Software Requirement Specification for
“Mark Entry Portal”

Name	DEEPAN KUMAR M
Roll no	7376221EC138
Seat no	242
Project ID	14
Problem Statement	MARK ENTRY

1. Introduction:

1.1. Purpose:

- The goal of this work is to provide an entire overview of the Mark Entry system. It will explain the system's goal and features, as well as its interfaces, what the system will do, the limitations under which it must operate, and how the system will respond to input from outside sources.

1.2. Scope of Project:

- This software system will serve as a module for academic management, enabling the entry of marks for periodical tests, calculation of CO-PO (Course Outcome - Program Outcome) mapping, and result analysis. The system will provide a comprehensive analytical dashboard for academic oversight.
- The system includes features such as the entry of marks for periodical tests, calculation of CO-PO (Course Outcome -

Program Outcome) mapping, comprehensive result analysis, and an analytical dashboard for academic oversight. Additionally, it supports role-based access for instructors, administrators, and other users involved in the academic process, ensuring each user has access to the functionalities relevant to their role.

2. System Overview:

2.1. Users:

1. Instructors:

- Enter and manage periodical test marks.
- Perform CO-PO calculations.
- Analyze results and generate reports.

2. Administrators:

- Oversee the entire mark entry process.
- Access analytical dashboards for comprehensive result analysis.
- Manage user access and permissions.

2.2. Features:

1. Login and Registration:

- Users can register for an account or log in with their existing account.

2. Mark Entry:

- Instructors can input periodical test marks CO-wise for students.

3. CO-PO Calculation:

- System performs calculations to map Course Outcomes (CO) to Program Outcomes (PO).

4. Result Analysis:

- Analytical tools for result analysis, allowing instructors to view detailed performance metrics.

5. Admin Access:

- Admins can view all mark entries, perform CO-PO calculations, and access analytical dashboards.

6. Analytics Dashboard:

- Provides insights into student performance, CO-PO mapping, and overall academic metrics.

3. System Requirements Specification

3.1. Functional Requirements

1. User Management:

- Users can register and log in.
- Admins have access control with an analytical dashboard and dedicated features.

2. Mark Entry:

- Instructors can enter periodical test marks CO-wise.
- Data entry forms include fields for student details and marks.

3. CO-PO Calculation:

- System calculates the mapping of Course Outcomes to Program Outcomes based on entered marks.

4. Result Analysis:

- Provides detailed analysis of student performance, CO-PO alignment, and other metrics.

5. Admin Dashboard:

- Admins can view and manage all mark entries.
- Includes tools for CO-PO calculations and result analysis.

3.2. Non-Functional Requirements

1. Performance:

- The system must respond to user actions within 2 seconds to ensure efficient usability and handle a concurrent user load of at least 100 users without significant performance degradation.

2. Security:

- User data must be encrypted during transmission and storage, and access to sensitive functionalities should be restricted to authorized admin users through secure authentication mechanisms.

3. Usability:

- The user interface should be intuitive and user-friendly, with clear and concise error messages to guide users in case of input errors or system failures.

4. Reliability:

- The system should be available 24/7 with minimal downtime and should have a backup and recovery mechanism to prevent data loss in case of system failures or crashes.

5. Scalability:

- The system should be designed to accommodate an increasing number of users and data volume over time and should be scalable to support additional features and functionalities as per future requirements.

4. System Architecture:

4.1. Backend Entities:

1. User Entity:

Name	String
Email	String
Password	Hash code
Role	String (Instructor/Admin)

2. Marks Entity:

StudentId	String
TestId	String
Date	Date

Marks	Array of Objects (CO-wise marks)

3. CO-PO Mapping Entity:

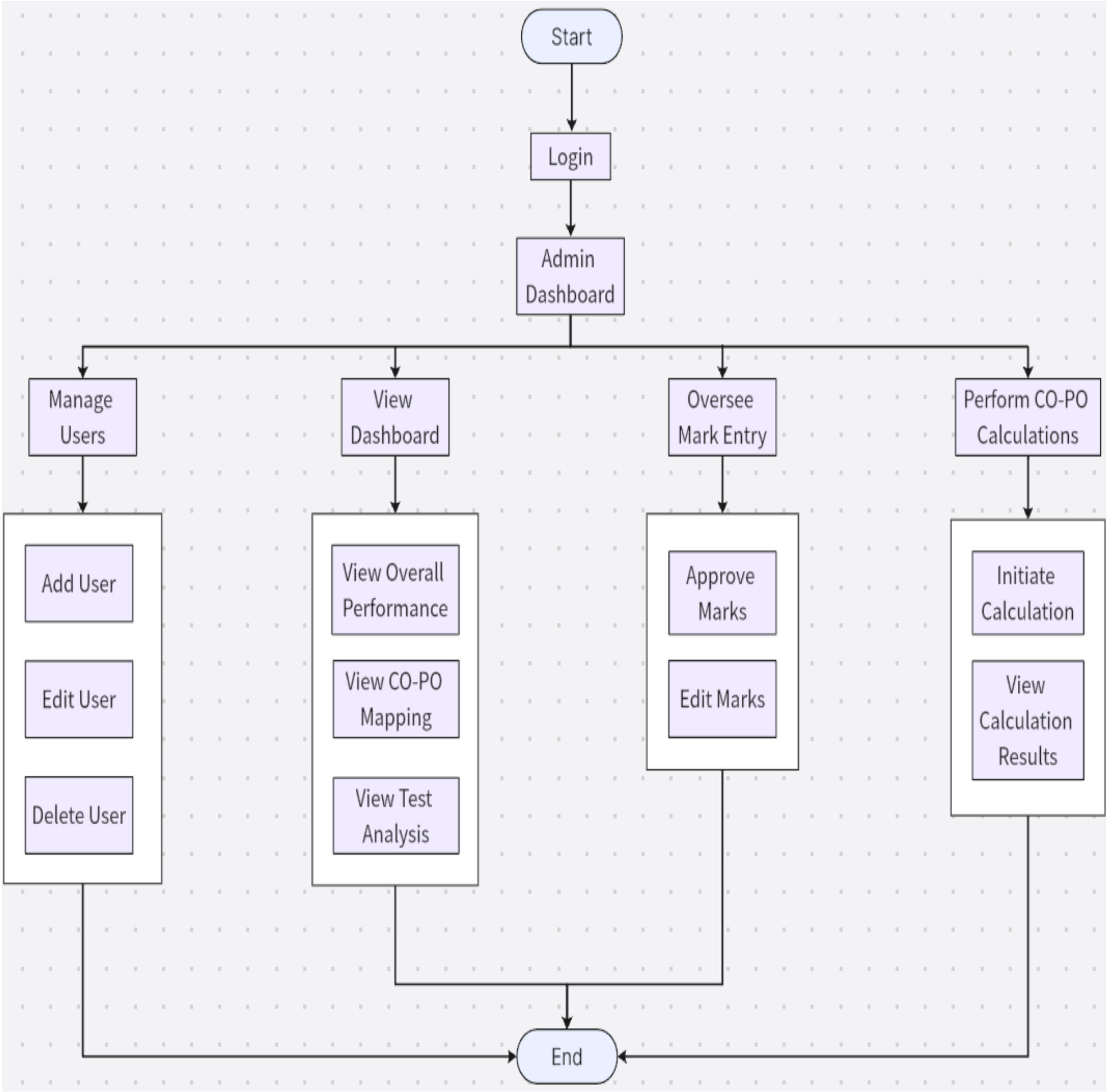
CourseId	String
COs	Array of Objects
POs	Array of Objects
Mapping	Array of Objects (CO-PO relations)

4.2. Technology Stack:

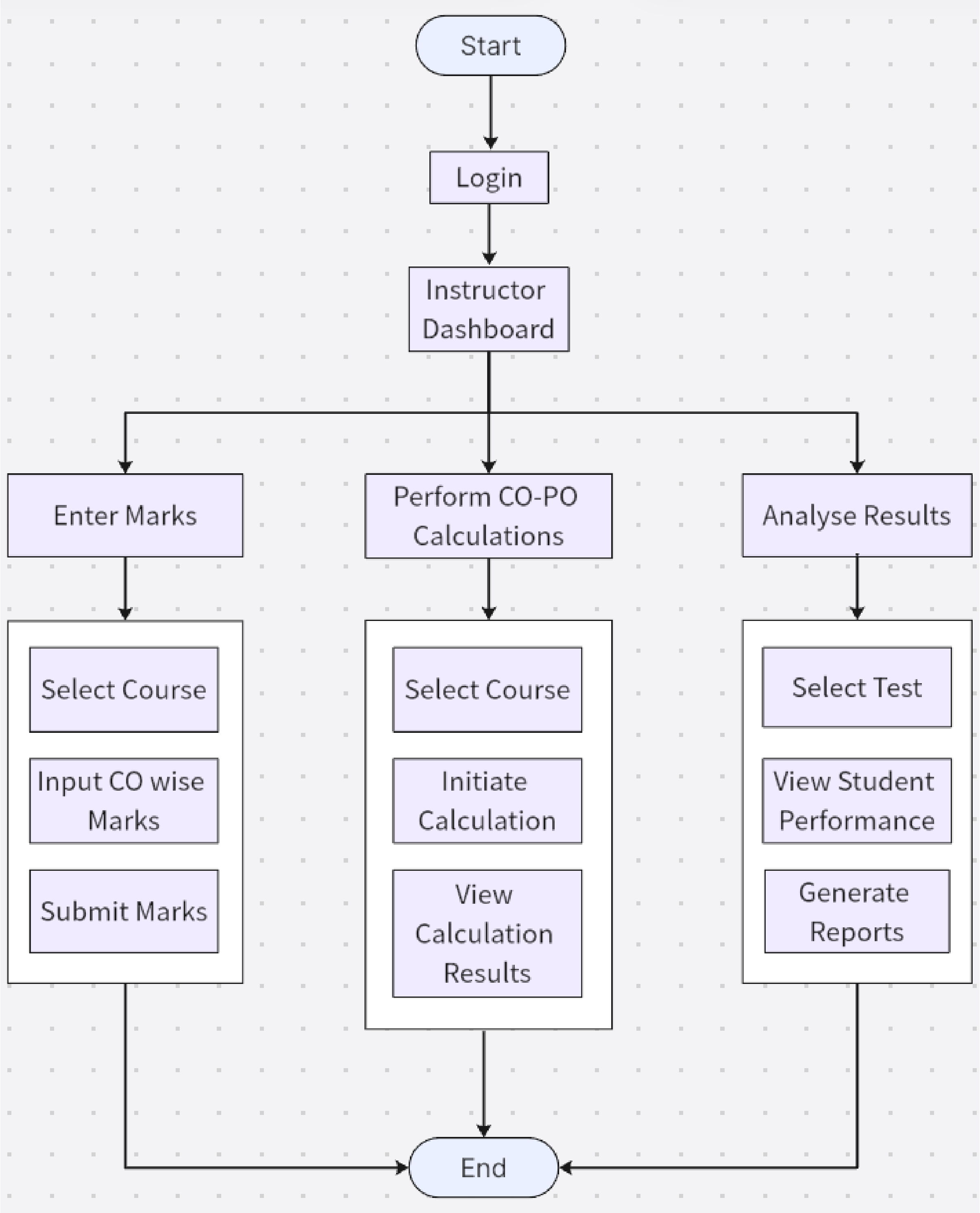
Front-End	1. HTML 2. CSS 3. JavaScript
Back-End	1. Django 2. Python
Database	1. MySQL

5. Work Flow Diagram:

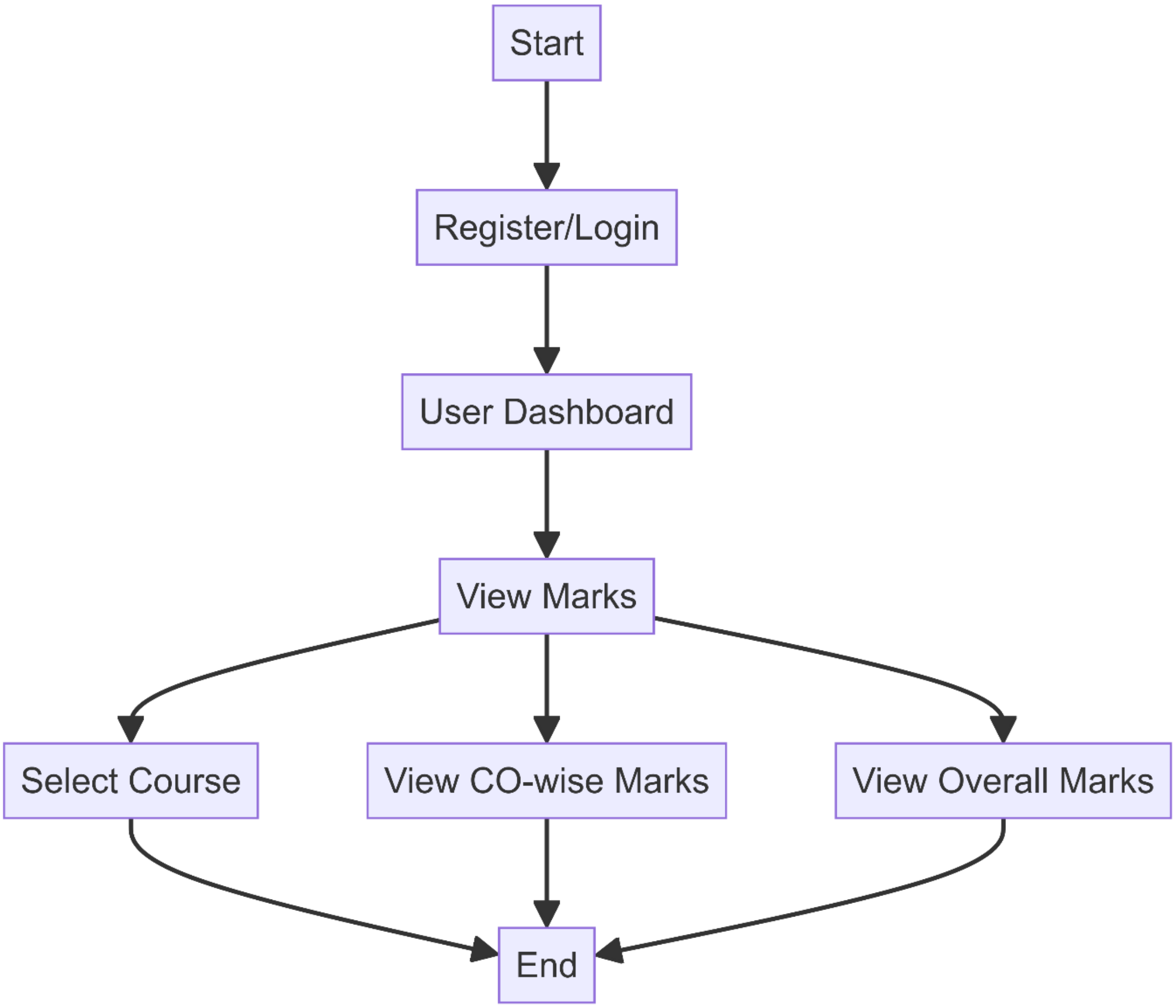
5.1. Admin Interface:



5.2. Instructor Interface:



5.3. User Interface:


Start
Register/Login
User Dashboard
View Marks
Select Course
View CO-wise Marks
View Overall Marks
End