# **x1. Introduction**

## **1.1 Purpose**

This document specifies the complete requirements for the "Study Buddy" application. It is intended for project stakeholders, designers, and developers.

## **1.2 Scope**

The application will allow Clemson students to create profiles, list their courses, manage their availability, and schedule study sessions with classmates. The system will be a standalone **desktop application** with a graphical user interface (GUI) built using **PyQt6**. It will not require external network connectivity.

## **1.3 Definitions**

* **User:** A Clemson student using the application.
* **Session:** A scheduled study meeting between two or more users.
* **Availability Slot:** A specific time block (e.g., Monday 2-3 PM) when a user is free to study.
* **GUI:** Graphical User Interface built using PyQt6.

# **2. Overall Description**

## **2.1 Product Perspective**

This is a new, self-contained software product. Data will be managed in memory during runtime and will not persist between application restarts.

## **2.2 Product Functions**

* User profile and course management.
* Availability scheduling.
* Automated matching of study partners based on course and availability.
* Session creation and confirmation.

## **2.3 User Characteristics**

Users are Clemson University students who are comfortable with basic computer applications. No prior experience with command-line interfaces is required, as the system will have a **menu-driven GUI** with buttons, dialogs, and tables for interaction.

## **2.4 Constraints**

* The application must be developed as a **Python desktop application using PyQt6**.
* The system shall not require a persistent database; all data is session-based.
* Development must strictly follow the Waterfall model.

# **3. Specific Requirements**

## **3.1 Functional Requirements**

**FR-1: User Profile Creation**

* **Description:** The system shall allow a new user to create a profile via a GUI form with text fields for username, password, and full name.
* **Inputs:** Username, password (entered securely), full name.
* **Processing:** The system shall validate that the username is not already taken.
* **Outputs:** A confirmation message dialog or an error message dialog.

**FR-2: User Login**

* **Description:** A registered user must be able to log in to the system using a login window.
* **Inputs:** Username, password.
* **Processing:** The system shall authenticate the user's credentials against stored records.
* **Outputs:** The main application window opens upon success; an error dialog is shown upon failure.

**FR-3: Course Management**

* **Description:** A logged-in user shall be able to add and remove courses from their profile using GUI controls such as buttons and input dialogs.
* **Inputs:** Course code.
* **Processing:** The system will add/remove the specified course from the user's list.
* **Outputs:** Confirmation message in the GUI and updated course list display.

**FR-4: Availability Management**

* **Description:** A logged-in user shall be able to add and remove availability slots for studying using a visual scheduler (e.g., a table or dropdowns for day/time).
* **Inputs:** Day, start time.
* **Processing:** The system will add/remove the time slot from the user's availability schedule.
* **Outputs:** Confirmation message in the GUI and updated availability list/table.

**FR-5: Suggest Study Matches**

* **Description:** A logged-in user shall be able to request a list of suggested study partners via a button or menu option.
* **Inputs:** Course code.
* **Processing:** The system shall identify all other users enrolled in the same course who have at least one overlapping availability slot.
* **Outputs:** A scrollable list or table of suggested partners displayed in the GUI.

**FR-6: Schedule a Study Session**

* **Description:** A user shall be able to schedule a study session with a suggested partner for a specific course at a mutually available time.
* **Inputs:** Partner's username, course code, day, and time (selected from GUI widgets).
* **Processing:** The system will verify that both users are free at the specified time and are enrolled in the course. It will then create a pending session invitation.
* **Outputs:** Confirmation dialog indicating the invitation has been sent.

**FR-7: View and Confirm Meetings**

* **Description:** A user shall be able to view their pending and confirmed study sessions in a GUI table or list. They must be able to confirm or decline pending invitations using buttons.
* **Inputs:** User selection to view sessions and confirm/decline invitations.
* **Processing:** The system will update the status of the session from 'Pending' to 'Confirmed' or 'Declined'.
* **Outputs:** Updated session status shown in the GUI.

## **3.2 Non-Functional Requirements**

**NFR-1: Usability:** The GUI must be intuitive, menu-driven, and provide clear, unambiguous labels and messages to the user. Forms and dialogs should have a consistent design.

**NFR-2: Performance:** All interface updates (e.g., list refreshes, navigation) must occur in under 1 second on a standard computer.

**NFR-3: Reliability:** The application should gracefully handle invalid inputs by showing error dialogs without crashing.

**NFR-4: Security:** Password input fields must mask characters (e.g., show bullets instead of plain text).

## **3.3 Interface Requirements**

### **3.3.1 User Interface**

All interaction will be through a **PyQt6 graphical user interface**.

* The main window will contain navigation buttons or menus for each major feature (Profile, Courses, Availability, Matches, Sessions).
* Dialog boxes will be used for data entry (e.g., creating a profile, adding courses).
* Tables or list widgets will display data such as courses, availability slots, partner suggestions, and sessions.
* Buttons will be provided for actions like Add, Remove, Confirm, and Decline.