**(Object Oriented Software Engineering)**

*Submitted by*

**Yuv Garg (2K21/SE/198)**

**Varun Goyal (2K21/SE/189)**

**Yash Gupta (2K21/SE/195)**

*Submitted to*

**Prof. Ruchika Malhotra**

**Head Of Department**

**Software Engineering**



Delhi Technological University

Bawana Road, Delhi - 110042

2022- 2023

INDEX

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sno.** | **Title** | **Page no.** | **Date** | **Signature** |
| 1 | Problem Statement |  |  |  |
| 2 | Initial Requirement Document |  |  |  |
| 3 | Software Requirements Specification Document |  |  |  |
| 4 | Appendix:  a) Use Case Diagram  b) Class Diagram  c) Sequence Diagram  d) Activity Diagram  e) State Chart Diagram  f) Website Screenshots |  |  |  |
| 5 | Test Case Matrix’s |  |  |  |
| 6 | References |  |  |  |

**Problem Statement**

Online Non-interactive website aims to provide an alternative and more flexible way of learning by offering wide range of course that can be accessed and completed with the comfort of time and place.

Limitations:

* Offline classrooms often have limited access to resources such as books, libraries, and educational materials. This can make it difficult for students sometimes to fully understand the topic.
* In an offline classroom, students are required to attend classes at specific times and locations, which can be inconvenient for certain group of students.
* Offline classrooms often involve traditional lectures, which limits student interaction and engagement with the material leading to disinterest from the learning process.
* The best educators of the industry are not accessed by everyone.
* Courses available may not be well structured.

Features:

* Online studying allows students to learn at their own pace. This is especially beneficial as now they can learn from where they like and at what time they like.
* The system includes a well-structured industry level course with proper step-by-step guidance.
* A certificate of completion is given after completing the course that can be valuable in industry.
* Courses includes industry experts as their educators.
* Feedback portal so that users can post all their doubts related to course.

**Initial Requirement Document**

|  |  |
| --- | --- |
| Title of the project | Online course website (Non-interactive) |
| Stakeholders involved in capturing requirements | Teachers, project leaders & students |
| Techniques used for requirement capturing | Interviewing and Brainstorming |
| Name of the persons along with designations | Yash Gupta  Varun Goyal  Yuv Garg |
| Date | January, 2023 |
| Version | 1.0 |
| **Consolidated list of initial requirements:**   1. The website should have a user registration and login system that allows teachers/learners to create an account and upload or access the course. 2. The website should have a catalogue of available courses, search functionality to easily find and enroll in courses. 3. The website should provide a roadmap for the completion of available courses. 4. The website shall be able to generate login ID and password to the admin/instructor/students. 5. Admin shall be able to manage course content and assessments. 6. Instructor shall be able to upload as well as update the courses. 7. The website should support a variety of multimedia content such as videos, readings, quizzes, and assessments to support the learning experience. 8. The website shall be able to provide availability of a particular course. 9. Students should be able to track their progress in each course, including completed lessons and assessments. 10. The courses should be available for single user for a limited period. 11. The website should provide an industry recognised certificate of completion for each course if completed within deadline. 12. The website should have a feedback system for learners to rate the course and provide feedback to the instructor. 13. The website should have a responsive design for viewing on mobile devices. 14. The website should have appropriate security measures to protect user’s data and prevent unauthorized access. 15. The website should be designed to handle a large number of users. 16. The website should have a user-friendly interface that is easy to navigate and understand. | |

**Software Requirements Specification (SRS) document for an Online Course Website:**

**Introduction: -**

* 1. Purpose:

The purpose of this Software Requirements Specification (SRS) is to detail the functional and non-functional requirements for an online course website like Coursera. The website will provide a platform for users to access a range of courses and educational content online, with features for creating, managing, and participating in courses.

* 1. Scope:

The Online course system will cover the following features:

* Issue login ID and password to the system operators.
* Course information management (adding, updating, and deleting courses)
* View availability of desired course
* Certificate of completion
* Proper guide to complete a certain course.
* User management (adding, deleting the details of the user)
* User feedback (e.g., handling customer queries and complaints)
  1. Definitions, Acronyms, and Abbreviations:

SRS: Software Requirement Specification

Operator: Admin, staff, data entry operator

RAM: Random Access Memory

Student: A candidate enrol in a course offered by the system

System Administrator/Administrator: User having all the privileges to operate the system.

Faculty: Teaching staff from various universities

* 1. References:
* Software Engineering by K.K. Aggarwal & Yogesh Singh, New Age Publishing House, 3rd Edition, 2008. Appendix λ 439
* IEEE Recommended Practice for Software Requirements Specifications—IEEE Std. 830-1998.
* IEEE Standard for Software Test Documentation—IEEE Std. 829-1998.
  1. Overview:

An educational course website is an online platform that provides access to a variety of courses and educational content. These websites offer a range of courses, often with different levels of difficulty, and typically cover various subjects and fields of study. They may be designed for both formal and informal learning, catering to different types of learners such as students, professionals, and lifelong learners.

**Overall Description: -**

2.1 Product Perspective:

The online course system will be developed for the skill development of the students by various recognised instructors, provide courses for the same, and improve overall customer experience.

2.2 Product Functions:

The system will provide the following functions to users:

* View courses and their own time table
* Check availability and charge for the course
* Avail one or more courses
* View course details and instructor details
* Different handle for students and instructors
* Certificate on the completion of course (within deadline)

The system will provide the following functions to the administrators:

* Manage course details (e.g., adding, updating, and deleting courses)
* View instructor details and give permission for course modifications
* Manage customer queries and complaints
* Monitor the performance of the system and resolve technical issues

2.3 User Classes and Characteristics:

The system will be used by the following user classes:

* Students: individuals who want to enrol in the courses
* Instructors: individuals who can post their courses on their handle
* Administrators: authorized personnel who can manage course, user information and handle customer queries

2.4 Operating Environment:

The system will run on modern web browsers and will be compatible with popular operating systems such as Windows, macOS and Linux. The system will be optimized for both desktop and mobile devices.

2.5 Design and Implementation Constraints:

The following design and implementation constraints must be considered while developing the online course system:

* The system must be developed using secure coding practices to prevent data breaches and unauthorized access
* The system must be scalable to accommodate a large number of users
* The payment gateway integration must comply with industry-standard security protocols (e.g., PCI DSS)
* The system must be easy to use and accessible to users with disabilities (e.g., support for screen readers)
* The system must be able to handle large volume of transactions
* The system must have a backup and recovery mechanism to ensure data integrity in case of failure

**External Interface Requirements:**

3.1 User Interface:

Since this is a Web based application so it should provide a very user-friendly interface. It should be easy to navigate. A decent and pleasant appearance with ease of navigation that should be of help to the users.

3.1.1 Student Interface

The interface for the student will provide the following features: -

* Personal Information-This enables the student to view and modify the user's personal information.
* Access Course Material- This enables the student to see lectures and download the study materials of the course.
* Post Queries- This will enable the student to send queries to their faculties and view answers to their queries.
* Send Feedback- This will enable the students to provide the feedback to the admin.
* Change Password- This enables the students to change the password for their account. Taking Exam- This enable the students to take the online automated exams on the courses they are enrolled in.

3.1.2 Faculty Interface

The interface for the faculty will provide the following features: -

* Personal Information-This enables the student to view and modify the user's personal

information.

* Uploading Course Material- This enables the faculty to upload the lectures and study materials.
* Answer Queries- This will enable the faculties to view queries of the students and post answers for the same.
* Change Password-This enable the faculties to change the password for their account
* Creating Exams- This enables the faculty to create an exam for the course. He will have to provide the questions and their answers, time, and other details.

3.1.3 Administrator Interface

The interface for the administrators will provide the following features: -

* Course Management- This enables an administrator to add, delete, modify course information, such course name, duration, fee, and other details.
* Student Management- This enables an administrator to add, delete and block the student accounts.
* Faculty Management- This enables an administrator to add, delete and block the faculty accounts.
* View Feedback- This enables an administrator to view the feedback received from the students and faculties.
* Post Notice- This enables an administrator to display notices and other messages on the website.
* Search Information- This enables an administrator to access all the information about all the users of the system.

3.2 Software Interface:

The application should support all major web browsers that will make it convenient for the user to access our system with ease. The back- end i.e., the database services will be used to a great extent and hence it will be quiet efficiently designed.

3.3 Hardware Interface:

The hardware requirement at the user end is simple and the website can also run on hardware that can run a basic simple browser, although the hardware may not work properly (overload) during peak times for the web servers.

**System Features: -**

4.1 Course Information Management:

The system should allow users to access course materials, such as videos, quizzes, readings, and assignments, and progress through the course at their own pace. Users should be able to search for and select courses based on subject, level, duration, pricing, and other criteria.

4.2 Payment Processing:

The website should allow users to have a secure payment gateway. The instructors should be to be able to get their payment for uploading courses while the students should be able to make payments for enrolling in one or more than one courses.

4.3 Assessment and feedback:

The system should provide users with regular assessments to track their progress and help them improve their understanding of the course content. The data collected from their assessments can be shown to the student in the form of graphs and their rankings. They should be able to give feedbacks to the instructor for the betterment of the courses.

4.4 User Management:

The system will store customer information in a database and will allow users to view and update their details. The system will also provide administrators with a platform to manage customer queries and complaints.

4.5 Analytics and Reporting:

The system should provide administrators with analytics and reporting features, such as user engagement and course completion rates, to help them improve the system's effectiveness.

4.6 Customer Support and Management:

The system should provide a platform for customer support and management, allowing administrators to handle customer queries and complaints.

**Non-functional Requirements:**

5.1 Performance Requirements

* The system must respond to user requests within 3 seconds
* The system must be able to handle at least 500 concurrent users
* The system must be able to process at least 100 transactions per second

5.2 Security Requirements:

The system must comply with industry-standard security protocols (e.g., PCI DSS)

The system must use encryption for sensitive data (e.g., passwords, payment information, etc.)

5.3 Usability Requirements:

The system must be user-friendly and intuitive, with clear and concise navigation

The system must be accessible to users with disabilities, following accessibility standards (e.g. WCAG)

5.4 Reliability Requirements:

* The system must have 99.5% uptime
* The system must automatically recover from crashes and errors
* The system must have backup and recovery procedures in place

5.5 Scalability Requirements:

The system must be scalable to accommodate growth in users and data

The system must be easily deployable in different environments (e.g., cloud, on-premises, etc.)

5.6 Technical Requirements:

* The system must be developed using a modern web development framework (e.g. Django, etc.)
* The system must be compatible with popular web browsers (e.g., Chrome, Firefox, etc.). The system must be optimized for both desktop and mobile devices.

Conclusion

The online course website will provide the users with an easy and convenient environment to meet the educational needs of modern learners, and provide them with necessary tools and resources to achieve their learning goals. The system will be secure, reliable, and scalable, with a user-friendly interface and efficient performance.

**Use Case Diagram**

­­­­­

**USE CASE DESCRIPTION:**

**Register Account:**

|  |
| --- |
| **Introduction:** This use case can describe how users can make an account on the system. |
| **Actors:** Student, Teacher, Employee, Admin |
| **Pre-condition:** None |
| **Post-condition:** If the use case is successful the users account is successfully made and he can log into the system. |
| **Event Flow:**  **Basic Flow:**   1. This use case starts when an unregistered user wishes to make his account on the website. 2. Once he enters his basic information, and sets his email and password, his/her data is authorized by third party authorization service. 3. After authorization the data of the user is saved into the database. |
| **Alternate Flow 1:** Username and email already exists.  While setting up a new profile if we enter a username that already exists, the system will display an error message.  **Alternate Flow 2:** Password doesn’t match the specific requirements.  If the password doesn’t fulfil the minimum requirements the user will have to change the user password.  **Alternate Flow 3:** Credentials left unentered  If the user details are left unentered, the system will display an error message. |
| **Special Requirements:** None |
| **Associated Use case:** None |

**Enroll in a course:**

|  |
| --- |
| **Introduction:** This use case can describe how students can enroll in a course on the system. |
| **Actors:** Students |
| **Pre-condition:** Students have already registered and logged into the account |
| **Post-condition:** If the use case is successful then the registered students have to pay for the course, only if the payment is successful the system can access the contents of the course. |
| **Event Flow:**  **Basic Flow:**   1. This use case starts when a student wishes to enroll in a course on the website 2. He/She selects the course in which the student wants to enroll |
| **Alternate Flow 1:** Already enrolled in the course  If the student is already enrolled in the course, the user can’t enroll for the course again.  **Alternate Flow 2:** The course has expired  If the course has expired, the user can’t enroll for the course. |
| **Special Requirements:** None |
| **Associated use case:** Login |

**Login:**

|  |
| --- |
| **Introduction:** This Use Case describes how students and teachers can login into the system  **Actors:** Students, Teachers, Staff, Admin  **Precondition:** Users have already registered their accounts  **Postcondition:** If the use case is successful then the users log into the system  **Event Flow:**  **Basic Flow:**   1. This use case starts when a user wishes to log into the system. 2. He/ She enters the email and password. If they match with the credentials present in the database then he/she successfully log into the system.   **Alternate Flow 1:** Username/E-mail is incorrect  While login if we enter a username which is incorrect, the system will display an error message.  **Alternate Flow 2:** Password is incorrect  While login if we enter a password that doesn’t matches with the actual password, the user has to enter the password again, the user also has the option of forgot password.  **Special Requirement:** None  **Associated use case:** Login |

**Feedback:**

|  |
| --- |
| **Introduction:** This use case describes how student can give feedback. |
| **Actors:**  Registered Student  Admin |
| **Precondition:** The student is already registered |
| **Postcondition:** If the use case is successful, the student will successfully be able to give feedback to the admin. |
| **Event Flow:**  **Basic Flow:**   1. This use case starts when the student wishes to provide feedback. 2. After filling necessary details, the information is sent to the concerned authority by the admin. |
| **Alternative Flow 1: Student is unregistered**  If the student is unregistered, the system shall generate an error.  **Alternative Flow 2: Student does not have a WhatsApp account**  If student does not have a WhatsApp account, the system shall generate an error. |
| **Special Requirement:**  None |
| **Associated Use case:**  Login |

**View Progress**

|  |
| --- |
| **Introduction:** This use case describes how the user can view progress of a student. |
| **Actors:**  Registered Student  Registered Teacher  Admin |
| **Precondition:** The student is already registered |
| **Postcondition:** If the use case is successful, the user will be able to view the progress of a student. |
| **Event Flow**  **Basic Flow:**   1. This use case starts when the student wishes to view his/her course progress. 2. A request is made to the admin to generate progress of the student. |
| **Alternative Flow 1: Student is unregistered**  Student is then redirected to login use case. |
| **Special Requirement:**  None |
| **Associated Use case:**  Login |

**Download Certificate**

|  |
| --- |
| **Introduction:** This use case describes how student can download certificate. |
| **Actors:**  Student |
| **Precondition:** The student is already registered and logged into their account. |
| **Postcondition:** If the use case is successful, the student will successfully be able to download the certificate. |
| **Event Flow**  **Basic Flow:**   1. This use case starts when the user wishes to download the course certificate. 2. Once ‘Download Certificate’ button is clicked, a request is made to the admin to generate the course certificate for the student. 3. If course progress is more than 90%, the user shall be able to download the certificate of completion. |
| **Alternative Flow 1:** **Student is not registered**  If the student is unregistered, the system shall generate an error.  **Alternative Flow 2: Course progress is less than 90%**  If course progress is less than 90%, the system shall generate an error and redirect the student to complete the course first. |
| **Special Requirement:**  None |
| **Associated Use case:**  Login |

**Update user Details**

|  |
| --- |
| **Introduction:** This use case describes how the registered students/educators can update their details |
| **Actors:** Students, Educators |
| **Precondition:** Users have already registered and logged into their accounts |
| **Postcondition:** If the use case is successful, then the registered users can successfully update their details. |
| **Event Flow:**  **Basic Flow:**   1. This use case starts when a user wishes to update their details 2. After entering the details, a request is made to the admin to update the details of the user. |
| **Alternative Flow 1:** **User is not registered**  If user is not registered, the system shall generate an error. |
| **Special Requirement:**  None |
| **Associated Use case:** Login |

**Payment Use case:**

|  |
| --- |
| **Introduction:** This use case describes how the registered students/educators can pay for the course they want to enrol in. |
| **Actors:** Students |
| **Precondition:** Users have already registered and logged in their accounts |
| **Postcondition:** If the use case is successful, then the users can have full access to the course details like slides, video lectures, quizzes and many more. |
| **Event Flow**  **Basic Flow:**   1. This use case starts when a user wishes to pay for the course posted on the site in which they want to enrol. 2. After selecting the course in which students wants to enrol they pay for the course which is managed by a third party payment service provider. |
| **Alternative Flow 1:** **Payment not successful**  If payment is failed user can’t enrol in the course.  **Alternative Flow 2: Course Expired**  If the course is expired, the user can’t pay for the course.  **Alternative Flow 3: Already enrolled in the course**  If the user already enrolled in the course, the user can’t pay for the course again. |
| **Special Requirement:**  None |
| **Associated Use case:**  Login |

**Add Courses:**

|  |
| --- |
| **Introduction:** This use case describes how the registered educators can add new courses. |
| **Actors:** Educators |
| **Precondition:** Users have already registered and logged in their accounts |
| **Postcondition:** If the use case is successful, then the educators can successfully add a course. |
| **Event Flow:**  **Basic Flow:**   1. This use case starts when a educator wishes to add a course. 2. After entering the details, a request is made to the admin to add the details of the new course in the website. |
| **Alternative Flow 1:** **Course already exists**  In this case the course will not be added. |
| **Special Requirement:**  None |
| **Associated Use case:**  Login |

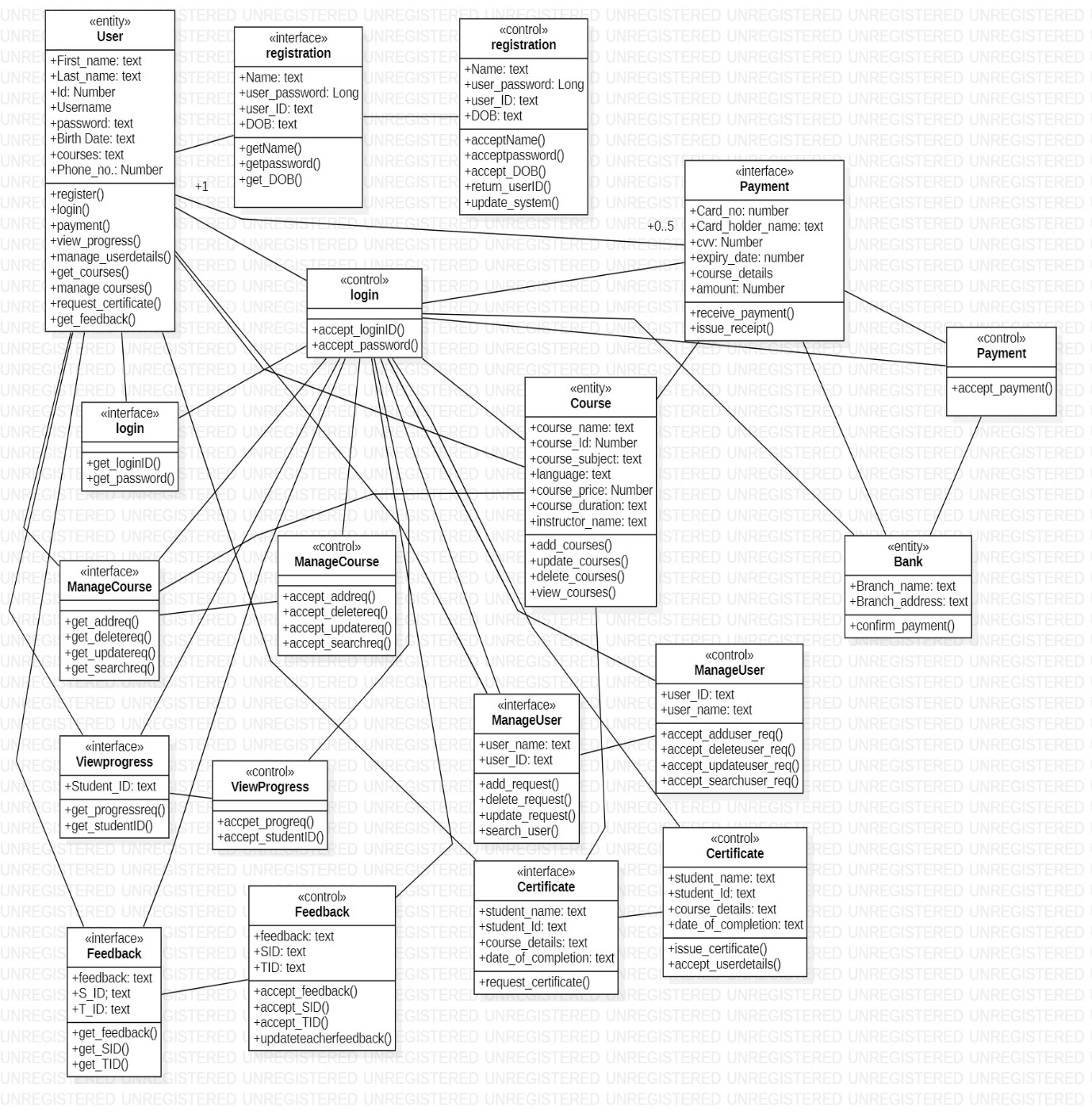
**Manage Course:**

|  |
| --- |
| **Introduction:** The use case documents the steps that the user follows in order to maintain book details and add, update, delete and view book information. |
| **Actors:** Admin, Educators |
| **Precondition:** Users have already registered and logged in their accounts |
| **Postcondition:** If the use case is successful, then the course information is added, updated, deleted, or viewed. |
| **Event Flow**  **Basic Flow:**  This use case starts when a user wishes add/update/delete/view course details.   1. The system requests that the user to specify the function he/she would like to perform (either add a course, update a course, delete a course or view a course). 2. Once the user provides the requested information, one of the sub flows is executed.   • If the user selects “Add a course”, the **Add a course** sub flow is executed.  • If the user selects “Update a course”, the **Update a course** sub flow is executed.  • If the user selects “Delete a course”, the **Delete a course** sub flow is executed.  • If the user selects “View a course”, the **View a course** sub flow is executed.  **Basic Flow 1:** **Add a Course**  The system requests that the user to enter the book information. This includes:   * Course Name * Course ID * Duration * Language * Educator   Once the user provides the requested information, the book is added to the system.  **Basic Flow 2: Update a Course**  1. The system requests that the user to enter the course ID.  2. The user enters the Course ID.  3. The system retrieves and displays the course information.  4. The admin makes the desired changes to the course information. This includes any of the information specified in the Add a course sub flow.  5. Once the admin updates the necessary information, the system updates the course information with the updated information.  **Basic Flow 3: Delete a Course**   1. The system requests the user to enter the Course ID. 2. The admin enters the Course ID. The system retrieves and displays the required information. 3. The system asks admin to confirm the deletion of the course details. 4. The admin verifies the deletion. 5. The system deletes the record.   **Basic Flow 4: View a Book**  1. The system requests the user to enter course ID.  2. The system retrieves and displays the book information. |
| **Alternative Flows**  **Alternative Flow 1: Invalid Entry**  If in the **Add a Course** or **Update a Course** flow, the actor enters invalid course details or leave any of the details empty, the system displays an appropriate error message. The actor returns to the basic flow and may re-enter the invalid entry.  **Alternative Flow 2: Course Already Exists**  If in the **Add a** **Course** flow, a **Course** with a specified ID already exists, the system displays an error message. The admin returns to the basic flow and may re-enter the Course.  **Alternative Flow 3: Course** **Not Found**  If in the **Update a Course** or **Delete a** **Course** or **View a Course** flow, the Course information with the specified ID does not exist, the system displays an error message. The admin returns to the basic flow and may re-enter the Course ID.  **Alternative Flow 4: Update Cancelled**  If in the **Update a Course** flow, the admin decides not to update the Course, the update is cancelled and the Basic Flow is restarted at the beginning.  **Alternative Flow 5: Delete Cancelled**  If in the **Delete a** **Course** flow, the admin decides not to delete the Course, the delete is cancelled and the Basic Flow is restarted at the beginning.  **Alternative Flow 6: User Exits**  This allows the user to exit at any time during the use case. The use case ends |
| **Special Requirement:**  None |
| **Associated Use case:**  Login |

**Manage User details:**

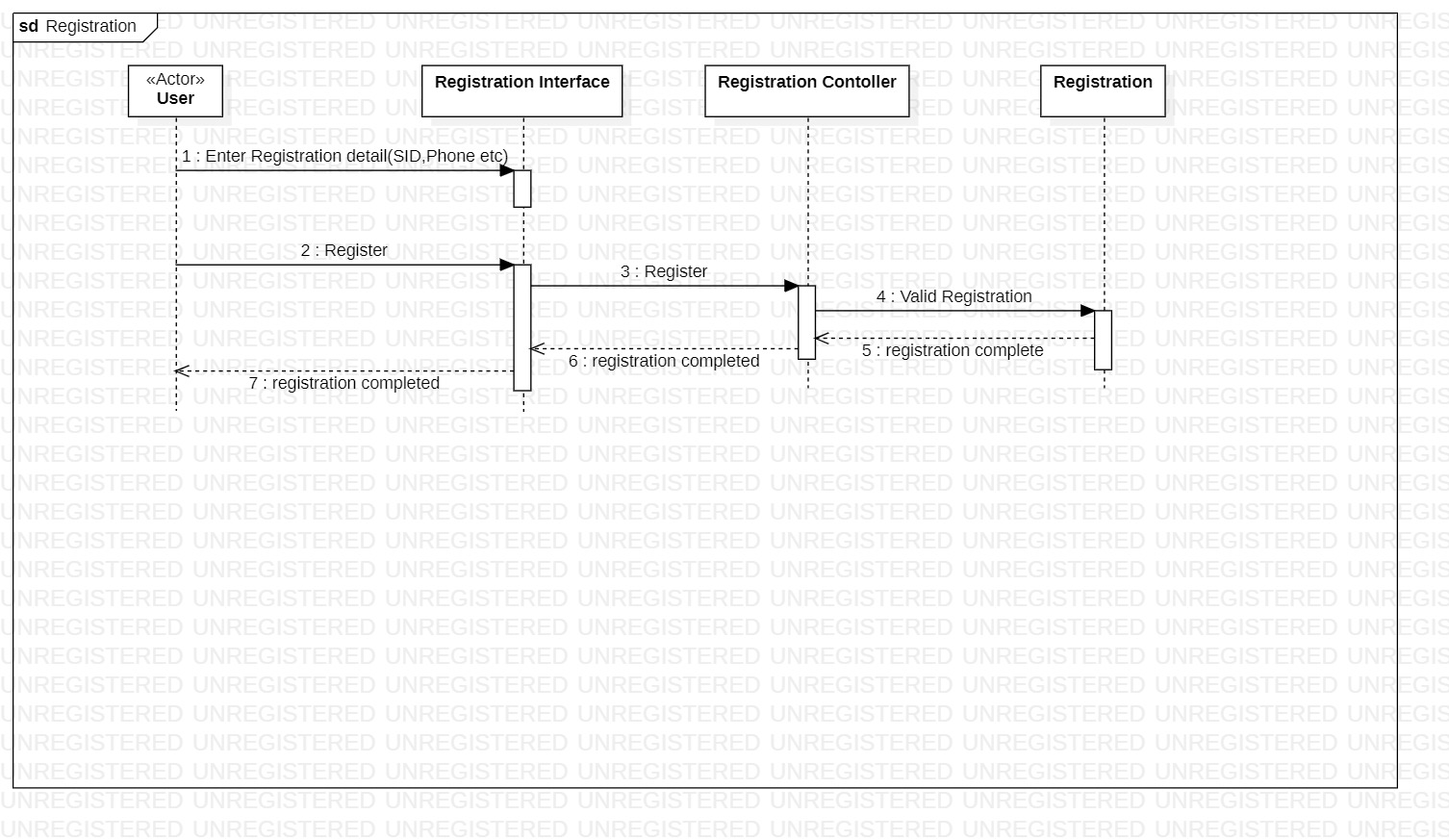
|  |
| --- |
| **Introduction:** This use case documents the steps that the user must follow in order to maintain members details. This includes adding, updating, deleting and viewing member information |
| **Actors:** Admin, Staff |
| **Precondition:** Users have already registered and logged in their accounts |
| **Postcondition:** If the use case is successful, then the member information is added, updated, deleted, or viewed. |
| **Event Flow**  **Basic Flow:**  This use case starts when a user wishes add/update/delete/view course details.   1. The system requests that the user to specify the function he/she would like to perform (either add, update, delete or view a member). 2. Once the user provides the requested information, one of the sub flows is executed.   • If the user selects “Add a member”, the **Add a member** sub flow is executed.  • If the user selects “Update a member”, the **Update a member** sub flow is executed.  • If the user selects “Delete a member”, the **Delete a member** sub flow is executed.  • If the user selects “View a member”, the **View a member** sub flow is executed.  **Basic Flow 1:** **Add a Member**  The system requests that the user to enter the member information. This includes:   * Member Name * Member ID * Phone No. * Date of Birth * Address * Password   Once the user provides the requested information, the member is added to the system.  **Basic Flow 2: Update a Member**  1. The system requests that the staff enter the student’s membership number.  2. The admin enters the member’s id.  3. The system retrieves and displays the member’s information.  4. The admin makes the desired changes to the member information. This includes any of the information specified in the Add a member sub flow.  5. Once the admin updates the necessary information, the system updates the member information with the updated information.  **Basic Flow 3: Delete a Member**   1. The system requests the user to enter the Member ID. 2. The admin enters the Member ID. The system retrieves and displays the required information. 3. The system asks admin to confirm the deletion of the member’s details. 4. The admin verifies the deletion. 5. The system deletes the record.   **Basic Flow 4: View a Member**  1. The system requests the user to enter Member ID.  2. The system retrieves and displays the Member information. |
| **Alternative Flows**  **Alternative Flow 1: Invalid Entry**  If in the **Add a Member** or **Update a Member** flow, the actor enters invalid member details or leave any of the details empty, the system displays an error message. The actor returns to the basic flow and may re-enter the invalid entry.  **Alternative Flow 2: Member Already Exists**  If in the **Add a** **Member** flow, a **Member** with a specified ID already exists, the system displays an error message. The staff returns to the basic flow and may re-enter the Member.  **Alternative Flow 3: Member Not Found**  If in the **Update a Member** or **Delete a** **Member** or **View a Member** flow, the Member information with the specified ID does not exist, the system displays an error message. The admin returns to the basic flow and may re-enter the Member ID.  **Alternative Flow 4: Update Cancelled**  If in the **Update a Member** flow, the staff decides not to update the Member, the update is cancelled and the Basic Flow is restarted at the beginning.  **Alternative Flow 5: Delete Cancelled**  If in the **Delete a** **Member** flow, the staff decides not to delete the Member, the delete is cancelled and the Basic Flow is restarted at the beginning.  **Alternative Flow 6: User Exits**  This allows the user to exit at any time during the use case. The use case ends |
| **Special Requirement:** None |
| **Associated Use case:** Login |

**Class Diagram**

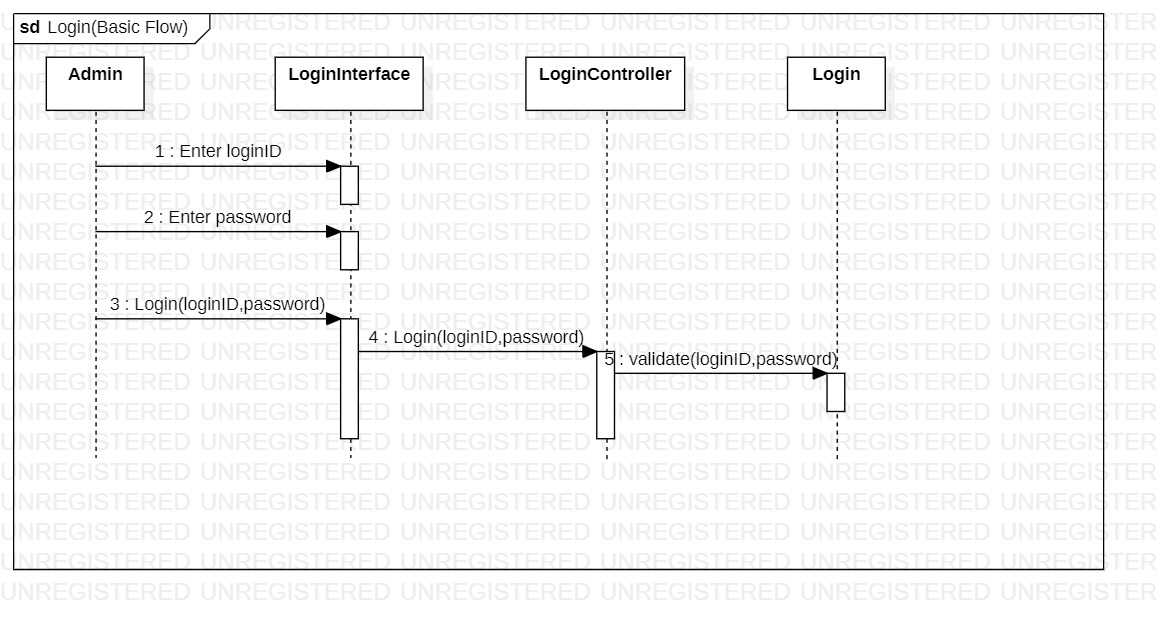


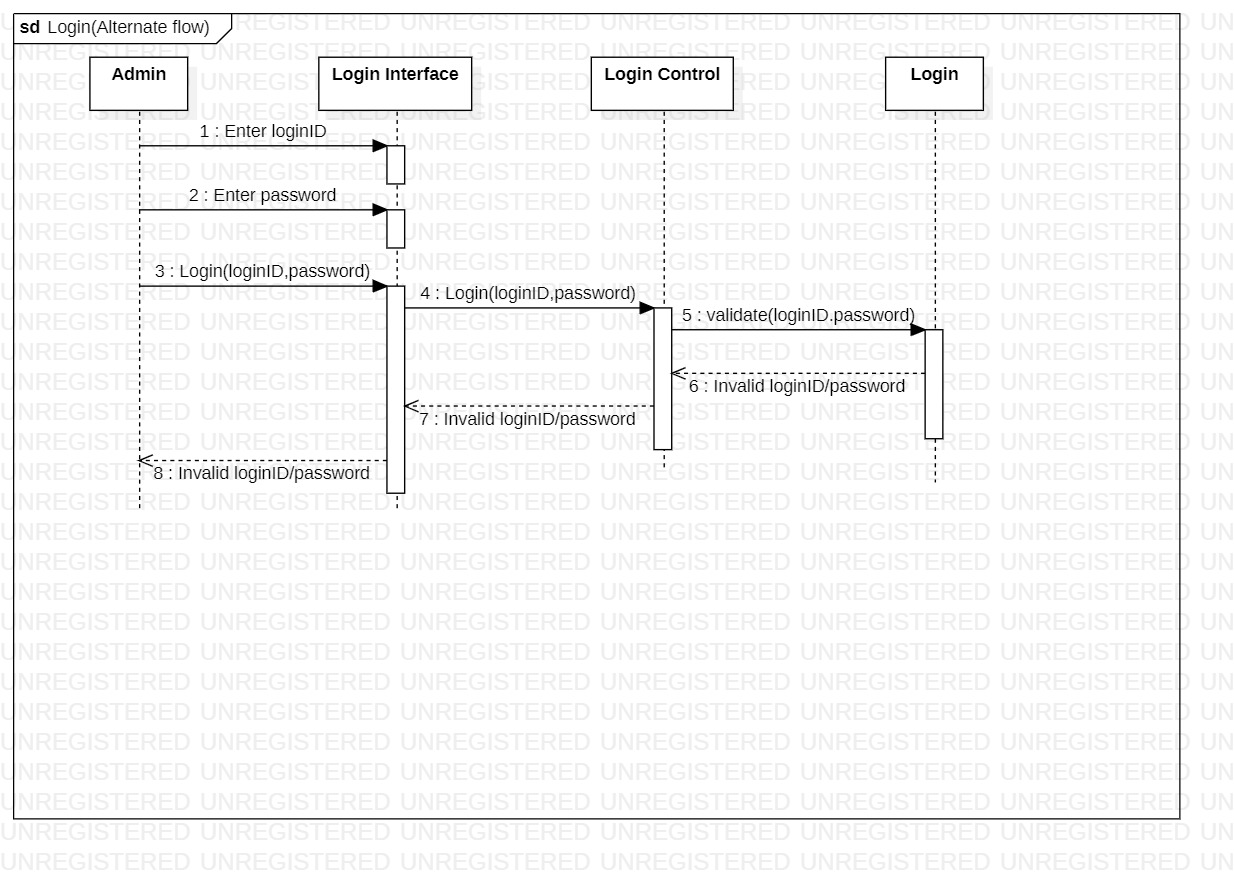
**Sequence Diagrams**

1.Registration

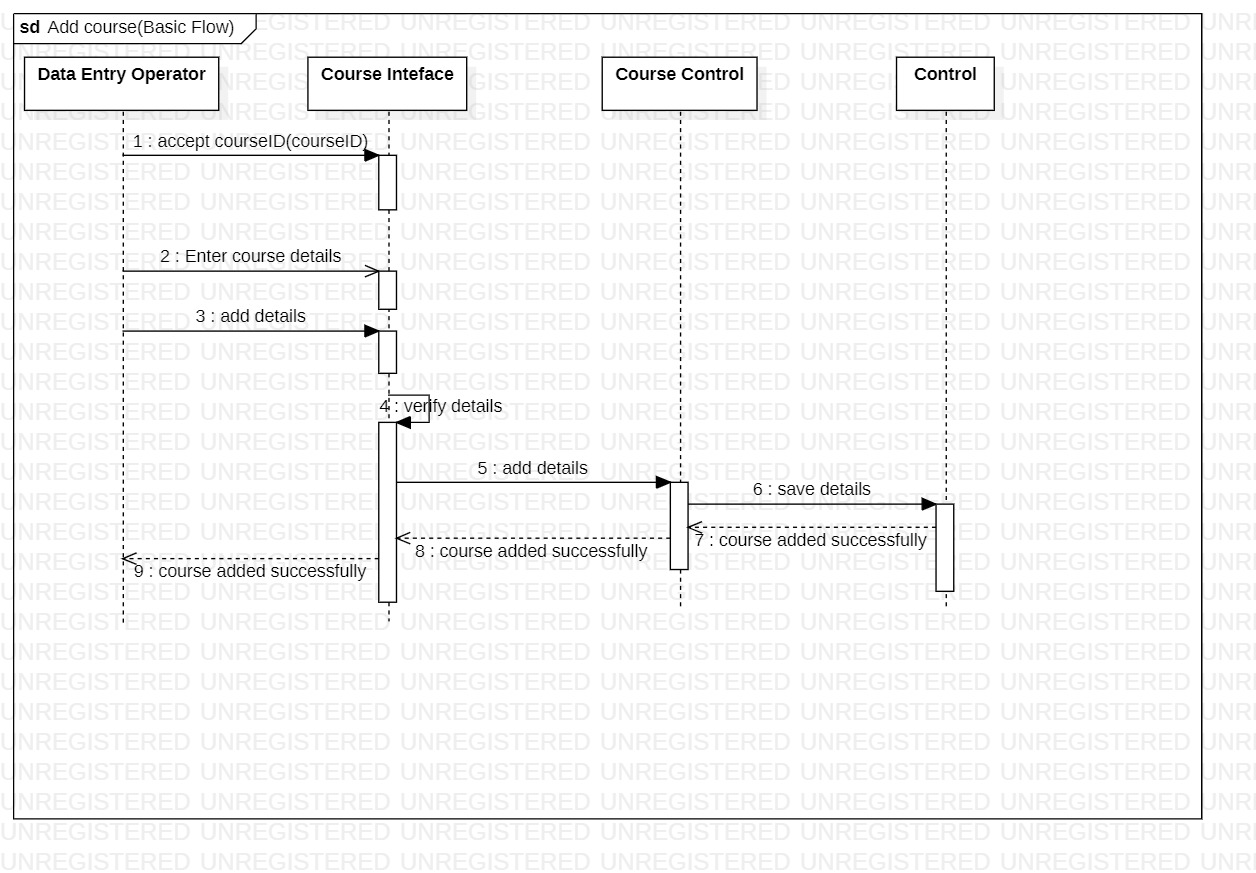


2. Login

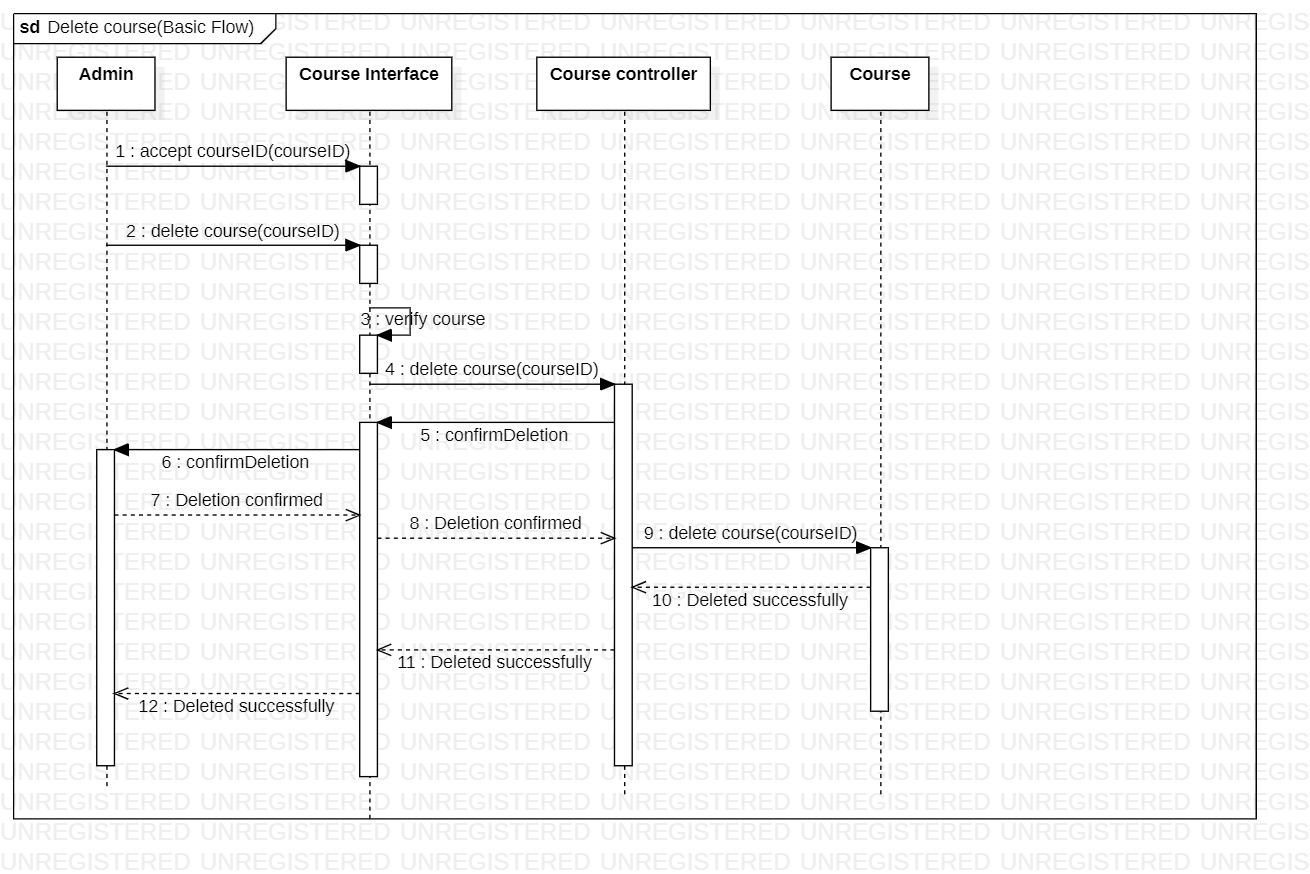


3. Login Alternate Flow

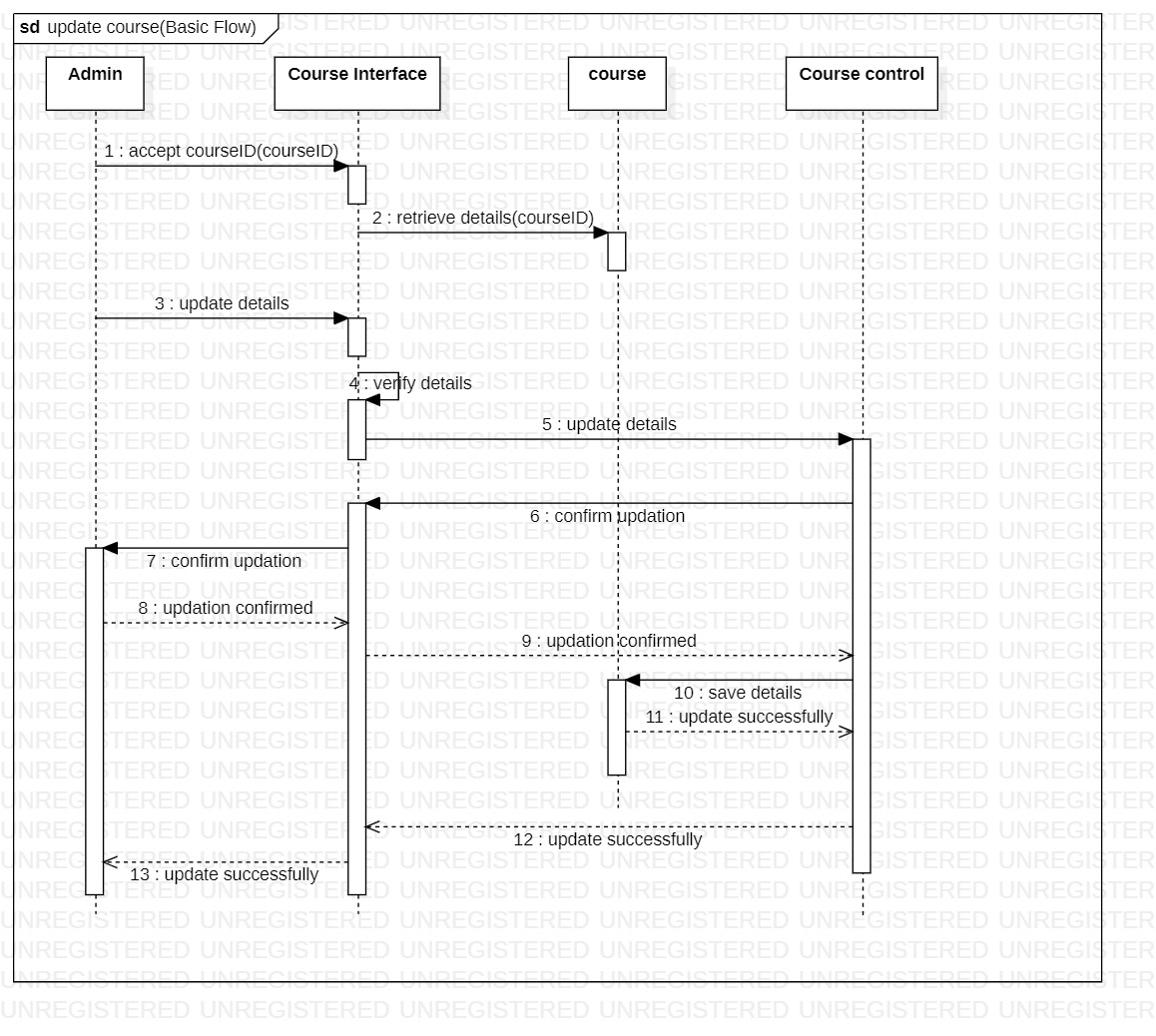
4.Add Course



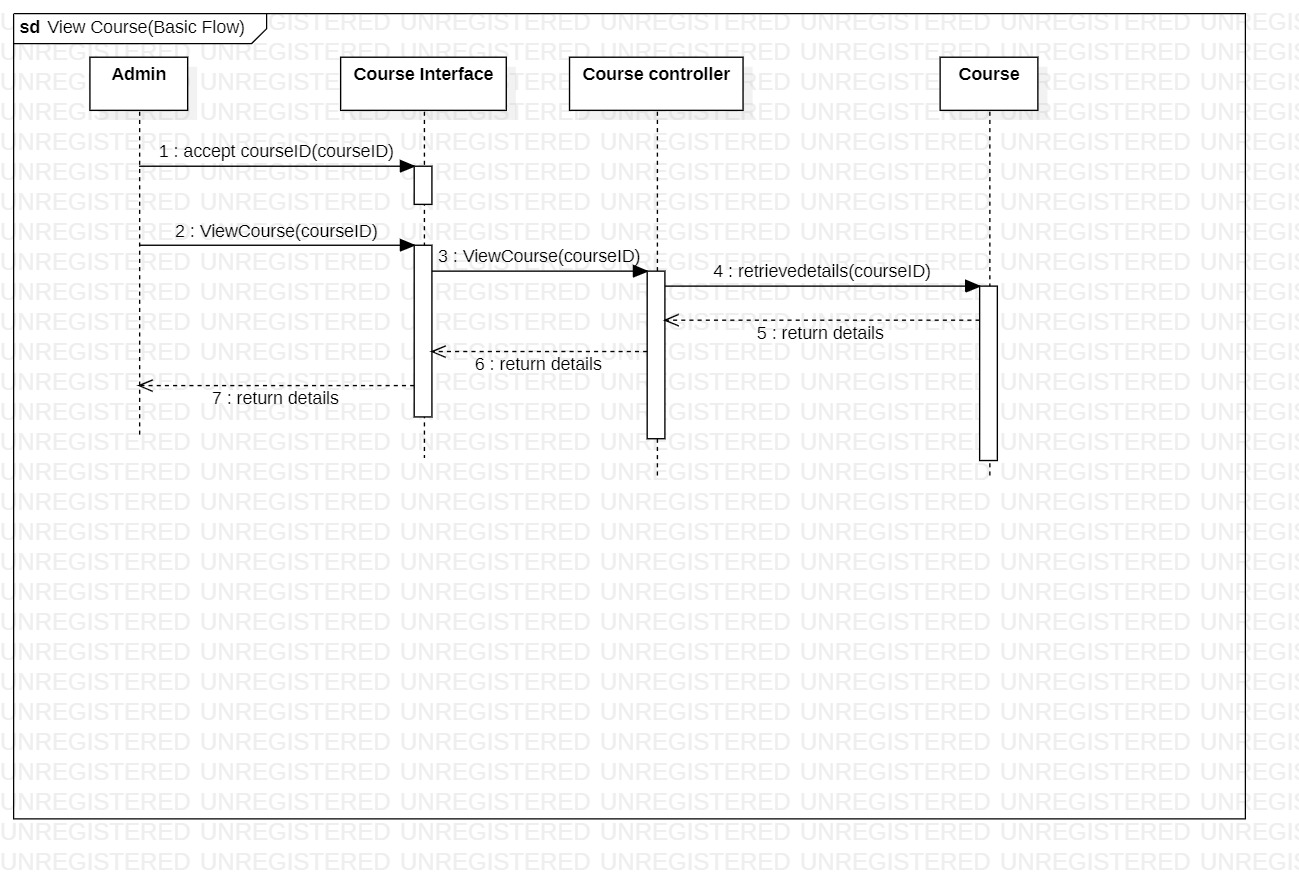
5.Delete Course



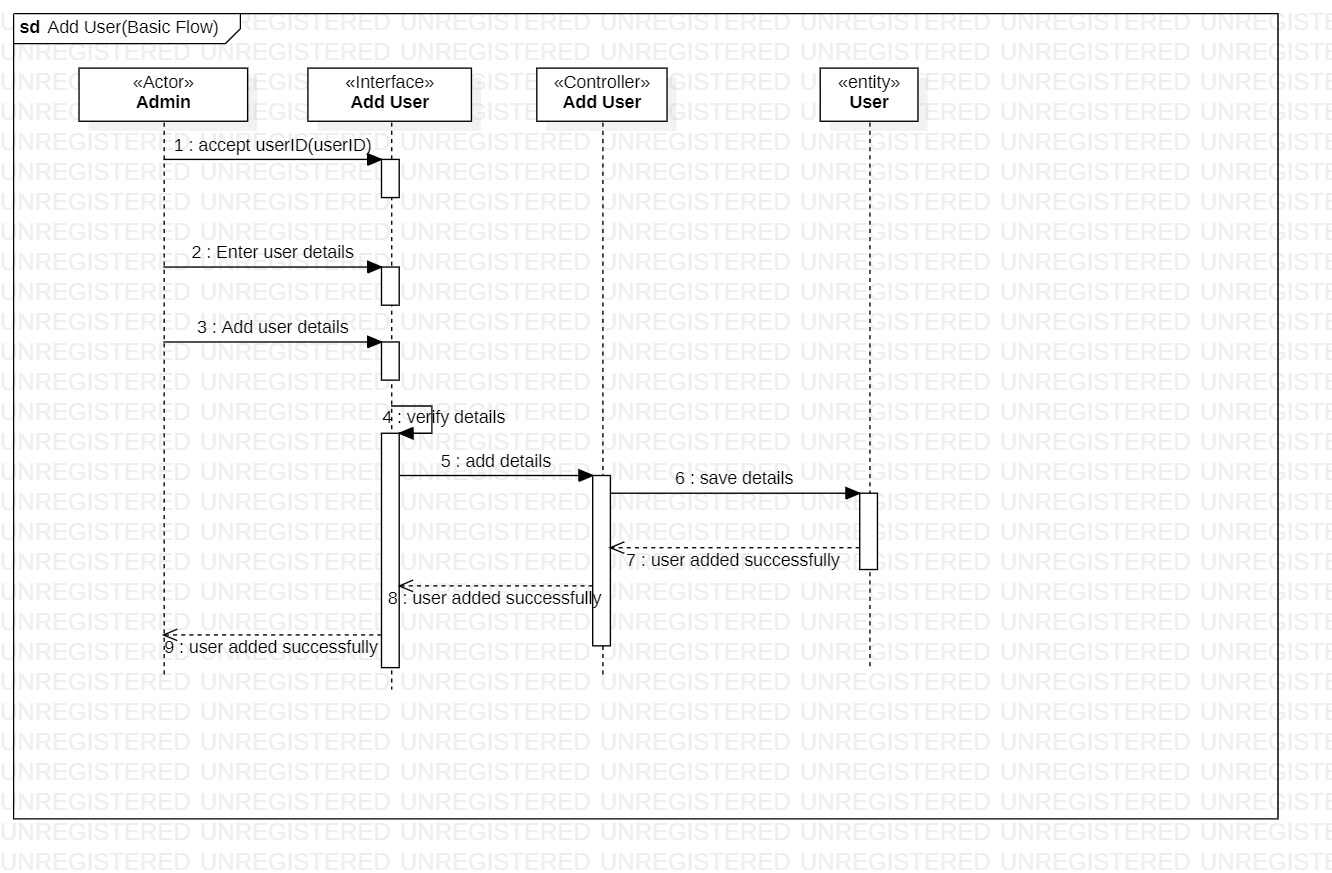
6.Update Course



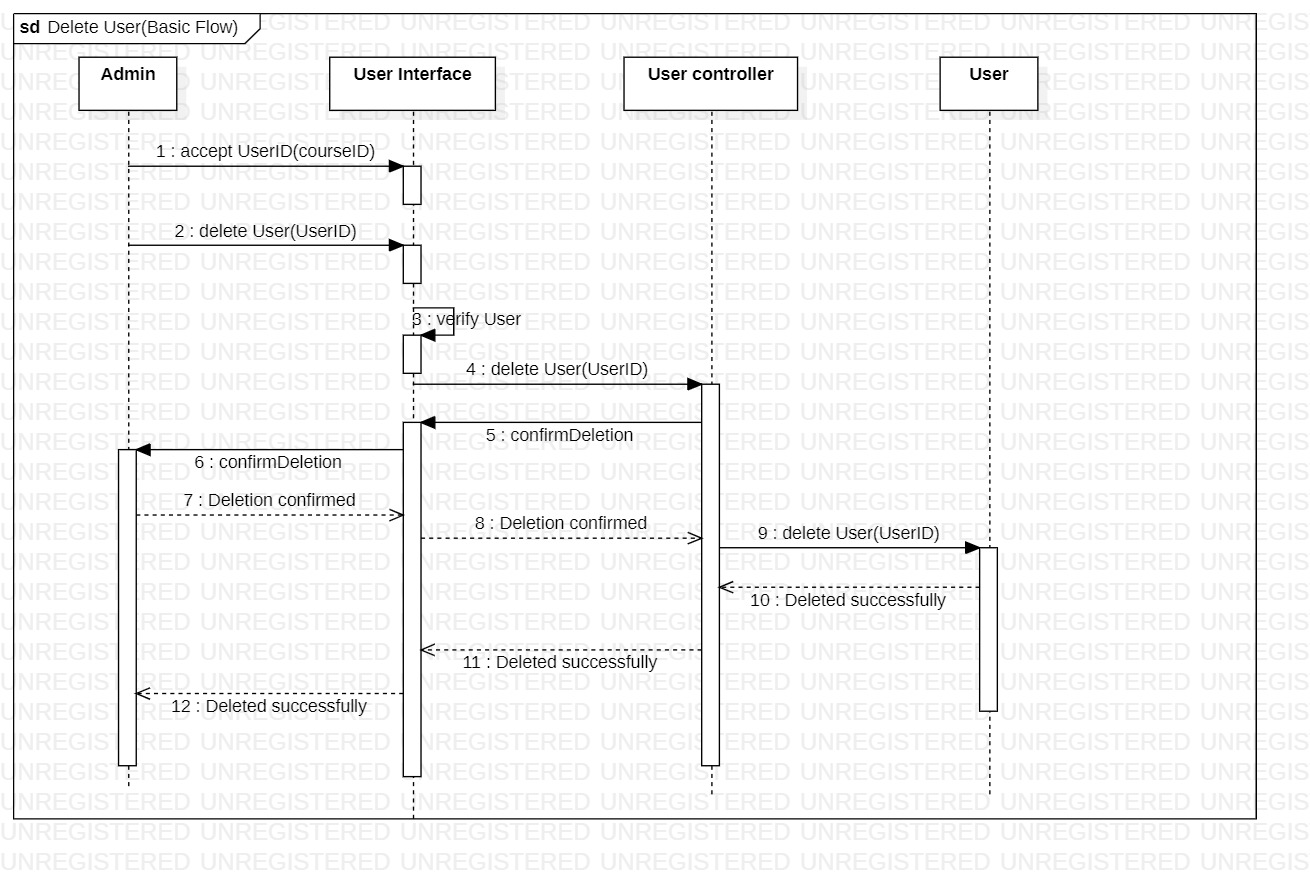
7.View Course



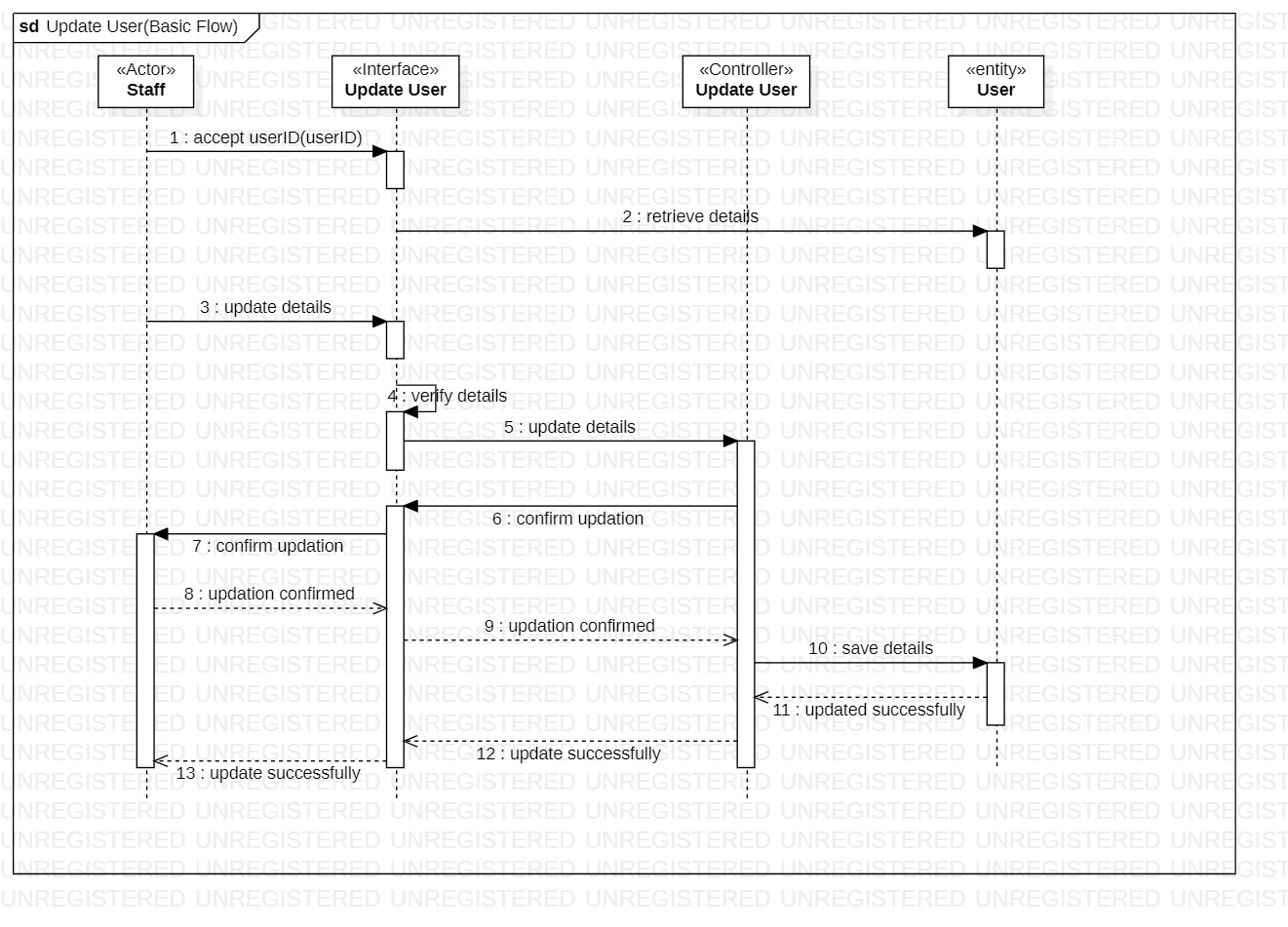
8.Add User



9.Delete User



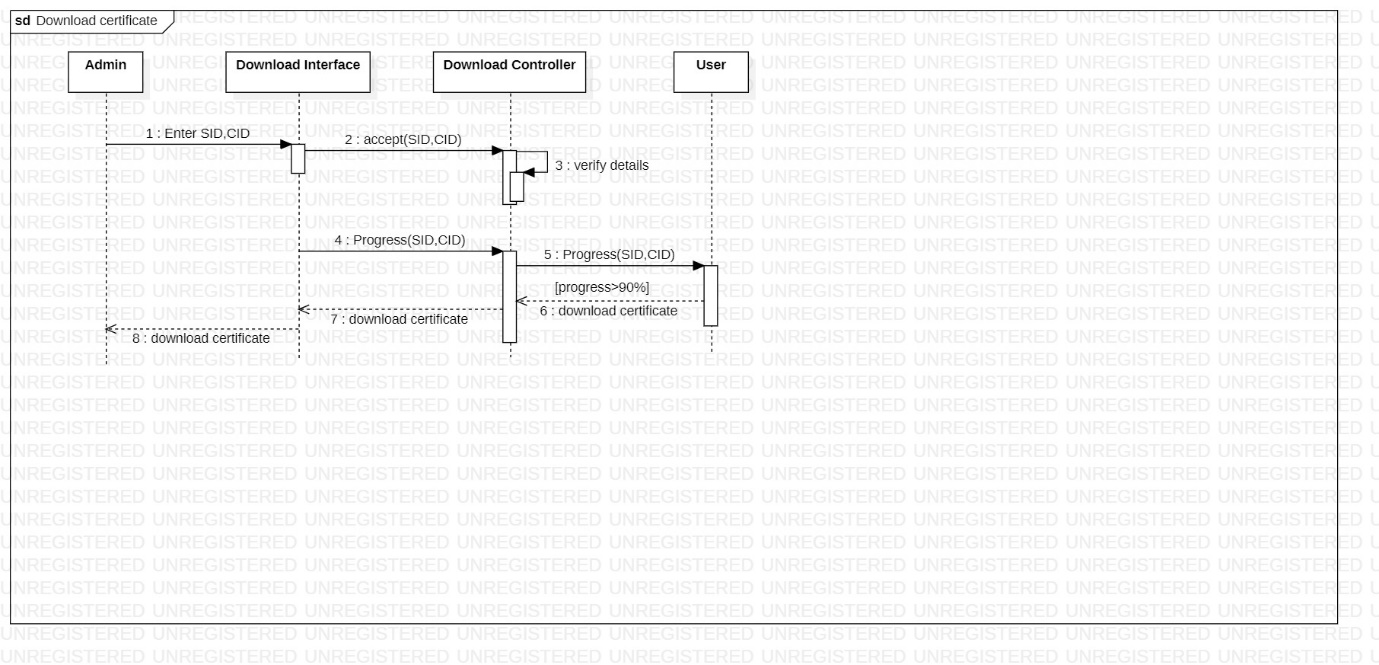
10.Update User



11.View Progress

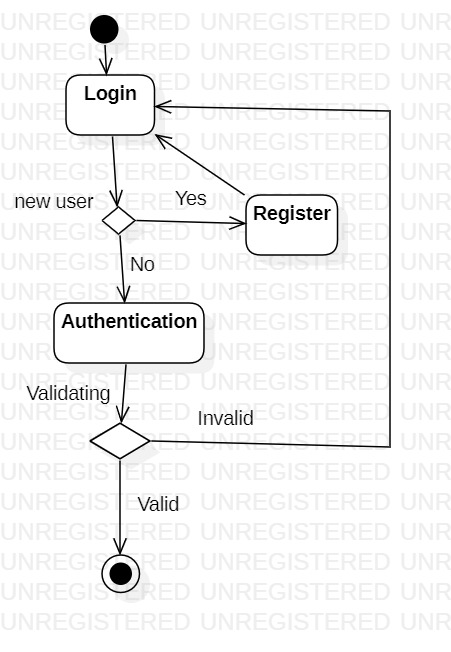


12.Download Certificate

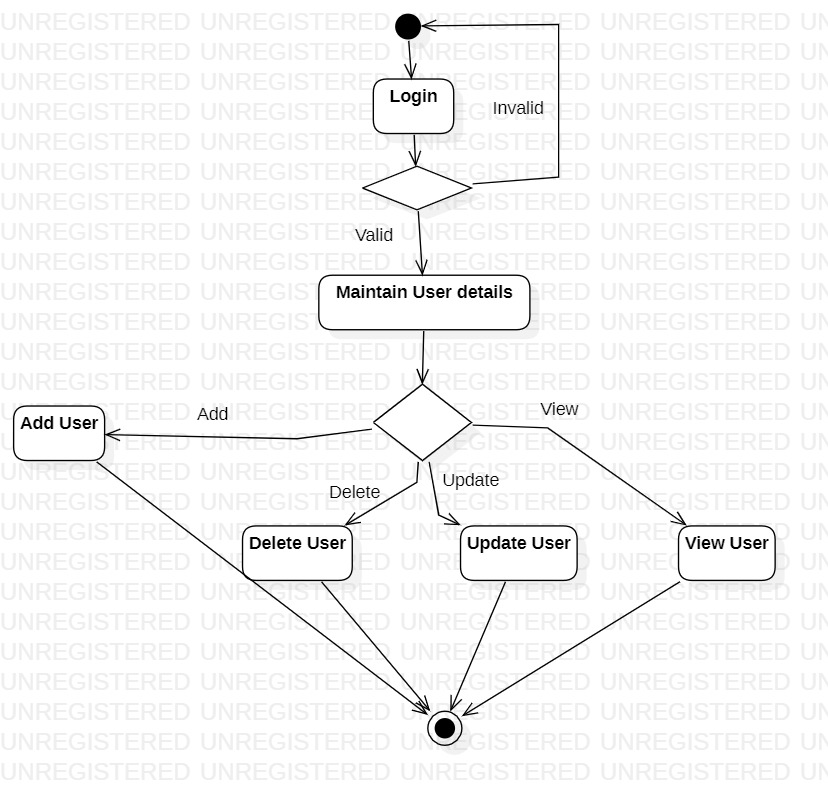


**Activity Diagrams**

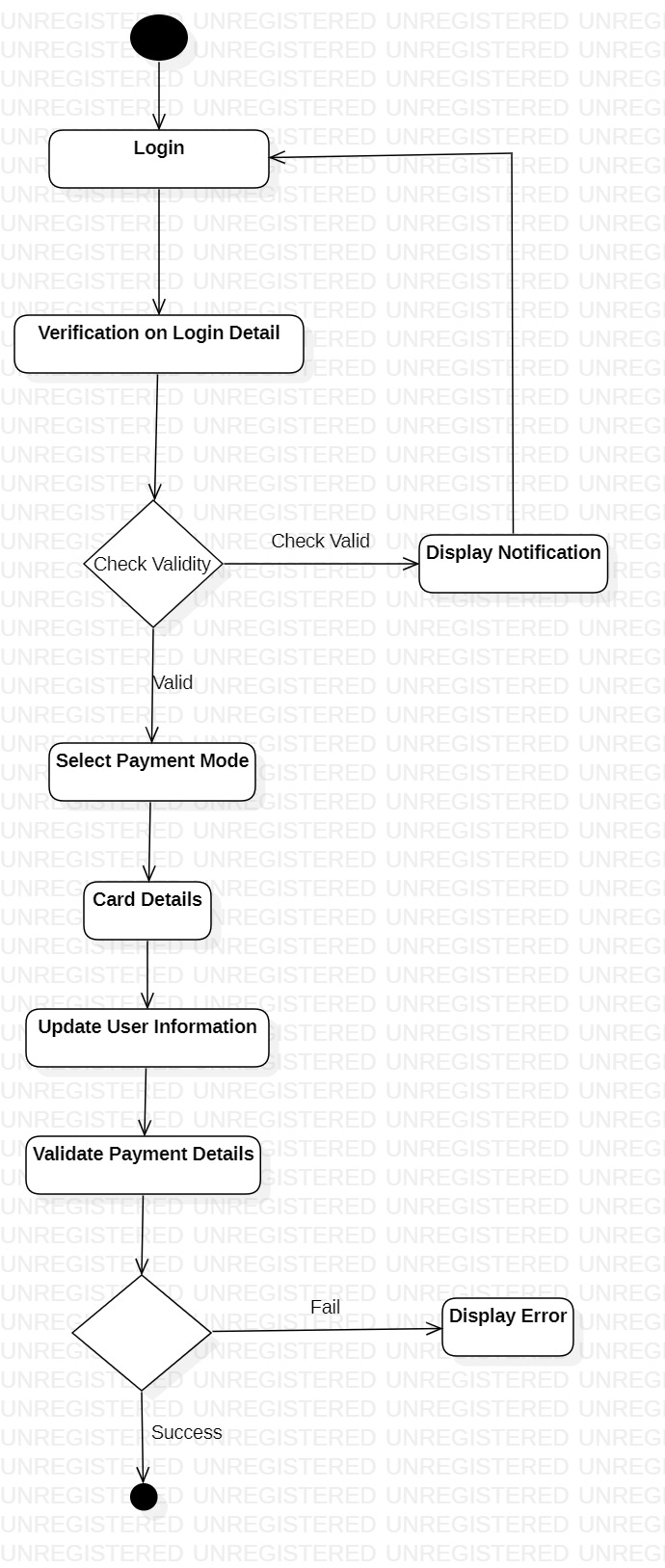
* + - 1. Login



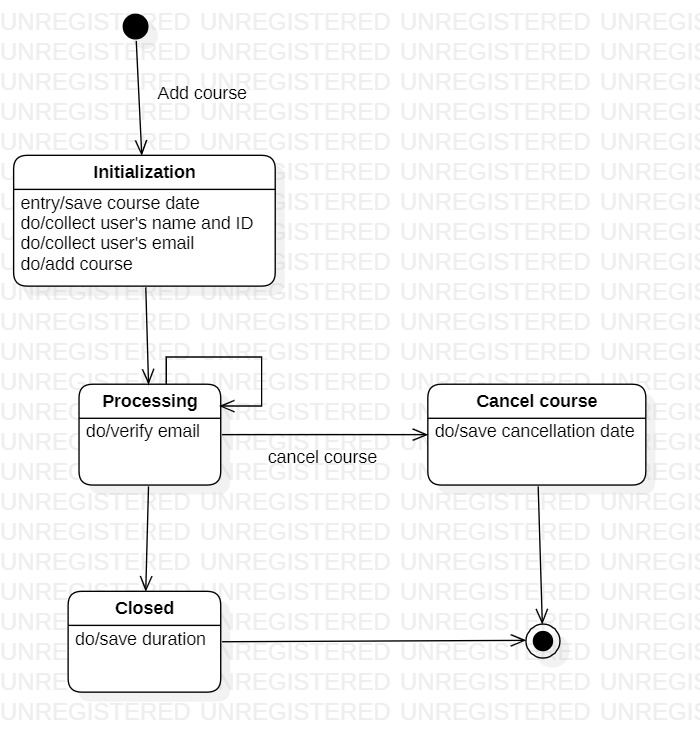
2. Maintain user details

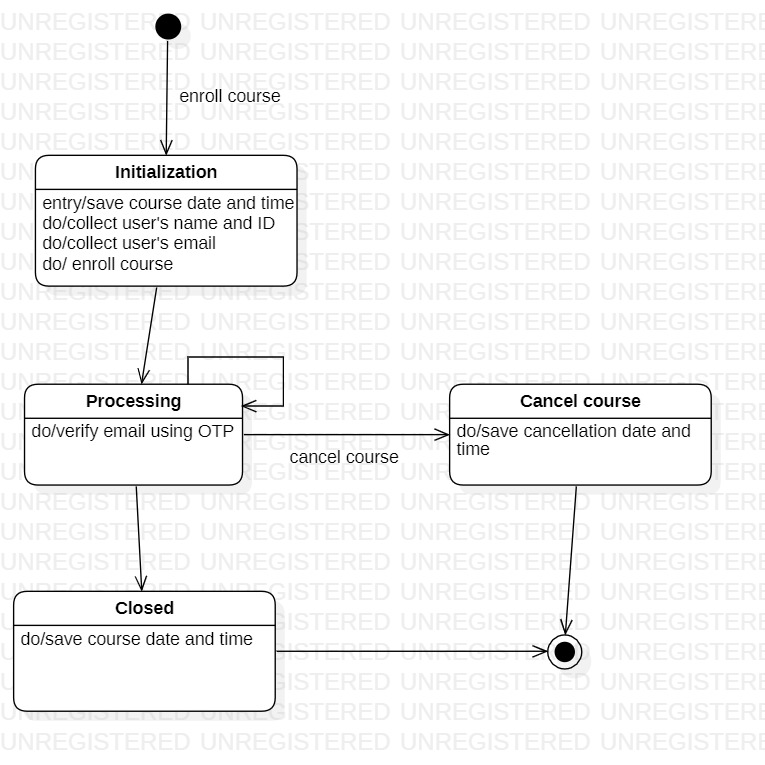


3. Payment

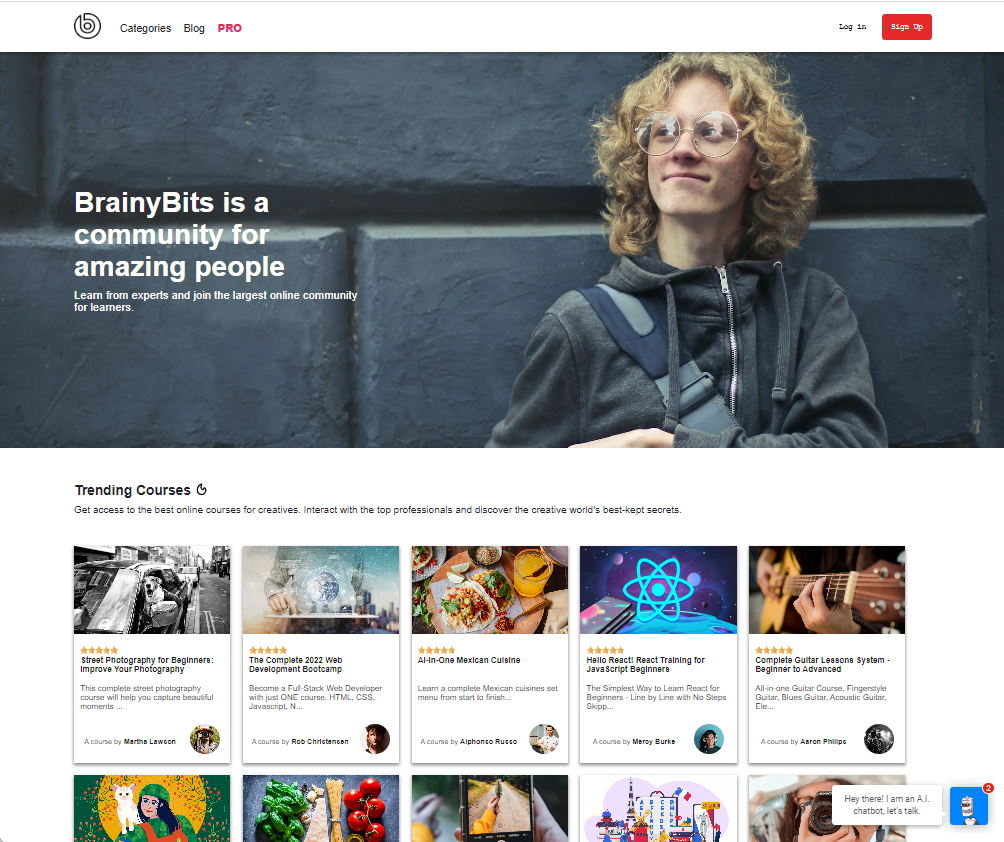


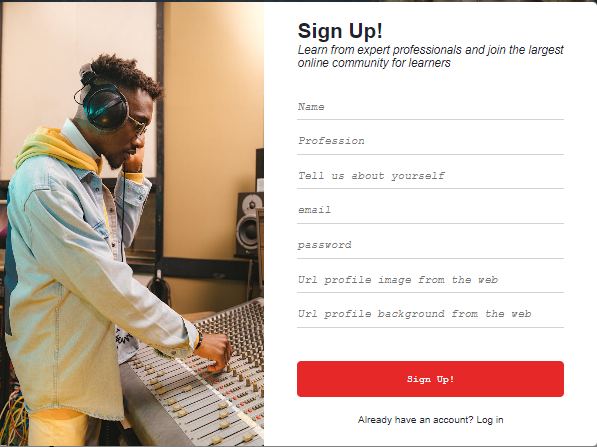
**State Chart Diagrams**

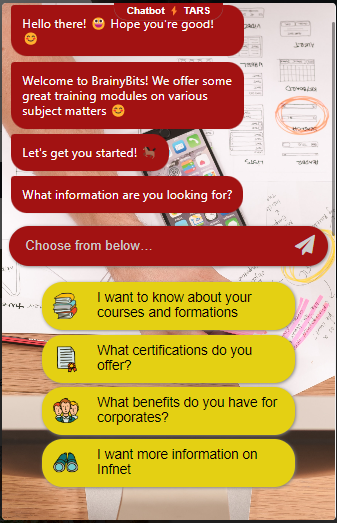
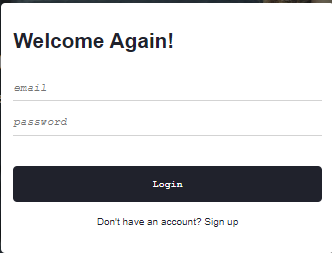
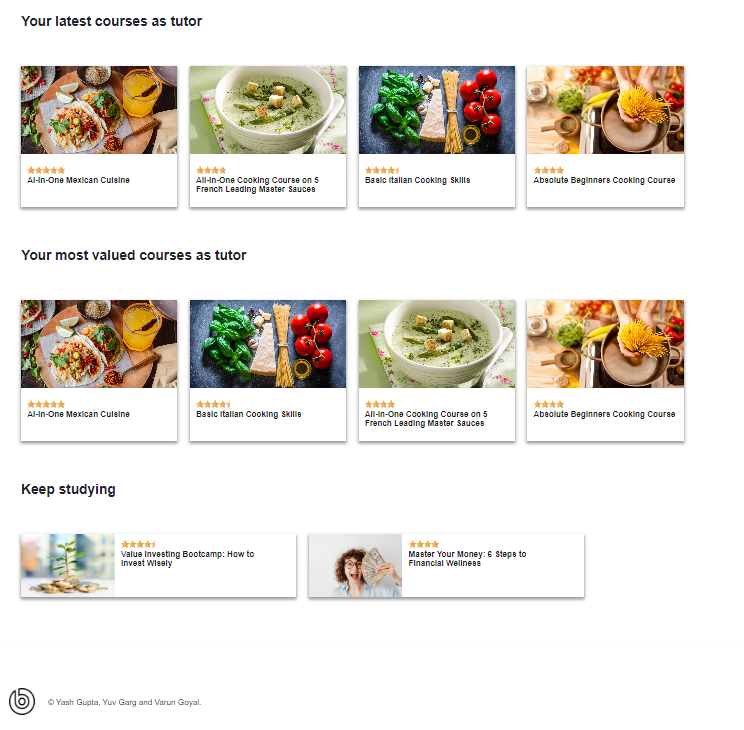


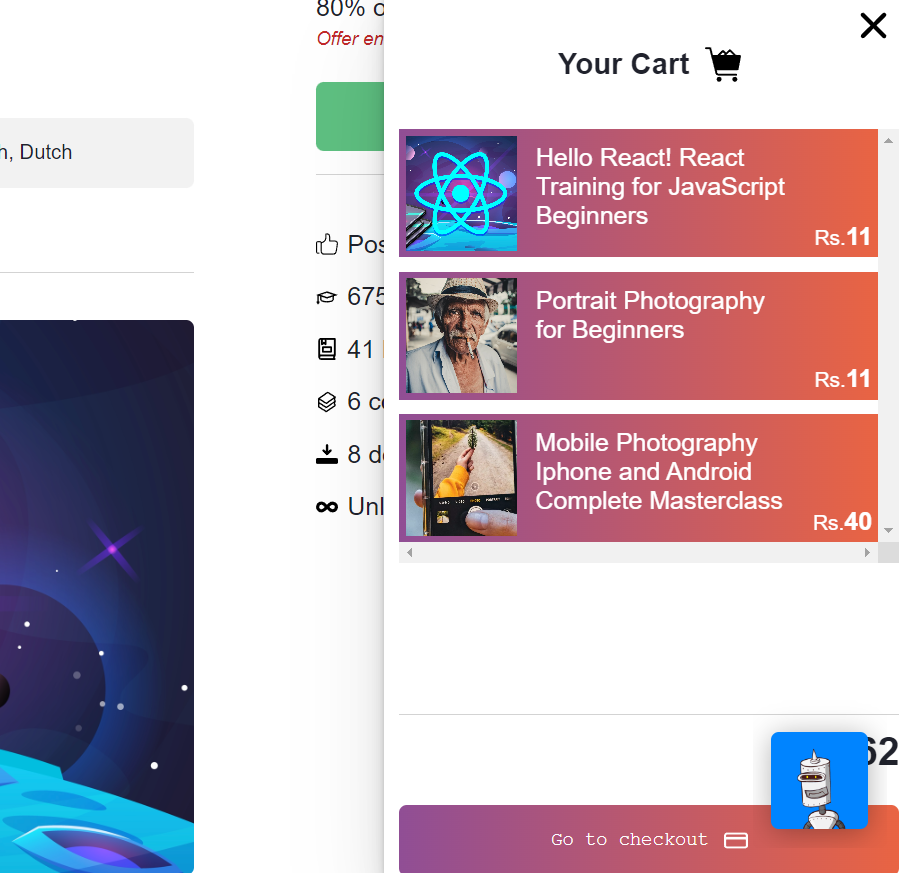


**Website Screenshots**







Test Case Matrix for Online Course Website

1. Test Case Matrix for Login.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case ID | Scenario Name  and Description | Input 1 User Id | Input 2 Password | Expected Output | Remarks |
| TC1 | Scenario 1 – Login | Valid | Valid | User is allowed to login |  |
| TC2 | Scenario 2 – Login Alternate flow:  Invalid Entry | Invalid | Valid/Invalid | User ID is invalid | User ID is incorrect |
| TC3 |  | Invalid | Valid | User ID is invalid | User ID does  not match with database |
| TC4 |  | Valid | Invalid | Password is Invalid | Password is incorrect |
| TC5 |  | Valid | Invalid | Password is Invalid | Password is not in database |
| TC6 |  | Invalid | Invalid | User ID and password is  incorrect | Login ID and Password not  in database |
| TC7 | Scenario 3 – Login Alternate flow  – User Exits | Valid/Invalid | n/a | User is allowed to exit and returns to login screen |  |

1. Test Case Matrix for Add Courses.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case ID | Scenario Name  and Description | Input 1  Course Id | Input 2  Title | Expected Output | Remarks |
| TC1 | Scenario 1 – Add course | Valid | Valid | Course added successfully |  |
| TC2 | Scenario 2 – list Course Alternate flow:  Invalid entry | Invalid | Valid/Invalid | Invalid Course Id | Course details are not correct |
| TC3 |  | Valid | Invalid | Invalid title |  |
| TC4 |  | Invalid | Invalid | Course Details are  Invalid | Course is not added in  database |
| TC5 | Scenario 3 – List Course Alternate flow:  User Exits | Valid/Invalid | Valid/Invalid | User comes and then exits from the  system |  |
| TC6 |  | n/a | n/a | User exits |  |

1. Test Case Matrix for Enroll in a Course.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case ID | Scenario Name and  Description | Input 1 Course Id | Input 2  User Id | Expected Output | Remarks |
| TC1 | Scenario 1 –  enroll Course | Valid | Valid | Enrolled successfully |  |
| TC2 | Scenario 2 – enroll Course Alternate flow:  Invalid Details | Invalid | Valid/Invalid | Course details are Invalid | Course details are not correct |
| TC3 |  | Valid | Invalid | User Id not valid | Database is not updated |
| TC4 |  | Invalid | Invalid | Course details not valid | Database is not updated |
| TC5 | Scenario 3 – enroll Course Alternate flow:  User Exits | Valid | n/a | User comes and then exits from the  system |  |

1. Test Case Matrix for View Courses.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case ID | Scenario Name and  Description | Input 1  Course Id | Expected Output | Remarks |
| TC1 | Scenario 1 – view course | Valid | Course is shown |  |
| TC | Scenario 2 – view course Alternate flow: Invalid entry | Invalid | Invalid  (Course ID) |  |
| TC2 | Scenario 3– view Course Alternate flow: User  Exits | Valid/Invalid | User comes and then exists from the system |  |
| TC3 |  | n/a | User exists |  |

1. Test Case Matrix for search Course.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case ID | Scenario Name and  Description | Input 1  (Course Id) | Input 2  (User Id) | Expected Output | Remarks |
| TC1 | Scenario 1 – search course | Valid | Valid | Suitable  Course is shown |  |
| TC2 | Scenario 2 – search course Alternate flow: Course not found | Invalid | Valid/Invalid | No course shown |  |
| TC3 |  | Valid | Invalid | Invalid combination of |  |
| TC4 | Scenario 3-  Search course  Alternate flow:  User exists | Valid | Valid | User exist | User comes and exists the system |