



CEBU INSTITUTE OF TECHNOLOGY
U N I V E R S I T Y

IT342-G4 SYSTEMS INTEGRATION AND ARCHITECTURE 1

FUNCTIONAL REQUIREMENTS SPECIFICATION (FRS)

Project Title: MedLink

Prepared By: Josan B. Sumarago

Date of Submission: February 3, 2026

Version:

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	4
2.3.	Operating Environment	4
2.4.	Assumptions and Dependencies	4
3.	System Features and Functional Requirements	4
3.1.	Feature 1:	4
3.2.	Feature 2:	5
4.	Non-Functional Requirements	5
5.	System Models (Diagrams).....	6
5.1.	ERD	7
5.2.	Use Case Diagram.....	8
5.3.	Activity Diagram	9
5.4.	Class Diagram	10
5.5.	Sequence Diagram	11
6.	Appendices.....	11

1. Introduction

1.1. Purpose

The purpose of this document is to define the requirements for the **MedLink System**, a web-based healthcare management portal. This document is intended for the project stakeholders, developers (Backend/Frontend), and quality assurance testers to understand the system's functionality, scope, and constraints.

1.2. Scope

The MedLink System is a web application designed to facilitate user access to healthcare services. The initial scope includes:

- Secure user registration and account creation.
- Authentication mechanisms (Login and Logout) using standard security protocols.
- A protected dashboard for authenticated users to view their profile.
- The system acts as a standalone module but is designed to expand into appointment booking and medical record management in future phases.

1.3. Definitions, Acronyms, and Abbreviations

- SRS: Software Requirements Specification.
- API: Application Programming Interface (The backend communication layer).
- JWT: JSON Web Token (Used for secure authentication).
- UI: User Interface (The React-based frontend).
- DBMS: Database Management System (e.g., MySQL or PostgreSQL).
- DTO: Data Transfer Object.

.

2. Overall Description

2.1. System Perspective

MedLink functions as a Client-Server application.

- Frontend: A responsive web interface (React.js) accessible via standard web browsers.
- Backend: A RESTful API (Spring Boot) that processes logic and connects to the database.

- Database: A relational database that stores user credentials and profile information.

2.2. User Classes and Characteristics

User Classes and Characteristics

1. Guest User: An unauthenticated visitor who can only access public pages (Landing Page, Login, Registration).
2. Authenticated User (Patient/Admin): A registered user who has successfully logged in. They have access to protected features like the Dashboard/Profile and Logout functions.

2.3. Operating Environment

- Client Side: Modern web browser (Chrome, Firefox, Edge, Safari) with JavaScript enabled.
- Server Side: Java Runtime Environment (JRE) 17 or higher.
- Database: MySQL or PostgreSQL server.
- Network: Stable internet connection for API communication.

2.4. Assumptions and Dependencies

- Assumption: Users have a valid email address for registration.
- Assumption: The browser supports local storage or cookies for session management.
- Dependency: The system relies on the availability of the hosting server and database connectivity.

3. System Features and Functional Requirements

Describe each major feature of the system and its functional requirements.

3.1. Feature 1:

Description: Allows a Guest User to create a new account by providing necessary personal details.

Functional Requirements:

- FR-01: The system shall provide a registration form requiring Name, Email, and Password.
- FR-02: The system shall validate that the email is in a correct format and does not already exist in the database.
- FR-03: The system shall encrypt the user's password using a hashing algorithm (e.g., BCrypt) before saving it to the database.
- FR-04: Upon successful registration, the system shall redirect the user to the Login page.

3.2. Feature 2: User Authentication (Login)

Description: Allows registered users to access their accounts securely using their credentials.

Functional Requirements:

- FR-05: The system shall validate the user's email and password against stored records.
- FR-06: The system shall generate a secure session token (JWT) upon successful login.
- FR-07: The system shall display an error message for invalid credentials (e.g., "Invalid email or password").
- FR-08: The system shall redirect the user to the Dashboard upon successful login.

3.3. Feature 3: Profile Management & Logout

Description: Allows authenticated users to view their dashboard and securely end their session.

Functional Requirements:

- FR-09: The system shall display the User Dashboard only to users holding a valid active session token.
- FR-10: The Dashboard shall display the user's name and role retrieved from the backend.
- FR-11: The system shall provide a "Logout" button that invalidates the user's session token or clears client-side storage.
- FR-12: Upon logout, the user shall be immediately redirected to the Login page/Landing page.

4. Non-Functional Requirements

Security:

- Passwords must never be stored in plain text; they must be hashed.
- API endpoints for the Profile must be protected against unauthorized access (401 Unauthorized).

Performance:

- The system shall respond to login/registration requests within 2 seconds under normal load.

Reliability:

- The system should be available 99% of the time during business hours.

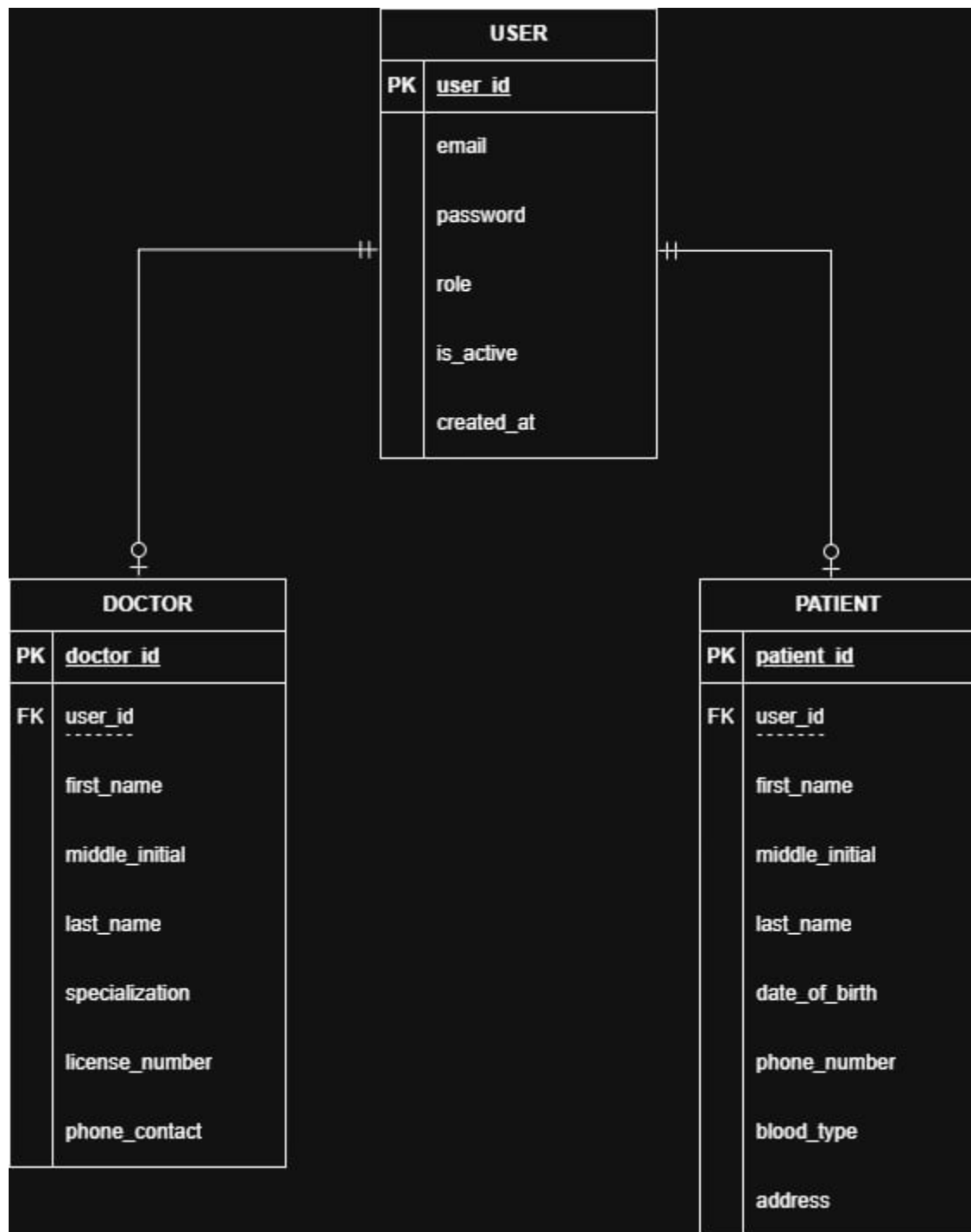
Usability:

- The user interface shall be responsive and usable on both desktop and mobile screens.
- Error messages shall be clear and user-friendly (non-technical).

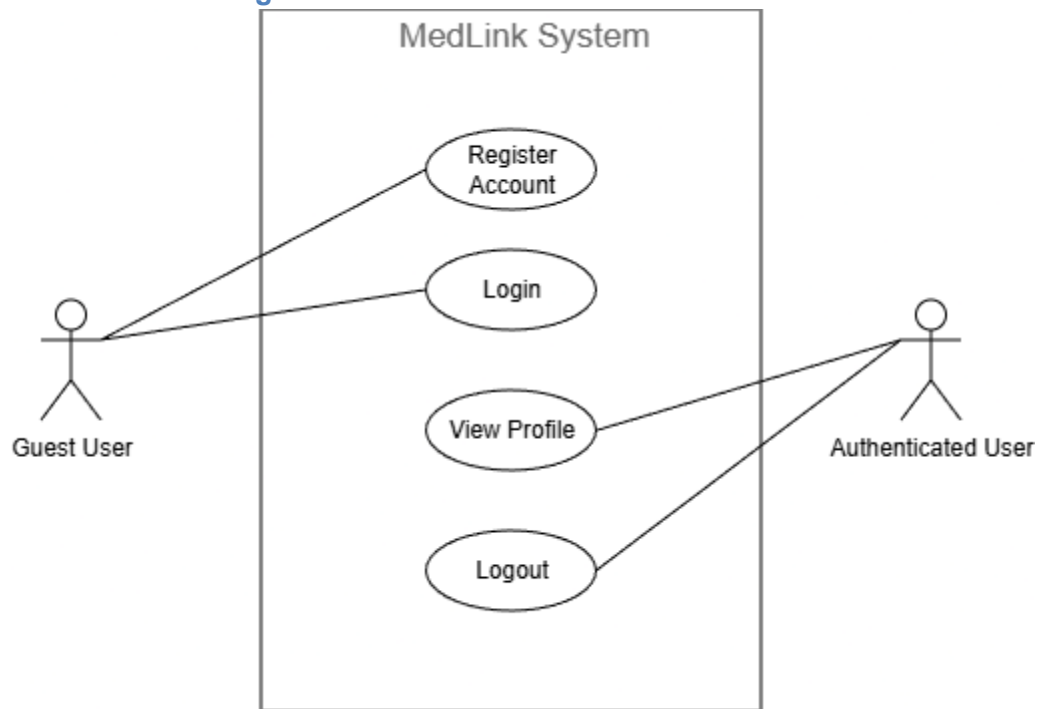
5. System Models (Diagrams)

Insert the necessary diagrams for the system:

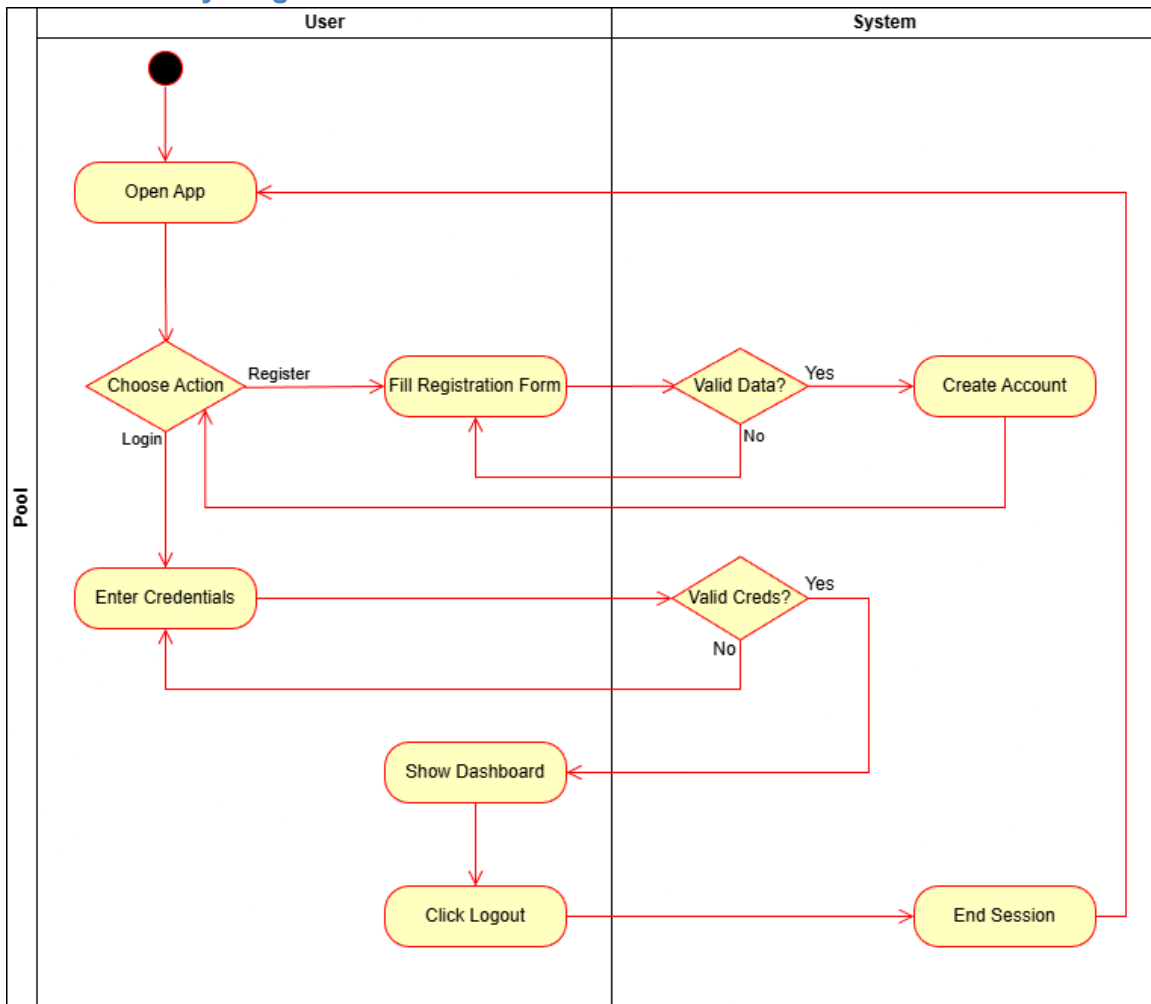
5.1. ERD



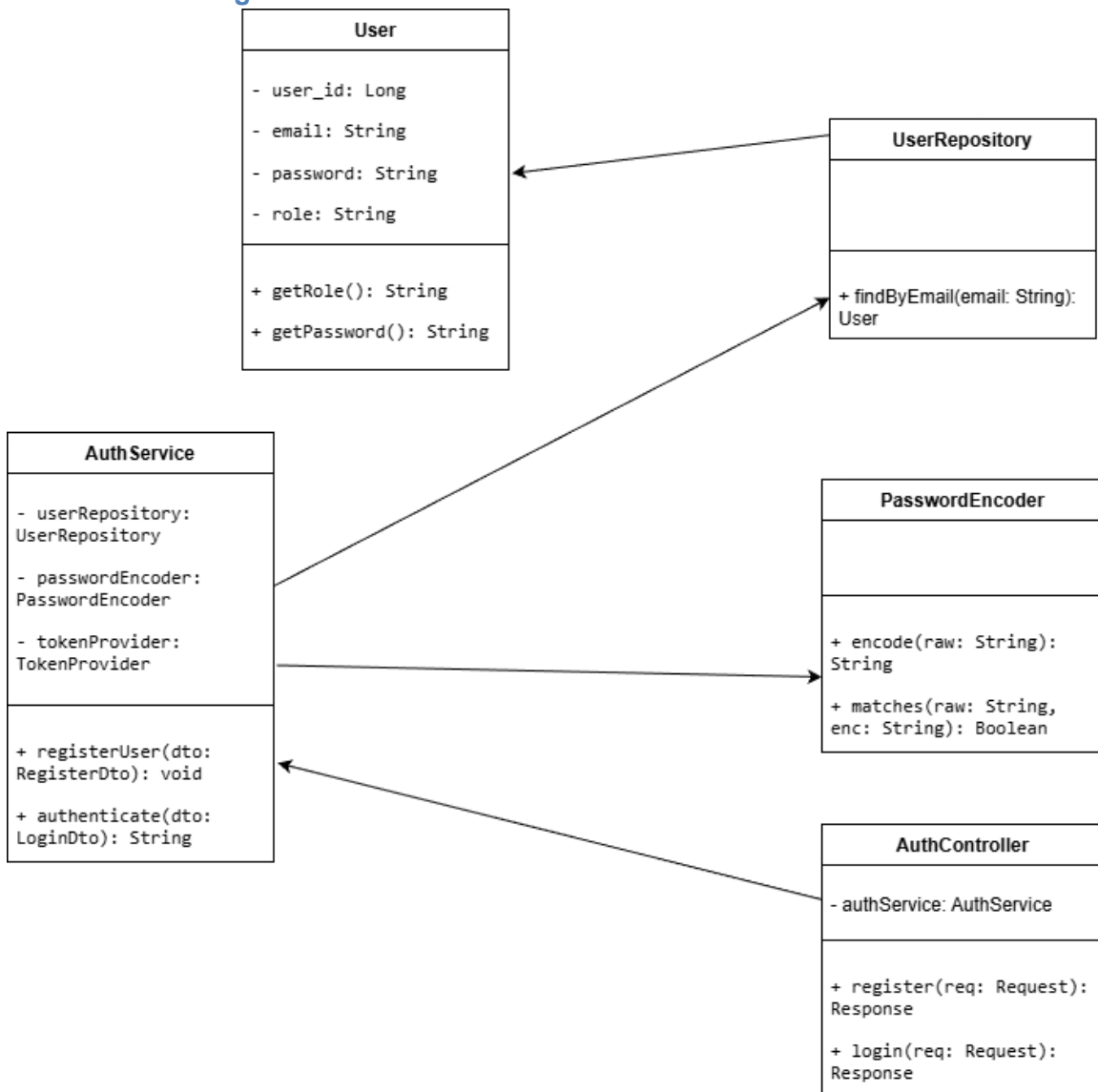
5.2. Use Case Diagram



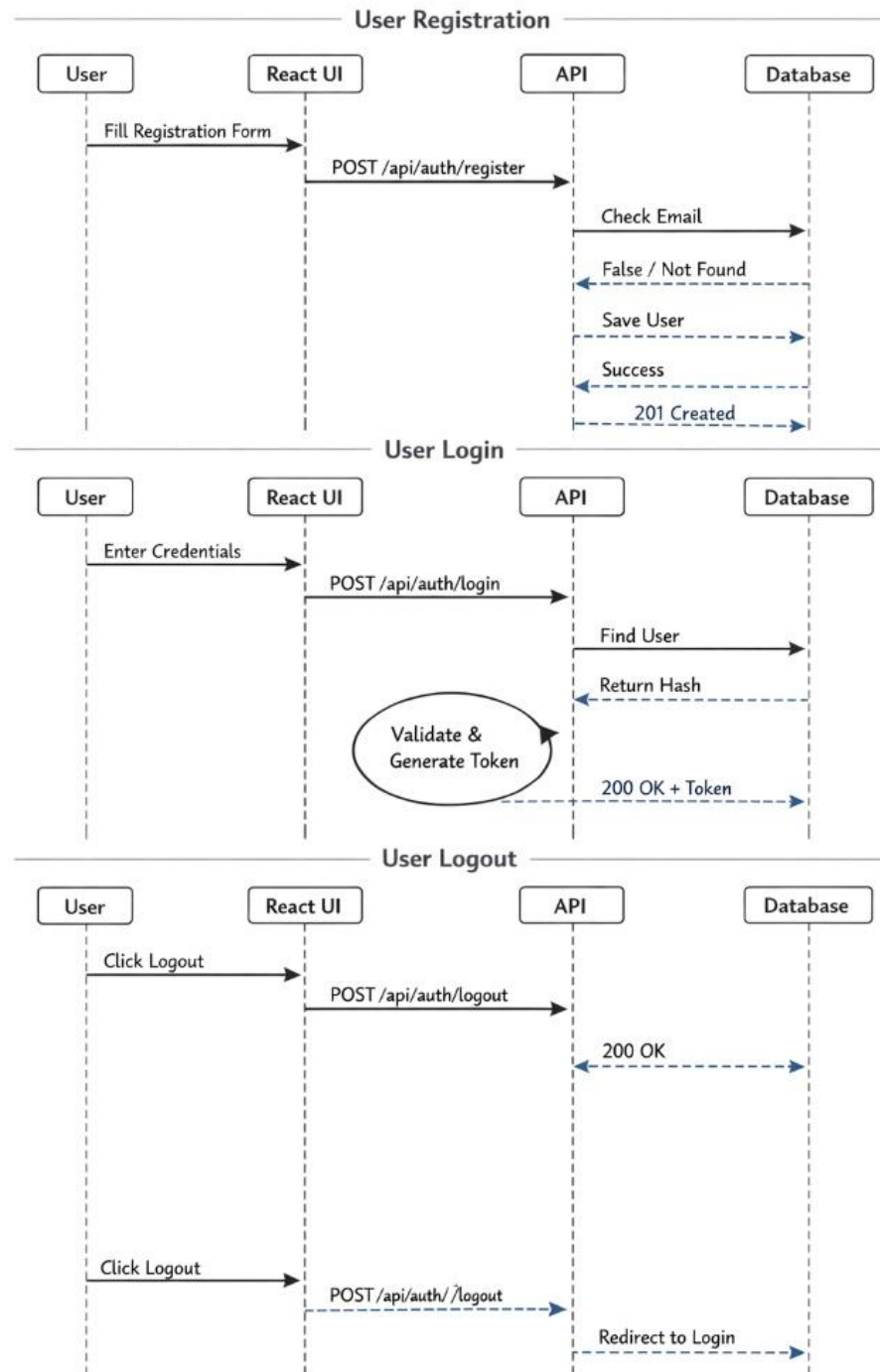
5.3. Activity Diagram



5.4. Class Diagram



5.5. Sequence Diagram



6. Appendices

Include any additional information, references, or support materials.