**Deom Blaze Software Test Design**

**Project Name:** Demo Blaze Web App Software Test Design (STD)

**Prepared by:** Ehab Khalil

**Document Created:** 2024-08-03

### Software Test Design (STD) for Demoblaze

#### 1. Introduction

* **Objective:** Creating a plan or strategy to test the entire Demoblaze application, including all of its features and functions. Test design aims to identify software defects early in the development cycle and ensures that the Demoblaze application meets its requirements and functions correctly before it is deployed to production.
* **Scope:** Test design aims to ensure that the testing process is efficient and effective and identify all the bugs, errors, and future mistakes in the software. Functional testing will be the main goal in this project with a focus on automated tests.

#### 2. Approach

##### **2.1 Testing Levels**

The following testing levels will be employed:

1. **Unit Testing**
   * Performed by developers to test individual components and ensure they function correctly.
   * Focus on testing methods and functions within classes.
2. **Integration Testing**
   * Test the interaction between integrated components.
   * Ensure that modules work together as expected.
   * Use of stubs and drivers if necessary.
3. **System Testing**
   * End-to-end testing of the entire application.
   * Verify that the system meets functional and non-functional requirements.
   * Conducted by the QA team.
4. **Acceptance Testing**
   * Performed to ensure the system meets the business requirements.
   * Involves end-users and stakeholders.
   * User Acceptance Testing (UAT) will be conducted before the release.

##### **2.2 Testing Types**

The following types of testing will be conducted:

1. **Functional Testing**
   * Validate that the application functions according to requirements.
   * Includes testing of user authentication, product browsing, cart management, order processing, user profile management, and product search.
2. **Regression Testing**
   * Ensure that new changes do not negatively impact existing functionality.
   * Automated regression tests will be run after each new build.
3. **Performance Testing**
   * Assess the application's responsiveness, stability, and scalability under load.
   * Includes load testing and stress testing.
4. **Usability Testing**
   * Evaluate the user interface and user experience.
   * Ensure the application is intuitive and user-friendly.
5. **Security Testing**
   * Identify vulnerabilities and ensure the application is secure from threats.

**3. Test Identification**

**4. Pass/Fail Criteria**

**Pass Criteria**

1. **Functional Requirements Met:**
   * The test item meets all specified functional requirements.
   * All test cases for the feature are executed and pass without any critical or high-severity defects.
2. **Performance Criteria Met**
   * The application performs within acceptable limits under various load conditions.
3. **Usability Criteria Met**
   * The application is intuitive and user-friendly.
4. **Security Criteria Met**
   * The application is secure from potential threats and vulnerabilities.
5. **Compatibility Criteria Met**
   * The application functions correctly across different browsers, devices, and operating systems.
6. **No Critical Defects**
   * No critical or high-severity defects are present in the application.

**Fail Criteria**

1. **Functional Requirements Not Met:**
   * The test item fails to meet one or more specified functional requirements.
   * Any critical or high-severity defects are identified in the feature.
2. **Performance Criteria Not Met**
   * The application does not perform within acceptable limits under load conditions.
3. **Usability Criteria Not Met**
   * The application is not intuitive or user-friendly.
4. **Security Criteria Not Met**
   * The application has vulnerabilities or security threats.

**5. Summary**

The Software Test Design (STD) consists of a number of stages which progressively elaborate the design of tests from an initial high-level strategy to detailed test procedures. The design of tests must be driven by the specification of the software. At the highest level, this means that tests will be designed to verify that the software faithfully implements the requirements of the Requirements Specification. At lower levels, tests will be designed to verify that items of software implement all design decisions made in the Architectural Design Specification and Detailed Design Specifications. As with any design process, each stage of the test design process should be subject to informal and formal review. The ease with which tests can be designed is highly dependent on the design of the software. It is important to consider testability as a key (but usually undocumented) requirement for any software development.