UNIVERSITATEA POLITEHNICA BUCUREȘTI
FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

Quick Report - A web application that allows users to report events

Serban Petre Bogdan

TABLE OF CONTESTS

ABSTRACT

Bureaucracy is starting to be a bigger problem in this era and people have a need to make these processes quicker. Quick Report is a web application that allows citizens to report new events from their community. Users have to authenticate in the platform and then, they can report new events for state authorities. After the event is concluded, state authorities' representatives can change the status of the event and users can see what happened to that event.

1. Introduction

In Romania is a constant need for upgrading and creating web platforms to ease the communication between the citizens and the state authorities. The work that has been done in the last years showed that this will help with the bureaucracy and can lead to smaller queues and to a better communication. An automated platform where people can upload their new discovered events can lead to quicker fixes of those and it is a great tool for state authorities to find out their priorities.

The web application described in this article aims to improve communication and streamline bureaucracy between citizens and state authorities in Romania. It allows citizens to easily report and share new events and provides state authorities with a valuable tool for identifying and addressing priorities. The need for such a platform is ongoing in Romania, as the country has a history of seeking ways to improve communication and reduce bureaucracy through the use of web platforms. The work that has been done in the past has demonstrated the effectiveness of these platforms in achieving these goals, with improvements in communication and reductions in queues and bureaucracy. By providing an automated platform for reporting and sharing events, the web application described in this article has the potential to further improve communication and efficiency between citizens and authorities in Romania.

One of the key benefits of this web application is its ability to facilitate real-time communication between citizens and authorities. By allowing citizens to quickly and easily report events as they occur, the web application can help to ensure that issues are addressed in a timely manner, rather than becoming lost or forgotten in the bureaucracy of traditional channels. This can be particularly important in situations where timely action is needed, such as in the case of emergency repairs or maintenance.

In addition to its real-time communication capabilities, the web application also provides a centralized hub for information and resources related to the events being reported. This can be particularly useful for authorities, as it allows them to easily access and review relevant information, and to coordinate their efforts in addressing the reported events.

Overall, the web application described in this article represents a valuable tool for improving communication and streamlining bureaucracy between citizens and state authorities in Romania. By providing an automated platform for reporting and sharing events, it has the potential to make a significant impact in improving the efficiency and effectiveness of public services in the country.

2. The Tech Stack

For creating this project, the MERN tech stack will be used in the developing phase:

- ReactJs - front-end
- NodeJs and Express - back-end
- MongoDb – database
- Google Analytics
- SeleniumAPI – testing
- FireBase - deployment
- GitLab – code base and CI/CD

These technologies offer great tools to develop web application based on http requests.

ReactJS is a JavaScript toolkit for creating reusable user interface components that is declarative, fast, and versatile. It's an open-source, component-based front-end library that's just responsible for the application's view layer. Taking security into consideration, the application will be using JWT tokens for the authentication process. Also, the ReactJs application will be based on axios requests to follow ajax development guidelines.

Node.js is a JavaScript run-time environment for servers. Node.js' standard library includes a set of asynchronous I/O primitives that prevent JavaScript code from blocking, and libraries in Node.js are often created following non-blocking paradigms, making blocking behavior the exception rather than the rule. It makes a great pairing for our React Frontend.

MongoDB is a NoSQL database management system that is free and open source. MongoDB is a database that allows you to organize, store, and retrieve document-oriented data. The documents resemble JavaScript Object Notation, although they employ a binary JSON variation (BSON). The choice for MongoDB stands in the ease of integration within our application (MERN Stack) and the fact that it could be easily used in a cloud deployment context by employing the help of Atlas, a fully-managed cloud database that handles all the complexity of deploying, managing, and healing your deployments.

Firebase is a set of hosting services for any type of application. It offers NoSQL and real-time hosting of databases and other content, and it suits well this project since we can make use of some free-to-use features for deploying our application.

Selenium is an umbrella project encapsulating a variety of tools and libraries enabling web browser automation, and we will be using the python framework of Selenium for creating a suite of automated test to cover the uses of the application.

Once a web application is developed and tested the next phase in its lifecycle is monitoring and reporting for keeping track of what is working and what not. For this purpose, we chose to integrate **Google Analytics** into the project for an easy but feature packed solution.

All code base will be versioned using **GitLab**. Also, this choice targets the robust CI/CD pipeline integrations that are provided by GitLab. This CI/CD will contain building the project, testing automatically and deployment.

3. Design and Preparation

To comply and follow better the properties of **accessibility and robustness**, being part of the **usability** concept, this project started with creating the design for the pages of the application and creating a plan for development. The application targets citizens of a country and employees of different state authorities that can change the status of an event. For this, two different types of users are used throughout the application. The design of the application consists of 3 pages (first page, dashboard page, and insert event page). First page needs to have two buttons, one for login and one for register, and each one will redirect the user to the specific action. Dashboard page needs to show the user events inserted by him, if the user is a citizen one, and the events corresponding to the state authority if the user is a state authority. Also, for the citizen user, a button is needed to redirect to the form that is used to create a new event. Insert event page consists of a minimalist form in which a title, a description and a category of the event is required. Proto.io was used for the design of the pages.

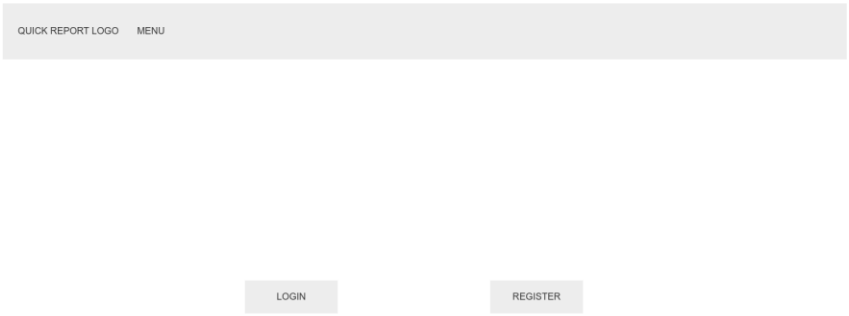

Figure1. Start Page

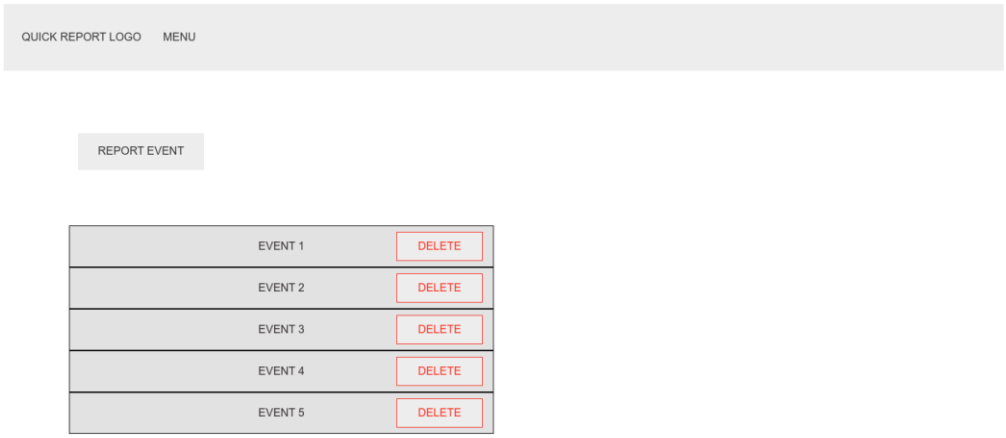

Figure 2. Dashboard

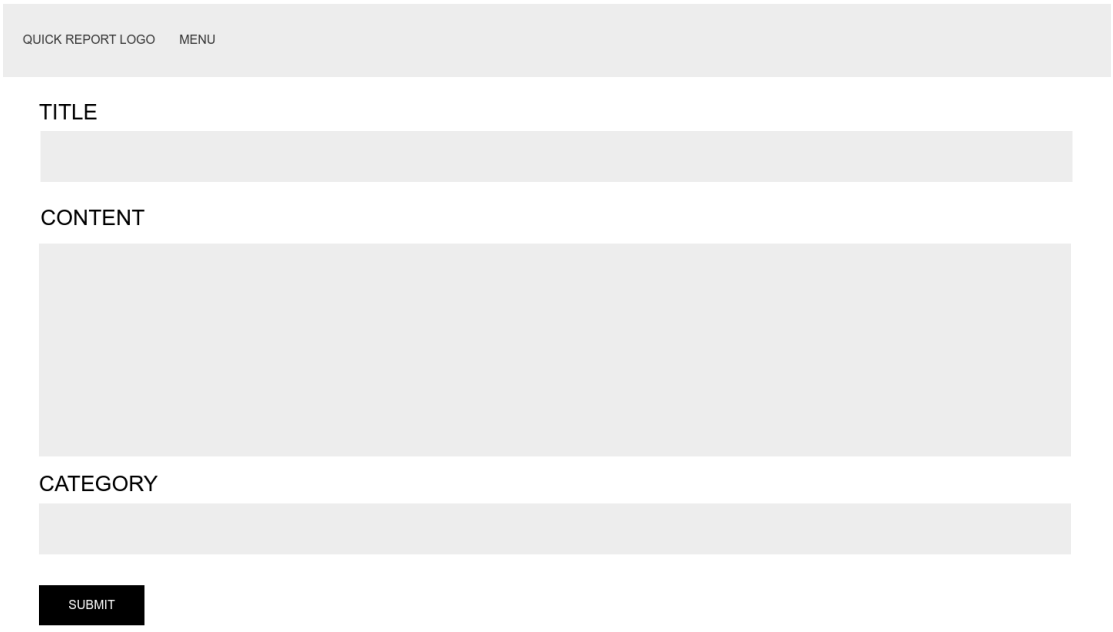

Figure 3. Insert Event

3.1. User Interface

The process of designing user interfaces in software or digital devices with a focus on appearance or style is known as user interface (UI) design. Designers strive to design user interfaces that are simple to use and enjoyable. UI design encompasses both graphical and non-graphical user interfaces, such as voice-controlled interfaces. In this project, the user interface is minimal and user friendly, easy to work with, and shows explicit messages to users.

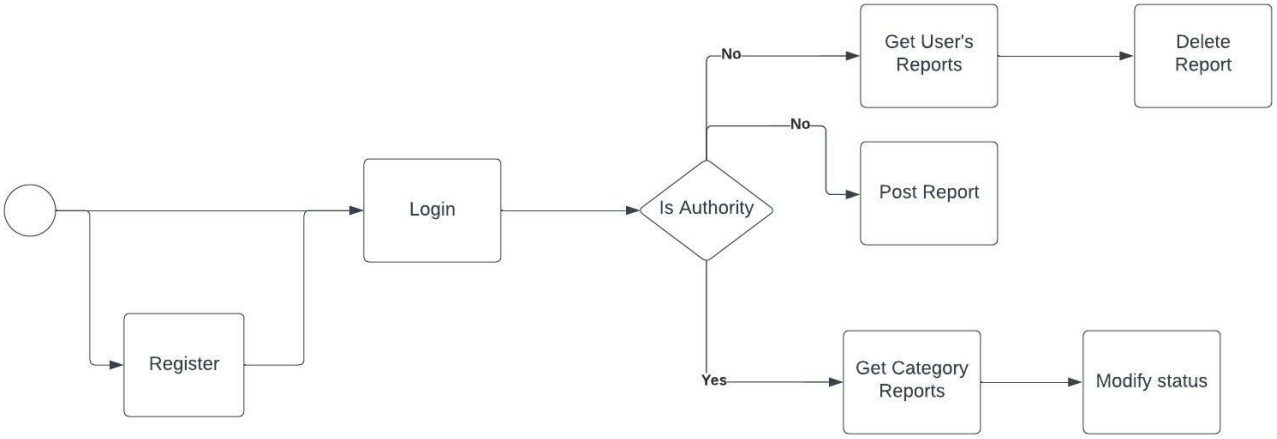

Figure 4. Flow Diagram.

When a user enters the application, will have the options to login if he already has an account, or to register if the user is new to the platform. As citizen users, they can add a new report to the platform and can see all of their previous reports. They can check every report status, having values of: created, in progress, solved. Users have the option to delete a report. As authority users, they can see all the reports that correspond to their authority, can see the details of the reports, and they can change the status of a report, change that will be displayed to the citizen user's page.

4. Implementation

Before the development of the application, the design should be considered. As for this reason, as a first step, a user flow will be determined so it could be followed into the development phase. Also, in terms of UI design, Proto.io will be used for the mock-up of the constituting pages.

In terms of architecture and implementation decisions, the project followed some key points of reference:

- AJAX approach - with use of axios library
- Minimal and Simple design with user focus
- Test driven development (TDD) - with use of SeleniumAPI
- Data Security - authentication based on JWT tokens
- App monitoring through Google Analytics
- CI/CD pipeline integration
- Firebase deployment

4.1. Authentication

The Authentication process is straight-forward as the industry dictates and takes into consideration the security of the user. The password is encrypted using **bcrypt** and the whole process is based on the standard **JWT** that defines a compact and self-contained way for securely transmitting information between parties, the user receives a unique token for the current session, everything being incorporated into the Mongo database.

4.2. Insert Event Form

For the insert event form, it was used a form from React Bootstrap that creates a simple and reliable form for the users to report an event. The data is sent to backend on a post request from the backend that will create a new entry in the table Events. A relation with a user is required for an event.

4.3. Dashboard

The dashboard page looks different based on the type of the user logged into the application. If a user is a citizen, it will display the events created by him, with details about the event and the status. Frontend will make a HTTP request to the backend and this one will provide the frontend with the events. Also, a button is present that redirects the user to the insert event form. For a state authority user, the dashboard shows the events assigned to that authority. In database, each event has a category(salubritate, rutier, ordine, administrativ). To check for authority, the authority from the user's email is evaluated(e.g. stefan@administrativ.ro -> administrativ is the category). JSON objects are used for communication between frontend and backend.

4.4. Database

MongoDB is a NoSQL database that is used in the Quick Report web application to store and manage the data related to events reported by users. It is a flexible, scalable database that can handle large volumes of data and allows for easy querying and indexing of data.

In the context of the Quick Report project, MongoDB is used to store the details of each reported event, including information such as the location, description, and status of the event. This data is then used to populate the dashboard and other features of the web application, allowing authorities to easily access and review the reported events.

MongoDB works by storing data in collections of documents, rather than in tables as in a traditional relational database. This allows for greater flexibility and scalability, as the structure of the data can be easily modified and new data can be easily added without the need to redesign the entire database.

Overall, the use of MongoDB in the Quick Report web application helps to ensure the efficient and reliable management of the data related to reported events, supporting the goal of improving communication and streamlining bureaucracy between citizens and authorities.

4.5. Deployment

Firebase is a cloud-based platform for developing and deploying web applications, and it is used in the Quick Report project for hosting and deploying the web application.

Firebase provides a range of tools and services for developing and deploying web applications, including a hosting service for hosting static and dynamic content, a real-time database for storing and syncing data, and tools for authenticating users and managing access to the application.

In the context of the Quick Report project, Firebase is used to host the web application and to provide the real-time database and authentication services needed to support the reporting and sharing of events. It allows the web application to be easily deployed and accessed by users and authorities and helps to ensure that the data related to reported events is securely stored and synced in real-time.

Overall, the use of Firebase in the Quick Report project helps to ensure the dependability of the web application, with a focus on security and safety. It provides a reliable and secure platform for hosting and deploying the web application, supporting the goal of improving communication and streamlining bureaucracy between citizens and authorities.

4.6. Overview on testing and maintenance

The Quick Report web application is supported by a continuous integration/continuous delivery (CI/CD) pipeline, which is used to automate the building, testing, and deployment of the application. This helps to ensure that the application is consistently built and deployed in a reliable and repeatable manner, reducing the risk of errors and downtime.

As part of **the CI/CD pipeline**, the Quick Report web application is also equipped with a suite of **automated tests**, which are used to validate the functionality and performance of the application. These

tests are run automatically as part of the pipeline, helping to ensure that the application is of high quality and meets the needs of users and authorities.

In addition to the CI/CD pipeline, the Quick Report web application also integrates with Google Analytics, a tool for tracking and analyzing web traffic. This allows us to monitor the usage of the web application and to understand how it is being used by users and authorities. This information can be used to identify areas for improvement and to optimize the performance and user experience of the application.

Overall, the use of a CI/CD pipeline and Google Analytics in the Quick Report project helps to ensure the dependability of the web application, with a focus on security and safety. By automating the build, test, and deployment process and by tracking and analyzing web traffic, we can ensure that the application is consistently delivered in a reliable and secure manner and that it meets the needs and expectations of users and authorities. These tools and practices help to support the goal of improving communication and streamlining bureaucracy between citizens and authorities, and contribute to the overall dependability of the web application.

5. Dependability & Quality properties

The Quick Report web application follows the principles of **dependability** in order to ensure the **safety**, **security**, and **reliability** of the application. These principles are critical to the success of the application, as they help to ensure that it can be trusted by users and authorities to accurately and reliably report and share events.

To ensure the **safety** of the application, we employed a test-driven development approach and used automated tests written in Selenium. These tests are run as part of the CI/CD pipeline, ensuring that the application is thoroughly tested and validated before being deployed. In addition, we also utilized app monitoring with Google Analytics to identify and address any issues or abnormalities that may arise in the application.

To ensure the **security** of the application, we implemented the use of JWT tokens for authentication. This helps to protect the user data and ensures that only authorized users can access the application.

In addition to dependability, the Quick Report web application also follows other principles of **quality**, such as **accessibility** and **robustness**. These principles are essential for ensuring that the application is easy to use and capable of handling a range of different scenarios. To achieve these goals, we employed a design process that included the use of mock-ups and flow diagrams to better understand the needs of users. We also utilized Lighthouse to test the usability of the user interface, ensuring that it is intuitive and easy to use.

6. Conclusions

In conclusion, the Quick Report web application represents a valuable tool for improving communication and streamlining bureaucracy between citizens and state authorities in Romania. By providing an automated platform for reporting and sharing events, it has the potential to make a significant impact in improving the efficiency and effectiveness of public services in the country.

The web application is built using a range of technologies, including MongoDB for data management, Firebase for hosting and deployment, and a CI/CD pipeline and Google Analytics for tracking and analyzing web traffic. These tools and practices help to ensure the dependability of the web application, with a focus on security and safety.

In a nutshell, the Quick Report web application represents a valuable solution for addressing the ongoing need for upgrading and creating web platforms to improve communication and reduce bureaucracy in Romania. It has the potential to significantly improve the efficiency and effectiveness of public services in the country, and to make a positive impact on the lives of citizens and authorities.