Test Document

for

S.M.A.R.T.E.X

Version 2.0

Prepared by

Group :16

Group Name: BitEngine

Danish Mehmood	**210297**	danishm21@iitk.ac.in
Rikesh Sharma	**180606**	rikesh@iitk.ac.in
Ajeet Meena	**210078**	ajeetm21@iitk.ac.in
Pratiksha Dawane	**210760**	ppdawane21@iitk.ac.in
Anuj Kumar	**200164**	kanuj20@iitk.ac.in
Kumar Gourav Songara	**170353**	songara@iitk.ac.in
Avinash Saini	**200232**	avinashs20@iitk.ac.in
Lohit P Talavar	**210564**	lohitp21@iitk.ac.in

Course: CS253

Mentor TA: *Rumit*

Date: 22/04/23

CONTENTS II

REVISIONS II

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Test Document Version 2.0	Danish Mehmood	This is the Test Doc v2.0 for S.M.A.R.T.E.X Final Draft	22/04/23

1 Introduction

For CAEES we have used both manual as well as automated testing methods at various places. For frontend we mostly had to test the JavaScript and it was tested manually, whereas in backend we used manual as well as automated methods.

The overall testing was conducted after the implementation but we also manually tested the functions while implementing also, while debugging the implemented code.

For this phase of testing the developers themselves were the testers. The developer tested their own functions along with the functions of other developers too.

The tools which we used for testing are Selenium and POSTMAN.

Selenium is an open-source, widely-used automated testing tool for web applications. It allows testers and developers to automate functional tests, integration tests, and acceptance tests, among others. Selenium provides a suite of tools for creating, executing, and managing automated tests, including a Selenium WebDriver for browser automation, a Selenium IDE for recording and replaying tests, and a Selenium Grid for parallel test execution.

Postman is a popular API testing tool used by developers and testers to streamline the testing process. It offers a user-friendly interface that simplifies API testing, allowing users to quickly create and execute tests. With Postman, users can send HTTP requests to APIs and receive responses, making it easy to verify the functionality of APIs. The tool also offers features such as automated testing, data-driven testing, and performance testing. Postman supports a wide range of APIs and is compatible with a lot of popular platforms. Its ease of use and powerful features make it an ideal tool for API testing, whether you are working on a small project or a large-scale application.

2 Unit Testing

1. **Unit Name**: Admin Login Component

Unit Details: (username, password), login function
Test Owner: Ajeet Meena
Test Date: 03/25/2023 - 03/26/2023
Test Results: The login function successfully logs in the admin with valid credentials and displays an error message for invalid or incorrect credentials. The system correctly identified valid admin's credentials and redirected to student information page for further activity, while showing error message for invalid credentials.
Structural Coverage: 100 % of branch (if/else statements) coverage.
Additional Comments: Admin login component appears to be working as expected.

2. **Unit Name:** Cre_dits Component

Unit Details: nav-item, function
Test Owner: Ajeet Meena
Test Date: 03/25/2023 - 03/26/2023
Test Results: Each navigation bar item click event handler successfully redirects the admin to the expected page.
Structural Coverage: All lines of codes were executed during testing.
Additional Comments: None.

3. **Unit Name**: Success Component

Unit Details: placeInput, submitForm function
Test Owner: Ajeet Meena
Test Date: 03/26/2023 - 03/26/2023
Test Results: The submitForm function successfully submits the student exit information with correct data type of place of visit and displays an error message for integer inputs in place of visit section.
Structural Coverage: N/A
Additional Comments: Success component appears to be working as expected.

4. **Unit Name**: Details Component

Unit Details: rollNoInput, manualEntry function
Test Owner: Ajeet Meena
Test Date: 03/26/2023 - 03/27/2023

Test Results: The manualEntry function successfully redirects the admin to the expected page after filling the valid roll number in right place. The system displaying error message for incorrect or invalid roll number and not redirects the admin to student information page.
Structural Coverage: All lines of codes were executed during testing.
Additional Comments: Details component appears to be working as expected and meets the requirements. In case of error message, admin have to re-enter the correct/valid roll number for further activities.

5. **Unit Name**: History Component

Unit Details: search-input, search function
Test Owner: Ajeet Meena
Test Date: 03/26/2023 - 03/27/2023
Test Results: The function successfully gives the required row/rows on entering that input text in search bar for which data are available on server. The system displays no row/rows of data for invalid input. One more thing, the system shows all the available rows of data for empty input.
Structural Coverage: if/else statements are working correctly with 100 %.
Additional Comments: History component appears to be working as expected and meets the requirements. The system is not giving any row of available data for incorrect input text or number.

6. **Unit Name**: Edit&delete Component

Unit Details: btn-primary, remove function
Test Owner: Ajeet Meena
Test Date: 03/26/2023 - 03/27/2023
Test Results: The remove function successfully removes or delete the required row for requested data which are available on server. The edit function successfully edits/changes the data for which admin had requested. For invalid or incorrect query, remove and edit function have not done anything. On successful row deletion, the system displays the success message. And for unsuccessful operation, the system displays error message.
Structural Coverage: All functions are doing work correctly.
Additional Comments: History component appears to be working as expected and meets the requirements. The system shows/displays the expected messages according to the successful/unsuccessful operation on query.

7. **Add to exit table**

The addExit function adds a record of a student's exit from their hall to the smartex-exit-table table in DynamoDB. It takes in an event object containing a request body that

specifies the student's roll number, place of visit, date, time, and guard ID, and a callback function that returns a response object containing the status code and a message.

Inputs:

- **event**: an object containing the following properties:
  - **body**: a JSON object containing the following properties:
    - **rollno**: a string representing the student's roll number.
    - **placeOfVisit**: (optional) a string representing the place the student is visiting.
    - **date**: a string representing the date of the student's exit in the format yyyy-mm-dd.
    - **time**: a string representing the time of the student's exit in the format hh:mm.
    - **gaurdID**: a string representing the ID of the guard who is logging the exit.
- **callback**: a function that takes in two arguments:
  - **error**: (optional) an object representing any errors that occurred during the execution of the function.
  - **response**: an object representing the response to be returned to the client, containing the following properties:
    - **statusCode**: an integer representing the HTTP status code of the response.
    - **body**: a JSON string representing the response body.

Outputs:

- If the operation is successful, the function should return a response object with a status code of 200 and a message containing the details of the student's exit.
- If the operation fails, the function should return an error object with details of the failure and a status code of 500.

Unit Details: Add Exit

Test Owner: Rikesh Sharma

Test Date: 22nd March 2023

Test Results: Successfully adding the valid input to the smartex-exit-table. If the input is invalid or incomplete it gives error.

Structural Coverage: statement coverage and Branch coverage

Additional Comments:

8. Retrieve Information from exit table

The getExit function retrieves exit details of a student from the smartex-exit-table in the DynamoDB database. It takes in an event object containing the rollno parameter of the student, and a callback function that returns either an error or the retrieved exit details in JSON format.

Unit Details: getExitDetail

Test Owner: Rikesh Sharma

Test Date: 22nd March 2023
Test Results:
Case 1: Retrieval of Exit Details for a Valid Student

Test Description	Test Data	Expected Result	Test Result
Retrieval of exit details for a valid student	event: {body: {rollno: "1810001"}}	A JSON object containing the exit details for the student with rollno 1810001 is returned	Passed
Retrieval of exit details for a valid student	event: {body: {rollno: "1820001"}}	A JSON object containing the exit details for the student with rollno 1820001 is returned	Passed

Test Case 2: Retrieval of Exit Details for an Invalid Student

Test Description	Test Data	Expected Result	Test Result
Retrieval of exit details for an invalid student	event: {body: {rollno: "1899999"}}	An error object is returned with the message "Unable to fetch data"	Passed
Retrieval of exit details for an invalid student	event: {body: {rollno: "2099999"}}	An error object is returned with the message "Unable to fetch data"	Passed

Test Case 3: Retrieval of Exit Details with Invalid Input

Test Description	Test Data	Expected Result	Test Result
Retrieval of exit details with invalid input	event: {body: {rollno: ""}}	An error object is returned with the message "Unable to fetch data"	Passed
Retrieval of exit details with invalid input	event: {body: {}}	An error object is returned with the message "Unable to fetch data"	Passed

Structural Coverage: Statement Coverage, Branch coverage, Error Handling and Invalid Input Handling
Additional Comments:

9. Deletion from exit table

Unit Details: deleteExit
Test Owner: Rikesh Sharma
Test Date: 22nd March 2023
Test Results:

Test Case 1: Delete an exit user successfully

Description: Test whether the function can delete an exit user from the "smartex-exit-table" table successfully.

Result: AWS SDK's `docClient.delete()` method returns no error and a success response.

Test Case 2: Error deleting an exit user

Description: Test whether the function returns an error response when there is an error deleting an exit user from the "smartex-exit-table" table.

Result: AWS SDK's `docClient.delete()` method returns an error.
Structural Coverage: Statement Coverage, Error Handling and Invalid Input Handling
Additional Comments:

10. Add to movement table

Description: This function takes an event and a callback function as input parameters. It fetches the student data from a table using roll number, deletes the corresponding row from the table, and then adds an entry in the movement table with the exit and entry details of the student.

Unit Details: addMovement
Test Owner: Rikesh Sharma
Test Date: 24th March 2023
Test Results:
Testing the function with valid input parameters Input:

```
event = { "body":
        "{ "rollno": "18110123",
        "date": "2022-04-02",
        "time": "16:00",
        "gaurdID": "123",
        "placeOfVisit": "IITK Gate" }"
        }
        callback = (error, result) => { console.log(result) }
```

Expected Output: The function should add a new entry in the movement table with the given input parameters and return a success response with status code 200.
Result: **Passed**

Testing the function with invalid roll number Input:
event = { "body":
"{ "rollno": "invalid_rollno",
"date": "2022-04-02",
"time": "16:00",
"gaurdID": "123",
"placeOfVisit": "IITK Gate" }"
} callback = (error, result) => { console.log(result) }
Expected Output: The function should return an error response with an appropriate message indicating that the given roll number is invalid.
Result: Passed

Testing the function with missing input parameters Input:
event = { "body":
"{ "date": "2022-04-02",
"time": "16:00",
"gaurdID": "123",
"placeOfVisit": "IITK Back Gate" }"
} callback = (error, result) => { console.log(result) }
Expected Output: The function should return an error response with an appropriate message indicating that the roll number is missing in the input parameters.
Result : **Passed**

Structural Coverage: Statement Coverage, Error Handling and Invalid Input Handling
Additional Comments:

11. Get information from movement table

Description: This function takes an event and a callback function as input parameters. It fetches the student data from a table using roll number, deletes the corresponding row from the table, and then adds an entry in the movement table with the exit and entry details of the student.

Unit Details: getMovement

Test Owner: Rikesh Sharma

Test Date: 24th March 2023

Test case for valid input:
Input: { "body": { "rollno": "123", "date": "2023-04-01", "time": "09:30:00" } }
Expected Output: The response object containing the exit-user details with a status code of 200.
Result: Passed

Test case for invalid input:
Input: { "body": { "rollno": "123", "date": "2023-04-01", "time": "" } }
Expected Output: An error message with a status code of 400.
Result: Passed

Test case for non-existing user:
Input: { "body": { "rollno": "456", "date": "2023-04-01", "time": "09:30:00" } }
Expected Output: An error message with a status code of 404.
Result: Passed

Structural Coverage: Statement Coverage, Error Handling and Invalid Input Handling.

12. **Unit Name**: Guard Login Component

[Include basic information about what was tested and what happened.]

Unit Details: (username, password), login function
Test Owner: Ajeet Meena
Test Date: 03/25/2023 - 03/26/2023
Test Results: The login function successfully logs in the guard with valid credentials and displays an error message for invalid or incorrect credentials. The system correctly identified valid guard's credentials and redirected to manual page for further activity, while showing error message for invalid credentials.
Structural Coverage: 100 % of branch (if/else statements) coverage.
Additional Comments: Guard login component appears to be working as expected.

13. **Unit Name**: Admin_pw Component

[Include basic information about what was tested and what happened.]

Unit Details: admin_pw_btn, changePassword function
Test Owner: Ajeet Meena
Test Date: 03/30/2023 - 03/31/2023
Test Results: The changePassword function successfully changes the password of admin after confirming its credentials and displays an error message for invalid or incorrect credentials or some other reasons. The system correctly changed the password of admin when admin entered its correct credentials and redirected to home page for further activity, while showing error message for invalid credentials or input.
Structural Coverage: 100 % of branch (if/else statements) coverage.
Additional Comments: Admin_pw component appears to be working as expected.

14. **Unit Name**: Guard_pw Component

[Include basic information about what was tested and what happened.]

Unit Details: guard_pw_btn, changePassword function
Test Owner: Ajeet Meena
Test Date: 03/30/2023 - 03/31/2023
Test Results: The changePassword function successfully changes the password of guard after confirming its credentials and displays an error message for invalid or incorrect credentials or some other reasons. The system correctly changed the password of guard when admin entered its correct credentials and redirected to home page for further activity, while showing error message for invalid credentials or input.
Structural Coverage: 100 % of branch (if/else statements) coverage.
Additional Comments: guard_pw component appears to be working as expected.

15. **Unit Name**: DeleteData Component

[Include basic information about what was tested and what happened.]

Unit Details: btn btn-primary, submitForm function
Test Owner: Ajeet Meena
Test Date: 04/01/2023 - 04/01/2023
Test Results: The submitForm function successfully submits the “update entry-exit form” after pressing the submit button on DeleteData page. An error message can be display due to network issue or server error or some other reasons. The system correctly deleted the requested row of data and redirected to history page and guard can take another action any number of times.
Structural Coverage: N/A.
Additional Comments: DeleteData component appears to be working as expected.

16. **Unit Name**: UpdateEntryExit Component

[Include basic information about what was tested and what happened.]

Unit Details: btn btn-primary, submitForm function
Test Owner: Ajeet Meena
Test Date: 04/01/2023 - 04/01/2023
Test Results: The submitForm function successfully submits the “update entry-exit form” after filling the update entry-exit form and then pressing the submit button of UpdateEntryExit page. An error message can be display due to network issue or server error or some other reasons. The system correctly edited or changed the data of the requested row of data and redirected to history page and guard can take another action any number of times.
Structural Coverage: All statements are executed carefully during the unit testing for particular component.
Additional Comments: UpdateEntryExit component appears to be working as expected.

17. **Unit Name**: nonstuedentregis Component

[Include basic information about what was tested and what happened.]

Unit Details: btn btn-primary, submitForm function
Test Owner: Ajeet Meena
Test Date: 04/02/2023 - 04/02/2023
Test Results: The submitForm function successfully registers the new non-student after filling the “student info” form nonstudentregis page. New non-student can get error message on screen if he/she enters unrequired/unexpected inputs in input fields or due to network issue or server. The system correctly added/registered the new non-student user into the database and after this redirected to home page and he/she can take another action like login to access the service for further activity.
Structural Coverage: All statements are executed during testing.
Additional Comments: nonstudentregis component appears to be working as expected

18. **Unit Name**: newadmin Component

[Include basic information about what was tested and what happened.]

Unit Details: btn btn-primary, addNewAdmin function
Test Owner: Ajeet Meena
Test Date: 04/02/2023 - 04/02/2023
Test Results: The addNewAdmin function successfully adds the new admin after successful validation of its credentials by filling the form of newadmin page. An error message can be display due to incorrect/invalid credentials or network issue or server error or some other reasons. The system correctly allowed the admin to access the service of the system and redirected to the expected page and admin can take another action for further process.
Structural Coverage: 100% of branch coverage (if-else statement).
Additional Comments: newadmin component appears to be working as expected.

19. Add General User

Unit Details: addGeneralUser()
Test Owner: Rikesh Sharma
Test Date: 4th April 2023

Test Results:

Test Cases:

1. Test to verify successful addition of general user data to the database.

Input: { "body": "{ "userID": "12345", "name": "John Doe", "address": "123 Main St, City", "gender": "male", "email": "johndoe@example.com" }" }

Output: { "statusCode": 200, "body": "{\n "userID": "12345",\n "name": "John Doe",\n "address": "123 Main St, City",\n "gender": "male",\n "email": "johndoe@example.com"\n}" }

Expected Output:

- statusCode: 200
- body: JSON object with the same properties as input

2. Test to verify error handling when unable to add general user data to the database.

Input: { "body": "{ "userID": "12345", "name": "John Doe", "address": "123 Main St, City", "gender": "male", "email": "johndoe@example.com" }" }

Output: { "message": "Unable to add General User data. Error JSON: {}" }

Output: message: "Unable to add General User data. Error JSON: {}"

3. Test to verify handling of missing required field in input.

Input: { "body": "{ "userID": "12345", "address": "123 Main St, City", "gender": "male", "email": "johndoe@example.com" }" }

Output: { "message": "Invalid input. 'name' field is required." }

Output: message: "Invalid input. 'name' field is required."

Structural Coverage: Statement coverage, Path coverage
Additional Comments:

20. Delete General User

Unit Details: deleteGeneralUser()
Test Owner: Rikesh Sharma
Test Date: 4th April 2023
Test Results:

Test Case Description	Test Data	Expected Output
Valid request	{ "userID": "user1" }	Success message
Invalid request	{}	Error message
Missing userID	{ "name": "John" }	Error message
Error in delete call	{ "userID": "nonexistent" }	Error message

Structural Coverage: statement coverage, path coverage

21. Update General User

The above code defines a function updateGeneralUser that updates a record in the "smartex-general-user-table" table in DynamoDB. The function takes an event and a callback as input parameters, and expects a JSON object in the body of the event, containing the fields userID, address, gender, email, and name.
Unit Details: updateGeneralUser()
Test Owner: Rikesh Sharma
Test Date: 4th April 2023
Test Results:

Test Scenario	Test Steps	Expected Result	Actual Result	Pass/Fail
Valid input is provided for updating general user details	1. Call the updateGeneralUser function with valid input parameters.	The function should successfully update the user details in the database and return a success response.	updated	Pass
Invalid input is provided for updating general user details	1. Call the updateGeneralUser function with invalid input parameters.	The function should throw an error and return an error response.	No such data found. Error.	Pass

Structural Coverage: statement coverage, path coverage

22. Change Password

Unit Details: changeAdminPassword(), changeGaurdPassword()
Test Owner: Rikesh Sharma
Test Date: 6th April 2023
Test Results:

Test case description	Test data	Expected result
Test for successful password change	{ "userID": "1234", "oldPassword": "password123", "newPassword": "newpassword456" }	{ "statusCode": 200, "message": "Password changed successfully" }

Test case description	Test data	Expected result
Test for incorrect old password	{ "userID": "1234", "oldPassword": "wrongpassword", "newPassword": "newpassword456" }	{ "statusCode": 401, "message": "Incorrect old password" }
Test for invalid user ID	{ "userID": "", "oldPassword": "password123", "newPassword": "newpassword456" }	{ "statusCode": 400, "message": "Invalid user ID" }
Test for invalid old password	{ "userID": "1234", "old Password": "", "newPassword": "newpassword456" }	{ "statusCode": 400, "message": "Invalid old password" }
Test for invalid new password	{ "userID": "1234", "oldPassword": "password123", "newPassword": "" }	{ "statusCode": 400, "message": "Invalid new password" }

Structural Coverage: statement coverage, branch coverage, path coverage

23. Add new guard

Unit Details: addGuard()
Test Owner: Rikesh Sharma
Test Date: 6th April 2023
Test Results:

Test Case Description	Test Data	Expected Result
Should add new guard to the table	{guardID: "001", username: "testuser", password: "password123", phoneNo: "1234567890", name: "John Doe", address: "123 Main St"}	{statusCode: 200, body: JSON.stringify({"guardID": "001", "username": "testuser", "password": "password123", "phoneNo": "1234567890", "name": "John Doe", "address": "123 Main St"})}
Should return error when guardID is missing	{username: "testuser", password: "password123", phoneNo: "1234567890", name: "John Doe", address: "123 Main St"}	{statusCode: 400, body: JSON.stringify({error: "Missing required parameter: guardID"})}
Should return error when username is missing	{guardID: "001", password: "password123", phoneNo: "1234567890", name: "John Doe", address: "123 Main St"}	{statusCode: 400, body: JSON.stringify({error: "Missing required parameter: username"})}
Should return error when password is missing	{guardID: "001", username: "testuser", phoneNo: "1234567890", name: "John Doe", address: "123 Main St"}	{statusCode: 400, body: JSON.stringify({error: "Missing required parameter: password"})}

Structural Coverage: Statement coverage, path coverage

12. Integration Testing

1. Module 1

Module details: Bar code scanner page, if bar code does not get scanned user will do manual entry

Test Owner: Pratiksha Dawane

Test Date: 03/28/2023 - 03/29/2023

Test Results:

1st case: Bar code get scanned so there is no need for manual entry.

2nd case: Bar code scanning shows error, then we had a manual entry with user's roll number.

2. Module 2

Module details: log in module with user management data

Test Owner: Pratiksha Dawane

Test Date:03/28/2023 - 03/29/2023

Test Results:

1st case: Test the login functionality with valid credentials - PASS

2nd case: Test the login functionality with invalid credentials - PASS

3rd case: Test the integration with the User Management module for user authentication - PASS

4th case: Test the integration with the User Management module for user role-based access control – PASS

Additional comments: Overall, the Login module was found to be functioning as expected and the integration with the User Management module was successful.

3. Module 3

Module details: Student Entry and Exit Database Integration Testing

Test owner: Pratiksha Dawane

Test date: 03/28/2023-03/28/2023

Test results: 1st case: Test the integration with the Student Management module for student data retrieval - PASS [retrieve student data from the 'Student Management module' and verify that the correct data is displayed in the Student Entry and Exit database software. If the software displays the correct student data, the test is marked as a PASS. If the software displays incorrect or incomplete data, the test is marked as a FAIL.]

4. **Module 4**

database at front end and backend

Test owner: Pratiksha, Lohit
Test date: 03/28/2023 - 03/28/2023

Test results: 1st case: Test the integration with bar code scan result for data base at history unit module –PASS [Check if data is getting placed at history page at frontend and giving edit access to admin and user as per need and as per rules to add or edit data and for backend data retrieval module 4 used.]

2nd case: Test with manual entry by user's Roll number allowing them to submit their student information form and data is getting added to the history page at frontend -PASS

5. **Module 5**

Module details: student information form submission and log out process

Test owner: Pratiksha Dawane

Test date: 03/28/2023 -03/28/2023

Test results: Test integration with form submission and history data base access and log out from the system –PASS [user can log out from system whenever he wants if his form has been submitted data will get saved at database and if at student info page, only data has been entered and not submitted then data would not get saved and user can log out directly]

6. **Module 6**

Module Name: Integration of Guard log in and Admin log in

Test owner: Danish Mehmood

Test Date: 04/07/2023

Test Results: Test integration with guard log in and admin log in – PASS
1st Case : log in with guard credentials -PASS { after successful guard log in page will redirect to bar code scanner page }
2nd case : log in with admin credentials – PASS { after successful admin log in page will redirect to admin dashboard page }

7. **Module 7**

Module Name: admin log in and admin dashboard page and backend user management module

Test owner: Team members

Test Date:04/08/2023

Description: This module aims to provide the admin with the statistics of the daily entry and exit logs. It allows the admin to view the number of entries and exits in a day, week, or month, and generate reports for the same. The module is implemented by integrating the admin dashboard page with the backend data statistics.

Test Results:
1st Test: Test for Daily Entry and Exit Statistics – PASS{The admin can view the number of entries and exits in a day on the dashboard page. The data is fetched from the backend and displayed in a graphical format for easy analysis.}
2nd Test: Test for student and non student entries differently – PASS{The admin can view the number of entries and exits on the dashboard page in graphical format of student and non student differently . The data is fetched from the backend and displayed in a graphical format for easy analysis by fetching student entries by their roll number and non student entries by their ids differently.}

8. **Module 8**

Module Name: Admin log in and admin password change

Test owner: Danish Mehmood

Test Date:04/08/20223

Test Results: Test integration of admin log in and admin password change - PASS {when admin logs in successfully with valid credentials, the software verifies the admin's information against the backend database and grants access to the admin dashboard page. After clicking the radio button on the page, the software retrieves the admin's information from the backend database and takes the admin to the change admin password page. The software then allows the admin to change their password, and the new password is encrypted and stored securely in the backend database.}

9. Module 9

Module Name: Admin log in and Guard password change

Test owner: Danish Mehmood, Ajeet Meena

Test Date: 04/08/20223

Test Results: Test integration of admin log in and guard password change – PASS {when admin logs in successfully, after accessing dashboard page clicking on change admin or guard password, software allows admin to change guard's password too by fetching backend database, the new password is encrypted and stored securely in the backend database. }

Additional comments: It is essential to ensure that the backend database is properly designed and configured to handle the storage and retrieval of sensitive information such as admin and guard passwords

10. **Module 10**

Module Name: Integration of Adding new admin or new guard and then log in to that new account check on the admin page that new added admin's dashboard

Test owner: Ajeet, Rikesh

Test Date: 04/08/20223

Test Results: Test integration of adding new admin or new guard page – PASS { Admin can add new guard or new admin by their information like id and all and if information matches to our database then addition of guard or admin is successful and if not then we can't add it }

11. **Module 11**

Module Name: Integration of admin log in and admin dashboard page and adding new admin and password change pages.

Test owner: Rikesh

Test Date: 04/10/2023

Test Results: Test Integration of admin log in and admin dashboard page – PASS {when admin logs in successfully, page will redirect to admin dashboard page which shows all data entries. }
1st Testt: test with integration of admin log in page and admin dashboard page - PASS {after successful log in, software will redirect admin to admin dashboard page }
2nd Test: test of add new admin – PASS { for now page will redirect to add admin or add guard page to fill information of and who is he admin or guard and then add them }
3rd Test: test of change password of guard or admin – PASS { admin can change his or guard's password by accessing radio button on admin dashboard page}
4th Test: integration of all admin access pages – PASS { after successful admin log in admin can also access all the pages guard can access , admin can go history page and can all the history of entry and exits and can also go for bar code scanning and manual entry page

Additional comment : The dashboard statistics page provides a quick overview of key data such as the number of entries and exits, average time spent on campus, and any other relevant data points. It allows the admin to quickly assess the situation and make informed decisions based on the data

provided.

12. Module 12

Module Name: Integration of guard log in and bar code scanning of user or the manual entry of user

Test owner: Rikesh, Ajeet, Danish , Gaurav

Test Date: 04/09/20223 -04/11/2023

Test Results: Test of Integration of guard log in and bar code scanning of user page or the manual entry of user – PASS
1ST CASE : Test with valid credential on log in page – PASS {when guard logs in successfully page will redirect to bar code scanner page
Test 1: Test with valid bar code- PASS { bar code gets scanned successfully, it allows user to enter his details like place of visit and his entry will be done.}
Test 2: Test with invalid bar code- PASS {as bar code is not found in data base software will show error will tell try manually }
Test 3: Test with valid roll number -PASS {if user tries to do manual entry by entering his roll number or IITK id, when the roll number or id is correct and found in IITK database then only page will redirect to student info page and allows user to enter his details like place of visit }
Test 4 : Test with invalid roll number- PASS { it will show user not found try with valid roll number }
Test 5 : Test with non-student id – PASS { if the id is correct then software will successfully allows user to add his details after redirecting page user info and complete his entry }
Test 6: Test with new registration id – PASS {software allows to do new registration with their id and allows them to have entry with that id like test 5 }

2nd CASE : Test with invalid credential- PASS { as guard log in failed it will show error message and will keep them on log in page only and will asks them to log in with correct id and password}

13. Module 13

Module Name: Integration of guard log in and all the pages guard can access

Test owner: Rikesh, Danish, Ajeet

Test Date: 04/10/20223

Test Results:1st CASE : Test with guard log in and user's entry page { module 12 }
2nd CASE : Test with guard log in and history page of software- PASS
Test 1 : Test guard can delete a entry row of any student or non student user – PASS { guard can delete it by clicking on radion button named delete }
Test 2: Test guard can edit a data entry row of any student or non-student user {guard can edit a data entry of any row by calling to backend and the updated information will be reflected in the database.}
Test 3: Cancel Edit - PASS {If the guard decides to cancel the edit, the information should remain unchanged in the database.}

Test 4: Test of search bar access -PASS { Guard can search anything in the search bar software will sort the data will display it on the page like

Input 1:IN – PASS {by searching IN guard can see data entry with all the user with IN status means who are in campus }

Input 2: 21 – PASS {by searching serial of any roll numbers guard can see all the user with serial of id 21}

Input 3: Rawatpur -PASS { by searching rawatpur guard can see all the entries who have entered rawatpur as their place of visit }

Additional comments: Adding search and filter options to the history page to allow users to quickly find specific entries or narrow down the list based on certain criteria.

13. System Testing

1. **Time Recording**: The system should be able to record time of entry and exit just after scanning ID-card.

Test Owner: Kumar Gourav Songara

Test Date: 04/01/2023

Test Result: First result failed. Ran 2 Test in 10.958s.

Selenium (python lib) has been used for test. Extracted table by class name "table table-striped"
And all the field values. Out Time couldn't be extracted.
Second Test passed. By fixing the bug on backend.

2. **Admin Check**: Admin must be able to know complete information about entry/exit of any student by putting his details like roll number or user-id.

Test Owner: Kumar Gourav Songara

Test Date: 04/01/2023

Test Result: First and Second result Passed. Ran 2 Test in 12.646s.

Extracted table value after putting value in filter with class name "btn btn-primary". First value found with all other values. And Second value not found because value was not in the table.

3. **Status Check**: Students should be able to check their status like where they are going/coming from, time of going/coming etc. by logging in using their credentials.

Test Owner: Kumar Gourav Songara

Test Date: 04/01/2023

Test Result: First test failed. Ran 1 test in 7.348s.
Extracted table attributes value of out time and value was not present.

4. **Performance testing**: The system must be able to process many transactions and users in real-time, with minimal delays and errors.

Test owner: Lohit, Pratiksha , Ajeet , Danish

Test date: 04/01/2023

Test results: System can do multiple log in processes at a time

5. **System testing for safety and security system** : The system must be secure and protect against unauthorized access, data breaches, and other security threats.

Test owner: lohit

Test date:04/01/2023

Test results: System only give access to iitk students to log inor we can say the only users whose database is always known by system , only admin has access of backend data .

6. **Usability Check:** The system must be easy to use and navigate for both end-users and administrators .

Test owner: lohit

Test date: 04/01/2023

Test results: System was very easy to handle its simple structure makes it easy to use, log in process was easy, both bar code scanning and manual data entry was easy.

7. **Maintainability check :** The system must be easy to maintain and upgrade, with minimal disruption to operations.

Test owner: Pratiksha , Lohit

Test date :03/31/2023- 04/01/2023

Test results: while testing whenever we got errors it was easy to update the code and upgrade the system and also keeping data at backend with safety has been taken care of.

8. **Error Check**: Verify the system's ability to generate an error message when an invalid entry is made in the student information form.

Test Owner: Ajeet, Danish, Rikesh, Gaurav

Test Date: 04/10/2023

Test Result: The system consistently generated error messages for all invalid data entries, indicating that it has the ability to catch and alert users to input errors in the student information form.

Input case1: In log in module with invalid id or password during both admin and guard log in system showed error messages and till we won't click on ok button of error message won't allow us to go for next process

Input case 2: In case of system fails to recognise bar code it shows error message to scan again or try manual entry

Input case 3: In case of invalid roll number of student system shows error and in case of invalid ID of non-student system shows error and does not proceed further

Input case 4: In case of adding new admin or new guard whe we fetch backend database to check whether id belong to any guard and admin or not so system shows error message to add valid ID

9. **Dashboard statistics check:**

Test Owner: Rikesh, Gaurav, Ajeet

Test Date: 04/12/2023

Test Scenario:
Verify that system take test data for a given date range, including student IDs, entry/exit times, and location. Like non student and student differently and number of entries and number of exits differently
Verify that the system generates accurate statistics for the given date range, including the total number of entries and exits, the average time spent on campus, and the busiest entry/exit times.
Verify that the system can handle large datasets and does not slow down or crash during the testing process.
Test Results: All test scenarios were passed, and the system generated accurate statistics for the given test data and displays it on Admin Dashboard .

10. **Settings and Management Check:**

Test Owner: Ajeet, Danish, Rikesh, Gaurav

Test Date: 04/10/2023-04/13/2023

Test Result:
1st test: System allows admin to create and manage guard and admin accounts-PASS
2nd test: System allows to change admin or guard's login credentials and also allows admin to add new admin/guard-PASS
3rd test: Allows admin to get info on entry and exit daily statistics-PASS
4th test: Non-Student Registration: People who are not students or don't already have their data in the database should be able to register for the first time so that their data can be stored and their entry/exit can be easier -PASS
5th test: Manage History: Admin/Guard must be able to know complete information about entry/exit of any student by putting his details like roll number or user-id. Also they should be able to edit or delete any entry from the history table in case of wrong information.
6th test: Admin/Guard should be able to logout of the system-PASS

11. **Editing and Deleting check**: We must be able to delete and edit entry and exit in the history table.

Test Owner: Kumar Gourav Songara

Test date: 04/15/20223

Test Result: We are able to delete the entry and exit data for both student and non-student in history table and that is reflected in statistics page (admin-dashboard).

12. **Non-Students Registration:** We must be able to register the non-student in system with valid achaar card.

Test Owner: Kumar Gourav Songara

Test date: 04/16/2023
Test Result:
Case 1: Test Failed: On trying to register the non-student without valid adhaar number test failed.
Case 2: Test Passed: On trying to register the non-student with valid adhaar number test passed
Case 3: Test Failed: On trying to register the non-student without valid mobile number test failed.
Case 2: Test Passed: On trying to register the non-student with valid mobile number test passed.

13.Non-student Aadhar Card number security:

Test Owner: Kumar Gourav Songara

Test date: 04/16/2023

Test Result: Test Failed: User-id is used for for process which is modified adhaar number and we were able to extract Aadhaar number for it. So, system is not highly secure on front-end. But have security on back end.

14.Conclusion

The tests we performed were taking care of pretty much every function we used in our software system, and also testing a group of functions performing one task and also the whole system in general.

The team adopted a manual testing strategy mostly, so testing with extreme values was not possible. This makes the testing not very exhaustive. However, the testing has been done adequately for the values that are expected and in the expected range, and our team has tried to account for all actions possible by all the actors.

There were some functions which were not directly involved with the general working of the system but had some side uses, they were also working fine.

The testing of individual functions which we did during the implementation phase of the project was very helpful and effective as it helped removing bugs which could have proven really dangerous in the future. The testing also helped us make the system smoother as while testing we got the idea to change the backend functions such that they work much more efficiently than before.

All the components of the software were tested successfully, more components were added, and their testing was also performed successfully.

Since there was no automated tool readily available for us to easily implement in our software, the testing process was quite time-consuming.

The testing process could have been much better if there were some automated testing tools specifically for our software or software's like ours, but as we were not able to find any we had to resort to manual testing only.