**Feature: Dashboard Functionality**

**Scenario: Load Dashboard**

Given the user accesses the dashboard page

When the dashboard is loaded

Then the dashboard should display without errors

**Scenario: Verify Visualization Accuracy**

Given the dashboard is displayed

When the user interacts with visualizations on the dashboard

Then the visualizations should accurately represent the underlying data

**Scenario: Test Widget Interactions**

Given the dashboard contains interactive widgets

When the user interacts with the widgets

Then the widgets should respond correctly to user actions

**Scenario: Navigate Between Dashboards**

Given the user is on the dashboard page

When the user navigates to another dashboard

Then the user should be able to switch between dashboards

**Scenario: Test Responsive Design**

Given the user accesses the dashboard page

When the dashboard is viewed on different devices

Then the layout should adjust correctly based on the screen size

**Scenario: Apply Data Filters**

Given the dashboard contains filterable data

When the user applies filters

Then the dashboard should display data based on the applied filters

**Scenario: Customize Dashboard**

Given the dashboard is displayed

When the user customizes visualizations and layout

Then the changes should be reflected in the dashboard

**Scenario: Handle Errors**

Given the dashboard encounters an error scenario

When the error occurs

Then the system should display an appropriate error message

**Scenario: Test Sharing and Collaboration**

Given the dashboard contains shared elements

When users collaborate and share the dashboard

Then shared elements should be accessible to authorized users

**Scenario: Test Accessibility**

Given the dashboard is loaded

When users navigate using keyboard and screen readers

Then all interactive elements should be accessible and usable

**Feature: Dashboard API Functionality**

**Scenario: Retrieve Dashboard List**

When the API endpoint to retrieve dashboards is called

Then the response should contain a list of available dashboards

**Scenario: Create Dashboard**

When the API endpoint to create a new dashboard is called

Then a new dashboard should be created and accessible through the API

**Scenario: Update Dashboard**

When the API endpoint to update an existing dashboard is called

Then the changes made through the API should be reflected in the dashboard

**Scenario: Delete Dashboard**

When the API endpoint to delete a dashboard is called

Then the dashboard should be removed from the list of available dashboards

**Scenario: Retrieve Visualization Data**

When the API endpoint to retrieve visualization data is called

Then the response should contain the correct data for the visualization

**Scenario: Share Dashboard via API**

When the API endpoint to share a dashboard is called

Then the dashboard should be accessible to authorized users specified in the API request

**Scenario: Apply Filters via API**

When the API endpoint to apply filters to a dashboard is called

Then the response should contain data filtered based on the applied filters

**Scenario: Customize Widget via API**

When the API endpoint to customize widgets is called

Then changes made through the API should be reflected in the dashboard

**Scenario: Manage Collaboration via API**

When API endpoints related to dashboard comments and annotations are called

Then comments and annotations should be displayed correctly in the dashboard

**Scenario: Handle API Errors**

When API endpoints receive invalid requests

Then the API should return appropriate error codes and messages