Table of Contents

Test Plan for Fabrilife

Introduction:
This document discusses the test plan for the **Fabrilife** application. It provides an in-depth overview of the various testing types and approaches, including functional, non-functional, security, usability, smoke, and many more. Fabrilife is an e-commerce platform that offers a wide range of health and fitness products. The goal of this test plan is to ensure that the platform is secure, reliable, user-friendly and performs optimally across various environments and browsers.

Web Page Overview:
Fabrilife is a cutting-edge e-commerce platform for health and fitness enthusiasts, offering an array of products and services. The site allows customers to browse, purchase, and manage their health-related purchases seamlessly. Its main features include personalized product recommendations, user accounts, order tracking, payment gateway integration, and responsive design.

Purpose:
The primary goal of this test plan is to ensure that the Fabrilife application is reliable, secure, and user-friendly. The application must work effectively across different browsers, devices, and operating systems. Additionally, we aim to ensure the site is free of bugs, responsive, and performs well under various loads.

Scope of Testing:
The scope of testing for Fabrilife includes the following functionalities and features:

- **Registration**
- **Login**
- **Product Browsing and Search**
- **Order and Checkout Process**
- **Payment Gateway Integration**
- **User Profile Management**
- **Reviews and Ratings**
- **Product Recommendations**
- **Admin Panel**
- **Cart Management**

Formation:
The features that need to be tested include:

- Header Section
- Registration/ Login/ Forgot Password
- Product Search/ Filters
- Product Pages
- Cart & Checkout Flow
- Payment Integration
- User Profile
- Reviews and Ratings
- Admin Panel
- Footer Section

Passing Over:

- **Automation Testing**
- **Third-Party Integrations**
- **External APIs**

Testing Environment:

- **Operating Systems:** Windows 10, MacOS, iOS, Android
- **Browsers:** Chrome, Firefox, Safari, Microsoft Edge
- **Mobile Devices:** Android, iPhone
- **Network Conditions:** Various bandwidth and latency conditions

Plan of Action:

Action-01:

Test Case and Test Scenario:
We will use multiple testing design techniques to ensure comprehensive coverage:

- **Equivalence Class Partitioning (ECP)**
- **Boundary Value Testing**
- **Decision Table-Based Testing**
- **State Transition**
- **Error Guessing**

Action-02:

Test Process After Getting an Application:
We will perform smoke testing for each new build to check the stability of the application. After positive results from the smoke test, we will proceed with more in-depth testing. If smoke testing fails, we will send the results to the development team for correction.

Test Types to Perform:

- **Smoke Testing / Sanity Testing**
- **Regression Testing**
- **Integration Testing**
- **Usability Testing**
- **Functionality Testing**
- **Security Testing**
- **Performance Testing**
- **Stress Testing**
- **Load Testing**
- **Acceptance Testing**
- **Compatibility Testing**
- **System Testing**
- **Positive Testing**
- **Negative Testing**
- **User Acceptance Testing (UAT)**

Test Cases:

- **Smoke Testing / Sanity Testing:**
  Test basic functionalities like login, checkout, and order placement to ensure the app works correctly after each build.
- **Functionality Testing:**
  Test critical features like user registration, login, and payment processes.
- **Usability Testing:**
  Ensure that the interface is intuitive, and the application is responsive across different platforms.
- **Compatibility Testing:**
  Check if the platform renders correctly across different browsers (Chrome, Firefox, Safari, Microsoft Edge) and devices (desktop, mobile, tablet).
- **Performance Testing:**
  Test load times, stress conditions, and the platform's performance under heavy traffic.

- **Security Testing:**
  Verify data protection protocols like HTTPS and encryption mechanisms, and test login and logout processes for security.
- **Link Testing:**
  Verify that all links (internal and external) are working correctly and redirect to the correct locations.
- **Accessibility Testing:**
  Ensure that the site is accessible to all users, including those with disabilities.
- **Regression Testing:**
  Test critical modules such as the cart, checkout, and payment process to ensure that no new defects have been introduced.

For the **Fabrilife Test Plan**, if there are no APIs or automation testing included in the testing process, you can specify it as follows:

Application Programming Interface (API):

➔ N/A
Currently, no Application Programming Interface (API) is being used or tested for the **Fabrilife** platform.

Automation Testing:

➔ N/A
At this stage, there is no automation testing planned for the **Fabrilife** platform. All testing will be conducted manually.

Test Diagram:
A detailed mind map or flowchart will be created to help visualize the testing process and interactions between different modules of the application.

Validation and Verification:

- **Verification:** Ensuring the implemented system meets the requirements provided by stakeholders.
- **Validation:** Checking that the system meets customer expectations and user requirements.

User Acceptance Testing (UAT):
Following successful completion of validation testing, the end users and stakeholders will perform UAT to ensure that the Fabrilife application meets all requirements.

Defect Reporting Procedure:
Any issues found during testing will be logged using **JIRA**. The defect report will contain:

- Expected Behavior
- Actual Behavior
- Defect Description
- Reproduction Steps

Test Schedule

Task Name	Date From-To	Days
Creating Test Plan	26.11.2024 -28.11.2024	**3**
Mind Mapping	25.11.2024	**1**
Test Scenario		
Test Summary Report		
Test Case Writing		
Bug Report		
Test Metrics		
Recommendation		

Testing Tools:

- **Bug Tracking Tool:** JIRA
- **Mind Mapping Tool:** Xmind
- **Test Documentation:** Microsoft Word, Excel
- **Performance Testing:** JMeter
- **Screenshot Tool:** Lightshot

Defect or Bug Fixing:
Once a defect is identified, a detailed report will be created and sent to the development team for resolution. The report will include all necessary information to reproduce the defect, its severity, and its potential impact.

Recommendations:

- Regular optimization of the application performance to ensure smooth user experiences during peak usage.
- Continuous security assessments to ensure data integrity and user protection.

Acknowledgement:
The successful completion of testing and the resulting quality product could not have been achieved without the dedication of the testing team, the development team, and stakeholders. Their continuous support has played a crucial role in the success of the **Fabrilife** application.