Test Plan: Project #1

Test Plan ID: Test Plan #1

Prepared by: William Crane
Created On: 27 November 2024

---

1. Introduction/Overview

This test plan outlines the manual testing activities for the **FakeStoreAPI**. The purpose of this testing is to ensure the API performs as expected across all endpoints, including product listings, cart management, user authentication, and checkout functionalities.

Key Features:

- Test Execution using Postman.
- Test Cases are managed in TestRail, stored and backed up on GitHub.
- Endpoints tested using HTTP methods: **GET**, **POST**, **PUT**, and **DELETE**.
- Test data and screenshots are stored in CSV and PNG formats, respectively.

---

2. Test Objectives

The objectives of this test plan include:

- Verifying the functionality of all API endpoints.
- Validating HTTP responses such as status codes, payload correctness, and error handling.
- Testing edge cases, including invalid inputs and unauthorised access.
- Ensuring integration with **TestRail** and **GitHub**.

---

3. Scope of Testing

In-Scope

- Product listing (GET all products, filtering, sorting).
- Cart management (creating, updating, deleting).
- User authentication (valid/invalid login).
- Checkout process.
- Error handling for invalid inputs and edge cases.

Out-of-Scope

- Performance testing (e.g., stress/load tests).
- UI testing.
- Security testing (e.g., penetration testing).

---

4. Test Approach

1. **Manual Testing** using Postman to interact with the API.
2. Test cases are written in CSV format and maintained in TestRail.
3. Validation includes:
   - **HTTP Status Codes**: 200 (Success), 400 (Bad Request), 401 (Unauthorized), 404 (Not Found).
   - **Response Data**: Consistency and accuracy in returned payloads.
4. Data and results are stored and tracked in GitHub.

---

5. Test Deliverables

- **Test Plan**: This document.
- **Test Cases**: Detailed steps are stored in TestRail and GitHub.
- **Test Results**: Execution logs from Postman and results summary.
- **Bug Reports**: If applicable, defects logged and tracked.
- **Screenshots**: Visual evidence of test failures.
- **Coverage Report**: Summary of tested API endpoints.

---

6. Test Resources

- **Postman**: API testing.
- **TestRail**: Test case management.
- **GitHub**: Test data and backup repository.
- **FakeStoreAPI**: Testing environment.

---

7. Test Cases

User Authentication

1. **Test Case ID**: C1
   Title: Valid User Login
   Steps:
   - Send a **POST** request to `/auth/login` with valid credentials.
   - Verify the response contains a valid token and status code 200.

2. **Test Case ID**: C2
   Title: Invalid User Login
   Steps:
   - Send a **POST** request to `/auth/login` with invalid credentials.
   - Verify the response contains an error message with status code 401.

Product Listing

3. **Test Case ID**: C3
   Title: Fetch All Products
   Steps:
   - Send a **GET** request to `/products`.
   - Verify all products are returned as a JSON array with status code 200.
4. **Test Case ID**: C4
   Title: Fetch a Single Product
   Steps:
   - Send a **GET** request to `/products/{id}` with a valid product ID.
   - Verify the product details match the expected data with status code 200.
5. **Test Case ID**: C5
   Title: Limit Product Results
   Steps:
   - Send a **GET** request to `/products?limit=5`.
   - Verify only 5 products are returned with status code 200.
6. **Test Case ID**: C6
   Title: Sort Product Results
   Steps:
   - Send a **GET** request to `/products?sort=desc`.
   - Verify the products are sorted in descending order with status code 200.

Cart Management

7. **Test Case ID**: C7
   Title: Add a New Cart
   Steps:
   - Send a **POST** request to `/carts` with valid payload.
   - Verify the response contains the new cart details with status code 201.
8. **Test Case ID**: C8
   Title: Fetch a User’s Cart
   Steps:
   - Send a **GET** request to `/carts/user/{userId}` with a valid user ID.
   - Verify the response contains the user’s cart details with status code 200.
9. **Test Case ID**: C9
   Title: Delete a Cart
   Steps:

    - Send a **DELETE** request to `/carts/{id}` with a valid cart ID.
    - Verify the response confirms cart deletion with status code 200.

Checkout Process

10. **Test Case ID**: C10
    Title: Validate Cart Before Checkout
    Steps:
    - Fetch the cart using `/carts/{id}`.
    - Verify all items in the cart are available, with status code 200.
11. **Test Case ID**: C11
    Title: Complete Checkout
    Steps:
    - Send a **POST** request to `/checkout` with the cart ID.
    - Verify the response confirms order completion with status code 200.

---

8. Test Environment

- **Postman**: Manual testing.
- **TestRail**: Test case tracking.
- **GitHub**: Repository for test data, test cases, and logs.
- **FakeStoreAPI**: API endpoint testing.

---

9. Test Metrics

- **Test Coverage**: Ensure 100% coverage for all endpoints.
- **Defects Logged**: Track critical, major, and minor issues.
- **Execution Summary**: Pass/fail status for all test cases.

---

10. Exit Criteria

- All critical test cases are executed and passed.
- No unresolved critical defects remain.
- Test logs and results are reviewed and documented.

---

11. Test Execution Schedule

Duration: 2 weeks
Daily Tasks:

- **Day 1-3**: Product Endpoints (GET, filter, sort).
- **Day 4-5**: Cart Management (Add, fetch, delete).
- **Day 6-8**: User Authentication and Persistence.
- **Day 9-10**: Edge cases and Error Handling.
- **Day 11-12**: Final Review and Results Compilation.
