Test Document

for

<ProfInfo Central>

Version <1.0>

Prepared by

Group 2: **Group Name:** Error 404:team not found

Pavani priya	220415	gppriya22@iitk.ac.in
Sontam Deekshitha	221075	sontamd22@iitk.ac.in
Lakshyta Mahajan	220581	lakshyta22@iik.ac.in
Kartik	220503	kartik22@iitk.ac.in
Atharv Moghe	220250	atharvm22@iitk.ac.in
Kuldeep Sandip Thakare	220557	kuldeeps22@iitk.ac.in
Mohd Nasar Siddiqui	220661	snasar22@iitk.ac.in
Prabhat Kumar Yadav	220774	prabhatky22@iitk.ac.in
Nilesh Maneshwar	220715	mnilesh22@iitk.ac.in
Sanapala Jaswanth	220955	sjaswanth22@iitk.ac.in

Course:	CS253
Mentor TA:	Abhilash Chandra
Date:	01-04-2024

CONTENTS.......... II

REVISIONS.......... II

1 INTRODUCTION.......... 1

2 UNIT TESTING.......... 2

3 INTEGRATION TESTING.......... 3

4 SYSTEM TESTING.......... 4

5 CONCLUSION.......... 5

APPENDIX A - GROUP LOG.......... 6

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Version 1	Error 404: team not found	Built a website where students can access all the professional and research-oriented information regarding many different professors of the campus from different departments and can also request projects under the professors according to their specific interests.	01/04/2024

1 Introduction

Test Strategy

1) For Unit Testing we have used manual testing and verified the output we got against the correct output.

2) For Integration Testing we have used manual testing. We developed the test cases based on specifications, requirements and system design documents.

3) For System testing we have used a manual testing method where we checked all functional as well as non functional requirements from the SRS document and tested our software.

When was Testing Conducted ?

Testing was mainly done after the implementation was over. Manual testing was done during the period of implementation but major part was done after implementing the code.

Who were the Testers ?

Testing was done by almost half of the team members who were majorly not involved in the backend development.

Coverage Criteria

We have tried to cover every case possible from the code. Manual Testing was done exhaustively to include every possible function coverage.

Tools For Testing

For Backend Testing we have used Postman tool. This was used to test the backend code. For Integration and System testing we used manual testing only.

2 Unit Testing

1. **POST /Login**

   Login function is created for both the Student login and the Professor’s Login. It was tested for a given username and its unique password. It functioned correctly as expected.

Unit Details: Backend Function for handling Post Request “/login”.

Test Owner: [Jaswanth]

Test Date: [03/26/2024] - [03/27/2024]

Test Results:

Correct Username and Correct Password - Login successful

Incorrect Username/ Password - No Login

Structural Coverage: 5 new usernames and their unique passwords were created and tested randomly on each other. Login was only successful in providing the correct username and the respective password.

Additional Comments: Unregistered emails failed to login.

2. **POST /Register**

   Register function is created to register new users to the website. New username is provided to them as well as a password is provided to each user.

Unit Details: Backend Function for handling Post Request “/Register”.

Test Owner: [Jaswanth]

Test Date: [03/26/2024] - [03/27/2024]

Test Results:

If New Username is provided then the user is registered.

If Existing Username is provided then Registration fails.

Structural Coverage: The register function was tested with existing usernames as well as with new usernames. 100% branch coverage was done.

Additional Comments: While Registering, In a few rows if we type an unrelated value even then it’s accepting which is incorrect. For example -ve values at CPI row, Numbers at branch row, random strings at resume row etc.. but we left that to the student concern.

3. **GET /Logout**

   Logout function is created to logout an user who is logged in to the website.

Unit Details: Backend Function for handling Get Request “/Logout” .

Test Owner: [Jaswanth]

Test Date: [03/26/2024] - [03/27/2024]

Test Results: User successfully Logged out from his account.

Structural Coverage: Branch Coverage - 100%

Additional Comments: Functioned correctly.

4. **GET /Fetch Projects**

   This function is created for a student to see his projects in the project page.

Unit Details: Backend Function for handling Get Request “/Fetch Projects”.

Test Owner: [Jaswanth]
Test Date: [03/26/2024] - [03/27/2024]
Test Results: Clicking on the Projects function correctly displays the projects the student has taken.
Structural Coverage: Branch Coverage - 100%
Additional Comments: Functioned correctly.

5. **GET /Fetch faculty**

   This function is created for students to see the professors of a particular department offering projects.

Unit Details: Backend Function for handling Get Request “/Faculty”.

Test Owner: [Jaswanth]
Test Date: [03/26/2024] - [03/27/2024]
Test Results: Successfully showed the faculties.
Structural Coverage: This function is particularly static as it is fed with data of all the professors of various departments. We tested it to see the professors of all branches which were fed up to the database. Branch Coverage - 100% .
Additional Comments: A student even can go through the projects offered by a particular faculty if he clicks on his card.

6. **PATCH /Your Projects**

   This function is created for the Professor to see the list of projects he/she has offered and to add a new project.

Unit Details: Backend Function to handle Patch Request “/Projects".

Test Owner: [Kuldeep Sandip Thakare]
Test Date: [03/26/2024] - [03/27/2024]
Test Results: Viewing projects was successful as well as adding new projects function was also functioning well.
Structural Coverage: Branch Coverage - 90%.
Additional Comments: Successful.

7. **GET /Profile(Student)**

   This function is Student on the Navbar created to view the profile of a student.

Unit Details: Backend Function to handle Get Request "/Profile".

Test Owner: [Jaswanth]
Test Date: [03/26/2024] - [03/27/2024]
Test Results: Unable to fetch image rest everything was successful.
Structural Coverage: Branch Coverage - 95%
Additional Comments: Functioned correctly for all text data entered only image was not fetched.

8. **GET /Profile(Professor)**

   This function is created for a professor to see his/her Profile.

Unit Details: Backend Function for handling Get Request "/Profile".

Test Owner: [Kuldeep Sandip Thakare]
Test Date: [03/26/2024] - [03/27/2024]
Test Results: Only one image is being fetched and rest data is also incorrect. Failed
Structural Coverage: Branch Coverage - 90%
Additional Comments: Function Failed.

9. **DELETE /Delete Project**

   This function is created for a Professor to delete a project. Click on Show More>>Delete.

Unit Details: Backend function to handle Delete Request "/Delete Project".

Test Owner: [Kuldeep Sandip Thakare]
Test Date: [03/26/2024] - [03/27/2024]
Test Results: Function successfully deleted the deleted project.
Structural Coverage: Branch Coverage - 90%
Additional Comments: Functioned correctly.

10. **GET/Get Resume**

    This backend function to fetch Resume from the register page is tested.All the testable branches functioned as expected.

Unit Details: Backend function to handle Get Request "/Get Resume".

Test Owner: [Jaswanth]
Test Date: [03/26/2024] - [03/27/2024]
Test Results: Function successfully fetched the Resume from the Register page.
Structural Coverage: Branch Coverage - 90%

Additional Comments: Functioned correctly.

11. Get/Get Project Request

The backend function that fetches project requests from student to professor was tested. All the testable branches functioned as expected

Unit Details: Backend function to handle Get Request “/Get Project Request”.

Test Owner: [Jaswanth]
Test Date: [03/26/2024] - [03/27/2024]
Test Results: Function successfully sent project request from student to professor.
Structural Coverage: Branch Coverage - 100%
Additional Comments: Functioned correctly.

12. GET /Display of Projects in Profile

This function is created for a student to see his projects in the project page.

Unit Details: Function to handle Get Request “/Display Projects”.

Test Owner: [Jaswanth]
Test Date: [03/26/2024] - [03/27/2024]
Test Results: Showed the request projects successfully.
Structural Coverage: Branch Coverage - 90%
Additional Comments: Functioned correctly.

13. GET /Contact Us

This function is set up for contacting our team in case of any issue it redirects to our Linkedin and Instagram handles.

Unit Details: Function to handle Get Request “/ContactUs”.

Test Owner: [Jaswanth]
Test Date: [03/26/2024] - [03/27/2024]
Test Results: Successful.
Structural Coverage: Branch Coverage - 100%
Additional Comments: Functioned correctly.

14. POST/Student Cannot register as faculty

Checking if the student can login as a faculty with the same username and password.We entered the student details and unfortunately we can log in as faculty. We have fixed the bug in such a way that at the time when faculty enters,He has to verify his mail. An OTP will be sent to his mail and after entering the OTP correctly, he will be logged in. The database is such that there were only the mails of the professors in it.

Unit Details Logging in with student credentials to faculty.

Test Owner: [Jaswanth]
Test Date: [03/26/2024] - [03/27/2024]
Test Results:
Incorrect Username or Password - Login Failed
Correct Username and Password of students - Login successful
After the bug fix,
User mail ID enter >> OTP will be sent if and only if User is a professor.
Enter OTP >> Login Successful
Wrong OTP >> “OTP does not match”
Structural Coverage: Branch Coverage - 100%
Additional Comments: Bug fixed and all the test cases were passed.

15. **GET/Accept**

The backend function that a professor accepts for a student to do a project is tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Get Accept “/Accept”

Test Owner: [Jaswanth]
Test Date: [03/26/2024] - [03/27/2024]
Test Results: The project is displayed in students project list
Structural Coverage: Branch Coverage - 100%
Additional Comments: Functioned correctly.

16. **GET/Enrolled Student**

The backend function which places the name of the student and his details who are doing projects in the Enrolled Students page were tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Get Enrolled Students “/Enrolled Student”

Test Owner: [Jaswanth]
Test Date: [03/26/2024] - [03/27/2024]
Test Results: The student is displayed in the enrolled Projects page to a professor.
Structural Coverage: Branch Coverage - 100%
Additional Comments: Functioned correctly.

17. **GET/Reject**

The backend function that a professor rejects for a student to do a project is tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Get Reject “/Reject”

Test Owner: [Jaswanth]
Test Date: [03/26/2024] - [03/27/2024]
Test Results: The student got rejected to do the project.
Structural Coverage: Branch Coverage - 100%
Additional Comments: Functioned correctly.

18. **POST/Login through Enter Key**

    In the Login page upon clicking on the "Enter" key, it should work as if I had clicked on the Login button.The function for it was tested and functioned as expected.

Unit Details: Used Enter button at Login page to login.

Test Owner: [Jaswanth]
Test Date: [03/26/2024] - [03/27/2024]
Test Results:
Correct Username and Correct Password - Login successful
Any other case - No Login
Structural Coverage: 5 new usernames and their unique passwords were created and tested randomly on each other. Login was only successful in providing the correct username and the respective password.100% branch coverage was done.
Additional Comments: Functioned correctly.

19. **POST /Register through Enter Key**

    In the Login page upon clicking on the "Enter" key, it should work as if I had clicked on the Login button.The function for it was tested and functioned as expected.

Unit Details: Used Enter button at the Register page to register.

Test Owner: [Jaswanth]
Test Date: [03/26/2024] - [03/27/2024]
Test Results:
If New Username is provided then the user is registered.
If Existing Username is provided then Registration fails.
Structural Coverage: The register function was tested with existing usernames as well as with new usernames. 100% branch coverage was done.
Additional Comments: Register function correctly registered new users to the website.

3 Integration Testing

1. **Checking the Interface between Login and Dashboard for User**

Module Details:
Test Owner: Atharv Moghe
Test Date: [03/27/2024] - [03/27/2024]
Test Results: When entering the correct user login credentials, the software redirects to the corresponding dashboard

2. **Logout redirects any user to the login page**

Module Details:
Test Owner: Sanapala Jaswanth
Test Date: [03/27/2024] - [03/27/2024]
Test Results: Test successful for all the roles

3. **Signup with registered Emails not possible**
Module Details:
Test Owner: Sontam Deekshitha
Test Date: [03/27/2024] - [03/27/2024]
Test Results: Shows " Registration failed! or User already exists. Please try again!!!"

4. **OTP Verification for registration**
Module Details: OTP Verification
Test Owner: Kartik
Test Date: [03/27/2024] - [03/27/2024]
Test Results: OTP is sent and verified successfully

5. **Registration done with IITK Mail-ID only**
Module Details:
Test Owner: Pavani Priya
Test Date: [03/27/2024] - [03/27/2024]
Test Results: Doesn't accept other email IDs for registration other than IITK Email-ID

6. **Login with incorrect credentials**
Module Details:
Test Owner: Lakshyta Mahajan
Test Date: [03/27/2024] - [03/27/2024]
Test Results: User is unable to login when he enters incorrect credentials.

7. Student can request a project

Module Details:

Test Owner: Nasar

Test Date: [03/27/2024] - [03/27/2024]

Test Results: Students are able to request the projects they are interested in, offered by the professors.

8. Requested project shown on student's profile

Module Details:

Test Owner: Nilesh

Test Date: [03/27/2024] - [03/27/2024]

Test Results: Requested projects are shown when students check their profile on the website.

9. Students can request any project only once

Module Details:

Test Owner: Kuldeep

Test Date: [03/27/2024] - [03/27/2024]

Test Results: A project can be requested only once by the student.

10. Restricted number of projects

Module Details:

Test Owner: Prabhat Kumar Yadav

Test Date: [03/27/2024] - [03/27/2024]

Test Results: A student can only request for 5 projects once.

11. Students viewing details and projects offered by professors

Module Details:

Test Owner: Atharv Moghe

Test Date: [03/27/2024] - [03/27/2024]

Test Results: Students can view the details of the professors available on the website and also the projects they offer.

12. Student can view list of all projects category wise

Module Details:

Test Owner: Jaswanth

Test Date: [03/27/2024] - [03/27/2024]

Test Results: All the projects available are shown category wise to the students in the website.

13. Faculty Registration and Login

Module Details:

Test Owner: Pavani priya

Test Date: [03/27/2024] - [03/27/2024]

Test Results: Faculty are able to register and login by entering the correct credentials

14. Requested projects and enrolled students shown in the respective professor's profile

Module Details:

Test Owner: Sontam Deekshitha

Test Date: [03/27/2024] - [03/27/2024]

Test Results: Professors can see the requests sent by students for the projects offered , in their respective profiles and also see the students enrolled in their projects

15. List of all projects

Module Details:

Test Owner: Kartik

Test Date: [03/27/2024] - [03/27/2024]

Test Results: Professors can view all the projects they offer in their profiles.

16. New projects added are shown on both the interfaces

Module Details:

Test Owner: Nasar

Test Date: [03/27/2024] - [03/27/2024]

Test Results: whatever projects are added by the professors, they are shown on both professors' and students' interface.

17. Deleting existing projects and updation on interfaces

Module Details:

Test Owner: Atharv Moghe

Test Date: [03/27/2024] - [03/27/2024]

Test Results: When professors delete any existing project, the updated available projects will be shown on both student and professors interface.

18. Accepting/Rejecting project requests

Module Details:

Test Owner: Jaswanth

Test Date: [03/27/2024] **-** [03/27/2024]

Test Results: Professors can accept or reject the requests from the students and can also make changes in them which will be shown in the student interfaces also.

4 System Testing

1. **Requirement: The student should be able to request a project from a professor**

The system should allow the student to request a project from a professor. The steps to complete this test are as follows:

1) The student logs into the system using their credentials.

2) The student navigates to the "Projects" section of the system.

3) The student selects the desired project.

4) The student selects the option "Request “

5)The student submits the request.

6)The system processes the request and notifies the professor.

.

Test Owner:Jaswanth
Test Date:[03/28/2024] - [03/28/2024]
Test Results:The test passed. The student was able to request a project from the professor without encountering any errors

2. **Requirement:The faculty should be able to see the requests**

1) The faculty member logs into the system using their credentials.

2)The faculty member navigates to the "Requests" section of the system.

3)The faculty member should be able to see a list of requests made by students, including details such as student name, project requested, and request status.

4)The faculty member can click on individual requests to view more details if needed

Test Owner: Atharv Moghe
Test Date: [03/28/2024] - [03/28/2024]
Test Results: The test passed. The functionality appears to be working as expected, allowing faculty members to seamlessly view requests made by students for projects

3. **Requirement: The faculty should be able to accept a particular student and give him a project**

   1)The faculty member logs into the system using their credentials.

   2)The faculty member navigates to the "Requests" section of the system.

   3)The faculty member reviews the list of requests made by students.

   4)The faculty member selects a particular request from a student they wish to accept.

   5)The faculty member clicks on the request to view more details.

   6)The faculty member finds an option to accept the student and assign them the project.

Test Owner: Kuldeep Thakare
Test Date: [03/28/2024] - [03/28/2024]
Test Results:The test yielded positive results. The faculty member effectively approved a specific student and allocated them a project

4. **Requirement: The faculty should be able to reject a particular student before accepting**

   1)The faculty member logs into the system using their credentials.

   2)The faculty member navigates to the "Requests" section of the system.

   3)The faculty member reviews the list of requests made by students.

   4)The faculty member selects a particular request from a student they wish to reject.

5)The faculty member clicks on the request to view more details.

6)The faculty member finds an option to reject the student's request.

7)The faculty member provides a reason for the rejection, if required.

8}The faculty member confirms the rejection of the student's request.

Test Owner:Prabhat Kumar Yadav
Test Date:[03/28/2024] - [03/28/2024]
Test Results:The functionality appears to be working as expected, allowing faculty members to seamlessly reject students' requests before accepting them

5. **Requirement:The faculty should be able to add or delete their project**

1. The faculty member logs into the system using their credentials.

2. The faculty member navigates to the "Projects" section of the system.

3. To add a new project:

   a. The faculty member finds an option to add a new project.
   b. The faculty member fills out the necessary details for the new project, such as title, description, and requirements.
   c. The faculty member saves the new project.

4. To delete an existing project:

   a. The faculty member navigates to the list of existing projects.
   b. The faculty member selects the project they wish to delete.
   c. The faculty member finds an option to delete the selected project.
   d. The faculty member confirms the deletion of the project.

Test Owner: Pavani Priya
Test Date: [03/29/2024] - [04/01/2024]
Test Results:The test concluded successfully. The faculty member effectively added new projects and removed existing ones

6. **Requirement:The student should be able to see all the projects which the faculties are offering**

    1. The student logs into the system using their credentials.

    2. The student navigates to the "Projects" section of the system.

    3. The student should be able to see a list of all the projects offered by faculties.

    4. Each project should display relevant details such as title, description, and faculty member offering it.

    5. The student should be able to click on individual projects to view more details if needed.

Test Owner:Jaswanth
Test Date: [03/29/2024] - [04/01/2024]
Test Results:The test passed, making it easy for students to find and explore projects offered by faculties.

7. **Requirement:The profile should be updated every time he gets a project if he is a student and he offers a project if he is a faculty**

The system should automatically update the profile of a user (student or faculty) every time they get assigned a project (if they are a student) or offer a project (if they are a faculty member).

The steps to complete this test are as follows:

a. If the user is a student:
- The system should update the student's profile information to reflect the newly assigned project.

b. If the user is a faculty member:
- The system should update the faculty member's profile information to reflect the newly offered project.

Test Owner:Lakshyta Mahajan
Test Date:[03/29/2024] - [04/01/2024]

Test Results:The test passed. The profile of the user was automatically updated as expected, reflecting the new project assignment (if the user is a student) or project offering (if the user is a faculty member)

8. **Requirement:The faculty should be able to view the details of the students like resume etc..**

The steps to complete this test are as follows:

1. The faculty member logs into the system using their credentials.
2. The faculty member navigates to the "Requests” section of the system.
3. The system should display the student's profile, including details like name, roll no, cpi , and resume.
5. The faculty member should be able to click on the resume link to view the student's resume if available.

Test Owner:Kartik
Test Date:[03/29/2024] - [04/01/2024]
Test Results:The test was successful. Faculty members could efficiently access and review student details, including their resumes.

9. **Requirement: The faculty should be able to reject a particular student after accepting if he wants**

1. The faculty member logs into the system using their credentials.

2. The faculty member navigates to the "Enrolled Students” section of the system.

3. The faculty member selects the student they previously accepted but now wish to reject.

4. The faculty member finds an option to reject the student.

5. The system should allow the faculty member to reject the student, even after they have previously accepted them.

Test Owner: Nasar
Test Date:[03/29/2024] - [04/01/2024]

Test Results:The test passed. The system successfully allowed the faculty member to reject a particular student after accepting them.

5 Conclusion

How Effective and exhaustive was the testing?

Testing was quite effective. We ensured that when any member pushed their local changes to the main repo, the code was working as intended and there were no major flaws. We have even tested manually with some test cases right before merging.

Our application has 2 users: student, faculty. We have tested in such a way that the student and faculty backend functions will not give any issue for all the possibilities. But since there are plenty of functions, the testing was quite exhaustive.

Which components have not been tested adequately?

All the components have been tested adequately up to our knowledge.

What difficulties have you faced during testing?

We have done testing manually which is a very time-consuming process since we have to test as many test cases as we can(Especially in integration testing). In unit testing, ensuring branch coverage was a little difficult.

How could the testing process be improved?

The GitHub repo has helped us a lot while testing, so maintaining a track of development would improve the testing process. Good documentation allows us to identify the portions that need testing. Automating all testing processes can result in a lot of improvement and even save a lot of time. At Least we can automate the extremely sophisticated queries made to the backend by Postman.

Appendix A - Group Log

Sl.No	Date	Time and Place	Remarks
1	22-03-2024	3pm - 4:30pm RM 4th floor	Discussed how to do testing and divided the work
2	25-03-2024	3pm - 4pm Zoom meet	Prepared a document which contains the tests we have to do
3	28-03-2024	4pm - 6pm Zoom meet	Completed unit testing and system testing and fixed the bugs
4	30-03-2024	4pm - 5:30pm Zoom meet	Completed Integration testing and fixed the bugs
5	01-04-2024	9pm - 10:30pm	Finalised the document.