**Software Requirements Specification (SRS)**

**Habib Unified Portal**  
**Version**: 1.0  
**Project Team**:

* Taha Hunaid Ali
* Shavez Niyazi  
  **Submission Date**: [Submission Date]

**Document History**

| **Version** | **Name of Person** | **Date** | **Description of Change** |
| --- | --- | --- | --- |
| 1.00 | Taha Hunaid Ali | 18/3/25 | Initial version of the SRS |
| 1.01 | Taha Hunaid Ali | 21/3/25 | Added functional and non-functional requirements |
| 1.02 | Taha Hunaid Ali | 21/4/25 | Completed system description and interface requirements |
| 1.03 | Taha Hunaid Ali | 25/4/25 | Final version with all sections completed |

**Distribution List**

| **Name** | **Role** |
| --- | --- |
| Ateeb Ahmed | Course Instructor |

**Document Sign-Off**

| **Version** | **Sign-off Authority** | **Project Role** | **Sign-off Date** |
| --- | --- | --- | --- |
| 1.03 | Ateeb Ahmed | Supervisor | 25/4/25 |

**Table of Contents**

1. **INTRODUCTION** 7  
   1.1. Purpose of Document 7  
   1.2. Intended Audience 7  
   1.3. Document Convention 7
2. **OVERALL SYSTEM DESCRIPTION** 8  
   2.1. Project Background 8  
   2.2. Project Scope 8  
   2.3. Not In Scope 8  
   2.4. Project Objectives 8  
   2.5. Stakeholders 9  
   2.6. Operating Environment 9  
   2.7. System Constraints 9  
   2.8. Assumptions & Dependencies 9
3. **SOFTWARE DEVELOPMENT METHODOLOGY** 10
4. **SYSTEM ANALYSIS & REQUIREMENT GATHERING** 10
5. **EXTERNAL INTERFACE REQUIREMENTS** 10  
   5.1. Hardware Interfaces 10  
   5.2. Software Interfaces 10  
   5.3. Communications Interfaces 10
6. **FUNCTIONAL REQUIREMENTS** 11  
   6.1. Data Flow Diagrams 11  
   6.2. Decision Tables/CRUD and Association Matrices 14  
   6.3. Use Cases 16
7. **NON-FUNCTIONAL REQUIREMENTS** 22  
   7.1. Performance Requirements 22  
   7.2. Safety Requirements 22  
   7.3. Security Requirements 22  
   7.4. User Documentation 22
8. **REFERENCES** 23
9. **APPENDICES** 24

**1. INTRODUCTION**

**1.1 Purpose of Document**

The purpose of this document is to outline the software requirements for the **Habib Unified Portal**. The portal is intended to provide a centralized platform for **academic resources**, **faculty-student interaction**, **event management**, and **appointment scheduling** at **Habib University**. This document specifies the **functional** and **non-functional requirements**, **system constraints**, **external interfaces**, and **development methodology** used for the project.

**1.2 Intended Audience**

* **Students**: Primary users of the portal who will access academic resources, schedule appointments, and register for events.
* **Faculty**: Professors and course assistants who will manage academic resources, appointments, and events.
* **Administrators**: University staff responsible for managing user roles and maintaining the portal.
* **Developers**: The team responsible for developing, testing, and deploying the system.

**1.3 Document Convention**

* **Font**: Calibri, size 12
* **Spacing**: 1.15 line spacing
* **Sections**: Bold section titles and numbered subsections (Font: Times New Roman)

**2. OVERALL SYSTEM DESCRIPTION**

**2.1 Project Background**

The **Habib Unified Portal** is designed to address the need for a centralized system where students and faculty can easily access academic resources, manage appointments, and register for university events. Currently, the university relies on fragmented communication methods such as emails, paper records, and ad-hoc scheduling.

**2.2 Project Scope**

The system will provide:

* **Academic Support**: A platform for students to access course materials, discussion forums, and academic resources.
* **Faculty Interaction**: Appointment scheduling with faculty members and course assistants.
* **Event Management**: A centralized event calendar for academic, co-curricular, and extracurricular events.
* **Notifications**: Email and push notifications for important updates and reminders.

**2.3 Not In Scope**

* The portal will not support **mobile applications** at the initial stage.
* **Remote payments or credit card transactions** will not be supported in the first version of the portal.

**2.4 Project Objectives**

The key objectives of the **Habib Unified Portal** are:

* Provide easy access to academic resources and facilitate student-faculty interactions.
* Centralize event management and ensure that students can easily register for events.
* Improve data security and reduce manual effort by automating appointment scheduling and event management.

**2.5 Stakeholders**

* **Owner**: Habib University (the institution funding the project).
* **Users**: Students, faculty, and administrators who will interact with the portal.
* **Project Manager**: Responsible for overseeing the development and delivery of the project.
* **Developers**: Responsible for implementing the system.
* **Quality Assurance**: Ensures that the system is working as expected through testing.

**2.6 Operating Environment**

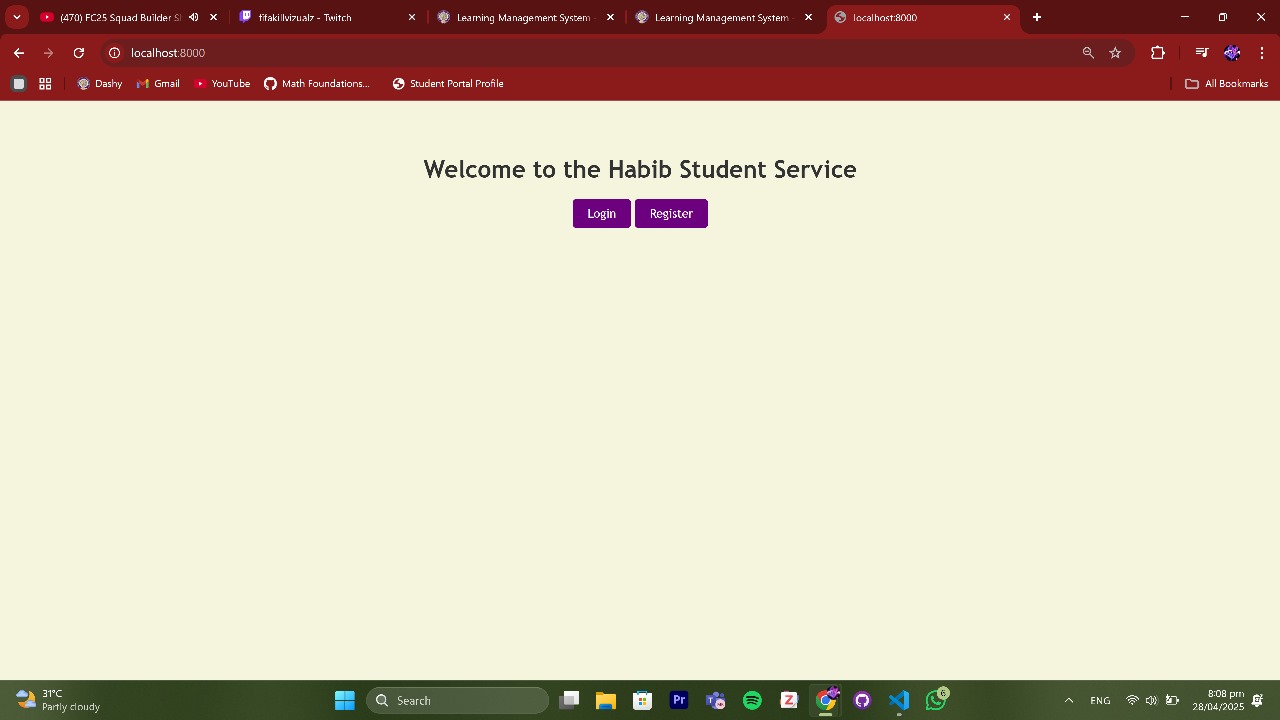
* **Database Management System**: SQLite (for simplicity and ease of integration).
* **Web Framework**: Flask (Python-based framework for backend development).
* **Operating System**: Cross-platform (Windows/Linux).

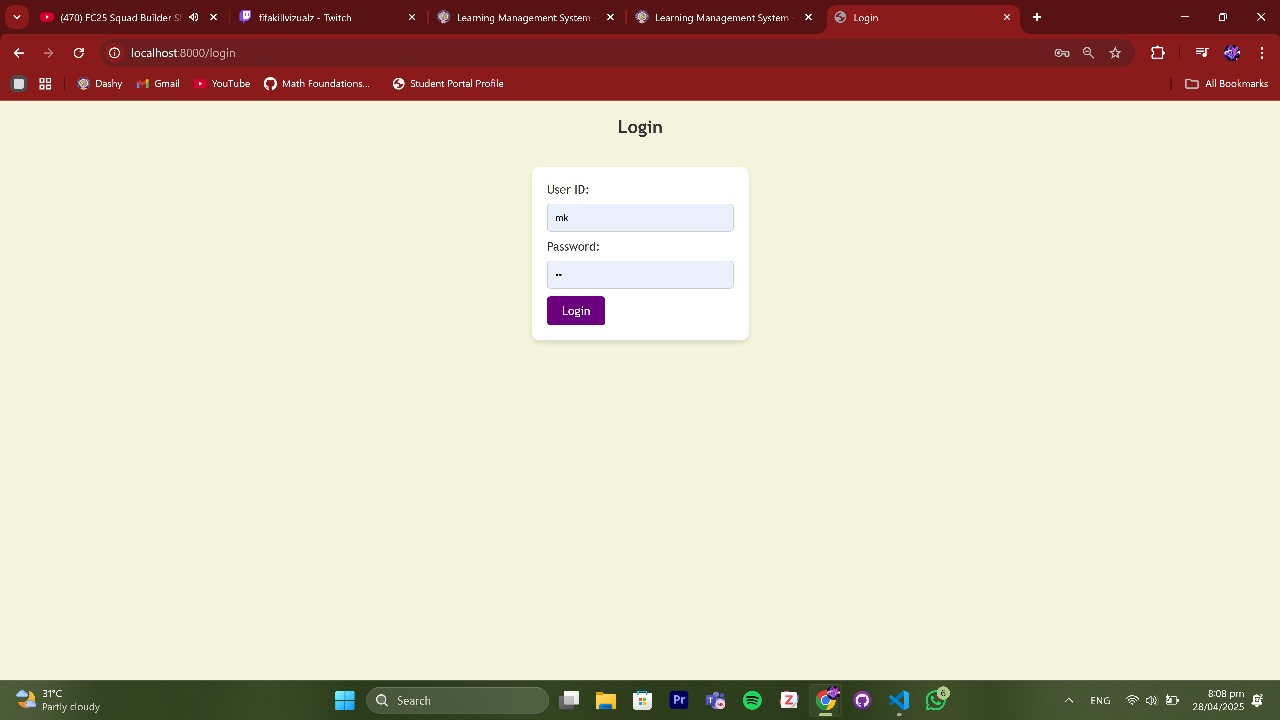
**2.7 System Constraints**

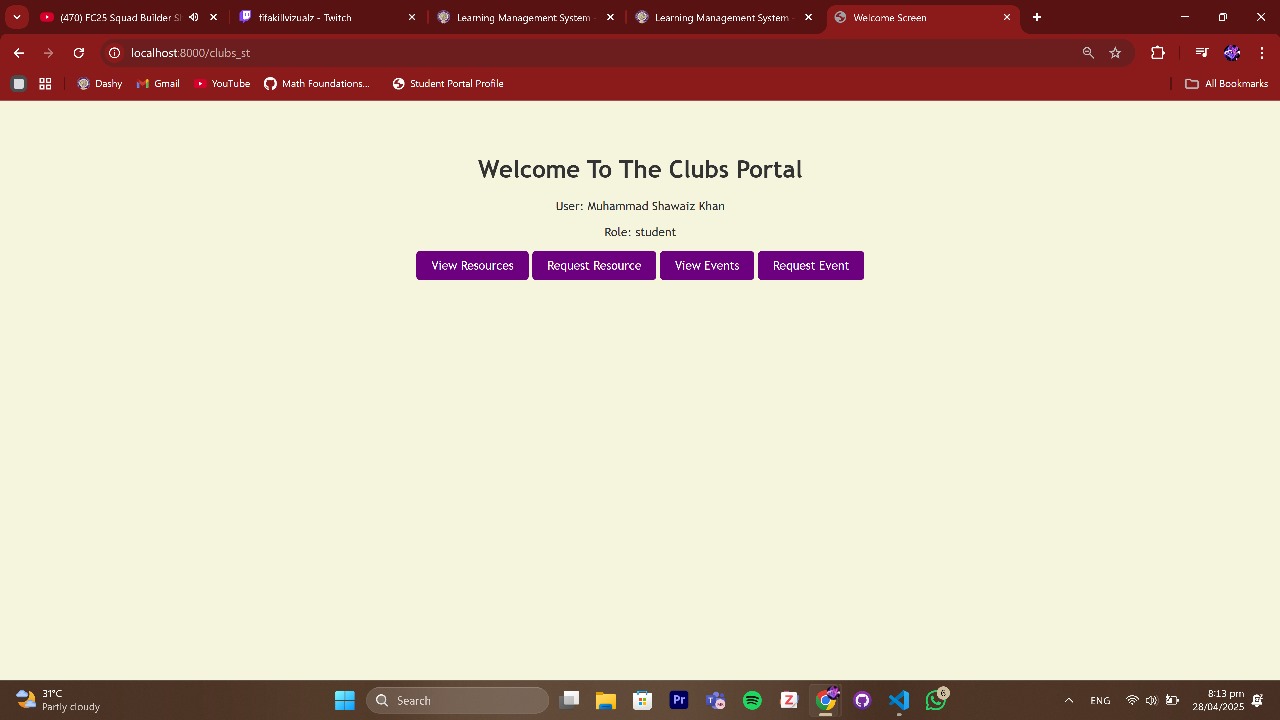
* The system will depend on the university’s **network infrastructure**.
* **System uptime** is crucial, especially during **high-traffic periods** (e.g., exam seasons and event registration periods).

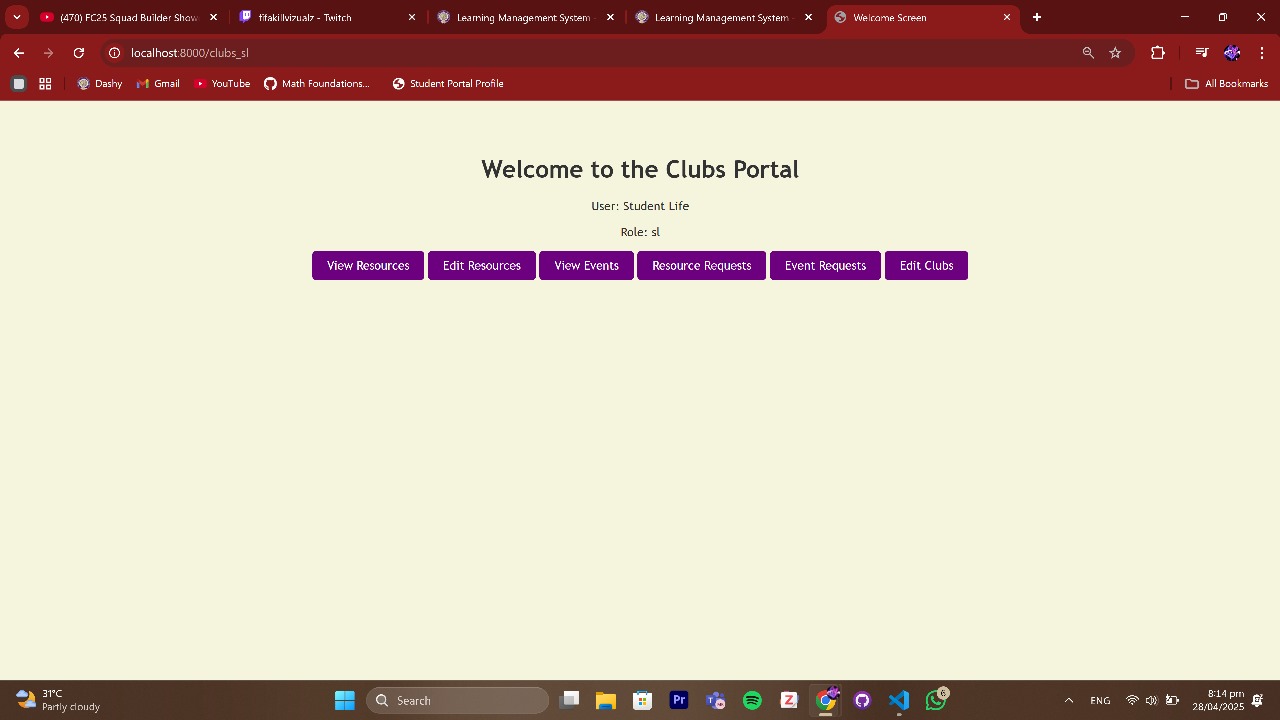
**2.8 Assumptions & Dependencies**

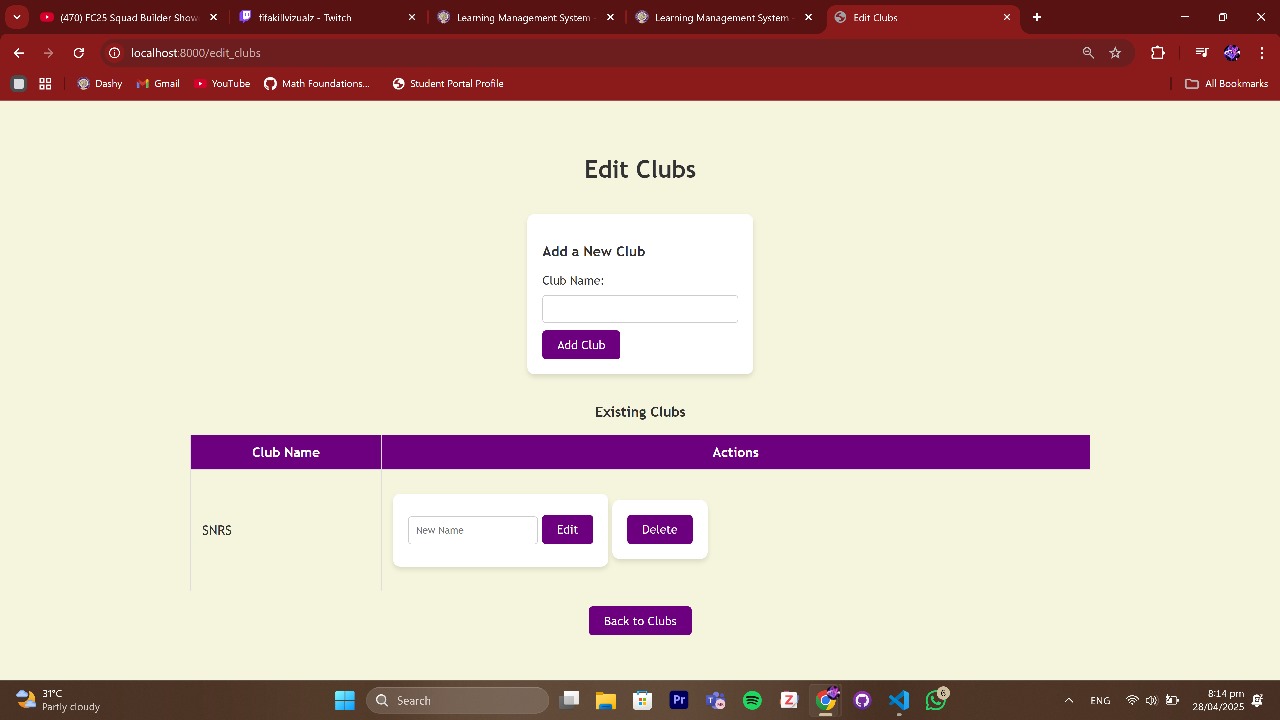
* All **students** and **faculty** will have **access to the internet** and **devices with browsers**.
* The system depends on the **university's existing authentication system** (likely Single Sign-On or LDAP) for login.

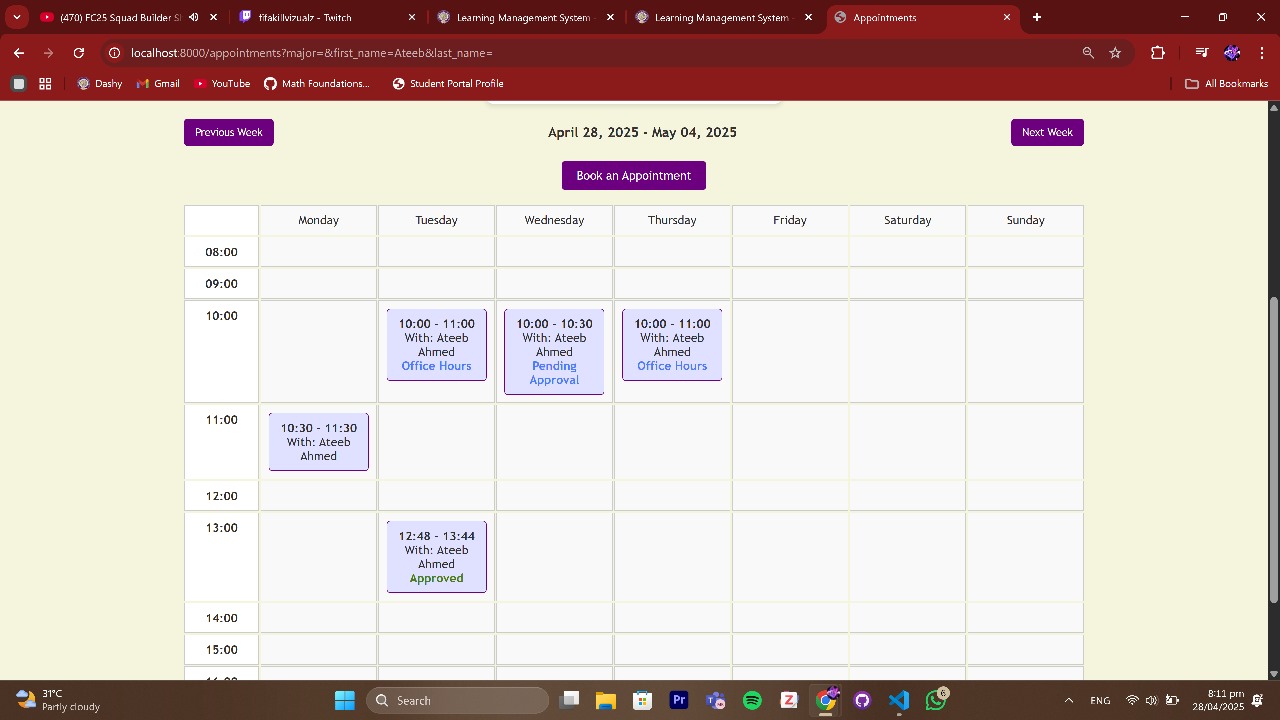


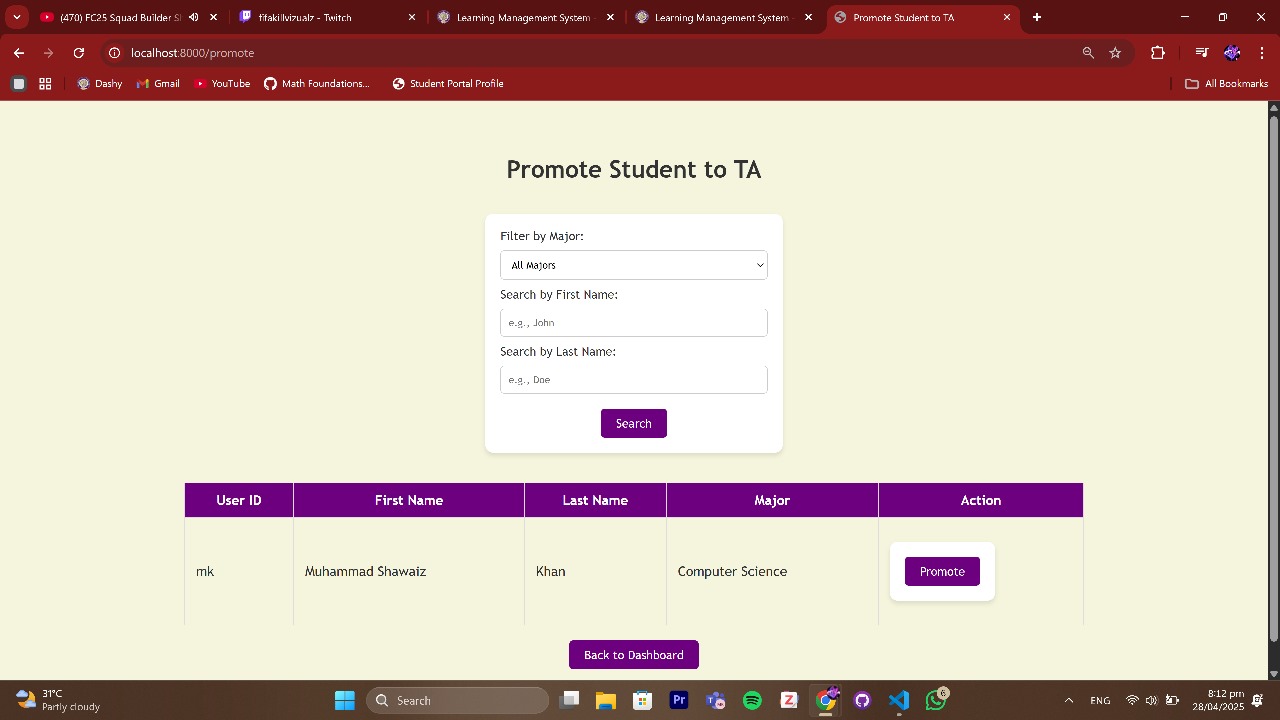












**3. SOFTWARE DEVELOPMENT METHODOLOGY**

The development of the portal will follow the **Agile methodology**, which focuses on incremental development and continuous feedback from stakeholders to ensure the development process aligns with user expectations and requirements.

**4. SYSTEM ANALYSIS & REQUIREMENT GATHERING**

Requirements will be gathered through:

* **Interviews**: Conducting interviews with faculty, students, and administrators to understand their needs.
* **Surveys**: Collecting data on existing pain points and gathering feedback on the user experience.
* **Workshops**: Discussing and refining system requirements with key stakeholders.

**5. EXTERNAL INTERFACE REQUIREMENTS**

**5.1 Hardware Interfaces**

* **Client devices**: Desktops, laptops, tablets with internet access.
* **Server**: A web server capable of running Flask and the SQLite database.

**5.2 Software Interfaces**

* **Web Framework**: Flask for backend development.
* **Database**: SQLite for data storage.
* **Authentication**: The university’s existing **SSO (Single Sign-On)** or LDAP system for user login.

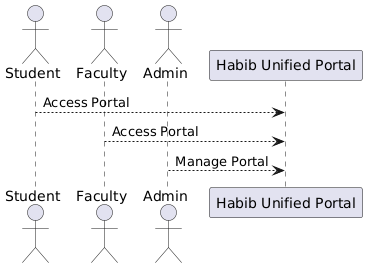
**5.3 Communications Interfaces**

* **HTTP/HTTPS**: Used for communication between clients and servers.
* **Email/SMS APIs**: Used for sending notifications.

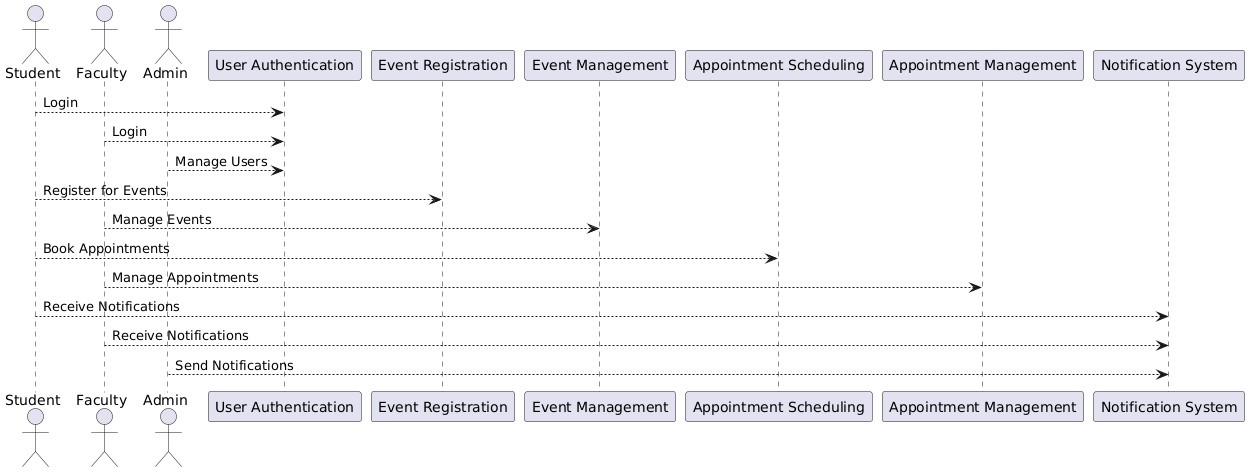
**6. FUNCTIONAL REQUIREMENTS**

**6.1 Data Flow Diagrams (DFDs)**

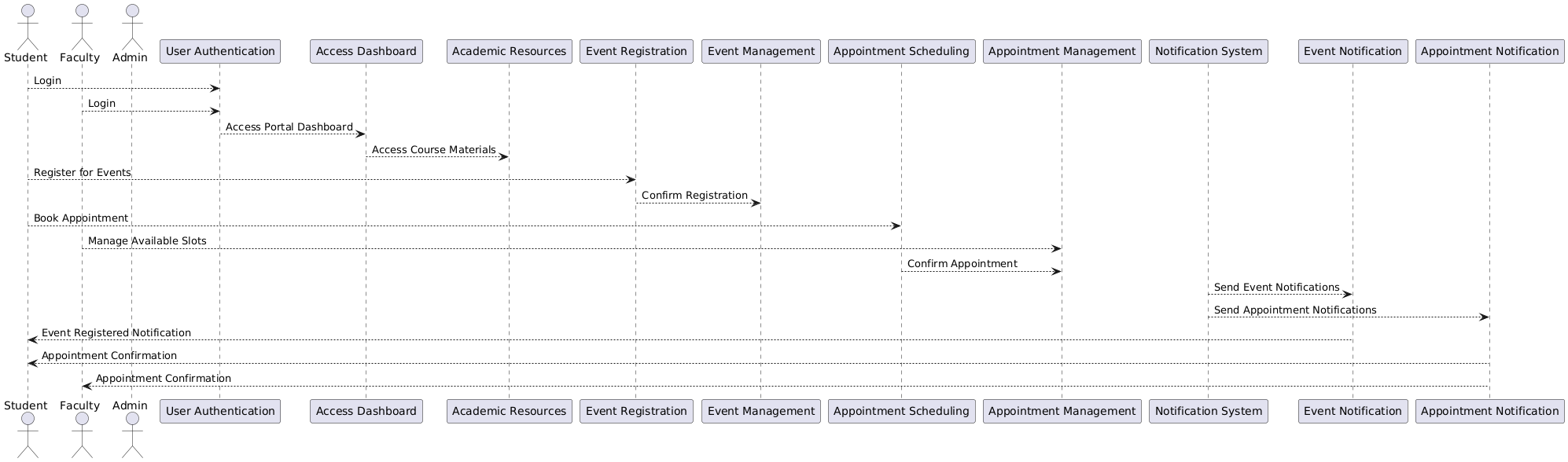
* **Context Level DFD**: Shows interactions between external entities (students, faculty, admins) and the **Habib Unified Portal**.



* **Level-0 DFD**: Breaks down the system processes such as **user authentication**, **event registration**, **appointment scheduling**, etc.



* **Level-1 DFD**: Details user flows for **academic resources**, **appointment scheduling**, and **notifications**.



**6.2 Decision Tables/CRUD and Association Matrices**

* **CRUD operations** to manage user data, events, appointments, and academic resources.

In this table, we will define the **Create, Read, Update, Delete (CRUD)** operations for managing **user data**, **events**, **appointments**, and **academic resources**.

| **Entity** | **Create** | **Read** | **Update** | **Delete** |
| --- | --- | --- | --- | --- |
| **User** | Register new user | View user details | Update user details | Delete user account |
| **Event** | Create new event | View event details | Update event details | Delete event |
| **Appointment** | Book an appointment | View appointment details | Update appointment details | Cancel appointment |
| **Academic Resources** | Upload course material | View course material | Update course material | Delete course material |

* **Association Matrices** define the relationships between data entities and processes (e.g., which entities have access to certain data).

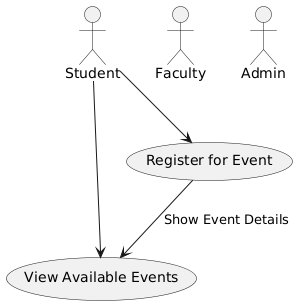
In this table, we define the relationships between **data entities** (e.g., **User**, **Event**, **Appointment**, **and Academic Resources**) and **processes** (e.g., **User Authentication**, **Event Registration**, **and Appointment Scheduling**). This will show which entities have access to certain data and processes.

Association Matrix for Habib Unified Portal

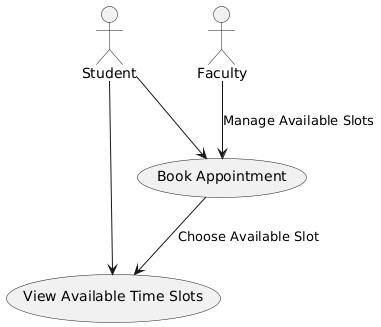
| **Process / Entity** | **User** | **Event** | **Appointment** | **Academic Resources** |
| --- | --- | --- | --- | --- |
| **User Authentication** | Create, Read | - | - | - |
| **Event Registration** | - | Create, Read, Update | - | - |
| **Appointment Scheduling** | - | - | Create, Read, Update, Delete | - |
| **Academic Resource Management** | - | - | - | Create, Read, Update, Delete |

**6.3 Use Cases**

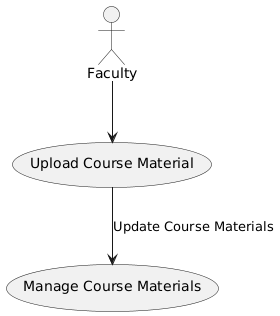
* **Use Case 1**: Register for Event



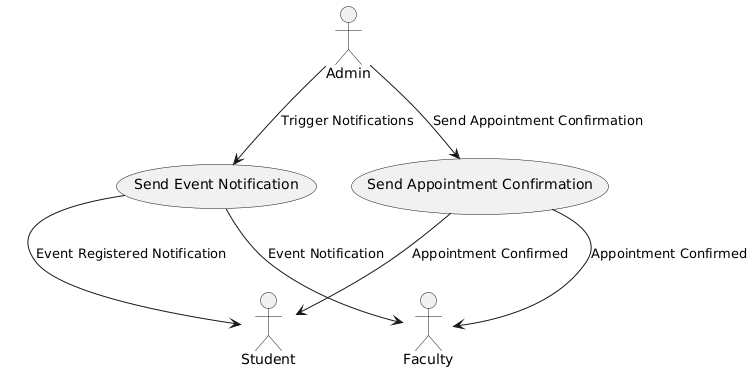
* The **Student** can view available events and register for an event.
* The use case diagram shows the interaction between the **Student** and the system (the **Habib Unified Portal**), where the system displays event details, and the **Student** registers for the selected event.
* **Use Case 2**: Book Appointment



* The **Student** views available time slots and books an appointment.
* The **Faculty** manages their available time slots.
* The **Student** and **Faculty** interact with the system to finalize the booking.
* **Use Case 3**: Upload Course Material



* The **Faculty** uploads and manages course materials in the portal, such as lecture notes or assignments.
* This use case highlights the interaction between the **Faculty** and the system in managing educational content.
* **Use Case 4**: Send Notifications



* The **Admin** triggers notifications for events and appointments.
* The system sends **Event Notifications** and **Appointment Confirmations** to the **Student** and **Faculty**.

**7. NON-FUNCTIONAL REQUIREMENTS**

**7.1 Performance Requirements**

* The system must be able to handle **500+ concurrent users** with minimal latency.

**7.2 Safety Requirements**

* **Data backup** will be performed regularly to ensure no data loss.

**7.3 Security Requirements**

* **User authentication** will be integrated with the university’s **SSO**.
* **Data will be encrypted**, and sensitive data (e.g., passwords) will be securely stored.

**7.4 User Documentation**

* A **user manual** will be provided for students, faculty, and administrators.
* The manual will include instructions for accessing academic resources, scheduling appointments, and registering for events.

**8. REFERENCES**

* University Authentication Documentation
* SQLite Documentation
* Flask Framework Documentation

**9. APPENDICES**

* **Appendix A**: Detailed use case descriptions.
* **Appendix B**: Database schema design.