RoomMate Project Specification

Course: CS-UY 4513 - Software Engineering
Professor: Dr. DePasquale
Project Title: RoomMate
Date: September, 28, 2025

1.0 Project Overview

The goal of this project is to design and develop a SaaS platform that helps students and newcomers to a city find compatible roommates to share housing costs.

The platform is named **RoomMate** and will serve as a reliable and user-friendly solution for individuals seeking to split rent and reduce living expenses.

The system will support a robust user management system with:

- Profile creation and verification for identity, budget, lifestyle preferences, and housing needs;
- Matching algorithms to connect users based on compatibility factors such as budget, location, habits, and schedules;
- A way for users to coordinate with potential roommates;
- Persistent data storage to maintain user preferences, search history, and active matches.

2.0 Core Requirements

The project must adhere to the mandatory characteristics outlined in the project specification template document.

2.1 User-Based System [SUMMARIZE] - Eric

[DESCRIBE THE REQUIREMENT AND HOW YOU WILL MEET IT]

2.2 User Roles (Minimum 3) - Eric

[DEFINE EACH ROLE AND SUMMARIZE ITS FUNCTIONALITY]:

- **ROLE 1:** [DESCRIBE AND SUMMARIZE]
- **ROLE 2:** [DESCRIBE AND SUMMARIZE]

- **ROLE 3:** ~~[DESCRIBE AN### 2.3 Persistent Storage~~
~~[DESCRIBE THE REQUIREMENT AND HOW YOU WILL MEET IT]~~

2.3 Persistent Storage - Terry

Proposed Database Schema (max 10 tables):
- **[table_name]:** [describe the purpose of the table]

2.4 Modular Architecture (3-5 Modules) - Kevin

The system must be logically divided into a minimum of 3 and a maximum of 5 modules. These modules must demonstrate clear dependencies and rely on each other for full functionality.

Proposed Modules:
1. **[MODULE NAME]**
   - [SUMMARIZE THE FUNCTIONALITY OF THE MODULE IN A LIST]
   - **Dependency:** [DESCRIBE ANY DEPENDENCIES TO/FROM THIS MODULE]

2.5 API Interfaces - Steven

Each module must expose a RESTful API interface. This is crucial for enabling the modules to communicate with each other and for potential future integrations. The API should follow standard REST conventions.

Example Endpoints:
- **User & Identity Management API:**
  - [LIST EACH TYPE AND ...]

- **Course & Assignment Management API:**
  - `GET /api/courses`
  - `POST /api/courses` (Requires Instructor/Admin role via User API)
  - `GET /api/courses/:id/assignments`

3.0 Technical Stack

- **Language:** [LIST AND DESCRIBE IF NECESSARY]
- **Framework:** [LIST AND DESCRIBE IF NECESSARY]
- **Database:** [LIST AND DESCRIBE IF NECESSARY]
- **Testing:** [LIST AND DESCRIBE IF NECESSARY]

4.0 Deliverables

The final submission must include:

1. **Complete Source Code:**
   A well-structured and commented codebase, including all necessary configuration files.

2. **Project Documentation:**
   A `README.md` file that explains the system's architecture, setup instructions, and how to run the application.

3. **API Documentation:**
   A separate document or file detailing all API endpoints, their expected parameters, and response formats.

4. **Presentation:**
   A brief presentation and demonstration of the application's core features.