# Test Document

for

# DBMS

Version 1.3

Prepared by

Group #: 5 Group Name: Dev Dynamos

| | | |
|---|---|---|
| Vishal Kumar | 221201 | vishalkmr22@iitk.ac.in |
| Yash Pratap Singh | 221223 | yashps22@iitk.ac.in |
| Shriya Garg | 221038 | shriyag22@iitk.ac.in |
| Aditya Gupta | 220065 | adityagu22@iitk.ac.in |
| Abhishek Kumar | 220041 | kumara22@iitk.ac.in |
| Nandini Akolkar | 220692 | anandini2222@iitk.ac.in |
| Rishikesh Sahil | 220892 | rishikeshs22@iitk.ac.in |
| Udbhav Singh Sikarwar | 221150 | udbhavss22@iitk.ac.in |
| Anshu Yadav | 220171 | anshuyadav22@iitk.ac.in |
| Kushagra Singh | 220572 | kushagras22@iitk.ac.in |

**Course:** CS253
**Mentor TA:** Bharat
**Date:**

## INDEX

## Revisions

| Version | Primary Author(s) | Description of Version | Date Completed |
|---|---|---|---|
| Draft Type and Number | Full Name | Information about the revision. This table does not need to be filled in whenever a document is touched, only when the version is being upgraded. | 01/04/24 |

# 1 Introduction

**Test Strategy:**

Our project, developed using the MERN stack, adopts Jest as the preferred framework for routes unit testing. Jest offers extensive support for JavaScript applications, ensuring thorough evaluation of individual components and functions. System-level testing is performed manually to assess overall functionality and user experience. The testing strategy combines automated and manual approaches, with specific parts automated for efficiency.

**When was the testing conducted?**

Testing occurred primarily post-implementation, with manual testing synchronized alongside development to identify and rectify issues promptly. Automated testing using Thunder Client was employed specifically for backend units to streamline the testing process.

**Who were the testers?**

Testers included both developers and independent testers. Developers took responsibility for manual testing of their respective components, while independent testers provided valuable insights and a fresh perspective on the application's functionality.

**Coverage Criteria:**

Our coverage criteria encompassed thorough manual testing, focusing on functional requirements coverage and user acceptance testing. Each aspect underwent meticulous evaluation to ensure the project's functionality and requirements were comprehensively addressed.

**Tools used for testing:**

.Jest served as the primary tool for unit testing backend routes, providing a robust framework for testing JavaScript code at the unit level. Automated testing of backend routes was also facilitated using Thunder Client during implementation. However, the majority of tests were conducted manually to ensure thorough evaluation of the application's functionality and user experience at the system level.

# 2 Unit Testing

## 1. Login Page

The page is rendered correctly and the unit is working as given in the table below.
**Unit Details:** [Function : Login() , verifyUser() , handleLogin()]
**Test Owner**: Vishal Kumar, Rishikesh Sahil
**Test Date**: 29/03/2024
**Test Results**:

| S.No. | Test | Unit(s) tested | Result |
|---|---|---|---|
| 1 | Entered correct credentials for a valid user | User Login | Successful Login |
| 2 | Entered wrong credentials | User Login | Does not not login and an alert displayed asking for entering correct user details. |

**Structural Coverage:[STATE] state :** Condition coverage : For userid[Phone number for washerman , roll no for student] , Password must match.

**Additional Comments**: The unit is working as expected.

## 2. Registration Page

The page is rendered correctly and the unit is working as given in the table below.
**Unit Details:** [Functions : RegisterStudent() , handleregister() ,sendOtpRequest() ,]
**Test Owner**: Vishal Kumar,Rishikesh Sahil
**Test Date**: 30/03/2024
**Test Results**: (Only for Student)

| S.No. | Test | Unit(s) tested | Result |
|---|---|---|---|
| 1. | Entered valid username,Roll-no, email-id, password ,selecting Hall and wing from the dropdown and registering as a student. | Student Registration<br>User Registration | OTP sent Successfully to the valid email provided. |

| | | | |
|---|---|---|---|
| 2. | Entered correct OTP | | Successful registration New user created in the backend. |
| 3. | The email should have a valid extension and it must be different from that of existing users. | | Successful |
| 4. | Passwords must be at least 6 characters long. | | Successful |

**Structural Coverage:Condition coverage :** For username not filled Password must match ;phone number,password and Email ID must be of the correct format.
**Additional Comments**: The unit works as expected

## 3. User Profile

The page is rendered correctly and the details of the Student including his name,photo,email id,hall,wing are correctly displayed.
**Unit Details:** [Functions : fetchRecord()]
**Test Owner**: Vishal Kumar,Aditya Gupta
**Test Date**: 30/03/2024
**Test Results**:

| S.No. | Test | Unit(s) tested | Result |
|---|---|---|---|
| 1 | The Student's Photo[ fetched from CC IITK],Email-id,Name,Hall, Wing should be displayed properly. | User Profile ,UI,(Frontend/Backend) | Successful |
| 2 | The washerman's Photo,Name,Contact-no should be displayed properly. | | Successful. |

**Structural Coverage:[State]** Branch Coverage:Checks the authenticity of the user.
**Additional Comments** : The unit is working as expected.

## 4. Payment Dues

The page is rendered correctly.
**Unit Details:** [Functions : paymentDates(),fetchDates()]
**Test Owner**: Vishal Kumar,Udbhav Sikarwar
**Test Date**: 30/03/2024
**Test Results**: The dues associated with all the Students is listed correctly.
**Structural Coverage**: **[State]** Condition coverage: Check for authentication of the user. Unless the user is an admin, he can't access the all dues list of another user. Additionally, while iterating over all transactions, if the status is pending, then only it is shown in all dues.
**Additional Comments**: The unit is working as expected.

## 5. User::Logout

The page is rendered correctly.
**Unit Details:** [Functions : logout() ,washerman logout()]
**Test Owner**: Vishal Kumar,Yash Pratap Singh
**Test Date**: 30/03/2024
**Test Results**: On clicking the logout button in the respective dashboard,successfully renders the landing Page.
**Structural Coverage**: **[State]** Condition coverage: Branch Coverage.
**Additional Comments**: The unit is working as expected.

## 6. Washerman::Add event

The page works as expected.
**Unit Details:** [Functions-Frontend : Date_Click_Fun() , Functions-Backend : upcomingDate() ]
**Test Owner**: Vishal Kumar,Yash Pratap Singh
**Test Date**: 30/03/2024
**Test Results**:
(i)On clicking the Add event button for a particular date,successfully leads to change in the color of that date in the dashboard.

(ii) The schema of the washerman contains a document - upcomingDate which get updated by the newly entered date.
**Structural Coverage**: **[State]** Condition coverage: Branch Coverage.
**Additional Comments**: The unit is working as expected.

## 7. Washerman::Print Summary

The page is rendered correctly.
**Unit Details:** [Functions :PrintSummary() ]
**Test Owner**: Vishal Kumar,Udbhav Singh Sikarwar
**Test Date**: 30/03/2024
**Test Results**: On clicking the print summary button displays the details of students ,clothes given till date and dues till date.
**Structural Coverage**: **[State]** Condition coverage: Branch Coverage.
**Additional Comments**: The unit is working as expected.

## 8. Washerman::Collect Clothes

The page is rendered correctly.
**Test Owner**: Udbhav Singh Sikarwar,Abhishek Kumar
**Test Date**: 30/03/2024
**Test Results**: On clicking the Collect clothes button,successfully renders the required Page.
**Structural Coverage**: **[State]** Condition coverage: Branch Coverage.
**Additional Comments**: The unit is working as expected.

## BACKEND TESTING USING JEST AND SUPERTEST

The following end-points are tested with supertest ,using the functions given below
The tests for session.t.js are as follows
**Unit Details**: admin Login
**/session/admin/login**
**Test Owner :** Yash Pratap Singh,Vishal Kumar

```
import app from '../../app.js';
import request from 'supertest';

describe('Testing Admin Login Route', () => {
    test('admin login with valid credentials', async () => {
        const req = {
            body: {
                username: process.env.ADMIN_USERNAME,
                password: process.env.ADMIN_PASSWORD
            }
        };

        const res = await
request(app).post('/session/admin/login').send(req.body);
        expect(res.status).toEqual(200);
        expect(res.body).toEqual({ message: 'Admin logged in successfully'
});
    });
```

Function for testing the **admin login with invalid credentials**

```
test('admin login with invalid credentials', async () => {
        const req = {
            body: {
                username: 'invalid_username',
                password: 'invalid_password'
            }
        };

        const res = await
request(app).post('/session/admin/login').send(req.body);
        expect(res.status).toEqual(401);
        expect(res.body).toEqual({ message: 'Invalid credentials' });
    });
});
```

**Unit Details**: Student Login
**/session/student/login/**
**Test Owner :** Yash Pratap Singh,Vishal Kumar

```
describe('Testing Student Login Route', () => {
    test('student login with valid credentials', async () => {
        const req = {
            body: {
                roll: '221223',
                pass: 'a1234567'
            }
        };

        const res = await
request(app).post('/session/student/login').send(req.body);
        expect(res.status).toEqual(200);
        expect(res.body).toEqual({ message: 'student logged in successfully'
});
    });
```

### Test for the invalid credentials while logging in by the student

```
test('student login with invalid credentials', async () => {
        const req = {
            body: {
                roll: '221223',
                pass: 'invalid_password'
            }
        };

        const res = await
request(app).post('/session/student/login').send(req.body);
        expect(res.status).toEqual(401);
        expect(res.body).toEqual({ message: 'Invalid credentials' });
    });
```

### Test for Student logging in with non-existing roll

```
test('student login with non-existing roll', async () => {
        const req = {
            body: {
                roll: 'non_existing_roll',
                pass: 'a1234567'
            }
        };

        const res = await
request(app).post('/session/student/login').send(req.body);
        expect(res.status).toEqual(401);
        expect(res.body).toEqual({ message: 'student not found' });
    });
});
```

**Unit Details**: Testing Washerman login route
**/session/washerman/login**

**Test Owner :** Yash Pratap Singh,Abhishek Kumar

```
describe('Testing Washerman Login Route', () => {
    test('washerman login with valid credentials', async () => {
        const req = {
            body: {
                contact: '1111111111',
                pass: 'washerman1'
            }
        };

        const res = await
request(app).post('/session/washerman/login').send(req.body);
        expect(res.status).toEqual(200);
        expect(res.body).toEqual({ message: 'Washerman logged in
successfully' });
    });
```

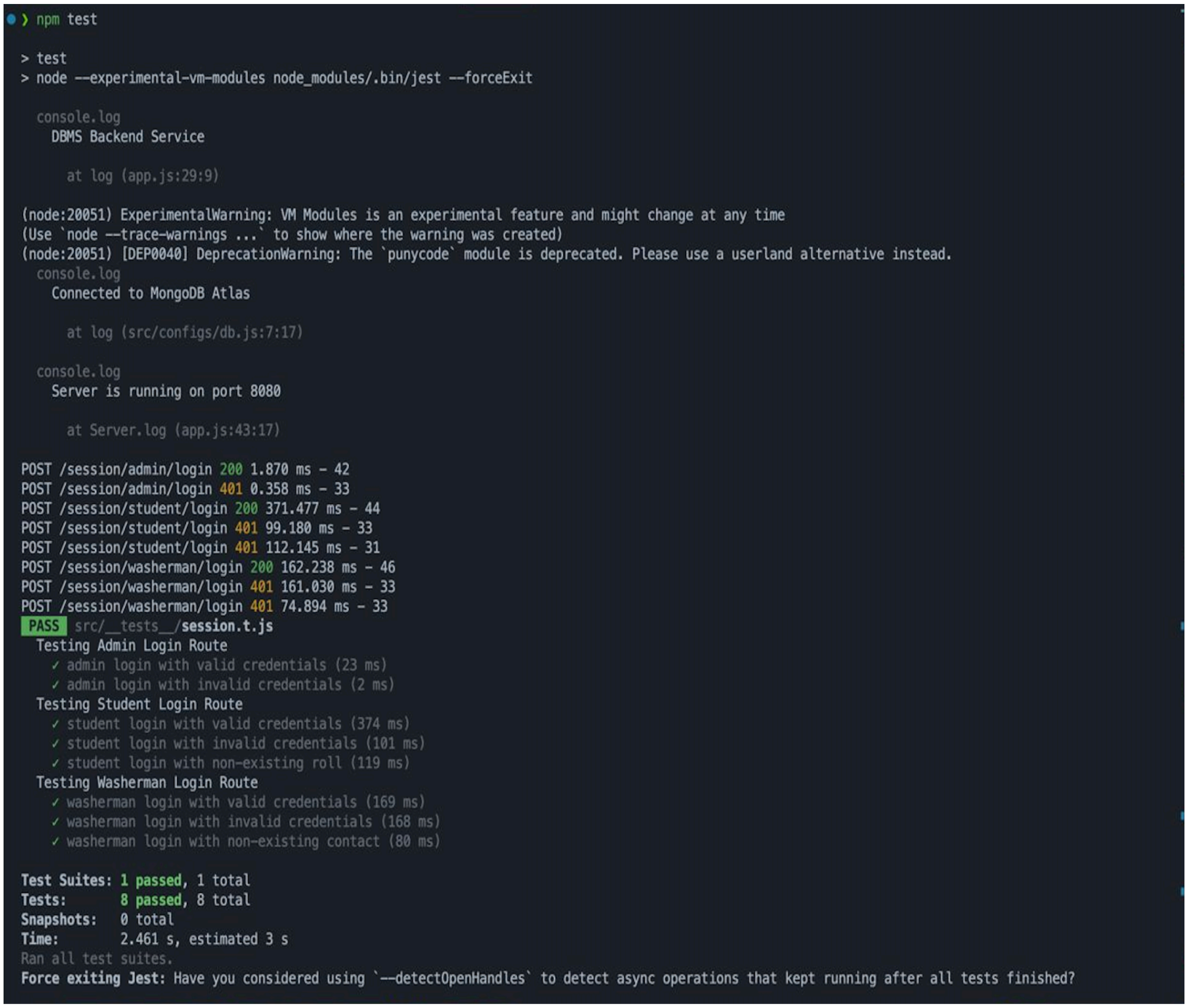
## Check function for washerman logging in with invalid credentials

```
test('washerman login with invalid credentials', async () => {
        const req = {
            body: {
                contact: 1111111111,
                pass: 'invalid_password'
            }
        };

        const res = await
request(app).post('/session/washerman/login').send(req.body);
        expect(res.status).toEqual(401);
        expect(res.body).toEqual({ message: 'Invalid credentials' });
    });
```

## Test function for washerman logging in with non-existing contact

```
test('washerman login with non-existing contact', async () => {
        const req = {
            body: {
                contact: 'non_existing_contact',
                pass: 'washerman1'
            }
        };

        const res = await
request(app).post('/session/washerman/login').send(req.body);
        expect(res.status).toEqual(401);
        expect(res.body).toEqual({ message: 'Washerman not found' });
    });
});
```

## Following are the results for the above tests



```
● ❯ npm test

> test
> node --experimental-vm-modules node_modules/.bin/jest --forceExit

  console.log
    DBMS Backend Service

      at log (app.js:29:9)

(node:20051) ExperimentalWarning: VM Modules is an experimental feature and might change at any time
(Use `node --trace-warnings ...` to show where the warning was created)
(node:20051) [DEP0040] DeprecationWarning: The `punycode` module is deprecated. Please use a userland alternative instead.
  console.log
    Connected to MongoDB Atlas

      at log (src/configs/db.js:7:17)

  console.log
    Server is running on port 8080

      at Server.log (app.js:43:17)

POST /session/admin/login 200 1.870 ms - 42
POST /session/admin/login 401 0.358 ms - 33
POST /session/student/login 200 371.477 ms - 44
POST /session/student/login 401 99.180 ms - 33
POST /session/student/login 401 112.145 ms - 31
POST /session/washerman/login 200 162.238 ms - 46
POST /session/washerman/login 401 161.030 ms - 33
POST /session/washerman/login 401 74.894 ms - 33
PASS src/__tests__/session.t.js
  Testing Admin Login Route
    ✓ admin login with valid credentials (23 ms)
    ✓ admin login with invalid credentials (2 ms)
  Testing Student Login Route
    ✓ student login with valid credentials (374 ms)
    ✓ student login with invalid credentials (101 ms)
    ✓ student login with non-existing roll (119 ms)
  Testing Washerman Login Route
    ✓ washerman login with valid credentials (169 ms)
    ✓ washerman login with invalid credentials (168 ms)
    ✓ washerman login with non-existing contact (80 ms)

Test Suites: 1 passed, 1 total
Tests:       8 passed, 8 total
Snapshots:   0 total
Time:        2.461 s, estimated 3 s
Ran all test suites.
Force exiting Jest: Have you considered using `--detectOpenHandles` to detect async operations that kept running after all tests finished?
```

**Following are the test cases for admin.test.js**

```
import app from '../../app.js';
import request from 'supertest';

const adminLogin = async () => {
    const loginDetails = {
        username: process.env.ADMIN_USERNAME,
        password: process.env.ADMIN_PASSWORD
    };

    const res = await request(app)
        .post('/session/admin/login')
        .send(loginDetails);

    // Extract token from Set-Cookie header
    const cookieHeader = res.headers['set-cookie'];
    const token = extractTokenFromCookieHeader(cookieHeader);
    console.log(token)
    return token;
};
```

## Helper Function to extract token from cookie header

```
const extractTokenFromCookieHeader = (cookieHeader) => {
    if (!cookieHeader) return null;

    const tokenCookie = cookieHeader.find(cookie =>
cookie.startsWith('token='));
    if (!tokenCookie) return null;

    const token = tokenCookie.split(';')[0].split('=')[1];
    return token;
};
```

```
let adminToken;

beforeAll(async () => {
    adminToken = await adminLogin();
});
```

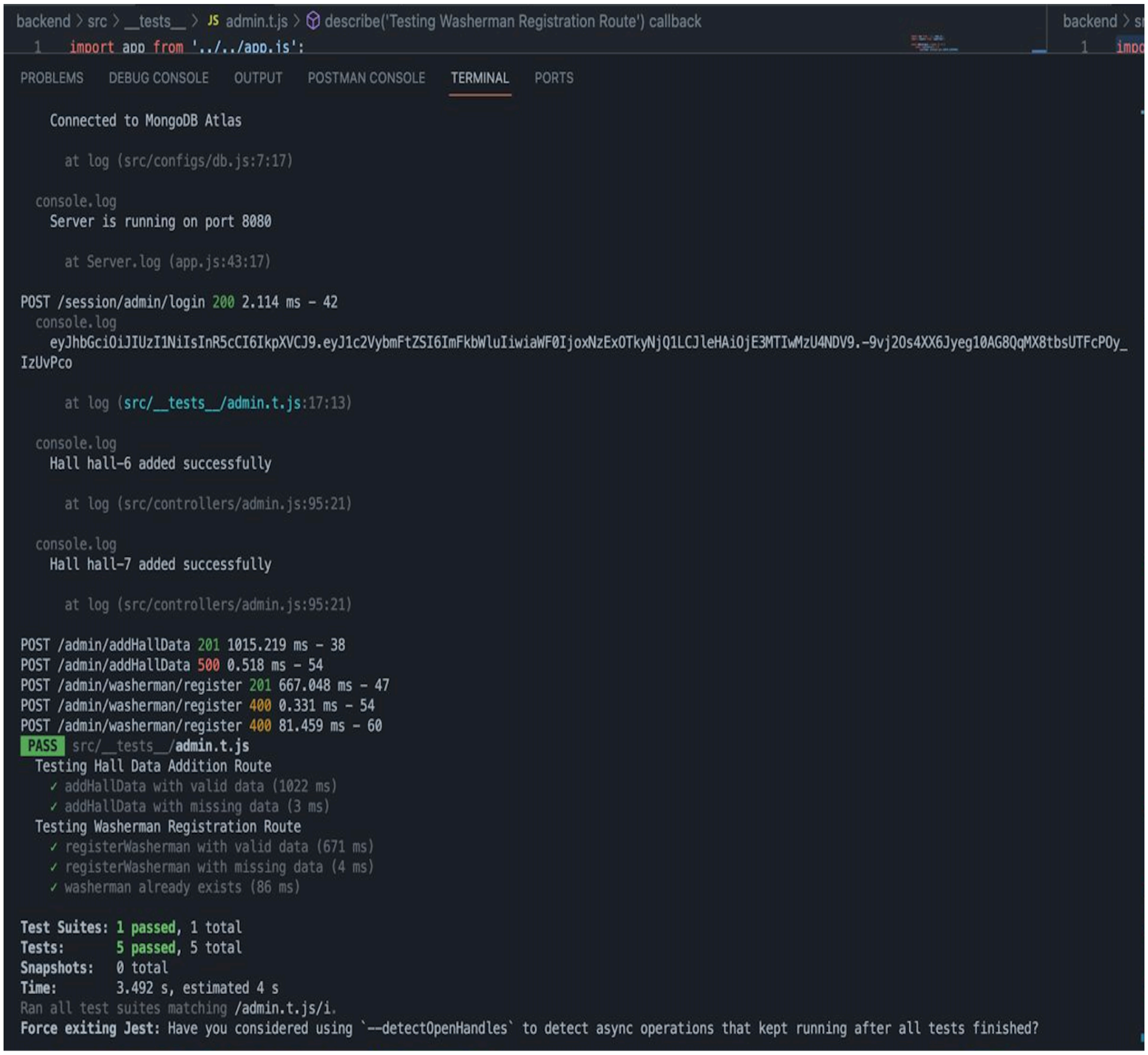
## Function for testing Hall Data Addition Routes

```


describe('Testing Hall Data Addition Route', () => {
    test('addHallData with valid data', async () => {
        const reqBody = {
            Halls: [
                { hallName: 'hall-6', wings: ['wing-x', 'wing-y'] },
                { hallName: 'hall-7', wings: ['wing-x', 'wing-y', 'wing-z']
}
            ]
        };

        const res = await request(app)
            .post('/admin/addHallData')
            .set('Cookie', `token=${adminToken}`)
            .send(reqBody);

        expect(res.status).toEqual(201);
        expect(res.body).toEqual({ message: 'Halls added successfully' });
    });

    test('addHallData with missing data', async () => {
        const reqBody = {
            // Missing required fields
        };

        const res = await request(app)
            .post('/admin/addHallData')
            .set('Cookie', `token=${adminToken}`)
            .send(reqBody);

        expect(res.status).toEqual(500);
        expect(res.body).toEqual({ message: 'Bad Request (Wrong/Missing Keys
in json)' });
    });
});
```

## Testing washeman Register Routes

```
describe('Testing Washerman Registration Route', () => {
    test('registerWasherman with valid data', async () => {
        const reqBody = {
            contact: '6666666666',
            name: 'washerman6',
            pass: 'washerman6',
            halls: [
                { name: 'hall-6', wings: ['wing-x', 'wing-y'] },
                { name: 'hall-7', wings: ['wing-z'] }
            ],
            accountID: 'razorPayID6'
        };

        const res = await request(app)
            .post('/admin/washerman/register')
            .set('Cookie', `token=${adminToken}`)
            .send(reqBody);

        expect(res.status).toEqual(201);
        expect(res.body).toEqual({ message: 'Washerman registered
successfully' });
    });

    test('registerWasherman with missing data', async () => {
        const reqBody = {
            // Missing required fields
        };
        const res = await request(app)
            .post('/admin/washerman/register')
            .set('Cookie', `token=${adminToken}`)
            .send(reqBody);

        expect(res.status).toEqual(400);
        expect(res.body).toEqual({ message: 'Bad Request (Wrong/Missing Keys
in JSON)' });
    });

    test('washerman already exists', async () => {
        const reqBody = {
            contact: '6666666666',
            name: 'washerman6',
            pass: 'washerman6',
            halls: [
                { name: 'hall-6', wings: ['wing-x', 'wing-y'] },
                { name: 'hall-7', wings: ['wing-z'] }
            ],
            accountID: 'razorPayID6'
        };

        const res = await request(app)
            .post('/admin/washerman/register')
            .set('Cookie', `token=${adminToken}`)
            .send(reqBody);

        expect(res.status).toEqual(400);
        expect(res.body).toEqual({ message: 'Washerman with the same contact
already exists' });
    });
});
```

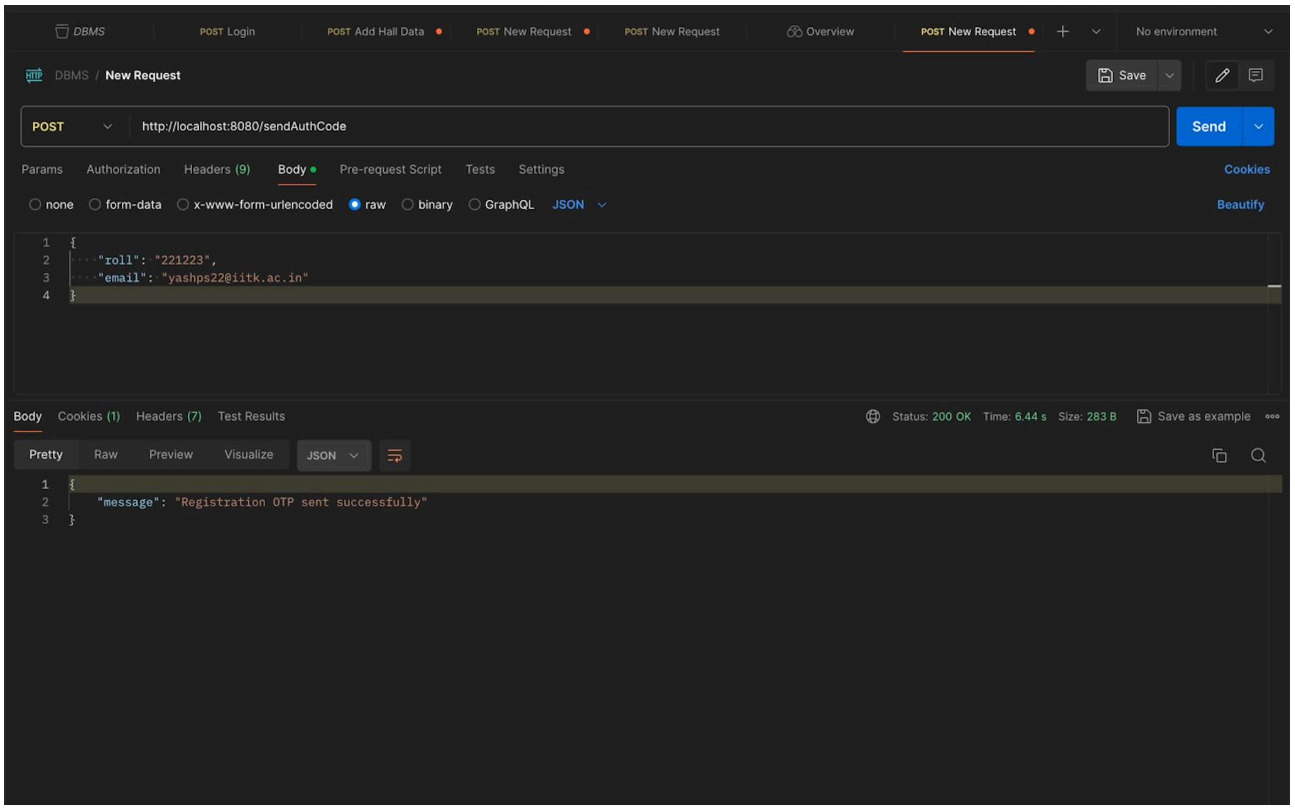
## Following are the results for the above tests



backend > src > __tests__ > admin.t.js > describe('Testing Washerman Registration Route') callback
PROBLEMS DEBUG CONSOLE OUTPUT POSTMAN CONSOLE TERMINAL PORTS

```
import app from '../../app.js';
```

```
    Connected to MongoDB Atlas

      at log (src/configs/db.js:7:17)

  console.log
    Server is running on port 8080

      at Server.log (app.js:43:17)

POST /session/admin/login 200 2.114 ms - 42
  console.log
    eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImFkbWluIiwiaWF0IjoxNzExOTkyNjQ1LCJleHAiOjE3MTIwMzU4NDV9.-9vj2Os4XX6Jyeg10AG8QqMX8tbsUTFcPOy_
IzUvPco

      at log (src/__tests__/admin.t.js:17:13)

  console.log
    Hall hall-6 added successfully

      at log (src/controllers/admin.js:95:21)

  console.log
    Hall hall-7 added successfully

      at log (src/controllers/admin.js:95:21)

POST /admin/addHallData 201 1015.219 ms - 38
POST /admin/addHallData 500 0.518 ms - 54
POST /admin/washerman/register 201 667.048 ms - 47
POST /admin/washerman/register 400 0.331 ms - 54
POST /admin/washerman/register 400 81.459 ms - 60
 PASS  src/__tests__/admin.t.js
  Testing Hall Data Addition Route
    ✓ addHallData with valid data (1022 ms)
    ✓ addHallData with missing data (3 ms)
  Testing Washerman Registration Route
    ✓ registerWasherman with valid data (671 ms)
    ✓ registerWasherman with missing data (4 ms)
    ✓ washerman already exists (86 ms)

Test Suites: 1 passed, 1 total
Tests:       5 passed, 5 total
Snapshots:   0 total
Time:        3.492 s, estimated 4 s
Ran all test suites matching /admin.t.js/i.
Force exiting Jest: Have you considered using `--detectOpenHandles` to detect async operations that kept running after all tests finished?
```



POST http://localhost:8080/sendAuthCode

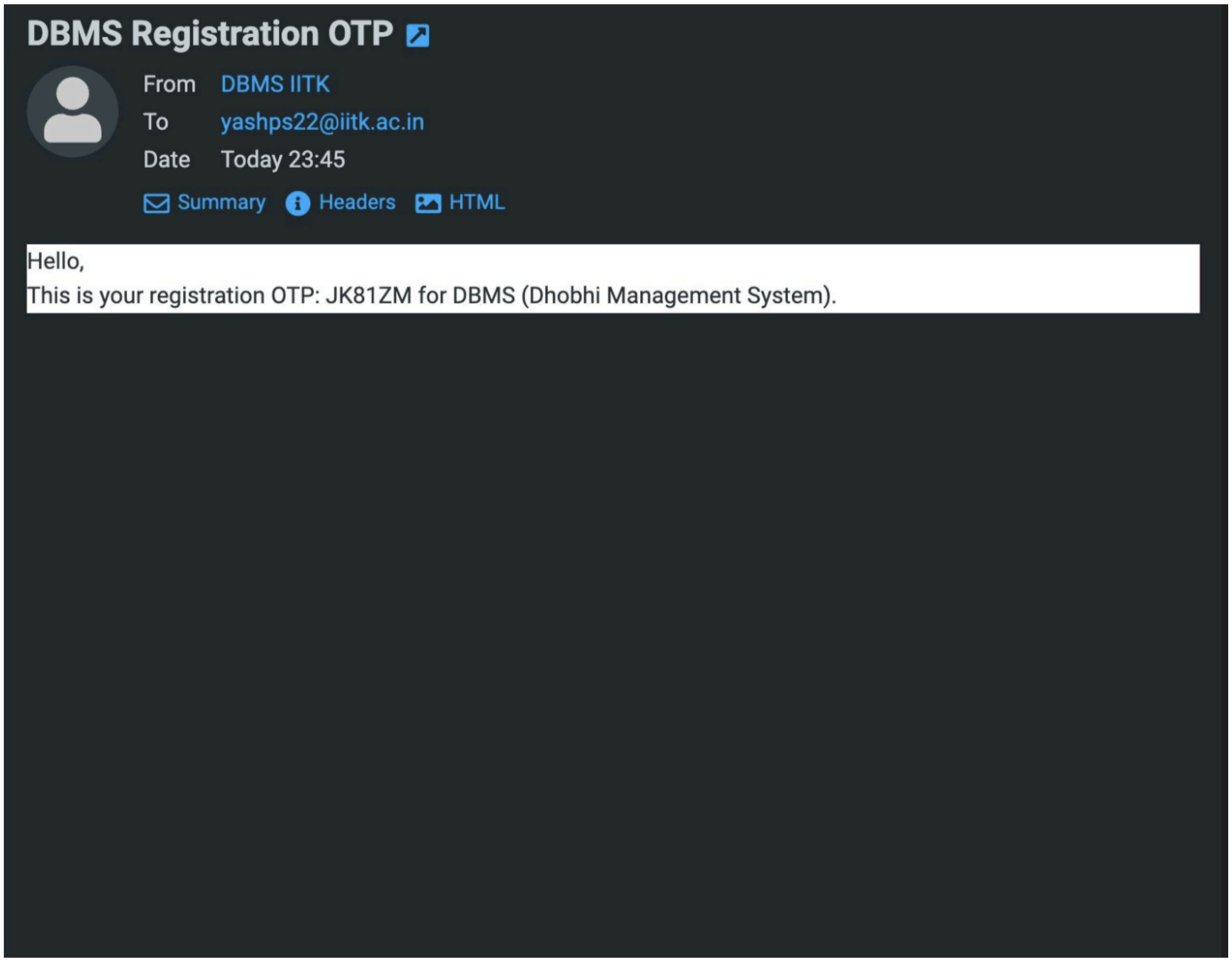
Body: none form-data x-www-form-urlencoded [✓ raw] binary GraphQL JSON

```
{
    "roll": "221223",
    "email": "yashps22@iitk.ac.in"
}
```

Status: 200 OK Time: 6.44 s Size: 283 B

JSON

```
{
    "message": "Registration OTP sent successfully"
}
```



DBMS Registration OTP

From DBMS IITK
To yashps22@iitk.ac.in
Date Today 23:45

Hello,
This is your registration OTP: JK81ZM for DBMS (Dhobhi Management System).

# 3 Integration Testing

**For the user being Student**

**Checking the interface between login and Dashboard for Student**

## 1.Authorization (Registration/Login)

**Tested functioning of new user registration and login on the platform**

**Module Details:** The module contains all the basic details of the class Student ,the necessary functions for the authentication of the student while logging in. register(),studentLogin()

**Test Owner**: Vishal Kumar , Rishikesh Sahil
**Test Date**: 28/03/2024
**Test Results**:

| S.NO. | TEST | TEST COMPONENT | RESULT |
|---|---|---|---|
| 1. | From “Login” page entered correct credentials for a valid student | student logging in | Successful login and redirected to dashboard page for student. |
| 2. | From “Login” page entered wrong credentials or credentials that don’t exists | | Does not login and an alert is generated for entering wrong credentials. |
| 3. | From the “Login” Page if the user is not registered,Clicks on “Register here”. | Login and Register page. | Successfully redirected to the Register page. |
| 4. | On the registration Page ,entering correct details and clicking on the “Get OTP” button sends OTP on the registered email. | | Successfully received OTP on the email id entered. |
| 5. | If you haven't | | Successfully sends |

| | | | |
|---|---|---|---|
| | received the OTP, click the "Resend OTP" button. | | OTP on the registered email. |
| 6. | After entering the OTP received on email , Click the ‘Register’ button. | | Alert box (‘Registration Successful’) ,Successfully redirects to the login Page. |
| 7. | From the "Login" page, if you've forgotten your password, click on "Create New Password." | | Successfully redirects to the verification page for creating a new password. |
| 8. | After writing username and roll no. , click on “Generate OTP”. | Verification page. | successfully sends OTP on the email and redirects to another page to enter OTP and create a new password. |
| 9. | Enter OTP and write a new password and write the same password on the “Confirm New Password”. | | This successfully changed Password and redirects you to the Login Page. |

## 2. Student Dashboard

**Tested Student Dashboard and overall functionalities**.

**Module Details:** The module contains all the necessary functions for wash request ,fetching dates of the clothes given, fetch receipt and payment dues. requestWash() ,fetchDates() ,clearDue() ,paymentDates() ,fetchReceipt() ,fetchRecord.

**Test Owner**: Vishal Kumar , Rishikesh Sahil
**Test Date**: 28/03/2024
**Test Results**:

| S.NO. | TEST | TEST COMPONENT | RESULT |
|---|---|---|---|
| 1. | Student clicks on a particular date on the calendar on which he gave clothes. | Dashboard Calendar | Successful mini view of the summary of the clothes just like a dropdown below that date. |
| 2. | For clothes submission ,Student clicks on the date of submission. | | Color changes appear on that particular date. |
| 3. | For a particular date, click on the “Wash clothes ” button. | Wash Clothes and Pay dues functionalities | Successfully redirected to the WashClothes Page.. |
| 4. | Clicking on the “Pay Dues” button. | | Successfully redirects to Pay dues Page. |
| 5. | Doing Payment using the third party “RazorPay”. | Payment through RazorPay | Amount debited from student account and credited to washerman account successfully. |
| 6. | Clicking on the “Logout” button the profile section. | Logout functionality | Successfully logouts the students and redirects to the landing page. |

## 3. Wash Clothes

**Tested Wash clothes Page..**

**Module Details:** The module contains all the necessary functions for washing clothes WashClothes() , incrementCounter() ,decrementCounter().

**Test Owner**: Vishal Kumar , Rishikesh Sahil
**Test Date**: 28/03/2024
**Test Results**:
….

| S.NO. | TEST | TEST COMPONENT | RESULT |
|---|---|---|---|
| 1. | For increasing the total number of clothes, click on the increment "+" button. Similarly, for decreasing the total number of clothes, click on the decrement "-" button. | Wash Clothes /Clothes submission page. | The section for "total clothes" , "Total cost" is accordingly updated successfully. |
| 2. | Clicking on "wash clothes" button. | | Sends a notification for washing clothes to the corresponding allotted washerman. |

**For the user being Washerman**

**Checking the interface between login and Dashboard for Washerman**

### 1.Washerman Login

**Module Details:** The module contains all the basic details of the Washerman class,the necessary functions for the authentication of the washerman while logging in that are Contact number and password.
washermanLogin()
**Test Owner**: Anshu Yadav, Abhishek Kumar
**Test Date**: 28/03/2024
**Test Results**:

| S.NO. | TEST | TEST COMPONENT | RESULT |
|---|---|---|---|
| 1. | From "Login" page entered correct credentials for a valid Washerman | correct login functionality | Successful login and redirected to dashboard page for washerman |
| 2. | From "Login" page entered wrong credentials or credentials that don't exists | For invalid credentials in Logging page | Does not login and an alert is generated for entering wrong credentials. |

### 2.Washerman Dashboard

**Tested washerman Dashboard's overall functionalities.**
**Module Details:** The module contains all the necessary functionalities
**Test Owner**: Vishal Kumar , Abhishek Kumar
**Test Date**: 28/03/2024
**Test Results**:

| S.NO. | TEST | TEST COMPONENT | RESULT |
|---|---|---|---|
| 1. | After logging in with correct contact number and password if if the washerman do not select any date for the event addition using “add event” | Events on Washerman Dashboard | Nothing will be append to the event list that's we want |
| 2. | On Trying to add blank event by selecting a particular date and clicking on “add event” | Events handling on Washerman Dashboard | Nothing will happen to the event list that's we expect |
| 3. | On the dashboard by clicking on the “Print Summary” button | printing summary on Washerman Dashboard | Successfully redirected to the page , where dropdowns of that allotted hall appear. Within each dropdown there is a list of rooms with total number of clothes given till date and total dues. |
| 4. | On the click the button “Collect Clothes” | Clothes collection feature | “Accept” or “deny” option will appear for students giving their cloths with detail of cloths item for verification |
| 5. | “Notification” button click | Notification view for the events | Visiting events by washerman will be shown |

| | | | |
|---|---|---|---|
| 6. | On clicking “Logout” button | logout functionality of Dashboard | Successfully landing page opened |

# 4 System Testing

1. **Requirement:** 3.2.1: Account creation and profile details input for students:

**Test Owner**: Shriya
**Test Date**: 28/03/2024
**Test Results**:

| S.No | Test | Frontend response | APIs called | Backend Response |
|---|---|---|---|---|
| 1. | Upon selecting the student and clicking on “Don't have an account? Register Here”. | We are redirected to a registration form, and prompted to enter the details. | Get request sent to backend at the API endpoint: ${process.env.REACT_APP_BACKEND_URL}/student/session/register` | record of the student will be stored in the database |
| 2. | Upon entering correct details in the registration page and clicking on get OTP. Enter the correct OTP. | The account is created and you are redirected to the login page. | Get request sent to backend at the API endpoint: ${process.env.REACT_APP_BACKEND_URL}/session/sendAuthCode | OTP is temporarily stored in database |
| 3. | Upon clicking, “Login here” on the registration page. | We are redirected to the login page in case of correct credentials. | Get request sent to backend at the API endpoint: ${process.env.REACT_APP_BACKEND_URL}/se | verification of the credential will done |

| | | | | |
|---|---|---|---|---|
| | | | ssion/student/login | |
| 4. | The user details are already existing. | An alert box is displayed. | Get request sent to backend at the API endpoint: ${process.env.REACT_APP_BACKEND_URL}/session/student/login | verification of the records will be done for prexisting and login page will be redirected |
| 5. | Password entered does not match or is not at least 6 characters long | Appropriate error message is displayed. | Get request sent to backend at the API endpoint: ${process.env.REACT_APP_BACKEND_URL}/session/student/login | Verification of the password is done for the record |

**Additional Comments**: Student registration is done according to the mentioned details in the SRS.But here anyone can register using their email id

2. **Requirement:3.2.2.**Student login via email/phone number:

**Test Owner**:Kushagra, Anshu Yadav
**Test Date**:29/03/24

**Test Results**:

| S.No | Test | Frontend response | APIs called | Backend Response |
|---|---|---|---|---|
| 1. | Login for student | Upon entering the correct credentials redirected to the dashboard | Get request sent to backend at the API endpoint: ${process.env.REACT_APP_BACKEND_URL/session/student/login} | verification of the data |
| 2. | Entering incorrect roll number or password | Alert box shown with “invalid Credentials” | Get request sent to backend at the API endpoint: ${process.env.REACT_APP_BACKEND_URL}/session/student/login | verification of the data |
| 3. | For “Forgot Password” and creating new password | Redirect to the resetPassword page for taking email and roll number as input then it takes the new password and the otp as input for the setting the new password | Get request sent to backend at the API endpoint: ${process.env.REACT_APP_BACKEND_URL}/sendAuthCode | Password for the corresponding roll no.,email will get changed to the newpassword if the otp and other credentials are entered correctly |

**Additional Comments**:Anyone can login by knowing the password and the email of a person there is not guaranteed by us on the right user.

3. **Requirement:** 3.2.3: Student Home page with a calendar showing total laundry submitted.

**Test Owner**: Yash pratap singh, Kushagra Singh

**Test Date**: 30/03/2024
**Test Results**:

| S.No | Test | Frontend response | APIs called | Backend Response |
|---|---|---|---|---|
| 1. | Dashboard view | The dashboard correctly displays the user information, along with a calendar view. | No API call was sent. | No change in backend |
| 2. | Upon clicking “Wash Clothes” button | We are redirected to a wash clothes page. | No API call was sent. | If no cloths item are selected then unchanged backend else if some cloths item are selected then changes may occur in backend |
|  | Upon adding/subtracting a cloth of a particular type | The total cost and the number of clothes are modified accordingly. | No API call was sent. | No change in backend |
| 3. | On clicking wash, after adding clothes. | We are redirected back to the dashboard. | API will be called which sends the notification to washerman for accepting or denying the wash requests | data stored regarding the due for the student will be update |
| 4. | Upon clicking the pay dues button . | the summary of the total dues of that student is displayed | Razorpay API call was sent | Due Detail will be shown for the student and then will be updated |

| | | | | |
|---|---|---|---|---|
| 5. | Upon clicking Log-Out Button. | We are redirected back to the landing page | Get request sent to backend at the API endpoint: ${process.env.REACT_APP_BACKEND_URL}/session/logout | No change in backend |

**Additional Comments**:The student home page with the calendar is working as specified in the initial documentation.

**4. Requirement:** 3.2.4: Testing the functionality of the calendar on student dashboard
**Test Owner**: Vishal, Nandini
**Test Date**: 30/03/24
**Test Results:**

| S.no. | Test | frontend response | API called | Backend response |
|---|---|---|---|---|
| 1. | Upon clicking a particular date on the calendar which has been marked | The summary of the clothes given on that day is displayed. | No API call was sent. | washerman events added will be on user calendar |

**Additional Comments**: This feature has been slightly modified as to what was mentioned in the SRS, with respect to the colors displayed on the dashboard.

**5. Requirement:** 3.2.6: Student Payment
**Test Owner**:  Nandini, Shriya
**Test Date**: 31/03/2024

| S.no. | Test | frontend response | API called | Backend response |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| 1. | Upon clicking pay dues. | We are redirected to the payment breakdown page which displays the breakdown of the total payment. | No API call was sent. | Amount due will be shown from backend data |
| 2. | Upon clicking Pay Via UPI | We are directed to razor-pay(Third party payment platform) and proceed from there. | RazorPay call will be sent | No change in response from backend |
| 3. | Upon clicking pay via cash. | The dues are shownA to be pending unless the washerman verifies them. | No API call was sent. | No change in backend |

**Additional Comments**:Here we have are assuming smooth transaction for cash payment and updation in the database will be done same as in the UPI

6. **Requirement:** 3.2.7.1: Washerman login page

**Test Owner**: Nandini, Shriya
**Test Date**: 31/03/2024

| S.no. | Test | frontend response | API called | Backend response |
|---|---|---|---|---|
| 1. | On entering the correct details | redirected to washerman dashboard with calendar | Get request sent to backend at the API endpoint: ${process.env.REACT_APP_BACKEND_URL}/session/washerman/login | Verifies the data |
| 2. | Entering incorrect mobile number or password | Alert prompt showing "Invalid Credentials." | Get request sent to backend at the API endpoint: ${process.env. | verifies the data |

| | | | | |
|---|---|---|---|---|
| | | | REACT_APP_BACKEND_URL}/session/washerman/logi | |

**Additional Comments**: Here we have implemented the correct static credentials not dynamic like verification via OTP

7. **Requirement:** 3.2.7.2: Washerman Dashboard with Calendar

**Test Owner**: Nandini, Shriya
**Test Date**: 31/03/2024

| **S.no.** | **Test** | **Frontend response** | **APIs called** | **Backend response** |
|---|---|---|---|---|
| 1. | On clicking the Logout button | the user is redirected to the Landing page | Get request sent to backend at the API endpoin: ${process.env.REACT_APP_BACKEND_URL}/session/logout | No change in backend |
| 2. | on selecting a date from the calender and clicking "ADD EVENT" in the create event dialog box | in the events dialog box, the visiting date with the option to delete the changes is displayed | Get request sent to backend at the API endpoin: ${process.env.REACT_APP_BACKEND_URL}/s | Data stored in Database and and in student |

| | | | | |
|---|---|---|---|---|
| | | | ession/washerman/upcomingDate | |
| 3. | on clicking “DELETE” in the events dialog box | that particular event is deleted from the list of all events | Get request sent to backend at the API endpoin: ${process.env.REACT_APP_BACKEND_URL}/session/washerman/upcomingDate | Data will be deleted from the database ,so from student dashboard |

**Additional Comments**:The washerman dashboard is implemented not exactly as mentioned in the SRS as we do not restricted to only 6 months past, like one can visit any month and can add as many event can want for a date may be repetitive

**8. Requirement:** 3.2.8:Summary for washerman
**Test Owner**: Nandini, Shriya
**Test Date**: 31/03/2024
**Test Results:**

| **S.no.** | **Test** | **frontend response** | **API called** | **Backend response** |
|---|---|---|---|---|
| 1. | on clicking the button “PRINT SUMMARY” | the summary of all the collected clothes of all students (wing wise) will be displayed | /washerman/printsummary/ | |

**Additional Comments**: This feature has been implemented somewhat similarly as mentioned in the requirement document.

9. **Requirement:** 3.2.9: Accepting or denying requests

**Test Owner**: Nandini, Shriya
**Test Date**: 31/03/2024
**Test Results:**

| S.no. | Test | frontend response | API called | Backend response |
|---|---|---|---|---|
| 1. | on clicking the button "COLLECT CLOTHES" | details of each student can be entered. | Get request sent to backend at the API endpoint: ${process.env.REACT_APP_BACKEND_URL}/session//student/fetchRecord | Data for the halls and wings allotted will be done from database |
| 2. | Upon the details of the student. | "accept" or "deny" option will appear for students giving their cloths with detail of cloths item for verification | Get request sent to backend at the API endpoint: ${process.env.REACT_APP_BACKEND_URL}/session/student/requestWash | Data of the student cloths will be shown at the washerman end for accepting or denying verification |

**Additional Comments**: Here washerman seen the details and then accept or deny we rely that Washerman do genuine acceptance or denial for request

10. **Requirement:** 3.2.8:Notification

**Test Owner**: Nandini, Shriya
**Test Date**: 31/03/2024
**Test Results:**

| S.no. | Test | frontend response | API called | Backend response |
|---|---|---|---|---|
| 1. | on clicking the button "Notification" | Visiting events by washerman will be shown | | |

**Additional Comments**: This feature has not been implemented as mentioned in the requirement document.

# 5 Conclusion

### Effectiveness of Testing Process

The testing process relied on automated as well as  manual execution, limiting the possibility of achieving exhaustive coverage.
Despite this constraint, the team dedicated efforts to address a wide range of anticipated scenarios that the system and its components might face.
Additionally, they strived to consider actions from various potential users of the system, ensuring a thorough evaluation of its functionality.

Manual testing techniques were the primary approach for system-level evaluation, ensuring a thorough assessment of the application's functionality and user experience. Additionally, **backend units were tested automatically using JEST** [a javascript framework], which facilitated efficient testing of backend routes, contributing to overall testing coverage and efficiency.

### Limitations in testing

All components have undergone testing to some extent, ensuring a baseline level of quality assurance. However, there are few areas in system testing where maximum exhaustion has not been achieved.

### Difficulty Faced

Manual testing of the software was a meticulous and resource-intensive endeavor. Each testing phase demanded considerable time and labor investment as testers meticulously executed test cases, observed system behavior, and documented results. Despite the challenges of manual testing, this approach helped us in a thorough examination of the software's functionality and user experience.

### Enhancement in Testing process

The integration of a separate testing team, distinct from the development team, would have mitigated potential biases introduced during testing. Such biases overlook critical aspects that may not be apparent to those deeply involved in the software's creation. Although structural coverage testing would have provided a more comprehensive evaluation, time limitations necessitated prioritizing functional testing instead.

# Appendix A - Group Log

| SL no. | Date | Timings | Venue | Description |
|---|---|---|---|---|
| **1** | 19 March 2024 | 2 pm -4pm | RM building | We discussed how to do the testing process for the website and distributed the work among us. |
| **2** | 20 March 2024 | 9pm -10pm | RM building | Met the TA and shared our plan and took his suggestions. |
| **3** | 22 March 2024 | 4pm-7pm | RM building | Started unit testing and user manually parallelly. |
| **4** | 24 March 2024 | 9pm -11:30pm | Google Meet | Completed unit testing and started integration testing. |
| **5** | 26 March 2024 | 8pm-10 pm | Google Meet | Finalised integration testing and started system testing. |
| **6** | 29 March 2024 | 9:30pm-10 pm | Google Meet | User manual and system testing continued. |
| **7** | 31 March 2024 | 6pm-9pm | RM building | Completed system testing and finalised user manual. |
| **8** | 01 April 2024 | 6pm-8pm | RM building | Verified the documents and submitted them. |