Test Document

for

TRACKit

Version 1.0

Group 5 **Group Name: Chill guys**

Aditya Gautam	**220064**	**agautam22@iitk.ac.in**
Aayush Singh	**220024**	**aayushs22@iitk.ac.in**
Dhruv Rai	**220365**	**dhruvrai22@iitk.ac.in**
Ved Prakash Vishwakarma	**221180**	**vedprakash22@iitk.ac.in**
Sharique Ahmad	**221002**	**asharique22@iitk.ac.in**
Dhruv Varshney	**220366**	**vdhruv22@iitk.ac.in**
Mayur Agrawal	**220641**	**mayurag22@iitk.ac.in**
Akash Verma	**220097**	**akashv22@iitk.ac.in**
Rahul Ahirwar	**220856**	**rahula22@iitk.ac.in**
Abhijeet Agarwal	**210025**	**abhijeeta21@iitk.ac.in**
Aryan Bansal	**200198**	**aryanb20@iitk.ac.in**

Course: **CS253**
Mentor TA: **Jeswaanth Gogula**
Date: **6 April 2025**

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
v1.00	Aditya Gautam Aayush Singh Dhruv Rai Ved Prakash Vishwakarma Sharique Ahmad Dhruv Varshney Mayur Agrawal Akash Verma Rahul Ahirwar Abhijeet Agarwal Aryan Bansal	Completed first version of the testing document	05/04/2025

1 Introduction

Testing Strategy

We used a hybrid testing strategy, as this was closest to the very high level of diversity of API requests in our App. For the functional testing of the Backend APIs and the Frontend, we used manual testing using Postman and visual inspection, while for non-functional testing related to performance, we used dedicated test scripts.

When was the Testing Conducted?

Informal testing by group members was done during every step in the project code development. Our repository, with over 290+ commits, saw the group members testing the code added in that git commit tested to the best of their abilities. We created dozens of issues in GitHub when another user spotted an issue.
Thus testing was an integral part of our development process itself.

In Addition to that, in the week preceding the submission of this document, the group engaged in vigorous and extensive formal testing of the product. Some bugs were found out, and immediately rectified/marked for rectification in the near future.

Who were the Testers?

Every group member contributed extensively to testing.

Coverage Criteria Used

We considered meeting the goals of the SRS, and the stability of the feature being tested, to be the coverage criteria.

The tool used for Testing

We used tools like Postman extensively during development and during testing.
Postman is a widely used API testing tool that facilitates the development, testing, and debugging of APIs. It allows users to send requests to API endpoints, inspect responses, and automate testing workflows.
We also used Browser Rendering, and manual UI-based operations to test the various features during Integration and System Testing.
We also wrote dedicated test scripts to stress test various aspects of the app, such as creating 2500 users at once.

2 Unit Testing

Login

Unit Details:

Involves the testing of the user login functionality.

Test Owner:

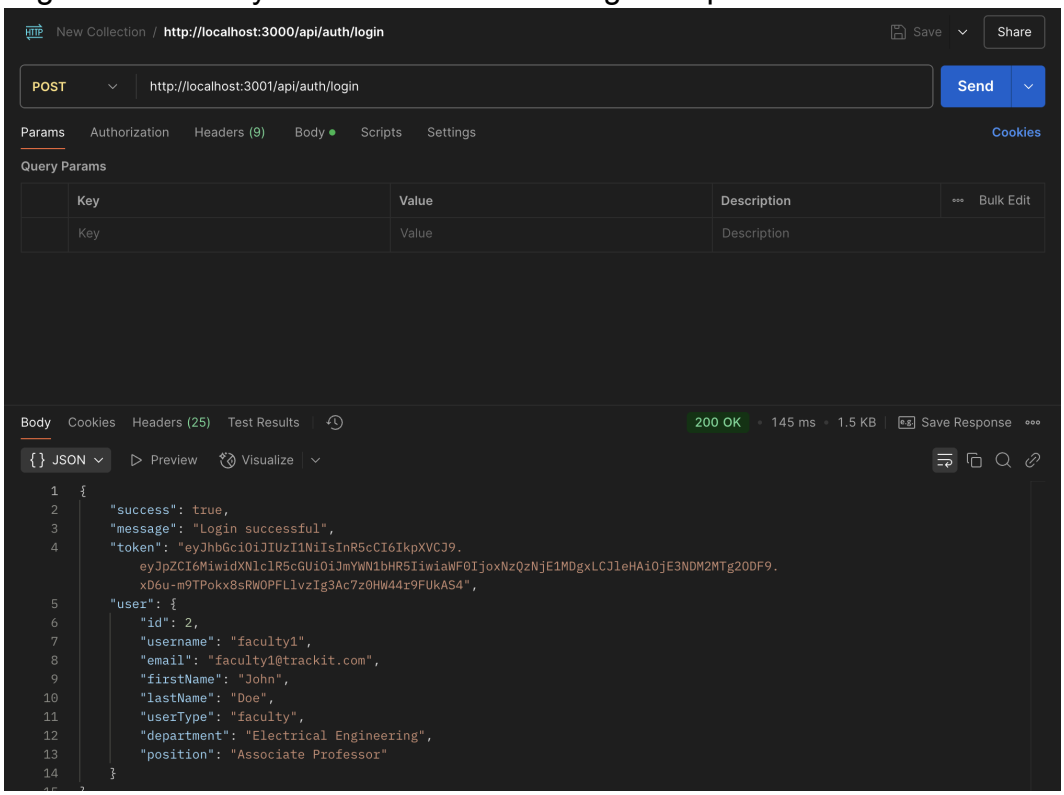
Aditya Gautam, 220024

Test Date:

2 April 2025

Test Results:

Login Functionality was found to be working as expected.



Structural Coverage:

The unit is working well and fulfils 100% of the requirements of the SRS.

Admin : User

Unit Details:

We have tested the functionality of user management within the admin interface to ensure seamless handling of user data. This includes verifying the ability to create new users, delete existing ones, and retrieve a comprehensive list of all registered users. Additionally, we have tested the critical features of adding and removing students from courses, ensuring proper enrollment and withdrawal processes.Throughout this testing process, our focus has been on maintaining data integrity and consistency in the database

Test Owner:

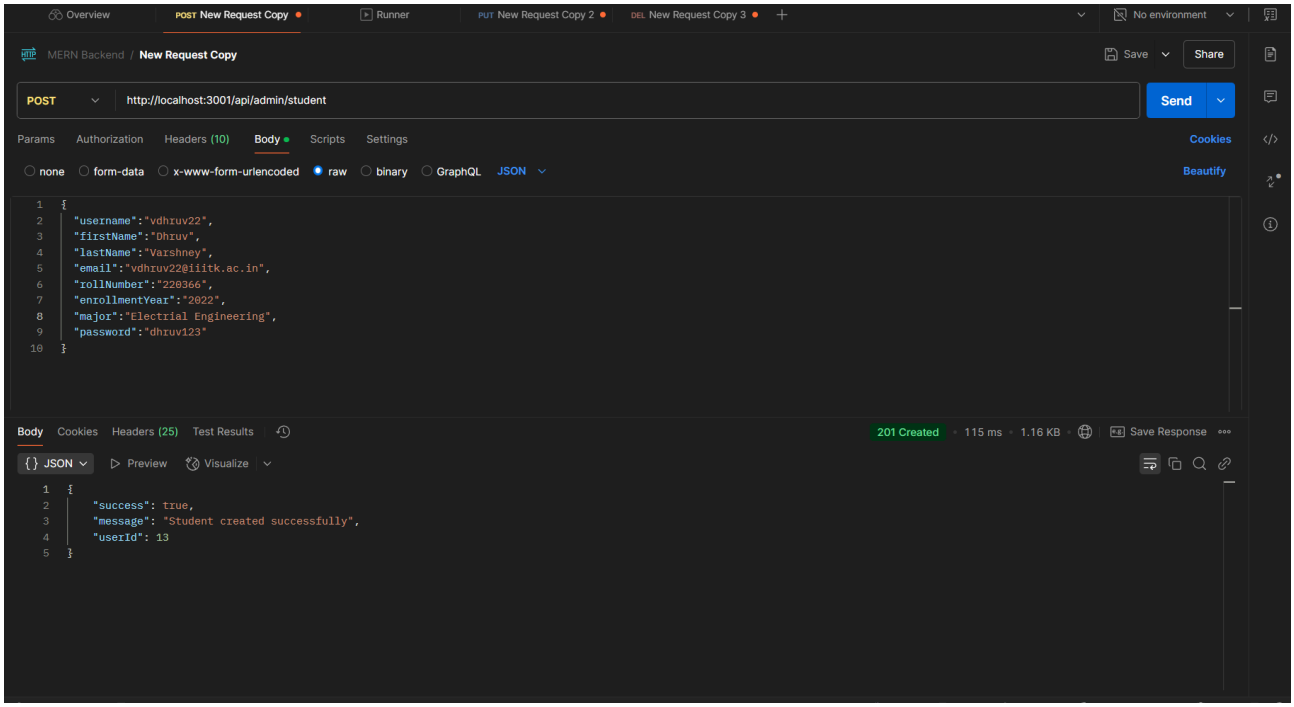
Sharique Ahmad: 221002
Dhruv Varshney: 220366

Test Date:

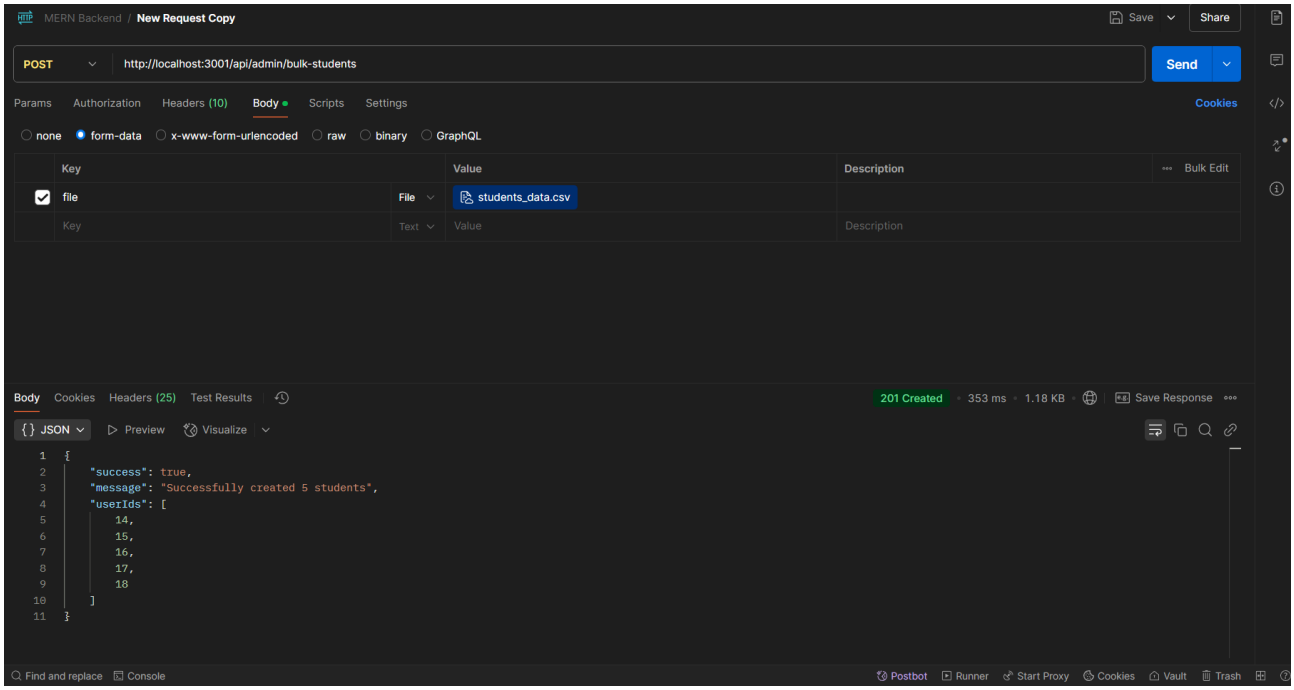
2 April 2025

Test Results:

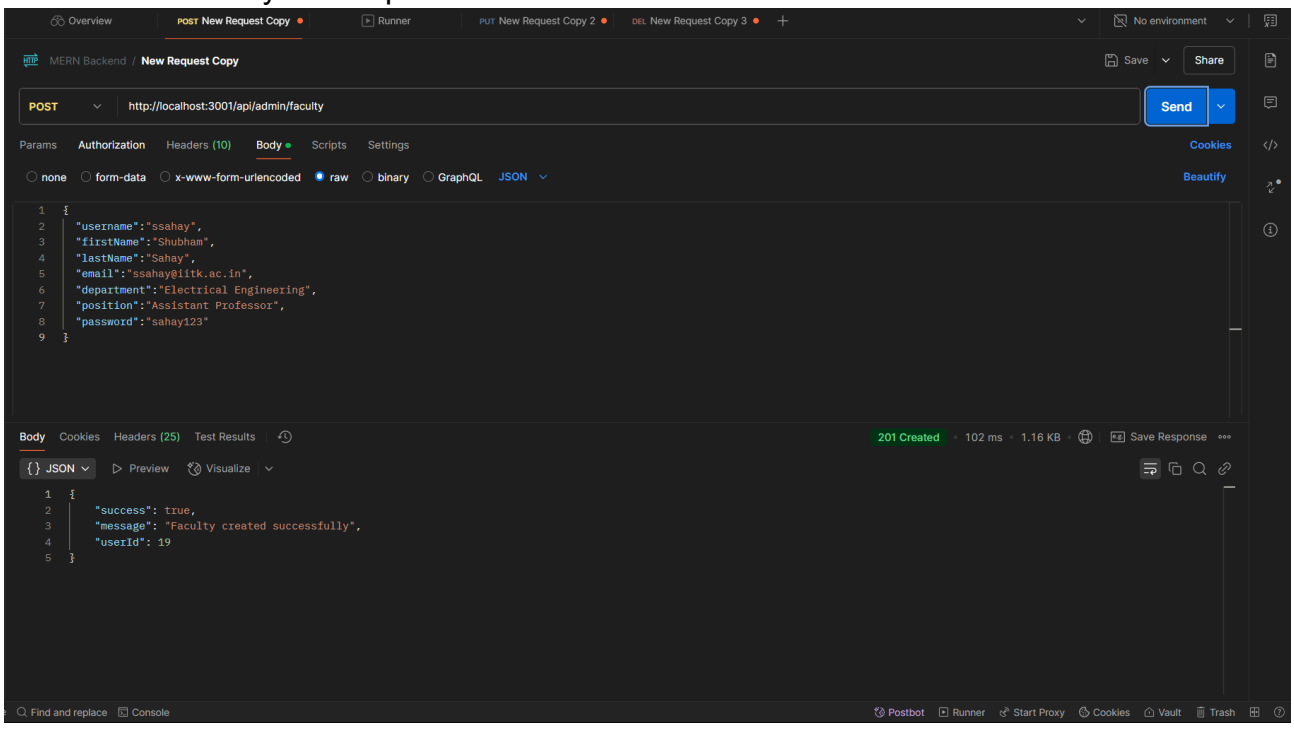
API to add student on the platform:



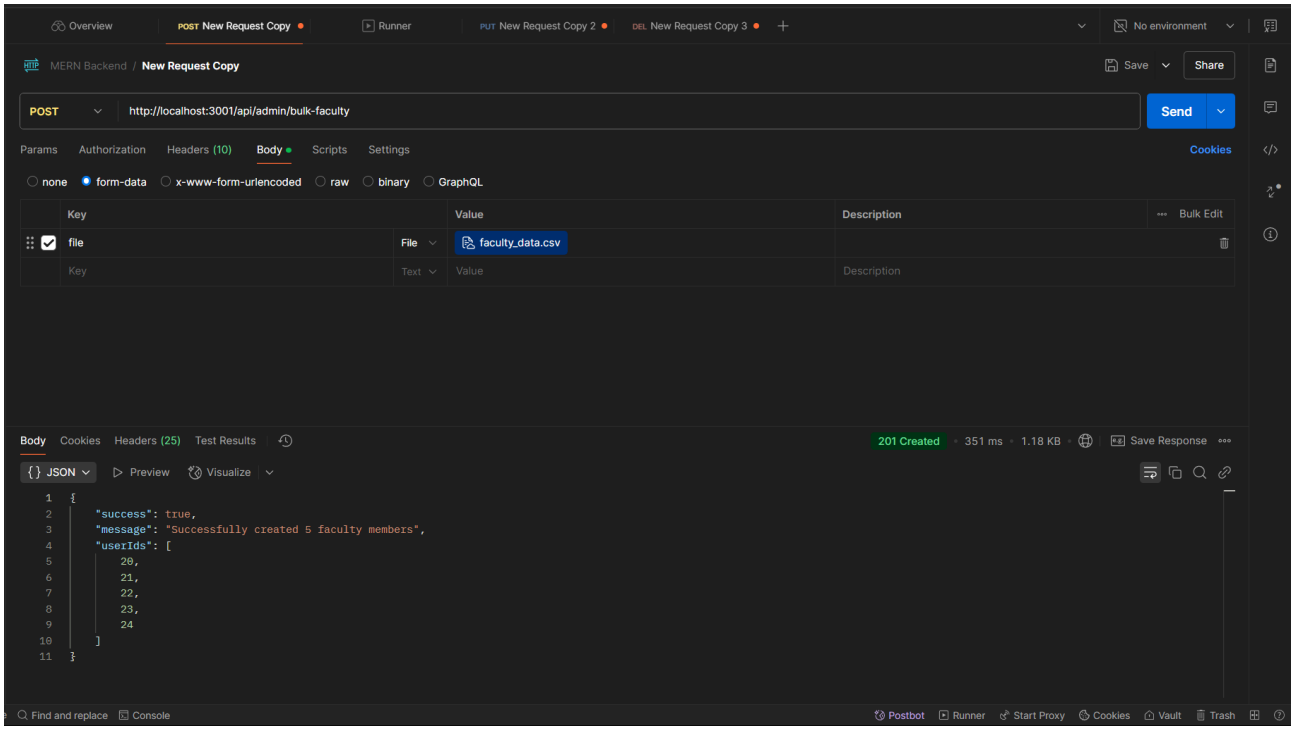
API to add students in bulk on the platform:



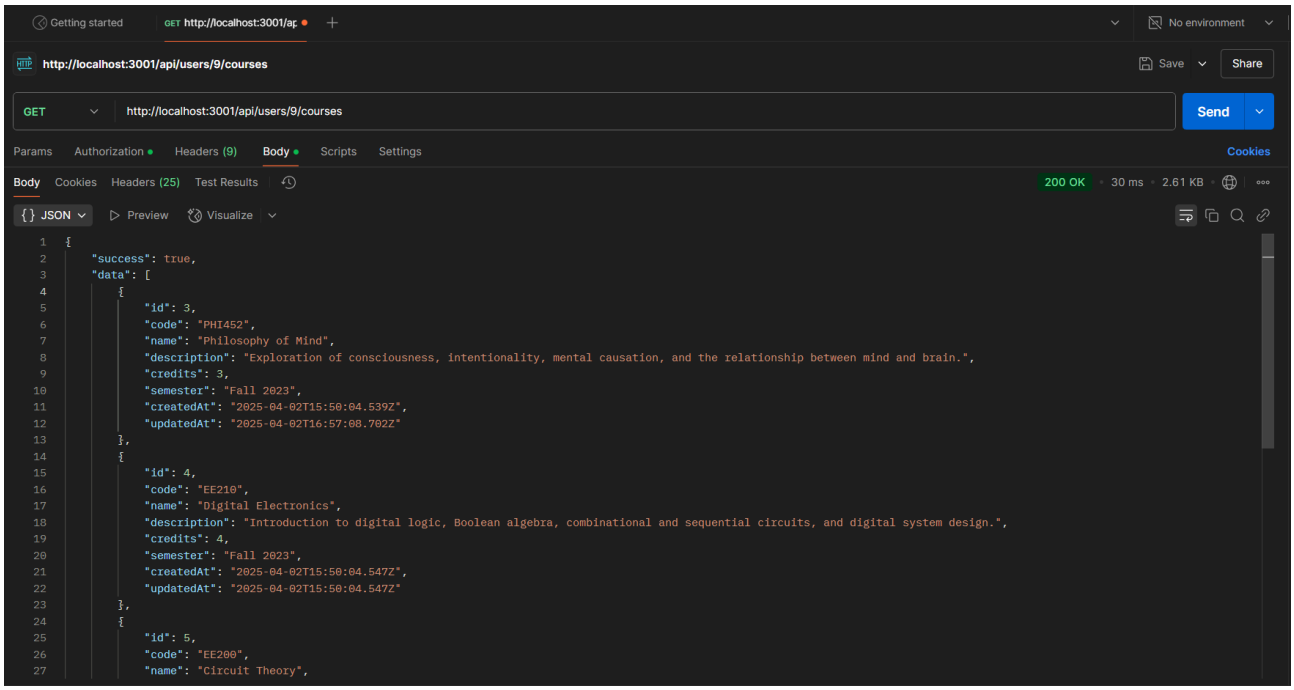
API to add a faculty on the platform:



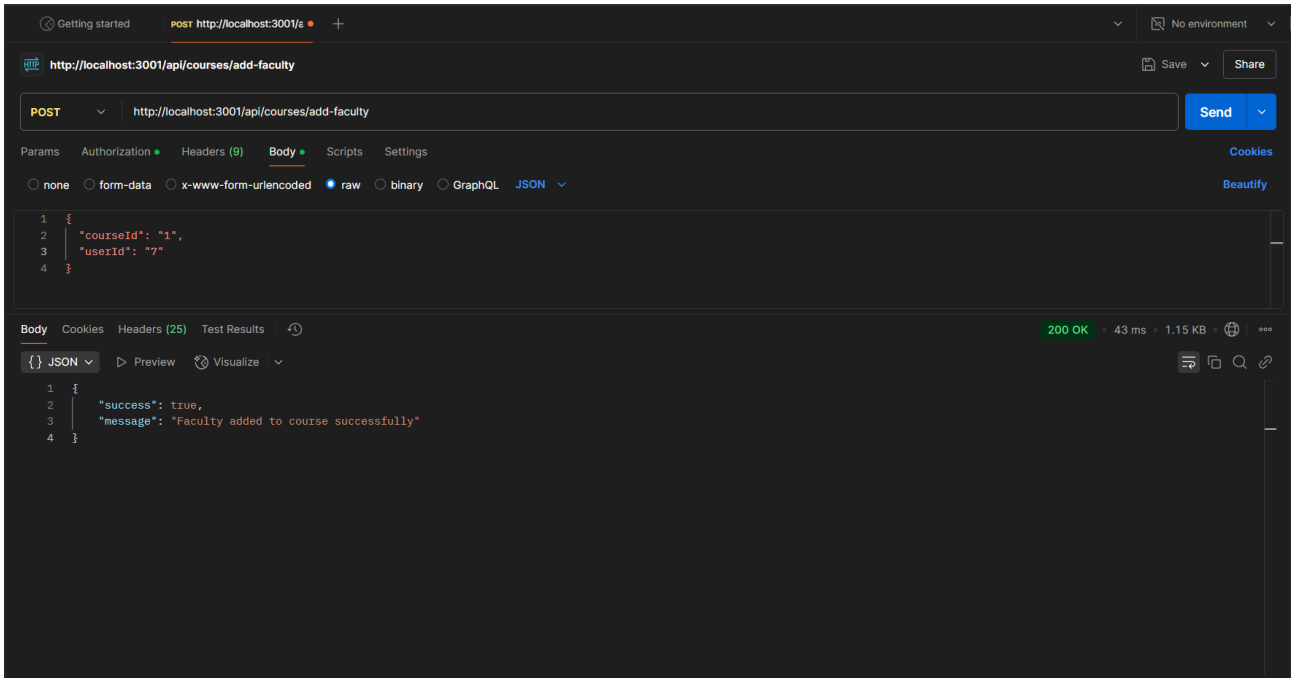
API to add faculty in bulk on the platform:



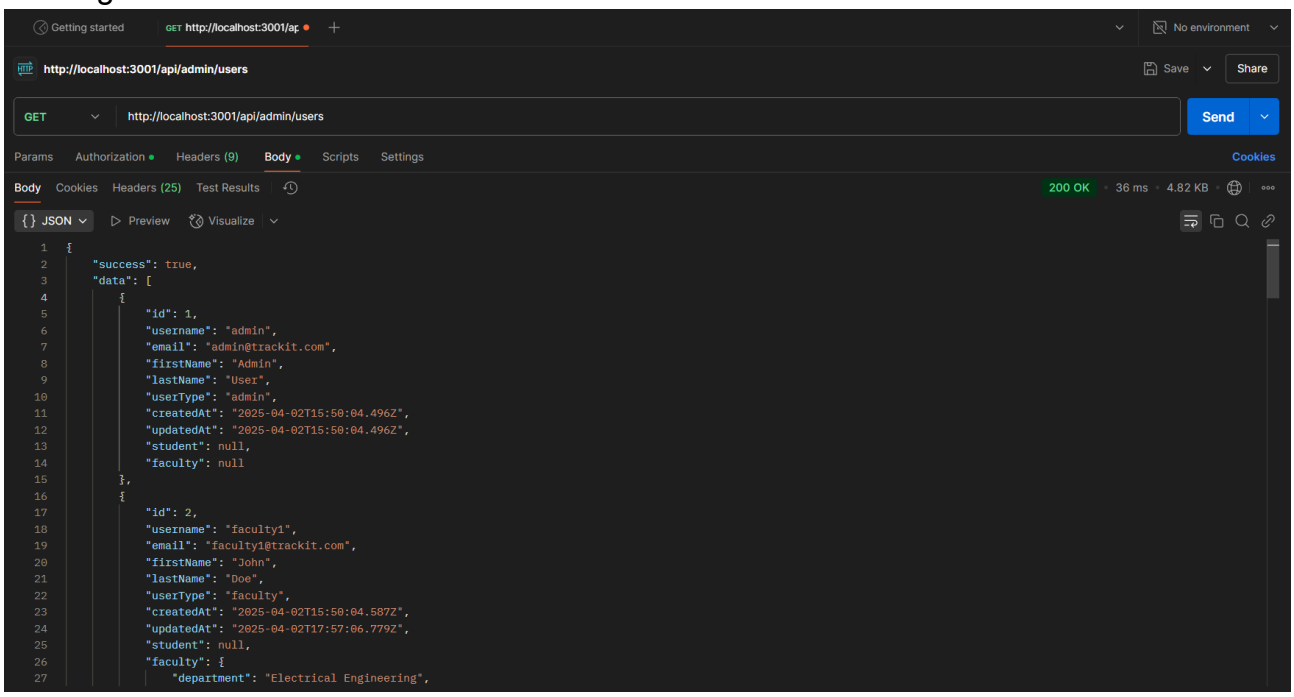
API to get the enrolled Courses of a User.



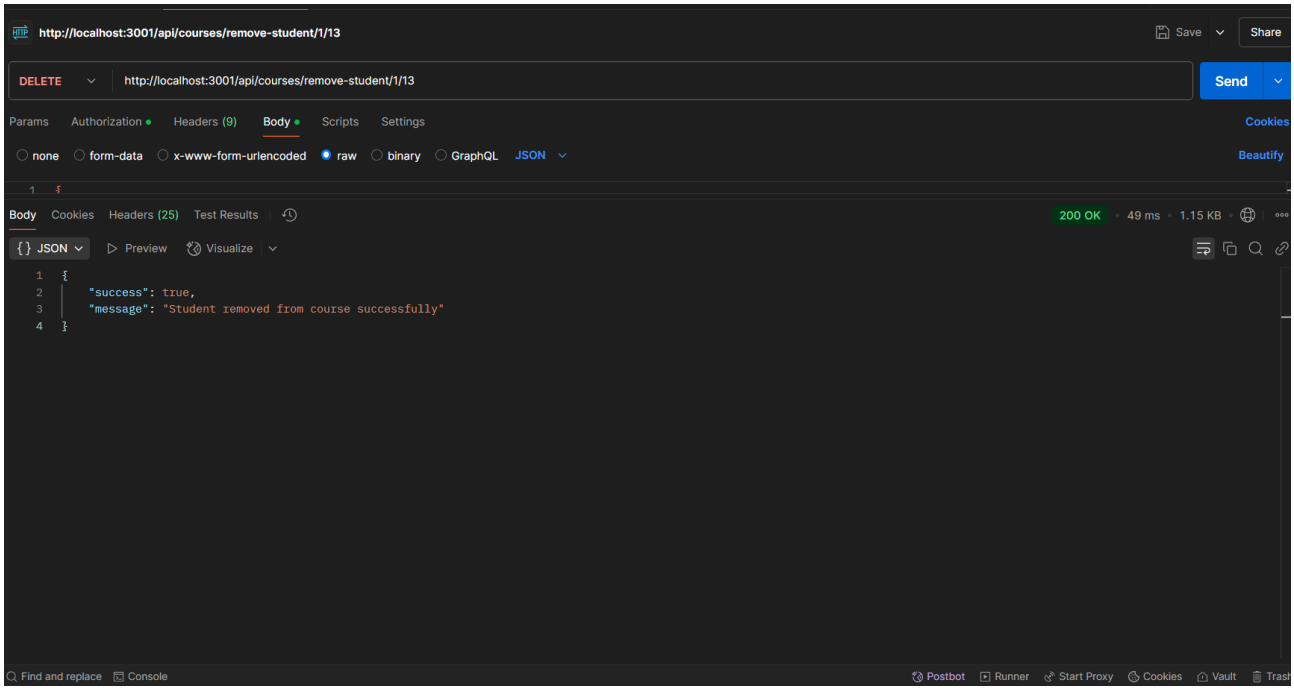
API to add a Faculty in a Course.



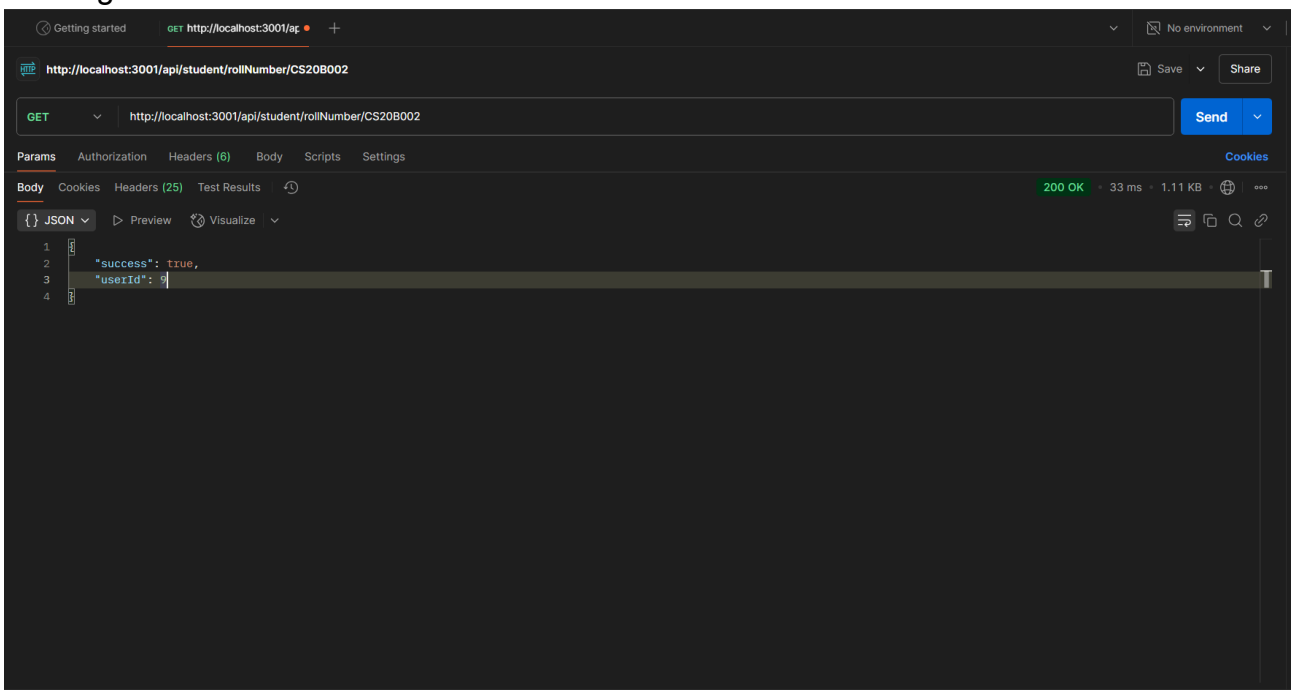
API to get all the users.



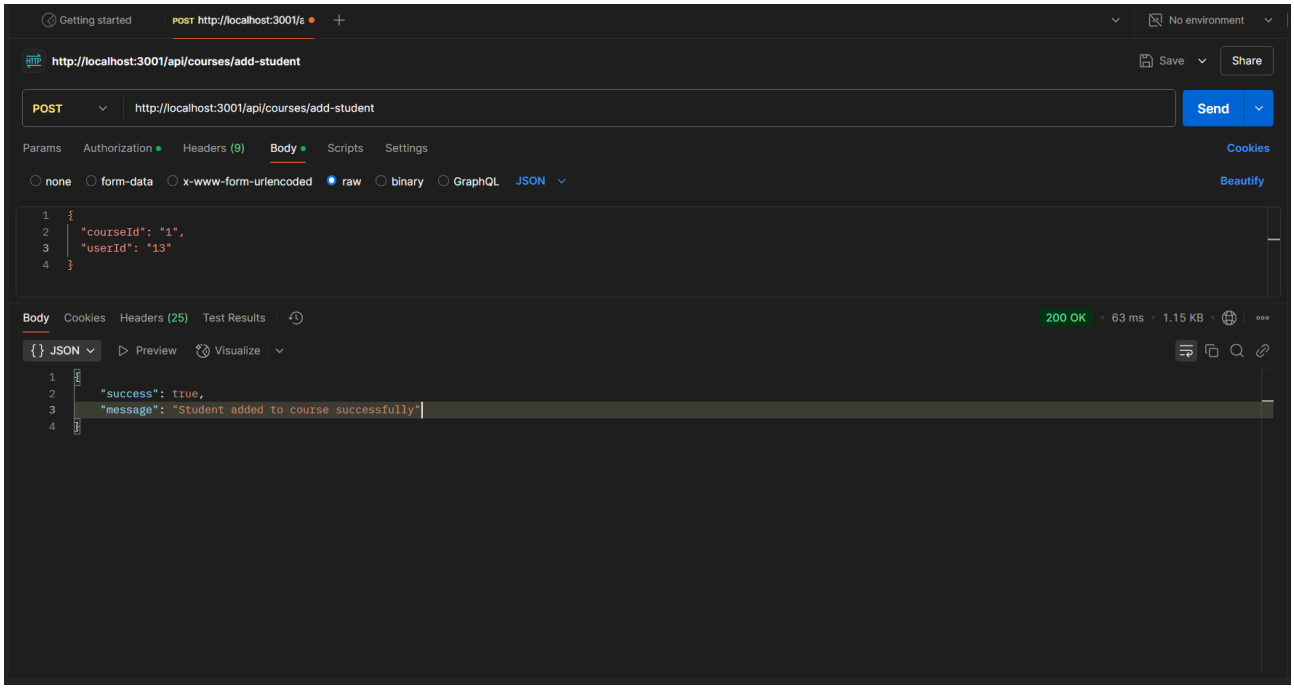
API for Removing a Student from a Course



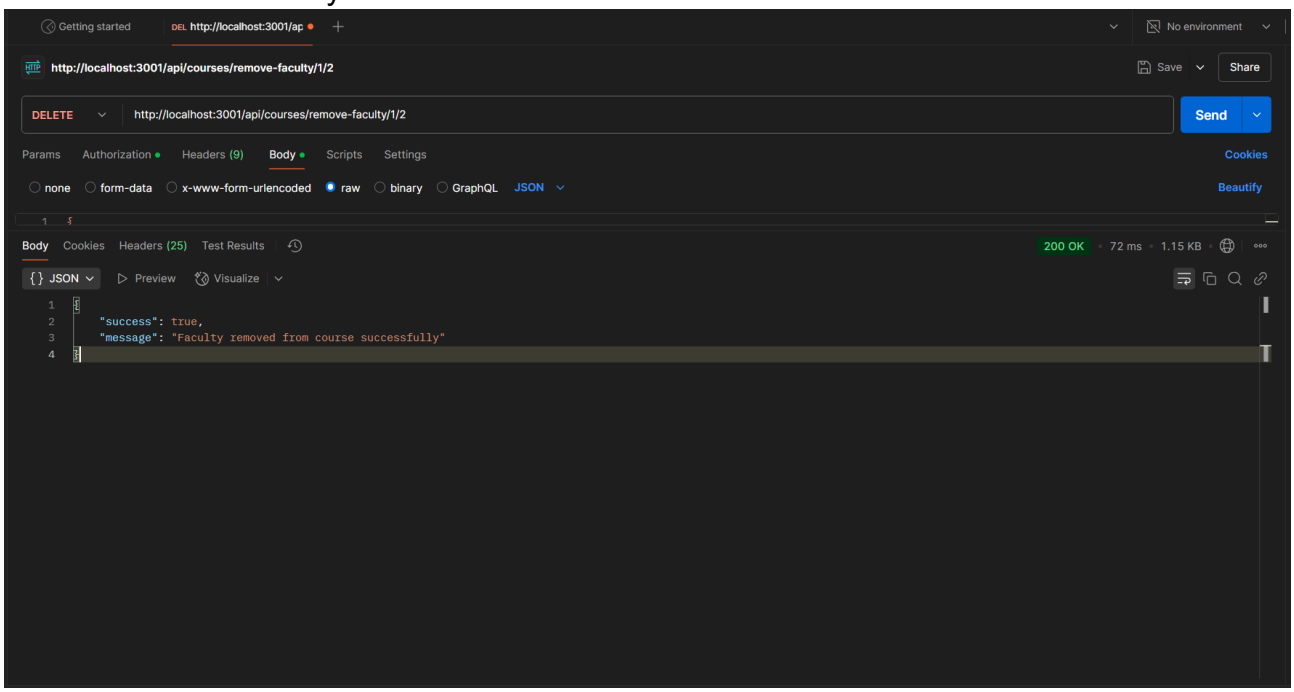
API to get UserID from Roll Number.



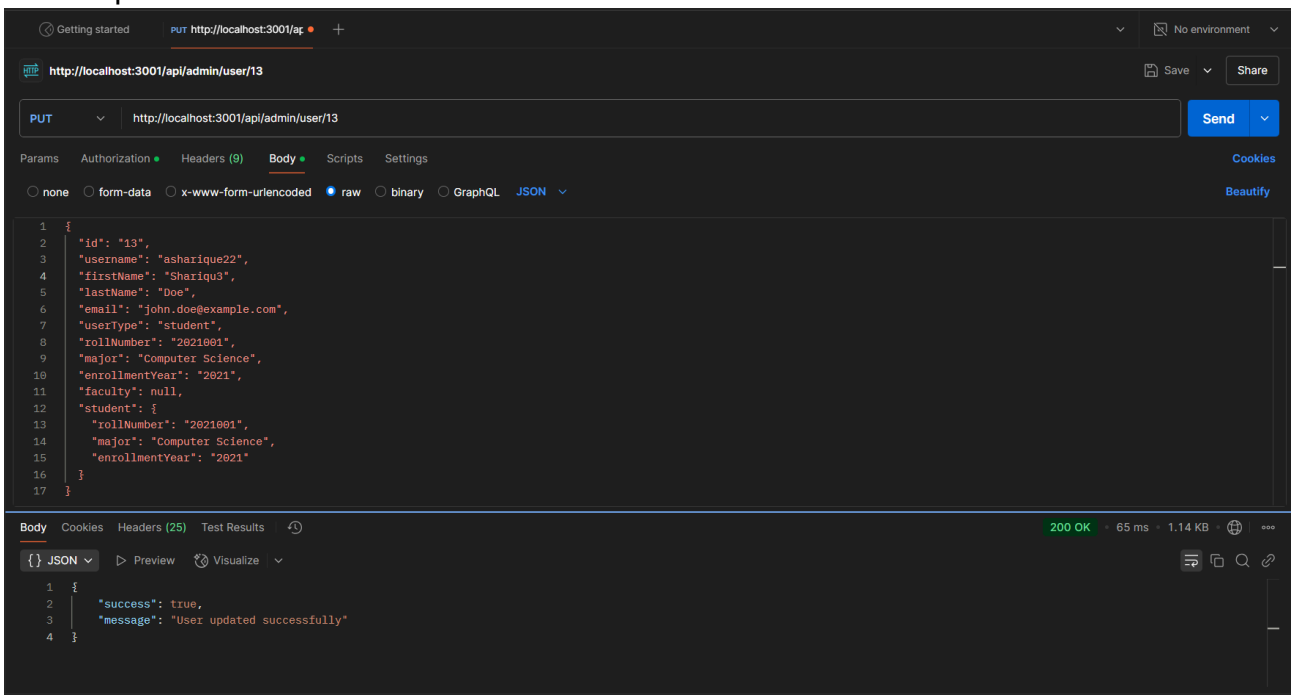
API to add Student in a Course



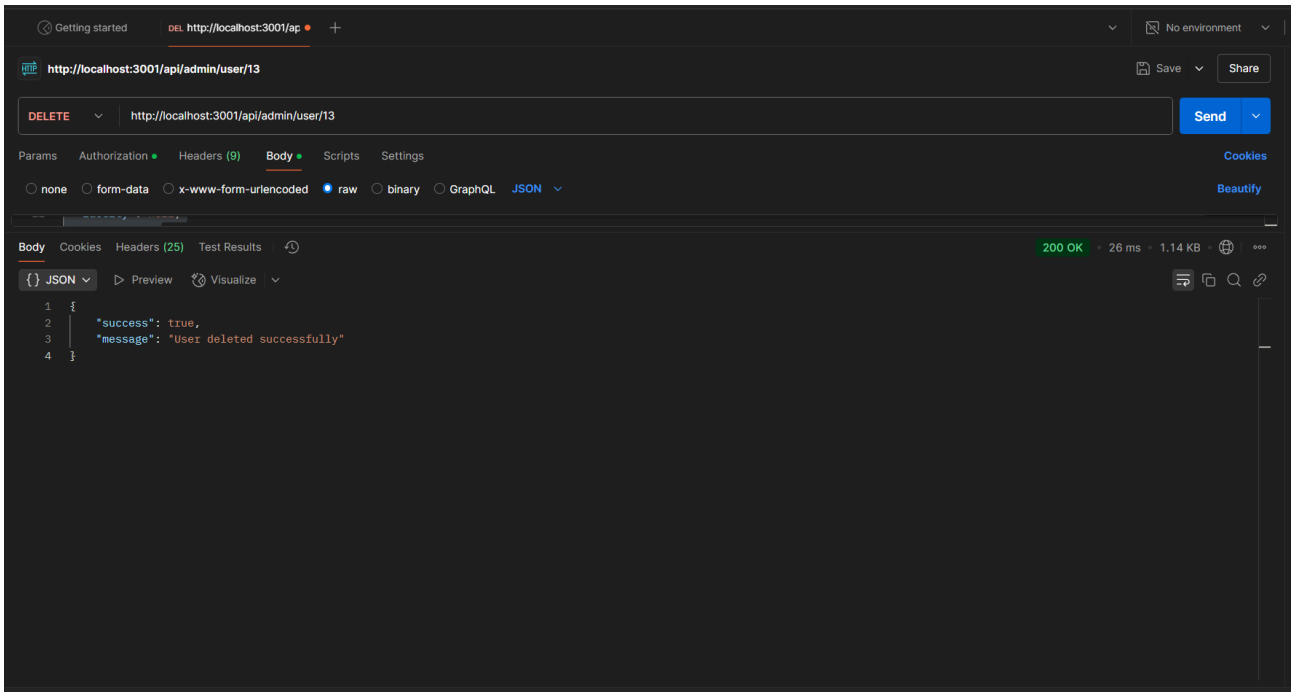
API to remove a Faculty from a Course.



API to update the user details.



API to delete a User



Structural Coverage:

The unit is working well and fulfils 100% of the requirements of the SRS

Admin: Courses

Unit Details:

This includes testing endpoints for creating, retrieving, updating, and deleting courses. The tests ensured proper validation of input data, appropriate error handling, and correct responses for both successful and failure scenarios.

Test Owner:

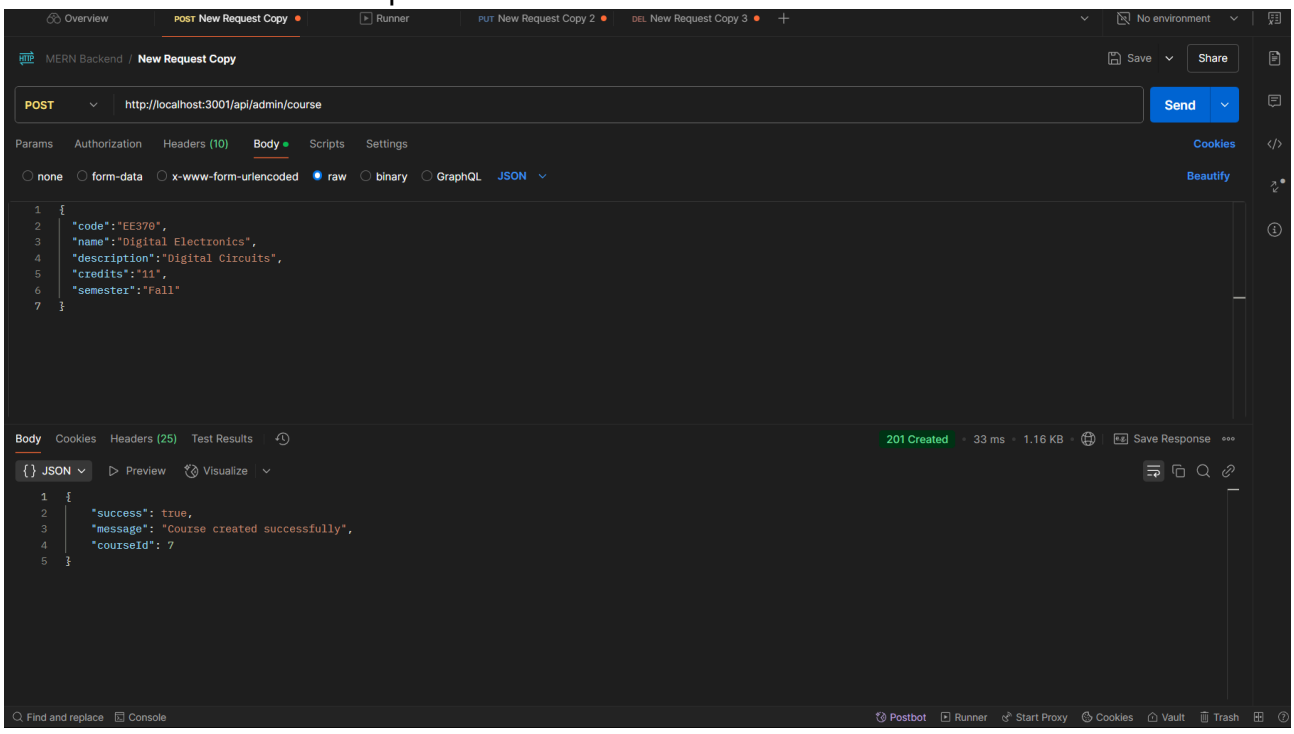
Sharique Ahmad: 221002
Dhruv Varshney: 220366

Test Date:

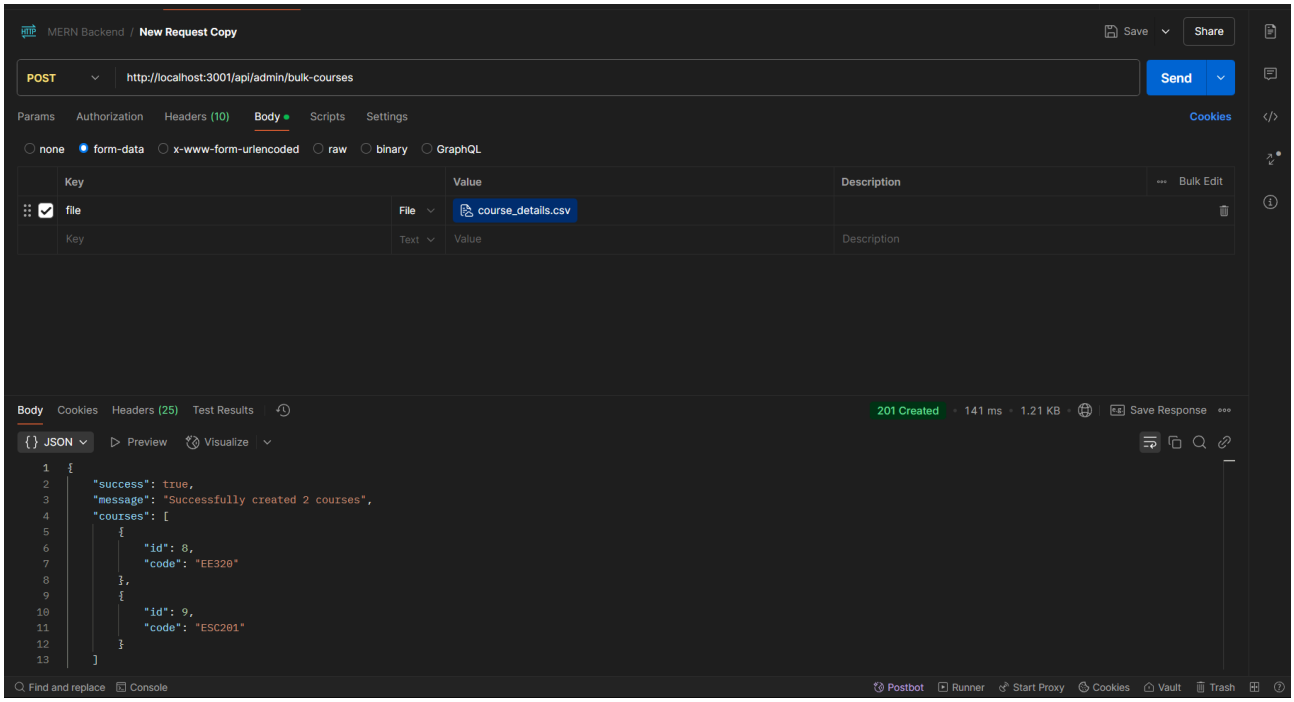
3 April 2025

Test Results:

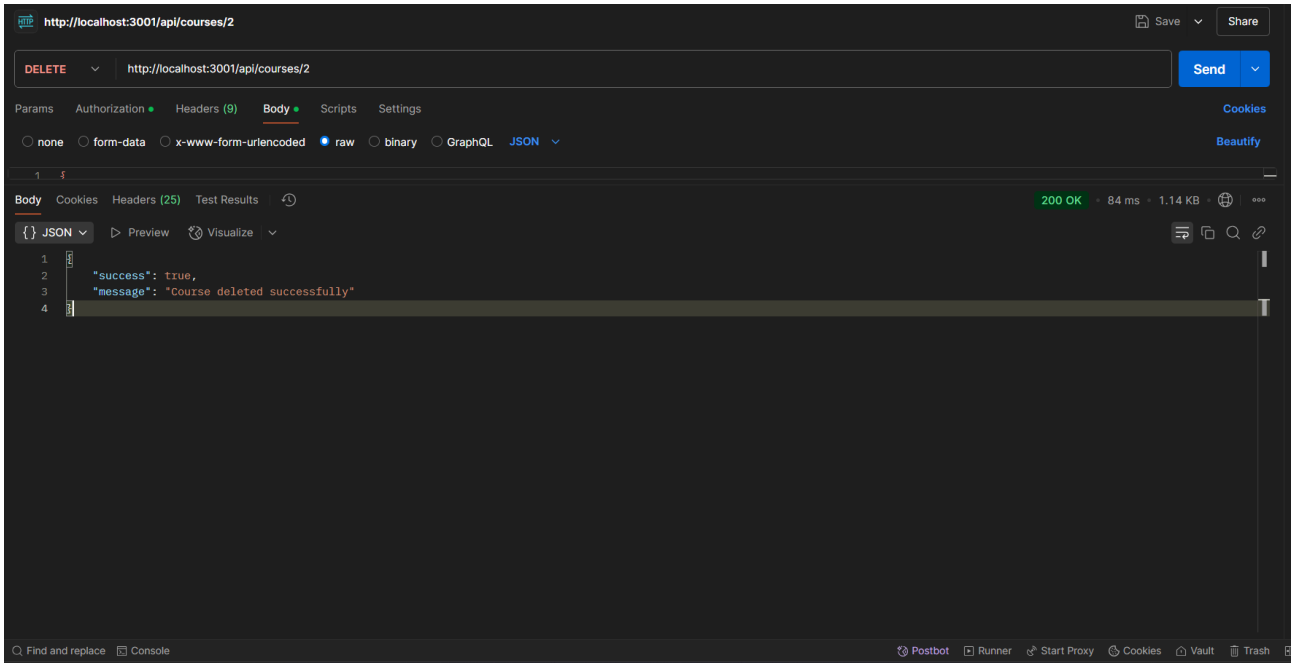
API to add a course on the platform:



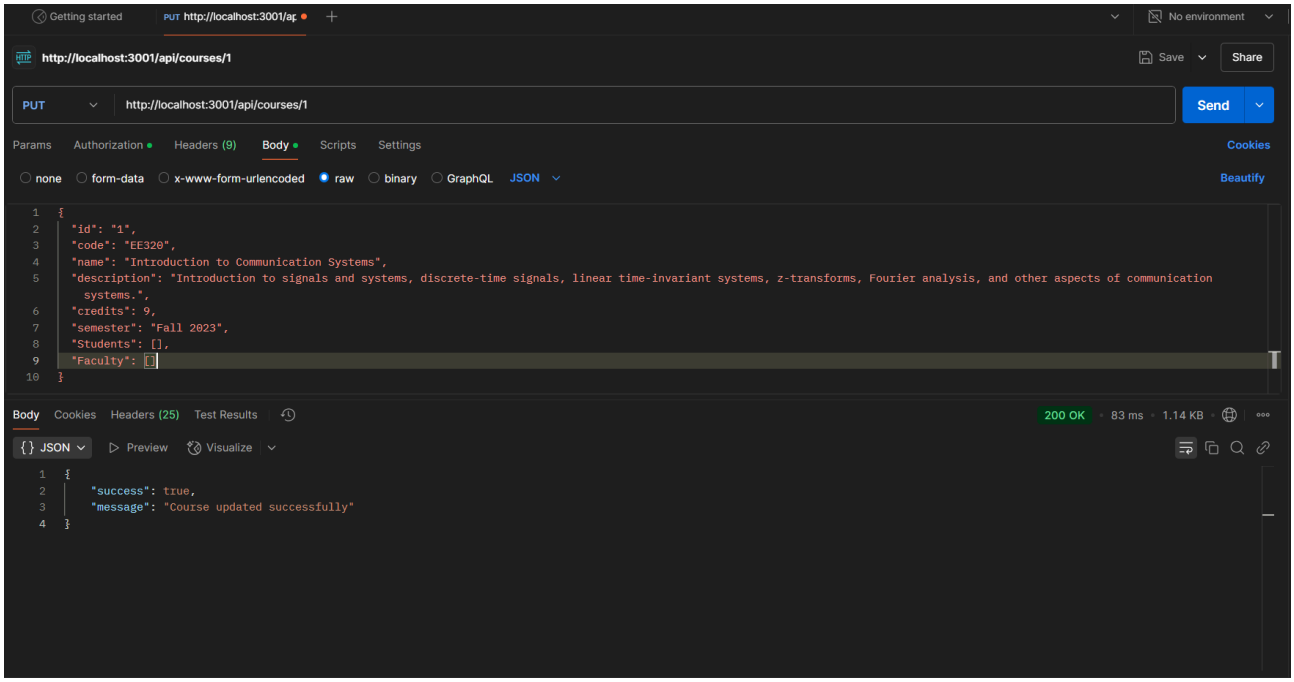
API to add courses in bulk on the platform:



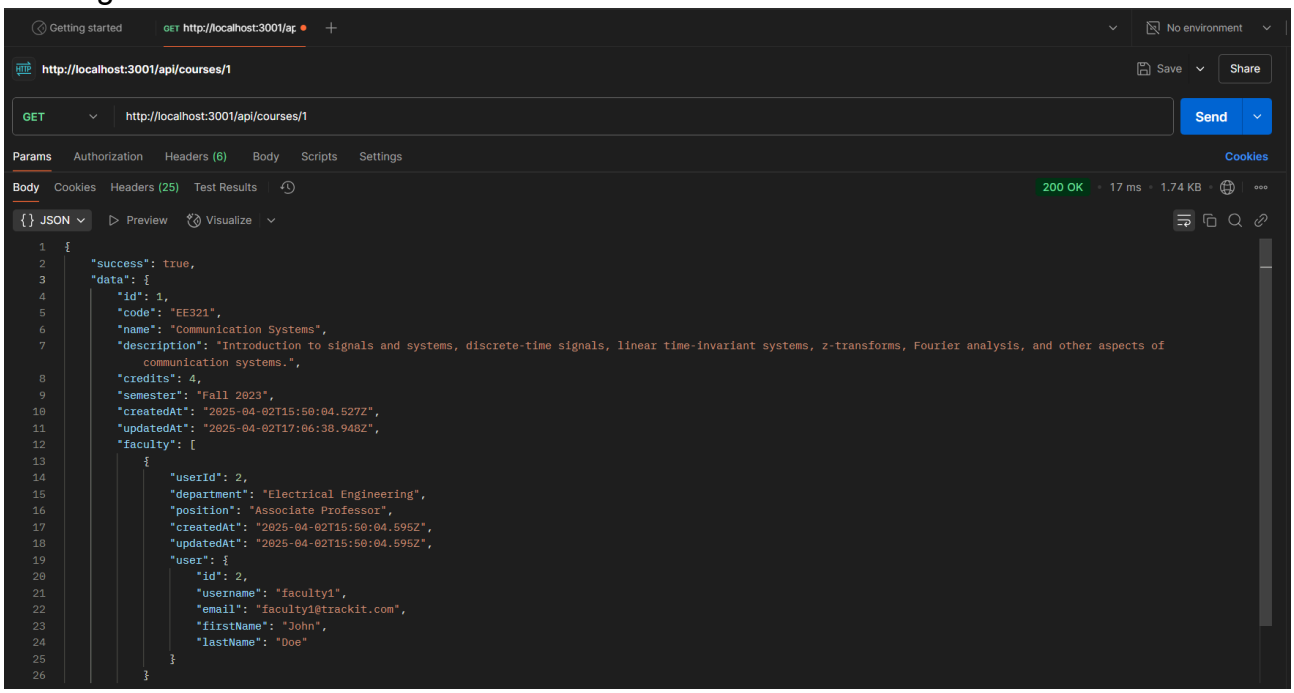
API to delete a Course.



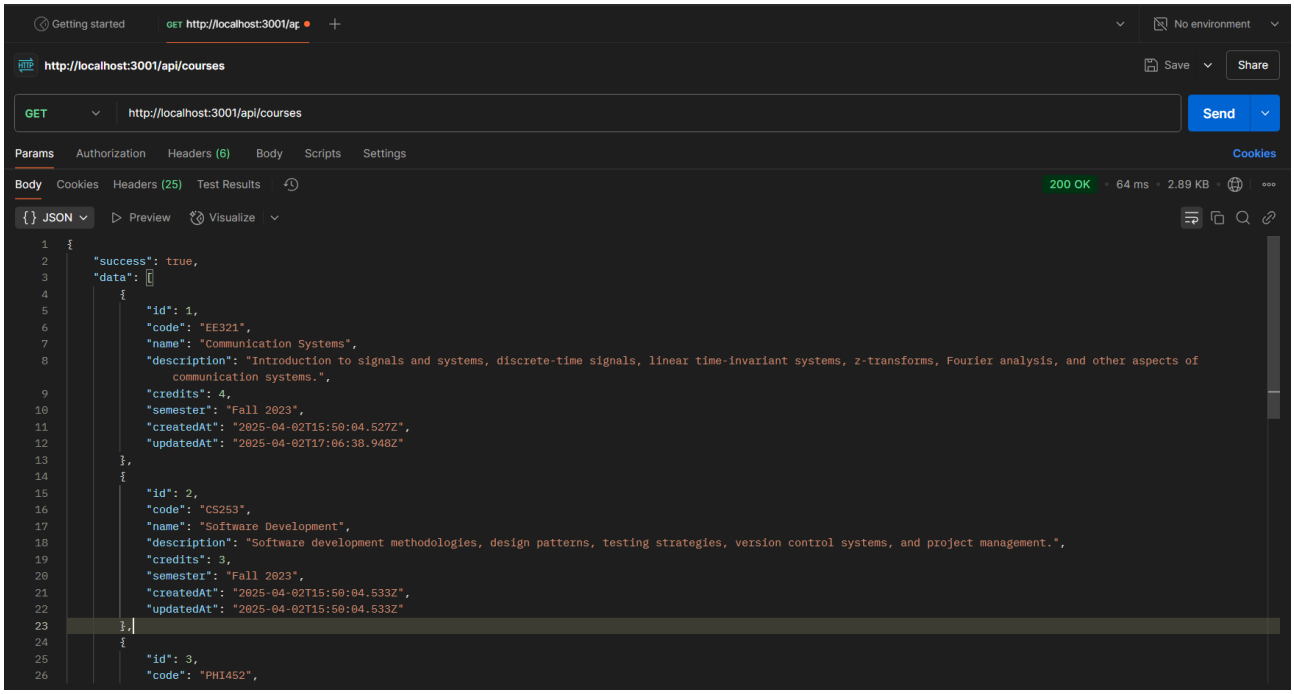
API to update the Course details.



API to get course details.



API to get all courses.



Events

Unit Details:

Involves the testing of the Events Database, APIs and Static Frontend.

Test Owner:

Aayush Singh, 220024

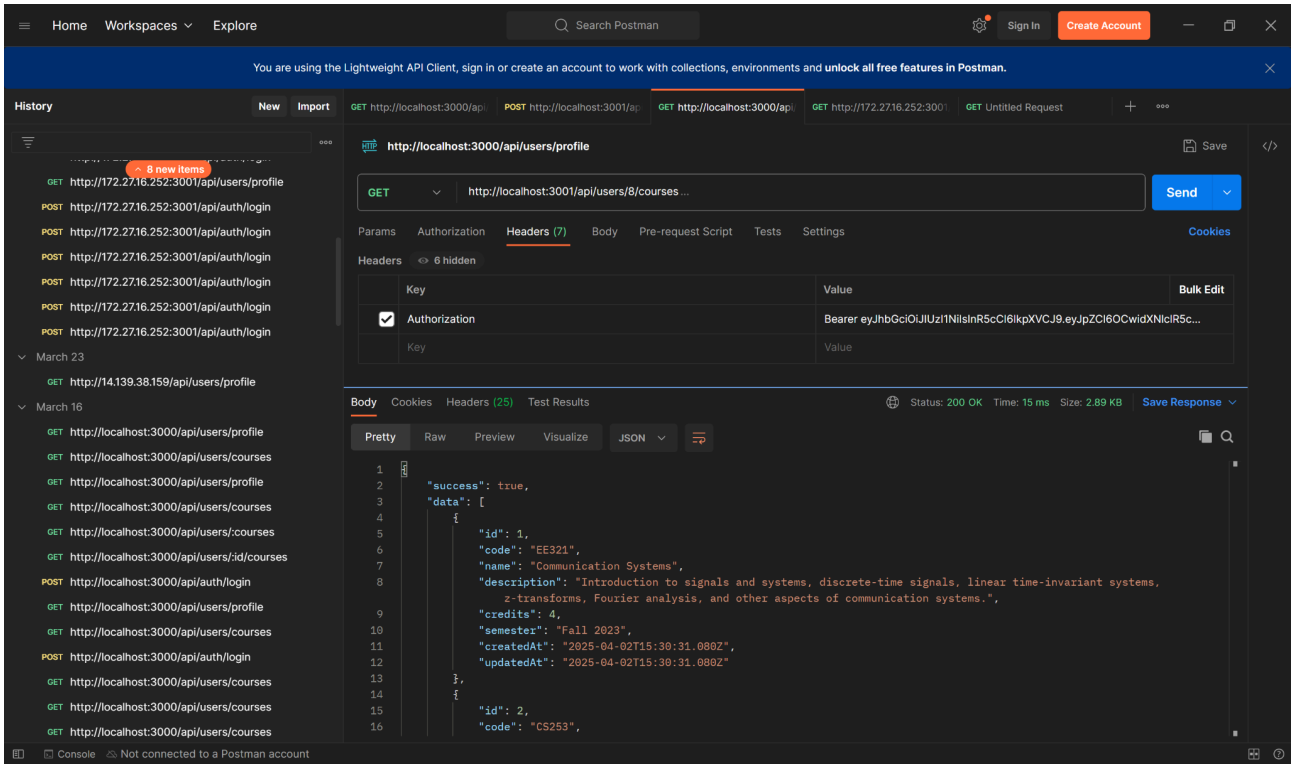
Test Date:

2 April 2025

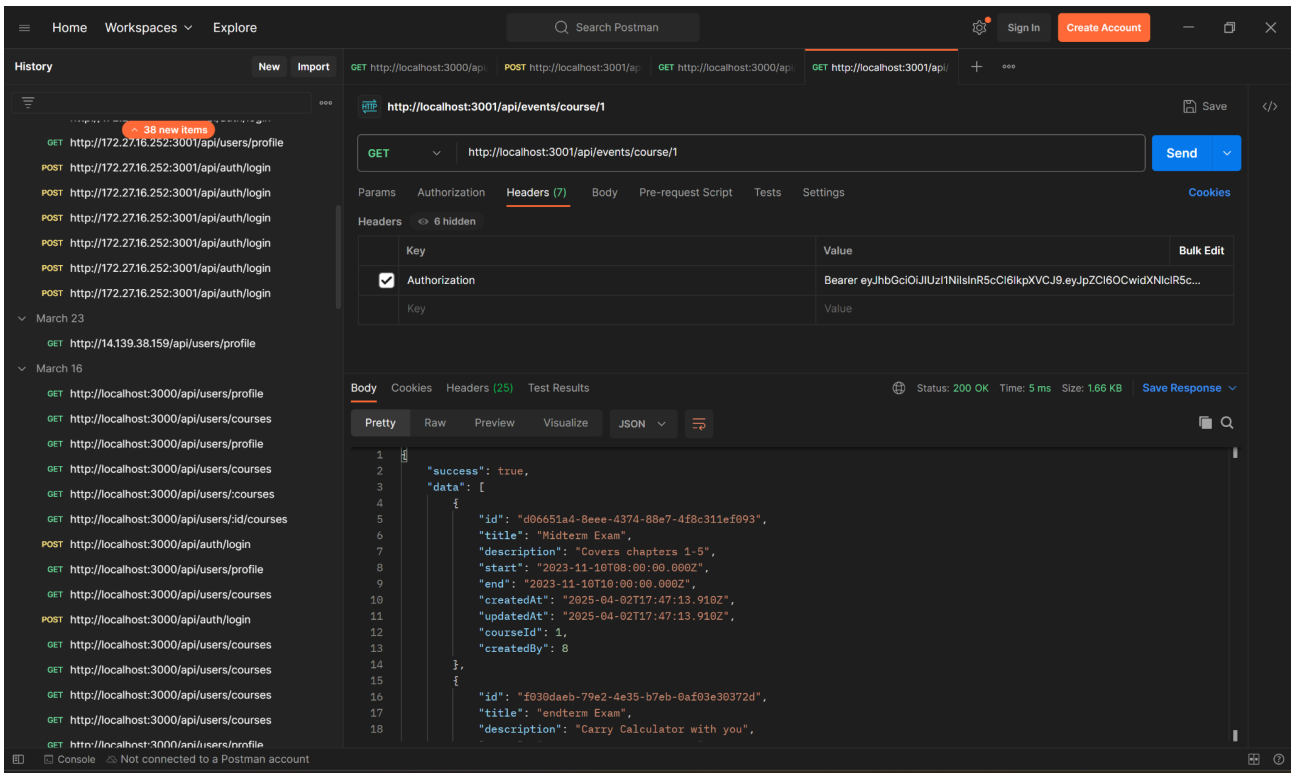
Test Results:

First, Testing the APIs required in Events.

API to fetch all the courses a user is enrolled in:
It will only work when we provide token(authorization that a user is signed in)

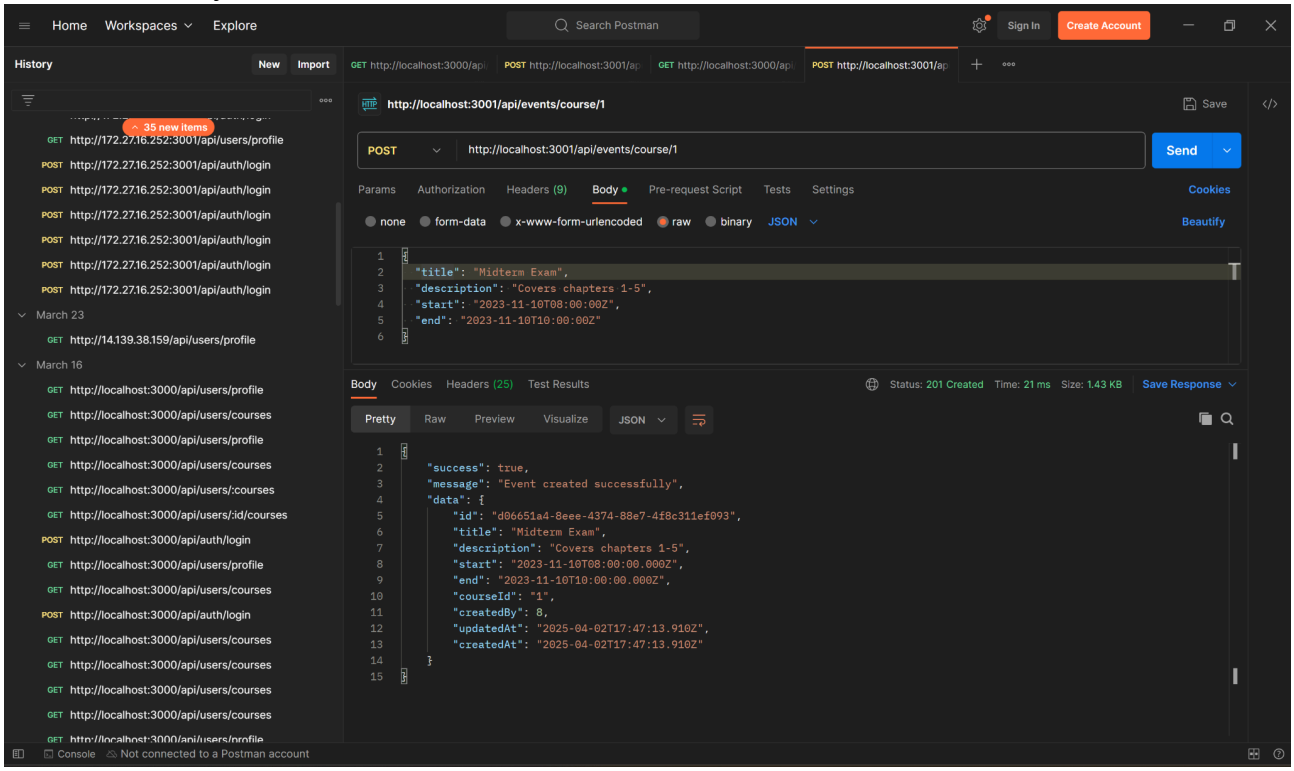


API to get events for a particular course:
This will also work with authorization

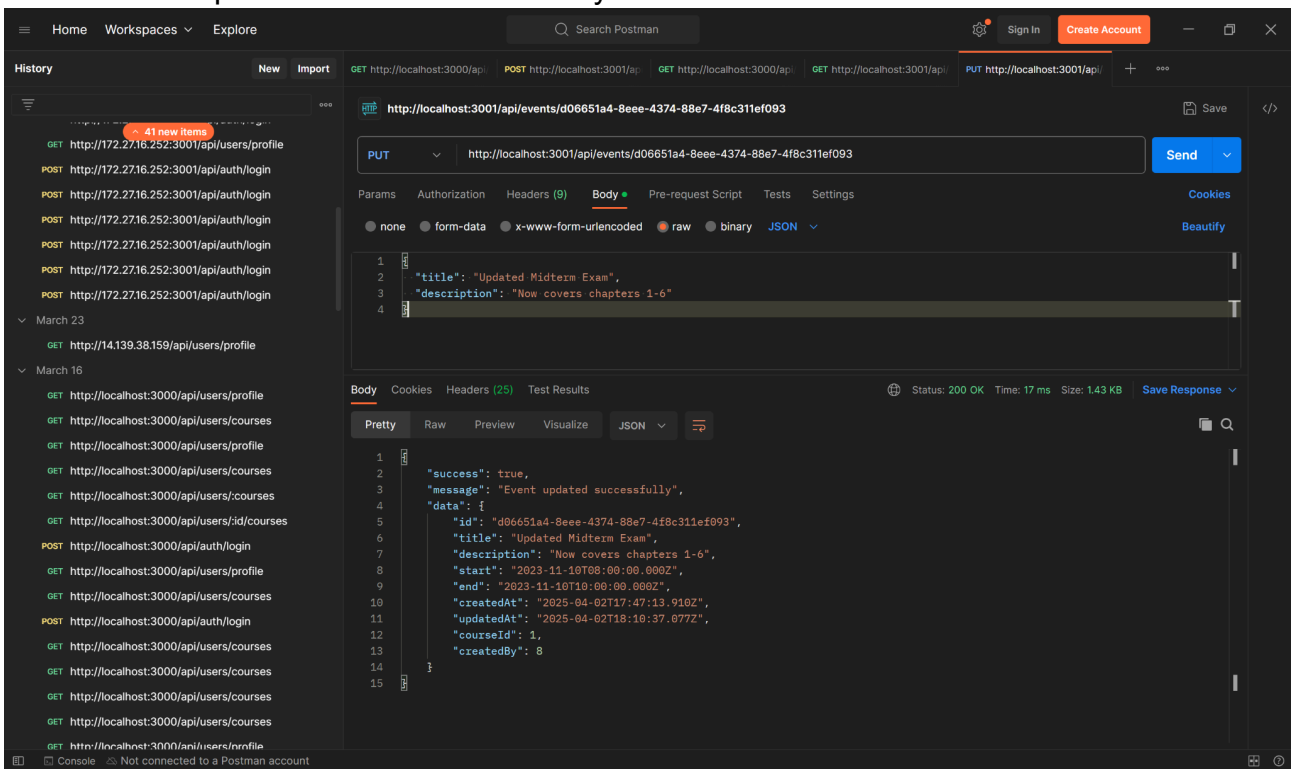


API to add events for a course by a faculty:

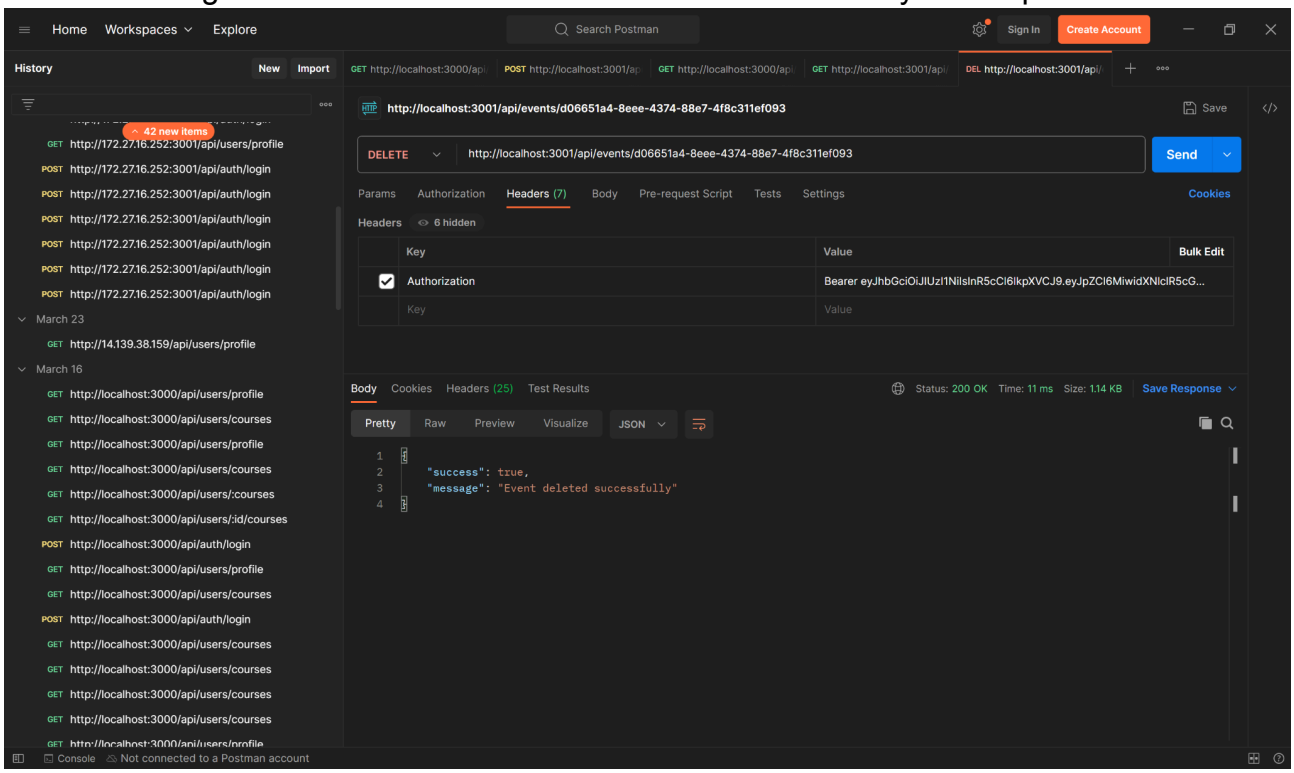
It has an authorization check for faculty of that particular course, also we would provide the data in the body of the API call in JSON format.



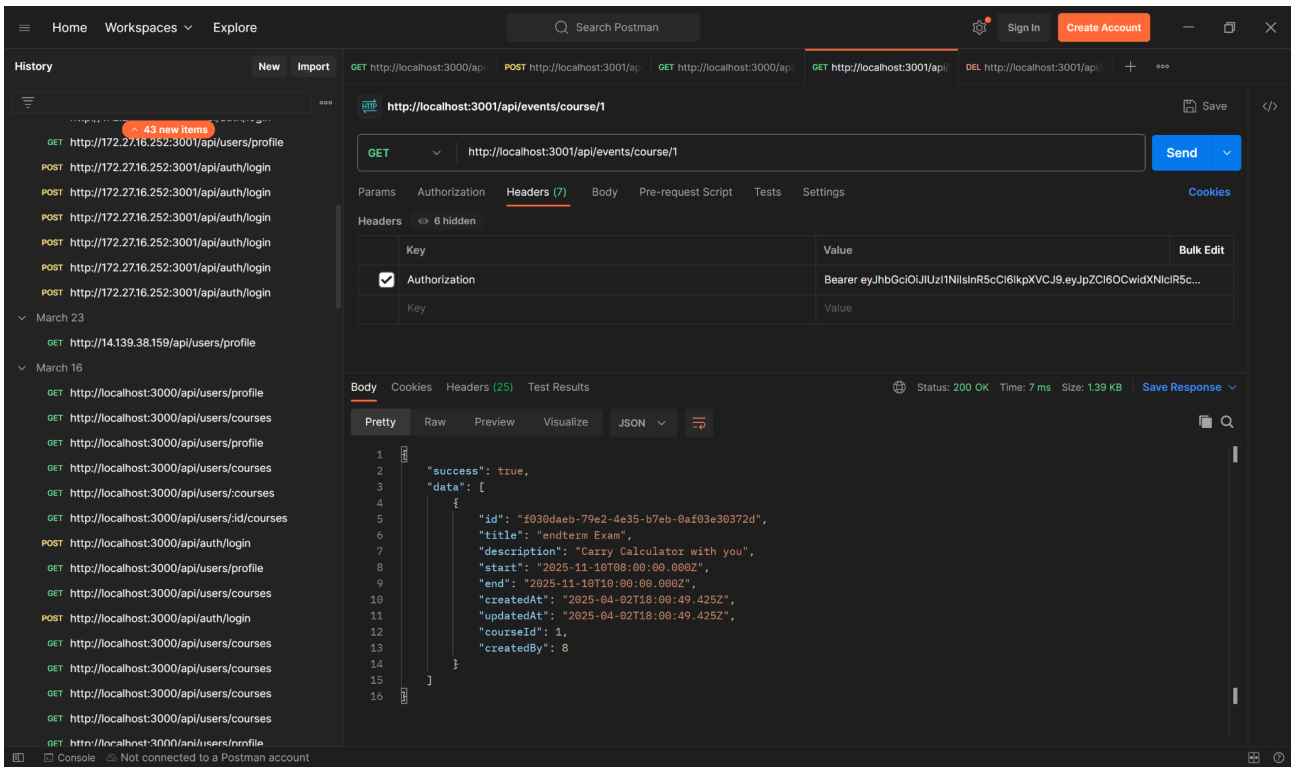
API for updating an event: It has an authorization check for faculty of that particular course. Also we would provide the data in the body of the API call in JSON format.



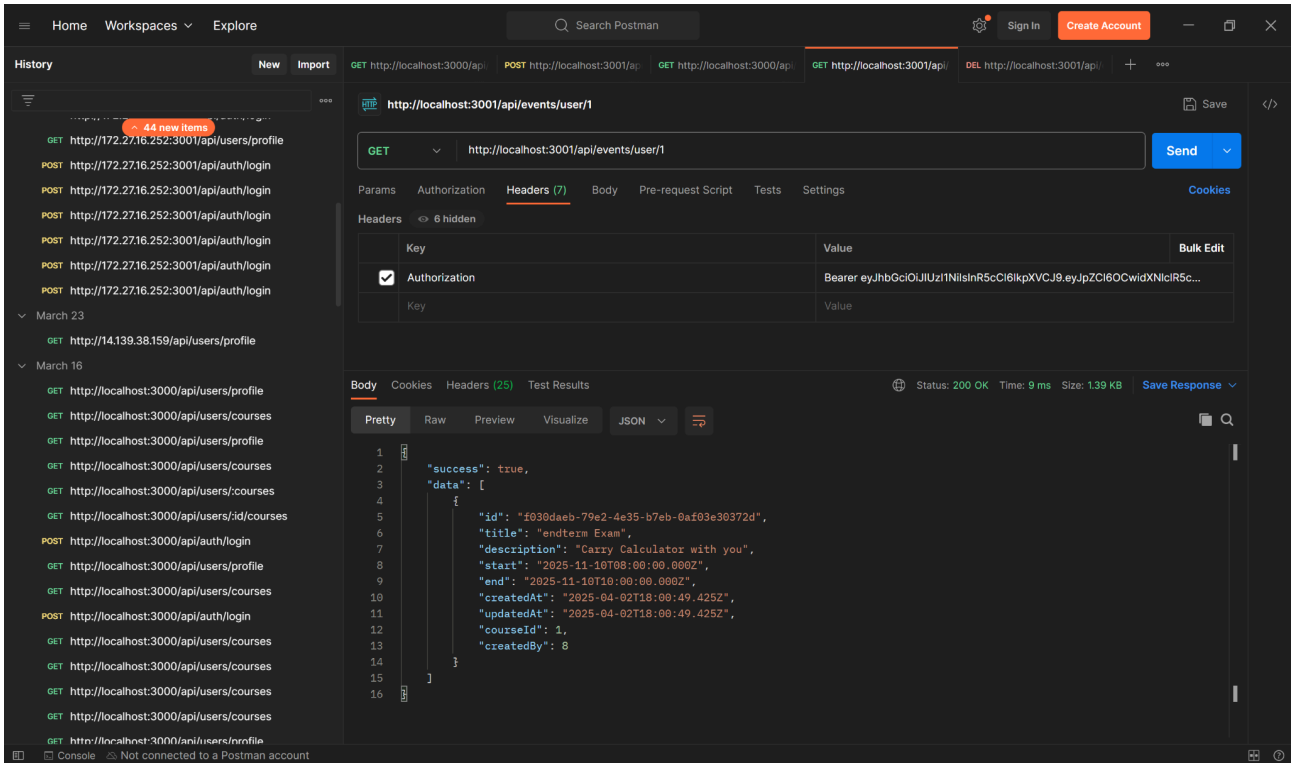
API for deleting an event:It has an authorization check for faculty of that particular course.



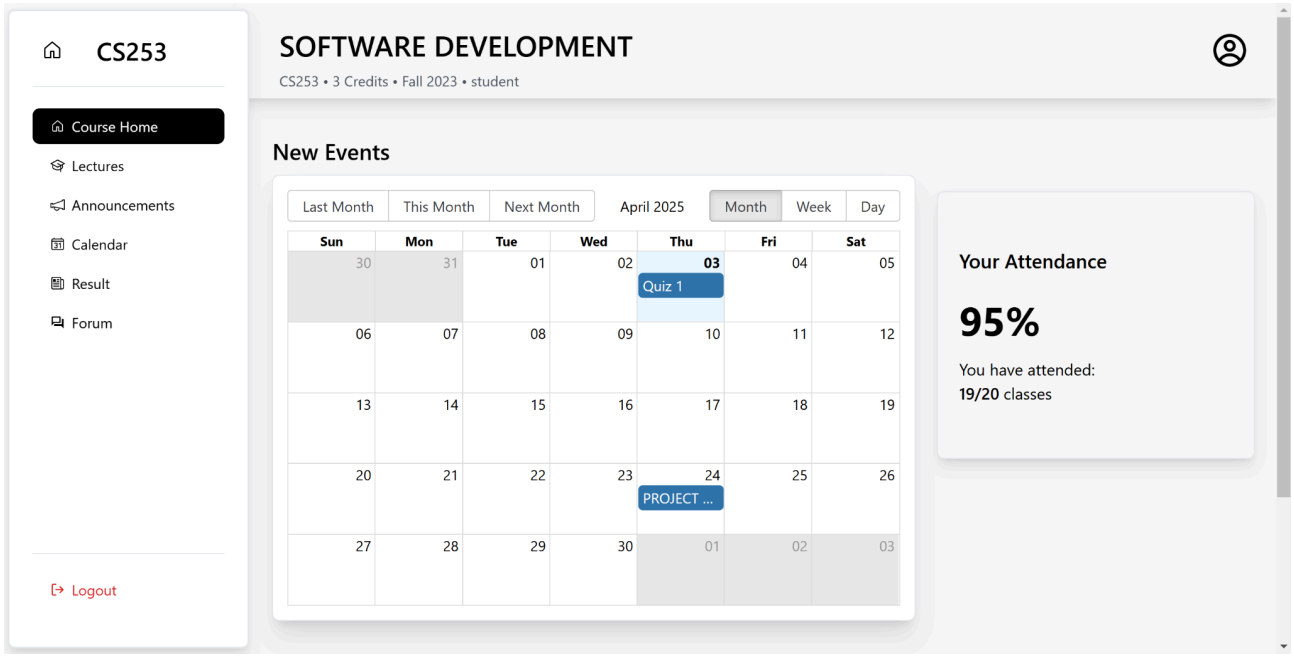
Deleted Event doesn't exist now: Works as intended.



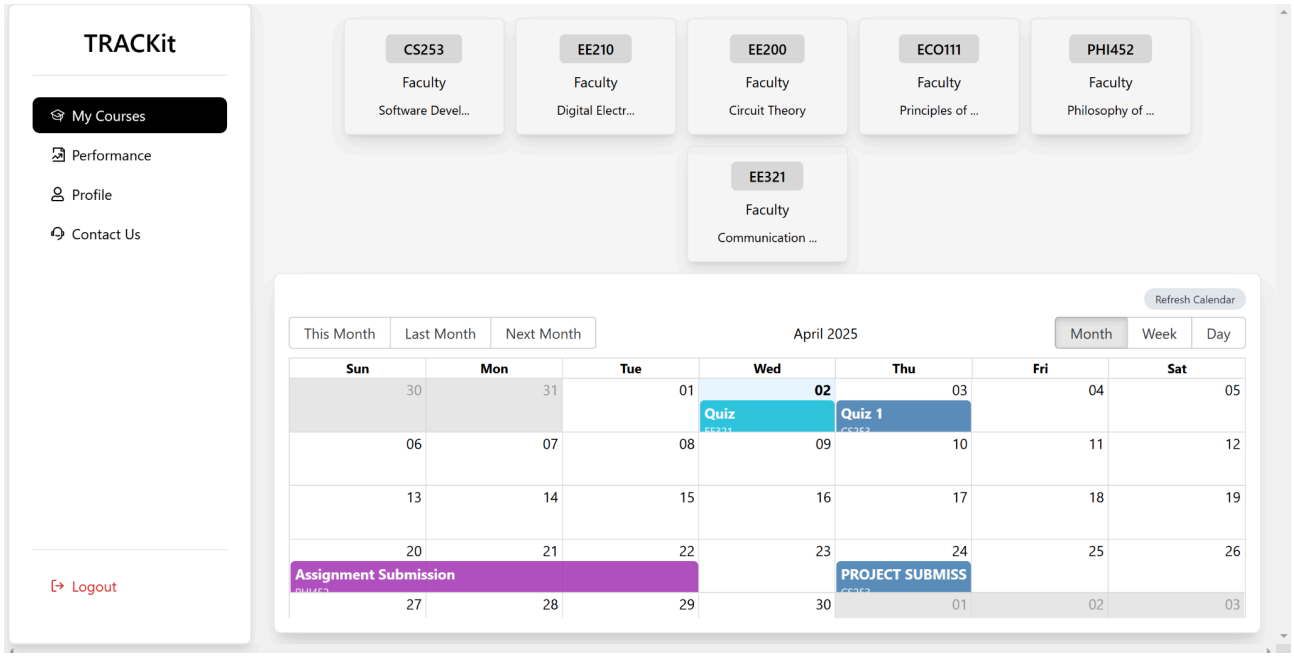
API for getting all events of a particular user: This API call is useful in the main dashboard calendar.

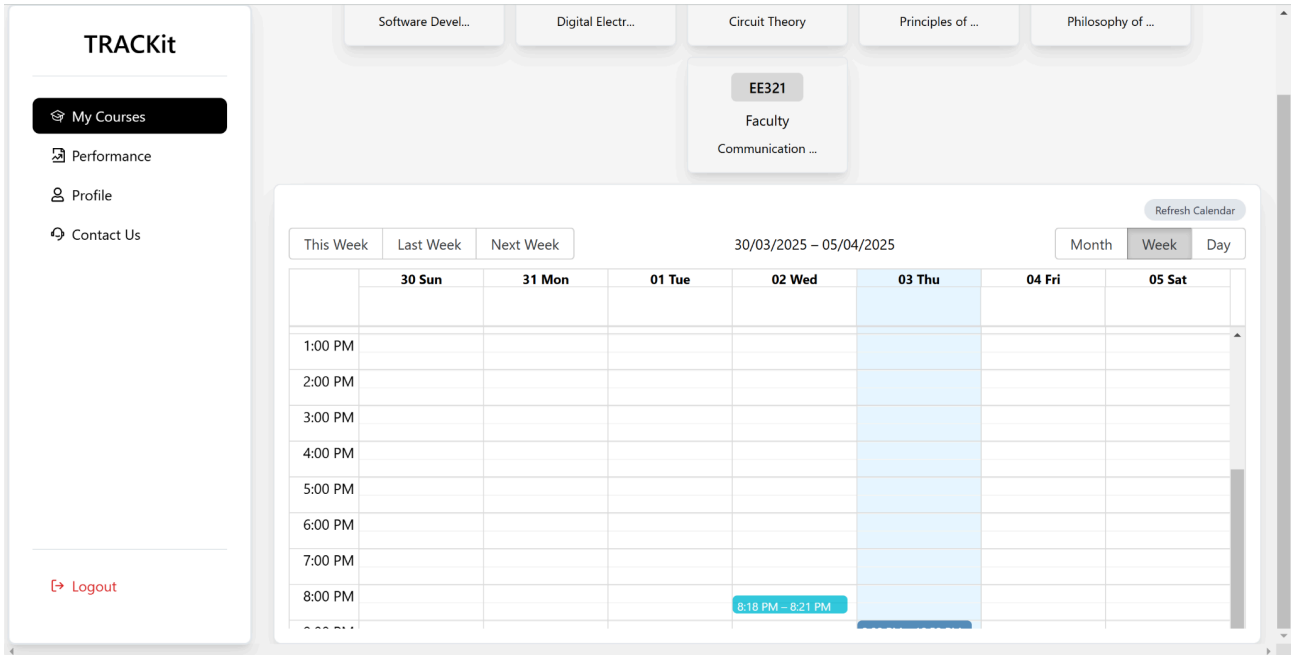


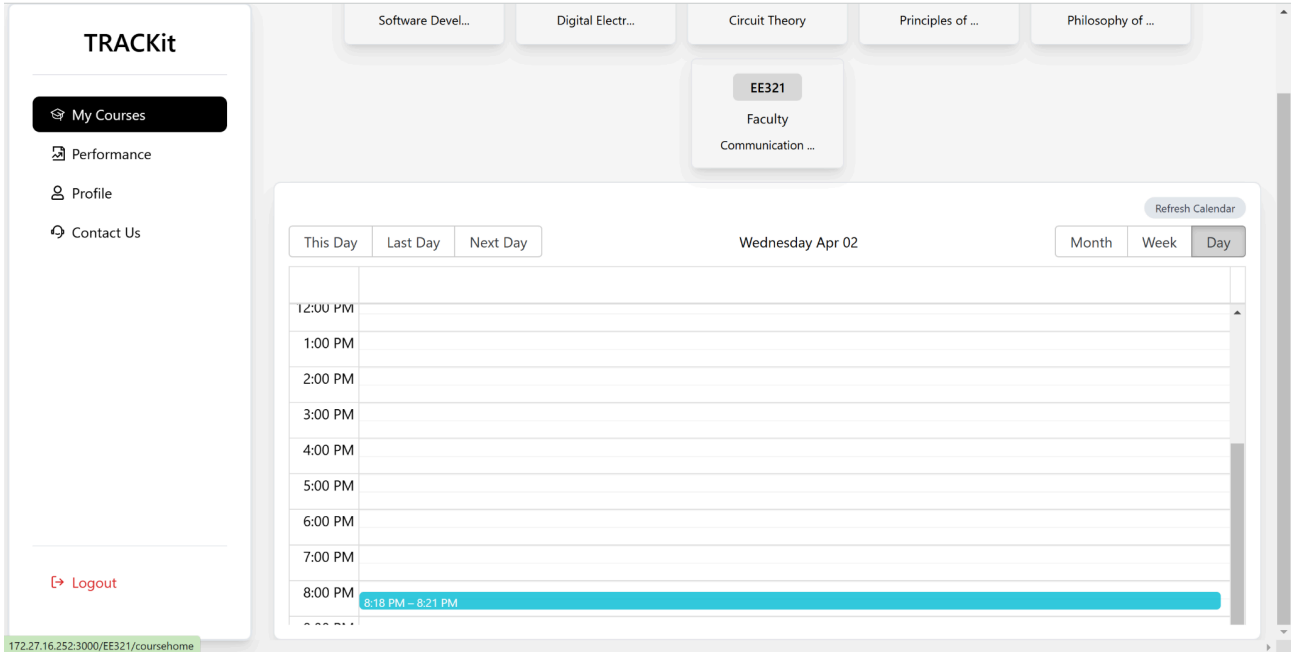
All the APIs are giving desired output. Now checking the frontend UI of events.
Course home:



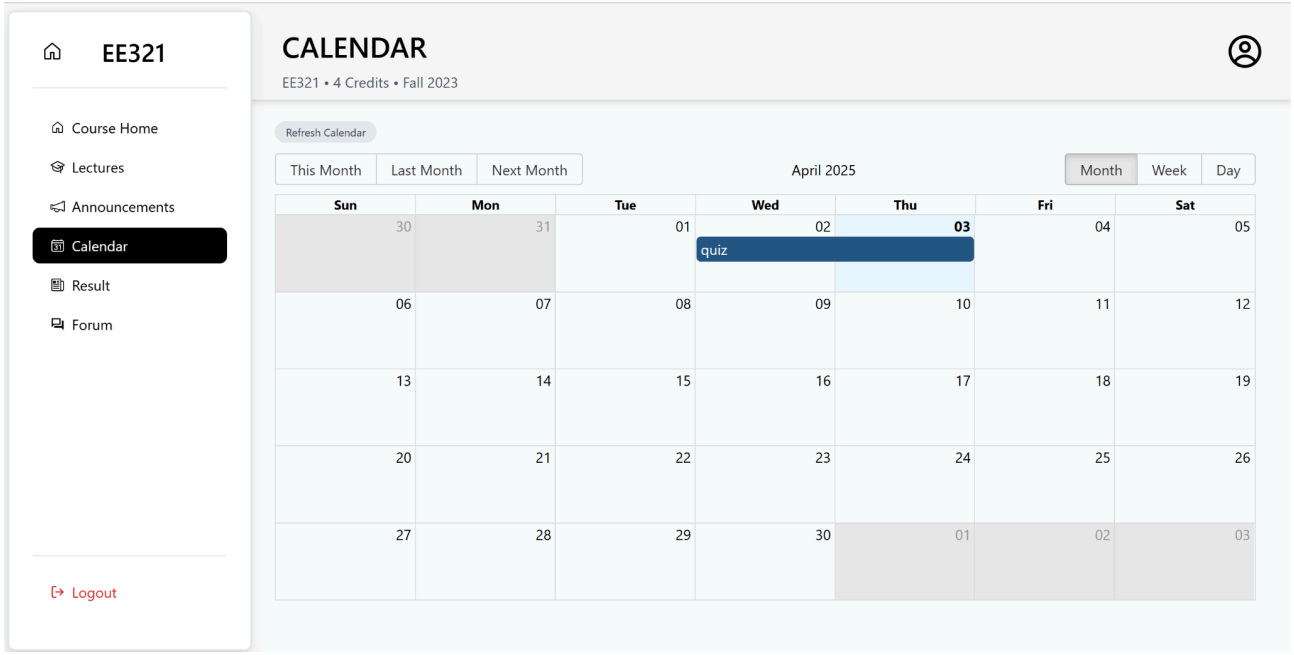
Main DashBoard:



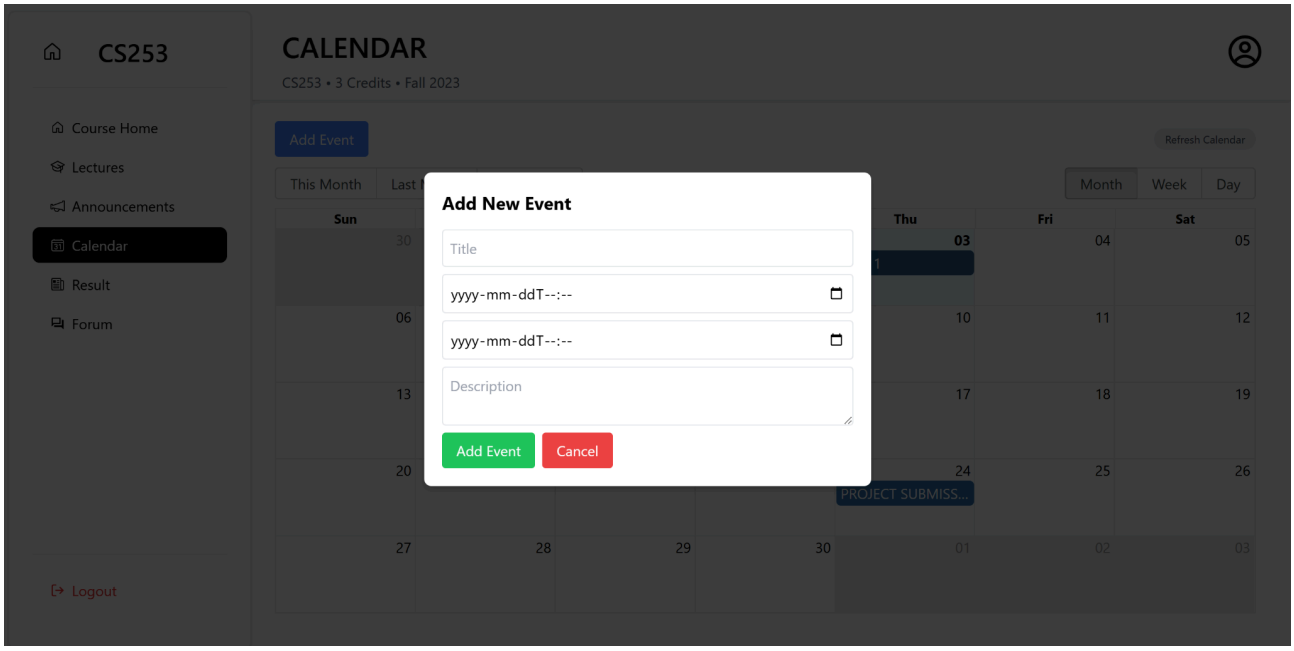


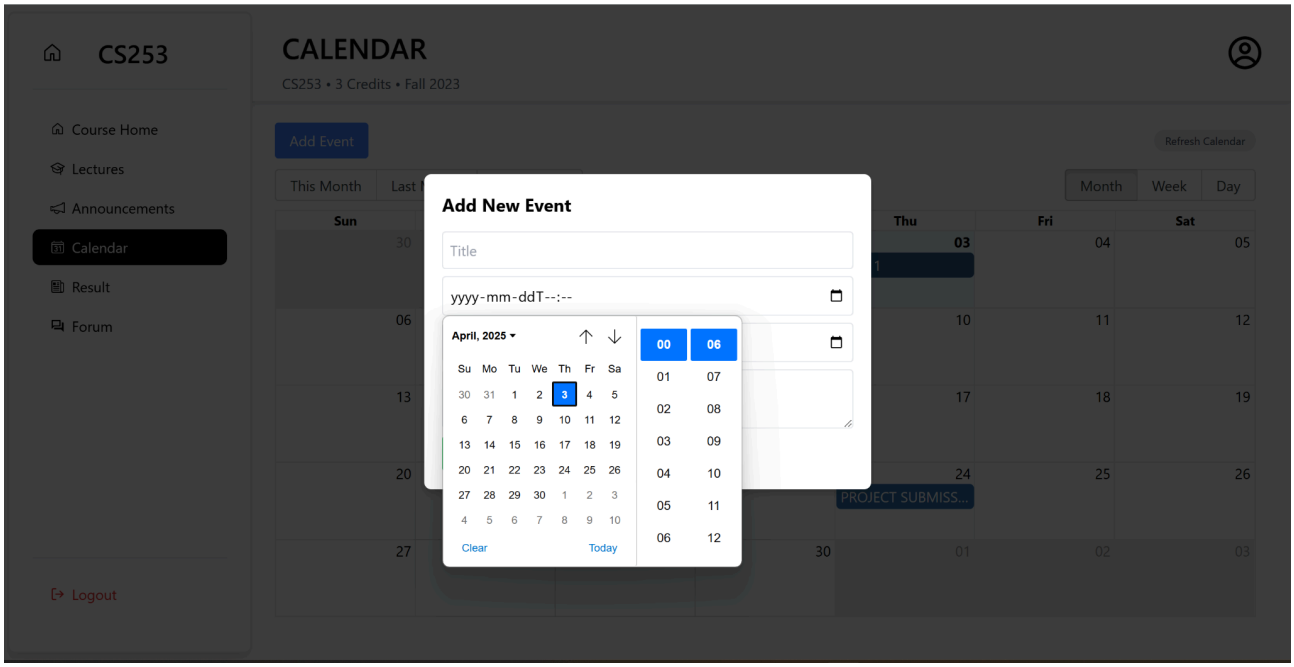


Student Side Calendar Page:



Faculty Side Calendar Page:





Structural Coverage:

The unit is working well and fulfils 100% of the requirements of the SRS. We tested backend and frontend separately in this unit testing, and would cover its integration in integration testing part.

Announcement

Unit Details:

Involves the testing of the Announcements Database and APIs. The announcement database is unique to each course, where an announcement object has a heading along with a body, along with the corresponding frontend. Only Faculties associated with that course can create announcements in that course, but they can be viewed by all students and other Faculty members.

Test Owner:

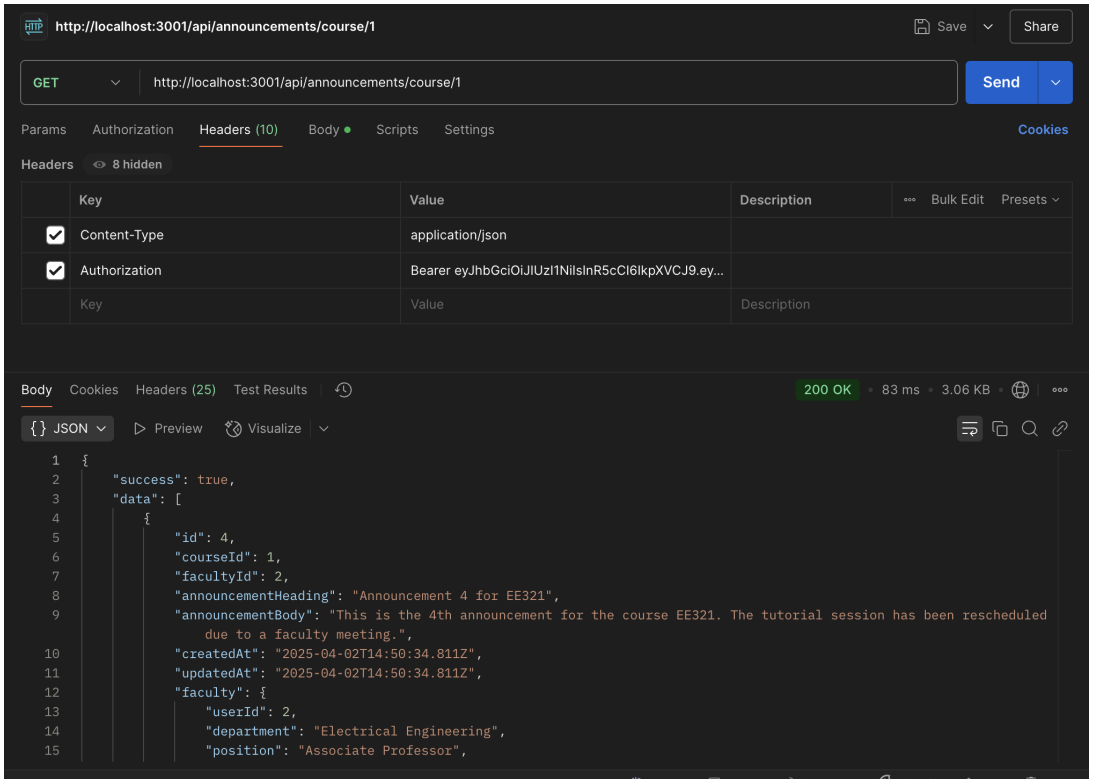
Aditya Gautam, 220064

Test Date:

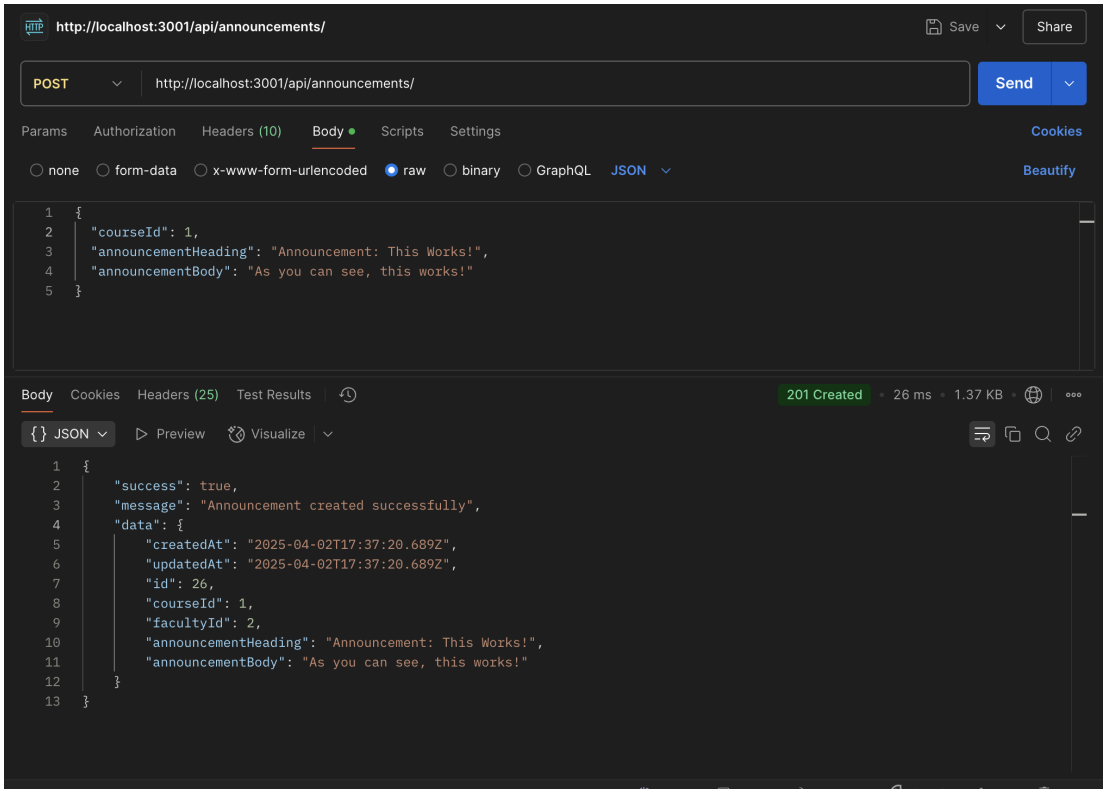
2 April 2025

Test Results:

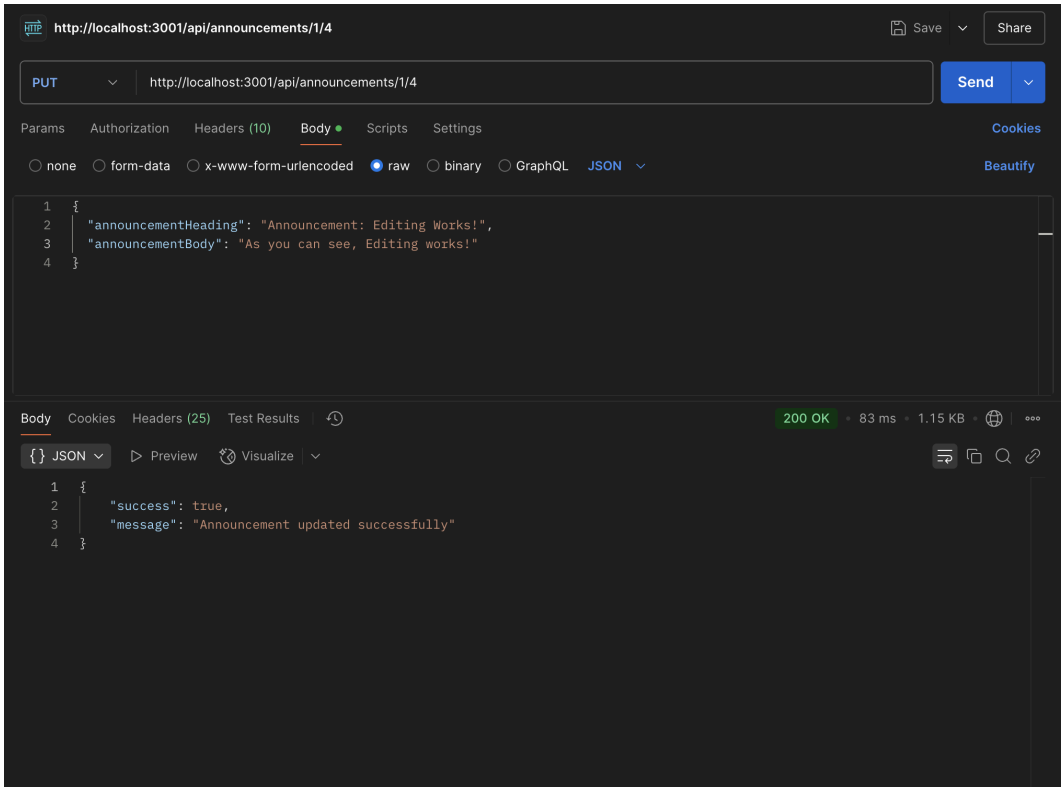
Getting all Announcements: Test Successful



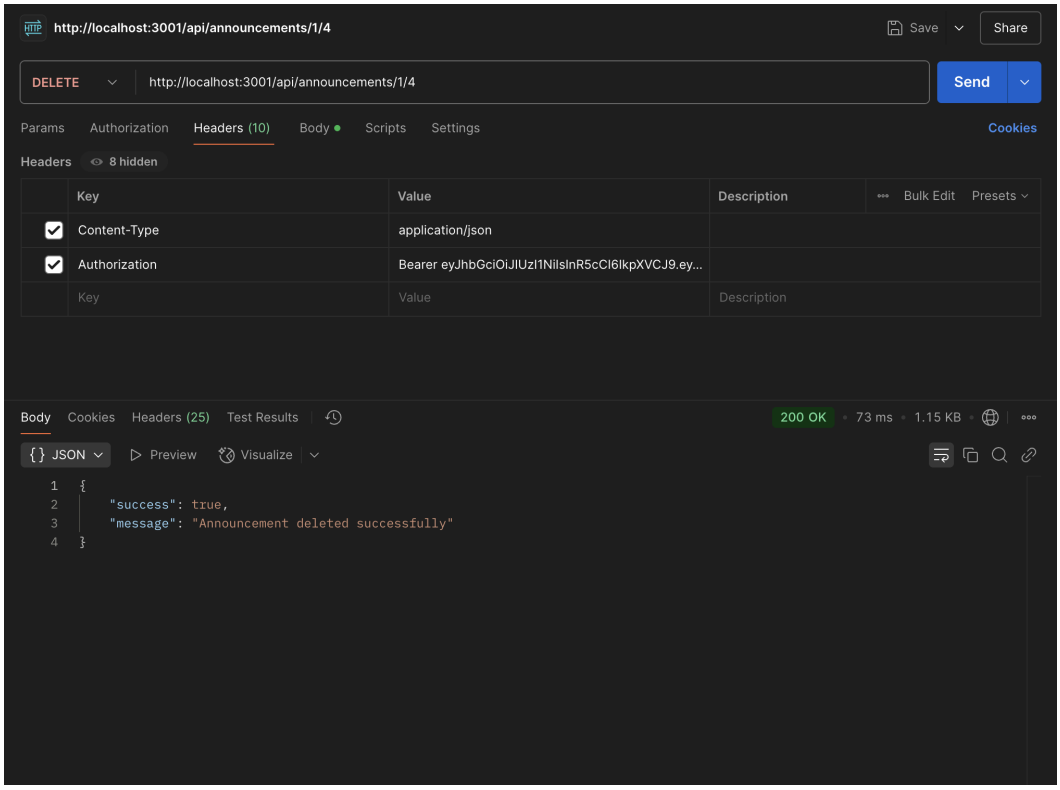
Creating new announcement (Faculty only): Test Successful



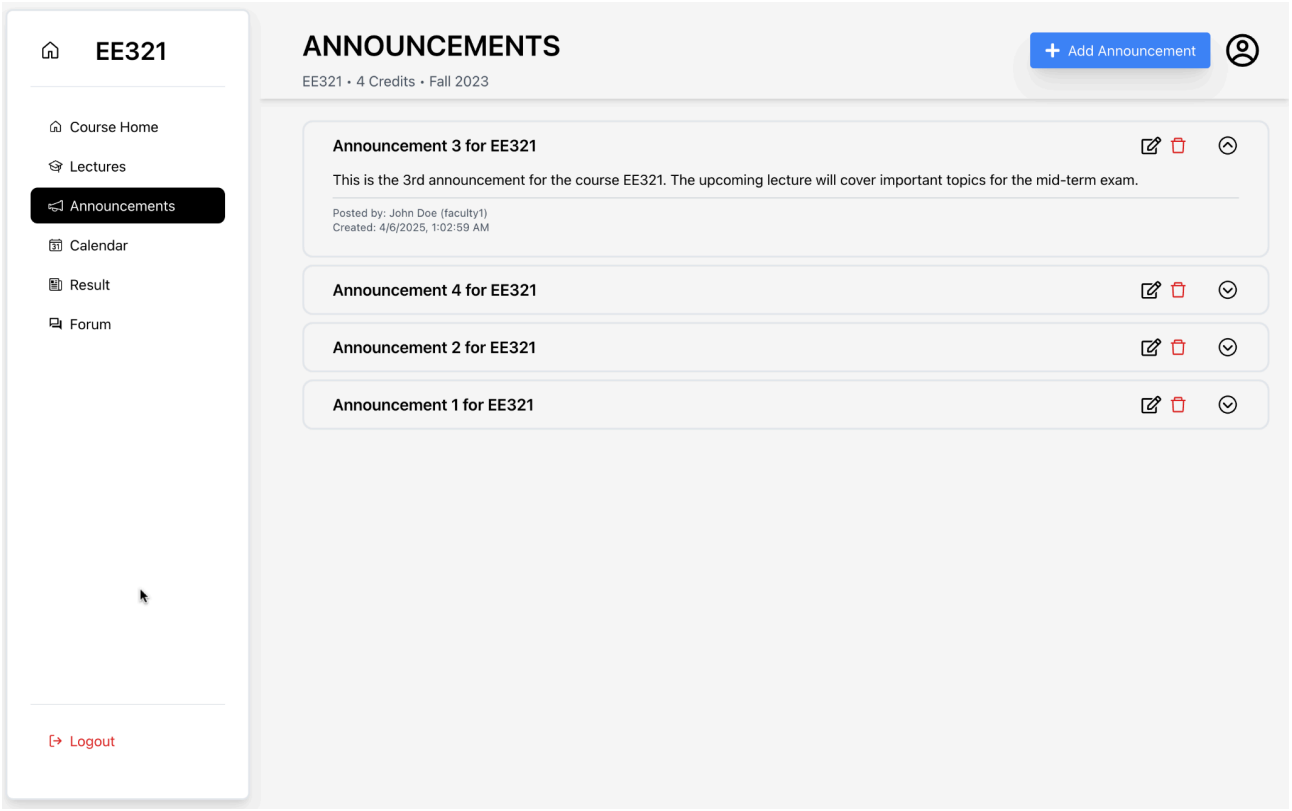
Editing Existing Announcement (Faculty only): Test Successful



Deleting Existing Announcement (Faculty only): Test Successful



Testing the Static Frontend: Test Successful, all components rendering flawlessly



Structural Coverage:

The unit is working well and fulfils 100% of the requirements of the SRS.

Course Description Entries

Unit Details:

Involves the testing of the Course Description Entry Database and APIs. The Course Description Entry database is unique to each course, where an Course Description Entry object has a heading along with a body, along with the corresponding frontend. Only Faculties associated with that course can create Course Description Entries in that course, but they can be viewed by all students and other Faculty members.

Test Owner:

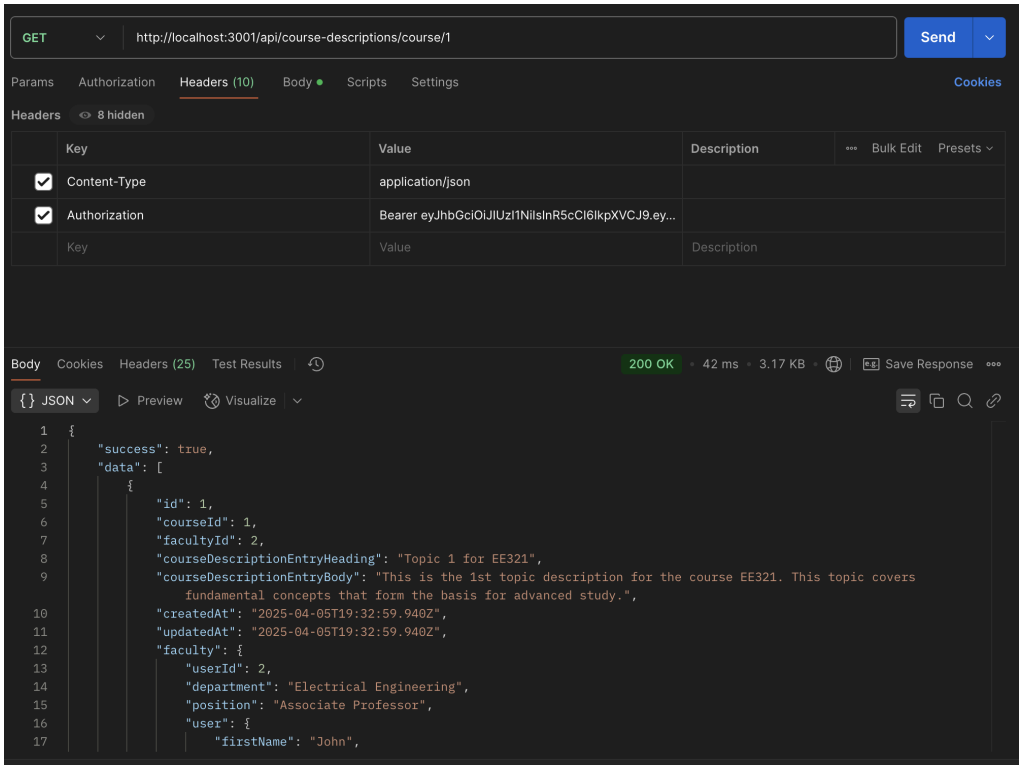
Aditya Gautam, 220064

Test Date:

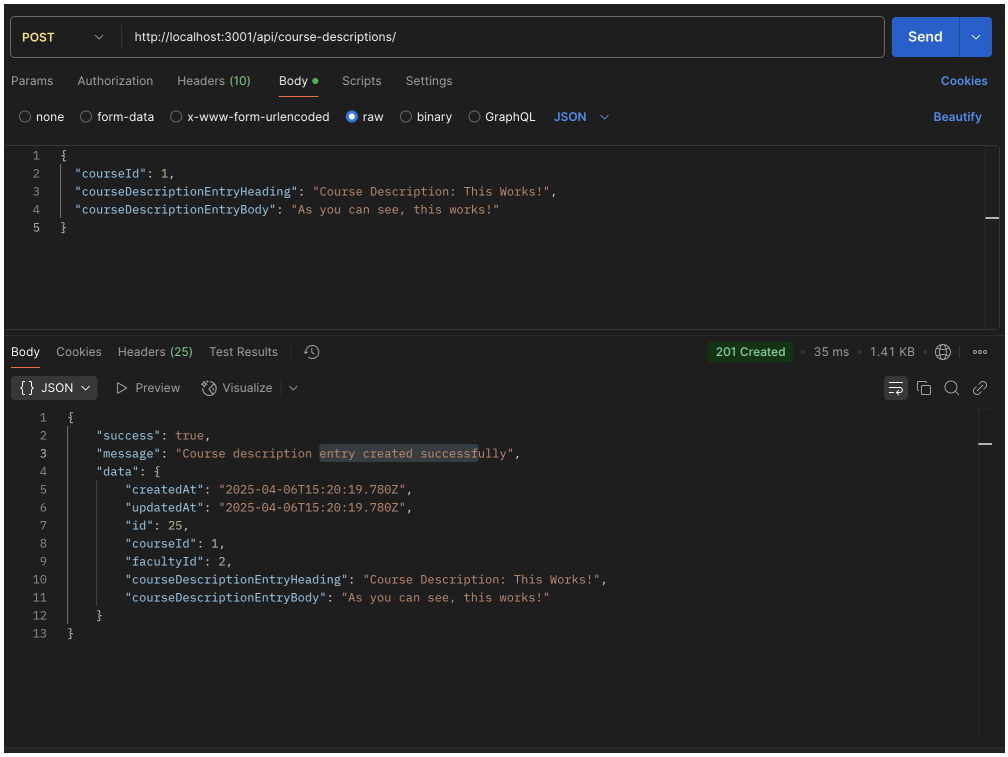
2 April 2025

Test Results:

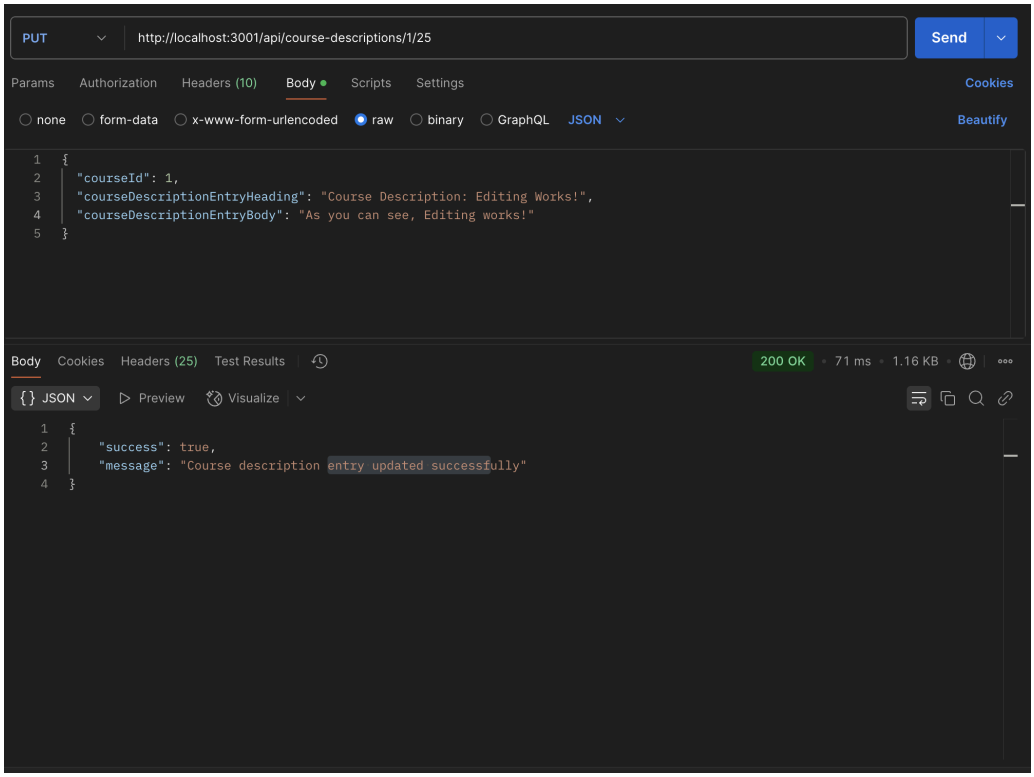
Getting all Course Description Entries: Test Successful



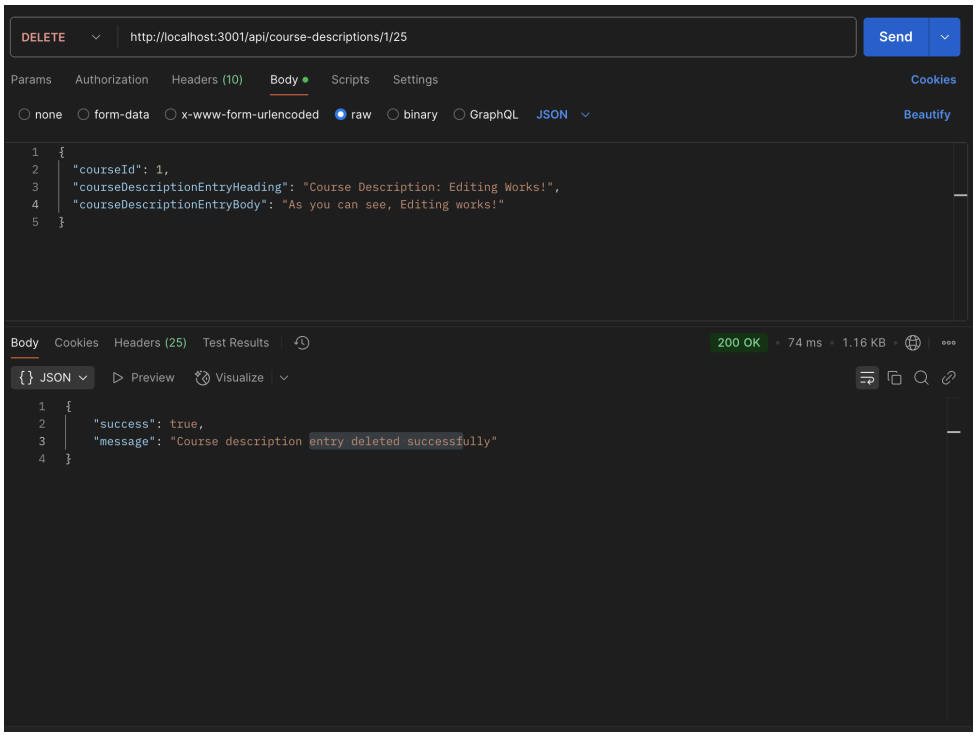
Creating new Course Description Entry (Faculty only): Test Successful



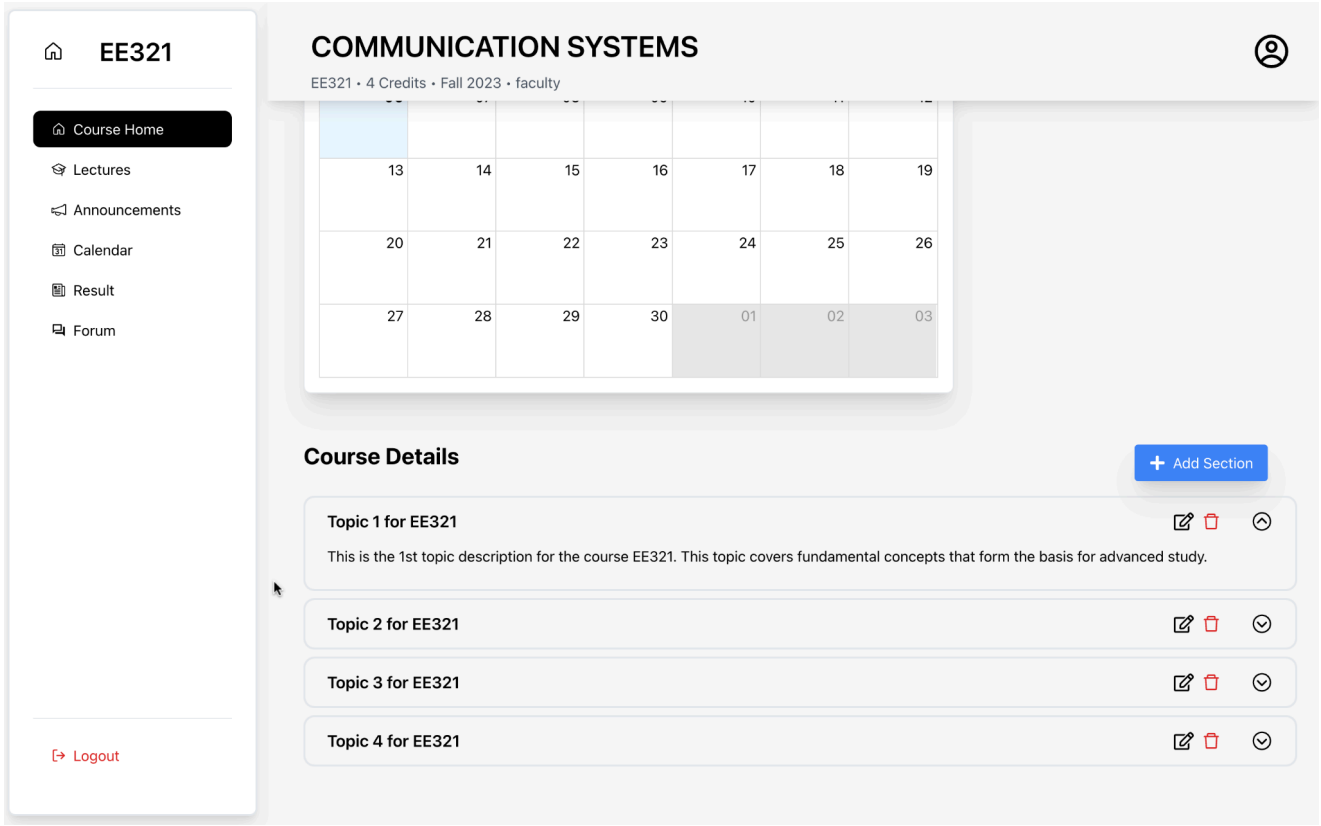
Editing Existing Course Description Entry (Faculty only): Test Successful



Deleting Existing Course Description Entry (Faculty only): Test Successful



Testing the Static Frontend: Test Successful, all components rendering flawlessly



Structural Coverage:

The unit is working well and fulfils 100% of the requirements of the SRS.

Forum

Unit Details:

Involves the testing of the Forum Database and APIs.

Test Owner:

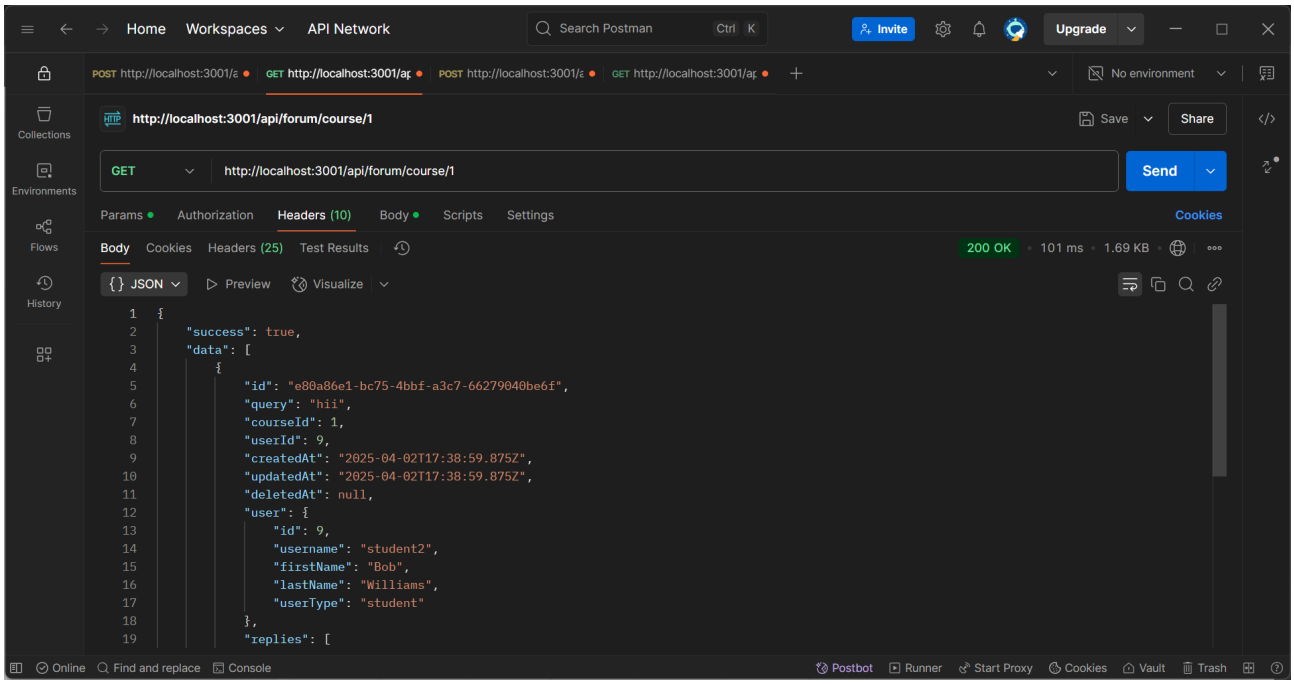
Rahul Ahirwar, 220856

Test Date:

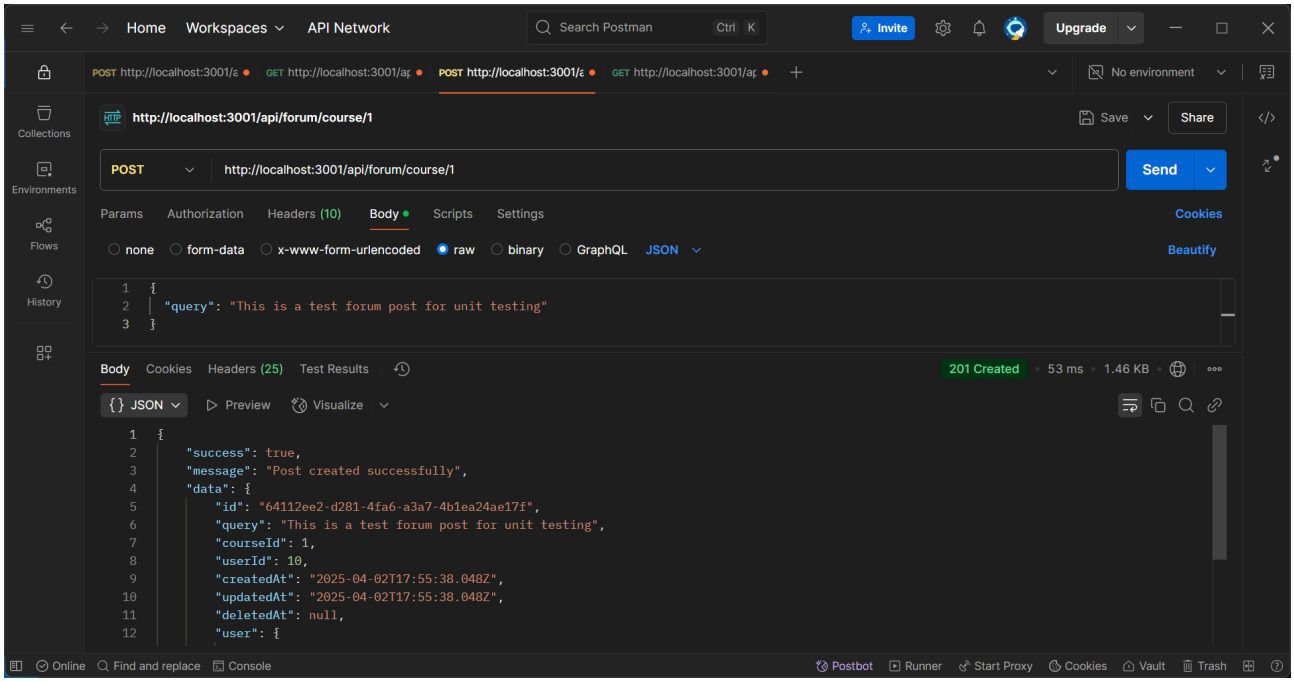
2 April 2025

Test Results: Testing the APIs required in Forum.

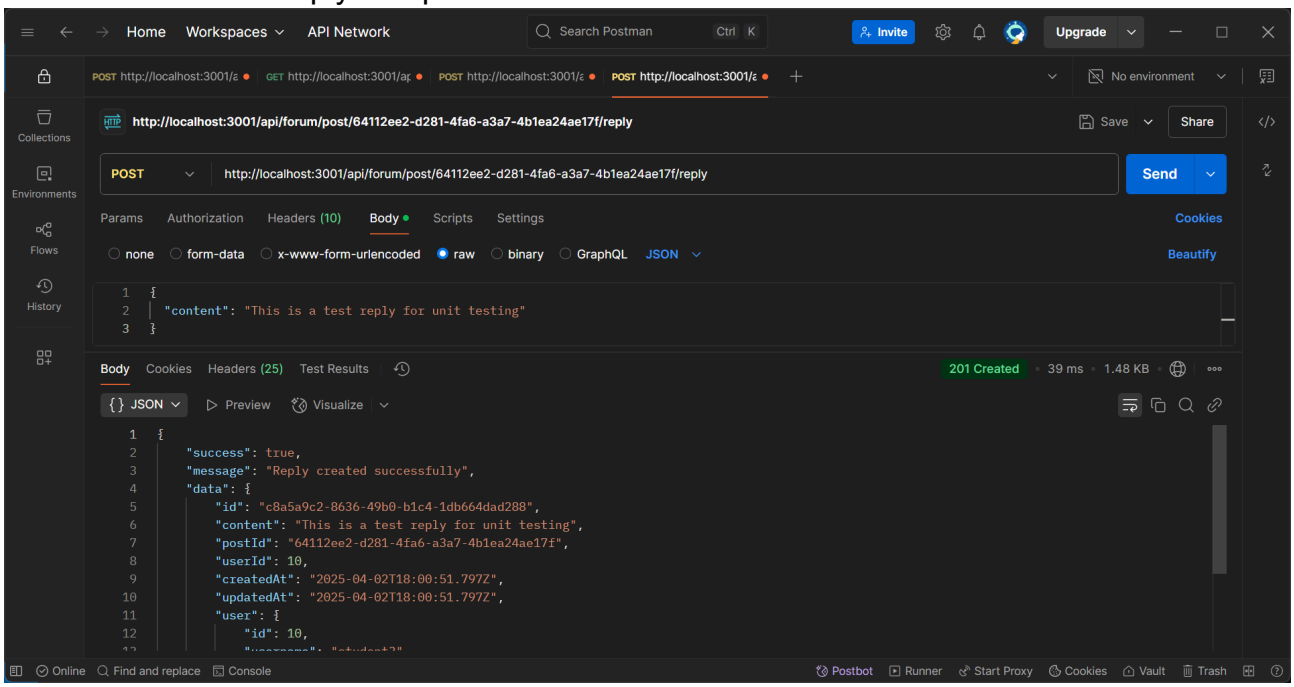
- API to get forum posts for a course.



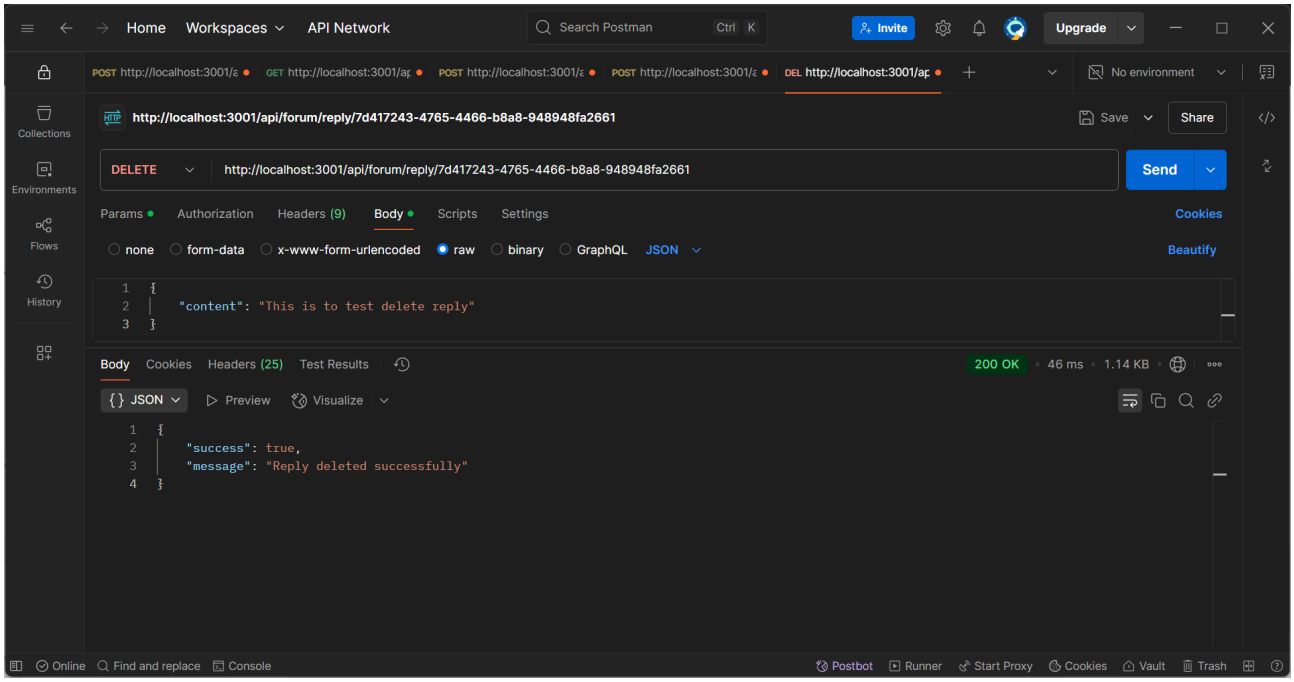
- API to create a new post.



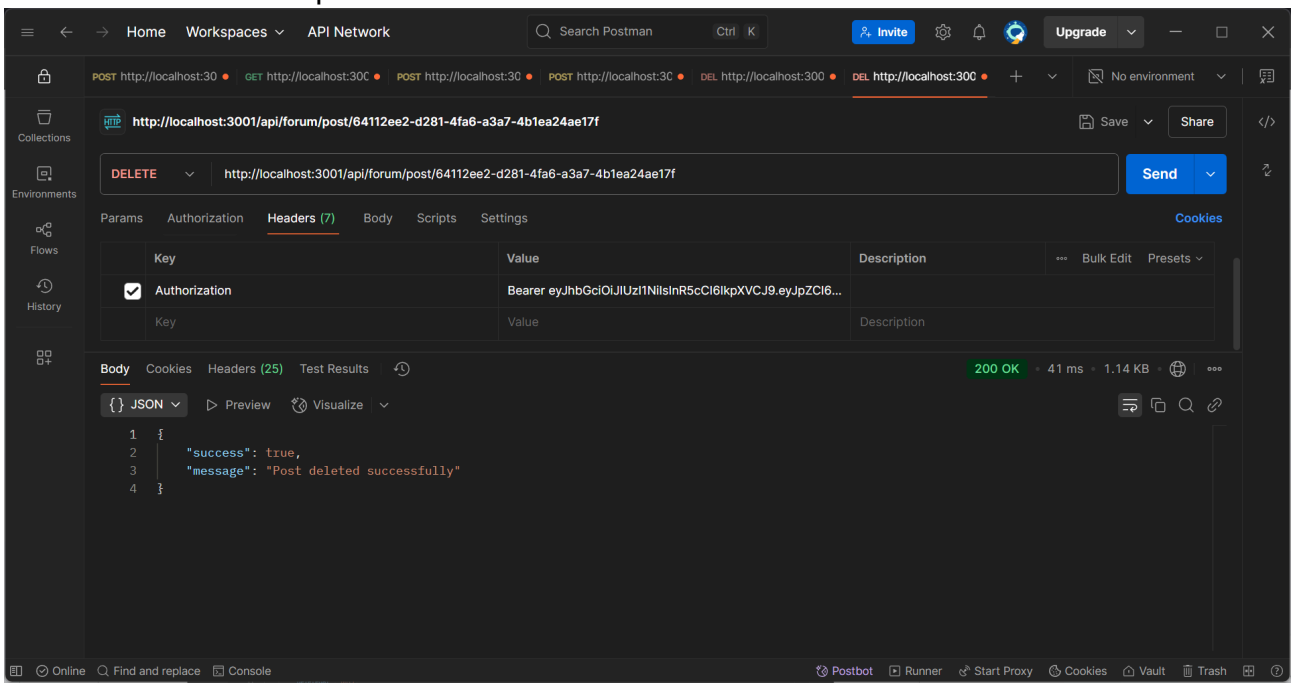
- API to add a reply to a post



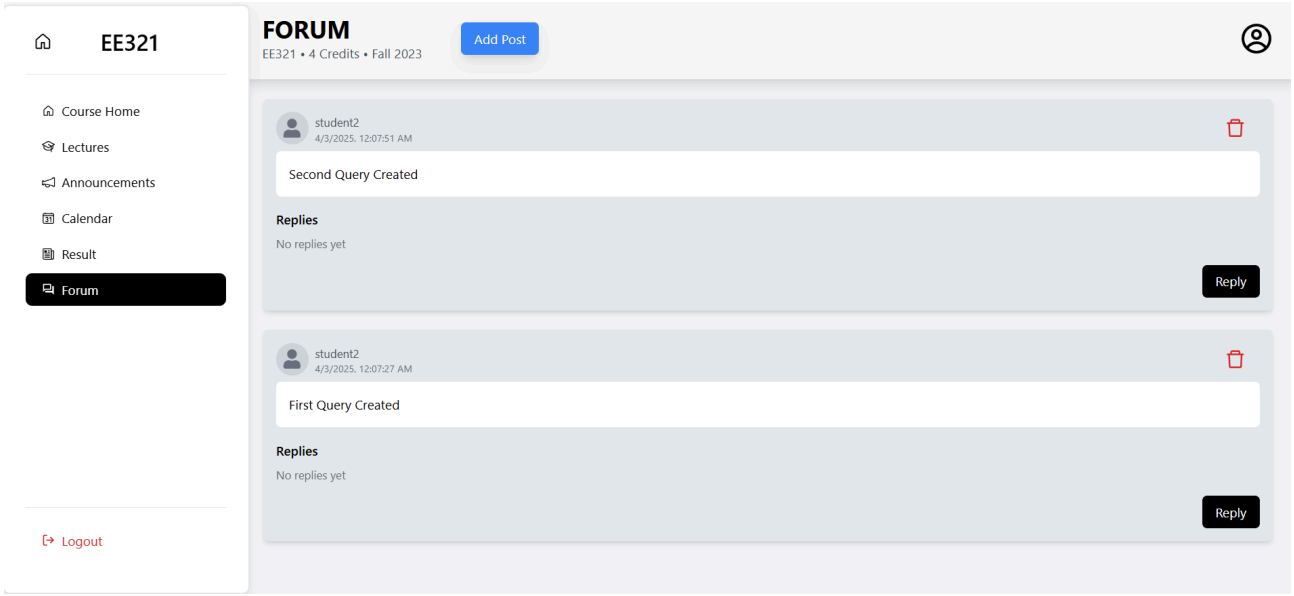
- API to delete a reply.



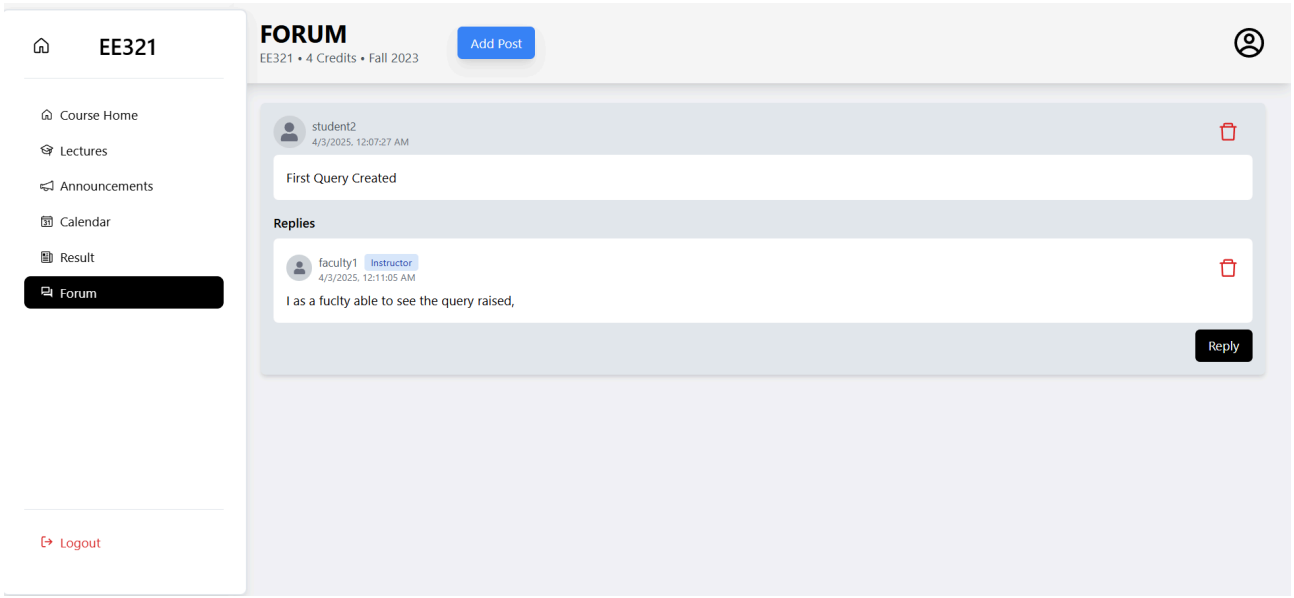
- API to delete a post



Students and faculty are able to create new queries over Forum.



Instructor and other students of the course are able to see and reply to the query raised.



Structural Coverage:

The unit is working well and fulfils 100% of the requirements of the SRS.

Results

Unit Details:

Classes: Result class, Exam class and its interaction with User class and Course class.

Functions: There are various functions for performing read, write, delete and modifying tasks on the result and exam data, description of each of them is given within the test results section.

Test Owner:

Dhruv Rai, 220365

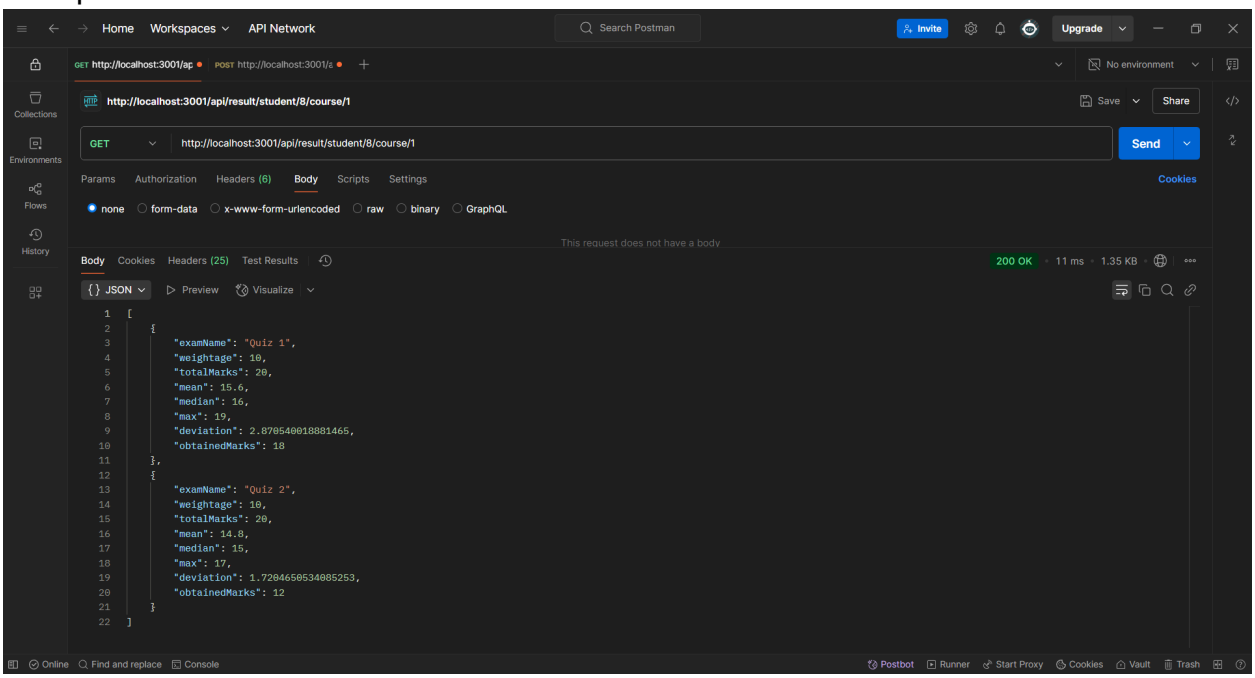
Test Date:

26/03/25-02/04/25

Test Results:

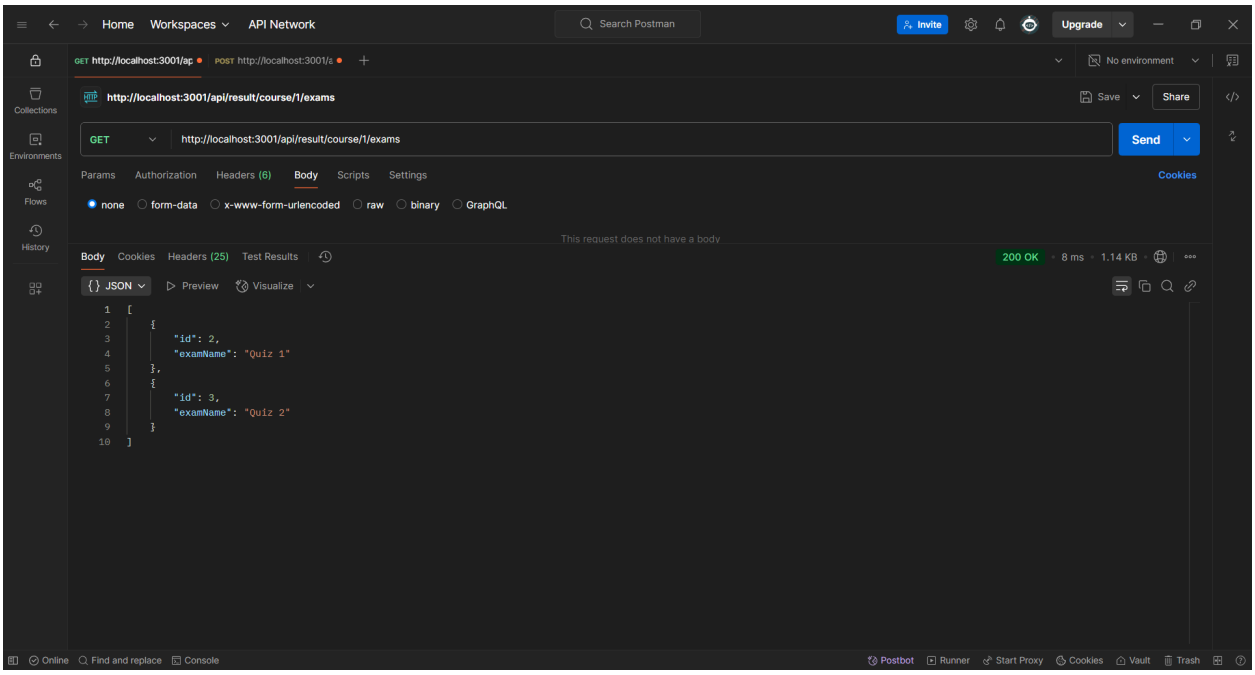
First, I presented the tests for the backend and then for the frontend part. The details of the tests are as follows-

Api to fetch result for a student-
I have tested this api and it is able to get data every time when it is present in the database. I have also tested it for the edge cases when the course id is wrong or the user id is wrong or the database is not present and in every scenario it worked as expected.



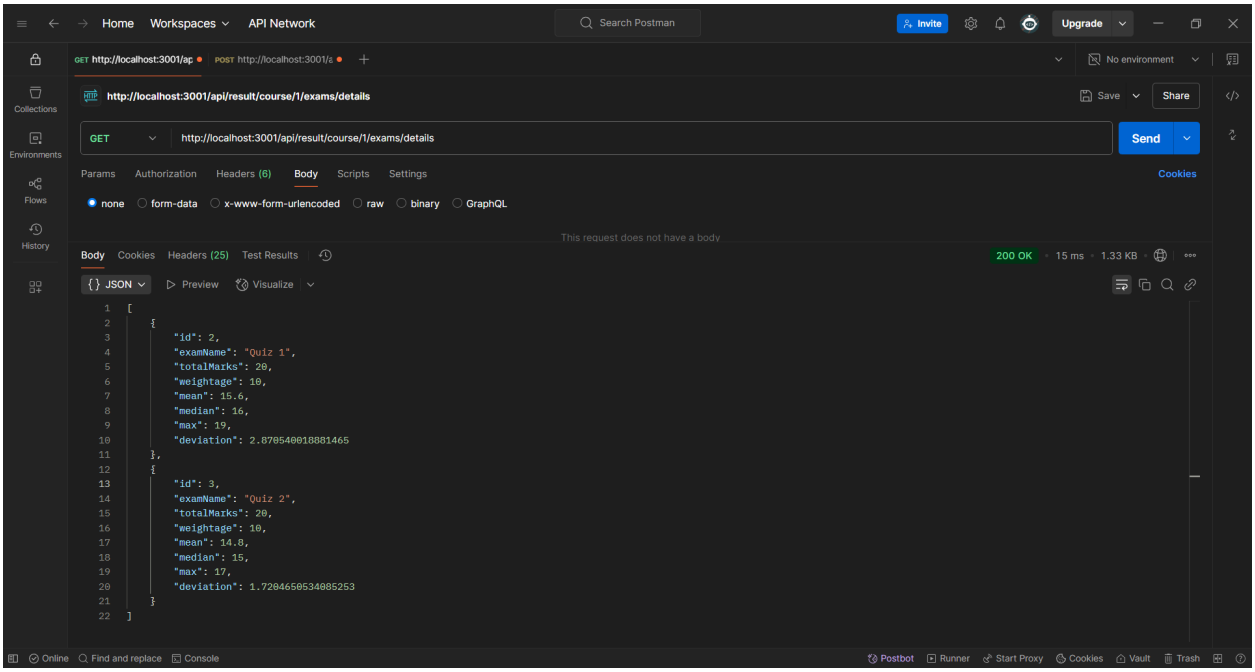
Api to fetch exam names and id for a course-

It takes course id as input and gives the name and id of all the exams present in the database for that particular course. It gives a major bug in initial testing in which this api is giving all the exams rather than the exams related to a particular course. I have successfully resolved this bug and now this is working fine in most of the scenarios.



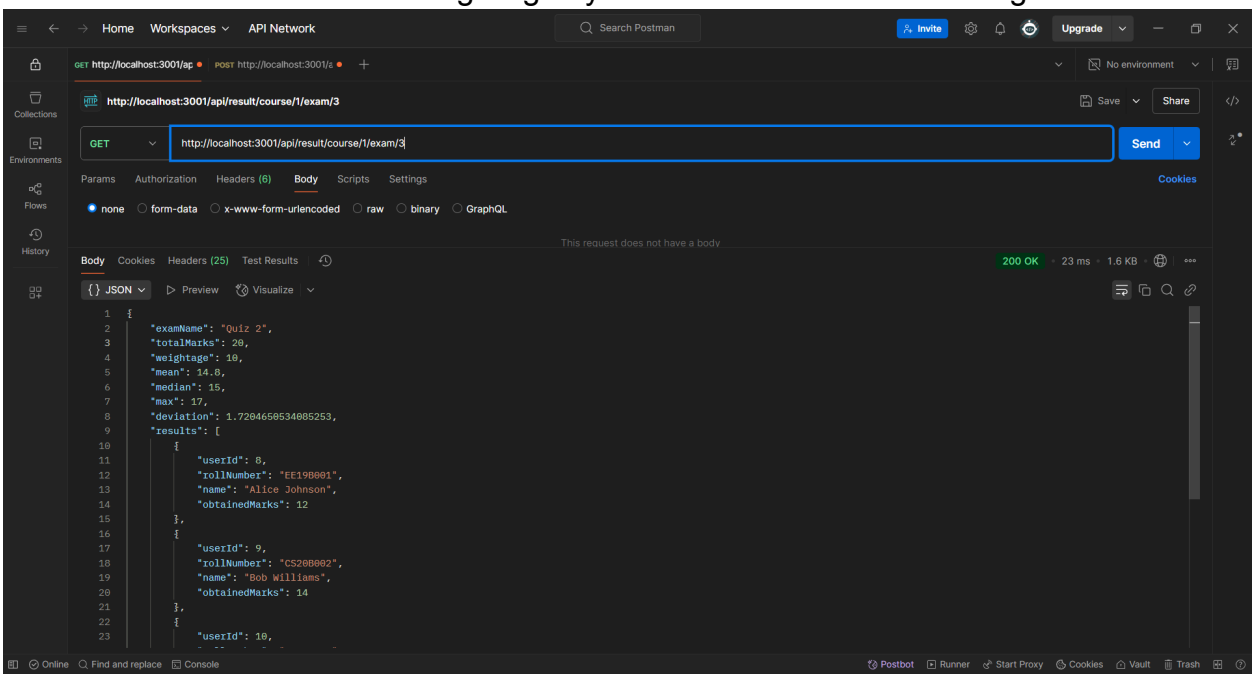
Api to fetch exams and details of them for a course-

It takes course id as input and gives all the details for all the exams present in the database for that particular course. This api worked fine in normal cases and also for the edge cases.



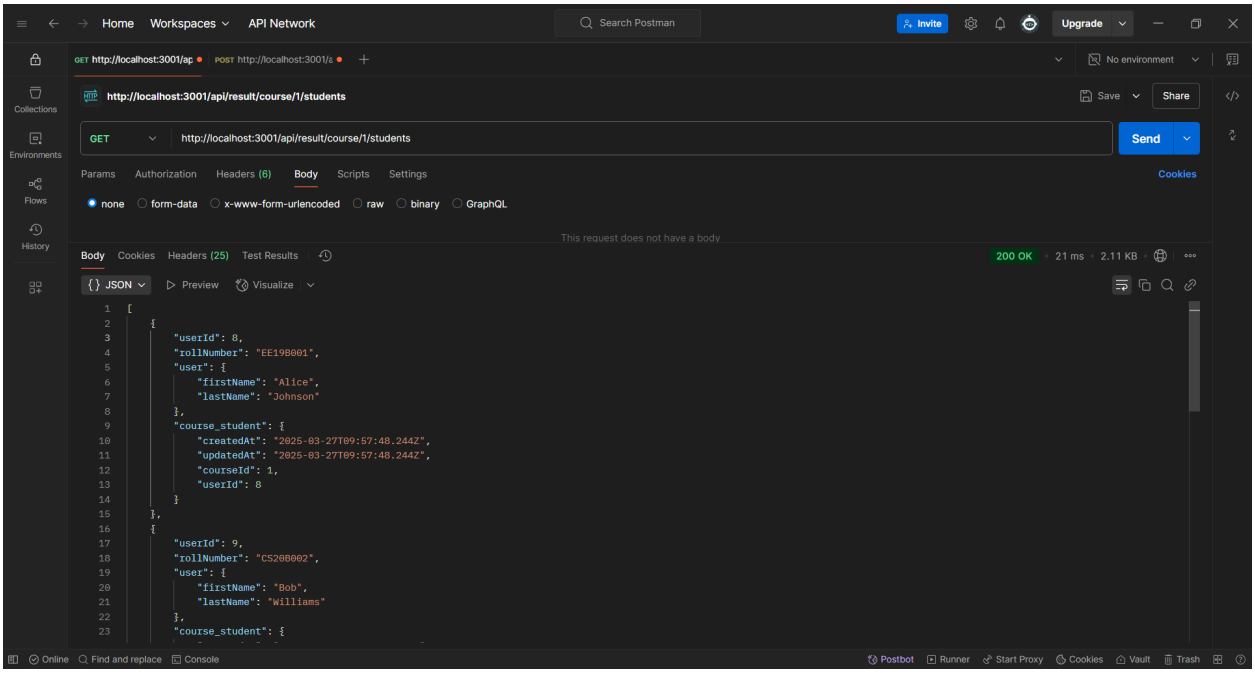
Api to fetch results for all the students for an exam-

It takes course id and exam id as input and gives results of all the students for that exam. It will be used by faculty to view results for a particular exam. This api worked fine in normal cases but has minor bug when obtained marks are not present for a student. I have successfully fixed this bug and now in absence of obtained marks it returns null for that user without giving any error. Now this unit is working fine.



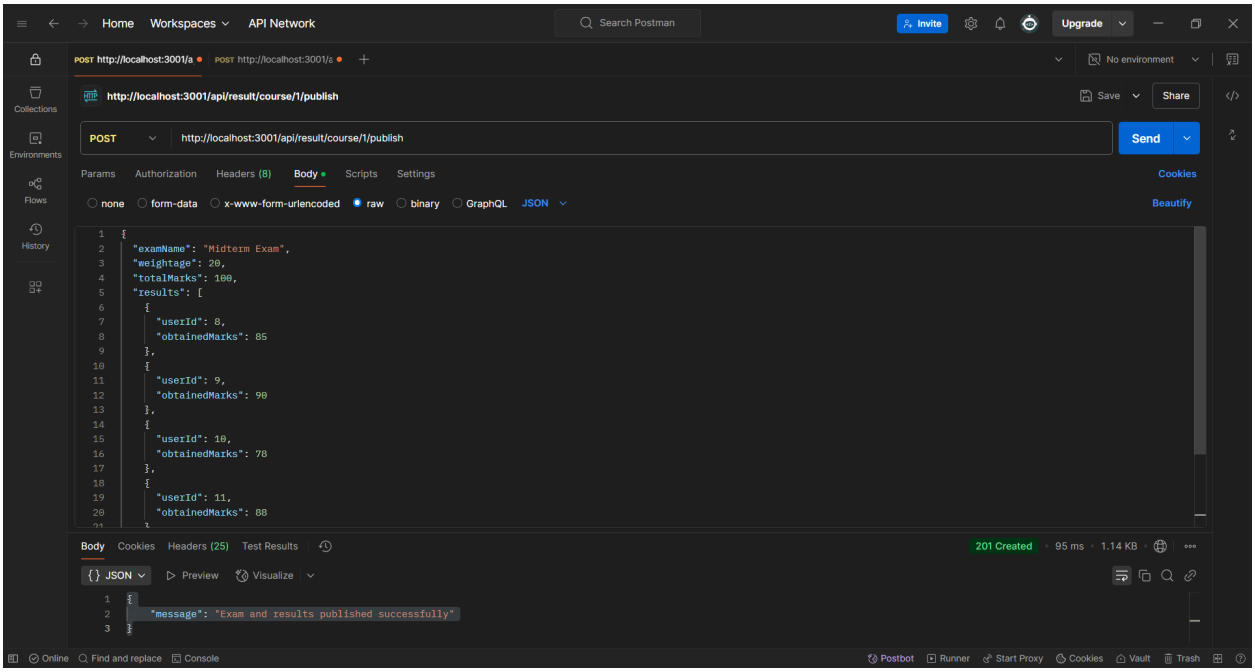
Api to fetch all the enrolled students in a course-

This unit will be used to get details of all the students enrolled in a course. This will be called by frontend when the faculty wants to add results for an exam. This unit passed all the tests successfully for most of the scenarios.

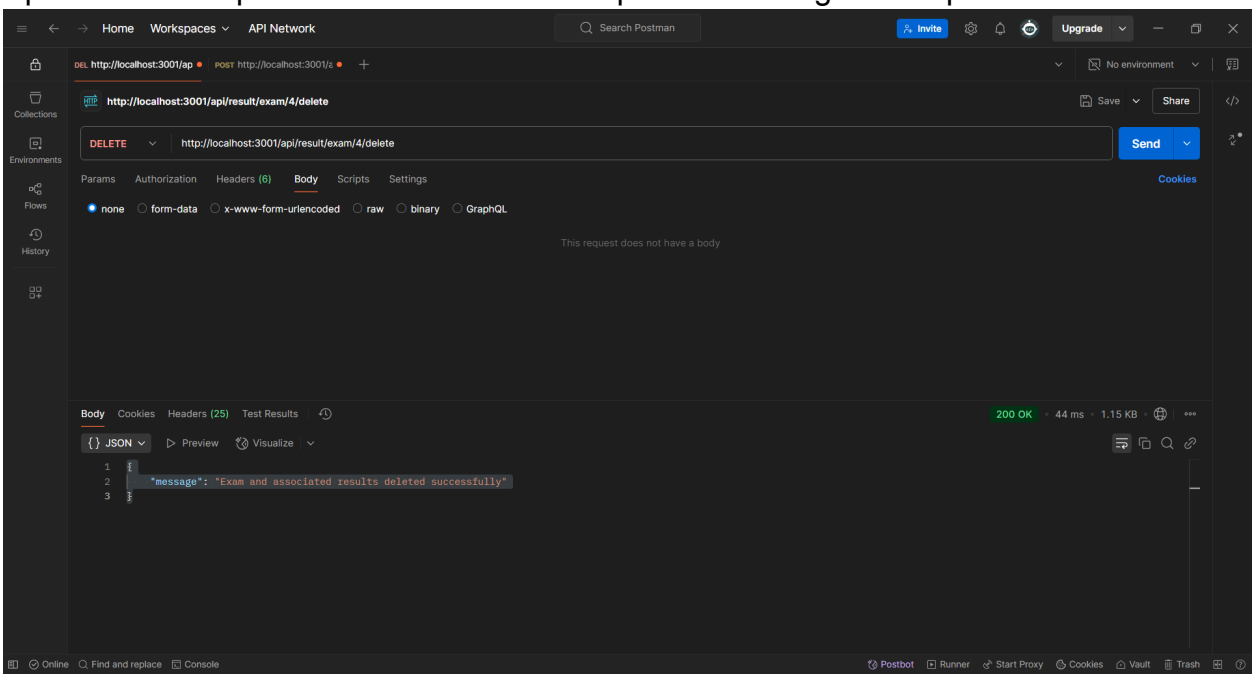


Api to publish result for a new exam-

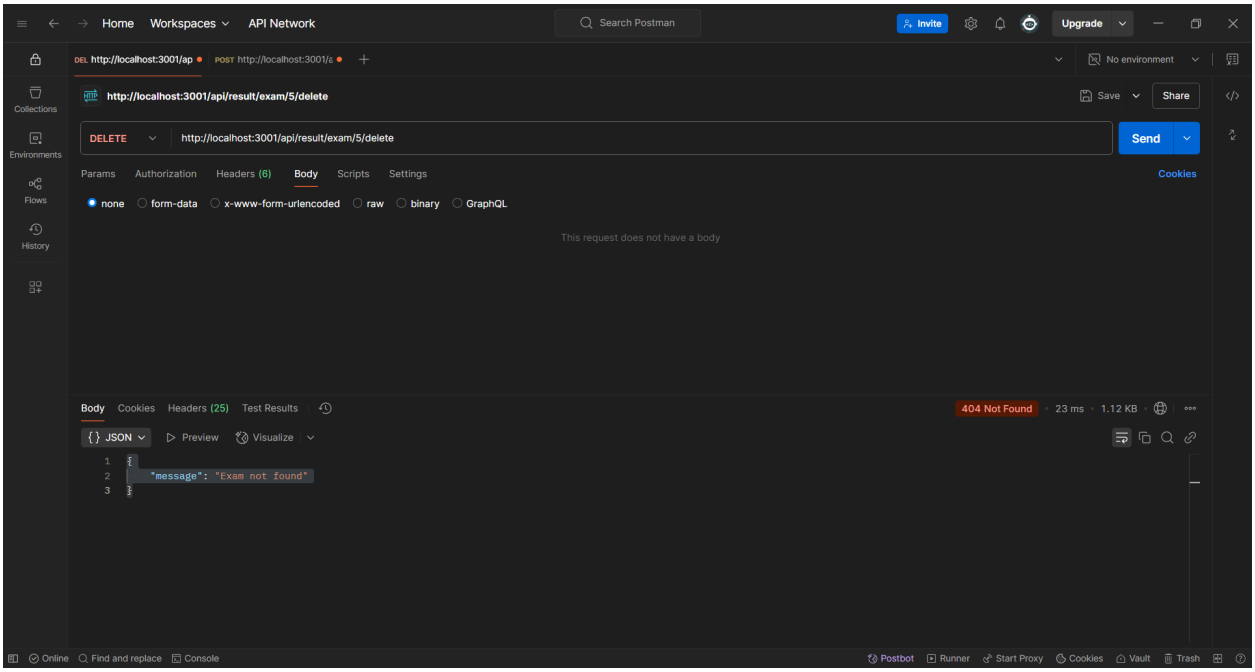
This unit will be used to publish results for a particular exam, this will be called by the faculty side frontend. One requirement for this api is that the data should only contain marks for the students enrolled in a course. If the data contain more or less students it can cause errors. Although it affects robustness but with a smart frontend this can be ensured that the data has proper format. Once data is in the correct format it works fine in all the scenarios.



Api to delete an exam and all its related results from the database-
This unit works fine for all the cases. When there is an exam with the given exam id it performs as expected and shows the completion message as output.

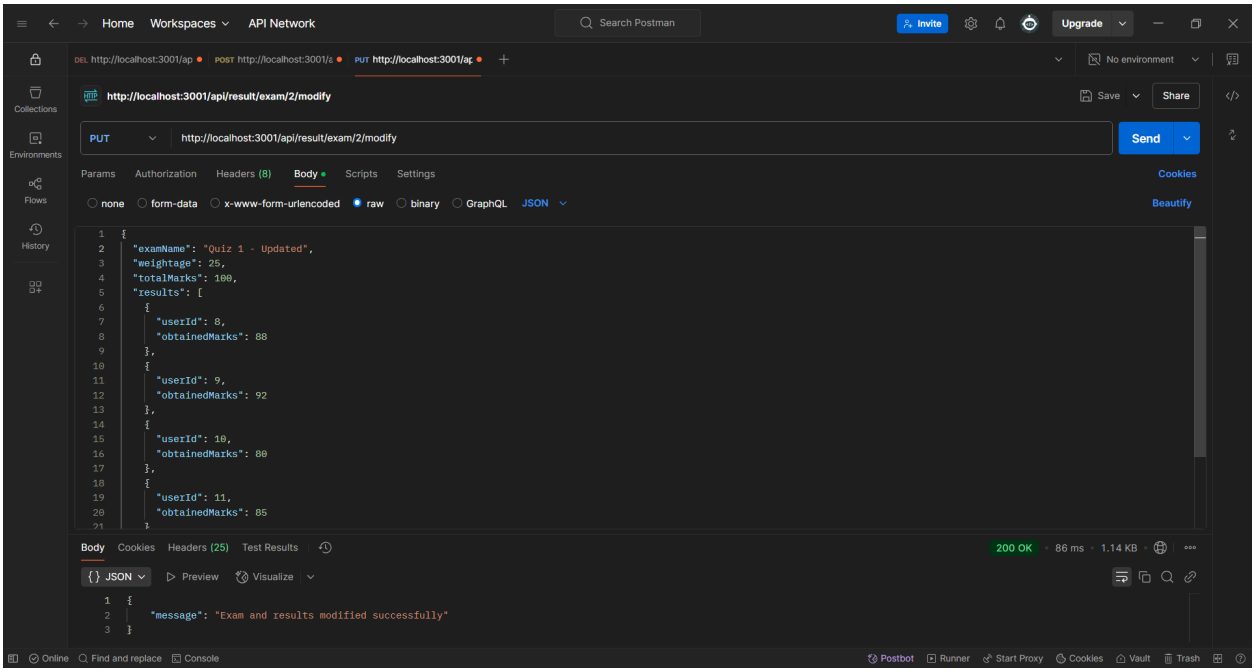


But when no exam with a given exam id is present in the database. It correctly presents an error message which says “Exam not found”. In all other cases it also performs well.



Api to modify exam result-

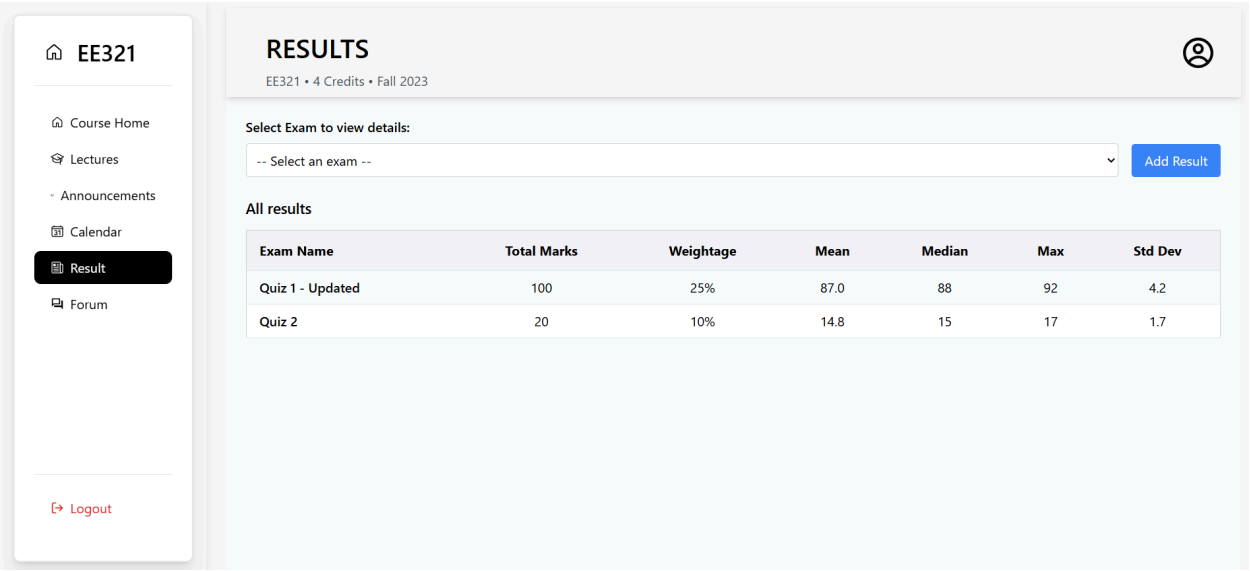
This unit performs as expected for all the normal cases as well as for the edge cases like cases when wrong exam id is given or data is not present in the database.



Now I will present the test results for the frontend units which will serve as the user interface for the result page in the system. For now all data is static as the backend is not connected it is tested only to ensure that frontend units are working fine.

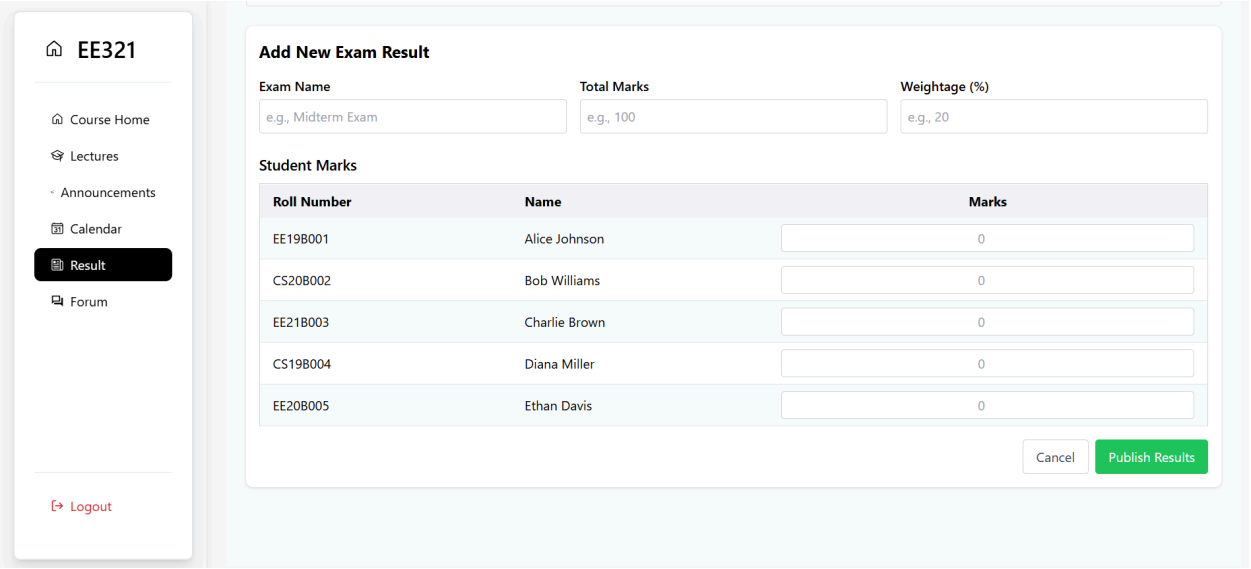
The faculty result page-

This page renders successfully in all the cases.

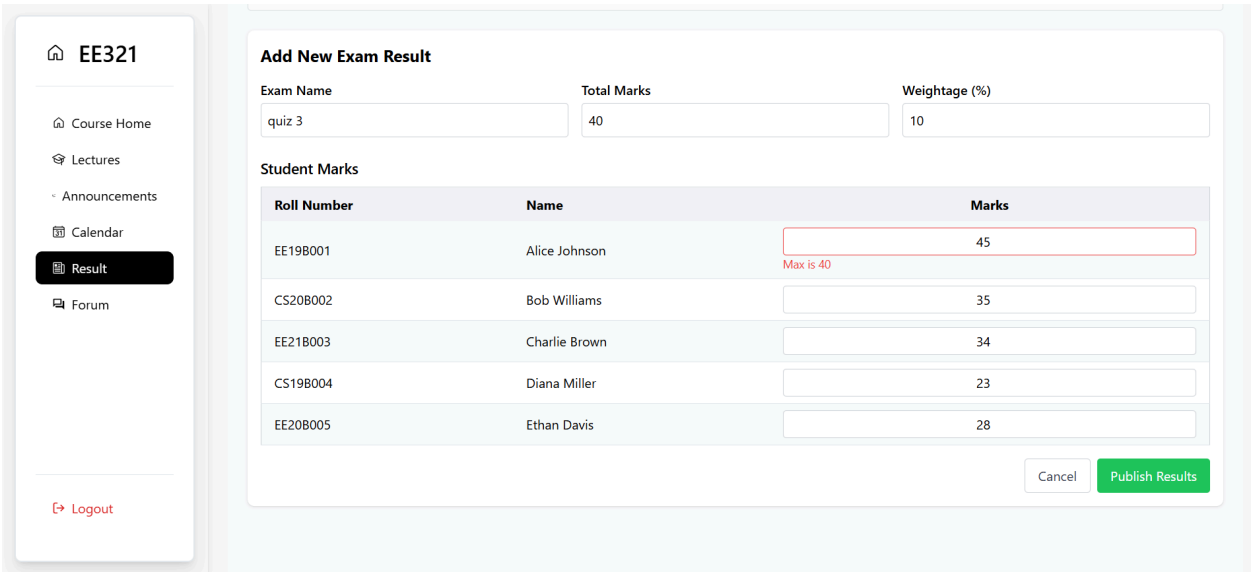


Add result form-

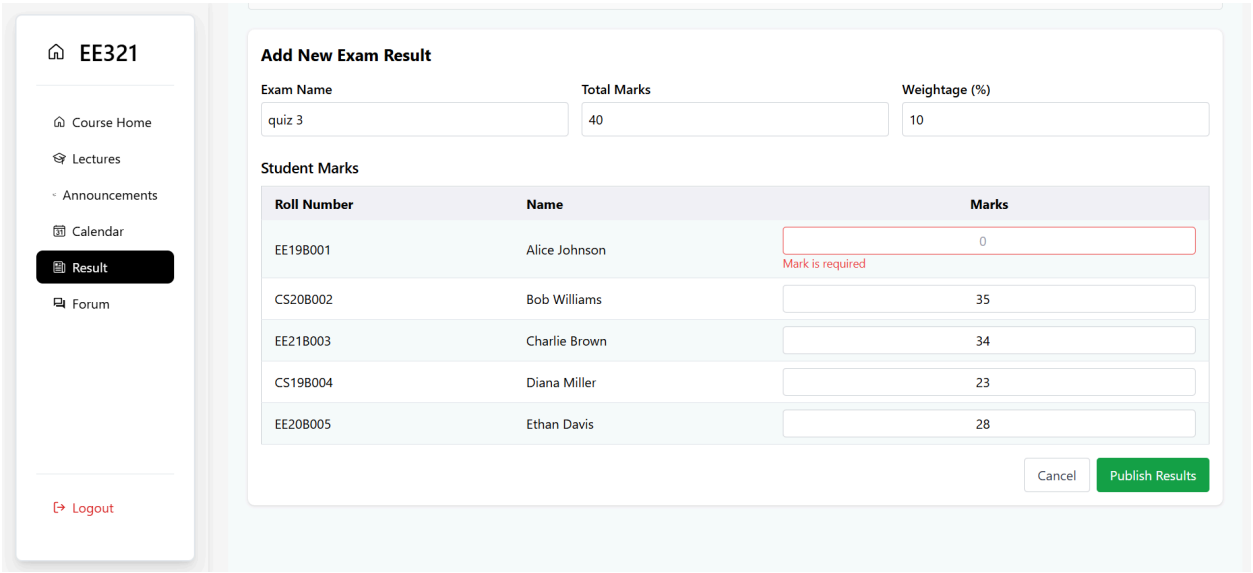
This page works successfully and shows the list of students enrolled in the course.



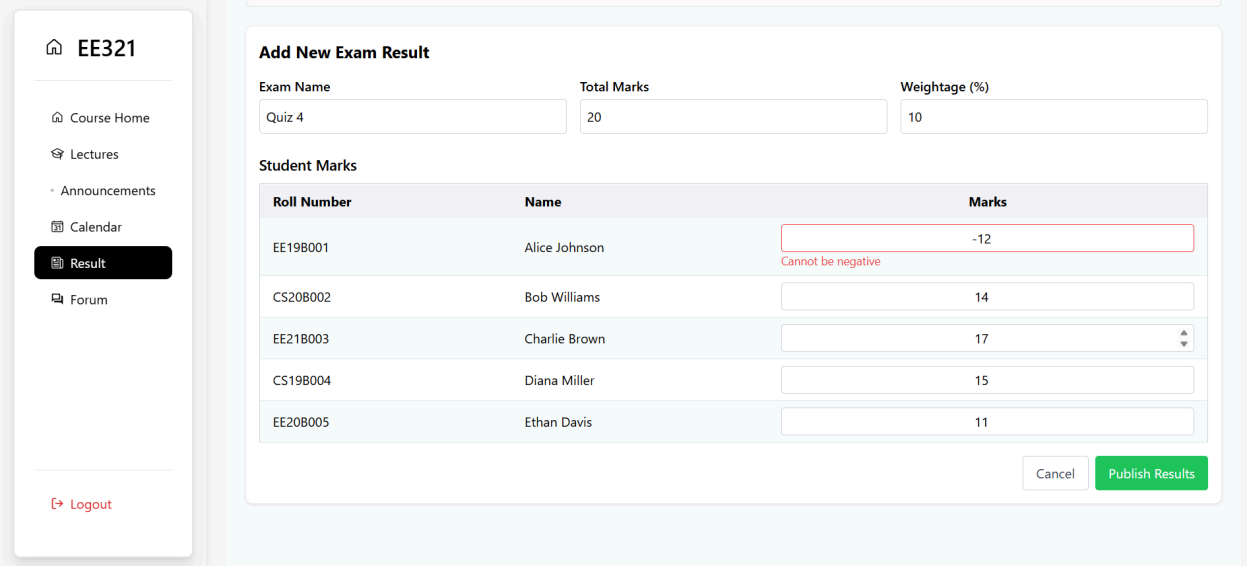
It also ensures that the obtained marks for any student does not exceed above the total marks available for the exam.



It also makes sure that faculty enter marks for every student enrolled in the exam which is consistent with our backend units used for publishing the exam result.



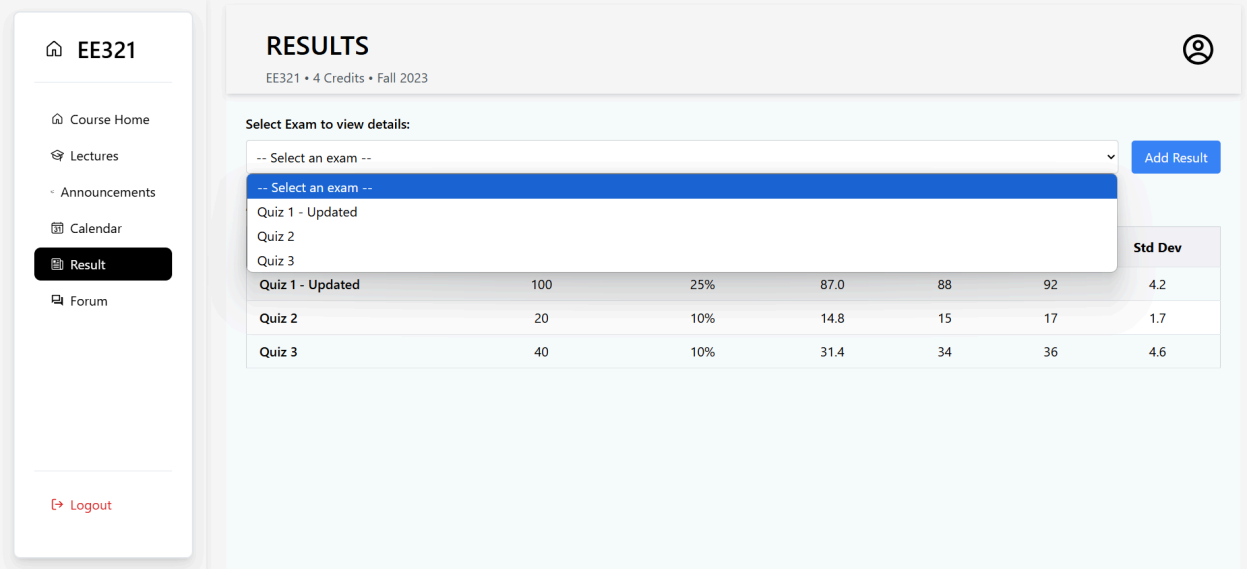
It also checks that marks obtained by students are not negative.



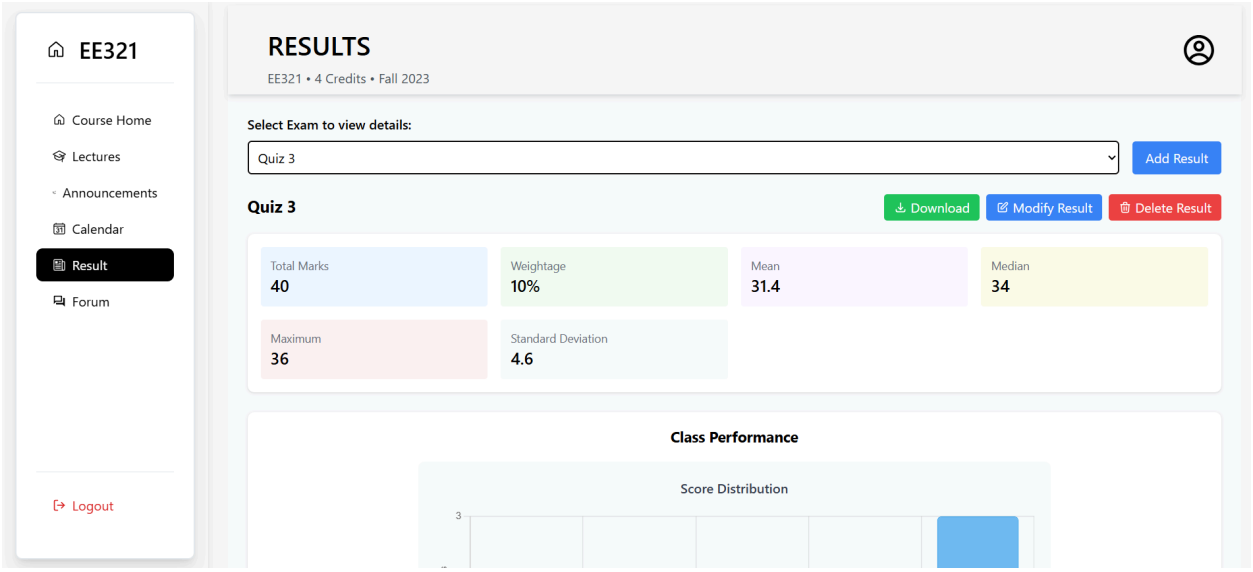
If faculty enter correct marks for every student the ‘publish result’ button works fine.

Detailed result view for an exam-

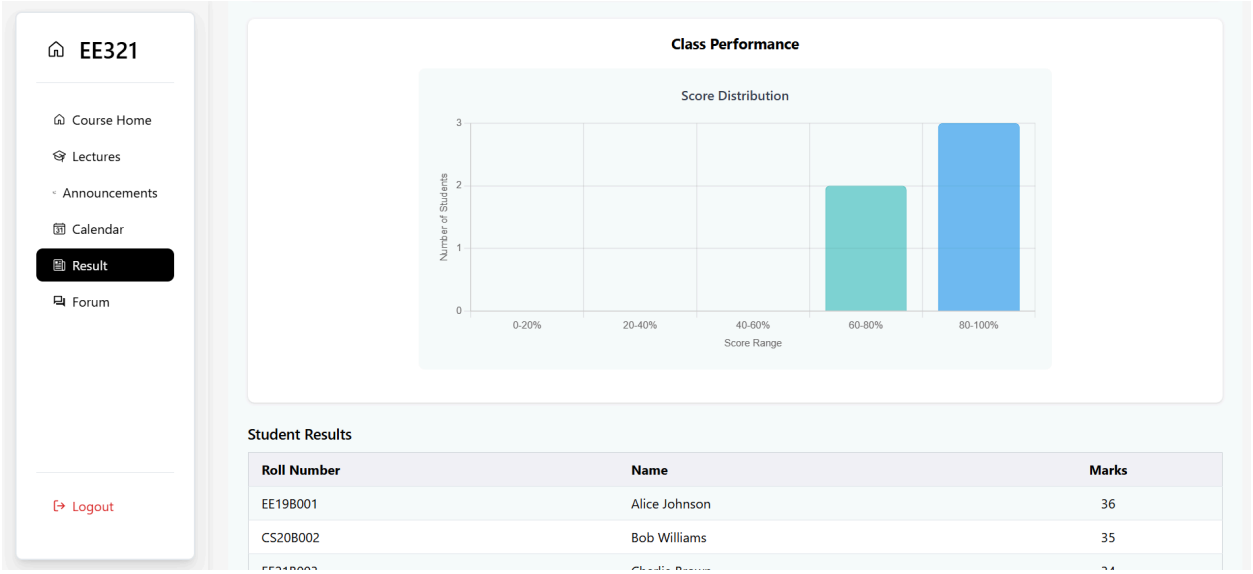
The faculty can view the details of an exam for which the results has been published by selecting the exam from the dropdown menu or from the table.



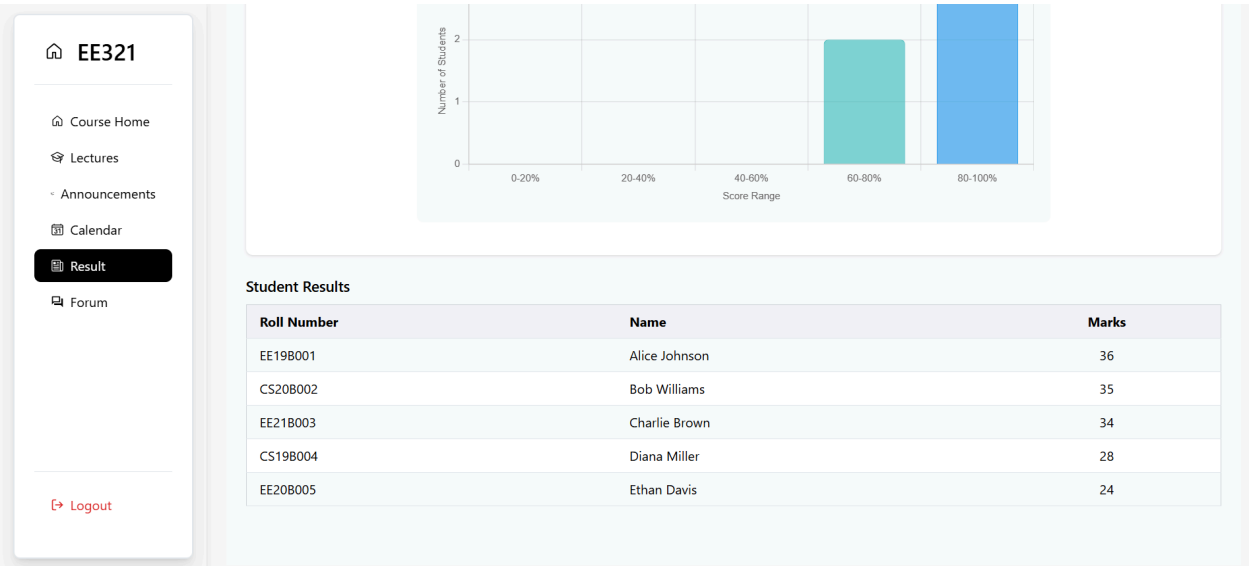
This unit also works fine and when faculty click on any exam the frontend displays the details of the exam correctly. At the top it displays basic details of the exam like exam name, total marks, weightage, mean, median, maximum, standard deviation.



Then it displays the graph of class performance which is basically a frequency distribution of marks. I have checked it with many data points and this unit correctly makes the graph for all of them so this unit is working as expected.

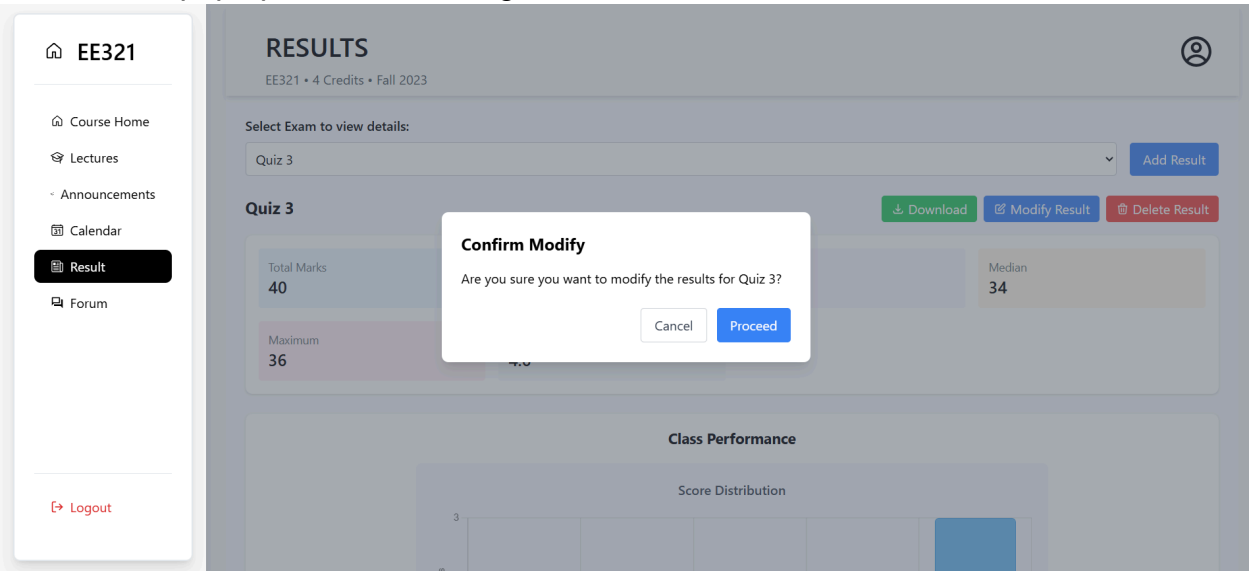


Then it display a table of marks obtained by every student enrolled in the course.

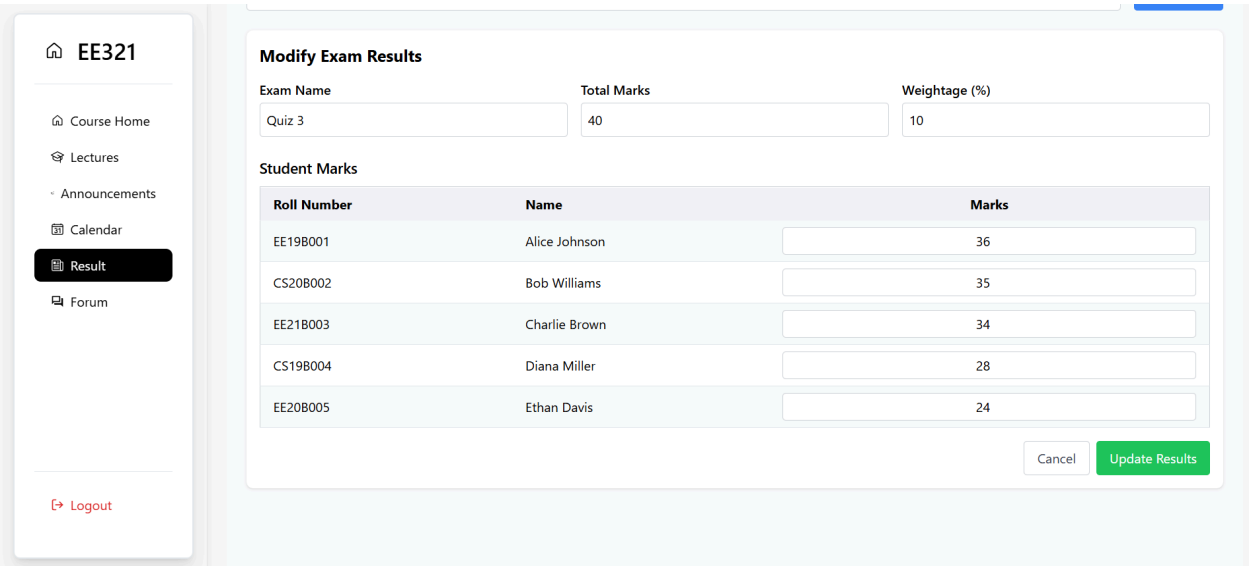


Modify result option-

When faculty click on the ‘modify result’ button the interface should give a confirmation pop up and it is working fine.

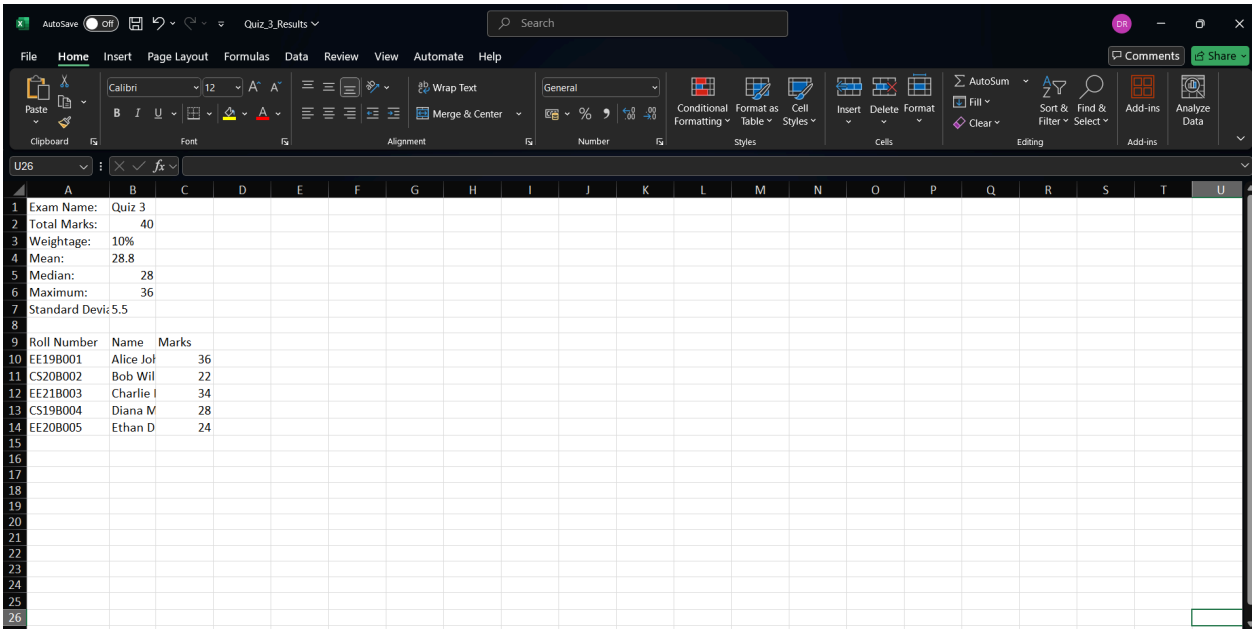


Once faculty click on the proceed button a form for modifying exam results opens as expected and faculty can update the details for the exam from here.



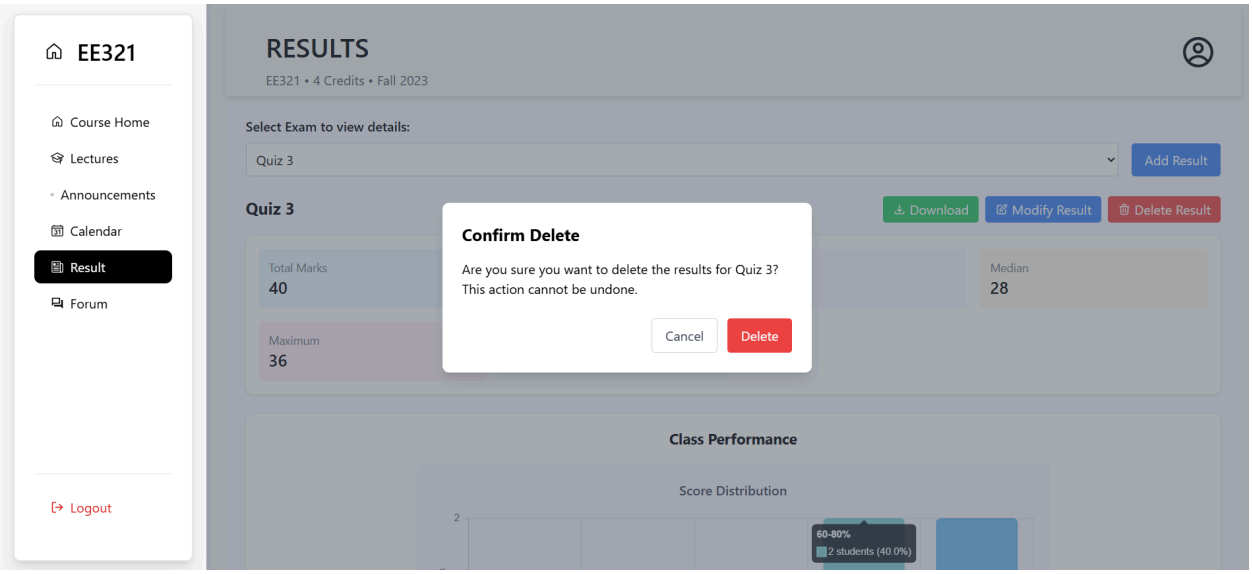
Download result-

When faculty click on the ‘Download’ button result details for the exam get downloaded in an excel format. This helps faculty to export the data. This feature works fine in all the scenarios.



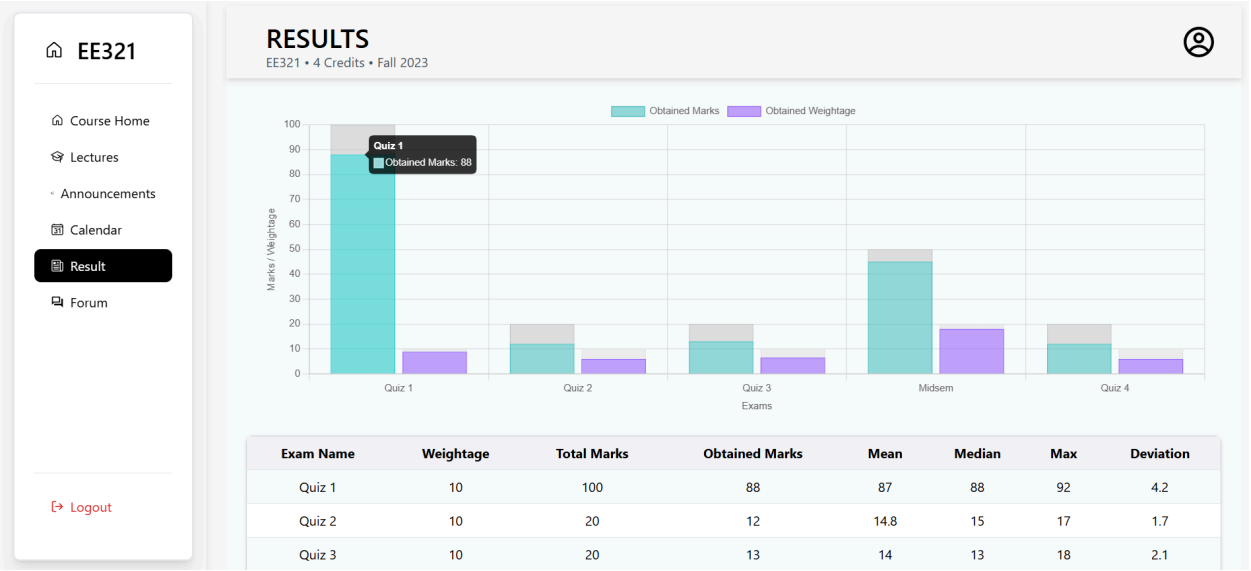
Delete result-

Faculty can delete the result for any exam by clicking on the ‘Delete result’ button. Once faculty clicks on it a confirmation pop-up comes up as expected and when faculty clicks on the ‘Delete’ button in the pop-up also the result gets deleted. This unit works fine in most of the cases.



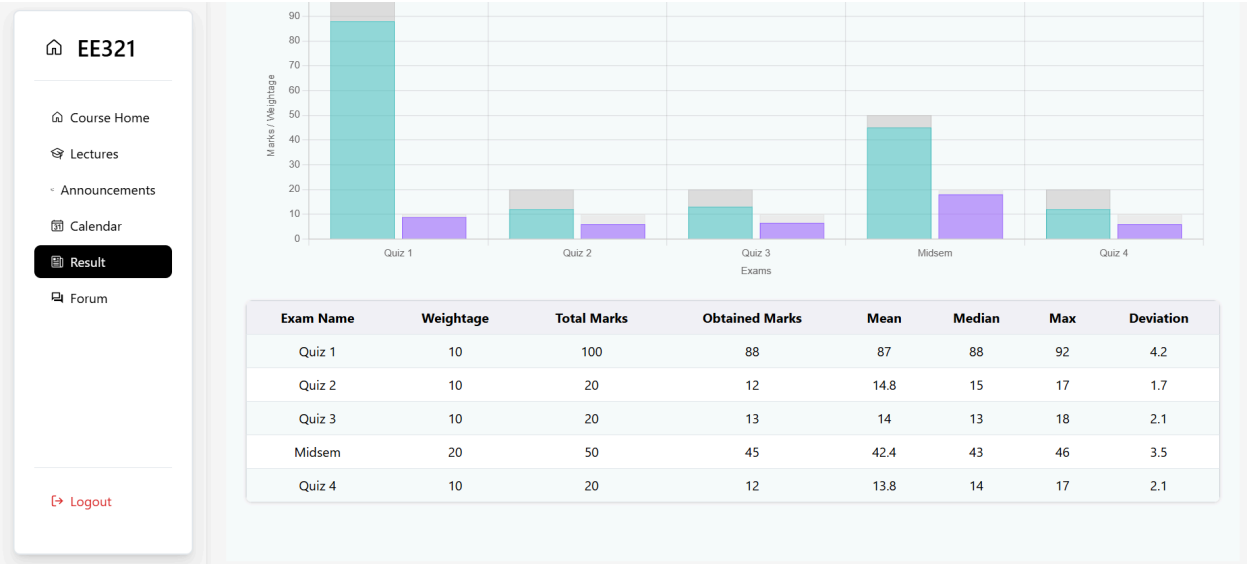
Student view-

The student only has read access to the results related to him. When a student clicks on the result tab he can see a graph telling his performance in various exams till then. The graph shows both the marks and the weightage that he has scored from the total.



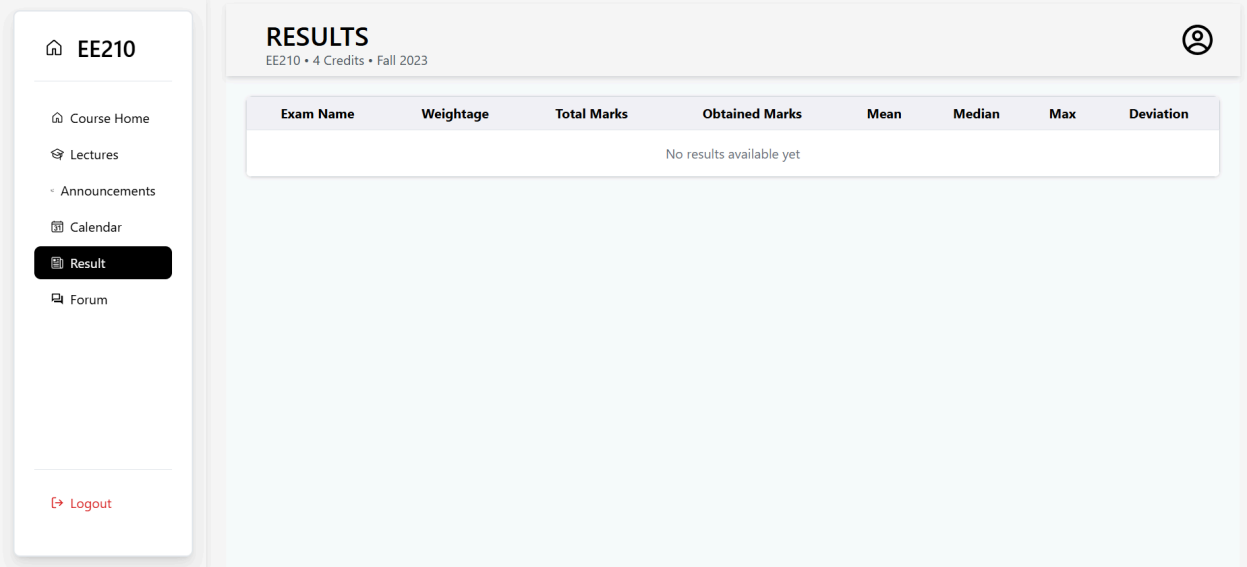
Exam Name	Weightage	Total Marks	Obtained Marks	Mean	Median	Max	Deviation
Quiz 1	10	100	88	87	88	92	4.2
Quiz 2	10	20	12	14.8	15	17	1.7
Quiz 3	10	20	13	14	13	18	2.1

Below the graph students can also view its result and other related information in a tabular format. This unit works fine in all the normal cases.



Exam Name	Weightage	Total Marks	Obtained Marks	Mean	Median	Max	Deviation
Quiz 1	10	100	88	87	88	92	4.2
Quiz 2	10	20	12	14.8	15	17	1.7
Quiz 3	10	20	13	14	13	18	2.1
Midsem	20	50	45	42.4	43	46	3.5
Quiz 4	10	20	12	13.8	14	17	2.1

One edge case occurs when there are no results available in a course. This situation is also handled by our unit by clearly indicating the absence of results.



Structural coverage:

In most of the backend units, I have used **function coverage** and **branch coverage** as the criteria for exhaustive testing.
For frontend units I have used **branch coverage** as the criteria as it ensures the proper working in all the different scenarios.

Additional comments:

One minor bug was found in one of the backend unit and it was rectified then only.

Lectures

Unit details:
Involves the testing of the Lectures Database and APIs. The Lectures database is unique to each course, where a Heading object has a subheading and a subheading object has a lecturer along with a body, along with the corresponding frontend. Only Faculties associated with that course can create lectures in that course, but they can be viewed by all students and other Faculty members.

Test Owner:

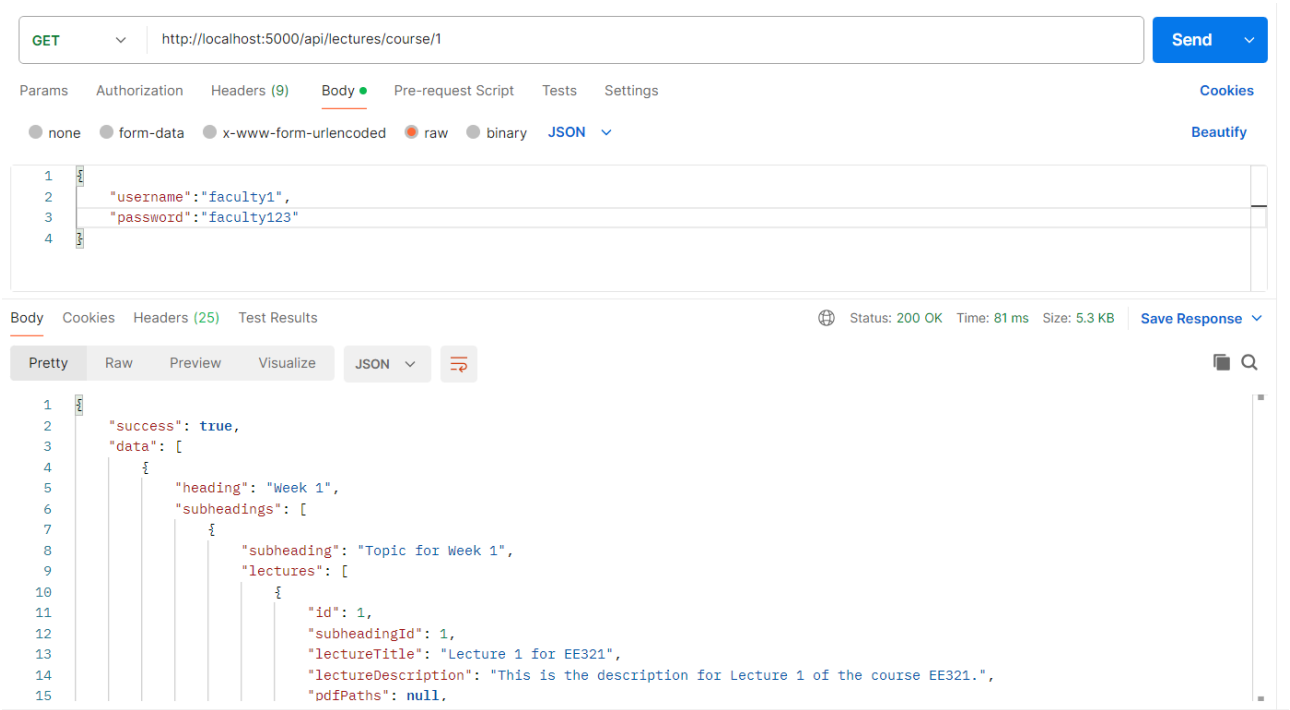
Abhijeet Agarwal, 210025

Test Date:

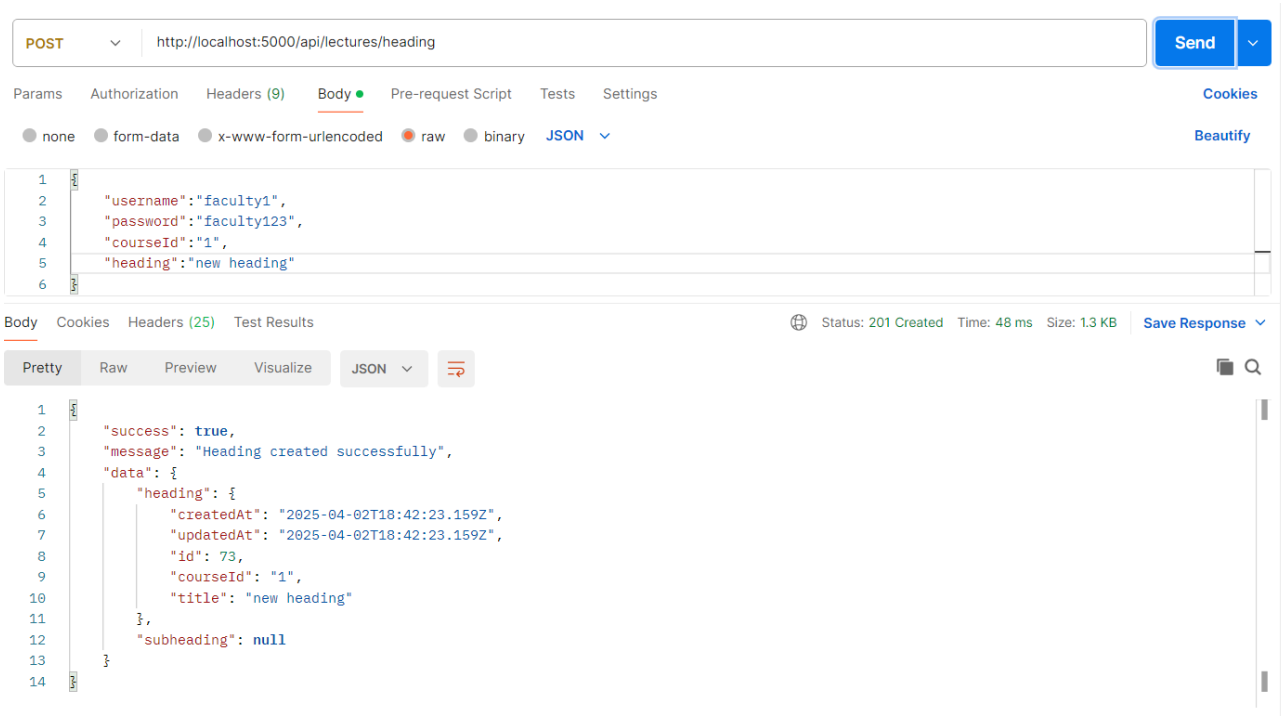
2 April 2025

Test Results:

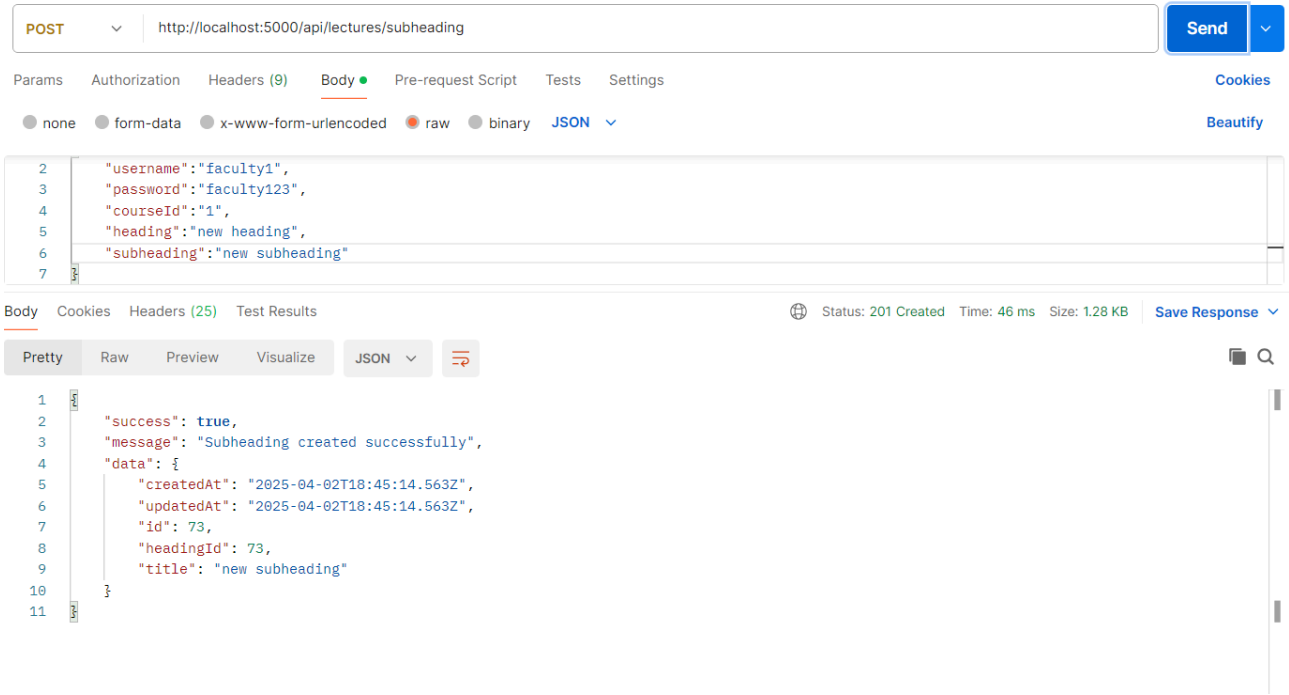
Getting all Lectures: Test Successful



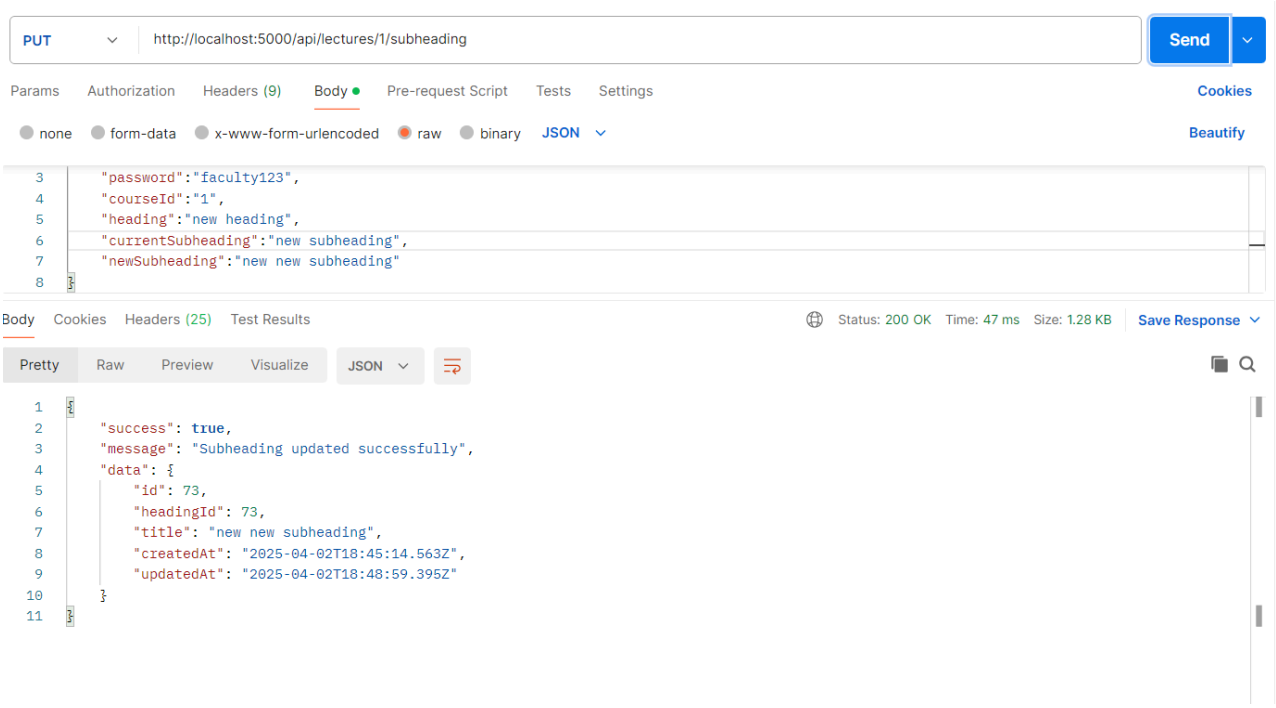
Creating new Module: Test Successful



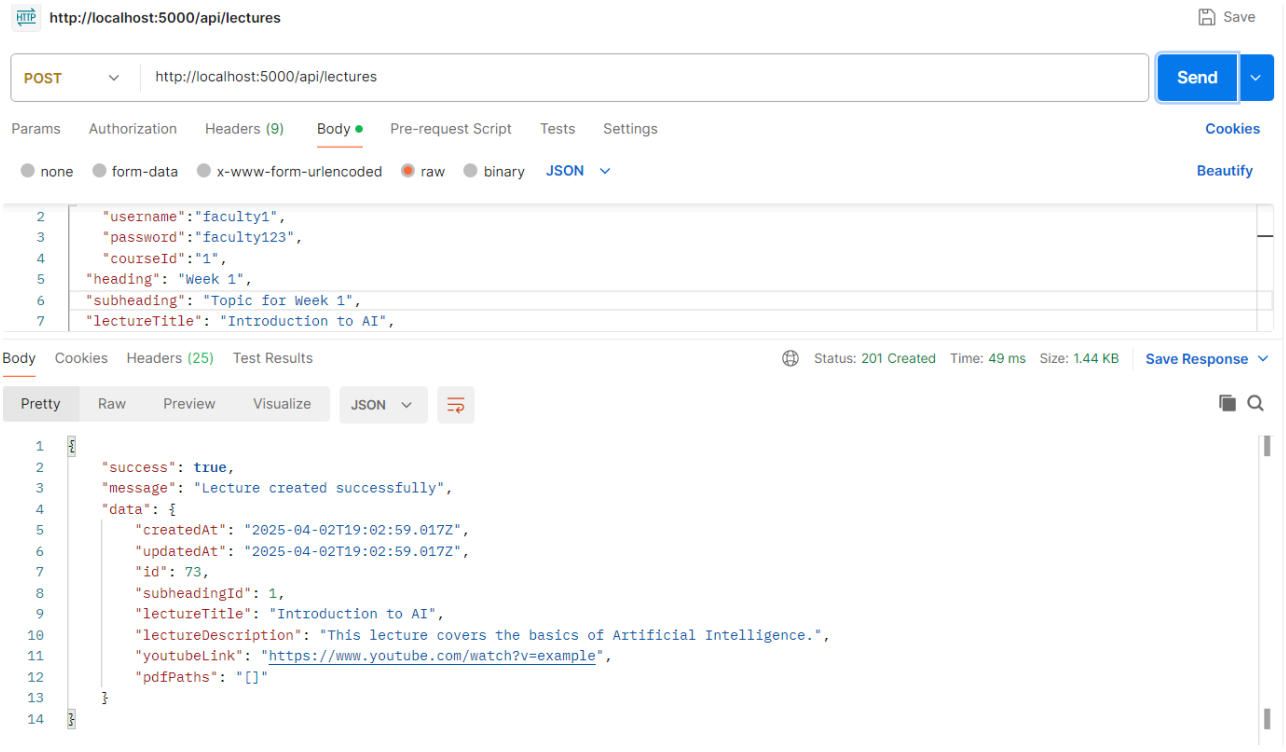
Creating new Section: Test Successful



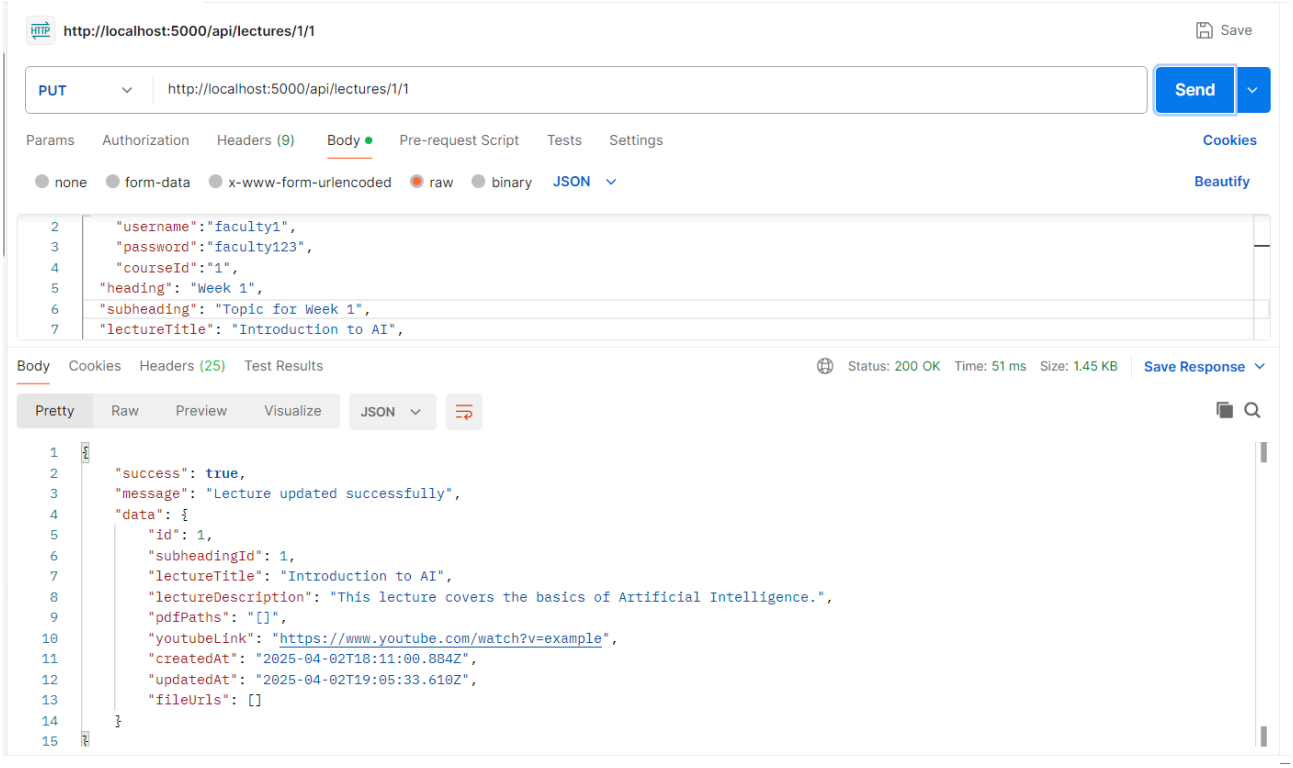
Editing Section: Test Successful



Creating new Lecture: Test Successful



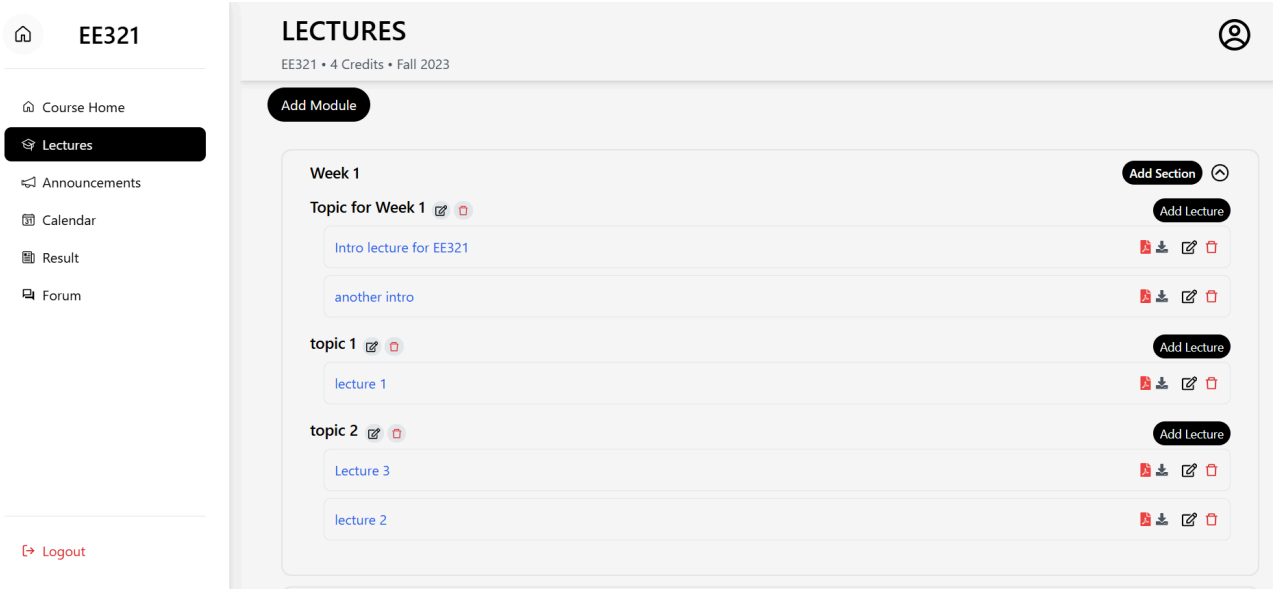
Editing Lecture: Test Successful



The test results for the frontend units, which will serve as the user interface for the lectures page, are as shown. Since the backend is not yet connected, the tests were conducted using static data to verify that the frontend components are functioning as expected

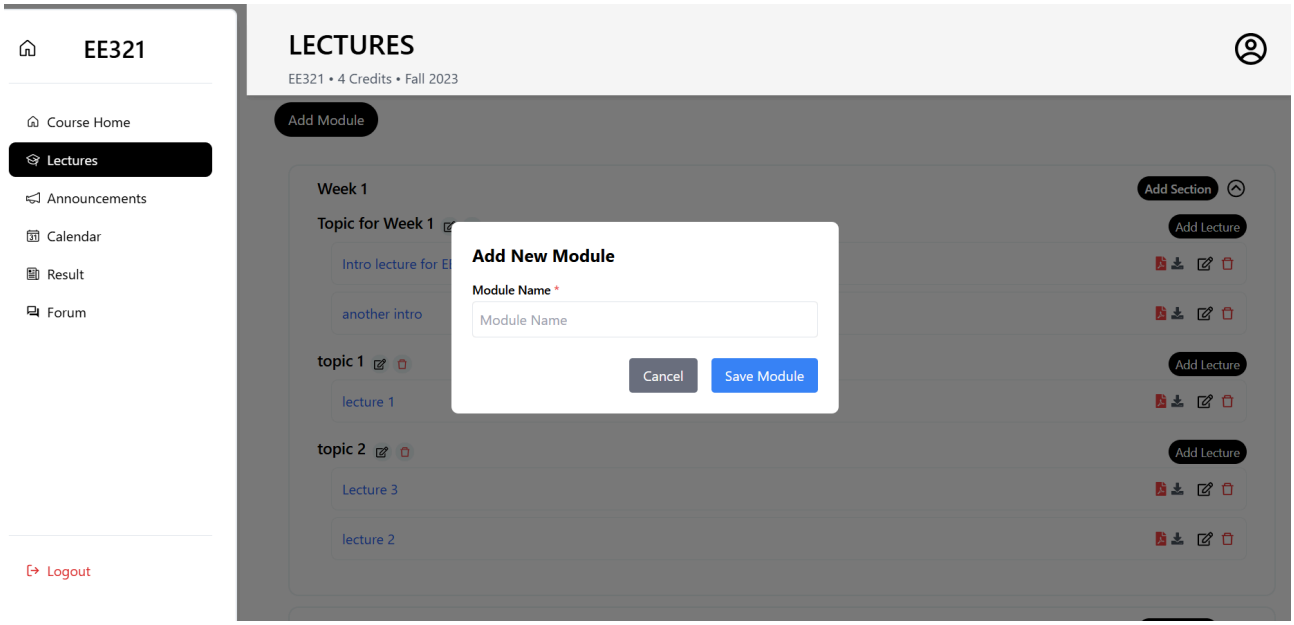
The faculty lectures page-

This page renders successfully in all the cases.



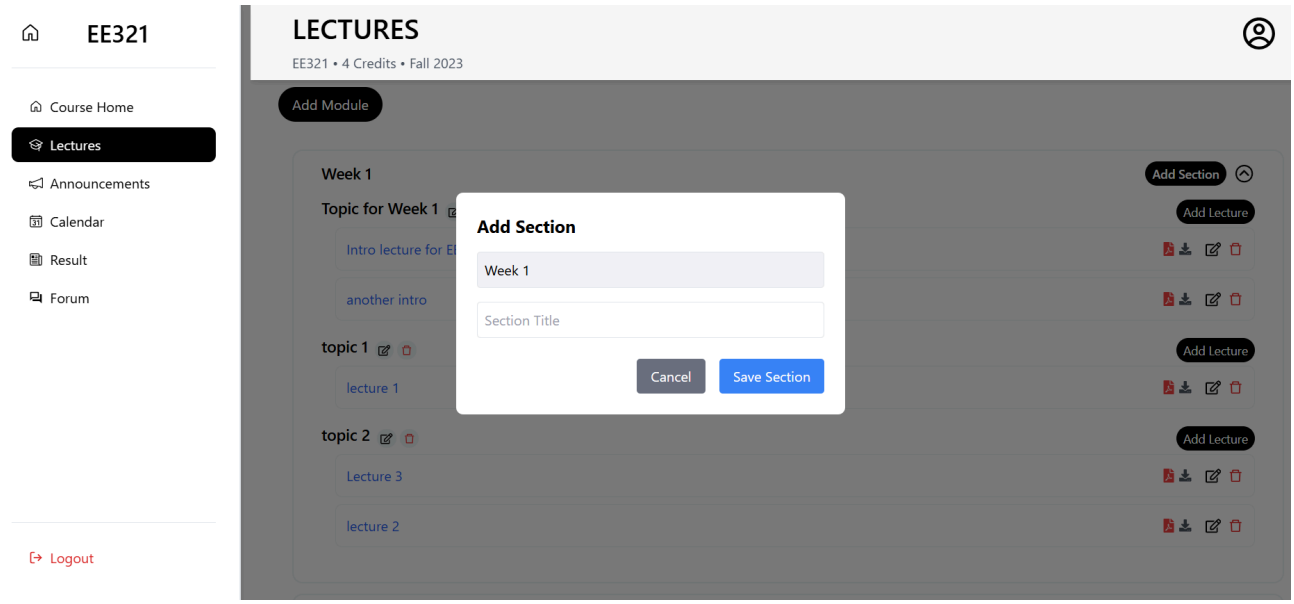
Adding new module:

This page works successfully and shows the required module name for creating a new module.

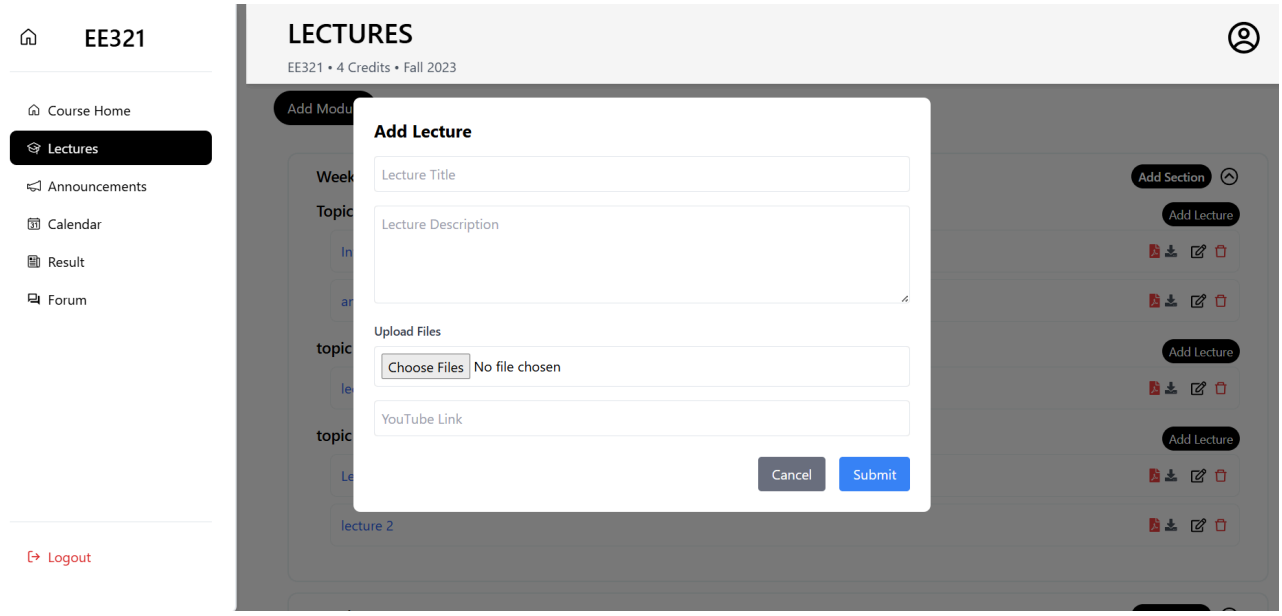


Adding new section:

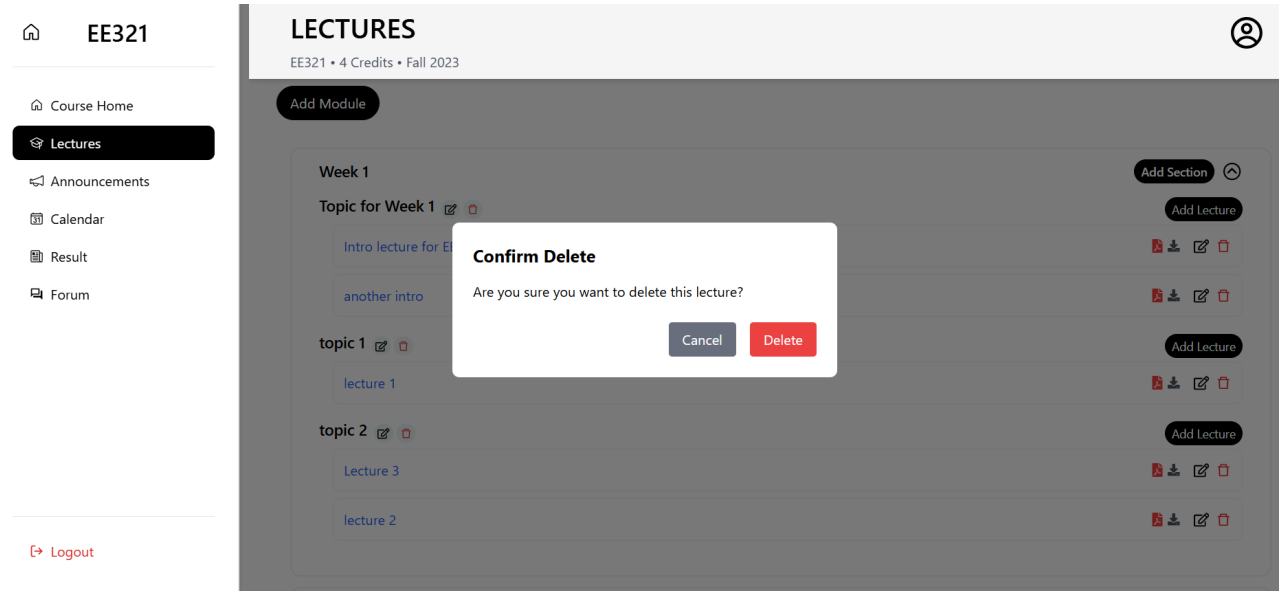
This page works successfully and shows the required section name for creating a new section.



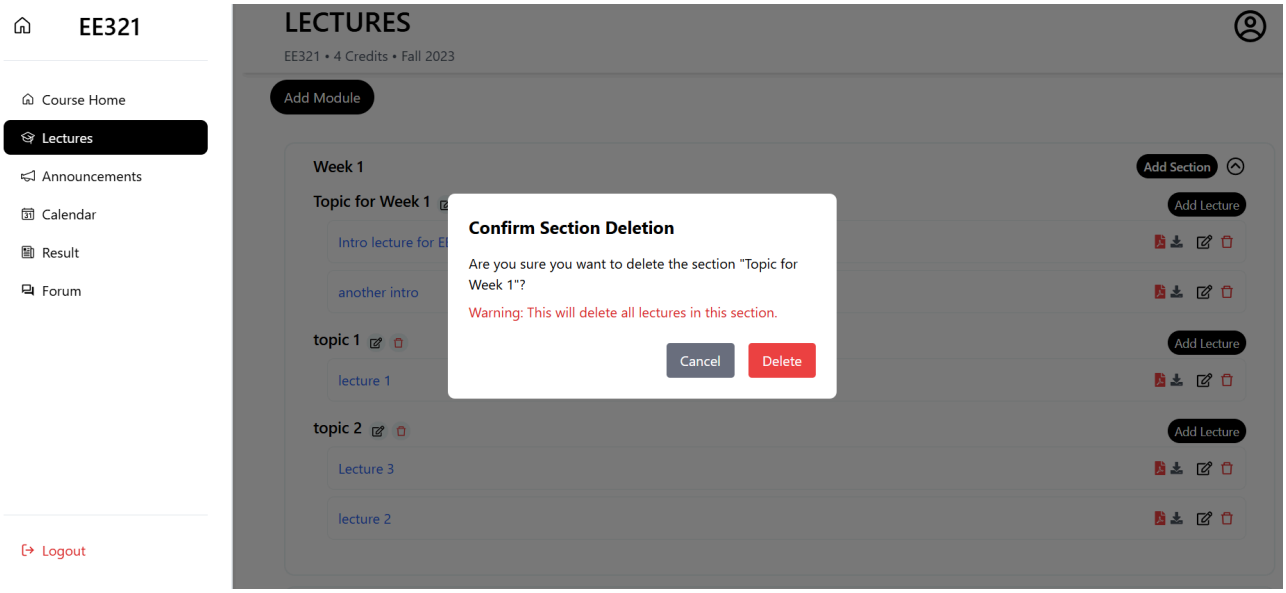
Adding new lecture:
This page works successfully and shows all the required fields for creating the lecture.



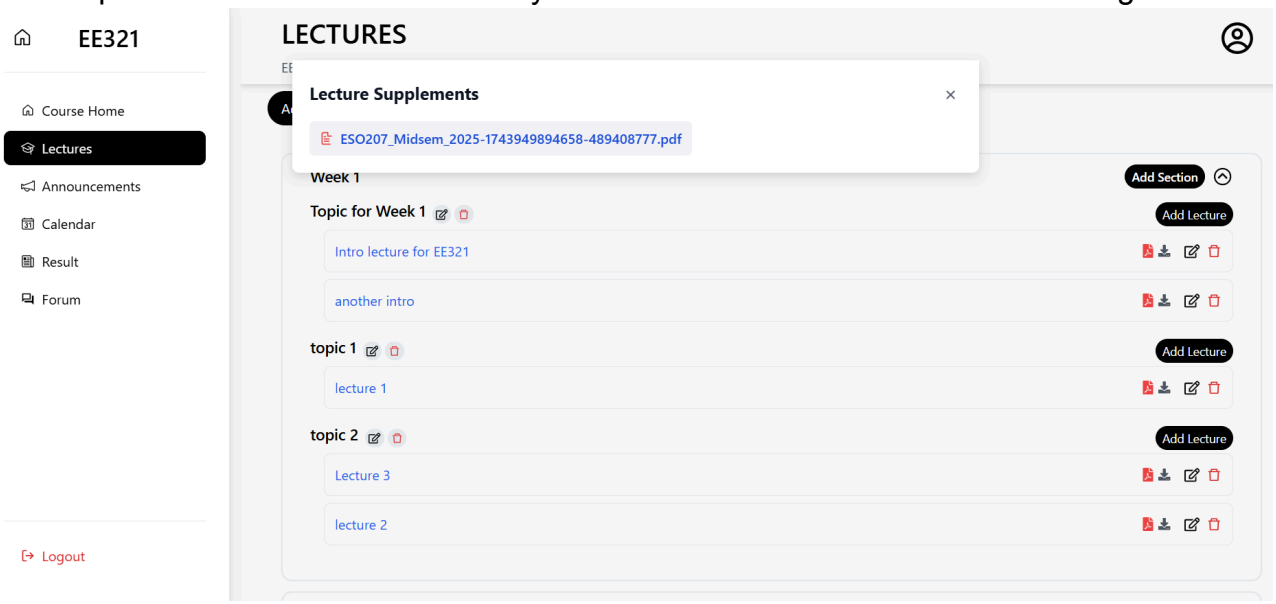
Deleting lecture:
This page works successfully and asks for confirmation before deleting a lecture.



Deleting section:
This page works successfully and asks for confirmation before deleting a section along with the warning that all the lectures within the section will be deleted.

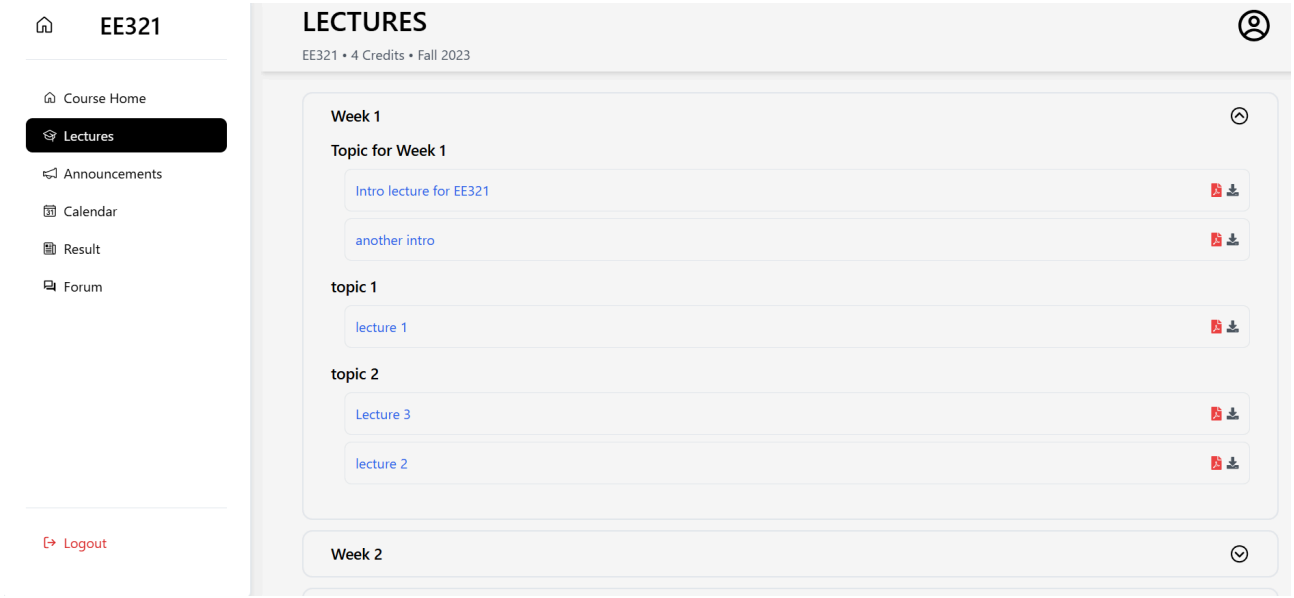


Downloading resources:
All the uploaded resources are correctly shown and are available for downloading.



The student lectures page-

This page renders successfully in all the cases and works correctly.



Structural Coverage:

For most backend units, exhaustive testing was performed using both **function coverage** and **branch coverage** as criteria. For frontend units, **branch coverage** alone was used as the testing criterion, ensuring proper functionality across all possible scenarios

Forgot Password

Unit Details:

Involves the testing of Forgot Password Database and APIs.

Test Owner:

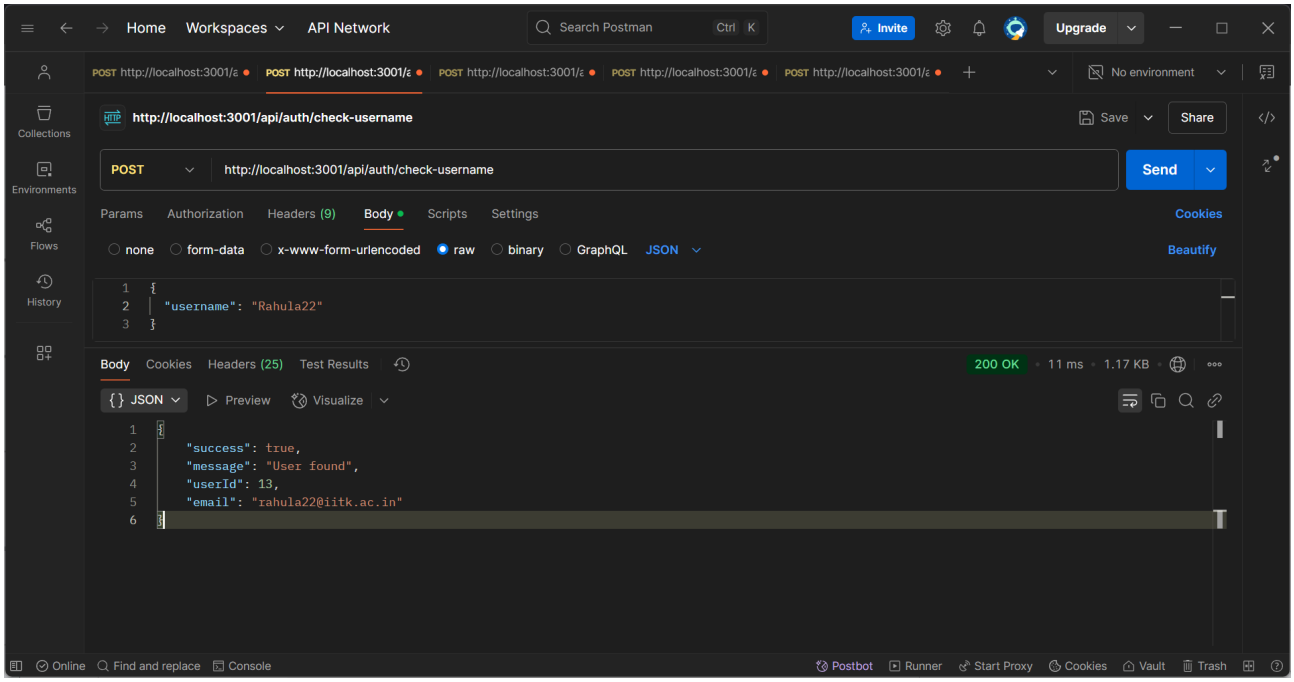
Rahul Ahirwar, 220856

Test Date:

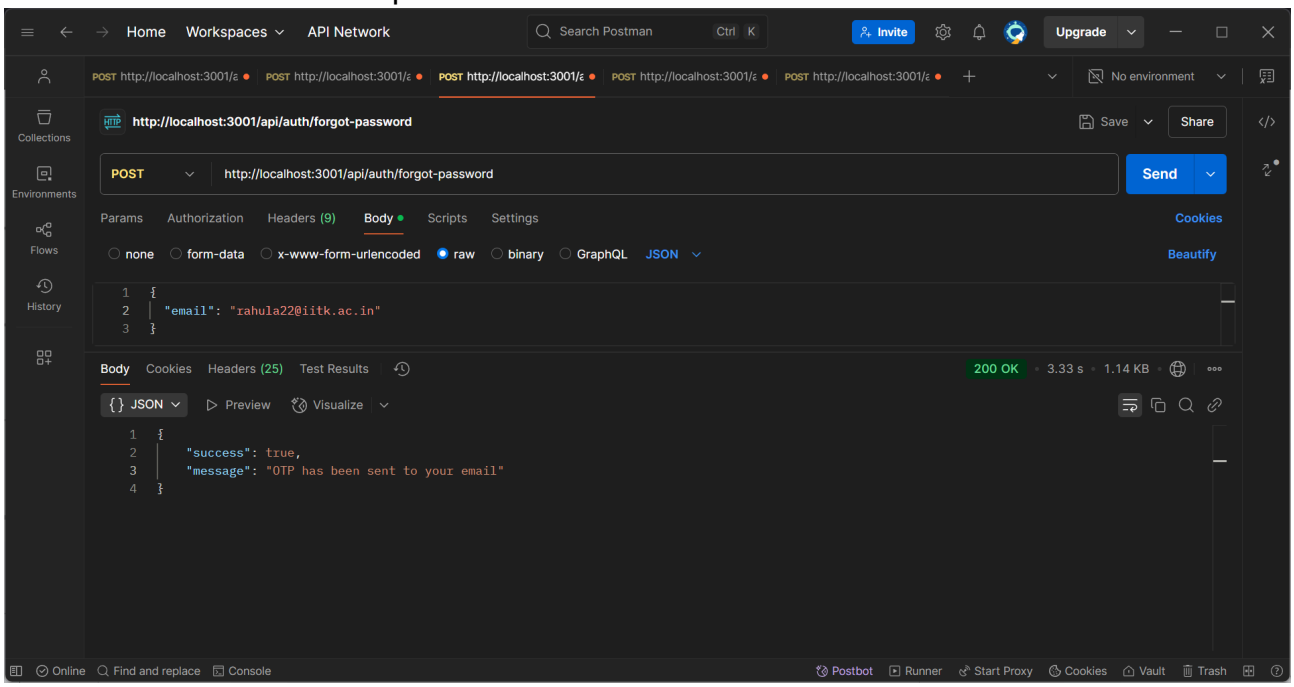
2 April 2025

Test Results:

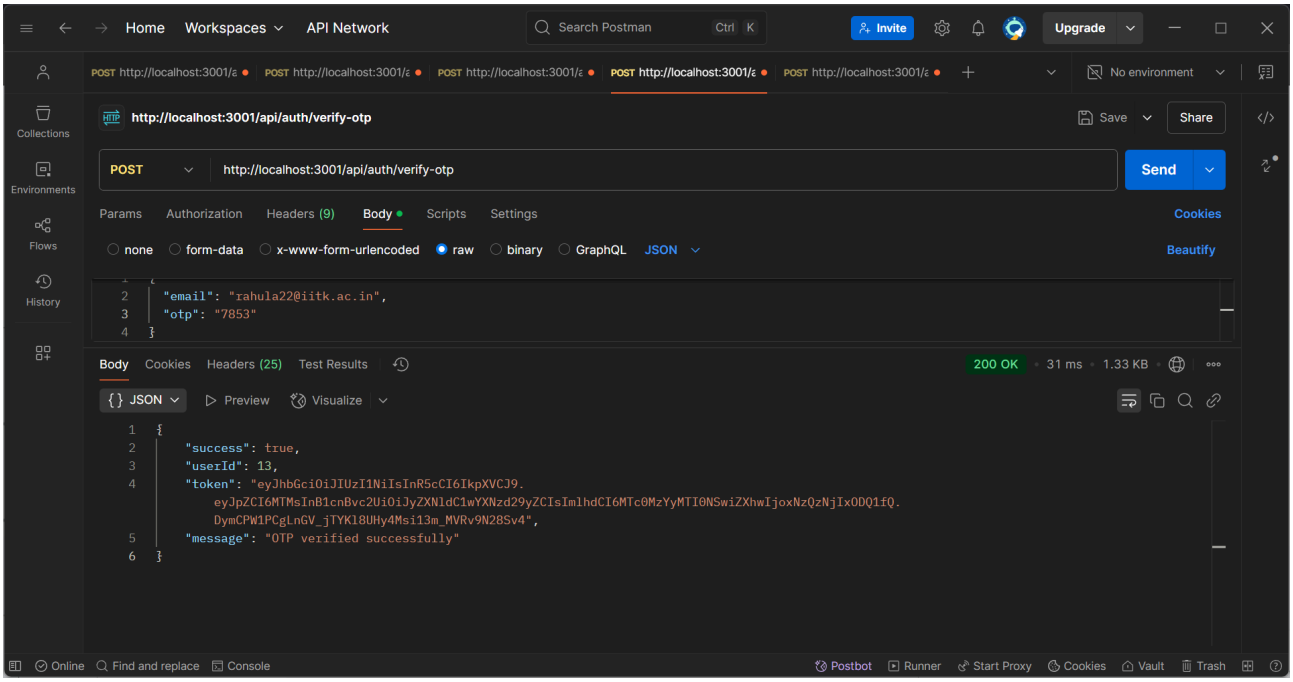
- API to check username exists : Test Successful



- API to send OTP for password reset : Test Successful

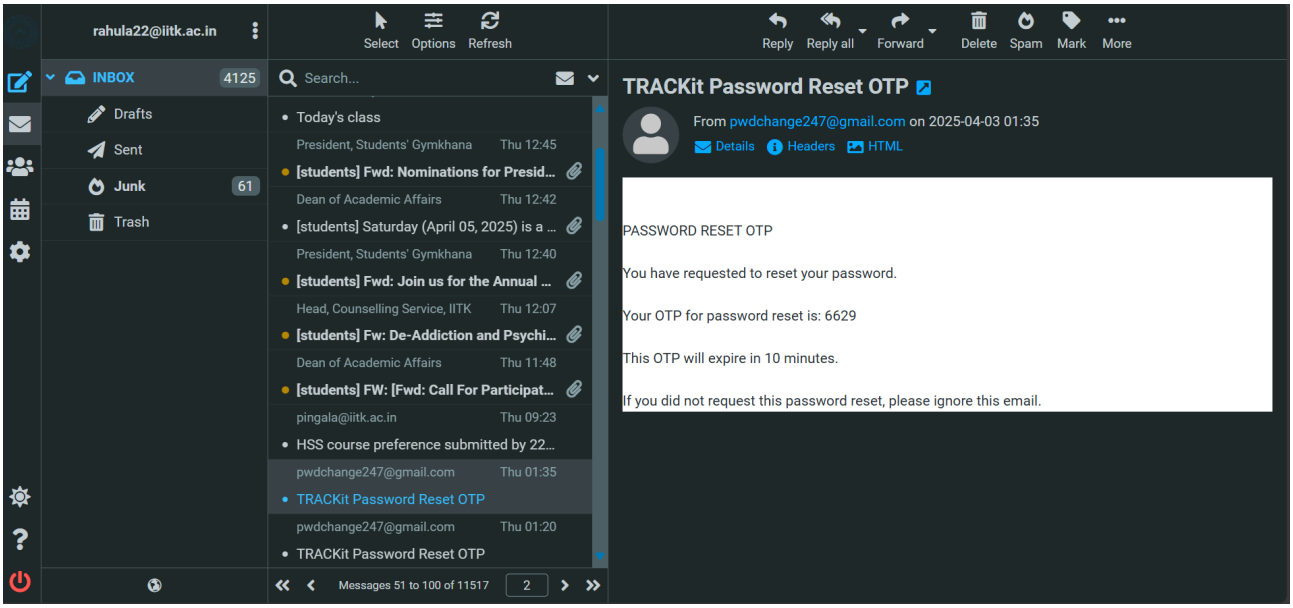


- API to Verify OTP : Test Successful

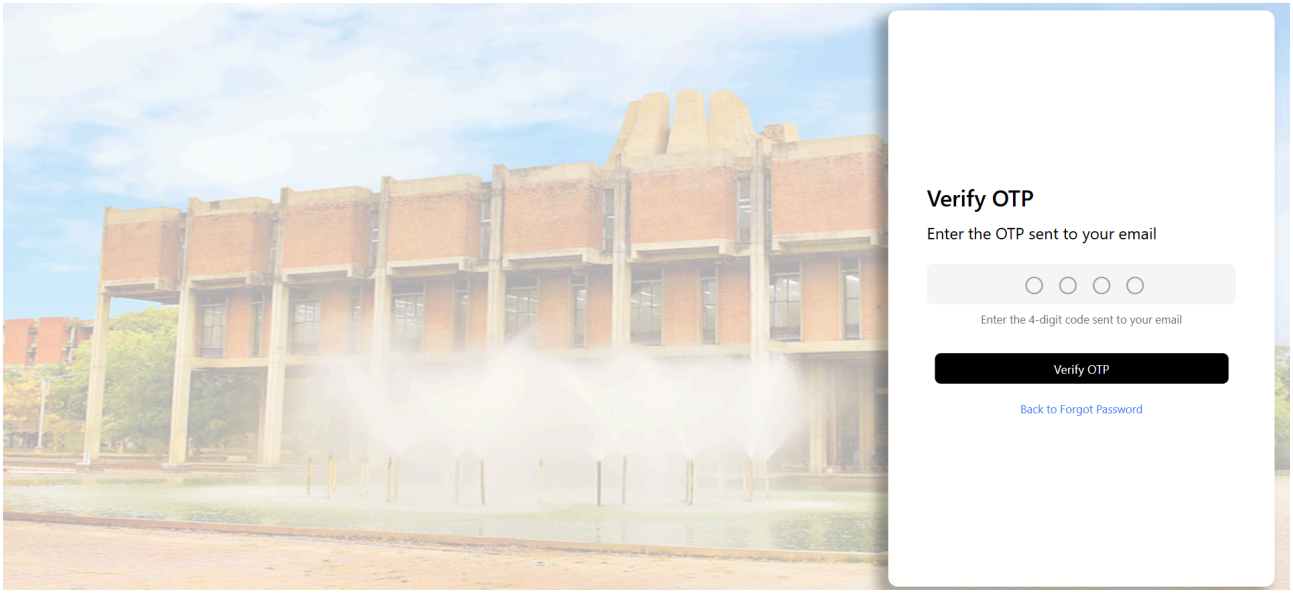


On Clicking the forgot password link, the user is navigated to this page where he/she can enter the username and continue with the process to change password.

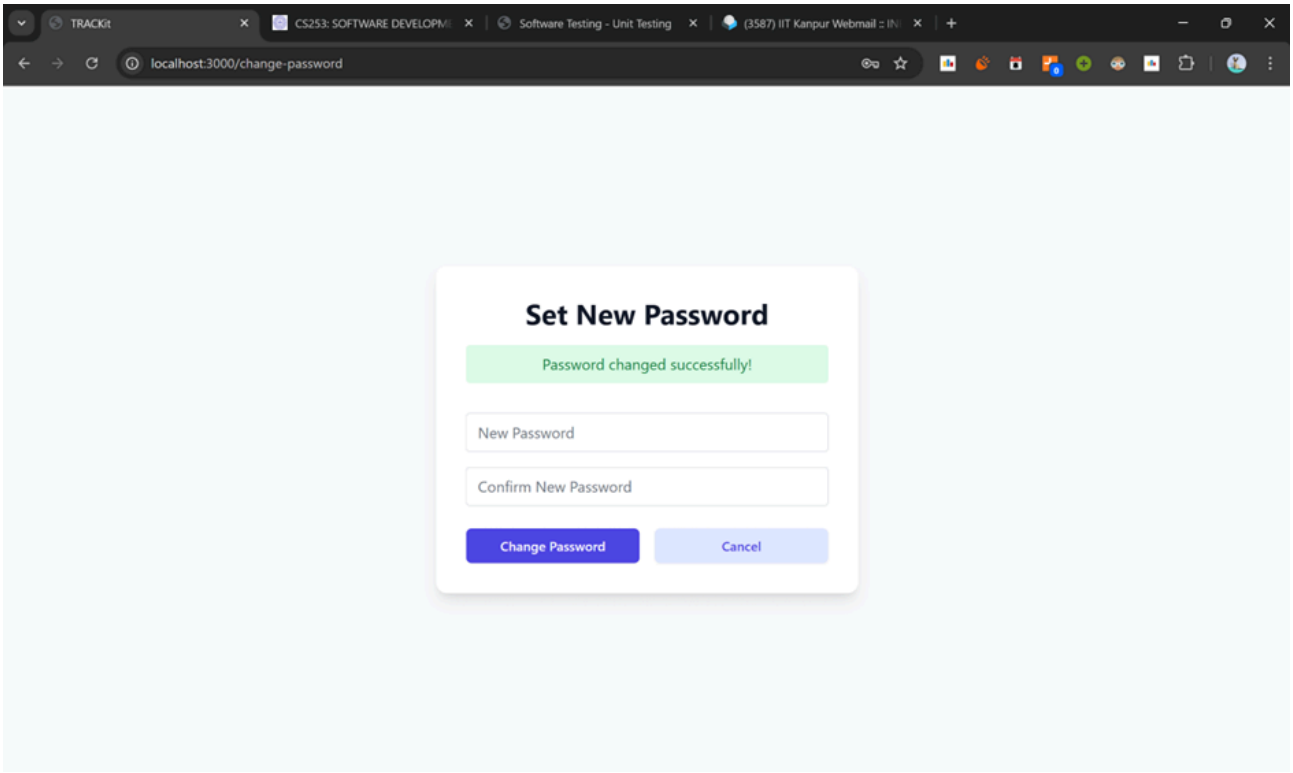
On verifying the username existence they will get the OTP on their email-Id to verify their identity.



OTP verification



Once the OTP gets verified, the user can change the password and get the message “Password changed successfully”.



Structural Coverage:

Functional coverage and **branch coverage** is used as the main criteria of coverage

3 Integration Testing

Integration testing involves testing whether the backend and frontend units interact in the right way and produce the desired behaviour. Both backend and frontend units have been individually tested by the unit testing team. We have performed integration testing as follows. We gave the input to the frontend and observed final changes that occur after the backend processes the request sent by the frontend. Correct final behaviour implies that the interaction between backend and frontend was alright. Apart from this, integration testing also involves the interaction between different components, redirection of pages and updation of pages whenever a change is made to the database from anywhere.

Admin : Add - Student

Module Details:
This unit focuses on the functionality for adding students to the system via two methods: manual entry and bulk upload using a CSV file. The manual entry form allows the admin to input individual student details, while the bulk upload feature enables adding multiple students at once by uploading a properly formatted CSV file. The unit also includes validation for input fields and file uploads, ensuring data integrity and error handling.
Key Features:

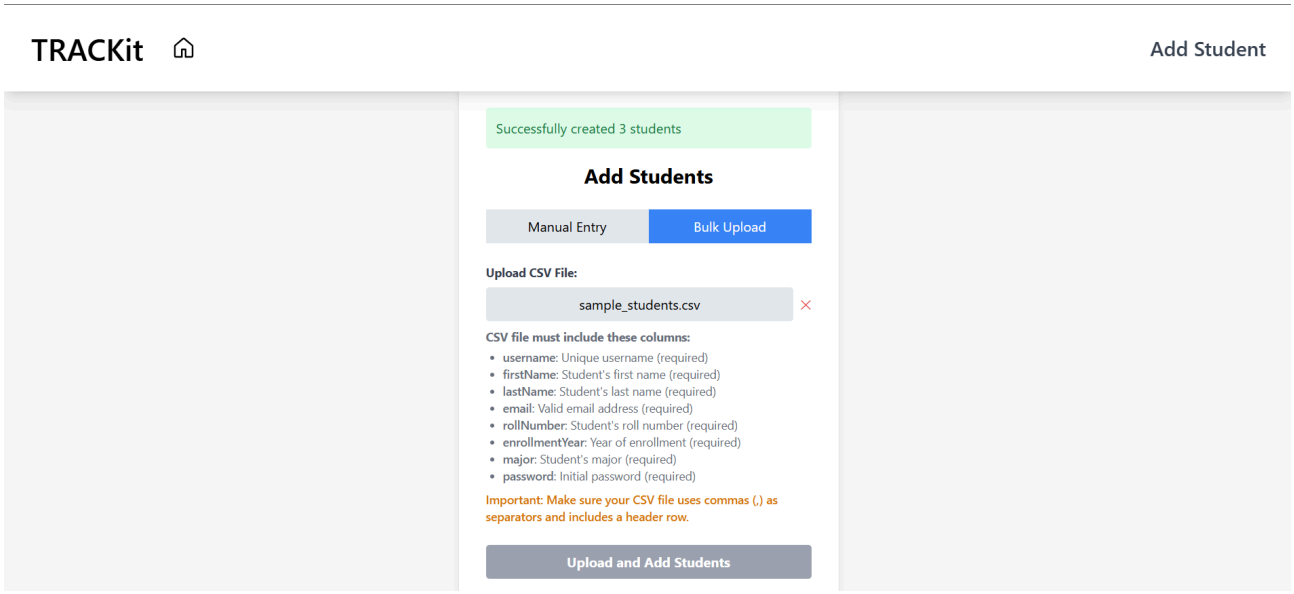
1. Manual entry of student details, including username, email, password, first name, last name, roll number, enrollment year, and major.
2. Bulk upload of student data using a CSV file with required fields.
3. Validation for required fields, file type, and file size (max 5MB).

Test Owner:
Sharique Ahmad, 221002

Test Date:
2 April 2025 - 3 April 2025

Test Result:
Manual Entry of User is successful as a New User named “Sharique Ahmad” has been added

Successfully Added 3 students using .csv file



All the Added students were correctly shown in the Manage User.

Admin : Add - Faculty

Module Details:
This unit focuses on the functionality for adding faculty members to the system. It provides two methods for adding faculty: manual entry and bulk upload using a CSV file. The manual entry form allows the admin to input individual faculty details, while the bulk upload feature enables adding multiple faculty members at once by uploading a properly formatted CSV file. The unit includes validation for input fields, file uploads, and error handling to ensure data integrity.
Key Features:

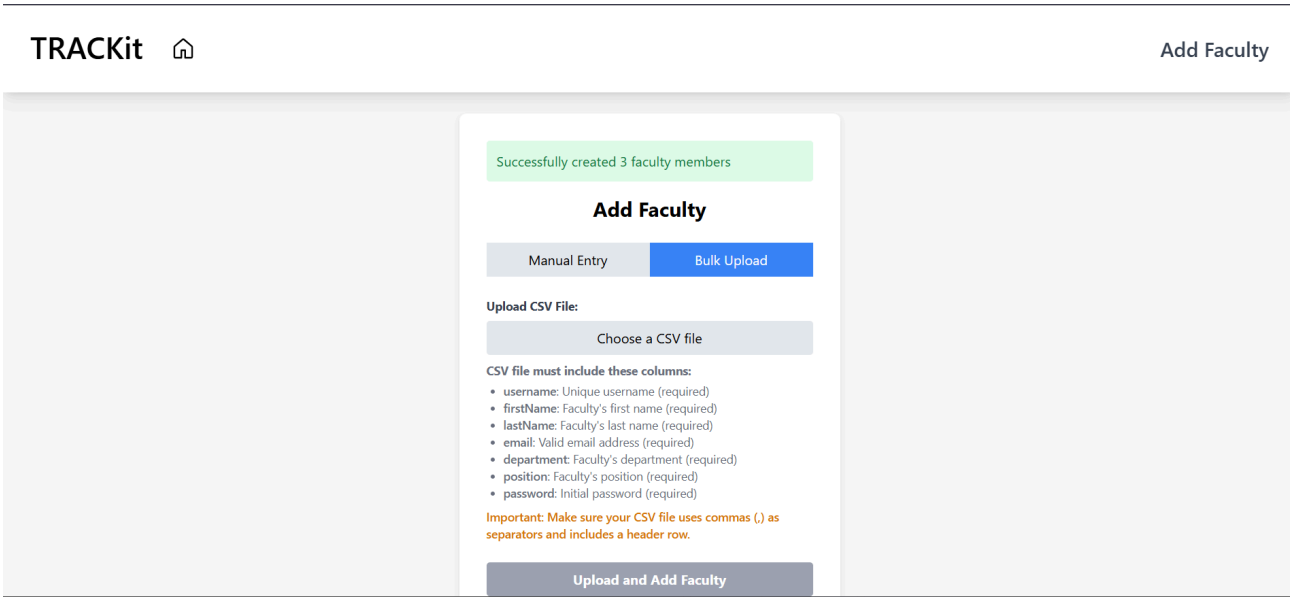
1. Manual entry of faculty details, including username, email, password, first name, last name, department, and position.
2. Bulk upload of faculty data using a CSV file with required fields.
3. Validation for required fields, file type, and file size (max 5MB).

Test Owner:
Sharique Ahmad, 221002

Test Date:
2 April 2025 - 3 April 2025

Test Result:
Successful addition of a Faculty through Manual Addition.

Successfully Added 3 Faculty through bulk addition using .csv file.



All the Added Faculty were Correctly shown in Manage Users with role “faculty”.

Admin : Create- Course

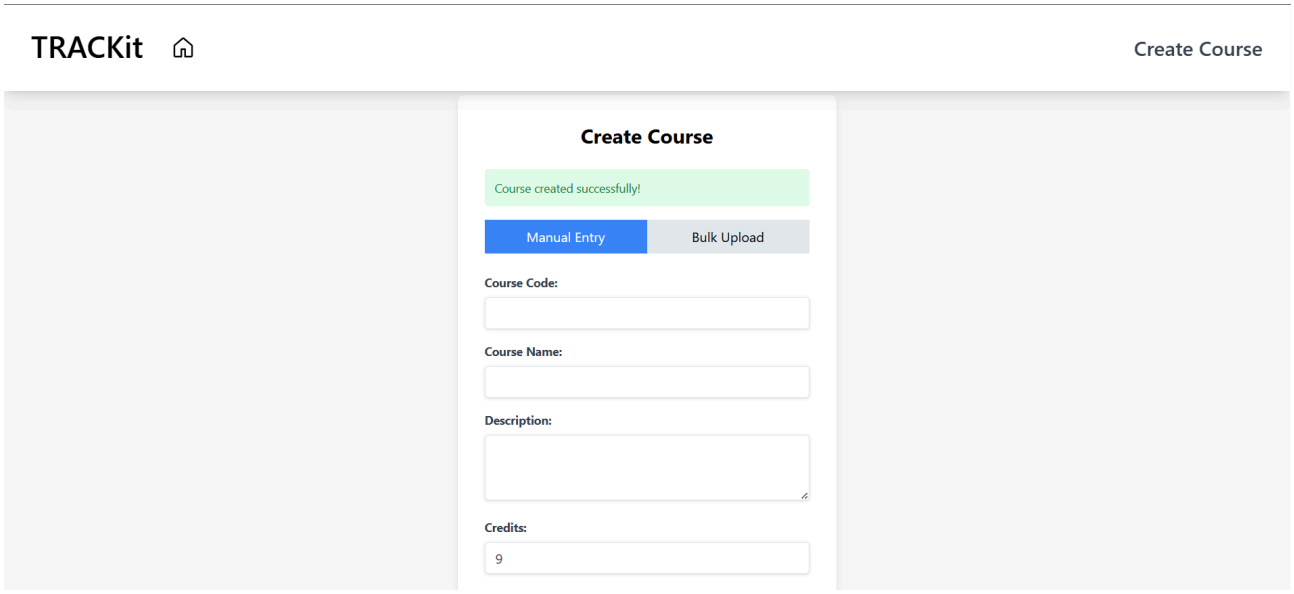
Module Details:
This unit focuses on the functionality for adding courses to the system. It provides two methods for adding courses: manual entry and bulk upload using a CSV file. The manual entry form allows the admin to input individual course details, while the bulk upload feature enables adding multiple courses at once by uploading a properly formatted CSV file. The unit includes validation for input fields, file uploads, and error handling to ensure data integrity.
Key Features:

1. Allows the admin to input individual course details
2. Enables the admin to upload a CSV file containing multiple course records
3. Validation for required fields, file type, and file size (max 5MB).

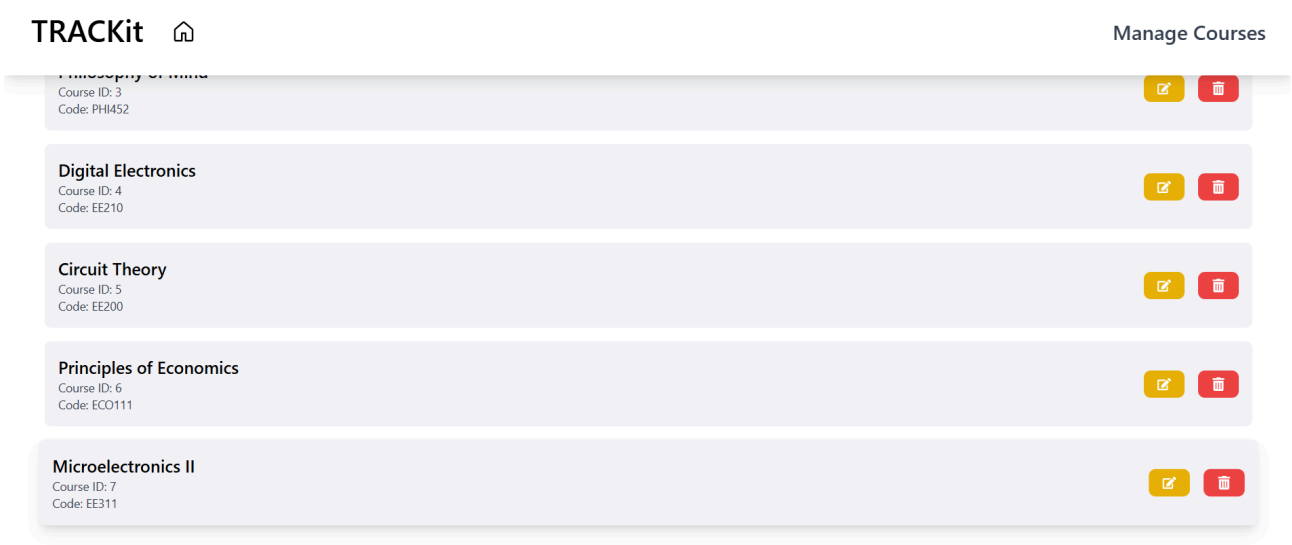
Test Owner:
Sharique Ahmad, 221002

Test Date:
2 April 2025 - 3 April 2025

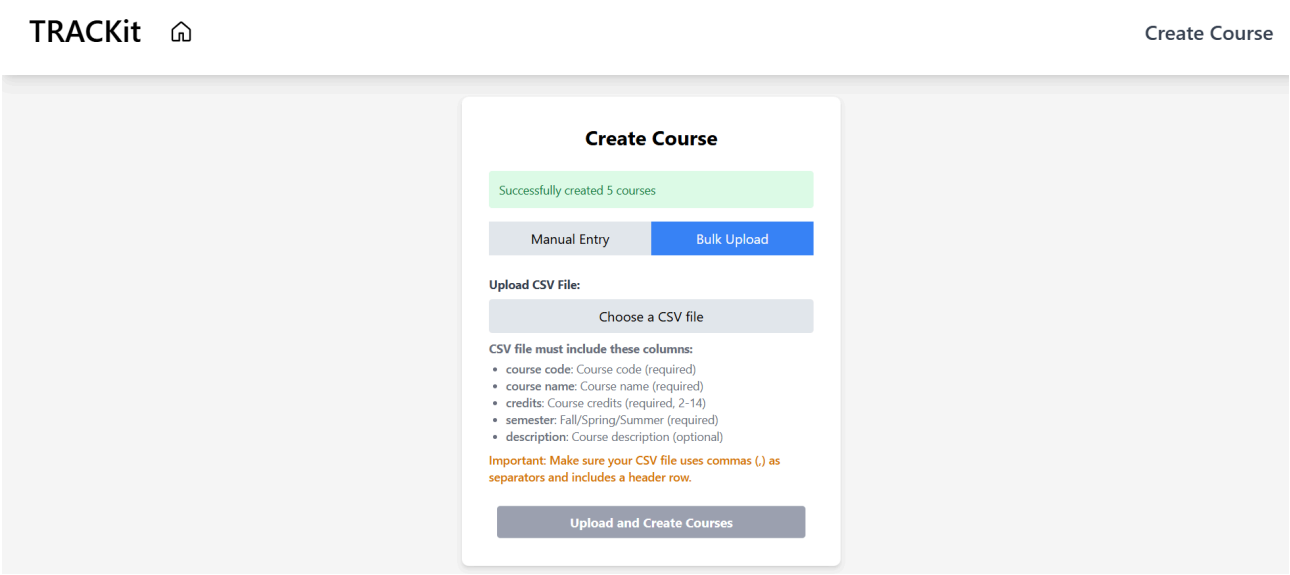
Test Result:
Successfully created a Course using manual Entry



Added course is successfully shown in the course list.



Successfully created 5 courses in bulk using .csv file.



All the Added courses were correctly shown in the Manage Course List.

Admin : Manage Users

Module Details:

This unit focuses on the "Manage Users" functionality within the Admin panel. It allows the Admin to perform CRUD (Create, Read, Update, Delete) operations on users, assign or remove courses for students and faculty, and manage role-specific fields such as department, position, roll number, and enrollment year. The unit also

includes search functionality for users and courses, ensuring seamless navigation and management.

Key Features:

1. Edit user details, including role-specific fields.
2. Assign and remove courses for students and faculty.
3. Delete users from the system.

Test Owner:

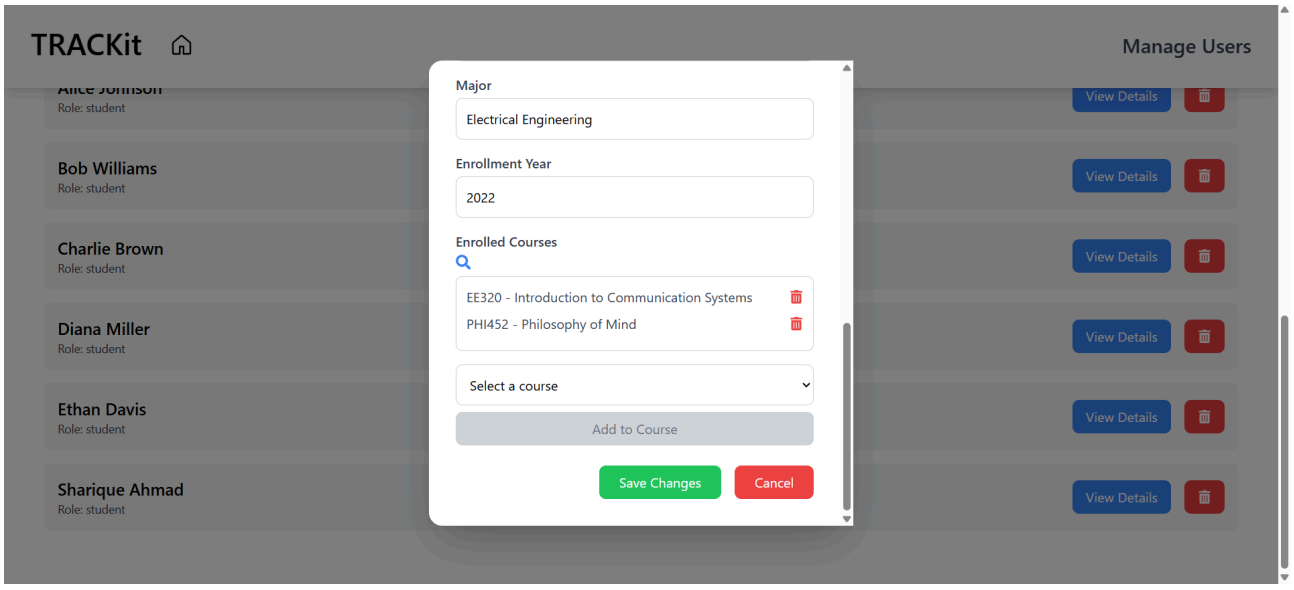
Sharique Ahmad, 221002

Test Date:

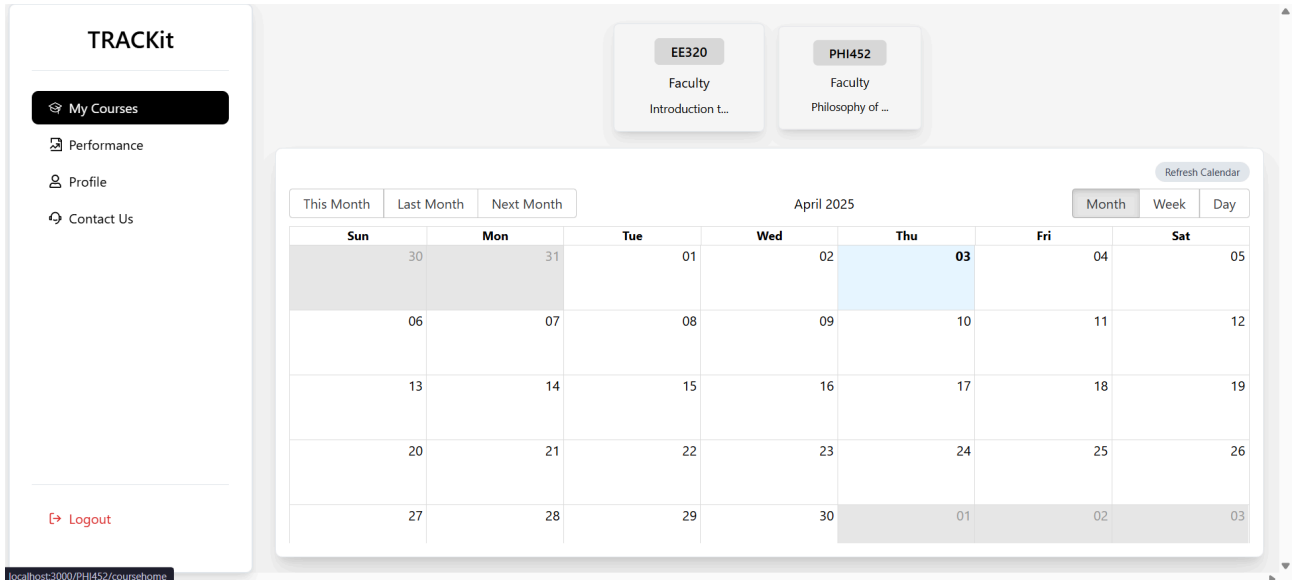
2 April 2025 - 3 April 2025

Test Result:

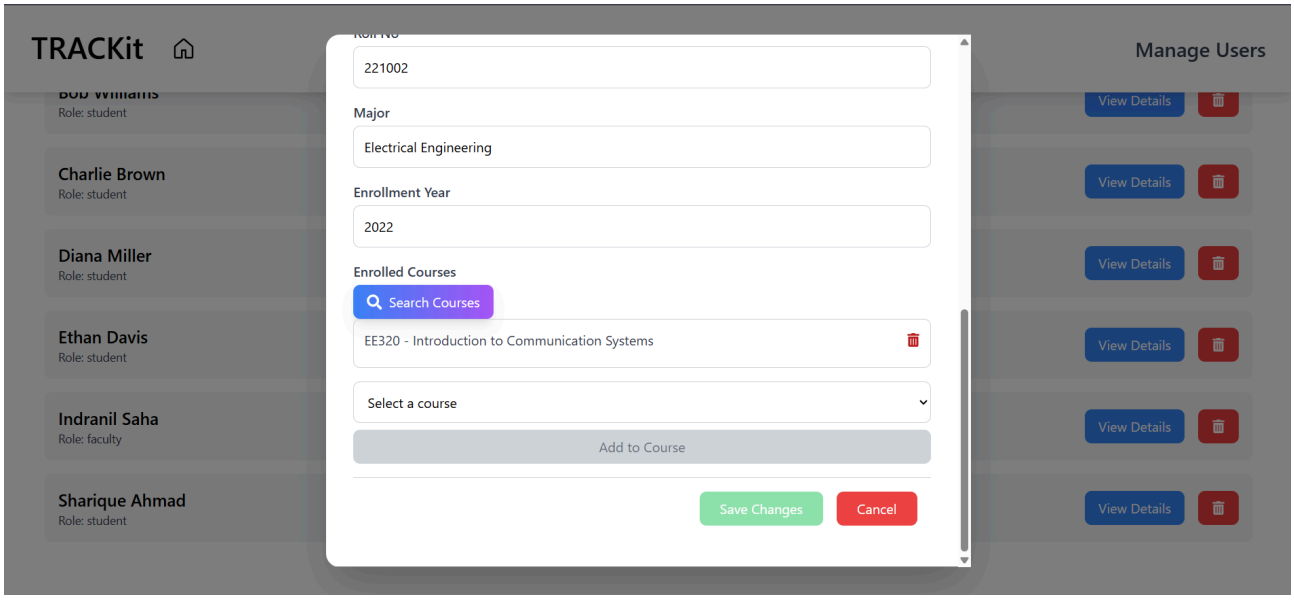
User added to some courses from Manage Users

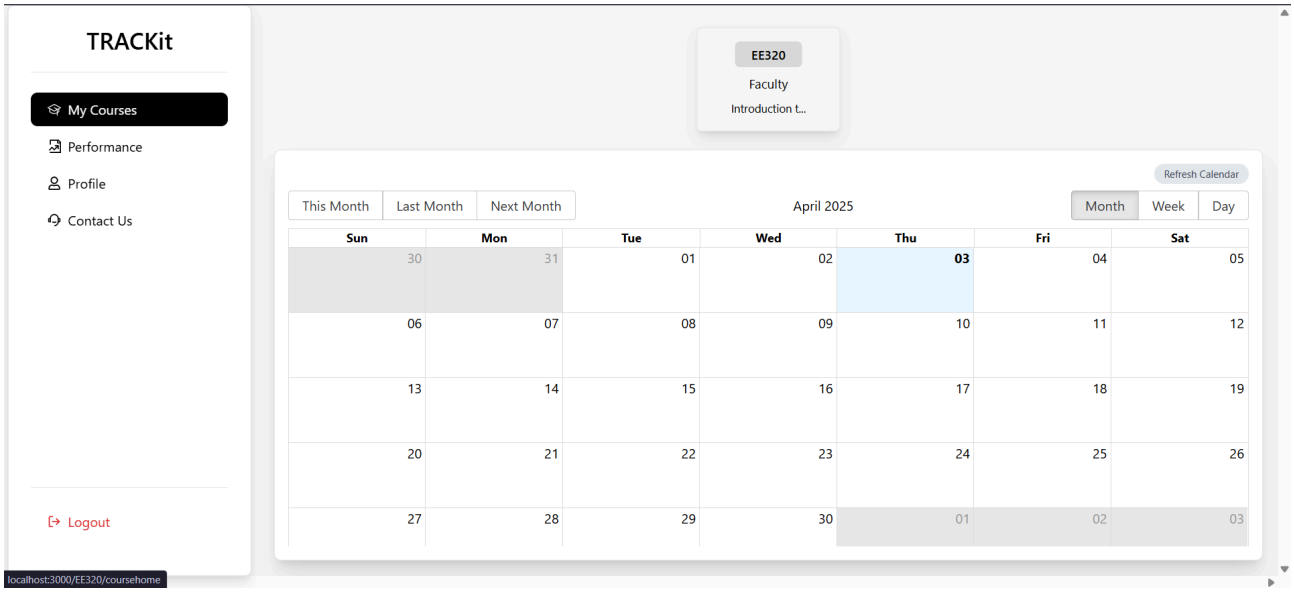


Added courses were correctly reflected in the user's Dashboard.

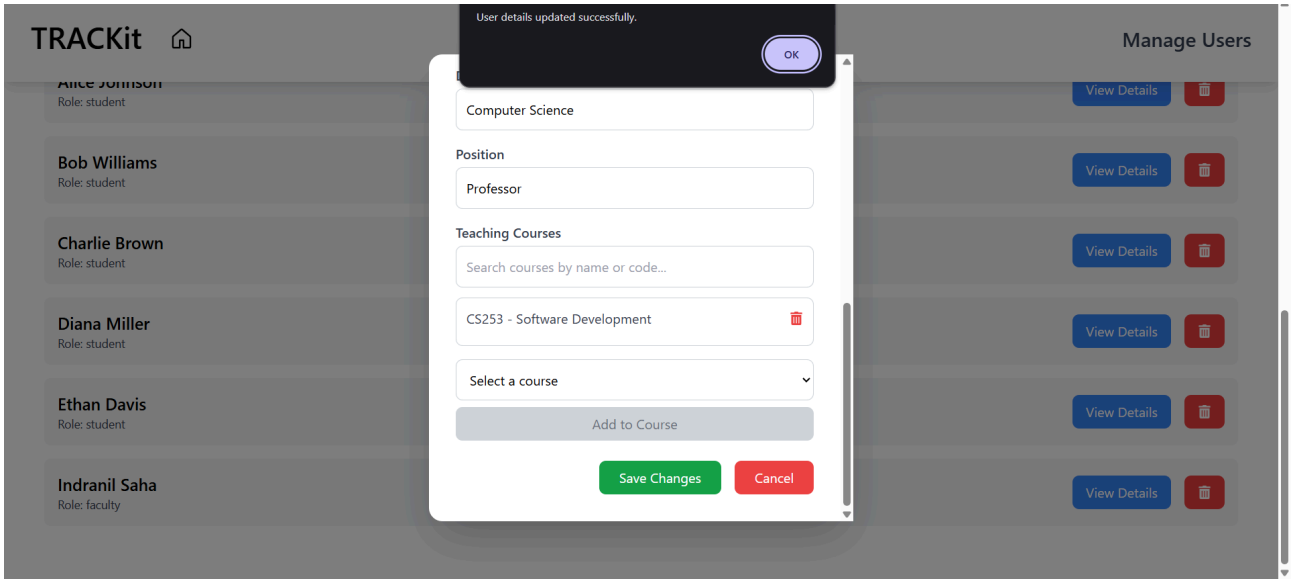


After Removing the User from a Course it is also correctly reflected in the User Dashboard.

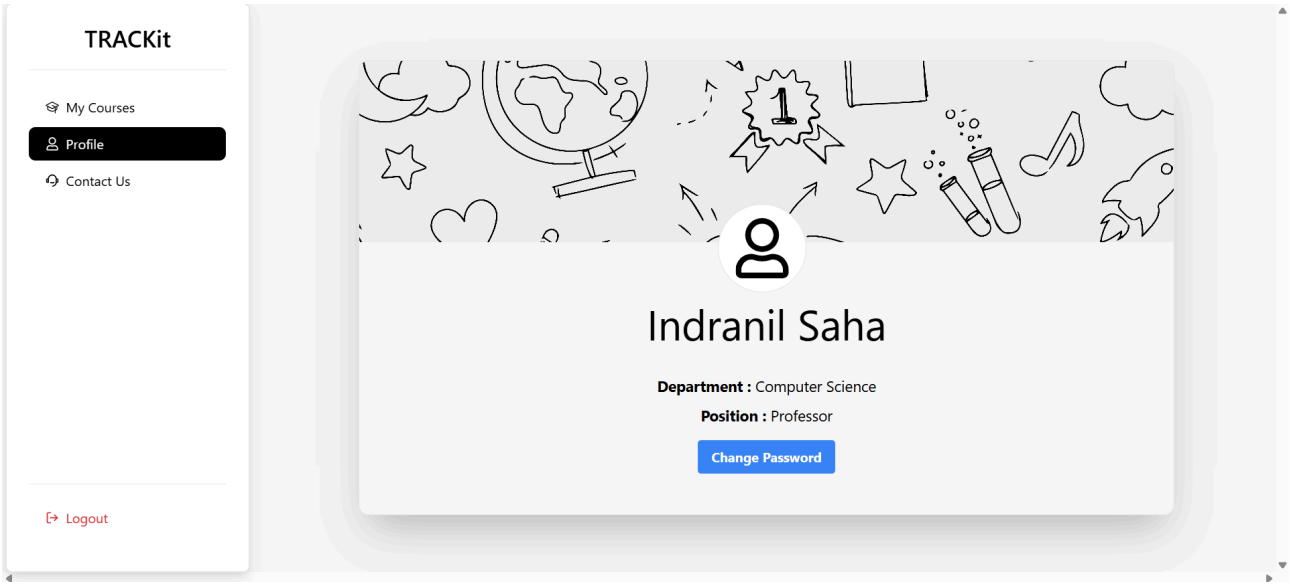




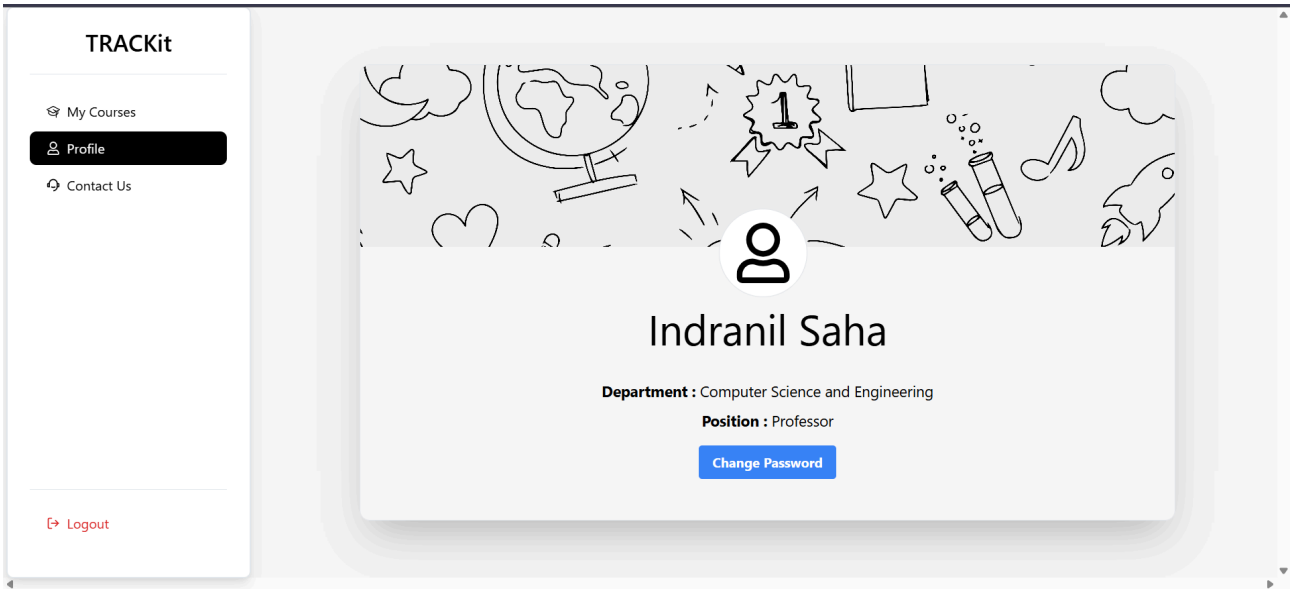
User details were updated from Manage User



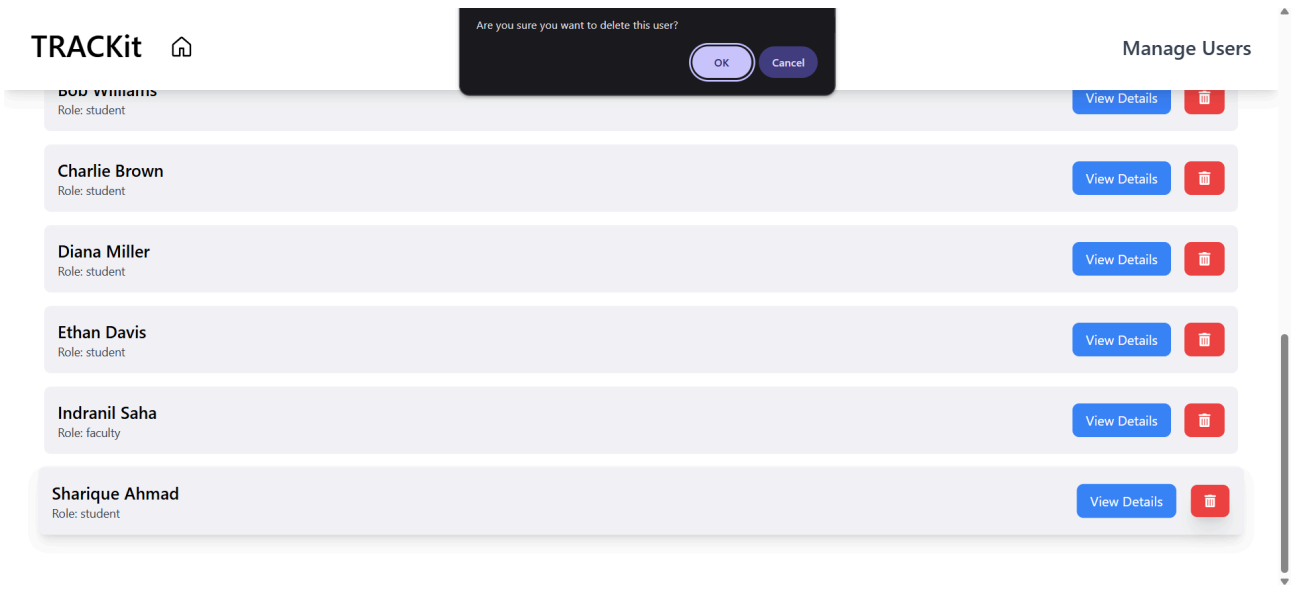
The department was Changed.



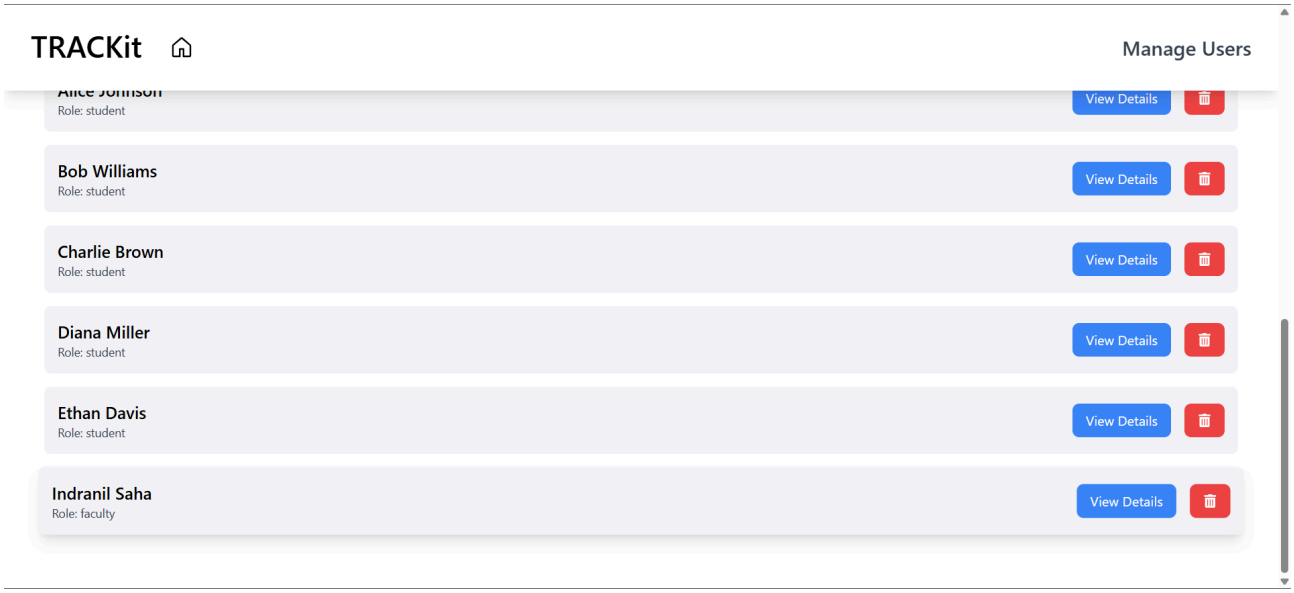
The change was successfully reflected in the user's Profile tab.



User named "Sharique Ahmad" is being deleted.



After deleting it is removed from the list and all its information is removed from the database



Admin - Manage Courses

Module Details:

This unit focuses on the "Manage Courses" functionality within the Admin panel. It allows the Admin to perform CRUD (Create, Read, Update, Delete) operations on courses, manage enrolled students and assigned faculty, and handle bulk student

additions via CSV uploads. The unit also includes search functionality for courses, students, and faculty, ensuring efficient course management.

Key Features:

1. Edit course details, including code, name, description, credits, and semester.
2. Delete courses from the system.
3. View course details, including enrolled students and assigned faculty.
4. Add or remove students and faculty from courses.
5. Bulk adds students to courses using a CSV file.

Test Owner:

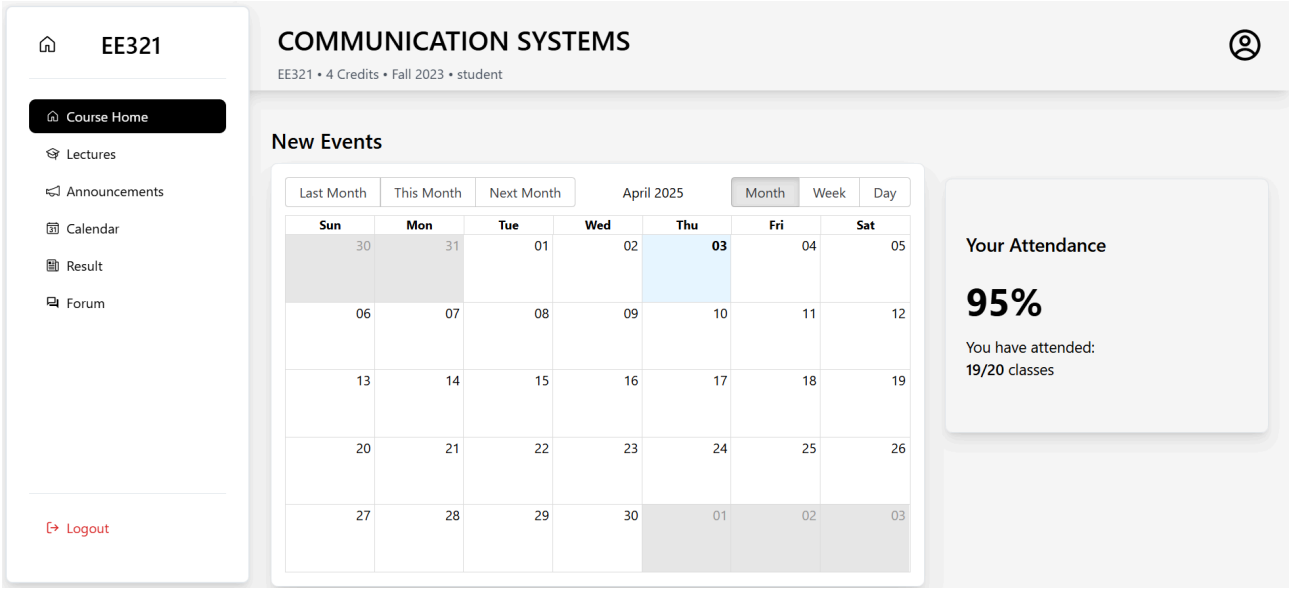
Sharique Ahmad: 221002

Test Date:

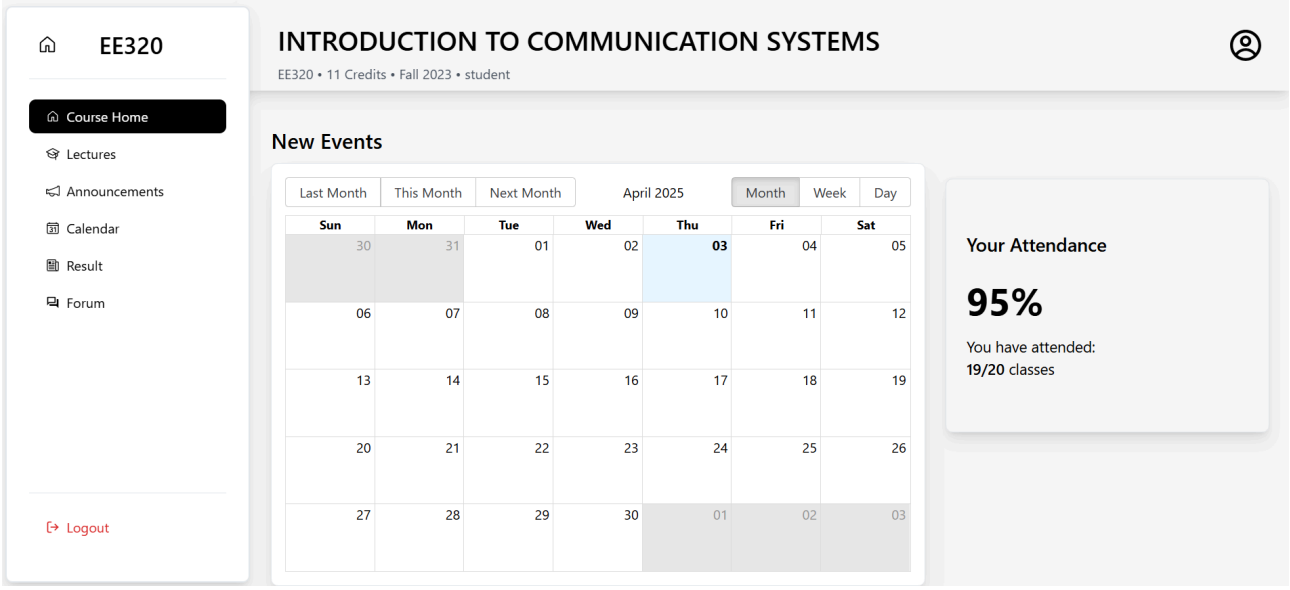
2 April 2025 - 3 April 2025

Test Results:

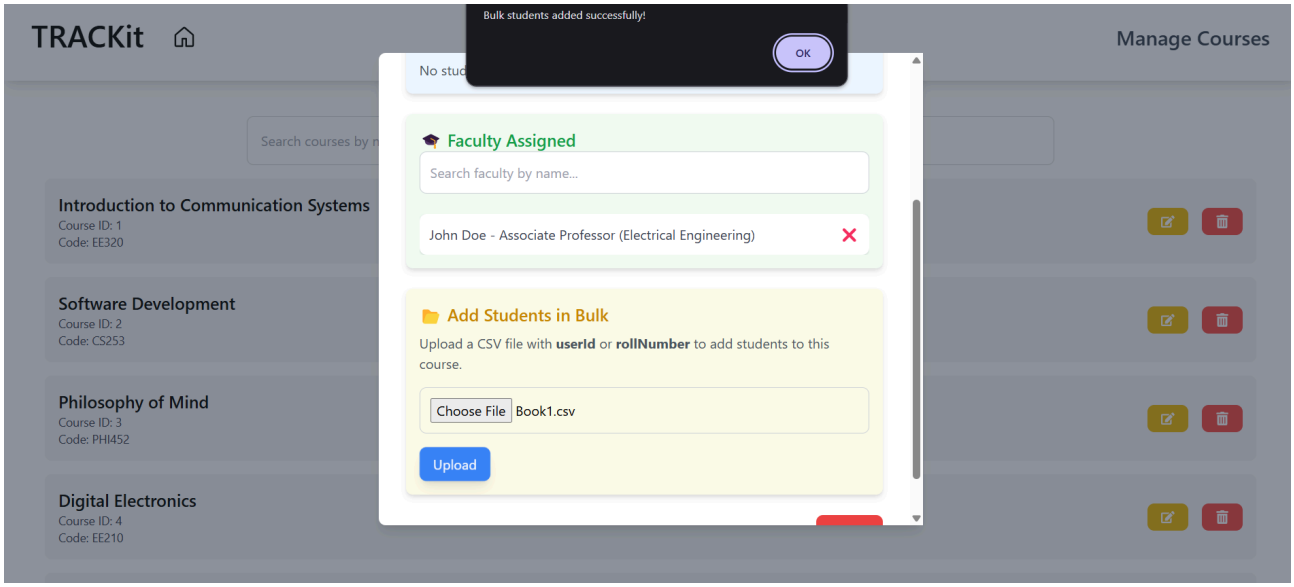
Before Updating the Course Details



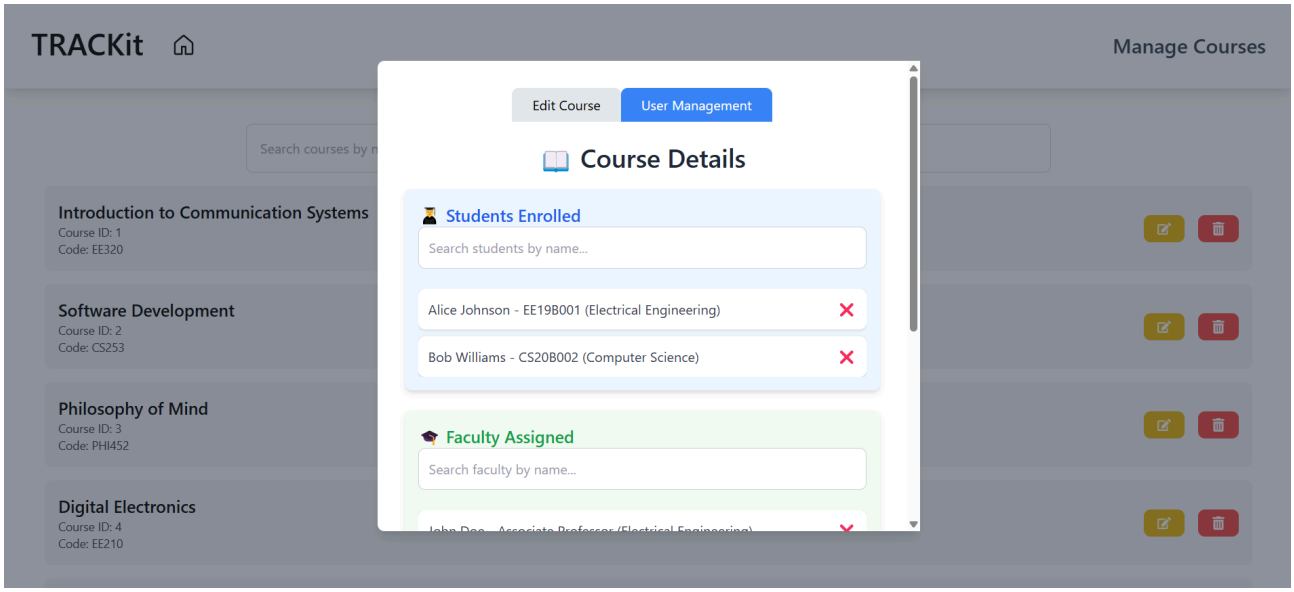
After updating the Course from the Manage Course all the changes were correctly reflected as the name and course code were changed.

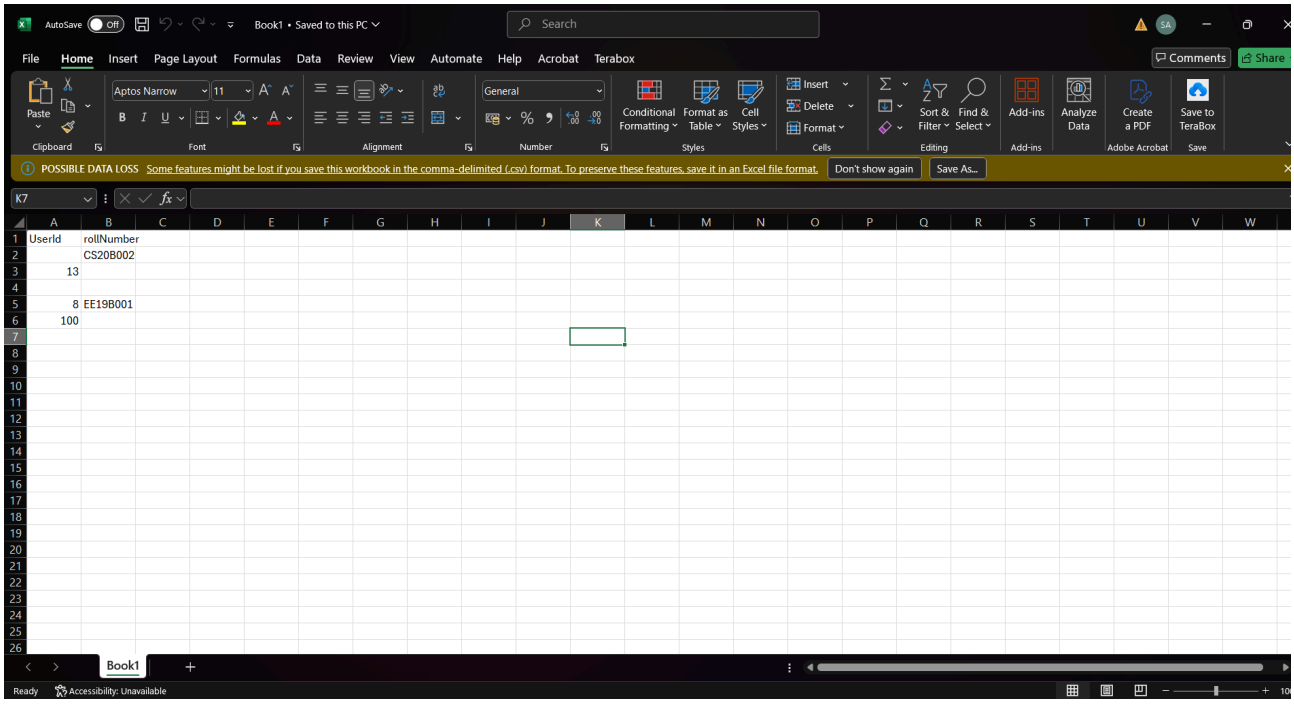


Bulk Addition of Students in a Course(EE320) is successful.

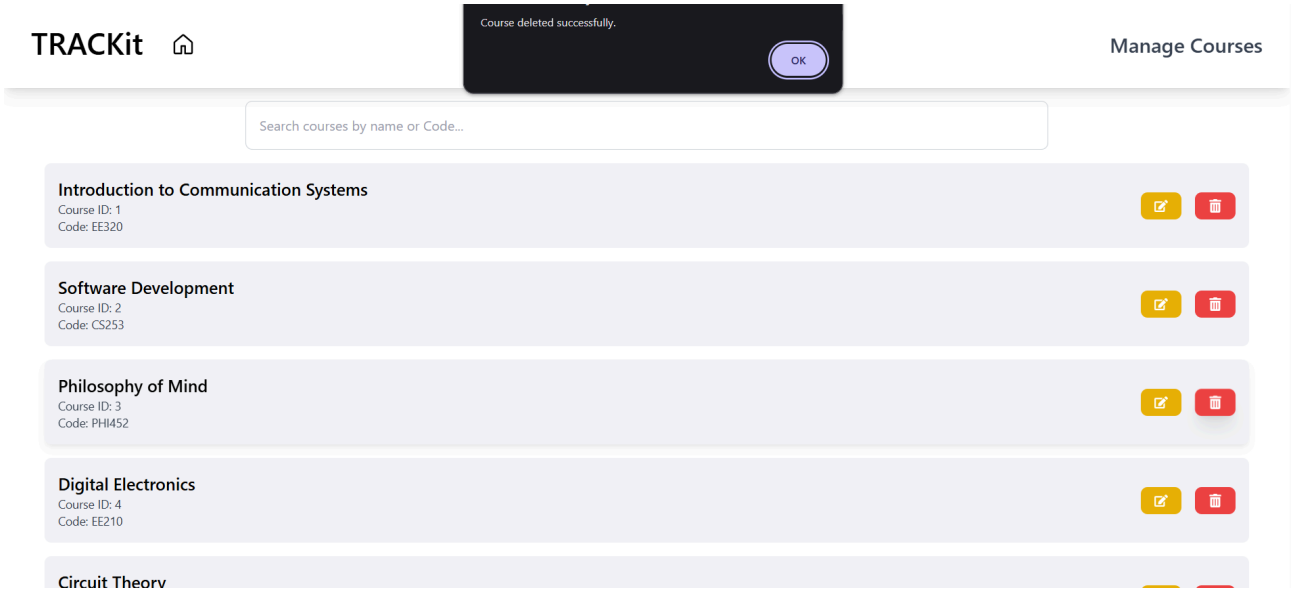


The User's which are already added are enrolled in the course. Rest are discarded.

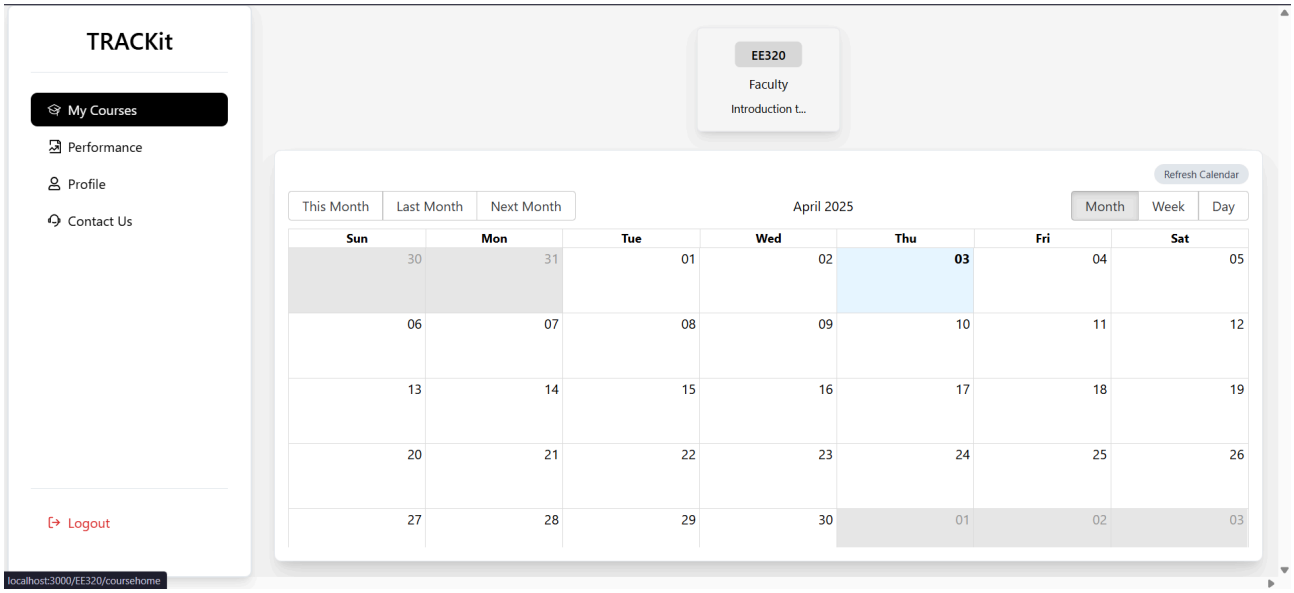




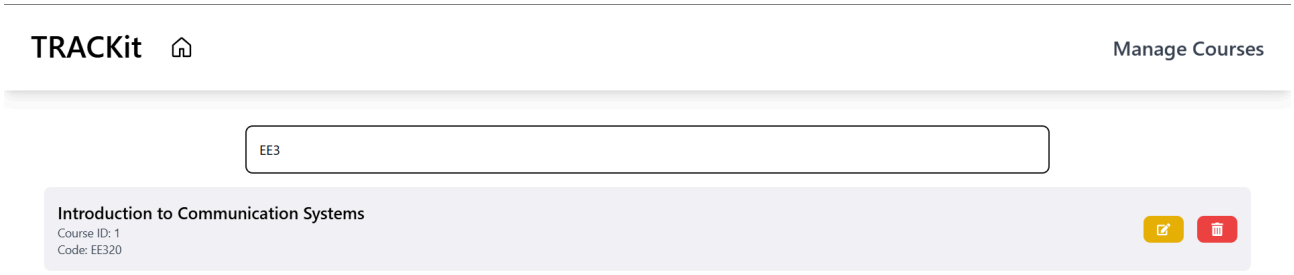
Course is successfully deleted from the Database.



The Dashboard of the Users enrolled in the Deleted Course was also updated correctly.



Search Functionality is also working correctly in Manage Courses.



Announcements

Module Details:

Involves the testing of the Announcements Database, unique to each course, where an announcement object, which has a heading along with a body, along with the corresponding frontend. Only Faculties associated with that course can create announcements in that course, but they can be viewed by all students and other Faculty members.

Test Owner:

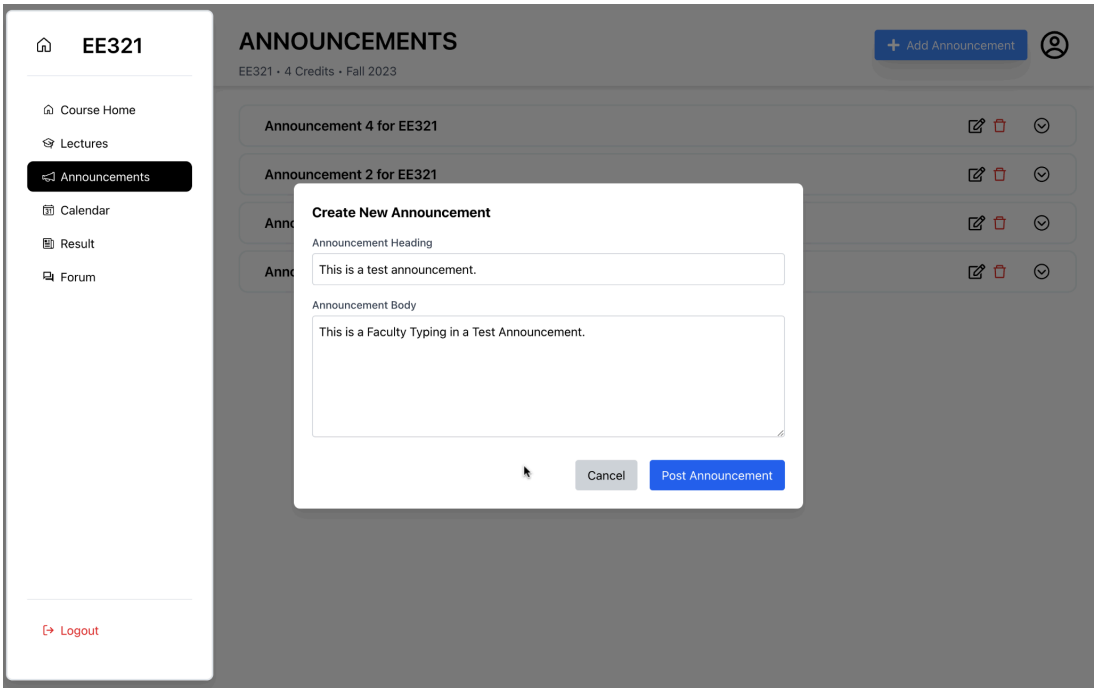
Aditya Gautam, 220064

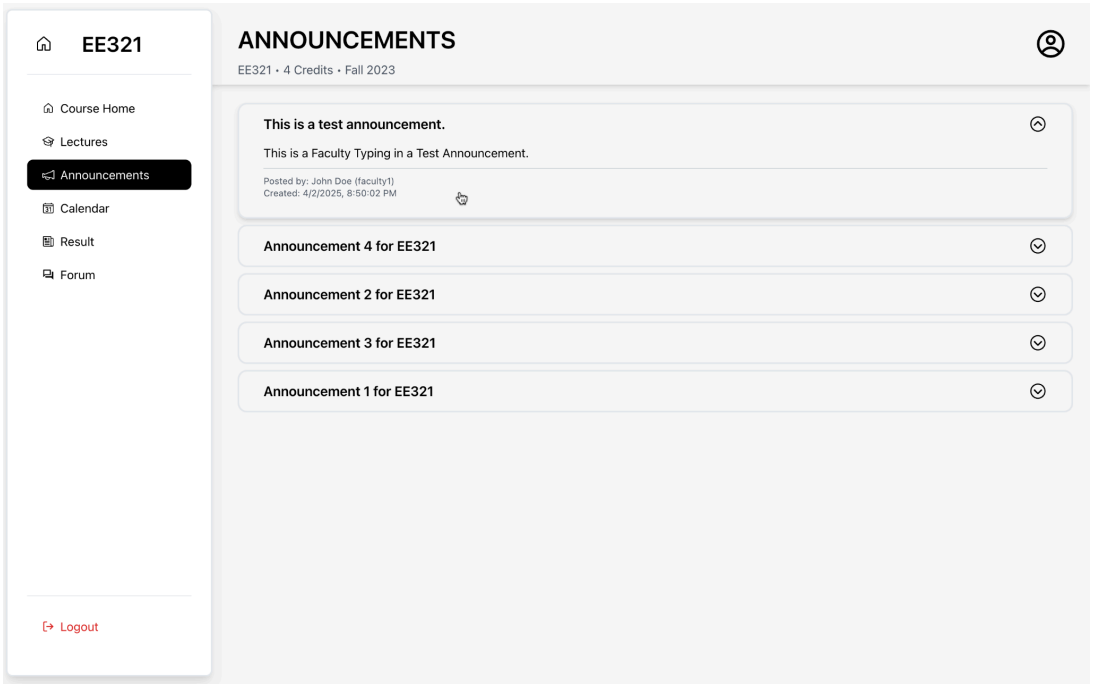
Test Date:

2 April 2025

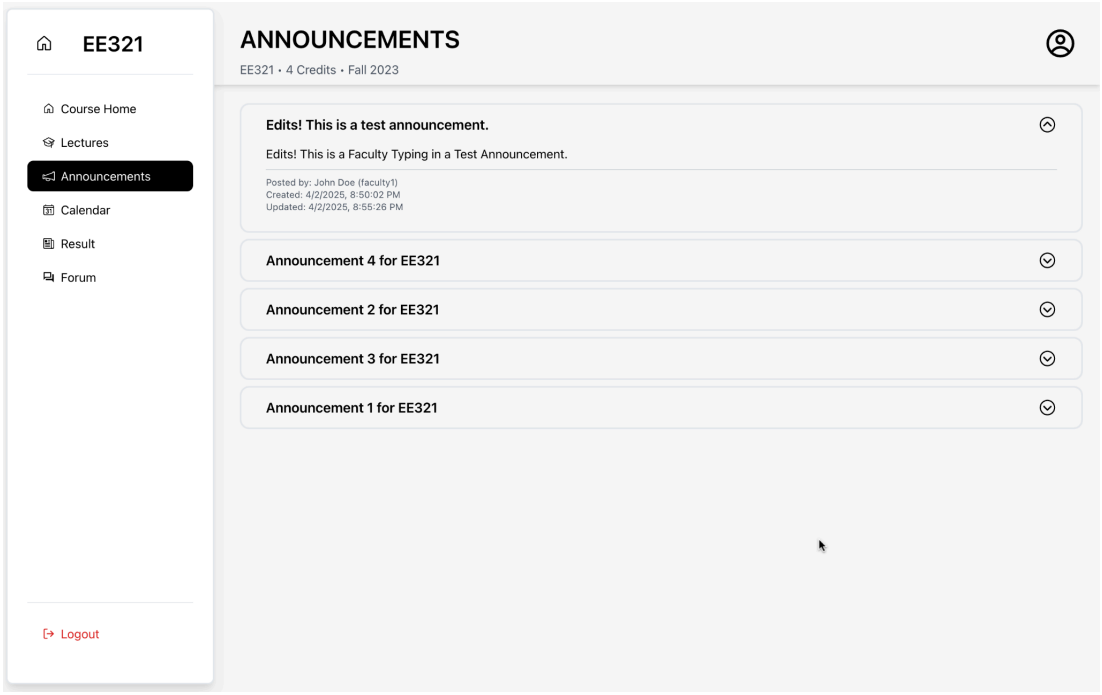
Test Results:

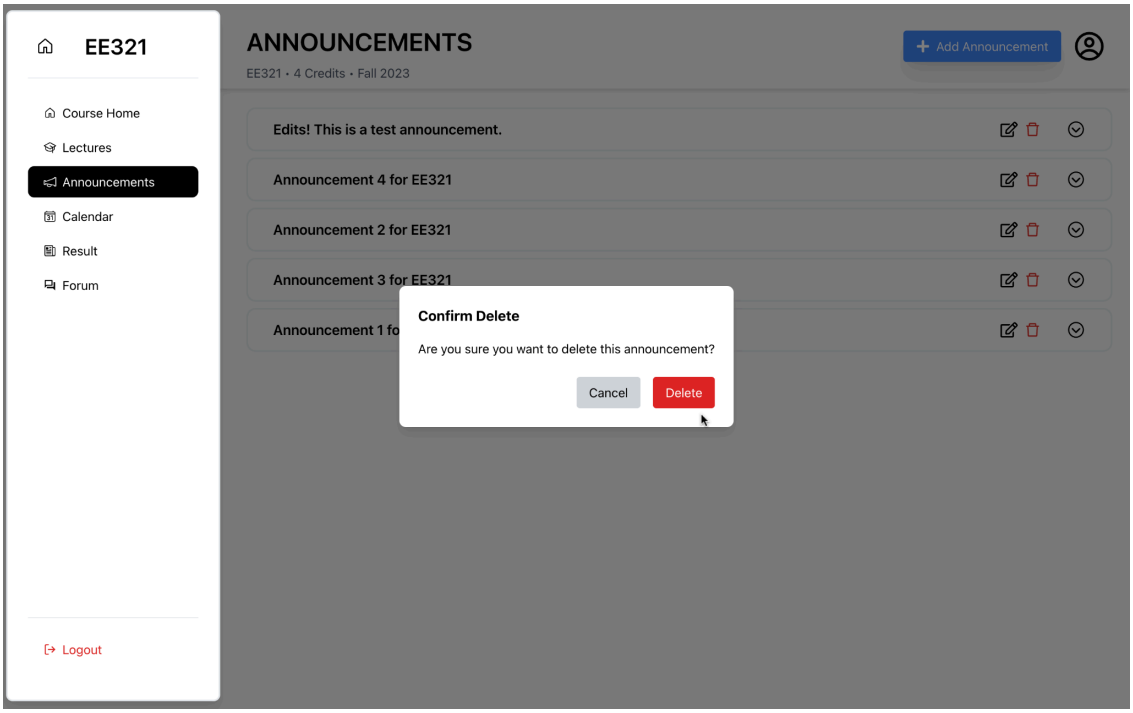
First, a dummy announcement was created using a faculty account. Then it was checked using the student accounts associated with that course.
Both Tests were fully successful.

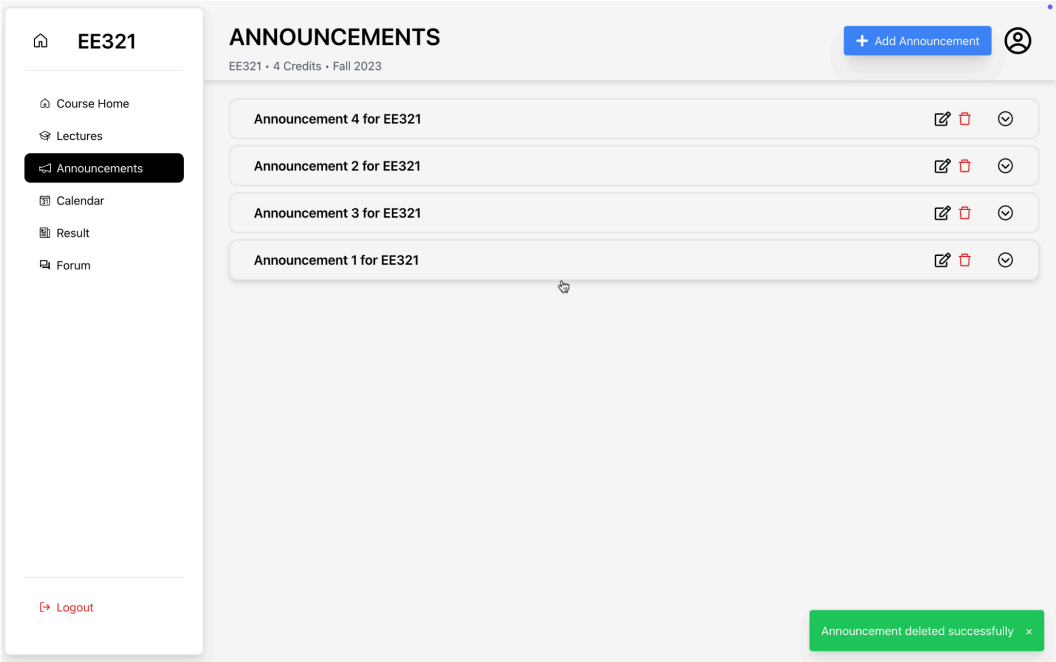




Following this, we tested the editing and deletion of the announcements by the faculty, and both were successful.


EE321
Course Home
Lectures
Announcements
Calendar
Result
Forum
Logout
ANNOUNCEMENTS
EE321 • 4 Credits • Fall 2023
Edits! This is a test announcement.
Edits! This is a Faculty Typing in a Test Announcement.
Posted by: John Doe (faculty1)
Created: 4/2/2025, 8:50:02 PM
Updated: 4/2/2025, 8:55:26 PM
Announcement 4 for EE321
Announcement 2 for EE321
Announcement 3 for EE321
Announcement 1 for EE321


EE321
Course Home
Lectures
Announcements
Calendar
Result
Forum
Logout
ANNOUNCEMENTS
EE321 • 4 Credits • Fall 2023
+ Add Announcement
Edits! This is a test announcement.
Announcement 4 for EE321
Announcement 2 for EE321
Announcement 3 for EE321
Announcement 1 fo
Confirm Delete
Are you sure you want to delete this announcement?
Cancel
Delete



Course Description Entry

Module Details:

Involves the testing of the Course Description Entry Database, unique to each course, where an announcement object, which has a heading along with a body, along with the corresponding frontend. Only faculties associated with that course can create course description entries, but they can be viewed by all students and other faculty members.

Test Owner:

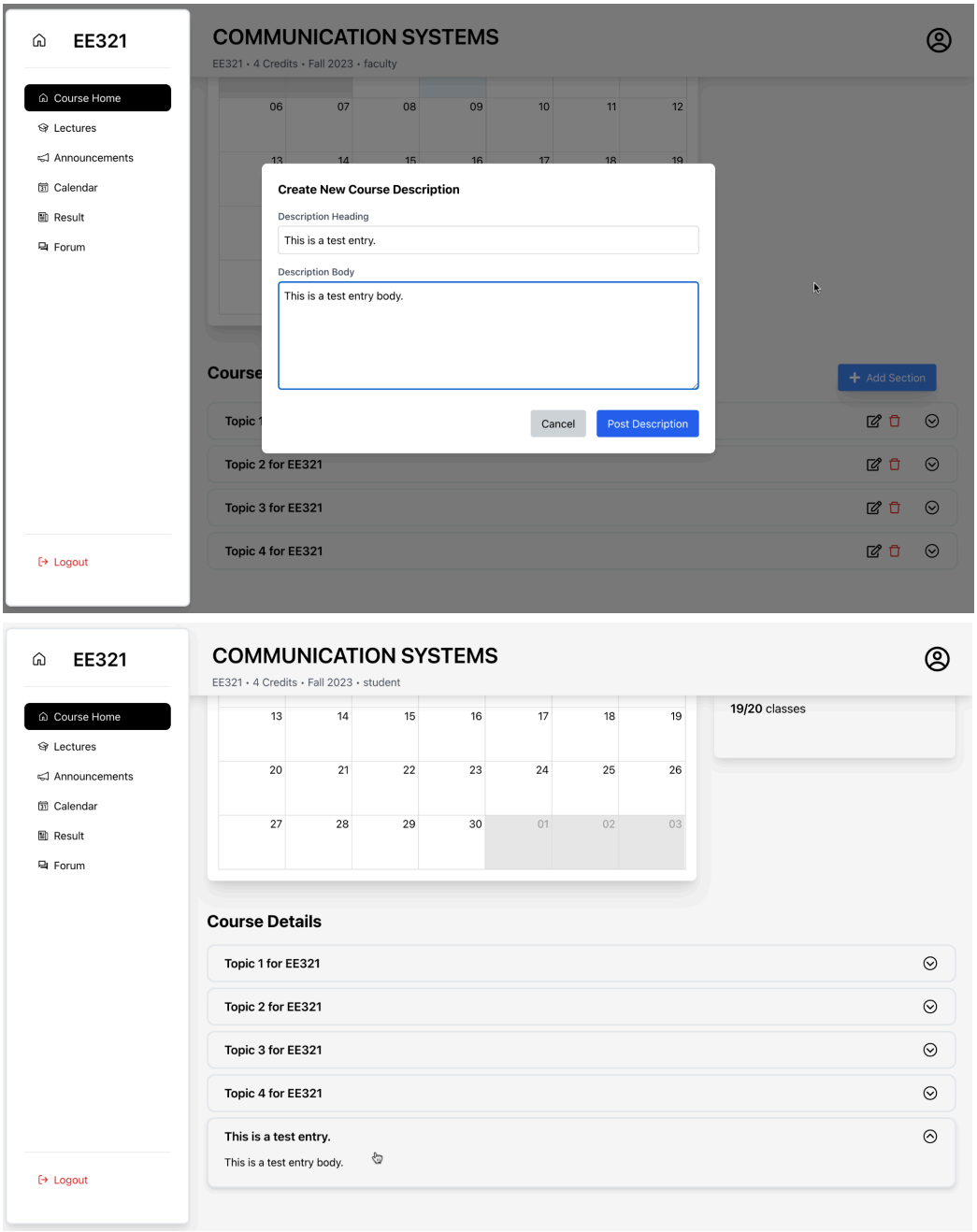
Aditya Gautam, 220064

Test Date:

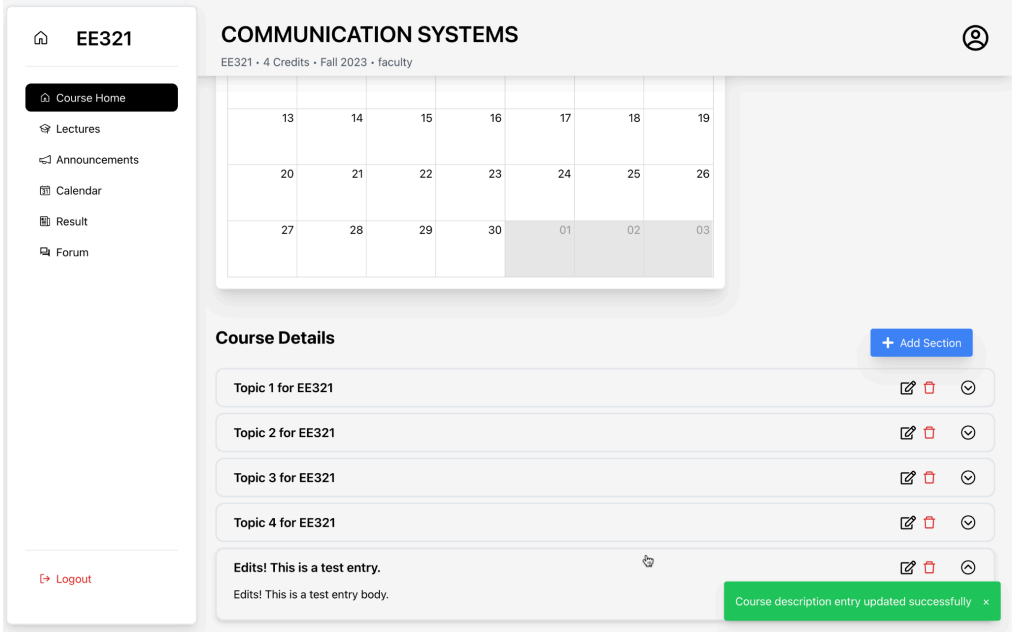
2 April 2025

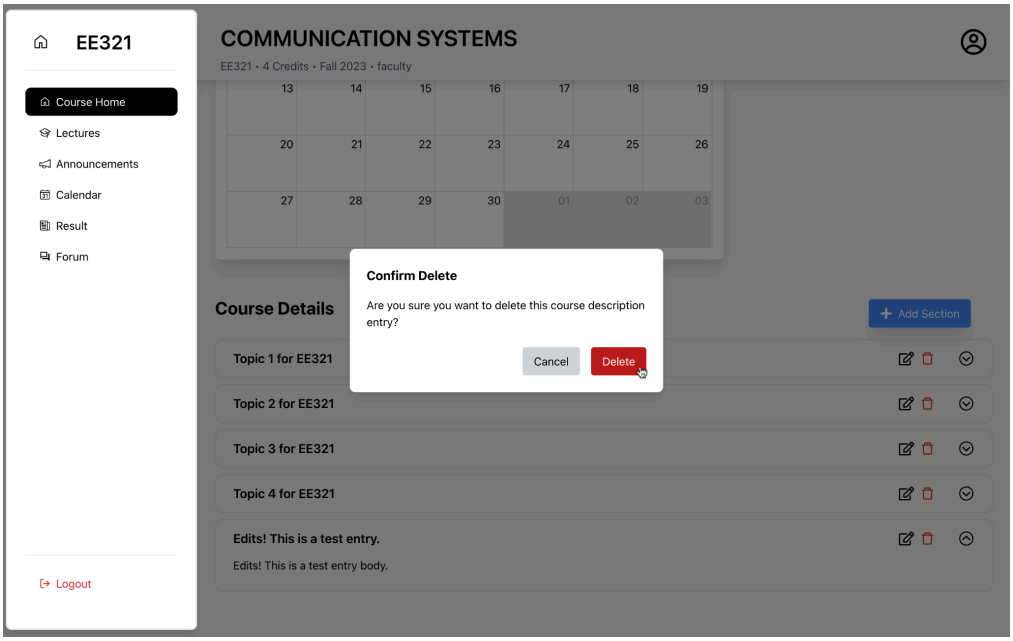
Test Results:

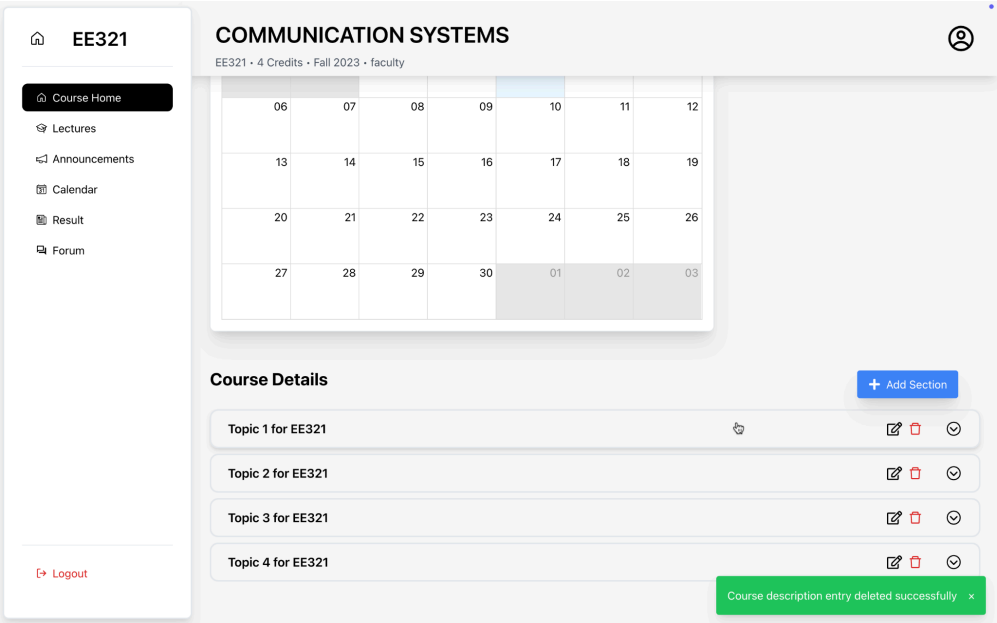
First, a dummy Course Description Entry was created using a faculty account. Then it was checked using the student accounts associated with that course.
Both Tests were fully successful.



Following this, we tested the editing and deletion of the Course Description Entry by the faculty, and both were successful.


EE321
Course Home
Lectures
Announcements
Calendar
Result
Forum
Logout
COMMUNICATION SYSTEMS
EE321 • 4 Credits • Fall 2023 • faculty
Course Details
Add Section
Topic 1 for EE321
Topic 2 for EE321
Topic 3 for EE321
Topic 4 for EE321
Edits! This is a test entry.
Edits! This is a test entry body.
Course description entry updated successfully


EE321
Course Home
Lectures
Announcements
Calendar
Result
Forum
Logout
COMMUNICATION SYSTEMS
EE321 • 4 Credits • Fall 2023 • faculty
Confirm Delete
Are you sure you want to delete this course description entry?
Cancel
Delete
Course Details
Add Section
Topic 1 for EE321
Topic 2 for EE321
Topic 3 for EE321
Topic 4 for EE321
Edits! This is a test entry.
Edits! This is a test entry body.



Events

Module Details:

Faculty can create Events in their course, which will be displayed to all students in that course. There are 3 places where the calendar is displayed, Main Dashboard (where events of all courses a user is enrolled in are shown, Course Home (a mini calendar where events of that course are shown) and Course Calendar (full-fledged calendar with course adding and removing support).

Test Owner:

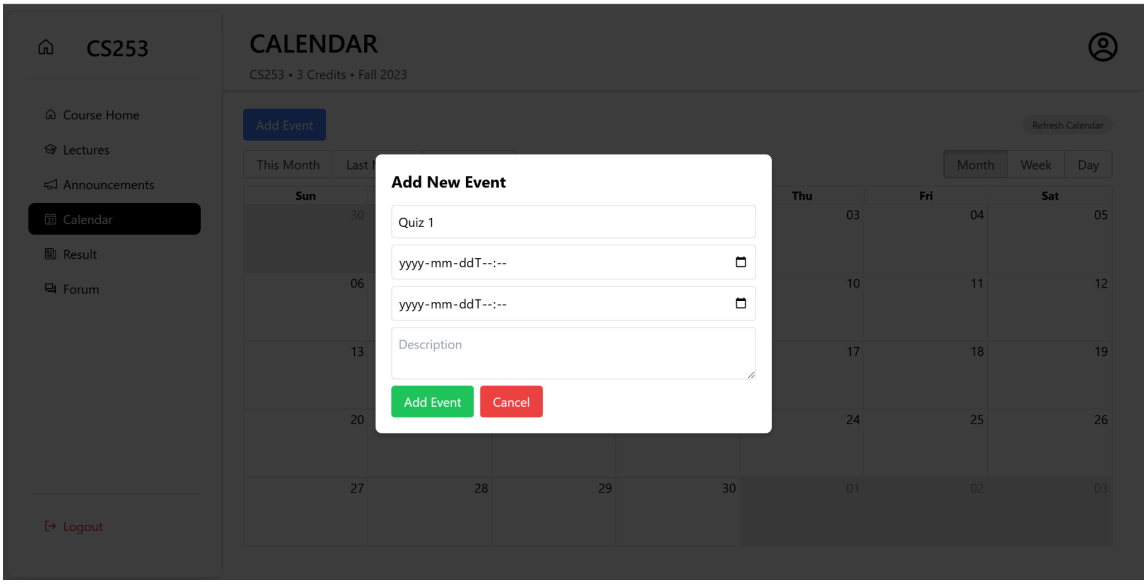
Aayush Singh,220024

Test Date:

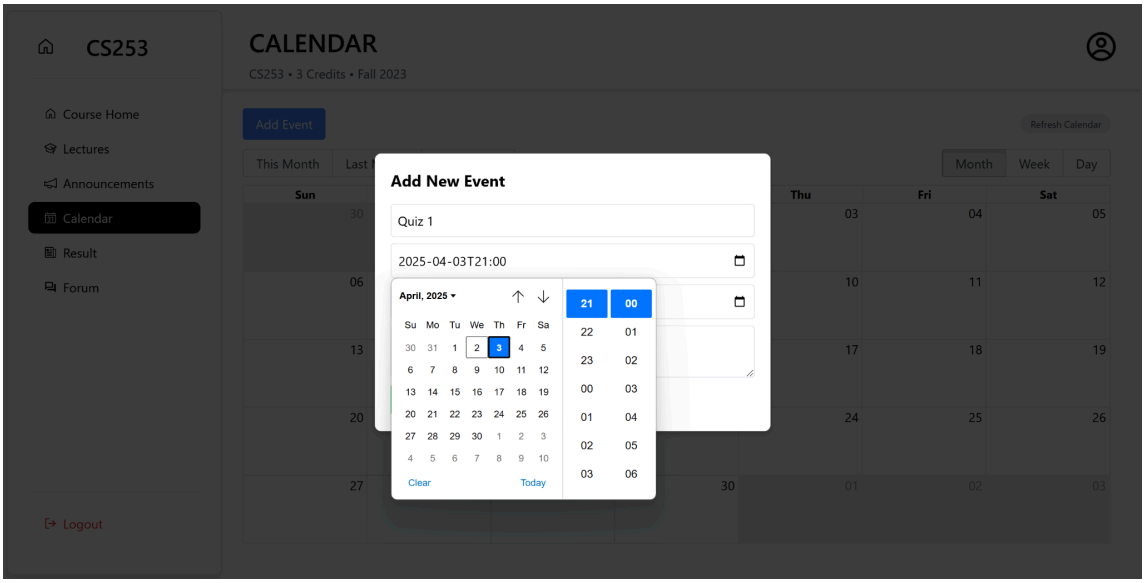
2 April 2025

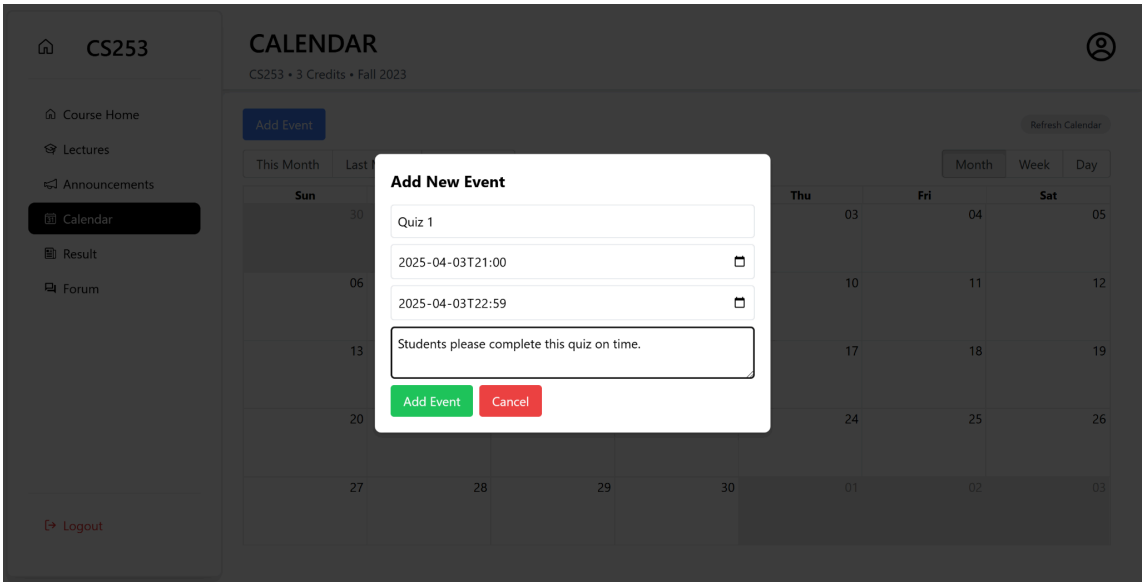
Test Results:

First, we added events in CS253 course from faculty1.We can put its Name, Start and End Time and date, and description

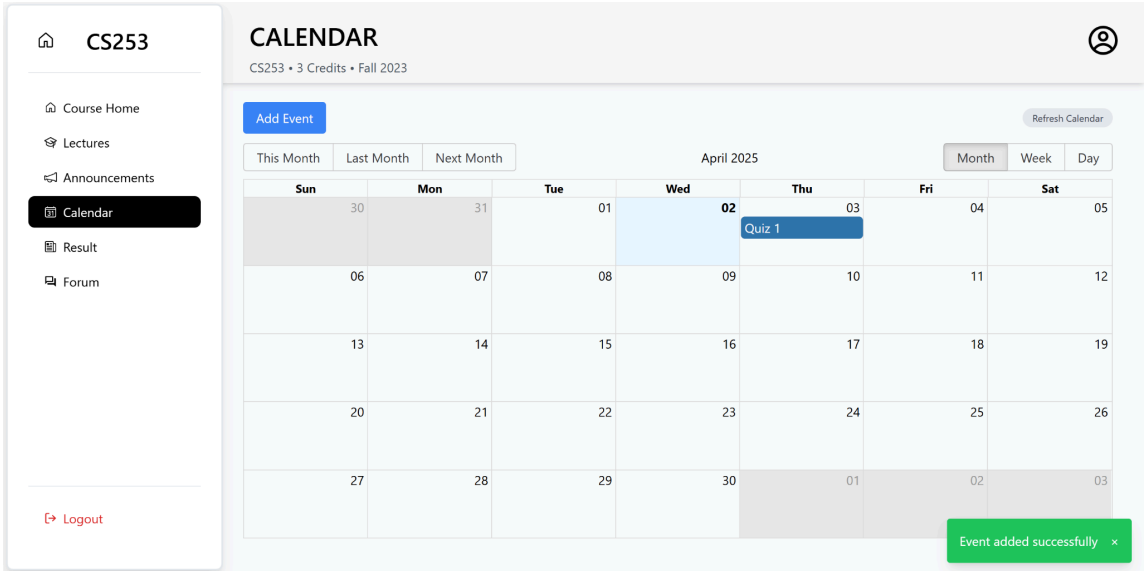


Time and Date selector UI.

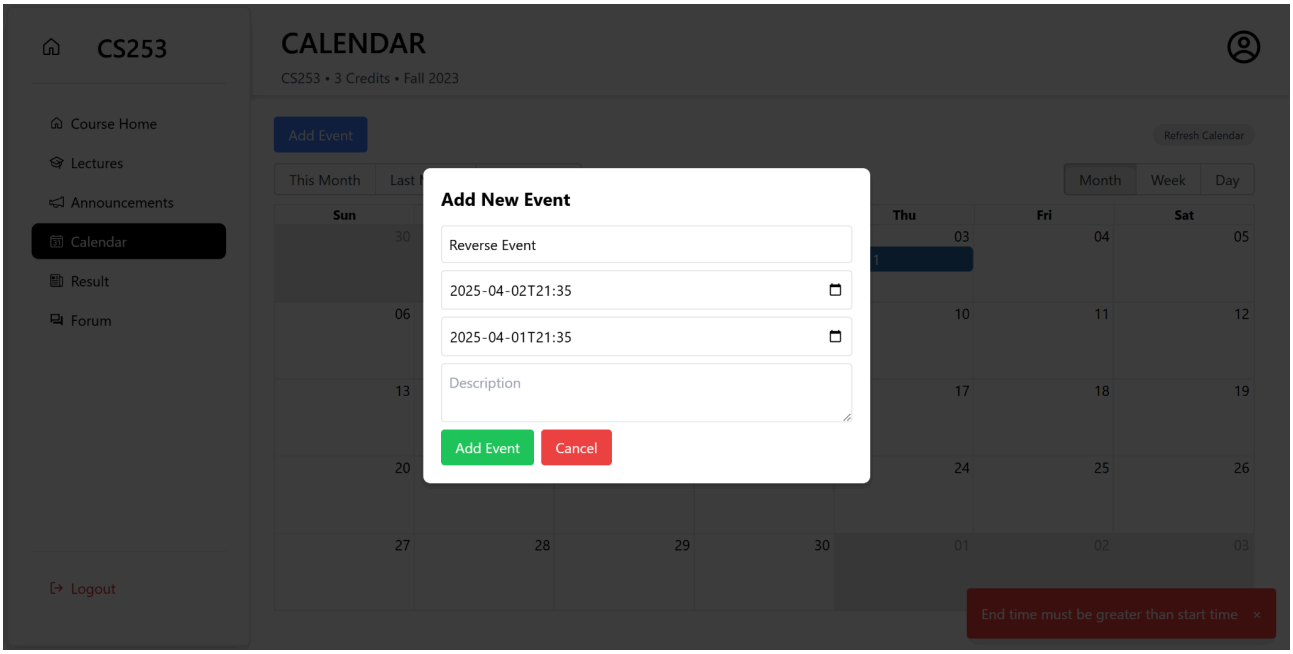




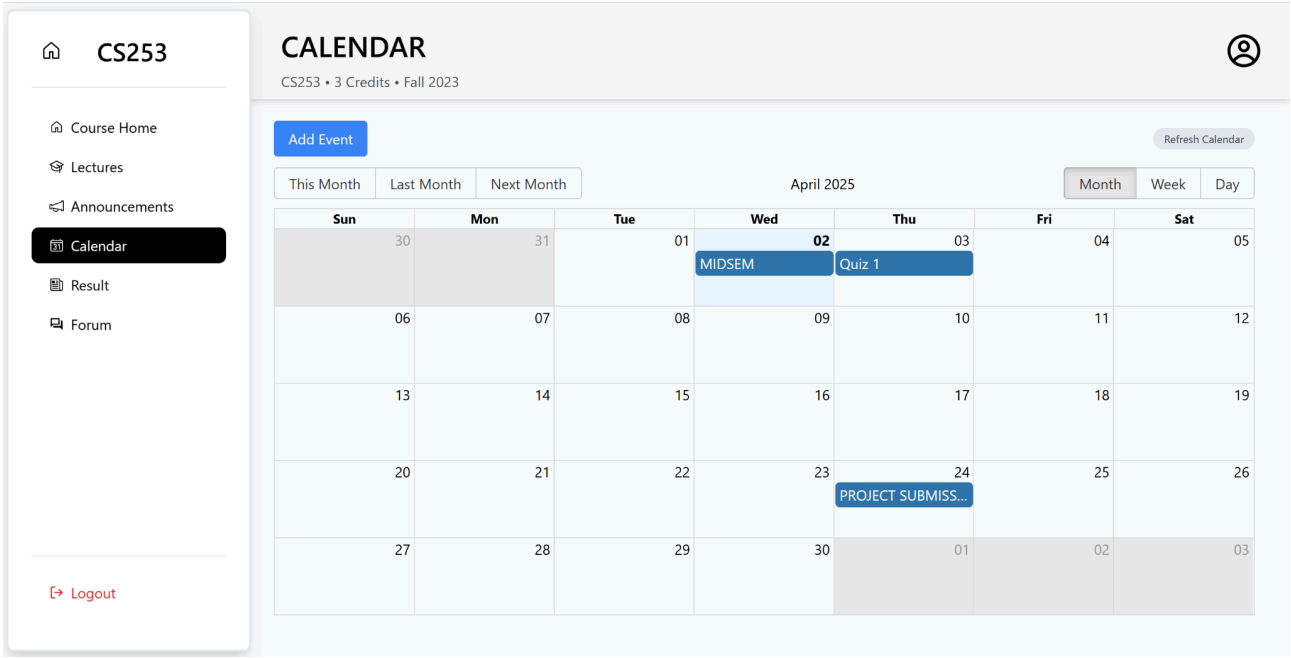
Event added successfully.



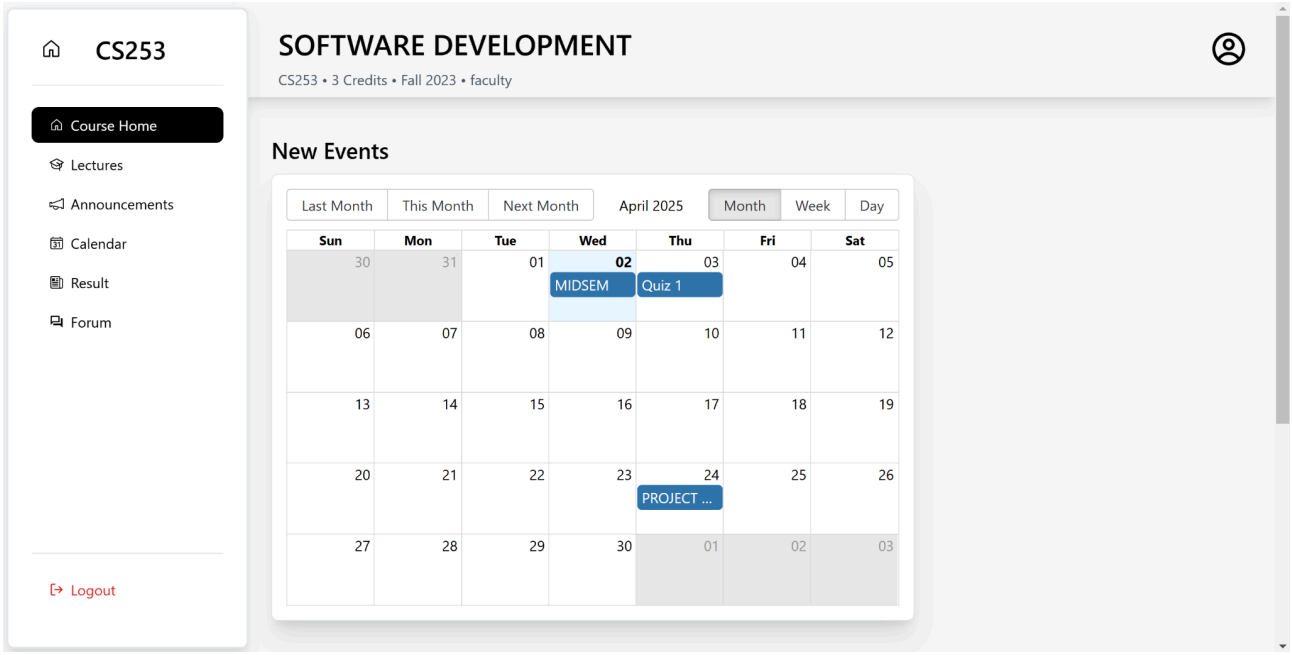
What if we try to put end time before start time?
It shows a pop up which says “End time must be greater than start time”.



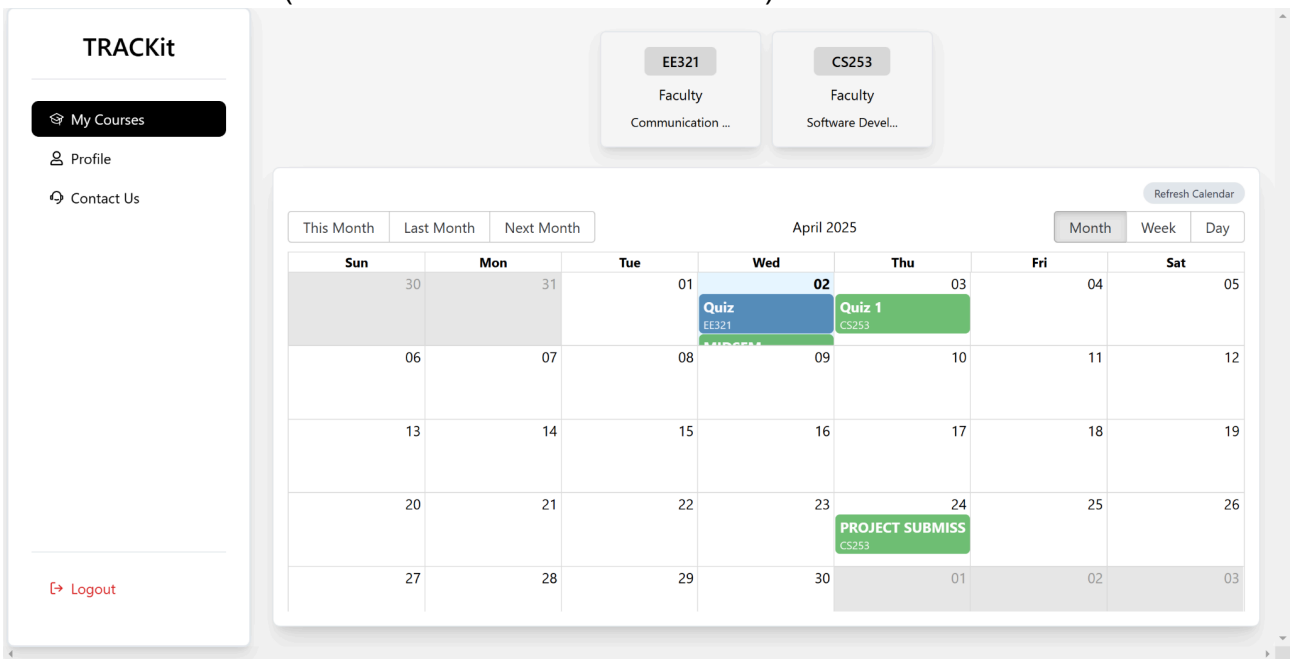
Adding some more events
In Course Calendar



In Course Home

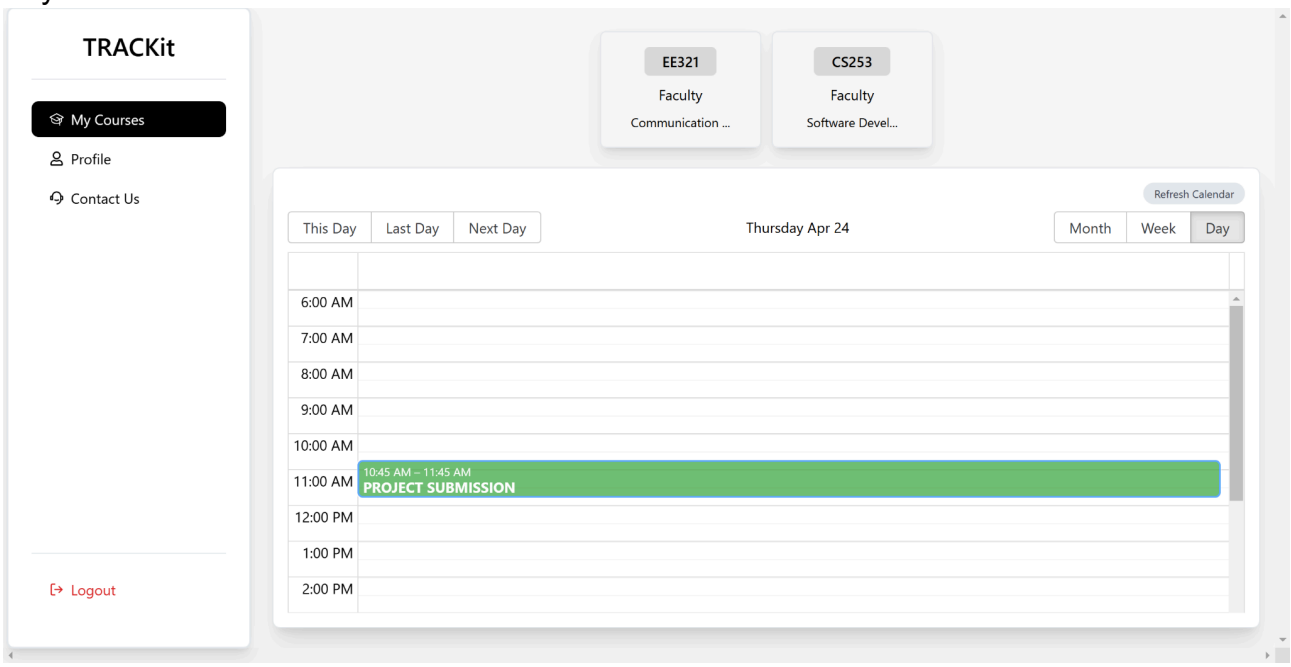


In Main DashBoard (Color Coded for different courses)

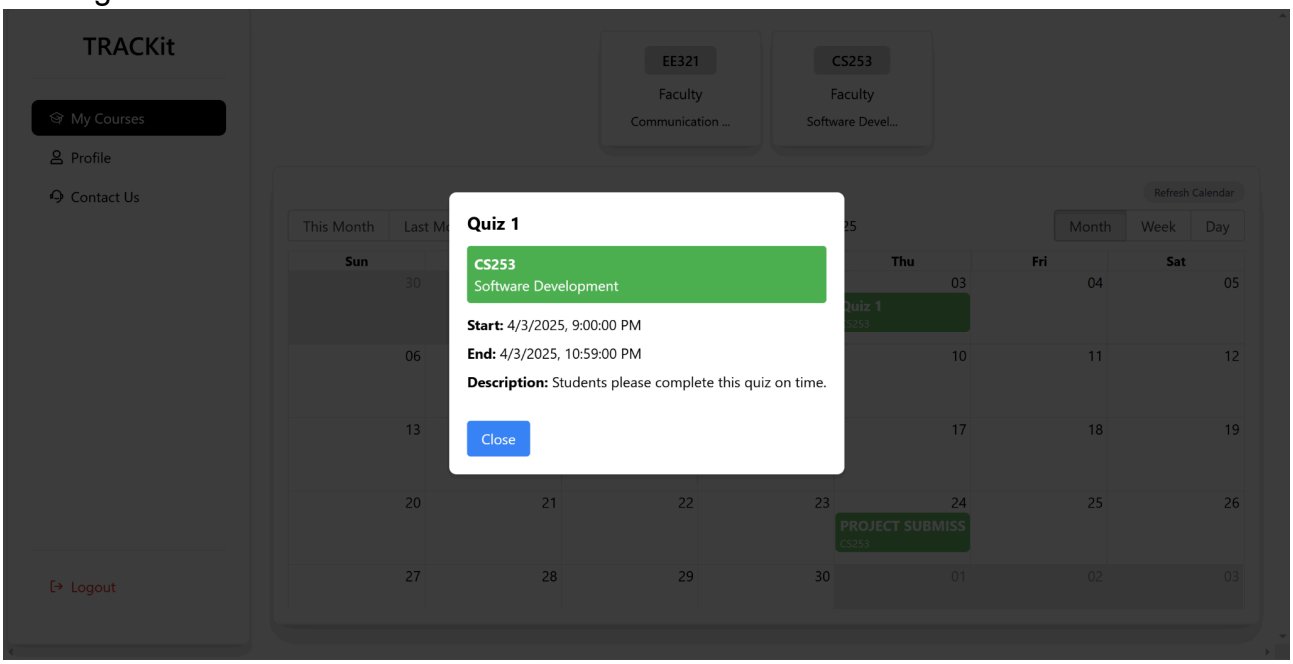


Weekly View

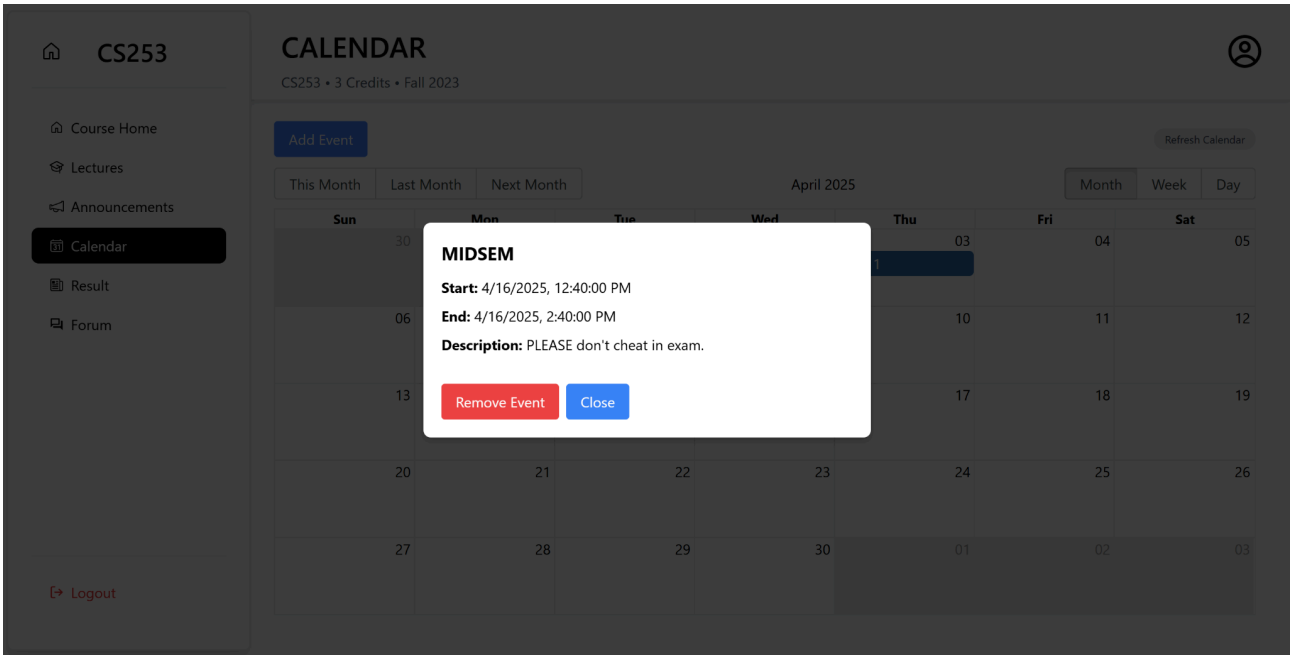
Day View



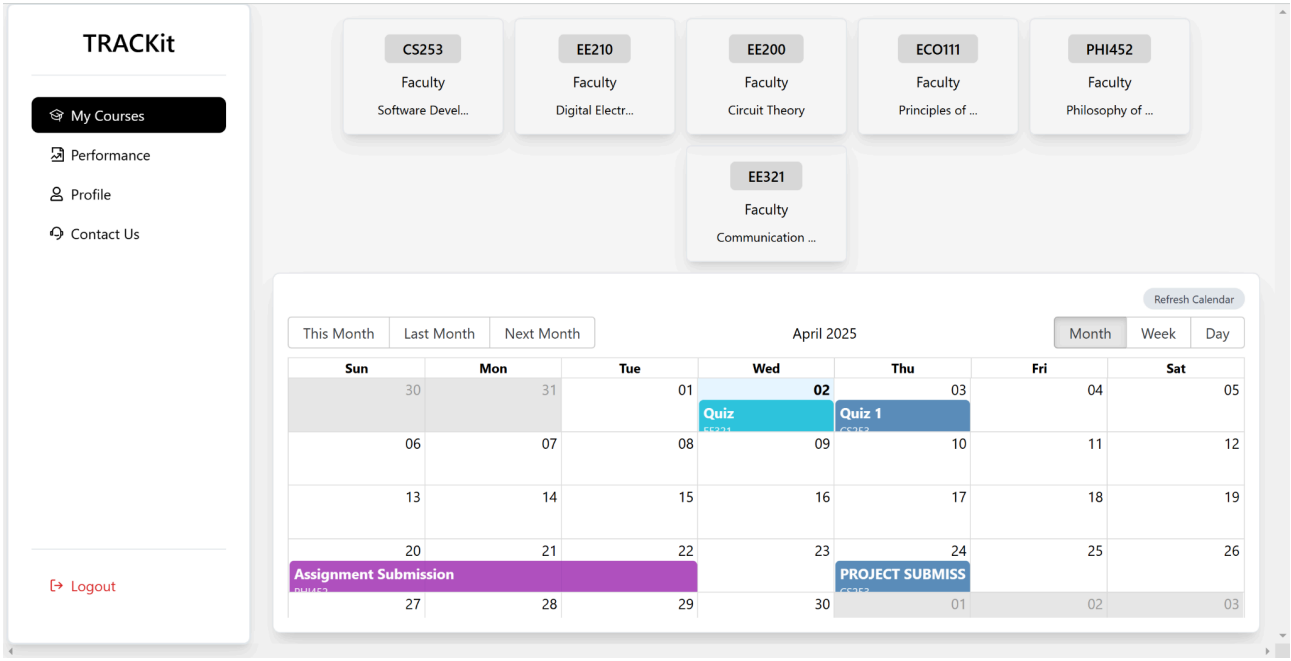
Clicking on an event



Removing an event (in course calendar page)



Testing whether the events are seen from a POV of student



Comments:

No critical bugs were found. But improvements can be done by adding repeating events.

Results

Module Details

The module encompasses testing procedures for the result page which involve creation, modification, deletion of students' results. It also tests how the views of both students and faculty get updated once changes are made by the faculty. We also check the download button for faculty which fetch data and convert it in excel format. Additionally automatic calculation of mean, median, standard deviation is also checked.

Test owner

Dhruv Rai (220365)

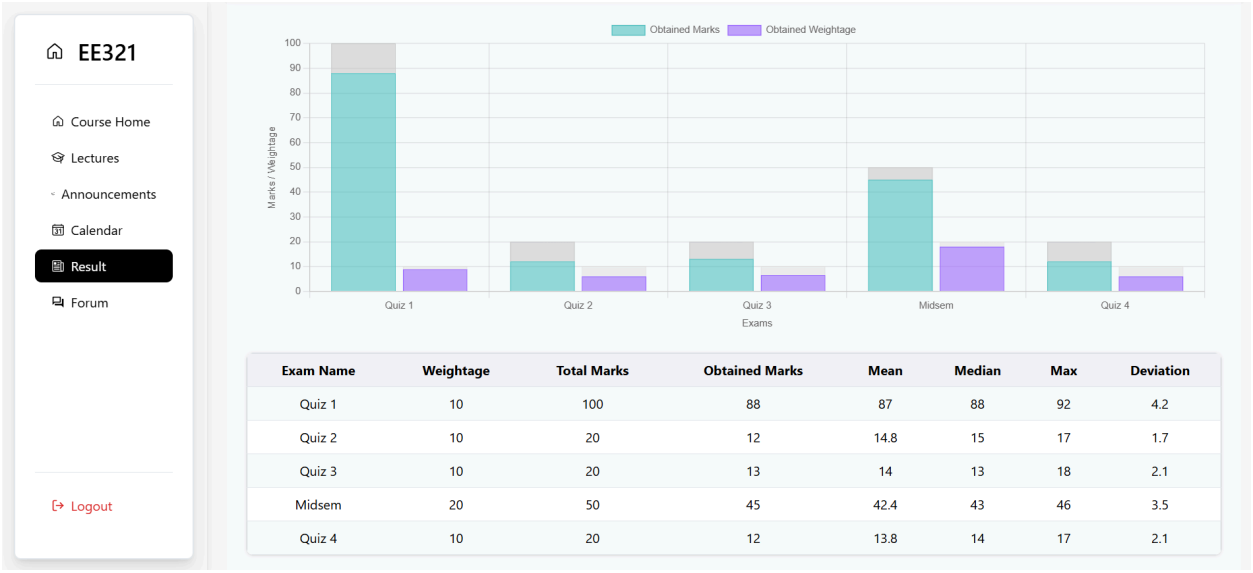
Test date

02/04/25 - 05/04/25

Test results

In this test we validated the successful integration of backend and frontend components of getAllResults, getAllExamsWithDetails, PublishExamResults, ModifyExamResults, getResultsForstudent, DeleteExamAndAssociatedResults and getEnrolledStudentsList. Apart from this, we successfully tested that the result table and the graph in both the student view and faculty view page represents the correct results for all students after all kinds of modifications. Throughout the testing process, we also scrutinized the redirection of users to the correct pages upon the completion of these actions.

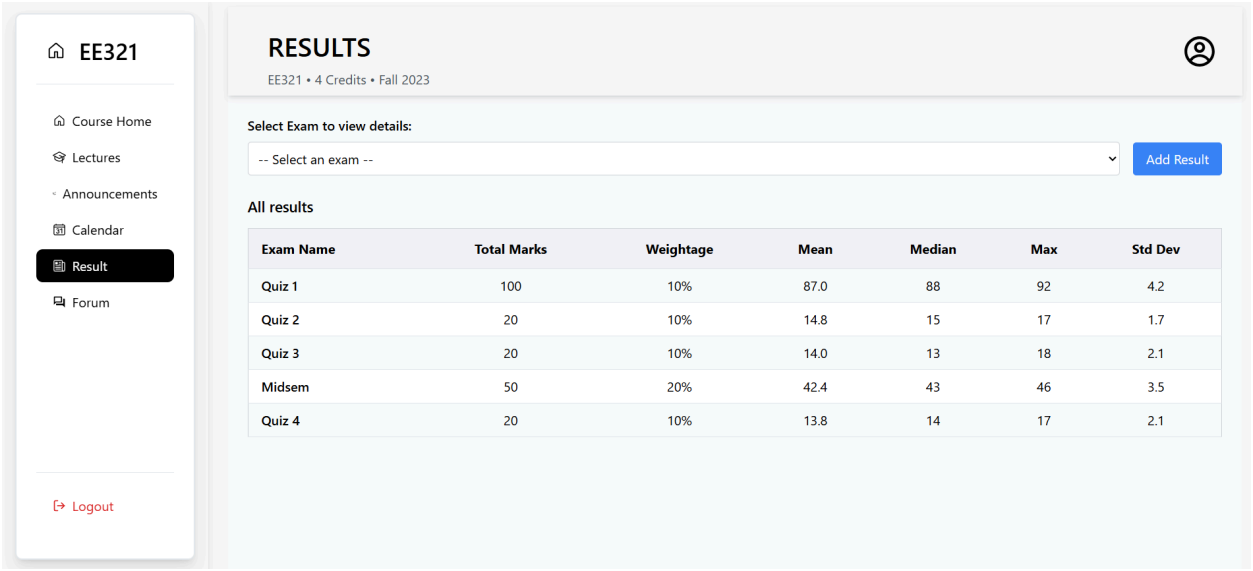
Details : To verify whether all the results associated with a particular student enrolled in a course get fetched correctly or not in the student view.



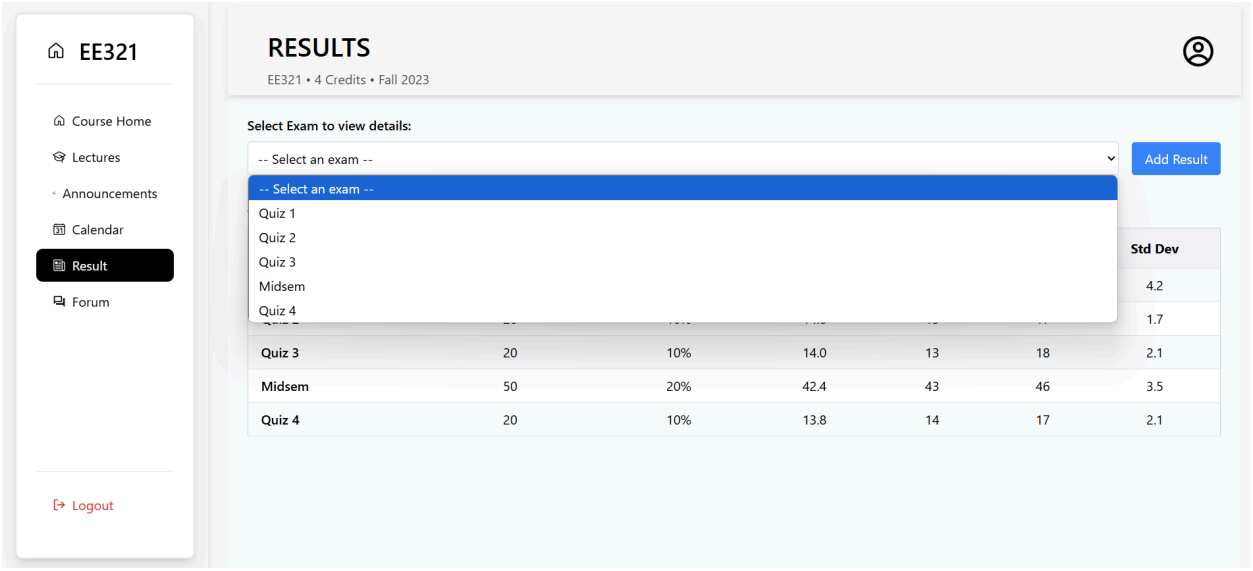
Exam Name	Weightage	Total Marks	Obtained Marks	Mean	Median	Max	Deviation
Quiz 1	10	100	88	87	88	92	4.2
Quiz 2	10	20	12	14.8	15	17	1.7
Quiz 3	10	20	13	14	13	18	2.1
Midsem	20	50	45	42.4	43	46	3.5
Quiz 4	10	20	12	13.8	14	17	2.1

Result : Displaying all the results with correct details for the student logged in and selected a particular course.

Details : To verify whether all the exam names for which result has been published are correctly shown or not in the faculty view after clicking on the result tab from the sidebar and also check the working of the dropdown menu.

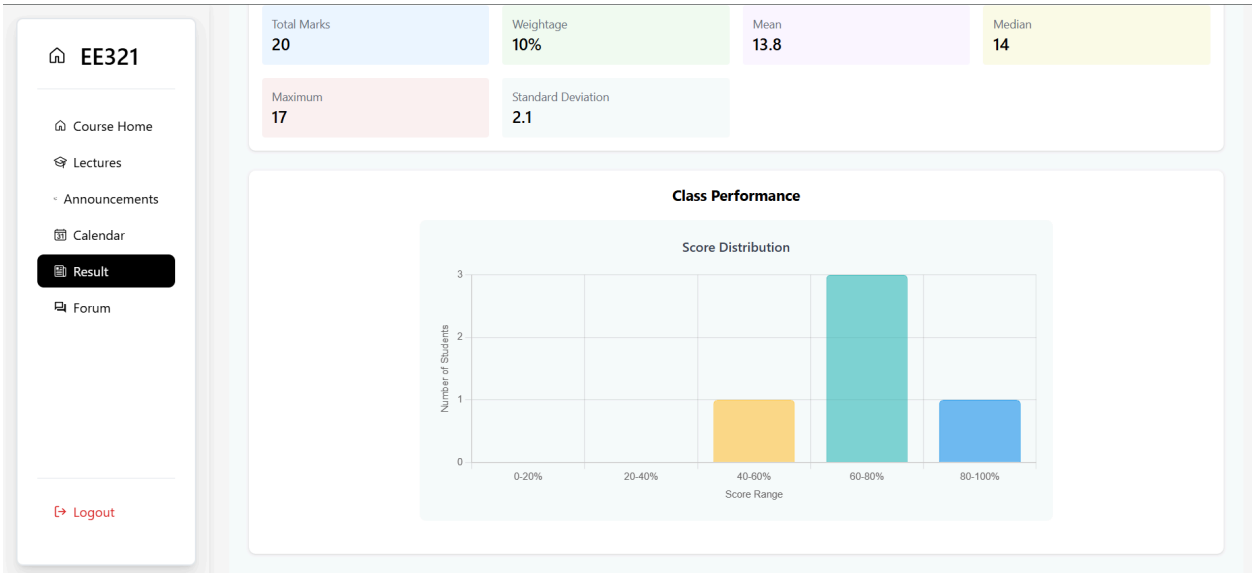


Exam Name	Total Marks	Weightage	Mean	Median	Max	Std Dev
Quiz 1	100	10%	87.0	88	92	4.2
Quiz 2	20	10%	14.8	15	17	1.7
Quiz 3	20	10%	14.0	13	18	2.1
Midsem	50	20%	42.4	43	46	3.5
Quiz 4	20	10%	13.8	14	17	2.1



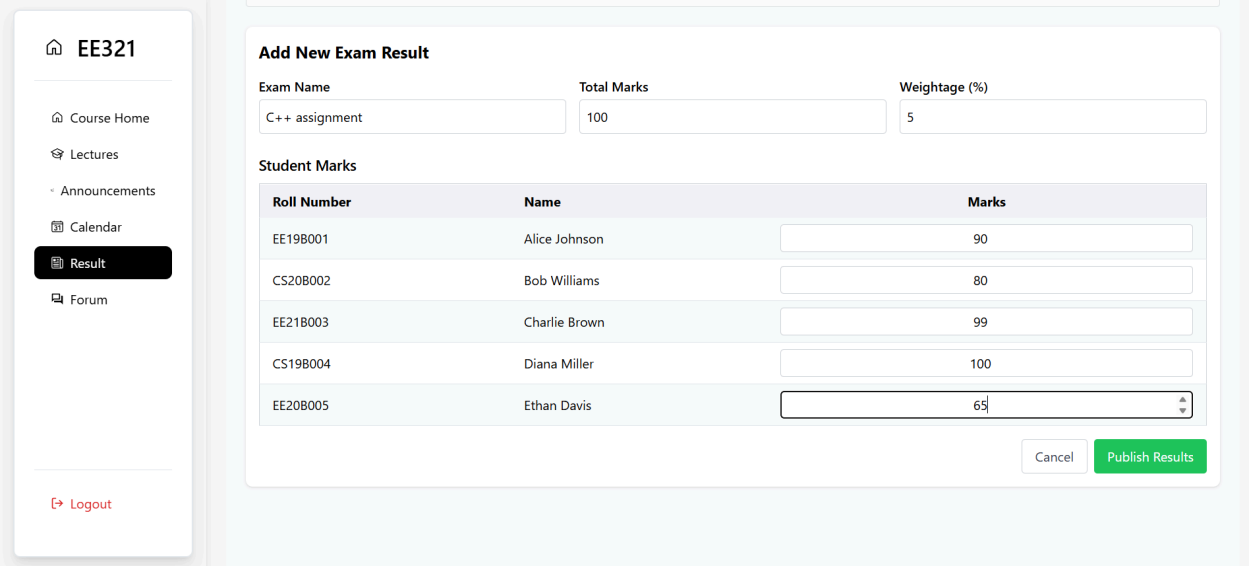
Result : Displaying all the exams with correct details once faculty get logged in and select a particular course and then click on the result tab. The dropdown menu is also working fine with a dynamic list of exams.

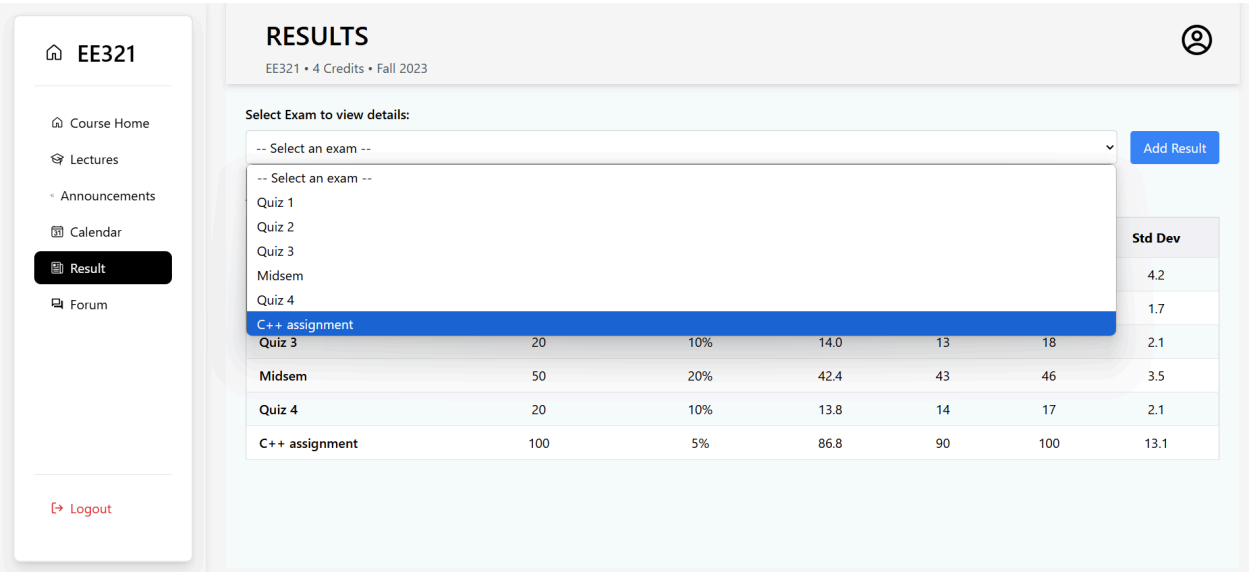
Details : To verify whether all the automatic calculations of mean, median, maximum, standard deviation are correct or not. Additionally also check the class performance graph.



Result : All the calculations are done correctly for the new data and the graph is also correct.

Details : To verify whether the add result option is working fine and does this get updated in real time for both student and faculty of the course. Additionally check if all the edge case checks are working fine or not.

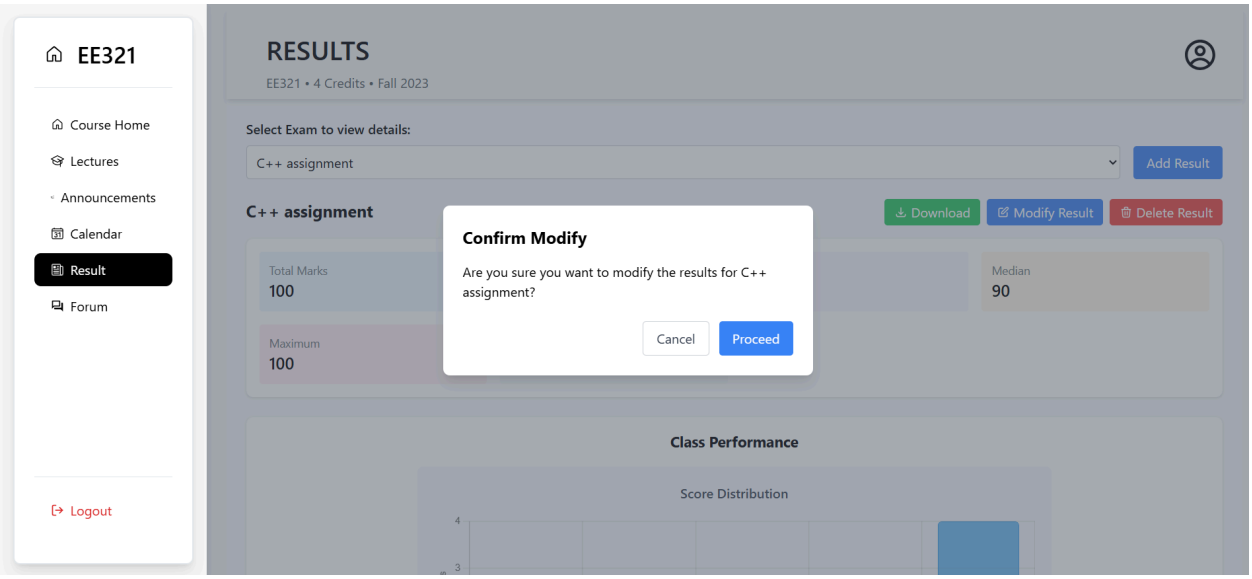


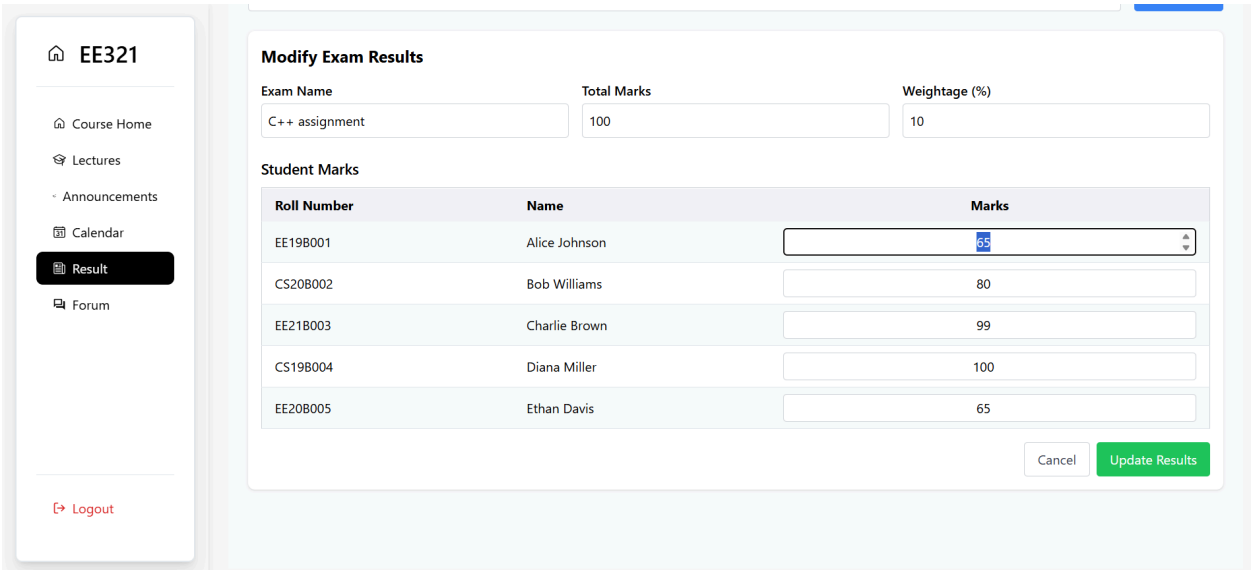


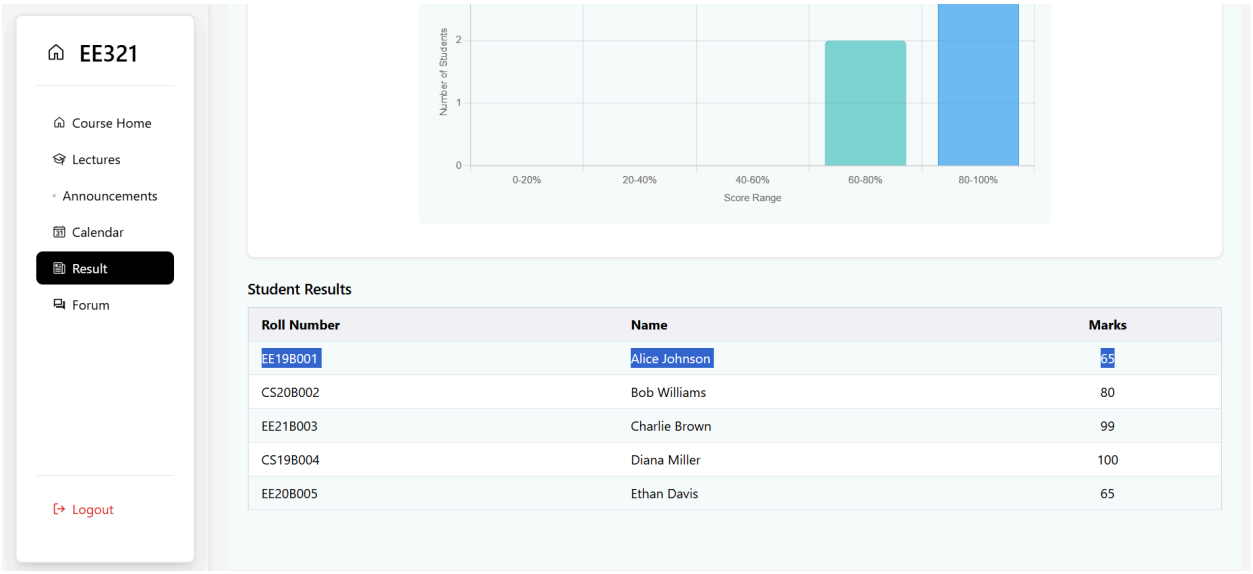
Exam Name	Weightage	Total Marks	Obtained Marks	Mean	Median	Max	Deviation
Quiz 1	10	100	88	87	88	92	4.2
Quiz 2	10	20	12	14.8	15	17	1.7
Quiz 3	10	20	13	14	13	18	2.1
Midsem	20	50	45	42.4	43	46	3.5
Quiz 4	10	20	12	13.8	14	17	2.1
C++ assignment	5	100	90	86.8	90	100	13.1

Result : The add result option worked fine and correctly updated the database. Once the exam result is added it correctly refreshes the screen for the faculty loading the details of the added exam and also loads the same for all the enrolled students.

Details : To verify whether the modify result option is working fine and does this get updated in real time for both student and faculty of the course. Additionally check if all the edge case checks are working fine or not.

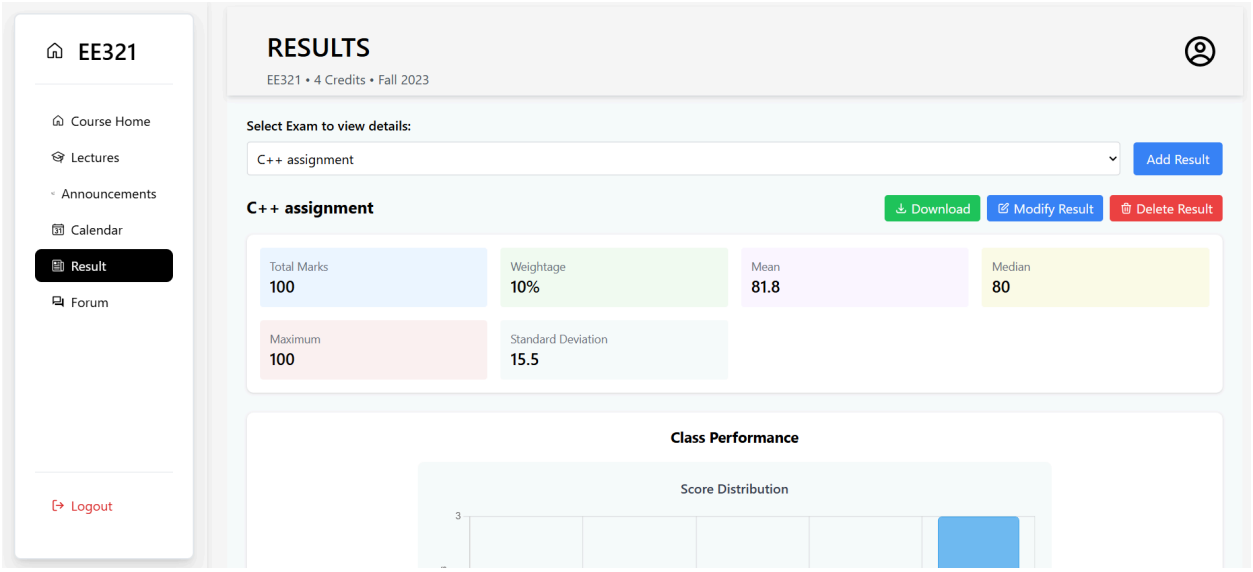


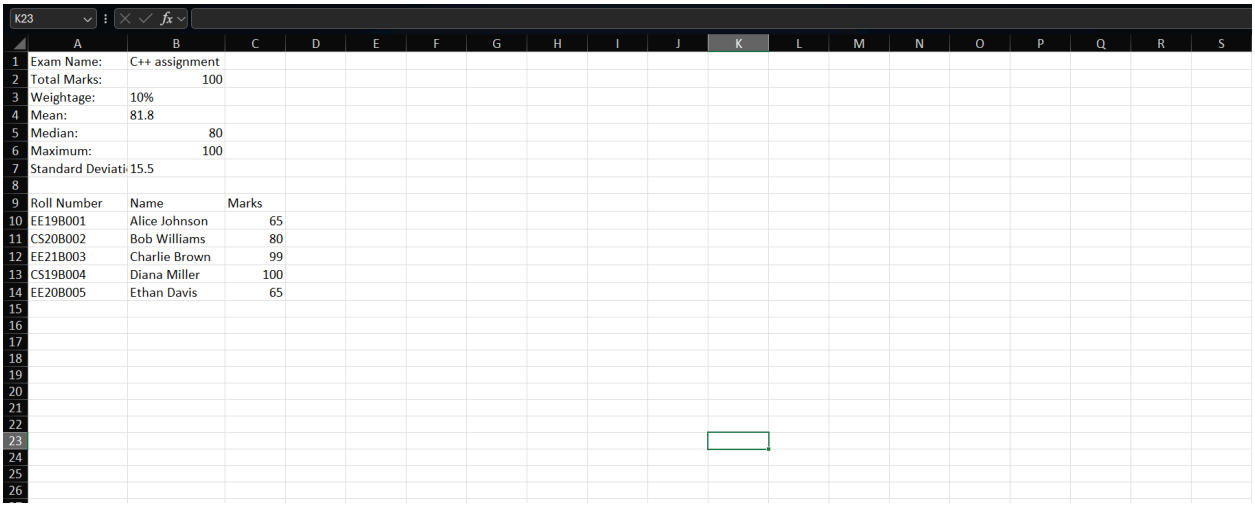




Result : The modify result option worked fine and correctly updated the database. Once the exam result is updated it correctly refreshes the screen for the faculty loading the details of the updated exam and also loads the same for all the enrolled students.

Details : To verify whether the download result option is working fine and does it provide data in expected excel format or not.

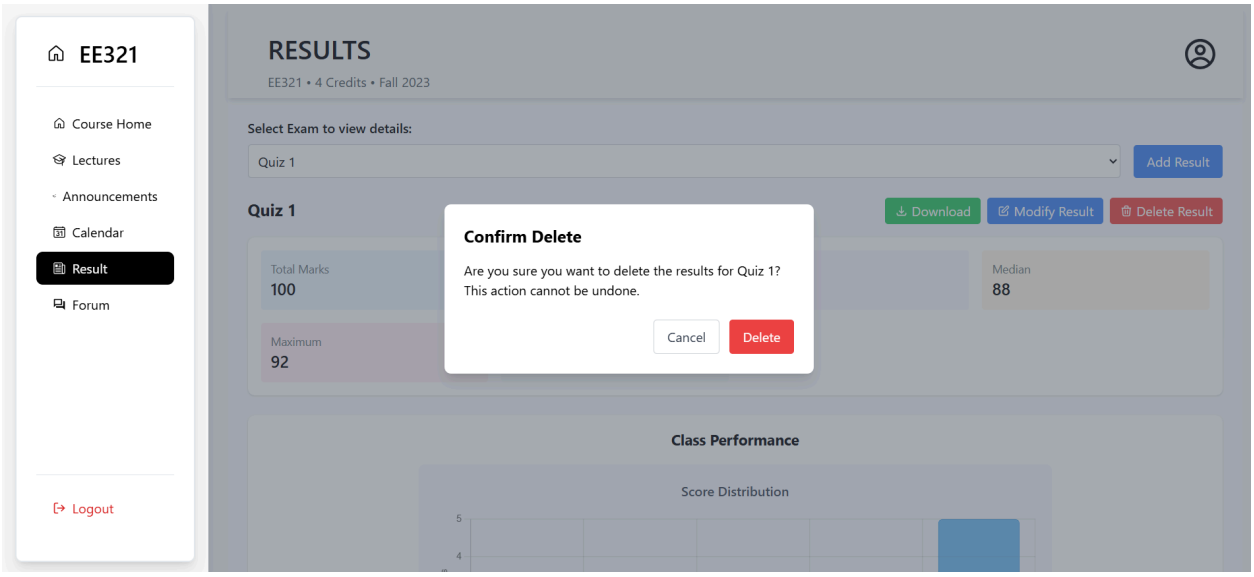


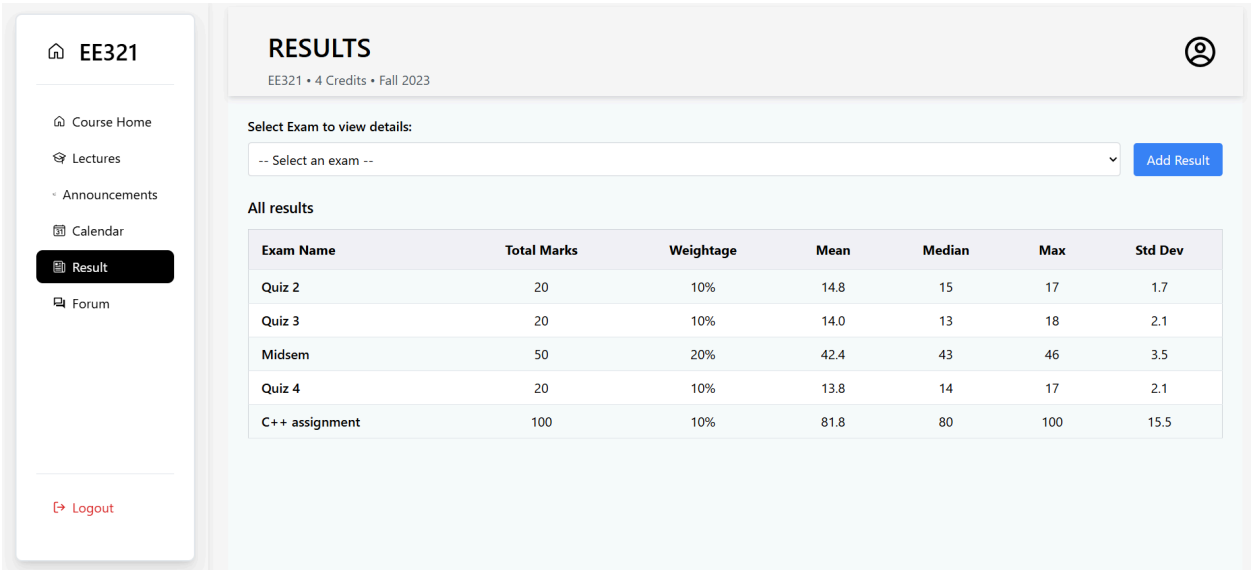


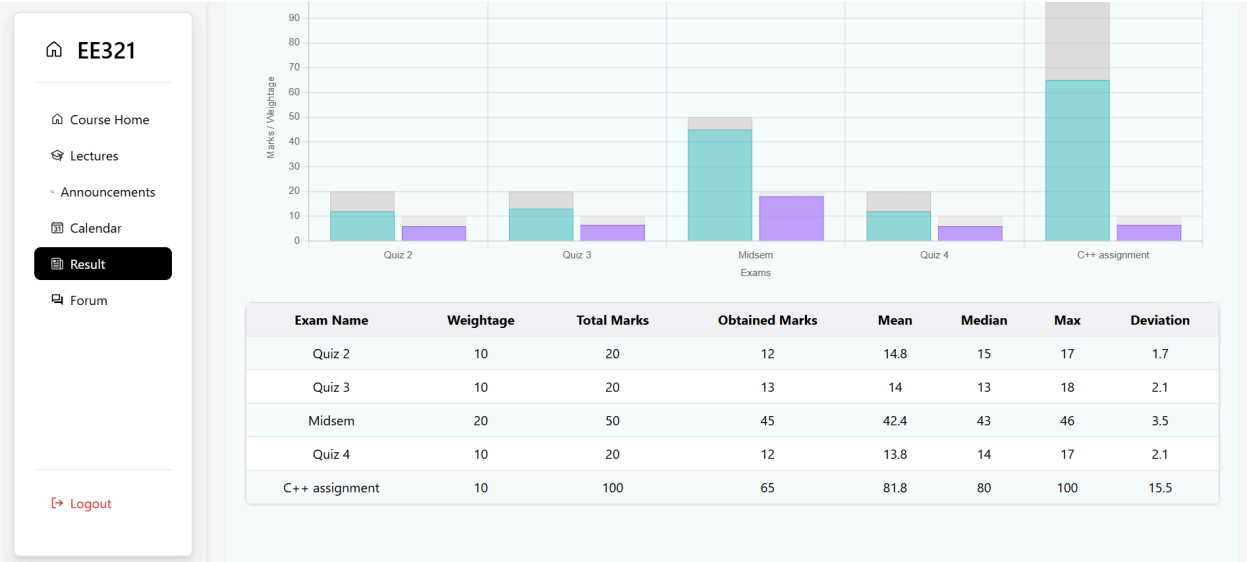
	A	B	C
1	Exam Name:	C++ assignment	
2	Total Marks:	100	
3	Weightage:	10%	
4	Mean:	81.8	
5	Median:	80	
6	Maximum:	100	
7	Standard Deviati	15.5	
8			
9	Roll Number	Name	Marks
10	EE19B001	Alice Johnson	65
11	CS20B002	Bob Williams	80
12	EE21B003	Charlie Brown	99
13	CS19B004	Diana Miller	100
14	EE20B005	Ethan Davis	65

Result : The download button worked fine as expected and it correctly exported data in expected format.

Details : To verify whether the delete result option is working fine and does this get updated in real time for both student and faculty of the course.


EE321
Course Home
Lectures
Announcements
Calendar
Result
Forum
Logout
RESULTS
EE321 • 4 Credits • Fall 2023
Select Exam to view details:
Quiz 1
Add Result
Quiz 1
Download
Modify Result
Delete Result
Total Marks
100
Median
88
Maximum
92
Confirm Delete
Are you sure you want to delete the results for Quiz 1? This action cannot be undone.
Cancel
Delete
Class Performance
Score Distribution


EE321
Course Home
Lectures
Announcements
Calendar
Result
Forum
Logout
RESULTS
EE321 • 4 Credits • Fall 2023
Select Exam to view details:
-- Select an exam --
Add Result
All results
Exam Name | Total Marks | Weightage | Mean | Median | Max | Std Dev
Quiz 2 | 20 | 10% | 14.8 | 15 | 17 | 1.7
Quiz 3 | 20 | 10% | 14.0 | 13 | 18 | 2.1
Midsem | 50 | 20% | 42.4 | 43 | 46 | 3.5
Quiz 4 | 20 | 10% | 13.8 | 14 | 17 | 2.1
C++ assignment | 100 | 10% | 81.8 | 80 | 100 | 15.5



Exam Name	Weightage	Total Marks	Obtained Marks	Mean	Median	Max	Deviation
Quiz 2	10	20	12	14.8	15	17	1.7
Quiz 3	10	20	13	14	13	18	2.1
Midsem	20	50	45	42.4	43	46	3.5
Quiz 4	10	20	12	13.8	14	17	2.1
C++ assignment	10	100	65	81.8	80	100	15.5

Result : The delete result option worked as expected and correctly updated the database. Once the exam result is deleted the module correctly refreshes the screen for the faculty removing the details of the deleted exam and also updates the view for all the enrolled students.

Additional Comments : NA

Lectures

Module Details:

Faculty members can design lectures for their courses, which will be accessible to all students enrolled in that course. The lecture page is structured as follows:

1. Modules: Professors can create modules, such as weekly units, to organize the course content.
2. Sections within Modules: Each module can be divided into multiple sections to further categorize the content.
3. Lectures within Sections: Professors can upload multiple lectures under each section, providing detailed materials for students.

Adding a Lecture:

Professors can create new lectures by filling out the required fields:

1. Lecture Title: Specify the name of the lecture.
2. Lecture Description: Provide a brief or detailed explanation of the lecture content.

3. Upload Files: Attach relevant files (e.g., PDFs, slides, or other resources).
4. YouTube Link: Include a link to a YouTube video for additional learning material.
5. Click Submit to save the lecture or Cancel to discard changes.

Editing a Lecture:

Professors can modify an existing lecture by:

1. Navigating to the lecture they want to edit.
2. Updating fields such as the title, description, uploaded files, or YouTube link.
3. Submitting changes to ensure the updated lecture is saved.

Deleting a Lecture:

If a lecture is no longer needed, professors can delete it:

1. Locate the lecture in the section.
2. Use the delete option (typically represented by a trash icon or button).
3. Confirm the deletion to permanently remove the lecture.

Managing Modules:

Professors can also manage modules effectively:

1. Creating Modules: Add new modules (e.g., "Week 1," "Week 2") to organize course content systematically.

Test Owner:

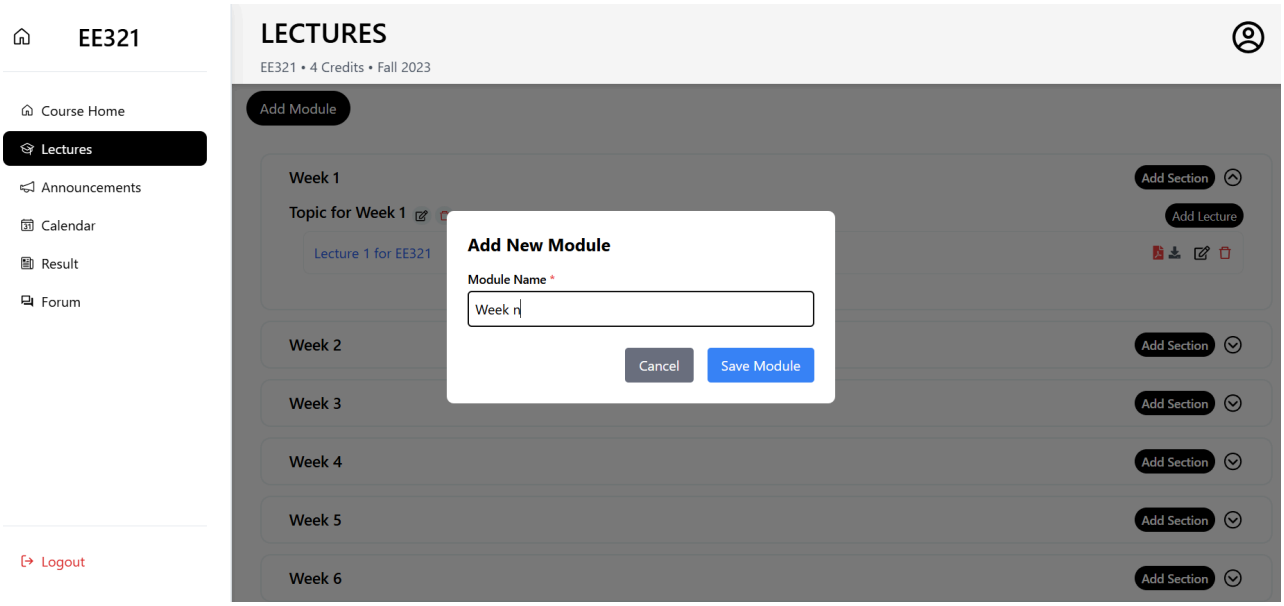
Abhijeet Agarwal, 210025

Test Date: `

2 April 2025

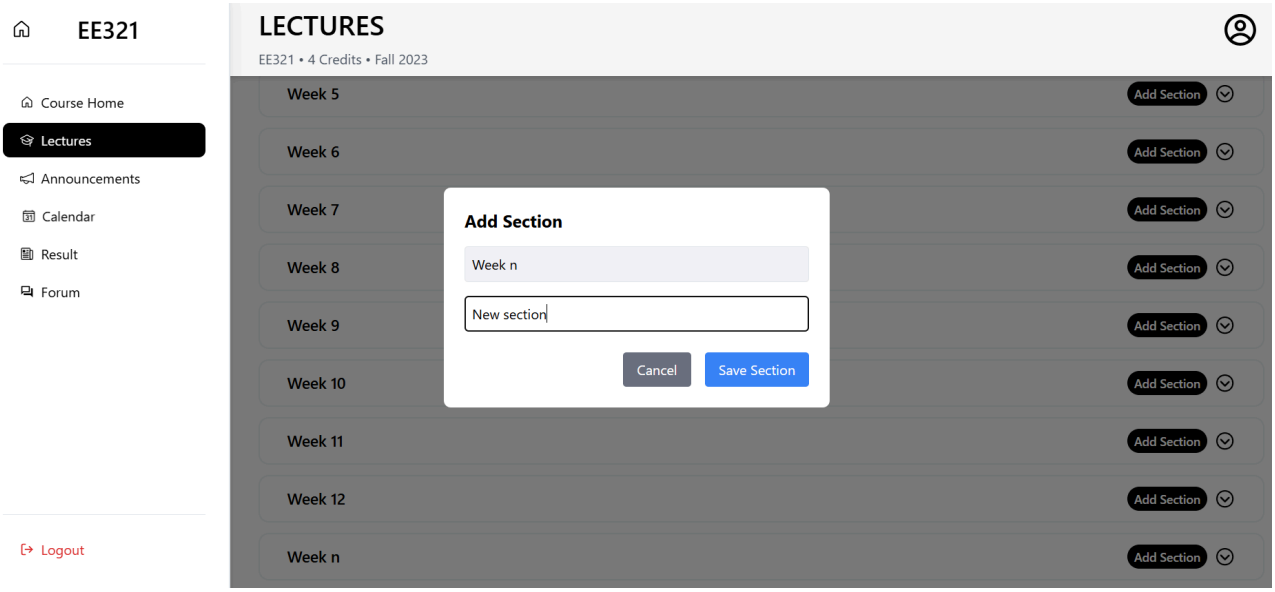
Test Results:

- When the "Add Module" button is clicked, and a module name is entered, the module is successfully created upon clicking "Save Module."

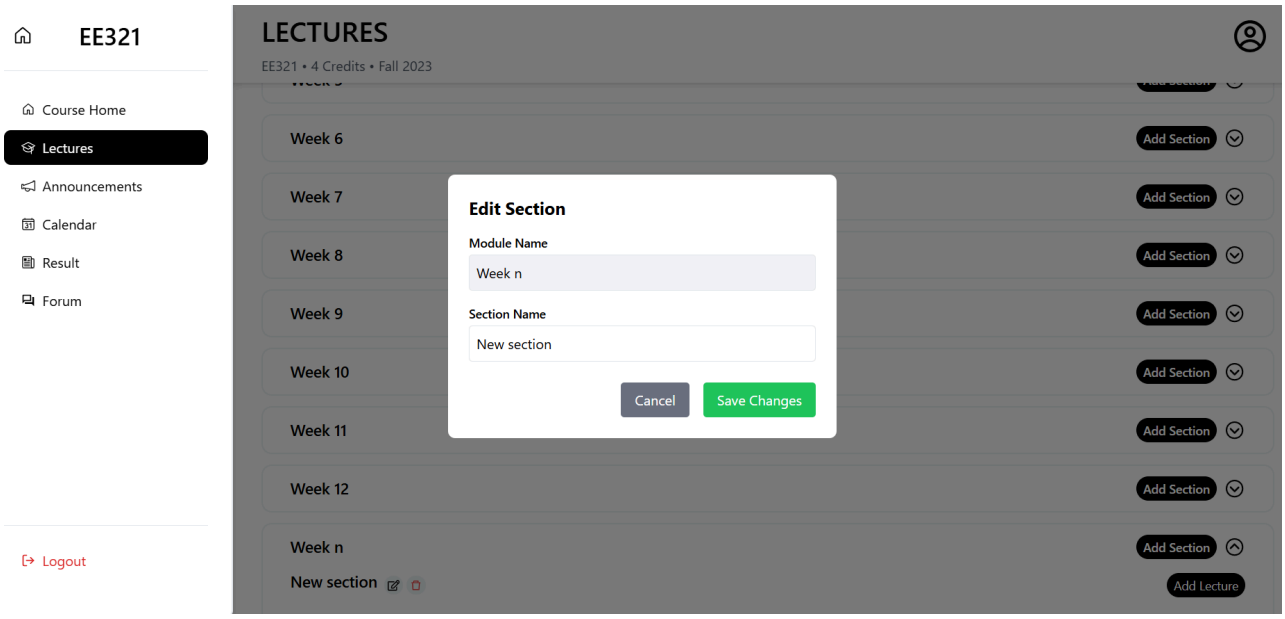


- When the "Add Section" button is clicked, and a section name is entered, the section is successfully created upon clicking "Save Section."

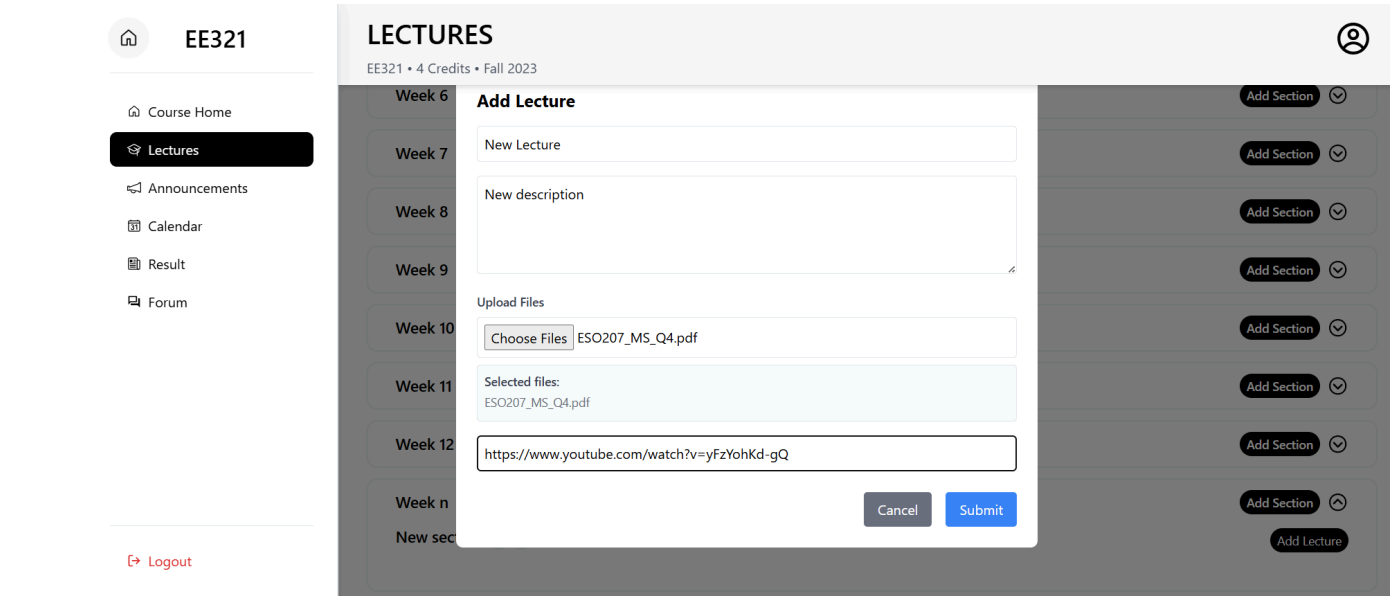
.



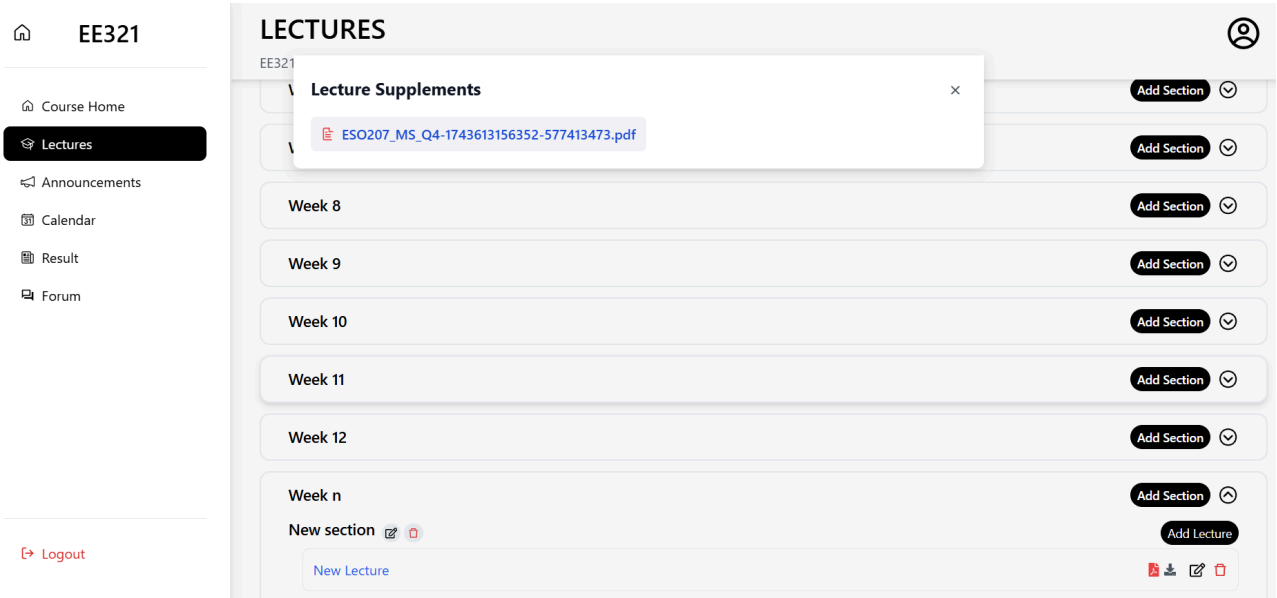
- The section name is updated successfully after modifying it and clicking "Save Changes."



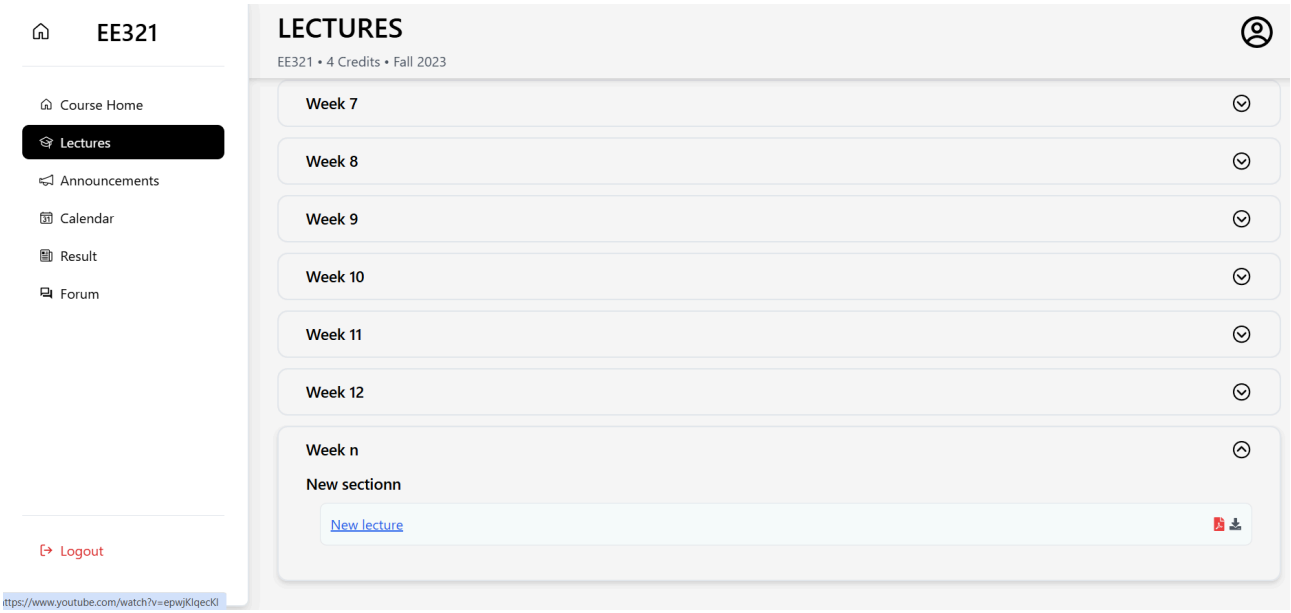
- When the "Add Lecture" button is clicked, and a lecture is added with the uploaded files and other details, the lecture is successfully created upon clicking "Submit."



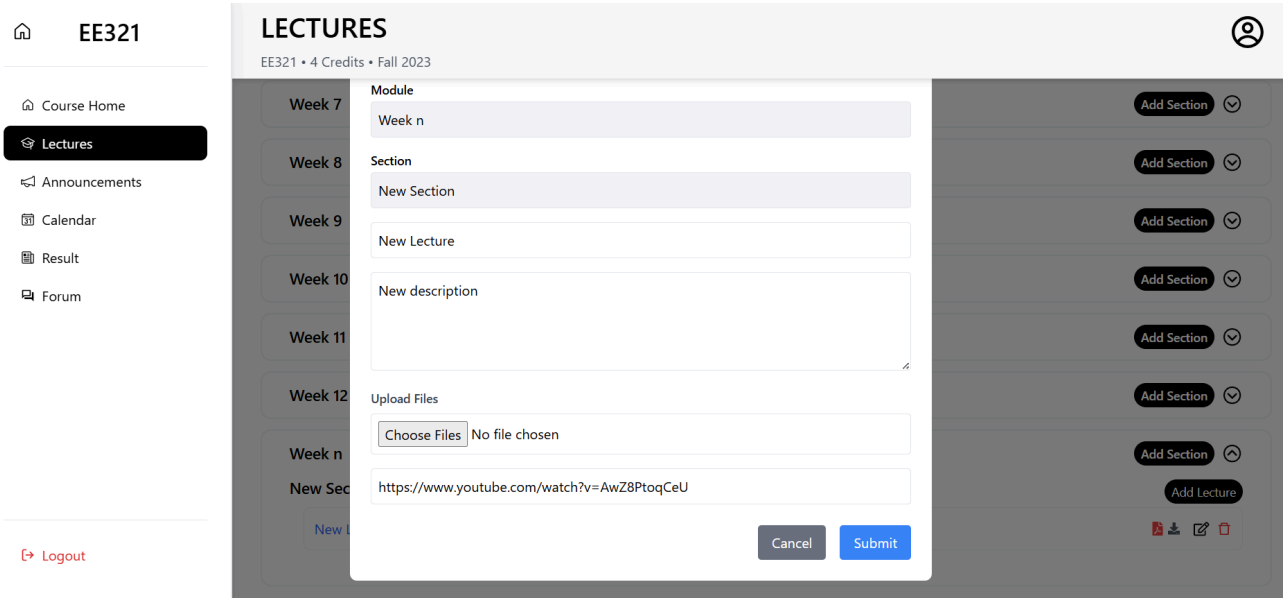
- Uploaded files are displayed with a "Download" button, allowing users to download them.



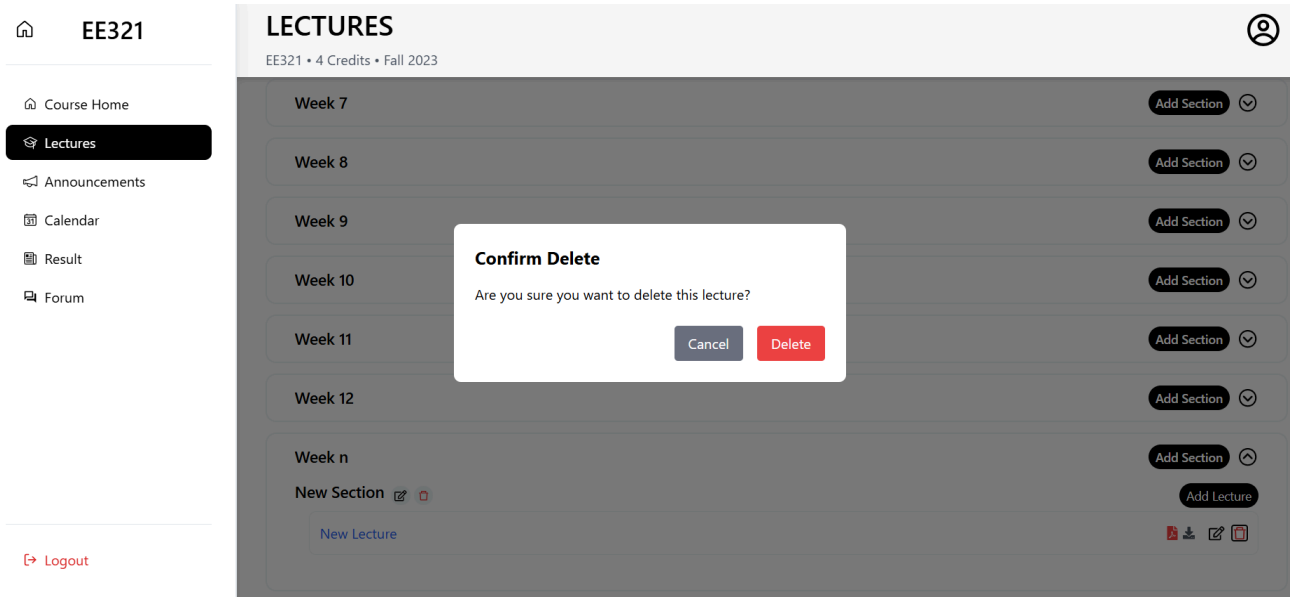
- The YouTube video link opens in a new tab when clicked on the lecture name.



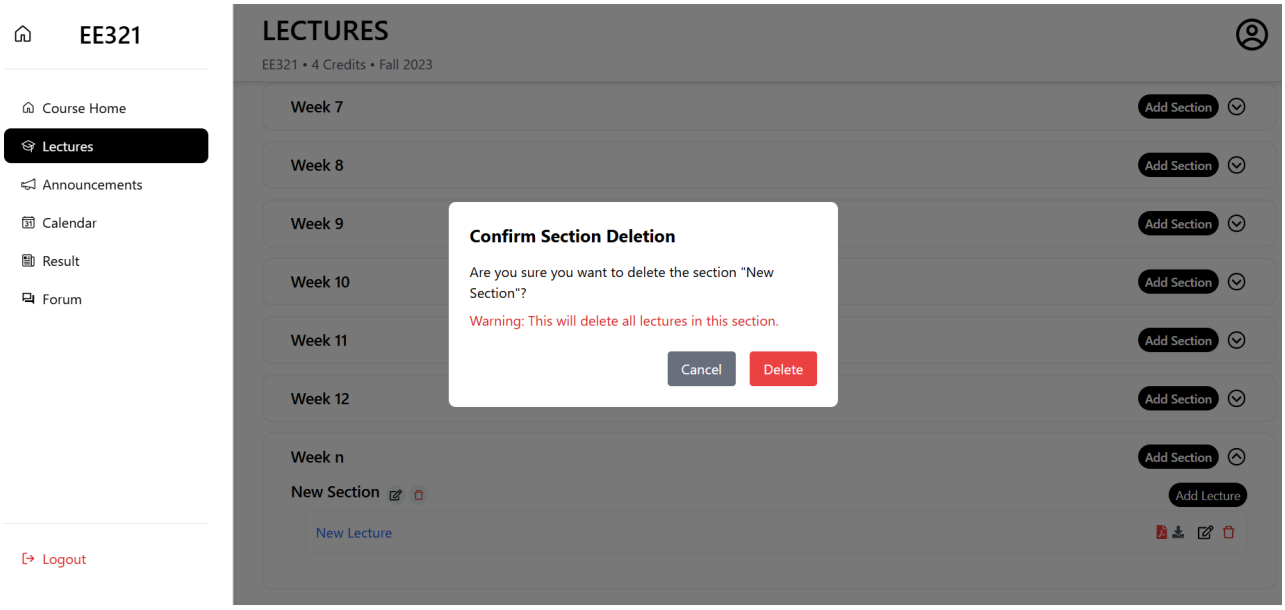
- Lectures are updated successfully with new resources and a modified name when edited by clicking on the edit button and then finally submitting the updated details with the submit button.



- Upon clicking "Delete Lecture," a confirmation dialog box appears. When confirmed, the lecture is deleted successfully.



- When deleting a section, a warning dialog box appears stating that all lectures within the section will be deleted. Upon confirmation, the section and its associated lectures are removed successfully.



Structural Coverage:

The unit is working well and fulfils 100% of the requirements of the SRS.

Additional Comments:

A visual bug was present in the add lecture and edit lecture dialogue box. It is fixed.

Forgot Password

Module Details:

Users can reset their password using the "Forgot Password" feature, which involves OTP verification via email. The process consists of three steps: First, the user enters their registered username, and if it exists in the system, an OTP is sent to the registered email address of the user. Next, the user inputs the received OTP, and the system verifies its correctness. If the OTP is valid, the user is allowed to set a new password. Finally, after confirming the new password, the system securely updates it in the database, completing the reset process.

Test Owner :

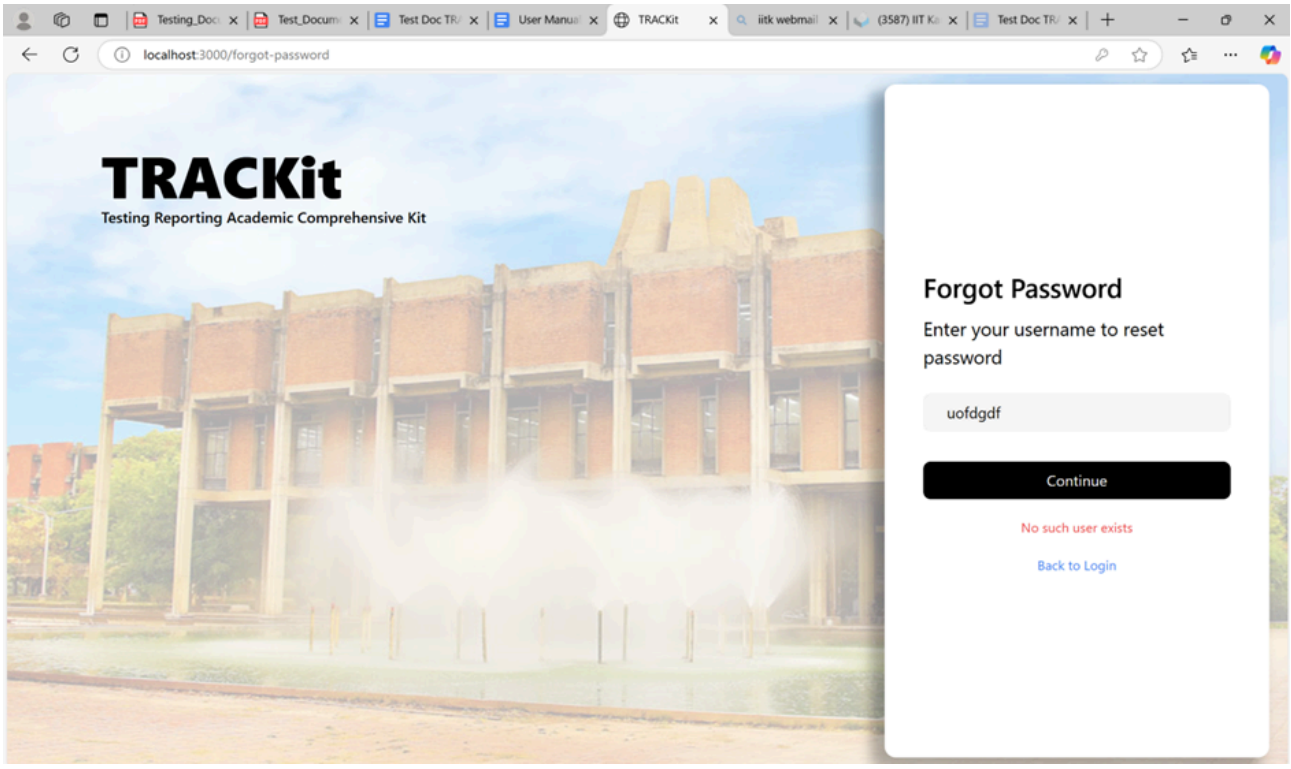
Akash Verma , 220097

Test Date :

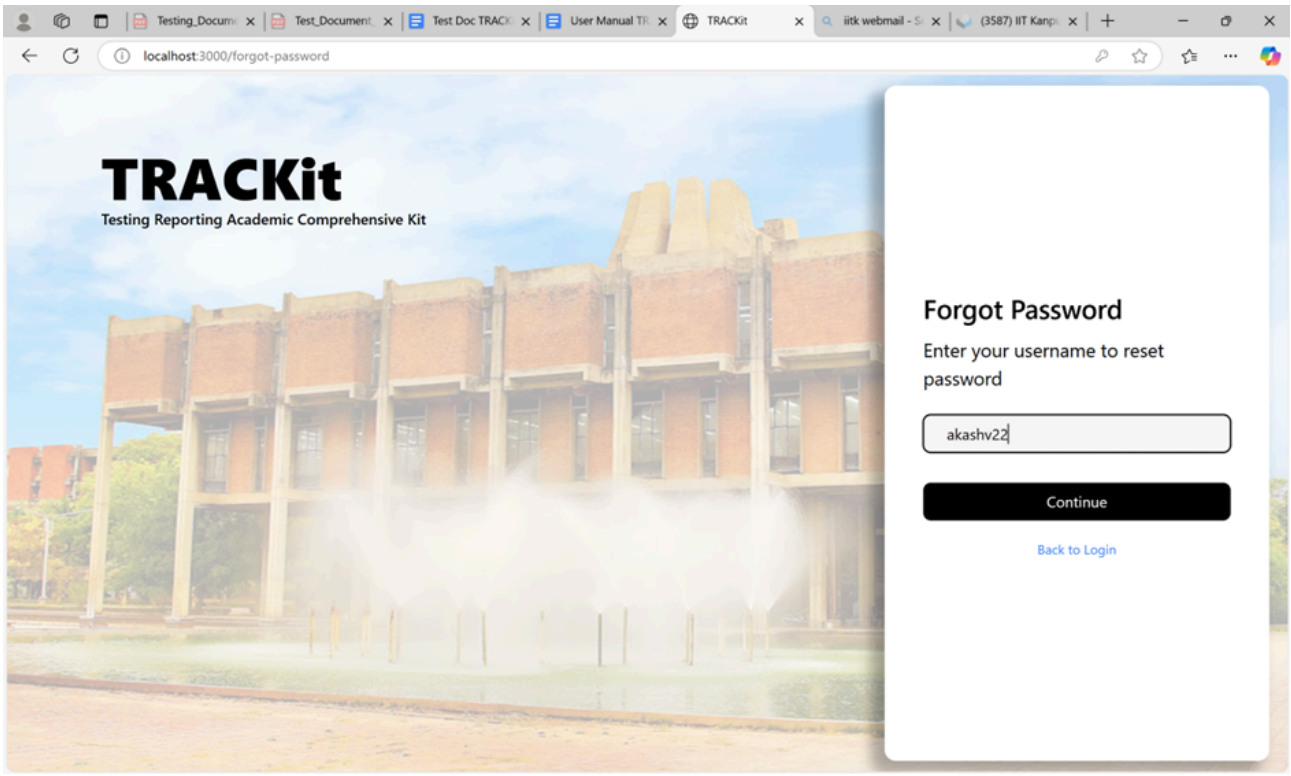
2 April 2025

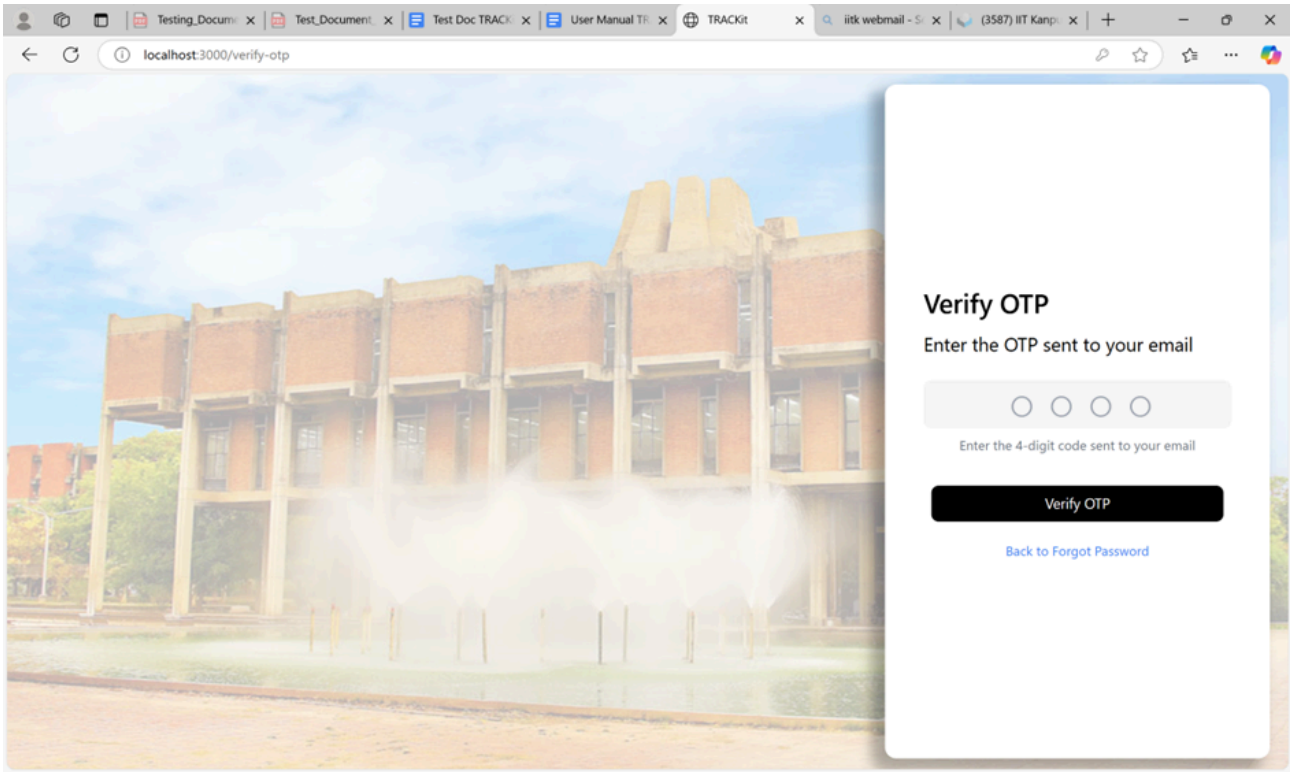
Test Results:

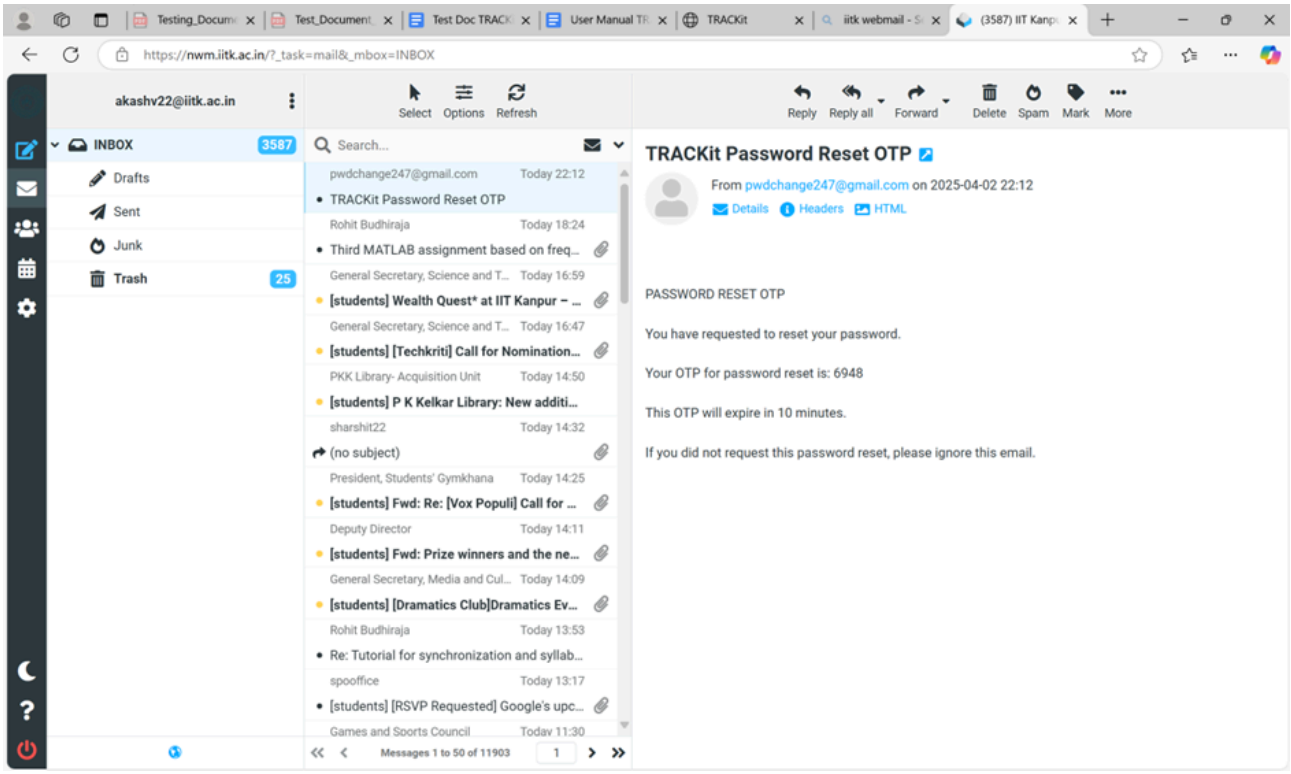
- On clicking the forget password button on the login page , the page for entering username correctly opens.



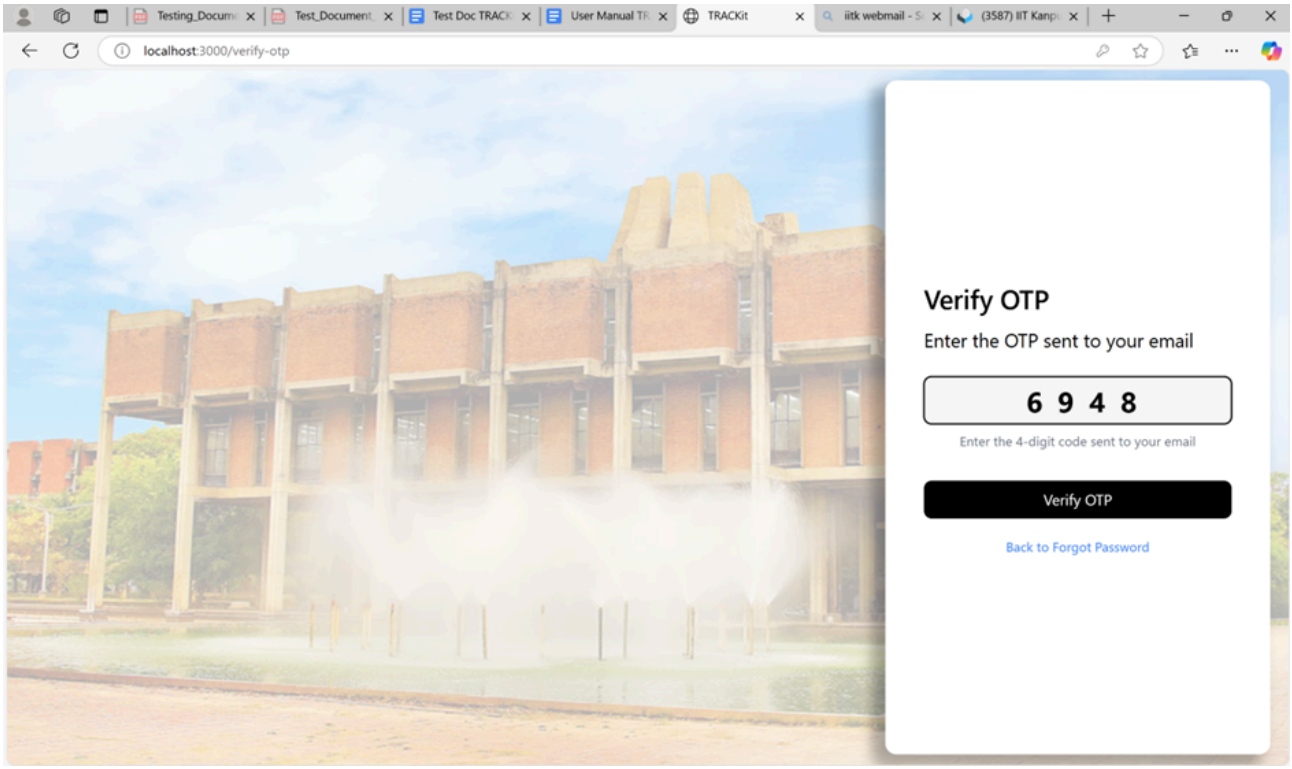
#Result1 : Correctly displays “No such user exists” on entering unregistered users.



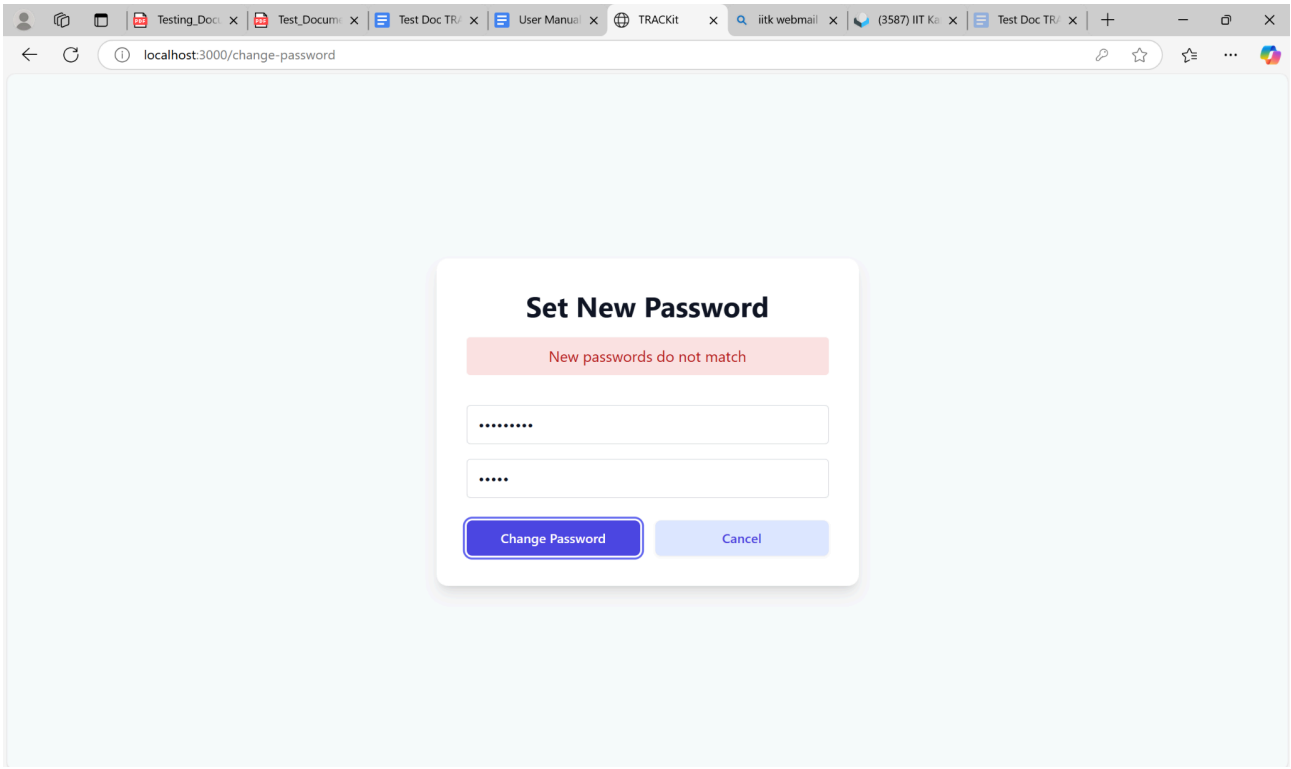


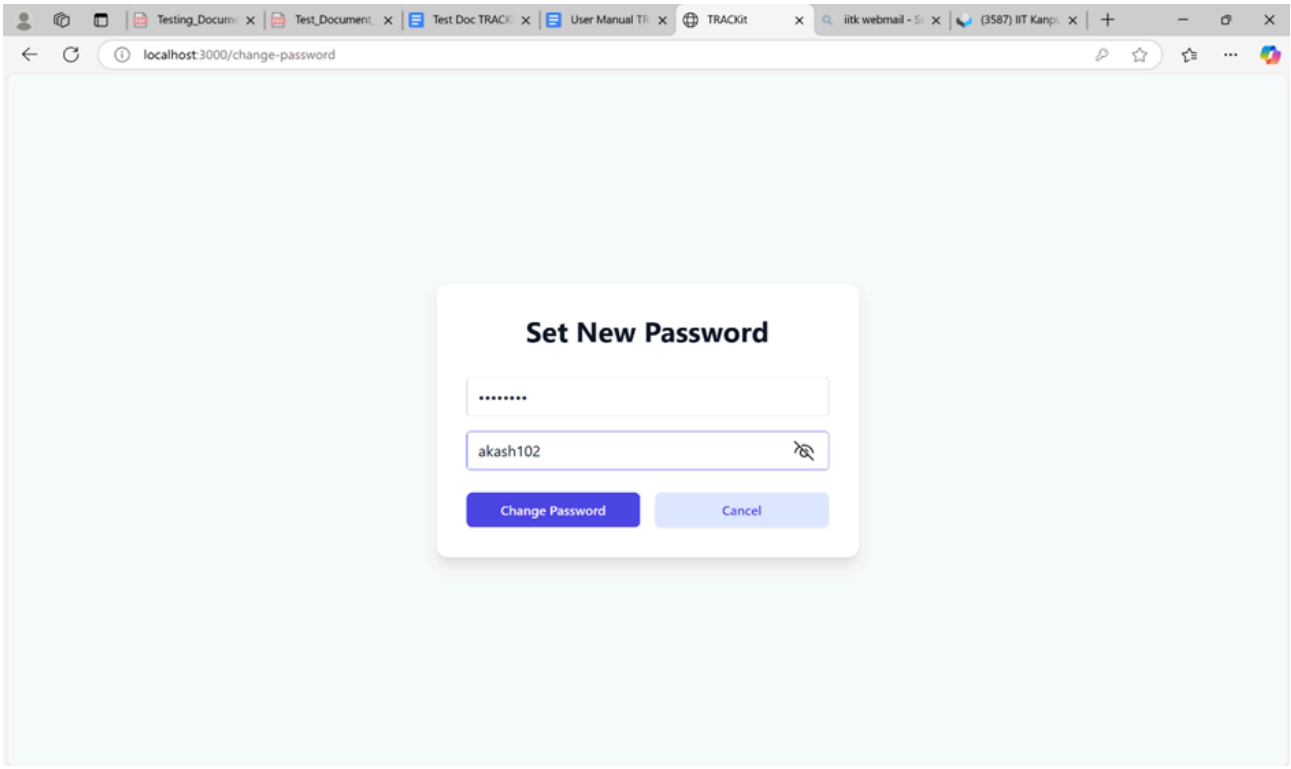


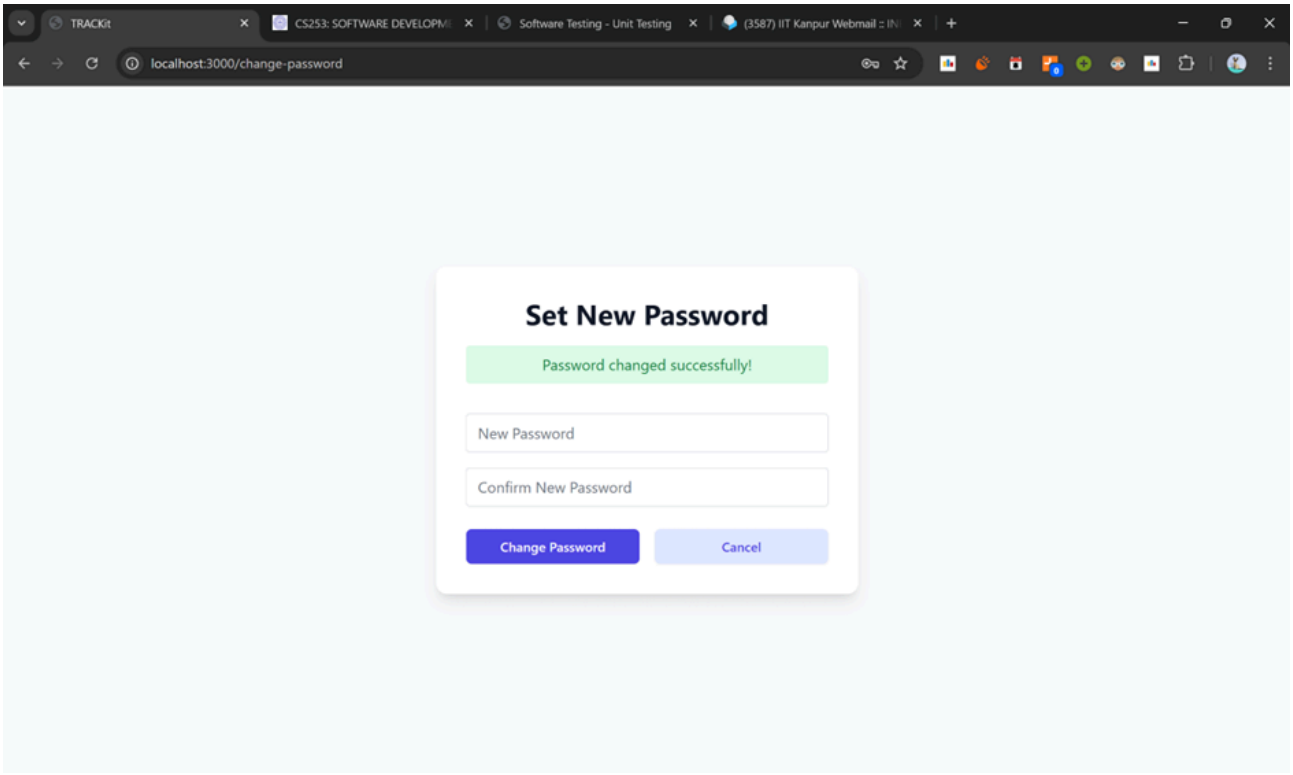
#Result2 : On entering **registered username** and pressing continue email is sent to **"registered email address"** and page to verify otp opens up correctly.



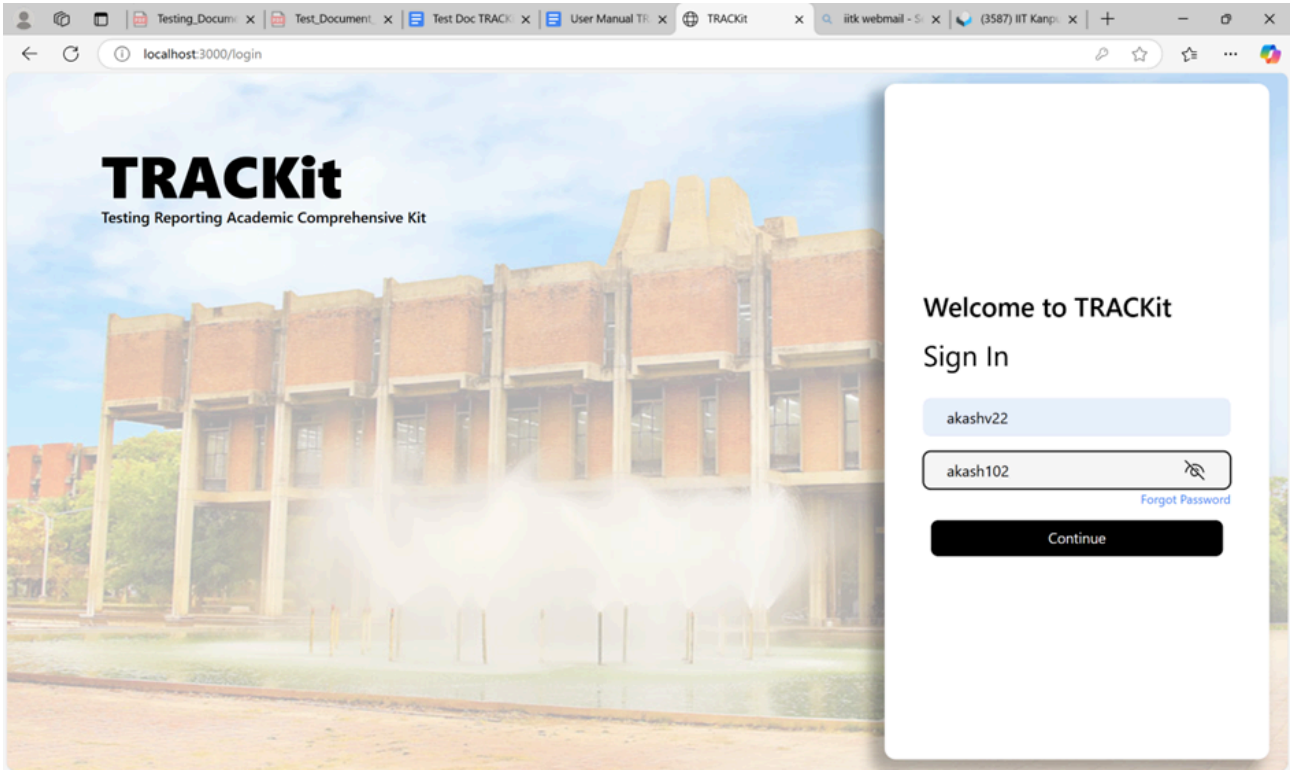
#Result3 : On entering **Correct OTP** , otp is verified successfully and user is correctly taken to **set new password page**

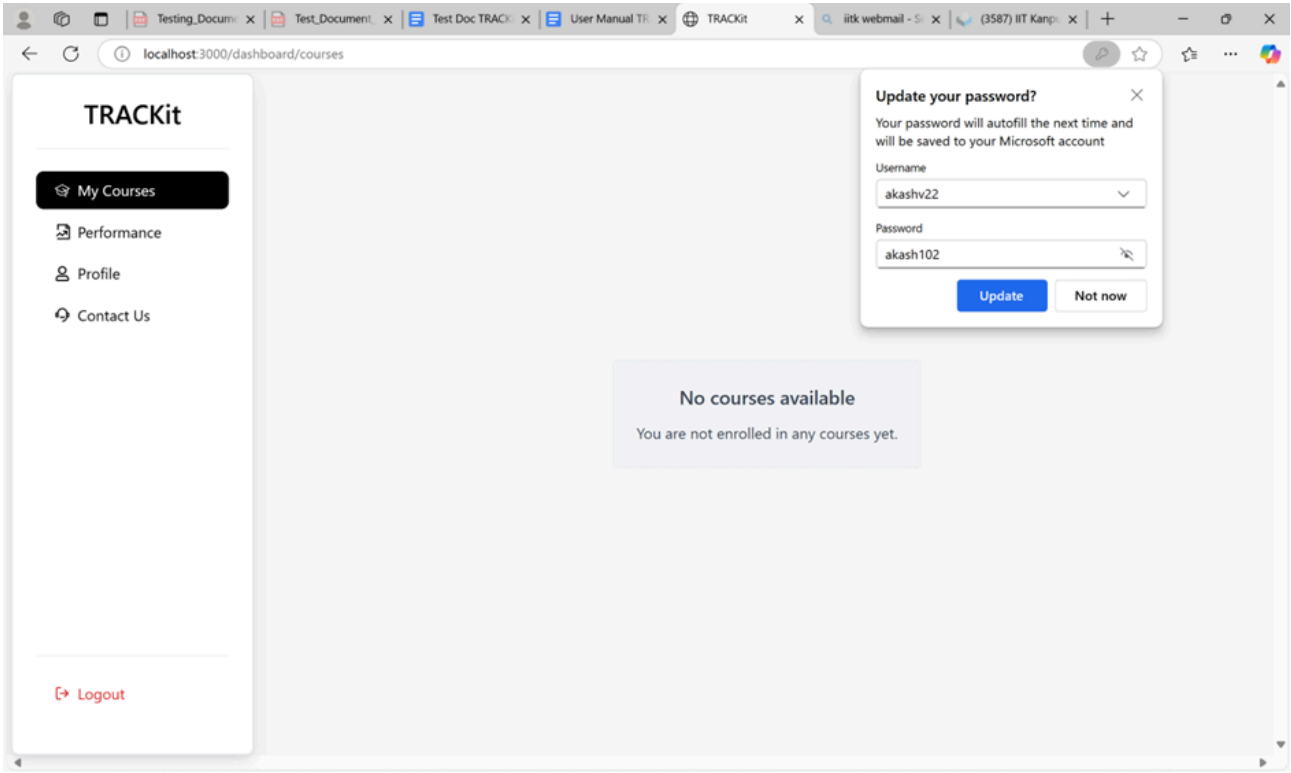



localhost:3000/change-password
Set New Password
akash102
Change Password
Cancel


TRACKit
Software Testing - Unit Testing
localhost:3000/change-password
Set New Password
Password changed successfully!
New Password
Confirm New Password
Change Password
Cancel

#Result4 : On entering a matching new and confirmed password, the password is getting changed correctly .





#Result5 : Password is successfuly updating in the backend and the user is able to login with the changed password perfectly.

Structural Coverage:

The module is working well and fulfills 100% of the requirements of the SRS.

Additional Comments: No bugs were found. Works perfectly smooth and fine .

FORUM :

Module Details :

This module encompasses testing procedures for Forum services inside a particular course. Api testing is done previously , following tests include testing of features like posting a forum , replying someone , deleting your own post and for faculty the feature to delete any student's response(s).

Test Owner :

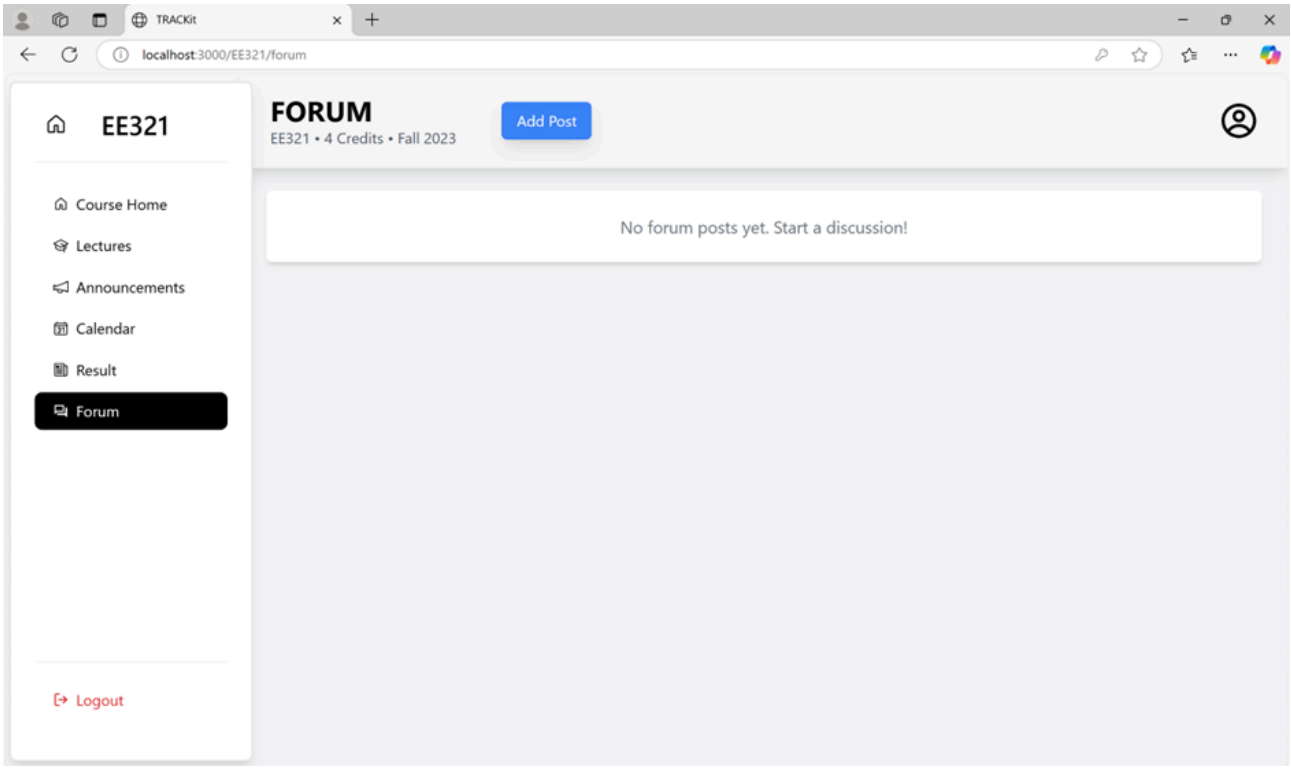
Akash Verma, 220097

Test Date :

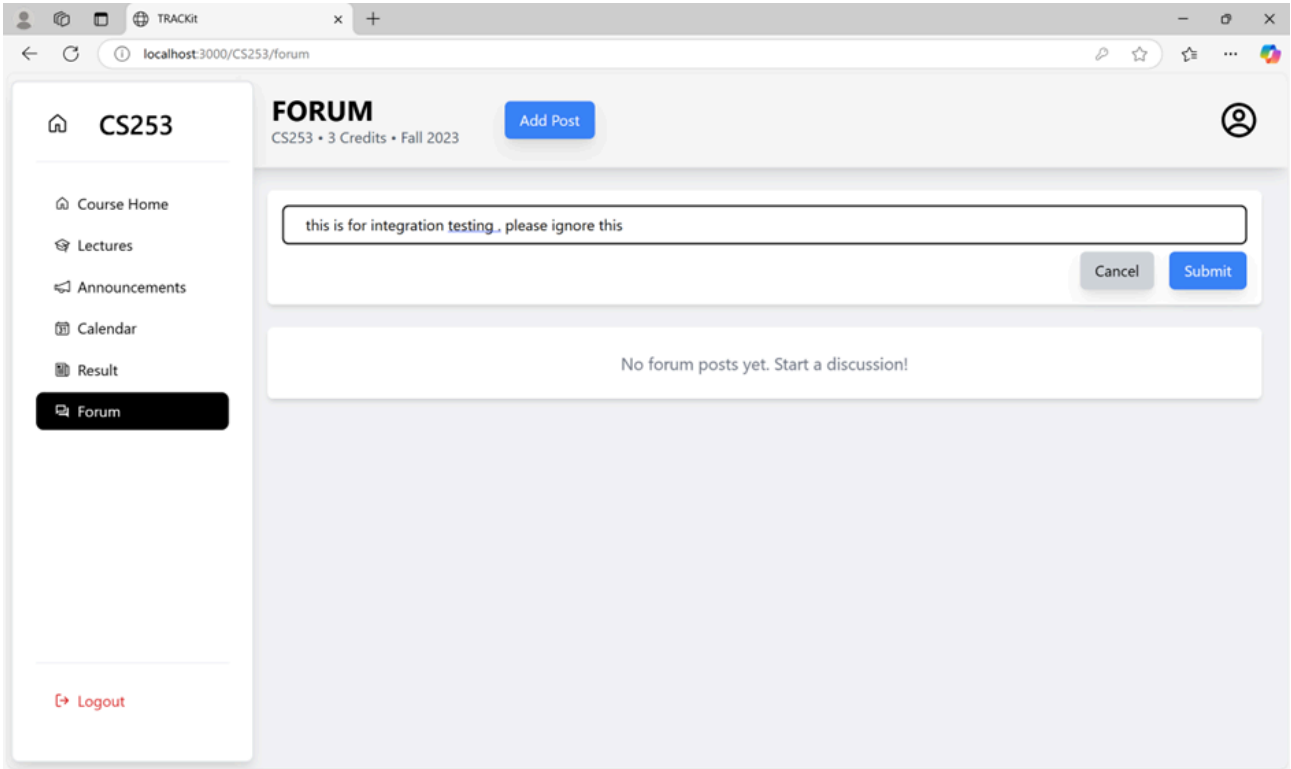
03/04/2025

Test Results :

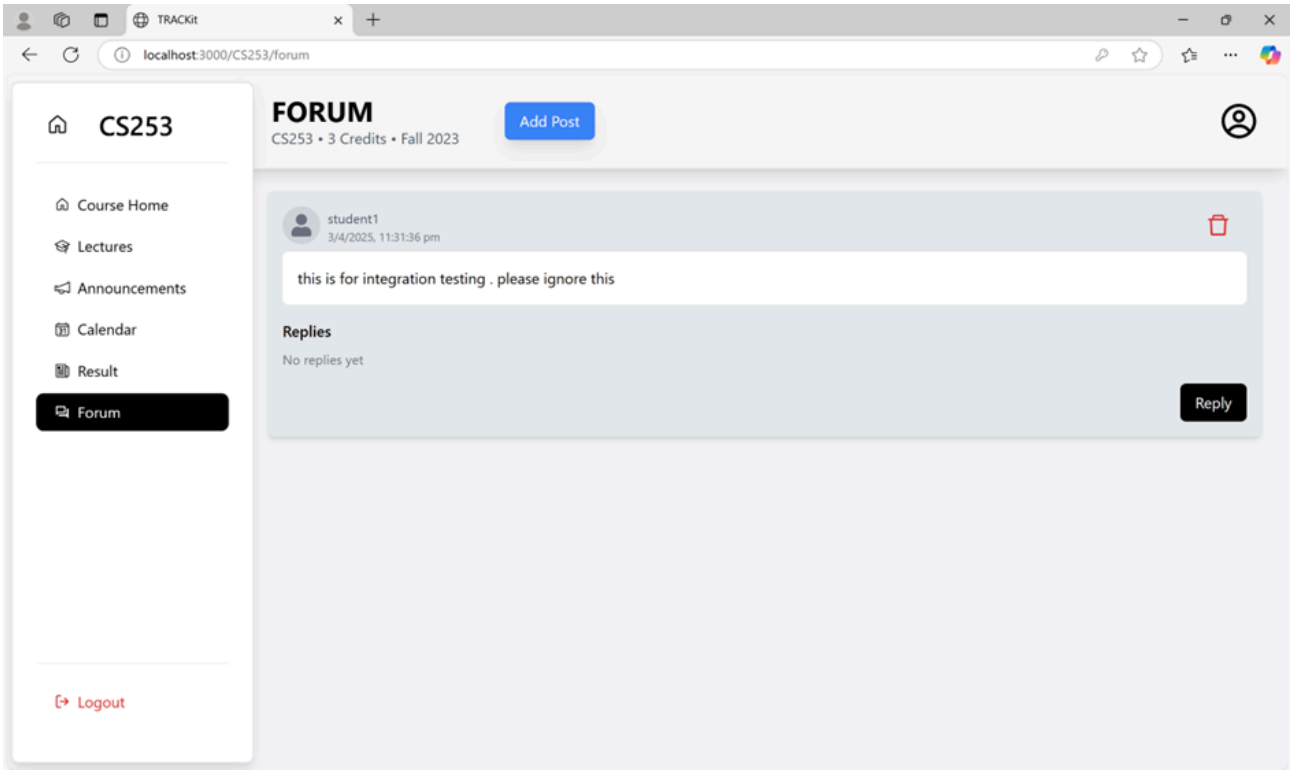
- After selecting an enrolled course in the main dashboard and selecting the forum option in the left panel , the forum page is opening correctly.



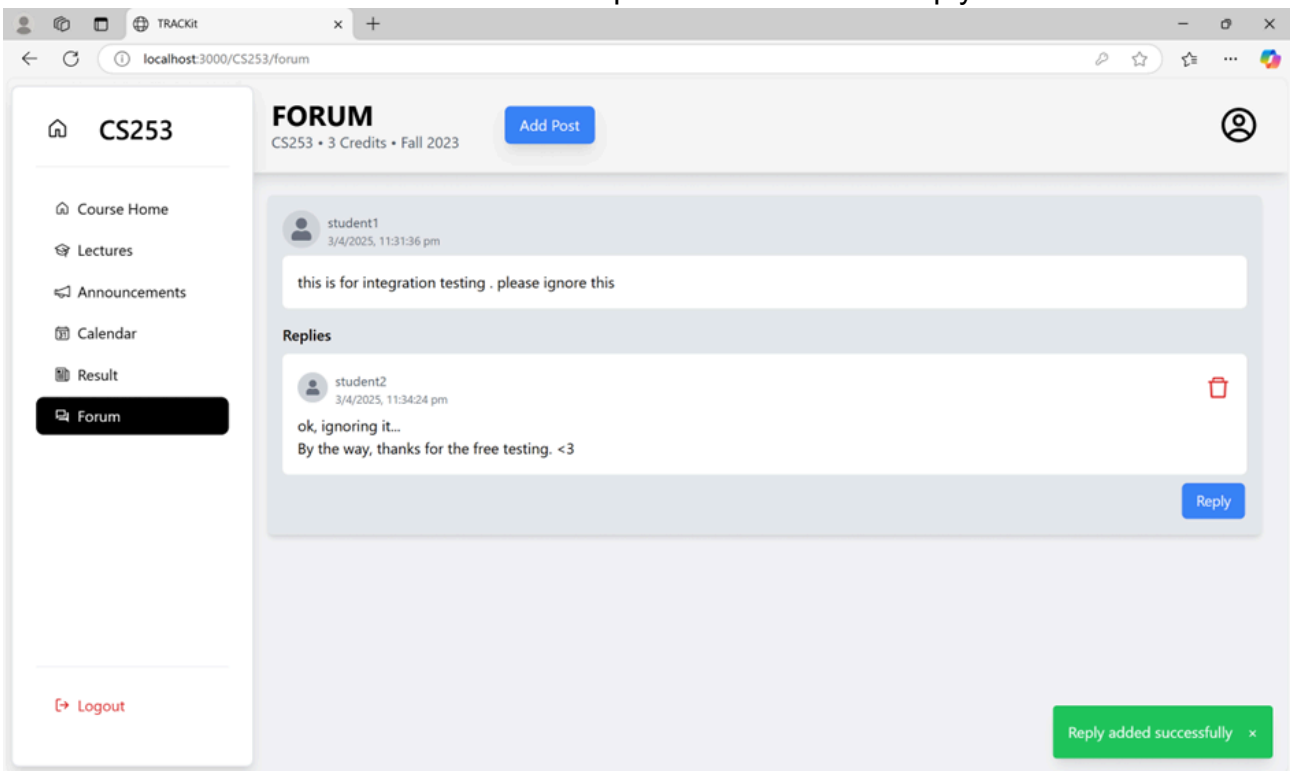
- On clicking upon Add post , a box to post new queries is opening successfully where one is able to write its queries:



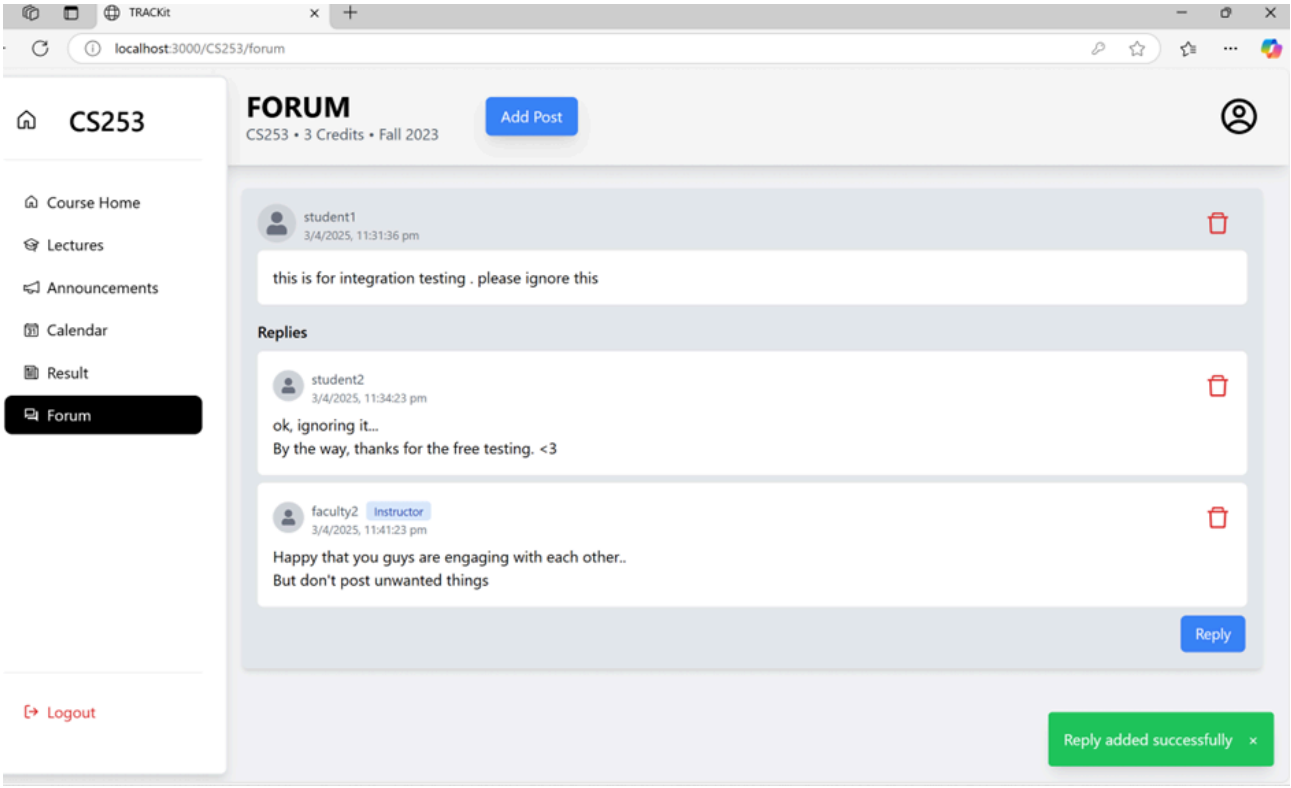
- On clicking “submit” button , forum is getting posted :



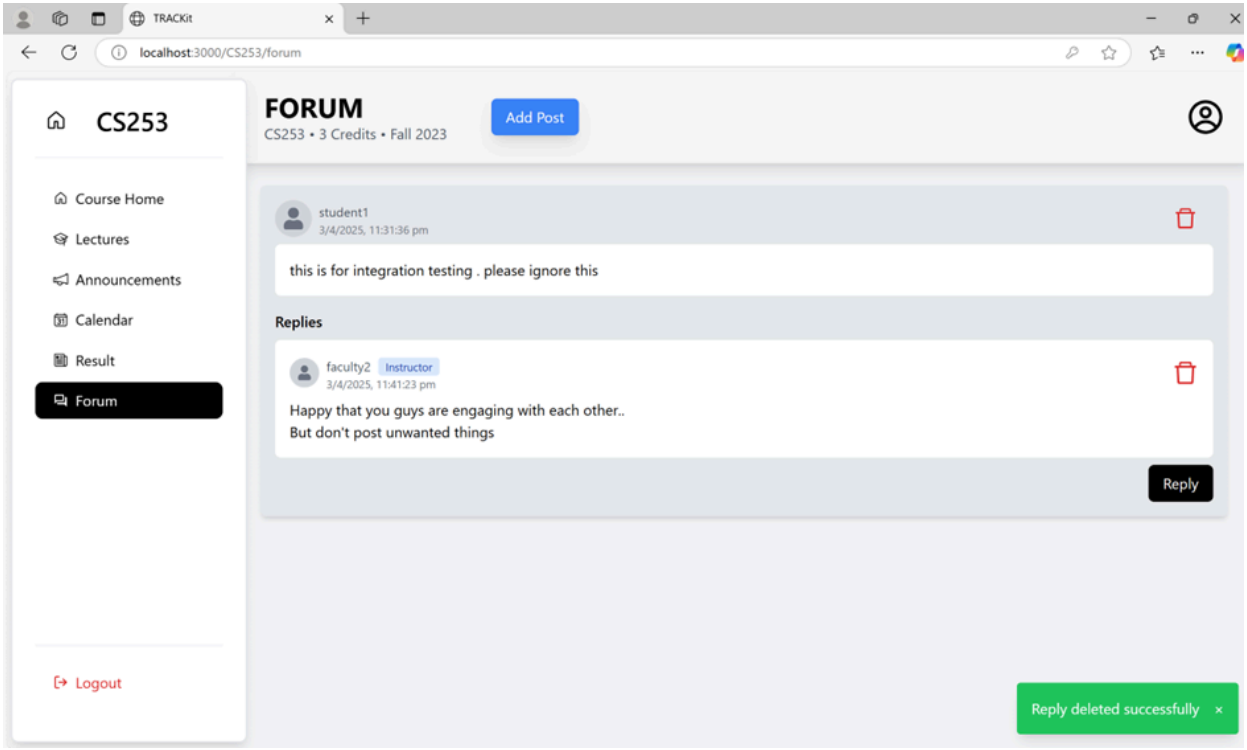
- Other users are also able to see the post and are able to reply to it.



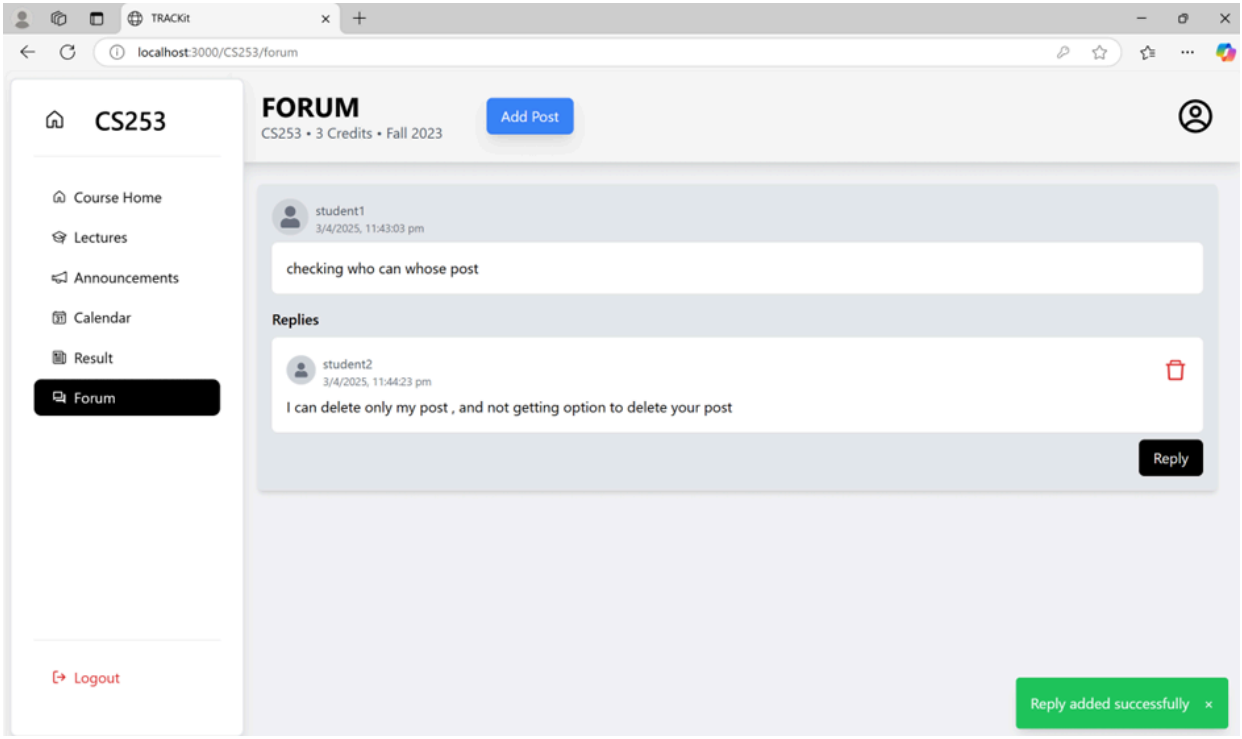
- Instructor is also able to see the post and able to reply on it :



- Instructor is also able to delete any students reply and also able to successfully delete the whole forum:



- Students are not able to delete any post **other than of their own** :



Structural Coverage:

The module is working well and fulfills 100% of the requirements of the SRS.

Additional comment :

- A bug was found that while creating a new forum post , the forum creator was not able to create a multiline post. Bug is fixed and now it is working fine.
- Apart from these attached tests , other things like occurrence of messages in increasing order of date and time etc. is also checked carefully .

4 System Testing

1. Requirement:
When the system is first initialized, no user accounts are present by default except for the **admin account**. The admin holds the authority to add students, faculty members, courses, and manage system-wide settings. Therefore, it is critical to verify whether the admin account is created and functional upon system startup.

We tested the admin login process by navigating to the login page and entering the default admin credentials provided during system setup. Upon successful authentication, the system redirected the user to the admin dashboard, confirming access to administrative functionalities such as adding users and courses.

To ensure robustness, we also attempted login with incorrect admin credentials and verified that the system displayed appropriate error messages (e.g., **“Invalid credentials”**). No issues or bugs were encountered. The admin account functioned correctly, satisfying the specified requirement.

Test Owner: Aryan Bansal
Test Date: 28/03/2025
Test Results: The admin account was successfully verified on the deployed system. No bugs were reported.

2. Requirement:
The **admin should be able to add students and faculty members** (i.e., users) to the system by manually entering their details, including **username**, **email**, **password**, and relevant **academic information**. Additionally, once added, users should appear in the system database and on the user management page accessible to the admin.

To validate this functionality, we performed a comprehensive test where the admin added multiple users with all necessary details. We confirmed that the newly created users were correctly inserted into the system's database. We then navigated to the **User Management Page**, where the admin can view and manage all users, and verified that the newly added users were accurately displayed.

All operations executed as expected. No bugs or inconsistencies were found during testing, confirming that the functionality met the requirement.

Test Owner: Aryan Bansal
Test Date: 28/03/2025
Test Results: User addition, listing in the admin view, and password update functionality worked as expected. No bugs were reported.

Additional Comments: The username and password could be e-mailed on the registered email address.

3. Requirement:
The system provides a **Bulk Upload** feature that allows the admin to add multiple users (students and faculty) at once by uploading a properly formatted **CSV file**. The system should validate the uploaded file and display an error message if the format is incorrect or if the file content is invalid. If the file is correct, all users listed in the CSV should be successfully added to the system database.

First, we uploaded a **correctly formatted CSV file** containing valid user details for both students and faculty. After submission, we verified that all users were accurately added to the system database and were visible on the **User Management Page**.

Next, we tested the system's validation logic by uploading an **incorrectly formatted CSV file**—one with missing columns, invalid headers, or incomplete data. In each case, the system displayed an appropriate **error message**, ensuring that no incorrect entries were added.

Test Owner: Aryan Bansal
Test Date: 28/03/2025
Test Results: Bulk upload with correct CSV succeeded; errors were appropriately handled for incorrect files. No bugs were reported.

4. Requirement:
While the admin is adding users (students or faculty), the system should **enforce uniqueness** for certain critical identifiers to maintain data integrity. Specifically:

- No two users should have the **same username**.
- No two users should be registered with the **same email address**.

- In the case of students, the **roll number** must also be unique across the system.

To test this functionality, we initially attempted to add users with duplicate usernames, email addresses, and student roll numbers. The system should **prevent duplication** for usernames and emails by displaying an appropriate error message.

However, during early testing, we discovered that the system allowed the creation of **multiple student accounts with the same roll number**, which was a bug. This issue was reported, and the development team addressed it promptly. After the fix was applied, we re-tested the scenario and confirmed that the system **no longer allows duplicate roll numbers** for students.

The final outcome verified that all three uniqueness checks were functioning as required.

Test Owner: Aryan Bansal
Test Date: 29/03/2025
Test Results: Duplicate prevention for username, email, and roll number now works as expected. One initial bug with roll number duplication was resolved and re-tested.

5. Requirement:
Once a user (student or faculty) has been added to the system, they should be able to **log in using their username and password**. Additionally, the system must support a **"Forgot Password"** feature to allow users to reset their password if forgotten or to update the temporary password provided by the admin or according to their needs.

The password reset workflow includes the following steps:

- The user clicks on the **"Forgot Password"** option.
- The system prompts the user to enter their **username**.
- If the username is valid, an **OTP is sent to the user's registered email address**.
- The user must then enter the received OTP to proceed to **reset the password**.
- The system should only accept the **new password** for future logins, rendering the old one invalid.

During testing:

- We first verified that users could **successfully log in** using the correct username and password.
- We then tested the **"Forgot Password"** functionality. Initially, the OTP **email was not being sent**, which was reported as a bug.
- After this issue was resolved, we verified that the email with the OTP was successfully received by the user, and the OTP validation and password change process worked as expected.
- Lastly, we confirmed that users were **able to log in using the new password** and that the old password was no longer valid.

This test validated both the core login functionality and the robustness of the password recovery mechanism.

Test Owner: Mayur Agrawal
Test Date: 29/03/2025
Test Results: Login and password reset functionalities worked as expected after an initial email delivery issue was resolved. No further bugs were reported.

6. Requirement:
The admin should be able to **add new courses** to the system. These courses will later be assigned to faculty and students as part of academic scheduling. A **Course Management** page is provided, where all existing courses can be viewed and managed.

During testing, we verified the following:

- The admin could **access the Course Management page**.
- New courses could be added by entering course-related details such as course name, code, and description.
- After submission, the course appeared in the **list of existing courses**, confirming that the addition was successful.
- We also checked whether the data was correctly stored in the backend/database.

Test Owner: Mayur Agrawal
Test Date: 29/03/2025
Test Results: Course addition and listing features worked as expected on the Course Management page. No bugs were reported.

7. Requirement:
The system should provide a **User Management** page accessible to the admin. This page should allow the admin to **view and manage all users** (students and faculty) within the system.

Key functionalities include:

- Viewing user details such as **username, email address, and academic information** (excluding passwords).
- Editing user information, such as **username, email**, or academic details. Ensuring that **if a username is updated**, the affected user can only log in using the **new username**, and the old one becomes invalid.
- **Deleting users** from the system, which should remove their data from the database and revoke access (i.e., deleted users should not be able to log in with old credentials).

During testing:

- We navigated to the User Management page and confirmed that all users were correctly listed.
- We successfully **edited user details**, including changing usernames and academic data, and verified that the updated information was reflected both in the UI and the database.
- We **deleted sample users** and ensured they were removed from the database and could no longer access the system.
- We confirmed that after a username change, **login was only possible using the updated username**.

The entire flow was tested across different types of users, and the feature behaved as intended.

Test Owner: Aryan Bansal
Test Date: 30/03/2025
Test Results: The User Management page worked as expected. All edit and delete operations were successfully performed. No bugs were encountered.

8. Requirement:

The admin should be able to **manually assign students or faculty members to specific courses** through the User Management page.

Functionality tested:

- The admin locates a user (student or faculty) via the User Management interface.
- The admin selects the desired course from the available list and assigns it to the user.
- Upon successful assignment, the user should **see the course listed** in their dashboard after logging in, with access to its content.

During testing:

- We verified that users were successfully added to the selected courses.
- We confirmed that these users could access the course and its material upon logging in.
- Backend validations ensured the mapping between users and courses was properly stored in the database.

Test Owner: Aryan Bansal
Test Date: 30/03/2025
Test Results: The manual user-to-course assignment feature functioned correctly. Users were able to access newly assigned courses. No issues were encountered.

Additional Comments:

- A suggested improvement is to allow **students to request enrollment** in a course, which the corresponding instructor can then **accept or reject**. This would significantly reduce the workload on the admin.
- Another potential enhancement is the **auto-enrollment** of students into mandatory courses based on their department and academic year, streamlining the process further.

9. Requirement:

The **Course Management** page should allow the admin to **view and manage all courses** in the system.

Functionality tested:

- The admin was able to **view a complete list of courses** currently added to the system. Each course entry displayed key information such as **course code, name, and id**.
- The admin could **edit any of these course details** (e.g., correcting course codes or updating descriptions), and these changes were **immediately reflected for all users** (students and faculty) enrolled in that course.
- The admin was also able to **delete any course** from the system. After deletion:
    - The course was no longer visible on the Course Management page.
    - Users previously enrolled in the course could no longer access it from their dashboard.
- Additionally, the admin was able to **view the list of users** (both students and faculty) assigned to each course and had the option to **remove any user** from the course if necessary.

Test Owner: Aryan Bansal
Test Date: 30/03/2025
Test Results: The Course Management page performed as expected. All functionalities, including course modification, deletion, and user management within courses, worked correctly. No bugs were encountered during testing.

10. Requirement:
The **Course Management** page should allow the admin to **bulk upload students into a course** using a properly formatted CSV file.

Functionality tested:

- The admin was able to **select a course** and upload a CSV file containing a list of student usernames to be added to that course.
- The system validated the uploaded file. If the **CSV file format was incorrect** (e.g., missing columns, improper headers), or if the file contained **usernames not already registered in the system**, the system displayed **clear error messages** and **prevented the upload**.
- For **correctly formatted CSV files** containing valid student usernames, all users were **successfully enrolled in the selected course**.
- Once uploaded, these students could **see the course in their dashboard** upon logging in and access course content.

Test Owner: Aryan Bansal
Test Date: 31/03/2025
Test Results: The bulk upload feature worked as expected. Proper error handling was observed for invalid files, and students listed in valid CSV files were correctly enrolled in the course. No bugs were reported.

11. Requirement:
There should be a **"Contact Us"** page accessible to the **admin**, where the contact details of the personnel responsible for the maintenance of the *Track It* system are displayed.

Functionality tested:

- The page was accessible from the admin dashboard.
- It correctly displayed the **names**, **email addresses**, **contact numbers**, and **roles** of all relevant maintenance team members.
- The information was **read-only** to the admin, ensuring data integrity.

Test Owner: Aryan Bansal
Test Date: 31/03/2025
Test Results: The contact information page worked as expected. All details were displayed correctly and were accessible to the admin without any errors. No issues were reported.

12. Requirement:
All types of users — **admin**, **faculty**, and **students** — should be able to log out from their respective accounts. Upon successful logout, users should be redirected to the **login page**, and their session should be terminated.

Functionality tested:

- The **logout button** was visible and accessible from the navigation bar/dashboard for all user types.
- On clicking logout, users were redirected to the login page without delay.
- Logout functionality was tested on different user roles to ensure uniform behavior.
- No residual session data remained post logout.

Test Owner: Mayur Agrawal
Test Date: 01/04/2025
Test Results: Logout functionality worked as expected for admin, faculty, and student accounts. All users were redirected to the login page after logout, and no bugs were reported.

13. Requirement: Only faculty members assigned as instructors of a course should have the ability to post, edit or delete announcements on the Course Announcement page. Students should only have viewing access. All users (faculty and students) added to a course must be able to view any announcements posted.

Testing Description:
We verified the access control functionality by logging in as both an instructor and a student. As a faculty member assigned to a course, we tested the ability to add a new announcement and confirmed that it was successfully posted. We also checked that the faculty member could delete or edit the announcement, with a confirmation prompt appearing before deletion.

Next, we logged in using a student account added to the same course. We validated that the student could see the announcement but was unable to interact with any posting or deletion functionalities.

Test Owner: Aryan Bansal
Test Date: 01/04/2025
Test Results: The Course Announcement page functions as expected. Role-based access control is enforced correctly, and students are restricted to view-only access. Faculty members can post and delete announcements. No bugs were reported.

Additional Comment:
A notification email containing the announcement details can be sent to all users enrolled in the course upon creation of a new announcement. This feature was suggested to ensure timely communication.

14. Requirement: Once a quiz, assignment, lab test, project, or any evaluative event has been conducted and evaluated, instructors should be able to publish results on the **Result Page**. The functionality must allow instructors to:

- Add results by specifying the **Exam Name**, **Total Marks**, and **Weightage**.
- View a list of all enrolled students for that course during result entry.
- Manually input the **marks scored** by each student in the corresponding field.

During testing, we verified:

- The instructor could successfully **access the result page** and initiate result entry for a specific course event.
- Upon entering exam name, total marks, and weightage, a **list of students** was displayed with editable fields for entering marks.
- Manual entry of scores was working correctly and persisted after submission.
- Students could only **view** their results and were not able to modify them.
- We also ensured if a student is added to the course later after a quiz has taken place the marks scored by him are shown as **N/A** to both the student and instructor.

Test Owner: Aryan Bansal
Test Date: 02/04/2025
Test Results: Instructor was able to successfully enter and publish results. Data visibility and access were correctly enforced.

15. Requirement: Once results are published for an evaluative event (quiz, assignment, lab test, etc.), the Results Page should support the following features for faculty and students:

For Faculty:

- **View** marks scored by all students for any specific exam.
- **Download** the complete result set for offline use.
- **Edit/Update** marks if errors are identified post-publication.
- View a **histogram** that visualizes the distribution of marks for analytical insights.

For Students:

- Students can **only view their own marks** and not the marks of others.
- For each exam, students are also shown aggregate statistics : Maximum marks, Mean, Median, Standard deviation, Weightage & Total marks of the exam

Testing Details:

- Faculty successfully retrieved full student-wise marks for selected exams and verified data.
- Download option worked as expected; a structured xlsx was generated with appropriate data.

- Faculty were able to update marks for individual students; updates were correctly saved and reflected on subsequent views.
- Histogram of mark distribution loaded correctly.
- Students were restricted from accessing peer marks and only saw their own scores and the relevant statistics.
- All data was accurate, and no calculation or access issues were encountered.

Test Owner: Aryan Bansal
Test Date: 01/04/2025
Test Results: All functionalities worked as intended. Role-based data visibility was correctly enforced. Histogram and statistics were rendered accurately.

16. Requirement: The forum should allow authorized users (original poster or instructor) to delete their posts and replies. The functionality must enforce proper access control so that only the original poster or an instructor can remove content they have created.

During testing, we logged in as a student who created a forum post and, using the delete icon, removed the post after confirming the deletion via a prompt. In a separate test, we logged in as an instructor and deleted a reply made by a student. After each deletion, we verified that the corresponding content was completely removed from the forum thread and that no residual data was visible. We also confirmed that unauthorized users could not access the delete function on content they did not create.

Test Owner: Mayur Agrawal
Test Date: 02/04/2025
Test Results: Deletion operations for forum posts and replies were successful.
Additional Comments: NA

17. Requirement: Faculty should be able to upload and organize course materials into modules/sections and lectures on the Lectures page. This includes creating a module/section structure and adding lecture content (title, description, and file or link).

Description: On the EE201 course, `prof_smith` navigated to the **Lectures** page. We performed the following:

- Clicked "Add Module" and entered "Week 1" as the module name, then saved.

- Within "Week 1", clicked "Add Section" and named it "Introduction".

- Under that section, click "Add Lecture". In the add lecture form, entered Title: "Lecture 1: Basics", Description: "Introduction to Circuits", and uploaded a PDF file of lecture notes. Then click "Submit".

The expected outcome was that a new module "Week 1" with a section "Introduction" appears on the Lectures page, and inside it the lecture "Lecture 1: Basics" is listed with options to download the PDF.

Test Owner: Mayur Agrawal
Test Date: 02/04/2025
Test Results: The Lectures page updated immediately. "Week 1" module was created and could be expanded to show the "Introduction" section. Under Introduction, **Lecture 1: Basics** was listed. An icon for a PDF and a download link were present next to the lecture title. We verified the file by clicking download, which successfully downloaded the uploaded PDF. Students enrolled in EE201 were also able to see "Week 1 > Introduction > Lecture 1: Basics" on their lecture page (verified with a student login). The content addition feature worked flawlessly.
Additional Comments: NA

18. Requirement: The system should require essential fields (like lecture title) when adding lecture content, and it should not allow adding a lecture without them.

Description: We attempted to add another lecture in "Week 1" but left the **Lecture Title** field empty to test validation. A file was selected and description provided, then "Submit" was clicked. The expected behavior: the system refuses to add the lecture and indicates that the title is required.

Test Owner: Mayur Agrawal
Test Date: 02/04/2025
Test Results: The lecture was not added. The interface highlighted the Title field in red and displayed a small error message "Please enter a lecture title." We filled in the title and resubmitted, after which the lecture was added successfully. This confirmed that the application enforces required fields for lecture content (PASS).
Additional Comments: NA.

19. Requirement: Calendar updates and event notifications should synchronize in real time (within 1-2 minutes).
The tester updated events on the instructor's interface and verified that changes appeared on the student's calendar within 2 minutes. Manual refresh and automatic notification checks confirmed timely updates.
Test Owner: Mayur Agrawal
Test Date: 03/04/2025
Test Results: Calendar synchronization and notifications worked as required.
Additional Comments: NA

20. Requirement: The Contact Us page should allow both faculty and student users to submit queries and report issues.

During testing, we navigated to the Contact Us page as both a faculty member and a student. We verified that the user's name and email address were automatically populated. We then entered a valid subject and message in the designated fields. Upon clicking "Send Message," a confirmation message appeared, and we confirmed that the query was logged for support follow-up. Additionally, we checked that an email containing the submitted query was successfully sent to the support team for both user types.

Test Owner: Mayur Agrawal
Test Date: 03/04/2025
Test Results: Contact form submission worked correctly for both faculties and students; a confirmation was displayed and the email notification was successfully sent.
Additional Comments: NA

21. Requirement: Users should be able to change their password from the Profile section in a secure and user-friendly manner.

The system must ensure that:

- Users can access the change password form via the "Change Password" button on their Profile page.
- The user must enter their current password correctly.
- The new password and confirm password fields must exactly match.
- Passwords must meet strength criteria and be securely stored.
- Appropriate confirmation or error messages must be displayed depending on the outcome.

Testing Details:
We tested the password change functionality for both student and faculty users.

- Users successfully navigated to the Profile section and accessed the “Change Password” form.
- The current password was entered accurately, followed by matching entries in the new password and confirm password fields.
- Upon submission, the system validated the inputs and displayed a success message ("Password changed successfully").
- We verified that users could log in using the new password immediately after the change, and that the old password was no longer valid.
- Additional tests were conducted with incorrect current passwords and mismatched new password entries. The system blocked the update in these cases and displayed appropriate error messages: "Current password is incorrect" and "New passwords do not match".
- All validation logic worked correctly and prevented unauthorized or erroneous changes.

Test Owner: Mayur Agrawal
Test Date: 04/04/2025
Test Results: Password change functionality worked as expected for both students and faculty. All validations, success responses, and error messages were accurate and securely enforced.

21. Requirement: The Profile section should display accurate user details including name, roll number, and department.
We logged in as a user for different users, navigated to the Profile section, and confirmed that all personal details were correctly populated from the database. The interface allowed for clear viewing of information.
Test Owner: Mayur Agrawal
Test Date: 04/04/2025
Test Results: User profile details were displayed accurately with no discrepancies.
Additional Comments: NA

22. Requirement: Students should be able to view the course calendar with all scheduled events.
On the course home page, we verified that the calendar displayed a merged view of all events. Navigation controls (day, week, month views) were tested and events appeared with correct details (title, date, course code).
Test Owner: Mayur Agrawal
Test Date: 03/04/2025

Test Results: The calendar view was fully functional and events were displayed as expected.
Additional Comments: NA

23. Requirement: Students should be able to view all course announcements on the course home page.
We logged in as a student and verified that all announcements (posted by the instructor) for that course were visible, sorted in reverse chronological order. Each announcement could be expanded to view details and was accompanied by the correct posting date and author information.
Test Owner: Mayur Agrawal
Test Date: 03/04/2025
Test Results: Announcements were displayed accurately and in the correct order.
Additional Comments: NA

24. Requirement: Students should be able to access all lecture modules, sections, and materials that instructors have uploaded, and download resources as needed.

Description: We navigated to the **Lectures** page of EE201 as a student. We expected to see the structured content that the faculty added, organized by week/module. Specifically, "Week 1" module with the "Introduction" section and inside it "Lecture 1: Basics" (or its updated title). We attempted to open the lecture content and download the attached PDF to ensure the student has access. We also tested that lectures from other weeks (if any) can be expanded or collapsed.

Test Results: We accessed the Lectures page for EE201 from a student account and observed that "Week 1" was displayed with an expandable arrow. Upon expanding it, the "Introduction" section became visible, containing **Lecture 1: Circuit Basics**. We clicked on the lecture title, and the uploaded PDF opened directly in the browser. We also tested the download functionality by clicking the download icon, which successfully downloaded the file. The weekly sections were collapsible, allowing us to hide or display content as needed. As students, we only had read-only access—no edit or delete options were available, as expected. This confirms that students can effectively access and retrieve lecture materials as intended.

Test Owner: Mayur Agrawal
Test Date: 03/04/2025
Test Results: Lectures were displayed and accessible correctly; collapsible sections functioned properly.
Additional Comments: None.

25. Requirement: The Performance section should provide students with a consolidated overview of their academic standing across all enrolled courses. It must include an overall summary of performance, as well as course-wise breakdowns with detailed metrics and visual analytics.

Testing Details:

We accessed the **Performance** tab from a student account enrolled in multiple courses. At the top of the page, we verified the summary dashboard which displayed:

- Total number of currently enrolled courses (e.g., 5).
- Weighted average performance percentage across all courses (e.g., 68.5%).
- Relative-to-median metric indicating how the student is performing compared to peers (e.g., 12.3% below median).

Below the summary, we tested the **Course-wise Breakdown** section. Each active course was shown as a collapsible card displaying the course title, credits, and a dropdown arrow. On clicking the arrow for a course (e.g., *CS253: Software Development*), a detailed view appeared containing:

- A **bar graph** comparing obtained marks (in teal) and weightage (in purple) for each evaluation component such as quizzes, assignments, and midterms.
- A **tabular breakdown** listing the evaluation name, weightage, total marks, obtained marks, mean, median, highest score (max), and standard deviation.

We verified the accuracy of these statistics against known result data. The graph visually reflected performance trends across evaluations, and the table allowed clear comparison. The visualization accurately highlighted performance gaps and strengths.

Additionally, we ensured that:

- The breakdown was only visible to the logged-in student and did not expose any peer information.
- The student could access only their own scores and the aggregate statistics (mean, median, max, deviation), with no edit options available.
- Expanding and collapsing course cards worked as expected without interface glitches.

All metrics, graphs, and interactions functioned smoothly, and data presented was consistent with the results published for that student.

Test Owner: Mayur Agrawal
Test Date: 04/04/2025
Test Results: The Performance section worked as intended. Summary statistics, course-wise breakdowns, and analytics were accurately presented and securely restricted to the student's own data.
Additional Comments: NA

26. Requirement: The Forum section should allow students to initiate, view, and participate in course-related discussions. They should be able to post new queries, view threads, reply to posts, and delete their own content.

Testing Details:
We tested the Forum section functionality from a student account across the following use cases:

- **Creating a New Post:**
  We navigated to the Forum tab within a course (e.g., EE210) and clicked the **"Add Post"** button. The interface expanded to reveal a text input field with the placeholder "Your Query." We entered a question: *"When is the deadline for assignment 1?"* and clicked **"Submit."** The post appeared instantly in the forum thread list, visible to all course participants.

- **Cancelling a Draft Post:**
  We initiated a new post and clicked **"Cancel"** instead of submitting. The input field collapsed, and the draft was discarded as expected, confirming that users can back out of creating a post without errors.

- **Viewing and Replying to Posts:**
  As a student, we opened an existing post and verified the following elements:

  - Author name (e.g., *student5*)
  - Timestamp of post creation
  - Post content
  - A red trash bin icon, visible only to the post's author, confirming access control

    Below the post, a **Replies** section was visible. We clicked the **"Reply"** button, typed a response (*"It's on 4th April."*), and submitted it. The reply appeared immediately below the original post in a threaded format, with

correct user info, timestamp, and a delete icon visible only to the reply's author.

- **Deleting Own Posts and Replies:**
  We tested deleting a post created earlier. The red trash bin icon was visible next to our post, and clicking it triggered a confirmation prompt. After confirming, the post was removed from the thread. We performed a similar test for replies: the delete icon next to our reply allowed us to delete the comment, which was removed instantly upon confirmation. In both cases, deletion was restricted only to the content's original author.

- **Thread Structure & Layout:**
  All posts and replies were displayed in a neatly threaded, hierarchical format. Usernames and timestamps were consistently visible, and there was no access to modify or delete others' content. The reply input box was correctly located beneath each post.

Additional Validation:

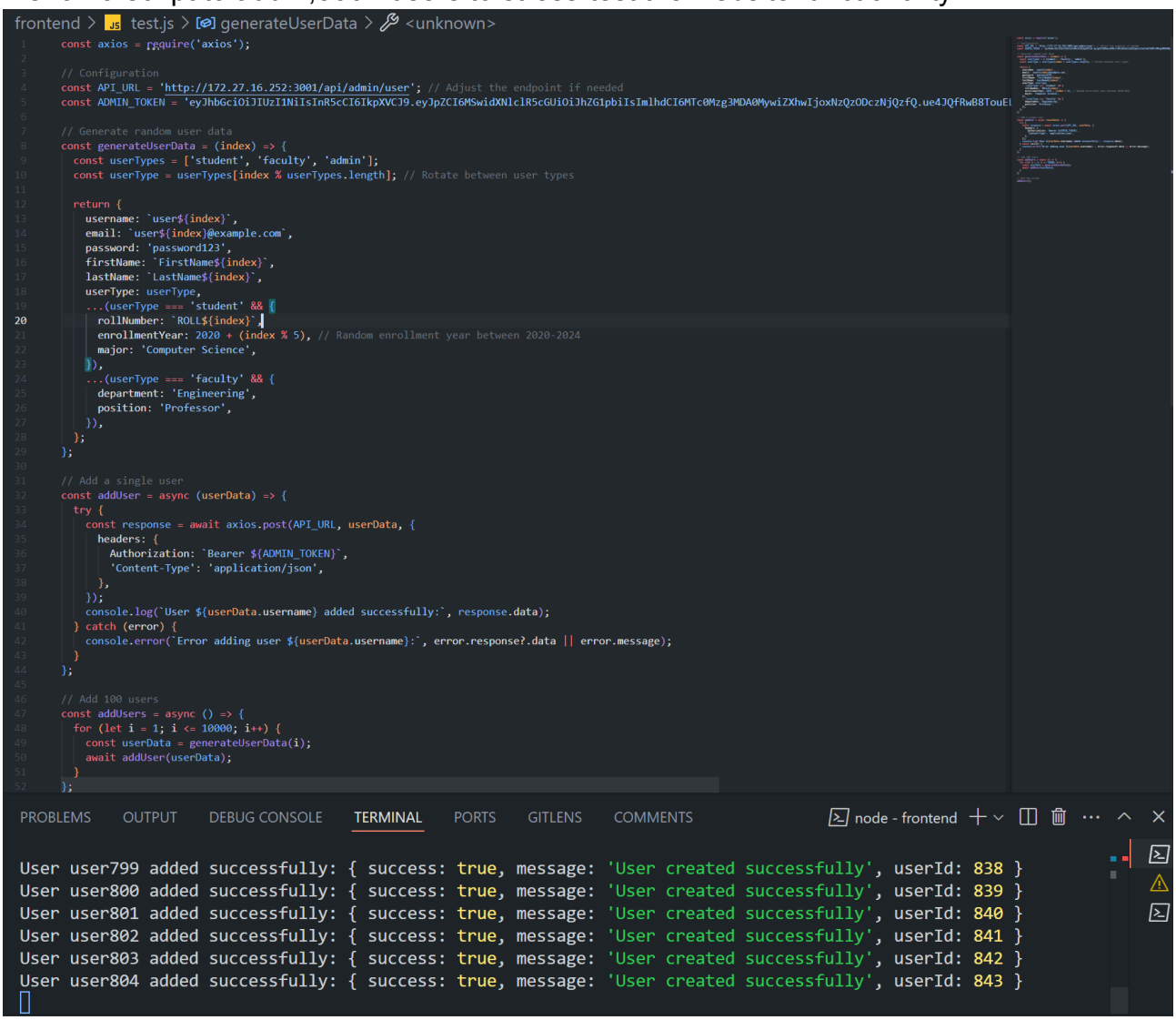
- Posts were not visible across unrelated courses.
- UI responded correctly without page reloads.
- Role-based permissions were strictly enforced.

Test Owner: Mayur Agrawal
Test Date: 04/04/2025
Test Results: All functionalities—post creation, reply, delete, and view—worked as expected. Proper access control and formatting were enforced, ensuring a smooth and collaborative discussion experience for students.
Additional Comments: NA

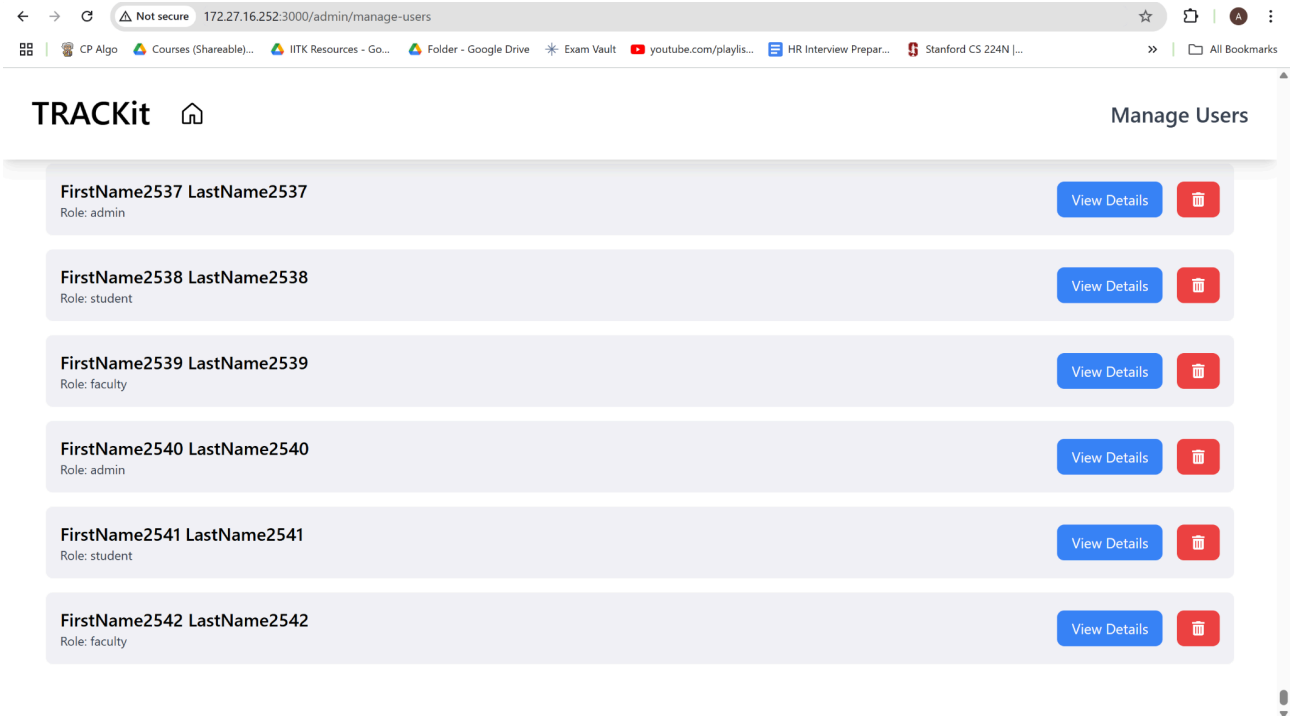
Testing the non-functional requirements

Scalability Testing:

1. **Test to add 2,500+ users in the server:**
   We ran a script to add 2,500+ users to stress-test the website functionality.

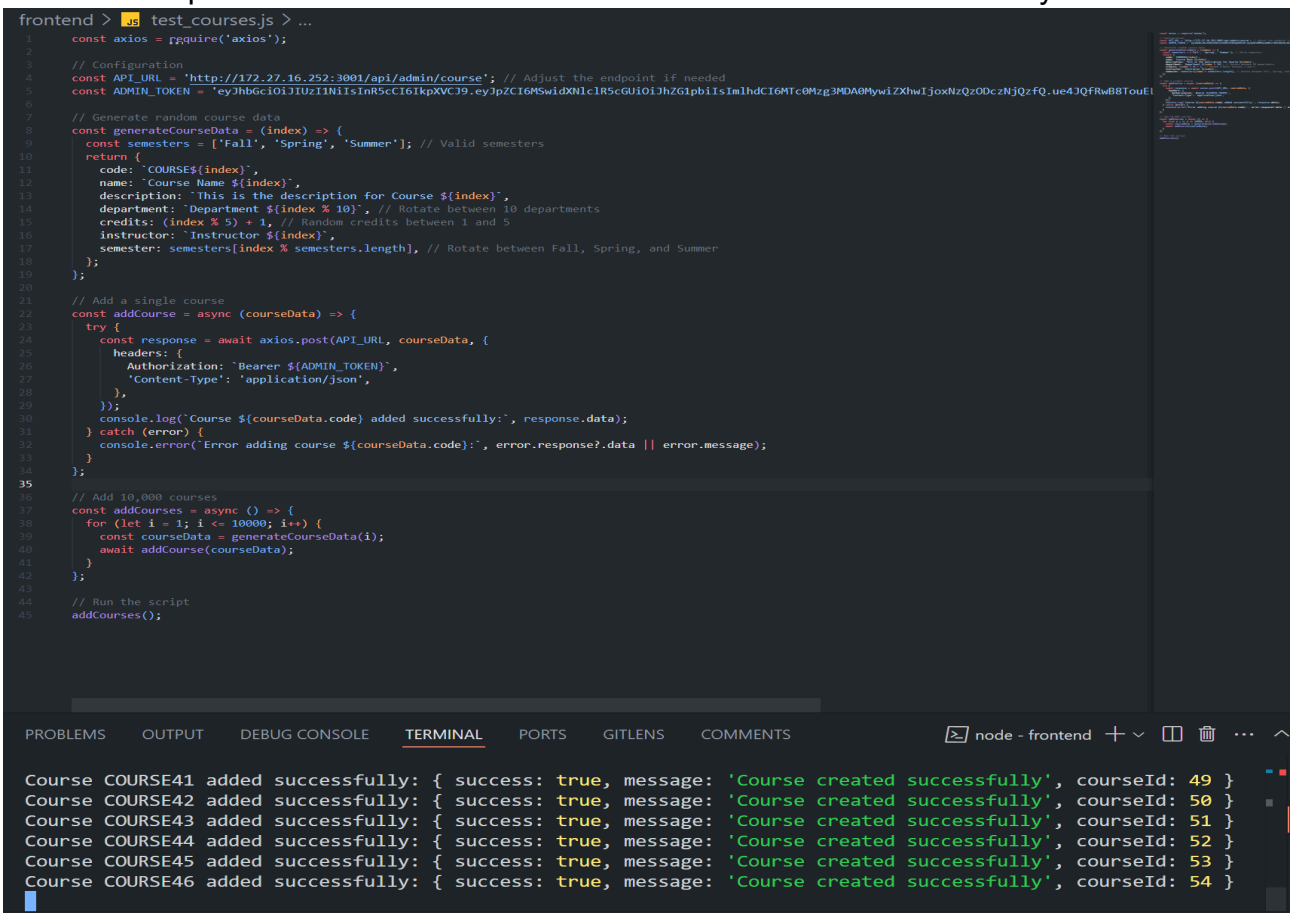


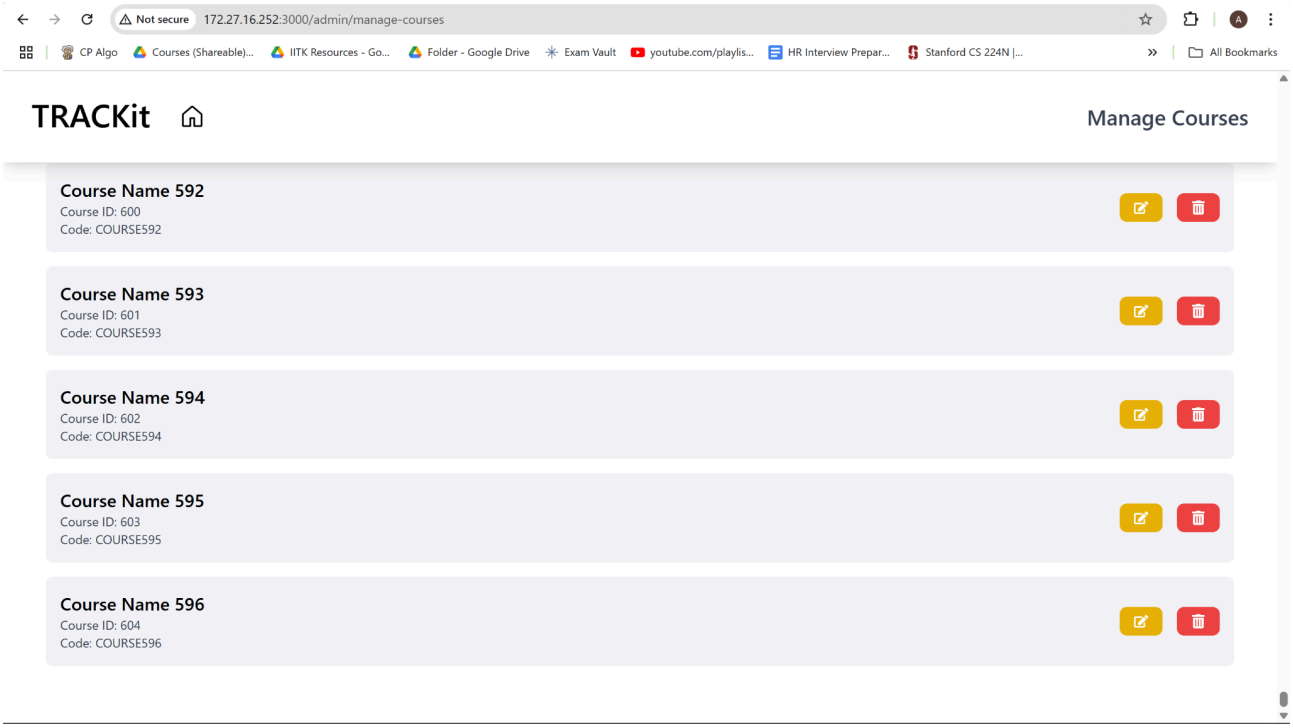
The website was fully functional after such a high student load.



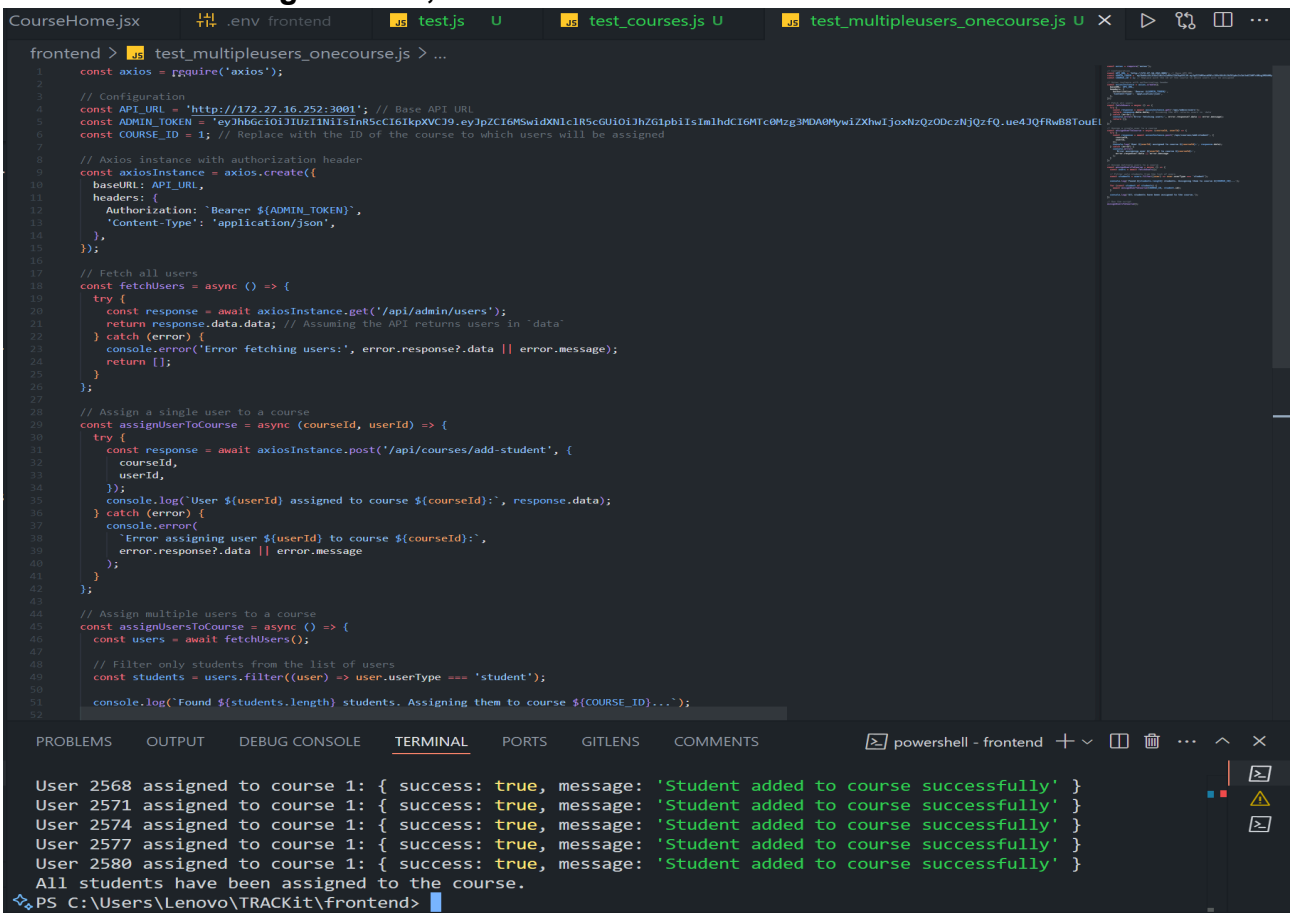
2. Test to add 600+ courses:

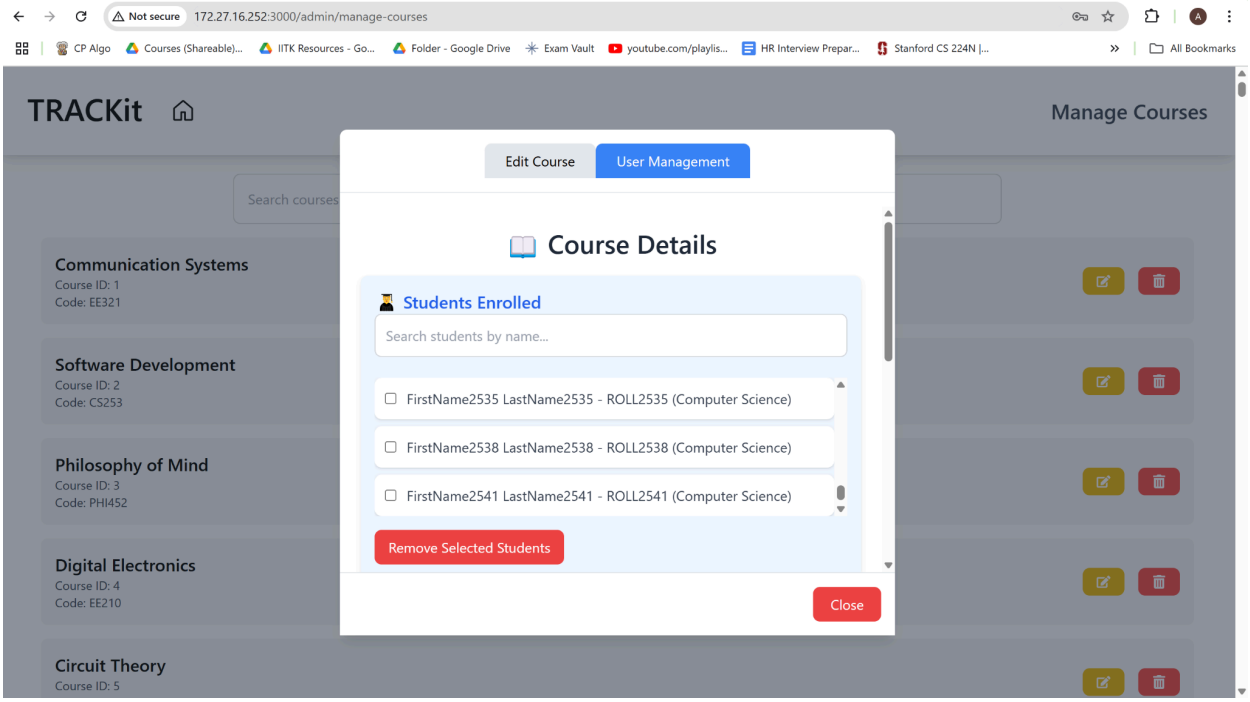
We ran a script to add 600+ courses to stress-test the website functionality.





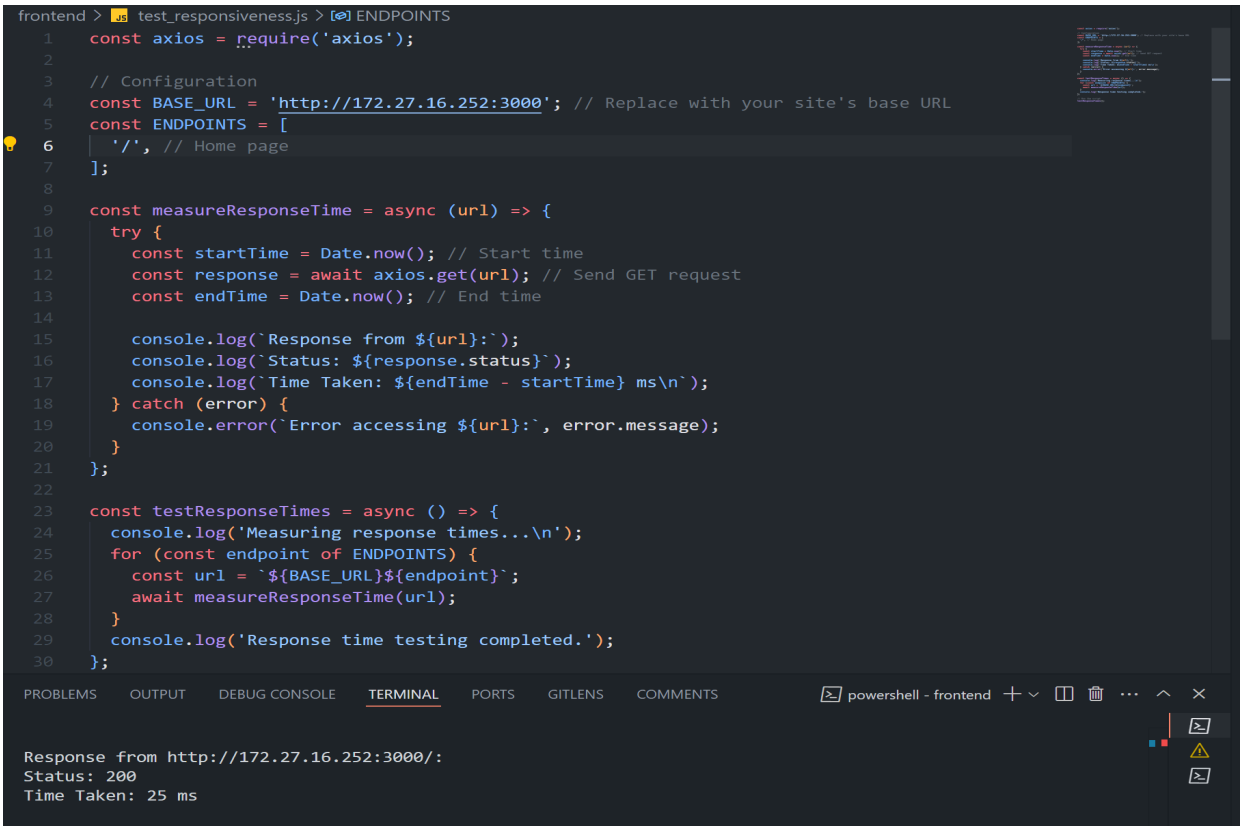
3. Now we test adding all the 2,500 users in one course




Not secure
172.27.16.252:3000/admin/manage-courses
TRACKit
Manage Courses
Search courses
Communication Systems
Course ID: 1
Code: EE321
Software Development
Course ID: 2
Code: CS253
Philosophy of Mind
Course ID: 3
Code: PHI452
Digital Electronics
Course ID: 4
Code: EE210
Circuit Theory
Course ID: 5
Edit Course
User Management
Course Details
Students Enrolled
Search students by name...
FirstName2535 LastName2535 - ROLL2535 (Computer Science)
FirstName2538 LastName2538 - ROLL2538 (Computer Science)
FirstName2541 LastName2541 - ROLL2541 (Computer Science)
Remove Selected Students
Close

Performance testing:

All APIs take less than 50ms on average, and loading the page for the first time takes about 25 ms.



Maintainability testing:

The software has been designed in an organized manner so that new features can be added and modifications can be done very easily within two working days. Various components of the software are well organised into folders and comments are added wherever required streamline future improvements

Reliability testing:

The system operates with remarkable robustness, consistently delivering reliable performance. The anticipated downtime of the website is significantly minimized, ensuring uninterrupted accessibility for users. This reliability enhances the overall user experience and fosters trust in the website's functionality.

Portability testing:

Being a web-based app, the software is easily accessible from any device with browser support. We tested the deployed website, and it works perfectly fine on Google Chrome, Mozilla Firefox and Microsoft Edge web browsers.

5 Conclusion

- **How Effective and exhaustive was the testing?**

  Testing was done almost exhaustively. Each API underwent testing to ensure that they were modifying the database and returning the desired information when needed. We have taken care of all cases, thus ensuring complete branch coverage. The frontend was also tested and bugs were resolved. Validity of each input has been checked in all forms. Appropriate alerts have been shown in such cases. Edge cases were also tested and were confirmed to work as expected. For example, when faculty tried to enter marks more than total marks of the exam, the system correctly prevented him from doing so.
  The testing was quite effective. Each of the modules has been tested by at least 2 members ensuring double check for every sub-system. Testing helps us to find many small bugs mostly at the time of integration testing and we tried to resolve most of them, however some are still pending and will be resolved soon.

- **Which components have not been tested adequately?**

  Our test coverage for non functional requirements outlined in the SRS document could be further expanded. Tests for non-functional requirements like reliability, maintainability and ease of learning could not be performed due to time constraint. Integration testing almost tries to check every interaction between the units but due to manual nature of testing some cases might got ignored so it can also be improved further.

- **What difficulties have we faced during testing?**

  The main difficulty faced during testing was ensuring non-functional requirements are satisfied, in system testing. Apart from this we also feel that testing every api manually for each possible scenario is quite time consuming and we could have same some time on that if we use automatic testing

- **How could the testing process be improved?**

  The testing could have been improved by doing the unit testing or integration testing as automated rather than manual. The developer might subconsciously assume a few crucial things which may be not so obvious to an end-user or someone who was not involved in the development of the software.

Appendix A - Group Log

"The following are the minutes of the collective group meeting held during the document submission. In addition to these discussions, each team member has contributed significantly through individual efforts that may not be fully captured in this documentation. The entire group sincerely appreciates and acknowledges each other's hard work and dedication."

Date	Timing	Duration	Minutes
01/04/25	14:00-16:00	2hrs .	● Discussed Various Types Of testing. ● Distributed Tasks Of Various types of testing among subgroups/individuals within the team. ● Divided work of making user manual
02/04/25	17:00-21:00	4 hrs.	● Discussed Various Doubts faced by each subgroup/individuals in their respective tasks.
03/04/25	22:00-02:00	4 hrs.	● Every testing unit updated and organised their work in doc and reported bugs to respective developer
04/04/25	16:00-18:00	2 hrs.	● Resolved bugs , reconducted the testing and updated the docs ● Revised the structure and content of the user manual.
05/04/25	20:00-23:00	3 hrs.	● Finalised the test document and User Manual