2.Scrum Master: Specify Sprint Backlog

Sprint 1 (1 week):

1. **User Story 1: Account Management (Points: 3)**
    - **Task 1:** Implement user registration functionality.
        - Subtask: Design registration form UI.
        - Subtask: Implement backend logic to handle user registration.
    - **Task 2:** Develop a secure login system.
        - Subtask: Design login form UI.
        - Subtask: Implement backend logic for secure user authentication.
    - **Acceptance Criteria:**
        - Users can successfully register for an account.
        - Users can securely log in.
2. **User Story 2: Course Management (Points: 5)**
    - **Task 4:** Create a course database schema.
        - Subtask: Define database tables for storing course information.
    - **Task 5:** Implement course search functionality.
        - Subtask: Design course search UI.
    - **Task 6:** Enable students to join and drop courses.
        - Subtask: Implement UI elements for joining and dropping courses.
        - Subtask: Implement backend logic for managing course registrations.
    - **Acceptance Criteria:**
        - Students can search for available courses.
        - Students can join and drop courses successfully.

Sprint 2 (1 week):

1. **User Story 3: Course Details (Points: 5)**
    - **Task 7:** Design and implement the course details interface.
        - Subtask: Define UI components for displaying course details.
    - **Task 8:** Retrieve and display relevant information for a selected course.
        - Subtask: Implement backend logic to fetch and display course details.
    - **Acceptance Criteria:**
        - Students can view detailed information about a course.

2. **User Story 4: Leave Feedback (Points: 8)**
    - **Task 9:** Create the feedback database schema.
        - Subtask: Define database tables for storing feedback.
    - **Task 10:** Develop the interface for leaving feedback.
        - Subtask: Design feedback form UI.
        - Subtask: Implement backend logic to handle feedback submissions.
    - **Task 11:** Implement logic to restrict one feedback per course per student.
        - Subtask: Ensure backend enforces the one-feedback-per-course rule.
    - **Acceptance Criteria:**
        - Students can leave feedback for registered courses.
        - The system enforces one feedback per course per student.

Sprint 3 (1 week):

1. **User Story 5: View Feedback (Points: 5)**
    - **Task 12:** Design the feedback display interface.
        - Subtask: Define UI components for displaying feedback.
    - **Task 13:** Implement sorting by date and display average ratings.
        - Subtask: Implement backend logic for sorting feedback by date.
        - Subtask: Calculate and display average ratings for each course.
    - **Acceptance Criteria:**
        - Students can view feedback for any course.
        - Feedback is displayed in chronological order with average ratings.
2. **User Story 6: View Top-rated Courses (Points: 3)**
    - **Task 14:** Develop the interface to display top-rated courses.
        - Subtask: Design UI components for displaying top-rated courses.
    - **Task 15:** Implement logic to calculate and display top-rated courses.
        - Subtask: Create backend functions to determine top-rated courses.
    - **Acceptance Criteria:**
        - Students can view the top 8 rated courses.

Sprint 4 (1 week):

1. **User Story 7: Course Calendar (Points: 5)**
    - **Task 16:** Design the course calendar interface.

        - Subtask: Define UI components for displaying courses in a calendar format.
    - **Task 17:** Implement functionality to display all courses in a calendar.
        - Subtask: Develop backend functions to retrieve and display courses in a calendar.
    - **Acceptance Criteria:**
        - Students can view all courses in a calendar.
2. **User Story 8: Remove Old Feedback (Points: 3)**
    - **Task 18:** Implement a scheduled task to identify and remove old feedback.
        - Subtask: Create a scheduled task for automated removal.
    - **Task 19:** Develop a log or record for auditing purposes.
        - Subtask: Design and implement a log for tracking removed feedback.
    - **Task 20:** Write Unit tests using the Junit testing framework.
    - **Acceptance Criteria:**
        - Feedback older than 1 year is automatically removed.
        - Tests reach no less than 50% of code coverage.