Form Application with Python and Java Backends for Response Management and Authentication

Daniel Santiago Pérez
Faculty of Systems Engineering
Universidad Distrital Francisco Jose de Caldas
dsperezm@udistrital.edu.co

Jaider Santiago Avila
Faculty of Systems Engineering
Universidad Distrital Francisco Jose de Caldas
snmendivelsom@udistrital.edu.co

***Abstract*—The context of this work focuses on the growing need for cloud-based applications that allow users to create and manage forms efficiently. With the rise of productivity tools like Google Forms, there has been a demand for platforms that facilitate the creation of surveys, contact forms, and questionnaires for a wide and diverse audience. The secure handling of user data and flexibility in form types are key aspects to ensure usability and security of these applications.**

The proposed solution in this project involves developing a form application based on a bifurcated backend in two languages: Python (using FastAPI) and Java (using Spring Boot). The application allows user authentication, form creation and management with different question types, and secure storage of responses. Additionally, notification functionality and a premium subscription system are implemented, offering features such as response export and advanced form management.

The results obtained during the development of this application have been satisfactory. A robust platform has been built capable of managing forms and responses in real-time, ensuring data security through JWT-based authentication and efficient database management. Tests conducted in PostgreSQL confirm the system's scalability and reliability, making it suitable for production environments.

I. Introduction

Today, web applications have taken a central role in people's daily lives, and among them, form creation applications have become essential for both individuals and organizations. Google Forms has led this type of solution, offering an easy-to-use platform for surveys, questionnaires, and contact forms. However, there is a latent need for more flexible systems that can be customized and used in various scenarios without the limitations of current commercial platforms.

This project aimed to develop a form application with similar features to Google Forms, but with a focus on data security, flexibility in form management, and the implementation of a modular backend in two languages, Python and Java. This bifurcation allows the application to leverage the strengths of each ecosystem: Python's speed and ease of development alongside Java's robustness and scalability.

User authentication is crucial in this application to ensure that only authorized users can create, modify, and view forms. A JWT (JSON Web Tokens)-based authentication system was implemented to ensure the security and scalability of the solution. Additionally, notification functionality allows form creators to receive alerts when responses are received, while the premium subscription system grants additional features to users, such as response export to CSV or PDF.

II. Methods and Materials

The development of this project was divided into two main sections: the creation of the backend in Python using FastAPI, and the backend in Java using Spring Boot. Both backends are connected to a PostgreSQL database, which is responsible for storing both the created forms and user responses.

A. Python Backend

The Python backend was developed using the FastAPI framework due to its efficiency and ease of use. FastAPI allows for the rapid creation of RESTful APIs, and its native support for data validation and security handling was crucial for implementing JWT-based authentication. Forms and responses are stored in the database using SQLAlchemy as the Object Relational Mapping (ORM) tool.

B. Java Backend

The Java backend was developed with Spring Boot, a robust framework that facilitates the development of enterprise applications. Spring Security was used to manage authentication and authorization, while Hibernate was chosen as the ORM for database communication. The decision to use Spring Boot was motivated by its high performance and scalability.

C. Database

PostgreSQL was chosen as the database due to its scalability, support for complex transactions, and high performance. Forms and responses are stored in separate tables, and one-to-many relationships were implemented between users and forms, and between forms and responses.

D. Response Handling and Notifications

The notification functionality was implemented using SMTP email services, allowing form creators to receive email notifications when a new response is recorded. Responses are stored in the database and can be queried in real-time.

III. Conclusions

This work has demonstrated the feasibility of developing a robust form application using backends in two different languages. Modularity and security were the fundamental pillars, and the system has proven efficient in managing users, forms, and responses.