CO421 SOFTWARE TESTING

“Testing of Shuup E-Commerce Platform”

SUBMITTED BY

Veena Nagar - 171CO151

Rounak Modi - 171CO236

VIII SEMESTER B-TECH

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA SURATHKAL

Overview of the Project

1. Purpose of the document
2. Application Overview
3. Testing Scope
    a. In Scope
    b. Out of Scope
    c. Items not tested
4. Metrics
5. Types of testing performed
    a. Functional Testing
        i. Unit testing
            Modules tested:
            a. - Login
            b. - Orders
            c. - Campaigns
            d. - Coupons
            e. - Discounts
        ii. Blackbox testing
            Modules tested:
            a. - Reports
            b. - Taxes
            c. - Users
            d. - API
            e. - Customer Tax Groups
            f. - Telemetry

    b. Non Functional Testing
        i. Usability
            Modules tested:
            1. - Contacts
            2. - Shops
        ii. Compatibility
            Quick Links module
            1. - form pages are tested in 2 different browsers
6. Test Environment
7. Test Tools
8. Best Practices
9. Exit Criteria
10. Conclusion

1. Purpose of the document

This document explains the various activities performed as part of the Testing of ‘Shuup E-Commerce platform’ application.

2. Application Overview

Shuup is an open-source e-commerce platform built on top of Django. It aims at satisfying both sellers and buyers: features B2B, B2C, and multi-vendor stores. Shuup store features unlimited products, product types, and categories. Built-in features also include customer groups and customer property management. You can add a person or company contacts, integrate Stripe payments and customize the checkout process.

Existing industry software were not usable for the complex enterprise and B2B solutions we were creating from the ground up. Shuup was innovated and developed internally by Shuup’s commerce experts as the perfect multi-vendor marketplace development platform.

There are several modules like Orders, Products, Shops and Reports which are integrated to fulfill the purpose.

3. Testing Scope

The scope of software testing itself is to cover both functional and non-functional aspects of the entire product under development/test. But there are some aspects of the application which cannot be tested by us. A functionality verification which needs connectivity to a third party application cannot be tested, as the connectivity could not be established due to some technical limitations.

3.1 In Scope

Login

1. Check the login capability
2. Only authentic users should be allowed to login to the system.

Browser Compatibility

1. Test for browsers which support the application on the following platforms(Firefox, Chrome)

Page Display

1. Check if the pages are displayed correctly.
2. Detect dead hyperlinks , plugin dependencies, font sizing etc

3. Check if runtime error messages are displayed correctly.
4. Check if alerts are displayed whenever required.

Usability

1. Check if the design is non intuitive.
2. Check if the site navigation is poor.
3. Check if help-support feature is poor.
4. Check if catalog navigation for the customer is not proper.

Modules

1. Orders
2. Products
3. Contacts
4. Campaigns
5. Content
6. Reports
7. Shops

3.2 Out of Scope

The following features will not be tested by us during the testing phase due to lack of resources and expertise in the field.

Session Testing

1. Session Expiration: To test whether session timeout is properly enforced by the server.
2. To test whether the data stored in sessionStorage gets cleared when the page session ends.

Content Analysis

1. Check for misleading, offensive and litigious content in the webpages.
2. Royalty free images and copyright infringement.

Availability Testing

1. Test for denial of service (DoS) attacks.
2. Unacceptable levels of unavailability.

Back-up and Recovery Testing

1. Test how each component of the module will react in case of failure.
2. Check for backup failures.

3. Test the fault tolerance capability of the application.

Performance Testing

1. Check for performance bottlenecks in the application.
2. Check how much load can the application handle.
3. Check for how much stress can be handled by the website.
4. Check for scalability for the application.

3.3 Items not tested

The following modules were not tested:

- Addons - Shuup contains facilities for installing, detecting, loading and configuring additional functionality with little or no system administration knowledge needed. Packages that can be loaded in this way are called Addons.
- Admin - creation of new admin was not tested.
- Settings - System settings, Notifications, Data Import and GDPR were not tested.

4. Metrics

- No. of test cases planned vs executed : Planned (100) Executed (100)
- No. of test cases passed/failed: Test Cases passed (100) Test Cases failed (0)
- No. of defects identified and their Status & Severity : 1, low severity.
- Defects distribution
  - If the asterisk field is left unfilled in Order status, Django key error is shown instead of alert box.

5. Types of testing performed

5.1 Functional Testing

5.1.1 Unit Testing

This is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output.

Number of test cases: 25
Modules tested:

- Login
- Orders
- Campaigns
- Coupons
- Discounts

5.1.2 Blackbox Testing

This is also known as Behavioral Testing, is a software testing method in which the internal structure/design/implementation of the item being tested is not known to the tester. Number of test cases: 25
Modules tested:

- Reports
- Taxes
- Users
- API
- Customer Tax Groups
- Telemetry

5.2 Non-Functional Testing

5.2.1 Usability Testing

Usability testing is a technique used in user-centered interaction design to evaluate a product by testing it on users. This can be seen as an irreplaceable usability practice, since it gives direct input on how real users use the system.
Number of test cases: 18
Modules tested:

- Contacts
- Shops

5.2.2 Compatibility Testing

Compatibility Testing is a type of Software testing to check whether your software is capable of running on different hardware, operating systems, applications, network environments or Mobile devices.
Number of test cases: 32
Modules tested: Quick- Forms

6. Test Environment

Test Environment consists of elements that support test execution with software, hardware and network configured. Test environment configuration must mimic the production environment in order to uncover any environment/configuration related issues.

We will be using the Python environment for automated testing. Automated testing is the execution of the test plan by a script instead of a human. Python comes with a set of tools and libraries to enable creation of automated tests for our application.

Environment requirements:

- Python3 must be installed on the system
- Selenium must be installed on the system.

Problems:

1. **Test process problems** often occur when testing and engineering processes are poorly integrated. Testing may not be adequately prioritized so that functional testing may be overemphasized. Testing of components, subsystems, or the system may begin before they are sufficiently mature for testing. Other problems include inadequate test evaluations and inadequate test maintenance.

2. **Test tools and environments problems** include an over-reliance on testing tools. Often, there are an insufficient number of test environments. Some of the test environments may also have poor quality (excessive defects) or insufficient fidelity to the actual system being tested. Moreover, the system and software under test may behave differently during testing than during operation. Other common problems are that tests were not delivered or the test software, test data, and test environments were not under sufficient configuration control.

7. Test Tools

Selenium

Selenium is an open-source tool that automates web browsers. We have used selenium-python that lets us write test scripts in python.
Selenium Webdriver then executes these scripts on a browser-instance on our device.
We performed functional tests and compatibility testing (non functional) using Selenium.

8. Best Practices

- A repetitive task done manually every time was time-consuming. This task was automated by creating scripts and run each time, which saved time and resources.
- Browser testing is important because the application should work equally well for users on all kinds of Browsers.
- Automation scripts were prepared to create new customers, new orders, new campaigns etc where a lot of records need to be created for Testing.

9. Exit Criteria

- All test cases should be executed : YES
- Run Rate (Verify if all the tests specified have been run) : 100%
- Verify if the level of requirement coverage has been met.: YES
- Verify if there are NO Critical or high severity defects that are left outstanding.: YES
- Pass Rate (should be high): 100%

10. Conclusion

As the Exit criteria was met and satisfied as mentioned in Section 9, this application is suggested to 'Go Live' by us. Appropriate User/Business acceptance testing should be performed before 'Go Live'.