**Test Execution Report for FinanceMan Dashboard**

#### **1. Test Execution Summary**

The test execution for the FinanceMan Dashboard was performed to ensure that the features and functionalities work as expected, as outlined in the test plan. The testing was executed on the URL:<https://finance-man.netlify.app/dashboard>. All test cases were executed on the latest versions of Google Chrome, Mozilla Firefox, and Microsoft Edge browsers, with additional tests for responsiveness across desktop, tablet, and mobile devices.

The following components were tested:

* Dashboard page load
* Header title and branding
* Sidebar menu items
* Financial data display
* Transaction history
* Search functionality
* Pagination for transactions
* Logout functionality
* Dashboard responsiveness
* Error handling for invalid input

**2. Test Execution Results**

| **Test Case ID** | **Test Case Type** | **Test Case Description** | **Pre-Requisites** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **Status** | **Comments** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| TC\_001 | Functionality | Verify Dashboard page loads successfully | User is logged in | 1. Open the URL:<https://finance-man.netlify.app/dashboard>.  2. Check if the page loads without errors. | N/A | The page should load without any errors and show the dashboard. | The page loads successfully without errors, and the dashboard is displayed successfully. | Pass | No issues encountered. |
| TC\_002 | Functionality | Verify the header displays the correct title and branding. | Dashboard is loaded | 1. Once the dashboard loads, check the header for the correct branding and page title (e.g., "Finance Dashboard").elements. | N/A | The header should display the correct branding and title of the dashboard. | The header displays "Finance Dashboard" with the appropriate branding. | Pass | Header displayed as expected. |
| TC\_003 | Functionality | Verify Sidebar Menu Items. | User is logged in | | 1. After the page loads, check if the sidebar menu is displayed.  2. Verify that all expected menu items are present (e.g., Dashboard, Transactions, Settings, etc.). | | --- |  |  | | --- | | N/A | |  | | --- |  | Sidebar should have all menu items as per design. | | --- | | The sidebar contains items like Dashboard, Transactions, Settings, and others. | Pass | Sidebar menu loaded correctly. |
| TC\_004 | Functionality | |  | | --- |  | Verify the dashboard displays correct financial data | | --- | | HUser is logged in with existing financial data | 1. Log into the dashboard.  2. Check that the financial data (e.g., balance, expenses, income) displayed is correct and corresponds to the user data. | Sample user data | The displayed financial data should match the expected values. | The financial data displayed is accurate and corresponds to the user's account. | Pass | Financial data verified successfully. |
| TC\_005 | Functionality | Verify Transaction History is displayed correctly | User has transactions | 1. After logging in, navigate to the transaction history section.  2. Verify that the transaction history is populated with the correct data (e.g., amounts, dates, and types). | Example transactions | The transaction history should show accurate data. | The transaction history displays correct transactions, including amounts and dates. | Pass | Transaction history displayed as expected. |
| TC\_006 | Functionality | Verify Search Functionality for Transactions. | User has transactions | 1. Click on the search bar in the transaction section.  2. Type a keyword (e.g., "Rent") and verify if the filtered results show matching transactions. | "Rent" transaction | The search should return transactions matching the search keyword. | The search functionality works correctly and returns matching results. | Pass | Search functionality is working as expected. |
| TC\_007 | Functionality | Verify Pagination for Transactions | User has more than 10 transactions | 1. Navigate to the transaction history section.  2. Verify that the pagination controls (Next/Previous) are visible and functional.  3. Click through the pages and ensure correct data is displayed. | More than 10 transactions | Pagination should work correctly, showing 10 items per page. | Pagination controls work, showing 10 transactions per page. | Pass | Pagination verified successfully. |
| TC\_008 | Functionality | Verify Logout functionality | User is logged in | 1. Click on the logout button from the dashboard.  2. Verify that the user is logged out and redirected to the login page. | N/A | |  | | --- |  | After clicking logout, the user should be logged out and redirected to the login screen. | | --- | | User is logged out and redirected to the login page.. | Pass | Logout functionality works as expected. |
| TC\_009 | Functionality | Verify Dashboard Responsiveness on Different Screen Sizes | User is logged in | 1. Open the dashboard on various devices or screen sizes (mobile, tablet, desktop).  2. Verify that the dashboard is responsive and adjusts its layout accordingly. | N/A | The dashboard layout should be responsive and correctly display on different screen sizes. | The dashboard adjusts its layout correctly on all screen sizes. | Pass | Responsiveness validated on all devices. |
| TC\_010 | Negative | Verify Error Message for Invalid Input | User is logged in | 1. Try submitting a form (e.g., add a transaction) with invalid data (e.g., empty fields, negative amount).  2. Verify if an appropriate error message is shown. | Invalid data (e.g., empty fields) | The system should show an error message indicating that the input is invalid. | An error message is shown for invalid data, indicating the issue (e.g., missing fields). | Pass | Error handling works as expected. |
| TC\_011 | Functionality | Verify that the Delete, Save, No Sort, Sort by Date, and Sort by Amount buttons are functioning properly. | 1. The application is installed and running.  2. User is logged into the application (if authentication is required).  3. Relevant data (such as records to delete or sort) is available. | 1. Navigate to the page containing the "Delete", "Save", "No Sort", "Sort by Date", and "Sort by Amount" buttons.  2. Click the "Delete" button on a record.  3. Click the "Save" button after making a change.  4. Click the "No Sort" button.  5. Click the "Sort by Date" button.  6. Click the "Sort by Amount" button. | 1. Sample records with varying dates and amounts for sorting functionality.  2. A record that can be deleted and edited for testing the Delete and Save buttons. | 1. The "Delete" button should remove the selected record.  2. The "Save" button should save any changes made to the record.  3. The "No Sort" button should return the list to its default order (if previously sorted).  4. The "Sort by Date" button should sort records by date in ascending or descending order.  5. The "Sort by Amount" button should sort records by the amount field in ascending or descending order. | All the buttons are showing bugs and not working as expected.  The Delete, Save, No Sort, Sort by Date, and Sort by Amount buttons do not trigger any action or display an error message. | Fail | Not works as expected. |

#### **3. Overall Test Execution Summary**

* **Total Test Cases Executed**: 11
* **Pass**: 10
* **Fail**: 1
* **Blocked**: 0
* **Not Executed**: 0

The test execution was completed without any critical issues. All the test cases passed successfully. The application worked as expected in all tested areas, including functional features like the dashboard page load, sidebar, transaction history, search functionality, and logout process. Additionally, the responsiveness of the application across multiple devices was validated.

#### **4. Conclusion**

* The FinanceMan Dashboard has successfully passed 90% of all the functional and negative test cases. The page loads correctly, all features (sidebar, search, transaction history, pagination, etc.) perform as expected, and the dashboard is responsive across different screen sizes. Error handling for invalid input was also verified and works as expected.
* Since 90% of all test cases passed, the application is not ready for production deployment. There were 1 - defects or blockers identified during this testing cycle.

#### **5. Next Steps**

* **Deployment to Production**: Since 90% all test cases passed, But the application cannot be ready for deployment to production.
* **Regression Testing**: In case of future updates or changes, regression testing should be performed to ensure that existing functionalities remain unaffected.
* **User Acceptance Testing (UAT)**: After deployment, user acceptance testing should be conducted to verify that the application meets end-user expectations.

#### **6. Approval**

| **Name** | **Role** | **Date** | **Signature** |
| --- | --- | --- | --- |
| Stephen Chitaranjan | Test Lead | 2024-12-13 | Stephen Chitaranjan |
| Biju | Project Manager | 2024-12-13 | Biju |