Testing Document

8.1 Testing Strategy

End-to-End Testing:

- Using **Selenium WebDriver** to simulate real user interactions like navigating the login page, entering credentials, and checking page transitions.
- **UI and Functional Testing** for elements like login buttons, error messages, and successful login redirects.
- **Screenshots** are taken during test execution for visual verification.

Unit Testing:

- Using **JSDOM** to simulate the **Dashboard** page's express structure, allowing for testing of static content without browser interaction.
- Verifies page titles, header content, buttons, and sections (like the tasks section) directly within the express DOM.

Dependencies:

- **Mocha**: Test framework for running and organizing tests.
- **Chai**: Assertion library for validating test results.
- **Selenium WebDriver**: Browser automation tool for E2E testing.
- **JSDOM**: Simulates a browser environment for unit testing of static HTML content.
- **fs (File System)**: For handling screenshots and reading HTML content for unit tests.
-

8.2 Test Plan

1. **Login Page Tests**:
   - Check if the login page loads correctly.
   - Validate error messages for invalid credentials.
   - Verify successful login with correct credentials and redirection.
2. **Dashboard Page Tests**:
   - Validate the page title, header content, and footer information.
   - Verify the presence of specific UI elements like the logout button, dashboard
   - links, tasks section, and messages (like "No tasks assigned yet.

8.3 Test Cases

Login Page Tests:

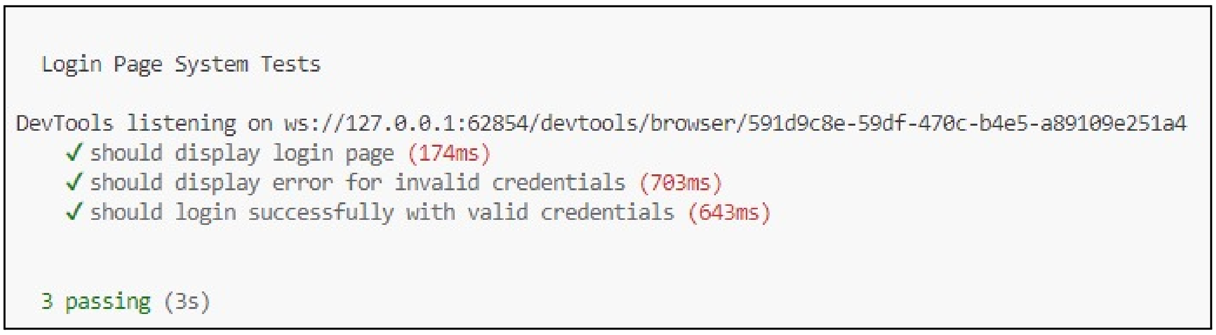
1. **Page Load**:
   - Check if the login page title is "Login".
   - Screenshot of the page is taken.
2. **Invalid Login**:
   - Enter invalid credentials and check for the error message "Invalid email or password".
   - Screenshot after submitting invalid credentials.
3. **Valid Login**:
   - Enter valid credentials and check if the user is redirected to the manager dashboard.
   - Screenshot of the dashboard after successful login.

Dashboard Page Tests:

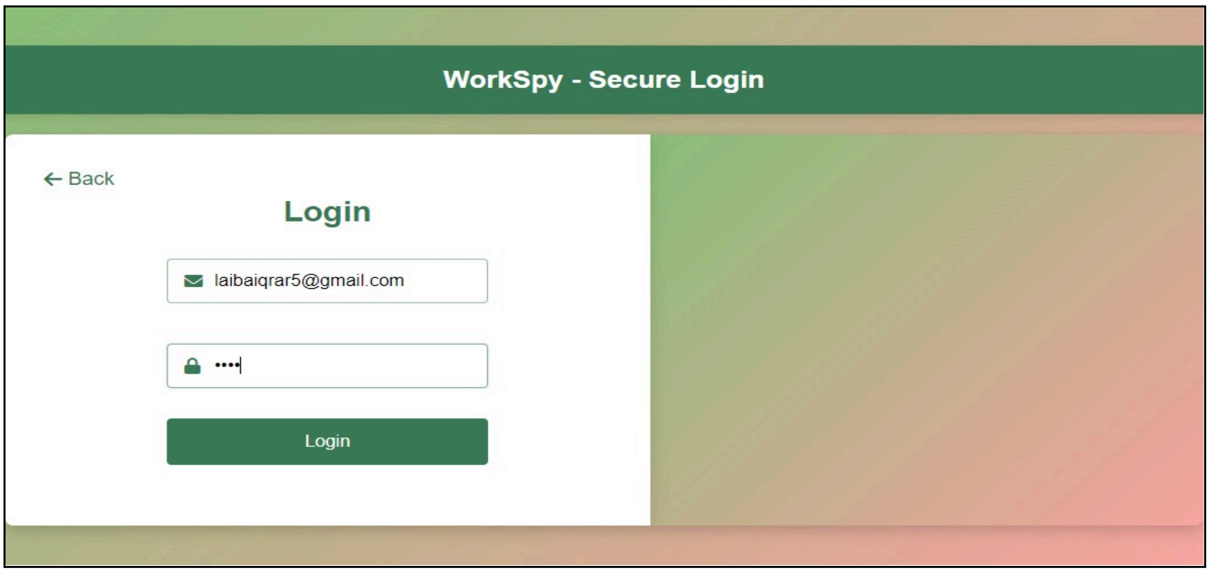
1. **Title Check**:
   - Validate the page title is "Employee Dashboard".
2. **Header Content**:
   - Verify the header contains the title "WORKSPY" and the correct slogan.
3. **Logout Button**:
   - Ensure the logout button is present and contains the text "Logout".
4. **Dashboard Links**:
   - Check if there are exactly 4 dashboard links.
5. **Tasks Section**:
   - Ensure the tasks section is present and displays the correct message when no tasks are assigned.
6. **Footer**:
   - Verify that the footer contains the copyright message.

8.4 Test Results

- Login Page Testing and screenshot from chromedriver

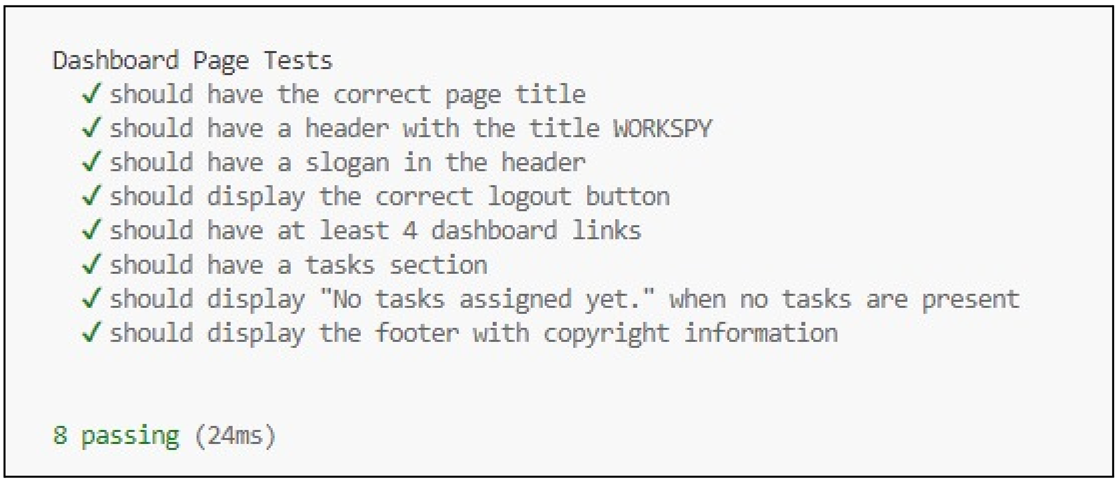


```
  Login Page System Tests

DevTools listening on ws://127.0.0.1:62854/devtools/browser/591d9c8e-59df-470c-b4e5-a89109e251a4
    ✓ should display login page (174ms)
    ✓ should display error for invalid credentials (703ms)
    ✓ should login successfully with valid credentials (643ms)


  3 passing (3s)
```

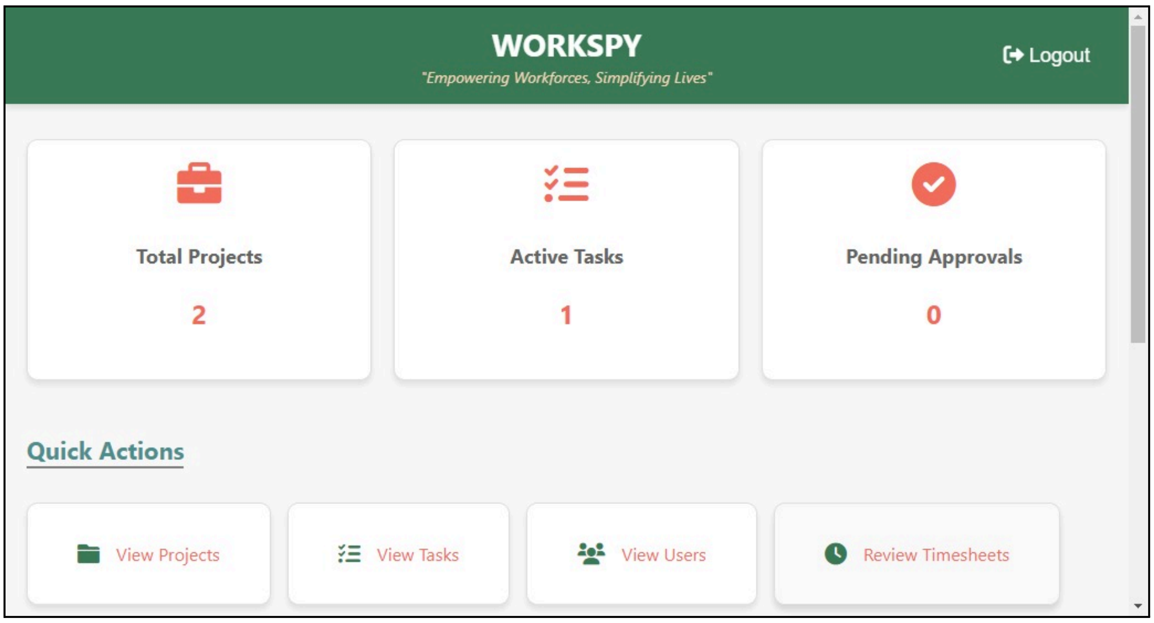


8.4 Test Results

- Landing page Testing and screenshot from chromedriver



```
Dashboard Page Tests
  ✓ should have the correct page title
  ✓ should have a header with the title WORKSPY
  ✓ should have a slogan in the header
  ✓ should display the correct logout button
  ✓ should have at least 4 dashboard links
  ✓ should have a tasks section
  ✓ should display "No tasks assigned yet." when no tasks are present
  ✓ should display the footer with copyright information


8 passing (24ms)
```

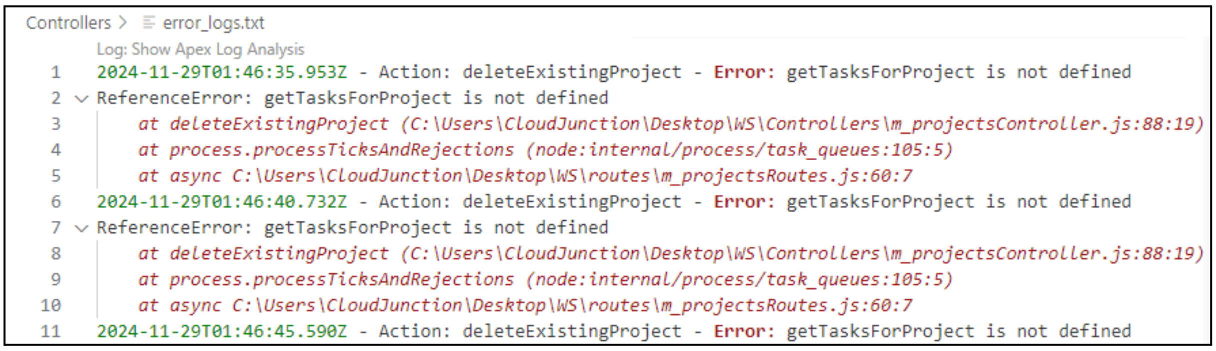


8.5 Bug Tracking

- **Error Logging:**

When an error occurs (e.g., in project deletion or any other function), it's caught in a try-catch block.

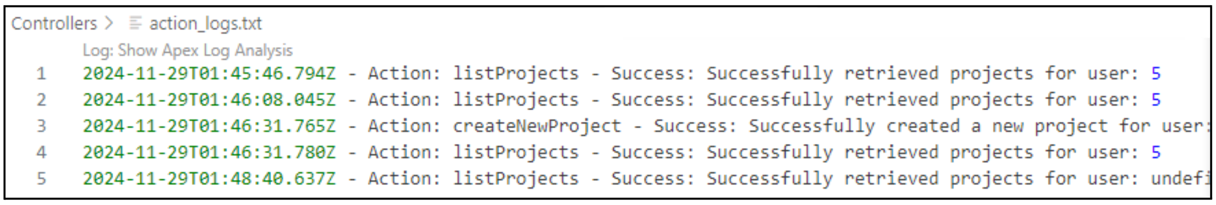
The error details (message and stack trace) are logged in a error_logs.txt file, along with the action that triggered the error (e.g., deleteExistingProject).



```
2024-11-29T01:46:35.953Z - Action: deleteExistingProject - Error: getTasksForProject is not defined
ReferenceError: getTasksForProject is not defined
    at deleteExistingProject (C:\Users\CloudJunction\Desktop\WS\Controllers\m_projectsController.js:88:19)
    at process.processTicksAndRejections (node:internal/process/task_queues:105:5)
    at async C:\Users\CloudJunction\Desktop\WS\routes\m_projectsRoutes.js:60:7
2024-11-29T01:46:40.732Z - Action: deleteExistingProject - Error: getTasksForProject is not defined
ReferenceError: getTasksForProject is not defined
    at deleteExistingProject (C:\Users\CloudJunction\Desktop\WS\Controllers\m_projectsController.js:88:19)
    at process.processTicksAndRejections (node:internal/process/task_queues:105:5)
    at async C:\Users\CloudJunction\Desktop\WS\routes\m_projectsRoutes.js:60:7
2024-11-29T01:46:45.590Z - Action: deleteExistingProject - Error: getTasksForProject is not defined
```

Success Logging:

When operations succeed (e.g., project creation or deletion), a success message with relevant details is logged in a action_logs.txt file.



```
2024-11-29T01:45:46.794Z - Action: listProjects - Success: Successfully retrieved projects for user: 5
2024-11-29T01:46:08.045Z - Action: listProjects - Success: Successfully retrieved projects for user: 5
2024-11-29T01:46:31.765Z - Action: createNewProject - Success: Successfully created a new project for user:
2024-11-29T01:46:31.780Z - Action: listProjects - Success: Successfully retrieved projects for user: 5
2024-11-29T01:48:40.637Z - Action: listProjects - Success: Successfully retrieved projects for user: undefi
```