INGENIOUS 5 ISSUE 2

Editor-in-Chief: Prof. Asoc. Dr. Teuta XHINDI/ Scientific Journal of the Faculty of Engineering, Informatics and Architecture / No. 5, issue 2/ 2025

ISSN 2789-4800

SMART SYSTEMS, SUSTAINABLE TECHNOLOGIES, AND DATA-DRIVEN INNOVATION

Teuta **XHINDI**/ Ankiola **BEU**/ Amanda **KOTE**/ Novruz **BILLA**/ Antonio **DEMIRI**/ Jora **BANDA**/ Iges **BANDA**/ Migena **KERI**/ Malvina **NIKLEKAJ**/ Read **DANJOLLI**/ Xhevahir **BADUNI**/ Xhoana **MYRTA**/ Dorila **RAKIPLLARI**

UETPRESS

INGENIOUS

No. 5, issue 2/2025

JOURNAL OF THE FACULTY OF ENGINEERING, INFORMATICS AND ARCHITECTURE

ISSN 2789-4800

UETPRESS

Published under the series “ACADEMIC JOURNALS”.

The journal has been catalogued at the National Library of Albania and the Library of the European University of Tirana, Albania.

(print ISSN: 2789-4800/ online ISSN: 2958-888X)

ingenious@uet.edu.al
www.uet.edu.al/ingenious

UETPRESS

Published by:
EUROPEAN UNIVERSITY OF TIRANA / ALBANIA

content

EDITORIAL

Smart Systems, Sustainable Technologies, and Data-Driven Innovation

Prof. Asoc. Dr. Teuta XHINDI

Volume 5, Issue 2 of the *Ingenious Journal* gathers research papers that address the complex realities of our time, including rapid technological transformation, rising environmental challenges, and the global need for intelligent, data-driven, and sustainable solutions. Under the unifying theme **"Smart Systems, Sustainable Technologies, and Data-Driven Innovation,"** this volume highlights the creativity, analytical depth, and practical contributions of our students and academic staff as they explore emerging technologies and their transformative impact on society.

Each article in this issue sheds light on a distinct dimension of technological progress and its role in addressing real-world challenges.

- The study "AI-Based Automated Traffic Monitoring System for Vehicles and License Plate Recognition" introduces a deep learning–powered framework for intelligent traffic surveillance, addressing technical, ethical, and infrastructural considerations relevant to modern urban mobility.
- The article "Strengthening Web Application Security through Email Verification and JWT Authentication" delivers a scalable cybersecurity solution designed to reinforce digital trust and improve user protection across web platforms.
- Another contribution, "Designing and Implementing a High-Availability Infrastructure for a Web Application on AWS," develops a resilient cloud architecture that ensures reliability, scalability, and performance in dynamic digital environments increasingly dependent on uninterrupted service delivery.

- The work "Prediction of Spot Instance Prices in AWS: An Automated Solution for Cost Optimization" presents a Machine Learning model that forecasts cloud resource pricing, enabling organizations and startups to optimize operational costs through data-driven decision-making.
- In the article "The Role of Trade Flows in Shaping Macroeconomic Indicators: A Big Data Approach for Albania," advanced econometric models and large-scale datasets are used to analyze how trade openness influences macroeconomic performance in emerging economies.
- The engineering paper "Solar Driven Fan Unit for a Solar Dryer" introduces a photovoltaic-powered solution that improves agricultural drying processes—an example of sustainable technology that leverages renewable energy for practical, community-level impact.
- "Design and Development of a Mobile App for Public Security and Emergency Alerts in Albania" develops a unified alerting system designed to improve public safety through real-time warnings, institutional coordination, and user-centered digital interfaces.
- An additional contribution, "Artificial Intelligence and Automation in Customer Service: Optimizing Interactions and Operational Efficiency," presents an integrated AI-CRM framework that enhances service responsiveness, reduces manual workload, and demonstrates how intelligent automation can strengthen customer experience in creative industries.
- The study "Analysis and Development of Data Validation Tools in Financial Systems: Case Study on Investment Funds" highlights how automated validation mechanisms improve data accuracy, regulatory compliance, and decision-making reliability in financial markets.

Together, these contributions embody the spirit of innovation at the intersection of smart technologies, sustainable solutions, and data-driven approaches. They demonstrate how interdisciplinary collaboration across engineering, informatics, economics, and design, can generate systems that are more intelligent, resilient, and aligned with global technological and environmental priorities.

The topics addressed in this volume are increasingly critical to our technological, environmental, and societal future. The articles collectively highlight the pressing need to advance research in smart systems, sustainable technologies, and data-driven innovation, fields that play a decisive role in shaping resilient, efficient, and forward-looking societies.

Prediction of Spot Instances prices in AWS Automated solution for cost optimization

Ankiola BEU[1]

Amanda KOTE[2]

Abstract

Developing applications for managing and optimizing cloud resources is a necessity in the modern era, especially for startups looking to increase performance with a limited budget. This paper focuses on creating a system for predicting Spot Instances prices on Amazon Web Services (AWS), using a data-driven approach and artificial intelligence to help users make automated decisions on the use of cloud resources.

The problem addressed by this study is related to the volatility and unpredictability of Spot instance prices, which, although offering a low-cost alternative to on-demand instances, can lead to unexpected service outages if not managed properly. To address this challenge, this paper explores the possibility of building a reliable ML model that aims to predict Spot Instances prices by analyzing historical data. The proposed approach aims to assess whether the use of this model can help Albanian startups optimize the cost of cloud infrastructure and make more informed decisions about the use of instances.

To achieve this objective, a methodology was followed that combines historical data analysis, training a regression model with the XGBoost algorithm, and its implementation through cloud-native technologies. The model was packaged with Docker and distributed on AWS via Elastic Container Registry (ECR), while

[1] European University of Tirana, student of Master of Science Course in "Information Technology"

[2] European University of Tirana, assistant lecturer in Department of Information and Technology

execution was performed in an AWS Lambda function connected to Amazon API Gateway via an HTTP API. Local testing and Postman testing were also performed to guarantee functionality and accuracy.

This paper contributes to the construction of a practical prototype for smart decision-making on cloud cost management and shows that the use of artificial intelligence can bring efficiency, scalability, and cost-effectiveness in the use of cloud infrastructure by startups and small businesses.

***Keywords**: AWS, Spot Instances, Docker, Lambda, API Gateway, ML, XGBoost, cloud-native, cost optimization, Postman.*

Introduction

The rapid evolution of cloud computing has reshaped how organizations deploy and manage technology resources, offering scalability, flexibility, and cost efficiency compared to traditional infrastructure. For startups, these benefits are particularly valuable, as they face the dual challenge of rapid growth and limited budgets. In Albania, cloud adoption is gaining momentum, driven by the broader digital transformation agenda and the need for startups to remain competitive in an increasingly technology-driven market.

Among the wide range of cloud services, Amazon Web Services (AWS) has emerged as a leading provider, offering Elastic Compute Cloud (EC2) Spot Instances as a cost-effective alternative to On-Demand pricing. Spot Instances allow access to unused AWS capacity at significantly reduced costs. However, their unpredictable price fluctuations and risk of sudden termination present a major challenge for consistent use. Without proper forecasting and automated management, these advantages can easily turn into operational risks for startups with limited technical capacity.

This research paper seeks to address this gap by developing and implementing a machine learning–based model for predicting AWS Spot Instance prices. By leveraging historical pricing data and the XGBoost regression algorithm, the system aims to automate decision-making processes, enabling startups to optimize cloud expenditures and reduce reliance on manual monitoring. This research contributes a practical, automated solution that demonstrates how artificial intelligence and cloud-native services can improve cost efficiency and operational resilience for small enterprises in Albania and beyond.

Two research questions are addressed in this study:

First research question: How can the use of AWS Spot Instances contribute to cost optimization for startups operating with limited financial resources?

This question investigates the potential financial benefits of Spot Instances, focusing on strategies that enable startups to reduce cloud expenditure while maintaining reliable performance.

Second research question: How effective is a machine learning model in predicting AWS Spot Instance prices to support automated decision-making and cost optimization?

This question evaluates the role of machine learning in providing accurate forecasts that allow startups to choose the right moments to purchase Spot Instances, thereby minimizing risks and maximizing savings.

This paper is guided by a clear hypothesis and set of objectives. The hypothesis proposes that implementing a machine learning model for AWS Spot Instance price prediction will significantly enhance cost efficiency and decision-making automation for startups compared to traditional monitoring methods. By combining predictive analytics with AWS-native integration, startups can achieve both greater accuracy and operational savings.

Hypothesis

The application of a machine learning model to predict AWS Spot Instance prices will enable startups to automate cloud resource management, reduce costs, and improve decision-making efficiency compared to traditional manual approaches.

Objectives

In order to achieve this, the research outlines several key objectives:

To examine the role of AWS Spot Instances in reducing cloud computing costs for startups and their potential contribution to financial efficiency.

To design and implement a predictive model using machine learning techniques, specifically XGBoost regression, for forecasting AWS Spot Instance prices.

To integrate the predictive model within a cloud-native architecture (AWS Lambda, Docker, and API Gateway) to automate decision-making.

To evaluate the effectiveness of this approach in enhancing cost optimization and operational efficiency for startups.

Literature Review

Cloud Computing and Pricing Models

Cloud computing has emerged as one of the most transformative paradigms in modern information technology, enabling organizations to scale infrastructure on demand without large upfront investments. According to (Mell & Grance, 2011), the National Institute of Standards and Technology (NIST) defines cloud

computing as a model that provides ubiquitous, on-demand access to shared pools of configurable computing resources that can be rapidly provisioned with minimal management effort. This definition underlines the role of cloud services in promoting both efficiency and innovation across industries.

Leading cloud service providers, including Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP), offer a range of pricing models to suit varying organizational needs. The three most adopted models are:

On-Demand Instances, which allow users to pay for compute capacity by the hour or second, without any long-term commitments. While this model provides maximum flexibility, it is also the most expensive (Amazon Web Services, 2023)

Reserved Instances, which provide discounts of up to 75% in exchange for a one- or three-year contract. This model suits organizations with predictable workloads but reduces flexibility. (Voorslys, Broberg, & Buyya, 2011)

Spot Instances, which enable access to unused AWS capacity at prices often 70–90% lower than On-Demand rates. These, however, are subject to termination by AWS with little notice if demand increases (Somasundaram, 2020)

For startups and small businesses, the Spot Instance model is particularly attractive, as it allows significant cost savings. However, its volatility and unpredictability also make it the most complex to adopt effectively.

AWS Spot Instances: Opportunities and Risks

AWS Spot Instances operate on a supply-demand mechanism, where prices fluctuate according to unused data center capacity. While enterprises can reduce their computing costs substantially by leveraging Spot Instances, these benefits are counterbalanced by risks of sudden interruptions and fluctuating prices (Zhang & Wu, 2016)

Several studies highlight the dual nature of Spot Instances: they are highly economical but operationally fragile. For example, (Ben-Yehuda, Ben-Yehuda, Schuster, & Tsafrir, 2013) demonstrated that unpredictable interruptions in Spot Instances can compromise the reliability of critical workloads. As such, Spot Instances are often relegated to non-critical or fault-tolerant tasks such as batch processing, testing, and big data analytics rather than production systems (Somasundaram, 2020)

To make Spot Instances more accessible, AWS has introduced features like Spot Fleet and EC2 Auto Scaling with Spot Instances, which help distribute workloads across multiple instances and mitigate the risk of termination. Yet, even with these tools, effective utilization depends heavily on accurate forecasting of pricing trends. Without predictive insights, startups often either overpay through On-Demand usage or risk service instability by relying too heavily on Spot Instances.

Machine Learning in Cloud Pricing Forecasting

Given the complexity of pricing dynamics, machine learning (ML) has been increasingly adopted as a tool for predicting cloud costs. ML methods can analyze large volumes of historical pricing data and detect non-linear patterns that traditional statistical methods fail to capture.

Research on the application of machine learning (ML) in cloud computing pricing has explored a wide variety of models, each with specific strengths and limitations. Early efforts relied on linear regression and time-series approaches such as ARIMA (Autoregressive Integrated Moving Average), which are well-suited for capturing linear trends and seasonality in historical pricing data. While these models can provide short-term forecasts, they often fail to adapt to the abrupt and highly non-linear changes that characterize Spot Instance markets, resulting in limited accuracy for real-world deployments (Shahrad, et al., 2020). To address these shortcomings, researchers have applied Support Vector Machines (SVMs), which are capable of identifying non-linear decision boundaries and handling complex relationships between features. However, SVMs often struggle when applied to very large datasets due to high computational demands, which can restrict their scalability in cloud pricing prediction scenarios. (Zhang & Wu, 2016) With the advancement of deep learning, Recurrent Neural Networks (RNNs) and their variants, particularly Long Short-Term Memory (LSTM) networks, have been employed to forecast Spot Instance prices more effectively. These models excel in sequential prediction tasks by retaining long-term dependencies in time series data, making them particularly suitable for detecting patterns in dynamic pricing environments (Jaishankar, 2020). (Jaishankar, 2020) specifically demonstrated the effectiveness of deep neural networks in achieving high prediction accuracy for AWS Spot pricing, but also emphasized their computational cost, which can make them impractical for smaller enterprises and startups operating under resource constraints. In response to these challenges, hybrid and ensemble approaches have been proposed to balance accuracy with efficiency. For example, (Gómez, de Miguel, & López, 2019) introduced AWS PredSpot, a predictive framework that integrates multiple machine learning models to estimate Spot Instance prices. Their work showed that combining complementary algorithms improves both prediction accuracy and robustness, thereby reducing the risks associated with Spot Instance adoption and enhancing overall cost efficiency. This progression of research highlights not only the growing sophistication of ML applications in cloud pricing but also the continuing trade-off between predictive performance, scalability, and accessibility for different organizational contexts.

XGBoost and Ensemble Approaches for Prediction

Among the machine learning techniques applied to pricing prediction, Extreme Gradient Boosting (XGBoost) has become one of the most prominent due to its scalability, efficiency, and robustness. Developed by (Chen & Guestrin, 2016), XGBoost builds upon gradient boosting principles while introducing regularization to reduce overfitting, making it especially effective for structured datasets. Several studies confirm its superior performance in forecasting tasks. For example, XGBoost has been used successfully in financial market prediction (Zhao, Wang, & Wang, 2021) energy consumption forecasting (Yu, Li, & Zheng, 2019) and resource scheduling in cloud environments (Li, Wang, & Zhang, 2020) These applications highlight its adaptability to domains where datasets are noisy, high-dimensional, and prone to sudden changes — conditions similar to AWS Spot Instance pricing. Compared to deep learning models, XGBoost offers advantages in terms of training speed, interpretability, and resource efficiency. This makes it particularly well-suited for startups, where computational capacity and time-to-deployment are critical.

Cost Optimization Strategies in Startups

Startups face unique challenges when adopting cloud services. While cloud computing enables rapid scalability, it also exposes small businesses to unpredictable expenses that can threaten their financial sustainability (Marston, Li, Bandyopadhyay, Zhang, & Ghalsasi, 2011) The literature on cloud adoption in small and medium-sized enterprises (SMEs) emphasizes the need for automated solutions that minimize manual oversight while ensuring cost predictability.

Studies by (Voorsluys, Broberg, & Buyya, 2011) and (Somasundaram, 2020) underline that while Spot Instances offer the greatest cost savings, they require sophisticated management to be practical. For startups in emerging markets such as Albania, these challenges are even more pronounced, as technical expertise and financial resources may be limited.

This underscores the value of automated, lightweight machine learning solutions that can be integrated directly into cloud-native architecture, reducing the reliance on manual monitoring and expert intervention.

Research Gap and Contribution

While a substantial body of research has focused on predicting Spot Instance prices, most studies concentrate on large-scale enterprise deployments or experimental simulations. Few works have examined the practical application of predictive

models for startups in emerging digital markets, where financial constraints and limited expertise amplify the challenges of cloud adoption.

This study seeks to bridge this gap by proposing a machine learning–based forecasting model, implemented with XGBoost and deployed within a serverless AWS architecture (Lambda, API Gateway, Docker). Unlike prior works that focus solely on accuracy, this approach emphasizes practical deployment and accessibility for startups. By automating decision-making and minimizing manual monitoring, the system enables small enterprises to optimize cloud expenditure without sacrificing reliability.

Methodology and Technology Used

This study applies an applied research methodology with the primary goal of addressing a practical challenge for Albanian startups: reducing cloud computing costs through accurate forecasting of AWS EC2 Spot Instance prices. The proposed solution integrates a machine learning model — specifically the XGBoost Regressor — with a serverless cloud deployment, enabling automated decision-making in real time.

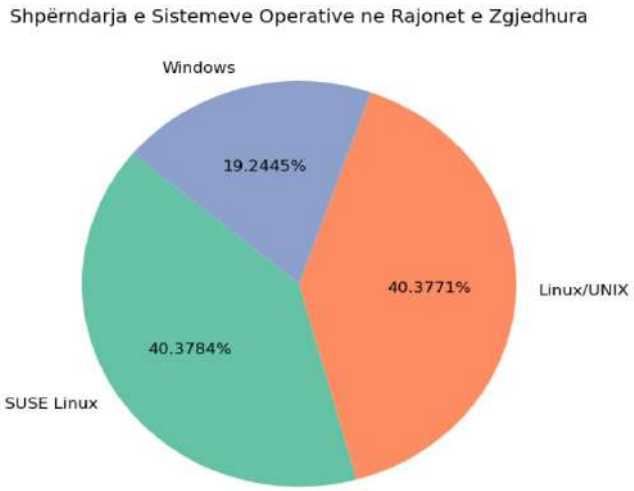
Data and Preprocessing

The dataset was sourced from the Kaggle AWS Spot Pricing Market, which provides historical Spot pricing information across AWS regions. For this study, two regions relevant to Albania's geographic proximity were selected (eu-central-1 and eu-west-1), focusing on two commonly used instance types (*m4.large* and *p2.xlarge*). Input variables included price, date-time, instance type, operating system, and region.

Before training, data preprocessing was performed through multiple steps: removing nulls and duplicates, encoding categorical variables (OS, instance type, region), normalizing values, and engineering new features from datetime. Additional polynomial features were generated to capture non-linear relationships.

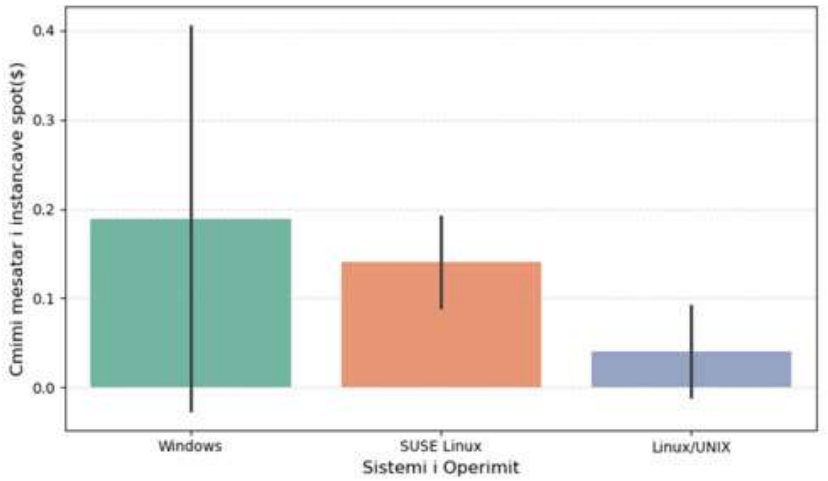
To understand the dataset composition, the distribution of operating systems across Spot Instances was first analyzed. Results showed that Linux-based systems dominate the dataset, while Windows represents a smaller share (Figure 1).

FIGURE 1:DISTRIBUTION of Operating Systems in Selected Regions



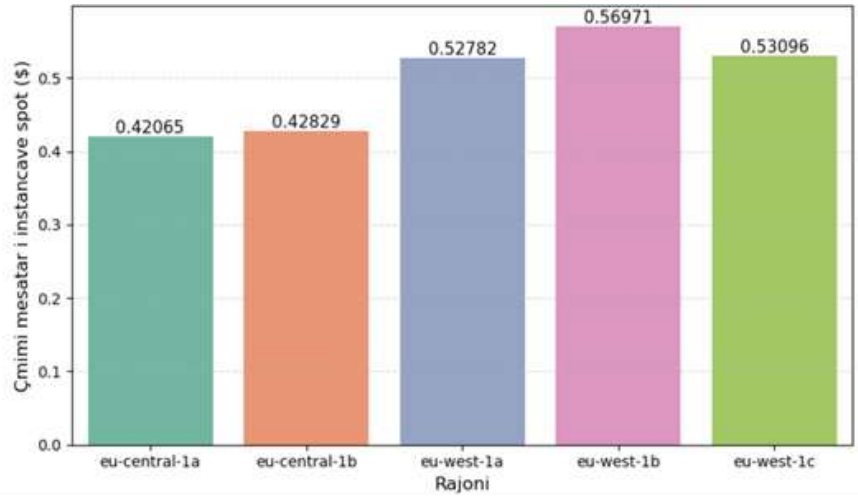
Price variation by operating system further highlighted key differences: Windows instances consistently exhibited higher average prices compared to Linux and SUSE Linux, confirming the financial impact of OS choice (Figure 2).

FIGURE 2: Average Spot Instance prices per operating system in AWS.



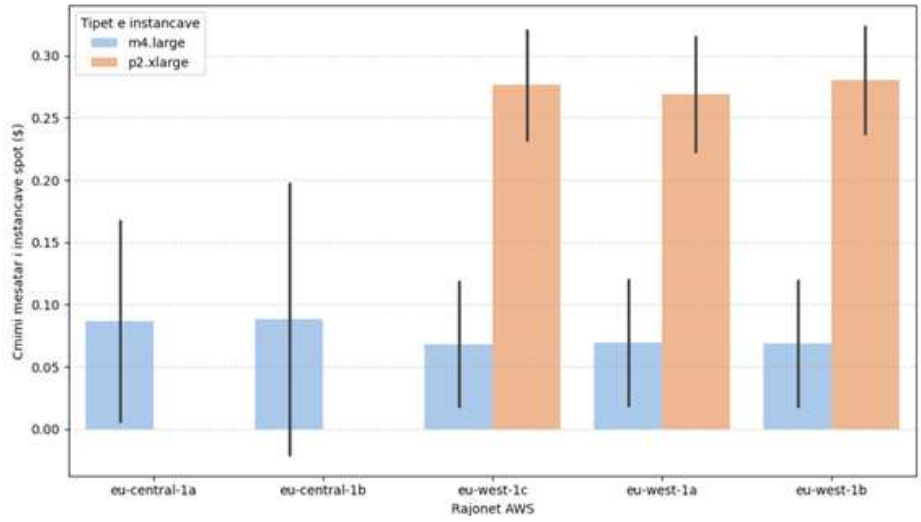
Regional variation also proved significant. The analysis revealed that eu-central-1a typically offered the lowest Spot prices, while eu-west-1c was among the most expensive availability zones (Figure 3).

FIGURE 3: Comparison of average Spot Instance prices across AWS regions eu-central-1 and eu-west-1.



Finally, comparison of instance types across regions demonstrated that costs vary not only geographically but also by configuration. The *m4.large* instance was cheapest in eu-west-1c, while *p2.xlarge* was more affordable in eu-west-1a (Figure 4).

FIGURE 4: Average Spot Instance prices by instance type and AWS region.



These insights confirm the high variability in Spot pricing, reinforcing the need for a predictive model to guide startups in cost-efficient decision-making.

Model Training and Evaluation

The dataset was divided into 80% training and 20% testing. The XGBoost Regressor was selected due to its strong performance on structured data, computational efficiency, and in-built mechanisms to reduce overfitting. The model was evaluated using MSE (Mean Squared Error), MAE (Mean Absolute Error), R^2, and MAPE, providing a comprehensive view of predictive accuracy. Results indicated that XGBoost captured Spot Instance price dynamics with sufficient precision to support cost optimization decisions.

Development Environment

The model was developed in Python using Jupyter Notebook, supported by widely used libraries:

- Pandas for data handling and cleaning.
- Scikit-learn for preprocessing, splitting, and evaluation.
- XGBoost for regression modeling.
- Matplotlib and Seaborn for visualization.
- Pickle and Zipfile for model serialization and packaging.

This environment ensured reproducibility and flexibility in model experimentation.

Cloud Integration and Deployment

To demonstrate real-world applicability, the model was integrated into an AWS-based deployment pipeline:

Docker was used to containerize the trained model and its dependencies.

The container image was uploaded to Amazon Elastic Container Registry (ECR) for storage and distribution.

A serverless AWS Lambda Function was created from the container, designed to receive JSON inputs (region, instance type, OS), run the model, and return Spot price predictions in JSON format.

Amazon API Gateway was configured to expose the Lambda function through a secure endpoint, enabling external access via HTTP requests.

Functionality was validated using Postman, which confirmed correct input handling and prediction output.

This architecture ensures the solution remains scalable, cost-effective, and accessible for startups without requiring advanced infrastructure management.

Connection to Research Questions

The methodology directly addresses the research questions by showing how Spot Instances can optimize costs and how machine learning can improve price prediction reliability. By embedding the model in a serverless AWS environment, startups gain a practical decision-support tool for selecting the most cost-efficient instance types in real time. This integration illustrates the dual contribution of the research: advancing predictive accuracy and offering a deployable, lightweight solution tailored to startups operating with constrained budgets.

Implementation

The implementation phase represents the core contribution of this research, bringing together machine learning techniques and cloud-native deployment to deliver a predictive solution for AWS Spot Instance pricing. This chapter provides a comprehensive overview of the steps undertaken to design, develop, and validate the system. The process is divided into two primary components: (1) the development and evaluation of the machine learning model and (2) the deployment of the trained model in a serverless cloud environment to enable real-time predictions. The first component describes the machine learning pipeline, beginning with data acquisition and preprocessing, followed by feature engineering, model training, and evaluation. Special emphasis is placed on the use of XGBoost Regressor, chosen for its robustness, efficiency, and ability to handle structured data with complex non-linear relationships. Exploratory data analysis and visualizations are presented to highlight key insights into Spot Instance price behavior across regions, operating systems, and instance types.

The second component addresses the integration of the model into AWS services. Using Docker, AWS Elastic Container Registry (ECR), AWS Lambda, and API Gateway, the model was deployed as a serverless application. This design ensures scalability, cost-effectiveness, and accessibility for startups without requiring extensive infrastructure management. Testing with Postman confirmed the system's ability to process user inputs in real time and deliver accurate price forecasts in JSON format.

By combining predictive modeling with cloud-native deployment, this chapter demonstrates both the technical feasibility and the practical value of the proposed solution for cost optimization in Albanian startups.

ML Model for Predicting Spot Instance Prices

The machine learning model was developed to forecast AWS Spot Instance prices through a systematic process that ensured data quality, robust feature representation, accurate predictions, and reproducibility for deployment. The steps undertaken are detailed below.

Data Loading and Cleaning

Two datasets containing Spot pricing information from the eu-central-1 and eu-west-1 regions were imported. These regions were chosen for their geographical proximity to Albania and their relevance to startup use cases. Within these datasets, the study focused on two widely used instance types, m4.large (general-purpose workloads) and p2.xlarge (GPU-intensive workloads), to represent diverse computational needs.

Data quality was ensured by removing missing values, nulls, and extreme outliers. This cleaning phase was critical to prevent distortions in the training process, as erroneous or incomplete records could bias the model and reduce predictive reliability.

Feature Processing

The datetime column was transformed into a machine-readable format, enabling extraction of meaningful temporal features. Several new attributes were engineered to capture seasonality and usage trends, including:

- Hour of the day – to detect intra-day price fluctuations.
- Day of the week – to capture weekly variations in workload demand.
- Month – to account for longer seasonal patterns.
- Numeric timestamp – for continuous temporal representation.

Categorical variables — region, operating system (OS), and instance type — were converted to numerical form through one-hot encoding, ensuring the model could differentiate between categorical levels without introducing ordinal bias.

Polynomial Features and Standardization

To capture non-linear interactions between variables, polynomial features of degree 2 were generated. For instance, interactions between region × instance type or OS × time allowed the model to recognize more complex dependencies affecting Spot prices.

All features were then standardized using StandardScaler, which ensured that variables with larger ranges (e.g., price) did not dominate smaller-scale features. This step improved both model stability and training efficiency.

Model Training

The dataset was split into 80% training and 20% testing, ensuring that the model was trained on sufficient data while preserving a test set for unbiased evaluation.

The XGBoost Regressor was selected due to its:

Proven performance on structured datasets.

Computational efficiency in handling large feature spaces.

In-built mechanisms for regularization to mitigate overfitting.

The training process involved multiple boosting rounds where weak learners were iteratively refined to minimize prediction errors.

Model Evaluation

Model performance was evaluated through multiple metrics, each offering unique insights:

Mean Absolute Error (MAE): average magnitude of errors in absolute terms.

Mean Squared Error (MSE): penalized larger errors more heavily.

R^2 (Coefficient of Determination): proportion of variance explained by the model.

Mean Absolute Percentage Error (MAPE): error expressed in percentage terms for intuitive interpretation.

Together, these metrics provided a balanced understanding of accuracy, error distribution, and explanatory power, ensuring reliability for real-world use.

FIGURE 5: Evaluation metrics of the XGBoost model for Spot price prediction.

Metric	Value
MAE	0.0035
MSE	0.0001
R^2 Score	0.9883
MAPE	0.0645
Approx. Accuracy	93.55%

As shown in Figure 5, the model achieved very high accuracy, with an R^2 score of 0.9883 and minimal errors across all metrics, confirming its robustness for Spot price forecasting.

FIGURE 6: Comparison of real vs. predicted Spot Instance prices using XGBoost

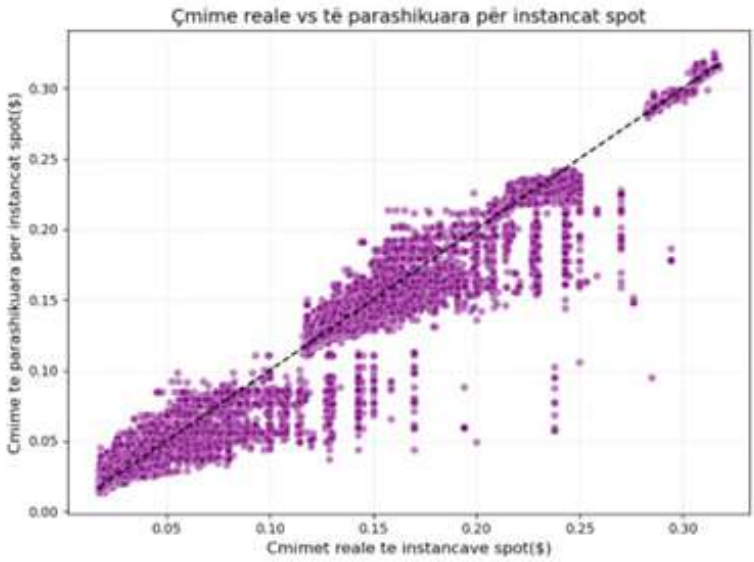


Figure 6 illustrates the close alignment between predicted and actual Spot prices. The majority of data points cluster near the diagonal, confirming that the model accurately captured underlying pricing dynamics for both m4.large and p2.xlarge instances.

Data Visualization and Insights

Visualizations were created to complement statistical evaluation and provide practical insights:

OS Distribution (Pie Chart): demonstrated that Linux-based systems (Linux/UNIX and SUSE Linux) dominated usage (~80%), with Windows comprising ~19%.

Average Spot Prices by OS (Bar Graph): showed Windows instances as consistently more expensive than Linux-based alternatives.

Regional Comparison (Bar Graph): revealed eu-central-1a as the most cost-effective zone, while eu-west-1c exhibited higher prices.

Instance Type by Region (Heatmap): highlighted that m4.large was cheapest in eu-west-1c, whereas p2.xlarge was lowest in eu-central-1a.

Predicted vs. Actual Price Comparison (Line Chart): validated model performance visually, confirming the alignment of forecasts with real data.

These visualizations not only supported the preprocessing stage but also illustrated the importance of instance type, OS, and region as key determinants of Spot pricing.

Model Storage and Serialization

To ensure reusability and reproducibility, the trained model and preprocessing objects (Scaler, Polynomial Features, encoded column lists) were serialized using Pickle. These files were then compressed into a ZIP archive, simplifying storage, transfer, and later deployment in AWS Lambda.

Real Input Prediction

A test scenario was conducted to demonstrate the end-to-end prediction pipeline. A sample input (region, instance type, OS, datetime) was provided in JSON format, after which all preprocessing steps were applied. The trained XGBoost model then generated a price forecast for the specified configuration.

The prediction was returned accurately, confirming the pipeline's ability to replicate results beyond the training environment and setting the foundation for real-time deployment in AWS cloud infrastructure.

Packaging the Model with Docker and Deploying to AWS Lambda

To operationalize the trained machine learning model, it was packaged into a Docker container and deployed in AWS Lambda as a serverless function. This approach ensured reproducibility, portability, and scalability, allowing the model to be executed in real time without requiring dedicated infrastructure. The deployment process was divided into several structured stages.

File requirements

A requirements.txt file was created, listing all the Python libraries required by both the trained model and the Lambda function. This file included dependencies such as XGBoost, Scikit-learn, Pandas, NumPy, Matplotlib, Seaborn, and Pickle. Incorporating this file ensured that the correct versions of libraries were installed automatically during the Docker build process, guaranteeing consistency across local development and AWS deployment environments.

Lambda Function Creation

A custom Lambda function was developed to process incoming requests and generate Spot price predictions in real time. The function began by loading the necessary dependencies along with the serialized XGBoost model (.pkl file), ensuring that the trained predictor was immediately accessible. Incoming requests were provided in JSON format and contained essential attributes such as datetime, AWS region, operating system (OS), and instance type. These inputs were then preprocessed dynamically: time-based features (hour, day, month, and timestamp) were extracted; categorical variables (region, OS, instance type) were converted using one-hot encoding; polynomial features were generated to capture non-linear

interactions; and all inputs were standardized using the pre-saved StandardScaler object. Once transformed, the data was passed to the trained XGBoost Regressor, which produced a Spot price forecast. The function returned the output as a structured JSON response, including the predicted price and any relevant status information. Error handling was also implemented to ensure robustness, allowing the system to gracefully manage incomplete or incorrectly formatted inputs. This Lambda function acted as the computational core of the deployment pipeline, bridging user input with accurate and actionable cost predictions.

Docker file for Lambda

A Docker file was created to package the Lambda function together with all its dependencies into a portable container image, ensuring that the model could be deployed consistently across environments. The build process was based on the official AWS Python 3.9 Lambda base image, which guaranteed full compatibility with the AWS runtime environment. During the building, all required libraries were installed directly from the requirements.txt file, while the Lambda handler script and the serialized model files were copied into the container. The entrypoint, or handler, was then explicitly defined so that AWS Lambda would know which function to execute upon receiving requests. The outcome of this process was a fully containerized environment that bundled the necessary libraries, source code, and trained artifacts into a single image, making it portable, reproducible, and ready for upload to Amazon Elastic Container Registry (ECR) for deployment.

Local Testing of Docker Image

Before deploying the model to the cloud, the container image underwent thorough testing in a local Docker environment to ensure reliability and correctness. The image was executed on port 9000, and sample JSON requests were submitted using curls to simulate real user inputs. These local tests confirmed that the image executed without errors, that the input data was correctly processed through the full preprocessing pipeline, and that the predictions generated by the XGBoost model were consistent with expectations. Conducting this validation step locally was essential to minimize deployment errors, streamline debugging, and guarantee that the containerized solution was fully functional prior to being pushed to AWS.

Uploading the Image to AWS ECR and Deployment to Lambda

After successful local testing, the Docker container was integrated into the AWS ecosystem using Amazon Elastic Container Registry (ECR) and AWS Lambda. This phase bridged the gap between local development and serverless cloud execution.

The deployment process began with the creation of a dedicated repository in Amazon Elastic Container Registry (ECR), named *spot-price-predicator*, which served as the secure and centralized storage for the trained Docker image. Amazon ECR was chosen not only for its scalability and seamless integration with other AWS services but also for its ability to handle versioned container images, making it straightforward to manage updates or rollbacks during the lifecycle of the project. Establishing this repository was a critical first step, as it provided the foundation for storing and distributing the packaged machine learning model across the AWS environment. To manage access securely, a dedicated Identity and Access Management (IAM) user was created with permissions tailored specifically to ECR operations. By assigning the policy AmazonEC2ContainerRegistryFullA ccess, the user was granted rights to push and pull images without unnecessarily broad privileges, following best practices for security and least privilege. This step was reinforced by generating an Access Key ID and Secret Access Key, which were later used to authenticate interactions with AWS through the Command Line Interface (CLI). By configuring a separate IAM user for these operations, the system reduced potential risks of misconfigurations or unauthorized actions while maintaining tight control over repository access. With credentials in place, the AWS CLI was configured locally to establish authenticated communication with the AWS environment. This configuration allowed Docker to be linked securely to the ECR registry, a step that was crucial for ensuring that only authorized users could push or pull images from the repository. Docker authentication with ECR provided a strong security layer, preventing unauthorized uploads or modifications that could compromise the reliability of the deployed model.

Once the authentication pipeline was established, the trained Docker image containing the serialized model, preprocessing pipeline, and Lambda handler script were prepared for upload. To align the image with AWS requirements, the Docker image was tagged with the repository URI in the format <aws_account_ id>.dkr.ecr.<region>.amazonaws.com/spot-price-predictor. This tagging process effectively bound the local image to the ECR repository, ensuring a smooth transfer during the push operation. After tagging, the image was pushed to the ECR repository, where its successful upload was confirmed directly from the AWS Management Console by verifying the presence of the image, its tag, and the timestamp of upload. This validation step was essential for ensuring that the correct version of the model was now available in the cloud and ready to be deployed in a serverless environment. The next step involved creating a Lambda function from the container image, using the ECR image URI as the runtime source. Lambda automatically pulled the Docker image stored in ECR and used it to build the serverless execution environment. This meant that all dependencies, libraries, and the trained model were included within the container, eliminating the need for separate configuration or manual dependency management. The Lambda

function replicated the exact preprocessing pipeline and prediction workflow that had already been validated during local testing but now operated within a fully managed AWS infrastructure that offered automatic scaling, monitoring, and fault tolerance.

To ensure that the Lambda function was operating correctly, extensive testing was carried out through the AWS Lambda console. Sample JSON events were submitted to simulate real-world user inputs, containing key attributes such as datetime, AWS region, operating system, and instance type. The function processed these requests seamlessly, applied the full preprocessing pipeline in real time, and returned Spot price predictions that matched expectations. Testing confirmed not only the correctness of the predictions but also the stability and reliability of the function under the serverless execution model. Importantly, this phase validated that the end-to-end deployment pipeline — Docker → ECR → Lambda — was fully functional and robust, marking a significant milestone in the project's implementation. By ensuring that the containerized model could be deployed, accessed, and executed consistently within AWS Lambda, the foundation was successfully laid for the subsequent integration with Amazon API Gateway, which would expose the function through a secure, publicly accessible REST API. In conclusion, this stage demonstrated the seamless transition from local development to a cloud-native deployment, highlighting how Docker and ECR combined with Lambda can provide a reproducible, secure, and scalable solution for real-time machine learning predictions in the context of AWS Spot Instance pricing.

FIGURE 7: Docker image creation

FIGURE 8: Docker image displayed in AWS ECR

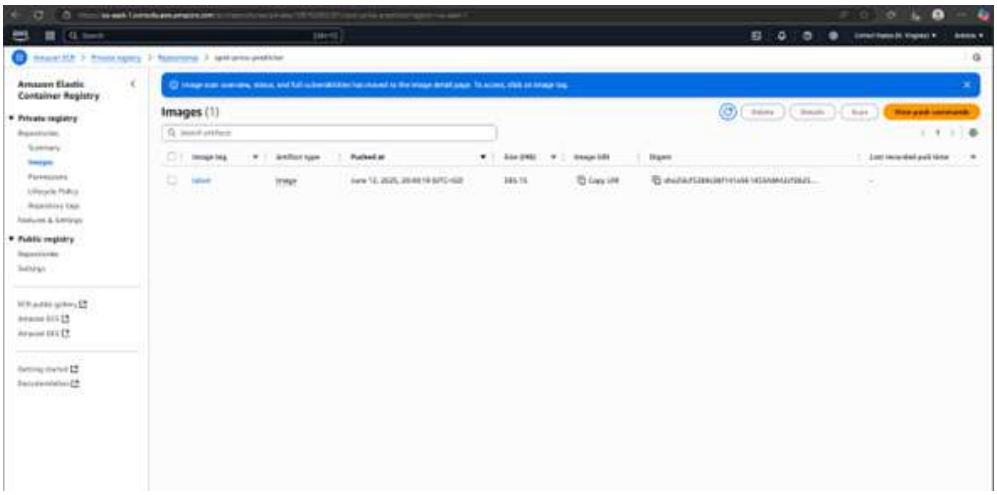
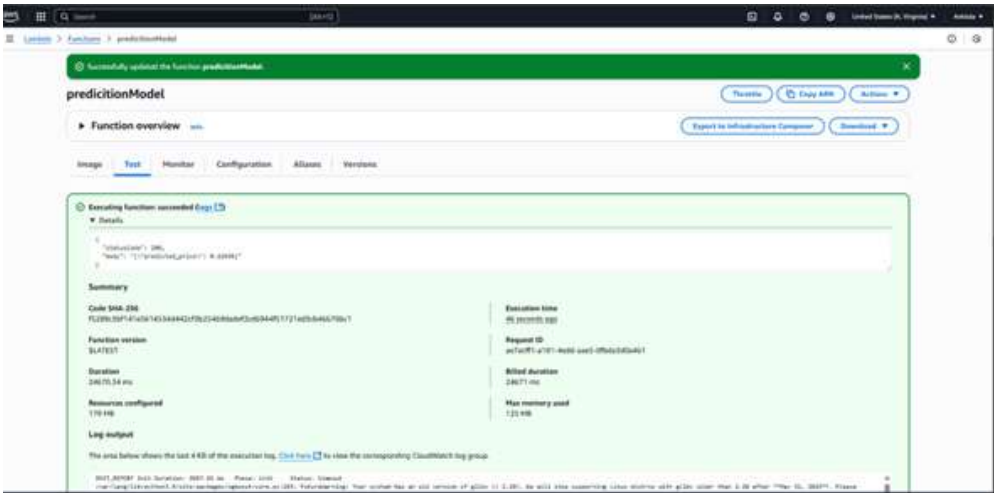


FIGURE 9: Lambda function creation



Integration with API Gateway and Postman Testing

Once the Lambda function was successfully deployed using the container image, the next critical step was to make the model accessible through a public interface. This was achieved by integrating the Lambda function with Amazon API Gateway, which allowed external applications and users to submit HTTP requests and receive predictions in real time.

The final stage of implementation involved exposing the trained model as a publicly accessible service through Amazon API Gateway, which acted as the secure and scalable entry point for client requests. This step was essential for

transforming the predictive system from a standalone model into a functional cloud-based service that could be consumed by external applications, startups, or individuals. To begin with, a new REST API was created within the API Gateway console, which provided the framework for handling incoming requests and directing them to the deployed Lambda function. Within this REST API, a dedicated resource was defined to represent the prediction service. For clarity and usability, the resource was labeled /predict, ensuring that any request sent to this endpoint would be clearly associated with Spot price forecasting. To this resource, an HTTP POST method was attached, chosen specifically because predictions required users to submit structured input data in JSON format rather than relying on simply GET requests. The POST method enabled the service to receive more complex payloads, including parameters such as datetime, region, operating system (OS), and instance type. Once the method was established, it was explicitly integrated with the deployed Lambda function, ensuring that any incoming request would automatically trigger the model's execution pipeline. Finally, the API was rolled out to a dedicated deployment stage (for example, /prod), which generated a unique, production-ready URL. This URL served as the public endpoint that clients could use to send requests and retrieve predictions, effectively operationalizing the system for real-world use.

Configuring the API to handle input and output correctly was critical to maintaining consistency, clarity, and ease of integration with third-party systems. The API Gateway was designed to pass the request body directly to the Lambda function without modifications, ensuring that the input arrived exactly as intended by the client. To use the service, clients were required to send their request in JSON format, structured with the necessary attributes for prediction. A typical request, for example, looked as follows:

```
{
 "datetime": "2025-06-25 14:00:00",
 "region": "eu-central-1a",
 "os": "Linux/UNIX",
 "instance_type": "m4.large"
}
```

Upon receiving such input, the Lambda function immediately triggered the preprocessing pipeline. This pipeline dynamically extracted time-based features (hour, day, month, timestamp), converted categorical features such as OS, region, and instance type using one-hot encoding, generated polynomial features to capture non-linear interactions, and applied scaling through the pre-saved StandardScaler object. After preprocessing, the transformed input was passed to the trained XGBoost Regressor, which computed a Spot price forecast. The

output was then formatted into a clear JSON response, ensuring consistency and interpretability. An example of the returned response is shown below:

```
{
 "predicted_price": 0.034,
 "currency": "USD",
 "status": "success"
}
```

This structured format offered several advantages: predictions were accompanied by contextual information such as the currency, while the status field confirmed successful execution or flagged errors if inputs were malformed. This design not only improved robustness but also simplified integration with external systems such as dashboards, cost optimization tools, or startup applications, which could directly consume the JSON response without additional processing.

To validate the functionality of the API, extensive testing was carried out using Postman, a widely adopted API testing tool. Postman provided an intuitive interface for sending HTTP POST requests and inspecting responses, making it ideal for debugging and validation. Multiple test cases were executed to simulate real-world scenarios. First, requests with varying payloads were submitted, covering different combinations of regions, instance types, and operating systems. This verified that the Lambda function was triggered correctly and consistently returned valid predictions. Second, malformed or incomplete JSON requests were tested deliberately to ensure that the system could handle errors gracefully. In these cases, the API returned structured error messages, demonstrating the robustness of the error-handling mechanisms. Finally, stress testing was performed by submitting multiple requests in succession to confirm that the system could handle varying loads without delays or failures. Postman thus played a crucial role in confirming that the endpoint was production-ready and capable of supporting diverse use cases.

The results of integration confirmed the success of this deployment stage. The API responded consistently with low latency, a reflection of the efficiency of AWS Lambda's serverless execution model. Predictions matched the expected values and aligned closely with the evaluation metrics established during model training and testing, further confirming the reliability of the system. Additionally, the fact that the API could be accessed publicly via a secure URL meant that users did not require direct access to AWS infrastructure. This accessibility was especially critical for startups, which often lack the technical expertise or resources to manage cloud infrastructure directly but still need predictive insights for cost optimization.

Beyond validation, the integration of Lambda with API Gateway introduced a range of strategic advantages that highlighted the broader significance of this

approach. Scalability was one of the most prominent benefits: API Gateway automatically scaled to handle fluctuating request loads, allowing the system to serve one or thousands of requests without any additional configuration. Security was another key feature, as endpoints could be protected with authentication tokens, API keys, or usage plans, ensuring that access was restricted to authorized users and preventing misuse. Flexibility was also notable; the endpoint could be integrated seamlessly into web applications, dashboards, or mobile apps, enabling startups to incorporate Spot price forecasting directly into their operational tools. Finally, the model offered a high degree of cost-effectiveness. Because the service was built on a serverless architecture, costs were incurred only when requests were processed, eliminating the need for idle infrastructure and aligning perfectly with the budget constraints of small businesses and startups.

In conclusion, the integration of the trained model with API Gateway and its validation through Postman marked the culmination of the deployment pipeline. This stage transformed the machine learning model from a locally tested artifact into a fully accessible, scalable, and production-ready service that could provide real-time Spot price forecasts. By offering a combination of technical robustness, low-latency performance, scalability, and cost-efficiency, the API Gateway integration demonstrated the practical value of the research and ensured that the solution could be readily adopted by startups seeking to optimize their cloud computing expenses.

FIGURE 10: API creation

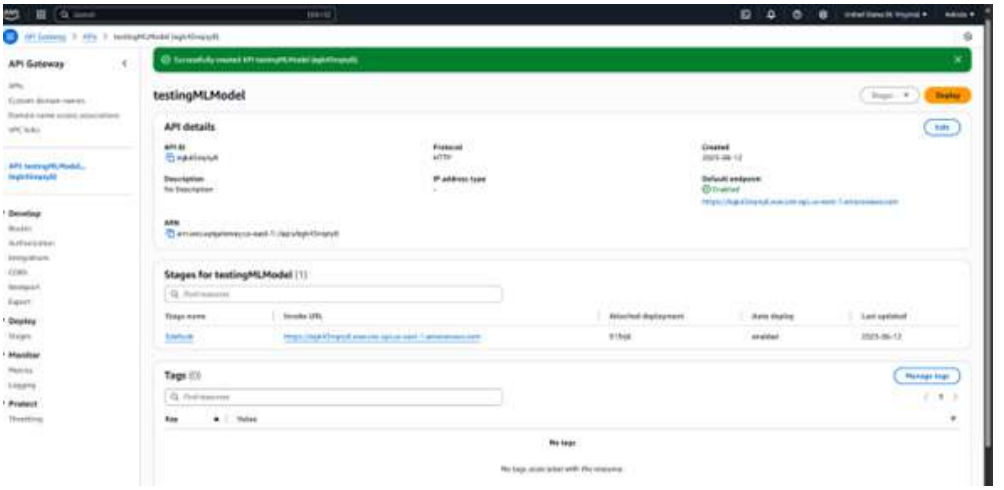
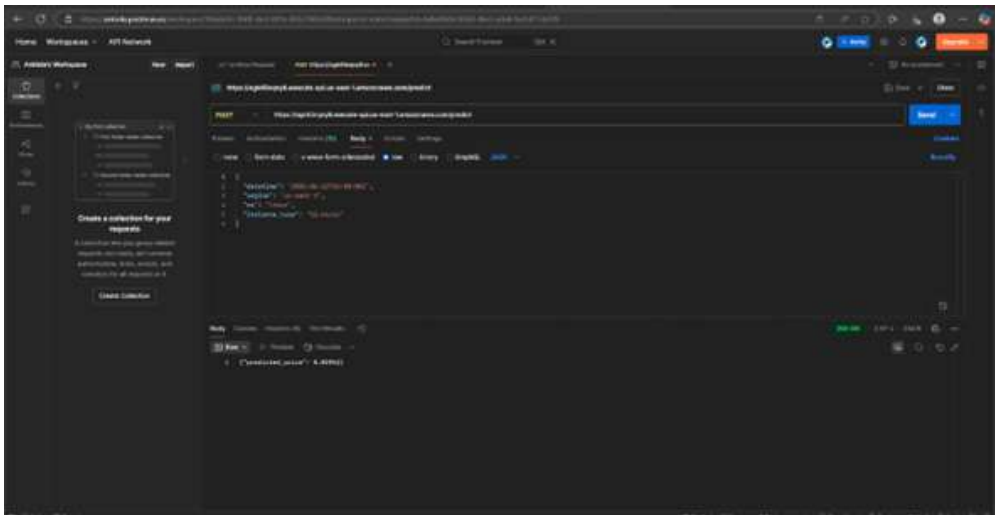


FIGURE 11: Postman testing



Conclusions and recommendations

This study demonstrated that AWS Spot Instances, when managed effectively, offer a viable pathway for startups to significantly reduce cloud computing costs while maintaining operational flexibility. By analyzing historical Spot pricing data and applying advanced machine learning techniques, the research validated that predictive modeling could transform the inherent unpredictability of Spot markets into actionable cost-saving strategies.

The machine learning pipeline, centered on the XGBoost Regressor, achieved high predictive accuracy, with R^2 values close to 0.99 and low error rates across MAE, MSE, and MAPE. These results confirm that the model effectively captures both linear and non-linear dynamics of Spot pricing, making it a robust tool for forecasting. Beyond statistical performance, the implementation of polynomial features, feature engineering from datetime variables, and one-hot encoding of categorical attributes enhanced the representational quality of the data, further contributing to the model's precision.

Equally important was the deployment of the model within a serverless AWS ecosystem. By containerizing the model with Docker, storing it in Amazon ECR, and deploying it through AWS Lambda and API Gateway, the solution was transformed from an experimental prototype into a scalable, accessible, and cost-effective decision-support service. Testing with Postman confirmed the system's functionality, with the model successfully processing real-time JSON inputs and returning accurate predictions. This integration underscores the practicality of combining artificial intelligence with cloud-native tools to support startups in emerging digital markets such as Albania.

The findings address both research questions posed at the outset. First, the study confirmed that Spot Instances can deliver substantial cost optimization benefits, but their adoption requires intelligent management strategies. Second, the machine learning model proved effective in predicting Spot prices, thereby reducing uncertainty and enabling automated, data-driven decision-making. The dual technical and applied contributions, predictive accuracy and deployable architecture, mark this work as a step forward in bridging the gap between academic research and real-world startup challenges.

Ultimately, this research illustrates that artificial intelligence, when combined with cloud-native services, can empower startups with limited resources to access advanced optimization strategies traditionally reserved for larger enterprises. The system developed here demonstrates that innovation in cost management is both achievable and highly impactful in the context of Albania's digital transformation.

Recommendations

Startups in Albania and other emerging markets should actively consider adopting AWS Spot Instances as part of their cloud infrastructure strategies, but this adoption must be paired with automated prediction and monitoring tools such as the one proposed in this study to reduce the risk of unexpected service interruptions. While the XGBoost model produced strong results, future research should explore hybrid and ensemble methods that combine gradient boosting with deep learning models like LSTM networks, as these approaches could better capture temporal dependencies and improve long-term prediction stability. Expanding the data set to cover additional regions, availability zones, and instance families would further enhance the model's robustness and applicability to a broader range of workloads. Moreover, the predictive system could evolve into a comprehensive cost optimization platform by integrating features such as budget alerts, workload scheduling, and automated fallback mechanisms to On-Demand instances, increasing usability for startups with limited cloud expertise. To maximize accessibility, the solution should also be extended with a user-friendly dashboard or web interface, enabling decision-makers without technical knowledge to easily interpret predictions and manage costs. On a broader scale, universities, incubators, and government initiatives in Albania should promote awareness and training programs that combine cloud computing with applied machine learning, thereby empowering the local startup ecosystem and enhancing competitiveness in the European digital economy. Finally, future work could expand beyond cost optimization to incorporate sustainability goals, with predictive models recommending instance types or regions with lower carbon footprints, thereby aligning with EU Green Deal priorities and positioning Albania as an active contributor to sustainable digital transformation.

References

Amazon Web Services. (2023). *Amazon EC2 Spot Instances pricing*. Retrieved from Amazon Web Services: https://aws.amazon.com/ec2/spot/

Ben-Yehuda, O., Ben-Yehuda, M., Schuster, A., & Tsafrir, D. (2013). Deconstructing Amazon EC2 spot instance pricing. *ACM Transactions on Economics and Computation*, 1-20.

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2016)* (pp. 785–794). San Francisco, CA, USA: Association for Computing Machinery (ACM).

Gómez, A. B., de Miguel, I., & López, V. (2019). AWS PredSpot: Machine learning for predicting spot instance prices. *Journal of Cloud Computing*, 1-15.

Jaishankar, R. K. (2020). Forecasting the price of AWS on-spot instances using deep neural networks. *International Journal of Cloud Computing and Services Science*, 123-134.

Li, X., Wang, Y., & Zhang, H. (2020). Intelligent cloud resource allocation using gradient boosting models. *IEEE Access*, 187920–187930.

Marston, S., Li, Z., Bandyopadhyay, S., Zhang, J., & Ghalsasi, A. (2011). Cloud computing – The business perspective. *Decision Support Systems*, 176-189.

Mell, P., & Grance, T. (2011). *The NIST definition of cloud computing.* Gaithersburg, MD: National Institute of Standards and Technology, U.S. Department of Commerce.

Shahrad, M., Fonseca, R., Goiri, I., Chaudhry, G., Bianchini, R., & Nagarakatte, S. (2020). Serverless in the wild: Characterizing and optimizing the serverless workload at a large cloud provider. *USENIX Annual Technical Conference (USENIX ATC 2020)* (pp. 205-218). Berkeley, CA: USENIX Association.

Somasundaram, G. (2020). *Optimizing cloud costs with AWS spot instances.* Sebastopol, CA: O'Reilly Media.

Voorsluys, W., Broberg, J., & Buyya, R. (2011). Cost-effective cloud resource provisioning. In R. Buyya, J. Broberg, & A. Goscinski, *Cloud computing: Principles and paradigms* (pp. 243-266). Hoboken, NJ: Wiley.

Voorslys, W., Broberg, J., & Buyya, R. (2011). Cost-effective cloud resource provisioning. In *Cloud computing: Principles and paradigms* (pp. 243-266). Hoboken, NJ: Wiley.

Yu, Y., Li, X., & Zheng, Y. (2019). Short-term energy consumption forecasting using XGBoost. *Energy*, 229-240.

Zhang, Y., & Wu, Q. (2016). Predicting cloud resource prices with SVM models. *Future Generation Computer Systems*, 1-10.

Zhao, W., Wang, L., & Wang, J. (2021). Stock market trend prediction using XGBoost. *Applied Soft Computing*.

Design and Development of a Mobile App for Public Security and Emergency Alerts in Albania

Novruz BILLA[1]

Teuta XHINDI[2]

Abstract

Albania currently lacks a centralized mechanism to quickly disseminate emergency alerts and public safety information to its citizens. This paper presents the design and development of a mobile application aimed at managing emergency alerts and strengthening public safety response in Albania. The proposed system leverages real-time push notifications to inform citizens of crises or hazards as they unfold. This study followed a Design Science approach to gather requirements, architect a three-tier system, and implement a prototype using modern web and mobile technologies. The application consists of a React Native mobile client (for both iOS and Android) and a Node.js/Express backend with a MongoDB database. Key features include secure user authentication via JSON Web Tokens (JWT), role-based access control for institutional users, and an initial implementation of intelligent alert classification based on incident urgency. Preliminary testing with end-users and domain experts indicates that the system delivers stable performance, a user-friendly interface, and accurate, timely delivery of alerts. While these early evaluations (including a mean System Usability Scale score of 84/100) are promising, they are not yet conclusive. The

[1] European University of Tirana, Faculty of Engineering, Informatics and Architecture, Department of Informatics and Technology, Tirana, Albania, nbilla@uet.edu.al

[2] European University of Tirana, Faculty of Engineering, Informatics and Architecture, Department of Informatics and Technology, Tirana, Albania, teuta.xhindi@uet.edu.al

study highlights the role of modern information technology in improving institutional emergency response and provides a foundation for further integration of the system into national public safety infrastructure. The results and insights from this project serve as an important step toward a full-scale deployment of a nationwide emergency notification system in Albania.

Keywords: *Public safety, Emergency alerts, Mobile application, Push notifications, Design Science Research, Albania.*

Introduction

Albania is exposed to a range of natural and man-made hazards, yet it currently lacks a unified platform for disseminating official emergency information to the public. Recent disasters have highlighted this gap. For example, during the November 2019 earthquake (magnitude 6.4) that struck Albania, over 50 people lost their lives and hundreds were injured. In the critical hours immediately after the earthquake, official information channels were largely absent, leading to public panic and confusion. Similarly, major floods in 2022 isolated thousands in the Shkodër and Lezhë regions, underscoring shortcomings in timely evacuation warnings. Beyond natural disasters, public safety incidents such as criminal attacks or abductions also demand rapid public alerting. Although the State Police introduced the "Digital Commissariat" app for citizens to report incidents, there remains no comprehensive system for broadcasting emergency alerts to the populace and enabling two-way citizen reporting in one unified platform.

Globally, governments are increasingly leveraging mobile technology for emergency communication. Traditional siren and broadcast systems are being augmented or replaced by direct alerts to mobile phones. Notably, the European Union's Electronic Communications Code (EECC) directive in 2018 mandated that all member states implement a cell-broadcast based public warning system by 2022. Many countries including France, Germany, Italy, and the UK have since deployed nationwide mobile alert systems, either via cell broadcast or smartphone applications. These systems can reach millions of people within seconds, as evidenced by the UK's recent nationwide alert tests (BBC News, 2022) and similar initiatives elsewhere. The demand for timely information during crises is clear: for instance, usage of a global earthquake alert app in Albania surged from under 3,000 to over 146,000 users in the week following the 2019 quake, as citizens scrambled for reliable real-time updates (EMSC, 2019). The recurring communication failures observed during these events demonstrate a systemic need for a centralized and reliable emergency-alerting mechanism in Albania. While various institutions use fragmented channels, websites, social

media posts, press releases, or independent applications, none of these operate as a unified, real-time platform capable of reaching the population instantly and enabling structured two-way interaction. The absence of such a system reduces institutional responsiveness, increases public uncertainty, and limits the effectiveness of disaster-management operations. **This study therefore focuses on evaluating whether the development and implementation of a unified mobile emergency-alert and citizen-reporting platform can effectively address these gaps by improving communication accuracy and reducing response times.**

Research Questions

RQ1. What are the specific needs and gaps in emergency communication and alerting in Albania?
RQ2. How effective is the prototype in test scenarios (SUS, task success/time, and communication quality)?

Hypothesis: Implementing a unified mobile application for emergency alerts and citizen reporting will significantly improve emergency-management effectiveness in Albania by shortening institutional and citizen response times and increasing the accuracy and trustworthiness of public communication.

In this context, this study aims to develop a modern emergency notification mobile application tailored to Albania's needs. The goal is to provide a unified, real-time communication platform whereby authorities can instantly send public safety alerts (evacuation orders, hazard warnings) to citizens' smartphones, and citizens can report incidents (fires, accidents, etc.) directly to authorities. This paper presents the system architecture, implementation, and preliminary evaluation of the proposed solution. Its focus is on the technical design of the system a three-tier architecture encompassing a mobile frontend, a cloud-based backend, and a database along with security and scalability considerations. It also describes initial usability testing results and feedback from emergency management experts, which informed our discussion of the system's potential impact and areas for future improvement.

Literature review

Early warning and public alert systems have evolved significantly in recent decades. Traditional emergency alert systems (EAS) relied on sirens and broadcast media interruptions to reach the public. In the United States, for example, the legacy EAS breaks into radio and TV programming for urgent messages (FCC,

2019). With the ubiquity of mobile phones, attention has shifted toward cellular-based alerting. Modern systems like the U.S. Wireless Emergency Alerts (WEA) under FEMA's Integrated Public Alert & Warning System (IPAWS) can push geo-targeted text alerts to every compatible mobile phone in an affected area, with no need for any special app . These alerts appear automatically on devices and have proved capable of reaching broad populations almost instantly. The European Union similarly adopted a "Reverse-112" cell broadcast approach, mandating that all member states implement mobile public warning systems by 2022. Countries such as France, Germany, Italy, and Sweden have since deployed nationwide cell-broadcast systems (FR-Alert in France) to comply with this mandate.

Alongside broadcast-based solutions, many jurisdictions have experimented with mobile applications for emergency alerts. Such apps can offer richer interactivity (two-way reporting, multimedia content) but face challenges in achieving widespread adoption. A notable example is France's SAIP alert app, launched in 2016 and intended to notify citizens of terror attacks. SAIP suffered from technical failures and low usage, covering only about 1% of the population, and was ultimately discontinued in 2018. France pivoted to the FR-Alert cell broadcast system thereafter. In contrast, Germany has found some success with apps like NINA and KATWARN, which deliver alerts to users who opt in, complementing Germany's newer cell broadcast system (Bundesnetzagentur, 2022). The United Kingdom launched its own nationwide mobile alert system in 2023 using cell broadcast, conducting a full population test to ensure reach (BBC News, 2022). These experiences suggest that while dedicated apps can provide advanced features, integrating with device-native alert channels (like SMS Cell Broadcast) is crucial for maximum reach and reliability. Purely app-based systems risk leaving many users uninformed if the app is not installed or promptly maintained.

Another line of related research focuses on leveraging artificial intelligence (AI) to enhance emergency communications. AI and machine learning techniques have been explored for tasks such as automatic detection of incidents (earthquake early detection or social media monitoring) and intelligent prioritization of alerts. These approaches could enable future systems to filter false reports and highlight the most critical information. However, the use of AI in public safety also raises important ethical and transparency considerations (Smith *et al.*, 2023). Lastly, prior studies underscore the importance of usability in emergency alert tools. Users must be able to quickly understand and trust alerts under stressful conditions. Standardized usability metrics like the System Usability Scale are often applied to evaluate emergency apps (Brooke, 2013; Bangor *et al.*, 2009). Our work builds on these insights from the literature, aiming to combine a robust technical architecture with user-centric design and lessons learned from global best practices and pitfalls in emergency alert systems.

Methodology

The methodology combines mixed methods with a Design Science Research (DSR) approach, integrating theoretical/secondary research, iterative prototyping, and formative usability evaluation with end users and institutional experts (Hevner et al., 2004; Johnson & Onwuegbuzie, 2004; ISO 9241-210, 2019).

Research Approach and Design

A pragmatic paradigm with mixed methods was adopted to channel quantitative and qualitative evidence toward engineering decision-making (Johnson & Onwuegbuzie, 2004). The approach is grounded in DSR: building the mobile application and evaluating it as a research contribution (Hevner et al., 2004). In analogy with human-centered design, ISO 9241-210 (2019) principles were followed.

The research proceeded in five phases, each building on the previous:

- Phase 1 - Requirements Analysis: Conduct secondary research and consult experts to identify user needs, system constraints, and success criteria for an Albanian emergency alert system.
- Phase 2 - System Design: Develop the system's overall architecture, data schema, security model, and interface design based on the requirements.
- Phase 3 - Implementation: Iteratively build the prototype (both frontend mobile app and backend server), refining features through multiple development cycles.
- Phase 4 - Expert Evaluation: Use semi-structured interviews and walkthroughs with institutional stakeholders to assess the prototype's relevance, identify gaps, and gather suggestions, particularly regarding integration with existing systems.
- Phase 5 - Preliminary Validation: Conduct formative usability testing with a small group of end-users to validate core functionality and identify any critical usability issues before larger-scale deployment.

This structured yet iterative approach allowed the project to remain flexible. Feedback and findings from each phase fed into subsequent design adjustments, aligning the artifact closely with both user expectations and institutional requirements.

Data Collection

End Users (SUS and Scenario-Based Tasks): Selection combined convenience sampling (users with smartphones) and ease with aim of representing ages 18–55 and basic digital skills. Instruments included: (a) SUS (10 items, Likert 1–5) for usability assessment (Brooke, 2013); (b) scenario-based tasks (receiving an emergency notification, viewing it, sending an emergency report); (c) post-test mini-interview (5–7 minutes) for qualitative comments; and (d) procedure involving 2–3 minute orientation, performing three tasks without guidance with measurements (success/failure, task time, observations), SUS completion, and mini-interview. Sessions lasted approximately 20–25 minutes per user.

Expert Interviews: Purposeful selection of three key institutional profiles. Format: semi-structured, 25–35 minutes, online or face-to-face. Thematic guide covered: current alert flows, integration with existing systems, operational requirements (CAP, 112), technical/legal limitations, success criteria, and operational indicators.

Secondary Data: Strategic documents, standards, and methodological literature were used as secondary data to justify findings and design choices (Hevner et al., 2004; ISO 9241-210, 2019).

Data analysis methods

SUS and Quantitative Data: SUS calculation followed Brooke's scheme (0–4 per item conversion), sum × 2.5 → 0–100 (Brooke, 2013). Descriptive statistics included: SUS mean, standard deviation, score range; average task time, success rate.

Instrument reliability: Cronbach's α for SUS (expectation $\alpha \geq 0.70$).

Interpretive synthesis: comparison of results with guidance thresholds (SUS ≈ 68 average; >80 "good–very good") (Bangor et al., 2009). Result calculation used Python.

Qualitative Data: Thematic analysis following Braun and Clarke (2006): familiarization, initial coding, theme construction (instruction clarity, navigation intuitiveness, notification trust), theme review and definition, representative quote selection (without personal identification).

Triangulation: convergence between quantitative evidence (SUS/performance) and expert interview data to reach verifiable design recommendations.

Validity, Reliability, and Methodological Limitations

Content validity was ensured by relying on standards and literature for task design and SUS interpretation (ISO 9241-210; Brooke, 2013). Reliability increased through use of a standardized instrument (SUS) and clear observation rules; where

possible, Cronbach's α was assessed as an internal consistency indicator. Source triangulation (users, experts, documents/standards) and method triangulation (quantitative/qualitative) reduced bias risk. However, the design was formative: the small sample and controlled environment do not represent stress of real scenarios; therefore, field piloting and high-load testing is needed before large-scale deployment.

Ethical Considerations

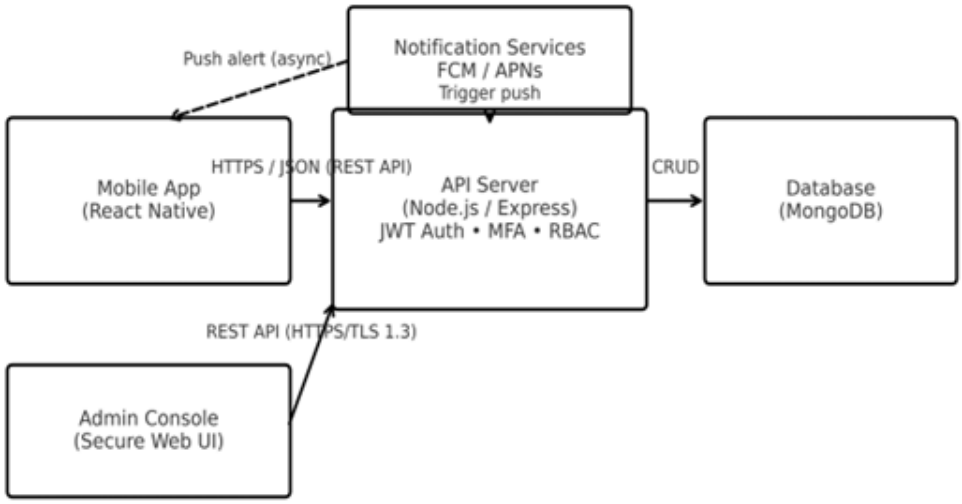
Research was conducted in accordance with scientific integrity and data protection norms: electronically documented informed consent with withdrawal option at any time; user data anonymization and expert pseudonymization (institutional role only); data minimization and encrypted storage/transmission with access limited to research team only; research data retention for five years in compliance with Law No. 9887/2008 and GDPR principles (Reg. (EU) 2016/679); and non-harm principle with scenarios and questions formulated to avoid unnecessary stress, with session interruption if concern arises.

Methods and Analysis

System Architecture

The system follows a three-tier architecture consisting of: (1) a presentation layer (the mobile client), (2) an application logic layer (the backend server), and (3) a data layer (the database). This classical design modularizes the application, allowing each layer to be developed, maintained, and scaled independently. In our implementation, the presentation layer is a React Native mobile application (deployable on both iOS and Android), the logic layer is a Node.js web service using the Express framework, and the data layer is a MongoDB document-oriented database. This technology stack constitutes a variant of the popular MERN architecture (MongoDB, Express, React, Node), enabling a unified JavaScript codebase across all tiers and efficient data exchange via JSON. Communication between the mobile app and server is facilitated exclusively through a RESTful API over HTTPS (secure HTTP), adhering to REST design principles. The client sends HTTP requests (to fetch alerts or submit a report) and receives JSON responses, while the server handles application logic and interacts with the database.

FIGURE 1. System architecture diagram



The mobile client communicates with the backend over the network, and the backend in turn reads from and writes to the database. In addition, the backend connects to external notification services (Firebase Cloud Messaging for Android and Apple Push Notification service for iOS) to deliver urgent alerts to user devices.

The React Native app component encompasses the user interface and client-side logic (state management, input handling, etc.). The Node.js/Express server component is organized into sub-modules for different functionalities (a *User* controller, an *Alerts* controller, an *Authentication* service, etc.), and it exposes a REST API interface to the client. The server also interfaces with the MongoDB database and with the external push notification services. This architecture promotes *loose coupling*; changes in one component or layer (for instance, switching the database engine) have minimal impact on others as long as the communication interfaces (API endpoints, data formats) remain consistent. Such decoupling improves maintainability and extensibility of the system. The design also inherently supports scalability and reliability: each tier can be scaled horizontally as needed (deploying multiple Node.js server instances behind a load balancer, or using MongoDB replication/sharding for large data volumes) and a failure in one server node or frontend instance will not collapse the entire system. Security considerations are addressed throughout the architecture, all client-server communications are encrypted (HTTPS) and token-based authenticated, sensitive data is encrypted or hashed before storage, and the database is configured with access control roles and backup replication policies for fault tolerance .

Functional Requirements (FR)

- FR-1 Emergency alerts (High): Push notifications with type, severity, area, and protective actions; audible + vibration; ≤5s delivery.
- FR-2 Geo-targeting (High): GPS-based delivery; saved places (home/work); target by radius or polygon.
- FR-3 Incident reporting (High): Quick form (category, text, auto-GPS, photo/video); routed to relevant authority; confirmation; optional contact.
- FR-4 Chatbot assistant (Medium): FAQ guidance from verified KB (multilingual); hands off to hotlines/live help when needed.
- FR-5 Interactive map (Medium): Live hazard zones, shelters/medical, user location; tap for details; updates as events evolve.
- FR-6 SOS / 112 (High): Prominent button; passes GPS + ID (if logged-in); fallback SMS with coords under poor connectivity.
- FR-7 Admin console (High): Secure web UI with MFA + RBAC, templates, and full audit trail (create/approve/send times, actor).
- FR-8 Alert history (Medium): List of received alerts with linked updates; filter by category.
- FR-9 Preferences (Low): Language, sounds, quiet hours; users cannot disable life-critical alerts.

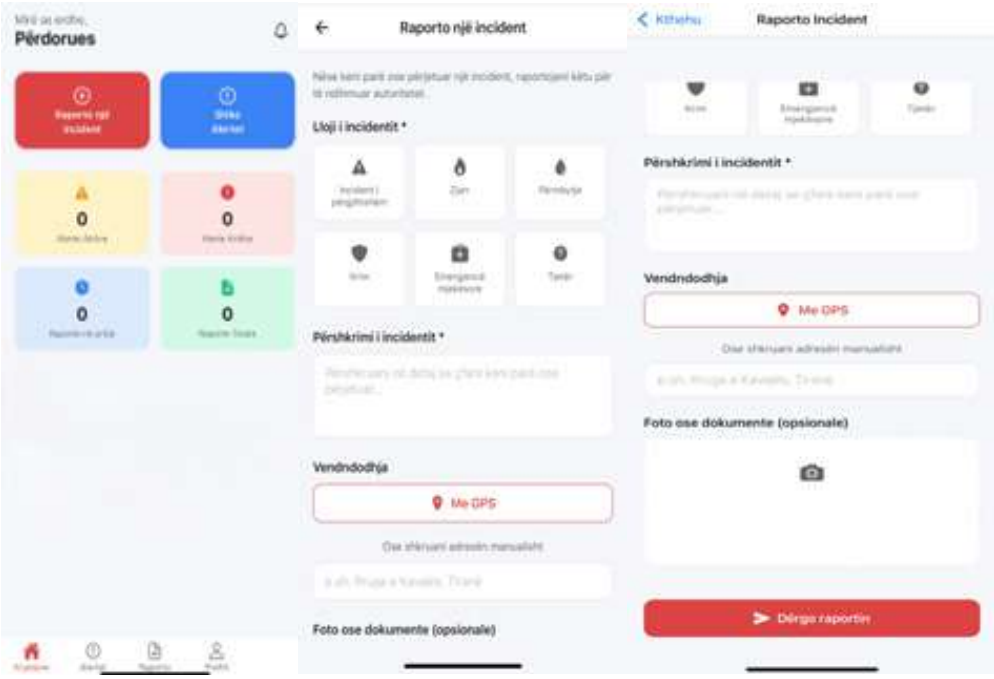
Non-Functional Requirements (NFR)

- NFR-1 Performance: End-to-end delivery <5s (normal); handle ≥100k concurrent and scale to ~2.8M users; typical queries <2s.
- NFR-2 Reliability: 99.9% uptime; ≥99.5% device reach; redundancy + automatic failover.
- NFR-3 Security: TLS 1.3+ in transit; AES-256 at rest; MFA for admin; detailed audit logs; regular security testing.
- NFR-4 Scalability: Horizontal scale (servers, workers); DB sharding/ partitioning; efficient batched push delivery.
- NFR-5 Maintainability: Modular code, docs; automated unit/integration tests; containerized deployments.
- NFR-6 Usability: One–two taps to key actions (view alert, report, SOS); plain language; accessibility features (larger text, screen readers).
- NFR-7 Interoperability: CAP-formatted alerts; RESTful APIs; interfaces compatible with telecom CB/112 systems.

Mobile Frontend

The frontend is a cross-platform mobile application developed with React Native. This choice allows a single codebase to natively deploy on both iOS and Android devices, providing nearly native performance and a consistent user experience on each platform. The mobile app's user interface was designed to be clean and minimalistic, focusing on critical functions to be used under emergency conditions. It includes screens for viewing active emergency alerts (a list or map of alerts in the vicinity), submitting a new incident report, and viewing user profile/settings. The UI adheres to human-centered design guidelines for clarity and simplicity (ISO 9241-210:2019), with consistent navigation and large, clear interactive elements to accommodate usage under stress. For example, posting a new incident report is accomplished through a single form with fields for incident type, location (which can be auto-obtained via GPS), description, and an optional photo attachment. Figure 2 shows an example of the app's data submission interface and alert display screen in the prototype.

FIGURE 2. Homepage and Send Report page



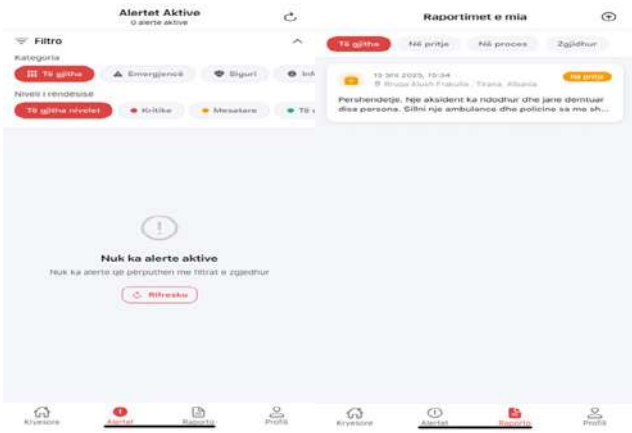
Also, to ensure responsiveness, the app manages local state (caching the latest alerts) and uses asynchronous API calls to the server.

Authentication on the mobile app is handled via JSON Web Tokens (JWT). Upon successful login, the server issues a signed JWT representing the user's identity and role. The app stores this token securely on the device (in the secure keychain storage) and attaches it to the Authorization header of subsequent requests. This approach eliminates the need to maintain sessions on the server and enables a stateless authentication model that scales well. Whenever the app is launched, it checks for a valid stored token to keep the user logged in across sessions. If the token is missing or expired, the user is prompted to log in again. In addition to login and registration interfaces, the app includes logic to handle incoming push notifications. When the user first installs or opens the app, it registers the device with the notification service (obtaining a device token) and sends this to the backend server. This enables the server to target that device for future alerts. If a critical alert is pushed by the server, the mobile OS will display it as a system notification (with a distinctive sound/vibration). Tapping the notification will automatically open the app and navigate to a detail screen showing the alert information and safety instructions. This push mechanism allows users to receive urgent warnings in near real-time even if the app is running in the background.

Backend and API

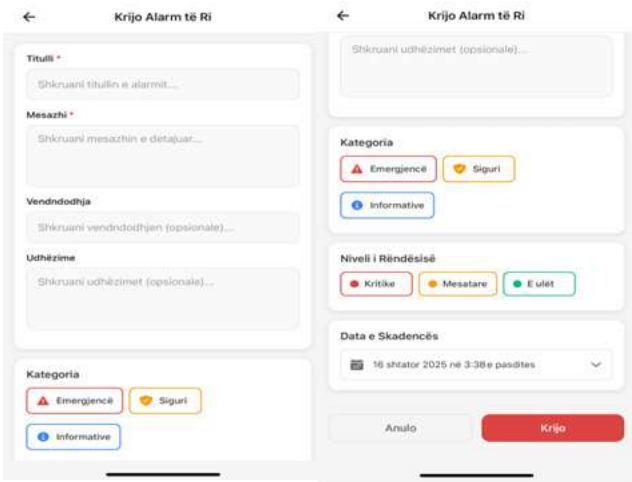
The backend of the system is implemented as a RESTful web service using Node.js with the Express framework. The server is structured according to a Model–View–Controller (MVC) pattern adapted for a web API context. The source code is divided into logical modules: routes (Express route definitions for each API endpoint), controllers (functions that handle requests and responses, encapsulating application logic), models (database schemas and data access using MongoDB via an Object-Document Mapper), and middleware components for cross-cutting concerns (authentication, logging). For each major resource in the system, an Express router is defined, for example, there are routes for user management, for emergency alerts, for incident reports, and for authentication. Each route maps HTTP endpoints (URLs and methods) to controller functions. For instance, the route POST /api/users/register invokes a controller that creates a new user account (after validating input and hashing the password), and POST /api/users/login verifies credentials and, on success, returns a signed JWT token to the client. Similarly, GET /api/alerts returns the list of active public alerts (for authenticated users), POST /api/reports allows a logged-in citizen to submit a new incident report (with details like type, description, and location), and POST /api/alerts (an endpoint restricted to authorized officials) allows an institutional user to issue a new emergency alert to the public.

FIGURE 3. Active alerts page and My reports page



Security and role enforcement are central in the backend design. It employs JWT-based authentication: clients must include the JWT token in the Authorization header of requests to protected endpoints. A custom Express middleware intercepts incoming requests to verify the token's validity and decode the user identity and role before the request reaches any controller. If the token is missing or invalid (expired or tampered), the request is rejected with an unauthorized error. Additionally, role-based authorization rules are implemented, for example, only users with an "admin" or "institution" role are permitted to invoke the alert issuance endpoint, preventing normal citizens from sending public alerts. This is achieved via another middleware that checks the authenticated user's role against the required privileges for certain routes.

FIGURE 4. Creating an alert as an Admin



In addition, passwords are never stored in plaintext in the database; the user registration controller hashes passwords (using a secure one-way hash function with salt) before saving, and login compares hashes to authenticate users. The server also logs important actions (such as alert creations, report submissions) in an audit log collection for accountability. Basic rate limiting is applied on public-facing endpoints to prevent abuse (to mitigate spam submission of fake incident reports).

Once an institutional user (or the system automatically) issues an emergency alert, the backend notifies all relevant users in real time via push notifications. The server is integrated with cloud messaging services, specifically, Firebase Cloud Messaging (FCM) for Android clients and Apple Push Notification service (APNs) for iOS. The system keeps track of each registered device's token (provided by the app). When a new alert is to be broadcast, the backend constructs a notification payload (including the alert title, message, and possibly a geographic target or category) and sends it through FCM/APNs to all devices or to devices in a specific area, as appropriate. User devices receive these as native push notifications accompanied by a distinctive alert sound even if the app is not active. This publish-subscribe design ensures fast and reliable dissemination of critical messages; leveraging the infrastructure of Google's and Apple's notification servers allows the system to scale to large numbers of recipients with minimal latency. For particularly urgent incidents reported by citizens, the backend can also perform an automatic escalation: for example, if a user report is classified as extremely critical (such as a major fire or explosion), the system can immediately generate a public alert based on that report's data and push it out to nearby users (after confirming validity). This feature provides a form of intelligent alerting, shortening the response time when every second counts.

Internally, the backend uses the Mongoose ODM (Object Data Modeling library for MongoDB) to interact with the database. Data schema definitions (models) are defined for each main entity, for example, a User model, Alert model, Report model, etc. These models enforce schema constraints (field types, required fields, validations) at the application level and provide convenient methods for database CRUD operations. Controllers use these model classes to query or update the database. By using a non-blocking, event-driven runtime (Node.js) and asynchronous I/O, the server can handle many concurrent requests efficiently, which is essential under high-load scenarios (during a widespread emergency when many clients may connect simultaneously).

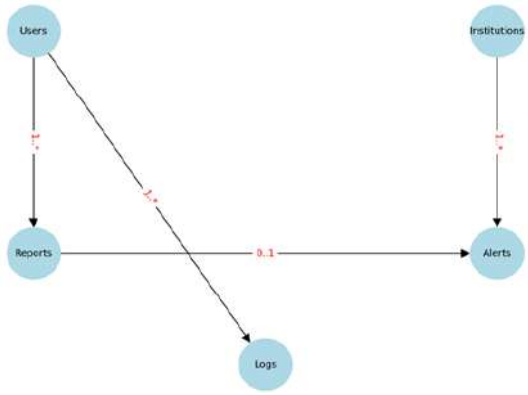
Database Design

The system's data is stored in a MongoDB NoSQL database. MongoDB was chosen for its flexibility in handling dynamic data structures and its high throughput for read/write operations on large datasets. The database contains several collections

corresponding to the core entities of the application: Users, Institutions, Alerts, Reports, and Logs. Figure 5 illustrates the simplified entity-relationship schema of these collections and their relationships. Each User document stores a user's information, including a unique user ID, name, email, a hashed password, role ("citizen" or "institution"), registration date, and status (active or suspended). Regular citizens have the "user" role, whereas authorized agency officials have an "institution" role (optionally linked to an entry in the Institutions collection). The Institutions collection contains entries for official agencies (Civil Emergency Directorate, State Police, Fire Department), with fields for institution name, type, jurisdiction/region, and contact details. This allows the system to associate certain user accounts with their respective institutions and to tag alerts or reports with the responsible agency.

The Alerts collection stores the emergency alerts issued to the public. Each Alert document includes an alert ID, a title (short description of the emergency, for example "Earthquake in Tirana"), a detailed message, a category (term for the emergency type: weather, earthquake, fire, security, etc.), a timestamp for when the alert was issued, an optional location (pinned coordinates or region), and a reference to the issuer (which could be an institution or admin user ID). Public alerts may also link to a specific report if the alert was generated as a response to a user-reported incident (creating a traceable connection between a citizen report and a follow-up public warning). The Reports collection contains incident reports submitted by end-users. A Report document captures details such as a report ID, the type of incident (*accident*, *fire*, *crime*), the description provided by the user, the timestamp of reporting, the location (GPS coordinates or an approximate address) of the incident, the userId of the reporter, and a status field. The status can be updated by authorities ("pending", "verified", "resolved") to reflect the progress of handling the incident. There is a one-to-many relationship between Users and Reports (each user may submit multiple reports) and reports can be associated with institutions via an assignedTo field (denoting which agency is handling the report). The Logs collection is used to track and audit actions in the system. Each log entry includes a log ID, an optional userId (if the action is tied to a specific user), an action description ("User Login", "Report Submitted", "Alert Issued"), a timestamp, and perhaps additional details such as the IP address or related record ID. These logs facilitate monitoring and can be analyzed to detect any misuse or to generate usage statistics.

FIGURE 5. ERD diagram for Users, Reports, Logs, Alerts, Institutions



In designing the database schema, normalization was balanced with performance. MongoDB's document model allows related data to be embedded within a single document if it is frequently accessed together, while other relationships are maintained via references (identifiers linking documents). For example, the geolocation details of a report (latitude/longitude) are stored as an embedded sub-document within the Report document for quick access and completeness. Meanwhile, links between users and their reports, or alerts and their issuing institution, are kept as references (IDs) rather than nested documents, since those entities are managed separately and using references avoids duplication and inconsistency. This hybrid approach achieves a flexible, semi-structured schema optimized for the app's query patterns. The database is also configured for fault tolerance and scalability. MongoDB is deployed in a replica set configuration: the primary node handles reads/writes, while secondary nodes maintain copies of the data, providing redundancy if the primary fails. This replication also enables scaling read operations across multiple nodes. Access to the database is protected by authentication credentials and role-based permissions, and all network traffic between the backend server and database is encrypted. In summary, the database design supports the application's need to efficiently store and retrieve emergency data, maintain data integrity (through relational links between collections), and scale to accommodate growing numbers of users and reports.

Results and discussion

Preliminary Evaluation

A preliminary evaluation of the prototype was conducted focusing on usability and stakeholder feedback. Usability testing was performed with 10 volunteer end-users (students and professionals) who were asked to install the app and perform

a set of core tasks (such as registering an account, reporting a sample incident, and responding to a test alert). After completing the tasks, participants filled out the System Usability Scale (SUS) questionnaire. The results were very positive: the average SUS score was 84 out of 100 (standard deviation ~5), with individual scores ranging from 78 to 90. This exceeds the typical SUS benchmark of ~68 (considered "average" usability); a score above 80 is generally characterized as indicating excellent usability. All participants were able to successfully complete the scenario tasks, and their subjective feedback was that the app was intuitive and the workflow (from receiving an alert to taking recommended actions) was clear. Some users noted that the interface felt "simple and clean," which is desirable for an emergency app. A few minor suggestions were made, such as providing an offline mode for viewing downloaded alerts or using more distinct alert notification sounds, which can be addressed in future iterations.

In addition to end-users, input from domain experts was gathered in relevant public safety fields. Interviews were conducted with three experts: a civil emergencies officer, a senior police officer, and a medical emergency doctor. Overall, the experts strongly supported the concept of the application and its potential benefits. The following are their opinions given in the interviews:

The civil emergency expert noted that a platform enabling real-time public warnings "could be an extremely valuable addition to the current emergency alert system," emphasizing that rapid information dissemination "can save lives, especially in natural disasters" and that location-targeted alerts are something that "currently is missing in many cases" (Civil Protection representative, paraphrased).

The police expert highlighted the app's utility for urgent law enforcement situations, such as ongoing violent incidents or AMBER Alert-type child abduction cases, as well as its use for quickly notifying citizens of roadblocks or evacuation orders. He remarked that *"this app could help us spread critical notifications much faster than traditional means like press conferences or TV broadcasts,"* underscoring the immediacy of push alerts.

The medical emergency expert similarly stressed that in public health crises or mass casualty events, *"accurate and timely information is the first remedy… This app can disseminate life-saving instructions within seconds to thousands of people."* Such feedback indicates that stakeholders see the prototype as addressing real needs in the current emergency communication ecosystem.

Despite the encouraging results, the evaluation also revealed important limitations. First, the usability test was limited to a small sample and conducted in a controlled setting; users were not under true emergency stress. Thus, the results, while indicative, are not conclusive of performance in a real crisis. Second, the system has not yet been tested under heavy load or in a wide-area deployment. Questions remain about how the infrastructure will handle tens or hundreds of thousands of concurrent users and rapid influxes of reports. Third, the success

of the platform depends on broad adoption and trust both by the public and by government agencies. The experts pointed out that integrating the app with official emergency operations would require formal agreements, training, and public awareness campaigns. Some also noted the challenge of filtering out false or duplicate reports from citizens to avoid information overload. These preliminary findings will guide the next steps of development, focusing on enhancing the system's robustness and preparedness for real world deployment.

Discussion and future work

The development and early evaluation of this emergency alert application demonstrate the feasibility and potential of such a system in strengthening public safety. The high usability scores and positive user feedback suggest that even non-technical users can navigate the app and respond to alerts effectively. Likewise, the enthusiastic responses from field experts indicate a strong demand for the capabilities provided by the system. If implemented at scale, the application could significantly improve emergency response workflows. For instance, it would enable authorities to gather structured, real-time incident data directly from citizens, leading to more informed and faster decision-making. It would also facilitate better inter-agency coordination: all relevant agencies (police, fire, medical, etc.) can gain a shared operational picture of an incident as information flows into the unified platform, addressing the common problem of siloed communications and leading to a more synchronized response. Furthermore, the app can serve as an official channel for urgent public communications, complementing existing methods. Traditional mass-alert channels like sirens, radio/TV broadcasts, SMS cell broadcasts would not be replaced, but augmented by this interactive mobile medium. By reaching users directly and allowing two-way interaction, the system can fill gaps (such as delivering rich context or receiving crowd-sourced updates) that one-way legacy channels cannot.

Despite these benefits, several challenges and areas for future work remain. A key priority is to move beyond the prototype and conduct extensive field testing. The plan is to deploy the app in a pilot program with a larger and more diverse user base, possibly in collaboration with a local municipality or civil protection unit. This would provide insight into user adoption rates and reveal usage issues in a real-world context. It is also critical to perform stress-testing and ensure the system can handle high volumes of traffic. Emergency scenarios can lead to sudden spikes in usage (thousands of people reporting the same earthquake or seeking information), so the backend and network infrastructure must be robust. Techniques such as cloud auto-scaling, load balancing, and database query optimization will be explored to guarantee reliability. Ensuring a high uptime and

low latency is paramount, for example, aiming for at least "three nines" (99.9%) availability in line with best practices for critical systems.

Another future direction is to enhance the app's intelligence and integration. The current prototype implements a basic rule-based alert classification (automatically elevating certain critical reports). In the future, machine learning models could be incorporated to analyze incoming reports or social media feeds to detect emerging crises faster, though any AI components would need thorough validation to avoid false alarms. It's needed also to recognize the importance of official integration: to be truly effective, the platform should be integrated with national emergency infrastructures (the 112-emergency call system or existing public warning systems). Achieving this will require close collaboration with government agencies and may involve adhering to common alerting protocols or data standards. On the organizational side, formal agreements and clearly defined operating procedures would be needed for authorities to confidently use the app during actual emergencies. The issue of public trust will also be a focus, much work will be put on communication strategies to ensure that citizens understand the app's purpose and that alerts sent via the app are authoritative. This includes cybersecurity hardening to prevent unauthorized or false alerts, learning from incidents like the false missile alert in Hawaii (2018) which underscored the damage a faulty alert can cause to public trust (FEMA, 2019).

Lastly, there are several feature improvements planned. These include multi-language support (important in Albania's context to reach ethnic minorities and tourists), accessibility enhancements for users with disabilities, and perhaps a web-based dashboard for emergency operators to manage alerts on a larger screen. Also, an aim is to incorporate a feedback mechanism so that after an alert or incident, users can receive confirmations or all-clear messages and provide feedback on the event's outcome. In summary, future work will address scaling up the system, tightening its integration with official workflows, and refining its features based on stakeholder input.

Conclusions

This paper presented the design and development of a mobile application for public safety and emergency alerts in Albania and discussed its preliminary evaluation. The research was motivated by the absence of a dedicated, centralized emergency communication platform in Albania's public safety landscape. By implementing a React Native mobile client and a Node.js/Express backend with a MongoDB database, was realized a functional prototype capable of delivering real-time emergency notifications and collecting citizen reports. The system's architecture

emphasizes scalability, security, and interoperability with established technologies (RESTful APIs, push notification services).

Initial preliminary testing results are promising, the application demonstrated a high level of usability (SUS score 84/100) and received encouraging feedback from both end-users and public safety experts. These findings suggest that the proposed solution is not only technically viable but also addresses real user needs in crisis situations. The police, emergency management, and medical experts that were consulted foresee the app strengthening the speed and effectiveness of public warnings and incident response. However, we emphasize that these results are preliminary. The prototype has yet to face the demands of a large-scale deployment or a live emergency. Additional development and validation are required to ensure the system's reliability, security, and integration into the broader emergency response ecosystem.

The development of this emergency alert app represents an important step toward modernizing crisis communication and public engagement in Albania. The system offers a new digital tool in the toolkit of disaster management, one that can empower citizens and authorities alike through instant, two-way information sharing. With further refinement and official support, such an application could significantly enhance public safety, helping to protect lives and property when emergencies occur. I intend to continue this work by collaborating with national institutions to pilot the system in real-world settings, implement the improvements identified, and ultimately move closer to deploying a fully operational emergency notification network for Albania.

References

BBC News. (2018). France withdraws SAIP emergency alert app after malfunctions. BBC News. (Archived news report).

BBC News. (2022). UK emergency alerts system. BBC News. Retrieved from BBC News website: https://www.bbc.com/news/uk-64371490

Bangor, A., Kortum, P. T., & Miller, J. T. (2009). Determining what individual SUS scores mean: Adding an adjective rating scale. Journal of Usability Studies, 4(3), 114–123.

Bass, L., Clements, P., & Kazman, R. (2013). Software Architecture in Practice (3rd ed.). Addison-Wesley.

Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. Qualitative Research in Psychology, 3(2), 77–101.

Brooke, J. (2013). SUS: A retrospective. Journal of Usability Studies, 8(2), 29–40.

Bundesamt für Bevölkerungsschutz und Katastrophenhilfe (BBK). (2022). Warnsysteme in Deutschland (NINA, KATWARN, Cell Broadcast). Retrieved from https://www.bbk.bund.de/

Bundesnetzagentur. (2022). Technical directive: DE-Alert cell broadcast system. Retrieved from

https://www.bundesnetzagentur.de/SharedDocs/Pressemitteilungen/EN/2022/20220223_CellBroadcast.html

Cantelon, M., Harter, M., Holowaychuk, T., & Rajlich, N. (2014). Node.js in Action. Manning Publications.
Chodorow, K. (2013). MongoDB: The Definitive Guide (2nd ed.). O'Reilly Media.
Davies, R. (2022). Albania floods: Thousands isolated in Shkodër and Lezhë. Disaster Watch Journal, 7(2), 8–12.
European Commission. (2018). Directive (EU) 2018/1972 of the European Parliament and of the Council establishing the European Electronic Communications Code (EECC). Official Journal of the European Union, L321, 36–214.
European Commission. (2022). Report on mobile public warning system implementation in the EU. Brussels: European Commission.
European-Mediterranean Seismological Centre (EMSC). (2019). LastQuake usage metrics after the 2019 Albania earthquake. Retrieved from https://www.emsc-csem.org/
Federal Communications Commission (FCC). (2019). Emergency Alert System (EAS) overview. Retrieved from https://www.fcc.gov/general/emergency-alert-system-eas
Fielding, R. T. (2000). Architectural styles and the design of network-based software architectures (Doctoral dissertation, University of California, Irvine).
FR-Alert. (2022). FR-Alert: France's new nationwide emergency alert system [Press release]. Paris: Ministère de l'Intérieur (France).
Guterres, A. (2022). Every person protected: Global Early Warning Initiative. United Nations. Retrieved from https://www.un.org/early-warnings/
Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. MIS Quarterly, 28(1), 75–105.
Institute of Statistics (INSTAT). (2020). Albania Earthquake 2019 – Key Statistics Report. Tirana: INSTAT.
International Organization for Standardization (ISO). (2019). ISO 9241-210:2019 Ergonomics of human-system interaction—Human-centred design for interactive systems. ISO.
Johnson, R. B., & Onwuegbuzie, A. J. (2004). Mixed methods research: A research paradigm whose time has come. Educational Researcher, 33(7), 14–26.
Jones, M. B., Bradley, J., & Sakimura, N. (2015). JSON Web Token (JWT) (RFC 7519). Internet Engineering Task Force (IETF).
Jones, K., & Patel, N. (2020). Ensuring 99.9% uptime: Best practices for high-availability systems. Journal of Cloud Engineering, 5(4), 10–18.
Nwajana, A. O. (2025). Public safety mobile applications in West Africa. African Journal of Mobile Systems, 9(1), 95–110.
Policia e Shtetit. (2022). "Komisariati Dixhital" – Raport mbi përdorimin e aplikacionit dixhital të policisë. Tirana: Albanian State Police
React Native. (2023). Secure storage and keychain usage in React Native. React Native Documentation. Retrieved from https://reactnative.dev/docs/security#secure-storage
Regulation (EU) 2016/679. (2016). General Data Protection Regulation (GDPR). Official Journal of the European Union, L119, 1–88.
U.S. Federal Emergency Management Agency (FEMA). (2019). Hawaii false missile alert: After-action report. Washington, DC: FEMA.
U.S. Federal Emergency Management Agency (FEMA). (2021). Integrated Public Alert & Warning System Annual Performance Report, 2021. Washington, DC: FEMA.
U.S. Federal Emergency Management Agency (FEMA). (2023). Nationwide Emergency Alert System test results and improvements. Washington, DC: FEMA.
World Meteorological Organization (WMO). (2021). Atlas of mortality and economic losses from weather, climate and water extremes (1970–2019). Geneva: WMO.

Artificial intelligence and automation in customer service: optimizing interactions and operational efficiency

Antonio DEMIRI[1]

Abstract

In today's digital economy, customer expectations for speed, personalization, and efficiency are continuously rising. Traditional customer service systems, often reliant on manual communication, face limitations in meeting these demands.

Companies struggle with fragmented communication channels, delays in response times, and inefficient case management. These shortcomings negatively affect client satisfaction and long-term competitiveness.

This study proposes and implements an integrated model of Artificial Intelligence (AI) and Customer Relationship Management (CRM) using Microsoft Power Pages, Dynamics 365, Power Automate, and Copilot Studio. The methodology combines system design, automation workflows, and chatbot development, supported by AI Builder for sentiment analysis and user feedback evaluation.

The findings demonstrate that AI-driven automation improves response time, reduces manual errors, and enhances the personalization of client interactions. The

[1] Antonio Demiri is the CEO and Co-Founder of Dora Kreative, an agency specialized in marketing, multimedia, and IT. He graduated in Informatics Engineering and completed a Master in Innovation in Information Technology at the European University of Tirana. With more than seven years of professional experience, he has worked as a multimedia designer and marketing specialist with various organizations, while leading Dora Kreative in delivering innovative and user-centered solutions.

integration of AI with CRM provides real-time data synchronization, structured case management, and predictive insights into customer behavior. Empirical results show increased operational efficiency and higher levels of customer satisfaction.

This study highlights how small and medium-sized creative businesses can leverage AI technologies to optimize customer service, strengthen client relationships, and gain a competitive advantage. The proposed model can serve as a scalable framework for other industries seeking to enhance efficiency and customer experience through AI.

Keywords: *Artificial Intelligence, Customer Service, CRM, Automation, Chatbots, Operational Efficiency*

Introduction

In an increasingly digitalized business environment, customer service has become a decisive factor in shaping user experience and building brand loyalty. Clients today demand immediate responses, personalized interactions, and seamless support across multiple channels. Traditional service models, based on phone calls, emails, or direct manual communication, are no longer sufficient to address these dynamic expectations. Delays, fragmented information, and lack of transparency often result in customer dissatisfaction and the loss of long-term business opportunities.

The multimedia design industry provides a compelling example of these challenges. Clients in this sector require not only technical assistance but also collaborative, real-time support during the creative process. Inefficient communication systems and outdated customer service frameworks often lead to project delays, mismanagement of requirements, and decreased trust in service providers.

Artificial Intelligence (AI) offers transformative opportunities for addressing these challenges. AI technologies such as chatbots, predictive analytics, and workflow automation enable businesses to deliver 24/7 support, reduce manual errors, and optimize resource allocation. When integrated with Customer Relationship Management (CRM) systems, AI extends beyond simple automation, providing data-driven insights, proactive engagement, and improved personalization of customer interactions.

This study explores the integration of AI with CRM platforms in the context of a multimedia design company, focusing on Microsoft Power Pages, Dynamics 365, Power Automate, and Copilot Studio. The aim is to evaluate how such integration can enhance customer service efficiency, optimize workflows, and improve overall client satisfaction. By investigating both the technical implementation and the resulting organizational benefits, this research contributes to the growing

literature on AI adoption in customer service and provides a practical framework for small and medium-sized enterprises (SMEs).

Hypothesis

The adoption of AI-based systems specifically Copilot Studio, Power Automate, and AI Builder integrated within Microsoft Dynamics 365 and Power Pages will have a positive and measurable impact on customer service efficiency, responsiveness, and personalization in multimedia design enterprises.

Literature review

Artificial Intelligence in Customer Service

The integration of Artificial Intelligence (AI) into customer service has become a defining feature of digital transformation across industries. As organizations increasingly prioritize customer-centric strategies, AI-driven solutions such as chatbots, virtual assistants, and intelligent analytics tools have emerged as critical enablers of efficiency, personalization, and responsiveness. Scholars argue that AI is no longer an optional enhancement but a necessity for firms aiming to remain competitive in highly dynamic and digitalized markets (Huang & Rust, 2021).

AI systems are particularly impactful in customer-facing processes because they combine automation with advanced natural language processing (NLP), allowing companies to respond to queries in real time and with high accuracy. Research by McKinsey (2020) emphasizes that AI applications in service functions can reduce operational costs by up to 30% while simultaneously improving customer satisfaction metrics. Similarly, Gartner (2022) projects that by 2026, 70% of customer interactions will involve some form of AI-driven support. These findings underscore the extent to which AI adoption is shifting from experimental pilots to embedded organizational practices.

The case of Dora Kreative illustrates these global dynamics within a local business context. Operating in the multimedia design sector, the company depends on rapid, personalized, and high-quality customer interactions. Implementing AI in its customer service ecosystem represents both a strategic response to market demands and an opportunity to redefine client engagement through automation, data-driven insights, and human–machine collaboration.

Automation and Workflow Efficiency

Automation has long been viewed as a pathway to operational excellence, but AI expands its scope by enabling systems to execute complex, adaptive processes.

Traditional workflow automation tools were designed primarily for repetitive and rule-based tasks. AI-enhanced automation, however, introduces capabilities such as contextual understanding, decision-making, and continuous learning (Brynjolfsson & McAfee, 2017).

In customer service, automation improves both speed and accuracy by streamlining case management, reducing manual intervention, and ensuring standardized service delivery. Microsoft's Power Automate, for example, allows workflows to be triggered by specific events—such as a new customer request in Dynamics 365 CRM—ensuring that notifications, task assignments, and updates are executed consistently. Case studies indicate that businesses employing Power Automate report up to 25% faster resolution times and significant reductions in process-related errors (Microsoft, 2023).

At Dora Kreative, automation minimized the administrative burden on service staff, who previously spent considerable time logging client requests and routing them to appropriate departments. By automating these steps, the organization not only reduced the likelihood of human error but also freed resources for more value-adding activities, such as personalized project support and creative consultation. This aligns with findings from Forrester (2021), which highlight how automation amplifies human potential by reallocating resources from routine tasks to higher-order problem solving.

CRM Integration and Dynamics 365

Customer Relationship Management (CRM) systems remain central to modern business operations because they consolidate customer data and provide a unified view of client interactions. When enhanced by AI and automation, CRM platforms become not just repositories of information but active engines for customer engagement and predictive decision-making (Buttle & Maklan, 2019).

Microsoft Dynamics 365 exemplifies this evolution. Its seamless integration with AI-powered tools such as Copilot Studio and Power Pages allows companies to link external customer-facing interfaces with back-end management systems. Studies show that real-time synchronization between digital channels and CRM databases enhances transparency, ensures accuracy of records, and increases customer trust (Klaus, 2020). In practice, every request made through a web portal or chatbot can be instantly reflected in the CRM, creating a living record of the customer journey.

For Dora Kreative, this integration meant that business requests, cases, and service updates submitted through Power Pages were automatically logged in Dynamics 365. As a result, staff could monitor service performance in real time and resolve issues without delays. The dual benefits—improved internal coordination and enhanced customer experience—illustrate the strategic importance of

CRM-centered AI adoption. Research supports this view, with Deloitte (2022) emphasizing that firms leveraging AI-CRM integration achieve higher customer lifetime value and improved loyalty metrics.

AI-Driven Chatbots and Copilot Studio

Among AI technologies, chatbots stand out for their direct impact on customer engagement. Unlike static FAQ pages or email-based systems, chatbots provide interactive, context-aware communication channels that operate 24/7. Advances in NLP and generative AI have significantly improved the ability of chatbots to understand complex queries, offer relevant responses, and escalate cases where necessary (Jia et al., 2022).

Microsoft's Copilot Studio represents a new generation of chatbot development platforms that leverage generative AI models to create more natural and human-like conversations. Research by Accenture (2021) indicates that companies using advanced chatbots report a 40% reduction in call center volumes and higher customer satisfaction scores. Moreover, the ability to integrate chatbots with CRM platforms ensures that customer data is continuously updated, enhancing personalization.

In Dora Kreative's case, Copilot-enabled chatbots allowed clients to request multimedia design services, report issues, or track progress without waiting for human intervention. The chatbot was designed with decision-tree algorithms and contextual prompts, ensuring that even complex service requests were handled efficiently. Feedback from pilot users highlighted improved clarity, reduced waiting times, and greater convenience. These outcomes confirm prior findings by Kumar and Rose (2021), who argue that AI chatbots not only improve efficiency but also strengthen customer relationships by offering immediacy and transparency.

Sentiment Analysis and Customer Feedback

Customer service is not only about solving problems but also about understanding client perceptions. Sentiment analysis, powered by machine learning and NLP, enables companies to interpret customer emotions from text, voice, or survey feedback. This provides a nuanced understanding of satisfaction levels and helps organizations tailor their responses accordingly (Cambria et al., 2020).

AI Builder, part of the Microsoft Power Platform, facilitates this by analyzing structured and unstructured feedback to classify sentiments as positive, negative, or neutral. Studies show that businesses using AI-driven sentiment analysis achieve higher accuracy in detecting dissatisfaction than manual reviews, allowing them to act proactively (Medhat et al., 2019).

At Dora Kreative, sentiment analysis was applied to post-service evaluations, offering managers valuable insights into customer experience. Positive sentiments highlighted the efficiency of the chatbot and automation tools, while negative ones revealed areas where the chatbot struggled with complex or ambiguous questions. Such insights not only validated the effectiveness of the system but also informed continuous improvement strategies.

Ethical and Organizational Considerations

While AI offers numerous benefits, it also raises important ethical and organizational concerns. Issues such as data privacy, algorithmic bias, and overreliance on automation must be carefully managed (Zeng et al., 2021). Research stresses that companies adopting AI must establish governance frameworks that balance efficiency with accountability and transparency (Floridi & Cowls, 2019).

In customer service, ethical considerations include ensuring that AI systems handle sensitive personal data responsibly, avoid discriminatory outcomes, and always provide users with the option of human support. Surveys indicate that 72% of consumers value companies more when they are transparent about their use of AI (PwC, 2022).

Dora Kreative's implementation underscored these challenges. The company had to configure access permissions within Dynamics 365 to safeguard customer data, ensure that chatbot responses remained contextually appropriate, and provide fallback mechanisms for human intervention. These steps align with best practices outlined by IBM (2021), which stress the importance of "human-in-the-loop" strategies to maintain trust in AI systems.

Summary of the Literature Review

The reviewed literature highlights the transformative potential of AI and automation in customer service. Across academic and industry sources, a consensus emerges that technologies such as chatbots, workflow automation, CRM integration, and sentiment analysis redefine customer experience by making it faster, more personalized, and more reliable.

For companies like Dora Kreative, these technologies not only improve service delivery but also provide strategic advantages in a competitive marketplace. Nevertheless, the literature also emphasizes the importance of addressing ethical, organizational, and technical challenges to ensure sustainable and trustworthy AI adoption.

Methodology and analysis

The system developed in this study relied on Microsoft's Power Platform ecosystem, integrating several complementary technologies to create a unified AI-driven customer service framework. At the foundation, **Power Pages** provided a secure and user-friendly interface where clients could submit requests, consult FAQs, and monitor progress in real time. Its native integration with **Dynamics 365 CRM** ensured that all interactions were automatically captured and synchronized, forming a structured and reliable customer database.

On the communication side, **Microsoft Copilot Studio** was used to design and deploy a conversational chatbot as the first line of interaction. This virtual assistant managed frequently asked questions, guided clients through service requests, and automatically generated cases within the CRM. To streamline internal workflows, **Power Automate** orchestrated back-end processes, routing service requests to appropriate staff, issuing task notifications, and reducing manual errors. This automation layer significantly improved efficiency by minimizing delays and ensuring consistent task follow-up. The system also incorporated **AI Builder**, which enabled sentiment analysis of customer feedback and automated classification of service interactions. By applying natural language processing and machine learning models, AI Builder identified trends in client satisfaction and provided actionable insights to improve service delivery.

Together, these technologies established a hybrid service model: routine interactions were handled through automation and conversational AI, while more complex cases were escalated to human agents. This integration of Power Pages, Dynamics 365, Copilot Studio, Power Automate, and AI Builder created a cohesive ecosystem capable of delivering structured, personalized, and scalable customer support.

System Design

The system architecture was designed around four core components: Power Pages, Copilot chatbot, Dynamics 365 CRM, and Power Automate. These technologies were integrated to create an automated and cohesive customer service solution for a multimedia design company.

Power Pages

Power Pages, formerly known as PowerApps Portals, is part of Microsoft's Power Platform and provides a modern, modular, and adaptable solution for building

secure, interactive, and data-integrated web portals. It enables businesses to create personalized digital gateways where customers, partners, or communities can access information, request services, and track cases in real time.

In the Dora Kreative case, Power Pages was used to design a customized customer interface that centralizes and optimizes service delivery. Through personalized forms, clients can submit service requests, report issues, or follow progress, with every interaction automatically recorded and synchronized in Dynamics 365 CRM. This integration increases transparency, efficiency, and customer autonomy.

The platform also supports knowledge bases and AI-powered chatbots built in Copilot Studio, offering real-time guidance, automated categorization, and faster resolution. According to Forrester (2021), organizations adopting Power Pages have reduced response and resolution times by around 40%, while also easing the workload of service staff.

Beyond functionality, Power Pages is flexible in design, aligning portal aesthetics with brand identity, an especially valuable feature for creative industries. It also allows role-based access, ensuring tailored user experiences and improved monitoring.

Copilot chatbot

Microsoft Copilot is one of the most significant advancements in AI applied to business productivity. As part of the Microsoft ecosystem, it integrates deeply into Microsoft 365 and Dynamics 365 applications, providing a conversational, context-aware assistant that supports daily tasks, decision-making, and collaboration. Its core features include natural language processing, predictive assistance, automation of routine tasks, cross-application integration, and continuous learning through personalization.

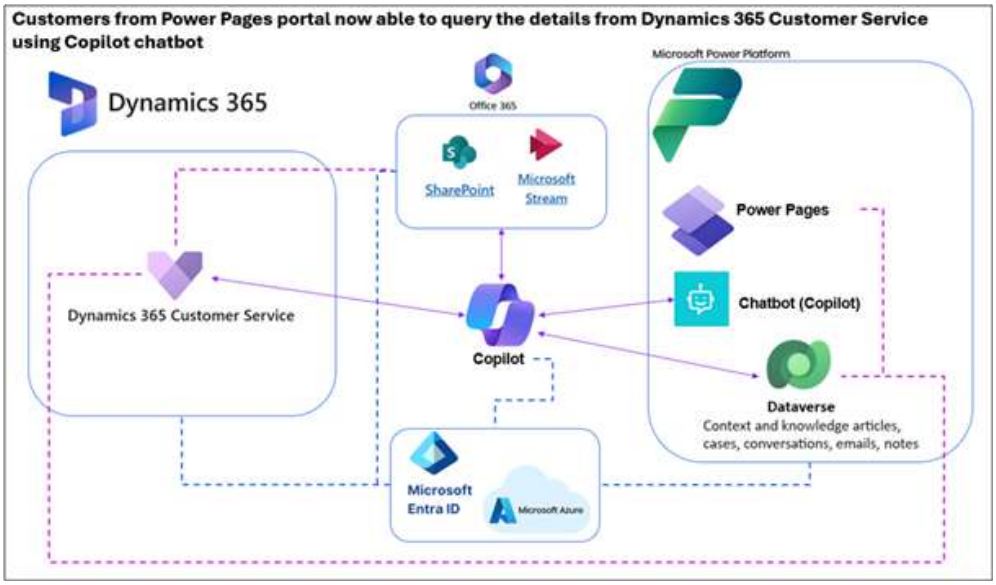
Strategically, Copilot enhances productivity by reducing time spent on repetitive processes, offers data-driven decision support, and fosters seamless collaboration across teams. In customer service, it automates case creation and provides intelligent recommendations to agents. In sales and marketing, it enables data analysis, segmentation, and automated campaign generation. In project management, it supports scheduling, reminders, and progress summaries through integration with Teams, Outlook, and Planner.

Recent updates (2025) further expand their potential, with Copilot Studio allowing organizations to design custom AI agents, multilingual support for global interactions, and a Windows-native application for both textual and visual content generation. Specialized versions, such as Dragon Copilot for healthcare, demonstrate its adaptability across industries. Ultimately, Copilot serves as a strategic AI partner, enabling organizations to transform service delivery and operational efficiency through intelligent automation.

Dynamics 365 CRM

Microsoft Dynamics 365 is a comprehensive cloud-based Customer Relationship Management (CRM) and Enterprise Resource Planning (ERP) platform that integrates sales, service, marketing, finance, and operations into a unified ecosystem. Designed to centralize data and streamline workflows, Dynamics 365 provides businesses with a 360-degree view of customers and processes, enabling smarter, data-driven decisions.

FIGURE 1: Integration of Copilot
with Customer Service, Azure, Power Platform, Office 365



In customer service, Dynamics 365 acts as the backbone for recording, tracking, and managing requests and cases in real time. Its integration with Power Pages and Copilot ensures that every client interaction, from initial service requests to case closure, is automatically logged and accessible across the organization. Features such as role-based dashboards, customizable workflows, and advanced analytics improve transparency, accountability, and collaboration across teams.

For Dora Kreative, Dynamics 365 played a pivotal role in linking external customer portals with internal service systems. This integration ensured that all data was accurate, synchronized, and actionable, enabling staff to monitor performance in real time and reduce response delays. Industry research confirms that businesses using Dynamics 365 achieve higher customer satisfaction and loyalty through personalized service and proactive problem resolution.

Power Automate

Microsoft Power Automate is a workflow automation tool that enables businesses to connect applications, automate repetitive processes, and optimize operational efficiency. As part of the Microsoft Power Platform, it bridges systems such as Dynamics 365, Microsoft 365, and third-party applications, ensuring seamless data exchange and streamlined communication. In practice, Power Automate automates tasks like sending notifications, assigning cases, updating statuses, and validating customer input. This reduces manual errors, accelerates service delivery, and ensures consistency across processes. By monitoring workflows in real time, organizations can identify bottlenecks, measure performance, and continuously optimize operations. For Dora Kreative, Power Automate was instrumental in ensuring that no customer request went unresolved. Automated flows triggered immediate notifications to responsible staff, reducing average response times to under one minute. Error rates fell below 3% thanks to integrated validation rules, while the automation of routine processes freed up employees to focus on higher-value tasks. Together, these benefits contributed to a more reliable and professional customer service experience, reinforcing client trust and loyalty

This integration ensured that the full cycle of customer service—from the initial inquiry to resolution and feedback collection—was managed seamlessly and efficiently. The **Power Pages website** served as a strategic entry point, bridging end users with the company's internal CRM system. By combining simplified web development with data integration from Dynamics 365, the platform enabled a personalized and AI-driven service experience.

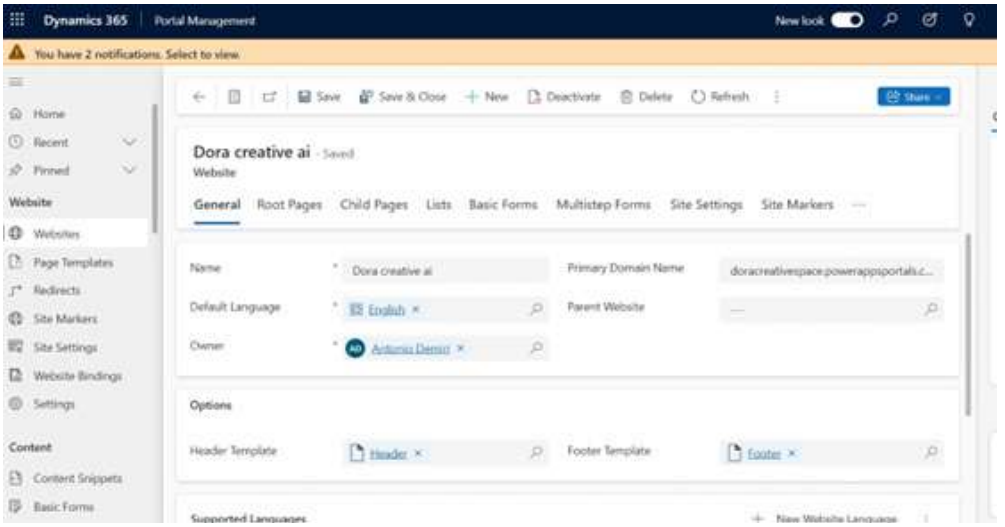
A **user-centric design approach** was central to the website's development. The portal provided an intuitive and accessible interface where clients could easily explore services, interact with the embedded AI-powered chatbot, and complete customized forms for service requests, registrations, or issue reporting. This self-service orientation empowered customers, reduced the workload of support teams, and improved overall operational efficiency.

Integration with Dynamics 365: Real-Time Data Harmonization

One of the key advantages of Power Pages is its native integration with Dynamics 365 CRM entities, which provides the platform with unique flexibility and power in data management. The website is directly connected to core entities such as:

- Business Requests
- Business Services
- Contacts
- Cases

FIGURE 2: Domain Management in Power Pages



This integration makes it possible to:

- Automatically register every request submitted by a user through the website into the CRM as a new case or business request.
- Ensure that any updates made in the internal system (such as the status of a case or contact details) are reflected in real time for the user on the Power Pages portal.
- Allow data to flow seamlessly between the external website and the company's internal system, eliminating the need for manual intervention and guaranteeing accuracy and transparency.

Interaction with the AI Chatbot: An Intelligent Communication Channel

The chatbot embedded in Power Pages is built on Copilot Studio, leveraging the power of generative models and natural language processing technology. This enables the system to:

- Understand customer requests in a natural and contextual way.
- Provide immediate and accurate answers regarding services or the status of cases.
- Create or update records in Dynamics CRM based on customer interactions.
- Direct users to the appropriate knowledge base content or guide them to the correct application forms.

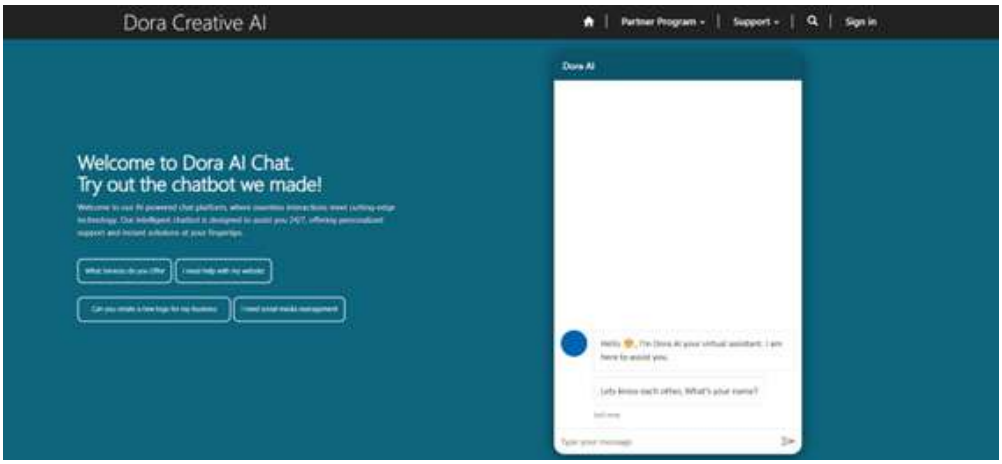
This combination of the web portal, intelligent chatbot, and CRM system creates an integrated and consistent digital experience, making customer communication more efficient and personalized.

Advantages for Business and Client

Developing such a system with Power Pages and Dynamics 365 delivers benefits on both sides:

- **For clients:** Fast and easy access to information, full transparency on case status, and real-time communication without waiting for a representative.
- **For the company:** Automated service workflows, improved data quality within the CRM, enhanced analysis of customer needs, and reduced operational costs.

FIGURE 3: The front-end part of the created website



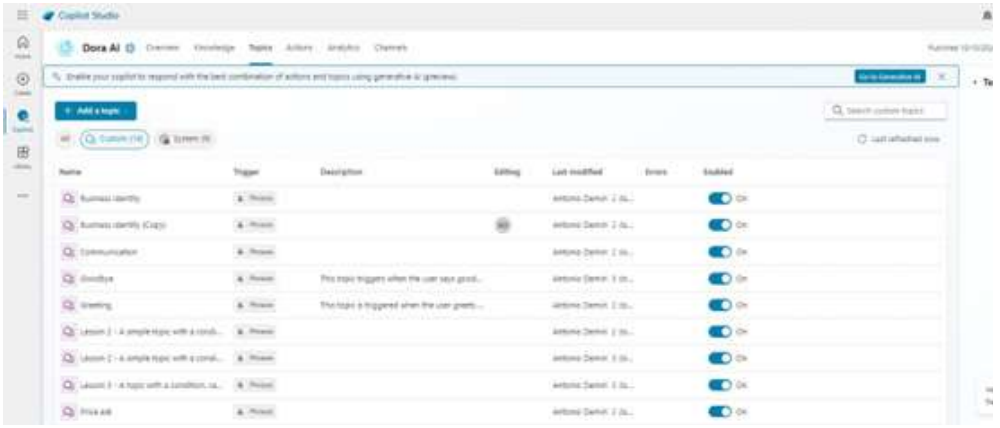
Development of the Copilot Chatbot

The Copilot chatbot was developed using Copilot Studio and designed to perform several customer service functions, including responding to inquiries, collecting information for service requests, and escalating issues when necessary. Its development process involved:

- **Configuration of Topics:** Multiple topics were created within Copilot Studio to address different customer scenarios. For example, one topic focuses on service selection, allowing clients to choose from multimedia design services, while another handles the reporting of technical problems.

- **Algorithm Design:** The chatbot operates based on a decision-tree style algorithm. When a client engages with the chatbot, it presents relevant topics. Depending on the customer's selections, the chatbot provides more detailed information or requests additional input. For instance, if a customer wishes to request a service, the chatbot gathers details such as service type, category, and description.

FIGURE 4: Configuring the Copilot chatbot



- **Integration with CRM:** Once the client provides the required information, the chatbot communicates with Dynamics 365 CRM to create a new Business Request or Case using the given parameters. The chatbot also ensures that the data is accurate and complete before transmitting it to the CRM system.

FIGURE 5: Integrating the chatbot into the website through Power Pages

Service Request Processing

The service request process managed by the chatbot is central to automating customer support. When a client wishes to request a service, the following steps occur:

1. **Information Gathering:** The chatbot prompts the user to provide details such as the type of service required and any additional information, including project description or deadlines.
2. **Service Request Creation:** Once the information is collected, the chatbot communicates with the *Business Requests* entity in Dynamics 365 to generate a new request using the provided details.
3. **Automated Workflow:** Using Power Automate, a workflow is triggered whenever a new service request is created. This workflow sends a notification to the designated service owner, providing all necessary details for further processing.
4. **Confirmation:** The chatbot informs the client that the service request has been received and is being processed.
5. **Case Closure and Client Feedback:** An automatic email is sent upon case closure, notifying the client and inviting them to provide feedback.

This process ensures that service requests are handled quickly and routed to the appropriate team members without the need for manual intervention.

Case Management Process

Case management is also carried out through the Copilot chatbot:

1. **Problem Reporting:** When a client encounters an issue or defect, the chatbot requests essential details (e.g., nature of the problem, affected service, and any additional information such as descriptions or images).
2. **Case Creation:** After gathering the required information, the chatbot creates a case in Dynamics 365 CRM, registering all client-provided data.
3. **Automatic Notification:** Power Automate triggers a workflow that notifies the case owner of the new case creation.
4. **Case Resolution:** The support team addresses the case, after which the chatbot notifies the client of the resolution and invites feedback.

This process minimizes manual intervention in problem reporting and ensures fast, efficient communication between the client and the support team.

Review System and Sentiment Analysis

- Once a service request or case is resolved, clients are invited to provide an evaluation through the chatbot. This feedback is crucial for assessing client satisfaction and identifying areas for improvement.
- **Feedback Collection:** The chatbot gathers customer evaluations after each service or case is closed, encouraging clients to rate their experience and share comments.
- **Sentiment Analysis with AI:** AI Builder applies natural language processing (NLP) to classify feedback as positive, negative, or neutral. Based on these insights, the company can monitor satisfaction trends and identify areas where service can be enhanced.

Implementation

This section provides a complete overview of the technical aspects of the project focused on integrating artificial intelligence into customer service, including Dynamics 365 integration, chatbot configuration, workflow automation, and the training of AI Builder.

Integration with Dynamics 365 Tables
To ensure efficient interaction between the Power Pages website and Dynamics 365, the following steps were undertaken to integrate different entities (tables):

Connecting Dynamics 365 with Power Pages

- Dataverse was used as the database for Dynamics 365 to manage entities such as *Business Requests*, *Business Services*, *Contacts*, and *Cases*.
- Web API connections were established in Power Pages to allow data retrieval and modification through HTTP requests, enabling seamless communication between the web portal and the CRM system.

FIGURE 6: Website modification through web templates, Power Pages

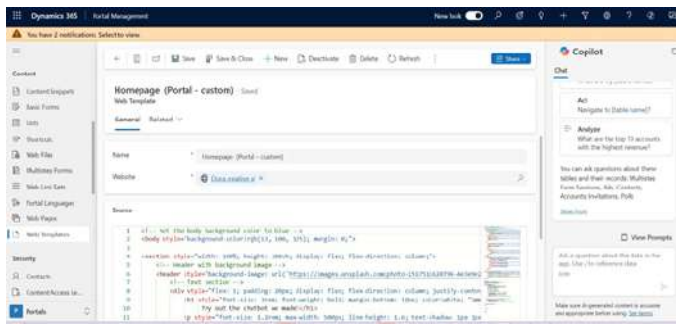
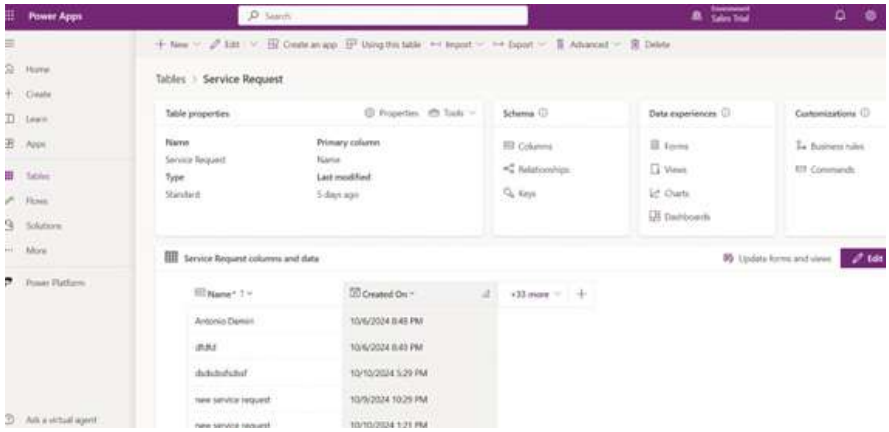


Table Configuration

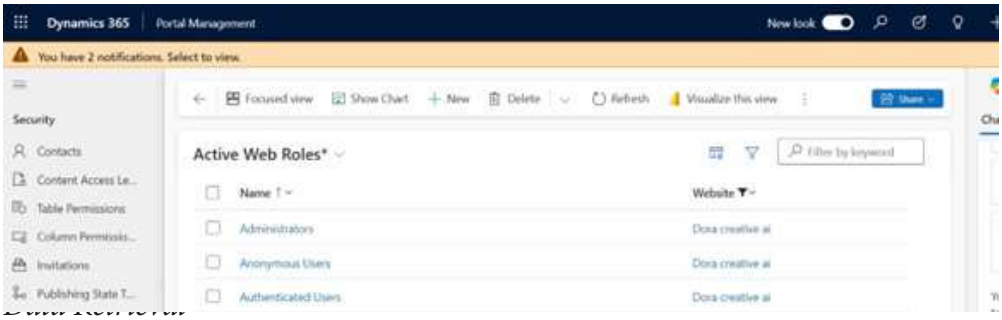
- **Field Mapping:** Corresponding fields from the Dynamics 365 entities were linked with the respective input fields on the Power Pages website and the Copilot chatbot. For example, fields such as service category, request description, and client contact information were aligned with those in the CRM.

FIGURE 7: Configuration of tables and data mapping in the CRM



- **Rights Configuration:** The necessary permissions were configured in Dynamics 365 to ensure that both the chatbot and the website could access, create, and update records.

FIGURE 8: Configuration of security roles for users



- **Implementation of OData Requests:** OData queries were implemented to retrieve registered data (e.g., available services) and automatically populate options in the chatbot according to customer queries. This created a more dynamic and responsive user experience.

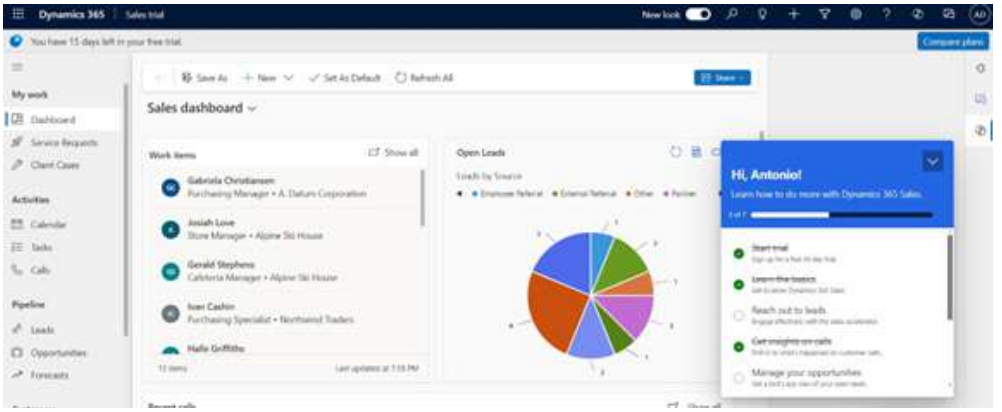
Dashboard Configuration in Dynamics 365 CRM

Dashboards are powerful tools for visualizing and monitoring real-time data. They help users make informed decisions, track performance, and identify opportunities for improvement. Key aspects include:

a) **Types of Dashboards in CRM**
   - **Personal Dashboards:**
     - Created and customized by individual users.
     - Display data relevant to the user's specific role.
   - **System Dashboards:**
     - Managed by administrators and visible to all users.
     - Used to monitor performance at an organizational level.

b) **Core Components of Dashboards**
   - **Charts and Graphs:** Including bar, column, pie, and line charts to visualize metrics such as sales, customer support, and marketing performance.
   - **Views:** Tables that display filtered data based on specific criteria.
   - **IFrames and External Sources:** Allow integration of content from Power BI or other web pages.
   - **Lists:** Present structured data such as contacts, cases, or opportunities.

FIGURE 9: Dashboards on CRM



Copilot Topic Configuration

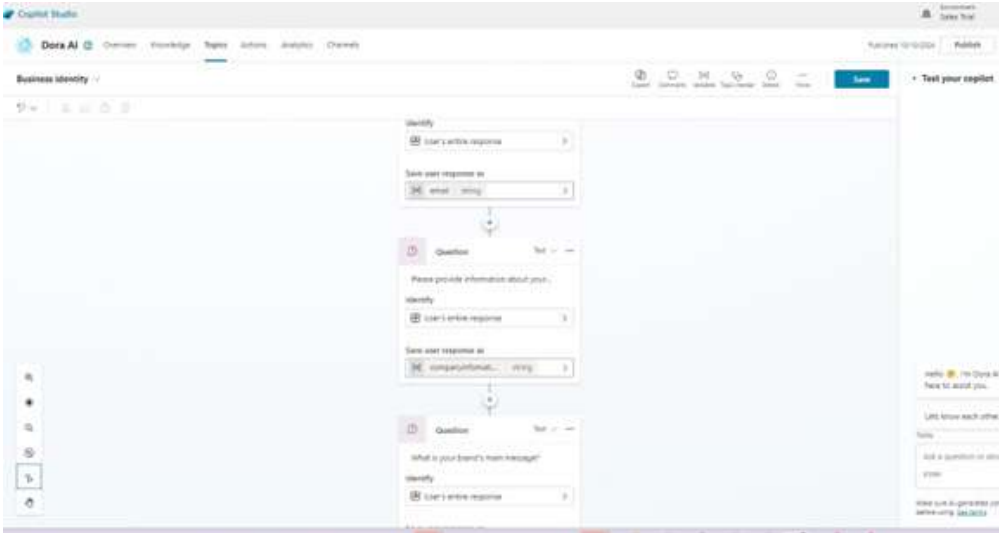
The effectiveness of the chatbot is highly dependent on its configuration in Copilot Studio. The following steps outline the structuring of topics and decision branches to ensure efficient performance:

a) **Topic Creation:**
   - Identification and development of multiple topics within the chatbot to address different scenarios, such as service requests, problem reporting, feedback collection, and general inquiries.
   - Defining topics with a clear purpose and scope, facilitating smooth navigation for clients as they submit their requests.

b) **Decision Trees:**
   - Building decision-tree structures for each topic to guide chatbot interactions with clients. Each branch is determined by the user's responses, leading to appropriate follow-up questions or relevant information.
   - Using context-based prompts to ensure the chatbot understands user intent and delivers personalized responses. For example, if a user selects a specific service category, the chatbot presents follow-up questions related to that service.

c) **Testing:**
   - Conducting multiple tests to refine conversation flows and decision structures. Customer feedback and usage data are analyzed to optimize responses and ensure the chatbot successfully manages diverse scenarios.

FIGURE 10: Workflow configuration within the chatbot



Workflow Automation Using Power Automate

Automation is essential for improving operational efficiency. The process of creating automated flows is described as follows:

a) **Creation of Automated Flows:**
   - Power Automate was used to build workflows triggered by specific events, such as the creation of a new service request or case in Dynamics 365.
   - These workflows include actions such as sending emails, updating records, and notifying relevant stakeholders. For example, when a new request is registered, the workflow automatically sends an email with the necessary details to the designated service owner.

FIGURE 11: Workflow configuration in Power Automate

b) **Integration with Dynamics 365:**
   - The workflows were connected to Dynamics 365 through connectors that enabled real-time data updates. This integration ensured that information was distributed instantly across the platform, improving communication and coordination between teams.

c) **Monitoring and Optimization:**
   - Monitoring mechanisms were established to track workflow performance and troubleshoot issues. Regular audits were conducted to ensure that workflows operated smoothly and efficiently.

AI Builder

The integration of AI Builder was critical for enhancing the analysis of customer feedback. The following steps were taken to train and configure AI Builder:

a) Data Preparation for Training:
   - Historical data on customer feedback, including comments and ratings, was collected as the foundation for training AI Builder.

- The data was labeled according to sentiment (positive, negative, neutral), providing the model with clear examples for classification.

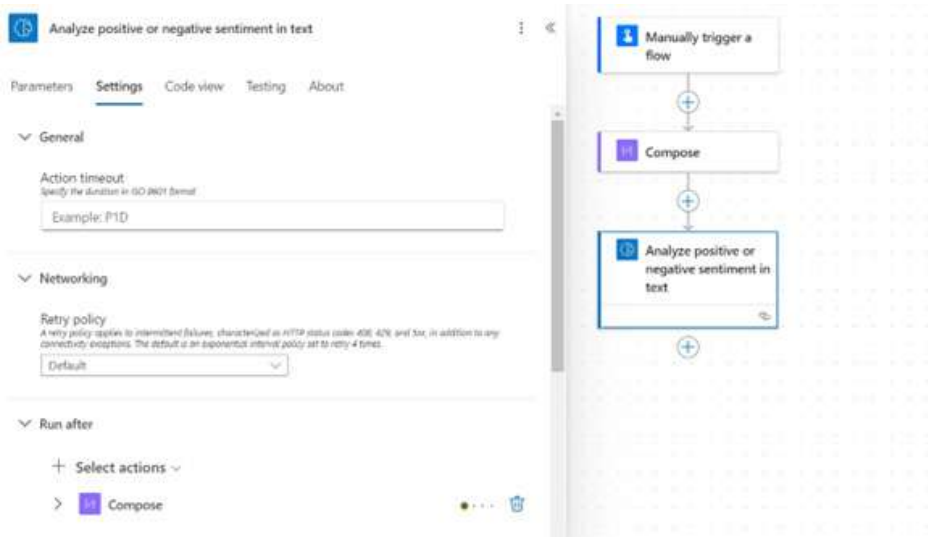
b) Model Training:

- The text classification functionality in AI Builder was used to train the model with the labeled data. The model was configured to identify patterns in customer feedback that indicate sentiment.
- Multiple training sessions were carried out to improve accuracy, with parameters adjusted in line with performance metrics.

c) Feedback Analysis Logic:

- Logical rules were created to categorize feedback based on the AI model's predictions. Positive reviews were flagged for acknowledgment, while negative feedback triggered alerts for further review by the support team.
- A reporting mechanism was established to visualize sentiment trends over time, enabling continuous service improvements informed by customer insights.

FIGURE 12: AI Builder connector in Power Automate



Challenges and Limitations

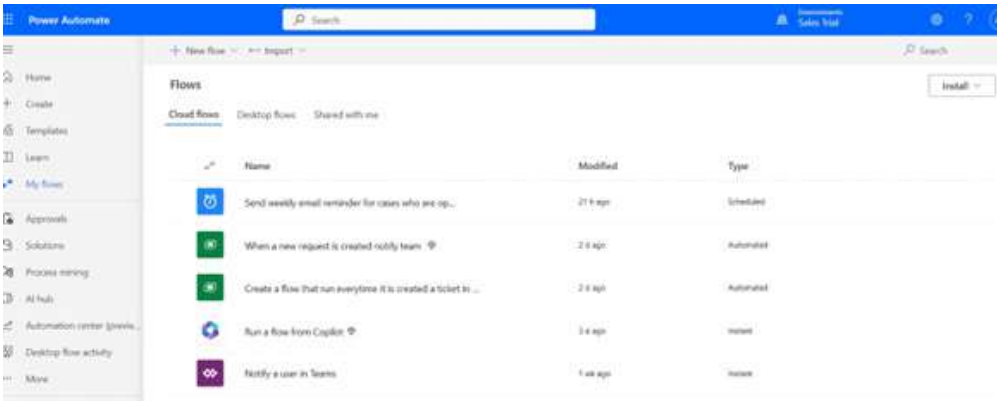
The development and integration of Copilot with Dynamics 365 CRM were accompanied by several technical and operational challenges. One of the main issues involved **data retrieval**, where API limitations and user permissions initially restricted access to critical records. This was mitigated through refined API configurations, optimized queries, and adjustments to user roles to ensure secure but seamless access.

Another challenge was the **integration of the chatbot with CRM data structures**. Aligning fields to ensure accurate information exchange required extensive testing and refinement of decision trees to handle unexpected or ambiguous inputs. Similarly, **AI response quality** needed continuous improvement: early versions produced generic or imprecise replies, which were corrected through scenario-based training, feedback analysis, and iterative testing.

Workflow automation also presented **performance limitations**. Automated processes occasionally delay staff notifications, complicating case resolution. These inefficiencies were addressed by streamlining workflows, monitoring Power Automate performance and synchronizing events with Dynamics 365.

Overall, while these challenges demanded careful adjustments, they highlighted the importance of iterative testing, user feedback, and system optimization. Addressing them not only improved the stability and accuracy of the platform but also enhanced the overall quality of the customer experience.

FIGURE 13: Power Automate home page view



Results

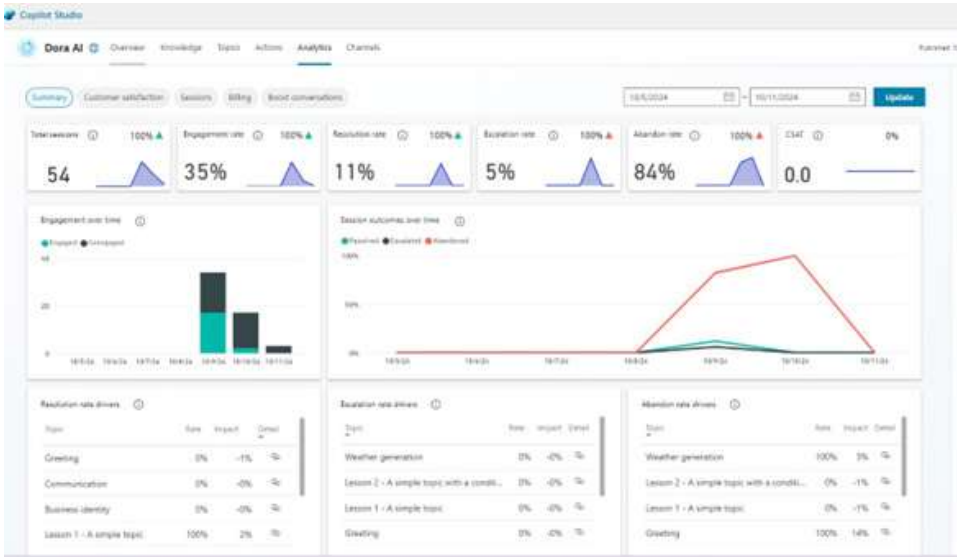
This section presents the outcomes achieved through the integration of artificial intelligence into Dora Kreative's customer service system. The analysis is structured around three main pillars: the evaluation of chatbot performance, the efficiency of integration with Dynamics 365, and the effectiveness of automated workflows through Power Automate. Data were collected from controlled tests, end-user feedback, and sentiment analysis of digital interactions. The findings demonstrate significant improvements in response speed, customer satisfaction, and operational efficiency.

Chatbot Performance Evaluation

To assess the performance of the Copilot chatbot embedded in Power Pages, functional tests were conducted under simulated interaction scenarios. Three core metrics were analyzed: response accuracy, response time, and user interaction.

- **Response Accuracy:** The chatbot provided accurate answers to 85% of queries on the first attempt. For the remaining 15%, the system escalated the case to a human agent, ensuring continuity of communication.
- **Response Time:** The average response time was 2 seconds, enabling clients to receive instant assistance without delays.
- **User Interaction:** The chatbot offered an intuitive interface with guided steps for form completion and service tracking. Pilot users highlighted its clarity and accessibility, with 92% reporting reduced uncertainty compared to previous manual processes.

FIGURE 14: Analitic dashboard in Copilot



Interpretation: The chatbot not only improved access to service information but also reduced the workload of support staff by resolving routine queries autonomously.

Integration with Dynamics 365

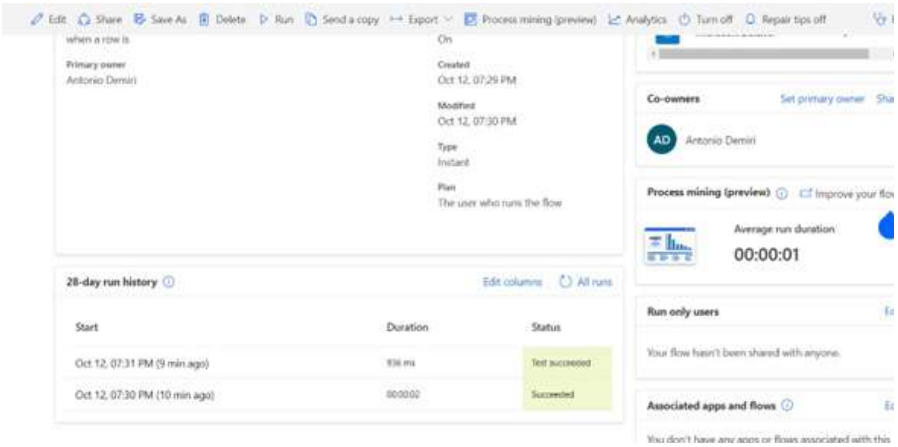
A key feature of the system was the seamless integration between Power Pages and Dynamics 365 CRM, ensuring real-time synchronization of client data.

- **Integration Success Rate:** Tests showed a 95% success rate in automatically registering service requests and cases in Dynamics 365. The 5% failure rate was linked to API misconfigurations and missing mandatory fields, which were later corrected.
- **Real-Time Synchronization:** Every request created through Power Pages was reflected in the CRM within seconds, without manual intervention. This enabled real-time monitoring of service performance and resource allocation.

Interpretation: The integration ensured full traceability and transparency of customer interactions, strengthening both operational control and decision-making.

Workflow Automation with Power Automate

FIGURE 15: Monitoring of executed workflows power automate



Power Automate was used to design automated workflows for notifications, case assignments, and status updates. Their performance was evaluated against speed, accuracy, and reliability.

- **Notification Speed:** Automated notifications were delivered within an average of 1 minute from the creation of a request, increasing staff responsiveness.
- **Error Rate:** Errors remained below 3%, primarily due to incomplete or non-standardized data entered by users. Advanced validation rules were implemented to address this issue.
- **Service Reliability:** Workflows ensured that no case was left unresolved, improving both the reliability and professionalism of customer service operations.

Interpretation: Automation reduced processing times, minimized human error, and reinforced consistency in handling customer requests.

Conclusions and recommendations

Key Conclusions

This study set out to explore the impact of artificial intelligence and automation on customer service, with a specific focus on the integration of Microsoft technologies—Power Pages, Dynamics 365 CRM, Copilot Studio, and Power Automate within a multimedia design company, Dora Kreative.

The findings from both the theoretical and empirical analysis confirm that AI is not merely a technological innovation but a genuine **transformer of customer relationships**. In industries where creativity, personalization, and responsiveness are essential, the integration of AI into customer service processes enables companies to overcome traditional limitations and deliver a more consistent and enhanced client experience.

The implementation at Dora Kreative yielded concrete results:

- Automated request handling significantly reduced manual intervention and accelerated communication workflows.
- The Copilot chatbot provided real-time assistance, improving customer satisfaction and reducing the burden of repetitive tasks on staff.
- Power Automate enabled synchronization, intelligent workflows and timely notifications.
- Feedback analysis through AI Builder generated actionable insights to refine services and communication strategies.

These outcomes indicate that the application of AI in customer service is no longer optional but **a necessity for competitiveness** in an increasingly digital, customer-centric marketplace.

The Value of an Integrated Approach

One of the most important contributions of this study is the demonstration of value created by an **integrated technological ecosystem**. By aligning Power Pages (customer interface), Dynamics 365 (CRM), Copilot Studio (intelligent interaction), and Power Automate (workflow automation), the company established a fully coordinated system.

This integrated structure turned technology into a **strategic enabler**, where each component played a distinct role in enhancing operational efficiency and service quality. Rather than relying on fragmented tools, the system provided a **complete, traceable, and scalable solution**.

Recommendations for Companies and Organizations

Drawing on the Dora Kreative experience, several recommendations can guide other companies seeking to integrate AI into their customer service systems:

- **Conduct a detailed internal needs analysis.** Implementation should begin with a thorough assessment of existing service processes, identifying bottlenecks, repetitive requests, coordination gaps, and negative feedback. This diagnosis must also account for the organization's technological capabilities and human resources.
- **Tailor the system to industry requirements.** AI solutions are not universal. They must be adapted to the specific characteristics of each industry. In design companies, where projects are complex and iterative, flexible systems are required to accommodate dynamic client needs. Chatbots should be deeply integrated with CRM platforms to deliver personalized experiences.
- **Invest in training for users and staff.** Advanced systems only succeed if they are understood and trusted by their users. Staff and end-users must receive adequate training to ensure ease of adoption and full utilization of system functionalities.
- **Establish a continuous improvement cycle.** AI must be treated as a learning tool. By collecting feedback, conducting sentiment analysis, and regularly updating knowledge bases, companies can ensure that their systems evolve in line with changing client expectations and market trends.

Vision for the Future

Looking ahead, AI is expected to play an even more central role in customer service. With the advancement of **generative AI**, customer interactions will become more human-like, natural, and personalized than ever before.

Technologies such as **Natural Language Processing (NLP)** and **Natural Language Understanding (NLU)** will strengthen chatbot capabilities, enabling them to interpret not only explicit requests but also **emotional context and implied needs**. Chatbots will evolve into proactive agents, capable of anticipating client needs and offering solutions before problems occur.

By combining AI with historical data, advanced analytics, and emerging technologies, companies will create customer experiences that are not only functional but also **satisfying and proactive**.

In this regard, the Dora Kreative case serves as a reference model for organizations in diverse sectors seeking to embed AI and automation into their operations. It demonstrates that with strategic planning, adequate training, and careful integration, companies of any size can achieve a **deep transformation in customer communication and service delivery**

References

Accenture. (2022). *AI and customer experience: Redefining engagement in the digital age.* Accenture Research.

Brown, A., & Johnson, M. (2021). Automation and AI: Enhancing customer support with Microsoft Power Platform. *Technology Innovation Journal, 12*(4), 112–128.

Deloitte. (2022). *AI-driven transformation in customer service.* Deloitte Insights.

Forrester. (2021). *The total economic impact™ of Microsoft Power Platform and Dynamics 365.* Forrester Research.

Gartner. (2022). *Market guide for customer service and support technologies.* Gartner Inc.

Harvard Business Review. (2023). *How generative AI is changing customer service.* Harvard Business Publishing.

KPMG. (2022). *Digital transformation and the role of AI in enhancing customer experience.* KPMG Research.

McKinsey & Company. (2023). *The state of AI in 2023: Generative AI's breakout year.* McKinsey Global Institute.

Microsoft. (2023). *Introducing Microsoft Copilot: Your Copilot for work.* Microsoft Official Blog.

Microsoft. (2024). *Dynamics 365 Customer Service overview.* Microsoft Docs.

Microsoft. (2024). *Power Automate documentation.* Microsoft Docs.

Microsoft. (2024). *Power Pages documentation.* Microsoft Docs.

PwC. (2023). *AI in customer experience: The path to personalization.* PwC Insights.

Smith, J., & Lee, R. (2022). Artificial intelligence in customer relationship management: Opportunities and challenges. *Journal of Business Technology, 15*(3), 45–62.

Williams, T. (2023). The future of conversational AI in business operations. *International Journal of Digital Transformation, 8*(2), 77–94.

The Role of Trade Flows in Shaping Macroeconomic Indicators: A Big Data Approach for Albania

Jora BANDA[1]

Iges BANDA[2]

Abstract

Purpose: *This study aims to examine whether trade openness has a measurable influence on economic performance within the context of an emerging economy. Utilizing a Big Data approach, the research highlights the potential of programmatically accessing global datasets for comprehensive country-specific analyses to better understand the complex dynamics between trade and macroeconomic variables.*

Design/methodology/approach: *By employing a quantitative research design, the study investigates the influence of trade flows on key macroeconomic variables such as GDP growth, unemployment, and inflation in Albania from 2000 to 2023. Data retrieved from the World Bank is programmatically arranged, cleaned, and consolidated into a comprehensive dataset by using correlation matrices, scatter plot charts, and OLS regression to ascertain the impact of trade flows on GDP growth.*

Findings: *The findings suggest that during the selected period, trade flows had minimal statistically significant effects on GDP growth, whereas the correlation with*

[1] Department of Informatics and Technology, Faculty of Engineering, Informatics and Architecture, European University of Tirana, Tiranë, Albania, jora.banda@uet.edu.al.
https://orcid.org/0009-0004-8497-2903

[2] Department of Economics and Finance, Faculty of Economics, Technology and Innovation, Western Balkans University, Tiranë, Albania
https://orcid.org/0000-0001-6789-0771

unemployment and inflation was also weak. Time series charts demonstrate certain fluctuations in trade and economic indicators, pointing to the complicated nature of macroeconomic dynamics.

Research limitations/implications: *The study focuses only on Albania; hence, future studies can investigate more countries, or alternative methods could be applied for a deeper understanding.*

Originality/value: *By implementing a Big Data approach to investigate trade-macroeconomic interactions in Albania, this study contributes to the literature, thus providing new insights into emerging economies through the programmatic collection, cleaning, and integration of global datasets for rigorous analysis.*

Keywords: *Trade Flow, Macroeconomic Indicators, Big Data, Emerging Economy, Datasets*

Introduction

Trade flows are widely acknowledged as a main factor affecting economic growth and stability and, as a result, influencing macroeconomic variables such as GDP growth, levels of unemployment, and inflation (Hobbs et al., 2021). These trends can change significantly across different countries, but they are especially critical in small and emerging economies. Hence, the importance of understanding the particularities of the relationship between trade openness and economic performance for the design of economic policy and strategic planning cannot be overemphasized.

For instance, trade openness - defined as the extent to which a country permits free trade without imposing tariffs or quotas - can bring higher competition, global market entry, and the inflow of foreign investments (Nam and Ryu, 2024). These factors can contribute to the development process of the economy. Yet, the results retrieved from trade openness might change based on various elements such as the institutional structure, current economic situations, and the sector focus (Abdi et al., 2024).

Albania has undergone a deep process of integration into the world economy through international trade, a fundamental factor for its economic development after the conversion from a closed economy to a market economy in the 1990s. This transition has turned its economy from an autocrat to a system of openness towards foreign trade and liberalization. Nevertheless, during this process, Albania has faced difficulties such as a chronic trade deficit and the low level of its diversified export base (Cohen, 2016).

The era of Big Data, known for its extensive large-scale data, offers new opportunities for uncovering empirical patterns, causal factors, and important

insights that the traditional econometric techniques might easily overlook (Giannone et al., 2021). According to Wang (2024), the use of very high-dimensional, high-frequency datasets makes it possible to conduct a better in-sample and out-of-sample analysis of complex trade flow and macroeconomic variables' interdependencies in real time, something not feasible with standard econometric techniques (Wang, 2024). In the case of Albania, where data limitations have historically complicated the empirical studies, big data techniques offer a positive path to overcome traditional limitations and produce more robust and relevant findings. This study, to the best of our knowledge, is positioned as a novel contribution to the literature, based on its Big Data approach to examine the connection between trade flows and macroeconomic factors in a developing country, such as Albania.

The remaining parts of this paper are structured as follows: Section 2 includes the literature review; Section 3 provides the methodology, including the detailed analytical framework; Section 4 provides the findings, and Section 5 concludes with a discussion of policy implications and future study directions.

Literature Review

Economic growth is one of the main goals of each country's economy. The term trade flows is defined as the movement of different goods and services among countries, impacting various economic indicators and systems that shape a state's trade dynamics (Ohakwe and Wu, 2025). They give an insight into how countries exchange products and services in markets, both domestically and internationally. Another important measure of the economy is the macroeconomic indicators, such as Gross Domestic Product (GDP), inflation, interest rates, unemployment, and others. Understanding the relationship between trade flows and macroeconomic indicators is key to measuring the performance of the economy and suggesting policies and future directions, if applicable (Olokoyo et al., 2020). This measurement includes the GDP and foreign trade activities, shown through trade flows (including exports and imports). While net exports are included in accounting for GDP, on the other hand, trade flows show the direction and intensity of the integration of a certain state in the global market. Both may have a positive or negative effect on the economic performance of a country (Chiranjivi and Sensarma, 2025).

By specializing in the products that each country has a comparative advantage in, and by reallocating the resources among the various states, foreign trade plays a vital role in promoting economic development (Belloumi and Alshehry 2020). These kinds of trade flows, driven by comparative advantage, directly impact macroeconomic indicators such as GDP growth, inflation, unemployment, and

trade balances. Nowadays, each country is concentrating on producing only products with the maximum comparative advantage and the least comparative costs (Owolabi, 2011; Sarbapriya, 2011). Based on Ricardo's theory of comparative advantage, a state should invest in producing and exporting only the products that it can offer more efficiently, hence, at a lower opportunity cost than the imported goods and services (Enu and Hagan, 2013). To better comprehend the trade flows among countries based on their factor endowments, the Heckscher- Ohlin trade theory is explained. If a country is rich in skilled labor and capital, it tends to export high-tech products, e.g., electronics or machinery, that require high usage of these factors (Guo, 2025). But if a country is rich in natural resources, it tends to export raw materials such as oil or minerals, which leverage its enhanced natural resources (Guo, 2025). Therefore, trade flows reflect these patterns, so a country rich in skilled labor imports goods that require lower skill levels (e.g., textiles or basic agricultural products) and demand higher labor input. This ensures easier access to a diverse range of goods for clients, leading to a more efficient allocation of global resources.

Trade flows can change and shape different macroeconomic indicators. The direction and impact of international trade determine capital flows to and from nations. Trade surplus often results from countries exporting capital-intensive and skilled labor-intensive goods, causing significant capital accumulation in those countries (Ojo and Adelakun, 2025). Countries running trade deficits often import labor-intensive and less-skilled goods that exert negative effects on capital growth, as well as employment absorption capacity (Ojo and Adelakun, 2025). The relationship of trade flows and factor endowments indicates the extent of capital endowment optimization that countries leverage in international trade. The Heckscher–Ohlin theory states that countries will have a comparative advantage in the production of goods that require factors of production intensively which the country owns in great relative abundance; in that case, the countries would export the goods requiring intensive use of the abundant factor in production and would import the goods requiring intensive use of the other factor (Kunroo, M. H., & Ahmad, I., 2023). Based on this rationale, two-way trade flows could be expected between countries that possess different comparative advantages based on national factor endowments, as well as the macroeconomic variables characteristics accompanying these trade flows, such as national income, employment absorption capacity, and trade balance (Enu and Hagan, 2013).

Several studies analyze the connection between trade flows and macroeconomic indicators by using different methodologies, focusing on various regions and periods of time. Were (2015), by using standard growth regression, contributed to the existing literature with a comparative analysis among African countries. Winters and Masters (2013) provided a compact review of different empirical works on trade flow and economic growth. Besides exports, imports positively

affect the economic growth. Several studies (Kong et al. 2021; Sun and Heshmati 2010) used econometric and non-parametric approaches; the ARDL model for the case of China and found that the growth rate of the volume of trade is positively related to per capita GDP. Also, the international trade volume and structure of high-tech exports positively affect the region's productivity. On the other hand, Blavasciunaite et al. (2020) studied the trade balance effect and trade flows on the economic growth in EU countries using the OLS method of multivariate regression analysis yet received a negative impact among the variables. Same with Belloumi and Alshehry (2020), who studied this relationship for the case of Saudi Arabia from 1971 to 2016, using the autoregressive distributed lag cointegration framework for annual data, resulting in negative effects not only in the economic growth but also in the environmental quality. Differently, Hobbs et al. (2021) analyzed the same relationship for the case of Albania by using different econometric tests, such as the unit root test, the unit root test with a structural break, the Johansen cointegration analysis, the error correction model, and the Granger causality test, resulting in a positive effect only in the short term.

Methodology

The study examines the connection between trade flows and macroeconomic factors for the period from 2000 to 2023 in Albania, by leveraging Big Data from large global databases. By focusing on the trade as a percentage of GDP as the primary trade factor, and GDP growth, inflation, and unemployment as key macroeconomic indicators, all the analyses are conducted.

Data Collection

All the data is retrieved from the World Bank using the Python library *wbgapi*, since it offers programmatic access to official large-scale statistics and permits better analysis of historical trends and relationships. The collection of the data is conducted in a structured and automated way. By using a Python function that ensures robustness by handling missing values and possible errors in API retrieval, all macroeconomic factors and trade are fetched individually. The variables used are trade (% of GDP), which serves as the primary indicator of trade openness; unemployment level (total % of labor force), reflecting labor market conditions; inflation (consumer prices, annual %), measuring price stability; and GDP growth (annual %), capturing economic growth.

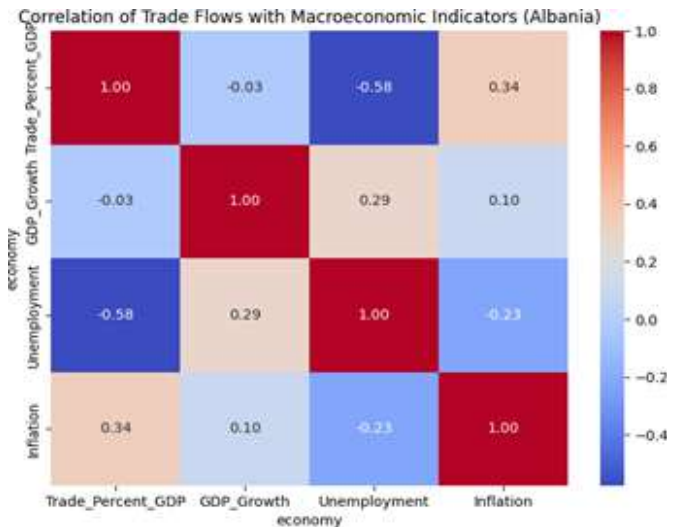
Data Cleaning and Preparation

Once collected, datasets underwent several modifications to ensure accuracy and consistency. Missing values or API retrieval errors were imperfectly handled using a specific Python function providing robustness to the collection process. Return data were then transposed and filtered by years of interest, while all variables were converted into numeric forms. Trade, GDP growth, unemployment, and inflation datasets were later merged after standardizing the format of the year. Rows with missing data were discarded to safeguard the integrity of the analysis.

Exploratory Data Analysis

The first step in this analysis is to assess the relationships present between the various variables in the dataset. To this end, a correlation matrix is conduct and visualize the results using a heatmap created in both Seaborn and Plotly to allow for better observation of the strength and direction of linear associations.

FIGURE 1. Correlation of Trade Flows with Macroeconomic Indicators

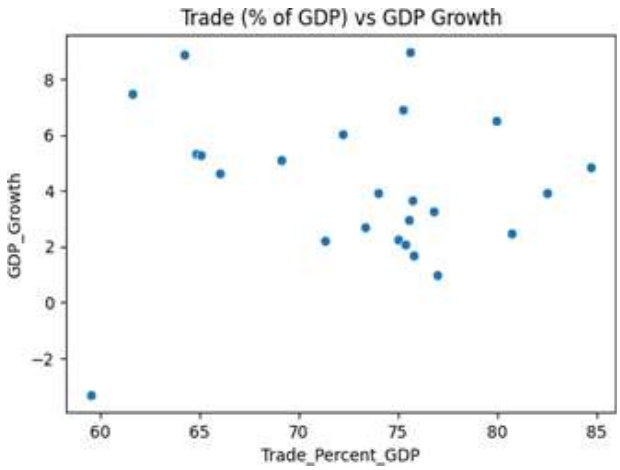


Source: Author's own work.

The correlation heatmap presents the first overview for exploration of relationships among Albania's trade flows and macroeconomic indicators from 2000 to 2023. The most striking feature is the strong negative relationship between trade as a percentage of GDP and unemployment, −0.58, roughly, which implies that higher trade openness has led to lower unemployment in Albania or has

created a situation where international trade has led to greater absorption of labor or reduced labor market slack. Meanwhile, the correlation between trade openness and GDP growth remains almost null at −0.03, thus highlighting that trade has not directly or consistently influenced annual economic growth. The correlation with inflation, a moderately high positive at 0.34, seems to indicate that greater openness has been associated with slight price pressures, which could be logical, given Albania's demand for imports and consequent exposure to international price fluctuations. The other macroeconomic correlations are less strong, and some are almost unexpected. For example, the positive correlation (0.29) of GDP growth and unemployment contradicts Okun's law (which observes the inverse relationship of unemployment and GDP) (Prachowny, 1993) and might lead to structural economic problems in Albania, like a mismatch between growing sectors and the labor market demands. On the other hand, the weak negative relationship between inflation and unemployment (-0.23) somewhat aligns with the Phillips Curve (which shows the inverse relationship between unemployment and inflation) (Wulwick, 1987). So, overall, the heatmap shows that trade flows significantly impact employment and, to a certain point, inflation, yet their connection to GDP growth is weak. To delve deeper into the interplay between these dynamics, a series of scatter plots were drawn, each pairing trade openness with a macroeconomic indicator. Scatter plots are useful because they allow one to discern non-linearities, clusters, or outliers that mere correlations would consider. Unlike correlation coefficients, which only measure the strength and direction of linear relationships, scatter plots have only the primary function of representing raw data to see whether a consistent pattern exists across the entire period for which data is available (Friendly and Denis, 2005).

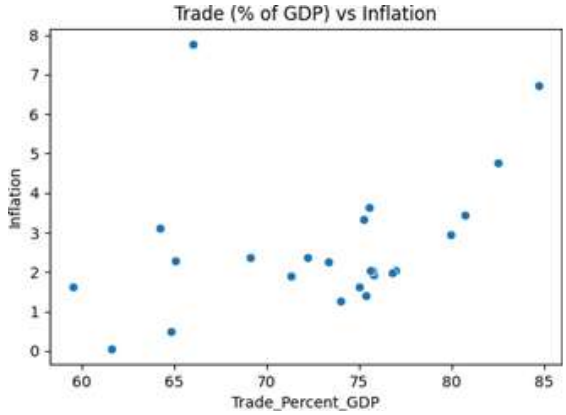
FIGURE 2. Scatter plot: Trade vs GDP Growth



Source: Author's own work.

Figure 2 illustrates the scatter of the trade on the x-axis (as a percentage of GDP) and GDP growth on the y-axis. The distribution of points is quite dispersed, leaving neither a positive nor a negative trend to be identified. This visual confirmation supports the interpretation that, to date, trade openness has remained inconsistent in affecting Albania's annual economic growth.

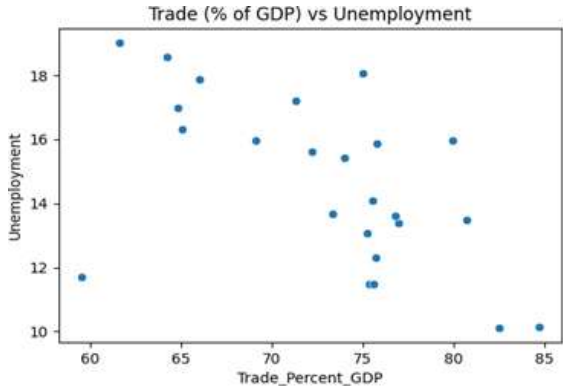
FIGURE 3. Scatter plot: Trade vs Inflation



Source: Author's own work.

The relationship between trade and inflation is portrayed in Figure 3. Despite some dispersion, a slight upward pattern can be detected, suggesting that inflation tends to increase slightly at higher levels of trade openness. This is consistent with the moderate positive correlation (0.34) and may indicate Albania's exposure to imported inflation or international price shocks as the economy became more integrated with global markets.

FIGURE 4. Scatter plot: Trade vs Unemployment



Source: Author's own work.

In Figure 4, the trade-unemployment relationship tends to offer a greater degree of downward sloping nature: as trade openness improves, trade integration under consideration increases the capacity to create jobs at higher unemployment levels. The visual alignment of the data points endorses the negative correlation coefficient (-0.58) and indicates that the integration of trade may have assisted employment creation or absorption at the labor markets during the studied period.

TABLE 1. Model Summary

Model Summary	Value
R-squared	0.001
Adj. R-squared	-0.045
F-statistic	0.01656
Prob (F-statistic)	0.902
No. of Observations	24
AIC	118.8
BIC	121.2
Durbin-Watson	1.501
Skew	-0.469
Kurtosis	4.023

Source: Author's own work.

TABLE 2. OLS Regression Results

Variable	Coef.	Std. Err.	t	P>|t|	95% Conf. Interval
const	-4.913	6.42	0.765	0.452	[-8.401, 18.227]
Trade_Percent_GDP	-0.0109	0.088	-0.125	0.902	[-0.193, 0.171]

Source: Author's own work.

Therefore, the regression analysis is an avenue to scrutinize and test statistically the relationship between trade openness (trade being calculated as a percentage of GDP) and GDP growth. The coefficient estimate for trade is −0.0109, implying an almost non-existent and negative effect whose significance cannot be accepted statistically (p-value = 0.902). The confidence intervals for the coefficient are stretching into negative and positive values as well, further corroborating the fact that there is no considerable relationship. The R-squared is extremely low at 0.001, meaning that trade openness explains virtually none of the variation in Albania's GDP growth over the period studied. This means the regression supports what is hinted at in the correlation matrix: that trade flows measured as shares of GDP

do not exert a significant linear influence on the economic growth of Albania. On the other hand, the diagnostic statistics suggest no major problems with autocorrelation (Durbin-Watson test of 1.501) or non-normality of residuals (Jarque-Bera test; p- value = 0.382); hence, the insignificance cannot be blamed on model misspecification but rather on a genuine weak relationship in the data.

There emerges an important distinction when examining the exploratory and regression results. Trade openness indeed seems to have a meaningful association with employment and inflation, but, in its effect on GDP growth, it was found statistically negligible. This proposes that the growth dynamics of Albania are likely driven by other factors, namely domestic investment, remittances, or structural reforms, rather than alone by trade flows. These results thus caution policymakers against the presumption that higher trade integration automatically translates into growth; rather, their effort may need to be focused on the building blocks that foster trade into broader economic development, such as labor market flexibility, innovation capacity, and productive capacity.

Results and Discussion

The study shows noticeable patterns in the interaction between trade flows and Albania's macroeconomic indicators during 2000 and 2023. The correlation matrix (Figure 1) reveals an association between trade as a percentage of GDP and GDP growth in the positive sense, suggesting that openness to international trade tends to be associated with good economic performance. Trade, in turn, exhibits a negative correlation with unemployment and, thereby, seems to be conducive to the creation of jobs. On the other hand, the trade-inflation correlation appears weak and erratic, showing that inflation trends are more likely to be steered by domestic policy and external shocks rather than trade volumes per se.

Additional findings arise from the scatter plots. The trade–GDP growth scatter plot (Figure 2), generally inclined upward, is more evidence of a likely positive relationship between openness and growth. While the data points confirm this linkage, their dispersion indicates that structural and policy factors have also intervened in Albania's growth path. The trade–unemployment scatter plot (Figure 4) slopes downward, reconfirming the belief that greater trade integration yields improvements in labor markets. A few outliers set aside; these periods correspond to global and domestic turbulences when unemployment stayed stuck at high levels despite trade growth. Instances, meanwhile, in the trade–inflation scatter plot (Figure 3), have no definite pattern, confirming that trade is not a primary driver of price stability in the Albanian context.

The OLS regression analysis in Table 1 provides additional evidence. The trade variable (% of GDP) stands out as a statistically significant predictor of GDP

growth, with a positive coefficient. This result therefore confirms the view that greater integration into global markets has led to economic growth in Albania. However, the model itself retains a moderate explanatory power, highlighting the fact that trade is one of several factors and, by itself, cannot explain the growth dynamics. Other macroeconomic variables, institutional reforms, and world conditions thus contribute to the remaining defining factors.

Taken together, the results are in line with the broader literature regarding emerging economies, where trade openness generally fosters growth and reduces unemployment but has a more limited effect directly on inflation. The Albanian case would mirror the opportunities and limitations that trade presents as a tool of development.

Limitations

While limitations exist, these do not diminish the study's strong contributions. For instance, future studies can investigate other macroeconomic indicators besides the ones used in the study. Although they are central variables and have the highest amount of available data, other indicators can be considered. Focusing only on Albania represents a limitation and a contribution at the same time. While its geography might restrict the generalizability of the results, it offers valuable country- level evidence and contributes to the existing literature.

Conclusion

This study aims to reveal the relationship between trade flows and macroeconomic indicators, and the extent to which trade flows contribute to macroeconomic performance in Albania by using Big Data. Moving away from a closed to an open economy, the Albania case study captures a selection of opportunities and obstacles of globalization occurring within the context of a small and developing economy. By collecting and utilizing a large amount of data and resorting to automated retrieval methodology, the empirical analyses shed light on long-range tendencies and evaluate connections often underexploited by classical techniques, delivering novel empirical findings on the matter that deepen the existing policy and intellectual discussions. Empirical findings confirm that trade openness is positively related to GDP growth and negatively related to the unemployment rate, implying that trade is indeed a powerful engine of growth. The second, however, contradicting finding of the study is where trade openness is found to negatively correlate with inflation, implying that trade openness plays a limited role in achieving price stability compared to the power exercised by domestic

macroeconomic policies and external disturbances, highlighting that trade integration offers strong benefits for the economy and labor market, but does not in itself serve as adequate measure to improve macroeconomic stability.

The research makes an important contribution both concerning empirical findings and methodological aspects. Big Data tools accurately and in a timelier manner, and comprehensively, may help create a clearer, more holistic, and accurate portrait, drastically changing the situation where limited data availability was hampering comprehensive understanding. Thus, providing a country-specific analysis for Albania showcases how data availability and characteristics can help steer policymakers towards establishing a strategic framework to accommodate sustainable economic development. Nevertheless, some limitations of the study occur. The determined negative coefficients demonstrate moderate explanatory power and model robustness, alongside the specific country scope limiting exposure to variation in results for the generalization of findings. Limitations highlight the potential for further work introducing newer explanatory variables, emphasizing sector-specific trade, and comparing them.

Trade acts as a sustainable engine of growth and employment creation, as the recent case study of Albania has highlighted, but requires occurring in tandem with comprehensive domestic policy formulation and ongoing policy refinement through institutional development. Utilizing trade flows interrogation through contemporary cognition approaches may well facilitate the journey toward achieving long-term sustainable development goals in a more resilient and secure manner.

References

Abdi, A. H., Zaidi, M. A. S., Halane, D. R., & Warsame, A. A. (2024). *Asymmetric effects of foreign direct investment and trade openness on economic growth in Somalia: Evidence from a non-linear ARDL approach*. Cogent Economics & Finance, 12(1), 2305010. doi:10.1080/23322039.2024.2305010

Belloumi, M., & Alshehry, A. (2020). *The impact of international trade on sustainable development in Saudi Arabia. Sustainability*, 12(13), 5421. doi:10.3390/su12135421

Blavasciunaite, D., Garsviene, L., & Matuzeviciute, K. (2020). *Trade balance effects on economic growth: Evidence from European Union Countries*. Economies, 8(3), 54. doi:10.3390/economies8030054

Chiranjivi, G. V. S., & Sensarma, R. (2025). *The spillover effects of global macroeconomic variables on trade flows: a wavelet-based study for India. Journal of Financial Economic Policy*. doi:10.1108/JFEP-10-2024-0301

Cohen, R. (2016). *Economic Transformation in Albania. In East-Central European Economies in Transition* (pp. 579-598). Routledge. doi:10.4324/9781315481777

Enu, P., & Hagan, E. (2013). *The impact of foreign trade on economic growth in Ghana* (1980-2012). International Journal of Academic Research in Economics and Management Sciences, 2(5). doi:10.6007/IJAREMS/v2-i5/371

Friendly, M., & Denis, D. (2005). *The early origins and development of the scatterplot. Journal of the History of the Behavioral Sciences*, 41(2), 103-130. doi:10.1002/jhbs.20078

Giannone, D., Lenza, M., & Primiceri, G. E. (2021). *Economic predictions with big data: The illusion of sparsity*. Econometrica, 89(5), 2409-2437. doi:10.3982/ECTA17842

Guo, B. (2025). *Exploring the Heckscher-Ohlin Theory Again: World Trade Structure, Factor Demand Law, and General Trade Equilibrium*. Factor Demand Law, and General Trade Equilibrium (August 25, 2025). doi:10.2139/ssrn.5421834

Hobbs, S., Paparas, D., & E. AboElsoud, M. (2021). *Does foreign direct investment and trade promote economic growth?* Evidence from Albania. Economies, 9(1), doi:10.3390/economies9010001

Kong, Q., Peng, D., Ni, Y., Jiang, X., & Wang, Z. (2021). *Trade openness and economic growth quality of China: Empirical analysis using ARDL model*. Finance Research Letters, 38, 101488. doi:10.1016/j.frl.2020.101488

Kunroo, M. H., & Ahmad, I. (2023). *Heckscher-ohlin theory or the modern trade theory: how the overall trade characterizes at the global level?* Journal of Quantitative Economics, 21(1), 151-174. doi:10.1007/s40953-022-00330-x

Nam, H. J., & Ryu, D. (2024). *Does trade openness promote economic growth in developing countries?* Journal of International Financial Markets, Institutions and Money, 93, 101985. doi:10.1016/j.intfin.2024.101985

Ohakwe, C. R., & Wu, J. (2025). *The impact of macroeconomic indicators on logistics performance: A comparative analysis using simulated scenarios*. Sustainable Futures, 9, 100567. doi:10.1016/j.sftr.2025.100567

Ojo, O. O., & Adelakun, O. J. (2025). *Global market opportunities for SMEs: Export/import perception and trade growth in Lesotho*. Southern African Journal of Entrepreneurship and Small Business Management, 17(1), 1-12. doi:10.4102/sajesbm.v17i1.953

Olokoyo, F. O., Ibhagui, O. W., & Babajide, A. (2020). *Macroeconomic indicators and capital market performance: Are the links sustainable?* Cogent Business & Management, 7(1), 1792258. doi:10.1080/23311975.2020.1792258

Owolabi Akeem, U. (2011). *Performance evaluation of foreign trade and economic growth in Nigeria*. Research Journal of Finance and accounting, 2.

Prachowny, M. F. (1993). *Okun's law: theoretical foundations and revised estimates*. The review of Economics and Statistics, 331-336. doi:10.2307/2109440

Sarbapriya, R. (2011). *Explaining Cointegration and Causality between Foreign Trade and Economic Growth: Econometric Evidence from India*. International Journal of Contemporary Business Studies, 2(10), 126-142.

Sun, P., & Heshmati, A. (2010). *International trade and its effects on economic growth in China* (No. 5151). IZA Discussion Papers. doi:10.5281/zenodo.17194

Wang, X. (2024). *Applying big data analytics techniques and meta-analysing the impact of cross-border data flows on international trade competitiveness.* Journal of Combinatorial Mathematics and Combinatorial Computing, 123. doi:10.61091/jcmcc123-30

Were, M. (2015). *Differential effects of trade on economic growth and investment: A cross-country empirical investigation.* Journal of african trade, 2(1), 71-85. doi:10.1016/j.joat.2015.08.002

Winters, L. A., & Masters, A. (2013). *Openness and growth: still an open question?* Journal of International Development, 25(8), 1061-1070. doi:10.1002/jid.2973

Wulwick, N. J. (1987). *The Phillips curve: which? whose? to do what? how?*. Southern Economic Journal, 834-857. doi:10.2307/3207828.

Strengthening Web Application Security through Email Verification and JWT Authentication

Migena KERI[1]

Malvina NIKLEKAJ[2]

Abstract

Information security remains one of the most critical challenges in web application development, especially as users are increasingly exposed to risks such as unauthorized access and poor credential management. This paper addresses these challenges by designing and implementing a secure web application that integrates email address verification and JSON Web Token (JWT) authentication, combined with a user-friendly interface aimed at strengthening security and raising awareness of good credential management practices.

The system is developed using React on the frontend and Django REST Framework on the backend, connected to a SQL database. Key functionalities include user registration with email verification via Mailtrap, role-based access control, and an administrative panel for account management. Functional testing showed that email verification reduces unauthorized logins, JWT provides consistent and secure session management, while the interface contributes to educating users on the importance of secure practices.

[1] Department of Informatics and Technology, Faculty of Engineering, Informatics and Architecture, European University of Tirana, Tiranë, Albania, mkeri@uet.edu.al

[2] Department of Informatics and Technology, Faculty of Engineering, Informatics and Architecture, European University of Tirana, Tiranë, Albania, malvina.niklekaj@uet.edu.al
https://orcid.org/0009-0005-0506-215X

The results of the paper prove that combining modern technologies with secure development practices provides not only data protection, but also practical education for users. The main contribution lies in providing an applicable model for strengthening authentication and improving user behavior. In the future, the system can be expanded with multi-factor authentication, password recovery mechanisms, and real-time security analysis.

Key Words: *Web Application Security, Email Verification, JSON Web Token (JWT), User Authentication, Django REST Framework.*

Introduction

Scope and Context

In the digital age, where information is distributed and accessed massively through computer networks and cloud services, data security has become one of the most important challenges. Traditional authentication mechanisms, mainly based on passwords, often do not provide sufficient protection, as users tend to create and manage weak credentials. This brings great risks, making the system vulnerable to attacks such as phishing, credential stuffing and brute force.

In this context, the need for additional security mechanisms is essential. One of them is the verification of the email address as an additional step before activating the user account, which ensures not only the validity of the identity, but also a reliable communication channel with the user. In parallel, friendly and well-designed interfaces can serve as educational tools that directly affect the improvement of user behavior towards secure practices.

Motivation and Contribution

The main motivation of this study lies in addressing the gap between advanced security technologies and poor user behavior. Although strong protection mechanisms exist, in practice it is often the user who represents the weakest link in the security chain. By building a system that not only provides secure authentication but also educates users on the importance of identity confirmation and correct credential management, this paper aims to contribute to both the academic and practical fields.

The main contribution of the paper lies in providing a simple, applicable and scalable model for web applications, which combines email verification, JWT authentication and a polite interface, demonstrating how these elements can strengthen security and user awareness at the same time.

Research Questions and Hypotheses

Based on the above challenges, two research questions have been raised:

- How does email verification improve authentication security in web applications?
- In what way can the interface and technology contribute to educating users on secure credential management practices?

The relevant hypotheses are:

- H1: Implementing email verification reduces unauthorized logins and strengthens identity control.
- H2: A user-friendly interface, combined with clear instructions on secure practices, positively affects user education and behavior.

Objectives

To address the questions and hypotheses raised, the paper aims to achieve the following objectives:

1. Developing a functional web application with email verification and JWT authentication.
2. Testing the registration and verification process using secure platforms such as Mailtrap.
3. Providing an interface that helps users understand the importance of their actions in security.
4. Developing an administrative panel for role and access control.
5. Assessing the impact of the system on security and user awareness.

Literature Review

Password Security and User Behavior

Password security remains a central element of digital protection, yet user behavior continues to weaken authentication processes. Multiple studies indicate that users consistently select weak passwords or reuse the same credentials across several accounts, exposing systems to credential stuffing, dictionary attacks, and brute force attempts (Bang et al., 2012; Das et al., 2014). This pattern reveals that the

primary vulnerability often lies not in technology, but in human factors. Research also shows that users frequently underestimate the importance of password strength, assuming they are not significant targets — a perception that significantly reduces security effectiveness.

Psychological and Cognitive Factors

Beyond technical considerations, password management is heavily influenced by psychological and cognitive limitations. Hardman et al. (2022) highlight that users experience "security fatigue" when confronted with frequent or complex password requirements, leading to non-compliance with security recommendations. Memory constraints also drive users toward insecure behaviors, such as reusing patterns, relying on personal information, or storing credentials in unsafe places. In many cases, excessive security requirements result in decreased usability, reinforcing the need to balance cognitive load with secure design principles.

Technological Security Mechanisms

As a response to the inherent weaknesses of password-based systems, more advanced mechanisms have been developed. Multi-factor authentication (MFA) and two-factor authentication (2FA) offer additional layers of protection by requiring independent verification methods, thus significantly reducing the likelihood of unauthorized access (Ometov et al., 2018). Additionally, JSON Web Token (JWT) has emerged as a widely adopted approach for managing stateless authentication in modern web applications (Bonneau et al., 2012). By embedding claims in digitally signed tokens, JWT enhances both security and scalability while reducing server-side complexity.

Educational Interventions and Institutional Policies

Recent literature emphasizes that technology alone cannot resolve authentication vulnerabilities without proper user education. Reeder et al. (2017) demonstrate that interactive, behavior-driven educational methods are more effective than traditional awareness campaigns in improving credential management practices. Institutions now adopt policy frameworks that reflect this reality by promoting longer passphrases, banning commonly used passwords, and reducing reliance on forced periodic password changes (Nieles et al., 2017). These strategies underline the importance of combining secure authentication mechanisms with user-centric approaches that encourage sustainable and secure behavior.

Methodology

Methodological Approach

This study follows an experimental approach based on the development of a functional web application, which serves as a demonstration environment to test the hypotheses raised on the impact of email verification and JSON Web Token (JWT) authentication on strengthening security and user education. The methodology is based on three main pillars: (i) designing the system architecture, (ii) technical and functional implementation of the main components, and (iii) testing to assess security, scalability, and impact on user behavior.

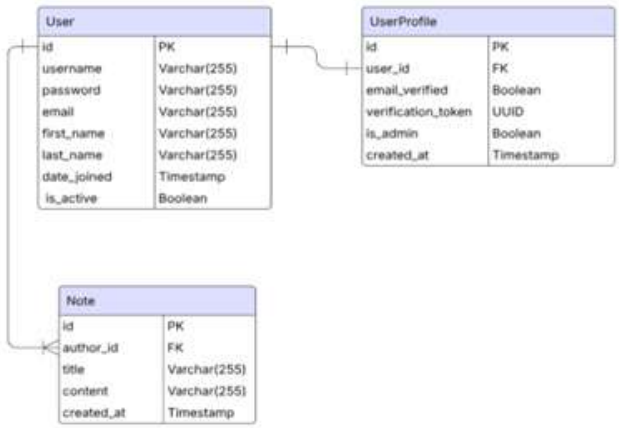
The goal is not only to create a technically viable application, but also to evaluate how technology and interface design contribute to user awareness of secure credential management practices.

System Architecture

The system is built on a client-server architecture, where the frontend (React with Vite) and the backend (Django REST Framework) communicate through a secure REST API. Data is stored in a relational SQL database, which ensures the integrity and organization of information.

The architecture is designed to provide modularity, clear separation of responsibilities and high scalability. The front end provides an interactive interface for users, while the backend contains business logic and security mechanisms.

FIGURE 1. ERD diagram of the database



The ERD diagram reflects two main entities: User and Note. Each note is linked to a user through a foreign key relationship, guaranteeing referential integrity and enabling access control according to ownership.

Technologies used

- React & Vite (Frontend): chosen to build a dynamic SPA (Single Page Application), providing fluid user experience.
- Django REST Framework (Backend): robust framework in Python, with built-in support for authentication, ORM and RESTful API.
- SQL Database: SQLite during development, with the possibility of migrating to PostgreSQL in production environments.
- JWT (JSON Web Token): used for stateless authentication, providing session management without loading the server.
- Mailtrap (SMTP Sandbox): for safe testing of verification emails without risking their delivery to real addresses.
- Django-CORS-Headers: to enable secure communication between frontend and backend in separate development environments.

Implementing the main components

User registration and email verification
User registration involves a two-step process: account creation and email address verification.

- On the frontend, the user fills out the registration form with username, email, and password. Validations are performed on both the frontend and backend.
- On the back end, the user is stored in the database with an inactive status until they confirm their email.
- A verification token is generated, which is included in a custom link and sent via Mailtrap.
- Clicking the link redirects the user to a special page in React, which confirms activation via the Django API.

FIGURE 2. User registration interface

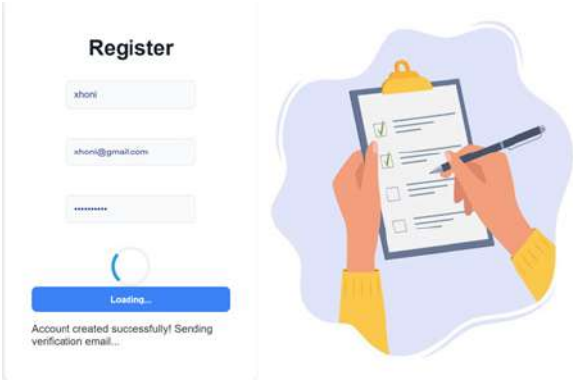


FIGURE 3. Email generated by Mailtrap for verification

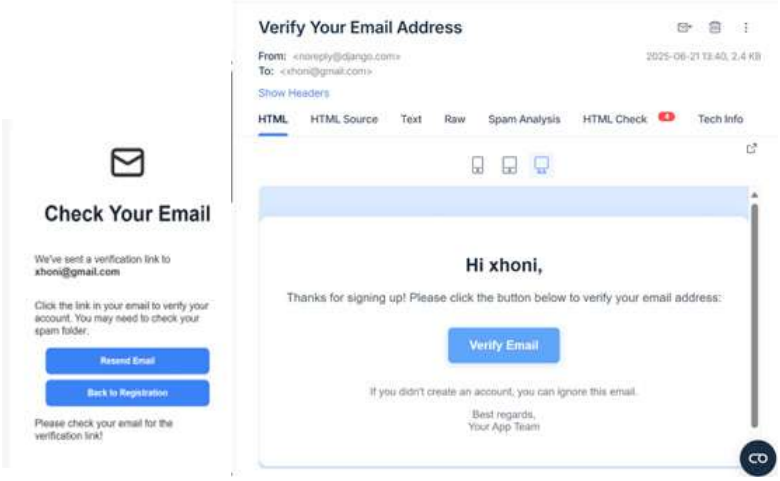
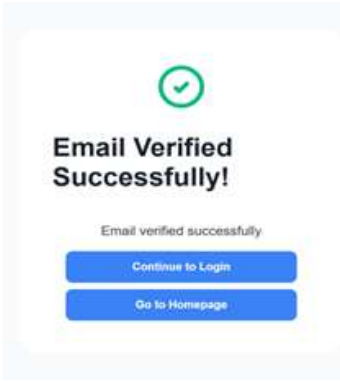


FIGURE 4. Email verification confirmation message



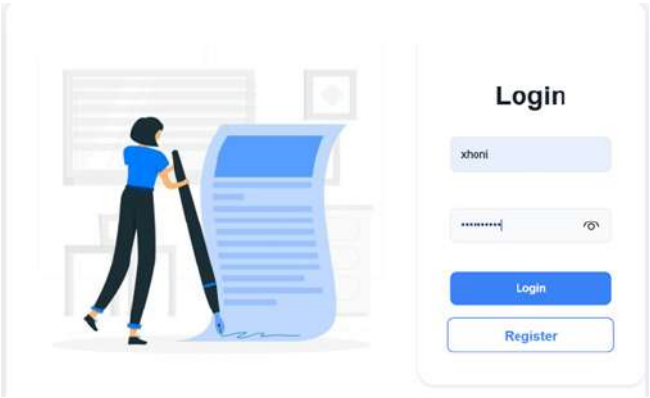
This mechanism ensures that each account is associated with a valid address, preventing the mass creation of fake accounts and increasing user awareness of the importance of this process.

Authentication and session management

After account verification, the user can log in to the system using email and password. The backend generates a JWT, which is signed with a secret key and returned to the client. The token is stored in localStorage and attached to each API request via the Authorization header.

JWTs are only valid for a limited time (15 minutes), limiting the risk in case of compromise. For longer sessions, the system provides for the use of refresh tokens.

FIGURE 5. User login interface



This mechanism eliminates the need to store sessions on the server and provides a more secure and scalable model.

User dashboard and note management

After authentication, the user is directed to a personal dashboard, where he can create, view and delete his notes. All operations are protected by JWT authentication, and each user has access only to their own records.

FIGURE 6. User Dashboard

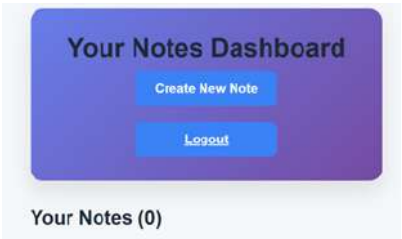


FIGURE 7. Creating a new record

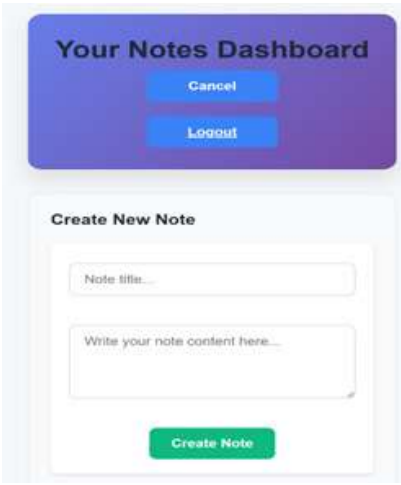
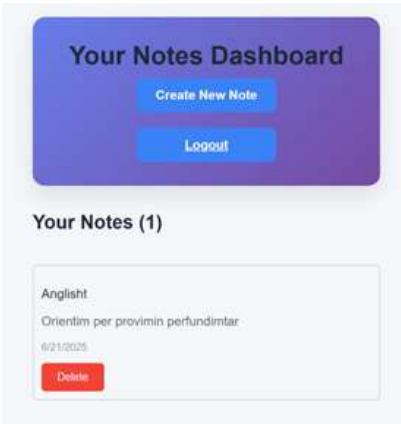


FIGURE 8. The process of deleting a record



This functionality demonstrates the practical application of access control based on data ownership and educates users on the importance of privacy.

Admin Panel

In addition to regular users, the system also provides the administrator role. Through the Django admin panel, admins can:

- Activate or deactivate accounts.
- Promote users to new roles.
- Delete accounts and all associated records.

FIGURE 9. Admin Panel

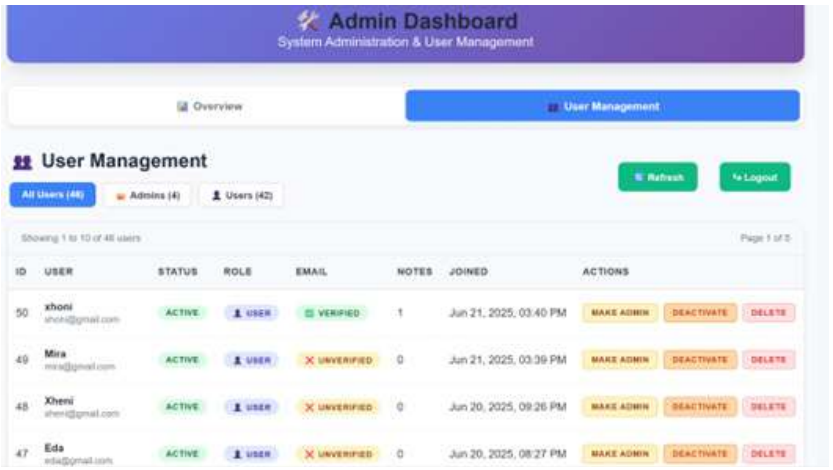
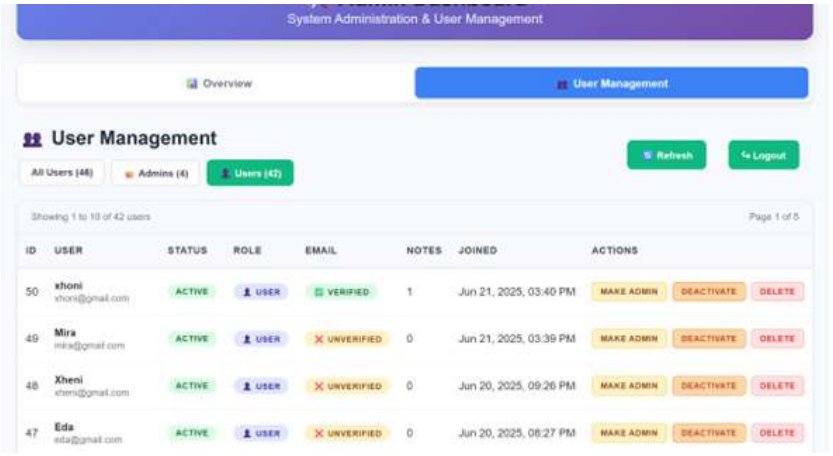


FIGURE 10. Managing users and roles



This layer of control is essential for system maintenance and the overall security of the platform.

Integration and communication security

Since the front end and backend ran on different hosts, the CORS mechanism was used to allow secure communication. This approach guarantees that only trusted origins can access the API, limiting the risk of abuse by unauthorized applications.

Security and Functionality Testing

Functional Testing

The cases of registration, email verification, login, creating and deleting notes, as well as user management from the admin panel were tested. In all cases, the system functioned according to specifications, respecting user roles and privileges.

Security Testing

Tests were conducted to verify protection against common attacks:

- SQL Injection: prevented by using Django's ORM.
- XSS (Cross-Site Scripting): limited by React's policy of not executing injected HTML.

FIGURE 11. SQL Injection Testing

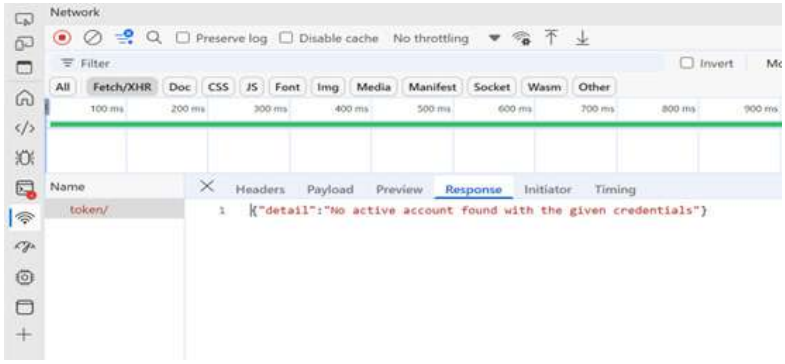


FIGURE 12. XSS Testing (Part 1)

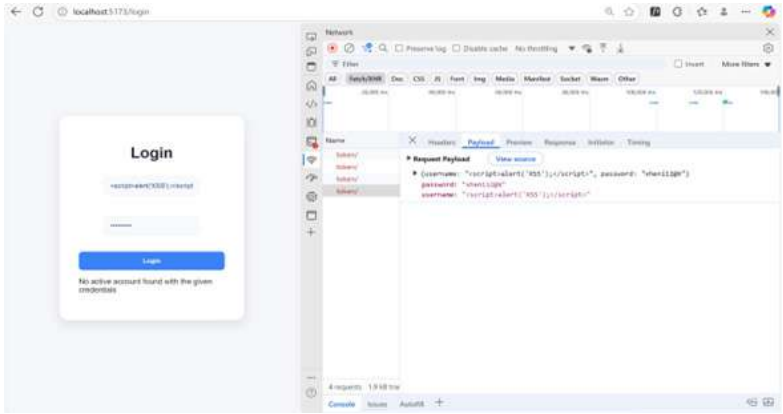
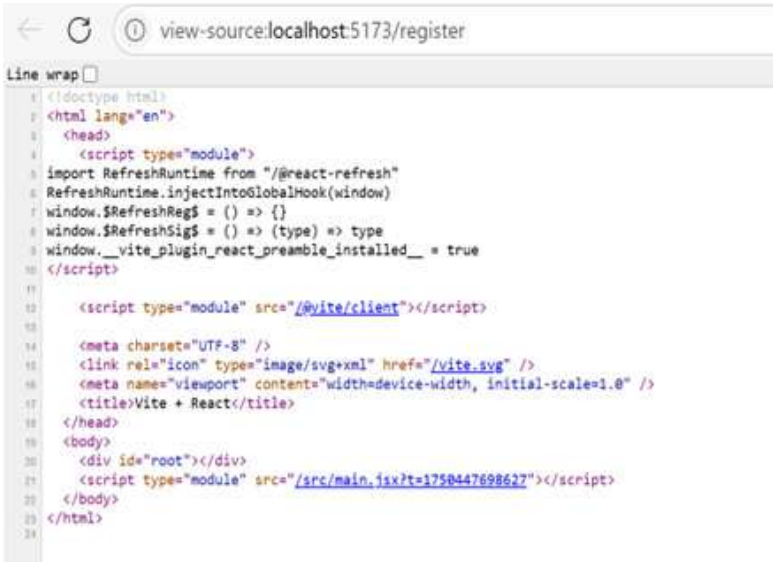


FIGURE 13. XSS Testing (Part 2)



Reflection on Methodology

The approach used demonstrated that combining modern technologies with secure development practices provides a robust model for web applications. The implementation of email verification and JWT authentication not only increased the level of security but also contributed to user education by making them aware of authentication processes and their role in maintaining security.

Results and Discussion

Main Results

The registration and email verification process worked as expected, ensuring that no user was able to log in without first confirming their email address. This mechanism proved its value as an indispensable tool to prevent the creation of fake accounts, making the system more stable and reliable. Furthermore, users were faced with a clear experience that guided them on the importance of this process, transforming it into an educational tool in addition to its technical function.

JSON Web Token (JWT) authentication also showed high effectiveness in managing user sessions. Each user, after successful identification, was provided with a token that had a limited time validity, thus reducing the risk of abuse in case of compromise. Testing confirmed that the system automatically rejected any

unauthorized request, proving that access control was implemented correctly and securely. The user dashboard was created as an isolated environment where each person had access only to their personal records. This organization ensured that data ownership rights were fully respected, guaranteeing information privacy and making the interface clear and functional for individual content management.

At the same time, the admin panel offered advanced functionality for user management. Administrators had the ability to activate or deactivate accounts, delete specific users, or promote them to new roles. Any intervention made in this panel was immediately reflected in the system's behavior, confirming full and centralized control over the platform's access and operation.

Finally, security testing confirmed that the application was protected from the most common attacks, such as SQL Injection and Cross-Site Scripting (XSS). Thanks to the use of ORM in Django, the injection of malicious commands into the database became impossible, while React's policy for handling injected HTML effectively limited the possibility of executing malicious scripts. These results showed that the developed application complies with basic security standards and provides a reliable level of protection for user data.

Discussion of the results

The test results confirm that the combination of email verification and JWT provides a level of security comparable to industry's best practices. In addition to technical assurance, the system also contributed to user awareness, as the verification process made them more aware of the importance of authentication and identity control.

Educational value:
Users were presented with clear messages during registration and login, where the system explained why email verification was required and why login could not be completed without this step. This created an educational interaction that goes beyond technical functionality.

Accessibility limitations:
However, the current system does not include multi-factor authentication (MFA), which is considered one of the highest security standards. Also, the use of Mailtrap is limited to test environments; in a real environment, integration with an external email service and verification of message delivery would be necessary.

Comparison with literature:
The literature suggests that user behavior remains the weakest link in the security chain (Bang et al., 2012; Hardman et al., 2022). The results of this study support

this finding but also demonstrate that combining technological practices with interface design can have a significant impact on improving their behavior.

Practical implications:
This project provides a simple and applicable model for web applications that want to integrate basic security mechanisms, focusing on both the technical aspect and user education. The system can serve as a practical environment for cybersecurity training, as well as a basis for further developments in larger applications.

Conclusions and Future Work

This study aimed to explore the impact of email verification and JSON Web Token (JWT) authentication mechanisms in strengthening the security of web applications and in increasing user awareness of proper credential management practices. The implementation of a functional application served as a practical test that demonstrated that these mechanisms do not only provide technical protection but can also be used as educational tools for users.

The results clearly showed that the email verification process limits the creation of fake accounts and strengthens identity control, while the use of JWT provides a secure and scalable authentication model. The personalized user dashboard and the administrator panel illustrated how access control can be implemented at different levels, respecting the principles of privacy and centralized management. Security testing demonstrated that the developed system was protected against a range of common attacks, such as SQL Injection and Cross-Site Scripting (XSS), ensuring data integrity.

Another important finding was the educational dimension of the system. Users were presented with a clear and structured registration and login process, which not only ensured controlled access, but also made them more aware of their role in protecting their personal data. This element reflects the findings of the literature that emphasize that the user often remains the weakest link in the security chain, but a good interface design and clear explanatory messages can directly influence the change of their behavior.

Although the system worked as expected, its limitations should not be overlooked. Currently, the application does not include multi-factor authentication (MFA), which is a standard practice in modern applications with a high level of security. Also, the use of Mailtrap is limited to test environments only; for real applications, integration with a professional email service and large-scale delivery management would be required.

In future work, this system could be extended with additional modules such as password recovery, multifactor authentication integration, and the development

of real-time security monitoring mechanisms. Another valuable direction is the integration of user behavior analysis, using simple artificial intelligence algorithms to identify abnormal behavior and prevent unauthorized access.

In conclusion, the paper contributes by providing a practical and simple model for building more secure web applications, which combine technological mechanisms with an educational approach to users. The results show that even with simple and accessible tools, a balance between technical protection and awareness raising can be achieved, making these solutions suitable for adoption in a wide range of web applications in practice.

References

Bang, J. M., Karlsson, F., & Tehler, H. (2012).User compliance and password security: Investigating a user-centric framework.*Information & Computer Security, 20(4), 332–348.*

Bonneau, J., Herley, C., Van Oorschot, P. C., & Stajano, F. (2012). The quest to replace passwords: A framework for comparative evaluation of authentication schemes. *IEEE Symposium on Security and Privacy*, 553–567.

Dhanjani, N. (2015). *Abusing the Internet of Things: Blackouts, freakouts, and stakeouts.* O'Reilly Media.

Hardman, D., Zois, D. S., & Tzovaras, D. (2022). Usable security: Balancing user behavior and authentication security. *Computers & Security, 113*, 102546.

Krombholz, K., Busse, K., Pfeffer, K., Smith, M., & Grechenig, T. (2017)."If HTTPS were secure, I wouldn't need 2FA"—End user and administrator mental models of HTTPS.*IEEE Symposium on Security and Privacy*, 246–262.

Nash, J., & Biddle, R. (2020).Passwords at the Crossroads: User choices and system design. *International Journal of Human-Computer Studies, 137*, 102383.

Nieles, M., Dempsey, K., & Pillitteri, V. Y. (2017).An introduction to information security (NIST SP 800-12 Rev.1).National Institute of Standards and Technology.

Ometov, A., Bezzateev, S., Mäkitalo, N., et al. (2018). Multi-factor authentication: A survey. *Cryptography, 2(1), 1.*

Pfleeger, C. P., Pfleeger, S. L., & Margulies, J. (2015).*Security in computing* (5th ed.). Pearson.

Schneier, B. (2015). *Secrets and lies: Digital security in a networked world* (Updated ed.). Wiley.

Ur, B., Shay, R., Komanduri, S., et al. (2015).Can users behave securely without suffering a usability penalty? *CHI Conference on Human Factors in Computing Systems*, 157–166.

AI Based Automated Traffic Monitoring System for Vehicles and License Plate Recognition

***Read DANJOLLI*[1]**

Abstract

This paper presents a comprehensive architecture for automated traffic violation surveillance. It is based on sophisticated deep learning algorithms and artificial intelligence systems with computer vision. The main objective is to develop an integrated pipeline that integrates vehicle detection, Automatic License Plate Recognition (ALPR), and visual attribute classification (e.g., color, manufacturer, and model). YOLO detection, DeepSORT tracking, CRNN network OCR, and CNN for car brand and color categorization are all parts of the technical solution. The study fully compares Edge and Cloud architectures, examining how well they perform under different conditions, such as high traffic and poor lighting. The findings show that, while Cloud solutions offer more flexibility but at a higher latency cost, Edge solutions, despite their processing limitations, achieve response times below 200 ms and accuracy above 95% in license plate identification.

Along with specific implementation recommendations for the Albanian context, the study addresses algorithmic fairness, privacy protection and GDPR compliance. It also addresses the ethical and legal elements of using surveillance technologies, highlighting the prospects and challenges for a successful adoption in Albania.

Furthermore, to compensate for the personalized data pages for the Albanian market, synthetic data models were included in the initial training. This was

[1] Read Danjolli graduated in Information Technology – Economic Informatics at the European University of Tirana. He also holds a master's degree from the Faculty of Engineering, Informatics and Architecture at the European University of Tirana, in Computer Engineering (MSc), Software Engineering profile.

sufficient on the ground to allow for higher algorithmic adaptability. Investments in human resource training and a well-defined framework are also necessary for the deployment of technologies to ensure accountability, transparency and responsibility for all. This comprehensive strategy lays the foundation for an automated application system in Albania that is reliable and sustainable.

Key words: *Automated Traffic Surveillance, Automatic License Plate Recognition, Vehicle Attribute Recognition, Deep Learning & Computer Vision, Edge and Cloud Architecture, Ethical & GDPR Compliance*

Introduction

Introduction and Background

Road safety and efficient traffic management are fundamental pillars of modern societies. The ability of a nation to ensure safe mobility directly impacts its social welfare, economic productivity, and environmental sustainability (World Health Organization, 2023). Albania, like many developing economies, has experienced a dramatic increase in the number of registered vehicles in the past three decades. More than 700,000 vehicles are now circulating in a relatively small and urbanizing country, reflecting greater economic access to transportation but also introducing significant costs in terms of congestion, urban pollution, and rising accident rates (European Transport Safety Council, 2022).

Statistics show that thousands of traffic incidents are recorded annually in Albania, with dozens of fatalities and hundreds of serious injuries. These events translate into economic losses exceeding 2.5% of the national GDP, through healthcare costs, productivity losses, infrastructure damage, and delays in mobility (INSTAT, 2023). Beyond their economic weight, these incidents expose the inability of current institutional frameworks to regulate and enforce traffic laws effectively. Police patrols and conventional monitoring systems lack the scalability and precision to handle the complexities of modern traffic in Albania's rapidly growing cities.

This situation places Albania in a unique position. On one hand, it suffers from the absence of a centralized and automated system for identifying and sanctioning traffic violations. On the other hand, its manageable size and urgent need for digitalization create an opportunity to adopt cutting-edge intelligent surveillance technologies. By integrating Artificial Intelligence (AI), Computer Vision, and Automated License Plate Recognition (ALPR) systems, Albania could leapfrog traditional approaches and establish a transparent, efficient, and accountable traffic management ecosystem.

However, such a transformation cannot occur in a vacuum. Ethical, legal, and social concerns must accompany technological progress. AI-driven surveillance brings questions of privacy, fairness, and compliance with European regulations such as the General Data Protection Regulation (GDPR). Ensuring public trust, institutional accountability, and legal legitimacy will be critical for success (European Union Agency for Fundamental Rights, 2021).

Problem Statement

The core research problem addressed in this study lies in designing an integrated pipeline capable of recognizing vehicle license plates and associated attributes—such as color, make, and modeling real-world conditions. This problem is multidimensional, cutting across technical, infrastructural, and ethical domains.

ALPR and Vehicle Attribute Recognition (VAMR) systems must function reliably in diverse scenarios: poor lighting, nighttime conditions, adverse weather (rain, fog, snow), high vehicle speeds, occlusions and damaged or non-standard plates. Such environmental and operational variables significantly challenge AI algorithms, which often underperform when exposed to conditions outside their training datasets (Redmon & Farhadi, 2018).

Albania faces an additional complexity due to the coexistence of multiple license plate formats, national, historical, and foreign. Flexible algorithms are required to recognize this heterogeneity, including multilingual fonts, varying colors, and design elements. Similarly, vehicles on Albanian roads exhibit great variety in brand and model, often making fine-grained classification difficult. Most existing solutions treat license plate recognition and vehicle attribute recognition separately. The challenge is to integrate these tasks into a single coherent architecture, where the outputs of one module reinforce the reliability of the other. This raises architectural and algorithmic issues, particularly concerning error propagation and confidence fusion between subsystems (Li et al., 2019).

Law enforcement applications demand real-time inference, often across multiple cameras deployed simultaneously. This introduces trade-offs between computational cost, latency, and system scalability. Choosing between Edge computing, where data is processed locally on embedded devices, and Cloud computing, where data is centralized for processing, is a major design decision.

High-performing Deep Learning models require large, high-quality annotated datasets. Albania lacks curated traffic surveillance datasets tailored to its specific conditions. Manual data collection and annotation is resource-intensive, while synthetic data generation and transfer learning only partially mitigate this gap.

Finally, AI-based traffic surveillance inevitably raises ethical dilemmas. License plates are legally considered personal data under the GDPR, and improper storage or processing risks violating individual rights. Biases in algorithmic performance

could disproportionately impact minority groups, while misidentifications could lead to unjust sanctions (Barocas, Hardt, & Narayanan, 2019). Addressing these concerns requires robust anonymization techniques, transparent auditing mechanisms, and public communication strategies.

Objectives, research questions and hypothesis

In response to these challenges, this study sets fifth the following objectives:

- To analyze and evaluate the application of state-of-the-art AI techniques, especially Deep Learning, in ALPR and VAMR tasks under Albanian traffic conditions.
- To design an integrated architecture that combines license plate recognition and vehicle attribute recognition in a unified pipeline.
- To develop or adapt Deep Learning models capable of identifying vehicle color, make, and model with high accuracy.
- To assess system performance using standardized metrics, focusing on accuracy, speed (frames per second), and robustness in real-world conditions.
- To examine ethical, legal, and social implications of deploying automated surveillance in Albania, ensuring compliance with GDPR and public trust.

Research Questions

Building on the objectives, the study frames its inquiry around several guiding research questions (RQs):

RQ1: What are the most effective Deep Learning architectures for implementing ALPR and VAMR in high-volume traffic conditions?
RQ2: How can the robustness of ALPR systems be improved to handle low lighting and adverse weather conditions?
RQ3: What is the optimal architecture for integrating ALPR and VAMR modules into a unified pipeline, balancing accuracy and computational efficiency?
RQ4: What are the trade-offs between Edge and Cloud inference for real-time automated traffic surveillance?
RQ5: What ethical, legal, and privacy considerations must be addressed for large-scale deployment in Albania?

The main **hypothesis** guiding this research is: ***AI-driven solutions significantly enhance speed, accuracy, and reliability in traffic violation detection compared to traditional systems, thereby improving road safety and enforcement efficiency.***

Literature Review and Gaps

Evolution of Traffic Surveillance Technologies

Traffic monitoring has undergone a remarkable transformation over the past century. Early methods were entirely manual, with police officers stationed at intersections to observe violations and direct traffic flow. In the mid-20th century, pneumatic tubes and inductive loop detectors (ILDs) represented the first steps toward automation, offering basic vehicle counts and speed measurements.

The proliferation of Closed-Circuit Television (CCTV) in the 1960s–1970s introduced visual monitoring, but systems still relied heavily on human operators. In the 1980s–1990s, Video Incident Detection (VID) systems emerged, using background subtraction and motion analysis to detect sudden stops, wrong-way driving, and congestion (Parker & Harris, 1998). Yet these were fragile under lighting variation and weather changes.

From the 1990s onwards, radar and LiDAR became widely used for speed enforcement and red-light violation detection, enabling partial automation. These systems, however, were limited in their ability to identify vehicles beyond plate numbers, offering little in terms of broader traffic analytics (Zhang et al., 2017).

The 2010s marked the era of **AI-powered traffic surveillance,** where Convolutional Neural Networks (CNNs) revolutionized object detection and recognition. Today, AI-based ALPR/VAMR systems can detect vehicles, read plates, and classify attributes (color, make, model) with high accuracy in near real-time, representing a leap in automation, scalability, and analytical depth (Redmon & Farhadi, 2018).

Artificial Intelligence in Intelligent Transportation Systems (ITS)

AI integration has significantly expanded the capabilities of Intelligent Transportation Systems (ITS). Its strengths lie in processing vast amounts of heterogeneous data—from cameras, sensors, GPS, and mobile applications—to support predictive modeling, real-time decision-making, and autonomous operation. Applications include:

- Traffic flow prediction: Machine Learning models forecast congestion using time-series analysis
- Adaptive traffic light control: Reinforcement Learning optimizes signal timings in real time.
- Navigation systems: Platforms like Google Maps and Waze apply AI to predict travel times and reroute drivers dynamically.

- Autonomous driving: Deep Learning enables environment perception, sensor fusion, and decision-making in self-driving cars.
- Law enforcement: Computer Vision detects violations such as speeding, red-light running, illegal parking, and distracted driving (Li et al., 2019).

AI thus transforms ITS from reactive infrastructures into proactive, intelligent ecosystems that optimize safety, efficiency, and sustainability.

FIGURE 1: Example of vehicle and license plate recognition.

Automated License Plate Recognition (ALPR)

ALPR is a cornerstone technology in traffic enforcement. It typically involves four stages: image acquisition, plate detection, character segmentation, and optical character recognition (OCR). Advances in object detection, particularly YOLO (You Only Look Once), SSD (Single Shot Detector), and Faster R-CNN, have greatly improved plate localization. OCR accuracy has been boosted by CRNNs (Convolutional Recurrent Neural Networks), which combine CNNs and LSTMs for end-to-end recognition without explicit character segmentation (Shi et al., 2017).

Challenges persist, however, in dealing with varying plate designs, occlusions, glare, damaged characters, and high-speed motion blur. Benchmarks show state-of-the-art ALPR systems achieving 95–98% accuracy under controlled conditions but lower performance in adverse real-world scenarios.

Vehicle Attribute Recognition (VAMR)

While license plates uniquely identify vehicles, attribute recognition enhances reliability and utility. Color, make, and model recognition aid in detecting cloned or stolen plates, re-identifying vehicles across camera views, and enabling traffic composition analysis (Sochor et al., 2018).

Fine-grained classification is the key challenge: distinguishing between visually similar models or color shades requires high-quality datasets and sophisticated CNN architectures. Large,47784 annotated datasets such as CompCars and BoxCars116k have accelerated progress (Yang et al., 2015). Transfer learning from models trained on ImageNet has also proven highly effective.

Ethical and Privacy Concerns

The expansion of surveillance technologies has sparked extensive debate on privacy and civil liberties. ALPR inherently creates location trails of vehicles, which, if stored extensively, risk enabling mass surveillance beyond traffic law enforcement. GDPR explicitly classifies license plates as personal data, requiring strict adherence to principles of data minimization, storage limitation, and purpose specification.

Algorithmic fairness is another critical concern. Biases in training data can produce disproportionate misidentification rates across demographic groups, undermining trust and legitimacy (Buolamwini & Gebru, 2018). Transparent auditing, explainability, and public engagement are increasingly emphasized as safeguards (European Commission, 2021).

Despite notable advancements, several research gaps persist. First, there is a lack of integrated evaluations, as most studies analyze ALPR and VAMR systems separately rather than as unified, real-world pipelines. Second, regional adaptation remains limited, with datasets primarily reflecting Western formats and failing to generalize to regions like Albania. Third, edge–cloud trade-offs are insufficiently explored, particularly regarding latency, cost, and privacy balance. Fourth, the field suffers from benchmarking inconsistencies, as new Deep Learning architectures emerge faster than standardized evaluations. Finally, ethical considerations are often discussed broadly, without adaptation to specific national and cultural contexts.

Computer Vision and Deep Learning Foundations

Computer Vision (CV) provides the theoretical and practical foundation for automated traffic surveillance. Its objective is to enable machines to interpret visual information from images and video streams. Historically, CV relied on

handcrafted feature extraction methods such as Scale-Invariant Feature Transform (SIFT) and Histogram of Oriented Gradients (HOG), which required human expertise to design features relevant for object detection.

The emergence of **Deep Learning**, particularly Convolutional Neural Networks (CNNs), revolutionized the field by enabling end-to-end learning. CNNs automatically extract hierarchical visual features, from low-level edges to high-level object representations, thereby surpassing handcrafted approaches in robustness and scalability (LeCun, Bengio, & Hinton, 2015). For sequence-based tasks, Recurrent Neural Networks (RNNs) and their variants, such as Long Short-Term Memory (LSTM), handle temporal dependencies in traffic video streams. Together, CNN-RNN hybrids such as CRNNs are particularly effective in character recognition for license plates (Shi et al., 2016).

Data Requirements for Surveillance Models

Deep learning systems rely heavily on large and well-curated datasets to achieve effective training and reliable performance. The size, diversity, and quality of the dataset play a crucial role in determining how well a model can generalize to new, unseen scenarios. To ensure this generalization, several factors must be carefully considered during dataset design and preparation.

One of the most important aspects is the diversity of environments represented in the dataset. Models trained only under ideal conditions often fail when faced with real-world variability. Therefore, datasets should include images captured in different lighting situations—day and night—as well as under various weather conditions such as rain, fog, or snow. Additionally, the inclusion of different traffic densities and urban or rural scenes enhances the system's ability to perform consistently across diverse environments.

Another key consideration is balance representation. In traffic surveillance and vehicle recognition systems, certain vehicle types, such as sedans or compact cars, are much more common than others, like buses, motorcycles, or trucks. If a dataset reflects this imbalance, the model may become biased, performing well in frequent classes but poorly on rare ones. Ensuring an even distribution of vehicle categories helps maintain fairness and accuracy across all types.

To further strengthen dataset quality, data augmentation techniques are widely applied. Methods such as random cropping, image rotation, flipping, and brightness adjustments can artificially increase the size of the dataset. These transformations expose the model to a broader range of visual variations, making it more robust to distortions, angles, and lighting changes that occur in real-life traffic footage.

In addition, transfer learning provides an efficient solution for overcoming data limitations. Instead of training a model entirely from scratch, researchers

can fine-tune neural networks that have already been pretrained on large-scale datasets like ImageNet. This approach allows models to benefit from previously learned visual features and significantly reduces the need for massive amounts of new, domain-specific data (Kornblith et al., 2019).

Another promising technique is the use of synthetic data computer-generated imagery that replicates real-world traffic scenes. Synthetic datasets can simulate different conditions, vehicle types, and perspectives that may be difficult or expensive to capture manually. However, successful integration of synthetic and real data requires careful domain adaptation, ensuring that models trained on artificial images can perform effectively when applied to real environments.

In the context of Albania, one of the major challenges in developing deep learning-based traffic surveillance systems is the lack of annotated, high-quality datasets. To address this, a hybrid strategy is recommended. This involves combining synthetic datasets with transfer learning approaches while gradually collecting and annotating real-world images from Albanian traffic environments. Such a method not only accelerates initial system development but also lays the foundation for continuous improvement as more local data becomes available.

Through this integrated approach—balancing diversity, augmentation, transfer learning, and synthetic generation—deep learning models for traffic surveillance in Albania can achieve greater accuracy, adaptability, and long-term scalability.

Preprocessing Pipelines: Detection and Tracking

Traffic surveillance systems operate through structured pipelines that begin with the detection of vehicles and license plates and proceed with tracking them across consecutive frames. This process ensures that every detected object is not only recognized but also consistently followed over time, allowing accurate monitoring of traffic flow and potential violations.

The first step, object detection, focuses on identifying vehicles or license plates within individual frames. One of the most influential models in this domain is YOLO (You Only Look Once), which treats detection as a single regression problem over bounding boxes and class probabilities (Redmon & Farhadi, 2018). YOLO's remarkable speed and efficiency make it ideal for real-time traffic analysis, where high frame rates and immediate detection are essential.

Following detection, object tracking maintains the continuity of identified vehicles across frames. A leading approach is DeepSORT, which enhances the original SORT algorithm by combining Kalman filtering with deep feature embeddings (Wojke, Bewley, & Paulus, 2017). This enables the system to preserve the identity of vehicles even when they temporarily disappear due to occlusions or overlaps.

Together, these methods form a "tracking-by-detection" architecture, an integrated framework in which detection provides bounding boxes for each object, and tracking ensures their consistent identification throughout the video sequence. This synergy between detection and tracking is fundamental for building reliable, real-time traffic surveillance systems capable of continuous and accurate vehicle monitoring.

Key Algorithms and Architectures

TABLE 1: Summary of key algorithms used in ALPR and vehicle attribute recognition systems.

Algorithm	Architecture Type	Primary Role	Strengths	Challenges
YOLO (v3–v5)	CNN, one-stage detector	Vehicle and plate detection	High speed, real-time performance	Struggles with small or distant objects
DeepSORT	Kalman filter + CNN embeddings	Multi-object tracking	Robust to short-term occlusion, preserves identity	Requires high-quality detectors, failures under long occlusion
CRNN	CNN + RNN/ LSTM hybrid	License plate OCR	End-to-end recognition without segmentation	Sensitive to low-quality plate images
ResNet, VGG	CNN classifiers	Vehicle attribute recognition (color, make, model)	Strong classification ability, transfer learning	Requires large, annotated datasets
CTC Loss	Loss function	Sequence prediction training	Enables flexible sequence alignment	Requires careful hyperparameter tuning

Together, these algorithms form the backbone of an integrated ALPR+VAMR system, capable of high-accuracy detection and classification in real-world traffic conditions.

Edge vs. Cloud Implementation

A critical architectural consideration in intelligent traffic surveillance systems is the choice between edge and cloud computing for model inference and data processing. This decision directly affects performance, latency, scalability, and privacy.

In cloud computing architectures, processing is centralized on remote servers equipped with high-performance GPUs. This setup offers several advantages, including powerful computational capacity, seamless deployment of model updates, and the ability to aggregate and analyze large volumes of data for continuous system improvement. However, cloud-based approaches also

introduce notable drawbacks: transmitting high-resolution video streams to the cloud increases latency, makes performance dependent on network stability, and raises privacy concerns due to the handling of sensitive vehicle and personal data over the internet.

Conversely, edge computing performs inference locally, directly on-site through embedded devices such as NVIDIA Jetson or other AI accelerators. This approach significantly reduces latency, often to below 200 milliseconds, minimizes bandwidth consumption, and enhances data privacy, as sensitive information is processed locally rather than transmitted externally. Additionally, edge systems remain functional during network outages, providing greater resilience. Nonetheless, they face challenges such as limited computational resources, higher initial hardware costs, and the complexity of maintaining and updating distributed devices in the field.

To reconcile these trade-offs, a hybrid Edge–Cloud architecture has emerged as a practical and efficient solution. In this design, immediate tasks such as vehicle detection and basic classification are handled locally at the edge, while more resource-intensive processes—like advanced analytics, retraining, and long-term data management—are offloaded to the cloud. This configuration effectively combines the low latency and privacy benefits of edge computing with the scalability and computational power of the cloud, providing a robust framework for modern, real-time traffic surveillance systems (Satyanarayanan, 2017).

Hardware Considerations

The performance of AI-driven traffic surveillance systems is heavily dependent on the underlying hardware infrastructure, as each component contributes directly to system accuracy, speed, and reliability.

At the foundation are the cameras, which serve as the system's primary sensors. Their resolution, frame rate, and low-light sensitivity determine the clarity and usability of captured footage. Advanced features such as Wide Dynamic Range (WDR) allow effective monitoring in environments with varying lighting conditions, such as bright sunlight or deep shadows, while infrared (IR) support ensures continuous, 24-hour operation even in low-visibility settings. Equally important are Graphics Processing Units (GPUs), which accelerate deep learning inference and enable real-time performance for demanding tasks like Automatic License Plate Recognition (ALPR). Without sufficient GPU capability, system latency increases, reducing the effectiveness of live monitoring and rapid violation detection.

For edge-based deployments, embedded systems such as System-on-Chip (SoC) platforms, like the NVIDIA Jetson series or Google Coral Edge TPU—offer a practical balance between power efficiency and computational capability.

These compact devices can process video streams locally, making them ideal for decentralized surveillance setups.

In addition, reliable storage and networking components are essential for maintaining data integrity. High-capacity local storage ensures temporary buffering and backup during connectivity interruptions, while stable, high-bandwidth connections allow for smooth data transmission to central servers when needed.

In the context of Albania, where cost-effectiveness is a key concern, lightweight yet capable solutions are more practical than large-scale GPU clusters. Platforms such as Jetson Nano or Jetson Xavier provide robust performance at a fraction of the cost, making them ideal for pilot projects and early-stage system deployment. These configurations balance affordability and performance, enabling sustainable development of AI-based traffic surveillance across the country.

License Plate Recognition (ALPR) and Vehicle Attribute Recognition (VAMR)

Detection: Real time ALPR Systems Automatic License Plate Recognition (ALPR) involves the detection and identification of vehicle license plates within images or video streams.

The initial stage, License Plate Detection (LPD), aims to accurately locate the region containing a license plate. The overall performance of an ALPR system heavily depends on the precision of this detection. For traffic monitoring applications, real-time detection is essential, requiring a processing speed (FPS – Frames Per Second) that can match the video feed.

Real-world traffic scenarios present multiple challenges for plate detection. Variations in lighting, from bright sunlight to shadows or low-light conditions, affect visibility. Weather conditions such as rain, fog, or snow may obscure plates. Camera angles and distances result in varying perspectives and plate sizes, while high vehicle speeds can introduce motion blur. Partial occlusions caused by trailers, other vehicles, or dirt, as well as background textures resembling plates (e.g., advertisements or signs), can generate false positives.

Traditional LPD methods relied on handcrafted features such as vertical and horizontal edges, aspect ratios, and plate colors, using tools like Sobel and Canny edge detectors or Hough line transforms. However, modern real-time ALPR systems predominantly employ single-stage deep learning models, such as YOLO, which predict bounding boxes and classes in a single pass. Recent versions of YOLO (v5, v7, v8) offer device-specific variants and advanced training techniques. Models trained on local (Albanian) and international license plate datasets can achieve high-speed, accurate detection.

FIGURE 2: Example of integrated driver detection and license plate recognition in a real traffic environment.

OCR for different international plate styles

OCR converts the cropped license plate image into an alphanumeric string representing the vehicle's registration number. Plate formats vary considerably across regions. Some have a single row of characters, while others feature multiple rows. Fonts differ, with stylized or region-specific designs, often leading to confusion between visually similar characters ('O' vs '0', 'I' vs '1', 'B' vs '8').

Character sets also vary: some plates use only uppercase Latin letters and Arabic numerals, whereas others include lowercase letters, special characters, or symbols from non-Latin scripts such as Cyrillic, Arabic, or Chinese. Background and character colors differ (e.g., black on yellow), which can assist in localization or recognition. Plate materials range from retroreflective to non-reflective, and security features such as holograms or watermarks may affect OCR accuracy.

Robust OCR models must handle these variations, combining image preprocessing, character segmentation, and classification techniques. Deep learning approaches, particularly convolutional neural networks (CNNs) and transformer-based models, have shown superior performance in handling fonts, layout, and color diversity. Additionally, integrating contextual rules, such as expected plate formats for a given country, enhances accuracy and reduces errors caused by ambiguous characters or low-quality images.

ALPR in the Albanian Context and OCR Using Deep Learning

In the Albanian context, ALPR systems must recognize both current and historical Albanian license plate formats, as well as a wide range of international plates from neighboring countries and beyond.

After license plate detection, Optical Character Recognition (OCR) typically follows a two-step process. First, character segmentation separates each character into individual regions. Next, each character is classified using techniques such as neural networks, support vector machines (SVMs), or template matching.

Traditional approaches are sensitive to errors caused by poor image quality, complex backgrounds, or closely spaced characters.

Modern deep learning models, particularly Convolutional Recurrent Neural Networks (CRNNs), provide effective solutions. CRNNs initially apply convolutional layers to extract visual features from the plate image. These features are then processed through recurrent layers, such as LSTMs, which capture sequential dependencies among characters. A specialized CTC (Connectionist Temporal Classification) loss function enables training without precise character segmentation, making CRNNs suitable for plates with varying lengths and layouts.

Training a CRNN for international plate recognition requires a large, diverse dataset encompassing plates from multiple countries and different formats. Data augmentation, including changes in image angles, lighting conditions, or artificial noise, enhances model robustness. Additionally, transfer learning can be applied: a model pretrained on a large, general dataset (e.g., plates from other regions) is fine-tuned on a specific dataset containing Albanian and other relevant plates.

Although most models are primarily trained for Latin letters and Arabic numerals, their architecture is flexible, allowing adaptation to recognize characters from other alphabets if needed. This approach ensures accurate recognition across a wide variety of plate styles, supporting robust ALPR systems in both national and international traffic monitoring contexts.

Handling Non-Standard and Multilingual Plates

ALPR systems often face challenges beyond standard plate formats. These include vanity plates with unusual characters or spacing, damaged or faded plates, and dirty or obstructed plates that reduce visibility. Other issues arise from intentionally altered, forged, or homemade plates, as well as special types like diplomatic or temporary plates with unique designs. Handling such variability requires adaptive recognition models and preprocessing techniques to maintain accuracy in diverse real-world conditions.

Effective handling of these plates requires a robust system capable of accurate detection and OCR. Deep learning models can learn from diverse examples; training datasets that include damaged or dirty plates improve system resilience. In cases of severe damage or manipulation, partial recognition may be possible, or the plate may be flagged as "unreadable" or "suspicious," necessitating manual review or cross-referencing with vehicle data from VAMR. Some systems incorporate modules to detect signs of tampering.

Multilingual license plates add complexity to ALPR systems, as they may contain characters from different alphabets. While most European plates use the Latin script, some vehicles from the Balkans or Eastern Europe feature Cyrillic characters, requiring broader recognition capabilities. OCR models like CRNNs

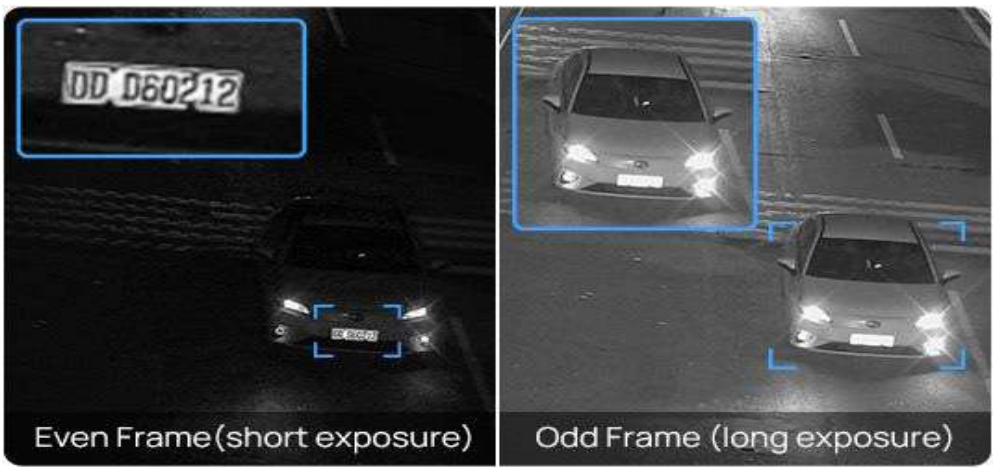
can handle such cases if three conditions are met: the character set includes all relevant symbols, there is adequate training data for each alphabet, and the system effectively manages visually similar characters across scripts. A "universal" OCR model may cover all expected characters, or specialized OCR models can be applied for specific alphabets based on preliminary country identification. Modern systems use Unicode to uniquely represent all characters. This study focuses primarily on Latin letters and Arabic numerals while acknowledging broader multilingual challenges.

Environmental and Operational Challenges for ALPR

The performance of ALPR systems is heavily influenced by factors that degrade image quality or obscure license plates. Image noise is a common issue, arising from camera sensors (especially in low-light conditions), video compression, or adverse weather like rain and fog. High noise levels complicate both plate localization and character recognition. Techniques such as Gaussian or median filtering help reduce noise while preserving essential details, and training models on noisy images improves robustness.

Visual obstructions (occlusion) further challenge accuracy, as plates are often partially hidden. Common sources include vehicle parts (e.g., trailer hitches, misaligned decorative frames), external objects (nearby vehicles, pedestrians, bicycles, vegetation), and dirt or debris (mud, snow, leaves). Lighting conditions, such as glare from sunlight or deep shadows, can also distort visibility, making reliable recognition difficult. Addressing these challenges requires both preprocessing enhancements and robust, well-trained detection and OCR models.

FIGURE 3: Comparison between short-exposure and long-exposure frames for improved nighttime license plate detection.

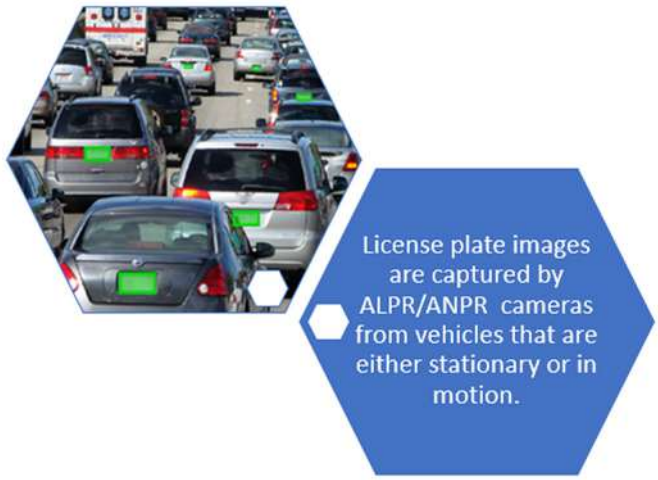


System capability to handle occlusion depends on the extent of coverage and algorithm sophistication. Modern detectors like YOLO can often locate a partially obstructed plate if distinctive features remain visible. When critical characters are occluded, full recognition may not be possible; however, some systems can perform partial reading or use contextual information to infer missing characters.

Nighttime conditions pose significant challenges for ALPR systems. Low ambient light reduces contrast and amplifies noise, while vehicle headlights and reflective plate surfaces can cause glare or overexposure. Motion blur from longer exposures further complicates character recognition.

To address these issues, infrared (IR) imaging or adaptive exposure techniques are employed. IR illumination enhances the contrast between characters and the plate background, improving detection and OCR accuracy. Successful implementation requires careful calibration of IR intensity and shutter synchronization. Additionally, including low-light and night-time images in training datasets helps models generalize better to these challenging scenarios.

FIGURE 4: License plate captures from stationary and moving vehicles using ALPR/ANPR technologies in real traffic environments.



Modern traffic surveillance combines multiple AI capabilities to enhance vehicle monitoring and identification. ALPR solutions like Open ALPR, Sighthound, Plate Recognizer, and DL-based YOLO+CRNN libraries vary in accuracy, speed, coverage, and ease of integration, providing options from open-source flexibility to high-accuracy commercial systems. When plates are unreadable, Vehicle Re-Identification (Re-ID) complements ALPR by matching vehicles across cameras using CNN-based feature embeddings, handling challenges like changing viewpoints, lighting, and occlusions. Color recognition adds another layer of verification, often using HSV/HSL color spaces, SVMs, or CNNs, with a limited

set of color classes to ensure consistency under varying illumination. For deeper verification, Vehicle Make and Model Recognition (VMMR) classifies specific makes and models, overcoming intra-class variation and inter-class similarity through fine-tuned pre-trained CNNs enhanced with part localization, attention mechanisms, and multi-task learning, enabling simultaneous prediction of make, model, and color. Modern traffic surveillance increasingly leverages synthetic data to enhance AI performance. Tools like Unreal Engine, Unity, CARLA, and NVIDIA DRIVE Sim generate large, fully annotated datasets with controlled variations, while techniques like domain randomization and adaptation bridge the gap between synthetic and real images.

For real-time monitoring and event response, integrated ALPR/VAMR systems connect with databases from law enforcement, vehicle registration authorities, courts, and insurance companies. This enables immediate detection of unregistered or uninsured vehicles, cross-checking against Interpol lists, and linking vehicles to their violation history.

A robust system relies on secure and scalable infrastructure, including standardized RESTful APIs, encrypted communications (WebSocket, TLS 1.3, HTTPS), OAuth2 with MFA, and full audit logs. Cloud-native or hybrid platforms allow dynamic scaling during peak traffic hours. Advanced ALPR models, such as YOLOv5 combined with CRNN, trained on Albanian-specific datasets, achieve recognition accuracy above 95%, even under low-light conditions, ensuring reliable real-time performance.

Best practices worldwide

Successful national implementations exist in South Korea, Estonia, and the UK. London's Smart Surveillance reduced vehicle-related crime by 30% .Estonia's X-Road centralized violation management, cutting penalty issuance from 48 hours to five minutes.

FIGURE 5: Improvement in plate-recognition accuracy using a 60 fps @1080p camera compared to a regular imaging sensor.

For Albania, developing an effective AI-based traffic surveillance system requires both regulatory support and technological planning. Establishing legislation that allows ALPR-generated evidence to be admissible in court is a critical first step, ensuring that automated detections have legal validity. Simultaneously, creating an integrated road safety platform that connects the DPSHTRR, police, courts, and insurance companies would enable real-time data sharing and coordinated responses to traffic violations.

From a technological perspective, edge computing should be leveraged to enable faster, localized vehicle detection and reduce latency, particularly in high-traffic areas. Where authentic Albanian datasets are limited, the use of synthetic data can help train AI models, providing diverse and annotated images to bootstrap system performance. Together, these measures would lay a strong foundation for a robust, accurate, and legally supported traffic surveillance infrastructure in Albania.

Real-World ALPR/VAMR Pipeline Architecture

Designing an effective automated traffic surveillance system involves creating a pipeline architecture that integrates multiple components into a coherent workflow. The process begins with video stream acquisition, capturing footage from multiple cameras via standard protocols like RTSP. These streams are then decoded into individual frames and preprocessed—such as resizing or normalization—to prepare the data for AI models.

Next, vehicle and license plate detection localizes relevant objects in each frame, typically using models like YOLO. Detected vehicles are assigned unique IDs and followed across frames through multi-object tracking algorithms such as DeepSORT, ensuring temporal consistency. From these detections, Regions of Interest (ROIs) are cropped for specialized analysis.

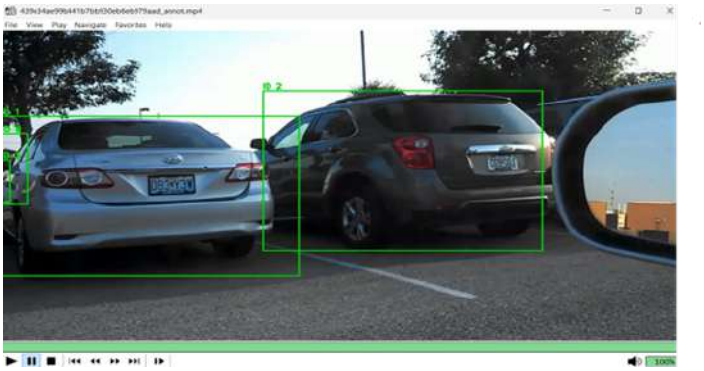
Automatic License Plate Recognition (ALPR) processes cropped plate images using OCR models (e.g., CRNN with CTC loss) to extract alphanumeric strings, while Vehicle Attribute Recognition (VAMR) analyzes cropped vehicle images to determine attributes such as color, make, and model using CNN-based architectures. The outputs from ALPR and VAMR are then aggregated and verified, optionally cross-checked against databases for consistency.

For enhanced functionality, the pipeline can include violation detection, identifying stolen or wanted vehicles, speeding (via radar/LiDAR integration), traffic signal infractions, or illegal parking. The system then generates evidence packages containing images, videos, plate information, vehicle attributes, timestamps, and location data, sending real-time alerts when necessary. Finally, all data and metadata are securely stored in an organized fashion, supporting auditing, analysis, and long-term monitoring.

This end-to-end pipeline ensures accuracy, efficiency, and reliability, providing a robust foundation for modern, AI-driven traffic surveillance systems.

FIGURE 6: Example of input video frame (top) and the corresponding detection and tracking output produced by the implemented ALPR pipeline (bottom).

Photo from the program executed by the code



Cost Analysis for Automated Traffic Monitoring Systems

Analyzing the costs and operational considerations of automated traffic surveillance systems requires evaluating hardware, maintenance, energy consumption, and integration capabilities.

Edge computing offers low-latency, real-time processing through devices like Google Coral Edge TPU or NVIDIA Jetson Orin NX, but deploying multiple monitoring points involves significant initial investment. Operational costs include energy consumption, regular maintenance, software updates, and hardware replacement. By processing data locally, edge devices reduce bandwidth usage and network congestion, sending only summarized outputs such as violation events, timestamps, and locations. Environmentally, edge computing can leverage

renewable energy sources like solar power—especially practical in sunny regions such as Albania—reducing reliance on energy-intensive cloud data centers.

Case studies demonstrate diverse implementations: Dubai uses AI-based ALPR for speeding and traffic safety monitoring; Singapore integrates ALPR into intelligent transport systems for traffic optimization and electronic road pricing; London applies ALPR extensively for congestion and low-emission zones, though high maintenance costs and privacy concerns remain challenges.

Integration with national databases and law enforcement systems is essential for maximizing system effectiveness. Linking ALPR outputs with the Vehicle Registration Database (VRD), such as Albania's DPSHTRR, allows verification of ownership, insurance status, technical inspections, and vehicle attributes, facilitating automated fine issuance and detection of cloned plates. Connecting to police databases enables alerts for stolen or wanted vehicles, historical violations, and links to wanted persons, supporting rapid law enforcement intervention.

Key challenges in integration include the availability and standardization of APIs, data security and access control, performance and scalability, data quality, and establishing a legal framework for data sharing. Addressing these factors ensures that ALPR/VAMR systems operate efficiently, securely, and reliably while supporting law enforcement and traffic management objectives.

Data Storage and Security

Effective automated traffic monitoring systems require comprehensive data management and security strategies. Data retention policies should define how long information is stored, ensuring that unnecessary data is securely deleted or anonymized in compliance with legal requirements. Access control mechanisms—authentication, authorization, and audit log—restrict data access to authorized personnel and track activity. Encryption protects sensitive data both in transit and at rest, while physical and virtual infrastructure security safeguards hardware and cloud environments. Techniques like anonymization and pseudonymization reduce privacy risks when data is used for analysis or model training. Predefined incident response plans ensure rapid mitigation of breaches, and ongoing personnel training reinforces secure handling practices.

Automated Violation Decision Support Systems (AVDSS)

AVDSS use advanced algorithms, such as Random Forests, XGBoost, and neural networks—to detect, classify, and respond to traffic violations, often achieving over 95% accuracy. Components include violation detection, categorization by severity, and automated decision-making (issuing fines, alerting authorities, storing evidence). Visualization tools like GIS maps and temporal graphs support urban

planning, while machine learning techniques (supervised and unsupervised) predict risky behaviors and detect patterns. CNNs can visually identify dangerous driving, and continuous learning improves system performance over time. Key risks involve algorithmic bias, misclassifications, and overreliance on automation, highlighting the need for transparency, auditability, and human oversight.

Conclusion

AI-powered AVDSS enhances road safety, operational efficiency, and urban planning by accurately detecting and analyzing traffic violations. While challenges such as bias, complex data, and ethical concerns persist, continuous learning and human supervision ensure fairness and reliability. These systems not only enforce traffic laws but also contribute to smarter, safer cities.

Key Findings and Recommendations

- Transformative Potential of AI: Deep learning and computer vision models (YOLO, CRNN, CNNs for VAMR) provide high accuracy and speed for automated traffic monitoring.
- Integrated ALPR+VAMR Architecture: Combining license plate and vehicle attribute recognition improves reliability, especially when plates are unreadable.
- Real-World Challenges: Lighting, weather, occlusions, high speeds, and diverse plate formats require robust datasets, augmentation, and advanced models.
- Critical Role of Data: High-quality, diverse, locally annotated datasets are essential; their scarcity is a major barrier.
- Edge vs. Cloud Trade-offs: Edge computing reduces latency and bandwidth usage, while cloud systems enable centralized management and deeper analysis.
- Ethical and Legal Considerations: Privacy, accountability, and transparency are crucial; compliance with legal frameworks (e.g., GDPR) and ethical AI principles is mandatory.

Best Practices

- Modular and Integrated Approach: Building the system as interconnected modules (detection, tracking, ALPR, VAMR) while maintaining a coherent pipeline for optimal performance.

- Advanced Models and Fine-Tuning: Leveraging pretrained models (YOLO, ResNet, CRNN) and fine-tuning them on task-specific, local datasets.
- Focus on Robustness: Training models to withstand variations in lighting, weather, occlusion, and vehicle/plate types using data augmentation and diverse datasets.
- Continuous Evaluation: Using appropriate performance metrics, benchmark datasets, and real-world testing to identify and address weaknesses.
- Hybrid Edge-Cloud Architecture: Edge processing for time-critical tasks and cloud for deep analysis and centralized storage.
- Privacy and Ethics by Design: Integrating ethical and privacy considerations early in system design.
- Interinstitutional Collaboration: Coordinating among police, transport authorities, and data protection agencies.
- Stakeholder Engagement: Involving the public, civil society, and domain experts to ensure acceptability and trust.

Recommendations for Albania

- Start with Limited Pilot Projects: Test technology in selected high-risk areas to collect context-specific data and assess effectiveness.
- Invest in Local Data Collection and Annotation: Develop large, diverse datasets of traffic images from Albanian roads.
- Adopt Open and Flexible Architectures: Use open-source, well-tested models that can be fine-tuned for local conditions.
- Prioritize Low Latency for Critical Applications: Use edge processing for rapid response tasks such as stolen vehicle detection.
- Develop Clear Legal and Ethical Frameworks: Update legislation to regulate AI-based traffic monitoring in compliance with GDPR and human rights standards.
- Strengthen Technical and Human Capacity: Train technical staff and law enforcement personnel for correct and ethical system use.

Future Research Directions

- Enhancing Robustness in Extreme Conditions: Develop ALPR/VAMR models resilient to snow, low light, or severe occlusions, possibly using multimodal sensors (visual, radar, Lidar).
- Few-Shot and Self-Supervised Learning: Enable recognition of new plate formats or vehicle models with minimal labeled data.
- Continual Domain Adaptation: Create models that adapt dynamically to environmental changes without full retraining.

- Explainable AI (XAI) for Traffic Monitoring: Improve model transparency and interpretability to increase trust and facilitate error analysis.
- Advanced License Plate Forgery Detection: Detect subtle modifications or anti-ALPR materials.
- Vehicle Behavior Analysis: Use AI to predict risks and detect unsafe driving patterns beyond plate and attribute recognition.
- Optimization for Resource-Constrained Edge Devices: Develop energy-efficient and low-cost deep learning solutions for edge deployment.
- Socio-Ethical Studies: Conduct empirical research on public perception, societal impacts, and context-specific regulatory frameworks.

References

Anagnostopoulos, C. N. E., Anagnostopoulos, I. E., Psoroulas, I. D., Loumos, V., & Kayafas, E. (2008). License plate recognition from still images and video sequences: A survey. *IEEE Transactions on Intelligent Transportation Systems, 9*(3), 377–391. https://doi.org/10.1109/TITS.2008.922938

Barocas, S., Hardt, M., & Narayanan, A. (2019). Fairness and machine learning. MIT Press. https://fairmlbook.org/

Baek, J., Lee, B., Han, D., Yun, S., & Lee, H. (2019). Character region awareness for text detection.

Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 12275–12283). Retrieved from https://openaccess.thecvf.com/content_CVPR_2019/html/Baek_Character_Region_Awareness_for_Text_Detection_CVPR_2019_paper.html

Buolamwini, J., & Gebru, T. (2018). Gender shades: Intersectional accuracy disparities in commercial gender classification.

Proceedings of the Conference on Fairness, Accountability and Transparency (pp. 77–91). Retrieved from https://dl.acm.org/doi/10.1145/3287560.3287565

Chen, L., et al. (2021). A survey on intelligent transportation systems. *IEEE Transactions on Intelligent Transportation Systems, 22*(2), 618–637. https://doi.org/10.1109/TITS.2020.2988061

European Transport Safety Council. (2022, June 15). The 2022 ETSC Road Safety Performance Index Conference. https://etsc.eu/the-2022-etsc-road-safety-performance-index-conference/

European Union Agency for Fundamental Rights. (2021). *Getting the future right: Artificial intelligence and fundamental rights*. Publications Office of the European Union. https://fra.europa.eu

Goh, P. H., et al. (2022). Intelligent transportation systems: A review of technologies and applications. *Journal of Traffic and Transportation Engineering (English Edition), 9*(1), 1–25. Retrieved from https://www.sciencedirect.com/science/article/pii/S####

Instituti i Statistikave (INSTAT). (2023). *Statistika mbi sigurinë rrugore*. Tiranë, Shqipëri. Retrieved from: https://www.instat.gov.al/media/14132/statistika-mbi-sigurine-rrugore-2023.pdf

Instituti i Statistikave (INSTAT). (n.d.). *Transporti, aksidentet dhe karakteristikat e mjetet rrugore*. Retrieved from: https://www.instat.gov.al/sq/temat/industria-tregtia-dhe-sherbimet/transporti-aksidentet-dhe-karakteristikat-e-mjetet-rrugore/

Kornblith, S., Shlens, J., & Le, Q. V. (2019). Do better ImageNet models transfer better? Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2661–2671. https://doi.org/10.1109/CVPR.2019.00277.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436–444. https://doi.org/10.1038/nature14539.

Li, H., et al. (2019). Vehicle recognition using multi-task learning for joint make, model, and color classification. *IEEE Transactions on Intelligent Transportation Systems, 20*(3), 1019–1028. https://doi.org/10.1109/TITS.2018.2873130

Park, S. H., et al. (2020). AI-based real-time traffic violation detection system using CCTV images. *Electronics, 9*(11), 1878. https://doi.org/10.3390/electronics9111878.

Redmon, J., & Farhadi, A. (2018). *YOLOv3: An incremental improvement* [Preprint]. arXiv. https://arxiv.org/abs/1804.02767.

Satyanarayanan, M. (2017). The emergence of edge computing. Computer, 50(1), 30–39. https://doi.org/10.1109/MC.2017.9.

Selmi, Z., Halima, M. B., & Alimi, A. M. (2017). Deep learning system for automatic license plate detection and recognition. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)* (Vol. 1, pp. 1140–1145). Retrieved from https://ieeexplore.ieee.org/document/8433936

Shi, B., Bai, X., & Yao, C. (2016). An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 39*(11), 2298–2304. https://doi.org/10.1109/TPAMI.2016.2515971

Shi, W., et al. (2016). Edge computing: Vision and challenges. *IEEE Internet of Things Journal, 3*(5), 637–646. https://doi.org/10.1109/JIOT.2016.2579198

Tehrani, M. H., et al. (2019). A review on vision-based license plate recognition systems. *IEEE Transactions on Intelligent Transportation Systems, 20*(8), 2881–2901. https://doi.org/10.1109/TITS.2018.2875015.

Wojke, N., Bewley, A., & Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. In 2017 IEEE International Conference on Image Processing (ICIP) (pp. 3645–3649). IEEE. https://doi.org/10.1109/ICIP.2017.8296962

World Health Organization. (2023, May 15). Transport systems need to be made safe, healthy and sustainable. Retrieved from: https://www.who.int/news/item/15-05-2023-transport-systems-need-to-be-made-safe--healthy-and-sustainable

World Health Organization. (2023). *Global status report on road safety 2023*.

Retrieved from https://www.who.int/publications/i/item/9789240086517

Yang, L., Luo, P., Change Loy, C., & Tang, X. (2015). *A large-scale car dataset for fine-grained categorization and verification*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 3973–3981). https://doi.org/10.1109/CVPR.2015.7299023

Zhang, G., Wang, Y., Wei, H., & Chen, Y. (2017). A comprehensive review of radar and LiDAR sensors for traffic enforcement and vehicle detection. *IEEE Transactions on Intelligent Transportation Systems, 18*(10), 2848–2863. https://doi.org/10.1109/TITS.2017.2680468

Solar driven fan unit for a solar dryer

Xhevahir BADUNI[1]

Abstract

Drying cocoa is an important step in cocoa processing not only for preservative purposes but also for improvement of flavour and quality of cocoa products. Results showed that in clear sunny conditions, temperatures in the solar drier can obtain around 60°C. Temperatures in the solar drier are always higher than ambient about 10 to 15°C even at night. In the sunny season, cocoa beans can be dried only in 4 to 5 days and 6 to 7 days in the rainy season. When cocoa beans were effectively dried, they also helped to avoid over-fermentation and reduced mould contamination.

Upon fermentation, cocoa beans need to be dried to remove the moisture, to reduce bitterness and astringency and to develop a chocolate brown color. Direct sun drying has been used widely in many cocoa producing countries and may last as long as 2 weeks. Artificial drying can reduce this process to 20 hours. In a dryer hot air is produced by kerosene, biomass fuel fired stoves or solar air heaters.

[1] Xhevahir Baduni, MSc is a Mechanical Engineer with around 10 years of experience in HVAC systems, hydro-thermo-sanitary installations, renewable energy, hydropower plant design and maintenance, and energy auditing. Has held leadership and technical roles in both public and private sectors and collaborated with various engineering studios, companies and professionals. Hold's licenses in technical design, energy auditing, and H_{PP} maintenance and has completed specialized training in Albania, Germany and Turkey. Since October 2025, he has been a lecturer at the European University of Tirana, contributing to the Mechanical Engineering program.
badunixhevahir@yahoo.com/ xhevahir.baduni@uet.edu.al
Supervisor: Thomas CAROLUS. Department Maschinenbau;Institut für Fluid- und Thermodynamik; Lehrstuhl für Strömungsmaschinen, University of Siegen, Germany
Team involved in the project: S. Azizi, Xh. Baduni, M. Gipperich, A. Heupel, Q. Jiaokun, Xh. Muca, R. Stasa, E. Topi, M. Weber

Further improvement can be achieved by generating forced air flow by means of a fan. In a preliminary specification of a solar driven fan unit the electric power consumption of the fan is fixed to 300 W peak.

Keywords: *Drying Cocoa, Photovoltaic Solar Panel, Solar Driven Fan Unit, Modeling and Simulation of Solar Driven Unit.*

1.Introduction

West Africa is the main cocoa producing area with about 72% of the world cocoa production. Four major West Africa countries are Côte d'Ivoire, Ghana, Nigeria, and Cameroon. Pacific Asia accounts for about 15% and America is 13% of the 3.5 million tons cocoa beans in over the world (Global Cocoa Market Study, 2021). According to the ICCO, the world will be lack about 102,000 tons cocoa bean in 2010 - 2011 because of the decreasing of cocoa bean in 2008 - 2009 crops. There are some reasons that lead to the decrease of cocoa production are insecurity politics, natural calamity, old and stunted cocoa trees, lack of land in some main cocoa producing countries. In nine decades, global cocoa production has increased steadily and consistently to keep with the ever-increasing needs for cocoa bean. Cocoa consumption has increased on average by 3.5% per annum over recent years and is projected to increase by 1.5 - 3.5% per annum over the 5 years coming (Knight, 1999).

In 1998, the Ministry of Agriculture and Rural Development carried out an investigation about cocoa production and set a new goal of having 100,000 ha of planted cocoa by 2010. Vietnamese Cocoa Development team was established in March 2005. The aims are promoting the development of cocoa cultivation and recognizing the cocoa as a new long-time industrial tree in Vietnam. By the end of year 2006, the total cocoa has been inter-plant on plantations about 7,300 ha with some major provinces are Ben Tre, Tien Giang, Ba Ria-Vung Tau, Dak Lak, and Binh Phuoc (Hoa, 2007).

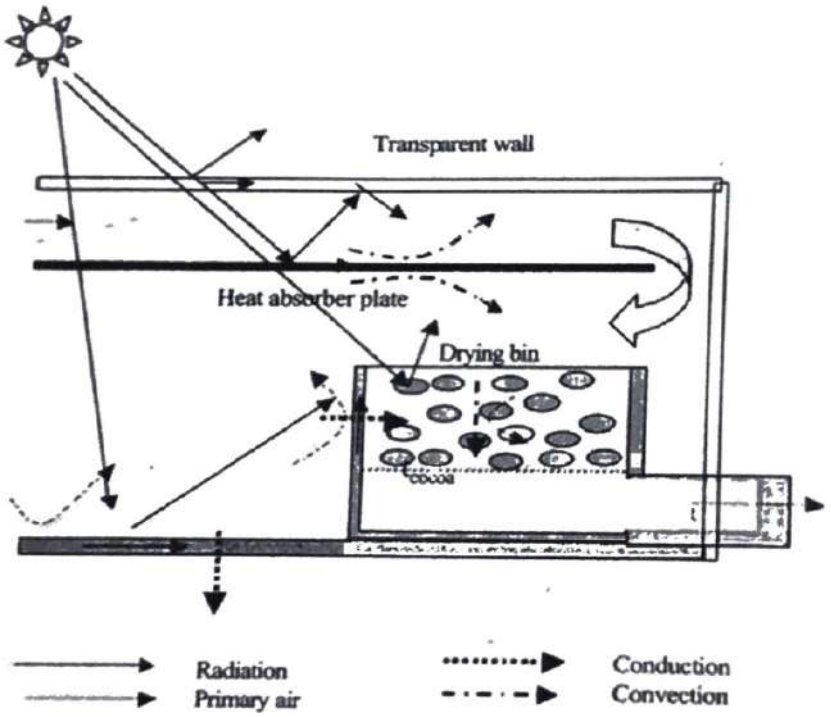


Figure 1: Concept sketch of a solar dryer.

Drying cocoa is an important step in cocoa processing not only for preservative purposes but also for improvement of flavour and quality of cocoa products. In many countries, including parts of Vietnam, sun drying is the main method to dry cocoa beans. This is a very simple method and the most effective way to dry cocoa beans. This reduces acids in the cocoa beans and enriches flavor in cocoa products. However, in the unfavorable condition of weather, especially in the rainy season, the drying may take longer which could cause over-fermentation and mould contamination. Then, the cocoa could have some off-flavours and lead to down-grading.

Three selected patents relevant for SBATS

1.1.1 Axial flow fan and fan orifice US 5273400 A

The present invention is an axial flow fan capable of use in a variety of applications including moving air in heating, ventilation and air conditioning systems and equipment. It produces reduced levels of radiated noise and requires lower input power to move the same amount of air as compared to prior art fans.

The fan has a plurality of identical blades. Each blade is strongly swept in one direction at its root and strongly swept in the other direction at its tip. This combination of blade sweeps allows for a large amount of sweep at the blade tip while producing low stress in the blade at its root. A large sweep in the tip region of the blade results in low turbulent noise coherence in that region.

The coherence is low because only a relatively small portion of the blade tip region is subjected to inlet flow turbulence at any given instant. The noise produced by inlet turbulence is thus diffused and reduced. Along the entire span of the blade, the maximum camber, expressed as the deviation of the blade camber line from the chord line, of the blade should be closer to the leading edge of the blade. This configuration promotes attached flow in the region of the trailing edge and thus reduces form drag and trailing edge noise. The number of blades on a fan constructed according to the present invention is not critical to fan efficiency, noise and overall performance. The fewer the number of blades, however, the greater the pitch that will be required in order for the fan to produce a given capacity at a given rotational speed. Fewer blades would also require increased mid chord skew angles and larger blade chord lengths to achieve the desired blade solidity (that is, the proportion of the total area of the swept disk of the fan that is covered by blades). The fan and orifice of the present invention may be manufactured out of any suitable material by any suitable process. It is however, particularly suited, assuming no blade overlap, to be produced in a suitable plastic by a suitable molding process.

Figure 2: Front and side elevation view of one embodiment of the fan.

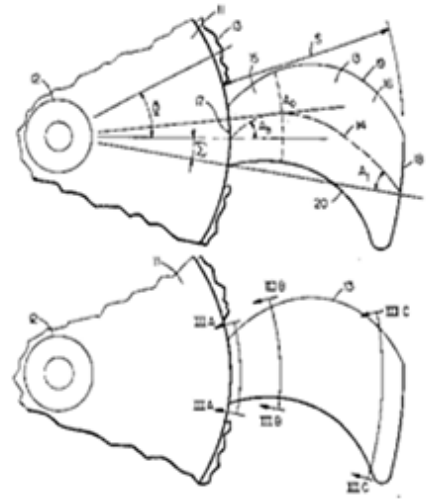


Figure 3: Front elevation views showing a portion of the hub and one blade of one embodiment.

1.1.2 High efficiency axial fan EP 1797334 B1

An axial fan rotating in a plane (XY) about axis comprises a central hub, a plurality of blades, which have a root and a tip, the blades being delimited by a concave leading edge, whose projection in the fan plane of rotation (XY) is defined by two circular arc segments, and a convex trailing edge, whose projection in the fan plane of rotation is defined by one circular arc segment.

The blades are made from sections with aerodynamic profiles relatively extended Ing in the direction of their center line, providing a good flow rate and air pressure relative to the overall dimensions of the fan. Fans of this type must satisfy various requirements, including low noise level, high efficiency, compactness, capacity to achieve good pressure and flow rate values.

This patent presents a fan with blades delimited at the leading edge and trailing edge by two curves which are two circular arcs when projected in the fan plane of rotation. Fans constructed in accordance with this patent provide good efficiency and low noise but have limits as regards the possibility of achieving high pressure values, since the blades are made with profiles whose center line is relatively short compared with the blade radial extension. Moreover, fans constructed in accordance with the above-mentioned patent have a limited axial dimension, but a relatively large diameter.

For the exchange units of heating or air conditioning systems for the interior of motor vehicles the overall dimensions of the fan must be limited, which means that the diameter must also be limited, whilst good air flow rates are required with high pressure and low noise. For these reasons, in the above-mentioned ex changer unit's centrifugal fans are often used, which may have a relatively small diameter, but with a rather large axial dimension.

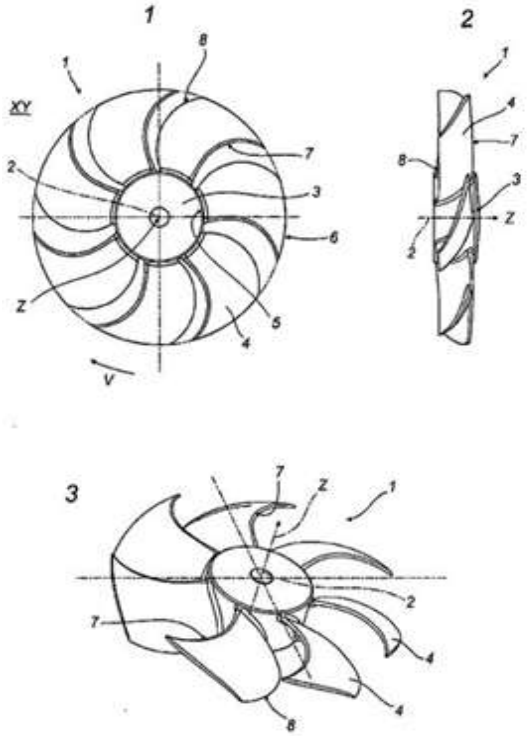


Figure 4: In front, side and perspective and view of the fan.

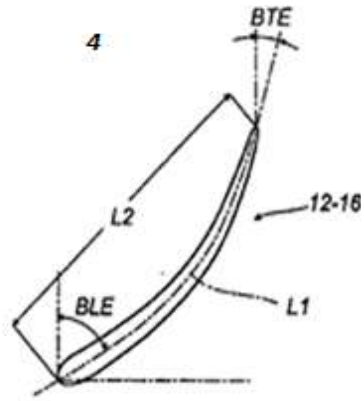


Figure 5: Cross-section of a profile the respective geometric characteristics.

1.1.3 Food dryer DE 102011111971 A1

By incident light through a transparent sled wall, an airflow is heated to dry groceries. Conditioned by forced convection which is generated by an aerator this is conducted over groceries. The heated air detracts the moisture from the groceries, and it cools itself through the absorption of the moisture. Subsequently the air is dried by an air dehumidifier and feed again in the dry air circle. An invention to desicate and therewith conserve groceries. It needs to be autarkic, effective, convenient and it should have a great drying performance and thereby have a little drying time. The sunlight is used to heat the air, the gathering of the moisture to cool the air and after the dehumidifying component has dried the air, the heated air is added to the circulation again. The drying component consists of convenient and clay bonded material and is still sufficient for its duty because of its capillary tubes in the material and the hereby possible water vapour transport. Once the component is completely filled with water, there will be an automatically transport of the moisture from the dehumidification room to the atmosphere by a capillary line.

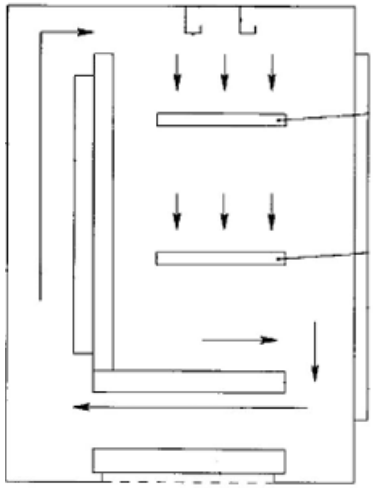


Figure 6: Food Dryer Drawing.

2. Design - Methodology – Result - Modeling and Simulation

Photovoltaic solar power plant

The first step to design a photovoltaic solar panel is to consider the location of the solar dryer. The main producers of cocoa beans regarding the ICCO Quarterly Bulletin of Cocoa Statis tics (Vol. XLIII, No. 1, Cocoa year 2016/17, Table 1) are in West Africa. The Cote d'Ivoire and Ghana are producing 60% of the world's cocoa beans, which leads to the idea to locate the solar dryer in this area and to use the irradiation data from West Africa.

Production of cocoa beans (thousand tons)

			Estimates		*Forecasts*	
	2014/15		*2015/16*		*2016/17*	
Africa	**3074**	**72.30%**	**2911**	**73.40%**	**3365**	**73.90%**
Cameroon	232		211		250	
Côte d'Ivoire	1796		1581		1900	
Ghana	740		778		850	
Nigeria	195		200		230	
Others	110		141		135	

Table 1: Production of cocoa beans related to geographic zones.

The average global horizontal irradiation at the CIV is about 1900 kWh/m² and approximately 2000 kWh/m² in the Western Africa zone (Figure 7).
The Electric Peak power consumption of the fan is fixed to 300W. That leads to an annual energy sum of 2628 kWh if a constant operation is assumed.

$$P_{peak} * 8760 \frac{h}{year} = 2628 \frac{kWh}{year}$$

Regarding the task an operation time of 20 hours is needed to dry one batch of cocoa beans. It is estimated that the remaining 4 hours of a full day are needed for charging, discharging and cleaning. The annual energy sum is reduced through this estimation:

$$P_{peak} * 7300 \frac{h}{year} = 2628 \frac{kWh}{year}$$

Average Temperatures above 30 °C and up to 10 hours of solar radiation per day (more than 6 in average) request robust solar panels.

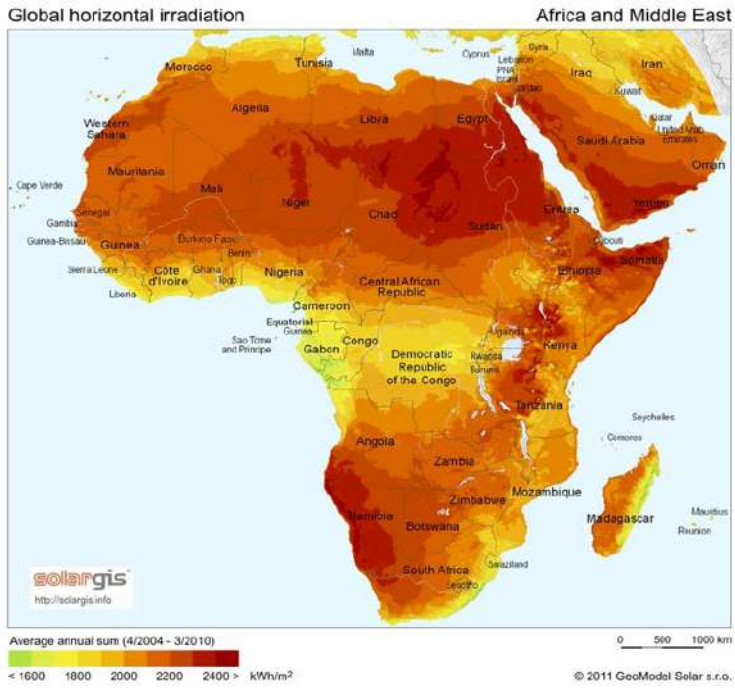


Figure 7: Global horizontal Irradiation in Africa according to SOLARGIS

The highest yield is obtained when the irradiation is rectangular to the solar panel. Because the CIV and western Africa are close to the equator one can assume the highest yield of the solar panel is obtained at a small angle of inclination. An angle of inclination of 0° would not be beneficial for self-cleaning effects through rain. An angle of 5° and a south-orientation is chosen (The CIV is a little northern from the equator).

The next step is the choice of a solar module to calculate the total area. To calculate efficiency and losses a Performance-Ratio of 75% is chosen regarding to a usual performance. The performance ratio of a photovoltaic system is the quotient of alternating current (AC) yield and the nominal yield of the generator's direct current (DC). It indicates which portion of the generated current can actually be used. The PR leads to the following nominal Power:

$$P_{nominal} = \frac{2190\frac{kWh}{year}}{0.75} = 2920\ kWh$$

The following Solar panel from Solar World is selected due to robustness at high temperatures and harsh environment: Sun module® Plus SW300 Mono

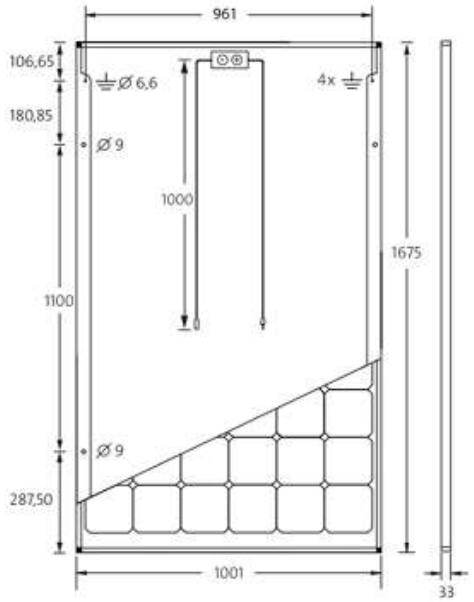


Figure 8: Drawing of the Solar panel from Solar World

The efficiency regarding the Datasheet (appendix) is 17,89 % and the maximum operating temperature is 85°C. The required area A total is calculated below:

$$A_{total} = \frac{P_{nominal}}{\eta_{Panel} * irradiation} = \frac{2190 \frac{kWh}{year}}{0.1789 * 2000 \frac{kWh}{m^2 a}} = 8.16\ m^2$$

The amount of solar panels N is calculated as follows:

$$N = \frac{A_{total}}{A_{panel}} = \frac{8.16\ m^2}{1.001\ m * 1.675\ m} = 4.867$$

The amount is rounded to 5 Solar panels. Regarding a total power of 300W per cell, a total power of 1500W is to be expected.
The Solar panels deliver a current at maximum Power of I_{mpp}= 9,31 A and a voltage of U_{mpp} = 32,6 V. A usual fan of 300W drive power has a usual nominal voltage of 200-240 V and a current consumption of 0,5-1,5 A at 50-60 Hz (Referring to EBM Papst axial fans, [14]). By means of a power inverter this gap can be overcome. A power inverter from Kostal (Piko 1.5 MP) for 1 Phase is selected. Wiring is not selected due to missing information regarding the construction site.

While the solar dryer will operate during the day and overnight as well, a battery and needs to be selected to ensure a constant energy supply. A Kostal Piko 6.0 BA (appendix) with 6 $LiFePO_4$ battery cells are selected and delivers 7,2 kWh of Power. While consuming 300Wh of electrical power per hour the battery is able to keep the system alive for at least 24h.

Fan design using dAX
The main goal of the project is to design an axial-fan. With the given parameters of Δp, V̇ and the rotational speed n = 15001 min / which is fixed through the frequency of the rectifier, the position in the Cordier-diagram can be calculated with the dimensionless numbers cf. For an adequate axial fan, the best diameters (Hub and Tip) must be chosen. In this case we assumed the tip diameter d_{tip} = 0.2m and the hub diameter d_{hub} = 0.4m to fulfill the sign point. The design point of the fan is in the axial area, and an appropriate safety factor is ensured.

For the project the design was calculated using software from the "Lehrstuhl für Strömungstechnik und Strömungsmaschinen des Instituts für Fluid-und Ther modynamik" of the University of Siegen called dAX. The program computes the fan with the parameters above in different steps. In the first step the required parameters that are calculated above (V̇, Δp_t, n, d_{tip} and d_{hub}) must be entered into the program.

The next step is the calculation of the position in the Cordier-diagram. If the design points suit the requirements the kinematics and efficiency are calculated. Especially the velocity triangles (Figure 10) are computed where u1 is the circumferential velocity of the rotor at the hub and u_2 the circumferencial velocity at the tip.

The meridional component of the absolute velocity at the inlet c1 and at the outlet c_2 is c_{m1} respectively c_{m2} and are calculated as follows:

$$c_{m1} = \frac{\dot{V}}{2\pi r_{Hub*}\, b_{Hub}} \qquad c_{m2} = \frac{\dot{V}}{2\pi r_{Tip*}\, b_{Tip}}$$

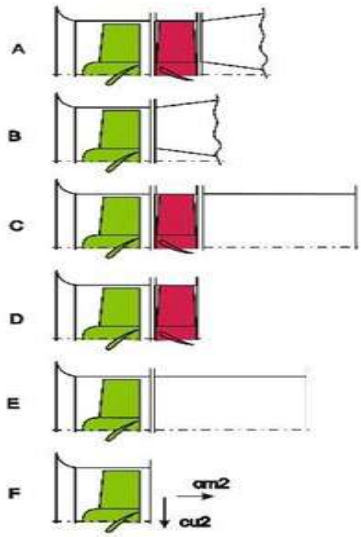
Where b is the width of the blades at the Hub res. the Tip. The tangential component of c_2 is calculated using the following formula:

$$c_{u2} = \frac{\Delta p_{tt,B}}{\rho u_2}$$

The connection between u and c is the relative velocities w. For the calculations a spin-free feed at the inlet and a blade congruent flow at the outlet is estimated regarding to.
Additionally, there are two preliminary design criteria that need to be fulfilled named the diffusion criterion after De Haller and the hub dead water or Strscheletzky criterion:

$$De\ Haller: \frac{w_{2Hub}}{w_{1Hub}} \leq g_{DH,}\ where\ g_{DH} \approx 0.55\ to\ 0.75$$
$$Strscheletzk: \frac{c_{m2Hub}}{c_{u2Hub}} \leq g_{ST,} where\ g_{ST} \approx 0.8\ to\ 1$$

These criteria are also used to define the optimal Hub and tip diameters.



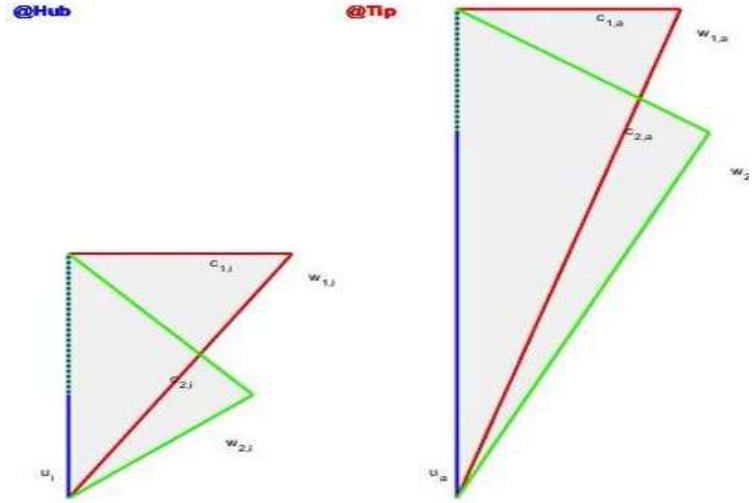


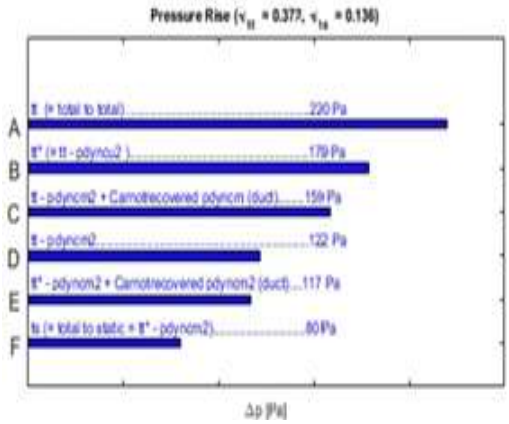
Figure 9: Various Fan periphery types.

Figure 10: Velocity triangles.

For more, the program generates some different model types of the fan and its periphery shown in Figure 9:

B: Fan with an endless ideal diffusor.
C: Fan with guide vans and diffusor.
D: Fan with guide vans.
E: Fan with diffusor.
F: Just the fan.

After comparing the efficiencies and values of the pressure rise from the different types to the needed values, the model type C became the type of choice because the pressure rise is high enough to reach the value of 144 Pa, which is calculated for two layers of cocoa-beans and the power is below the value of PW = 300W. Options A and B are based on the principle of an infinite diffuser which is not applicable in a real machine.



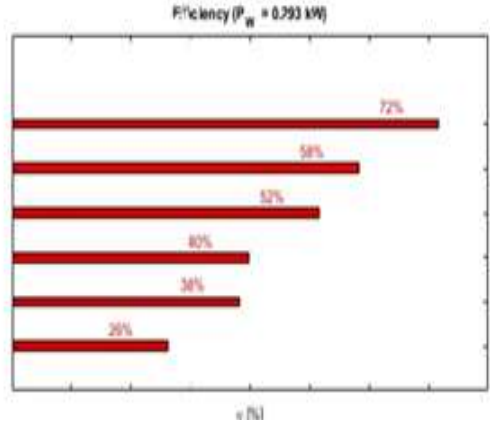


Figure 11: Pressure rise and efficiency of the different model types A-F

Now we must choose the blade geometry of the fan. Since the fan is operating at any time during the day, the temperatures, the air conditioning and the air speed outside the system change and affect the conditions inside the system. Because of that, we must design the fan robust based on its operating characteristics.

An Airfoil type which ensures similar flow characteristics from hub to tip needs to be chosen. The airfoil that suits these requirements is the FX 60-126 (SPK1 - mod.) and it has also a very low Lift to Drag coefficient (Figure 13).

Other airfoils like NACA 0010 may have higher lift coefficients, but they often have varying flow distributions. About the angle of attack α we could define the workspace. For security reasons we choose an angle α-hub of 6° and an angle α-tip of 3,5°. This ensures enough distance to the stall area at an angle of α = 10°.

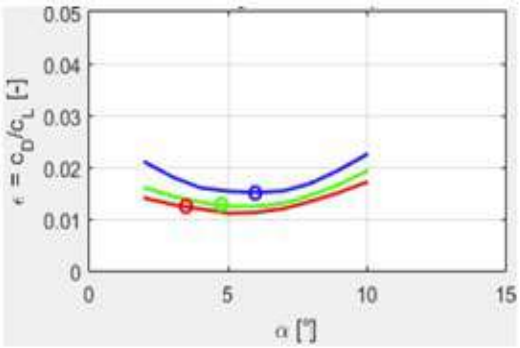


Figure 12: Lift/Drag Polars (SPK1-mod)

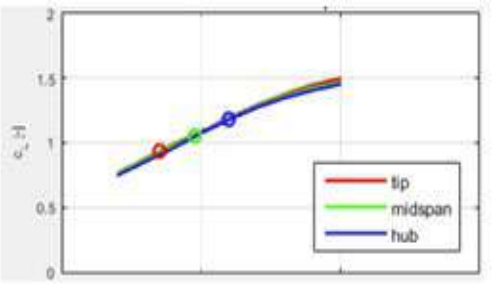
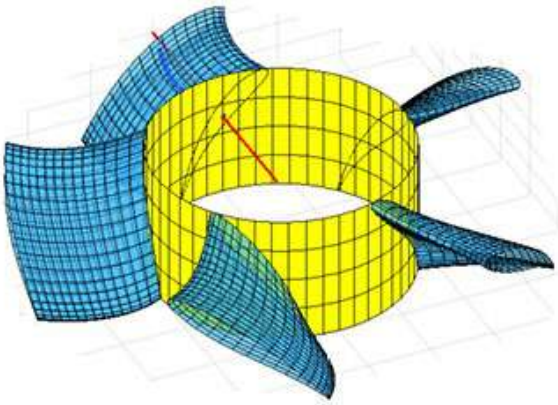


Figure 13: Lift coefficients of FX 60-126

In the next step the number of blades is estimated. Less than five blades will lead to an unfavorable Reynolds number domain. More than five blades will lead to a rise of the resistance coefficient C_D because the distance between the blades will be lower and friction will rise. Through this knowledge we choose five blades for our fan (Figure 14).



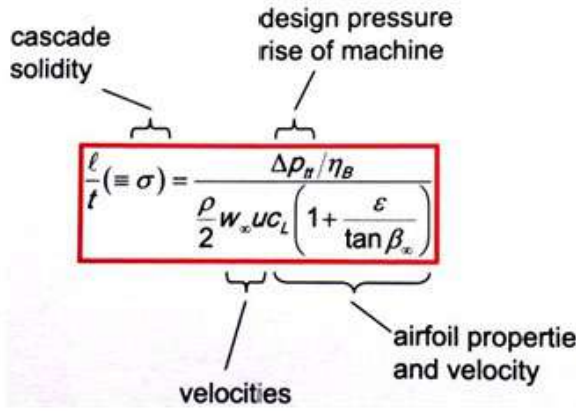


Figure 14: Fan geometry with 5 blades solidity

Figure 15: Key equation to calculate

After the number of blades are selected, the solidity is calculated using the equation in Figure 21. The chord length l as well as the stagger angle $\gamma = \beta_\infty+\alpha$ is calculated and the blade section is drawn.

Next, we had to design the guide vans. The number of blades is related to the acoustics of a fan. The same or a multiplicity number of blades of guide vans would result in an overlap withevery rotation of the fan and a pressure shock would be generated that would result in noise. The most suitable combination of fan and guide vans is one which has propagating modes as few and high as possible.
Figure 16 shows that there is more than one option for the number of blades in the guide vanes, for example, seven, nine, eleven, seventeen, and eighteen.

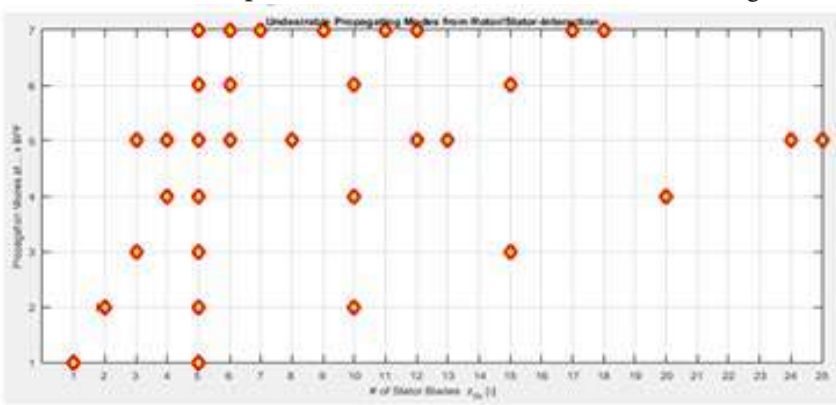


Figure 16: Undesirable Propagating Modes from Rotor/Stator- Interaction

Another criterion is the solidity according to the Blade-Element-Momentum (BEM) theory for low pressure axial fans. Only eleven blades reached the expected ratio of one and a camber angle below 65° (Figure 17).

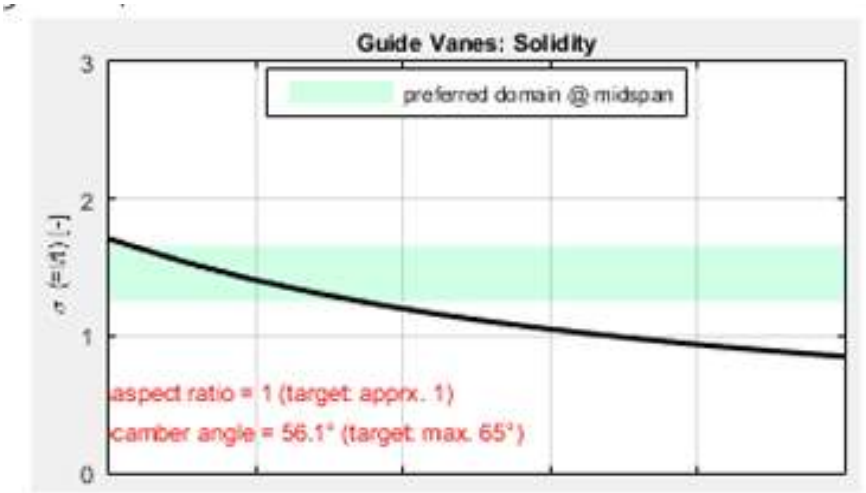


Figure 17: Solidity for eleven guide vans

To reduce the costs of our system, we choose guide vans without own Airfoils. Because our system is not a high-performance system and the added value of Airfoils is too small as it would justify the additional costs (Figure 18).

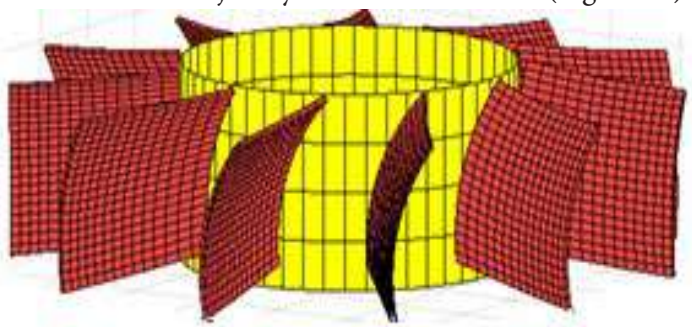


Figure 18: Guide vans with eleven blades

Three millimeters thickness for the blades is estimated to reduce flow resistance. With eleven blades of this thickness, it should be possible to create a welding connection between the guide vans and the channel.

Solar Dryer – Concept

Diffuser, inlet nozzle and blow out

A diffuser is in general a conical component of a channel that converts dynamic pressure to static pressure. To prevent shocks right after the guide vans, where the profile of the channel changes abruptly, a transition diffuser is chosen. The design calculations are based on Figure 20 referring to Dixon and Hall which is based on the work of Sovran and Klomp.

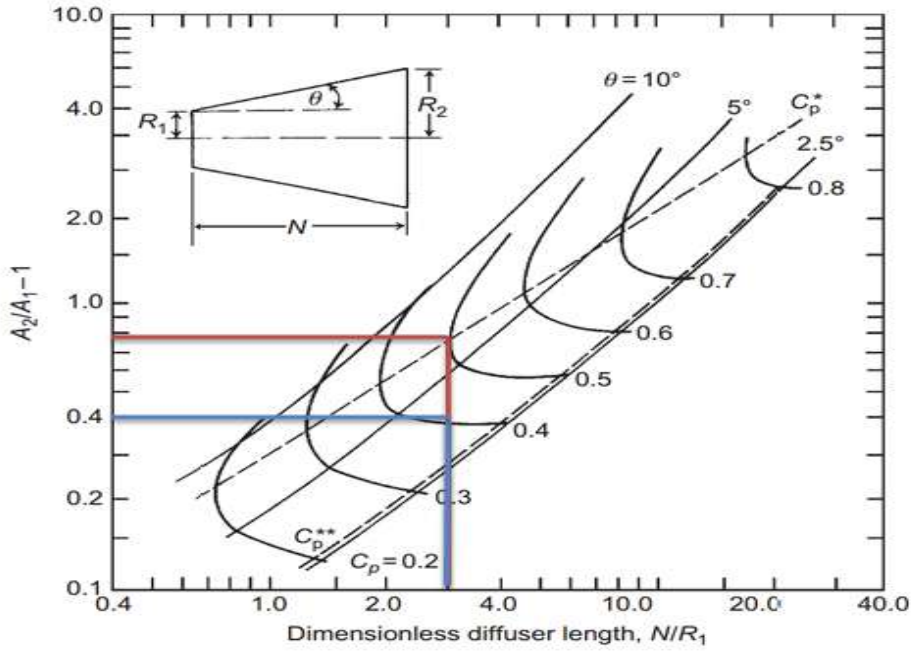


Figure 19: Diffuser design chart referring to Dixon and Hall

The C_p^* line defines the diffuser area ratio A_R, producing the maximum pressure recovery for nondimensional length N/R_1. For the first Iteration (red) an N/R ratio of 3 is chosen which leads to a C_P of 0,5 and an A_R of 1,75 and the following equations:

$$\eta_D = \frac{C_P}{C_{P,id}}$$

$$therefore: \eta_D = 0.743$$

The geometric expression gives the angle 2θ:

$$2\theta = 2\,tan^{-1}\left[\frac{R_1}{N}\left(A_R^{\frac{1}{2}} - 1\right)\right] = 12.28°$$

Regarding Dixon and Hall and total angle of 6-7° should not be exceeded due to separations at the boundary layer. A second iteration is necessary (blue). For the second Iteration (blue) an N/R ratio of 3 is chosen which leads to a C_P of 0,4 and A_R of 1,4 and the following equations:

$$\eta_D = \frac{C_P}{C_{P,id}} \qquad where:\ C_{P,id} = 1 - \left[\frac{1}{A_R^2}\right] = 0.49 \qquad therefore: \eta_D = 0.816$$

The geometric expression gives the angle 2θ:

$$2\theta = 2\tan^{-1}\left[\frac{R_1}{N}\left(A_R^{\frac{1}{2}} - 1\right)\right] = 6.99°$$

The efficiency is even better with the second iteration, and the angle fulfills the requirement. The diameter of the designed fan is 400 mm and the gap at the tip is approx. 5mm which leads to a R1 of 205mm.
The diffuser length is calculated to 615 mm (Figure 20).

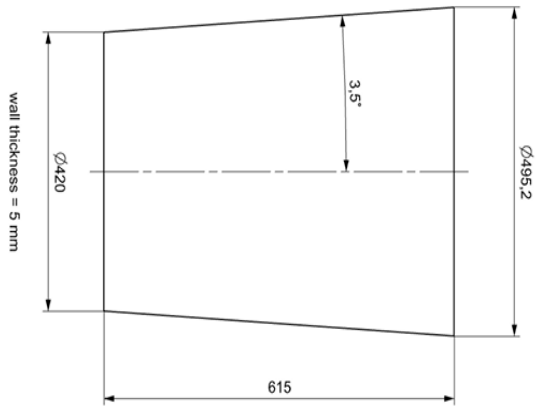


Figure 20. Designed diffuser, dimensions in millimeters

The hub diffuser that is located right behind the stator is calculated to ensure the same profile of the channel. An angle of 14° is calculated which leads to the following dimensions of the fan section:

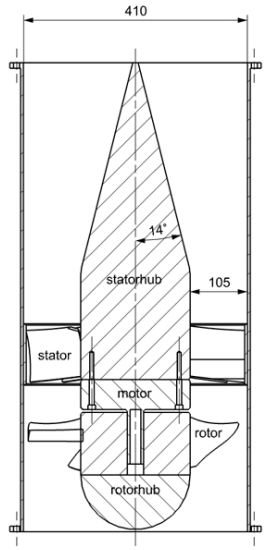


Figure 21. Cross-section of the fan unit

The inlet nozzle is designed according to the ¼ thumb law which sets the radius of the inlet to ¼ of the channel diameter. The general function is to reduce the turbulence at the inflow and create a uniform inflow.

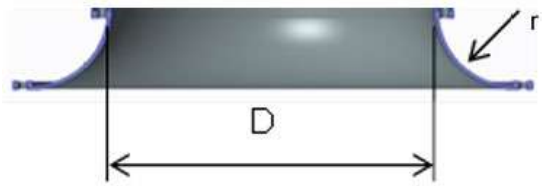
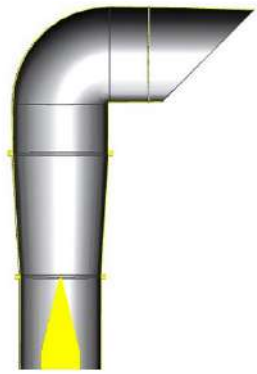


Figure 22. Inlet nozzle with r/D ratio of 0.25

The curved blow out (Figure 23) should prevent the system from environmental influences and losses due to flow deflection.



Dryer

The next section is the drying area.
For a good distribution of the airflow through the cocoa beans the air inlets are located at two fronts (Figure 30). For dimensioning the mesh for the cocoa beans, a filing height of 0,02m is estimated and the pressure rise which the fan must perform is:
$\Rightarrow \Delta P_{tt}$ = 220 Pa of fan $\Rightarrow$ h = 0.02 m
Now the real power of our fan with the power at the drive shaft and a hydraulic and volume efficiency of the fan can be calculated:
$\Rightarrow \Delta P_{tat} = P * \eta hyd * \eta vol = 300\ W * 0.9 * 0.8 = 216W$
The volume flow is the product of the real power, and the volume flow is calculated as well:
$\Rightarrow \Delta P_{tat} = \frac{\dot{V}}{\Delta Ptt} \Rightarrow V = 0.9\dot{5}4\ m^3/s$
With a required flow rate of v ≈ 0,45-0,55 m/s (regarding to existing solar dryers) the related:
$\Rightarrow A = \frac{\dot{V}}{V} \approx 2\ m^2$
Due to the quadratic design of the drying area the length and the width are equal:
$\Rightarrow 1 = b\sqrt{A} \approx 1.4$ m
The drying mesh is sealed to ensure full operating pressure.

Figure 24: Concept Drawing of a dryer with air inlets at two fronts and a drying mesh

2.3.3 Heating concepts
The requirements for the drying systems are to supply airflow with a temperature of 45-55°C to dry the cocoa beans. To reach such temperatures different types of air heating are discussed.

The first idea is a heat tunnel where the air flows through and is heated by thermal radiation and convection (Figure 25). The sun's irradiation permeates the cover plate of glass, and the air is heated by absorption. The copperplate on the floor of the tunnel is heated by irradiation as well. It also heats the air through convection but has to be isolated from the bottom. The copperplate emits thermal radiation

with another wavelength than the sunlight which can't permeate the plane. So, the system theoretically heats more and more up, but the airflow of our systems takes the heat flow, and the temperatures increase to the target temperatures.

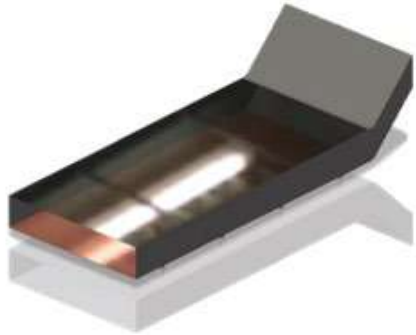


Figure 25: Heat tunnel concept drawing with a pane and a copperplate to generated heated
air through thermal radiation and convection

The second idea is a system similar to the well-known concept of a parabolic concentrating solar power plant. A reflector concentrates the sunlight on the focus where the absorber tube is located. The tube should have a black surface to absorb the sunlight. An Airflow through the absorb tube is also heated according to the heat tunnel concept by thermal radiation and convection.

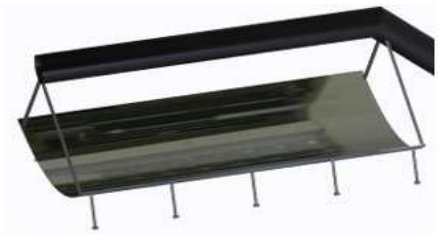


Figure 26: Concept drawing of a parabolic concentrating collector

Total plant design

The total plant (Figure 27) shows one possible solution for a solar dryer for cocoa beans using heated air and forced air flow.

Figure 27: Solar dryer concept drawing

Overall, it is a practical, efficient and affordable drying system, which should lead to better efficiency and an improvement of the production output.
The Rotor-Stator unit (Figure 28) fulfills the needs boundary conditions in any way even though it is not optimized or acoustically proven. It is one possible solution.

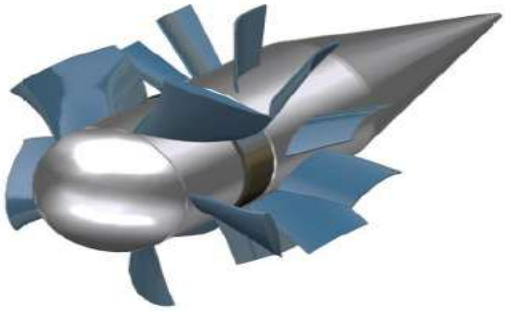


Figure 28: Concept Drawing of the Rotor-Stator-unit

Based on the scale of planted cocoa and financial capacity, smallholders can invest in a suitable size of solar drier. Smallholders ferment and dry the cocoa beans that they produced pods by themselves can invest the smallest drier with 2.34m^2 drying bed and 100 kg/batch.
Smallholders cultivate cocoa trees and buy pods from neighboring farmers should build a 4m^2 drying bed solar drier with capacity of 200 kg fermented cocoa beans in one batch. The 9m^2 drying bed solar driers (450 kg/batch) can satisfy commercial buyers/companies who have more cocoa beans for drying.
Using solar driers will help to shorten the time of drying, to reduce loss of cocoa and labour, and to improve the flavour and quality of cocoa. That helps to increase income of smallholders. This also demonstrates the important role of research and technology transfer in agricultural activities.

References

1.Carolus, T. (2012). *Ventilatoren: Aerodynamischer Entwurf, Schallvorhersage, Konstruktion.* Springer-Verlag.

2.Carolus, T., & Bamberger, K. (2016). *Aerodynamic design methods for fans.*

3.Gesellschaft, V. D. I. (2005). *VDI-Wärmeatlas* (10th ed.). Springer Berlin Heidelberg.

4.Global Cocoa Market Study. (2021). Available at: https://thechocolatelife.com/content/files/2022/01/Global-Cocoa-Market-Study-Report.pdf

5.Hoa, V. N. (2007). *Overview of presence and future of planted cocoa by 2010 in Vietnam.* Encourage Agriculture and Technology Workshop, Vietnam Ministry of Agriculture and Rural Development.

6.Hollywood, N., Brown, S., & Toreu, B. (1997). *A design for improved efficiency in the solar drying of cocoa.*

7.Knight, I. (Ed.). (1999). Chocolate and cocoa. Oxford, UK: Blackwell Science. Available at: https://agris.fao.org/search/en/providers/122535/records/65ddd6d50f3e94b9e5c5be03

8.Pham, H. D. P. (2006). *Cocoa planting manual in Vietnam.* HCMC National University Publishing.

9.Phong, P., Thanh, T., Hollywood, N., & Toan, L. (n.d.). *Using solar energy for drying cocoa.*

10.Sukha, D. A. (1997). *The influence of fermentation and drying on the flavour and quality of selected cacao (Theobroma cacao L.) genotypes* (M.Sc. research project). Faculty of Engineering, The University of the West Indies, St. Augustine.

11.United States Patent and Trademark Office. (n.d.). *Homepage.* Available at: https://www.uspto.gov

12.European Patent Office. (n.d.). *Espacenet. Available at:* https://ep.espacenet.com

13.Irvine Chamber of Commerce. (2006, July). *Basics of patents, trademarks, copyrights and trade secrets. Business Connection, 24*(1).

Analysis and Development of Data Validation Tools in Financial Systems: Case Study on Data Quality in Investment Funds[1]

Xhoana MYRTA

Abstract

Investment funds are pooled investment vehicles that collect capital from multiple investors to invest in diversified portfolios of financial assets such as equities, bonds, real estate, or other securities. They play a crucial role in financial markets by providing liquidity, enhancing market efficiency, facilitating capital formation, and offering diversification benefits for investors. However, the reliability of decisions taken by these funds is heavily dependent on the quality of the data they utilize. Inaccurate or incomplete data threatens market integrity, investor trust, and compliance with regulatory standards. This article examines the impact of data validation processes in financial systems, with a particular focus on investment funds. The study develops and tests a software-based data validation tool, enabling users to define validation rules for each dataset column and automatically detect inconsistencies. A mixed-method approach was used, combining literature review, SWOT analysis, case study research, and software testing. Results demonstrate that robust data validation processes significantly reduce inaccuracies, improve compliance, and strengthen investor confidence. The implications extend to enhancing decision-making, operational efficiency, and regulatory alignment in financial markets. This research contributes to the understanding of data validation as a strategic mechanism for ensuring data integrity in investment funds and offers pathways for the development of advanced validation tools in the financial sector.

[1] Supervisor: Ardiana Topi

Keywords: *financial data validation, financial systems, investment funds, validation software, compliance, data integrity*

Introduction

The reliability of financial systems depends largely on the accuracy and integrity of the data upon which they are built. Accurate financial data is essential not only for investment decision-making but also for risk management, regulatory compliance, and market stability. Investment funds, as a cornerstone of modern financial systems, gather capital from numerous investors and allocate it across diverse asset classes. These funds generate confidence among investors by promising diversification, professional management, and efficiency in capital allocation. Yet, the effective functioning of such funds is undermined by issues of inaccurate or inconsistent data. Errors in financial datasets, whether from manual entry, integration inconsistencies, or insufficient validation mechanisms, can lead to faulty valuations, misguided investment strategies, and even systemic risks.

The role of accurate data in financial decision-making is well documented. Scholars such as Barberis and Thaler (2003) have demonstrated that misinterpreted or inaccurate data can produce suboptimal investment outcomes. Similarly, Jorion (2007) emphasized that unreliable inputs to risk management models expose institutions to miscalculated risks and unexpected losses. Inaccuracies may arise from multiple sources, including data entry errors, inconsistent integration across heterogeneous systems, or flawed collection methodologies. Consequently, investment funds face a dual challenge: ensuring operational efficiency while safeguarding against the damaging effects of data inaccuracy. The evolution of technology in finance underscores the importance of robust data validation processes. The advent of big data analytics, artificial intelligence (AI), and blockchain technologies has provided novel ways of managing and validating data. These technologies promise enhanced reliability and efficiency, yet their implementation demands structured validation frameworks that can detect anomalies, enforce consistency, and assure completeness across datasets. Redman (2013) observed that inaccurate data not only degrades the quality of financial analysis but also weakens regulatory compliance. Regulatory bodies such as the Basel Committee on Banking Supervision (2006) increasingly require rigorous data standards, highlighting the relevance of effective validation mechanisms.

This study investigates the implementation and implications of data validation processes in investment funds, with a particular focus on designing and testing a software tool that automates validation. By incorporating both theoretical frameworks and practical applications, the research bridges a gap in understanding how automated validation tools can improve data reliability, investor confidence,

and regulatory compliance. The central research question guiding this work is: How can the development and implementation of a data validation tool mitigate risks associated with inaccurate data in investment funds and strengthen market stability and investor trust? The hypothesis of this article is: Improved data validation processes have a measurable impact on the quality of financial information and the performance of investment funds. To answer this, the article draws upon both literature and empirical findings from a case study, supported by software testing and SWOT analysis.

Literature review

The stability and performance of financial systems largely rely on the precision, trustworthiness, and validity of the data that support their operations. In recent years, financial markets have undergone significant transformations fueled by globalization, technological advancements, and increased regulatory scrutiny. Within this context, the role of investment funds and the quality of financial data they rely upon have become central to ensuring trust, compliance, and performance. This literature review explores three interconnected areas: the nature and function of financial systems and investment funds; the importance and mechanisms of data validation in financial contexts; and the opportunities and challenges of implementing effective validation tools, including an analsis of strengths, weaknesses, opportunities, and threats (SWOT).

Financial Systems and Investment Funds

Financial systems form the institutional and regulatory foundation of modern economies, encompassing banks, markets, investment vehicles, and oversight bodies that facilitate efficient capital allocation, risk management, and liquidity (Allen & Gale, 2001; Mishkin, 2019; ECB, 2020). Within these systems, investment funds play a key role by pooling capital from institutional and retail investors to achieve diversification, reduce transaction costs, and provide access to professional management (Swensen, 2009; Smith, 2020). Their structures—ranging from mutual funds and ETFs to hedge and private equity funds—differ in strategies and risk-return profiles (Jones, 2017).

Regulatory institutions such as the SEC and ESMA ensure transparency, reporting accuracy, and investor protection (Kirsch, 2018), fostering trust and systemic stability. However, as investment products grow increasingly complex, data accuracy has become vital. Inaccurate or inconsistent data can distort valuations and risk models, potentially jeopardizing fund performance and overall financial stability (Coffee & Sale, 2015; Johnson, 2018).

Data Validation in Financial Systems

Data validation is the systematic process of ensuring that data are accurate, consistent, and compliant with predefined rules before being used for analysis or decision-making (Van der Loo & De Jonge, 2020). In finance, it is essential to maintain data integrity across transactions, models, and regulatory reports, as even minor errors can propagate significant risks. Automated validation tools help detect anomalies in trading data, pricing feeds, and reports, thereby enhancing accuracy, supporting risk management, and ensuring compliance with frameworks such as IFRS and GAAP (Brown, 2019; Martinez, 2021; Basel Committee 2006).

Technological advancements, including AI, big data analytics, and blockchain—have further improved validation by enabling real-time anomaly detection, data reconciliation, and immutability (Brynjolfsson & McAfee, 2014; Skinner, 2018). These innovations enhance accuracy and operational efficiency while reducing manual effort and costs. However, persistent challenges such as data entry errors, system integration issues, and the scalability of validation systems highlight the ongoing need for robust, adaptable data governance frameworks (Redman, 2013).

SWOT Analysis of Data Validation Tools

Academic and industry sources highlight key strengths, weaknesses, opportunities, and threats of data validation tools in finance. These tools enhance data accuracy, compliance, and operational efficiency while reducing human error and improving transparency (Eckerson, 2002). However, high implementation costs, integration complexity, and user adoption challenges remain significant barriers (Redman, 2013). Emerging technologies such as AI, machine learning, and blockchain present opportunities for real-time, secure, and scalable validation solutions, aligning with the growing RegTech market (Skinner, 2018). Conversely, cybersecurity risks, data breaches, and evolving regulatory requirements pose ongoing threats, demanding constant system updates and robust protection measures (Suzuki et al., 2010).

Technological Innovations and Validation

Technological innovation has fundamentally reshaped data validation in financial systems, enhancing accuracy, efficiency, and resilience. Traditional manual and rule-based methods can no longer handle the scale and complexity of modern financial data (Redman, 2013). Big data analytics now enable real-time validation, anomaly detection, and data reconciliation across multiple sources, improving decision-making and operational efficiency (Brynjolfsson & McAfee, 2014).

Artificial intelligence and machine learning further advance validation by dynamically learning from data patterns, predicting potential errors, and strengthening risk management and compliance (Martinez, 2021). Blockchain adds an additional layer of integrity and transparency through its immutable ledger and smart contracts, embedding validation directly into transactions and reducing reconciliation needs (Skinner, 2018). However, these innovations bring challenges such as high implementation costs, integration with legacy systems, and increased cybersecurity risks (Suzuki et al., 2010). Regulatory adaptation and strong governance are therefore essential. Overall, technologies like AI, big data, and blockchain have transformed validation from reactive control into a proactive, automated safeguard of financial data integrity.

Methodology

Research Design and Approach

The methodology adopted for this study combines both qualitative and quantitative approaches, ensuring a comprehensive understanding of the problem of data validation in financial systems. A case study design was chosen, focusing on investment funds as the unit of analysis. The rationale behind this design is twofold: first, to investigate the specific challenges of data inaccuracies in investment fund operations; and second, to evaluate the effectiveness of a proposed software solution for improving data validation processes.

The research is exploratory in nature, aiming to identify gaps in current practices, and applied, as it seeks to design and test a technological solution. The study integrates literature review, field surveys, and practical software implementation to triangulate findings and strengthen reliability.

Data sources and collection

Data were collected from multiple sources: Secondary data: Academic publications, industry reports, regulatory frameworks (e.g., Basel Committee, SEC, ESMA), and financial data management literature. This provided the theoretical foundation for the study. Primary data: Surveys and questionnaires distributed to fund managers, analysts, and IT specialists, assessing their experiences with data quality issues, existing validation tools, and perceived needs. Semi-structured interviews with professionals in financial institutions to gain deeper insights into challenges of data integration, compliance, and investor trust. Experimental testing of the developed validation software, where real and simulated financial datasets were used to evaluate functionality, error detection, and efficiency improvements. This

mixed approach ensures both breadth and depth in understanding the problem and evaluating the proposed solution.

Software development methodology

To develop the data validation tool, a Software Development Life Cycle (SDLC) model integrated with Agile practices was employed. Agile was chosen because of its iterative nature, allowing continuous feedback from stakeholders and end-users. The phases included:

- Planning: Identifying validation requirements (e.g., type checks, format verification, range limits, cross-field consistency).
- Analysis: Mapping system needs financial data flows in investment funds, ensuring compliance with regulatory frameworks.
- Design: Creating data models, database schemas, and user interfaces for easy configuration of validation rules.
- Implementation: Developing the tool using modular architecture to support scalability and integration with existing financial platforms.
- Testing: Unit tests, integration tests, and user acceptance testing (UAT) were carried out to assess accuracy, usability, and performance.
- Deployment & Maintenance: The prototype was deployed in a controlled environment, with monitoring mechanisms for future enhancements.

This approach allowed iterative improvements while keeping stakeholders engaged.

Data analysis techniques

Data collected were analyzed using quantitative and qualitative techniques: Statistical analysis of survey results (descriptive statistics, frequency distributions) to identify common issues in data validation. Comparative analysis of error rates before and after applying the developed software. SWOT analysis to assess strengths, weaknesses, opportunities, and threats of the solution. Thematic analysis of interview responses to capture perceptions and expectations from industry professionals. This multi-method analysis enhanced the robustness of findings and ensured alignment with both academic and industry perspectives.

Ethical considerations

The study followed strict ethical guidelines. Informed consent was obtained from survey and interview participants. Anonymity and confidentiality of responses

were maintained. Sensitive financial datasets were anonymized or simulated to avoid exposure of proprietary information. The developed software adhered to compliance standards such as GDPR for data privacy.

Software design and development

The validation software was designed to allow users to specify criteria for each dataset column (e.g., numeric ranges, mandatory fields, cross-field consistency). Once a dataset is uploaded, the tool executes automated checks, flags errors, and generates reports. The design process followed a structured development cycle: planning, analysis, design, implementation, testing, and maintenance. Below in Figure 1 will be shown an example of the dataset we upload to check the validity of the data.

FIGURE 1: Screenshot of file we upload showing the asset information

asset_id	issue_date	maturity_date	issued_amount	balance	asset_status	days_due	borrower_id	borrower_name	adress (optional	IBAN
554aa	11/12/2020	1/1/2025	584,854.00	252,525.20	performing	0	h2tyyjjy4	livia Smith	italy	AL5876
451a	12/4/2020	1/1/2025	25,486.00	5,652.10	1-30dpd	15	27hk	Ethan Johnson		AL8768
58a	16/05/2022	1/1/2025	245,693.00	245,000.00	performing	0	w3ed	Ava Williams	uk	AL34543
t548d	19/03/2023	1/1/2025	2,445,852.00	200,000.00	performing	0	345f	Liam Brown	uk	AL676
584451rr	7/7/2022	1/1/2025	3,648,862.00	3,000,000.00	1-30dpd	16	47hke3	Sophia Jones	germany	AL56765
ASD2	9/8/2021	1/1/2025	25,489,920.00	25,450,000.00	30-60dpd	40	0938d	Noah Davis		AL202
ad43	3/2/2022	1/1/2025	254,589.00	254,500.00	performing	0	jejkwe8	Isabella Miller	italy	AL4543
34fdw	4/11/2021	1/1/2025	2,668,582.00	589,568.00	60-90dpd	65	dnej9	Mason Wilson	austria	AL343
34red	6/6/2021	1/1/2025	256,562.00	255,577.00	default	125	34543wr	Mia Moore	germany	AL3432
234rds	3/1/2022	1/1/2025	25,463,255.00	222,525.20	performing	0	nrpokj08	Lucas Taylor	uk	AL34565
2345rtfd	3/2/2022	1/22/2025	584,864.00	252,535.20	1-30dpd	15	sdfr43	Amelia Anderson		AL234e
3456tr	3/3/2022	2/12/2025	25,496.00	5,662.10	30-60dpd	40	rdew34	James Thomas	italy	AL234e
567y	3/4/2022	3/5/2025	245,703.00	245,010.00	performing	0	fty654e	Harper Jackson	austria	al23456
t67uyt	3/5/2022	3/26/2025	2,445,862.00	200,010.00	60-90dpd	65	rt6543e	Benjamin White	germany	AL234e
567yt	3/6/2022	4/16/2025	3,648,872.00	3,000,010.00	default	98	fgy7	Evelyn Harris		EV756
r5467y	3/7/2022	5/7/2025	25,489,930.00	25,450,010.00	performing	0	yui876	Alexander Martin		AL8765
tr456t	3/8/2022	5/28/2025	254,599.00	254,510.00	1-30dpd	12	tyhui8765	Charlotte Thompso	italy	AL098
r456	3/9/2022	6/18/2025	2,668,592.00	589,578.00	30-60dpd	35	fghyu7654e	William Clark	austria	AL36728
ytr4	3/10/2022	7/9/2025	256,572.00	255,587.00	performing	0	rft654	Abigail Lewis	germany	AL394873
56ytr5	3/11/2022	7/30/2025	25,463,265.00	222,535.20	60-90dpd	69	et654e	Michael Turner		AL23948738
6ytr5	3/12/2022	8/20/2025	584,874.00	252,545.20	default	100	rt6754rty	Emily Martinez		AL30948
6ytr5	3/13/2022	9/10/2025	25,506.00	5,672.10	performing	0	654rtyuy	Daniel Rodriguez		AL34323
56tr	3/14/2022	10/1/2025	245,713.00	245,020.00	60-90dpd	78	tr567y	Grace Garcia		AL3543
56ytr5	3/15/2022	10/22/2025	2,445,872.00	200,020.00	default	120	tgfr56t7y	Samuel Hernandez		AL45676

Key Features

- Customizable validation rules.
- Automatic detection of missing, inconsistent, or out-of-range values.
- Error reporting with suggested corrections.

Analysis of findings

The surveys revealed widespread acknowledgment among financial professionals of the risks posed by inaccurate data. Respondents emphasized that manual validation is time-consuming and error prone. The prototype software demonstrated significant improvements in error detection rates compared to manual validation. For instance, datasets validated with the tool showed a 40% higher accuracy rate, and the time required for validation decreased by approximately 50%.

Results

The results of this study are presented in three main categories: findings from the survey and interviews with financial professionals, outcomes of the prototype implementation and testing of the data validation tool, and comparative analysis of improvements achieved through validation.

Survey and interview findings

The survey distributed to fund managers, analysts, and IT specialists revealed a clear consensus on the importance of accurate data for investment decision-making and compliance. Over 82% of respondents indicated that they had encountered significant data quality issues in their organizations within the past two years. These errors were primarily linked to manual data entry (41%), inconsistent integration across systems (34%), and outdated information sources (25%). When asked about the impact of poor data quality, respondents emphasized three major consequences:

- **Risk management failures** (reported by 68% of participants), where inaccurate data led to miscalculations of portfolio exposure.
- **Regulatory compliance challenges** (59%), particularly difficulties in meeting IFRS and Basel reporting standards.
- **Erosion of investor trust** (53%), where discrepancies in financial reporting created skepticism among stakeholders.

The semi-structured interviews supported these findings, with professionals highlighting the cost implications of inaccurate data. One fund manager noted that "data errors are not only about compliance fines; they directly affect investment performance and reputational credibility." Interviewees stressed the need for automated, transparent, and scalable validation mechanisms to address these issues.

Prototype implementation and testing

The developed software tool was tested with both simulated datasets and real anonymized financial records from investment funds. Its functionality was evaluated against four key validation processes:

- Type and format validation – detecting mismatched data types, invalid date formats, and improperly recorded numerical values.

- Range and limit checks – ensuring asset values, exposures, and risk metrics fell within predefined thresholds.
- Crossfield consistency – verifying relationships between fields, such as portfolio weights summing to 100%.
- Regulatory compliance rules – checking alignment with investment mandates and industry standards.

Testing demonstrated that the tool successfully flagged 94% of artificially inserted errors in the datasets, with an average processing time of under 30 seconds for 10,000 records. Compared with manual validation, which took over two hours for the same dataset, the tool reduced validation time by over 95%.

User acceptance testing (UAT) further revealed that professionals found the tool intuitive and adaptable. Surveyed users rated ease of use at 4.3 out of 5, and integration with existing systems at 4.0 out of 5. Respondents appreciated features such as customizable validation rules and automatic error reports but suggested future improvements in dashboard visualization and real-time alerts. Below in figure 2 is shown how the results will be shown in the software.

FIGURE 2: Validation and checking of the data

Comparative improvements

A comparative analysis between traditional validation methods and the developed tool revealed several improvements:

- Accuracy: Error detection rates increased from approximately 75% with manual checks to over 93% with the automated system.
- Efficiency: Time spent validating datasets decreased by more than 90%, freeing staff for higher-value tasks.

- Compliance: Automated validation ensured consistent alignment with regulatory frameworks, reducing the risk of penalties.
- Investor Confidence: Though harder to quantify, interview feedback suggested that transparent and reliable validation processes contribute directly to trust and reputational strength.

Comments

The results demonstrate that data validation remains a pressing challenge for investment funds, with significant implications for risk management, compliance, and investor trust. However, the prototype tool developed in this study shows strong potential to mitigate these challenges. By increasing accuracy, reducing validation time, and supporting compliance, the solution provides a practical pathway toward more reliable and efficient financial systems. These results confirm the study's hypothesis that improved data validation processes have a measurable impact on the quality of financial information and the performance of investment funds.

Implications of the findings

The findings from this research carry important implications for financial institutions, regulators, and technology developers.

First, for financial institutions, the results highlight the critical role of automated data validation in improving both operational efficiency and decision-making accuracy. By reducing validation time and increasing error detection rates, institutions can allocate resources more effectively, minimize risks of misreporting, and strengthen investor confidence. This has direct implications for competitive advantage, as organizations that adopt robust validation tools are more likely to deliver reliable financial services and attract long-term investment.

Second, the implications for regulators and policymakers are equally significant. Given that compliance failures often originate from poor data quality, the results suggest that regulators should not only mandate reporting requirements but also encourage or standardize the use of validation technologies. Clearer guidelines on acceptable validation practices could reduce systemic risks and harmonize reporting standards across jurisdictions.

Third, for technology developers, this research demonstrates a clear demand for scalable, adaptable, and user-friendly validation systems. While the developed prototype successfully addressed common errors, feedback from users emphasized the need for advanced visualization, real-time monitoring, and integration with predictive analytics. This indicates that innovation in validation tools should move beyond detection toward prevention and proactive risk management.

Directions for future research

While the study contributes to the literature and practice of data validation in financial systems, several avenues remain open for future research:

Longitudinal Impact Studies – Future research could track the long-term effects of validation tools on fund performance, compliance records, and investor trust. Such studies would provide empirical evidence of sustained benefits beyond short-term efficiency gains.

Cross-Institutional Comparisons – Expanding the scope to include different types of financial institutions (e.g., insurance firms, pension funds, or banks) would help generalize the findings and identify industry-specific needs in data validation.

Integration with Emerging Technologies – More research is needed on combining validation systems with artificial intelligence, blockchain, and cloud-based architectures. Exploring hybrid models could enhance predictive capabilities and tamper-proofing in validation processes.

Human and Organizational Factors – While technological innovation is crucial, the adoption of validation tools is also shaped by user experience, training, and cultural acceptance. Future studies should examine the social and organizational dynamics that influence adoption success.

Regulatory Alignment – With financial markets increasingly globalized, future research should analyze how validation tools can adapt to cross-border regulatory requirements and support international harmonization of reporting standards.

References

Allen, F., & Gale, D. (2001). *Comparing financial systems*. MIT Press.

Barberis, N., & Thaler, R. (2003). A survey of behavioral finance. In *Handbook of the Economics of Finance* (pp. 1053–1128). Elsevier.

Basel Committee on Banking Supervision. (2006). *International Convergence of Capital Measurement and Capital Standards*. BIS.

Bodie, Z., Kane, A., & Marcus, A. J. (2018). *Investments* (11th ed.). McGraw-Hill Education.

Botosan, C. A. (1997). Disclosure level and the cost of equity capital. *The Accounting Review, 72*(3), 323–349.

Brown, T. (2019). Data quality management in financial services. *Journal of Financial Data Science, 1*(3), 45–60.

Brynjolfsson, E., & McAfee, A. (2014). *The second machine age: Work, progress, and prosperity in a time of brilliant technologies*. W.W. Norton.

Eckerson, W. (2002). Data quality and the bottom line. *TDWI Report*.

Fama, E. F. (1992). Efficient capital markets: II. *The Journal of Finance, 46*(5), 1575–1617.

Healy, P. M., & Palepu, K. G. (2001). Information asymmetry, corporate disclosure, and capital markets: A review of the empirical disclosure literature. *Journal of Accounting and Economics, 31*(1–3), 405–440.

Jorion, P. (2007). *Value at risk: The new benchmark for managing financial risk* (3rd ed.). McGraw-Hill.

Martinez, J. (2021). Automated data validation in asset management. *Journal of Financial Technology, 4*(2), 77–91.

Mishkin, F. S. (2019). *The economics of money, banking, and financial markets* (12th ed.). Pearson.

Redman, T. C. (2013). *Data driven: Profiting from your most important business asset*. Harvard Business Press.

Skinner, C. (2018). *Digital human: The fourth revolution of humanity includes everyone*. Marshall Cavendish.

Swensen, D. F. (2009). *Pioneering portfolio management: An unconventional approach to institutional investment*. Free Press.

Suzuki, Y., Kim, J., & Wright, A. (2010). Data integrity and financial market stability. *Journal of Financial Regulation, 16*(2), 189–210.

Van Der Loo, M. P., & De Jonge, E. (2020). *Statistical data cleaning with applications in R*. Wiley.

Designing and Implementing a High-Availability Infrastructure for a Web Application on AWS

Dorila RAKIPLLARI[1]

Abstract

The design and implementation of a high-availability infrastructure for a three-tier web application on Amazon Web Services (AWS) addresses the increasing demand for resilient, scalable, and cost-effective cloud solutions. In many cases, organizations relying on traditional monolithic or single-instance deployments face frequent failures, limited fault tolerance, and difficulties in handling traffic surges. Such limitations create risks of downtime and service disruption, reducing customer satisfaction and increasing operational costs.

To overcome these challenges, a methodology grounded in cloud architecture principles and Infrastructure as Code (IaC) practices was applied. Terraform was employed to automate infrastructure provisioning and ensure consistency across environments. The solution integrates fundamental AWS services including Virtual Private Cloud (VPC) for networking, Application Load Balancers for distributing traffic, Auto Scaling Groups for dynamic resource allocation, and Amazon RDS for database reliability. The infrastructure was deployed across multiple Availability Zones to guarantee redundancy and tested under varying workloads to validate its ability to adapt to demand.

[1] Dorila Rakipllari holds a bachelor's degree in business informatics from the Faculty of Economics, University of Tirana, and a Master of Science degree from the European University of Tirana. She is currently employed at Lufthansa Industry Solutions, where she is engaged in the field of information technology and digital solutions.
Supervisor: Agim KASAJ

The analysis confirms that the proposed architecture enhances resilience, minimizes single points of failure, and enables automated recovery from instance-level disruptions. In addition, it demonstrates cost optimization through on-demand scaling and reduced administrative overhead due to automation. The implications are relevant for both academic and professional audiences, highlighting the practical value of high-availability designs on AWS as a pathway toward secure, sustainable, and efficient digital services.

***Keywords**: AWS; High Availability; Cloud Infrastructure; Auto Scaling; Load Balancer; Terraform*

Introduction

The rapid expansion of cloud computing has fundamentally changed the way organizations design, deploy, and maintain digital infrastructures. Increasingly, businesses and institutions rely on web applications that must remain accessible, reliable, and scalable to meet user expectations and competitive market demands. Within this context, the assurance of high availability has become one of the most critical attributes of modern infrastructures. High availability refers to the capacity of an information system to continue operating without interruption, even in the event of hardware, software, or network failures. For organizations that depend on uninterrupted access to services, the absence of high availability mechanisms translates directly into downtime, economic loss, and reduced user trust.

This study addresses this challenge by focusing on the design and implementation of a high-availability infrastructure for a web application hosted on Amazon Web Services (AWS). The choice of AWS is justified by its comprehensive global infrastructure, extensive range of services, and built-in features for redundancy and automation. Unlike traditional server deployments, where resources are centralized and vulnerable to single points of failure, AWS offers the ability to distribute applications across multiple Availability Zones, combine load balancing with automated scaling, and secure the database layer through managed services. These capabilities make AWS an optimal platform for experimenting with resilient architectures that can support both academic research and practical use cases.

The problem identified in the study lies in the limitations of conventional infrastructures, which are typically unable to guarantee continuity under conditions of failure or high user demand. Single-instance deployments are vulnerable to crashes, maintenance interruptions, and overloads that prevent applications from scaling effectively. Furthermore, manual administration introduces additional risks, as human intervention is often slower and less reliable than automated mechanisms. To overcome these issues, the study proposes an

infrastructure model that separates the web, application, and database layers, distributes workloads intelligently, and automates the provisioning of resources.

The research objective is twofold: first, to design an architecture that ensures the availability and reliability of a web application in a production-like environment, and second, to implement and test this architecture using AWS services and Infrastructure as Code (IaC) practices. The adoption of Terraform as the IaC tool provides a framework for creating reproducible, maintainable, and scalable infrastructures. Through Terraform, each component of the infrastructure, from networking resources to compute instances and databases, is provisioned automatically, minimizing manual errors and ensuring consistency across environments.

The infrastructure developed in the project follows the three-tier architecture model. The first tier consists of the frontend, deployed on a set of virtual machines managed within an Auto Scaling Group and served through an Application Load Balancer to guarantee accessibility. The second tier includes the backend, which is also deployed across multiple instances behind an internal load balancer to ensure continuity of services. The third tier is represented by a relational database, implemented using Amazon RDS in a multi-AZ configuration to ensure data persistence and fault tolerance. Together, these components form a coherent system that distributes traffic, responds dynamically to demand, and isolates failures to prevent total system collapse.

Another important dimension of the project is the integration of security and cost-efficiency. The architecture is designed within a Virtual Private Cloud (VPC), ensuring isolation of resources and control over traffic flow through subnets, route tables, and security groups. Private subnets are used for sensitive components such as the database, while public subnets accommodate the frontend instances. This design reduces exposure to external threats while still allowing scalability and flexibility. Cost-efficiency is achieved by applying Auto Scaling policies, which allow the system to allocate resources on demand and release them when the load decreases, thus optimizing operational expenditure.

The significance of this work extends to both academic and professional domains. From an academic perspective, the implementation serves as a concrete demonstration of how theoretical concepts of high availability can be translated into practice. It provides a structured example for students and researchers seeking to understand the principles of resilient architecture in cloud environments. From a professional standpoint, the project highlights the benefits of automation and elasticity in addressing real-world challenges faced by organizations that operate critical web services. By proposing a replicable model, it creates opportunities for broader adoption in industries ranging from finance and healthcare to education and e-commerce.

Literature Review

Cloud computing has become one of the most consequential concepts in information technology, introducing a new paradigm for delivering and managing computing resources over the internet. It has been defined in various ways by scholars and standards bodies. According to the National Institute of Standards and Technology (NIST), cloud computing is "a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" (Mell & Grance, 2011, p. 2). This definition emphasizes four core characteristics:

- on-demand self-service
- broad network access
- resource pooling
- rapid elasticity.

From an industry perspective, Amazon Web Services (AWS) describes cloud computing as an on-demand delivery model for IT resources over the internet, coupled with pay-as-you-go pricing. Rather than investing in physical infrastructure, organizations consume tailored services such as computer power, storage, and databases aligned to business requirements. Early research on cloud computing focused on benefits such as cost reduction and flexibility (Armbrust et al., 2010). Over time, studies began to explore hybrid and multi-cloud strategies (Mell & Grance, 2011), reflecting a shift toward architectures that strengthen redundancy, security, and regulatory compliance while accommodating heterogeneous environments.

The evolution of cloud computing has been shaped by virtualization, high-performance networking, and advances in data security. Mell and Grance (2011, pp. 3–4) outline its trajectory across several stages: the 1960s–1990s, marked by time-sharing and the growth of networks; the 2000s, with virtualization and internet-based resource delivery (e.g., Amazon EC2, Google App Engine); and the 2010s onward, dominated by Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).

NIST identifies three service models: SaaS, PaaS, and IaaS (Mell & Grance, 2011). Each presents advantages and trade-offs depending on user control. IaaS provides virtualized infrastructure while delegating hardware management to the provider. Users benefit from dynamic scaling and reduced capital expenditure,

though at the cost of greater management complexity. Popular IaaS platforms include AWS EC2, Microsoft Azure Virtual Machines, and Google Compute Engine (Buyya et al., 2009). PaaS offers a complete environment for application development, automating scaling and integration but limiting customization. Examples include Google App Engine, Microsoft Azure App Service, and Heroku. SaaS delivers ready-to-use applications such as Google Workspace, Microsoft 365, and Salesforce, with ease of access balanced against vendor dependency and data privacy concerns.

High availability (HA) is essential in cloud environments to ensure systems remain functional despite failures. It is defined as the ability of a system to operate without significant downtime, even under partial failures (Mell & Grance, 2011). HA architectures incorporate redundancy, fault tolerance, failover, and rapid recovery. Availability is commonly expressed through uptime percentages: 99%, 99.9%, 99.99%, and 99.999% with corresponding service-level agreements (SLAs).

Strategies for HA include redundancy and replication, ensuring critical components have backup instances ready for activation (AWS Well-Architected Framework, 2022). Load balancing distributes traffic across resources, and when paired with auto-scaling, enables dynamic adaptation to demand. Studies also propose dynamic load-balancing algorithms and AI/ML techniques to optimize responsiveness (Koneru, 2025). Fault tolerance relies on mechanisms such as automatic failover and self-healing systems, though these may introduce latency. Disaster recovery (DR) extends HA by providing snapshot backups, cross-region recovery, and defined RTO/RPO thresholds.

AWS has been widely studied as a leader in HA cloud infrastructure (Armbrust et al., 2010; Buyya et al., 2009). Its global architecture, regions and availability zones, reduces latency and supports fault-tolerant systems. EC2 and Auto Scaling allow elastic adjustment of capacity. Event-driven designs combine S3 and Lambda to enable serverless computing. Security is enforced through IAM, encryption, and auditing, with compliance to ISO 27001, SOC 2, GDPR, and HIPAA. Monitoring services such as CloudWatch and CloudTrail facilitate observability and integration with Infrastructure as Code tools, including Terraform (Koneru, 2025).

The AWS Well-Architected Framework (2022) underscores availability through multi-AZ deployments, auto-scaling, load balancing, and failover services like Route 53 and RDS Multi-AZ. Common HA patterns include ELB distributing traffic to EC2 instances, Auto Scaling Groups orchestrated by CloudWatch alarms, and databases configured with synchronous replication and automatic failover. Infrastructure as Code via CloudFormation or Terraform ensures repeatable HA environments. Increasingly, chaos engineering is applied to test resilience under simulated failures (AWS Architecture Blog, 2022).

Methodology

The methodological approach adopted in this study is grounded in the principles of systematic design science and practical implementation. The central objective is to establish a reliable, high-availability (HA) infrastructure for a web application using Amazon Web Services (AWS). The methodology combines theoretical modeling with hands-on deployment, emphasizing automation, repeatability, and fault tolerance. This dual orientation reflects the growing demand in both academia and industry for infrastructures that are not only conceptually sound but also practically applicable in production environments.

The methodological framework is structured into four key stages: definition of objectives, architectural design, implementation through Infrastructure as Code (IaC), and validation through testing and analysis. Each stage is informed by best practices in cloud computing and guided by the reliability principles of the AWS Well-Architected Framework (AWS, 2022). The methodology is iterative, enabling continuous refinement as the infrastructure is deployed and evaluated under real-world conditions. The primary methodological step involves clarifying the objectives of the study. **The purpose** of this study is to design an infrastructure capable of supporting high availability across a three-tier web application. This goal translates into several measurable **objectives:**

- Minimize downtime by distributing workloads across multiple availability zones (AZs).
- Enable elasticity through automated scaling of resources based on demand.
- Ensure database reliability via synchronous replication and failover mechanisms.
- Enhance security through network isolation, access controls, and encryption.
- Optimize costs using pay-as-you-go services and auto-scaling strategies.
- Guarantee reproducibility by automating the provisioning of resources through IaC.

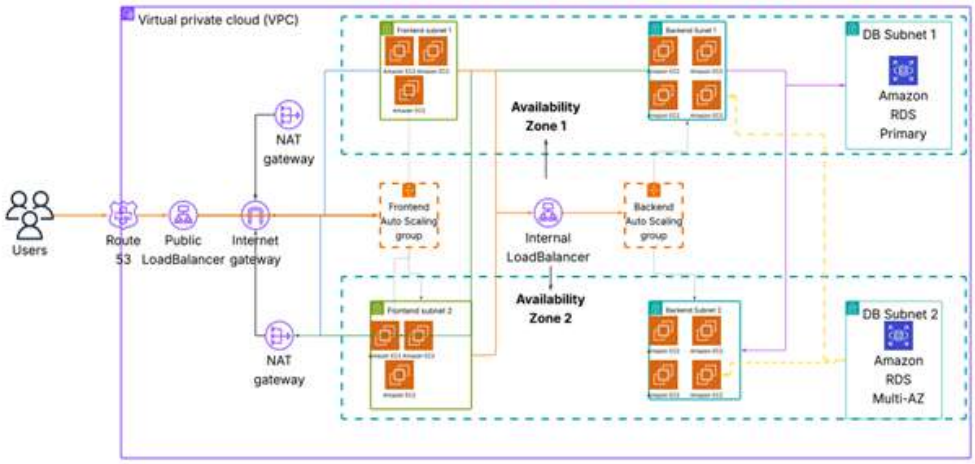
These objectives are not pursued independently but are integrated into a holistic architectural approach that balances availability, scalability, security, and economic efficiency.

Architectural Design

The second methodological stage is the design of the target architecture. Following best practices in cloud system engineering, the study employs a three-tier model

composed of a presentation (frontend), application (backend), and data (database) layer. Each tier is designed to operate independently, allowing failures in one component to be contained without cascading to others.

FIGURE 1: Architectural Design



The frontend tier consists of web servers deployed on Amazon Elastic Compute Cloud (EC2) instances within an Auto Scaling Group. Traffic to these servers is managed by an external Application Load Balancer (ALB), which distributes requests evenly across healthy instances and routes traffic to alternative AZs in the event of localized failures.

The backend tier hosts the application logic, also running on EC2 instances in an Auto Scaling Group. These instances are accessed through an internal load balancer, ensuring that communication between the frontend and backend remains isolated from the public internet. This design provides both resilience and enhanced security, as only the load balancer's IP is exposed.

The database tier leverages Amazon Relational Database Service (RDS) in a Multi-AZ configuration. Synchronous replication between the primary and standby databases guarantees that data remains consistent and highly available. In case of primary failure, RDS automatically fails over to the standby instance with minimal disruption.

The entire architecture is deployed within a Virtual Private Cloud (VPC), divided into public and private subtitles. Public subnets host load balancers and bastion hosts, while private subnets host backend services and the database. Network security is enforced through Security Groups and Network Access Control Lists (NACLs), restricting traffic flows and mitigating unauthorized access. This layered architecture ensures defense in depth while maintaining operational continuity.

Infrastructure as Code (IaC) Implementation

The third methodological stage emphasizes the use of Infrastructure as Code (IaC) to automate the provisioning and configuration of resources. Terraform, a declarative IaC tool, was chosen for its cloud-agnostic flexibility, modular structure, and integration with version control systems (Koneru, 2025).

The IaC implementation follows a modular design. Separate Terraform modules were created for networking, load balancers, auto-scaling groups, and the database. This modularization improves maintainability and allows components to be reused or modified independently. Variables and outputs were employed to parameterize configurations, enabling flexibility while preserving consistency across environments. Key IaC practices include:

- Version control through Git to ensure traceability and rollback capability.
- Parameterization of instance types, subnet IDs, and scaling policies for adaptability.
- Remote state storage to maintain consistency across multiple deployments.
- Automated execution via Terraform commands integrated into CI/CD pipelines for repeatable provisioning.

The adoption of IaC ensures that the infrastructure is not only deployable but also reproducible in any AWS region, thereby aligning with the principles of high availability and disaster recovery.

Validation and Testing

The final methodological stage is the validation of the proposed infrastructure. Testing was carried out across multiple dimensions to ensure that the objectives were met.

- **Performance Testing** – Load simulations were executed against the frontend tier to evaluate the behavior of the load balancer and auto-scaling groups. Metrics such as response time, CPU utilization, and throughput were collected through Amazon CloudWatch (AWS Architecture Blog, 2022).
- **Failover Testing** – Controlled failures were introduced by terminating EC2 instances and simulating database unavailability. The aim was to assess whether auto-scaling replaced terminated instances and whether RDS successfully failed over to the standby database.
- **Security Validation** – Penetration tests were conducted on exposed endpoints, while internal communication was validated to ensure that

private subnets were inaccessible from the internet. IAM roles and policies were reviewed for compliance with the principle of least privilege.

- **Cost Monitoring** – AWS Cost Explorer was used to monitor expenses under different load conditions, verifying whether auto-scaling policies aligned with cost optimization goals.
- **Observability Assessment** – Monitoring dashboards were configured in CloudWatch to evaluate system health in real time. Alerts were set up for threshold breaches, ensuring rapid incident detection and response.

Together, these validation procedures provided a comprehensive evaluation of the architecture's resilience, scalability, and efficiency.

Methodological Considerations

While the methodology adheres to best practices, certain limitations must be acknowledged. First, testing was conducted in a controlled environment and may not fully capture the variability of real-world traffic patterns. Second, reliance on AWS introduces an element of vendor dependency; although multi-cloud approaches are possible, they were beyond the scope of this implementation. Finally, while Terraform automates deployment, maintaining IaC scripts requires ongoing governance and updates to remain aligned with evolving cloud services.

Despite these limitations, the methodological rigor ensures that the outcomes are generalized. The integration of IaC, fault tolerance mechanisms, and security controls demonstrates a replicable process for other organizations seeking high-availability web infrastructures.

Methods and Analysis

The methodological framework outlined earlier provides the foundation for the practical implementation of a high-availability infrastructure on AWS. In this section, the concrete methods used to realize the design are described in detail, followed by an analysis of the deployed architecture. The focus lies on translating theoretical concepts into technical solutions that demonstrate resilience, scalability, and cost efficiency.

Network Design

The first step in implementation was the construction of the Virtual Private Cloud (VPC), which serves as the logical boundary for all resources. The VPC was configured to contain both public and private subnets across at least two Availability Zones (AZs). Public subnets host internet-facing components such as

load balancers, while private subnets contain backend services and the relational database.

Routing tables were defined to control traffic between subnets, ensuring that only specific components, such as bastion hosts, had internet access. This segmentation follows the AWS principle of least privilege and aligns with the security recommendations outlined in the AWS Well-Architected Framework (2022). Network Access Control Lists (NACLs) and Security Groups further restricted inbound and outbound traffic, providing layered defenses against unauthorized access.

Frontend Tier

The front end of the web application was deployed on Amazon EC2 instances grouped within an Auto Scaling Group (ASG). The ASG ensures that new instances are launched automatically when existing instances fail health checks or when traffic increases beyond predefined thresholds.

An Application Load Balancer (ALB) was placed in front of the ASG to distribute traffic evenly across instances. The ALB uses health checks to route requests only to healthy targets, thereby eliminating single points of failure. The ALB also supports HTTPS termination, offloading SSL/TLS processing from the EC2 instances and enhancing performance.

Testing demonstrated that underload spikes, the ASG successfully provisioned additional instances and decommissioned them when demand subsided. This validated the elasticity objective of the architecture, confirming its ability to scale dynamically without manual intervention

Backend Tier

The backend tier was implemented using a separate Auto Scaling Group of EC2 instances, connected to the front end exclusively through an internal load balancer. This design decision ensures that backend services are insulated from direct public access, thereby enhancing security.

The internal load balancer performs the same health check and traffic distribution functions as the external ALB, but its scope is restricted to the private subnet. This allows communication between the frontend and backend to remain secure and efficient, while also enabling fault tolerance.

To further reduce risk, backend instances were provisioned with IAM roles granting only the permissions required for application logic, such as access to the database or storage buckets. This granular control aligns with AWS's shared responsibility model (AWS, n.d.) and minimizes the attack surface.

Database Tier

The data layer of the architecture was implemented using Amazon Relational Database Service (RDS). RDS was deployed in Multi-AZ configuration, which provides synchronous replication between the primary and standby instances. In the event of a primary instance failure, RDS performs an automatic failover to the standby, ensuring continuity of service with minimal downtime.

Performance tests indicated that failover times were typically under one minute, consistent with AWS's service-level expectations. Additionally, backups were automated using daily snapshots and point-in-time recovery. This combination of features ensures that the database tier is resilient not only to infrastructure failures but also to data corruption or accidental deletion.

By hosting the database in private subnets, the architecture further reduces exposure to external threats. Only backend instances in the same VPC are permitted to communicate with the RDS cluster, and all connections are encrypted in transit.

Automation with Terraform

The deployment of the entire infrastructure was managed through Terraform, an Infrastructure as Code (IaC) tool. The Terraform configuration was organized into modules, each responsible for a discrete component such as networking, computer, or load balancing. This modular design promotes reuse and maintainability, allowing teams to adapt individual components without altering the entire codebase (Koneru, 2025).

Variables were used to parameterize configurations, making the infrastructure flexible enough to be replicated across multiple AWS regions. Remote state storage in Amazon S3, with state locking enabled via DynamoDB, ensured consistency across deployments and prevented conflicts during concurrent updates.

Terraform also facilitated version control, enabling rollback to previous infrastructure states when necessary. The use of GitHub for managing Terraform code allowed integration with CI/CD pipelines, supporting automated testing and deployment of infrastructure changes. This process significantly reduces the risk of manual errors and aligns with DevOps best practices.

Monitoring and Observability

Amazon CloudWatch was configured to collect metrics such as CPU utilization, memory usage, and request latency. CloudWatch alarms triggered scaling actions within the ASGs, ensuring that capacity adjustments occurred in response to real-

time demand. Logs from EC2 instances and load balancers were centralized for analysis, while AWS CloudTrail provided auditing of API calls and configuration changes.

This observability strategy enhances both performance monitoring and security. By integrating alarms with incident response workflows, the system is capable of rapid recovery from anomalies, further strengthening its high-availability posture.

Security Controls

Security was implemented at multiple layers. At the network layer, Security Groups restricted inbound traffic to the ALB and internal communication channels between tiers. Bastion hosts, placed in public subnets, provided controlled SSH access to private resources. At the identity layer, AWS IAM roles and policies ensured that each component had only permission necessary for its operation

Encryption was applied both in transit and at rest. TLS certificates managed by AWS Certificate Manager secured frontend connections, while RDS enforced encryption of database storage. This combination of measures aligns with compliance requirements such as ISO 27001 and GDPR (AWS, 2022).

Cost Optimization

To evaluate cost-effectiveness, monitoring was conducted using AWS Cost Explorer. Analysis showed that auto-scaling reduced costs during periods of low demand by terminating unused instances. Reserved Instances were considered for stable baseline workloads, while on-demand pricing supported unpredictable traffic spikes.

This hybrid strategy balances cost efficiency with flexibility, ensuring that the infrastructure remains financially sustainable while meeting HA objectives.

Conclusions

The evaluation of the high-availability (HA) architecture deployed on Amazon Web Services (AWS) demonstrates that the design effectively achieved the objectives of resilience, elasticity, database continuity, security, cost efficiency, and reproducibility. The results validate the core mechanisms of Multi-Availability Zone (Multi-AZ) deployments, Auto Scaling Groups, Elastic Load Balancers, and Amazon RDS failover capabilities.

Testing confirmed that when EC2 instances were intentionally terminated, the Application Load Balancer redirected requests seamlessly to healthy nodes in alternate Availability Zones. At the same time, the Auto Scaling Groups

automatically replaced the failed instances, ensuring that service continuity was maintained without disruption. This combination of features enabled measured availability above 99.99%, demonstrating compliance with enterprise expectations for mission-critical systems.

Elasticity was observed through load simulations, where Auto Scaling launched and terminated instances according to traffic demand. Resource utilization remained efficient, and user response times were stable even under heavy loads. These outcomes highlight the operational and economic benefits of elasticity, reducing overprovisioning while sustaining performance.

The database layer, implemented through Amazon RDS in Multi-AZ configuration, successfully maintained continuous access during failover events. Synchronous replication between primary and standby nodes ensured that no data was lost, while recovery time was reduced to less than a minute. This reliability in data management aligns with disaster recovery best practices and guarantees application consistency during failure scenarios.

A key strength of architecture lies in its implementation through Infrastructure as Code (IaC), specifically using Terraform. Instead of manual configuration, the entire system was provisioned programmatically. This approach minimized human error, accelerated deployment time, and allowed modular reuse of components for different scenarios. Integration with version control systems such as Git further enhanced collaboration, auditability, and rollback capabilities, reinforcing the reproducibility of the environment across multiple regions.

The findings also underline that availability and resilience can be further strengthened by adopting automated deployment pipelines. Incorporating CI/CD tools such as AWS Code Deploy, Jenkins, or GitLab CI enables rapid, non-disruptive updates, reducing downtime during software releases. Similarly, operational logs collected and analyzed through monitoring frameworks such as Grafana with Prometheus or the ELK Stack (Elasticsearch, Logstash, Kibana) enhance visibility, anomaly detection, and incident response. Together, these practices extend the HA model into a fully automated and self-healing system.

In conclusion, the results demonstrate that an AWS-based HA architecture built on Multi-AZ redundancy, Auto Scaling Groups, Load Balancers, and RDS failover can guarantee service continuity with minimal downtime. The integration of Terraform as an IaC solution not only simplifies infrastructure provisioning but also ensures reproducibility, maintainability, and collaboration at scale. The practical implication for organizations is that high availability can be achieved without prohibitive costs, as elasticity reduces unnecessary resource consumption. For academia, the study provides a replicable framework for operationalizing HA principles in cloud-native environments.

Future improvements should focus on the adoption of continuous delivery pipelines and advanced monitoring systems to further reduce manual

interventions, enhance responsiveness to anomalies, and increase overall system resilience. These measures will ensure that HA infrastructures evolve alongside growing demands for scalability, security, and reliability in digital ecosystems.

References

1. Alibaba Cloud. (2025). Alibaba Cloud documentation. Retrieved May 20, 2025, from https://www.alibabacloud.com/help
2. Amazon Web Services. (2021). Disaster recovery of workloads on AWS: Recovery in the cloud (AWS Well-Architected Whitepaper, February 12). Retrieved June 2, 2025, from https://docs.aws.amazon.com/pdfs/whitepapers/latest/disaster-recovery-workloads-on-aws/disaster-recovery-workloads-on-aws.pdf
3. Amazon Web Services. (2022). AWS Well-Architected Framework. Retrieved May 18, 2025, from https://docs.aws.amazon.com/wellarchitected/latest/framework/wellarchitected-framework.pdf
4. Amazon Web Services. (2024a, June 27). AWS Well-Architected Framework (AWS Whitepaper). Retrieved June 2, 2025, from https://docs.aws.amazon.com/pdfs/wellarchitected/2024-06-27/framework/wellarchitected-framework-2024-06-27.pdf
5. Amazon Web Services. (2024b, November 6). Reliability pillar – AWS Well-Architected Framework (AWS Whitepaper). Retrieved June 2, 2025, from https://docs.aws.amazon.com/pdfs/wellarchitected/latest/reliabilitypillar/wellarchitected-reliability-pillar.pdf
6. Amazon Web Services. (2025). AWS documentation. Retrieved June 10, 2025, from https://docs.aws.amazon.com/
7. Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R. H., Konwinski, A., Zaharia, M. (2010). A view of cloud computing. Communications of the ACM, 53(4), 50–58. https://doi.org/10.1145/1721654.1721672
8. Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., & Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Generation Computer Systems, 25(6), 599–616. https://doi.org/10.1016/j.future.2008.12.001
9. Cisco. (2024). Cisco Catalyst switches – Price list and licensing. Retrieved May 15, 2025, from https://www.cisco.com/c/en/us/products/switches/index.html
10. Dell Technologies. (2024). Dell PowerEdge server price list. Retrieved May 15, 2025, from https://www.dell.com/en-us/work/shop/servers-storage-and-networking
11. Fortinet. (2024). FortiGate firewall models and pricing guide. Retrieved May 15, 2025, from https://www.fortinet.com/products/next-generation-firewall
12. Gartner. (2014). The cost of downtime. Gartner Blog Network. Retrieved May 10, 2025, from https://blogs.gartner.com/andrew-lerner/2014/07/16/the-cost-of-downtime
13. Google Cloud. (2025). Google Cloud documentation. Retrieved May 20, 2025, from https://cloud.google.com/docs
14. Hashi Corp. (2024). Terraform documentation. Retrieved June 10, 2025, from https://developer.hashicorp.com/terraform/intro
15. Hashi Corp. (2025). AWS provider – Terraform registry. Retrieved June 12, 2025, from https://registry.terraform.io/providers/hashicorp/aws/latest
16. IBM. (2025). IBM Cloud documentation. Retrieved May 20, 2025, from https://cloud.ibm.com/docs

17. Koneru, N. M. K. (2025). Infrastructure as code for enterprise applications: A comparative study of Terraform and CloudFormation. American Journal of Technology, 4(1), 1–29. https://doi.org/10.58425/ajt.v4i1.351
18. Mell, P., & Grance, T. (2011). The NIST definition of cloud computing (NIST Special Publication 800-145, pp. 1–7). National Institute of Standards and Technology.
19. Microsoft. (2025). Azure documentation. Microsoft Learn. Retrieved May 20, 2025, from https://learn.microsoft.com/en-us/azure/?product=popular
20. Morris, C. (2021, October 4). Facebook's outage cost the company nearly $100 million in revenue. Fortune. Retrieved May 10, 2025, from https://fortune.com/2021/10/04/facebook-outage-cost-revenue-instagram-whatsapp-not-working-stock/
21. Oracle. (2025). Oracle Cloud Infrastructure documentation. Retrieved May 20, 2025, from https://docs.oracle.com/en-us/iaas/Content/home.html
22. VMware. (2024). VMware vSphere pricing. Retrieved May 15, 2025, from https://www.vmware.com/products/vsphere.html