Test plan for SneakerHeadz

ChangeLog

Version #	Change Date	By	Description

1 Introduction

The method of testing used was the agile model method of building and testing simultaneously for this project. CircleCI was introduced as a workflow integration application to allow the testing of functions quickly. Pytest was used in conjunction with CircleCI to test functions that were created for this project.

1.1 Scope

1.1.1 In-Scope

In-Scope features include: front-end and back-end listing page display, user account page, login, logout, and registration functions, the time accuracy of those functions, and a SQL database for users and inventory. These will be tested including payment and possible cart functions.

1.1.2 Out-of-Scope

Out-of-Scope features include: front-end and back-end search, selling/listing shoes, admin, ratings and other functions related to user stories as well as a SQL database for transactions. These will not be tested in at least the first testing period.

1.2 Quality Objective

Overall Objective: To create a working application that will allow users to interact with designated functions for their needs to be met.

Breakdown of Objectives:

- Ensure the Application Under Test conforms to functional and nonfunctional requirements
    - The main functions for this testing plan are successful databases, logging in and out (as well as registration), and an interactive user account page
- Ensure the AUT meets the quality specifications defined by the client
    - Mainly, create acceptance testing for the user stories for testing period(s)
- Bugs/issues are identified and fixed before going live
    - Before deploying website and testing functions through CircleCI, make sure everything runs well on localhost window

1.3 Roles and Responsibilities

Our roles were splitten amongst each other while some took the roles of others to help production and testing in time. We were all installation team members as well. Those specific roles and tasks were broken down more in our GitHub Wiki Pages.

- QA Analyst
- Test Manager
- Configuration Manager
- Developers
- Installation Team Member

Name	**Net ID**	**GitHub username**	**Role**
Ander Talley	ajt432	MaverickDSmith	Configuration/Test Manager/Developer
Kennedy Keyes	kfk38	CodingKen02	QA Analyst/Developer
Austin Wheeler	aaw345	LaBrav0	Developer
Ehren Achtermann	eta55	eta55	Developer
Trey Fullwood	rf802	TreyPilgrim	Installation Team Member

2 Test Methodology

2.1 Overview

The Methodology chosen for this project is the Agile Model. This model is best suited for the time constraints in which we have to work on this project. The short time for each sprint means that testing during the building of the application is necessary to achieve an optimal result. The model also allows the team to minimize the risks by incrementing testing.

2.2 Test Levels

For our test level, we are primarily doing Integration Testing with CircleCI based on our project, time, and budget constraints. Through integration testing, we are able to ensure that any additions to the site are smooth. In the event that anything goes awry, frontend and backend teams work together to solve any lingering bugs. The primary purpose of this is to guarantee we won’t have a pile of errors to navigate through at the end, as well as helping the development teams know what direction to take in designing the site.

2.3 Bug Triage

The Bug Triage helped us fix bugs on time. We accomplished this by:

- Defining the type of resolution for each bug
- Prioritizing bugs and determining a schedule for all "To Be Fixed Bugs'.

Through good communication, the group is able to tackle errors. We tackled bugs as they arose. On average, the person that made the code was also the one to fix the bug; occasionally, other group mates would help but this was the primary method for handling errors found on the site.

2.4 Suspension Criteria and Resumption Requirements

We thankfully never had an error that stopped all production. The only time anything like that would happen was in the event that the front end was not working properly; this usually led to a halt in backend production until it worked properly.

Suspension Criteria: HTML files won't appear or link properly
Resumption Criteria: When HTML files are corrected for proper use of the functions

2.5 Test Completeness

Here are the defined criterias that will deem our testing complete:

- 100% test coverage
- All Open Bugs are fixed *or* will be fixed in next release
- All automated test cases executed with expected outcomes
- Acceptance Criteria for User Stories is met and doable.
- Pytest files passes with no errors or failures

3 Test Deliverables

Here are all the Test Artifacts that will be delivered during different phases of the testing lifecycle:

- Test Plan
- Test Cases and Outcomes per Feature
- Pytest Reports
- Acceptance Testing/Criteria for User Stories

4 Resource & Environment Needs

4.1 Testing Tools

These are required to test the project:

- pytest (testing Python function files)
- CircleCI, a continuous integration (CI) tool website

4.2 Test Environment

The test environment will be on a desktop/laptop device that can handle inputs from mouse and keyboard as well as displaying information on a screen.
The following **software is** required to test the application:

- Platform Linux / pytest
    - Python Flask, SQLAlchemy, and other listed dependencies in requirements.txt file
- Windows 8 and above; mac OS Montgomery or above
- CircleCI for Continuous Integration Testing Tools
- GitHub for Source Code to sync with CircleCI

5 Terms/Acronyms

Make a mention of any terms or acronyms used in the project

TERM/ACRONYM	DEFINITION
API	Application Program Interface
AUT	Application Under Test
CI	Continuous Integration
SQL	Structured Query Language
OS	Operating Systems
HTML	HyperText Markup Language