Test Document

for

Academics Management Software

Version 1.0

Prepared by

Group #: 17 **Group Name:** same_storm

Achint Agrawal	180028	achintag@iitk.ac.in
Somya Lohani	190848	somyaloh@iitk.ac.in
Sachin	180639	thakan@iitk.ac.in
Udhav Gupta	180829	udhavgup@iitk.ac.in
Yash Gupta	190998	gyash@iitk.ac.in
Himanshu Sood	190381	himsood@iitk.ac.in
Piyush Agarwal	190600	piyuagr@iitk.ac.in
Aditya Ranjan	190068	aranjan@iitk.ac.in
Yash Gupta	190997	yashbg@iitk.ac.in
Utkarsh Jain	190928	utkjain@iitk.ac.in

Course:	CS253
Mentor TA:	*Ms. Shatroopa Saxena*
Date:	3 April 2022

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Initial Draft (v1)	Yash Gupta (190998), Yash	Built a web application that will allow a user to register himself/herself into the application to see	04/04/2022

Version	Primary Author(s)	Description of Version	Date Completed
	Gutpa (190997), Aditya Ranjan, Sachin, Somya Lohani, Udhav Gupta	course reviews of various courses and also to plan future semester by selecting various course and import course timings into the calendar UI	

1 Introduction

Mention the test strategy
(Manual testing? Test automation? …)
Our testing consisted of both automated and manual testing. We tried using automated testing as much as we could but for some set of APIs user interaction was needed which was a bit complex to automate, and thus for these sets of APIs we used manual testing.

When was the testing conducted?
(in parallel with the implementation? After the implementation was completed? …)
The testing of the application has been done both during the implementation and after the completion of the application. During the implementation, we mostly carried out manual testing using the Postman software which basically provides us a client to test our APIs. So as to ensure that the logic used in the implementation of the application is correct, we tried testing the APIs on various edge cases and thus improved the code accordingly.
After the completion of the implementation, we started making use of the Jest package which automated our tests. During this time, we collaboratively wrote unit tests for the various APIs and suggested the needed changes to the developers of those APIs.

Who were the testers?
(the developers? Independent testers? …)
In this project, the team members were both developers and testers of the application. We tested our own APIs as well as those of other developers by reviewing their PRs.

What coverage criteria were used?
While carrying out the automated testing we used the in-built functionality of the Jest package in NodeJS to check the coverage percentage of the test cases. For the manual testing part we manually checked the number of functions getting invoked by the execution of those test cases.

Have you used any tool for testing?

We have used the JavaScript package Jest which is primarily used for carrying out automated testing of functions and modules. Also we have used Postman for manually testing the APIs.

2 Unit Testing

1. User signup

Unit Details: *Unit for a user signup - [registerUser() and addUser()]*
Test Owner: *Yash Gupta (190998)*
Test Date: 03/04/2022
Test Results: A new user could sign up successfully for and user details got saved in the database.
Structural Coverage: Gave request containing details of a new user and then a new user got created in the database

2. User signup

Unit Details: *Unit for a user signup - [registerUser() and findUser()]*
Test Owner: *Yash Gupta (190998)*
Test Date: 03/04/2022
Test Results: An existing user could not sign up because *findUser()* returned a non-null value.
Structural Coverage: Gave a request containing details of an existing user

3. User login

Unit Details: *Unit for user login - [loginUser() and findUser()]*
Test Owner: *Yash Gupta (190998)*
Test Date: 03/04/2022
Test Results: On different sets of inputs (requests), we got the desired result in the response body.
Structural Coverage:

- Tried giving valid credentials of an existing user (user logged in and a JWT token got generated)
- Tried giving invalid credentials of both an existing and non-existing user (user couldn't log in)

4. User logout

Unit Details: *Unit for user logout - [routing statements in credRouter.js]*
Test Owner: *Yash Gupta (190998)*
Test Date: 03/04/2022
Test Results: For different users logged in, users could logout from the application successfully.
Structural Coverage:

- Tried running the DELETE endpoint when a user was logged in (and the token wasn't expired) (got the cookies cleared and user logged out successfully)
- Tried running the DELETE endpoint when no user was logged in (got an authorization failed message)

5. Get Course Details

Unit Details: *Unit for getting course details - [apiGetCourseDetails(), getCourseDetails() and getReviewContent()]*
Test Owner: *Yash Gupta (190998)*
Test Date: 03/04/2022
Test Results: For different course IDs in the parameters of the URL, we got the corresponding course details in the response body, along with a list of reviews given by the users.
Structural Coverage:

- Tried running the GET endpoint with an existing course (got the course details and a list of reviews, if there was any)
- Tried running the GET endpoint with a non-existing course (got a message saying the course doesn't exist in the database)

6. Get Review Details

Unit Details: *Unit for getting a review details - [apiGetReviewDetails(), getReviewDetails() and getCommentContent()]*
Test Owner: *Yash Gupta (190998)*
Test Date: 03/04/2022
Test Results: For different review IDs in the parameters of the URL, we got the corresponding review details in the response body, along with a list of comments given by the users.
Structural Coverage:

- Tried running the GET endpoint with an existing review ID (got the correct review details)
- Tried running the GET endpoint with a non-existing review ID (got a message saying no such review exists)

7. Get Favorite Courses

Unit Details: *Unit for getting favorite courses of a user - [apiGetFavoriteCourses() and getFavoriteCourses()]*
Test Owner: *Yash Gupta (190998)*
Test Date: 03/04/2022

Test Results: For different user logins, the test gave a different and valid list of favorite courses.

Structural Coverage:

- Tried running the GET endpoint for an existing user and got a valid list of favorite courses
- Tried running the API call for a non-existing user and got an message of user not found.

8. Add Review to a course

Unit Details: *Unit for adding a review to a course - [apiAddReview(), addReview() and updateCourseRating()]*
Test Owner: *Yash Gupta (190998)*
Test Date: 03/04/2022
Test Results: Users could add the reviews to any course available in the database and also update the rating of that course
Structural Coverage:

- Tried adding a review for the first time of a course (got the review added into the list of reviews of a course and rating got updated accordingly)
- Tried adding a review when a review already exists (review got updated and rating got updated too)

9. Add Comment to a course review

Unit Details: *Unit for adding a comment to a course review - [apiAddCommetToReview(), addCommentToReview() and addComment()]*
Test Owner: *Yash Gupta (190998)*
Test Date: 03/04/2022
Test Results: Users could add comments to a course review
Structural Coverage:

- Tried adding a comment to course review with a valid reviewID (got it successfully added)
- Tried adding a comment to a course review already having a comment from that user (edited the already present comment)
- Tried adding a comment to a course review with an invalid reviewID (got an error message saying no such review exists)

10. Add Comment to a course review

Unit Details: *Unit for adding a comment to a course review - [apiAddCommetToReview(), addCommentToReview() and addComment()]*
Test Owner: *Yash Gupta (190998)*

Test Date: 03/04/2022
Test Results: Users could add comments to a course review
Structural Coverage:

- Tried adding a comment to course review with a valid reviewID (got it successfully added)
- Tried adding a comment to a course review already having a comment from that user (edited the already present comment)
- Tried adding a comment to a course review with an invalid reviewID (got an error message saying no such review exists)

11. Add course in course planner

Unit Details: *Unit for adding a course in course planner [apiAddCourse(), addCourse()]*
Test Owner: *Yash Gupta (190997)*
Test Date: 03/04/2022
Test Results: A course could be added successfully in the given semester.
Structural Coverage:

- Tried adding a course in a semester that exists
- Tried adding a course in a semester that does not exist
- Tried adding a course which already exists in the given semester
  Got correct results for all the cases.

12. Delete course from course planner

Unit Details: *Unit for deleting a course from course planner [apiDeleteCourse(), deleteCourse()]*
Test Owner: *Yash Gupta (190997)*
Test Date: 03/04/2022
Test Results: A course could be deleted successfully in the given semester.
Structural Coverage:

- Tried deleting a course from a semester containing that course
- Tried deleting a course from a semester that does not contain that course
- Tried deleting a course from a semester that does not exist
  Got correct results for all the cases.

13. Add a new semester in course planner after a given semester

Unit Details: *Unit for adding a new semester in course planner after a given semester [apiAddSemester(), addSemester()]*
Test Owner: *Yash Gupta (190997)*
Test Date: 03/04/2022
Test Results: A new semester could be added successfully after the given semester.
Structural Coverage:

- Tried adding a new semester after an existing semester
- Tried adding a new semester after a semester that does not exist

  Got correct results for all the cases.

14. Add a new semester at the end in course planner

Unit Details: *Unit for adding a new semester at the end in course planner [apiAddSemesterAtEnd(), addSemesterAtEnd()]*
Test Owner: *Yash Gupta (190997)*
Test Date: 03/04/2022
Test Results: A new semester could be added successfully at the end.
Structural Coverage: Tried adding a new semester at the end and got the correct result.

15. Delete a semester from course planner

Unit Details: *Unit for deleting a semester from course planner [apiDeleteSemester(), deleteSemester()]*
Test Owner: *Yash Gupta (190997)*
Test Date: 03/04/2022
Test Results: A semester could be deleted successfully.
Structural Coverage:

- Tried deleting an existing semester
- Tried deleting a semester that does not exist

  Got correct results for all the cases.

16. Get number of existing semesters in course planner

Unit Details: *Unit for getting number of existing semesters in course planner [apiGetNumberOfSemesters(), getNumberOfSemesters()]*
Test Owner: *Yash Gupta (190997)*
Test Date: 03/04/2022
Test Results: Number of existing semesters could be found successfully.
Structural Coverage: Tried getting the number of existing semesters and got the correct result.

17. Get the contents of a semester from course planner

Unit Details: *Unit for getting the contents of a semester from course planner [apiGetSemester(), getSemester()]*
Test Owner: *Yash Gupta (190997)*
Test Date: 03/04/2022
Test Results: The contents of a semester could be found successfully.
Structural Coverage:

- Tried getting the contents of an existing semester
- Tried getting the contents of a semester that does not exist
  Got correct results for all the cases.

18. *Get the contents of a course in course planner*

Unit Details: *Unit for getting the contents of a course in course planner [apiGetCourse(), getCourse()]*
Test Owner: *Yash Gupta (190997)*
Test Date: 03/04/2022
Test Results: The contents of a course in the given semester could be found successfully.
Structural Coverage:

- Tried getting the contents of a course in a semester containing that course
- Tried getting the contents of a course in a semester that does not contain the couse
- Tried getting the contents of a course that does not exist
  Got correct results for all the cases.

19. *Mark a course as a favourite course*

Unit Details: *Unit for adding a course as a favourite course given course ID [apiAddToFavourites(), addToFavourites()]*
Test Owner: *Aditya Ranjan (190068)*
Test Date: 04/04/2022
Test Results: A course could be successfully added as a favourite course given its ID
Structural Coverage:

- Tried adding a non-favourite valid course as a favourite course
- Tried adding a favourite valid course as a favourite course
- Tried adding an invalid course as a favourite course
  Got correct results for all cases.

20. *Liking a course review*

Unit Details: *Unit for giving a like to a course review given its review ID [apiLikeReview(), likeReview()]*
Test Owner: *Aditya Ranjan (190068)*
Test Date: 04/04/2022
Test Results: The review's like counter could be successfully incremented given its review ID
Structural Coverage:

- Tried liking an existing course review with a valid review ID
- Tried liking a non-existing course review with a valid review ID
- Tried liking a non-existing course review with an invalid review ID
  Got correct results for all cases.

21. Liking a comment on a course review

Unit Details: *Unit for liking a comment on a course review given the comment ID ("<review-author-username>-<course ID>-<comment-author-username>") [apiLikeComment(), likeComment()]*
Test Owner: *Aditya Ranjan (190068)*
Test Date: 04/04/2022
Test Results: The comment's like counter could be successfully incremented given its comment ID
Structural Coverage:

- Tried liking an existing comment with a valid comment ID
- Tried liking a non-existing comment with a valid comment ID
- Tried adding a non-existing comment with an invalid comment ID
  Got correct results for all cases.

22. Create a note for a course

Unit Details: *Unit for creating a note on a course given its course ID and note title, content and tags [apiCreateNote, insertNote]*
Test Owner: *Aditya Ranjan (190068)*
Test Date: 04/04/2022
Test Results: The note could be added successfully for the course
Structural Coverage:

- Tried adding a note to an existing course with a non-empty title
- Tried addind a note to an existing course with an empty title
- Tried adding a note to a non-existing course with a non-empty title
- Tried adding a note to a non-existing course with an empty title
  Got correct results for all cases

23. Delete a note from course

Unit Details: *Unit for deleting a note on a course given its note ID [apiDeleteNote, deleteNote]*
Test Owner: *Aditya Ranjan (190068)*
Test Date: 04/04/2022
Test Results: The note could be deleted successfully from the course
Structural Coverage:

- Tried deleting an existing note with a valid ID
- Tried deleting a non-existing note with an invalid ID
  Got correct results for all cases.

24. Update a course note

Unit Details: *Unit for updating a course note given its note ID, contents, title and tags [apiUpdateNote, updateNote]*
Test Owner: *Aditya Ranjan (190068)*
Test Date: 04/04/2022
Test Results: The course note could be updated successfully
Structural Coverage:

- Tried updating an existing note with valid note ID and non-empty title
- Tried updating an existing note with valid note ID and empty title
- Tried updating a non-existing note with invalid ID and non-empty title
- Tried updating a non-existing note with invalid note ID and empty title
- Tried updating a note via note ID which belongs to another user
  Got correct results in all cases

25. Fetching all Course Notes

Unit Details: *Unit for fetching all course notes corresponding to a course given the course ID [apiFetchNotes, fetchNotes]*
Test Owner: *Aditya Ranjan (190068)*
Test Date: 04/04/2022
Test Results: All notes belonging to the course could be fetched successfully
Structural Coverage:

- Tried fetching all course notes corresponding to a valid course ID
- Tried fetching all course notes corresponding to an invalid course ID
  Got correct results in all cases

3 Integration Testing

1. Checking the interface between the signup and the login module

Module Details: signup and login modules tested
Test Owner: *Yash Gupta (190998)*
Test Date: 04/04/2022
Test Results: On successfully signing up in the application using the valid credentials, the user gets redirected to the login page. If the user already exists then an alert message shows up and then the window gets redirected to the login page.

2. Checking the interface between the login module and the showProfile module

Module Details: login and showProfile modules tested
Test Owner: *Yash Gupta (190998)*
Test Date: 04/04/2022
Test Results: On entering the correct credentials in the login form, the user is redirected to the profile page showing the user specific details.

3. User authentication fails on entering incorrect credentials

Module Details: login and UI framework tested
Test Owner: *Yash Gupta (190998)*
Test Date: 04/04/2022
Test Results: On entering wrong credentials in the login form, user gets an alert message of incorrect credentials, and thus is not allowed to access the authorization protected endpoints.

4. Checking the effect of logout on the application

Module Details: logout module tested
Test Owner: *Yash Gupta (190998)*
Test Date: 04/04/2022
Test Results: On logging out from the application, the user gets redirected to the login page and the associated cookies get cleared so as not to authorize the user anymore.

5. Already existing users cannot register i.e., already existing username/rollno cannot be used while signing up

Module Details: signup and findUser modules tested
Test Owner: *Yash Gupta (190998)*
Test Date: 04/04/2022
Test Results: On using a username or rollno that already exists in the database, the user gets an alert message showing that a user with the same username or rollno already exists.

6. All endpoints except the home, signup and login page are accessible only when a user is authorized

Module Details: authorization middleware along with all other modules tested

Test Owner: *Yash Gupta (190998)*

Test Date: 04/04/2022

Test Results: To access the GET and POST endpoints except the one for signup and login can only be accessed when a user is logged into the system i.e., have an active session.

7. User can see the course details by giving a valid course ID in the getCourseDetails endpoint

Module Details: getCourseDetails and UI framework tested

Test Owner: *Yash Gupta (190998)*

Test Date: 04/04/2022

Test Results: On giving a valid course ID a user is able to see details specific to a course including the reviews given to that course by the other users.

8. After adding a review to a course, other users can see the newly added review

Module Details: addReview and getCourseDetails modules tested

Test Owner: *Yash Gupta (190998)*

Test Date: 04/04/2022

Test Results: On adding a review to a course, it gets saved to the database through which other users can access the review with the help of getCourseDetails module.

9. After giving a rating to a course, other users can see the modified rating along with other course details

Module Details: giveRating and getCourseDetails modules tested

Test Owner: *Yash Gupta (190998)*

Test Date: 04/04/2022

Test Results: On giving some non-zero (negative not allowed) rating to a course, other users can also see the modified rating on their respective sessions after refreshing the getCourseDetails page.

10. After adding a comment to a course review, the list of comments to that course review gets changed for other users as well

Module Details: addComment and getReviewDetails modules tested

Test Owner: *Yash Gupta (190998)*

Test Date: 04/04/2022

Test Results: On adding a comment to a course review, the list of comments of a course review gets updated in the database and eventually on other users’ screens as well.

11. Liking a course review also changes the number of likes of a course review shown on other users' screens

Module Details: likeReview and getReviewDetails modules tested

Test Owner: *Yash Gupta (190998)*

Test Date: 04/04/2022

Test Results: On liking a course review, the count of likes given to a course review gets changed for other users as well i.e., the count of likes gets changed in the database for that particular course.

12. Liking a course comment also changes the number of likes of a comment shown on other users' screens

Module Details: likeComment and getReviewDetails modules tested

Test Owner: *Yash Gupta (190998)*

Test Date: 04/04/2022

Test Results: On liking a comment given by a user to a review, the count of likes gets increased globally i.e., other users can also see the increased number of likes to that comment.

13. Marking a course favorite modified the number of stars for a course on other users' screens

Module Details: addToFavorite and getCourseDetails modules tested

Test Owner: *Yash Gupta (190998)*

Test Date: 04/04/2022

Test Results: On marking a course as favorite, the number of stars of that course changes globally i.e., the current user and other users as well can see the increased number of stars for that course.

14. Checking if a newly added note can be viewed in the browse notes section

Module Details: *insertNote, fetchNotes*

Test Owner: *Aditya Ranjan (190068)*

Test Date: *04/04/2022*

Test Results: *After adding a note by entering details and clicking Submit in the Add Notes section, clicking on Notes → Browse Notes shows all notes added by the user including the newly added note*

15. Checking if a newly deleted note is removed from the browse notes section

Module Details: *deleteNote, fetchNotes*

Test Owner: *Aditya Ranjan (190068)*

Test Date: *04/04/2022*

Test Results: *After deleting a note by clicking on the delete note button, clicking on Notes → Browse Notes shows all notes added by the user but not the newly deleted note which was previously visible.*

16. Checking if a newly updated note can be accessed

Module Details: *updateNote, fetchNotes*

Test Owner: *Aditya Ranjan (190068)*

Test Date: *04/04/2022*

Test Results: *After updating a note and submitting the updated note, clicking on Note → Browse Notes shows all notes added by the user including the newly updated note.*

17. All the added courses with their details in all existing semesters are visible in the frontend in the course planner

Module Details: *getNumberOfSemesters, getSemester, getCourse*

Test Owner: *Yash Gupta (190997)*

Test Date: *04/04/2022*

Test Results: *The frontend shows all the added courses with their details in all existing semesters*

18. New semesters and courses are visible in the frontend when new semesters and courses are added

Module Details: *addSemester, addSemesterAtEnd, addCourse*

Test Owner: *Yash Gupta (190997)*

Test Date: *04/04/2022*

Test Results:

- *Whenever a new semester is added in the backend, a new column shows up in the course planner in the frontend*
- *Whenever a new course is added in an existing semester in the backend, it is visible in that semester in the frontend*

19. Deleting a semester deletes all its courses in the course planner

Module Details: *deleteSemester*

Test Owner: *Aditya Ranjan (190068)*

Test Date: *04/04/2022*

Test Results: *Deleting a semester removes an entire column from the plan table, frontend as well as backend*

20. Courses are removed from the frontend when deleted from an existing semester

Module Details: *deleteCourse*

Test Owner: *Yash Gupta (190997)*

Test Date: *04/04/2022*

Test Results: *Deleting a course from an existing semester in the backend removes that course from the semester in the frontend*

21. Courses selected in the course planner get their timings imported into the calendar UI

Module Details: *addCalendarEvent, addSemester, addCourse*

Test Owner: *Yash Gupta (190997)*

Test Date: *04/04/2022*

Test Results: *Adding a course while planning a semester and submitting a form imports the weekly timing of that course into the calendar UI and the user is able to see the course schedule on the calendar*

22. Courses selected in the course planner get their timings imported into the calendar UI

Module Details: *addCalendarEvent, addSemester, addCourse*

Test Owner: *Yash Gupta (190997)*

Test Date: *04/04/2022*

Test Results: *Adding a course while planning a semester and submitting a form imports the weekly timing of that course into the calendar UI and the user is able to see the course schedule on the calendar*

4 System Testing

1. **Requirement:** *Testing the complete functionality of the Course Planner*

   Modules used -
   - *addSemester*
   - *addSemesterAtEnd*
   - *addCourse*
   - *getNumberOfSemesters*
   - *getSemester*
   - *getCourse*
   - *deleteCourse*
   - *deleteSemester*

Test Owner: *Yash Gupta (190997)*
Test Date: *04/04/2022*
Test Results:
- *Able to add a new semester*
- *Able to add a course in an existing semester*
- *Able to view all existing semesters with their courses and their details in the frontend*
- *Able to delete existing semester along with all the courses in that semester*
- *Able to delete individual courses in existing semesters*

2. **Requirement:** *Testing the complete functionality of the Course Note*

   Modules used:
   - *insertNote*
   - *deleteNote*
   - *fetchNotes*
   - *updateNote*

Test Owner: *Aditya Ranjan (190068)*
Test Date: *04/04/2022*
Test Results:
- *Able to add a new note to a course*
- *Able to delete an existing note from the list of current notes*
- *Able to fetch all notes from the database and search on them*
- *Able to update an existing note to change its content, title, tags, course*

….

3. **Requirement:** *Testing the complete functionality of the course feedback requirement (a user should be able to add a review to a course and view the same as well. Also, a user can add comments to a course review and view other comments as well. He/she should also be able to mark a course as their favorite).*

   Modules used -
   - *addToFavorite*
   - *addReview*
   - *addComment*
   - *likeReview*
   - *likeComment*
   - *getCourseDetails*
   - *getReviewDetails*
   - *getReviews*
   - *getCommentDetails*
   - *giveRating*

Test Owner: *Yash Gupta (190998)*
Test Date: *[29/03/2022] - [04/04/2022]*
Test Results: *All the tests related to testing this functionality were successfully executed.*

- *Able to add a course to the favorite list (mark a course as favorite)*
- *Able to add a review for a course along with tags*
- *Able to get all the details related to a course including the reviews given and the number of stars*
- *Able to get all the details related to a particular course review including the comments given and number of likes*
- *Able to get a list of courses marked as favorite by a user (requires a user to be logged into the session)*

4. **Requirement:** *Testing the complete functionality of the calendar management feature of the application (a user should be able to add events into the calendar and also can import the class timings of a course while planning the course schedule for a semester.*

   Modules used -
   - *addCalendarEvent*
   - *viewCalendarEvent*
   - *deleteCalendarEvent*

Test Owner: *Yash Gupta (190998)*
Test Date: *[29/03/2022] - [04/04/2022]*
Test Results: *All the tests related to testing this functionality were successfully executed.*

- *Able to manually add events into the calendar, similar to what happens in Google Calendar*
- *Able to view the added events in the calendar along with their specified timings and titles*
- *Able to get the course timings imported into the calendar chosen using the course planner*
- *Able to view clashing events in the calendar and also the clashing course while planning a semester*
- *Able to delete events from the calendar and the database as well*

5 Beta Testing

BUG 1: While Signing up - in the rollNo field, alphabets and other non-numeric characters are allowed causing an internal server error when submitted.

Tested Feature: Sign in

Tester Name: Akansh and Akshan

Testing Date: 16/04/2022

Bug Details:

Test Case 1 : Signup using RollNo as : Abgdv
Test Case 1 Details : Internal server error occurred
Test Case 2 : Signup using RollNo as (special characters) : $@^
Test Case 2 Details : Internal server error occurred

Bug Report Date: 16/04/2022

Has the bug been fixed? Yes

Date of Bug fixing: 17/04/2022

Any other comment: Bug was resolved quickly by the other team.

BUG 2: In sign up - username and rollNo both should be such that no user in the database should have them as one of the attributes, but if during signup, we enter a unique username with an existing rollNo, it registers that user successfully.

Tested Feature: Sign in

Tester Name: Akansh and Akshan

Testing Date: 17/04/22

Bug Details:
Test Case : Signup using any RollNo as mentioned in the database and choosing the username field randomly or vice versa.
Test Case Details : Register the user successfully

Bug Report Date: 17/04/22

Has the bug been fixed? Yes

Date of Bug fixing: 18/04/2022

Any other comment : No comments.

BUG 3: The overall course rating given by one user was not reflected at the end of the other user in the feedback.

Tested Feature: Feedback (add review)

Tester Name: Akansh and Akshan

Testing Date: 17/04/22

Bug Details:
Test Case : Rating given at the end of one user and its effect viewed at other ends.
Test Case Detail: Rating given at the end of one user was not reflected at the end of another user.

Bug Report Date: 17/04/22

Has the bug been fixed? No

Date of Bug fixing: N/A

Any other comment : No comments.

BUG 4: In the month wise calendar, the month names were missing .

Tested Feature: Calendar

Tester Name: Akansh and Akshan

Testing Date: 17/04/22

Bug Details:
Test Case : Checking if the calendar day, month and year wise are correct.
Test Case Detail: Month wise calendar missing the names of the months.

Bug Report Date: 17/04/22

Has the bug been fixed? Yes

Date of Bug fixing: 18/04/22

Any other comment : No comments.

6 Conclusion

How Effective and exhaustive was the testing?
Testing was effective as each module was rigorously tested after its completion via tools like Postman. This helped detecting bugs early during the development and helped streamline the development process.

Which components have not been tested adequately?
The modules for implementing the calendar and checking the pre-requisite for courses put in the course planner have not been tested adequately.

What difficulties have you faced during testing?
The manual testing process took a fair amount of time, as automated testing frameworks were time consuming and hard to setup and work with.

How could the testing process be improved?
The design process could’ve been made more robust which would have reduced the amount of testing needed in the first place. Other than that, automated tests can great reduce the time taken for testing once set up properly.

Appendix A - Group Log

<*Please include here all the minutes from your group meetings, your group activities, and any other relevant information that will assist in determining the effort put forth to produce this document*>

1. Organized meetings over Zoom as well as GMeet and discussed the project status and implementation status
2. Organized meetings to discuss the best package to be used for automating the testing of the applications
3. Carried out discussions for finding out the corner cases to test the application exhaustively to find out any potential bugs