CEBU INSTITUTE OF TECHNOLOGY
UNIVERSITY

IT342-Section SYSTEMS INTEGRATION AND ARCHITECTURE 1

FUNCTIONAL REQUIREMENTS SPECIFICATION (FRS)

Project Title: User-Registration-Authentication System

Prepared By: Earl John G. Guimoc

Date of Submission: 2/2/2026

Version: 1

Table of Contents

1. Introduction

○ Purpose

The purpose of this system is to provide a secure and reliable user authentication mechanism that allows users to register an account, log in, access protected pages such as a dashboard or profile, and log out safely.

This is intended for students who will use these diagrams and specifications as the basis for implementing the authentication system using React for the frontend and Spring Boot for the backend.

○ Scope

The system will:

- Allow a new user to register an account
- Allow an existing user to log in using valid credentials
- Allow authenticated users to access protected pages (dashboard/profile)
- Prevent unauthorized access to protected pages
- Allow users to log out and end their session

Boundaries of the system:

- The system focuses only on authentication and basic user access control
- It does not include advanced features such as password recovery, email verification, or role-based access control beyond basic user authentication
- The system covers only the flow between React UI, Spring Boot API, and the database

○ Definitions, Acronyms, and Abbreviations

Authentication - Process of verifying a user's identity

Authorization - Process of allowing access to protected resources

JWT (JSON Web Token) - Token used to maintain user session after login

API (Application Programming Interface)- used for communication between frontend and backend

UI - User Interface (React frontend)

DB - Database where user data is stored

ERD - Entity Relationship Diagram

UML - Unified Modeling Language

2. Overall Description

○ System Perspective

The system is a client-server architecture composed of:

- React UI (frontend) for user interaction
- Spring Boot API (backend) for processing authentication logic
- Database for storing user information

The authentication system acts as a foundational module that can later be integrated into larger systems requiring user access control.

○ User Classes and Characteristics

Guest User - Has no account or is not logged in; can only register or log in

Authenticated User - Has successfully logged in; can access dashboard/profile and logout

○ Operating Environment

Hardware:

- Desktop or laptop computer
- Internet connection

Software/Tools:

- Web browser (Chrome, Edge, Firefox)
- React (Frontend)
- Spring Boot (Backend)
- MySQL / PostgreSQL (Database)
- draw.io / diagrams.net (for diagrams)

○ Assumptions and Dependencies

Assumptions:

- Users have stable internet access
- Users provide valid and truthful information during registration
- The backend server is always available

Dependencies:

- React for frontend implementation
- Spring Boot framework for backend services
- Database system for storing user data
- JWT library for token generation and validation

3. System Features and Functional Requirements

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

○ Feature 1: User Registration

Description: This feature allows a new user to create an account by providing a username, email, and password.
Functional Requirements:

- The system shall allow users to input username, email, and password
- The system shall validate that the username and email are unique
- The system shall encrypt the password before storing it in the database
- The system shall store the new user data in the database
- The system shall display a success or error message after registration

○ Feature 2: User Login and Logout

Description: This feature allows registered users to log in using valid credentials, access protected pages, and log out securely.
Functional Requirements:

- The system shall allow users to enter username and password to log in
- The system shall verify credentials against the database
- The system shall generate a JWT token upon successful login
- The system shall allow access to dashboard/profile only when authenticated
- The system shall prevent access to protected pages when not logged in
- The system shall allow users to log out and remove the authentication token

4. Non-Functional Requirements

Performance - The system shall respond to login and registration requests within 2–3 seconds

Security - Passwords shall be encrypted; JWT tokens shall be used for session management

Usability - The interface shall be simple and easy to navigate for basic users

Reliability - The system shall handle invalid inputs and errors gracefully

Scalability - The design shall allow future addition of roles and more user features

Maintainability - The system shall follow layered architecture (Controller, Service, Repository) for easy maintenance

Availability - The system shall be accessible whenever the server is running

Compatibility - The system shall run on modern web browsers

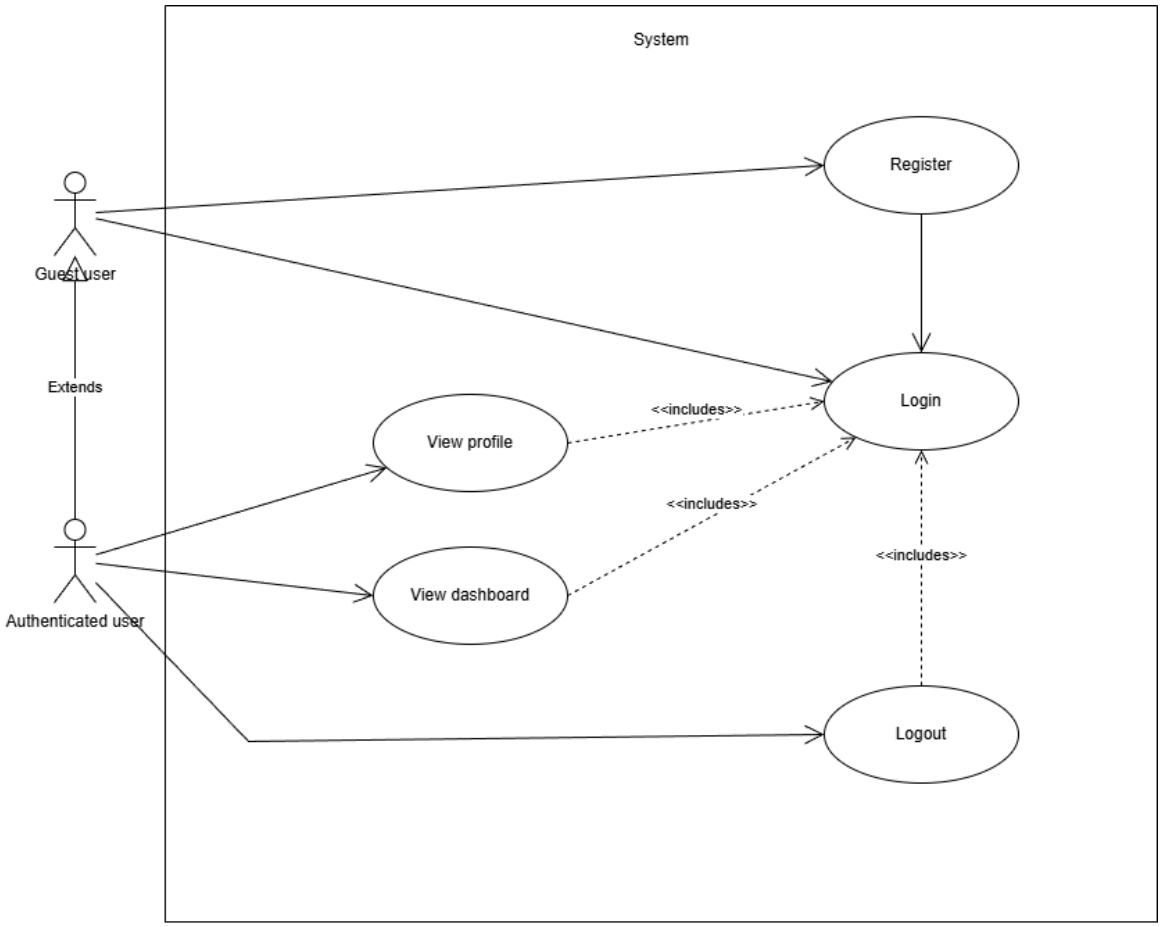
5. System Models (Diagrams)

Insert the necessary diagrams for the system:

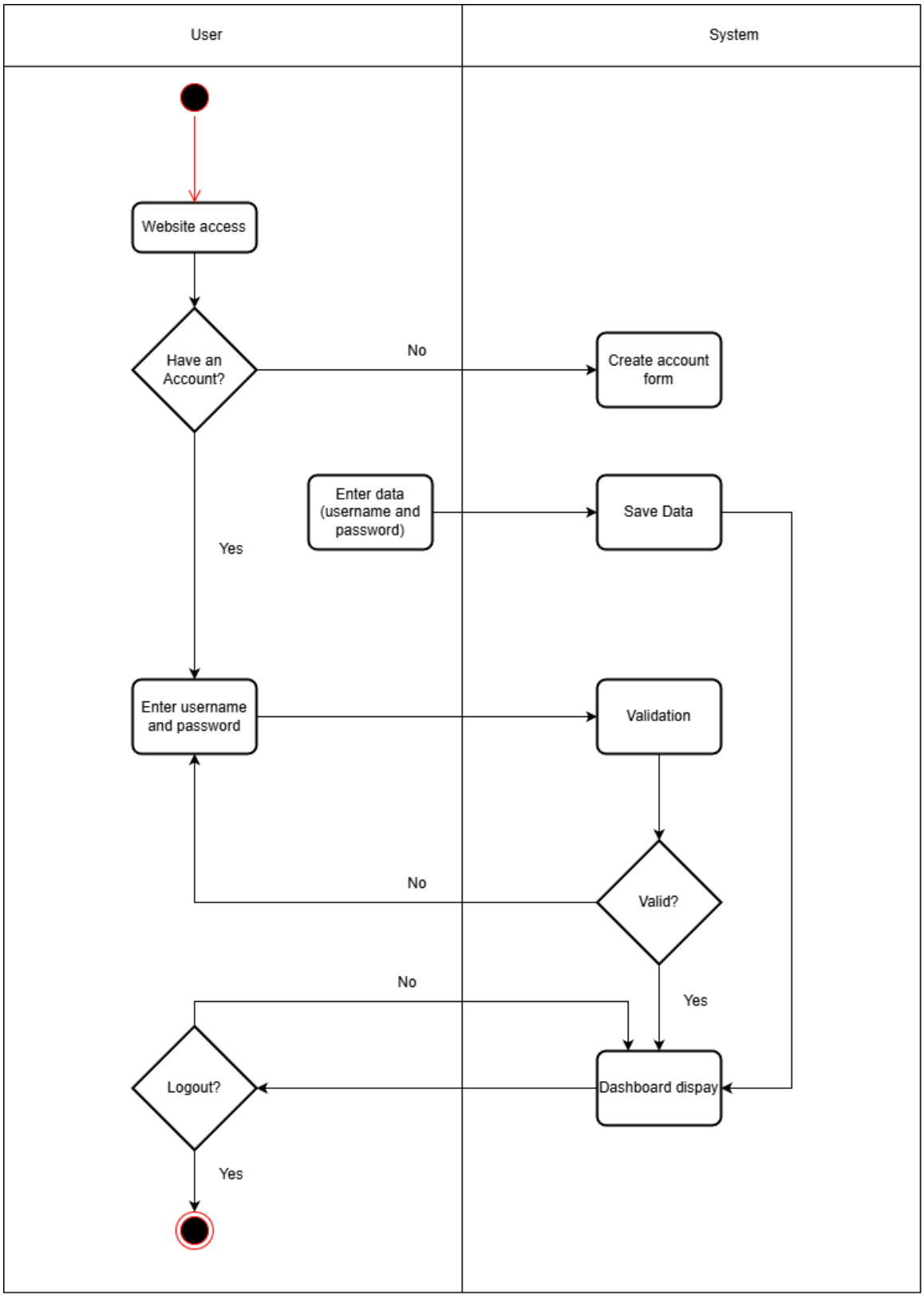
- **ERD**

Users	
PK	UserID BIGINT (20)
	user_name VARCHAR (50)
	password VARCHAR (50)
	email VARCHAR (50)
	first_name VARCHAR (50)
	last_name VARCHAR (50)

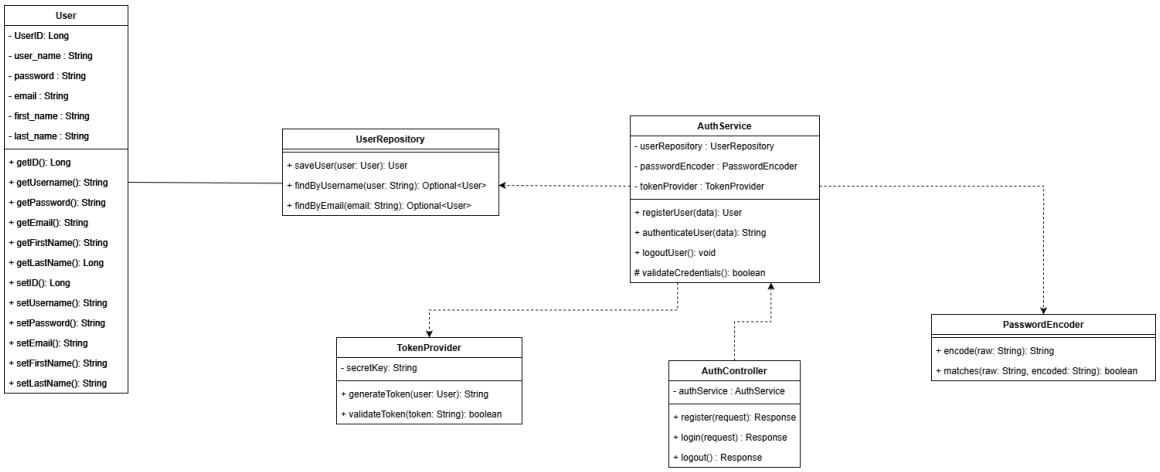
- **Use Case Diagram**



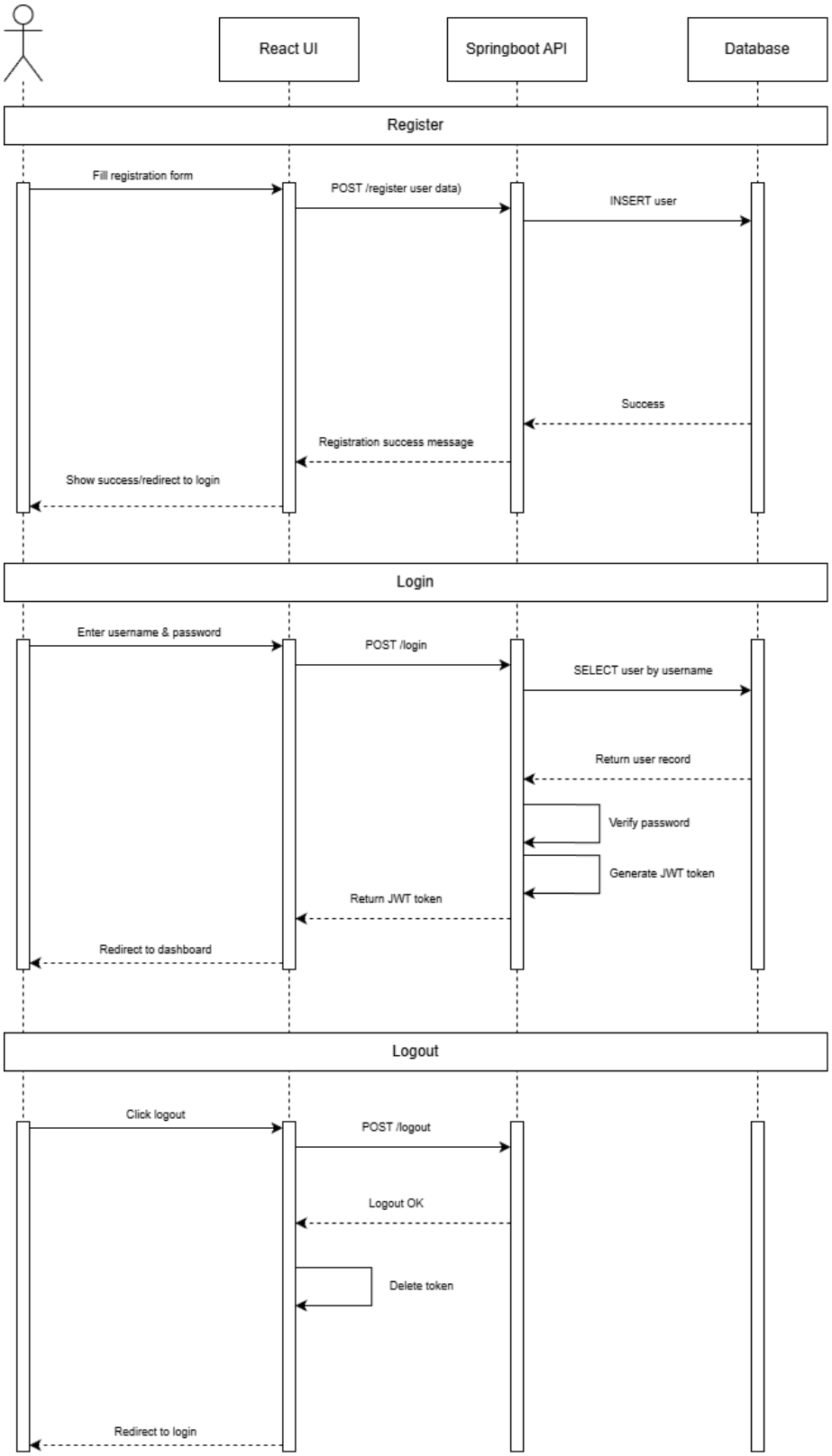
- **Activity Diagram**



○ Class Diagram



- ## Sequence Diagram

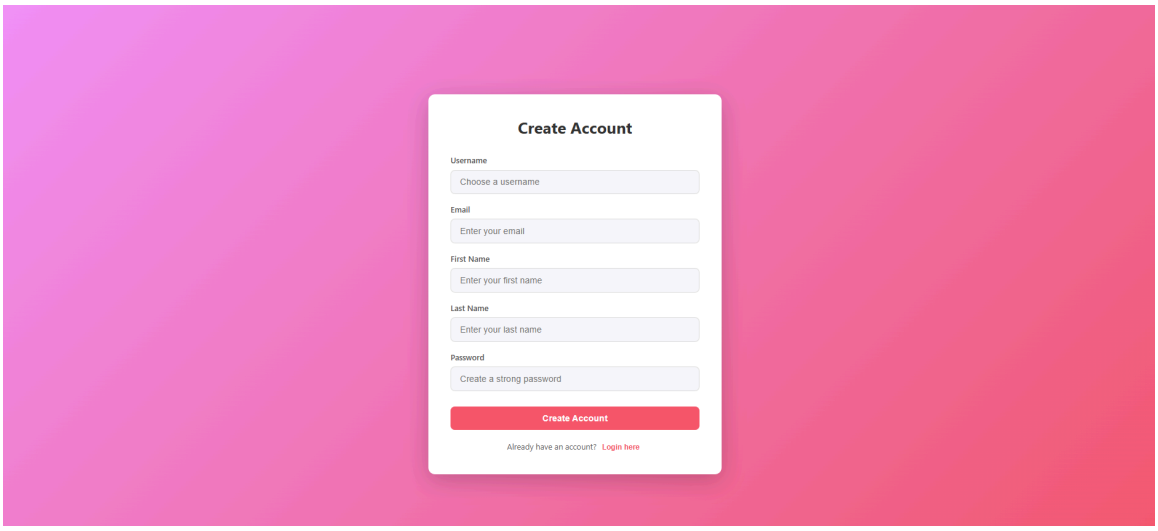


6. Appendices

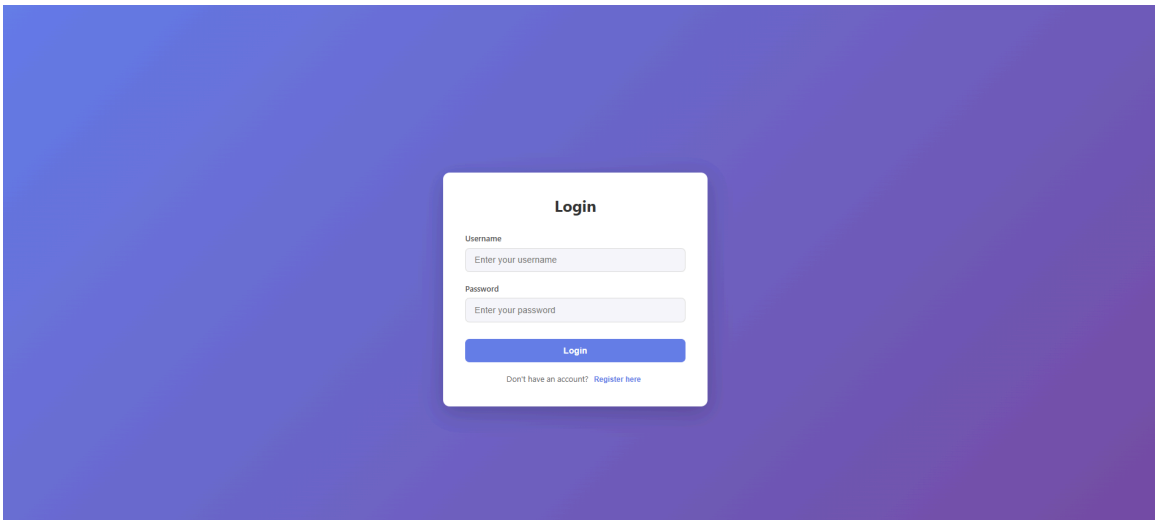
Include any additional information, references, or support materials.

7. Web UI Screenshots

- Register



- Login



- Dashboard/Profile and Logout