IT314: Software Engineering
Group-33

GROUP MEMBERS:

ID	Name
202201470	Priyank Asodariya
202201471	Dhwani Joshi
202201472	Kaushik Prajapati
202201475	Krutarth Kadia
202201476	Parv Patel
202201483	Neel Tandel
202201501	Jish Chanchapra
202201524	Yash Rathod
202201525	Heer Shah
202201527	Zenil Rupareliya

Initial Ideation of the the Project and its documentation

Sprint 1: Setup and Basic User Authentication

- ## User Stories

1. As an unregistered user I want to register to the platform so that I can take courses / I can create courses.

2. As a registered user I want to login to the platform so that I can take courses or I can create courses.

3. As a Developer, I want to directly manage user accounts and credentials in the database so that I can maintain platform integrity and ensure secure authentication.

4. As a Developer, I want to access system logs and modify the platform's configurations so that I can address security concerns and enhance functionality.

- ## Use Cases

Use Case 1: Registering as a New User

➔ **Actors**

- **Primary Actor**: Unregistered User
- **System**: The application's registration module

➔ **Preconditions**

- The user does not have an existing account.

- The registration button is accessible.

➔ **Main Flow**
- **User** navigates to the home page and clicks on the register button.
- **User** provides required information (e.g., name, email, password).
- **User** selects the type of account (e.g., "Student" for taking courses or "Instructor" for creating courses).
- **User** submits the registration form.
- **System** validates the information.
- **System** creates the user account and confirms registration.

➔ **Alternate Flows**
- **Invalid Information**:
    - **System** displays an error message if any field is invalid (e.g., invalid email).
    - **User** corrects the information and resubmits.
- **Duplicate Account**:
    - If the email is already in use, **System** notifies the user and prompts them to log in or use a different email.
    - If the username is already in use, **System** notifies the user and prompts them to use a different username.

➔ **Postconditions**
- The user has a new account and can log in to take or create courses, depending on the selected account type.

Use Case 2: User Login

➔ **Actors**
- **Primary Actor**: Unlogged User
- **System**: The platform’s authentication module

➔ **Preconditions**
- The login button is accessible.

➔ **Main Flow**

- **User** navigates to the login button and clicks it.
- **User** enters their email and password.
- **User** submits the login form.
- **System** authenticates the credentials.
- **System** grants access and redirects the user to their respective dashboard.

➔ **Alternate Flows**

- **Invalid Credentials**:
    - If credentials are incorrect, **System** displays an error message.
    - **User** can retry.
- **Unregistered Credentials**:
    - If the **User** is unregistered then they can click on the register link displayed below to create a new account.

➔ **Postconditions**

- The user is logged in and can access features based on their account type (taking or creating courses).

Use Case 3: Full Access Management by Developer

➔ **Actors:**

- Developer

➔ **Description:**

The Developer, as the creator and maintainer of the platform, has full access to the codebase and database, allowing them to manage users, modify system functionality, and ensure the platform's integrity and security.

➔ **Preconditions:**

- The Developer has access to the system’s codebase and database tools.

➔ **Main Flow:**

- Developer accesses the system (via the codebase or database management tools).
- Developer performs any of the following tasks:
    - Adds, updates, or deletes user accounts directly in the database.
    - Monitors user activity logs to detect potential misuse or security issues.
    - Removes malicious or inactive users from the system.
    - Modifies system settings or configurations to improve functionality or resolve issues.
    - Conducts database maintenance (e.g., backups and restores).
- Developer ensures data is securely stored and applies encryption where necessary.
- Developer reviews logs to identify bugs, vulnerabilities, or areas for improvement.

➔ **Alternate Flow:**

- If a system malfunction occurs:
    - Developer accesses the codebase to debug and resolve the issue.
    - Performs immediate database fixes if needed (e.g., clearing corrupted records).

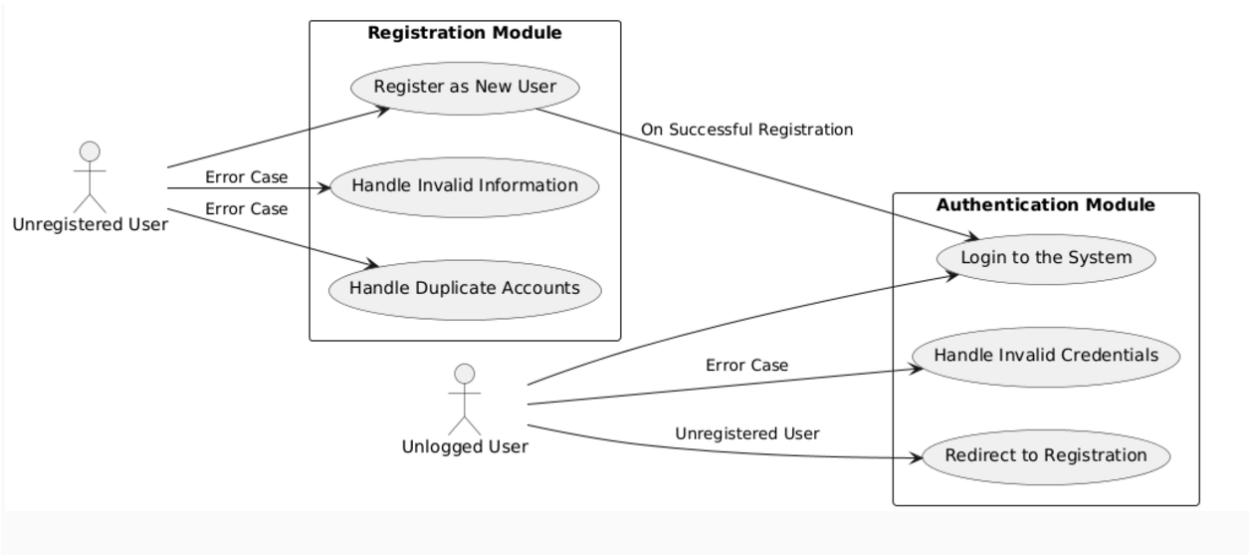
➔ **Postconditions:**

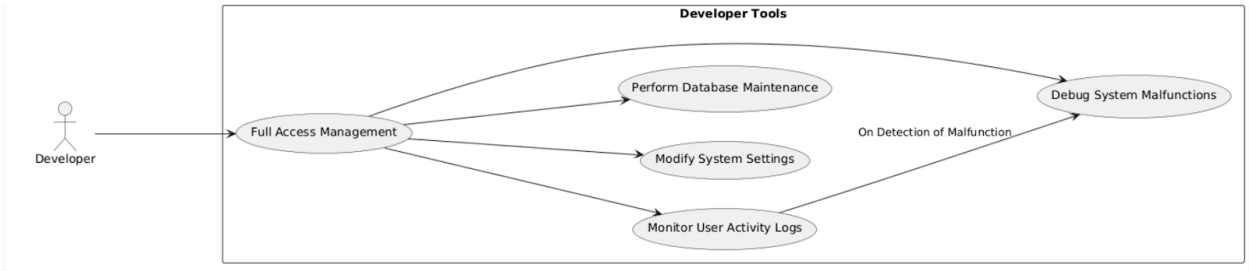
- User accounts and system configurations are updated as needed.
- Platform functionality and security are maintained.

➔ **Exceptions:**

- Developer encounters unexpected downtime or a server crash, requiring recovery actions.
- Developer unintentionally introduces a bug, requiring rollback or patching.

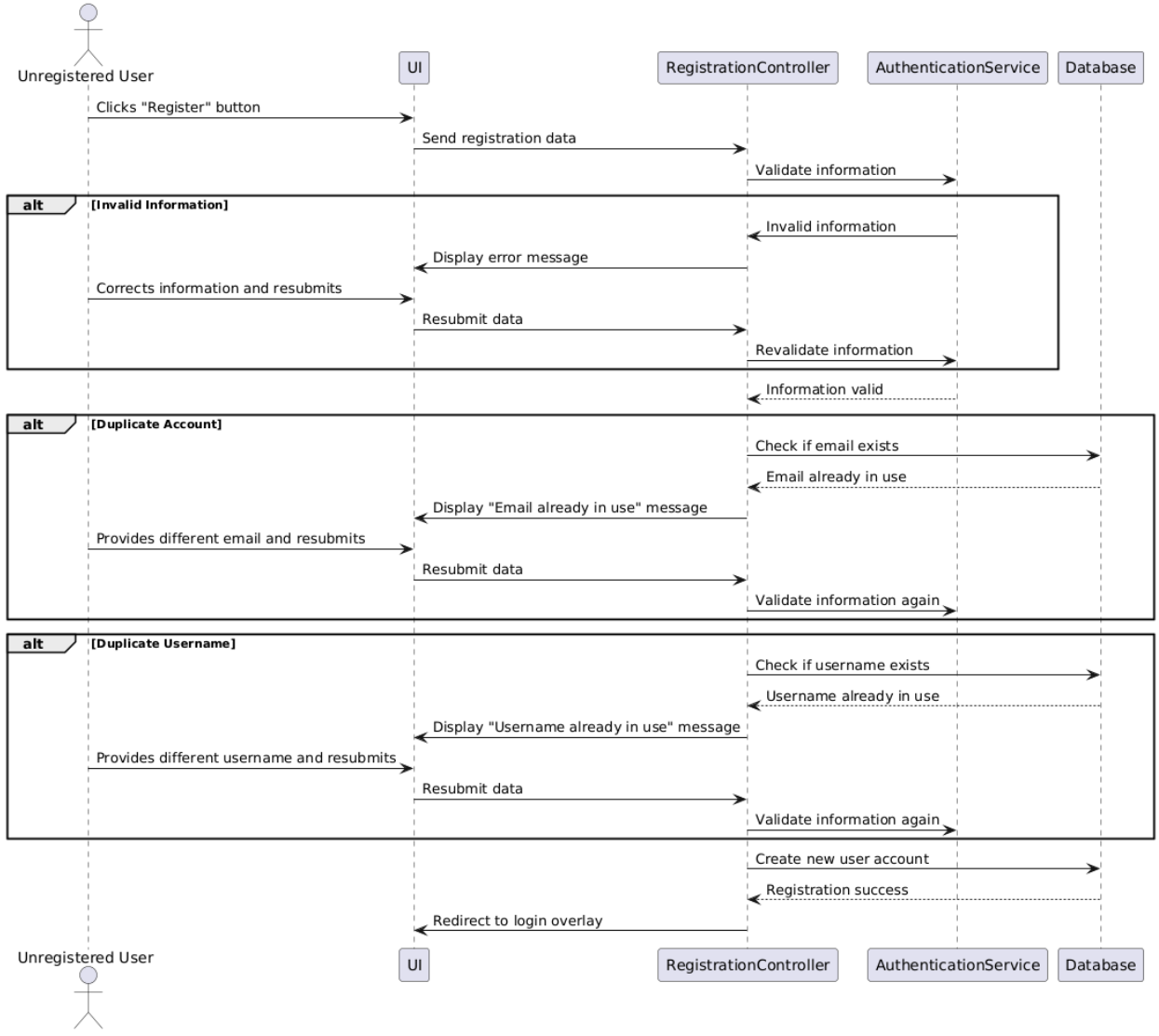
- **Use Case Diagram**



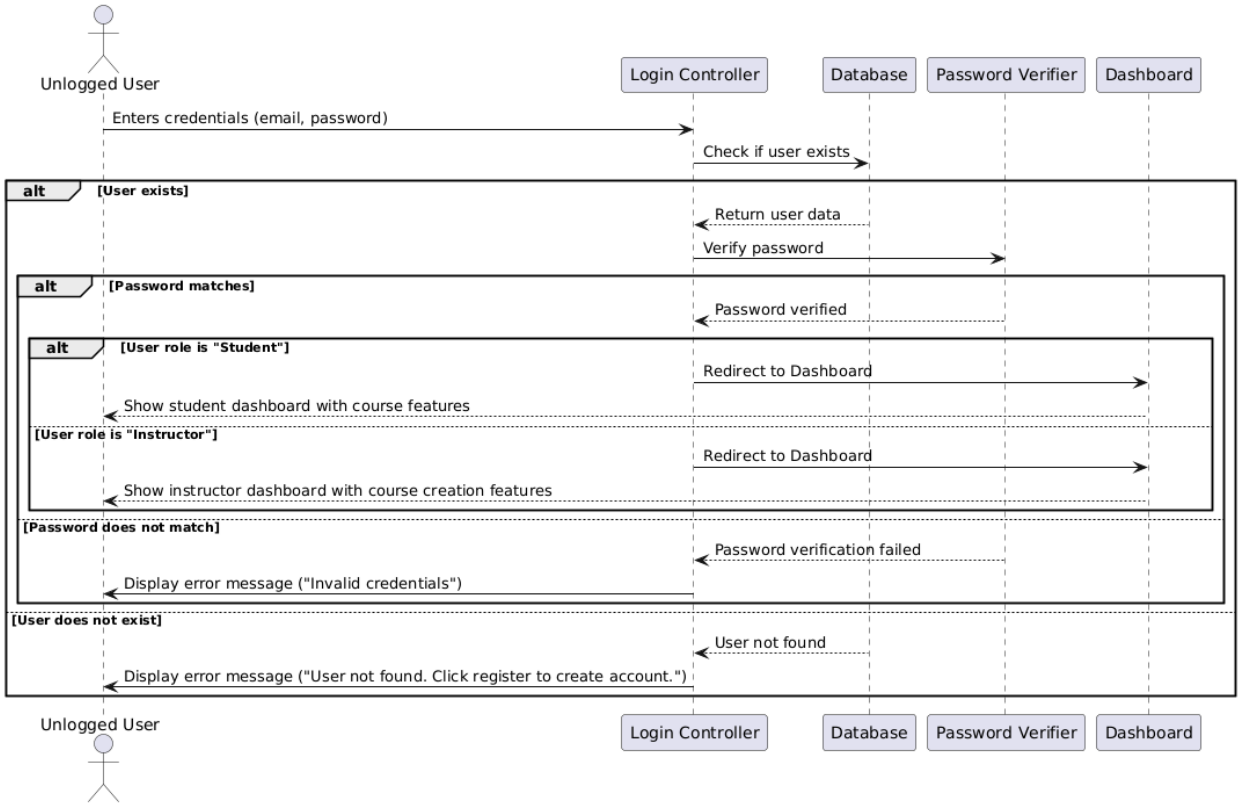


- **Sequence Diagram**

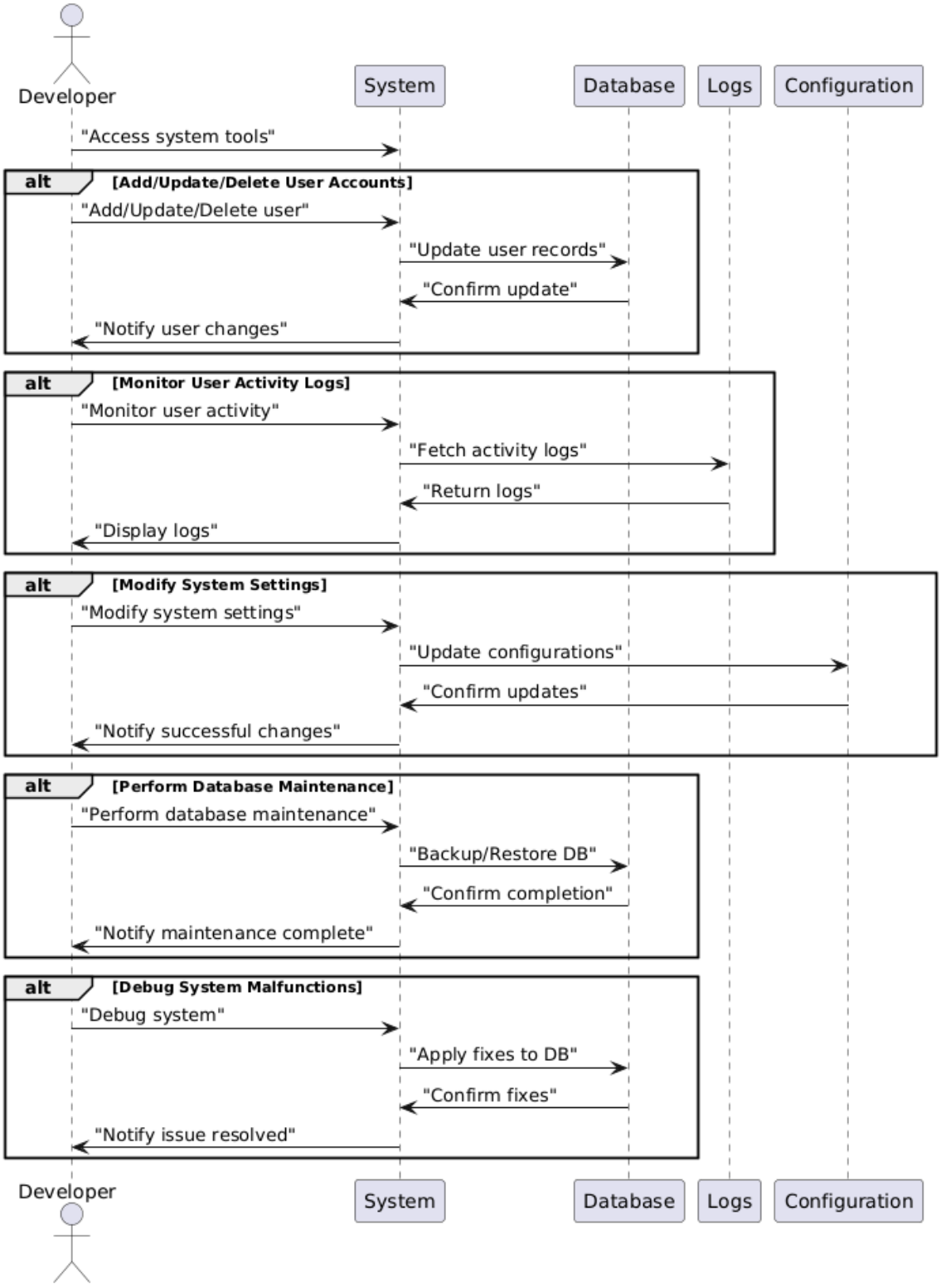
➔ **For registering**



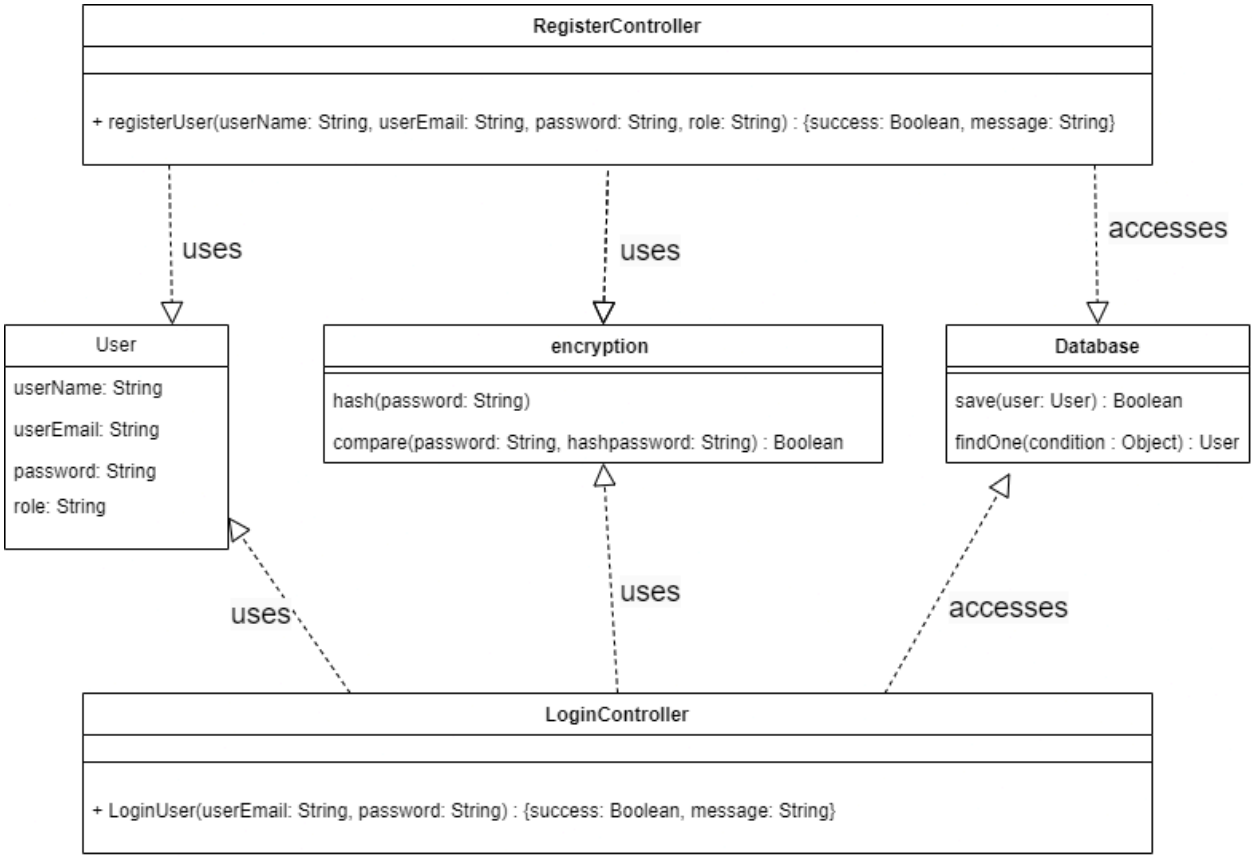
➔ **For logging in**



➔ **For Database management**

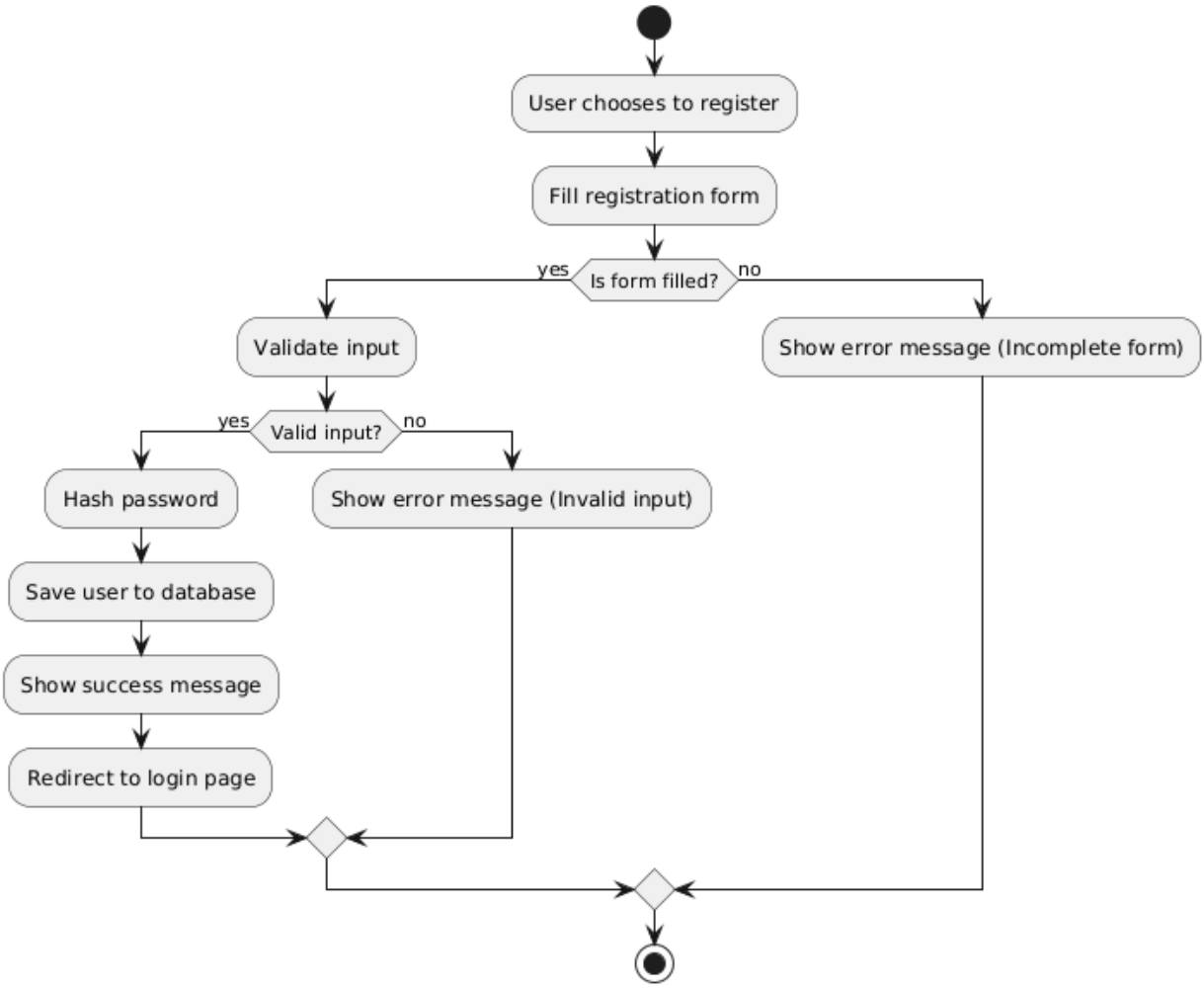


- **Class Diagram**

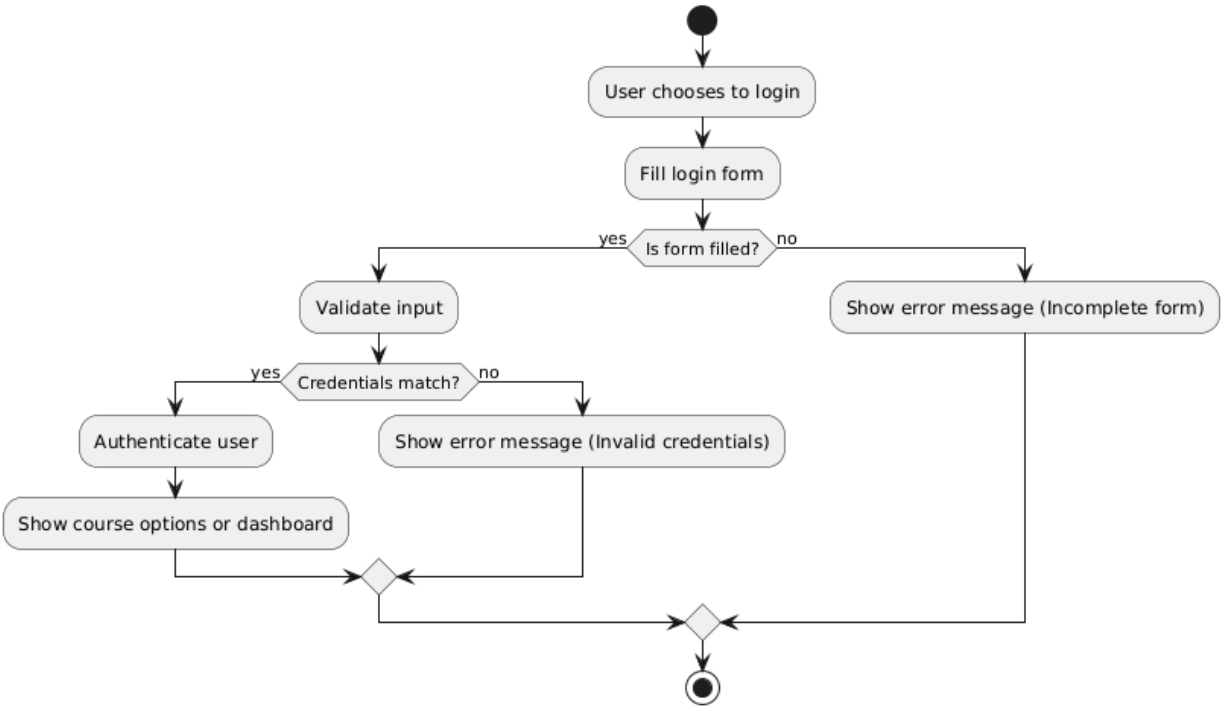


- **Activity Diagram**

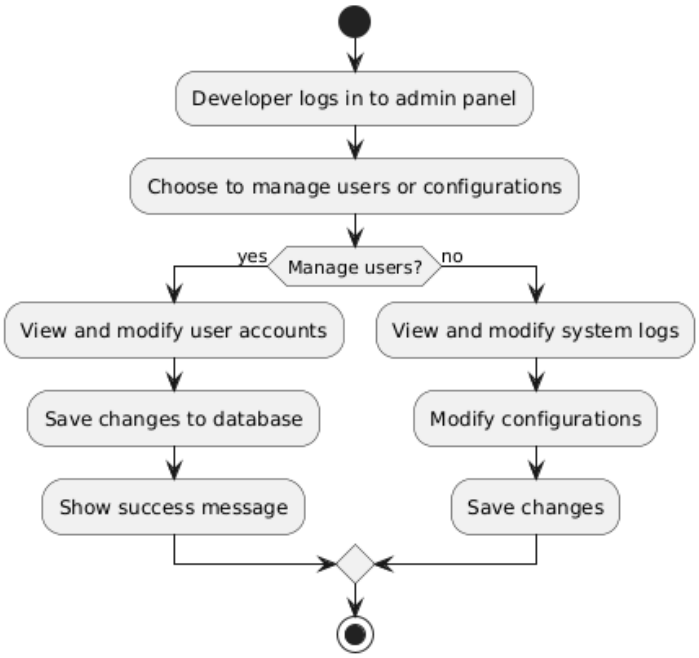
➔ **For registering**



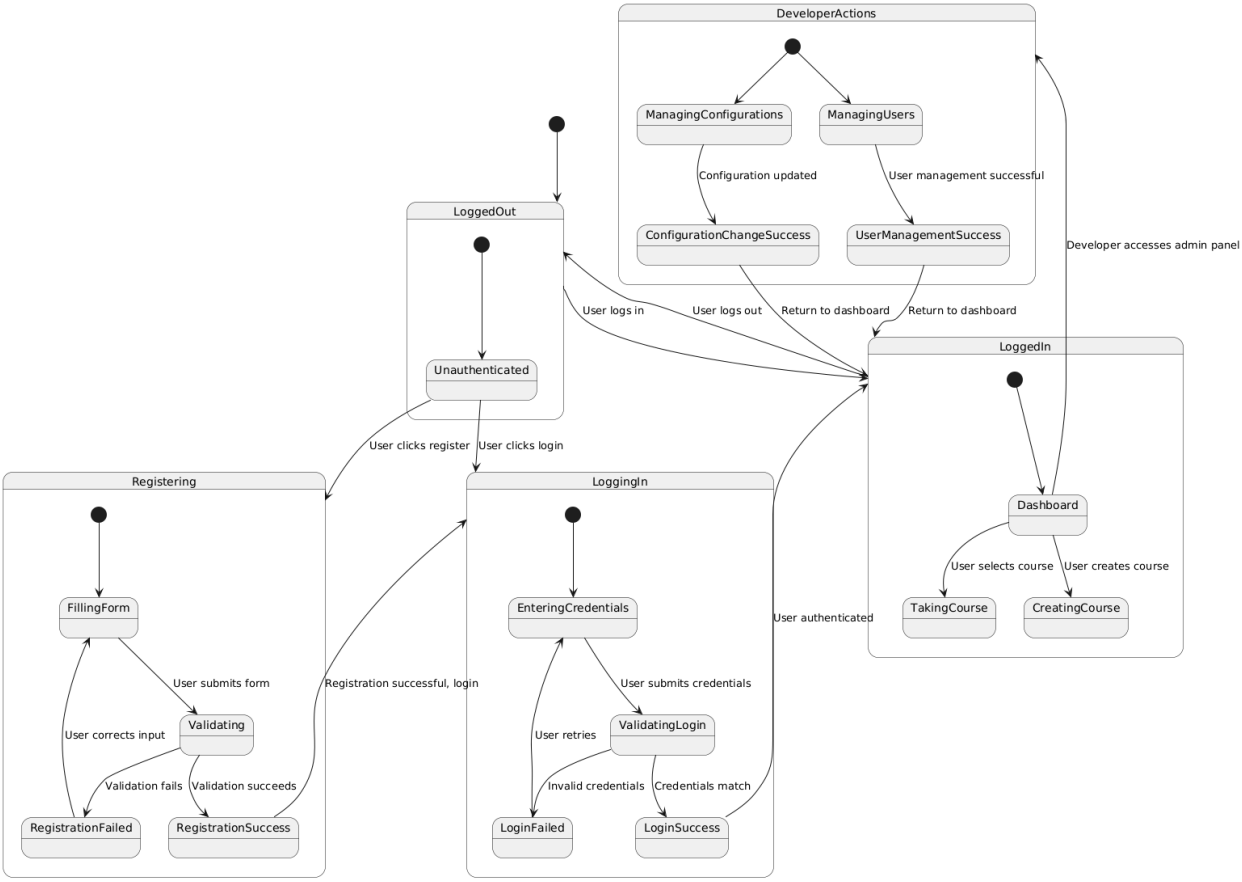
➔ **For logging in**



➔ **For Database management**



- **State Diagram**



Sprint 2: Profile Management

- ## User Stories

1. As a user I want to update my profile page so that my profile has up-to-date information.

2. As a Student, I want my personalized dashboard so that I can see my progress and quizzes.

- ## Use Cases

Use Case 1: Update Profile

➜ **Actor:** User
➜ **Preconditions:**
- User is logged in.
- Profile page is accessible.

➜ **Postconditions:**
- Profile information is updated in the system database.’

➜ **Main Flow:**
- User navigates to the profile page.
- User clicks the "Edit" button to modify the profile.
- User updates fields (e.g., name, email, password).
- User submits the changes.
- System validates the inputs.
- System saves the changes and displays a success message.

➜ **Alternative Flow:**
- If the user cancels the operation, the system discards any changes.

➜ **Exceptions:**
- Invalid input: System displays an error message and highlights incorrect fields.
- System error: Profile updates fail, and the system displays a failure notification.

Use Case 2: View Personalized Dashboard

➜ **Actor:** Student

➜ **Preconditions:**
- Student is logged in.
- Dashboard is configured for the specific student.

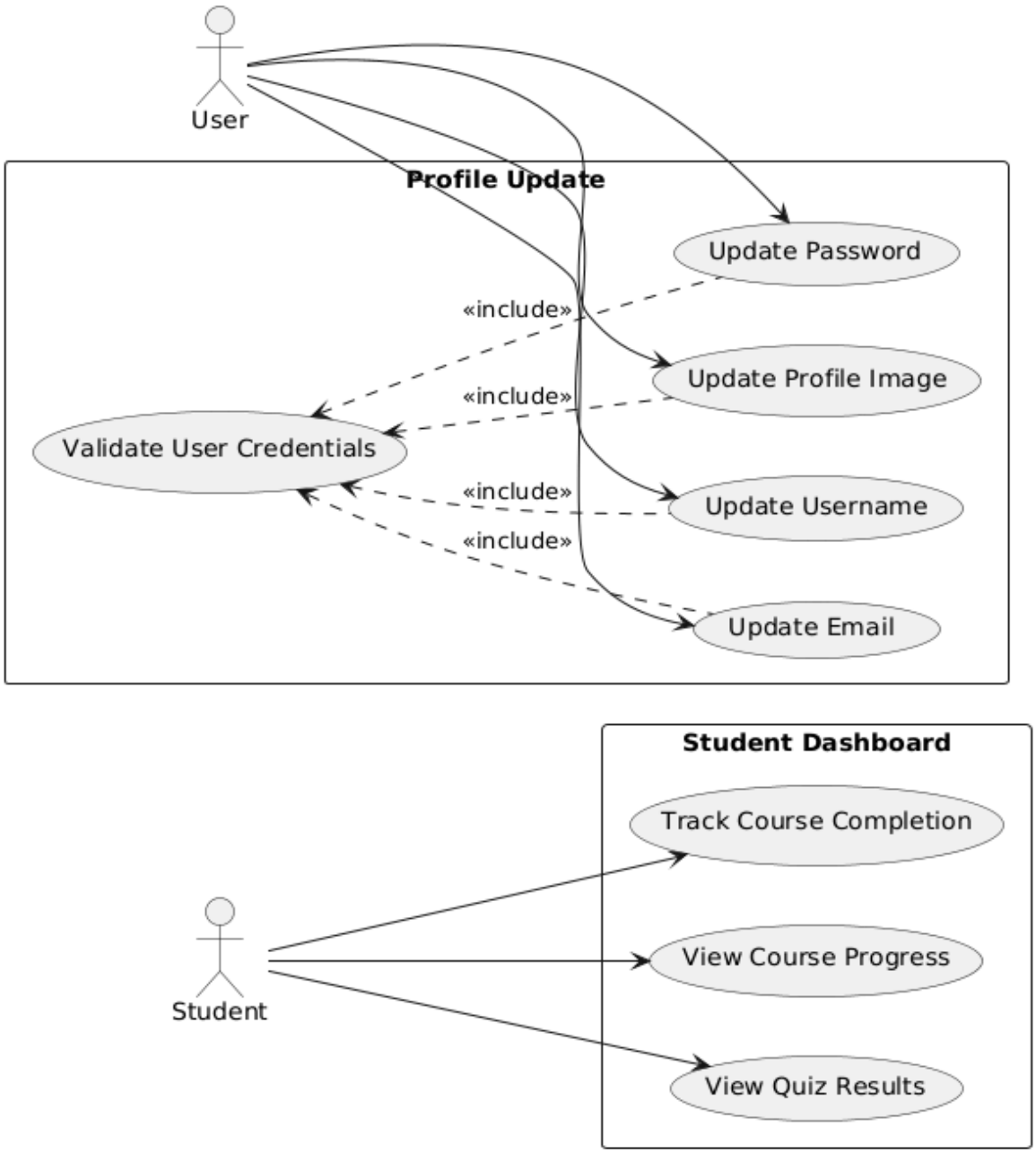
➜ **Postconditions:**
- Dashboard displays current progress and quizzes.

➜ **Main Flow:**
- Student logs into the system.
- Student navigates to the dashboard page.
- System displays the student's all courses, finished courses, in-progress courses and list of quizzes.

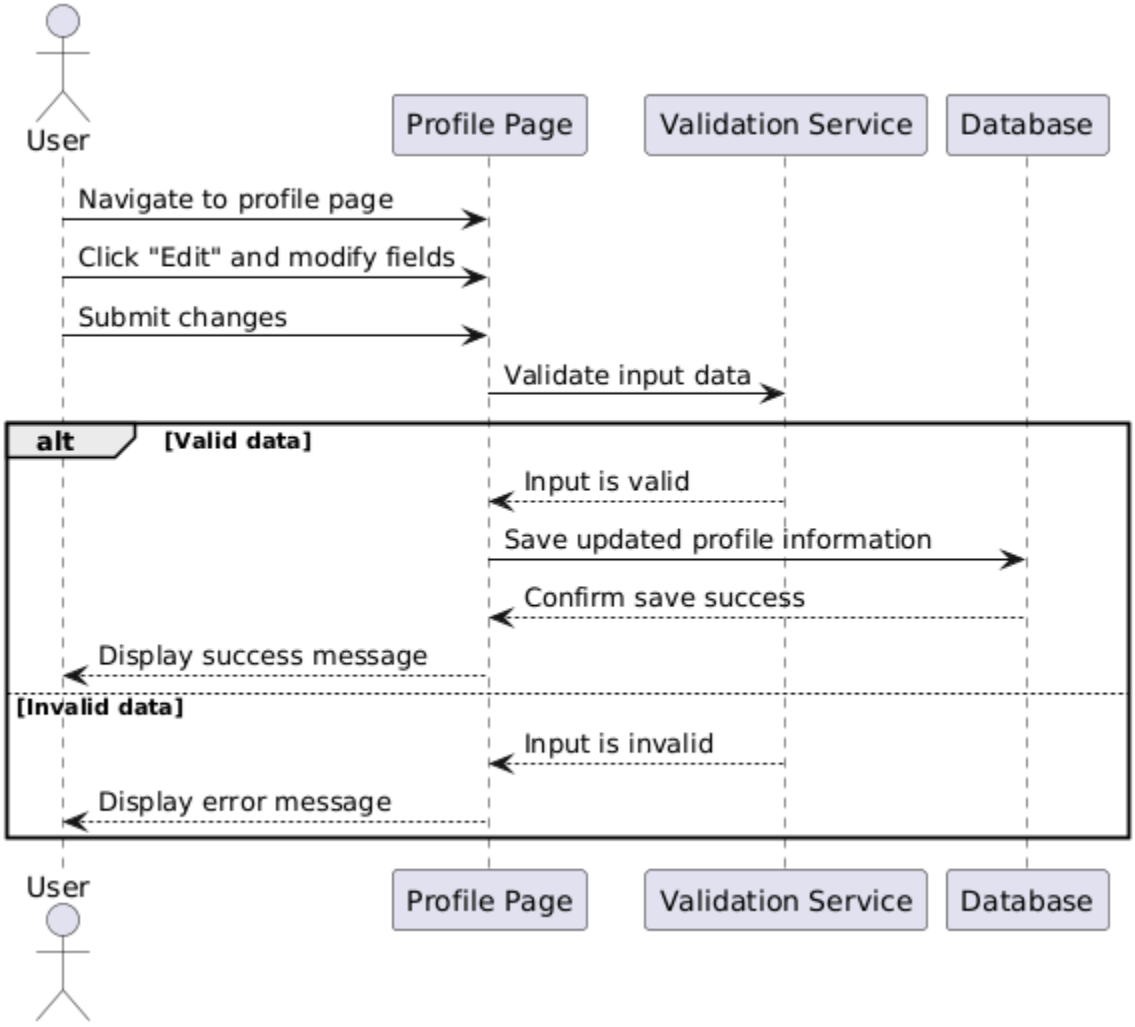
➜ **Alternative Flow:**
- If there are no quizzes available, the dashboard only shows progress information.

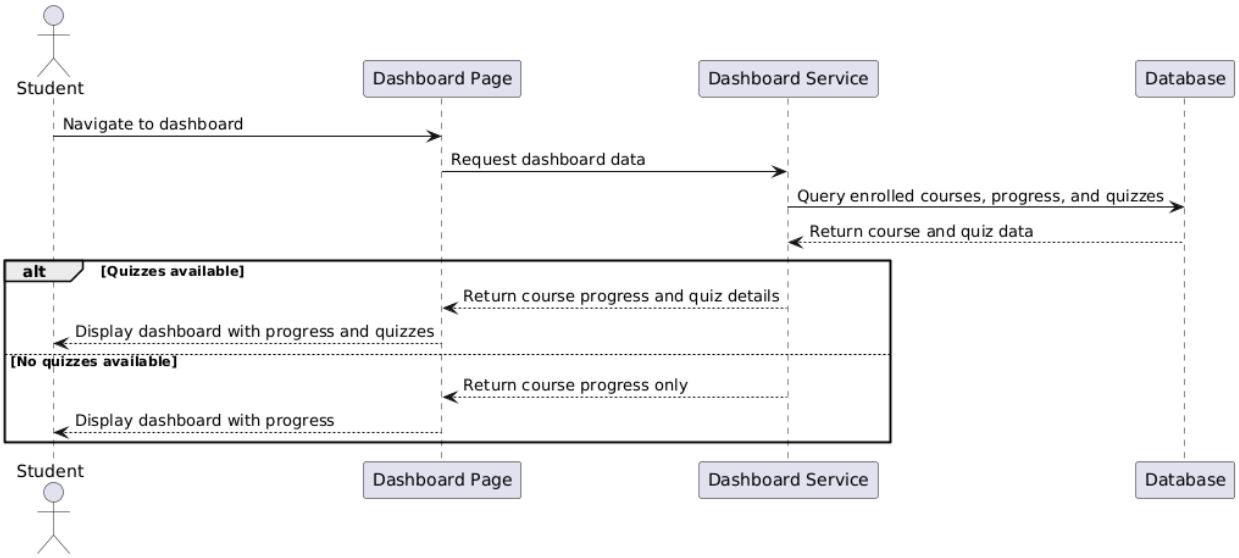
➜ **Exceptions:**
- System error: Dashboard fails to load, and the system shows an error message.

- **Use Case Diagram**

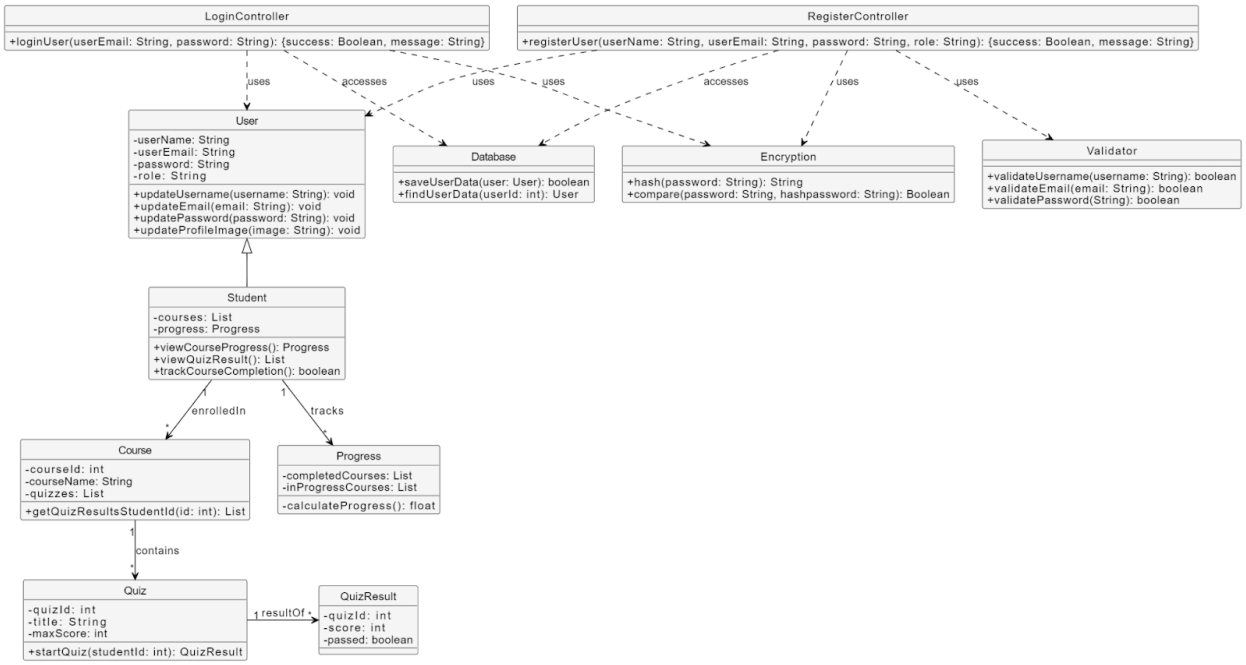


- **Sequence Diagram**

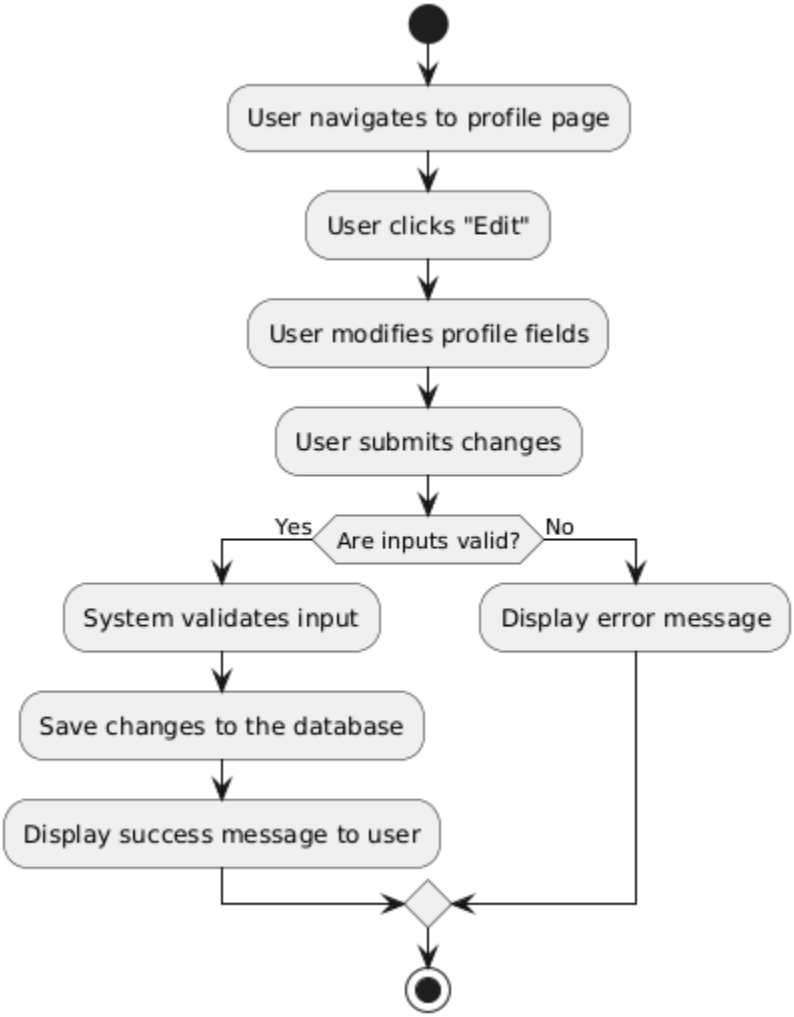


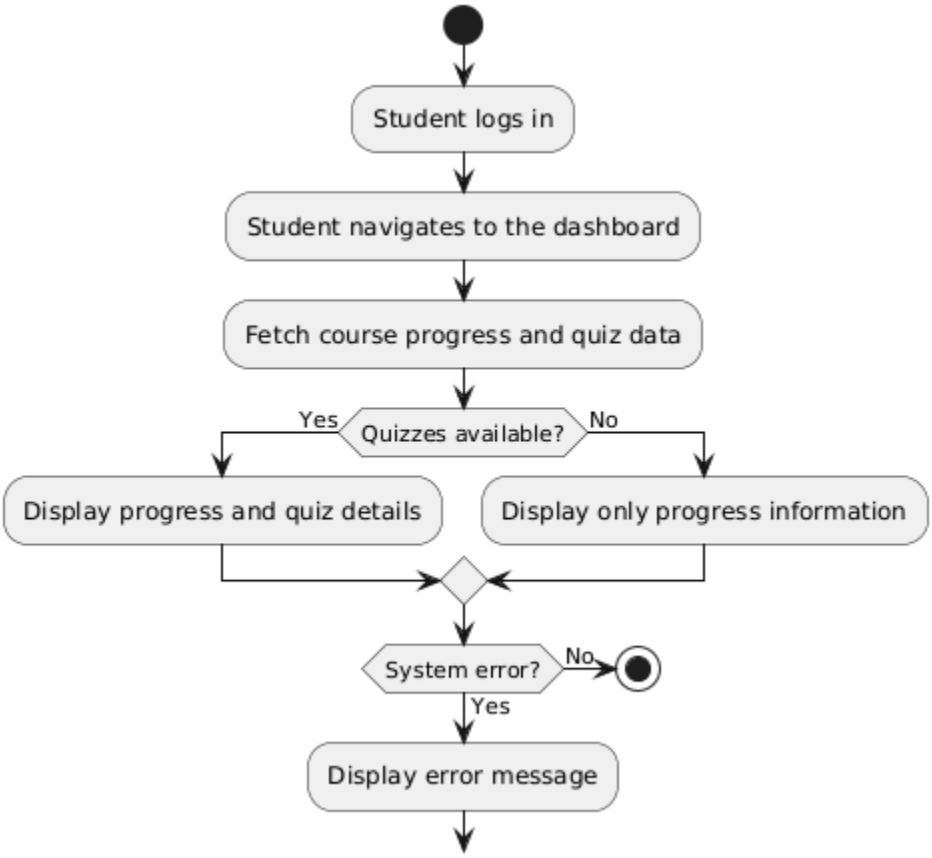


- **Class Diagram**

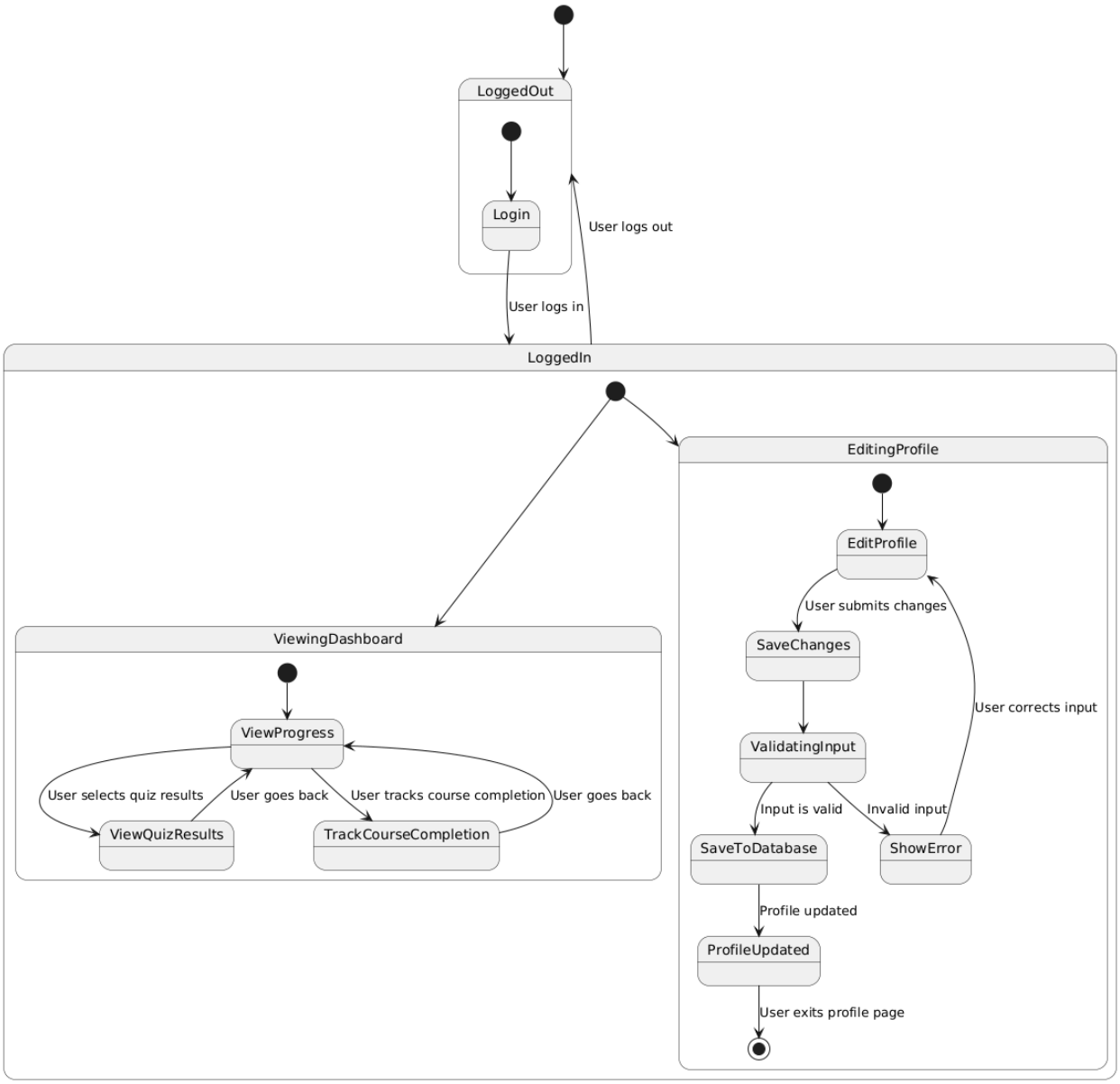


- **Activity Diagram**




Student logs in
Student navigates to the dashboard
Fetch course progress and quiz data
Quizzes available?
Yes
No
Display progress and quiz details
Display only progress information
System error?
No
Yes
Display error message

- **State Diagram**



Sprint 3: Course Management

- ## User Stories

1. As a user I want to search and filter courses so that I can see available courses of my preference.
2. As a user I want to preview a course and watch demo lectures of a course before purchasing a course so that I can make better decisions for a course to take or not to take.
3. As an instructor, I want to create a course so that I can share my knowledge with learners.
4. As an instructor, I want a dashboard so that I can efficiently manage all course-related details from one place.
5. As an instructor, I want to collaborate with other instructors to develop courses together.
6. As an instructor, I want to be able to set pricing and discounts on my offered course so that more students will enrol.

- ## Use Cases

Use Case 1: Search and Filter Courses

➔ **Actors**

- **Primary Actor:** Logged User
- **System:** The platform's course search and filter module

➔ **Preconditions**

- The search bar and filter options are accessible on the platform.

➔ **Main Flow**

- User navigates to the search bar.
- User types keywords related to their desired course (e.g., “Python programming”).
- User clicks the search button.
- System fetches and displays a list of matching courses.
- User applies filters (e.g., difficulty level, price range, language).
- System refines the search results based on the selected filters.

➔ **Alternate Flows**

- **No Matching Results:**
    - If no courses match the search query, the System displays a “No results found” message.

➔ **Postconditions**

- The user views a customised list of courses matching their preferences.

Use Case 2: Preview a Course and Watch Demo Lectures

➔ **Actors**

- **Primary Actor:** Logged User
- **System:** The platform’s course content and preview module

➔ **Preconditions**

- The course catalogue and preview options are accessible.
- Demo content is available for selected courses.

➔ **Main Flow**

1. User selects a course from the course catalogue.
2. System displays course details, including title, description, instructor bio, and reviews.
3. User clicks the “Preview” or “Watch Demo Lecture” button.

4. System streams the demo content for the user.

➔ **Alternate Flows**

- **No Demo Available:**
    - If demo content is unavailable, the System displays a message indicating “No demo available for this course.”
    - User can still review other course details to make a decision.

➔ **Postconditions**

- The user views the course demo and can decide whether to purchase the course.

Use Case 3: Create a Course

➔ **Actors**

- **Primary Actor:** Instructor
- **System:** The platform’s course creation module

➔ **Preconditions**

- The instructor is logged in and has access to the “Create Course” section.

➔ **Main Flow**

- Instructor navigates to the “Create Course” section.
- Instructor enters course details (e.g., title, description, category).
- Instructor uploads course content (e.g., videos, PDFs, quizzes).
- System saves the course as a draft or publishes it, based on the instructor’s choice.

➔ **Alternate Flows**

- **Missing Information:**
    - If any required fields are incomplete, the System prompts the instructor to fill in the missing details.

➔ **Postconditions**

- The course is created and saved as a draft or published.

Use Case 4: Manage Course Details via a Dashboard

➔ **Actors**

- **Primary Actor:** Instructor
- **System:** The platform's instructor dashboard module

➔ **Preconditions**

- The instructor is logged in and has active courses to manage.

➔ **Main Flow**

- Instructor navigates to their dashboard.
- Instructor edits course details, adds new content, or adjusts pricing.
- System saves the changes and updates the course listing.

➔ **Alternate Flows**

- **No Active Courses:**
    - If no courses are available, the System displays a message encouraging the instructor to create a course.

➔ **Postconditions**

- The instructor efficiently manages course-related activities through the dashboard.

Use Case 5: Collaborate with Other Instructors

➔ **Actors**
- **Primary Actor:** Instructor
- **System:** The platform's collaboration module

➔ **Preconditions**
- The instructor has collaboration permissions.

➔ **Main Flow**
- Instructor invites a co-instructor to collaborate on a course.
- Co-instructor receives and accepts the invitation.
- Both instructors edit and manage the course content collaboratively.

➔ **Alternate Flows**
- **Invitation Declined:**
    - If the co-instructor declines the invitation, the System notifies the primary instructor.

➔ **Postconditions**
- Both instructors collaborate effectively to develop the course.

Use Case 6: Set Pricing and Discounts for a Course

➔ **Actors**
- **Primary Actor:** Instructor
- **System:** The platform's pricing and discount module

➔ **Preconditions**
- The instructor has at least one active course.

➔ **Main Flow**

- Instructor navigates to the pricing section for a course.
- Instructor sets the base price and defines discounts.
- System calculates and displays the final price for students.
- Instructor saves the pricing details.

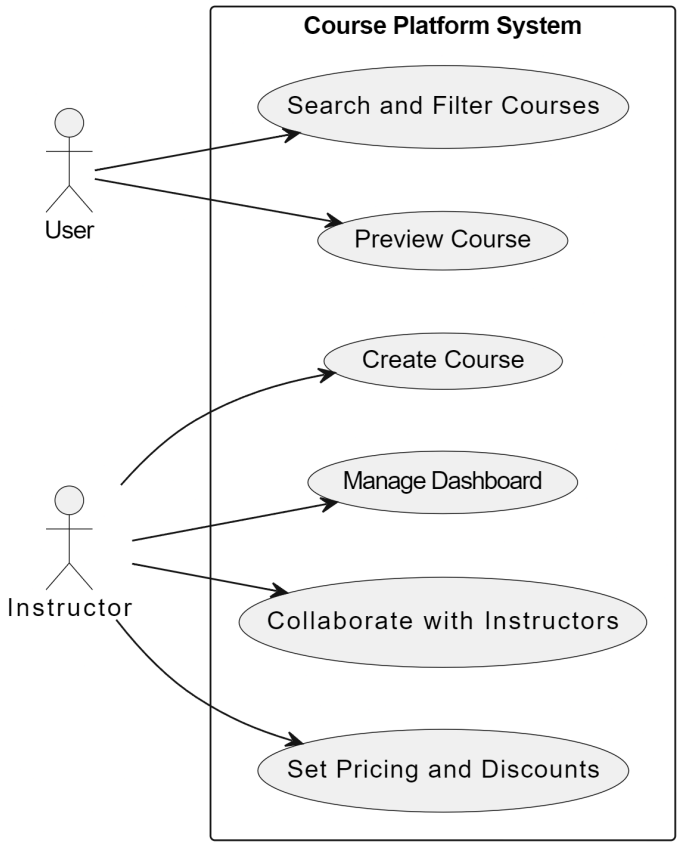
➔ **Alternate Flows**

- **Invalid Pricing Input:**
    - If the entered price or discount is invalid, the System prompts the instructor to correct it.

➔ **Postconditions**

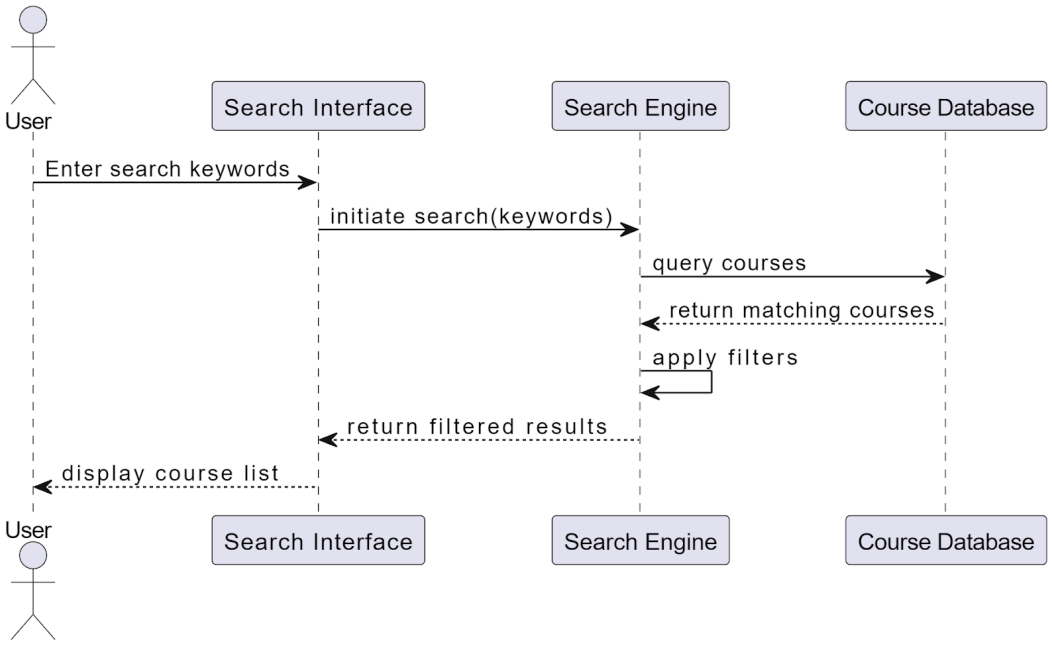
- The course listing is updated with the new pricing and discount structure.

- **Use Case Diagram**

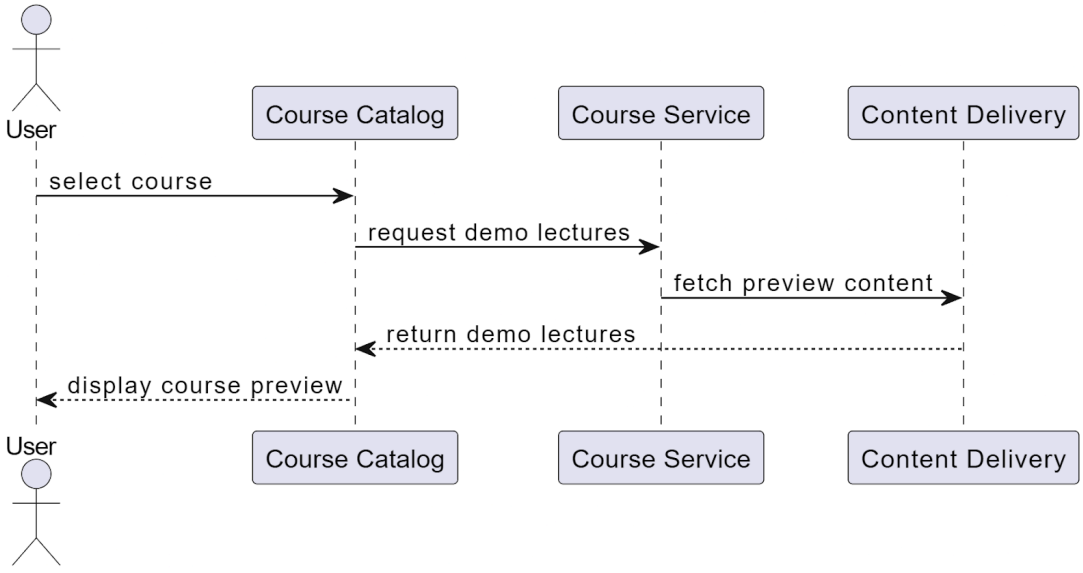


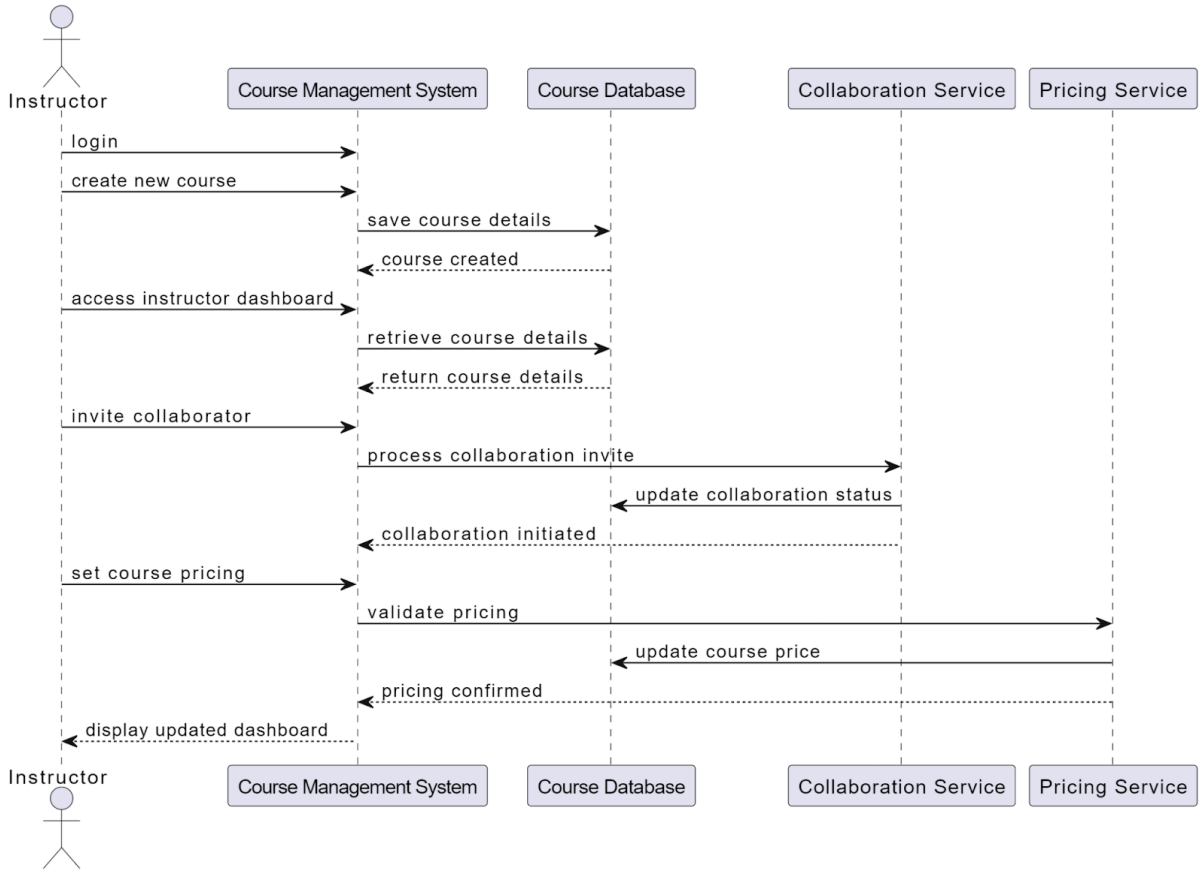
- **Sequence Diagram**

➔ **Search and Filter Courses**

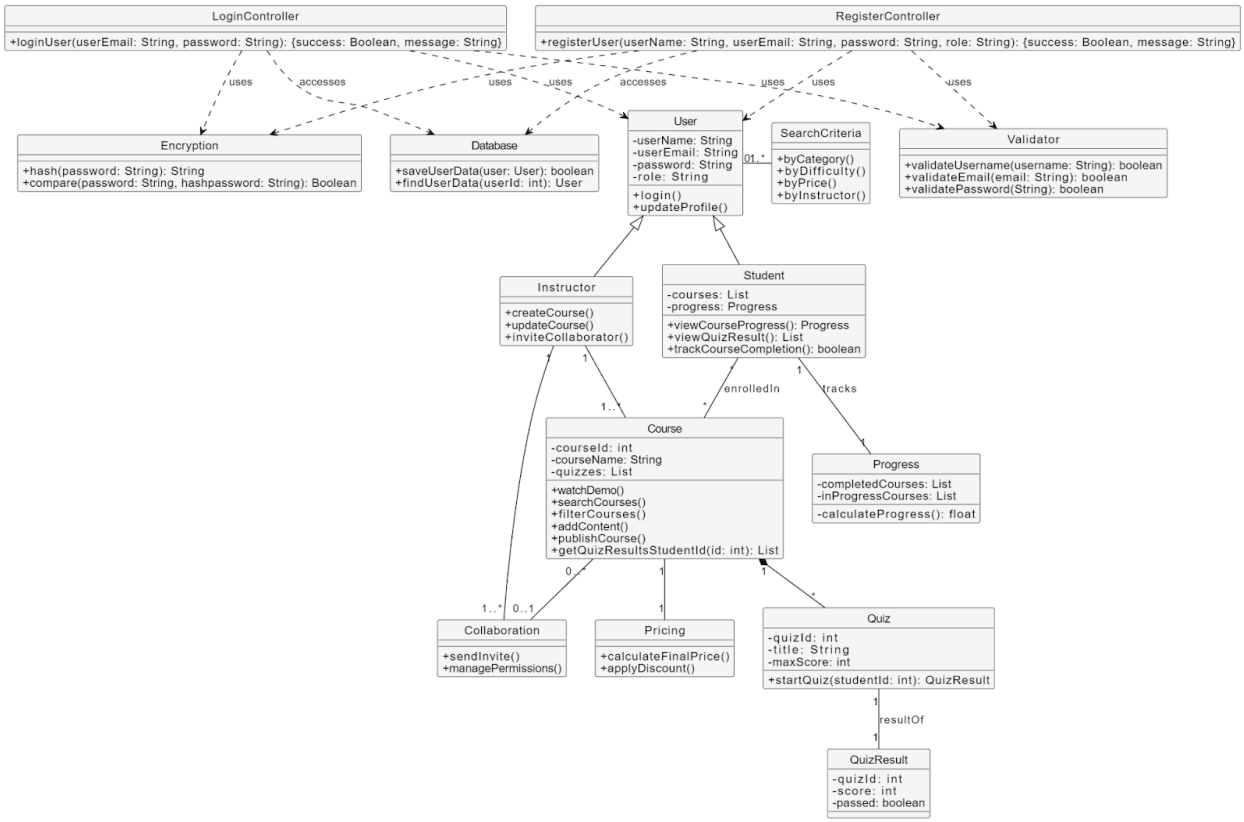


➔ **Demo Lectures**



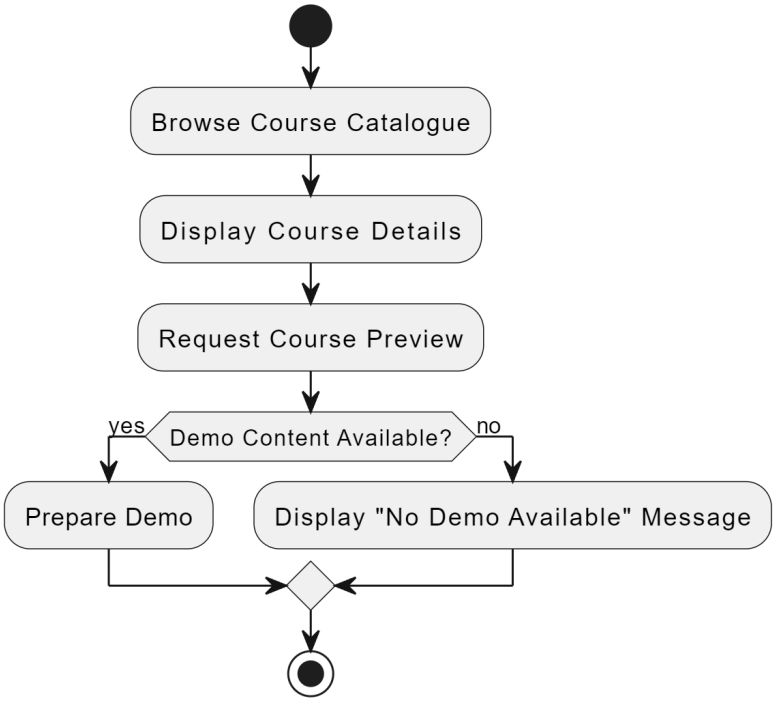

Instructor
Course Management System
Course Database
Collaboration Service
Pricing Service
login
create new course
save course details
course created
access instructor dashboard
retrieve course details
return course details
invite collaborator
process collaboration invite
update collaboration status
collaboration initiated
set course pricing
validate pricing
update course price
pricing confirmed
display updated dashboard
Instructor
Course Management System
Course Database
Collaboration Service
Pricing Service

- **Class Diagram**

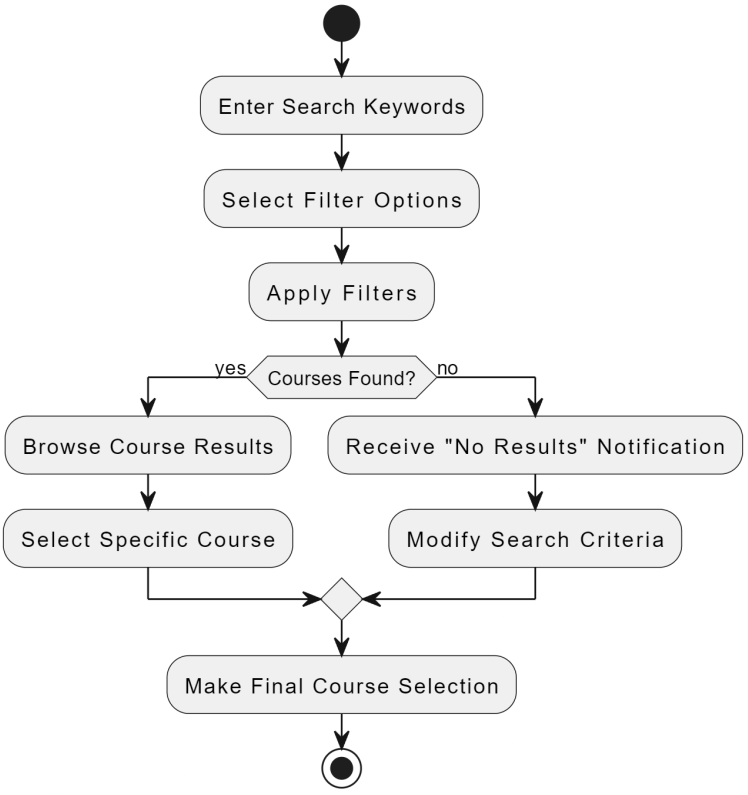


- **Activity Diagram**

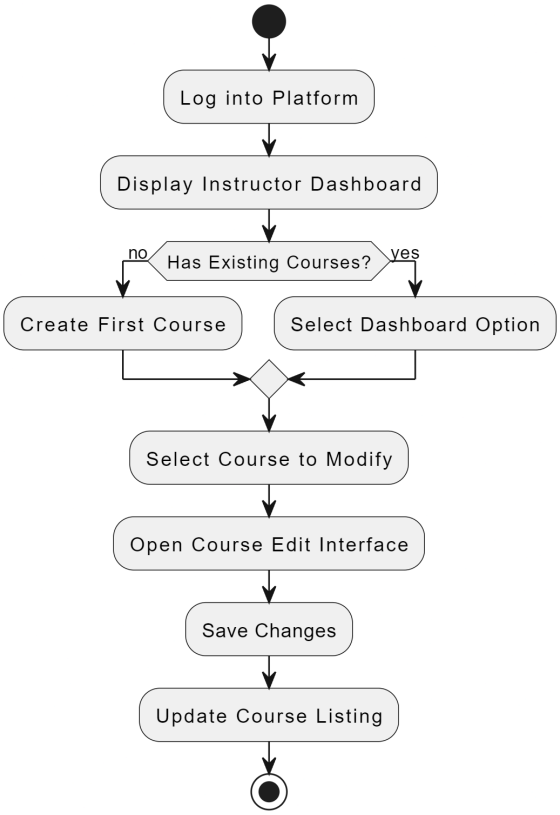
➔ **Demo Courses**



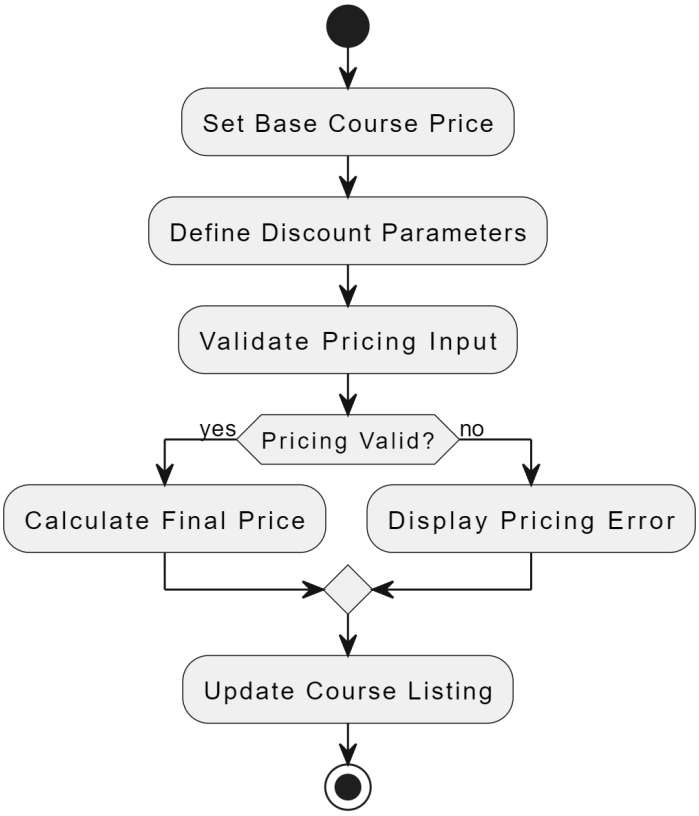
➔ Search and Filters



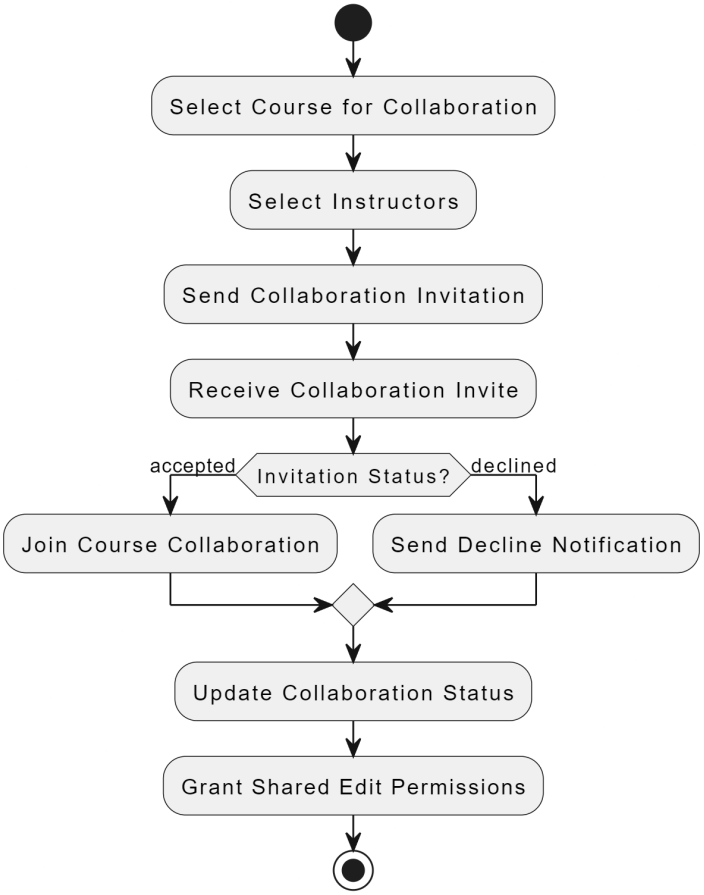
➔ Create Course and Dashboard



➔ **Set Pricing and Discounts**

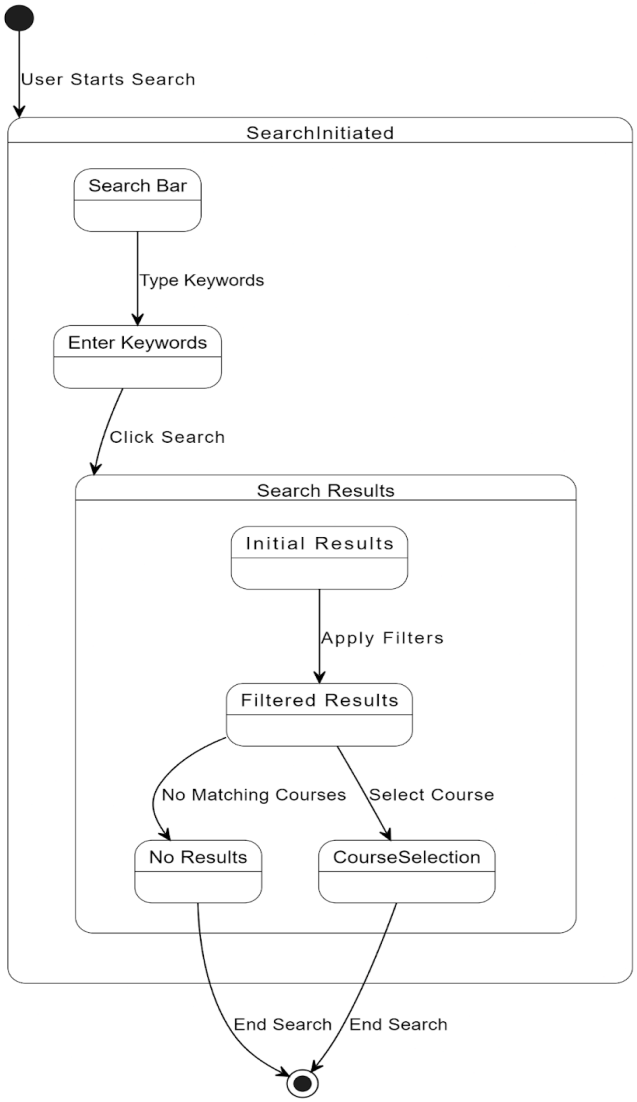


➔ Collaboration among Instructors

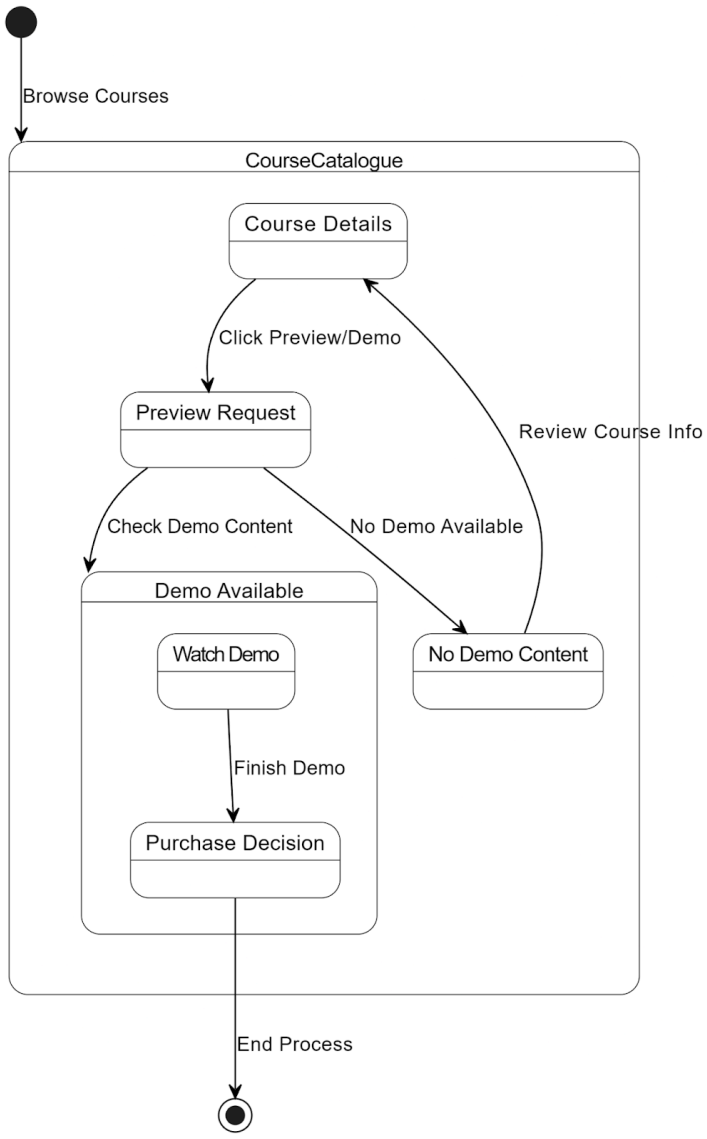


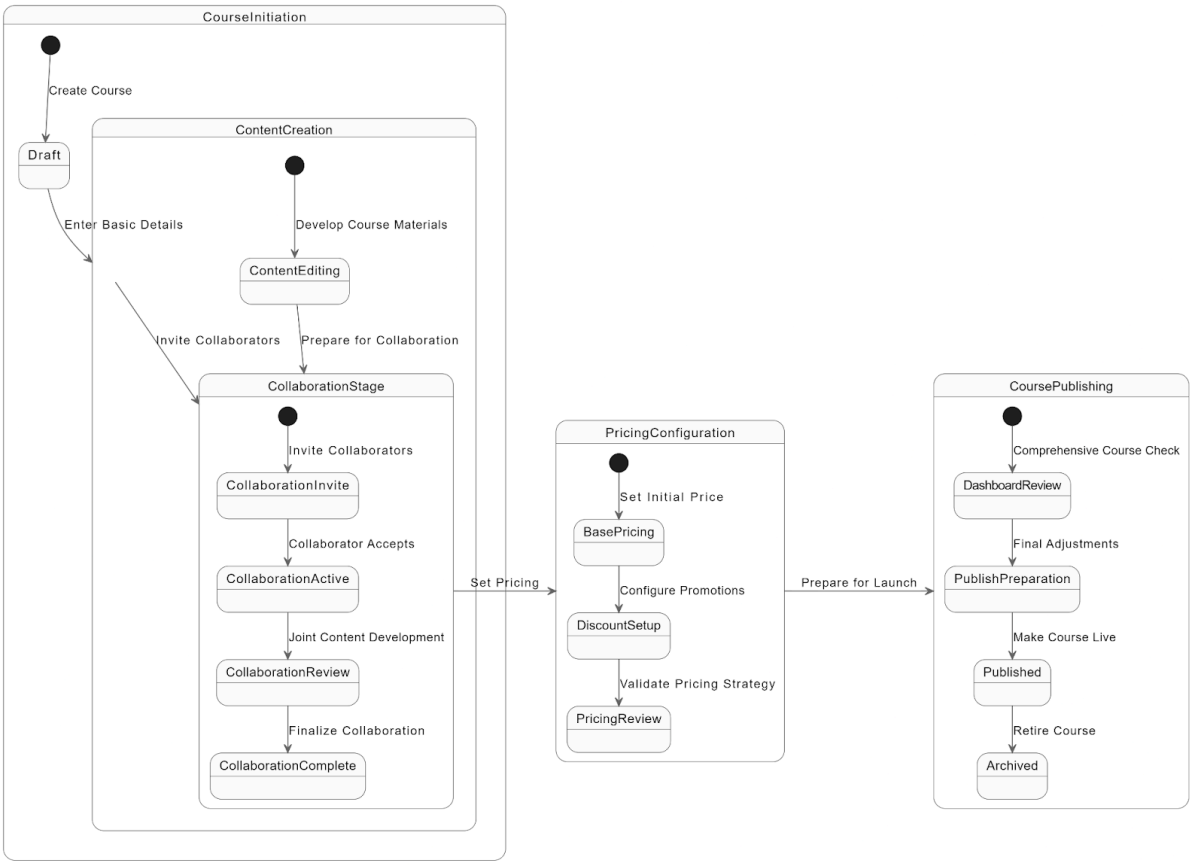
- **State Diagram**

➔ **Search and Filtering**



➔ **Demo course**


Browse Courses
CourseCatalogue
Course Details
Click Preview/Demo
Preview Request
Review Course Info
Check Demo Content
No Demo Available
Demo Available
Watch Demo
No Demo Content
Finish Demo
Purchase Decision
End Process


CourseInitiation
Create Course
Draft
ContentCreation
Enter Basic Details
Develop Course Materials
ContentEditing
Invite Collaborators
Prepare for Collaboration
CollaborationStage
Invite Collaborators
CollaborationInvite
Collaborator Accepts
CollaborationActive
Joint Content Development
CollaborationReview
Finalize Collaboration
CollaborationComplete
Set Pricing
PricingConfiguration
Set Initial Price
BasePricing
Configure Promotions
DiscountSetup
Validate Pricing Strategy
PricingReview
Prepare for Launch
CoursePublishing
Comprehensive Course Check
DashboardReview
Final Adjustments
PublishPreparation
Make Course Live
Published
Retire Course
Archived

Sprint 4: Learning Tools

● User Stories

1. As an instructor I want to upload videos to my course so that students can access it remotely.
2. As an instructor I want to take live lectures so that students can learn interactively.
3. As an instructor, I want to be able to upload course materials so that students can revise it.
4. As an instructor, I want to create a quiz so that students can evaluate their progress.
5. As a student, I want to download course materials of enrolled courses, so that I can access them offline.
6. As a user, I want to participate in discussion forums so that I can interact with other students and the course instructor to solve my doubts about the course content.
7. As a system, I want to grade quizzes, so that students can see their scores and understand their performance.

● Use Cases

Use Case 1: Upload Videos

➔ **Actors**

- **Primary Actor:** Instructor
- **System:** The platform's course management module

➔ **Preconditions**

- Instructor is logged into the platform.
- The instructor has the necessary permissions to upload videos.

➔ **Main Flow**

- Instructor navigates to their course dashboard.
- Instructor selects the "Upload Video" option.
- Instructor chooses a video file from their device and provides details.
- System uploads the video to the course.
- System confirms the upload with a success message, and the video is added to the course materials.

➔ **Alternate Flows**

- **Invalid File Format:**
    - If the uploaded file is not in an accepted format, the system displays an error message and prompts the instructor to upload a valid file.

➔ **Postconditions**

- The video is accessible to students enrolled in the course.

Use Case 2: Conduct Live Lectures

➔ **Actors**

- **Primary Actor:** Instructor
- **System:** The platform’s live session module

➔ **Preconditions**

- Instructor and students are logged into the platform.
- The live session feature is enabled for the course.

➔ **Main Flow**

- Instructor schedules a live lecture.
- System sends notifications to enrolled students about the scheduled lecture.
- At the scheduled time, the instructor starts the live session.
- Students join the session via a provided link or course dashboard.
- Instructor conducts the lecture.

➔ **Alternate Flows**

- **Student Unable to Join:**
    - If a student cannot join or the instructor cannot conduct the live session due to technical reasons.

➔ **Postconditions**

- Students participate in the live session.

Use Case 3: Upload Course Materials

➔ **Actors**

- **Primary Actor:** Instructor
- **System:** The platform's course material module

➔ **Preconditions**

- Instructor is logged in and has permissions to upload materials.

➔ **Main Flow**

- Instructor navigates to the course dashboard.
- Instructor selects the "Upload Materials" option.
- Instructor uploads files and adds descriptions if necessary.
- System validates and uploads the files.
- System confirms the upload with a success message, and materials become available to students.

➔ **Alternate Flows**

- **Upload Error:**
    - If the file fails to upload due to size or format issues, the system provides an error message and allows retrying.

➔ **Postconditions**
- Course materials are available for students to view or download.

Use Case 4: Create a Quiz

➔ **Actors**
- **Primary Actor:** Instructor
- **System:** The platform's quiz creation module

➔ **Preconditions**
- Instructor is logged into the platform and has access to the course dashboard.

➔ **Main Flow**
- Instructor navigates to the course dashboard and selects the "Create Quiz" option.
- Instructor enters quiz details.
- System saves the quiz and links it to the course.
- System notifies enrolled students that a new quiz is available.

➔ **Alternate Flows**
- **Incomplete Quiz:**
    - If the instructor exits without saving, the system prompts them to save progress or discard changes.
    - If the instructor enters harmful content, the system removes it.

➔ **Postconditions**
- Quiz is available for students to attempt.

Use Case 5: Download Course Materials

➔ **Actors**
- **Primary Actor:** Student
- **System:** The platform's course material module

➔ **Preconditions**
- Student is logged into the platform and enrolled in a course.

➔ **Main Flow**
- Student navigates to the course dashboard.
- Student browses the list of available course materials.
- Student selects materials to download.
- System processes the download and provides the file to the student.

➔ **Alternate Flows**
- **No Materials Available:**
    - If no materials are uploaded for the course, the system displays a "No materials available" message.

➔ **Postconditions**
- Student downloads and accesses the selected materials offline.

Use Case 6: Participate in Discussion Forums

➔ **Actors**
- **Primary Actor:** User (Instructor or Student)
- **System:** The platform's discussion forum module.

➔ **Preconditions**

- User is logged into the platform and enrolled in a course.

➔ **Main Flow**
- User navigates to the course discussion forum.
- User views or posts a question or topic.
- Other users (instructors or students) respond to the question.
- System notifies the users of new responses or activity.

➔ **Alternate Flows**
- **Inactive Forum:**
    - If no questions are available, the system displays a "No discussions yet" message and encourages users to start one.

➔ **Postconditions**
- Users interact through the forum, clarifying doubts or discussing course-related topics.

Use Case 7: Grade Quizzes

➔ **Actors**
- **Primary Actor:** System

➔ **Preconditions**
- Quiz has been created by instructor and attempted by students.
- Grading rules are defined for the quiz.

➔ **Main Flow**
- Student submits a completed quiz.
- System evaluates the responses against the correct answers.
- System calculates the score based on grading rules.
- System displays the score to the student.

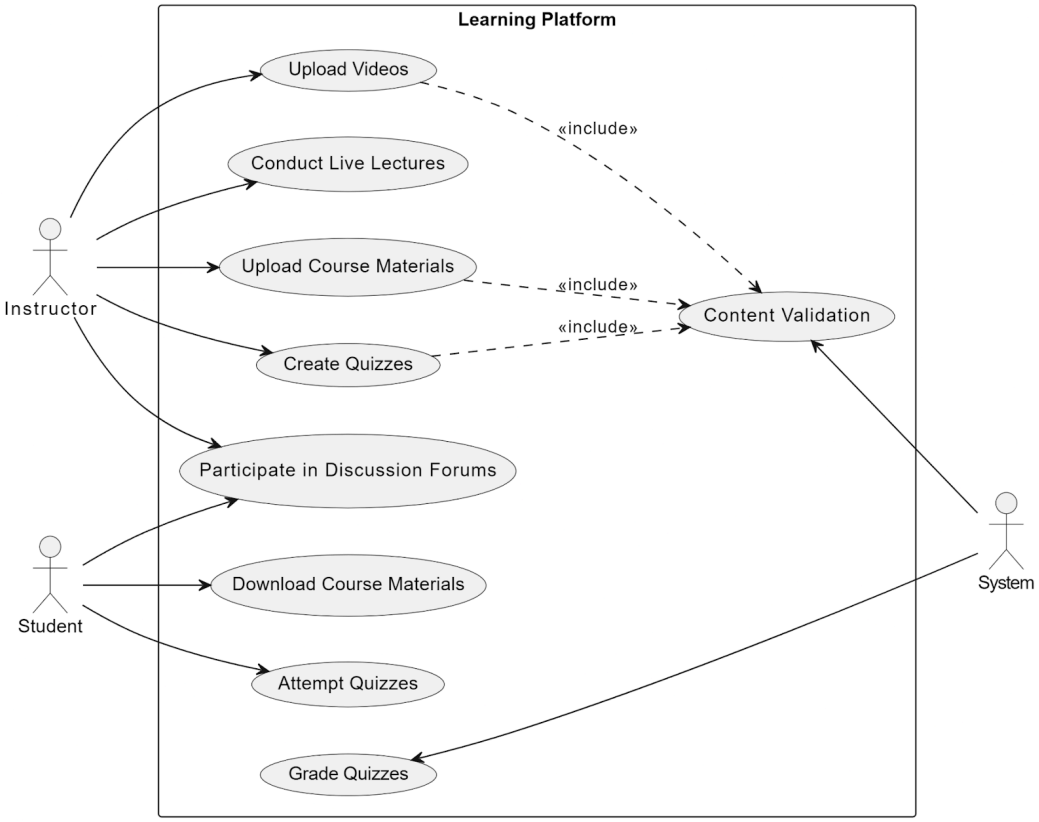
➔ **Alternate Flows**

- **Manual Grading Required:**
    - If the quiz contains subjective questions, the system notifies the instructor to grade manually.

➔ **Postconditions**

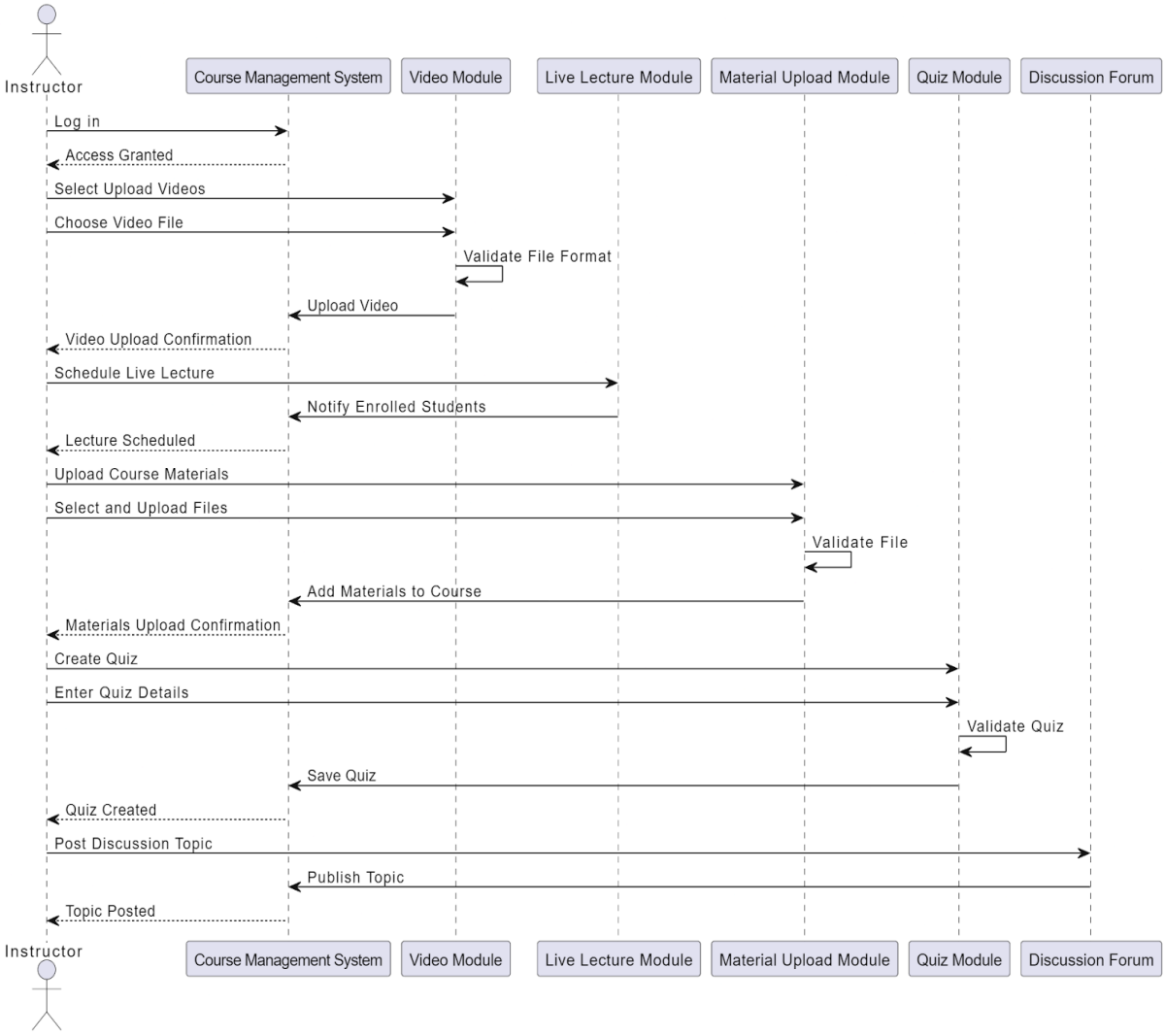
- Quiz results are available for students and instructors to review.

• Use Case Diagram

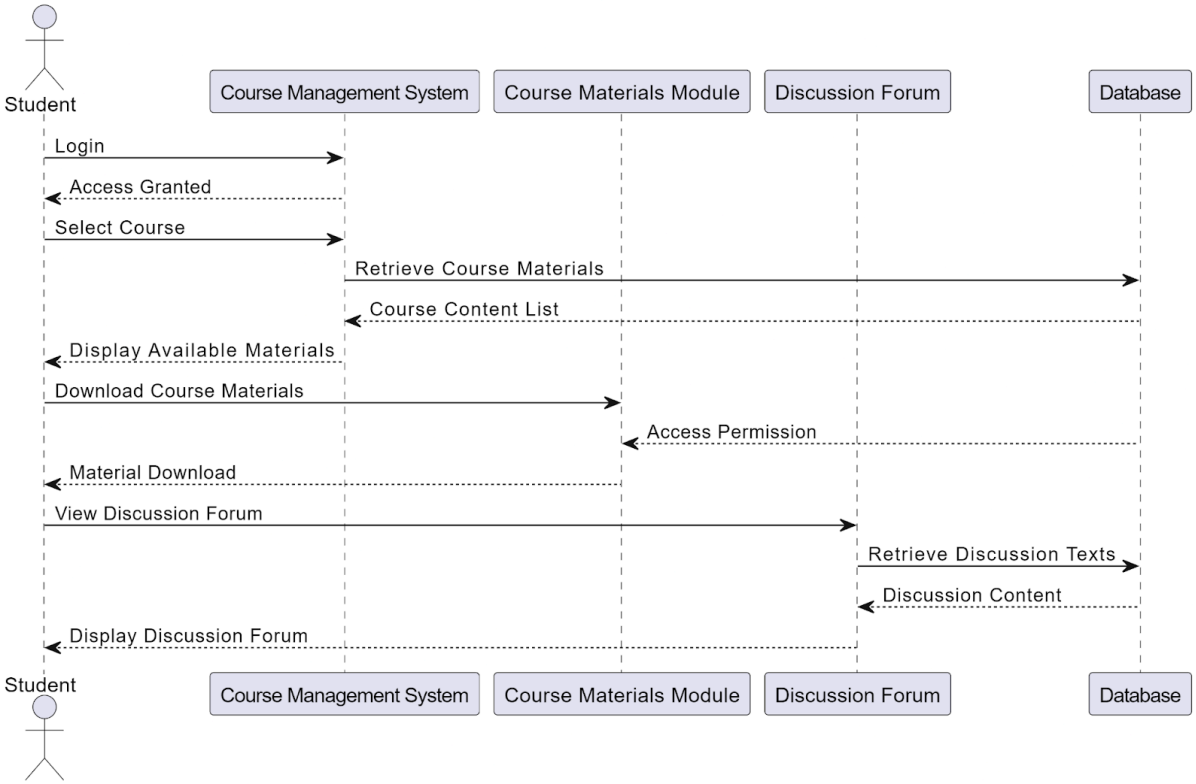


- **Sequence Diagram**

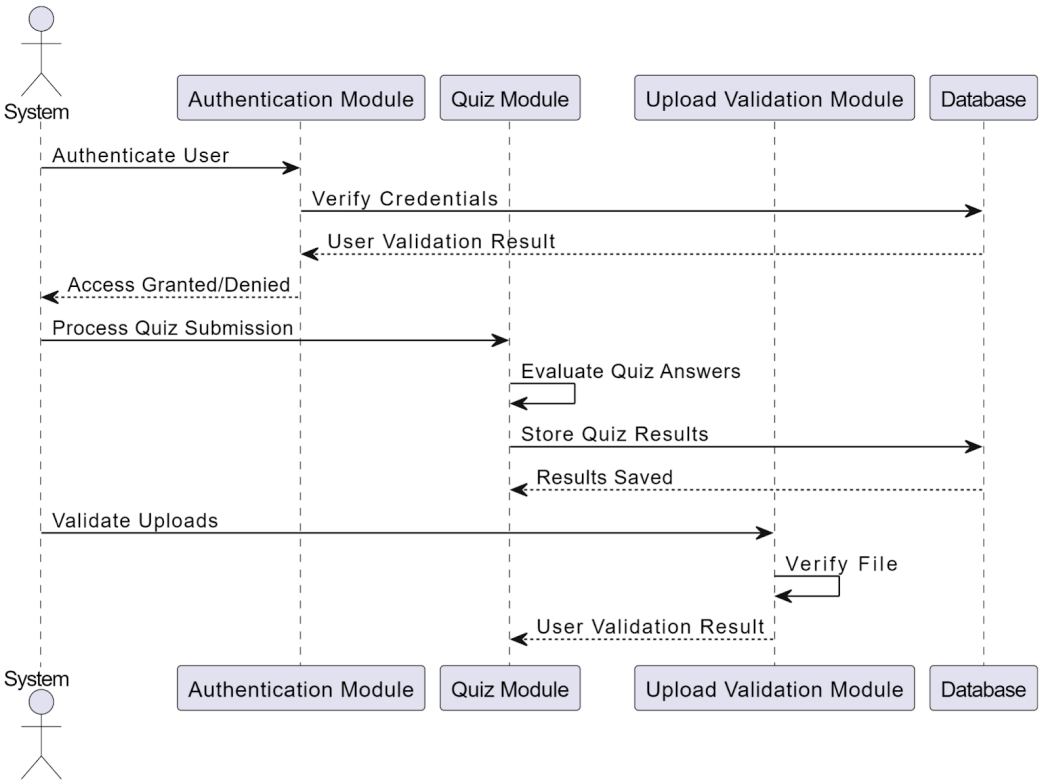
➔ Instructor



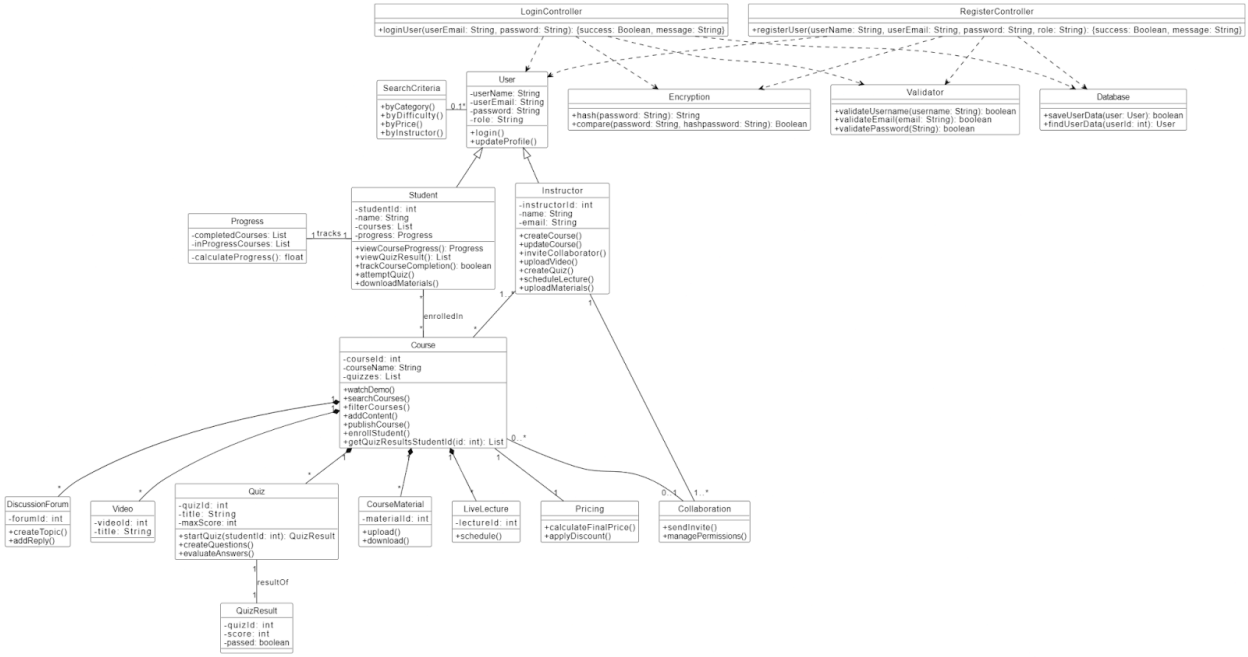
➔ Student


Student
Course Management System
Course Materials Module
Discussion Forum
Database
Login
Access Granted
Select Course
Retrieve Course Materials
Course Content List
Display Available Materials
Download Course Materials
Access Permission
Material Download
View Discussion Forum
Retrieve Discussion Texts
Discussion Content
Display Discussion Forum
Student
Course Management System
Course Materials Module
Discussion Forum
Database

➔ **System**

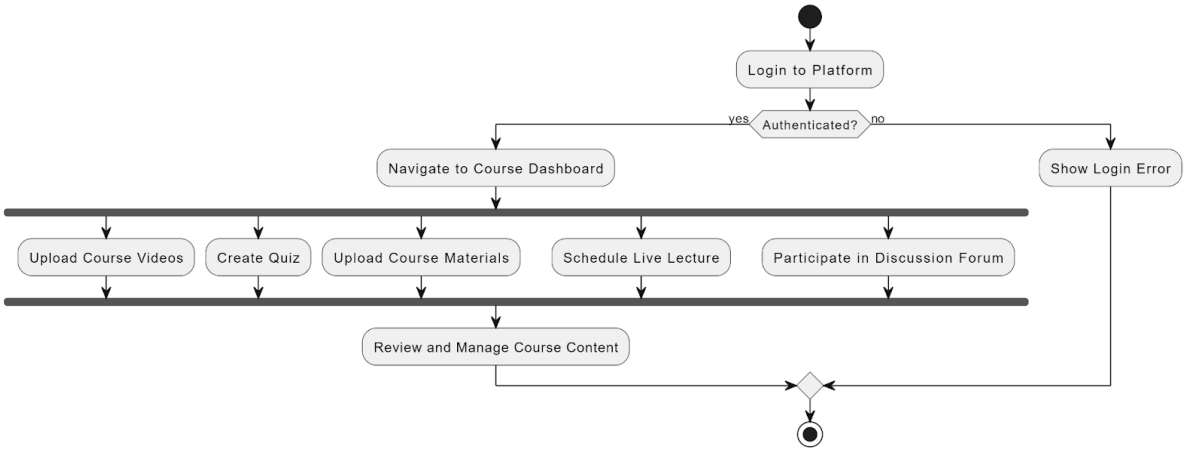

System
Authentication Module
Quiz Module
Upload Validation Module
Database
Authenticate User
Verify Credentials
User Validation Result
Access Granted/Denied
Process Quiz Submission
Evaluate Quiz Answers
Store Quiz Results
Results Saved
Validate Uploads
Verify File
User Validation Result
System
Authentication Module
Quiz Module
Upload Validation Module
Database

- **Class Diagram**

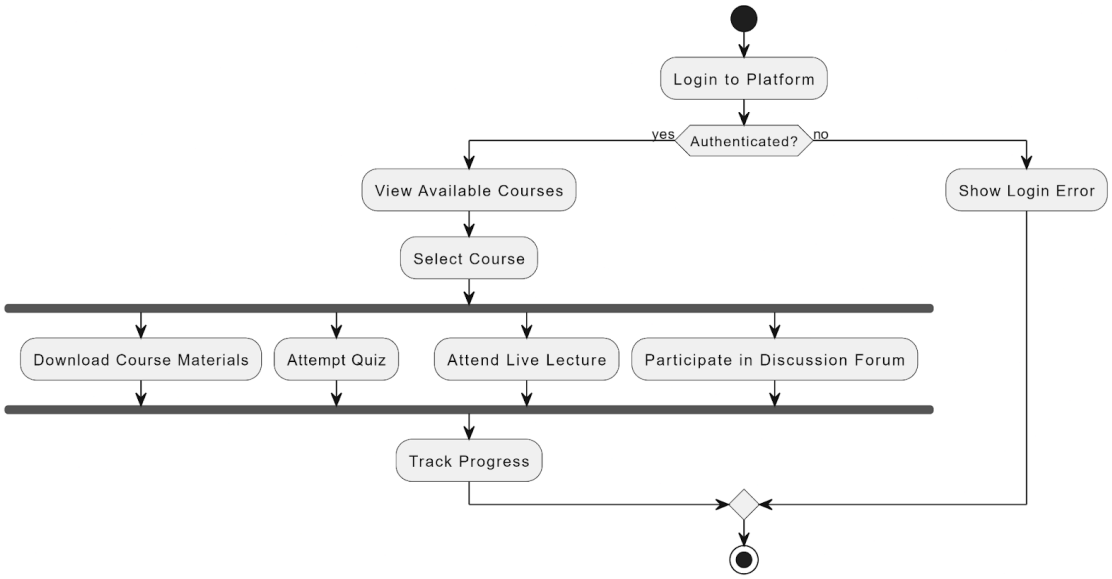


- # Activity Diagram

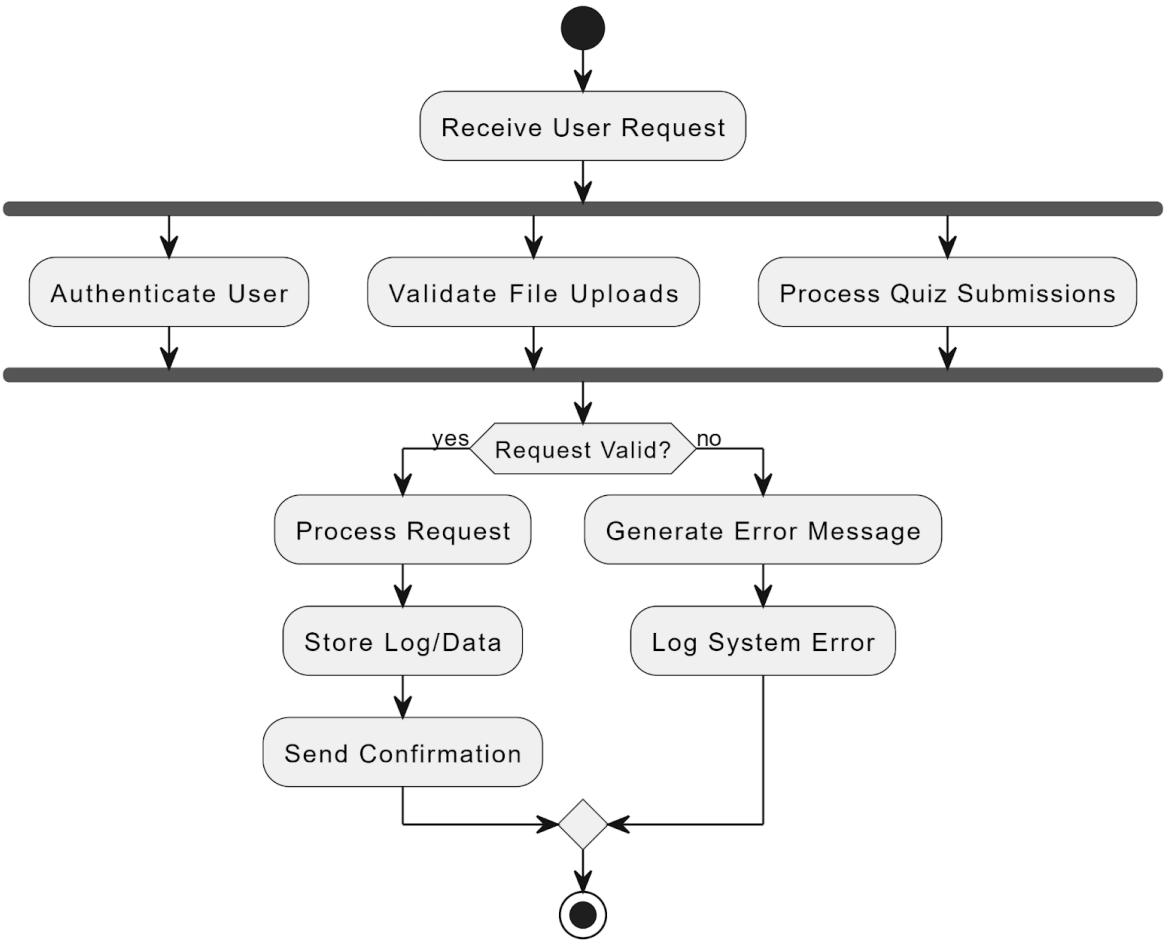
➔ **Instructor**



➔ **Student**

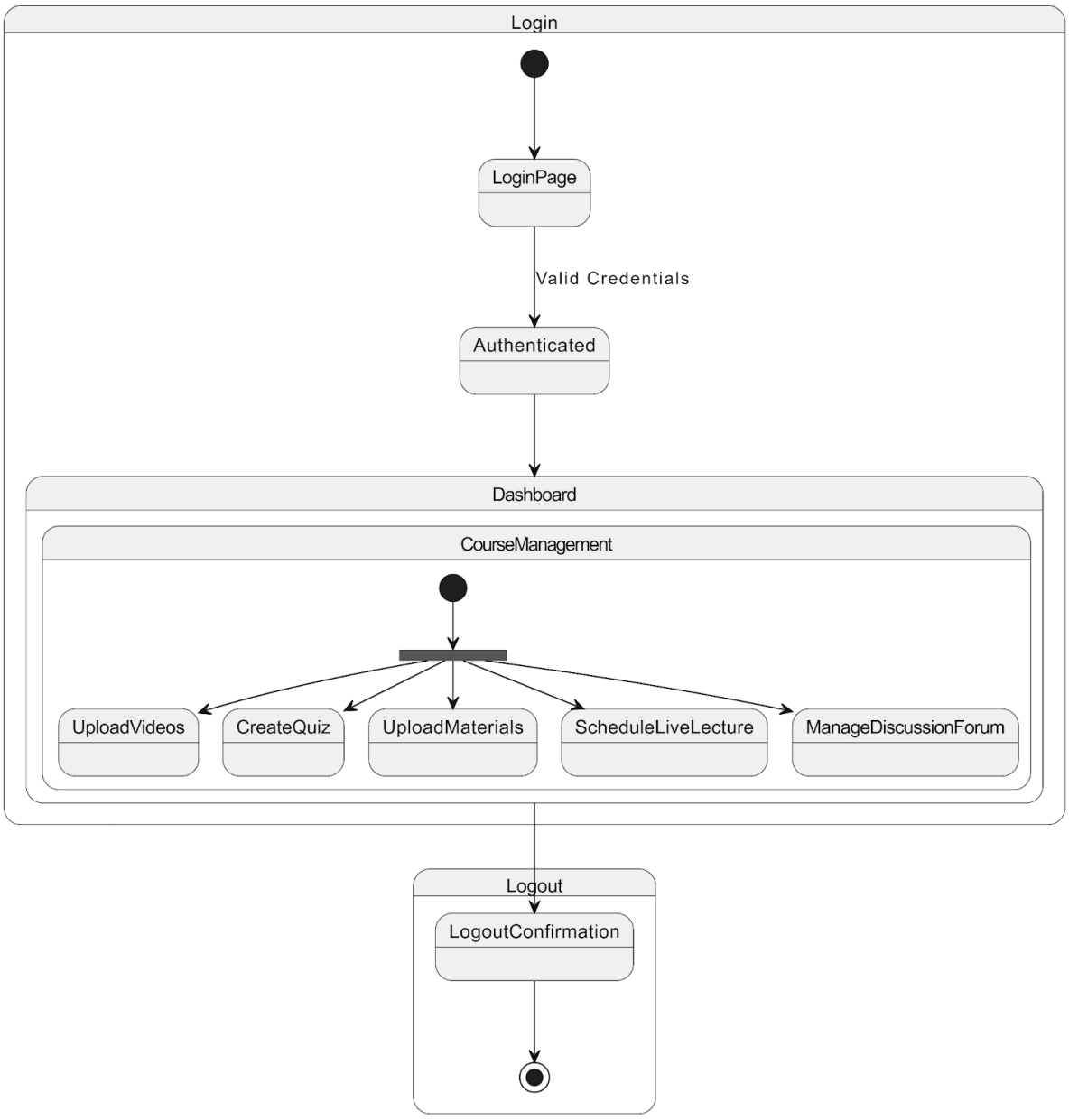


➔ **System**

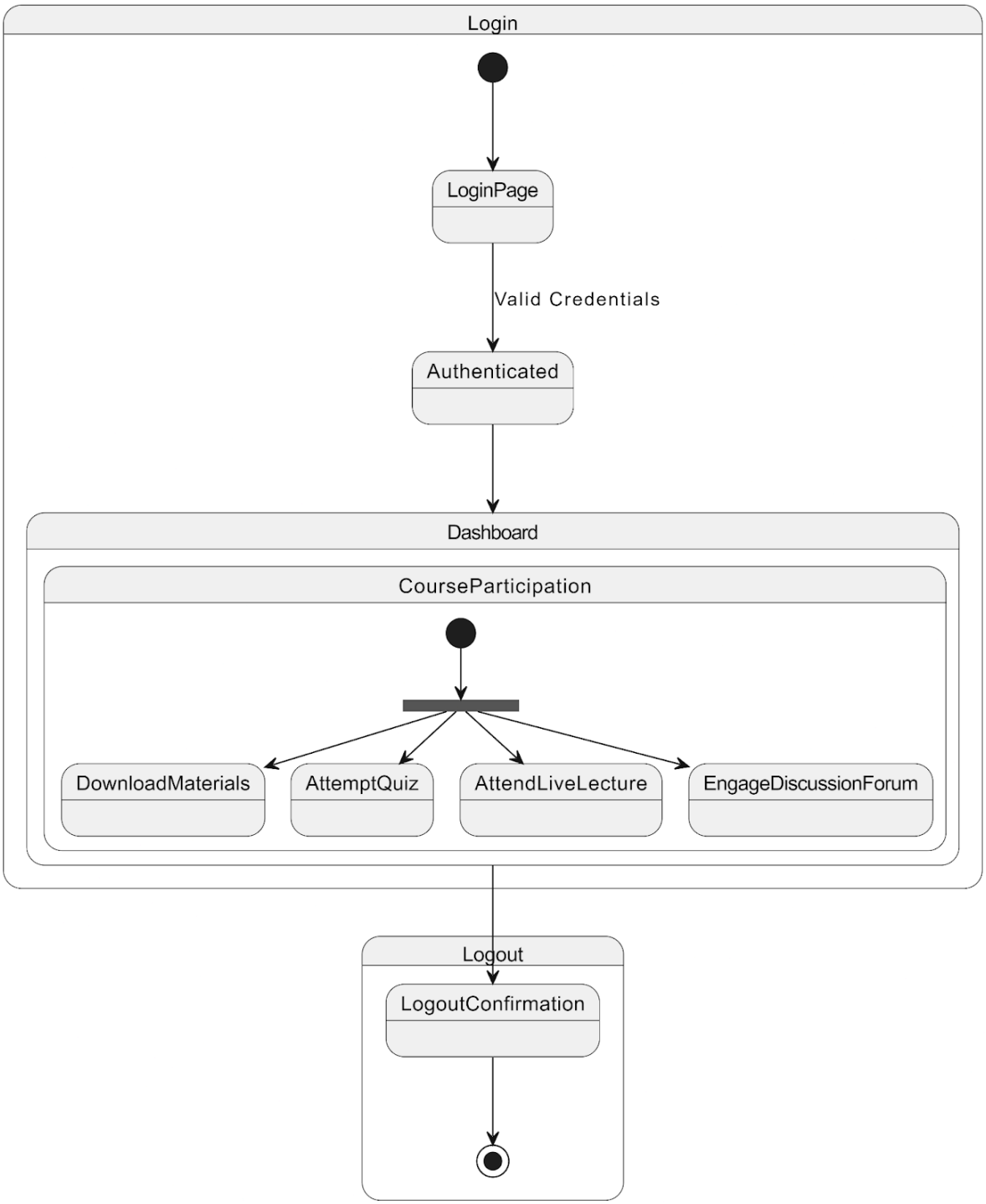

Receive User Request
Authenticate User
Validate File Uploads
Process Quiz Submissions
yes
Request Valid?
no
Process Request
Generate Error Message
Store Log/Data
Log System Error
Send Confirmation

- **State Diagram**

➔ **Instructor**



➔ **Student**


Login
LoginPage
Valid Credentials
Authenticated
Dashboard
CourseParticipation
DownloadMaterials
AttemptQuiz
AttendLiveLecture
EngageDiscussionForum
Logout
LogoutConfirmation

Sprint 5: User Engagement and Support

● User Stories

1. As a student, I want to bookmark a course or course materials, so that I can access them easily.
2. As a student, I want to see my past history, so that I can resume my lecture progress.
3. As a student, I want to view the course leaderboard, so that I can understand my ranking and performance compared to my peers.
4. As a student,I want to review and rate a course after I complete it, So that I can share feedback and help future students make decisions.
5. As a user, I want customer support, so that I can resolve my technical queries with this feature.
6. As a system, I want to send notifications about their courses/enrol courses to users so that they stay updated about course activities.

● Use Cases

Use Case 1: Bookmark Course Materials

Actors
Primary Actor: Student
System: Course Management Module

Preconditions

- The student is logged into the platform.
- The student is enrolled in a course.

Main Flow

1. The student navigates to the course materials section.
2. The student selects the option to bookmark specific material.
3. The system saves the bookmark.

4. The system adds the material to the student's bookmarks list.

Alternate Flows

- **Material Already Bookmarked:**
    - The system removes the bookmark.

Postconditions

- The bookmarked materials are easily accessible from the student dashboard.

Use Case 2: View Course History

Actors
Primary Actor: Student
System: Progress Tracking Module

Preconditions

- The student is logged into the platform.

Main Flow

1. The student navigates to the course history section.
2. The student selects a course to view detailed progress.

Alternate Flows

- **No Course History:**
    - The system displays a "No history available" message.

Postconditions

- The student can view and track their learning progress.

Use Case 3: View Course Leaderboard

Actors
Primary Actor: Student
System: Performance Tracking Module

Preconditions

- The student is logged into the platform.
- The student is enrolled in a course with multiple students.

Main Flow

1. The student navigates to the course leaderboard.
2. The student can view a detailed performance breakdown.

Alternate Flows

- **Insufficient Data:**
    - The system shows limited or no ranking.

Postconditions

- The student understands their performance relative to peers.

Use Case 4: Course Review and Rating

Actors
Primary Actor: Student
System: Feedback Management Module

Preconditions

- The student has completed the course.
- The student is logged into the platform.

Main Flow

1. The system prompts the student to review the course.
2. The student provides rating and feedback
3. The system saves and aggregates the review.
4. The system updates the course rating.

Alternate Flows

- **Student Skips Review:**
    - The student can choose not to provide a review.
- **Validation Error:**
    - The system validates the review content and displays an error for invalid input.

Postconditions

- The course receives an updated rating and feedback.

Use Case 5: Customer Support

Actors
Primary Actor: User
System: Support Query Module

Preconditions

- The user is logged into the platform.

Main Flow

1. The user navigates to the support section.
2. The user creates a query by describing a technical issue.
3. The system logs the query.
4. The support team reviews and responds to the query.

Alternate Flows

- **query Escalation:**
    - The system routes the query to advanced support if required.
- **Unresolved queries:**
    - The system links the query to any previous unresolved queries.

Postconditions

- The support query is tracked and resolved.

Use Case 6: Course Notification System

Actors
Primary Actor: System

Preconditions

- The user is registered on the platform.
- The user has set notification preferences.

Main Flow

1. The system identifies notification triggers.
2. The system generates personalised notifications.
3. The system sends notifications.

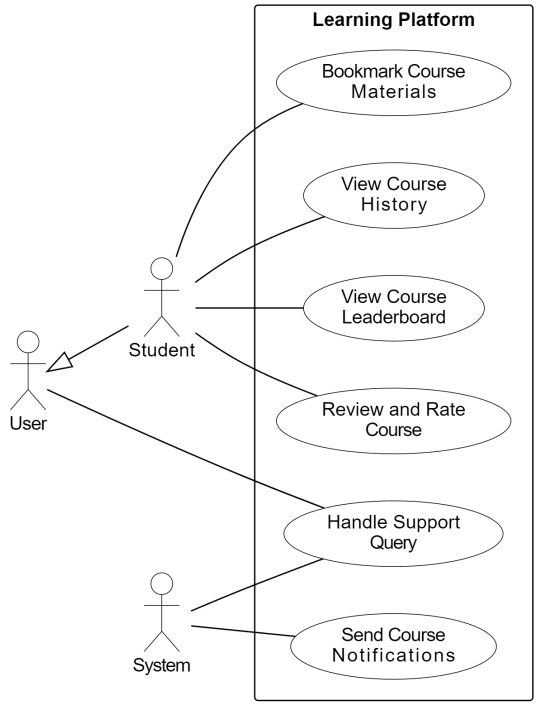
Alternate Flows

- **Notifications Disabled:**
  - The system skips sending notifications.
- **Delivery Failure:**
  - The system logs and retries failed notifications.

Postconditions

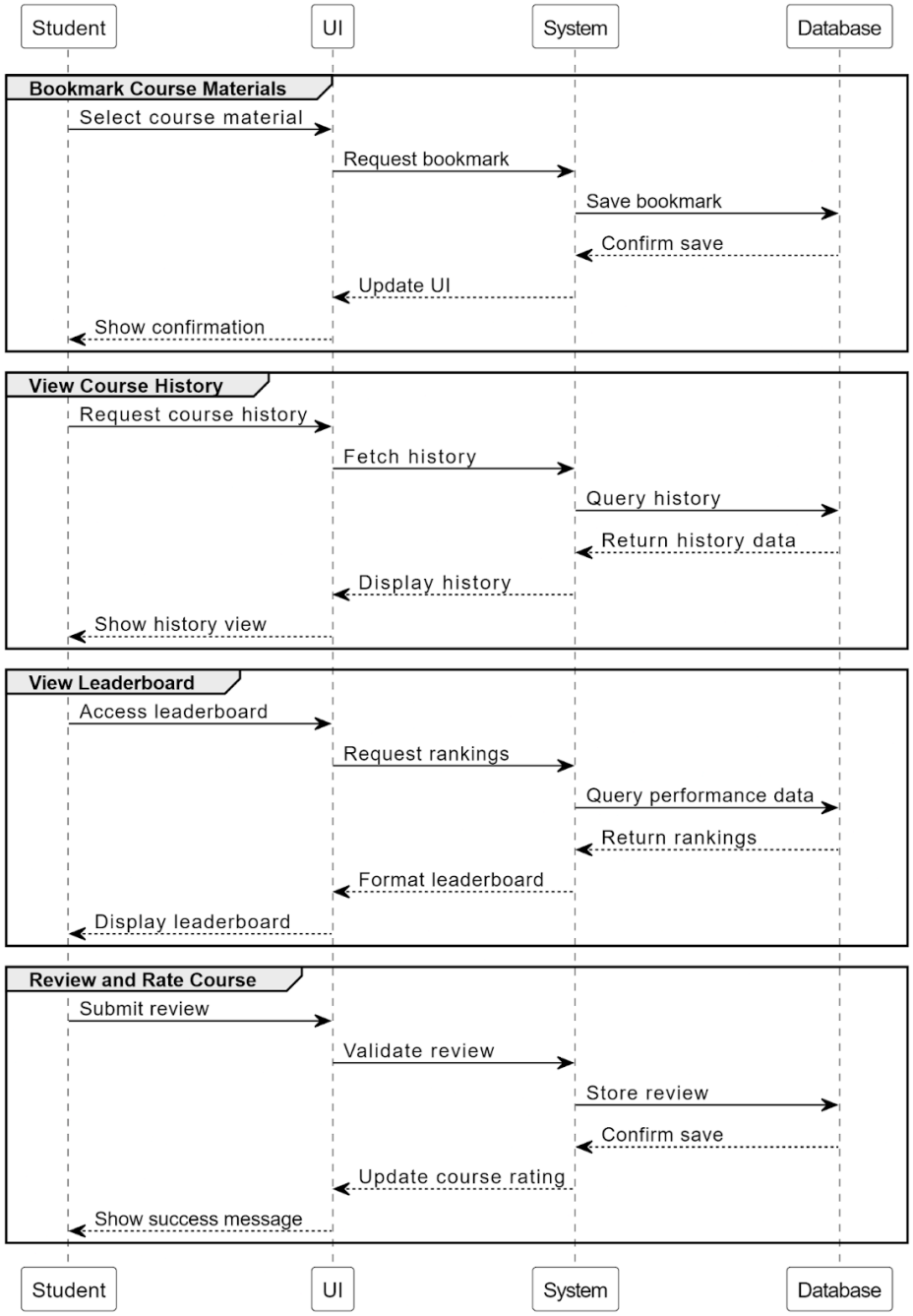
- The user receives timely, relevant course updates.

● Use Case Diagram

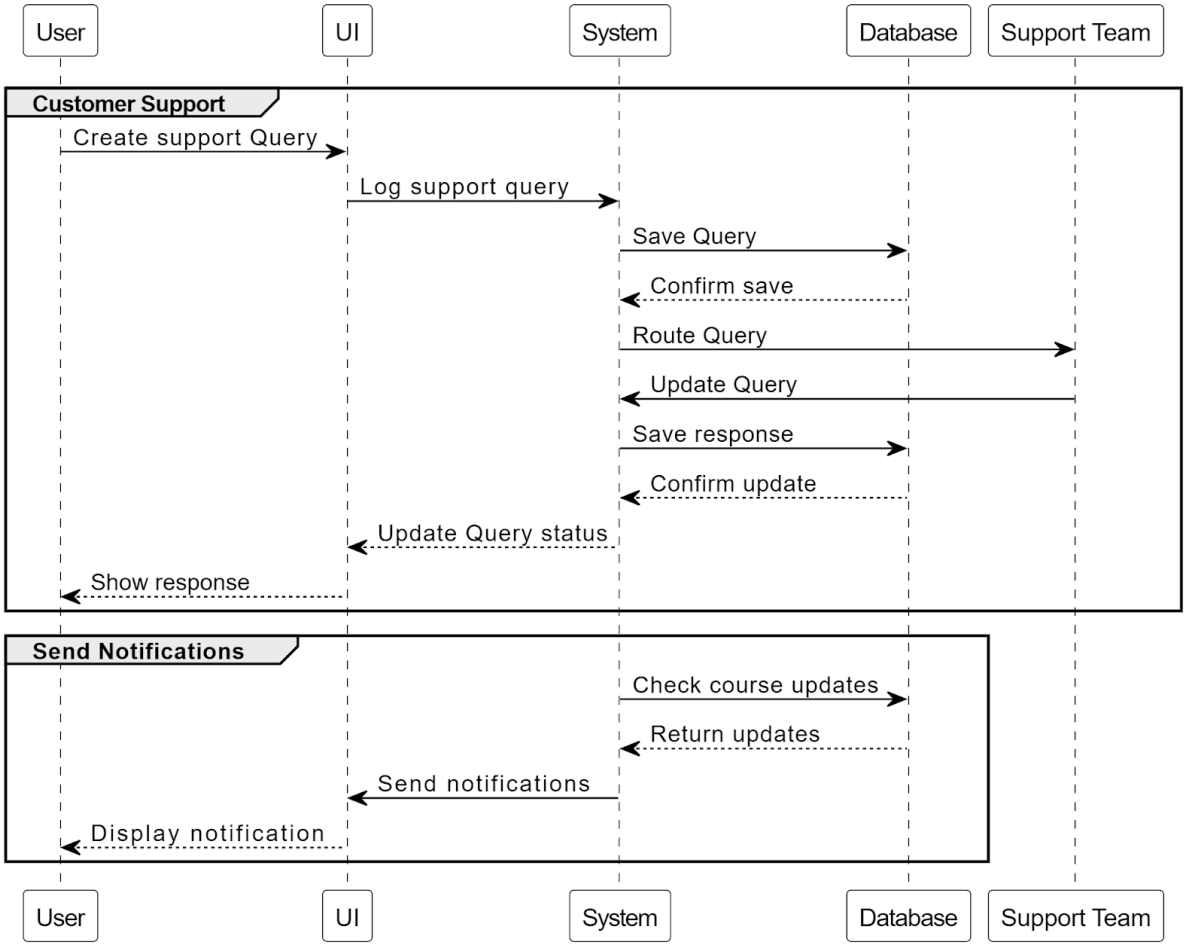


- **Sequence Diagram**

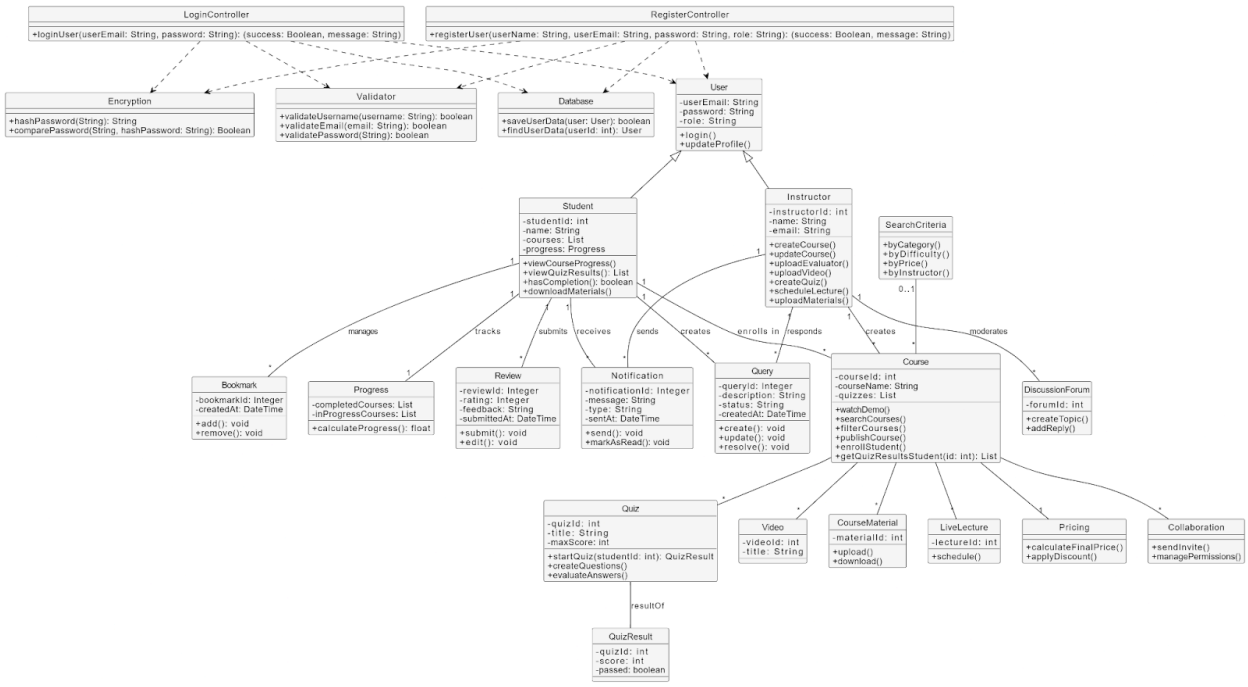
➔ Student



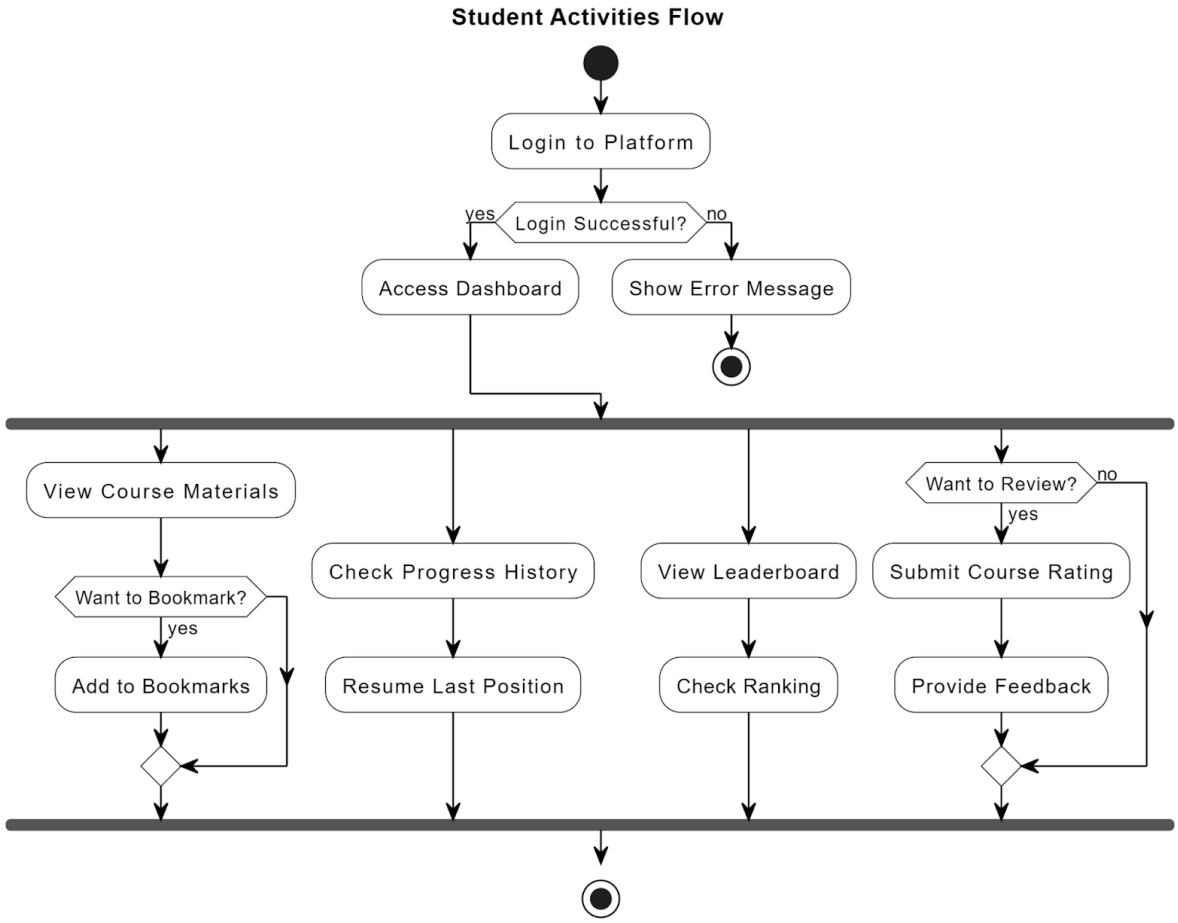
➔ **User**


User
UI
System
Database
Support Team
Customer Support
Create support Query
Log support query
Save Query
Confirm save
Route Query
Update Query
Save response
Confirm update
Update Query status
Show response
Send Notifications
Check course updates
Return updates
Send notifications
Display notification
User
UI
System
Database
Support Team

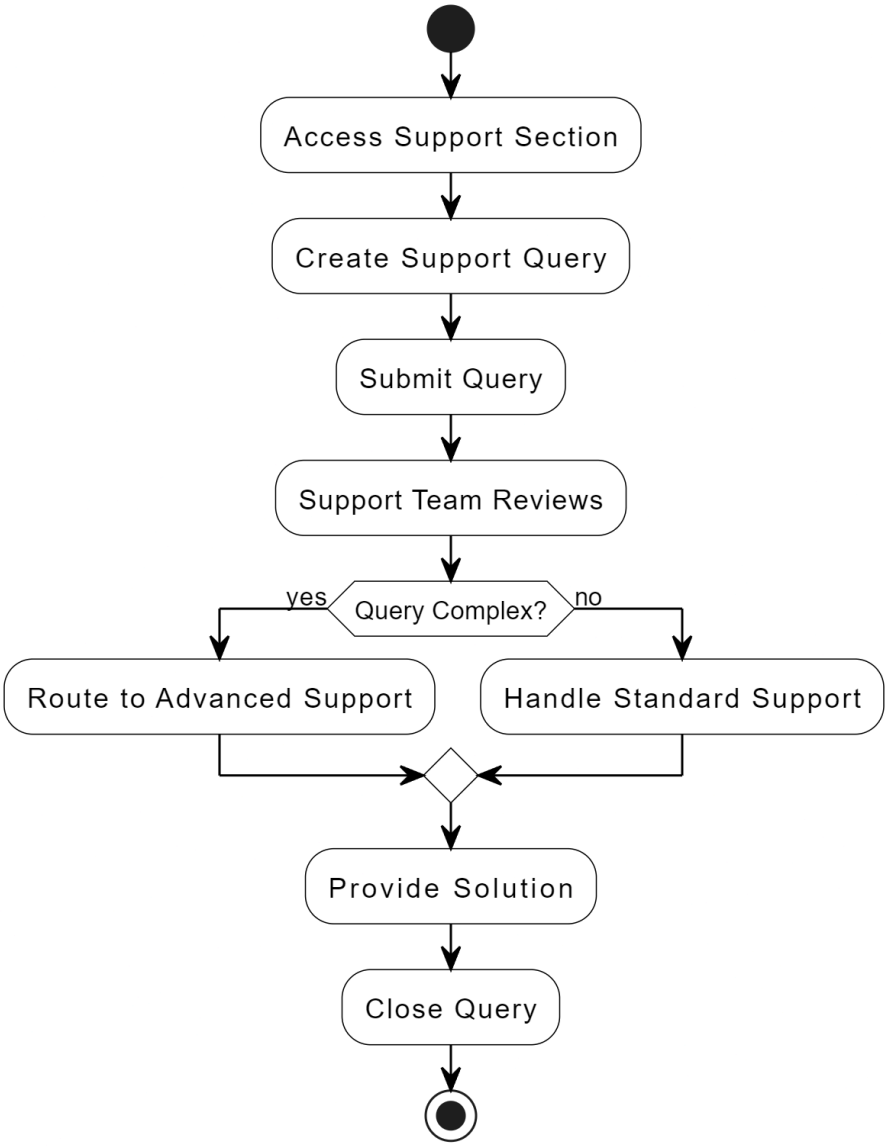
- **Class Diagram**



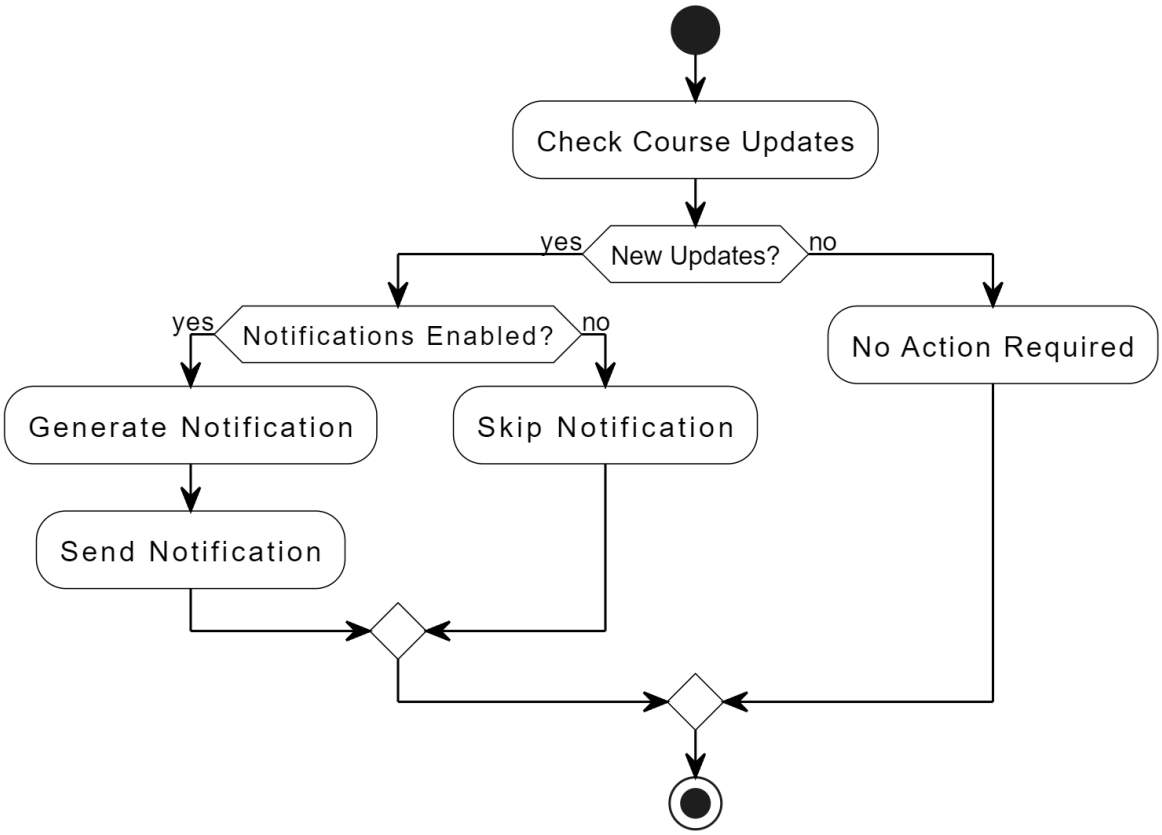
- **Activity Diagram**



➔ **Support Query**

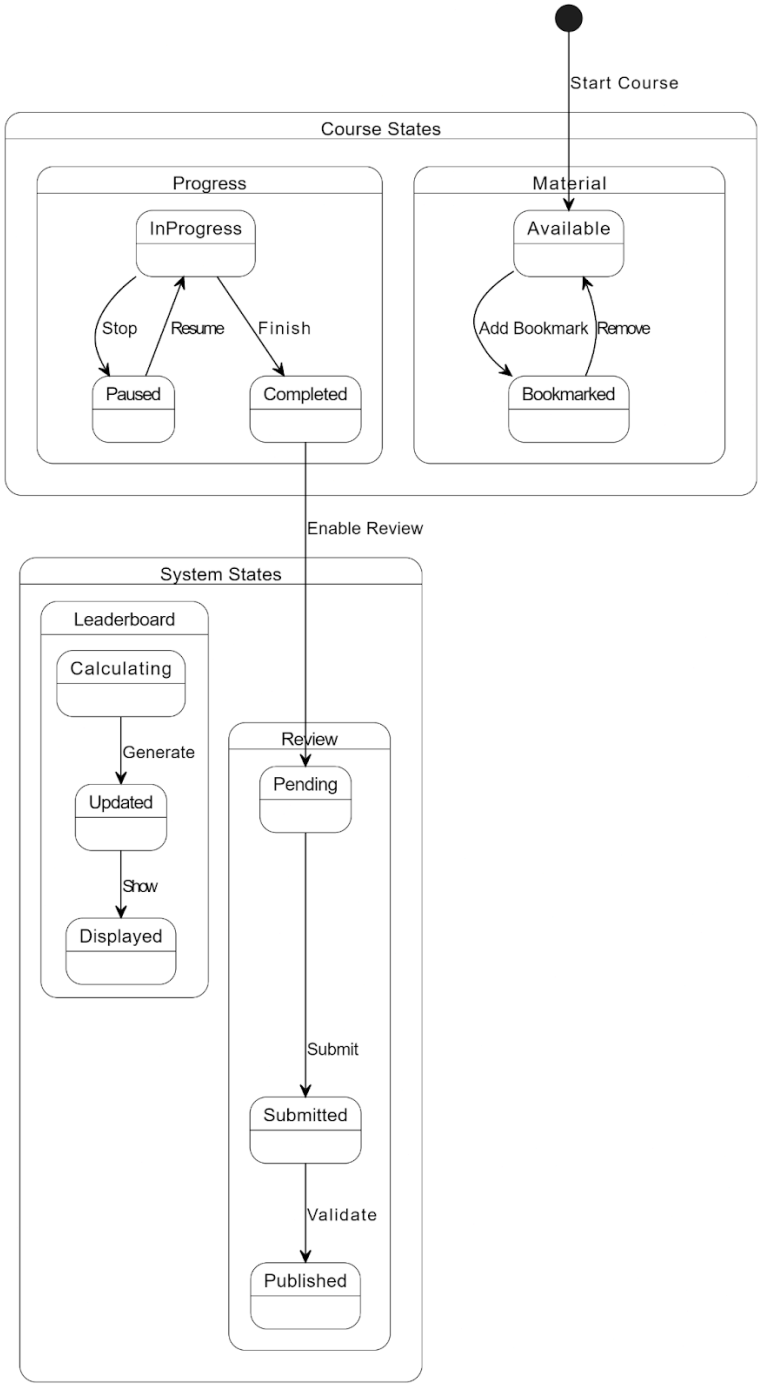


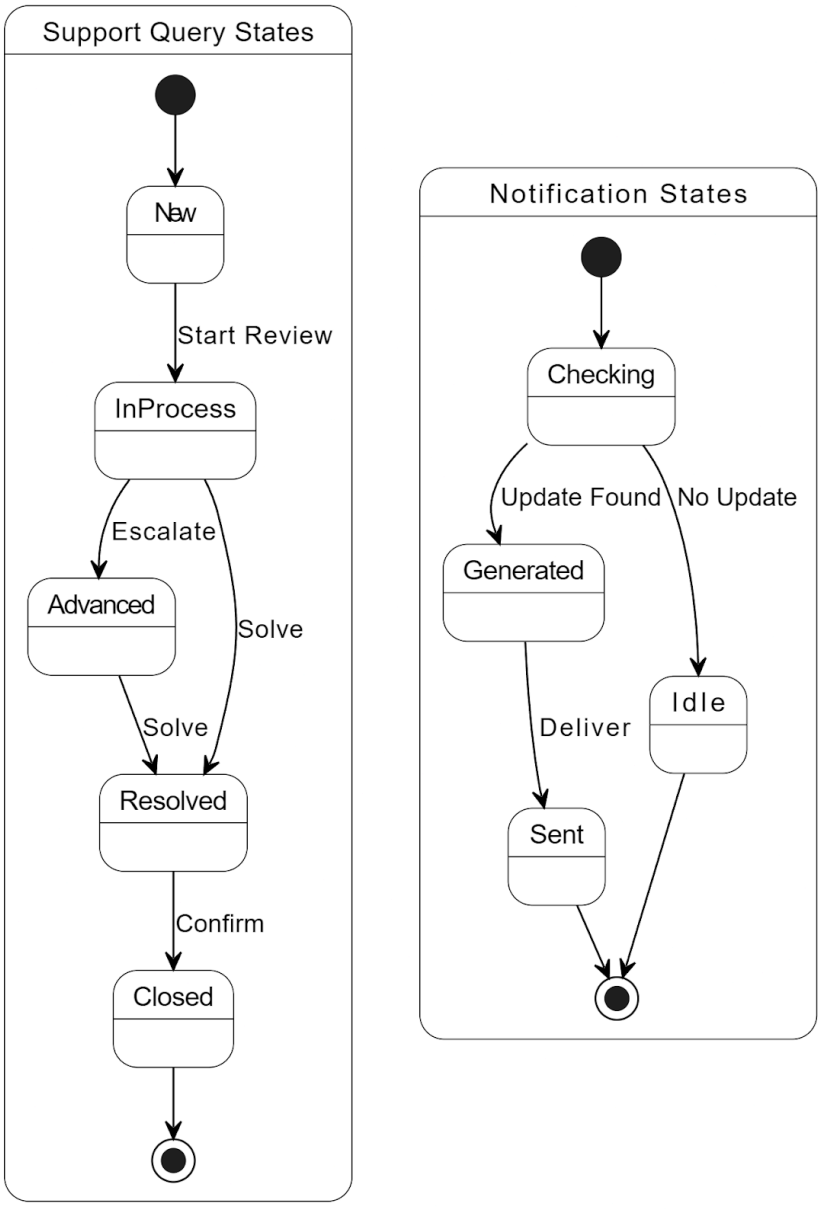
➔ **Support Query**



- **State Diagram**

➔ **Student Activities**




Support Query States
New
Start Review
InProcess
Escalate
Solve
Advanced
Solve
Resolved
Confirm
Closed
Notification States
Checking
Update Found
No Update
Generated
Deliver
Idle
Sent

Sprint 6: Admin Features and Security

• User Stories

1. As a system, I want to automatically generate and send certificates to users who completed a course, so that they can have proof of their course completion.
2. As a system, I want to securely handle the payment process of course purchases, so that users can complete their payment safely.
3. As a system, I can assign badges to the users based on their performances, consistency, and achievements.
4. As an admin I want to verify course content so that no inappropriate content is uploaded.
5. As an admin, I want to be able to receive customer queries so that I can help them to resolve.
6. As an admin, I want to approve or reject course submissions so that only quality and appropriate content is available on the platform.
7. As an admin, I want to handle payments and process refunds so that financial transactions are managed accurately.

• Use Cases

1. Automatically Generate and Send Certificates

➔ **Actor**: System

➔ **Preconditions**:

- User has completed all course requirements.
- The course's certificate template is predefined.

➔ **Main Flow**:

- The system identifies the user who has completed the course.
- The system retrieves the course details and user information.
- The system generates a personalised certificate.

➔ **Postconditions**:
- The user receives the course completion certificate.

➔ **Alternate Flows**:
- **If certificate generation fails**: Log the error and notify the admin.

2. Secure Payment Process

➔ **Actor**: System, User

➔ **Preconditions**:
- The user has selected a course for purchase.
- Payment gateway integration is functional.

➔ **Main Flow**:
- The user selects a course and proceeds to checkout.
- The system directs the user to the secure payment gateway.
- The user inputs payment details and confirms payment.
- The payment gateway verifies the transaction.
- The system records the successful payment and enrols the user in the course.

➔ **Postconditions**:
- The course is accessible to the user upon successful payment.

➔ **Alternate Flows**:
- **If payment fails**: Notify the user and allow them to retry.

3. Assign Badges Based on Achievements

➔ **Actor**: System

➔ **Preconditions**:
- Badge criteria are clearly defined in the system.

➔ **Basic Flow**:
- The system monitors user activity and achievements.
- The system checks if the user satisfies any badge criteria.
- If criteria are met, the system assigns the badge to the user profile.
- The user is notified about the badge assignment

.

➔ **Postconditions**:
- The badge is visible on the user's profile.

➔ **Alternate Flows**:
- **If badge assignment fails**: Log the issue for further investigation.

4. Verify Course Content

➔ **Actor**: Admin

➔ **Preconditions**:
- The course content is pending verification.

➔ **Basic Flow**:
- The admin reviews the uploaded course content.
- The admin checks for inappropriate or low-quality material.
- The admin approves or rejects the content.
- The system notifies the course creator of the decision.

➔ **Postconditions**:
- Approved content becomes visible on the platform.
- Rejected content is returned to the creator for corrections.

5. Handle Customer Queries

➔ **Actor**: Admin

➔ **Preconditions**:
- A query is submitted through the platform's support channel.

➔ **Basic Flow**:
- The system routes the query to the admin.
- The admin reviews the query details.
- The admin responds to the customer or resolves the issue.
- The system updates the query status (resolved/pending).

➔ **Postconditions**:
- The customer receives assistance.

➔ **Alternate Flows**:
- **If query requires escalation**: Admin forwards the issue to higher support tiers.

6. Approve or Reject Course Submissions

➔ **Actor**: Admin

➔ **Preconditions**:
- The course submission is pending review.

➔ **Basic Flow**:
- The admin reviews the course submission.

- The admin approves or rejects the course.
- The creator is notified of the decision.

➔ **Postconditions**:
- Approved course submissions are published on the platform.
- Rejected course submissions remain unpublished.

7. Handle Payments and Refunds

➔ **Actor**: Admin

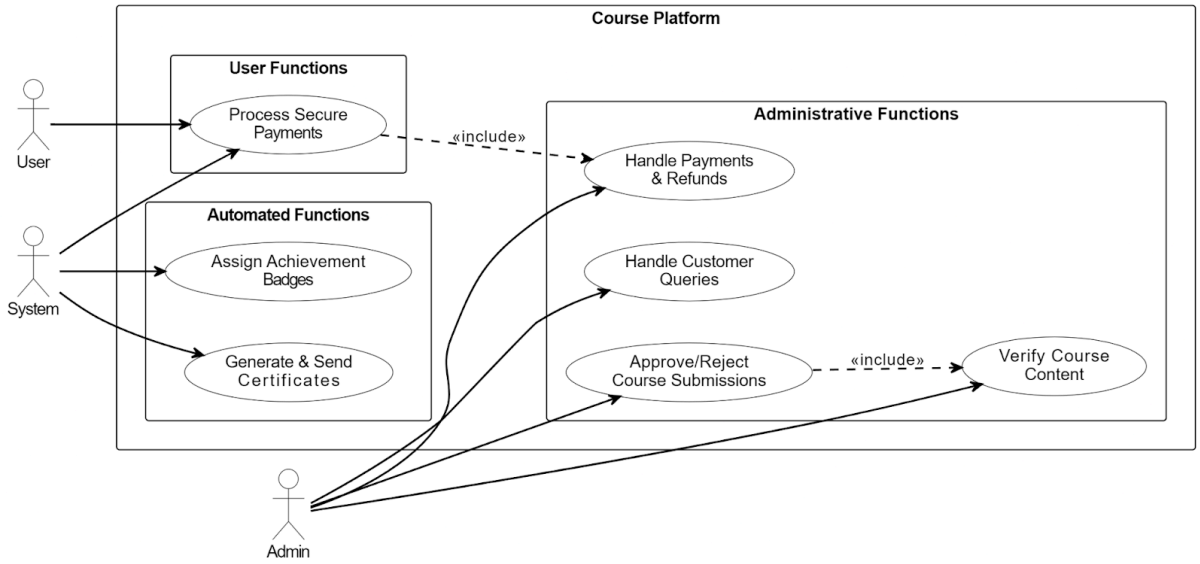
➔ **Preconditions**:
- The payment or refund request is valid.

➔ **Basic Flow**:
- The admin reviews payment details or refund requests.
- For refunds, the admin verifies the request validity.
- The admin processes the payment via the payment gateway.
- The system notifies the user about the transaction status.

➔ **Postconditions**:
- Payments and refunds are recorded and processed accurately and securely.

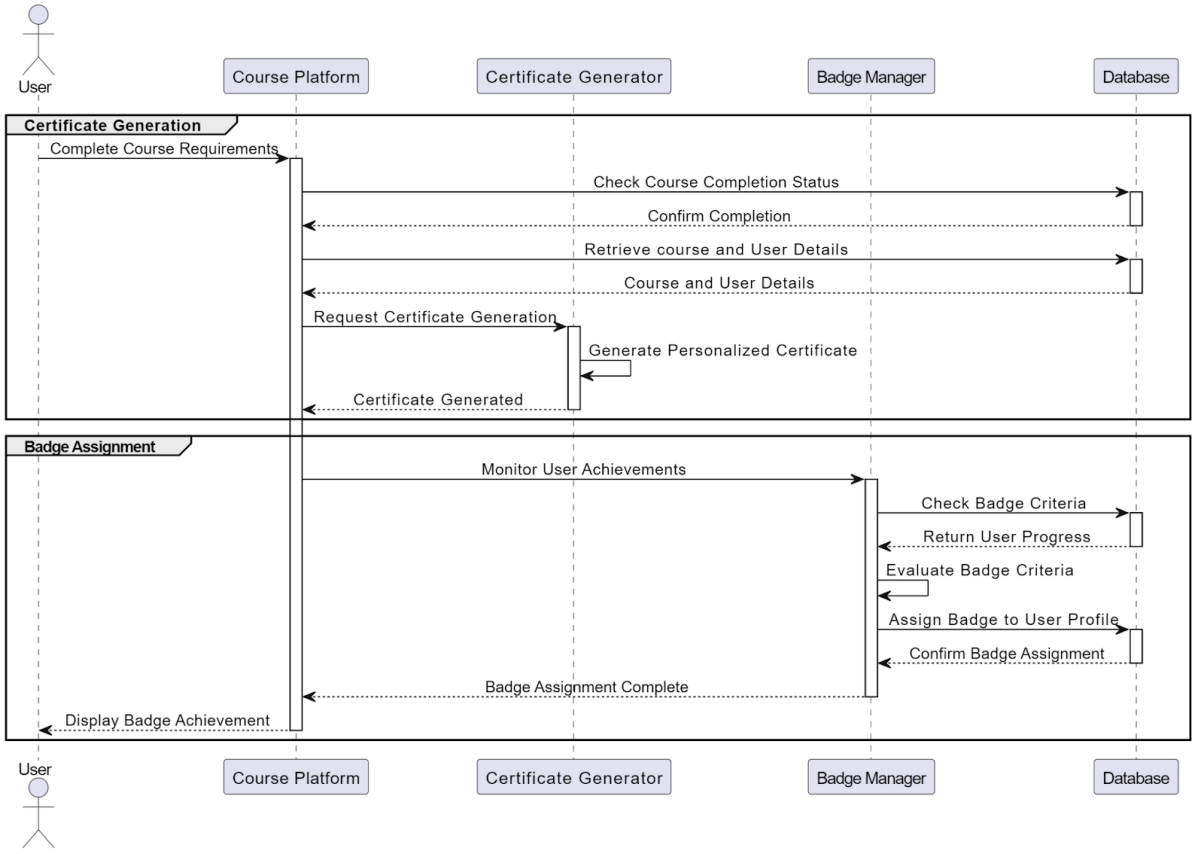
➔ **Alternate Flows**:
- **If payment or refund fails**: Notify the admin for manual resolution.

- **Use Case Diagram**

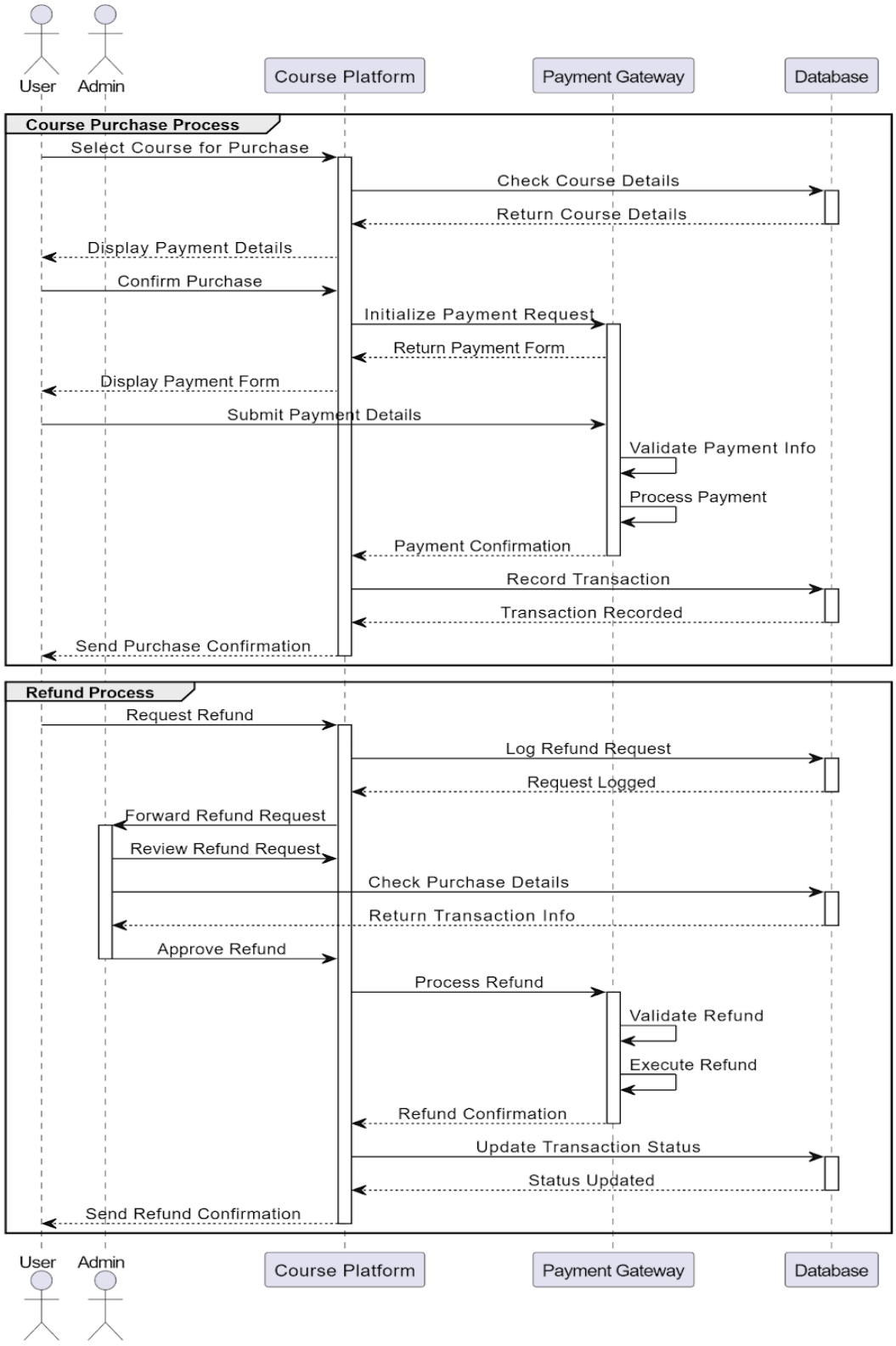


- # Sequence Diagram

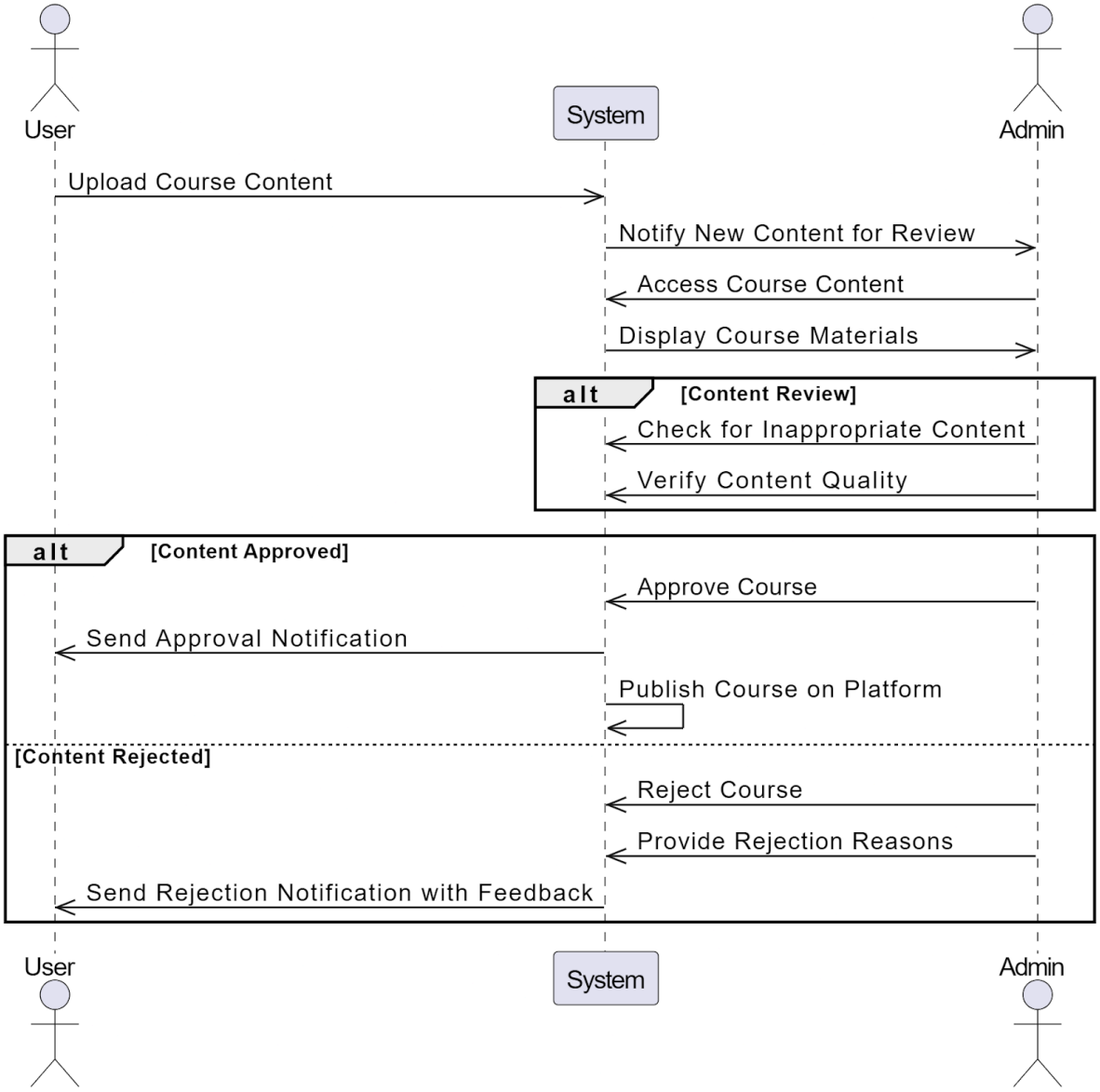
➔ Certificate and Badges



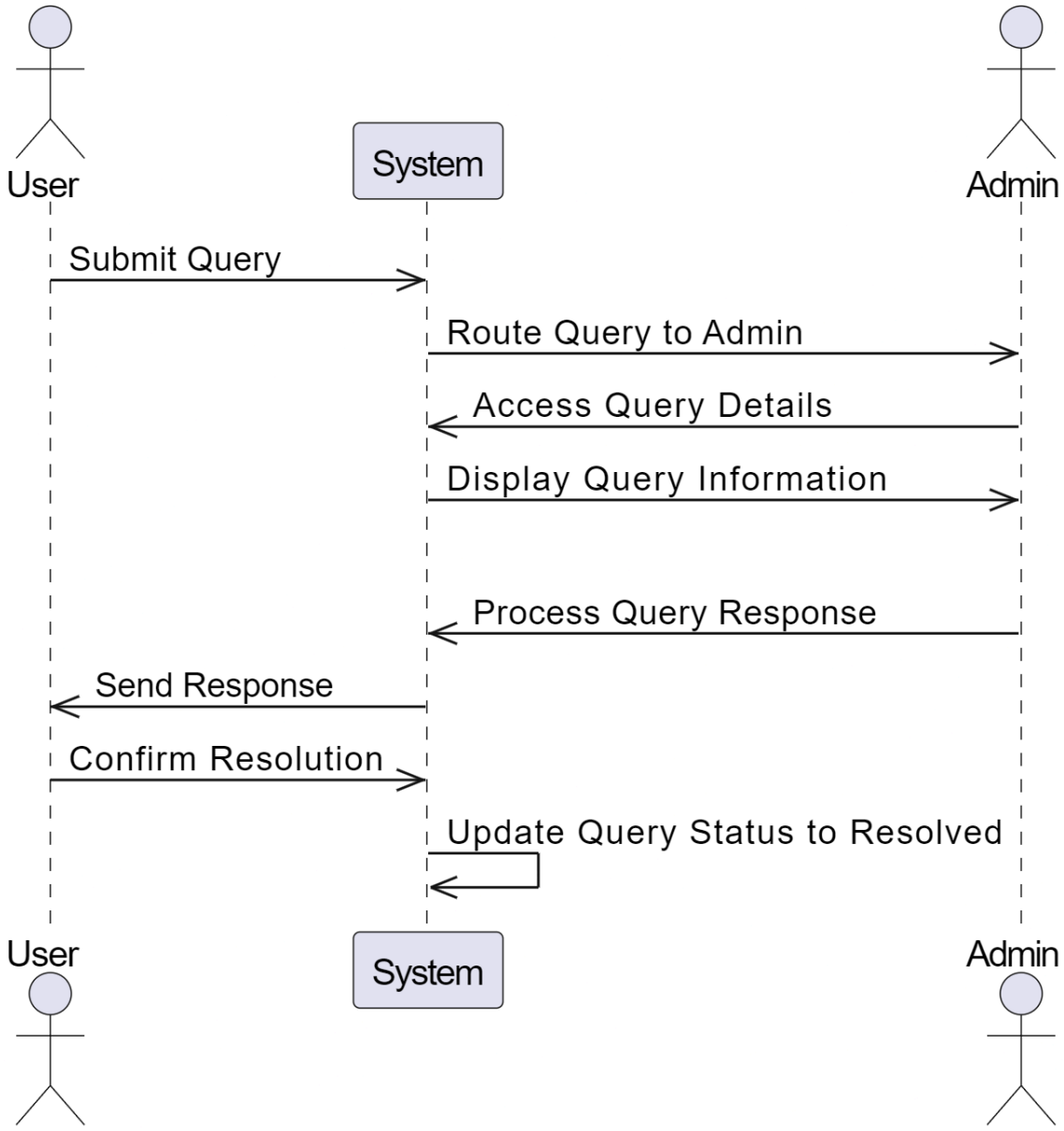
➔ Secure Payment and refund



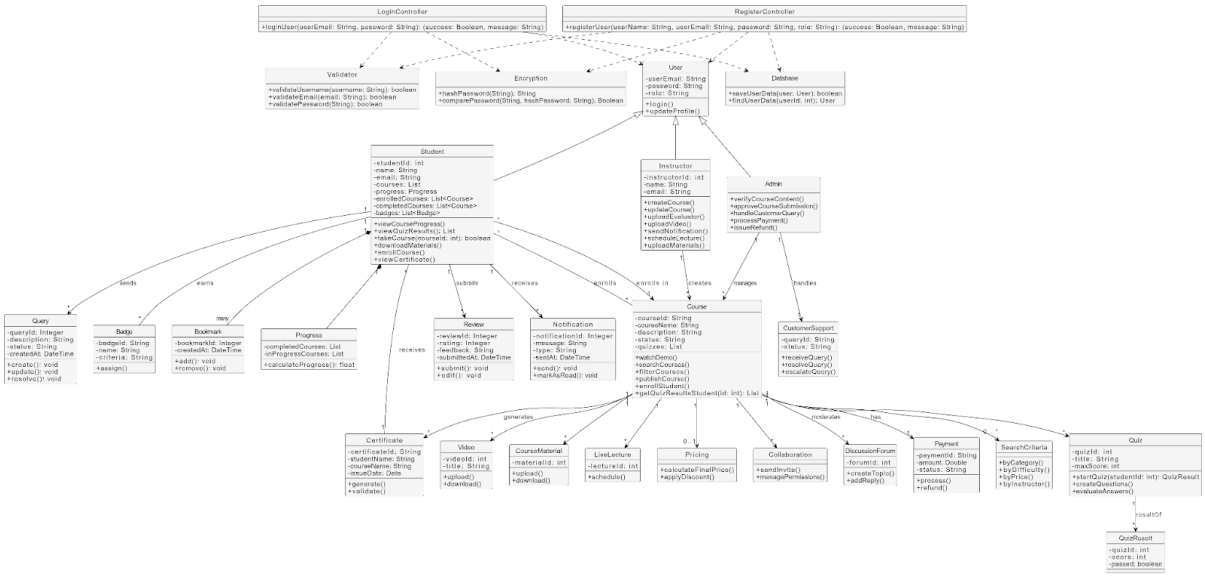
➔ **Verify Course Content**



➔ **Queries**

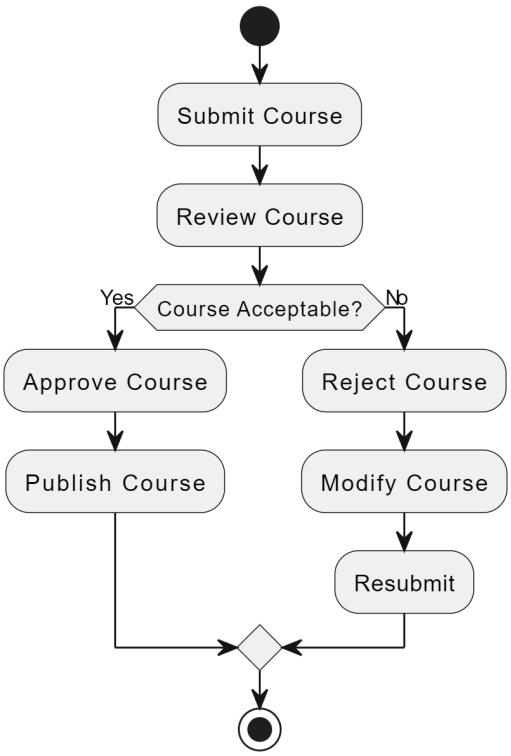

User
System
Admin
Submit Query
Route Query to Admin
Access Query Details
Display Query Information
Process Query Response
Send Response
Confirm Resolution
Update Query Status to Resolved
User
System
Admin

- **Class Diagram**

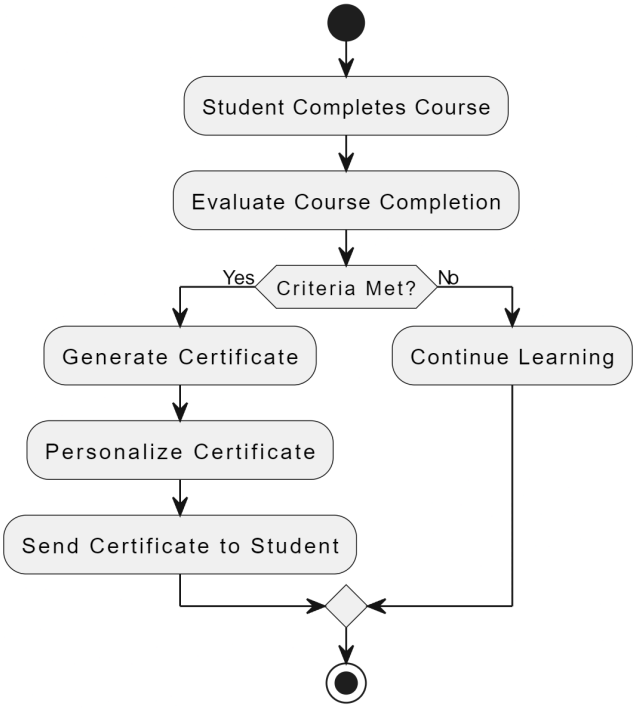


- **Activity Diagram**

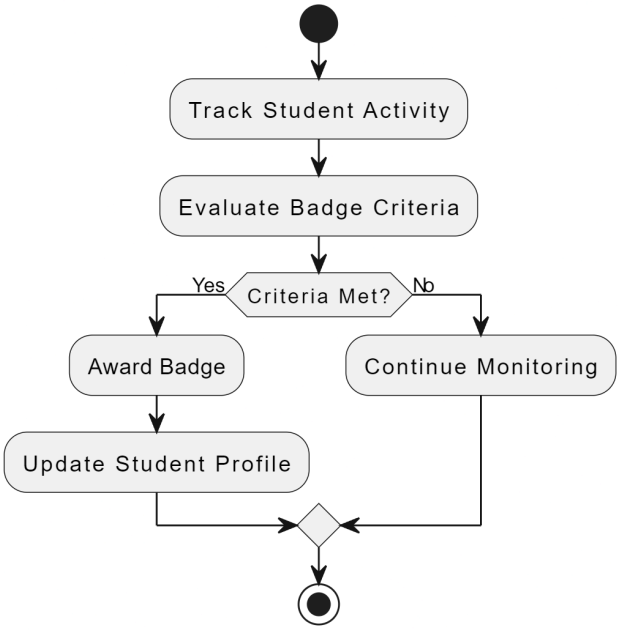
➔ **Verifying course content**



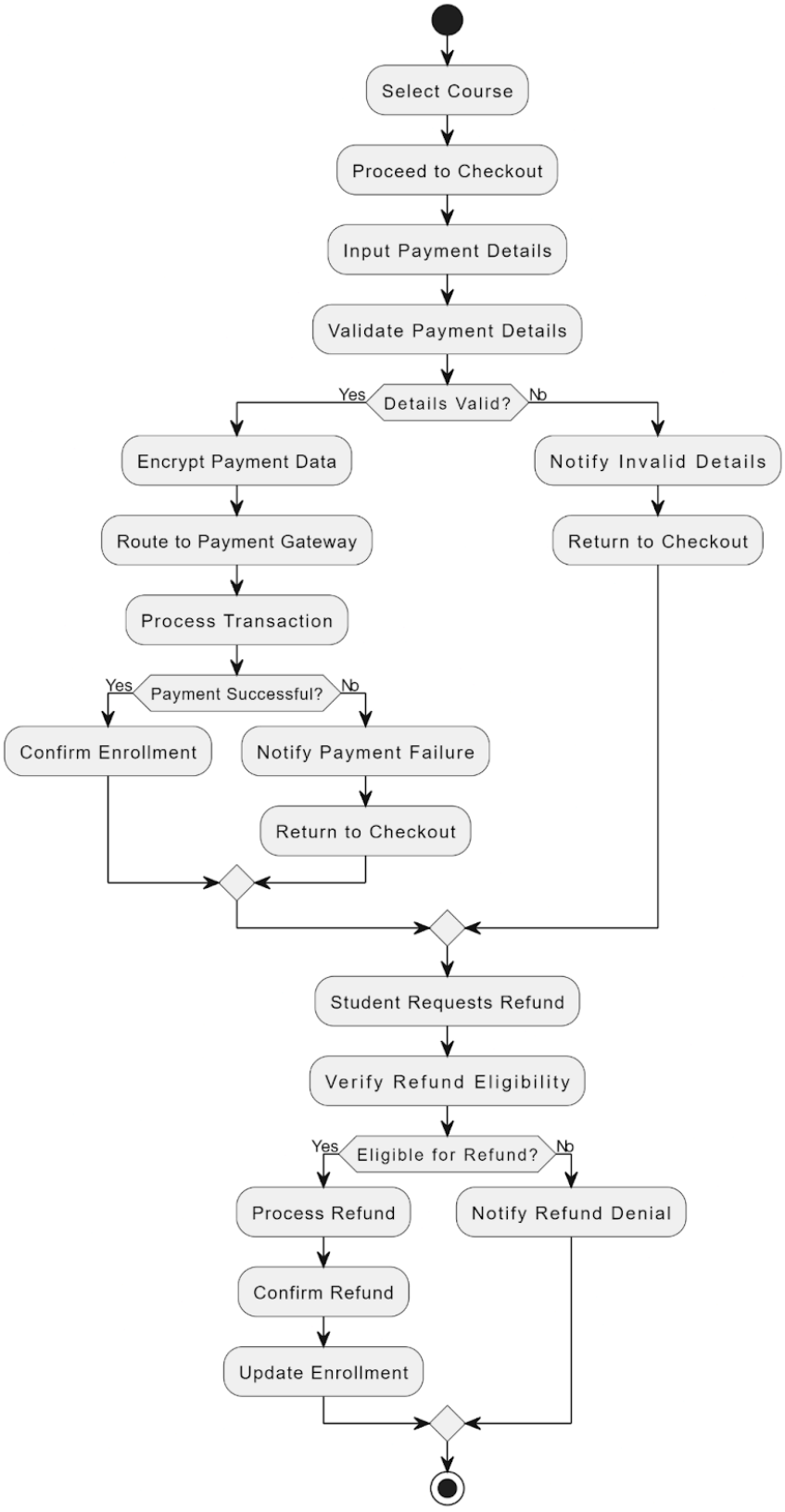
➔ Assigning Course Certificates



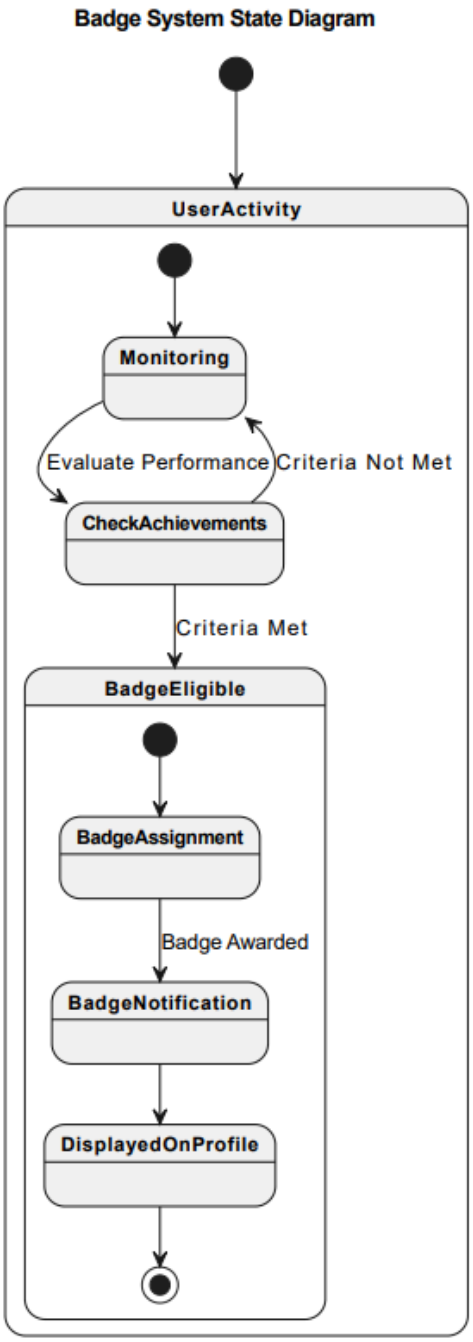
➔ Assigning Badges

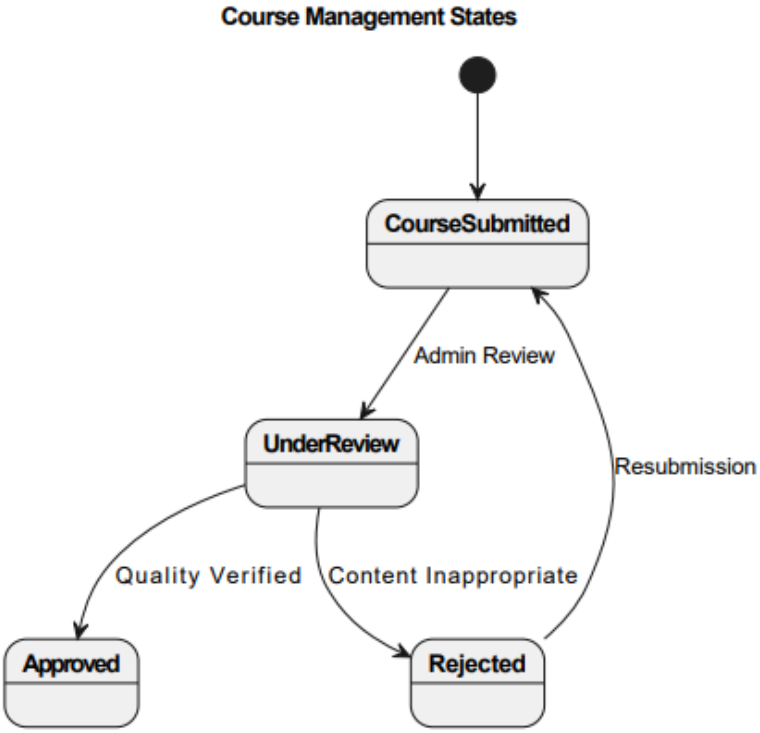


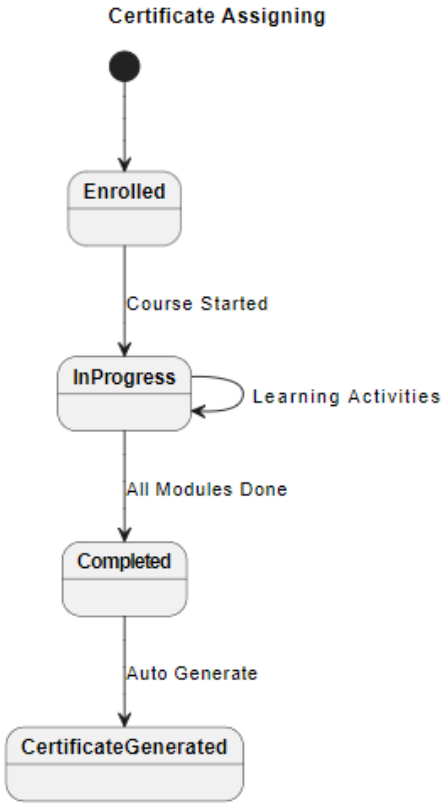
➔ **Secure Payment and refund**

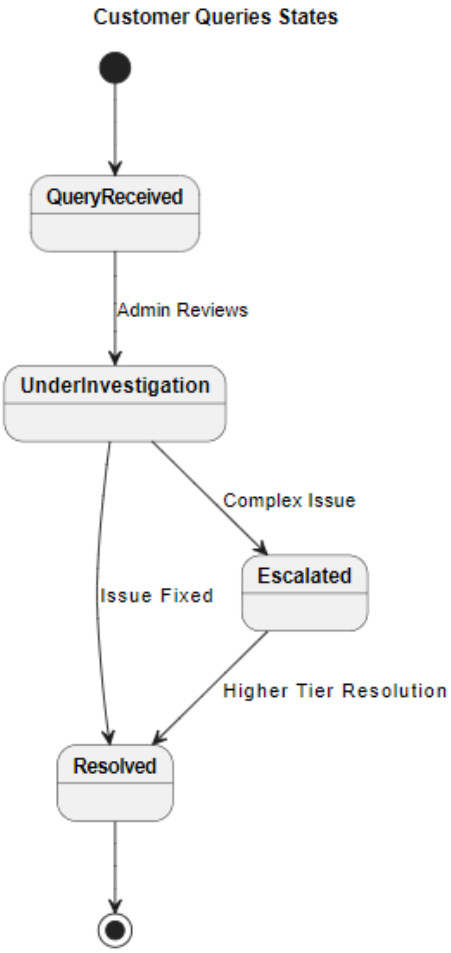


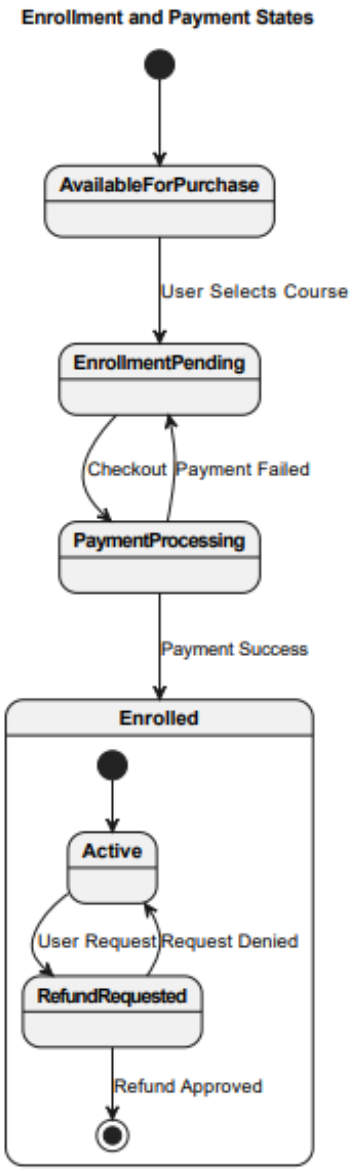
- **State Diagram**




Course Management States
CourseSubmitted
Admin Review
UnderReview
Resubmission
Quality Verified
Content Inappropriate
Approved
Rejected


Certificate Assigning
Enrolled
Course Started
InProgress
Learning Activities
All Modules Done
Completed
Auto Generate
CertificateGenerated


Customer Queries States
QueryReceived
Admin Reviews
UnderInvestigation
Complex Issue
Escalated
Issue Fixed
Higher Tier Resolution
Resolved


Enrollment and Payment States
AvailableForPurchase
User Selects Course
EnrollmentPending
Checkout
Payment Failed
PaymentProcessing
Payment Success
Enrolled
Active
User Request
Request Denied
RefundRequested
Refund Approved