Software **Requirements Specification**

**for**

SOFTWARE TESTING PLAN

**Version 1.0 approved**

**Prepared by Team 3**

**R.M.K**

**December 1,2022**

**Table of Contents**

**Table of Contents ii**

**Revision History ii**

**1. Introduction 1**

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Project Scope 1

1.5 References 1

**2. Overall Description 2**

2.1 Product Perspective 2

2.2 Product Features 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints

2.6 Assumptions and Dependencies 3

**3. System Features 3**

3.1 System Feature 1 3

3.2 System Feature 2 (and so on) 4

**4. External Interface Requirements 4**

4.1 User Interfaces 4

4.2 Hardware Interfaces 4

4.3 Software Interfaces 4

4.4 Communications Interfaces 4

**5. Other Nonfunctional Requirements 5**

5.1 Diagram 5

**6. Other Requirements 5**

**Appendix A: Glossary 5**

**Appendix B: Analysis Models 6**

**Appendix C: Issues List 6**

# **Introduction**

## **Purpose**

*The purpose of this document is to determine the effort needed to validate the quality of the application under test. The test plan serves as a blueprint to conduct software testing activities as a defined process, which is minutely monitored and controlled by the test manager.*

## **Document Conventions**

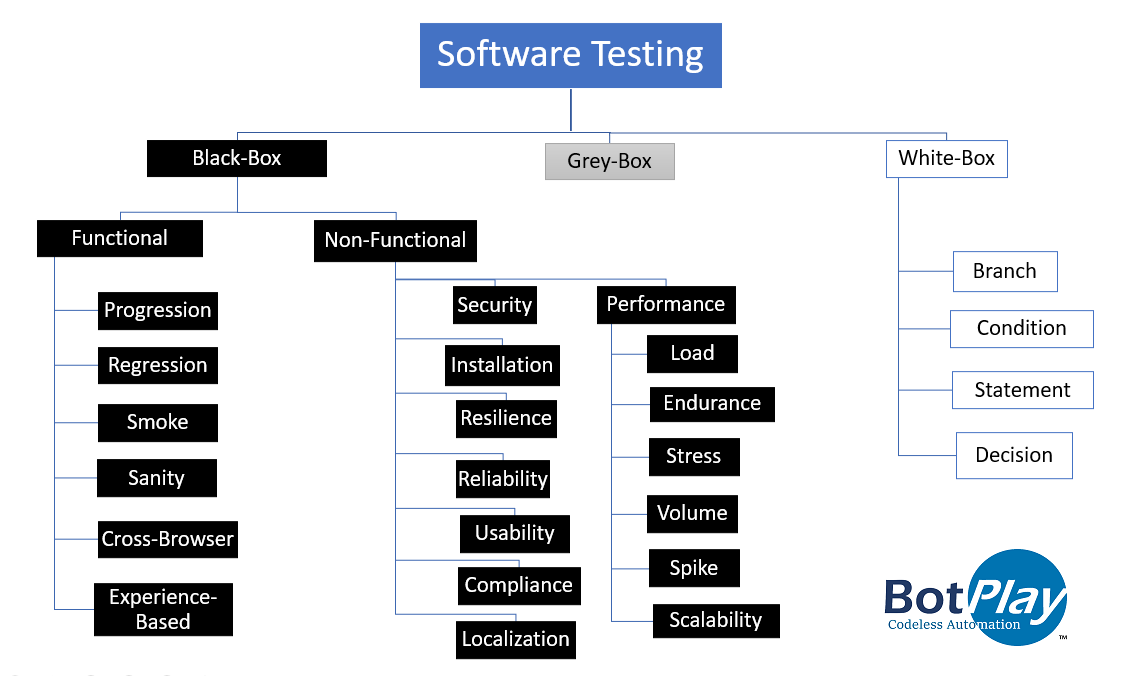
*This document uses the following conventions:*

*This includes* ***defining test objectives, test approach, test tools, test environment, test schedules and team responsibilities and composition****. However, before the right test approach and other planning details can be defined, a larger view of the organizational and project objectives must be defined first.*

## **Intended Audience and Reading Suggestions**

*Every Software Testing Life Cycle (STLC) begins with test planning. This article will go through the entire planning process and highlight all that is necessary to create result-oriented software tests, no matter the nature of the software or the project in question.*

## **Project Scope**

* *Articulates what the project entails so that all stakeholders can understand what's involved;*
* *Provides a roadmap that managers can use to assign tasks, schedule work and budget appropriately;*
* *Helps focus team members on common objectives; and*
* *Prevents projects, particularly complex ones, from expanding beyond the established vision.*

## **References**

[*https://www.onestoptesting.com*](https://www.onestoptesting.com)

# **Overall Description**

## **Product Perspective**

The distributed software testing plan contains the following components:

*Scope: Details the objectives of the particular project. Also, it details user scenarios to be used in tests. If necessary, the scope can specify what scenarios or issues the project will not cover.*

*Schedule: Details start dates and deadlines for testers to deliver results.*

*Resource Allocation: Details which tester will work on which test.*

*Environment: Details the nature, configuration, and availability of the test environment.*

*Tools: Details what tools are to be used for testing, bug reporting, and other relevant activities.*

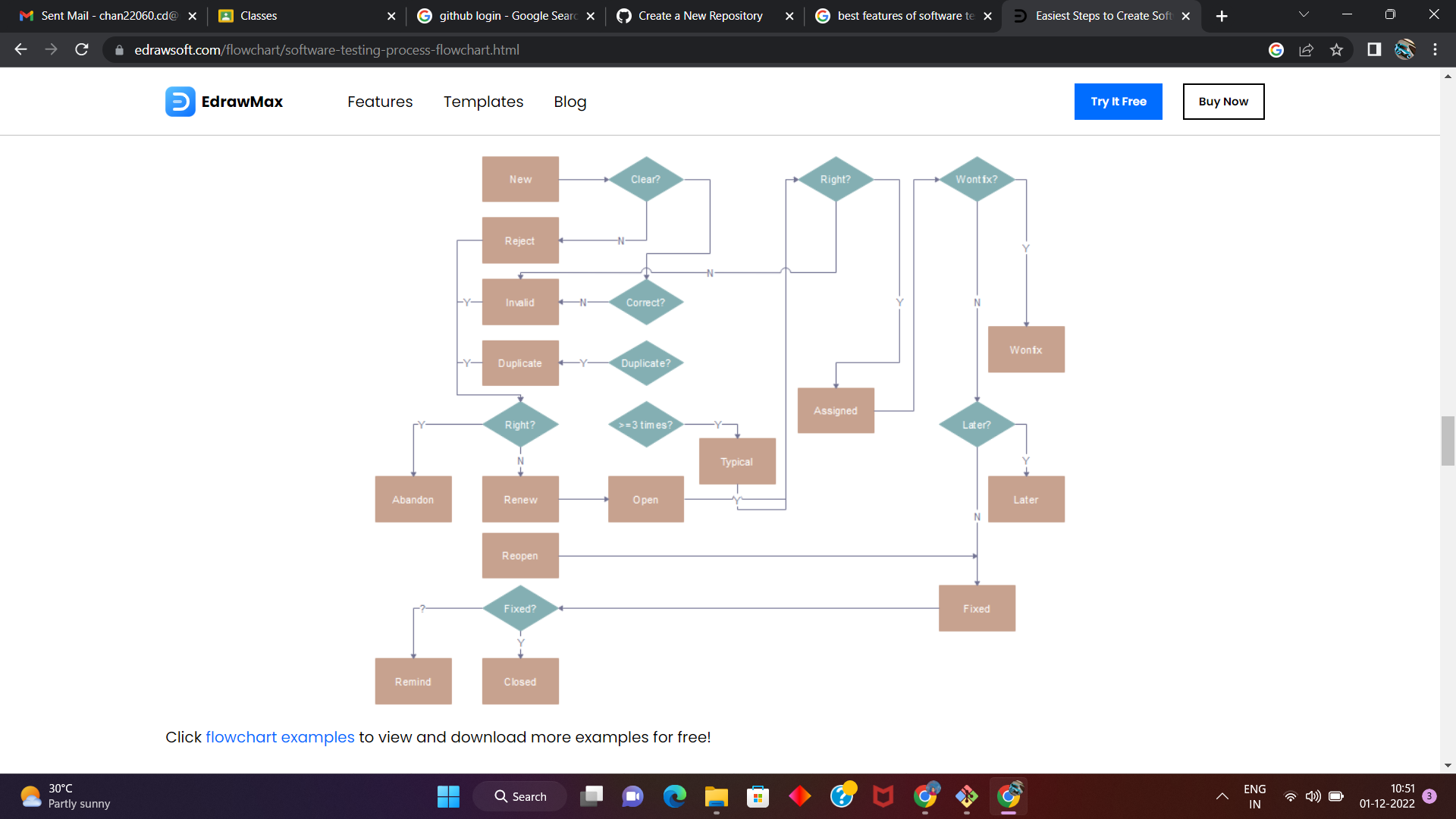
*Defect Management: Details how bugs will be reported, to whom and what each bug report needs to be accompanied by. For example, should bugs be reported with screenshots, text logs, or videos of their occurrence in the code?*

*Risk Management: Details what risks may occur during software testing, and what risks the software itself may suffer if released without sufficient testing.*

*Exit Parameters: Details when testing activities must stop. This part describes the results that are expected from the QA operations, giving testers a benchmark to compare actual results to.*

## **Product Features**

The software testing plan process flowchart:

**

## **User Classes and Characteristics**

* *Uncover as many as errors (or bugs) as possible in a*

*given product.*

* *Demonstrate a given software product matching its*
* *requirement specifications.*
* *Validate the quality of a software testing using the*

*minimum cost and efforts.*

* *Generate high quality test cases, perform effective*

*tests, and issue correct and helpful problem reports.*

## **Operating Environment**

*Operating environment for the software testing plan is as listed below:*

* *System and applications*
* *Test data*
* *Database server*
* *Front-end running environment*
* *Client operating system*
* *Browser*
* *Hardware includes Server Operating system*
* *Network*
* *Documentation required like reference documents/configuration guides/installation guides/ user manual*

## **Design and Implementation Constraints**

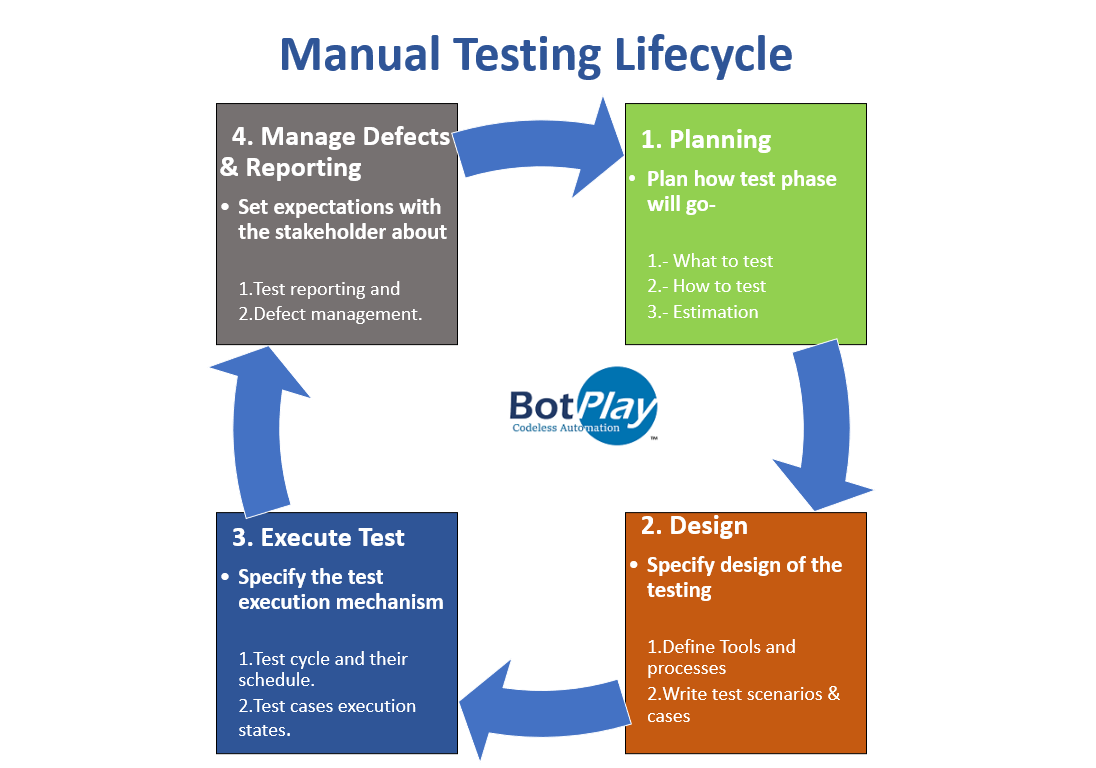
*Software design and implementation is the stage in the software engineering process at which an executable software system is developed.*

* *Software design and implementation activities are invariably inter-leaved. –*
* *Software design is a creative activity in which you identify software components and their relationships, based on a customer’s requirements.*
* *Implementation is the process of realizing the design as a program.*

## **Assumptions and Dependencies**

*Not everything is clear at the start of the project. Any assumptions or unclear requirements are specified in this section. Assumptions can be of any aspect of the project like resourcing, test environment, requirements, etc. Test Plan specifies assumptions to be made while testing the software.*

*Now for easy management of the testing phase, we can divide the test phase into four major sub-phases to complete the testing life-cycle and can draw a testing plan around it.*

**

# **System Features**

*<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>*

## **System Feature 1**

3.1.1 Description and Priority

*Priority can be defined as how fast or how early the defect should be addressed. The defects having highest priority should be fixed first followed by the defects having lesser priority.*

***Levels:***

*As with severity, priority levels may also differ in different companies. However it can be broadly classified in 3 levels.*

* ***High (Priority 1/P1)****: A defect which causes a significant damage to application is given a high priority. Defects having high priority should be fixed as soon as possible.*
* ***Medium (Priority 2/P2)****: Defects having medium priority should be fixed once high priority defects are addressed. Fixing of normal priority defects takes precedence over low priority defects.*
* ***Low (Priority 3/P3)****: Defects with low priority doesn’t impact the functionality much and they should be fixed once high and medium priority defects are addressed.*

3.1.2 Stimulus/Response Sequences

*A “blackbox” description of software requirements describes the behaviour of the system in terms of its external stimuli (inputs) and external responses (outputs). In general, every requirement is specified in terms of a relationship between an externally generated stimulus and an externally visible response. The advantages of using a “blackbox” approach for describing requirements include minimizing the potential for including internal design details in the specification and maximizing the suitability of the specification in testing the system’s software. Discouraging the inclusion of design details in the specification decreases the likelihood of overly constraining the design and makes the specification simpler to maintain as the design details may change as the project develops. Minimizing the software’s internal processing descriptions simplifies the development of “blackbox” test cases as the test engineers do not have to derive the requirements based test cases from descriptions of internal processing.*

3.1.3 Functional Requirements

* Helps you to check whether the application is providing all the functionalities that were mentioned in the functional requirement of that application
* A functional requirement document helps you to define the functionality of a system or one of its subsystems.
* Functional requirements along with requirement analysis help identify missing requirements. They help clearly define the expected system service and behavior.
* Errors caught in the Functional requirement gathering stage are the cheapest to fix.
* Support user goals, tasks, or activities

## **System Feature 2 (and so on)**

# **External Interface Requirements**

## **User Interfaces**

*The user interface for the software shall be compatible to any browser such as Internet*

*Explorer, Mozilla or Netscape Navigator by which user can access to the system.*

*The user interface shall be implemented using any tool or software package like Java*

*Applet, MS Front Page, EJB etc.*

## **Hardware Interfaces**

*e-Store Project Version: &lt;3.0&gt;*

*Software Requirements Specification Date: &lt;04/15//07&gt;*

*&lt;document identifier&gt;*

*Confidential &lt;Company Name&gt;, 12/1/2022 Page 14*

*Since the application must run over the internet, all the hardware shall require to connect*

*internet will be hardware interface for the system. As for e.g. Modem, WAN – LAN,*

*Ethernet Cross-Cable.*

## **Software Interfaces**

*1. The e-store system shall communicate with the Configurator to identify all the*

*available components to configure the product.*

*2. The e-store shall communicate with the content manager to get the product*

*specifications, offerings and promotions.*

*3. The e-store system shall communicate with billPay system to identify available*

*payment methods , validate the payments and process payment.*

*4. The e-store system shall communicate to credit management system for handling*

*financing options.*

*5. The e-store system shall communicate with CRM system to provide support.*

*6. The e-store system shall communicate with Sales system for order management.*

*7. The e-store system shall communicate with shipping system for tracking orders and*

*updating of shipping methods.*

*8. The e-store system shall communicate with external Tax system to calculate tax.*

*9. The e-store system shall communicate with export regulation system to validate*

*export regulations.*

*10. The system shall be verisign like software which shall allow the users to complete*

*secured transaction. This usually shall be the third party software system which is widely*

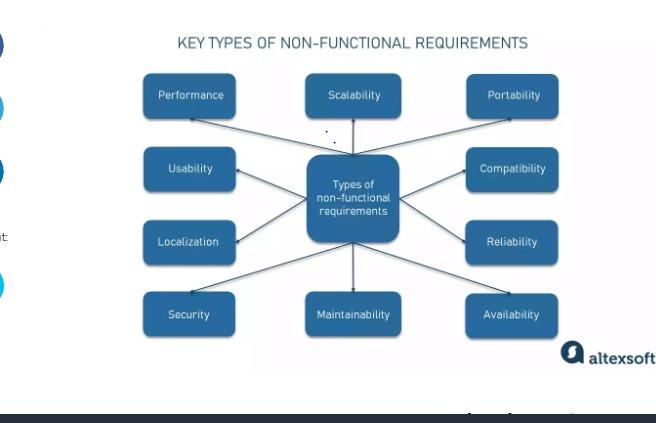
*used for internet transaction.*

## **Communications Interfaces**

*The e-store system shall use the HTTP protocol for communication over the internet and*

*for the intranet communication will be through TCP/IP protocol suite.*

# **Other Nonfunctional Requirements**



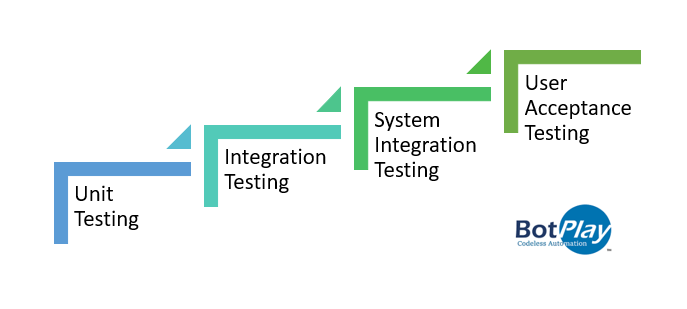
# **Other Requirements**

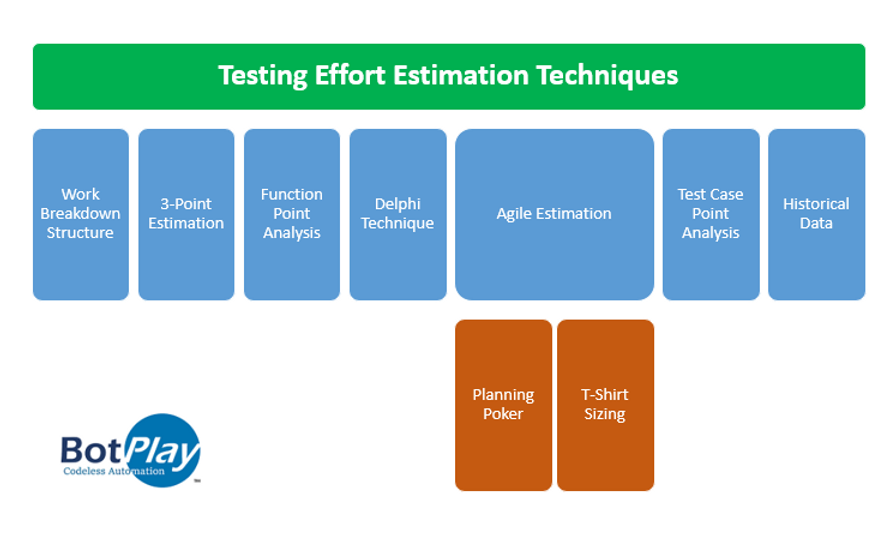
**Appendix A: Glossary**

*A systematic and properly planned end-to-end testing plan ensures achieving the goals of the testing phase effectively and efficiently. Without a test plan, the process will not only become chaotic, slow, and unaccountable. Therefore, the formulation of a thorough and ordered test plan will be beneficial for the project and the quality of the software.*

**Appendix B: Analysis Models**

**Levels of software testing:**

**Software testing effort estimation techniqu**

****

**Appendix C: Issues List**

***Project Risk*** *- An activity that can hinder the advancement of the project.*

***Product Risk*** *- The probability of a system failure that may not be able to meet the requirements and expectations of the client.*

***Impact analysis of the Risk*** *- The impact of the identified risk is analyzed to determine the extent of impact ranging from High, Medium, or Low Impact.*

***Mitigation of the Risk*** *- It involves choosing strategies that make the risk negligible or at least having as least impact as possible. To be able to efficiently do this, the risk must be properly documented and accessible to all stakeholders and team members. To ensure that risk is mitigated, it must be continuously monitored and controlled.*