CEBU INSTITUTE OF TECHNOLOGY
UNIVERSITY

IT342-Section SYSTEMS INTEGRATION AND ARCHITECTURE 1

FUNCTIONAL REQUIREMENTS SPECIFICATION (FRS)

Project Title: Mini App – User Registration & Authentication

Prepared By: Florence Azriel R. Migallos

Date of Submission: 02/06/2026

Version: 1.0

Table of Contents

1. Introduction

1.1. Purpose

The purpose of this document is to define the functional and non-functional requirements for the “Mini App – User Registration & Authentication” system. It serves as a guide for the development phase by detailing how the system should behave, how data is managed, and how users interact with the interface. This document is intended for students, system developers, and instructors.

1.2. Scope

The "Mini App" is a web-based system designed to manage user identity. The scope of the system includes:

- Allowing new users ("Guest Users") to register for an account.
- Allowing registered users to log in securely using credentials.
- Providing authenticated users access to protected views (Profile and Dashboard).
- Enabling users to terminate their session via logout.
- Restricting access to protected pages for unauthenticated users.

1.3. Definitions, Acronyms, and Abbreviations

Term	Definition
FRS	**Functional Requirements Specification**. This document, which details the behavior and requirements of the Mini App.
ERD	**Entity Relationship Diagram**. The diagram that defines the database structure, specifically the Users table and its columns.
API	**Application Programming Interface**. The backend system (Spring Boot) that receives requests from the User Interface (React) to perform tasks like logging in.
JWT	**JSON Web Token**. The specific type of "Token" generated by the TokenProvider class to validate a user's session after logging in.
JSON	**JavaScript Object Notation**. The format used to structure the data sent between the React UI and the Spring Boot API.
BCrypt	A password-hashing algorithm mentioned in the security logic to ensure user passwords are not stored as plain text.

Spring Boot	The Java-based framework used for the backend API.
React	The JavaScript library used to build the User Interface (UI) where users interact with the forms.

2. Overall Description

2.1. System Perspective

This system operates as a standalone client-server application. It consists of a frontend interface developed in React (React UI) and a backend REST API developed in Spring Boot. The system interacts with a relational database to store user records and authentication data.

2.2. User Classes and Characteristics

The system identifies two primary user classes:

1. **Guest User**: An unauthenticated user who can only access the Registration and Login pages. Their primary goal is to gain access to the system.
2. **Authenticated User**: A user who has successfully logged in. They have permissions to view the Dashboard, view their Profile, and Logout.

2.3. Operating Environment

The system requires the following hardware and software environments to function correctly:

Client Side (User Interface):

- **Browser**: A modern, standard-compliant web browser such as Google Chrome (v90+), Microsoft Edge, or Mozilla Firefox.
- **Configuration**: JavaScript must be enabled in the browser settings.
- **Display**: Optimized for desktop and laptop screens (minimum resolution 1366x768), but responsive enough for mobile viewports.

Server Side (Backend API):

- **Runtime**: Java Development Kit (JDK) 17 or higher (LTS version recommended for Spring Boot 3.x).
- **Framework**: Spring Boot 3.0+ running on an embedded Apache Tomcat web server.
- **Port**: The application server requires port 8080 (default) to be open and available on the host machine.

Database Server:

- **DBMS**: MySQL 8.0+ or PostgreSQL 14+.
- **Connection**: Must support JDBC connectivity.
- **Storage**: Sufficient disk space to store user records (minimal requirement for this mini-app).
- **Port**: The database service must be listening on its default port (e.g., 3306 for MySQL or 5432 for PostgreSQL).

2.4. Assumptions and Dependencies

- The user device has an active internet connection to communicate with the Spring Boot API.
- The database service is running and accessible by the API.
- Users possess a valid email address for registration.

3. System Features and Functional Requirements

3.1. Feature 1: User Registration

Description:
This feature allows a Guest User to create a new account by providing personal details. The system validates the input and stores the record in the database.

Functional Requirements:

- **FR-1.1:** The system shall provide a registration form requiring: Username, Email, First Name, Last Name, Phone, and Password.
- **FR-1.2:** The system shall check if the provided email already exists in the database.
- **FR-1.3:** If the email is unique, the system shall hash the password before saving.
- **FR-1.4:** The system shall confirm successful registration with a "201 Created" message.

3.2. Feature 2: User Authentication (Login)

Description: This feature verifies the identity of a registered user. It grants access to the system if the provided credentials match the stored records

Functional Requirements:

- **FR-2.1:** The system shall provide a login form accepting Username and Password.
- **FR-2.2:** The system shall validate the password against the stored password hash using BCrypt.
- **FR-2.3:** If credentials are valid, the system shall generate a session token (e.g., JWT)
- **FR-2.4:** If credentials are invalid, the system shall deny access and return an " Unauthorized" status (implied by typical flow).

3.3. Feature 3: View Dashboard & Profile

Description: This feature allows an Authenticated User to view their personal dashboard and profile details. These pages display data specific to the logged-in user.

Functional Requirements:

- **FR-3.1:** The system shall retrieve and display the logged-in user's details (Username, Email, Name) on the Profile page.
- **FR-3.2:** The system shall present a Dashboard view that serves as the main landing page after a successful login.
- **FR-3.3:** The system must ensure that the data displayed corresponds strictly to the user_id in the active session token.

3.4. Feature 4: Protected Access & Logout

Description: This feature enforces security by restricting access to protected pages and handling session termination.

Functional Requirements:

- **FR-4.1:** The system shall intercept requests to the Dashboard and Profile routes and verify the presence of a valid session token.
- **FR-4.2:** The system shall redirect unauthenticated users to the Login page if they attempt to access protected routes.
- **FR-4.3:** Upon clicking "Logout," the system shall invalidate the current session and redirect the user back to the Login page.

4. Non-Functional Requirements

Security:

- All user passwords must be stored as hashes (e.g., using BCrypt), never as plain text.
- Authentication tokens must be transmitted securely (e.g., via HTTPS).

Usability:

- The interface should provide clear feedback messages (e.g., "Registration Successful").

Reliability:

- The system should prevent duplicate account creation by enforcing unique email constraints.

Data Integrity:

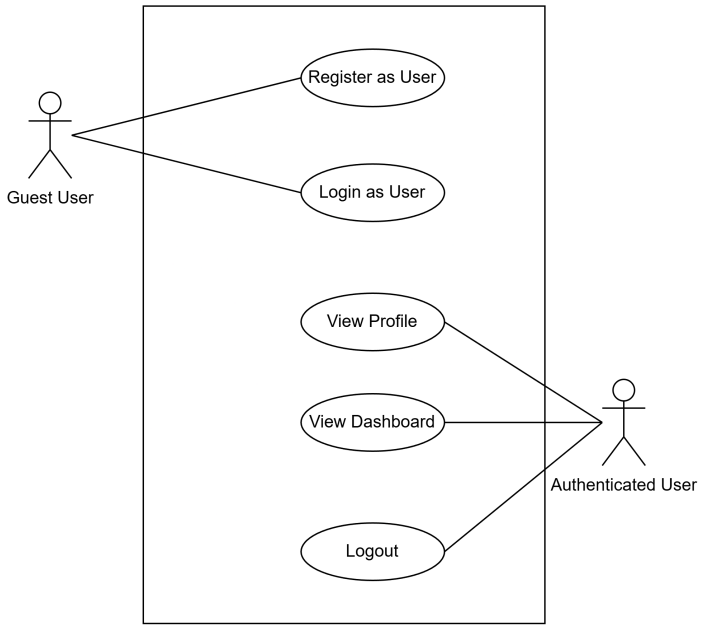
- User records must include a unique primary key (user_id) to ensure accurate data retrieval

5. System Models (Diagrams)

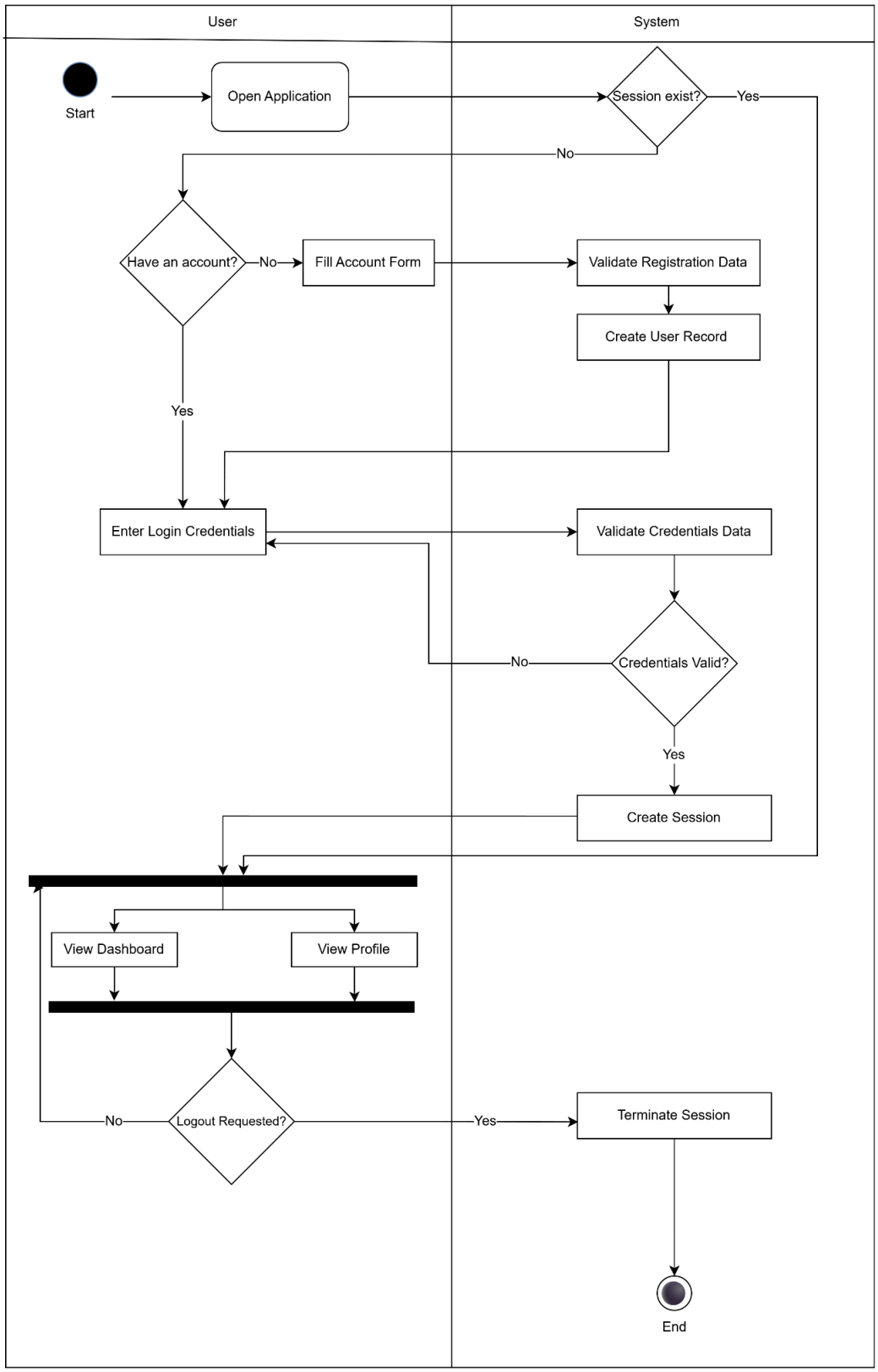
5.1. ERD

User	
PK	**userId : INT**
	username : VARCHAR(50)
	email : VARCHAR(100)
	firstName : VARCHAR(50)
	lastName : VARCHAR(50)
	phone : VARCHAR(20)
	passwordHash: VARCHAR(255)

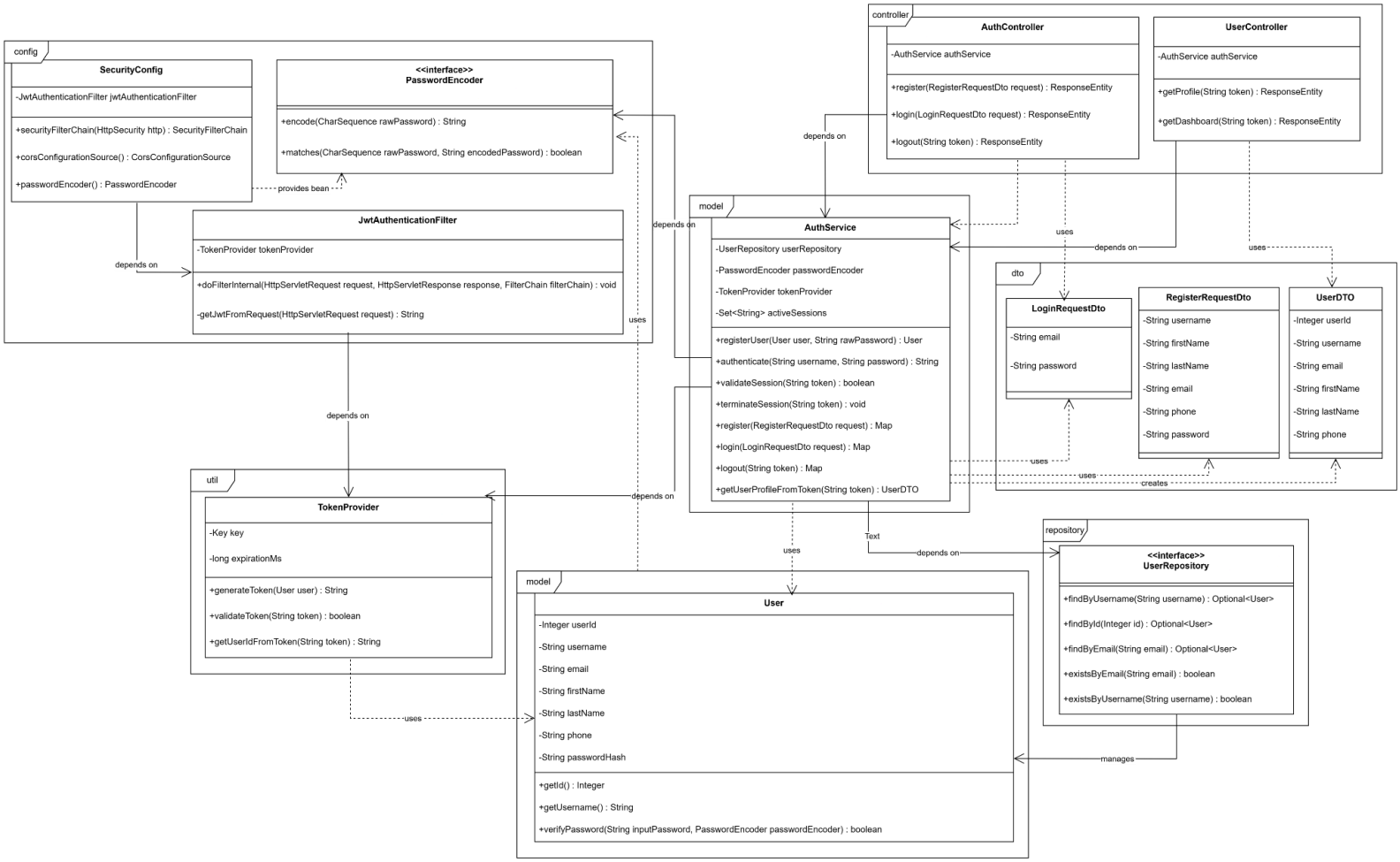
5.2. Use Case Diagram



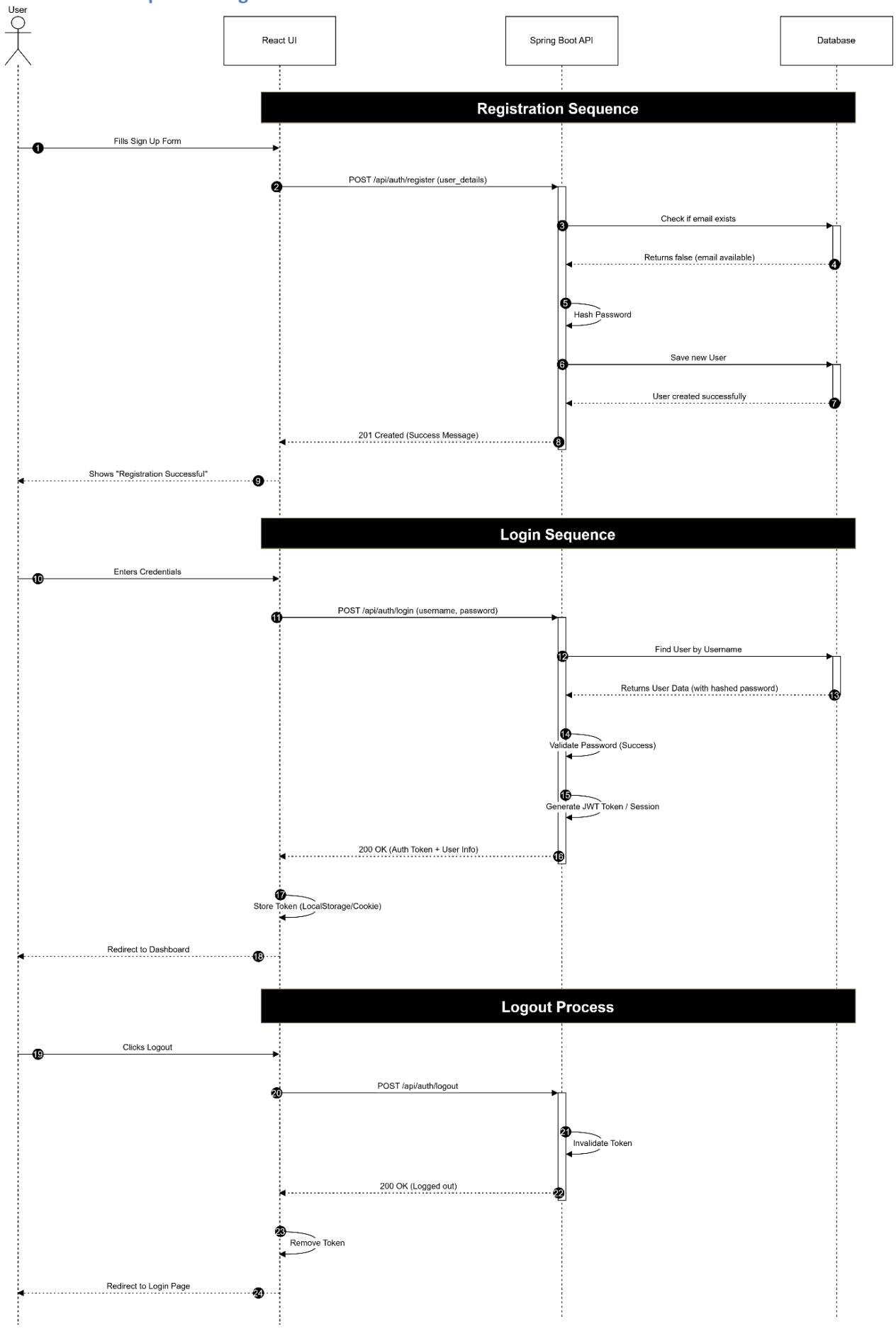
5.3. Activity Diagram



5.4. Class Diagram



5.5. Sequence Diagram



6. Appendices

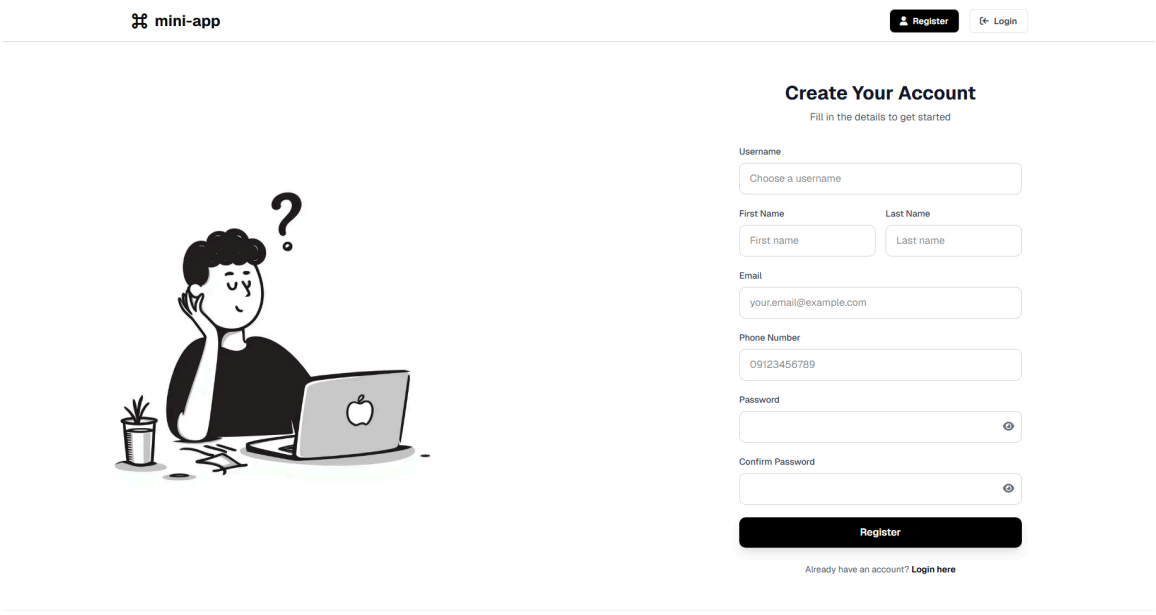
Appendix A: Database Script

The following SQL script defines the structure for the users table required for the authentication module:

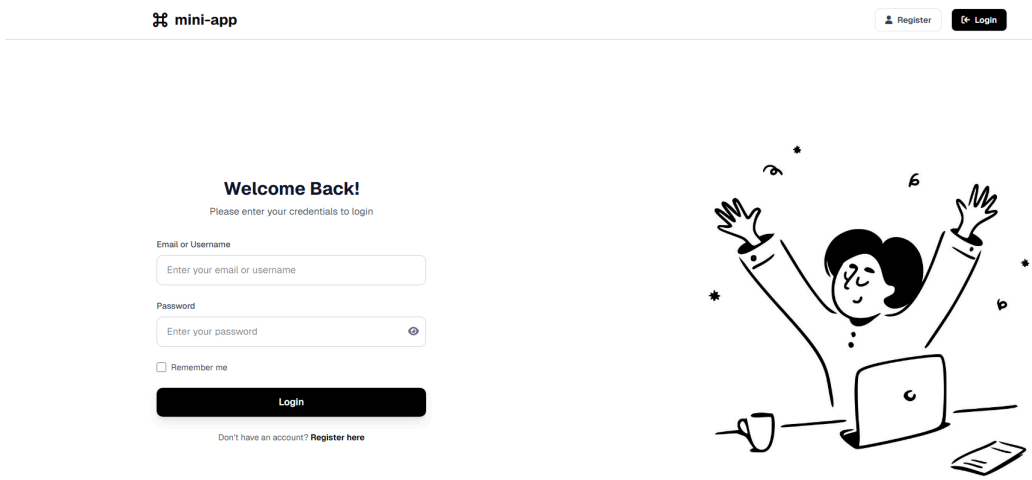
```
CREATE TABLE users (
    userId INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(50) NOT NULL UNIQUE,
    email VARCHAR(100) NOT NULL UNIQUE,
    firstName VARCHAR(50) NOT NULL,
    lastName VARCHAR(50) NOT NULL,
    phone VARCHAR(20),
    passwordHash VARCHAR(255) NOT NULL
);
```

Appendix B: Web Screenshots

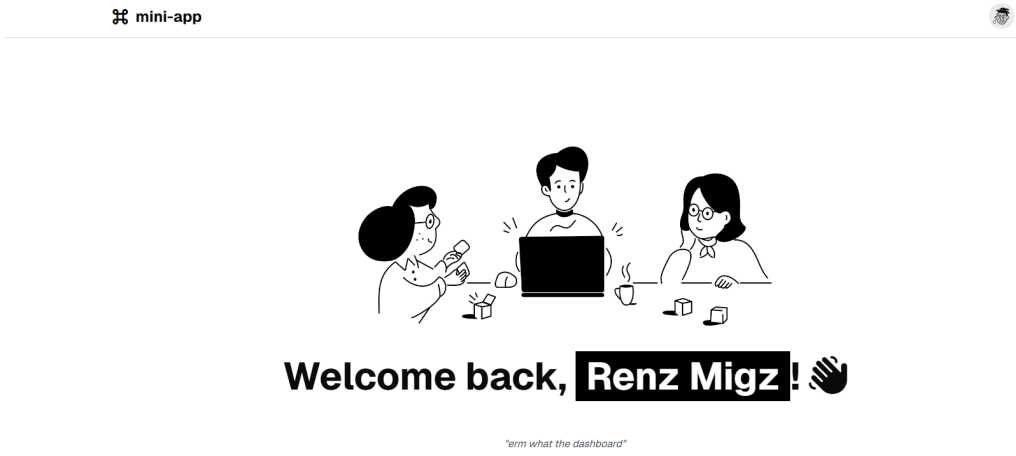
Register:



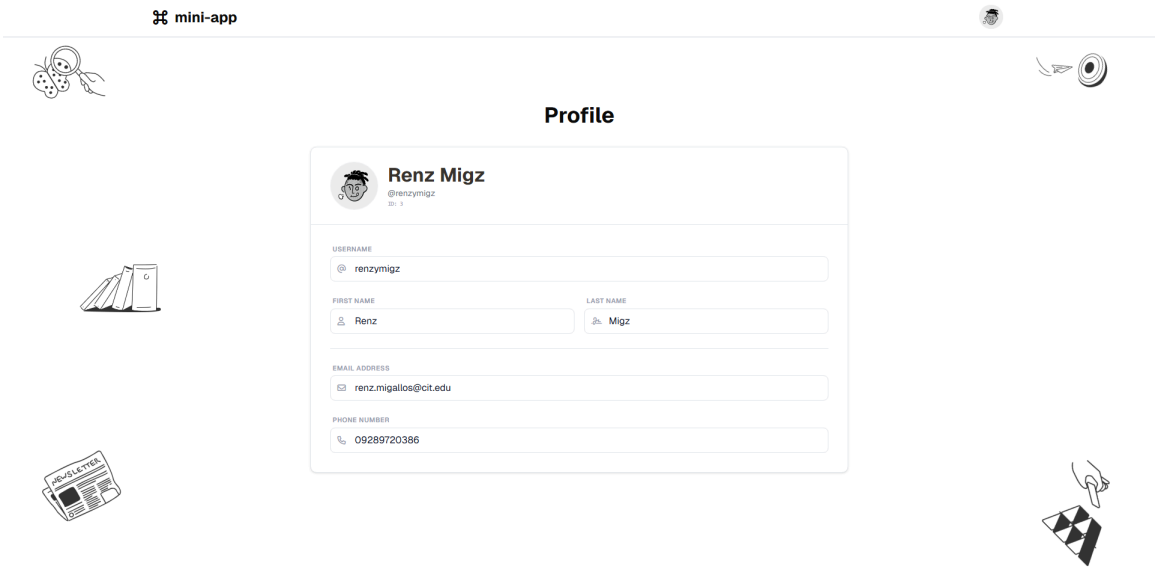
Login:



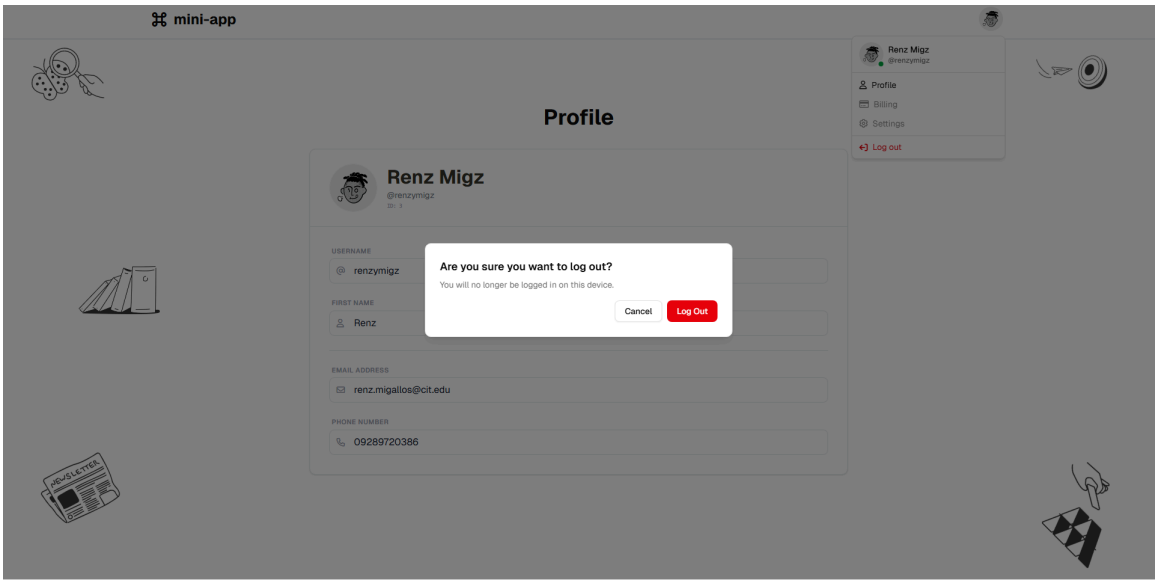
Dashboard:



Profile:

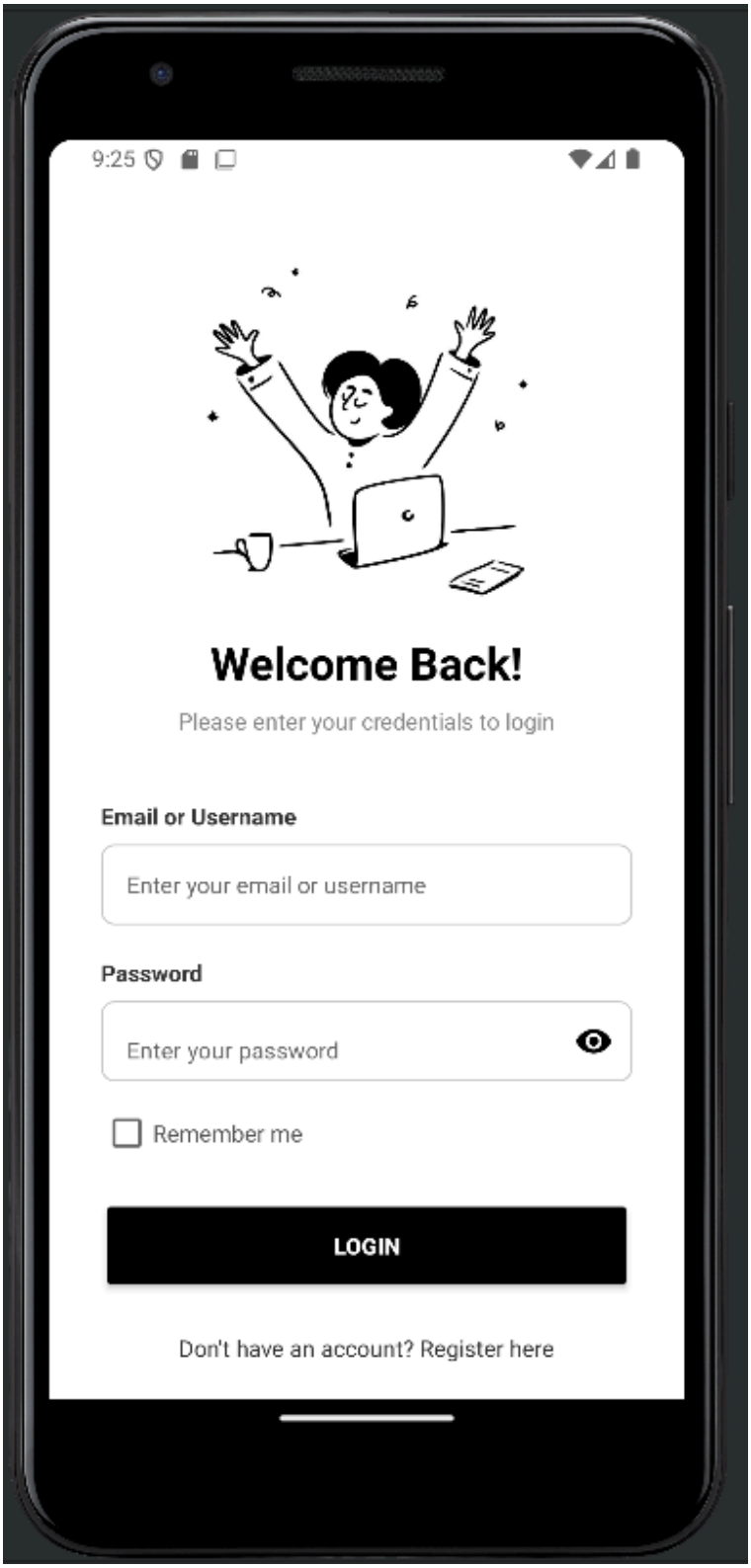


Logout:

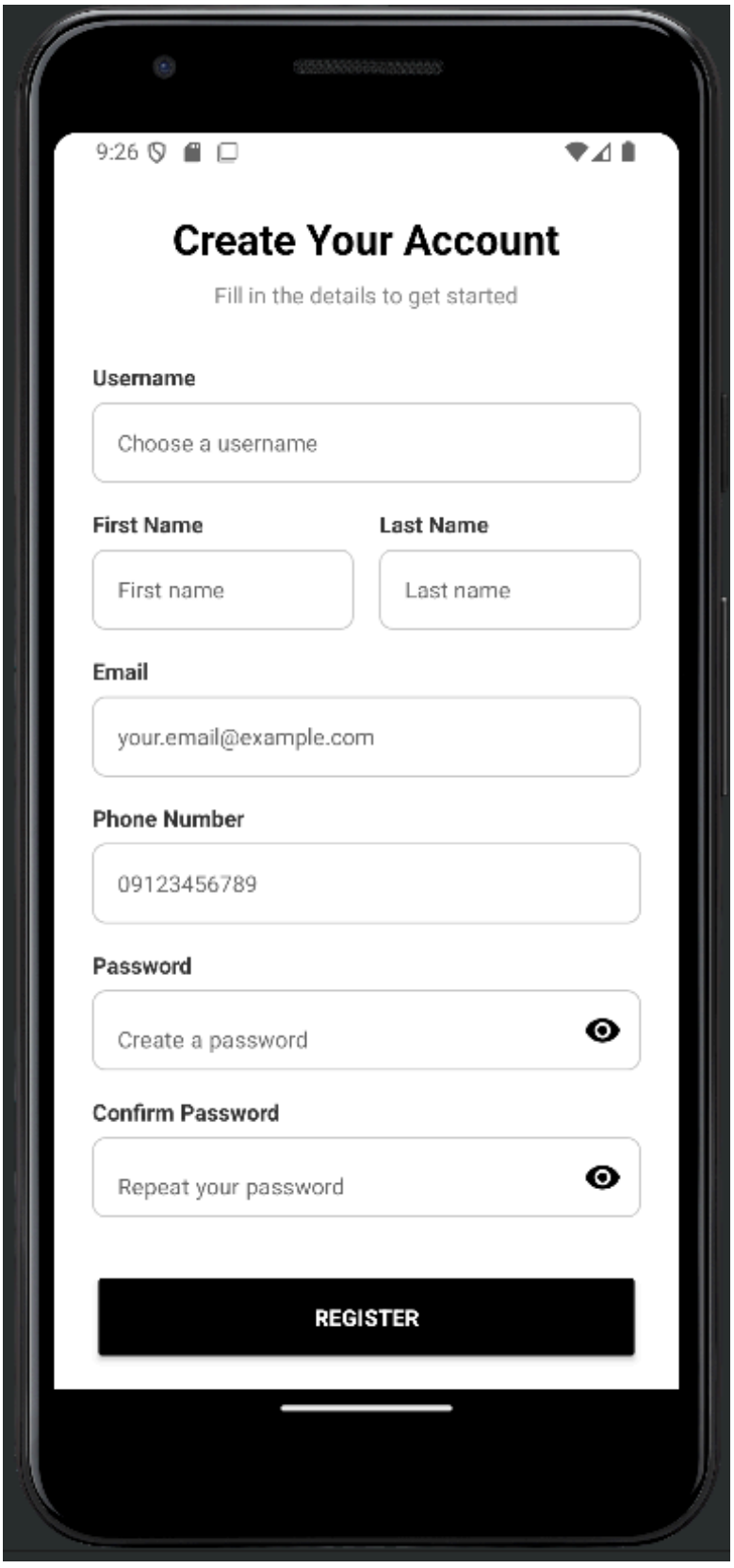


Appendix C: Mobile Screenshots

Login:



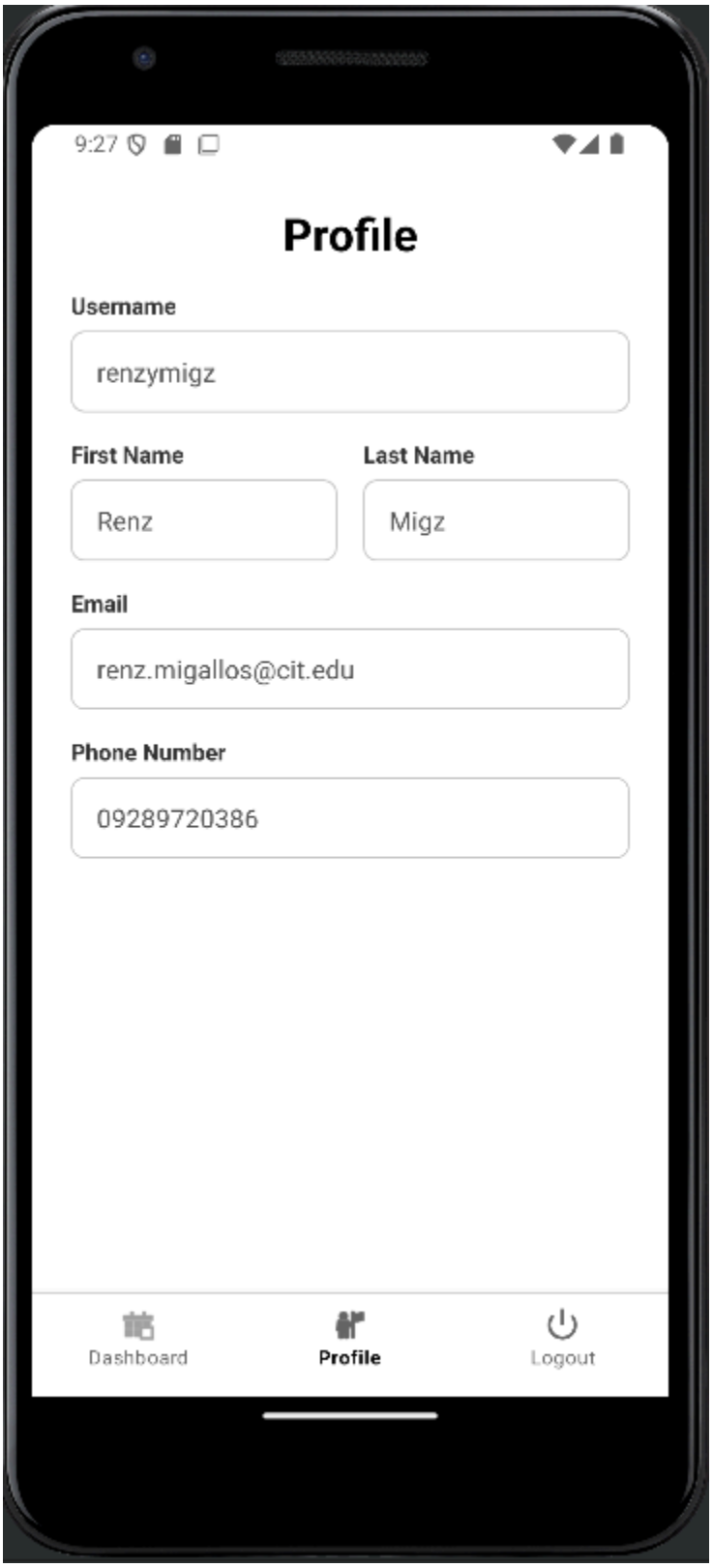
Register:


9:26
Create Your Account
Fill in the details to get started
Username
Choose a username
First Name
Last Name
First name
Last name
Email
your.email@example.com
Phone Number
09123456789
Password
Create a password
Confirm Password
Repeat your password
REGISTER

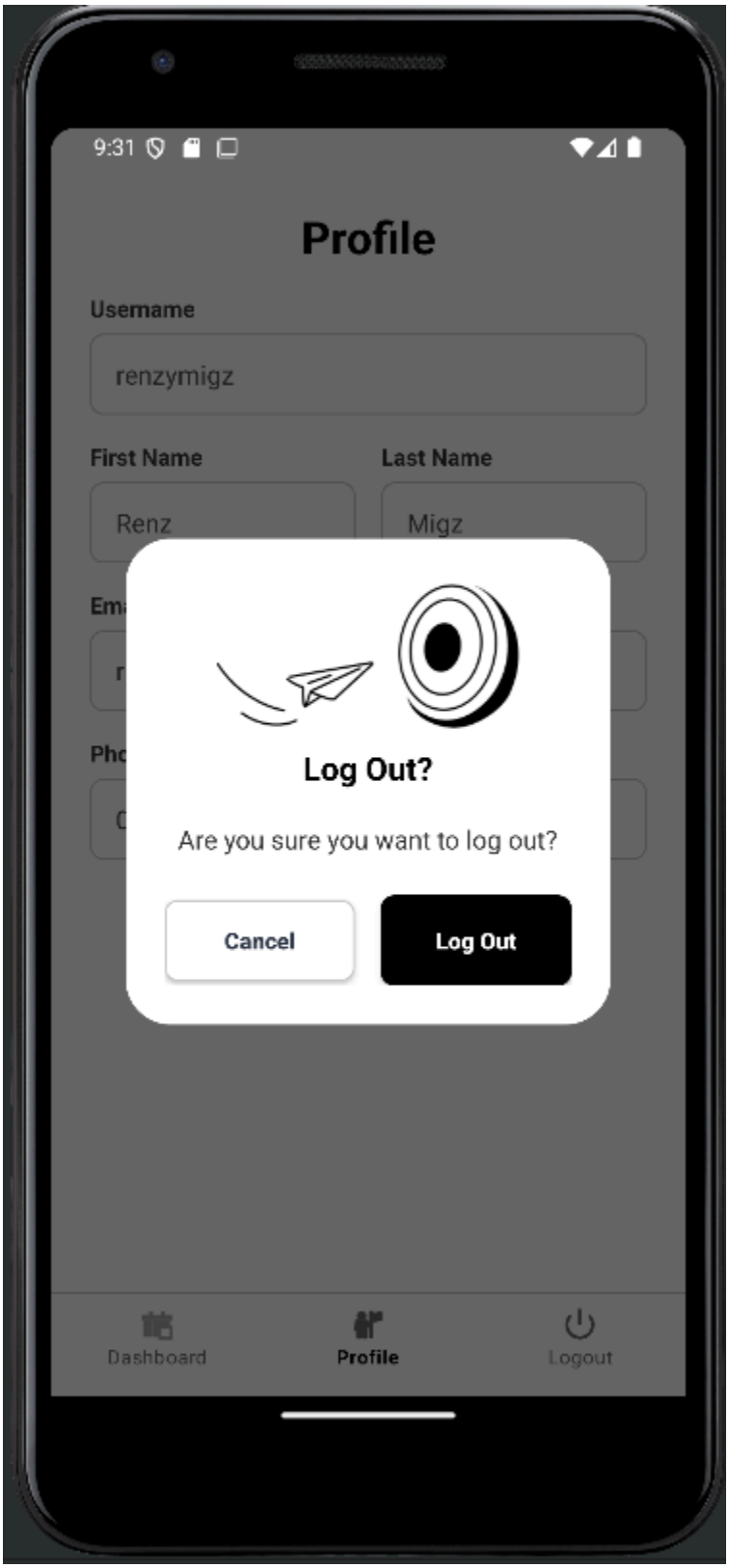
Dashboard:


9:27
Welcome back,
Renz Migz !
'erm what the dashboard xd"
Dashboard
Profile
Logout

Profile:


9:27
Profile
Username
renzymigz
First Name
Renz
Last Name
Migz
Email
renz.migallos@cit.edu
Phone Number
09289720386
Dashboard
Profile
Logout

Logout:


9:31
Profile
Username
renzymigz
First Name
Last Name
Renz
Migz
Log Out?
Are you sure you want to log out?
Cancel
Log Out
Dashboard
Profile
Logout