CEBU INSTITUTE OF TECHNOLOGY
UNIVERSITY

IT342-G2
SYSTEMS INTEGRATION AND ARCHITECTURE 1

FUNCTIONAL REQUIREMENTS SPECIFICATION (FRS)

Project Title: HealthGate

Prepared By: Vein Carmell G. Pangilinan

Date of Submission: 02/09/2026

Version: 3.0

Table of Contents

1. Introduction 3
   1.1. Purpose 3
   1.2. Scope 3
   1.3. Definitions, Acronyms, and Abbreviations 3
2. Overall Description 3
   2.1. System Perspective 3
   2.2. User Classes and Characteristics 3
   2.3. Operating Environment 3
   2.4. Assumptions and Dependencies 3
3. System Features and Functional Requirements 3
   3.1. Feature 1: 3
   3.2. Feature 2: 3
4. Non-Functional Requirements 3
5. System Models (Diagrams) 4
   5.1. ERD 4
   5.2. Use Case Diagram 4
   5.3. Activity Diagram 4
   5.4. Class Diagram 4
   5.5. Sequence Diagram 4
6. Appendices 4

1. Introduction

1.1. Purpose

This document defines the functional and non-functional requirements of the HealthGate system. It is intended for students, instructors, developers, and system designers who will design, implement, and evaluate the user registration and authentication system.

1.2. Scope

HealthGate is a secure authentication system that allows users to register, log in, view a protected dashboard, and log out. It ensures that only authenticated users can access protected pages. The system focuses on authentication and user management only.

1.3. Definitions, Acronyms, and Abbreviations

List and define important terms used in this document.

2. Overall Description

2.1. System Perspective

HealthGate is a standalone authentication module that can be integrated into larger systems such as healthcare apps, clinic systems, or service platforms. It communicates between a frontend client and backend server through secure APIs.

2.2. User Classes and Characteristics

a. **Guest User** – Unregistered or logged-out user; can register and log in.
b. **Authenticated User** – Logged-in user; can access dashboard and profile.

2.3. Operating Environment

Frontend: React Web Application

Backend: Spring Boot REST API

Database: MySQL / PostgreSQL

Tools: VS Code, Postman, Git, Draw.io (Diagrams.net)

2.4. Assumptions and Dependencies

- Users have internet access.
- The system depends on database availability.
- Secure token authentication (JWT) is implemented.
- Backend services must be running for system operation.

3. System Features and Functional Requirements

3.1. Feature 1: User Registration

Description: Allows new users to create an account.
Functional Requirements:

- The system shall allow users to register using email and password.
- The system shall validate input fields.
- The system shall encrypt passwords before storage.
- The system shall prevent duplicate email registration.

3.2. Feature 2: User Login

Description: Allows registered users to log in.
Functional Requirements:

- The system shall authenticate users using credentials.
- The system shall generate authentication tokens.
- The system shall reject invalid login attempts.
- The system shall create user sessions.

3.3. Feature 3: User Dashboard/Profile

Description: Allows logged-in users to access protected content.
Functional Requirements:

- The system shall restrict access to authenticated users only.
- The system shall display user profile information.
- The system shall verify authentication tokens.

3.4. Feature 4: User Logout

Description: Allows users to securely log out.
Functional Requirements:

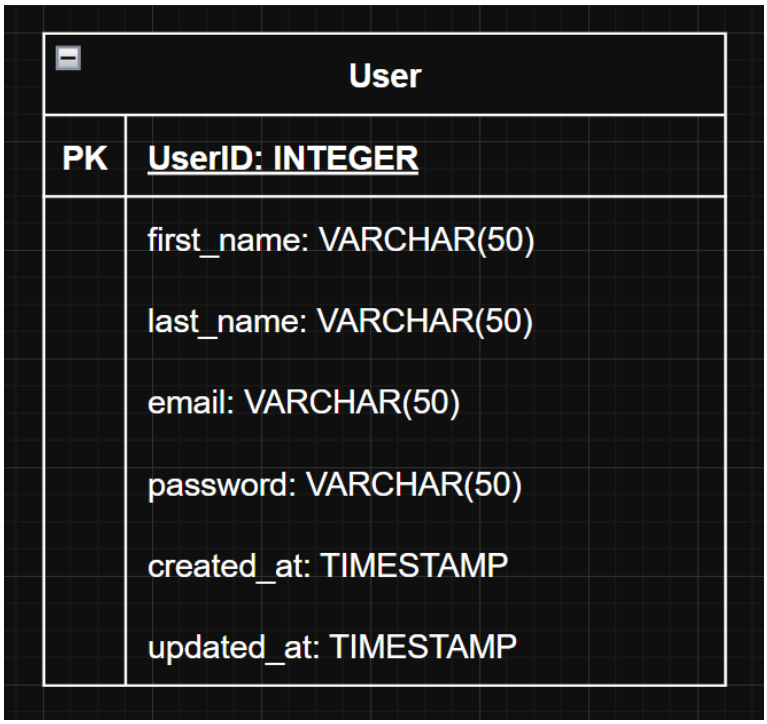
- The system shall invalidate authentication tokens.
- The system shall destroy active sessions.
- The system shall redirect users to login page.

4. Non-Functional Requirements

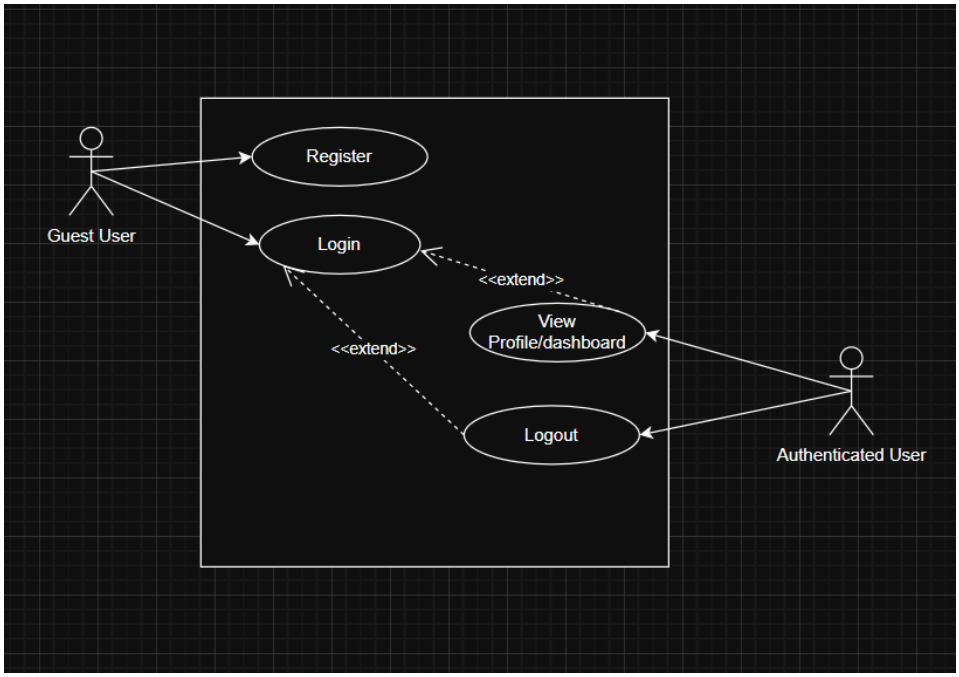
- **Security:** Password hashing, token-based authentication, secure APIs.
- **Performance:** Login and registration responses under 2 seconds.
- **Usability:** Simple UI and clear error messages.
- **Reliability:** System uptime and data consistency.
- **Scalability:** Supports future expansion and more users.

5. System Models (Diagrams)

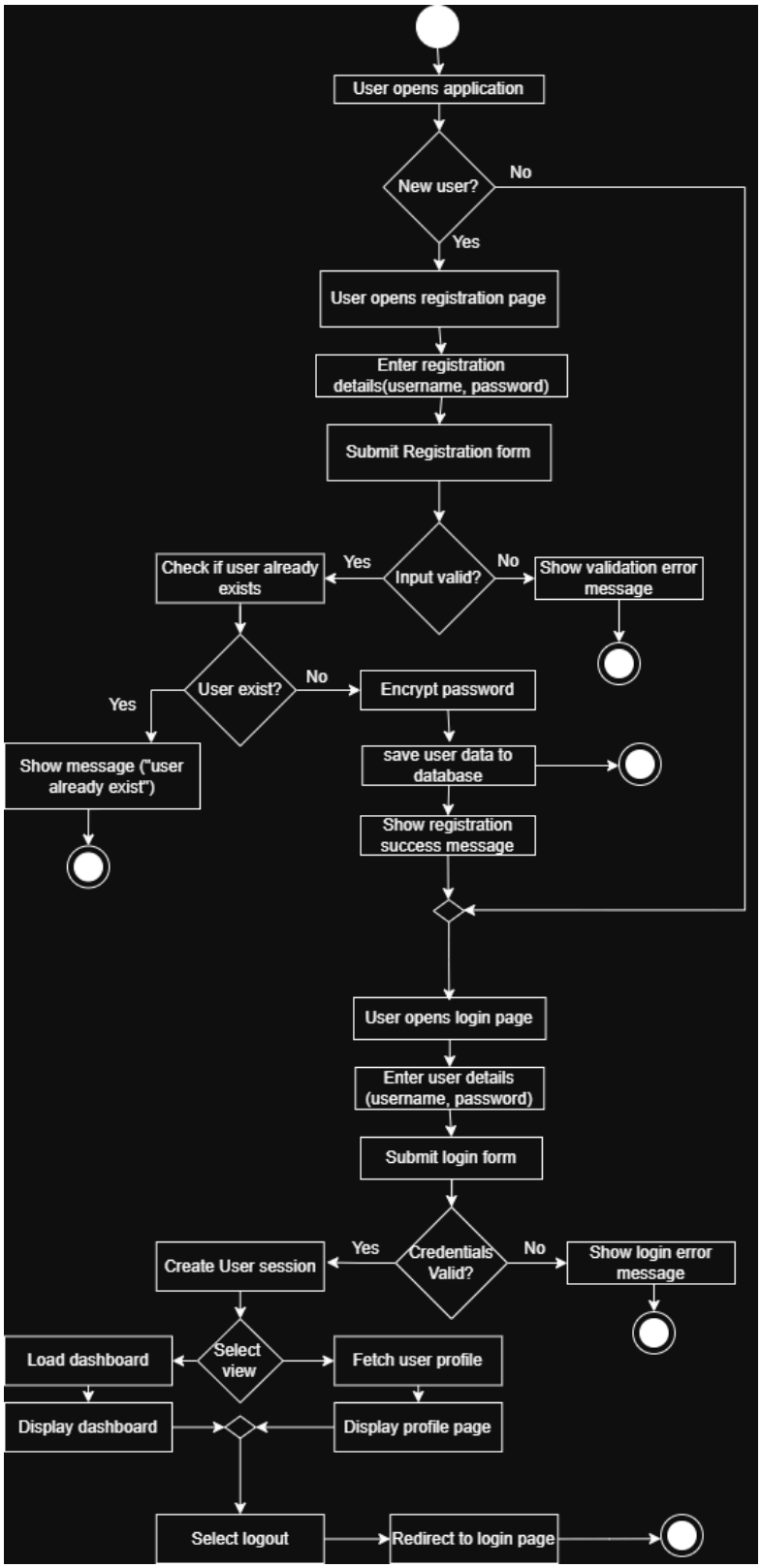
5.1. ERD



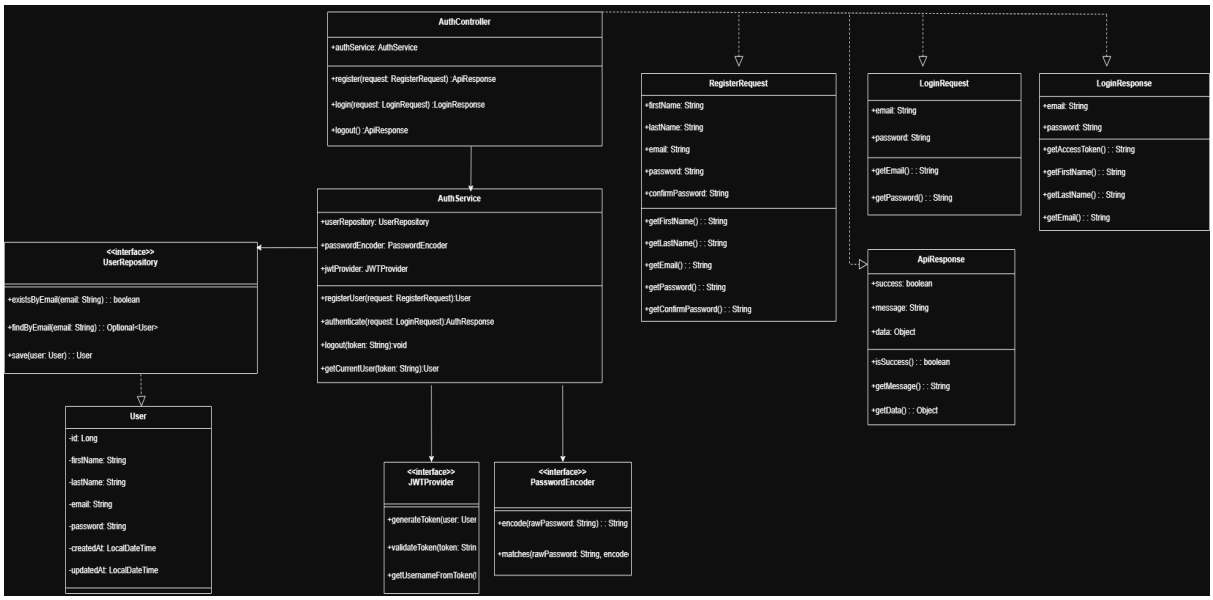
5.2. Use Case Diagram



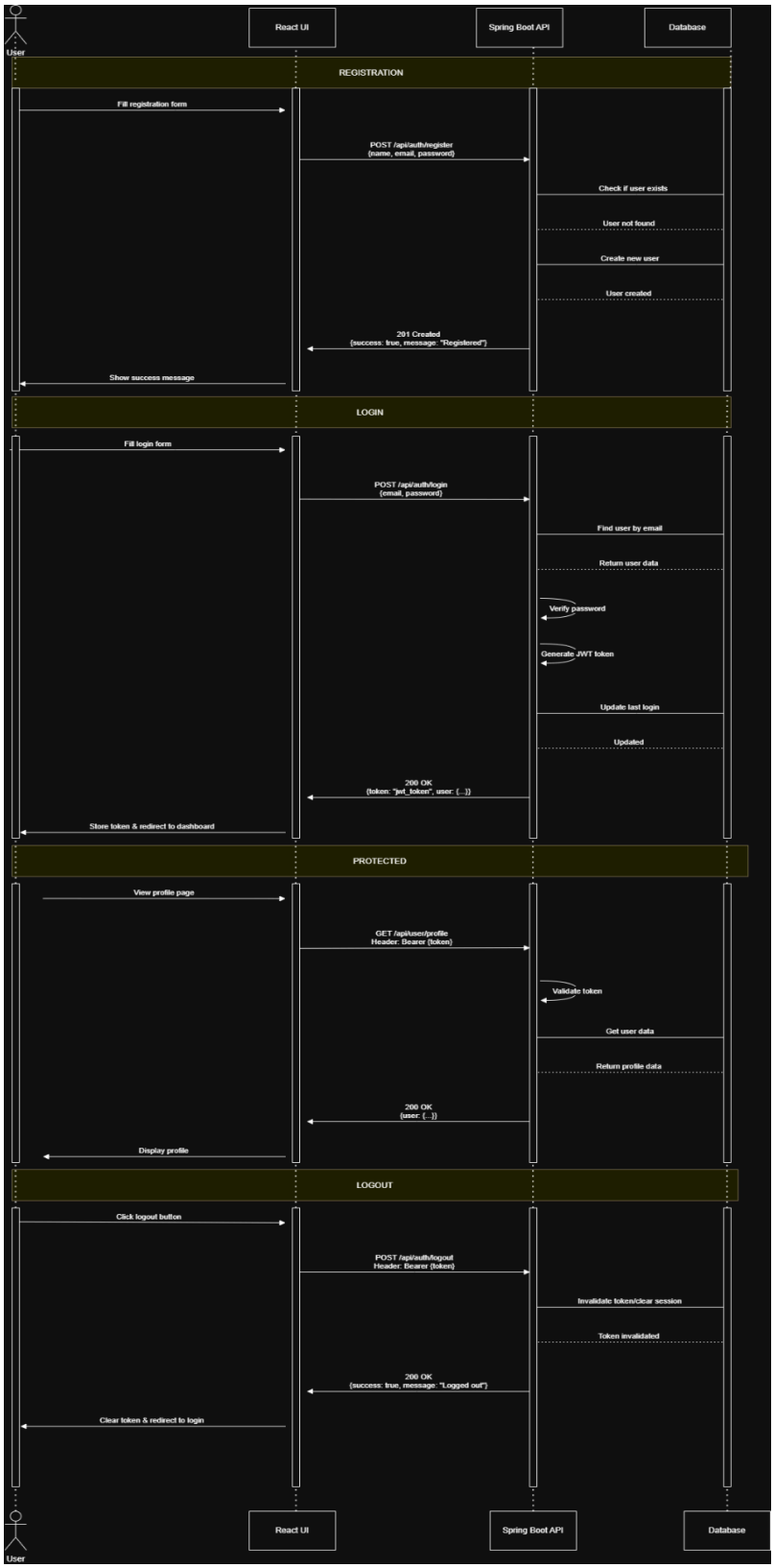
5.3. Activity Diagram



5.4. Class Diagram

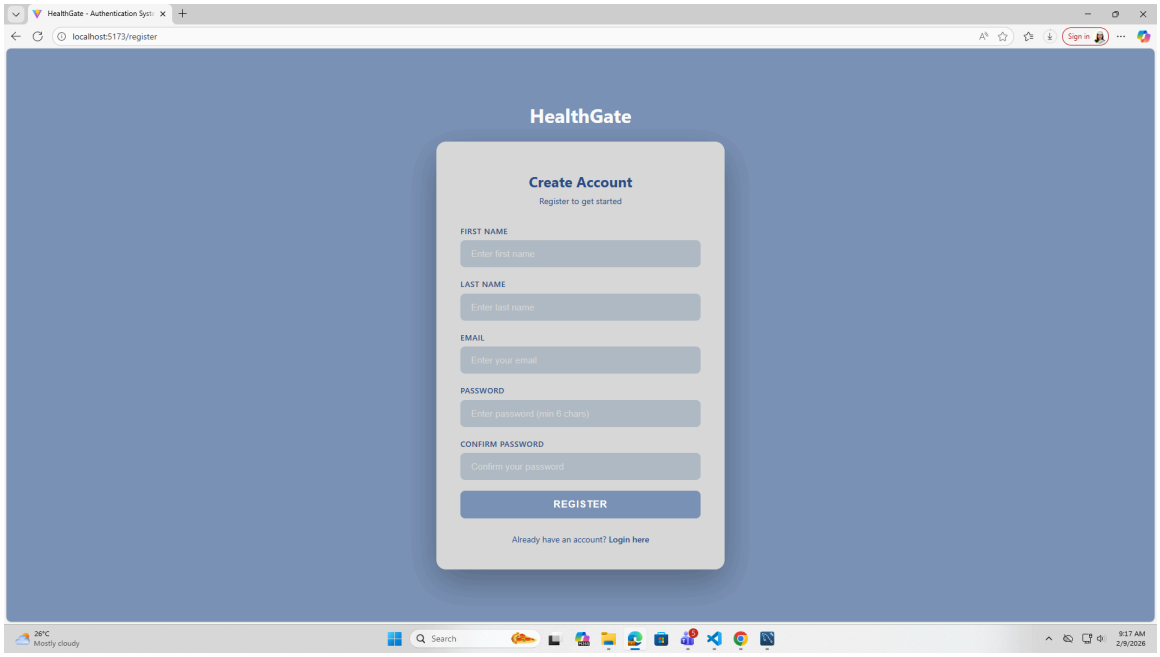


5.5. Sequence Diagram

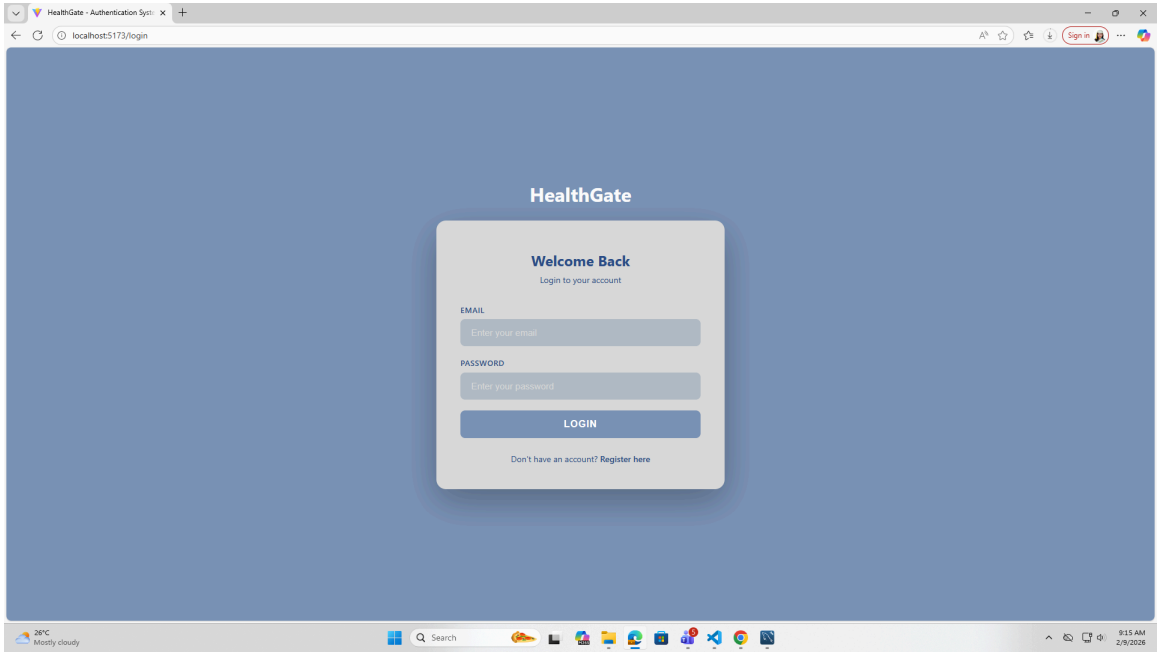


6. Screenshots of the Web UI:

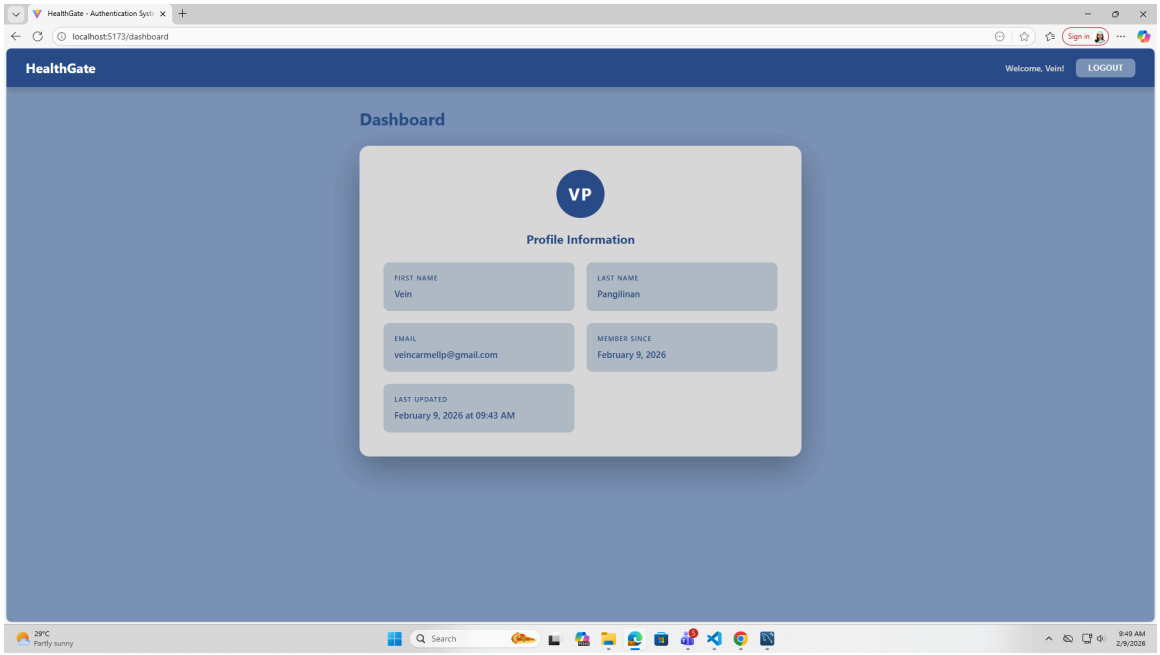
6.1. Register



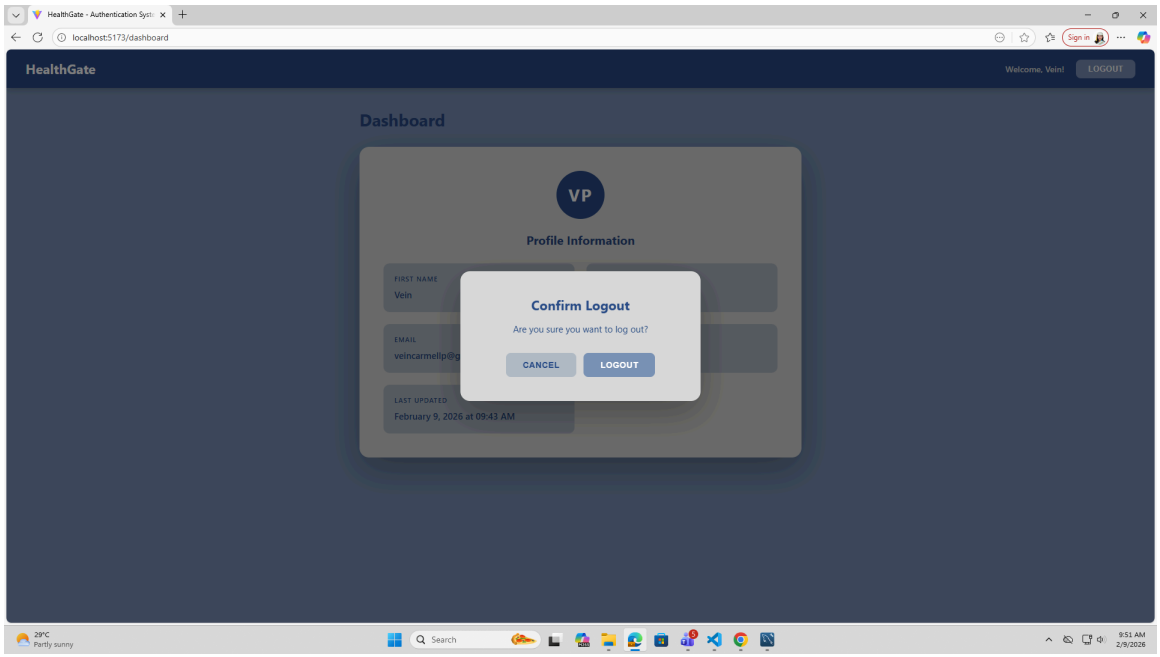
6.2. Login



6.3 Dashboard/Profile



6.4 Logout



7. Appendices

GeeksforGeeks. (2026, January 21). *Sequence diagrams Unified Modeling Language (UML)*. GeeksforGeeks. https://www.geeksforgeeks.org/system-design/unified-modeling-language-uml-sequence-diagrams/

GeeksforGeeks. (2026a, January 21). *Activity Diagrams Unified Modeling Language (UML)*. GeeksforGeeks. https://www.geeksforgeeks.org/system-design/unified-modeling-language-uml-activity-diagrams/