CEBU INSTITUTE OF TECHNOLOGY
UNIVERSITY

IT342-Section SYSTEMS INTEGRATION AND ARCHITECTURE 1

FUNCTIONAL REQUIREMENTS SPECIFICATION (FRS)

Project Title: User Registration & Authentication

Prepared By: Mark Anton L. Camoro

Date of Submission: 02 / 03 / 2026

Version: 1.0

Table of Contents

1. Introduction

1.1. Purpose

The purpose of this system is to provide a simple and secure platform where users can register an account, log in, view their profile or dashboard, and log out. This document is intended for developers, students, and stakeholders who need to understand how the system is designed and how it works.

1.2. Scope

The system allows users to create an account, authenticate using login credentials, access protected pages, and log out securely. It includes basic authentication features such as user registration, login validation, session management, and access control. The system does not include advanced features such as password recovery, email verification, or multi-factor authentication.

1.3. Definitions, Acronyms, and Abbreviations

UI - User Interface
API - Application Programming Interface
ERD - Entity Relationship Diagram
UML - Unified Modeling Language
DB - Database
Authentication - Process of verifying user identity
Authorization - Granting access to protected resources
Token - A generated key used for user authentication

2. Overall Description

2.1. System Perspective

The system is a simple authentication application that connects a React-based frontend to a Spring Boot backend and a database. It works as part of a larger software environment where users interact with a web interface while the backend processes authentication and data storage.

2.2. User Classes and Characteristics

Identify the different types of users and their characteristics.

The system has two main types of users:

1. Guest User
    - Not logged in
    - Can register a new account
    - Can log in to the system

2. Authenticated User
    - Already logged in
    - Can view dashboard and profile
    - Can log out to the system

2.3. Operating Environment

Specify the hardware, software, and tools required to operate the system.

The system requires the following environment:

- Frontend: React Web Application
- Backend: Spring Boot API
- Database: Relational Database (such as MySQL or PostgreSQL)
- Development Tools:
    - Java Development Kit (JDK)
    - Node.js
    - Web browser
    - Draw.io / UML tools for documentation

2.4. Assumptions and Dependencies

List any assumptions and external dependencies that may affect the system.

- Users have a valid email address for registration.
- The system depends on a working database to store user information.
- Internet connection is required to access the system.
- The backend API must be running for the frontend to function properly.
- Passwords are securely encoded before being stored.

3. System Features and Functional Requirements

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

3.1. Feature 1: User Registration

Description: Allows a new user to create an account by providing required personal and login details.
Functional Requirements:
- The system shall allow a guest user to access the registration page.
- The system shall validate all input data.
- The system shall save valid user details to the database.
- The system shall display a success message after registration.
- The system shall prevent duplicate email registration.

3.2. Feature 2: User Login

Description: Allows registered users to log in using their email and password.
Functional Requirements:
- The system shall allow users to enter login credentials.
- The system shall validate email and password.
- The system shall authenticate valid users.
- The system shall deny access for invalid credentials.
- The system shall redirect authenticated users to the dashboard.

3.3. Feature 3: Profile and Dashboard Access

Description: Allows authenticated users to access protected pages.
Functional Requirements:
- The system shall allow only logged-in users to view the dashboard.
- The system shall display user profile information.
- The system shall block access to protected pages for guest users.

3.4. Feature 4: Logout

Description: Allows authenticated users to securely log out of the system.
Functional Requirements:
- The system shall allow users to log out.
- The system shall destroy the active session or token.
- The system shall redirect the user to the login page after logout.

4. Non-Functional Requirements

Specify system quality attributes such as performance, security, usability, reliability, etc.

Security:

- Passwords must be encrypted before storage.
- Authentication tokens must be validated.

Performance:

- Login and registration should process quickly.

Usability:

- The system must have a simple and user-friendly interface.

Reliability:

- The system must correctly validate user input.

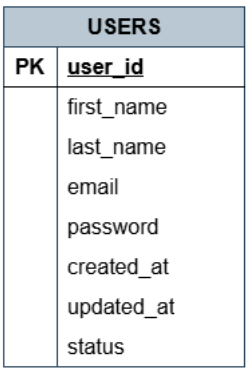
Availability:

- The system should be accessible whenever the server is running.

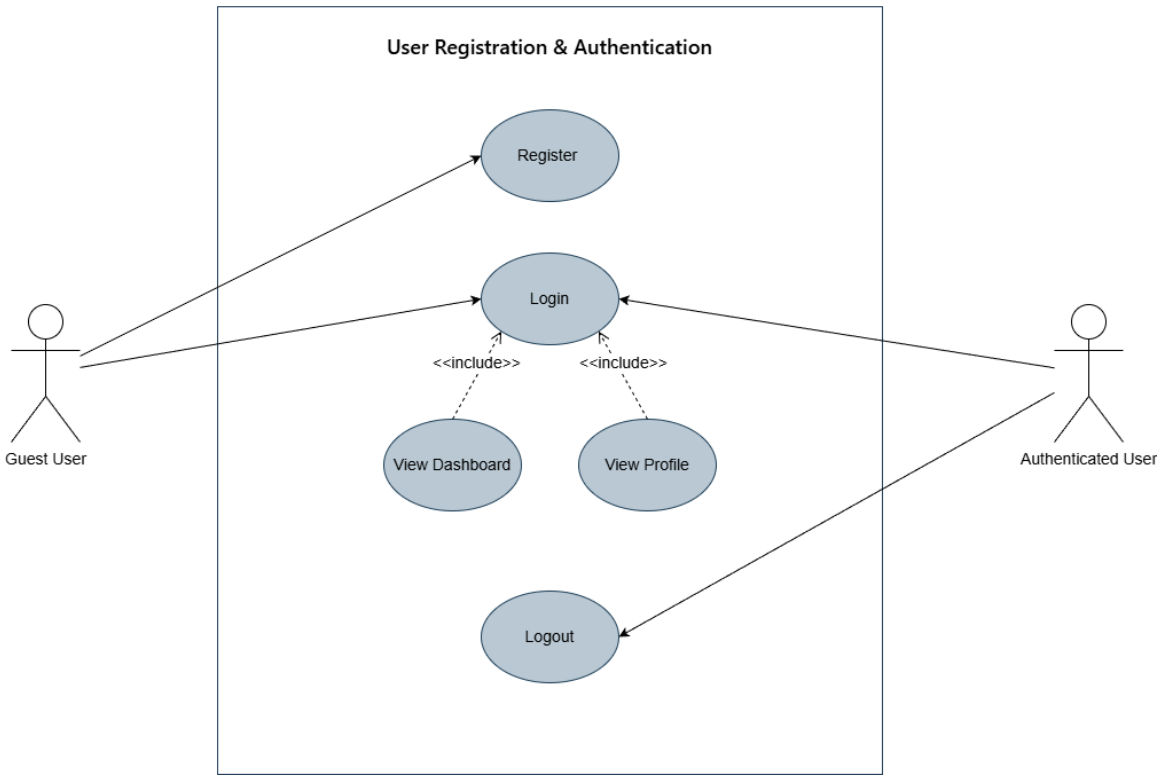
5. System Models (Diagrams)

Insert the necessary diagrams for the system:

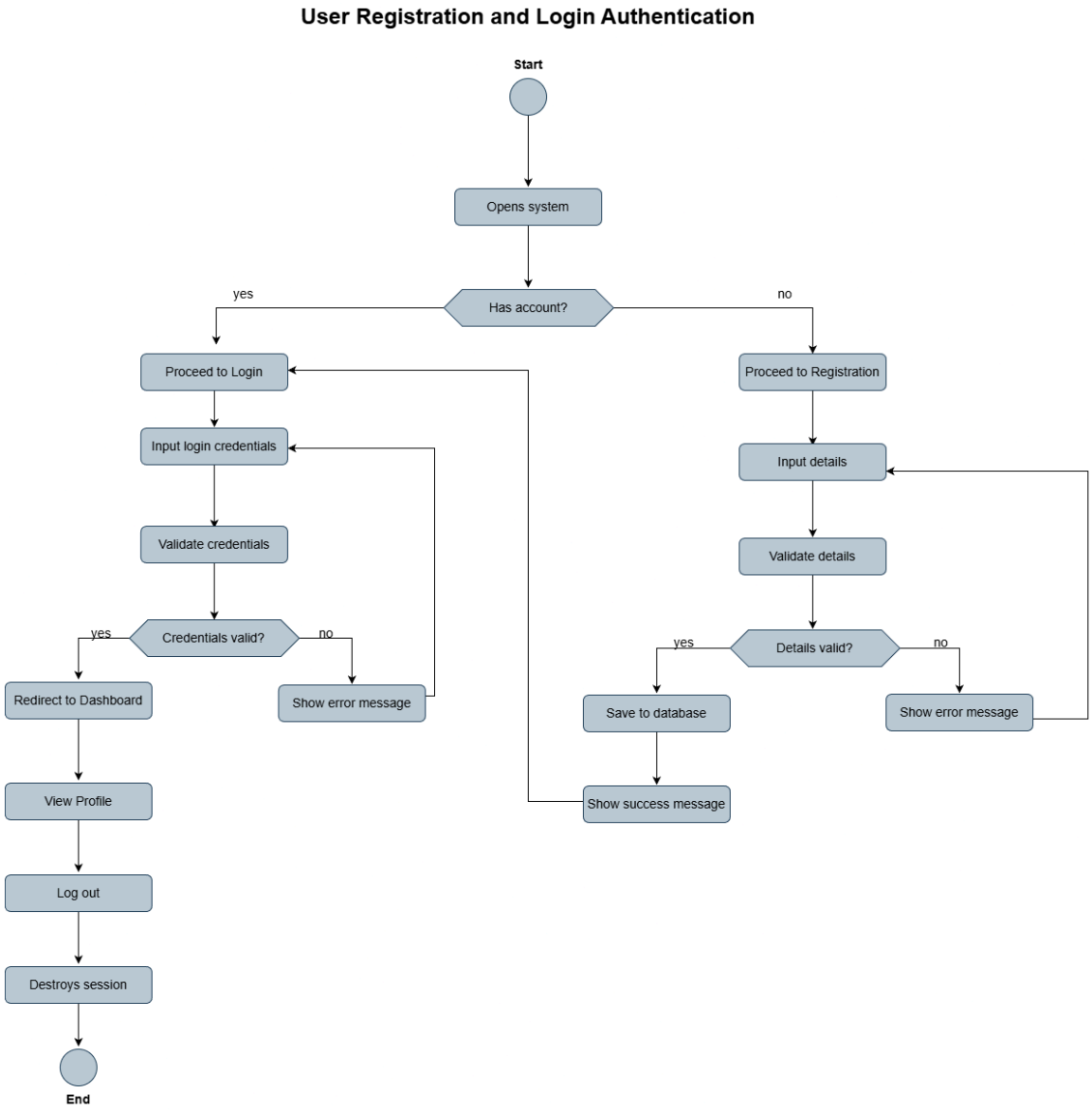
5.1. ERD



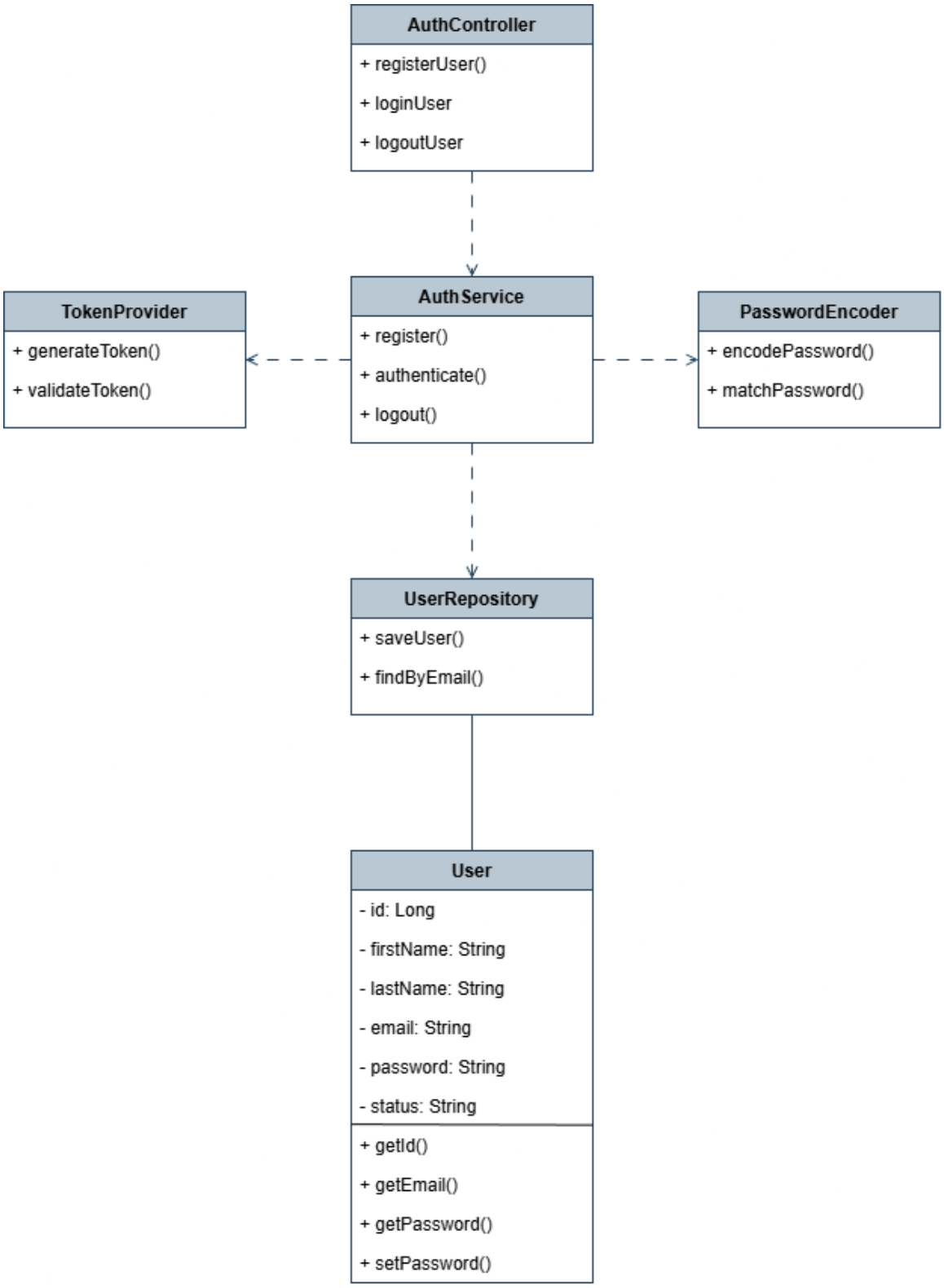
5.2. Use Case Diagram



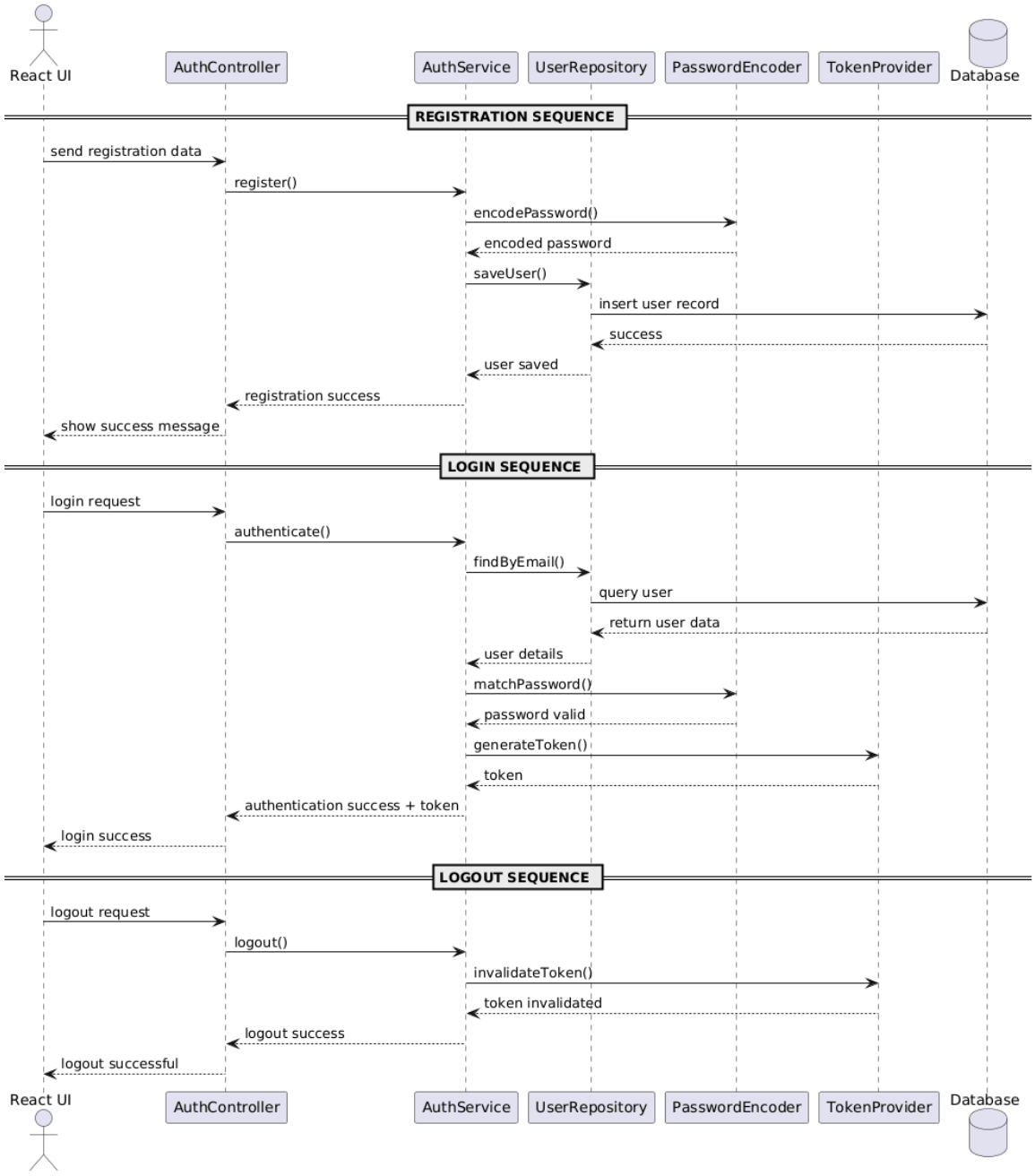
5.3. Activity Diagram



5.4. Class Diagram



5.5. Sequence Diagram



6. Appendices

Include any additional information, references, or support materials.

References:

[1]Nishadha, “Use Case Diagram Relationships Explained with Examples,” *Creately Blog*, Feb. 17, 2015. https://creately.com/blog/diagrams/use-case-diagram-relationships/

[2]Visual Paradigm, “UML Class Diagram Tutorial,” *Visual-paradigm.com*, 2024. https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/

Support Materials:

- UML Diagrams created using Draw.io