Sprint 3

Course Feedback Web Guide

Spring 2024

Group #3

Gharam Mansour, Daniel Shin, Kevin Nguyen, Javaris Johnson, Eriic Graham, & Justin Khor

Section 1:

Assignee Name	Email	Task	Duration (hours)	Dependency	Due Date	Evaluation
Gharam Mansour	mansourgharam@gmail.com	Section 1, Section 2, Section 8, Section 9, Section 10,	7 hours	Planning/Scheduling Table, Implementation, Test Cases, Use Cases, 2nd Diagram for Uml in Section 8	Monday, March 18th. before class @ 5:30 pm	100%
Javaris Johnson (Coordinator)	johnson.javaris@gmail.com	Section 9, Section 10, & Submission		Implementation, Test Cases	Monday, March 18th. before class @ 5:30 pm	100%
Eric Howard-Graham	ehowgra@gmail.com	Section 6, Section 8		Database Tables	Monday, March 18th. before class @ 5:30 pm	100%
Daniel Shin	danielshiny15@gmail.com	Section 9, Section 10		Implementation, Test Cases	Monday, March 18th. before class @ 5:30 pm	100%
Justin Khor	Justinkhor113@gmail.com	Section 3, Section 4, Section 5, Section 7,		Context Diagrams, Activity Diagrams,	Monday, March 18th. before class	100%

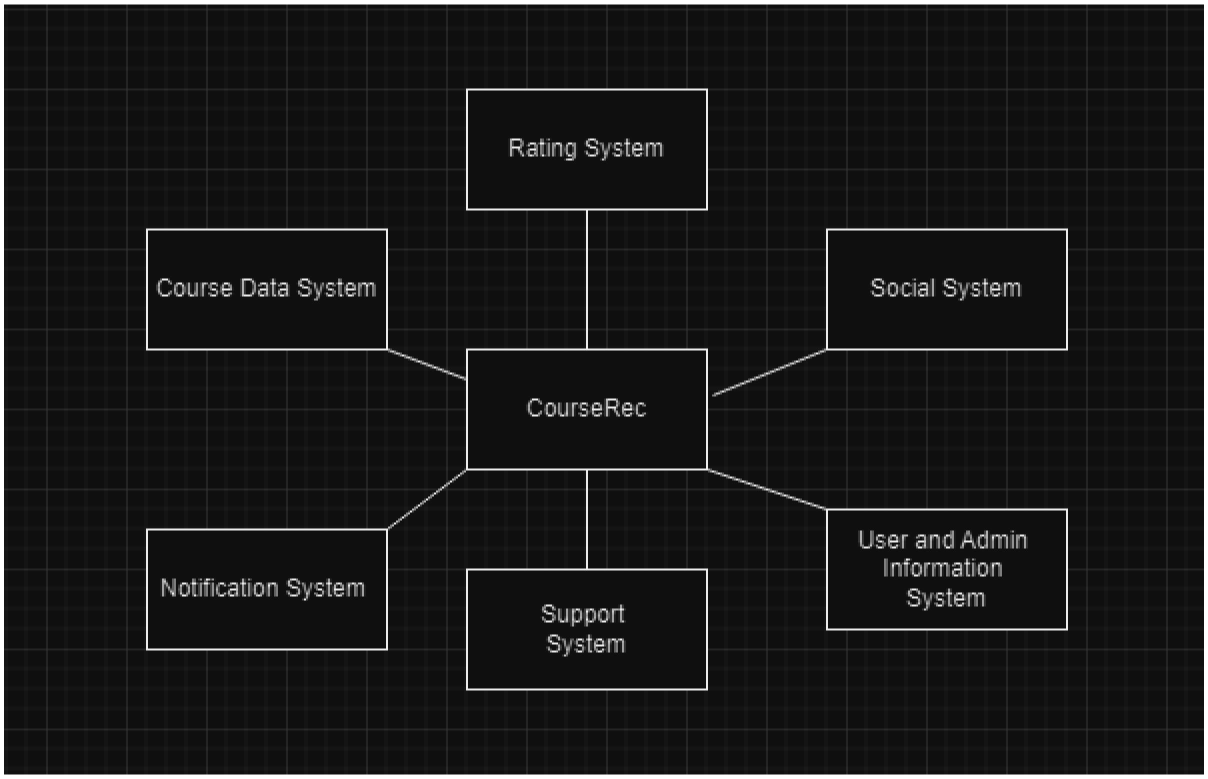
		Section 12		Use Cases, Behavioral Modeling, Github Updates	@ 5:30 pm	
Kevin Nguyen	nkevin46309@gmail.com	Section 3, Section 4, Section 5, Section 7, Section 12		Activity Diagrams, Use Cases, Behavioral Modeling, Github Updates	Monday, March 18th. before class @ 5:30 pm	100%

Section 2:

The product is a Course Feedback system designed to provide comprehensive reviews of courses. It caters to students, faculty, and universities, aiming to address the problem of accessing detailed evaluations of courses before enrollment. This database will contain student or anonymous user evaluations, offering insights into course content, learning styles, difficulty levels, and pacing. Alternatives like university websites and RateMyProfessor exist, but the proposed system stands out by offering organized course reviews with Likert Scale ratings and open-ended feedback sections. Courses will be categorized by major, prerequisites, and teaching styles, with options for anonymous or user-revealed identities. Admin powers will enable management of comments and database control. The system differentiates itself by incorporating social media-like features, allowing users to connect, message, and share information on courses. A user survey will capture interests and learning styles upon registration. Technically, the project utilizes SQL for database management, Likert Scales, and open-ended discussions. It offers client, admin, and anonymous logins, with the potential for robust client-admin interactions. The system's technical aspects, including database management and user interactions, make it an intriguing project for developers, leveraging available resources and technology effectively.

Section 3:

Context Diagram



Section 4:

Activity Diagram

Based on the feedback provided and additional topics covered in class, you are to revise, refine, complete and include your context diagram with sprint 3. Therefore, you will have an improved version of the activity diagram you provided in Sprint 1 and 2.

Section 5: System Requirements

5.1: Use Cases

Use Case no: 1
Use Case Name: User Login
Actors: User, Admin, and Database
Description:

- The user is greeted with a login page. The user will email account information (email and password). Forgot Password is an extended option. The user submits information. Information is validated with the database. Successful input will allow users to access the rest of the application. If there is an unsuccessful input, applications pops up a “try again” message. Admin can access the database in order to update or change information.

Alternate Path:

- Only one path where successful login will lead to rest of the application.

Exception Path:

- If the user does not have an account, there will be a Sign Up page or an option to opt for anonymous browsing.

Pre-Condition:

- Existing account.

Post-Condition:

- Allows you to enter the rest of the application after the pre-condition has been met.

Use Case no: 2
Use Case Name: Sign Up Page
Actors: User, Database
Description:

- The user does not have an existing account. The user is led to a sign up page to create an account. The user enters the email and password associated with an account. If there is a valid input for email and password, it gets stored in a database which stores this information along with the new account ID. If there is a invalid input, the page will warn the user and allow them to try again. If there is an input that is not available such as entering an existing email, the app will send the user a message saying the email has already been used. After a successful input has been stored in the database, the user will be prompted to try to log in again. Alternate Path: There is only one path, in this case where the User can create an account, unless there is an error which prompts the user to try again.

Exception Path:

- If the user does have an account, it will lead to the login page.

Pre-Condition:

- The user does not have an account with the application.

Post-Condition:

- The user is able to log in due to creating an account with user inputted information.

Use Case no: 3

Use Case Name: Account Update
Actors: User, Database
Description:

- The user will be given an option to update additional account information such as university name, degree, and learning style. Information is stored within the database and will be used later when the application makes class recommendations to users.

Alternate Path:

- Users can access this page from any other page in order to allow them to update information at any time.

Exception Path:

- If the user enters a university or degree that is not listed, they will be allowed to type in their information rather than clicking on available options.

Pre-Condition:

- User accounts must already be created.

Post-Condition:

- Users will be allowed to update this information any time and as many times as they need.

Use Case no: 4
Use Case Name: Course Lookup
Actors: User, Database
Description:

- The user wants to search for a specific course to gather information and reviews about it. The user interacts with the Course Feedback web application interface to input the course details or keywords. The system then queries the database to retrieve relevant information, including course structure, difficulty level, teaching style, and student reviews. The user can view detailed course descriptions and ratings to make an informed decision about enrollment.

Alternate Path:

- If the user enters incomplete or incorrect course information, the system prompts them to refine their search criteria or suggests similar courses based on the provided keywords.

Exception Path:

- If the database encounters technical issues or fails to retrieve course information, the system notifies the user about the temporary unavailability and prompts them to try again later.

Pre-Condition:

- The Course Feedback web application is accessible and operational.The database contains updated information about courses, reviews, and user feedback.The user has access to the internet and a compatible device to interact with the application.

Post-Condition:

- The user successfully retrieves relevant information about the desired course.The user may choose to enroll in the course based on the gathered insights and reviews.The database logs the user's search query for future analysis and system improvement.

Use Case no: 5
Use Case Name: Likert Scale Class Rating
Actors: User, Database
Description:

- The user intends to provide a rating for a specific course using the Likert Scale provided by the Course Feedback web application. The user selects the course of interest and submits their rating based on predetermined criteria such as course content, teaching effectiveness, workload, and overall satisfaction. The system records the user's rating in the database for future reference and analysis.

Alternate Path:

- If the user decides not to provide a rating for a particular course, they can navigate back to the course listing or explore other features of the application.

Exception Path:

- If the database encounters errors during the rating submission process, such as connectivity issues or data validation problems, the system alerts the user about the issue and prompts them to try again later.

Pre-Condition:

- The Course Feedback web application is accessible and functional. The user is logged into their account or accessing the application anonymously. The user has navigated to the page where they can submit ratings for courses.

Post-Condition:

- The user's rating for the selected course is successfully recorded in the database. The overall rating for the course is updated based on the collective ratings from users. The user may choose to provide additional feedback or explore other courses within the application.

Use Case no: 6
Use Case Name: Anonymous User Login
Actors: User, Database
Description:

- The user intends to access the Course Feedback web application anonymously to explore course evaluations and reviews without revealing their identity. The user navigates to the application's login page and selects the option for anonymous login. Upon selecting this option, the system grants the user access to the application's features and functionalities while maintaining their anonymity throughout the session. The user can search for courses, view ratings, and read reviews without creating a personalized account.

Alternate Path:

- If the user initially attempts to log in using their credentials or register for an account but decides to remain anonymous instead, they can select the anonymous login option from the login page or registration form.

Exception Path:

- If the user encounters difficulties while attempting to access the application anonymously, such as technical issues or access restrictions, the system provides clear instructions and troubleshooting steps to facilitate the anonymous login process.

Pre-Condition:

- The Course Feedback web application is accessible and operational. The user has reached the application's login or registration page. The option for anonymous login is available and clearly visible to the user.

Post-Condition:

- The user successfully gains access to the application's features and content while remaining anonymous. The system securely manages the user's session data without storing any personally identifiable information. The user can explore course evaluations and reviews without the need for creating or logging into a personalized account. Upon exiting the session or closing the application, the user's anonymity is preserved, and no trace of their activity is retained within the system.

Use Case no: 7
Use Case Name: Non-Anonymous User Login
Actors: User, Database
Description:

- The user intends to access the Course Feedback web application using a non-anonymous login to take advantage of personalized features and functionalities. The user navigates to the application's login page and enters their credentials, including username and password, to authenticate their identity. Upon successful authentication, the system grants the user access to personalized features such as saving favorite courses, accessing past feedback submissions, and engaging in community discussions.

Alternate Path:

- If the user forgets their password, they can utilize the "Forgot Password" feature to reset their credentials and regain access to their account.

Exception Path:

- If the user encounters authentication issues, such as entering incorrect credentials multiple times or experiencing technical difficulties, the system provides error messages and guidance to assist the user in resolving the issue.

Pre-Condition:

- The Course Feedback web application is accessible and operational. The user has previously registered for an account on the application and possesses valid login credentials. The login page of the application provides options for entering username/email and password for non-anonymous login.

Post-Condition:

- The user successfully gains access to personalized features and functionalities within the application. The system securely manages the user's session data, ensuring confidentiality and data integrity throughout the user's interaction with the application. The user can explore course evaluations, submit feedback, and interact with other users while leveraging the benefits of a personalized account. Upon logging out or terminating the session, the user's account remains securely logged out, preventing unauthorized access to their account information.

Use Case no: 8

Use Case Name: Non-anonymous User Profile Details
Actors: User, Database
Description:

- Non-anonymous users of the Course Feedback web application can view and update their profile details, including grade received in class, academic year, major, and hours spent in class. Upon accessing their profile settings, the user can input or modify their academic information to provide context for their course feedback submissions and enhance community engagement.

Alternate Path:

- If the user wishes to update their profile details at a later time, they can navigate to the profile settings section from the application's dashboard or user menu.

Exception Path:

- If the user encounters difficulties while updating their profile details, such as data validation errors or technical issues, the system provides informative error messages and assistance to help the user resolve the issue.

Pre-Condition:

- The Course Feedback web application is accessible and operational. The user is logged into their non-anonymous account and has navigated to the profile settings section. The user has valid information regarding their grade received in class, academic year, major, and hours spent in class.

Post-Condition:

- The user's profile details, including grade received in class, academic year, major, and hours spent in class, are accurately updated in the database. The updated profile information is securely stored and associated with the user's account, ensuring consistency and reliability across the application. Non-anonymous users can leverage their profile details to provide more insightful course feedback, connect with peers in similar academic programs, and contribute to a vibrant learning community. The user may choose to revisit their profile settings in the future to make further updates or adjustments as needed.

Use Case no: 9
Use Case Name: Admin Powers Management
Actors: Admin, Database
Description:

- The administrator of the Course Feedback web application possesses exclusive powers to manage user activities, review feedback submissions, and maintain database integrity.

The administrator interacts with the application's administrative interface to execute various administrative tasks and ensure the smooth operation of the platform.

Alternate Path:

- If the administrator encounters a high volume of feedback submissions or user activities, they may prioritize tasks based on urgency and importance to streamline administrative operations effectively.

Exception Path:

- If the administrator experiences technical difficulties or system errors while performing administrative tasks, the system provides error notifications and troubleshooting guidance to help resolve the issue promptly.

Pre-Condition:

- The Course Feedback web application is accessible and operational. The administrator is logged into their administrative account and has access to administrative privileges and features. The administrative interface provides options and tools for managing user activities, feedback submissions, and database operations.

Post-Condition:

- The administrator successfully executes administrative tasks, such as reviewing feedback submissions, managing user accounts, and moderating user interactions. The system maintains database integrity by ensuring that user activities and feedback submissions adhere to community guidelines and platform policies. Administrative actions and decisions are securely recorded and logged within the system for audit and accountability purposes. The administrator may monitor system performance, identify areas for improvement, and implement enhancements to enhance user experience and platform functionality.

Use Case no: 10
Use Case Name: User Messaging for Course Inquiry
Actors: User, Database
Description:

- Users of the Course Feedback web application can communicate with each other through a messaging feature to inquire about classes, seek advice, and share insights regarding course experiences. The messaging functionality facilitates peer-to-peer interaction and promotes knowledge exchange within the application's user community.

Alternate Path:

- If users wish to initiate a new conversation or respond to an existing message thread, they can navigate to the messaging interface from the application's dashboard or user menu.

Exception Path:

- If users encounter difficulties while sending or receiving messages, such as network connectivity issues or system errors, the system provides notifications and troubleshooting guidance to help resolve the issue.

Pre-Condition:

- The Course Feedback web application is accessible and operational. Users are logged into their respective accounts and have access to the messaging feature within the application. The messaging interface provides options for users to compose new messages, view message history, and manage message notifications.

Post-Condition:

- Users successfully engage in meaningful conversations and exchange information about classes, course content, teaching styles, and academic experiences. The messaging feature enhances user engagement and fosters a sense of community within the application, facilitating peer support and collaboration. Users may choose to archive or delete message threads, manage message settings, and customize their messaging preferences based on their communication needs. The messaging system securely stores and encrypts user messages, ensuring confidentiality and data privacy throughout the messaging exchange.

Use Case no: 11
Use Case Name: User Following System
Actors: User, Database
Description:

- Users can follow other users within the Course Feedback system. Once a user follows another, they receive notifications or updates about new class ratings and feedback posted by those they follow. This feature allows users to stay informed about the opinions and experiences of users they trust or find insightful.

Alternate Path:

- Users can unfollow other users at any time if they no longer wish to receive updates.

Exception Path:

- If a user tries to follow an account that does not exist or has been deactivated, the system will display an error message indicating that the follow action cannot be completed.

Pre-Condition:

- Users must have an account and be logged into the system.

Post-Condition:

- Users receive updates on new class ratings and feedback from users they follow.

Use Case no: 12
Use Case Name: Social Group Creation
Actors: User, Database
Description:

- Users can create social groups within the app based on course subjects or other interests. These groups serve as forums for discussion and advice. Once a group is created, it is stored in the database for other users to join.

Alternate Path:

- Users can join existing groups if they don't want to create a new one.

Exception Path:

- If a group name already exists, the user is prompted to choose a different name.

Pre-Condition:

- Users must be logged in.

Post-Condition:
- New social group is available for others to join.

Use Case no: 13
Use Case Name: Course Content Feedback
Actors: User, Database
Description:

- Users can provide detailed feedback specifically on the course content, including the curriculum, relevance, and resources provided. After selecting a course, the user can enter their feedback through a form that prompts them for specific aspects of the content they wish to comment on. This information is then stored in the database and made available for review by other students and faculty for course improvement.

Alternate Path:

- If the user wants to provide general feedback not specific to any content area, they can use a general feedback form.

Exception Path:

- If the user attempts to submit feedback for a course they have not enrolled in, the system will prompt them to select a course they have taken.

Pre-Condition:

- The user must be logged in and must have either taken the course or be currently enrolled.

Post-Condition:

- Detailed feedback on the course content is stored in the database and is made visible in the course's feedback section for other users.

Post-Condition:

- Feedback is stored and visible to other users.

Use Case no: 14
Use Case Name: Course Material Sharing
Actors: User, Database
Description:

- Users can share course materials such as study guides or notes. After uploading the material and associating it with the relevant course, the system stores it in the database for access by other students.

Alternate Path:

- Users can request materials for a course if not available.

Exception Path:

- If the upload fails due to file size or format, the user is notified to make adjustments.

Pre-Condition:

- User must be logged in.

Post-Condition:

- Course materials are available for other users.

Use Case no: 15
Use Case Name: User Account Deactivation
Actors: User, Database
Description:

- Users can deactivate their account through the settings menu. Upon confirmation, the account is deactivated, and personal data is anonymized in the database.

Alternate Path:

- Before account deactivation, the system could offer the user an option to temporarily disable the account instead. This path would allow users to take a break without permanently removing their data, with the option to reactivate their account later.

Exception Path:

- If there is an error during deactivation, the user is informed and asked to try again.

Pre-Condition:

- Users must be logged in and must navigate to account settings.

Post-Condition:

- User account is deactivated and personal data is anonymized.

5.2: Requirements

Requirements number: 1
Use Case number: 1
Introduction:
- A simple layout with two input form for our users to login

Inputs:
- The inputs are the user email and password associated with their account.

Requirements Description:
- The user should be able login to their account on the application given all preconditions are met.

Outputs:
- The user should be able to access the rest of the application once logged in.

Requirements number: 2
Use Case number: 2
Introduction:
- A user friendly interface to allow users to register for an account if the precondition of having an account is not met.

Inputs:
- User will input name, email, password, and confirm password

Requirements Description:
- The user will be able to sign up for an account in order to satisfy the precondition of having an account to access the rest of the application.

Outputs:
- After the user has successfully registered, the user will be directed to the homepage of the application.

Requirements number: 3
Use Case number: 3
Introduction:
- The user is given an option to update account information like university name, degree, and learning style. This information will be used for recommendations that will be implemented later in the development cycle.

Inputs:
- The inputs are university name, degree, and learning style.

Requirements Description:
- The system will provide input fields for the user to update university, degree, and learning style. The database must validate the entered information in order to ensure data accuracy.

Outputs:
- Once data has been updated, the system will return a confirmation to the user notifying them of the saved inputs.

Requirements number: 4
Use Case number: 4
Introduction:
- A feature in our application to allow users to search for a specific course and read reviews about the course.

Inputs:
- User inputs course name.

Requirements Description:
- The system will provide an input field for users to enter course details which will then query the database to search for relevant information.

Outputs:

- After the database has found the relevant course information, the system will display the information found with the course name along with reviews and suggestions.

Requirements number: 5
Use Case number: 5
Introduction:
- This is a feature that allows users to provide ratings for specific courses using a Likert Scale. This rating will be displayed once a user searches for a certain course.

Inputs:
- User inputted course as well as Likert Scale ratings from other users.

Requirements Description:
- The system will provide an interface for users to rate their experience of a certain course using the Likert Scale. The system will record this rating in the database. The criteria of the rating should be created from course material, professor teaching style, workload, and satisfaction.

Outputs:
- After a submission of a rating of a course, the system will record and confirm the user input and update the course's overall rating.

Requirements number: 6
Use Case number: 6
Introduction:
- This feature allows users to access the application anonymously in order to evaluate and review courses and professors without revealing their identity.

Inputs:
- Selection of anonymous login options at the application's login page.

Requirements Description:
- The system will provide an option to opt for anonymous login at the first login page. Once selected, the system allows the user to access the application features while maintaining anonymity.

Outputs:
- After selecting the anonymous login option, the system will display a confirmation message to the user who is then free to explore the application anonymously.

Requirements number: 7
Use Case number: 7
Introduction:
- This feature is for registered users to access the application using the information used at account creation to utilize the personalized suggestions. The users should be able to access features like favorite courses, previous review submissions, and engage in discussions with other users.

Inputs:
- Username or email and the password for account authentication

Requirements Description:

- The system allows for users to enter non anonymously in order to access personalized features.

Outputs:

- Once the user has been successfully authenticated from the database, the system will send a confirmation message to the user and grant access to the personalized features of the application.

Requirements number: 8
Use Case number: 8
Introduction:

- This feature allows non-anonymous users to view and update profile details such as grade received in class, academic year, and hours spent in class.

Inputs:

- User details such as grade received, academic year, major, hours spent in and out of class.

Requirements Description:

- The system will have a dedicated section in the application for non-anonymous users to view and update their profile details relating to their education and course history.

Outputs:

- After successfully updating their details, the system will store the information in the database for future analysis and review.

Requirements number: 9
Use Case number: 9
Introduction:

- A feature to allow the admin of the application to manage user activities, review feedback submissions, and maintain database integrity.

Inputs:

- Administrative actions that are only executable from the administrative interface.

Requirements Description:

- The system will provide the administrator with unique access and a specific interface for executing administrative tasks.

Outputs:

- After executing administrative tasks, the system will update the changes made and record the actions made in order to maintain integrity within the application.

Requirements number: 10
Use Case number: 10
Introduction:

- The feature is to allow users to communicate with others through direct messaging to discuss courses, ask advice, and share educational insights. This feature facilitates peer-to-peer interaction and allows for knowledge exchange within the community.

Inputs:

- User messages between other users in the community.

Requirements Description:

- The system should provide users the ability to message one another within the application as well as respond to existing threads.

Outputs:

- After sending or receiving a message, the system can display a chat between users in order to interact and store messages.

Requirements number: 11

Use Case number: 11

Introduction:

- Implementing a user following system within the Course Feedback platform to allow users to follow others and receive updates on their new class ratings and feedback.

Inputs:

- The ID or username of the user to follow. The action initiated by the user to follow or unfollow another user.

Requirements Description:

- The system must provide an interface that allows users to search for other users by ID or username and initiate a follow action. The system must update the database to reflect the new follower relationship and ensure that notifications or updates about new class ratings and feedback are sent to the followers. There should also be an option to unfollow users.

Outputs:

- Confirmation or the follow or unfollow action to the user. Notifications or a feed update showing new class ratings and feedback from followed users.

Requirements number: 12

Use Case number: 12

Introduction:

- This requirement supports the creation of social groups within the application enabling users to discuss courses, share advice, and provide support.

Inputs:

- User input for the new social group's name, description, and possible course or interest tags.

Requirements Description:

- The system must provide an interface for creating new social groups, including input fields for the group's name and description, and a selection of tags. It should check for duplicate group names and confirm successful creation.

Outputs:

- A new social group is created in the database and is made discoverable to other users.

Requirements number: 13

Use Case number: 13

Introduction:

- This requirement enables users to provide feedback on course content, including curriculum relevance and resource quality.

Inputs:

- User selection of a course, feedback on various aspects of course content through a structured form.

Requirements Description:

- The system must allow users to select courses they have taken or are taking and provide detailed feedback on specific content aspects. Feedback is stored in the database and made accessible for course improvement considerations.

Outputs:

- User feedback is stored, and a summary of feedback is accessible to users and potentially faculty for course enhancement.

Requirements number: 14
Use Case number: 14
Introduction:

- This requirement allows users to share and access course materials, such as study guides or notes, fostering a collaborative learning environment.

Inputs:

- User upload of course material files, selection of associated course, and optional description.

Requirements Description:

- The system should provide a feature for users to upload and categorize course materials, validate file types and sizes, and store materials in the database for access by other students.

Outputs:

- Course materials are available for download or viewing by other users, with the system tracking and displaying the number of downloads/views.

Requirements number: 15
Use Case number: 15
Introduction:

- This requirement outlines the process for users to deactivate their accounts, ensuring their data is handled appropriately.

Inputs:

- User initiation of account deactivation from the settings menu, confirmation of deactivation.

Requirements Description:

- The system must offer users the option to deactivate their account, requiring confirmation to proceed. Upon deactivation, the user's personal data is anonymized in the database.

Outputs:

The user's account is deactivated, their session is terminated, and a confirmation of deactivation is provided. Personal data associated with the account is anonymized in the database.

5.3 : Use Case Diagram
Diagram #1 (Use Case #1)

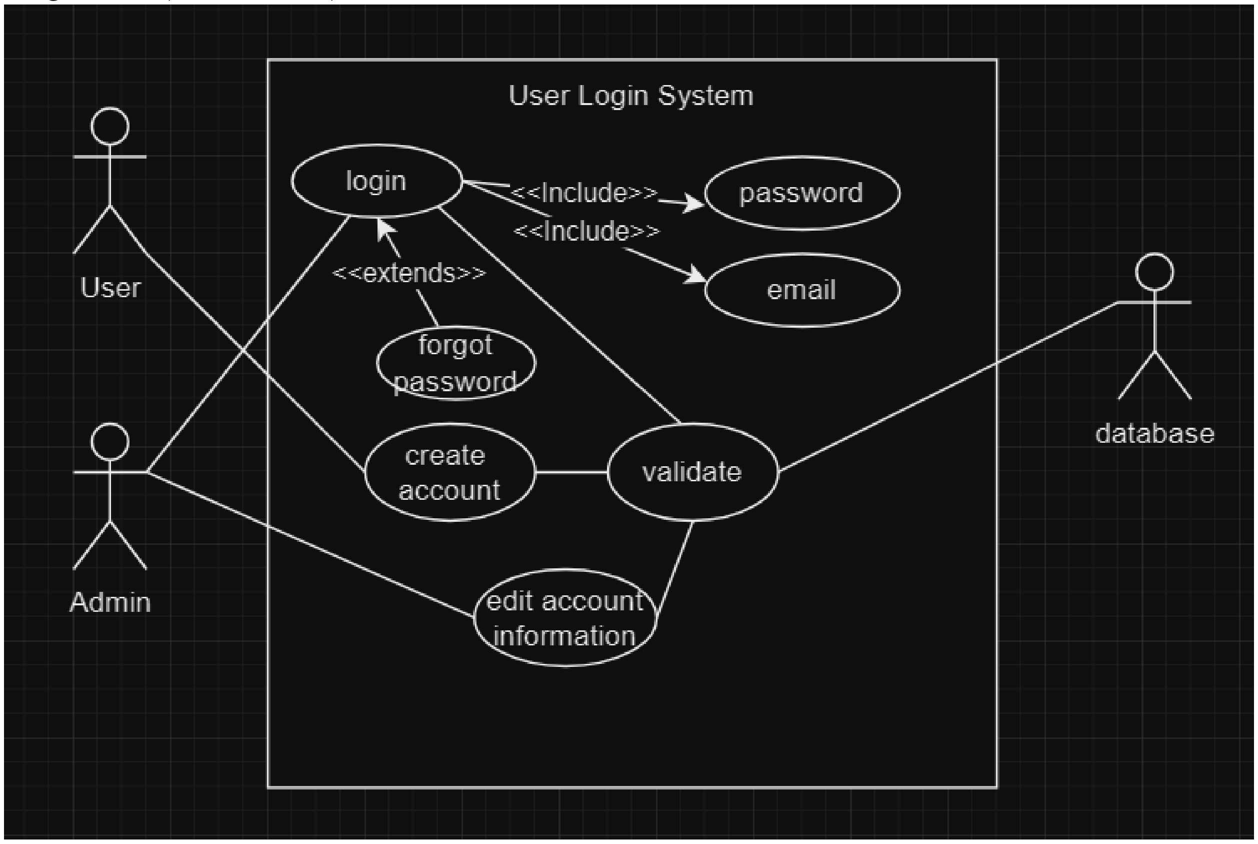


Diagram #2(Use Case #4)

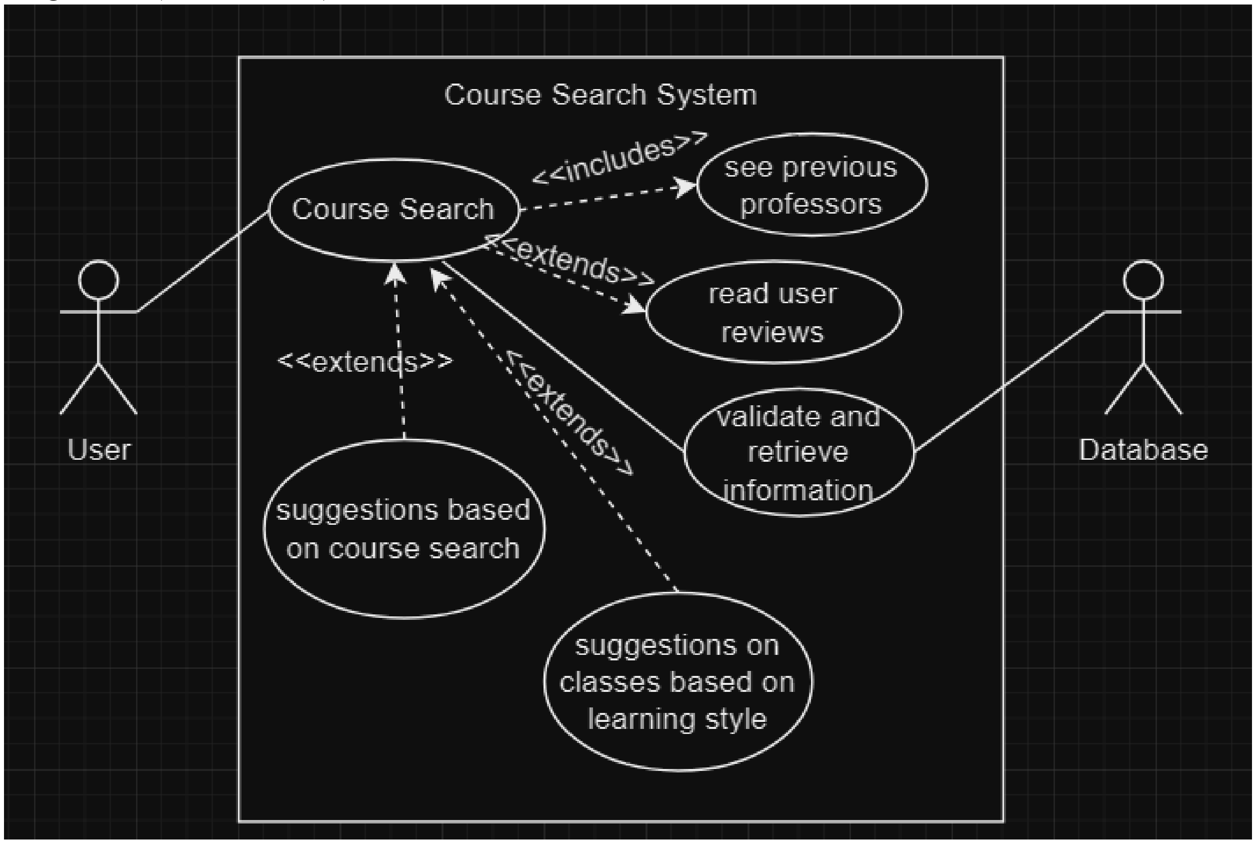


Diagram #3(Use Case #15)

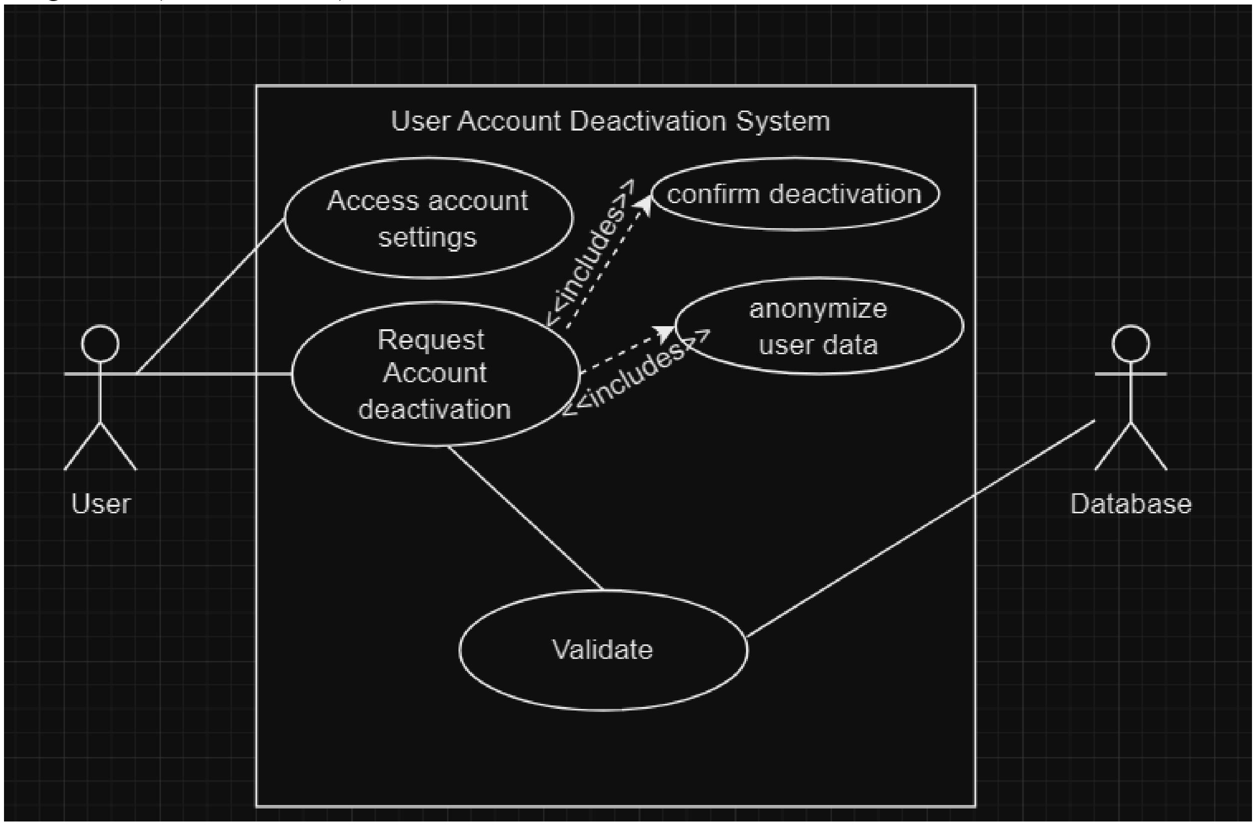
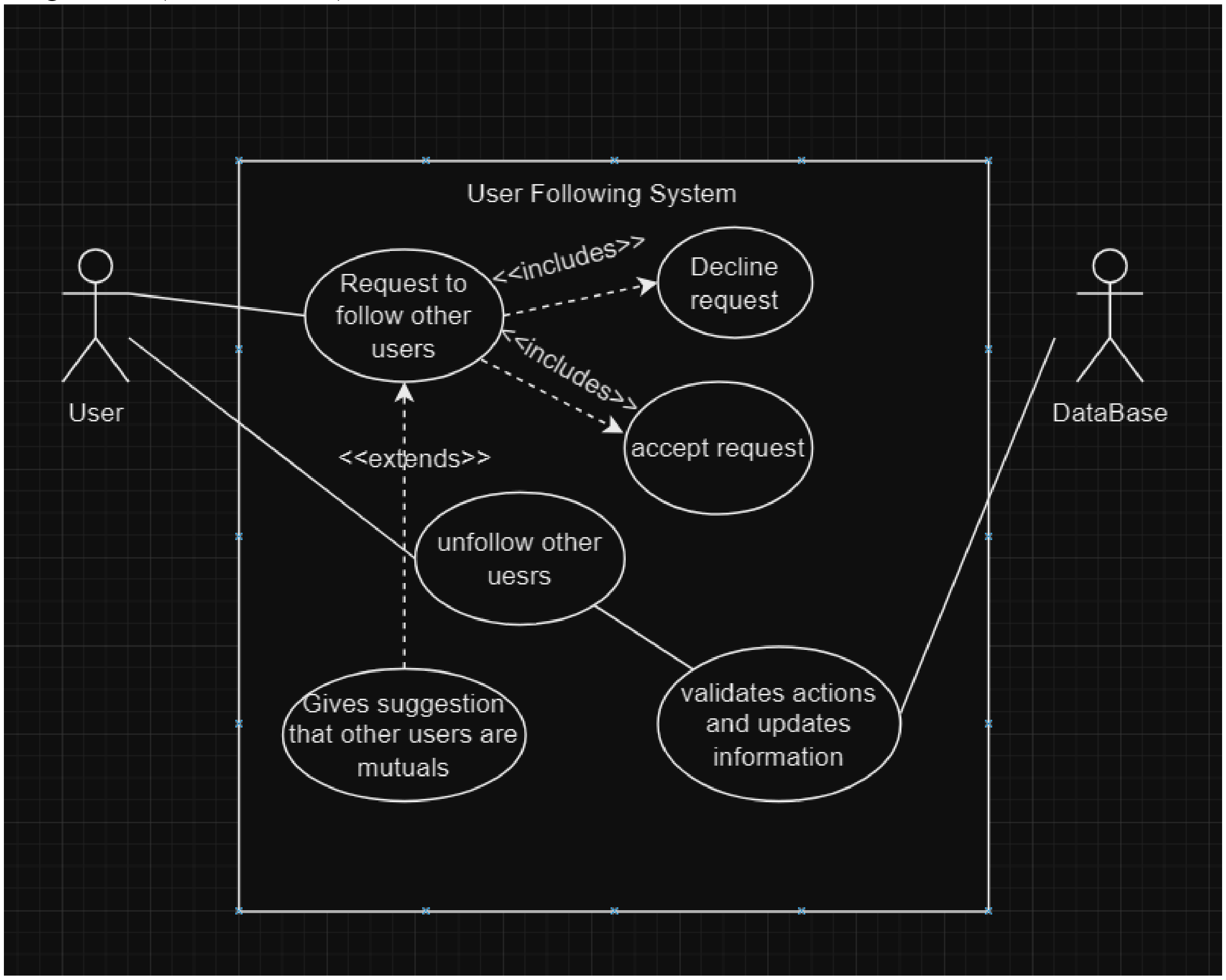


Diagram #4 (Use Case #11)

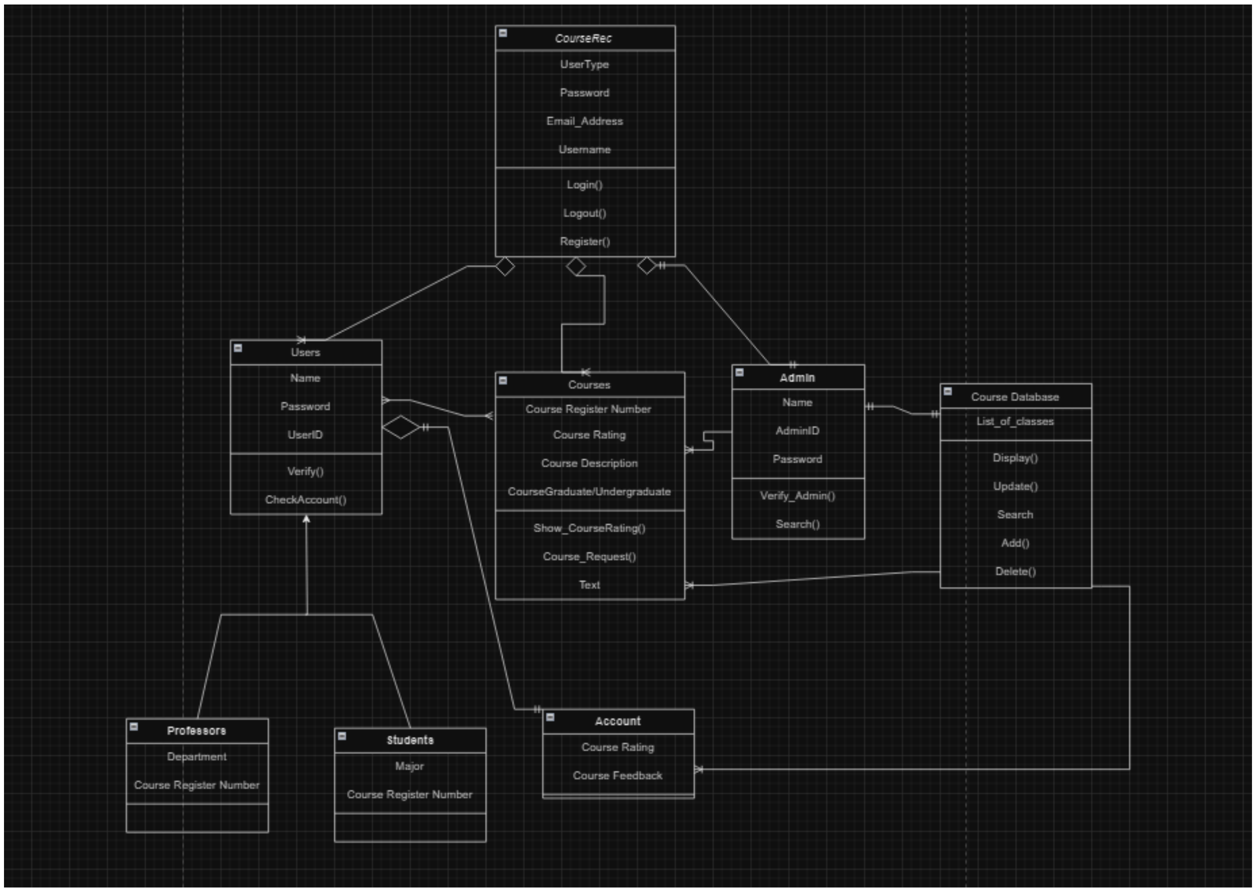


Section 6:

Database Tables

Based on the feedback provided and additional topics covered in class, you are to revise, refine, complete and include your database tables with sprint 2. Therefore, you will have an improved version of the database tables you provided in Sprint 2.

Section 7:



Section 8:

Behavioral Modeling

Create a UML sequence diagram for two major use cases, with one focusing on the creation of the system. (do NOT make sequence diagram for login, logout, or registration modules).

Show appropriate lifelines, activations, and message types. You may also use loop, alternative (alt), and optional fragments if needed. You may use your class diagram to identify the objects' names and messages (methods) that you need to develop your sequence diagram.

A sequence diagram is a visual representation of how objects in a system interact. Keep in mind that the reasons to create a sequence diagram is to:

- refine your use case diagram and uses cases (adding missed cases [functionalities]),
- refine your class diagram (adding missed methods [messages]),
- transition from the conceptual model and start thinking about the implementation, which is the most important.

UML Sequence Diagram #1:

Use Case no: 4
Use Case Name: Course Lookup
Actors: User, Database
Description:

- The user wants to search for a specific course to gather information and reviews about it. The user interacts with the Course Feedback web application interface to input the course details or keywords. The system then queries the database to retrieve relevant information, including course structure, difficulty level, teaching style, and student reviews. The user can view detailed course descriptions and ratings to make an informed decision about enrollment.

Alternate Path:

- If the user enters incomplete or incorrect course information, the system prompts them to refine their search criteria or suggests similar courses based on the provided keywords.

Exception Path:

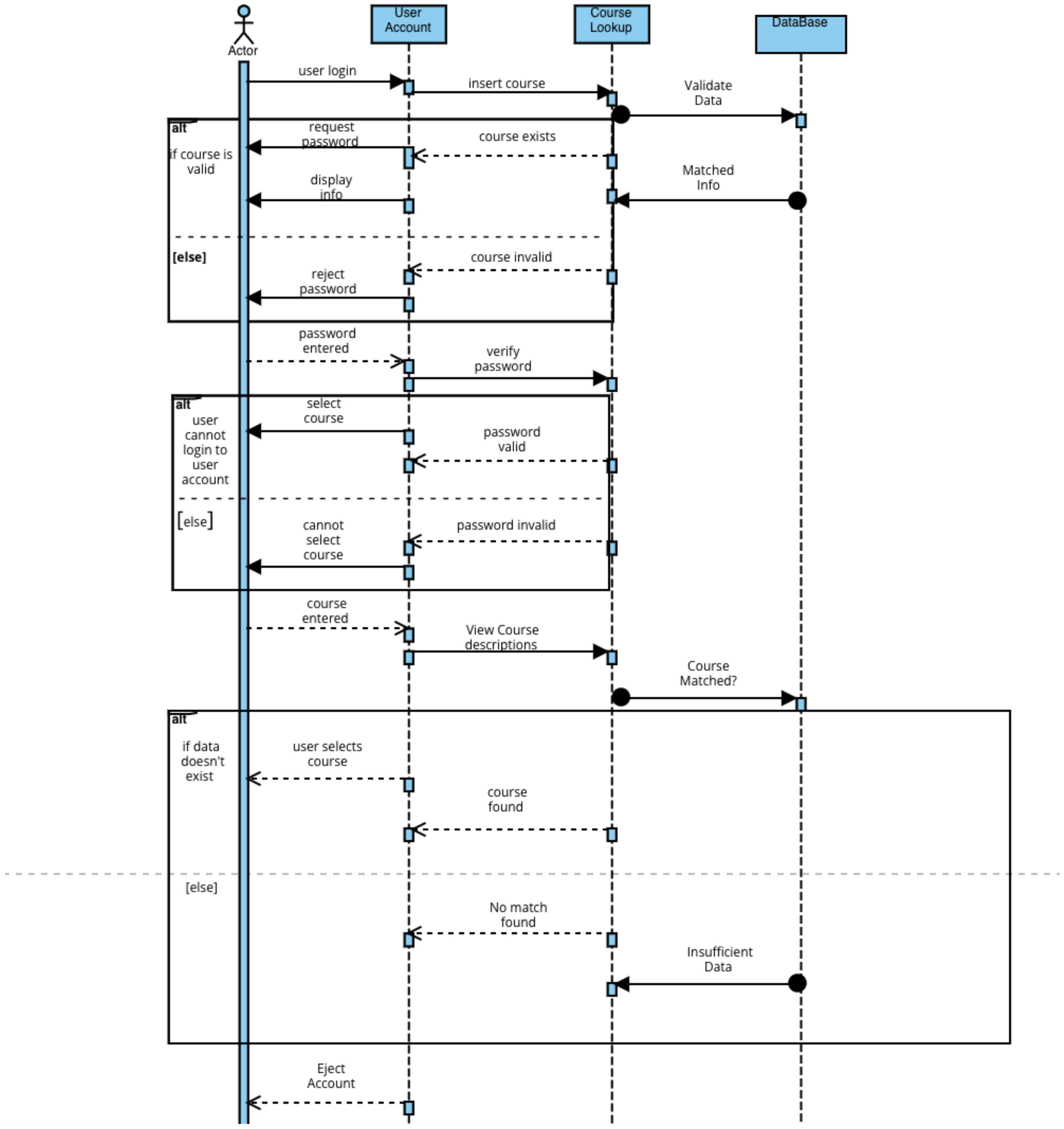
- If the database encounters technical issues or fails to retrieve course information, the system notifies the user about the temporary unavailability and prompts them to try again later.

Pre-Condition:

- The Course Feedback web application is accessible and operational.The database contains updated information about courses, reviews, and user feedback.The user has access to the internet and a compatible device to interact with the application.

Post-Condition:

- The user successfully retrieves relevant information about the desired course.The user may choose to enroll in the course based on the gathered insights and reviews.The database logs the user's search query for future analysis and system improvement.



Section 9:

Implementation

Implement the Database Design (Tables, Backend):

- Select the tables associated with four of the major use cases in your system.
- Implement the chosen tables using the selected database management system (MySQL, MS-SQL server, Oracle, etc.)

Creating tables using SQL scripts makes it easy to create the necessary tables on another computer: Development Environment, Test Environment, Production Environment; Deployment

Note: These are some free hosting database management systems:

  - Amazon RDS (Relational Database Service) has a free usage tier for 12 months to run a Micro DB instance with 20GB of storage and 10 million I/Os
  - Google Cloud Firestore / Datastore is a document-store with a permanent free tier of 1GB storage, 50k reads and 20k writes per day.
  - MongoDB Atlas has a free tier with a 3-node replica set and 512MB storage.
  - Redis Labs offers a 30MB Redis instance for free.
  - Heroku Postgres has a free tier limited to 10k individual rows of data.

Implement the Class Diagram Design (Frontend and Logic): (develop/write code)

- Get an overview of the software frameworks or platforms, programming languages, host, etc. Install the necessary Software
- Start creating the main parts (two of the major use cases) for your application (both Frontend, logic (CODE), and GUI).
- Create a good structure for your code.
- Test that you can communicate with the Database (data go from Frontend [GUI] to Backend [Tables]).
- A description of how to compile and run your code
- You should not instruct us to compile and execute your code in any IDE.

- We recommend writing your own build script or generating one in an IDE (ant, mvn, etc.).
- Or, if you chose to host your system in the cloud, make sure to provide the host link, username and password in the section of "description for how to compile your code".

Note: There are some Web Based Frameworks and IDE tools

Backend Frameworks	Frontend JavaScript Frameworks	Integrated Development Environment (IDE) Software
Spring (Java)	Angular (JavaScript, by Google)	Visual Studio
ASP.NET (C#)	React	IntelliJ IDEA
Django (Python)	Vue	Xcode
Laravel (PHP)	Ember	Eclipse

Express (Node.js)	Backbone	WebStorm
Ruby on Rails		PhpStorm
		NetBeans
		AWS Cloud9

Section 10:

Testing

Create test cases:

- Develop a collection of requirements-based tests (functionality testing):
- ○ Use the following steps to develop your test cases:
- ○ Identify features (functionality) related to the main 5 of the major use cases
- ■ E.g.: insert (int N, list A)
- ○ Partition inputs into equivalence classes: (for each feature)
- ■ E.g.: int N is 5-digit integer between 10000 and 99999,
- ■ One possible partition is: <10000, 10000-99999, >100000)
- ■ E.g.: list A is a list of length 1-10.
- ■ One partition is: empty list, list of length 1, list of length 2-10, list of length > 10
- ○ Test Specification: (for each feature)
- ■ E.g.: insert (< 10000, empty list)
- ○ Test cases:
- ■ E.g.: insert (50000, {})
- When you are giving test specification, clearly mention the use case it is being given for.
- Use the below example format:

Test Case ID:

Description:

Test Inputs:

Expected Results:

Dependencies:

Initialization:

Test Steps:

Post-conditions:

Test Case for User Login (Use Case 1)

Test Case ID: TC001
Description: Verify successful login with valid credentials
Test Inputs: Email: user@example.com, Password: validPassword
Expected Results: User is logged in and redirected to the homepage
Dependencies: User account exists
Initialization: Ensure user account exists in the database
Test Steps: 1. Navigate to login page. 2. Input email and password. 3. Click login.
Post-conditions: User sees the homepage.

Test Case for Sign Up Page (Use Case 2)

Test Case ID: TC002
Description: Validate account creation with a unique email
Test Inputs: Email: new@example.com, Password: newPassword
Expected Results: Account is created, user is prompted to log in
Dependencies: Email is not in use
Initialization: Check no account exists with the email
Test Steps: 1. Navigate to sign-up page. 2. Input new email and password. 3. Submit.
Post-conditions: User is directed to login page.

Test Case for Account Update (Use Case 3)

Test Case ID: TC003
Description: Ensure users can update account information
Test Inputs: University: "Tech University", Degree: "Computer Science"
Expected Results: Account information is updated in the database

Dependencies: User is logged in
Initialization: User logs in and navigates to account settings
Test Steps: 1. Open account settings. 2. Update university and degree. 3. Save changes.
Post-conditions: User sees a success message.

Test Case for Course Lookup (Use Case 4)

Test Case ID: TC004
Description: Validate searching for a course by name
Test Inputs: Course name: "Introduction to Programming"
Expected Results: Relevant course information is displayed
Dependencies: Course exists in the database
Initialization: Ensure course data is present
Test Steps: 1. Navigate to course search. 2. Enter course name. 3. Initiate search.
Post-conditions: Course details are shown.

Test Case for Likert Scale Class Rating (Use Case 5)

Test Case ID: TC005
Description: Verify users can rate a course using a Likert scale
Test Inputs: Course ID: 101, Rating: 4
Expected Results: Rating is recorded and affects course's overall rating
Dependencies: User has completed the course
Initialization: User selects a completed course
Test Steps: 1. Choose course. 2. Select rating. 3. Submit rating.
Post-conditions: Success message is displayed.

Test Case for Anonymous User Login (Use Case 6)

Test Case ID: TC006
Description: Ensure anonymous login provides access to course reviews
Test Inputs: Action: Select anonymous login
Expected Results: User accesses the platform anonymously, can view courses and reviews
Dependencies: None
Initialization: Navigate to login page
Test Steps: 1. Select anonymous login option.
Post-conditions: User views courses and reviews without personal data.

Test Case for Non-Anonymous User Login (Use Case 7)

Test Case ID: TC007
Description: Validate non-anonymous login functionality
Test Inputs: Email: registeredUser@example.com, Password: userPassword

Expected Results: User logs in and accesses personalized features
Dependencies: User account exists and is verified
Initialization: User is registered and confirmed
Test Steps: 1. Input email and password. 2. Click login.
Post-conditions: User sees personalized dashboard.

Test Case for Non-anonymous User Profile Details (Use Case 8)

Test Case ID: TC008
Description: Confirm users can update and save profile details
Test Inputs: Major: "Software Engineering", Year: "Junior"
Expected Results: Profile details are updated in the user's account
Dependencies: User is logged in
Initialization: User accesses profile settings
Test Steps: 1. Edit profile details. 2. Save changes.
Post-conditions: Changes are reflected in the user's profile.

Test Case for Admin Powers Management (Use Case 9)

Test Case ID: TC009
Description: Verify admin can manage user feedback and accounts
Test Inputs: Action: Review and approve feedback, Action: Deactivate user account
Expected Results: Feedback is published or archived, user accounts are managed appropriately
Dependencies: Admin login
Initialization: Admin logs into the admin panel
Test Steps: 1. Navigate to feedback section. 2. Approve feedback. 3. Manage user accounts.
Post-conditions: System updates based on admin actions.

Test Case for User Messaging for Course Inquiry (Use Case 10)

Test Case ID: TC010
Description: Ensure users can send and receive messages about courses
Test Inputs: Recipient: user2@example.com, Message: "How was the workload for CS101?"
Expected Results: Message is sent to the recipient, and they can reply
Dependencies: Both users are registered and logged in
Initialization: User navigates to the messaging feature
Test Steps: 1. Compose a new message. 2. Select recipient. 3. Send message.
Post-conditions: Recipient receives and can respond to the message.

Test Case for Course Difficulty Rating (Revised to User Following System) (Use Case 11)

Test Case ID: TC011
Description: Validate users can follow others and receive updates on new class ratings and feedback
Test Inputs: Action: Follow User ID 102
Expected Results: User starts receiving updates on new ratings and feedback from User ID 102
Dependencies: Both users exist in the system
Initialization: User searches for another user to follow
Test Steps: 1. Search for User ID 102. 2. Click "Follow".
Post-conditions: User receives notifications for new activity from followed user.

Test Case for Social Group Creation (Use Case 12)

Test Case ID: TC012
Description: Validate that users can create and join social groups within the platform.
Test Inputs: Group Name: "AI Enthusiasts", Description: "A group for AI hobbyists and professionals."
Expected Results: A new social group named "AI Enthusiasts" is created and listed for others to join.
Dependencies: User must be logged in.
Initialization: User logs into the system.
Test Steps:

1. Navigate to the social groups section.
2. Click on "Create Group."
3. Enter the group name and description.
4. Submit the creation form.

Post-conditions: The new group is visible in the user's and platform's group listings.

Test Case for Course Content Feedback (Use Case 13)

Test Case ID: TC013
Description: Ensure users can submit feedback on specific course content.
Test Inputs: Course ID: 305, Feedback: "Needs more practical examples in the second module."
Expected Results: Feedback is submitted successfully and is available for faculty review.

Dependencies: User must be logged in and have taken the course.
Initialization: Course with ID 305 exists and user has enrolled in it.
Test Steps:

1. Navigate to the feedback section of Course ID 305.
2. Enter the feedback in the provided textbox.
3. Click the "Submit Feedback" button.

   Post-conditions: Feedback is stored in the database associated with the course.

Test Case for Course Material Sharing (Use Case 14)

Test Case ID: TC014
Description: Validate that users can upload and share course-related materials.
Test Inputs: Course ID: 210, Material Description: "Midterm Study Guide", File: midterm_guide.pdf
Expected Results: The study guide is uploaded successfully and is accessible to other students enrolled in the course.
Dependencies: User must be logged in and have taken or currently be enrolled in the course.
Initialization: Course with ID 210 exists.
Test Steps:

1. Navigate to the materials section of Course ID 210.
2. Click on "Upload Material."
3. Choose the file and enter a description.
4. Submit the upload form.

   Post-conditions: The material is available for download by students in the course.

Test Case for User Account Deactivation (Use Case 15)

Test Case ID: TC015
Description: Verify that users can deactivate their accounts and have their data anonymized.
Test Inputs: User confirmation: "Yes, deactivate my account."
Expected Results: Account is deactivated, user is logged out, and personal data is anonymized or deleted as per the platform's policy.
Dependencies: User must be logged in.
Initialization: User logs into their account.
Test Steps:

1. Navigate to account settings.

2. Click on "Deactivate Account."
3. Confirm the deactivation request.
Post-conditions: User's account is no longer active, and they are logged out of the system.

GitHub link and Screenshot

- Using the project, you created for Sprint 1, identify a new to do; In Progress and Done columns for Sprint 3 Under these columns, create your own cards and have the assigned tasks and their status (in progress and done) written there as a list.
- Give the link and submit a screenshot of your project.

REPORT

Report Sections:

1. Title page
2. Section 1: Planning and Scheduling Table
3. Section 2: Problem Statement
4. Section 3: Context Diagram
5. Section 4: Activity Diagram(s)
6. Section 5: Use Case, Requirement and Use case diagram
7. Section 6: Database tables
8. Section 7: Class Diagram
9. Section 8: Behavioral Modeling
10. Section 9: Implementation
11. Section 10: Testing
    - Section 10.1: Test cases
12. GitHub link and screenshot
13. Demo Demonstration in the class

Report Format

- The first page will have: Project Title, group number, members’ name (underline the coordinator’s name), semester and year.
- Font size 12, Font type is times new roman, single space between lines.
- All paragraphs must be Text Justified.
- Pages are numbered
- Diagrams and tables must be centered.

Report Submission

- Strictly one submission per group
- The file should be named CoordinatorName_SprintNumber_GroupNumber.pdf
- The latest document submitted by the team coordinator will be graded.
- You (team coordinator) will also print out a copy of the report and submit it before the class on the submission day.