**STP**

**ECOMMERCE DJANGO PROJECT**

**Haytam Khalil**

**Document code: 234567**

|  |  |
| --- | --- |
| **STP document template** | Document Name |
| **1.0** | version |
| **17.7.2022** | Version date |

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Date | role | signature |
| Haitham Khalil | 17/7/2022 | Team leader - QA | H.KH |
|  |  |  |  |

**Tracking changes:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Document version** | **Changes documentation** | | **Done by** | **day** |
|  |  |  |  |  |
| **1.0** |  |  | Me | today |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Contents

[1.1 Document purpose 3](#_Toc109127972)

[1.2 General description of the system 3](#_Toc109127973)

[1.3 Applicable Documents 3](#_Toc109127974)

[*1.4* Terms and concepts 3](#_Toc109127975)

[2 Testing strategy 4](#_Toc109127976)

[2.1 General strategy 4](#_Toc109127977)

[2.2 Test levels 5](#_Toc109127978)

[2.3 planned time schedule 5](#_Toc109127979)

[2.4 Tests content 5](#_Toc109127980)

[2.4.1 Tests included in the document 5](#_Toc109127981)

[2.4.2 Tests not included in the document 5](#_Toc109127982)

[3 Training & HR 5](#_Toc109127983)

[4 General maintenance 5](#_Toc109127984)

[4.1 Changes in system content 5](#_Toc109127985)

[4.2 Criteria to accept system for testing 6](#_Toc109127986)

[4.3 Criteria to accept system for the next step 6](#_Toc109127987)

[5 Test subjects 7](#_Toc109127988)

[Coverage table 8](#_Toc109127989)

[1 Tests data 8](#_Toc109127990)

[2 Requirements to procced test 8](#_Toc109127991)

[3 Risk management 9](#_Toc109127992)

[4 Fault managemnet 9](#_Toc109127993)

[#1. New 10](#_Toc109127994)

[#2. Assigned 10](#_Toc109127995)

[#3. Open 10](#_Toc109127996)

[#4. Fixed 10](#_Toc109127997)

[#5. Test 10](#_Toc109127998)

[#6. Verified 11](#_Toc109127999)

[#7. Closed 11](#_Toc109128000)

[#8. Reopen 11](#_Toc109128001)

[#9. Duplicate 11](#_Toc109128002)

[#10. Deferred 11](#_Toc109128003)

[#11. Rejected 11](#_Toc109128004)

**1 General**

## Document purpose

* Functional and nonfunctional tests
* The tests will be performed according to the AGILE Methodology
* Work Approach: Dividing the system by Functionality
* Managing project and team using Trello
* Integration tests
* Api tests
* Unit tests

## General description of the system

A completely eCommerce / shopping cart application using Django, REACT and REDUX with the following functionality:

* Full featured shopping cart
* Product reviews and Ratings
* Top products carousel
* Product pagination
* Product search feature
* User profile with orders
* Admin product management
* Admin user management
* Admin Order details page
* Mark orders as a delivered option
* Checkout process (shipping, payment method, etc)
* PayPal / credit card integration

## Terms and concepts

* **Functional testing -**  is the process through which QAs determine if a piece of software is acting in accordance with pre-determined requirements.
* **Non-functional** - It is designed to test the readiness of a system as per nonfunctional parameters which are never addressed by functional testing. (Security, Ssability, Performance).
* **Integration tests** - type of testing where software modules are integrated logically and tested as a group
* **Unit tests** – a type of software testing where individual units or components of a software are tested.
* **Docker**: is a software development platform for virtualization with multiple Operating systems running on the same host. It helps to separate infrastructure and applications in order to deliver software quickly
* **Jenkins**: Jenkins is an open-source automation server. open-source Continuous Integration tool.
* **Selenium**: open-source web-based automation tool.
* **Python –** an object-oriented programming language, it is ideally designed for rapid prototyping of complex applications.
* **SQL -** is the standard language to query a database.

# Testing strategy

## General strategy

* Functional tests
* Integration tests
* Unit tests
* E2e tests
  + 1. **Functional tests**

We will perform these tests to establish whether the application features works as per the software requirements.

* + 1. **Integration tests**

We will perform these tests to test the interfaces between the modules and expose any defects that may arise when these components are integrated and need to interact with each other.

* + 1. **Unit tests**

We will perform these tests on an individual blocks of code. To isolate written code to test and determine if it works as intended.

* + 1. **E2e tests**

We will perform these tests in order to verify that all components of a system are able to run and perform optimally under real-world scenarios

## Test levels

* + 1. components testing
    2. integration testing
    3. **system testing**
    4. **acceptance testing**

## planned time schedule

|  |  |  |
| --- | --- | --- |
| **To** | **From** | **step** |
| 23.09.2022 | 18.07.2022 | STP |
| 28.09.2022 | 24.08.2022 | STD |
| 29.09.2022 | 29.08.2022 | Perform first testing cycle |
|  |  | Perform second testing cycle |

## Tests content

### Tests included in the document

All the sections inside specification document

### Tests not included in the document

N/A

# Training & HR

* 2 testers to check the requirements and specification documents.
* Training the test team for the platform we will use (Jira)
* 3 people will work on the project: 1 team leader ,1 developer, 1 automation testers.

## 

## Criteria to accept system for testing

The following criteria define the conditions to accept the system for testing.

The purpose of the criteria is to ensure that the system is indeed ready for testing, to avoid situations of instability and double work.

Below given criteria’s:

|  |  |
| --- | --- |
| criteria | Step |
| All planned tests were performed | Sanity tests |
| All tests performed were successful | Sanity tests |

## Criteria to accept system for the next step

The following criteria define the "red line" for moving the system to the next step. The "red line" is defined by:

* The number of open (unfixed) faults and their level of severity
* Percentage of tests performed from the planned tests
* Percentage of tests successfully passed from the tests performed

Faults that are discovered during the tests are documented and marked with a "severity level". Possible severity levels will be

▪ Critical: For a malfunction that causes "failure" in the system

▪ Severe: For a malfunction that does not cause "failure" but does not allow continued normal operation.

▪ Moderate: A malfunction that does not cause "failure" and allows continued normal operation.

▪ Minor: Slight GUI-level malfunction or similar.

The following is the definition of the criteria to accept the system to production, according to the percentage of faults (or CR) from the total number of tests performed:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **minor** | **moderate** | Severe | **critical** | **Criteria** |
| <15% | <10% | 0 | 0 | Opened defects |

The following is the definition of the criteria to accept the system to production, according to Tests levels:

|  |  |
| --- | --- |
| **Percentage %** | **Criteria’s** |
| 90% | Percentage of planned tests were performed |
| 80% | Percentage of tests performed were successful |

# Test subjects

* 1. **Test User**

**Unit tests**

* Test create regular user
* Test set password
* Test product creation

**Integration tests**

* Test API user creation
* Test API product creation

**E2E tests**

* **#Positive scenario**
* Test Search product
* Test register new user
* **#Negative scenario**
* Test log in by input wrong password and username
  1. **Test products**
* Test product creation

Coverage table

|  |  |  |  |
| --- | --- | --- | --- |
| Requirement document | Specification document | Design document | STP |
|  | 1.1 |  | 1.0 |
|  | 1.2 | 1.2 | 1.2.4 |
|  |  |  | 1.2.5 |
| 2 | 1.3 | 1.3.1 | 1.3.1.2 |
|  |  |  | 1.3.3.3 |

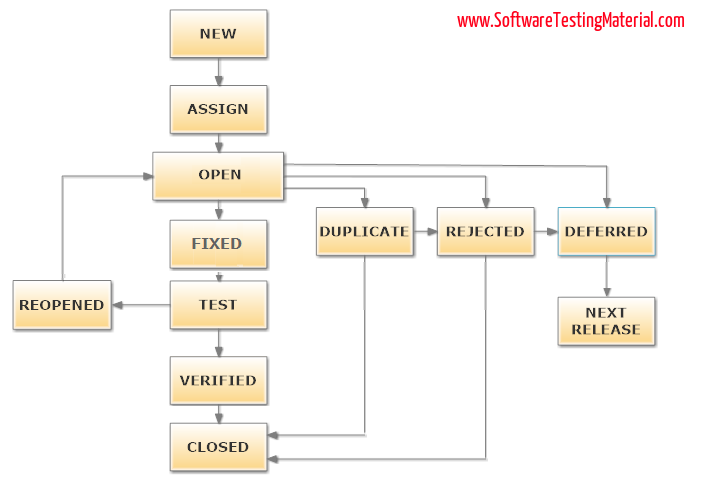
# Requirements to procced test

|  |  |  |
| --- | --- | --- |
| **#** | **Cause** | **Requirements** |
| **Hardware** | | |
| **312** |  | Working computer |
|  |  | Internet conniction |
|  |  |  |
| **Software** | | |
| **313** |  | Docker, Jenkins, Selenium Grid |
|  |  |  |
| **others** | | |
|  |  |  |
|  |  |  |

# Risk management

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| # |  |  |  |  |  |  |  |  |  |  |
| 1 | *Tester goes to vacation* | 5 | 3 | 15 | Part of system will not be tested |  | Increase schedule or assign additional testers | project | Team leader | 10/7/22 |
| 2 | Testers without experience | 5 | 3 | 15 | New junior testers | monitoring | Monitor the work of testers | project | Team leader | All time |

# Fault management

**

### **#1. New**

When a tester finds a new defect.

### **#2. Assigned**

Defects that are in the status of New will be approved (if valid) and assigned to the development team by Test Lead/Project Lead/Project Manager.

### **#3. Open**

The development team starts analysing and works on the defect fix.

### **#4. Fixed**

When a developer makes the necessary code change and verifies the change, then the status of the bug will be changed as “Fixed” and the bug is passed to the testing team.

### **#5. Test**

If the status is “Test”, it means the defect is fixed and ready to do test whether it is fixed or not.

### **#6. Verified**

The tester re-tests the bug after it got fixed by the developer. If there is no bug detected in the software, then the bug is fixed and the status assigned is “verified.”

### **#7. Closed**

After verified the fix, if the bug is no longer exits then the status of the bug will be assigned as “Closed.”

### **#8. Reopen**

If the defect remains the same after the retest, then the tester posts the defect using the defect retesting document and changes the status to “Reopen”. Again the bug goes through the life cycle to be fixed.

### **#9. Duplicate**

If the defect is repeated twice or the defect corresponds to the same concept of the bug, the status is changed to “duplicate” by the development team.

### **#10. Deferred**

In some cases, the Project Manager/Lead may set the bug status as deferred.

* If the bug found during the end of the release and the bug is minor or not important to fix immediately.
* If the bug is not related to the current build.
* If it is expected to get fixed in the next release.
* The customer is thinking to change the requirement.
* In such cases the status will be changed as “deferred” and it will be fixed in the next release.

### **#11. Rejected**

If the system is working according to specifications and the bug is just due to some misinterpretation (such as referring to old requirements or extra features) then the Team lead or developers can mark such bugs as “Rejected”.