**Test plan for**

**KSG Store**

*ChangeLog*

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Change Date** | **By** | **Description** |
| 1.0 | 4/12/2023 | George Anim | Started document |
| 1.1 | 4/16/2023 | Spencer Hall | Updated Document |
| 1.2 | 4/17/2023 | Spencer Hall | Finished document |

1 Introduction 2

1.1 Scope 2

1.1.1 In-Scope 2

1.1.2 Out-of-Scope 3

1.2 Quality Objective 3

1.3 Roles and Responsibilities 3

2 Test Methodology 4

2.1 Overview 4

2.2 Test Levels 4

2.3 Bug Triage 5

2.4 Suspension Criteria and Resumption Requirements 6

2.5 Test Completeness 7

3 Test Deliverables 7

4 Resource & Environment Needs 8

4.1 Testing Tools 8

4.2 Test Environment 9

5 Terms/Acronyms 9

# Introduction

This test plan outlines the strategies, processes, workflows, and methodologies that will be used to test the e-commerce platform under development. We will use a combination of automated and manual testing techniques, including unit testing, integration testing, and system testing. We will also incorporate continuous testing and delivery practices to ensure that the platform is thoroughly tested at every stage of development and deployment.

This test plan will provide a comprehensive overview of our testing approach, including the scope of testing, the quality objectives we aim to achieve, and the metrics we will use to measure our progress. It will also outline the roles and responsibilities of the testing team and the tools and technologies we will use.

## Scope

### In-Scope

The following features and functional/non-functional requirements of the e-commerce platform will be tested:

*User Authentication*

* Verify that all user roles (buyer, seller, admin) can log in to their accounts and perform actions specific to their role
* Test that users can log out of their accounts without any issues

*Buyer Functionality*

* Verify that buyers can search for products by name and can compare two products
* Test that buyers can add products to their cart and checkout securely
* Verify that buyers can initiate returns

*Seller Functionality*

* Test that sellers can add products to their inventory and set prices
* Verify that sellers can receive payments for their products via the chosen payment gateway

*Admin Functionality*

* Verify that admins can approve/block new user accounts and products
* Verify that admins can perform actions on behalf of users, such as resetting a password or cancelling an order

### Out-of-Scope

The following features and functional/non-functional requirements of the e-commerce platform will NOT be tested:

*Performance Requirements*

* Verify that the system can handle high volumes of traffic and transactions without crashing or slowing down significantly
* Test that the response time of the system is within acceptable limits
* Verify that the system can handle multiple concurrent requests without data inconsistency issues

*Compatibility Requirements*

* Test that the platform works correctly on different web browsers, such as Chrome, Firefox, and Safari
* Verify that the platform is compatible with different devices, including desktop computers, laptops, tablets, and smartphones
* Test that the platform works correctly on different operating systems, such as Windows, macOS, and Linux.

## Quality Objective

The objective of the testing effort for this e-commerce platform is to ensure that the Application Under Test (AUT) conforms to the functional and non-functional requirements specified in the project documentation. Additionally, we aim to verify that the AUT meets the quality specifications defined by the client and that any bugs or issues are identified and fixed before the platform goes live.

To achieve this objective, we will perform unit and integration testing using a number of automated tools, including PyTest and CircleCI. We will also manually conduct system testing to verify that the platform works as expected and to identify any usability issues. Ultimately, our goal is to deliver a high-quality e-commerce platform that meets the needs of users as well as the requirements set out by the client.

## Roles and Responsibilities

As our team consists of only three members, the roles and responsibilities are limited but crucial to the success of the project. The roles and responsibilities of our team members are as follows:

* *Back-end developer*: Responsible for developing and maintaining the server side of the application. This includes the design, implementation, testing, and deployment of the server-side components of the application.
* *Front-end developer*: Responsible for developing and maintaining the user interface of the application. This includes the design, implementation, testing, and deployment of the client-side components of the application.
* *Tester*: Responsible for ensuring that the application meets the functional and non-functional requirements. This includes creating and executing test cases, reporting bugs, and verifying that defects have been fixed.

In addition to these primary roles, each team member is expected to be flexible and take on additional responsibilities as needed to ensure the success of the project.

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Net ID** | **GitHub Username** | **Roles** |
| George Anim | gba37 | animkofi | back-end developer  unit testing |
| Kyana Conway | ktc188 | KyanaConway | front-end developer  system testing |
| Spencer Hall | jsh278 | spencerhall01 | group leader  back-end developer  integration Testing |

# Test Methodology

## Overview

Agile was chosen as the test methodology for this project because it emphasizes collaboration, adaptability, and flexibility, all of which are essential for delivering a high-quality e-commerce platform. In an Agile environment, testing and development are intertwined to ensure that quality is built into every stage of development and that any issues are identified and resolved as quickly as possible. Agile is particularly well-suited for this project because it allows us to break down complex requirements into smaller, more manageable tasks, which can be tested and validated on an ongoing basis.

As part of our Agile methodology, we will use Scrum as our project management framework, which will help us to prioritize tasks, track progress, and ensure that testing is aligned with development goals and timelines. We will also incorporate continuous testing, integration, and delivery practices to ensure that we catch any bugs or issues early in the development process and that the platform is thoroughly tested at every stage of development and deployment.

## Test Levels

Testing of the e-commerce platform will be conducted at several levels, including unit testing, integration testing, and system testing. Each level of testing serves a specific purpose and helps ensure the platform is thoroughly tested before it is released. The testing levels selected for this project take into account the scope of the project and time constraints.

**Unit Testing**

Unit testing is the first level of testing and is performed on individual units of code to ensure that they meet their functional requirements. We will use PyTest to automate our unit testing and ensure that each unit of code performs as intended and integrates smoothly with the rest of the platform.

**Integration Testing**

Integration testing is the next level of testing and is performed to ensure that different units of code work together as expected. We will use CircleCI, integrated into GitHub, to automate our integration testing and ensure that each module of the platform works seamlessly with the rest of the system.

**System Testing**

System testing is the third level of testing and is performed on the complete system to ensure that it meets its functional and non-functional requirements. We will primarily use manual testing to validate the system's behavior and ensure that it meets the expectations of our client.

## Bug Triage

Bug triage is a critical process that involves identifying, prioritizing, and resolving bugs in the e-commerce platform. The goal of bug triage is to ensure that each bug is assigned the appropriate type of resolution and prioritized based on its severity and impact on the platform.

**Types of Resolution**

Each bug will be assigned one of the following types of resolution:

* *Fixed*: The bug has been identified and fixed by the development team.
* *Won't Fix*: The bug has been identified but will not be fixed due to low priority, lack of resources, or other reasons.
* *Duplicate*: The bug has been identified but is a duplicate of another bug that has already been reported.
* *Cannot Reproduce*: The bug has been reported but cannot be reproduced by the development team.
* *Deferred*: The bug has been identified but will not be fixed in this release and will be deferred to a future release.

**Prioritization**

Each bug will be prioritized based on its severity and impact on the platform. Bugs will be classified into one of the following categories:

* *Critical*: The bug has a severe impact on the platform's functionality and must be resolved before the platform can be released.
* *High*: The bug has a significant impact on the platform's functionality and should be resolved as soon as possible.
* *Medium*: The bug has a moderate impact on the platform's functionality and should be resolved in a reasonable timeframe.
* *Low*: The bug has a minor impact on the platform's functionality and can be resolved at a later time.

**Schedule**

Once bugs have been prioritized and assigned a type of resolution, a schedule will be created for all "To Be Fixed Bugs." The schedule will take into account the severity and impact of each bug, as well as the availability of development resources. Bugs will be resolved in the order in which they appear on the schedule until all "To Be Fixed Bugs" have been resolved. The bug triage process will be ongoing throughout the testing process to ensure that any new bugs are identified and resolved as quickly as possible.

## Suspension Criteria and Resumption Requirements

During the testing process, it may be necessary to temporarily suspend testing activities due to a variety of factors, such as the need for additional information or resources, unexpected issues, or changes in project priorities. To ensure that testing activities can be resumed as efficiently as possible, and to minimize any impacts on the project schedule, to it's important to define clear suspension criteria and resumption requirements.

**Suspension Criteria**

Testing activities may be suspended if any of the following criteria are met:

* *Critical defects*: If critical defects are identified during testing, testing may be suspended until the defects are resolved and retesting can be completed.
* *Insufficient resources*: If there are insufficient resources, such as test environments or testing personnel, testing may be suspended until the necessary resources are available.
* *Changes to requirements or design*: If there are changes to the project requirements or design, testing may be suspended until the changes can be reviewed and any necessary updates can be made.
* *Unforeseen events*: If unforeseen events, such as power outages or network failures, occur during testing, testing may be suspended until the issue is resolved.

**Resumption Requirements**

Once testing activities have been suspended, they can be resumed once the following requirements have been met:

* *Clear communication*: All team members involved in testing should be notified of the suspension and the reason for it. When testing resumes, all team members should be notified of the resumption and the testing activities that will be resumed.
* *Verification*: Any completed testing activities should be verified before testing resumes. This ensures that the test results are still valid and that any defects that were identified before the suspension have been resolved.
* *Replanning*: If the suspension of testing results in a delay in the project schedule, the testing plan may need to be updated to reflect the delay.

## Test Completeness

The following criteria will be used to measure the completeness of the testing process for this project. When all of these criteria are met, we will consider the testing process complete and the product ready for release.

* All identified test cases have been executed and passed
* All high and medium priority defects have been resolved and validated
* All known risks have been addressed and mitigated
* Test coverage of at least 95% has been achieved
* All test deliverables have been created, reviewed, and approved

# Test Deliverables

Throughout the testing lifecycle, various test artifacts will be created and delivered. These test deliverables serve as documentation of the testing process and provide visibility into the progress and outcomes of testing activities. The following test artifacts will be delivered during the testing lifecycle:

* *Test Plan*: The test plan outlines the scope, approach, and objectives of testing. It includes details on the testing strategy, test environment, testing schedule, and test deliverables.
* *Test Cases*: Test cases are step-by-step instructions for executing specific tests. They include details on the test data, expected results, and pass/fail criteria.
* *Requirement Traceability Matrix (RTM)*: The RTM is a document that maps the requirements to the corresponding test cases. It helps ensure that all requirements have been covered by testing and provides visibility into the test coverage.
* *Bug Reports*: Bug reports document defects found during testing. They include details on the defect, steps to reproduce, and severity level.
* *Test Strategy*: The test strategy provides an overview of the testing approach, including the types of testing that will be performed and the resources required for testing.
* *Test Metrics*: Test metrics are quantitative measurements of the testing process, such as the number of defects found, the number of test cases executed, and the test coverage.

These test deliverables provide documentation of the testing process and serve as a reference for future testing activities. By delivering these artifacts, stakeholders can track the progress of testing, ensure that requirements have been met, and make informed decisions about the release of the application.

# Resource & Environment Needs

## Testing Tools

The following tools are essential for ensuring that the software meets the functional and non-functional requirements and that issues are identified and resolved promptly.

**Tracking Tools**

* *GitHub*: The primary tracking tool for this project is GitHub, which is used for tracking requirements, bugs, and tasks. User stories and task breakdowns are maintained in the project wiki, while issues are created and assigned for tracking and resolving bugs.

**Automation Tools**

* *PyTest*: We will be using PyTest for automated unit and integration testing. This tool will allow us to write and execute automated tests to ensure that the application functions as intended.
* *CircleCI*: We will be using CircleCI for continuous integration and deployment. This tool will be used to automate integration testing by running tests every time changes are made to the codebase.

## Test Environment

The Application Under Test will primarily be tested in a Linux command-line environment (CLI), running on Ubuntu under the Windows Subsystem for Linux (WSL). The minimum hardware requirements for testing the application are as follows:

* A computer running Windows 11 or above
* Intel Core i5 processor or equivalent
* 8 GB RAM or above
* At least 1 GB of free disk space
* High-speed internet connection

In addition, the following software is required:

* Ubuntu 20.04 or above
* Python 3.8 or above
* Django 3.2 or above
* Pytest 6.2 or above
* Git or GitHub for version control and issue tracking
* CircleCI for continuous integration and deployment

Please note that this is the minimum requirement for testing the application, and higher-end hardware configurations may provide better performance.

# Terms/Acronyms

| TERM/ACRONYM | DEFINITION |
| --- | --- |
| API | Application Program Interface |
| AUT | Application Under Test |