**Scrum Framework**

**Student Management Portal**

**Sprint 1: User Authentication and Dashboard UI**

1. User Authentication (Backend - Spring Boot, JWT, Bcrypt)

- Implement a JWT-based authentication system to ensure secure login.

- Create APIs for user registration, login, and logout.

- Store passwords securely using bcrypt hashing.

- Define user roles (Admin, Faculty, Student) with proper access control.

2. Frontend Authentication (React/Angular)

- Develop Login and Registration pages with proper form validation.

- Implement conditional rendering based on authentication status.

- Secure frontend routes to restrict unauthorized access to dashboard and management pages.

3. Home Page UI Design

- Design an intuitive dashboard for users after login.

- Implement a navigation bar for easy access to different sections.

- Add a welcome banner and quick action buttons for a better user experience.

**Sprint 2: Student and Course Management**

Tasks:

1. Student Management (Backend & Database - Spring Boot, MySQL)

- Design a Student entity with details like name, roll number, course, email, etc.

- Implement \*\*APIs\*\* to \*\*add, update, view, and delete\*\* student records.

- Ensure \*\*data validation\*\* (e.g., unique roll numbers, valid email formats).

- Enable \*\*search and filtering\*\* (by name, course, or roll number).

2. Course Management (Backend & Database - Spring Boot, MySQL)

- Design a Course entity with course name, code, and faculty details.

- Implement APIs to add, update, view, and delete courses.

- Allow students to enroll in courses and view their enrolled courses.

- Ensure proper data relationships (students linked to courses).

3. Frontend Integration (React/Angular)

- Design pages for adding and managing students & courses.

- Display student lists and course lists with options to edit/delete.

- Implement search and filter features for easy navigation.

- Provide forms with validation for adding/updating records.

**Sprint 3: Attendance Tracking and Reporting**

Tasks

1. UI Design: Create attendance management pages with a responsive layout.

2. Attendance Marking: Implement a form for instructors to mark attendance for students.

3. Attendance Records: Display attendance data in a table with filters (student, course, date range).

4. API Integration: Connect UI with backend APIs for fetching and updating attendance.

5. Report Generation & Export: Add options to download attendance reports in PDF/Excel format.

**Implementation**

1. Database Design: Create an `attendance` table linking students and courses with attendance status (`PRESENT`/`ABSENT`).

2. Entity & Repository: Define JPA entities for `Attendance`, `Student`, and `Course`, and implement repository methods for attendance retrieval.

3. Service Layer: Implement business logic for marking attendance and fetching records.

4. Controller Layer: Create REST APIs for marking, fetching, and generating attendance reports.

5. Reporting & Export: Implement functionality to generate reports in PDF/Excel.