IT 314

SOFTWARE ENGINEERING

GROUP 6

AUTO ATTENDANCE SYSTEM

INDEX

1. PROBLEM STATEMENT

APP for automated attendance in Lab/Lectures This project is an APP application that provides the students an option to automatically mark attendance in the lecture and lab sessions. The instructor opens the sessions for a specific duration and students have to automatically mark the same. Auto Attendance reduces the time of marking attendance and encourages proxies.

2. NEED FOR THIS SYSTEM

The need for this system is as follows:

1) To save user time, cost, and institute resources.
2) The system is helpful as it generates a systematic overall report of every class attendance.
3) It maintains the record in a large database instead of the conventional method of maintaining a register or a biometric attendance, which further simplifies the process of searching for a particular record.
4) It provides accuracy.
5) This web application records data automatically.
6) To provide a good analysis for students as well as for professors to view the student attendance data.
7)) Helps the student to be punctual as there is specific duration to mark the attendance.
8) Helps to lessen the paper-work and human efforts.

3. FEATURES OF THIS SYSTEM

The features of this system are as follows:

1) Provides a separate login page for students and Instructors.

2) Instructors can create a new course and instructors can submit the student information in the portal to enrol that student.
3) During a lecture, Instructors can open an attendance frame for a specific period and during this period, students can mark their attendance.
4) Students will be able to view which lecture he/she has taken and which is missed.
5) Instructors will be able to view the dashboard of the attendance of students of each lecture and in each course.
6) Instructor will be able to download the report of attendance.
7) Student will be notified for his/her enrolment and attendance in a course.

4. FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS

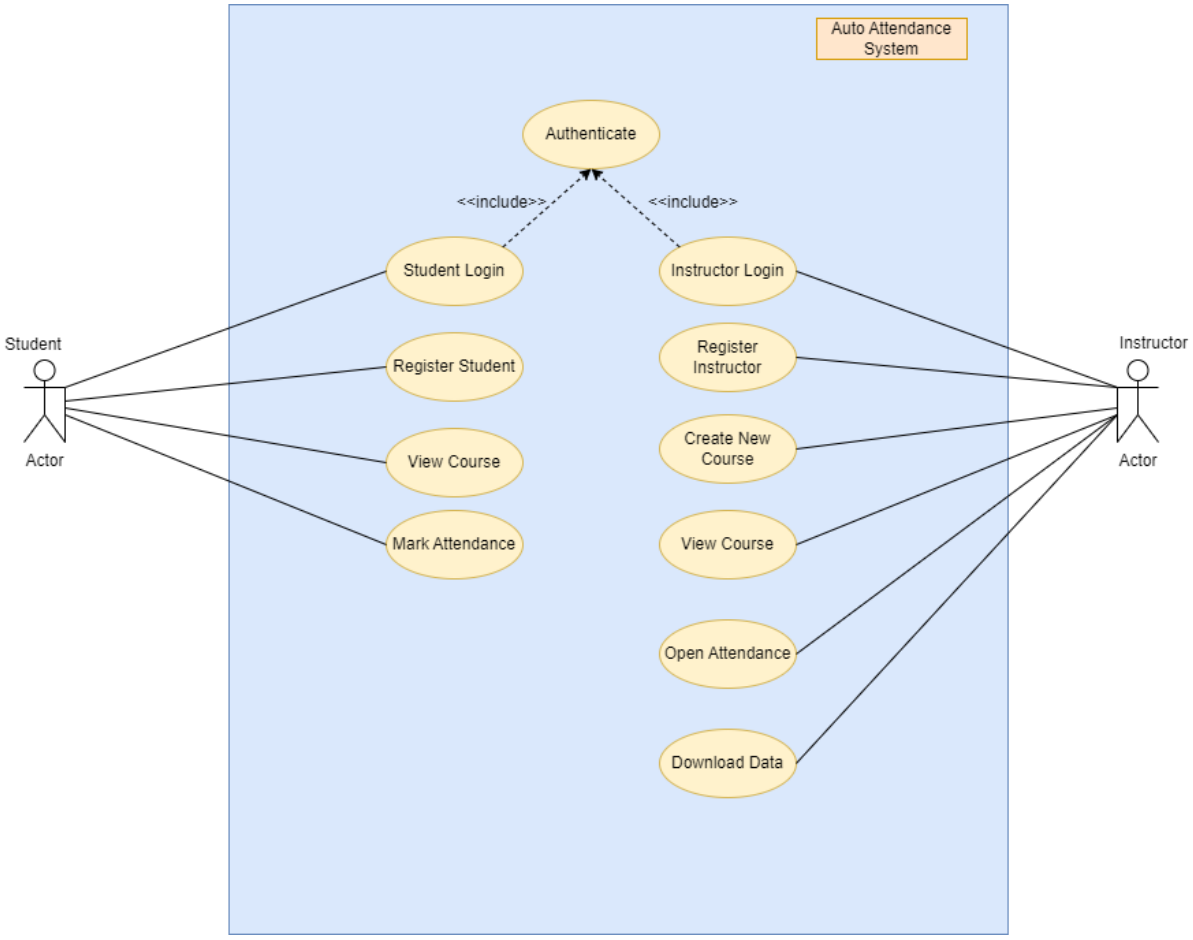
The functional requirements are as follows:

1) To provide a separate login feature for student and instructor.
2) Instructors will be able to create a new course and submit the student information in the portal to enrol that student.
3) Students will get enrolment confirmation mail for that course.
4) Students will be able to see their attendance graph.
5) Instructor will be able to specify the time frame for attendance.
6) Students will get mail which consists of a link for marking his/her attendance.
7) Instructors will be able to make multiple courses and can see the attendance graph for each course.

Non-Functional Requirements with justifications:

1) **Security:** The website should store the data securely and provide protection against threats.
2) **Compatibility and Portability:** The website should adapt according to the specifications of the user system. The website should support all the modern web browsers as well as older versions.
3) **Reliability and Availability:** The website should remain up 24*7 and the databases and servers should be ready and available to establish the connection all the time.
4) **Scalability and Performance:** Multiple users should be able to access the website at a time and the process of authentication and validation should be fast even for a high number of users at the same time.
5) **Localization:** The website should fit into the local specifics of the user system. It should be responsive to the screen size.
6) **Usability:** The UI should be easy to use for all kinds of users.(i.e. Technical as well as Non-Technical users)

5. USE CASE DIAGRAM



6. PROCESS MODEL

1) The process model that we had decided to use is the simple **waterfall model.**
2) The reason for using this is that our project is small.
3) Our requirements are already known.
4) Our requirements will not change frequently.
5) Waterfall model is simple to use.

7. USE CASE DESCRIPTION

1. Use Case: Student Registration

Use Case Id: UC_1

Goal: This describes how a student registers the Attendance system.

Actors: Student

Pre-condition: User must have a valid email-address.

Description:

1. AAS asks for valid credentials like first name, last name, password.
2. User enters the credentials.
3. AAS creates a new student account.

Post condition: AAS displays login page.

Exception:

2.a If student enter invalid email address.

2.a.1 System will again redirect to the registration page.

2. Use Case: Student Login

Use Case Id: UC_2

Goal: This describes how a student logs into the Attendance system.

Actors: Student

Pre-condition: User must create a valid account using UC_1

Description:

1. AAS asks for valid credentials.
2. User enters the credentials.
3. AAS authenticates and allows the user to log in.

Post condition: AAS displays relevant homepage.

Exception:

3.a. Authentication fails.

3.a.1 AAS displays the error message.

3.a.2. AAS again returns back to the login page.

3. Use Case: View course

Use Case Id: UC_3

Goal: This shows how to view the course.

Actors: Student

Pre-condition: Students should be successfully logged in the system by performing UC_2.

Description:

1. AAS shows the list of all the courses that the student has enrolled.
2. Students click on the particular course that he/she wants to inspect.
3. AAS shows the valid course page which consists of attendance statistics.

Post condition: AAS displays course page.

Exception:

None.

4. Use Case: Mark Attendance

Use Case ID: UC_3

Goal: To mark 'Present' for the student in particular course.

Actors: Student

Pre-condition:

1. Students should be successfully logged in to the website using UC_2 and he should be able to view the courses using UC_3.
2. Students should be notified about the attendance in a particular course.

Description:

1. Students will go to the list of courses and select a particular ongoing course-lecture.
2. AAS will validate the course-lecture if the attendance-marking portal is active.
3. Students will click on 'Mark my attendance'.

Post-condition:

Students will get a confirmation status about attendance of that course-lecture.

Exception:

2.a. The attendance portal expired.

2.a.1 AAS shows an error message.

5. **Use Case:** Instructor Registration

Use Case Id: UC_5

Goal: This describes how a instructor registers the Attendance system.

Actors: Instructor

Pre-condition: Instructor must have a valid email-address.

Description:

1. AAS asks for valid credentials like first name, last name, password.
2. User enters the credentials.
3. AAS creates a new student account.

Post condition: AAS displays login page.

Exception:

2.a If instructor enter invalid email address.

2.a.1 System will again redirect to the registration page.

6. Use Case: Instructor Login

Use Case Id: UC_6

Goal: This describes how an instructor logs into the attendance system.

Actors: Instructor

Pre-condition: Instructor must create a valid account using UC_5.

Description:

1. AAS asks for valid credentials.
2. Instructor enters the credentials.
3. AAS authenticates and allows the user to log in.

Post condition: AAS displays relevant homepage.

Exception:

3.a. Authentication fails.

3.a.1 AAS displays the error message.

3.a.2. AAS again returns back to the login page.

7. Use Case: Create new course

Use Case Id: UC_7

Goal: This shows how to create a new course.

Actors: Instructor

Pre-condition: Instructor should be successfully logged in the system by performing UC_6.

Description:

1. Instructor clicks on the “Create a new course” button.
2. AAS displays create a new course page.
3. Instructor enters all the course information that the AAS asks.
4. Instructor clicks on the submit button.
5. AAS shows the success message.

Post condition: AAS displays the valid course page that the instructor has newly created.

Exception:

None.

8. Use Case: View course

Use Case Id: UC_7

Goal: This shows how to view the course.

Actors: Instructor

Pre-condition: Instructor should be successfully logged in the Auto-Attendance System by performing UC_6.

Description:

1. AAS shows the list of all the courses that he/she has created using UC_7.
2. Instructor clicks on the particular course that he/she wants to inspect.
3. AAS shows the valid course page which consists of attendance statistics.

Post condition: AAS displays the valid course page.

Exception:

None.

9. Use Case: Open attendance

Use Case Id: UC_9

Goal: This shows how to open a new attendance portal.

Actors: Instructor

Pre-condition: Instructor should be successfully logged in the system by performing UC_6 and instructor should be able to view that course that he/she wants to open attendance using UC_8.

Description:

1. Instructor opens the course page.
2. Instructor clicks on the “Open new attendance” button.
3. AAS displays a form.
4. Instructor enters the lecture name and time frame.
5. AAS validates and shows the confirmation page.

Exception:

5.a. AAS validation fails.

5.a.1 AAS shows the error message.

5.a.2 AAS redirects the user to the same form page.

Post condition: AAS returns back to the same course page with the confirmation message.

10. Use Case: Download data

Use Case Id: UC_10

Goal: This shows how to download attendance data.

Actors: Instructor

Pre-condition: Instructor should be successfully logged in the system by performing UC_6 and instructor should be able to view that course using UC_8.

Description:

1. Instructor opens the course page.
2. Instructor clicks on the “Download data” button.
3. AAS starts downloading the attendance data for that course.

Post condition: AAS returns back to the same course page with the confirmation message.

11. Use case: Authentication

Use case: UC_11
Goal: To authenticate user's credentials.
Actors: AAS
Pre-condition: None
Description:
Post condition: AAS shows success or error message.

8.TOOLS AND TECHNOLOGY

1. **Front-end:** We are planning to use html, css and javascript and also to make it responsive as well as beautiful we are planning to use designing framework like bootstrap, tailwind css.
2. **Backend:** To make backend we are planning to use express js, because it is fast, effective as well as more resources are available for express js.
3. **Database:** We are planning to use open source NoSQL database MongoDB, as it is flexible and scalable.
4. **IDE:** We are planning to use an open-source IDE namely Visual Studio Code integrated with Github Co-pilot.

9.EFFORT ESTIMATION USING USE CASE SIZE POINT

9.1 Unadjusted Use-Case Weight (UUCW)

Use-Case Complexity	Number of Transactions	Use-Case Weight
Simple	≤3	5
Average	4 to 7	10
Complex	>7	15

Use case name	Number of transaction	Category
Authenticate	2	Simple
Student Login	2	Simple
View Course (For Students)	1	Simple
Mark Attendance	1	Simple
Instructor Login	2	Simple
Create New Course	1	Simple
View Course (For Instructors)	1	Simple
Open Attendance	1	Simple
Download Data	1	Simple
Student Registration	1	Simple
Instructor Registration	1	Simple

Use-Case Complexity	Weight	Number of Use-Cases	Product
Simple	5	11	55
Average	10	0	0
Complex	15	0	0
Unadjusted Use-case Weight (UUCW)			55

9.2 Unadjusted Actor Weight (UAW)

Actor Complexity	Example	Actor Weight
Simple	A System with defined API	1
Average	A System interacting through a Protocol	2

Complex	A User interacting through GUI	3

Actor Name	Category	Weight
Instructor	Complex	3
Student	Complex	3
Unadjusted Actor Weight (UAW)		6

9.3 Unadjusted Use Cast Point (UUCP)

Now, Unadjusted Use Case Point = Unadjusted Actor Weight(UUCW) + Unadjusted Use Case Weight(UAW)

Unadjusted Use Case Point = 6 + 55 = 61

2.4 Technical Complexity Factor (TCF)

Factor	Description	Weight (W)	Rated Value (0 to 5) (RV)	Impact (I = W × RV)
T1	Distributed System	2.0	4	8
T2	Response time or throughput performance objectives	1.0	5	5
T3	End user efficiency	1.0	4	4
T4	Complex internal processing	1.0	4	4
T5	Code must be reusable	1.0	4	4

T6	Easy to install	.5	1	0.5
T7	Easy to use	.5	4	2
T8	Portable	2.0	3	6
T9	Easy to change	1.0	4	4
T10	Concurrent	1.0	5	5
T11	Includes special security objectives	1.0	5	5
T12	Provides direct access for third parties	1.0	2	2
T13	Special user training facilities are required	1.0	0	0
Total Technical Factor (TFactor)				49.5

Technical Complexity Factor can be calculated as follows:

∴ TCF = 0.6 + (0.01 × TFactor)
∴ TCF = 0.6 + 0.01*49.5
∴ TCF = 0.6 + 0.495
∴ TCF = 1.095

2.5 Environmental Complexity Factor (EF)

Factor	Description	Weight (W)	Rated Value (0 to 5) (RV)	Impact (I = W × RV)
F1	Familiar with the project model that is used	1.5	5	7.5
F2	Application experience	.5	2	1

F3	Object-oriented experience	1.0	3	3
F4	Lead analyst capability	.5	4	2
F5	Motivation	1.0	3	3
F6	Stable requirements	2.0	4	8
F7	Part-time staff	-1.0	0	0
F8	Difficult programming language	-1.0	2	-2
Total Environment Factor (EFactor)				22.5

Environmental Factor = 1.4 + (-0.03 × EFactor)=0.725

Factor	Description	Weight
UUCP	Unadjusted use case point	51
TCF	Technical Complexity Factor	1.095
EF	Environmental factor	0.725

UCP = UUCP × TCF × EF
UCP = 48.426375

Total Working Hours = UCP X Working Hours/UCP
Working Hours/UCP = 15
Total Working Hours = 40.487625 X 15
Total Working Hours = 726.395625 Hours

10.DOMAIN ANALYSIS

Entities:

1. **Students**: The individuals who attend the classes and whose attendance is being tracked.
2. **Instructors**: The individuals who are responsible for taking attendance and managing the courses.
3. **Courses**: The courses that are being taught and the specific sessions that students attend.
4. **Attendance records**: The data that is collected about each student's attendance for each class session.

Processes:

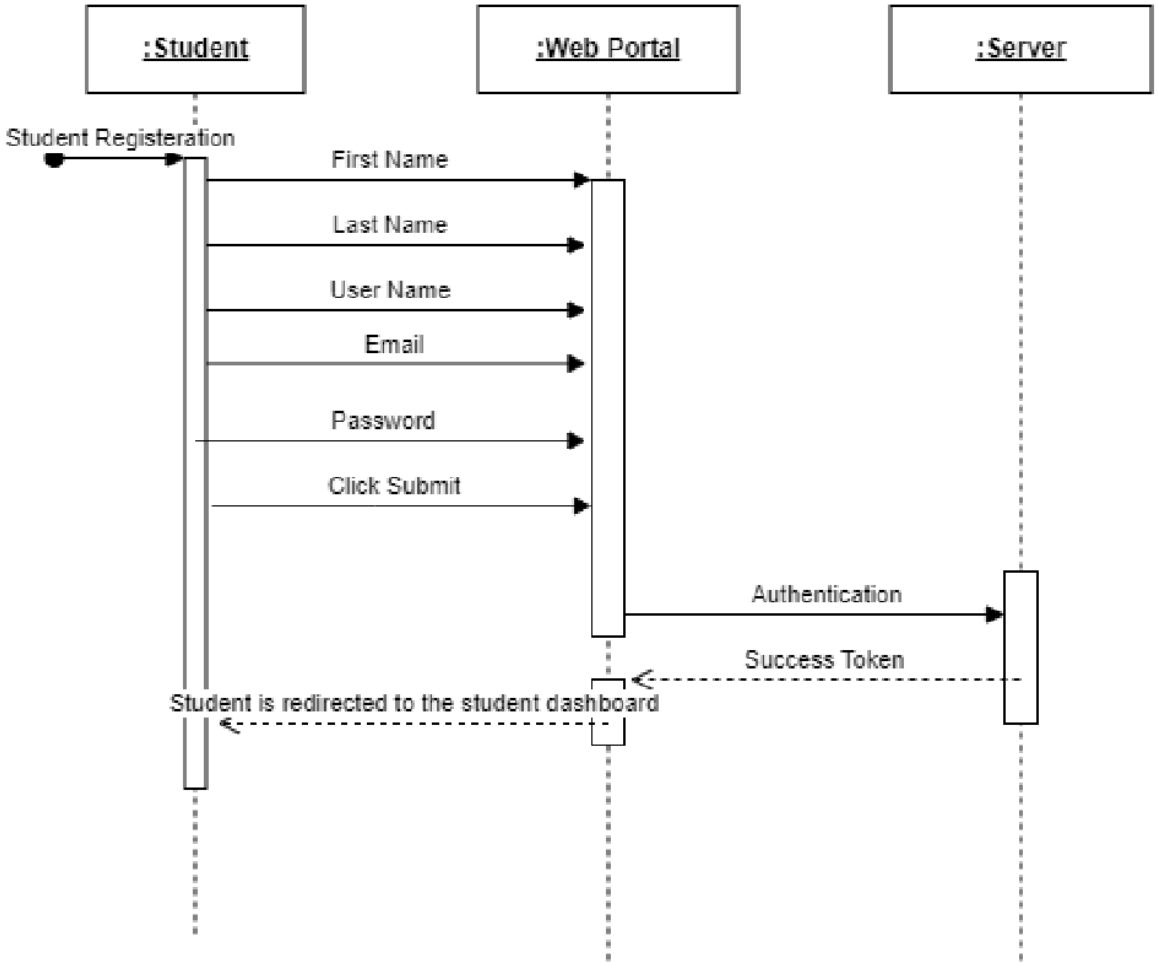
1. **Taking attendance**: The process by which instructors record the attendance of each student in a given class session.
2. **Automatic attendance**: The process by which the system automatically detects and records the attendance of students using methods such as facial recognition or geolocation.
3. **Managing attendance records**: The process by which instructors and administrators access and manage the attendance records for each class session.

Relationships:

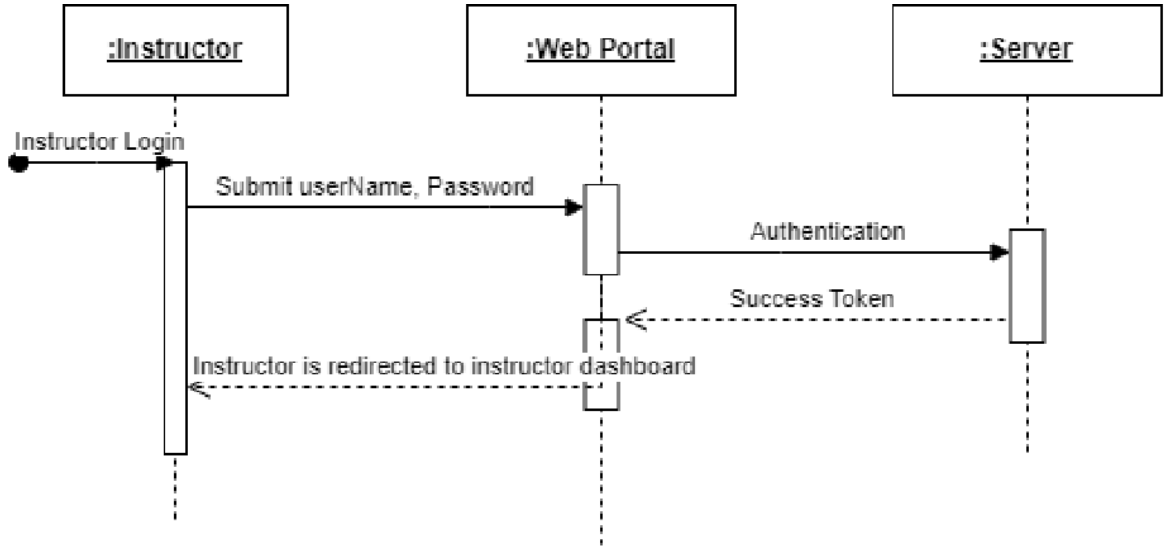
1. Students are enrolled in classes and attend specific class sessions.
2. Instructors are responsible for taking attendance and managing class sessions.
3. Attendance records are associated with specific class sessions and students.

11.SEQUENCE DIAGRAM

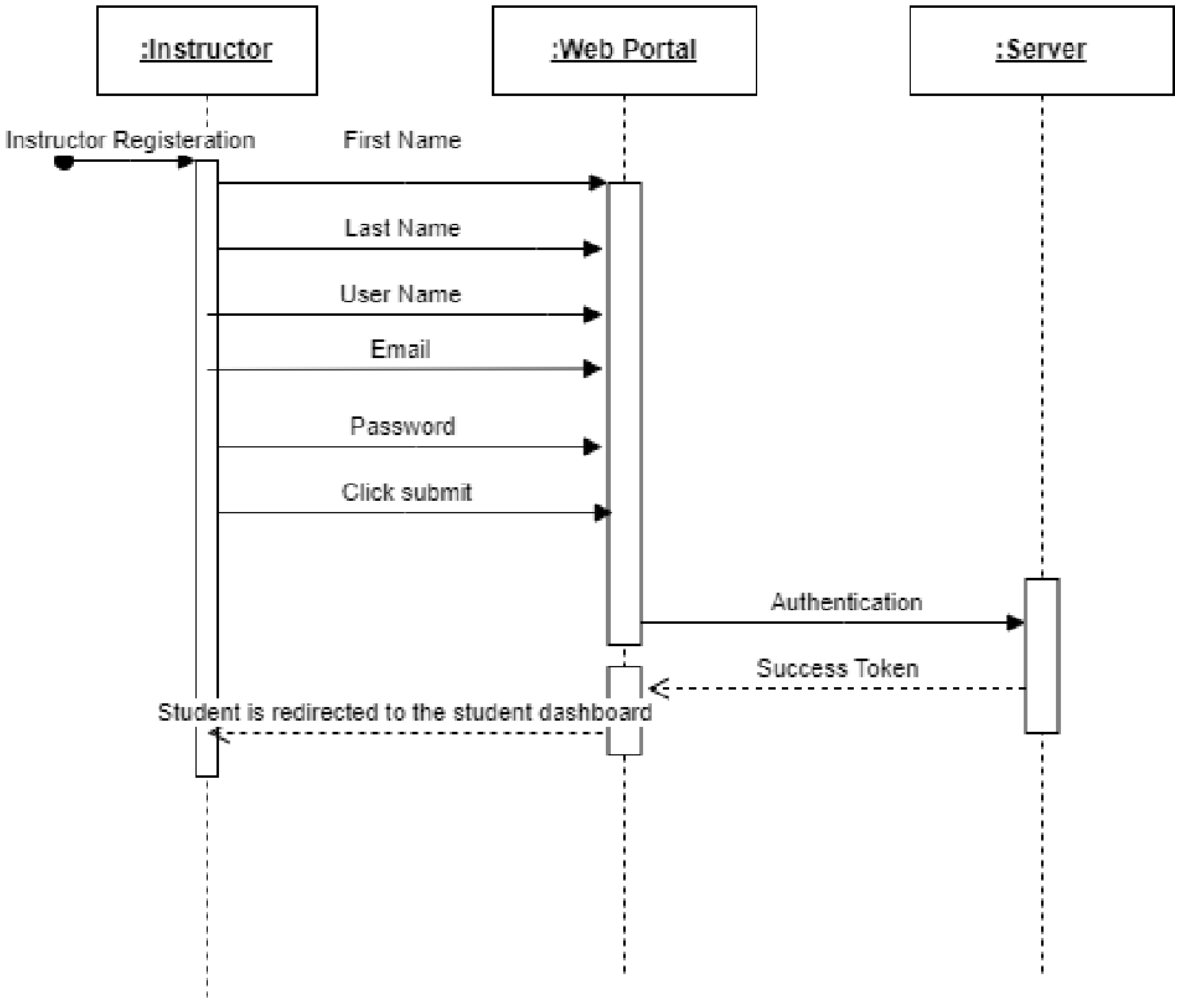
Student Login



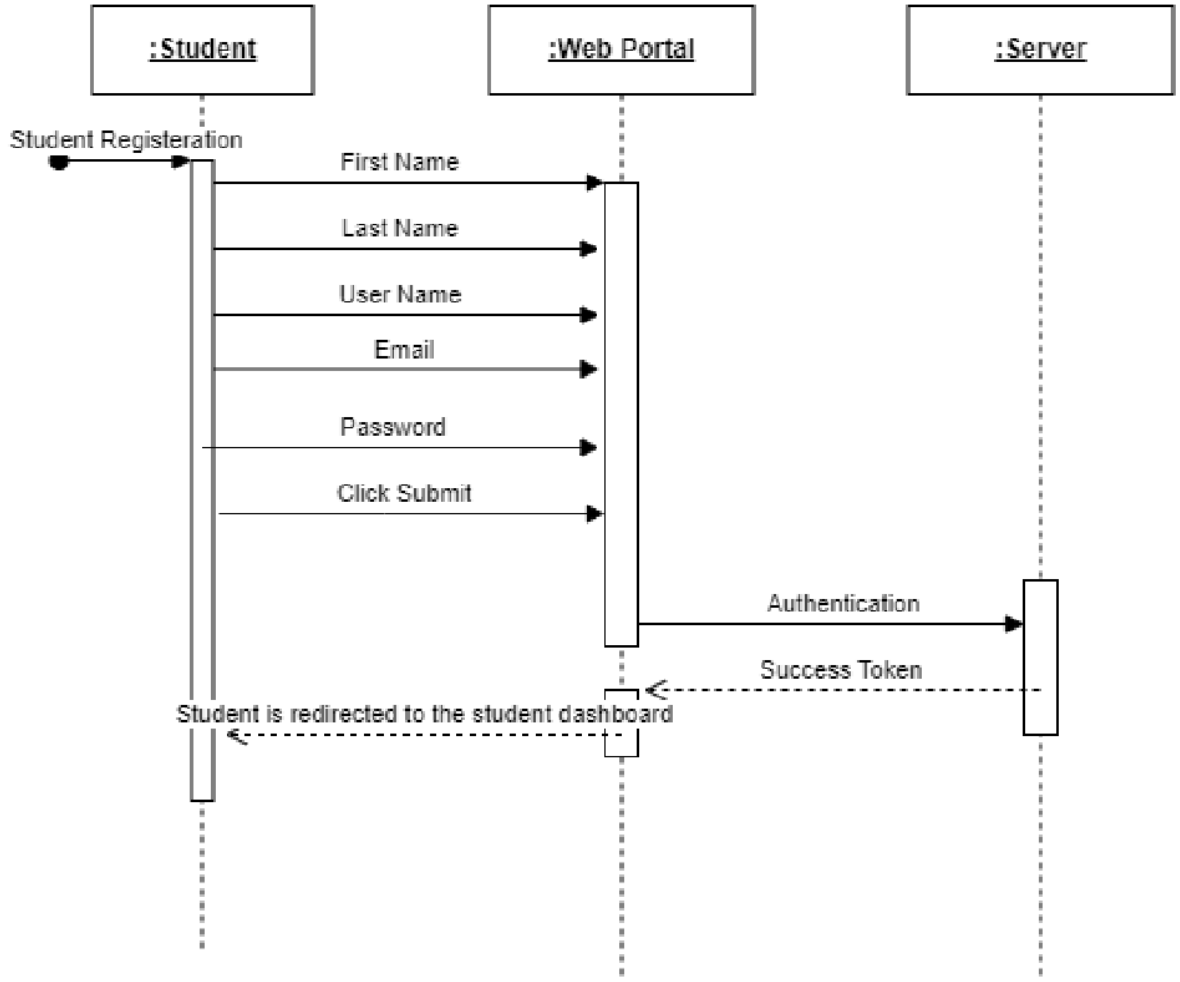
Instructor Login



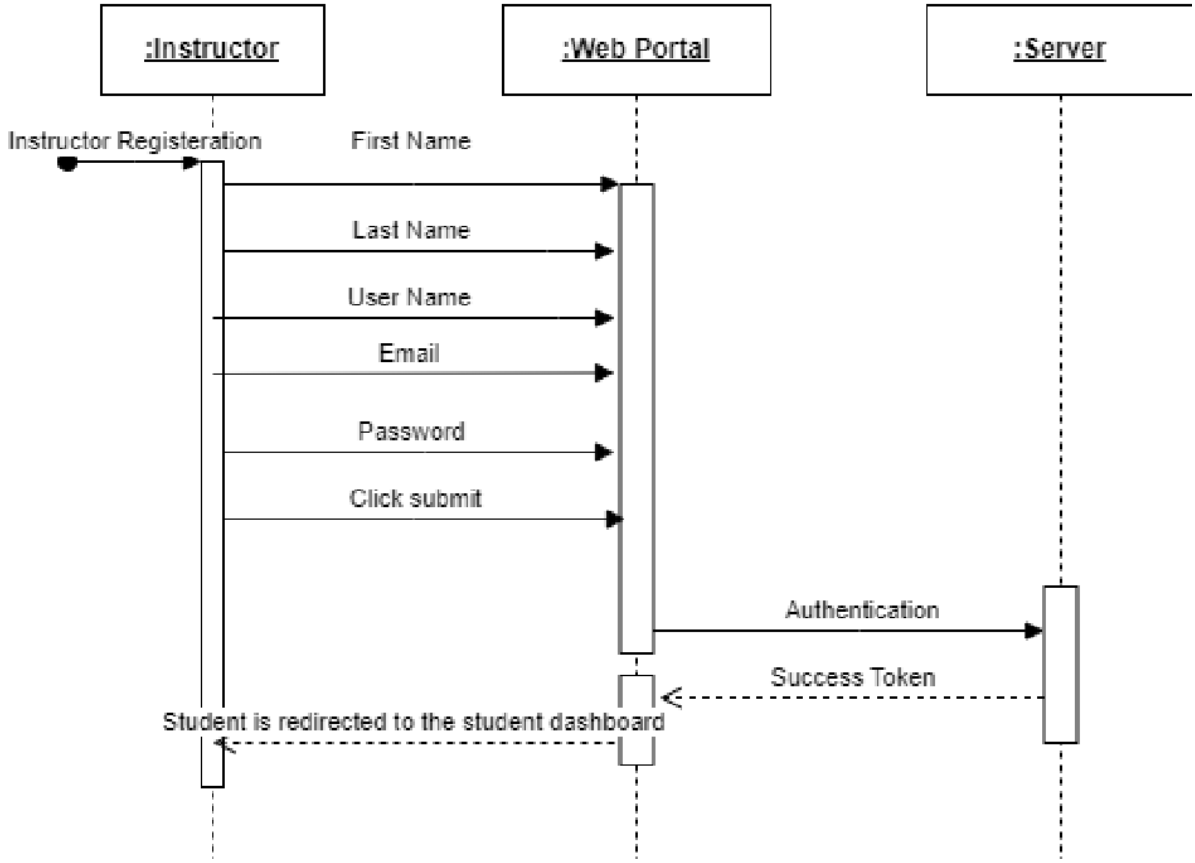
Instructor Registration



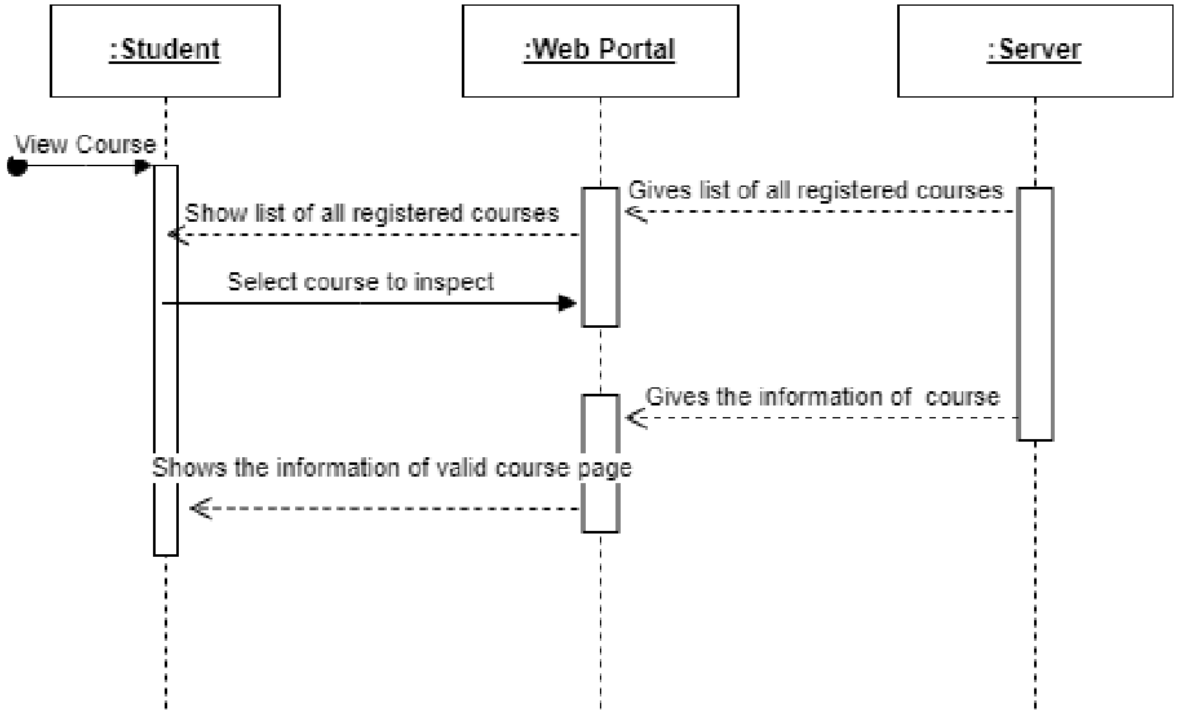
Student Registration



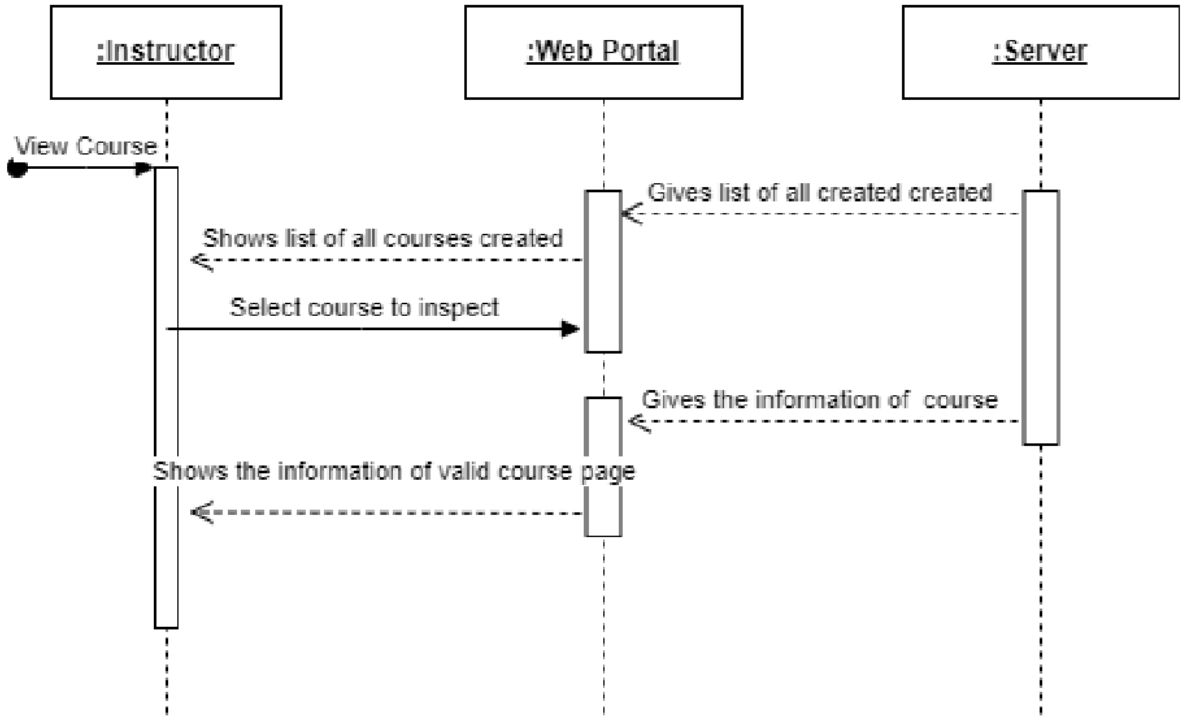
Instructor Registration



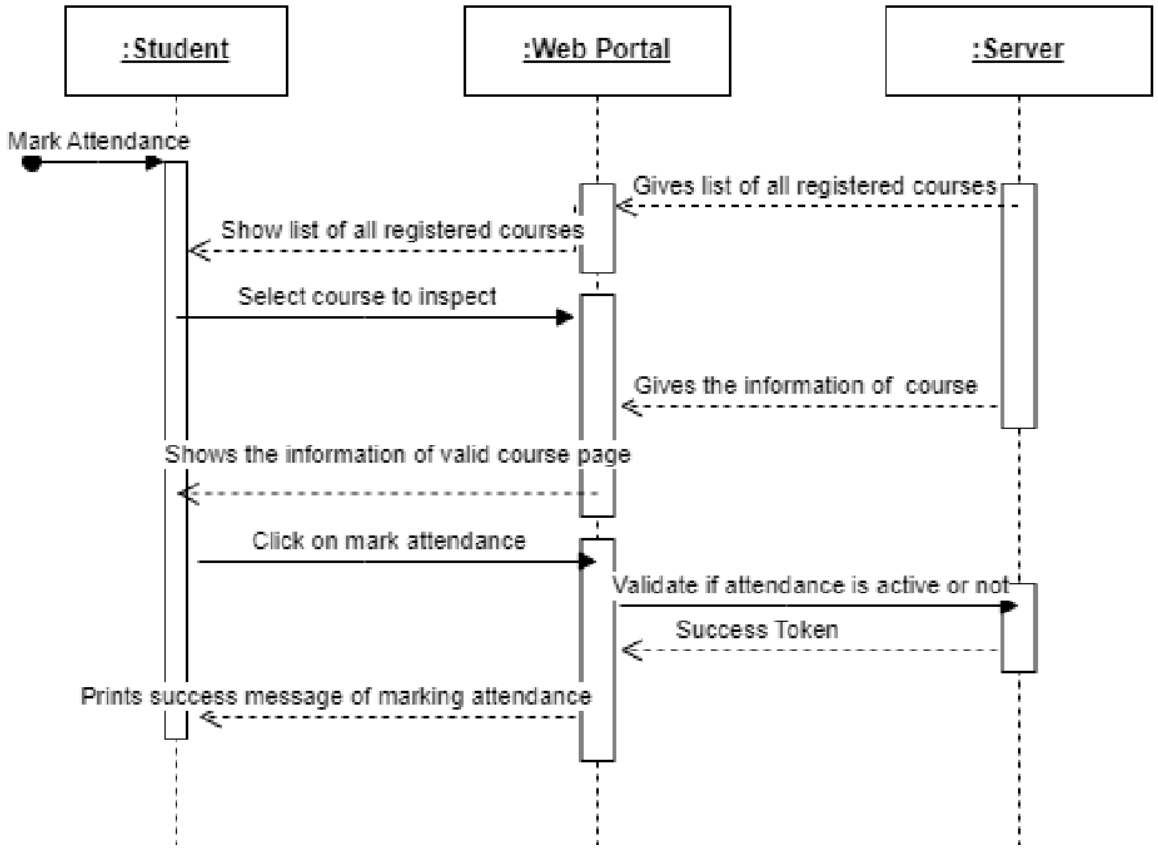
View course



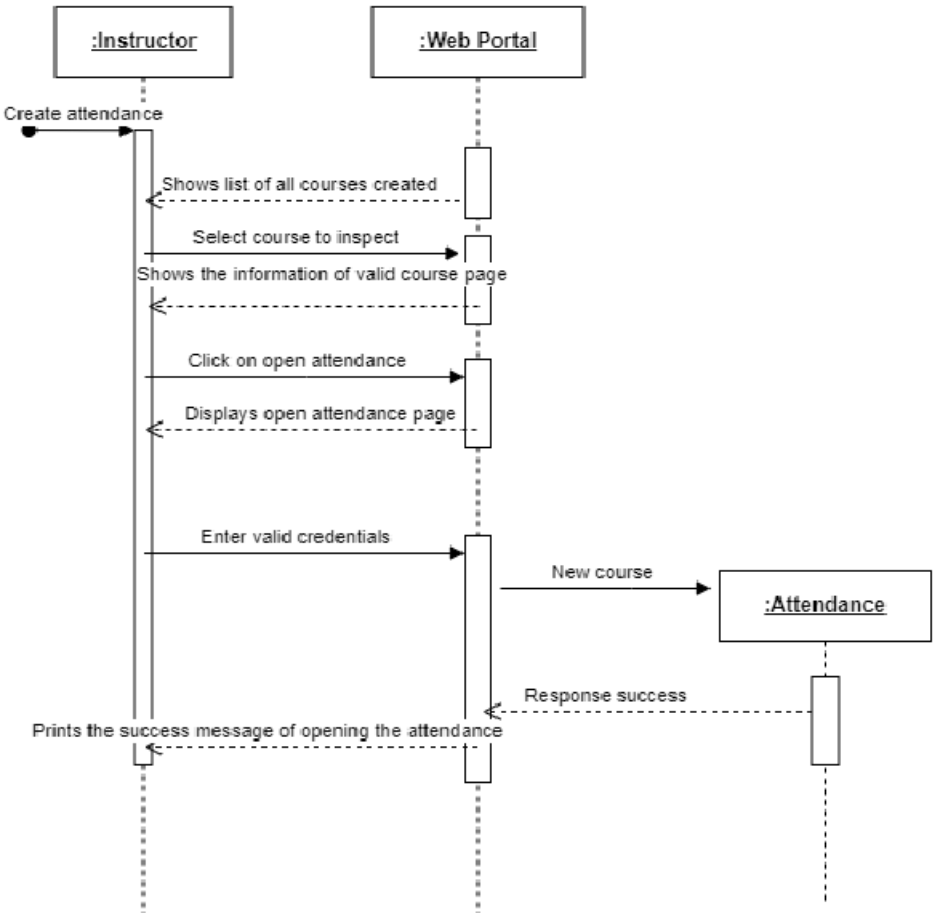
View course for instructor.



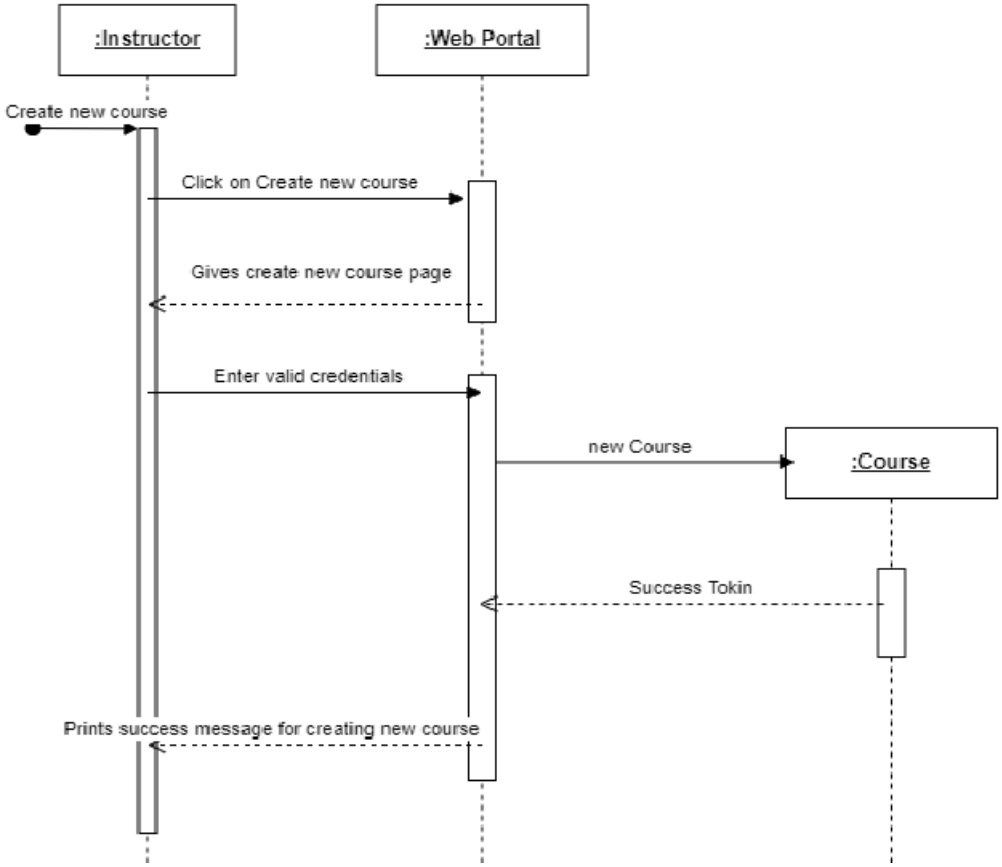
Mark attendance for student.



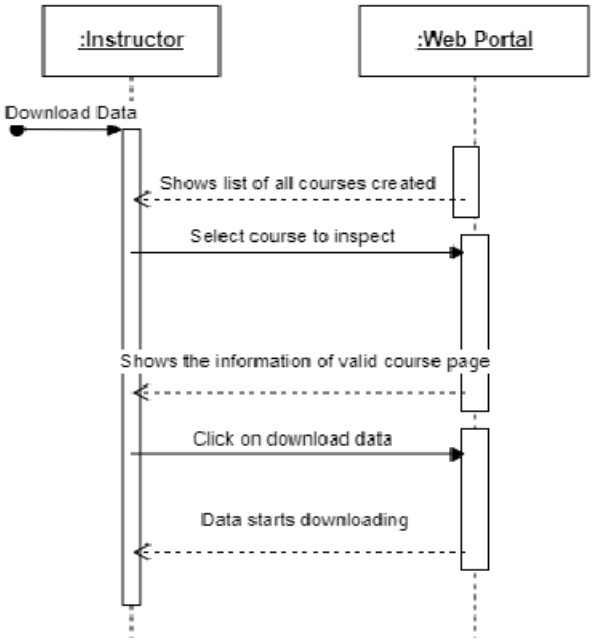
Create new attendance for instructor.



Create new course for instructor.



Download data for instructor



12.CLASS DIAGRAM