Manual Test Plan for Shortest Route Optimizer

Application Under Test: Shortest Route Optimizer

Purpose: To verify the functional behavior of the “Shortest Route Optimizer application”.

Scope: This test plan covers the core functionalities, UI responsiveness, and data integrity of the application.

Test Environment:

- Browser: Google Chrome (latest version), Firefox.
- OS: Windows 10.
- Network: Stable internet connection.

Requirement Analysis:

The Shortest Route Optimizer application aims to calculate the shortest path between two nodes within a connected graph using Dijkstra's Algorithm. The application provides two input modes: manual selection of nodes and automatic selection via an external API for random node generation. The results should be visually displayed to the user and logged to the browser console, as well as sent to an HTTP echo API.

Key functionalities include selecting FROM and TO nodes, refreshing random nodes, calculating the shortest path, and clearing selections. The user interface must adhere to the V1 design specification, maintain responsiveness across different screen sizes, and ensure error-free API integration. Additionally, the system must handle unidirectional paths and display results in a structured Data Transfer Object (DTO) format in the console.

Testing must cover the functional behavior, UI responsiveness, and API interactions, ensuring accurate path calculations, proper element display, and correct console outputs.

Approach:

Testing will be divided into specific modules:

- Functional Testing: Verifying individual features.
- UI Testing: Ensuring the design and responsiveness align with the specification.
- API Testing: Confirming the correctness of API responses and backend integration.
- Console Output Testing: Verifying that expected data is correctly logged to the browser console.

Test Cases Overview:

1. Random Mode Selector
2. Refreshing the Random Nodes
3. Selection of From/To Nodes
4. Calculating the Path
5. Display of Visual Results
6. Clearing the Current Selection
7. Adherence to the V1 UI Design Specification
8. Viewport Scaling Responsiveness of UI
9. API Echo Response OK
10. Browser Console Output of Result

Test Cases and Scenarios:

Test Case ID	Test Case Name	Objective	Precondition	Steps	Expected Result	Pass Criteria
TC-01	Random Mode Selector	Verify that the random mode selector works correctly.	Application is loaded successfully.	1. Navigate to the application. 2. Click on the "Random Mode" button.	The application should select random FROM and TO nodes automatically.	Random nodes are selected without user input.

Test Case ID	Test Case Name	Objective	Precondition	Steps	Expected Result	Pass Criteria
TC-02	Refreshing the Random Nodes	Verify the refresh functionality for random nodes.	Random Mode is active with nodes selected.	1. Click the "Refresh" button for random nodes.	The selected nodes should refresh, and new random nodes should be displayed.	Nodes refresh successfully on each click.

Test Case ID	Test Case Name	Objective	Precondition	Steps	Expected Result	Pass Criteria
TC-03	Selection of From/To Nodes	Verify manual selection of FROM and TO nodes.	Application is loaded, and Random Mode is off.	1. Click on the dropdown for FROM node. 2. Select a node. 3. Repeat for the TO node.	The selected nodes should appear correctly.	Nodes are correctly selectable and displayed.

Test Case ID	Test Case Name	Objective	Precondition	Steps	Expected Result	Pass Criteria
TC-04	Calculating the Path	Verify the shortest path calculation.	FROM and TO nodes are selected.	1. Click the "Calculate" button.	The shortest path between the selected nodes is calculated and displayed.	Correct path and distance are displayed based on nodes..

Test Case ID	Test Case Name	Objective	Precondition	Steps	Expected Result	Pass Criteria
TC-05	Display of Visual Results	Verify the display of visual results..	Path calculation is completed.	1. Observe the visual representation of the calculated path.	The path should be visually represented on the UI with connecting nodes.	Path display matches calculated results.

Test Case ID	Test Case Name	Objective	Precondition	Steps	Expected Result	Pass Criteria
TC-06	Clearing the Current Selection	Verify the clearing of node selections and results.	FROM and TO nodes are selected with a calculated path.	1. Click the "Clear" button.	All selections and results should be cleared, and return the app to the initial state.	Selections are reset without errors.

Test Case ID	Test Case Name	Objective	Precondition	Steps	Expected Result	Pass Criteria
TC-07	Adherence to the V1 UI Design Spec	Verify UI elements adhere to the V1 design.	Application is loaded.	1. Compare UI against the V1 design mockup.	UI should match the design specification (layout, colors, button placements).	No discrepancies between the application UI and design.

Test Case ID	Test Case Name	Objective	Precondition	Steps	Expected Result	Pass Criteria
TC-08	Viewport Scaling Responsiveness	Verify the application's responsiveness to different screen sizes.	Application is loaded.	1. Resize the browser window to different dimensions (desktop, tablet, mobile).	UI elements should adjust and maintain usability across different viewports.	UI elements should adjust and maintain usability across different viewports.

Test Case ID	Test Case Name	Objective	Precondition	Steps	Expected Result	Pass Criteria
TC-09	API Echo Response OK	Verify that the API echo response returns OK.	A calculation has been completed, and results are ready to send.	1. Trigger the result DTO to be sent to the HTTP echo API.	The API should return an OK response.	Response is captured in the browser console as OK.

Test Case ID	Test Case Name	Objective	Precondition	Steps	Expected Result	Pass Criteria
TC-10	Browser Console Output of Result	Verify correct data is outputted to the browser console.	Path calculation is complete.	1. Check the browser console for the output message.	Correct DTO structure and values (nodeNames and distance) are logged in the console.	Console displays the correct data format and values.

Test Case ID	Test Case Name	Objective	Precondition	Steps	Expected Result	Pass Criteria
TC-11	Input Validation	Verify that the application validates user inputs correctly.	Application is loaded, Random Mode is off.	1. Attempt to submit the form without selecting FROM and TO nodes. 2. Try entering invalid data via developer tools or form manipulation.	The application should prompt the user to select valid nodes and prevent form submission with invalid inputs.	The application should prompt the user to select valid nodes and prevent form submission with invalid inputs.

Test Case ID	Test Case Name	Objective	Precondition	Steps	Expected Result	Pass Criteria
TC-12	Data Persistence after Refresh	Verify that the application handles browser refreshes correctly..	FROM and TO nodes are selected, and a path is calculated.	1. Refresh the browser page. 2. Observe whether selections and results persist or reset.	The application should either retain the state or reset to the initial state gracefully.	The application should reset to the initial state.

Tools and Technologies Used

Selenium WebDriver (Java):

Selenium WebDriver is utilized for automating browser interactions and conducting end-to-end testing of the web application. Java is used as the programming language for writing the test scripts.

Maven Dependencies:

Maven is used for project build automation and managing dependencies. Key dependencies include:

- Selenium WebDriver for browser automation.
- TestNG for test management and execution.
- Additional dependencies as needed for project-specific requirements (e.g., Selenide).

Test Data Selection for Automation

The following nodes are used for the From-Nodes and To-Nodes selections in the test cases:

- Available Nodes: A, B, C, D, E, F, G, I
- Excluded Node: H
- Rationale for Exclusion:

Node H was excluded from the test data because it does not return the expected total distance value during automation testing.

Reporting

For reporting, TestNG's built-in reporting libraries have been utilized. After executing the TestRunner.xml, a detailed HTML report is automatically generated and saved in the designated /report folder.

Key Details:

- Report Format: HTML
- Location: /report folder
- Trigger: Generated upon execution of the TestRunner.xml file

This report provides a comprehensive overview of the test execution results, including pass/fail status, exceptions, and execution times.

Exit Criteria

The testing process will be considered complete when the following conditions are met:

- **Test Script Execution:**

    A. All test scripts have been executed without any critical issues or errors.

    B. No significant defects or blockers remain unresolved that would impede the execution or validation of the test scripts.

- **Pass/Fail Status**:

    A. A majority of test cases must pass, and any failures are reviewed and addressed according to the severity and priority defined in the Test Plan.

Test Coverage:

Adequate test coverage is achieved, and all critical paths and functionalities are tested as per the requirements and acceptance criteria.