
CEBU INSTITUTE OF TECHNOLOGY
UNIVERSITY

# IT342-G1
# SYSTEMS INTEGRATION AND ARCHITECTURE 1

---

# FUNCTIONAL REQUIREMENTS SPECIFICATION (FRS)

---

Project Title: ResearchCenter

Prepared By: Emman Jay C. Uy

Date of Submission: 02/06/2026

Version: 1.1

# Table of Contents

# 1. Introduction

## 1.1. Purpose

This document defines the functional and non-functional requirements for the authentication module of ResearchCenter, which will be accessible through both a web application and an Android mobile application. It is intended for students, developers, and system designers who will implement and test user registration, login, dashboard and profile access, and logout functionalities. The purpose of this document is to ensure that the authentication system is clearly specified and that diagrams are accurately represented to serve as the basis for the coding phase.

## 1.2. Scope

ResearchCenter provides secure authentication services that allow users to create accounts, log in, view protected dashboard and profile pages, and log out. Only authenticated users can access restricted areas of the platform. The system is designed to be accessible through both a React-based web frontend and a Kotlin-based Android mobile application, with the backend implemented in Spring Boot and user data stored in Supabase. This FRS focuses exclusively on the authentication module, which will later integrate with the full ResearchCenter platform.

## 1.3. Definitions, Acronyms, and Abbreviations

| Term | Definition |
|---|---|
| JWT | JSON Web Token for secure authentication |
| Guest User | User not logged in or unregistered |
| Authenticated User | User who has logged in and can access protected content |

# 2. Overall Description

## 2.1. System Perspective

The authentication module functions as a standalone system that supports both web and mobile clients. It communicates with the backend via REST APIs implemented in Spring Boot, and all user data is stored in Supabase. The React web application runs on modern web browsers, while the Kotlin Android mobile application runs on Android devices version 8.0 and above.

### 2.2. User Classes and Characteristics

The system recognizes two types of users. Guest users are those who are not registered or not logged in and can only access the registration and login functionalities. Authenticated users have successfully logged in and can access the dashboard and profile pages.

### 2.3. Operating Environment

- Backend: Spring Boot (Java)
- Frontend: ReactJS (Web)
- Mobile: Android Kotlin (Android 8.0+)
- Database: Supabase (PostgreSQL)
- Version Control: Git and GitHub
- Deployment: Cloud hosting (Vercel / Railway / Render)

### 2.4. Assumptions and Dependencies

- Users have stable internet connectivity.
- The Supabase database must be running and accessible.
- The backend Spring Boot services must be operational for both web and mobile clients.
- JWT token-based authentication is implemented to ensure secure access for authenticated users.
- Web frontend (ReactJS) and mobile frontend (Android Kotlin) are compatible with supported browsers and devices (Android 8.0+).
- Cloud hosting platforms (Vercel, Railway, Render) are available and running for deployment.

## 3. System Features and Functional Requirements

### 3.1. Feature 1: User Registration

Description: The system allows new users to register accounts using either the web or mobile application.
Functional Requirements:
- The system shall allow users to register using a valid email and password.
- The system shall validate input fields and prevent registration with duplicate emails.
- The system shall securely hash passwords before storing them in the database.

### 3.2. Feature 2: User Login

Description: Registered users can log in through both web and mobile applications.
Functional Requirements:
- The system shall authenticate users using their email and password.
- The system shall generate a JWT token for successful logins.
- Invalid login attempts shall be rejected, and sessions shall be created for authenticated users.

### 3.3. Feature 3: User Dashboard

Description: Authenticated users can access the dashboard page after logging in.
Functional Requirements:
- Access restricted to authenticated users.
- Verify authentication tokens before granting access.

### 3.4. Feature 4: User Profile

Description: Authenticated users can view their profile information.
Functional Requirements:
- Access restricted to authenticated users.
- Display basic user information (username, email).
- Verify authentication tokens before granting access.

### 3.5. Feature 5: User Logout

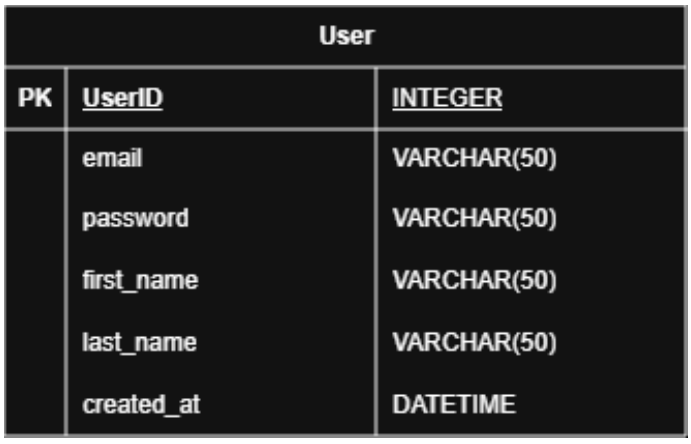
Description: Users can securely log out from both web and mobile platforms.
Functional Requirements:
- The system shall invalidate active authentication tokens.
- User sessions shall be destroyed upon logout.
- Users shall be redirected to the login page after logout.

## 4. Non-Functional Requirements

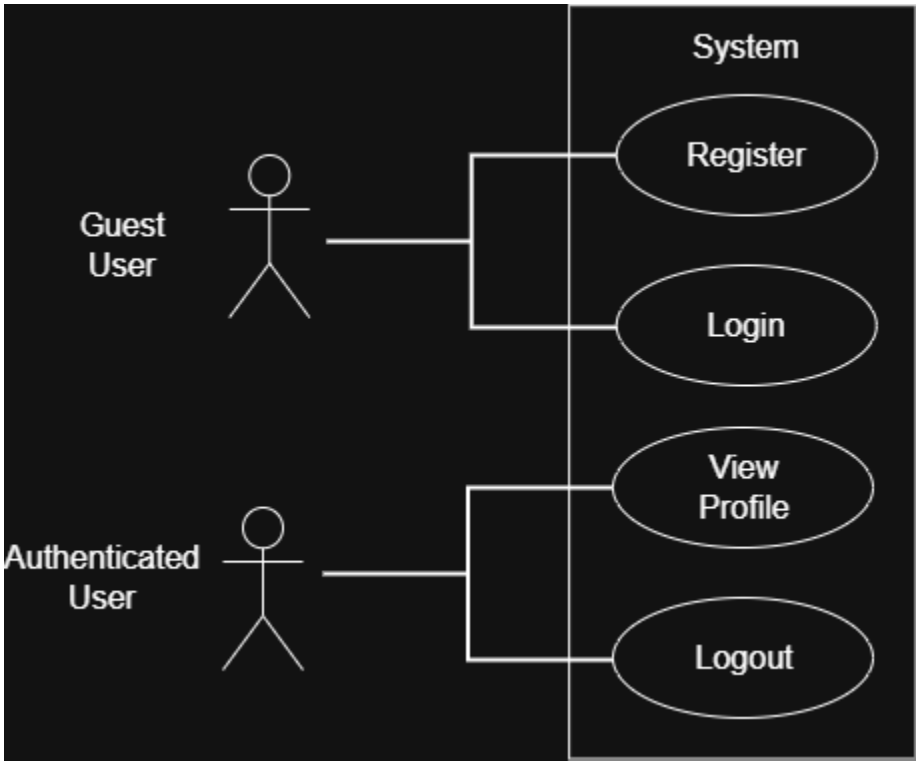
- Security: Passwords shall be hashed and authentication shall use JWT tokens.
- Performance: Login and registration responses shall occur within two seconds.
- Usability: User interface shall be simple, intuitive, and provide clear error messages.
- Reliability: System shall ensure data consistency and availability at all times.
- Scalability: System shall support future expansion and additional users.

## 5. System Models (Diagrams)

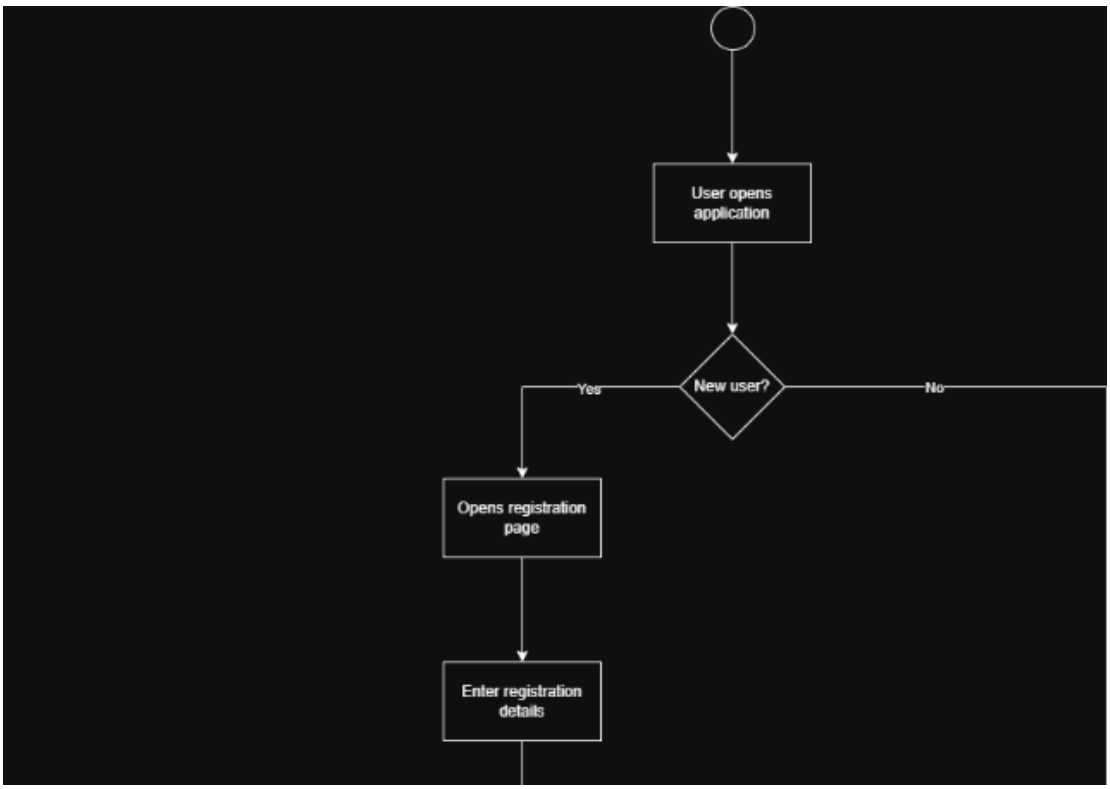
### 5.1. ERD

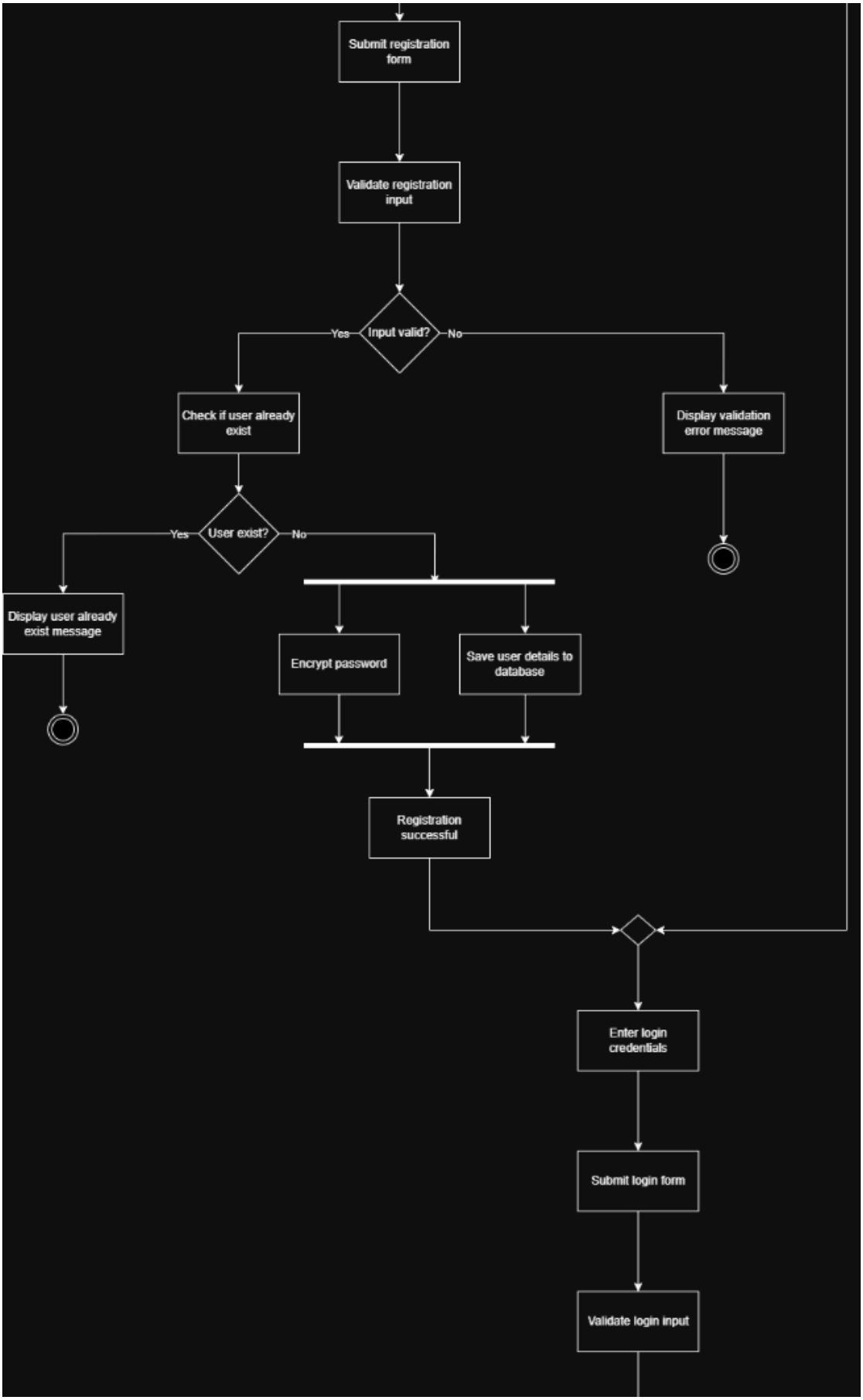


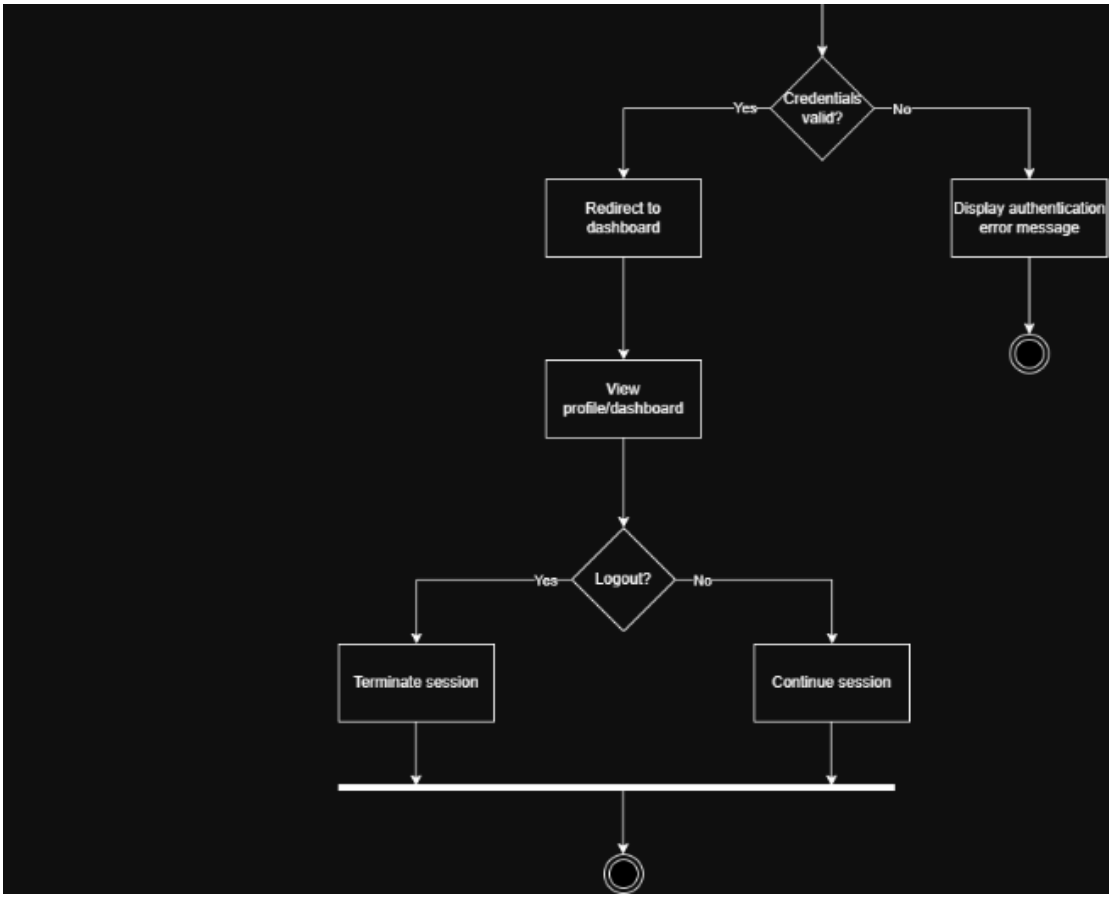
## 5.2. Use Case Diagram



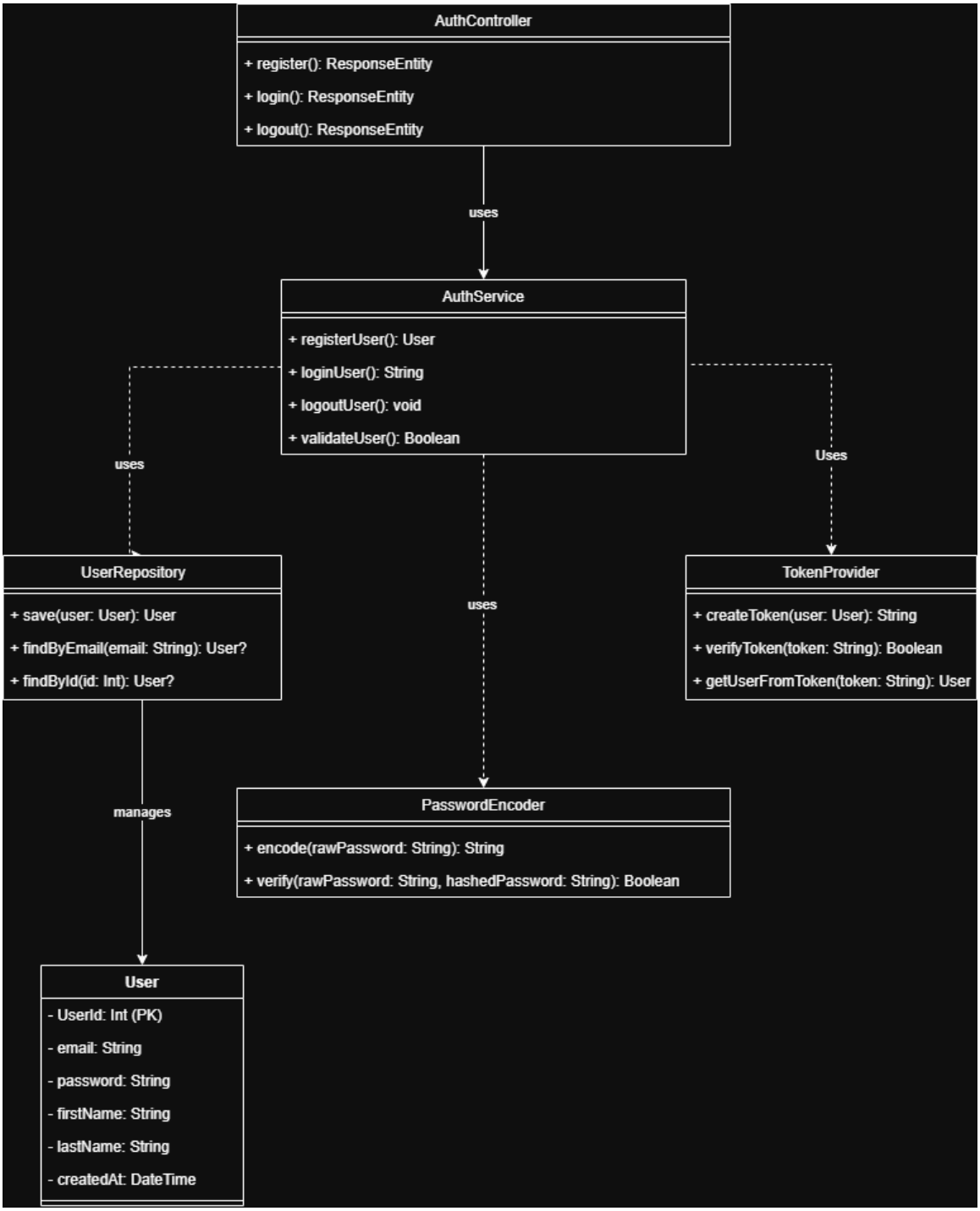

## 5.3. Activity Diagram



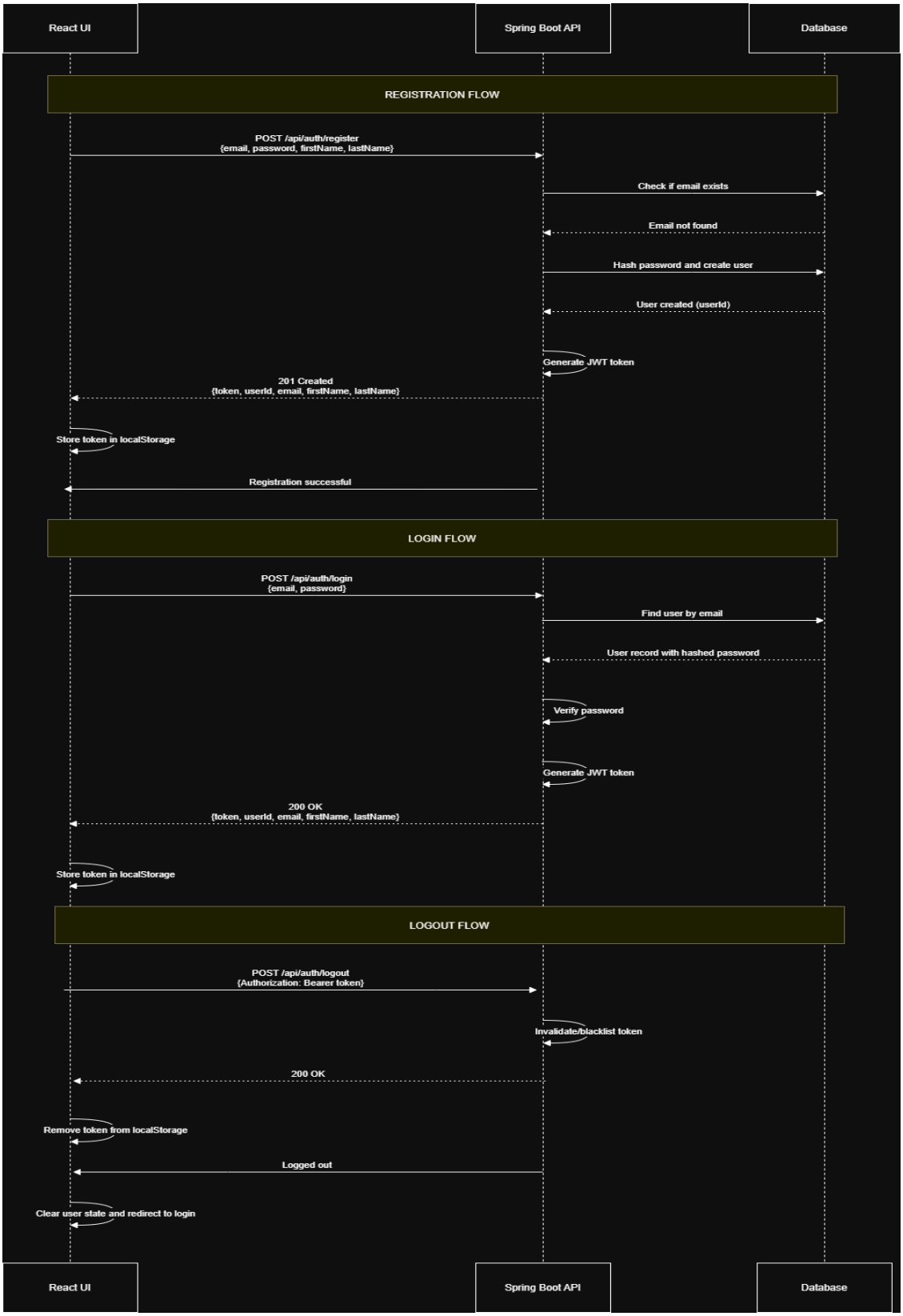

Submit registration form
Validate registration input
Input valid?
Yes
No
Check if user already exist
Display validation error message
User exist?
Yes
No
Display user already exist message
Encrypt password
Save user details to database
Registration successful
Enter login credentials
Submit login form
Validate login input


Credentials valid?
Yes
No
Redirect to dashboard
Display authentication error message
View profile/dashboard
Logout?
Yes
No
Terminate session
Continue session

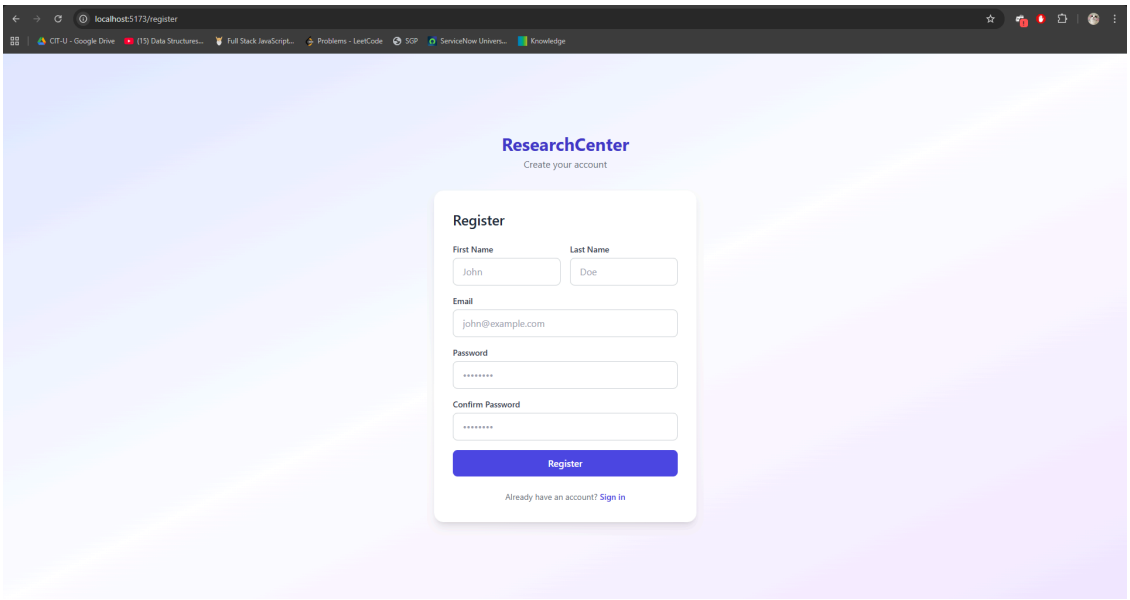
## 5.4. Class Diagram

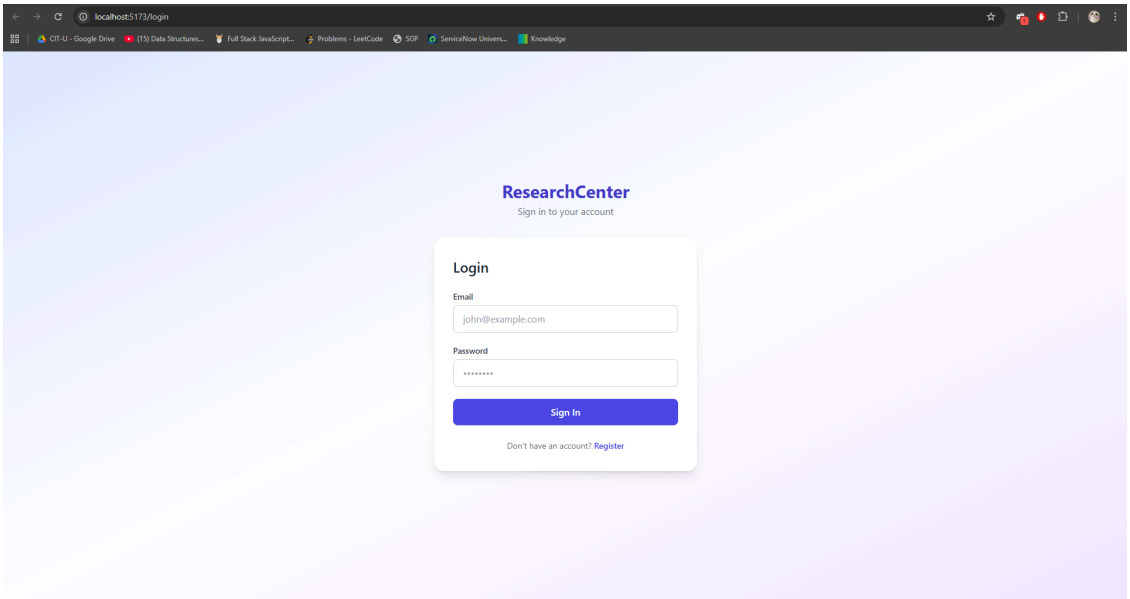


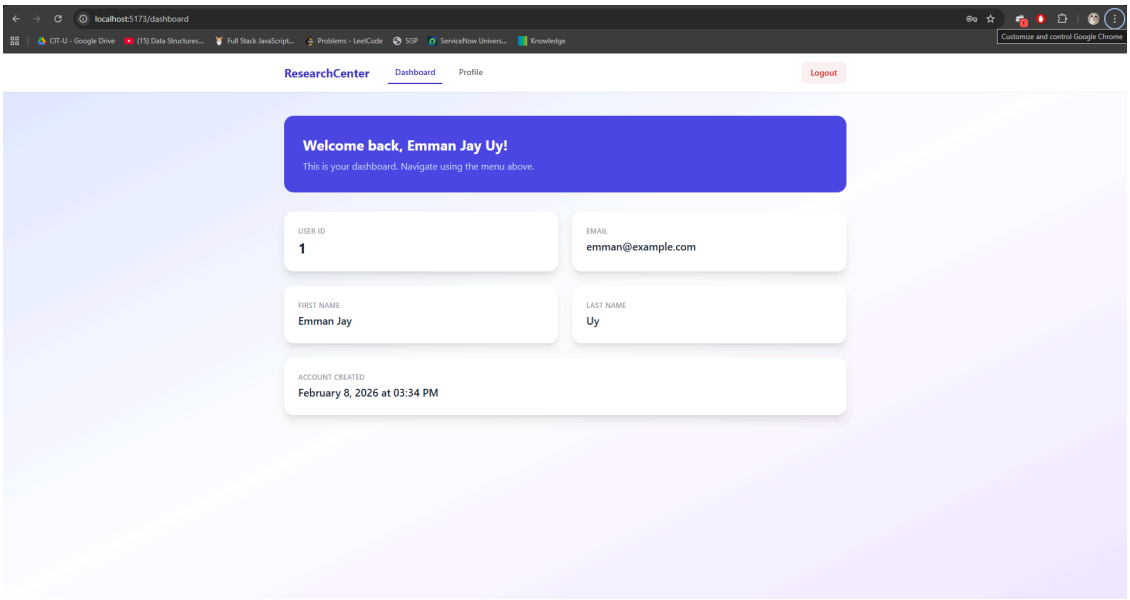
## 5.5. Sequence Diagram

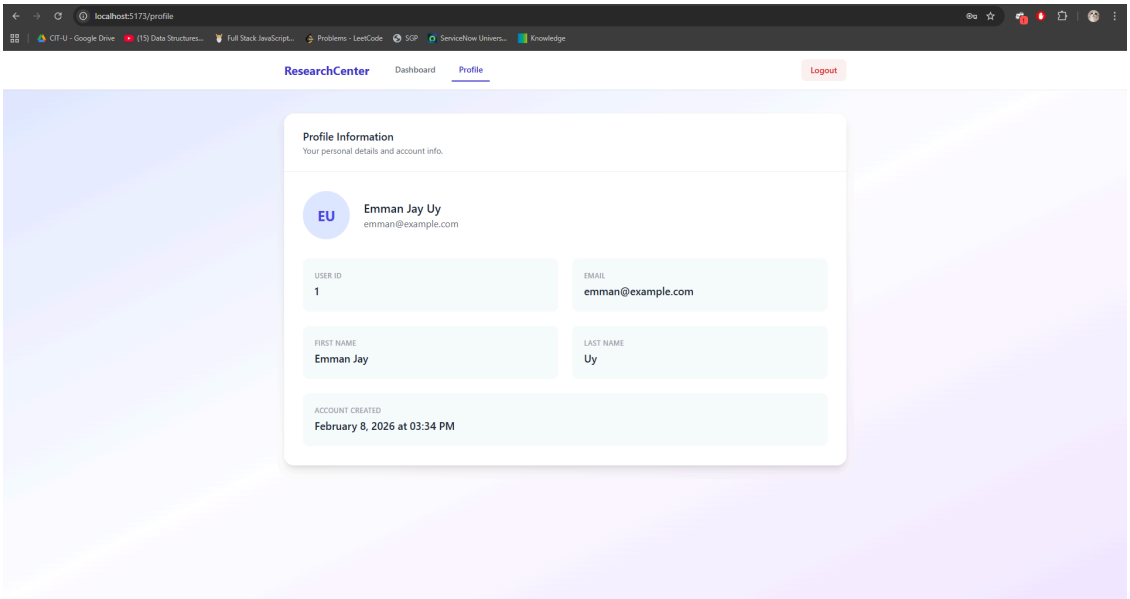


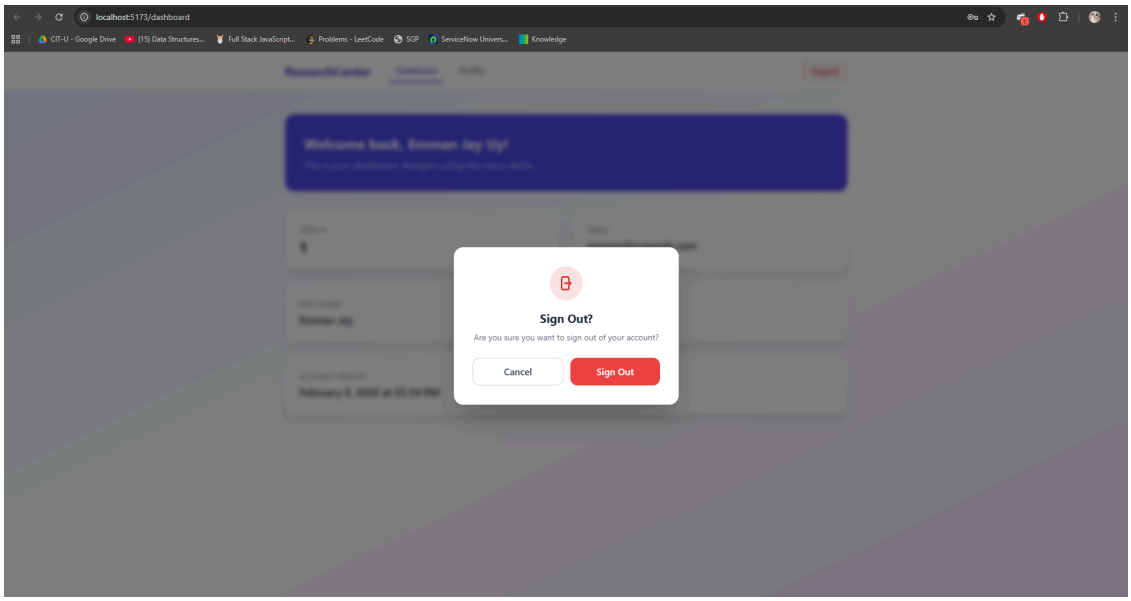
## Screenshots of the Web UI:




localhost:5173/dashboard
Customize and control Google Chrome
CIT-U - Google Drive
(15) Data Structures...
Full Stack JavaScript...
Problems - LeetCode
SGP
ServiceNow Univers...
Knowledge
ResearchCenter
Dashboard
Profile
Logout
Welcome back, Emman Jay Uy!
This is your dashboard. Navigate using the menu above.
USER ID
1
EMAIL
emman@example.com
FIRST NAME
Emman Jay
LAST NAME
Uy
ACCOUNT CREATED
February 8, 2026 at 03:34 PM


localhost:5173/profile
CIT-U - Google Drive
(15) Data Structures...
Full Stack JavaScript...
Problems - LeetCode
SGP
ServiceNow Univers...
Knowledge
ResearchCenter
Dashboard
Profile
Logout
Profile Information
Your personal details and account info.
EU
Emman Jay Uy
emman@example.com
USER ID
1
EMAIL
emman@example.com
FIRST NAME
Emman Jay
LAST NAME
Uy
ACCOUNT CREATED
February 8, 2026 at 03:34 PM




## 6. Appendices

*UML Class Diagram Tutorial*. (n.d.).

https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/

*Username/Password Authentication :: Spring Security*. (n.d.).

https://docs.spring.io/spring-security/reference/servlet/authentication/passwords/index.html

*What is Activity Diagram?* (n.d.).

https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/

*What is Entity Relationship Diagram (ERD)?* (n.d.).

https://www.visual-paradigm.com/guide/data-modeling/what-is-entity-relationship-diagram/

*What is Sequence Diagram?* (n.d.).

https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-sequence-diagram/

*What is Use Case Diagram?* (n.d.).

https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/