Test Plan

1. Objective

This test plan outlines the approach to testing the core functionalities of a mobile e-commerce application. The application allows users to browse products, view product details, add products to the cart, and remove products from the cart. The goal is to ensure that these functionalities work as expected across both iOS and Android platforms.

2. Scope

- Browse a list of products.
- View product details.
- Add a product to the cart.
- Remove a product from the cart.

3. Key Scenarios to be Tested

Functional Scenarios

TC-001: Viewing the Product List

- Description: Ensure that the product list is displayed correctly with all products and their details.
- Steps:
  - Open the app.
  - Navigate to any collection.
  - Navigate to the product list page.
  - Verify the list of products.
- Expected Result: The product list should display all products with their correct names and prices.

TC-002: Viewing Product Details

- Description: Ensure that the product details page displays the correct information.
- Steps:
  - Open the app.
  - Navigate to the product list page.
  - Select a product to view its details.
  - Verify the product details (name, price, description).
- Expected Result: The product details page should display the correct product information, and the "Add to Cart" button should be present.

TC-003: Adding a Product to the Cart

- Description: Ensure that a product can be added to the cart and the cart updates correctly.
- Steps:
  - Open the app.
  - Navigate to the product list page.
  - Select a product and view its details.
  - Click "Add to Cart."
  - Navigate to the cart page.
- Expected Result: The product should be added to the cart, and the cart summary should reflect the correct product and price.

TC-004: Removing a Product from the Cart

- Description: Ensure that a product can be removed from the cart and the cart updates correctly.
- Steps:
  - Open the app.
  - Add a product to the cart.
  - Navigate to the cart page.
  - Click "Remove" for the product.
- Expected Result: The product should be removed from the cart, and the cart summary should update accordingly.

Edge Cases

TC-005: Adding More Quantity of the Same Product

- Description: Ensure that the app properly updates the quantity of the same product in the cart when the user adds more of it.
- Steps:
  - Open the app and navigate to the product list page.
  - Select a product to add to the cart.
  - Add the same product again to the cart to increase the quantity.
  - Check the cart for the updated product quantity.
- Expected Result: The app should correctly update the quantity of the product in the cart and display the new total quantity.

TC-006: Adding Multiple Products to the Cart

- Description: Ensure that multiple products can be added to the cart and the cart summary updates correctly.
- Steps:
  - Open the app.
  - Add multiple products to the cart.

    - Navigate to the cart page.
- Expected Result: All added products should be listed in the cart, and the cart summary should reflect the correct total price.

TC-007: Removing the Only Product in the Cart

- Description: Ensure that the app handles the removal of the last product in the cart gracefully.
- Steps:
    - Open the app.
    - Add a product to the cart.
    - Navigate to the cart page.
    - Remove the product.
- Expected Result: The cart should display an appropriate message (e.g., "Your cart is empty").

TC-008: Network Issues

- ID: TC-008
- Description: Ensure that the app handles network issues gracefully.
- Steps:
    - Simulate a network issue (e.g., no internet connection).
    - Open the app and attempt to browse products.
- Expected Result: The app should display an appropriate error message (e.g., "No internet connection").

4. Assumptions

- The app is stable and functional for basic navigation.
- The app supports both iOS and Android platforms.
- The app has been tested manually for basic functionality before automation and has no known bugs affecting the core functionalities listed.
- The test environment (simulators/emulators) is pre-configured and ready for use.

5. High-Level Approach to Testing

Manual Testing

- Initial exploratory testing will be conducted to understand the app's behavior and identify any obvious issues.
- Manual testing will be used to verify edge cases and scenarios that are difficult to automate.
- Perform basic nonfunctional testing such as accessibility, usability, security, and performance.

Automated Testing

- Automated tests will be written using a mobile testing framework: Appium
- The focus will be on automating the core functionalities:
  - Viewing the product list.
  - Viewing product details.
  - Adding a product to the cart.
  - Removing a product from the cart.
- Automated tests will be executed on both Android and iOS simulators/emulators.

Tools and Frameworks

- Appium: For cross-platform mobile automation.
- **Mocha**: For test scripting and assertions.
- GitHub: For version control and collaboration.
- JIRA/TestRail: For bug tracking and test case management.

6. Test Deliverables

- Test Plan Document: This document outlining the test approach.
- Test Scripts: Automated test scripts for the core functionalities.
- Bug Report: Documented issues with steps to reproduce, expected vs. actual behavior, and severity.
- Test Execution Report: Summary of test execution results.

7. Risks and Mitigation

- Risk: Flaky tests due to unstable test environment.
  - Mitigation: Use stable simulators/emulators and retry mechanisms for flaky tests.
- **Risk**: Flaky tests due to the absence of unique IDs for UI elements.
  - **Mitigation**: Use relative locators or other strategies to minimize flakiness. Implement retry mechanisms for flaky tests
- Risk: Limited access to real devices for testing.
  - Mitigation: Use cloud-based device farms (e.g., BrowserStack, Sauce Labs) if possible.