# Test Strategy Document

It is a high-level document which is set at organization level and can be used by multiple projects to achieve testing objectives. A test strategy document describes the testing approach of the software development cycle and sets standard for testing process.

Other documents such as the Test Plan draws its contents from those standards set in the Test Strategy Document

## Software Development Methodology

We adopted "Agile software development- Scrum" which based on iterative and incremental development approach, where requirements and solutions evolve via collaboration between self-organizing cross-functional teams. It’s a holistic product development strategy where a team irrespective of their functional role as Software Consultant or Quality analyst works as a unit to reach the product deliverables. So, software development and its verification goes hand in hand.

During requirement phase, Scrum master gathers all requirements prepare BRD [Business Requirement Documents]. After review and sign off from stakeholder product backlog will be created by creating User story against each product feature. Based on business priority few chunks i.e. User Stories from Product backlog will be chosen as Sprint backlog. Whole development team will have brainstorming session for Sprint Planning where detail discussion about user story for design, development and verification will happen along with respective roles and responsibilities assigned and respective efforts will be agreed. The respective sprint will be opened and should be completed as per defined timelines and finally shipped to end user.

## Test Planning

The Test Plan document defines the scope and approach of a project. It basically emphasis on a software testing effort. It is usually prepared against a Project to describe what to test, how to test, who will test and when to test. It has below components

### Introduction -

Small introduction about the documents and product domain

### Roles and Responsibilities

* Testing Manager – Test Manager is responsible for defining the testing activities and to check that if the team has all the necessary resources to execute these activities.
* Testing Lead –Test lead needs to ensure that testing is going parallelly with the software development in all phases. He also carries out all the responsibilities of test planning and execution as per time lines.
* Test Designer – This is the person responsible for creating the test scripts, scenarios and so on that make up the tests to be performed. There should be different test designers for different Functional [Primary Owner] and Non-Functional testing [Secondary Owner]
* Test Approver – This is the person responsible for reviewing, validating, and approving the test artefacts created by the Test Designer.
* Tester – This is the person responsible for executing the test scripts, and reporting the results. There should be separate Testers for Functional [Primary Owner] and Non-functional test [Secondary Owner] execution.
* Reviewer – This is the person responsible for reviewing reports from the testers and determining what subsequent actions will be taken.

### Test Items

List of the test items i.e. software/products and their versions

### Features to be tested

List of the features of the software/product to be tested. Its references should be provided to the Requirements and/or Design specifications of the features to be tested

### Features not to be tested

List of the features of the software/product which will not be tested. Against those features reasons these features won’t be tested should be specified.

### Entry criteria and Exit criteria

* **Entry criteria** - It defines the activities or specific conditions that must be present before a process/Test activity can begin such as:

e.g. Verify if the Test environment is available and ready for use.

Verify if test tools installed in the environment are ready for use.

Verify if Testable code is available.

Verify if Test Data is available and Validated for correctness of Data

This also includes the input from the development team such as Design document and Business requirement document

* **Exit criteria** - Exit criterion can be defined for all the test activities right from planning specification and execution. Basically, it determines whether the test activities are completed or NOT

**e**.g. Verify if All tests planned have been run.

Verify if the level of requirement coverage has been met.

Verify if there are NO Critical or high severity defects that are left outstanding.

Verify if all high-risk areas are completely tested.

Verify if software development activities are completed within the projected cost.

Verify if software development activities are completed within the projected timelines.

### Suspension and Resumption criteria

* **Suspension Criteria** - A very high sever incident which stops testing activities or further testing does not add value to testing efforts.
* **Resumption Criteria** – Once a fix for the problem has been delivered or new major version of the software deployed

### Features pass or fail criteria i.e. Acceptance Criteria

These are minimum expectations or conditions which a software application should satisfy to be accepted by a user or customer. Acceptance criteria to be defined at each feature

### Test deliverables

Test Deliverables are the entire document which are designed/maintained during testing starting from kick off to the sign off phase. Some test deliverables are provided before testing phase, some are provided during the testing phase and some after the testing cycles is over. Below are the artefacts which should be submitted:

Test Plan, Test Scenarios, Test Cases – containing Test Data, Test Scripts, Test Suite, Test Results/Test Reports, Defect Logs/Enhancement Logs, Release notes

### Test Estimates

Provide a summary of test estimates (in man hours or man days) and/or provide a link to the detailed estimation with respect to User story.

### Risk and Mitigation

Risk is exposure to a chance of loss or damage to the system also can be a potential threat to the system. This may vary from project to project so list down all the risks associated with people management, tools and technology related, environment related etc.

Risk mitigation are the steps to be taken to reduce adverse effects. Proactive actions to be taken at project level to minimize the risk. E.g. Creating a backup plan for the resource to share work load, arrangement of regular Knowledge Transfer sessions.

## Test Artefacts

We will have two separate approaches for testing viz.

### Functional Testing –

Currently we do not have scope for Web Services or API testing, so testing a web application will be a prime focus. For functional testing of web application below test artefacts should be maintained

* **Test Scenarios** - As per assigned project, Test Designer will analyze the BRD/User story to create different **Test Scenarios**.
* **Test Cases** - The given scenario will be break down to detailed test cases. Each test case will have Test Case Id, Summary, Prerequisites, Detail steps, Test data, Expected Result.
* **Test Data** - Based on input parameters and input validation criteria, using Test Data Design Techniques test data will be agreed. [For single input - Boundary Value Analysis and Equivalence Partitioning. For multiple inputs – Decision Table, State Transition and Use case techniques to be used]
* **Test Result** – On subsequent each build, given Test cases will be executed and its results should be broadcasted in project team and reporting manager.
* **Defect Log/Enhancement Log** – During test execution defect identified must be logged in tracking system with proper priority and root cause. For better quality or improvement valid suggestions are always welcome with proper information.
* **Test Script** – Each test case will undergo Feasibility Analysis for automation. Based on Business value, respective test cases will be automated.
* **Test Suite** – Test suite is a collection of test scripts which are functionally co-related and used for verification of given Test Case or User Story or System module.
* **Release Note** – After a sprint backlogs implemented and demonstrated to stakeholder, product should be delivered. So against each shipment below information should be broadcasted: Issues fixed, Enhancements, Known defects

### Non-Functional Testing –

Under non-functional testing for web application Compatibility Testing, Performance Testing, Load Testing and Security Testing will be in scope.

**Compatibility Testing** - During execution for Compatibility testing above mentioned functional Test Cases will be executed across all the browsers so no any other test artefacts are required.

**Performance Testing** - For Performance testing separate **Test Scenarios** will be created but no any Test Data required and after execution **Test Results** to be broadcasted

**Load Testing** - For Load testing test scenario will be same as that of Performance Testing but should have separate **Test Data** for No of Users/Request and after execution **Test Results** to be broadcasted

## Test Methodologies with Scope

### Manual Testing –

Under manually testing, test artefacts are documented and executed manually. Manual testing should have functional test coverage as well as non-functional test coverage.

#### Functional Testing –

All features should be documented in form test scenarios/test cases. Respective test cases should be executed when build is available.

During functional testing, whenever required data validation on UI then Database Testing should be performed by designing appropriate SQL queries.

#### Non-Functional Testing –

During non-functional testing below testing types should be considered:

**Compatibility Testing** – Application\*[Developed onwards Jan 2017] should be cross browser functional. All test cases should be executed on below browser and respective versions

Internet Explorer – IE 11, Google Chrome – Latest Version, Mozilla Firefox – Latest Version

**Performance Testing** – Response time for different Page loading and control loading especially grids with large number of rows [with multiple paging] and comparing with expected one

**Load Testing** – Under load test below scenarios to be covered.

1. Capacity Test – Determines the maximum number of concurrent users that the application server can support under a given configuration while maintaining an acceptable response time

2. Consistent Load Test – Long-running stress test that drives a continuous load on the application server for an extended period (at least n hours).

The main purpose of this type of test is to ensure the application can sustain acceptable levels of performance over an extended period without exhibiting degradation.

3. Single Function Stress Test - A test where 100\*n users perform the same function with no wait times and no ramp up time.

This test will help determine how the application reacts to periods of extreme test in a very narrow area of the code.

4. Graphical analysis of the results and Summary Reports based on input assertions

**Security Testing** – Under Security testing below test to be carried out:

1. Cross site scripting - In Cross Site Scripting (XSS) testing, we test if user can inject client-side scripts into web pages which may disturb expected behavior of application.
2. SQL Injection - In SQL injection, we try to identify SQL loop holes in web application i.e. confirm user inputs provided not directly retrieving or manipulating the backend data
3. Penetration testing/ Broken Authentication and Session Management/ Sensitive Data Exposure/ Invalidated Redirects and Forwards

**Configuration Testing** – Configuration testing is the process of testing the system with each one of the supported software and hardware configurations. Viz. Operating system and machine processor. Operating Systems - Win XP, Win 7-32 bit, Win 7-64 bit, Win 8-32 bit, Win 8-64 bit, Linux, Mac. But currently we have scope only for Win 7-64 bit.

Devices such as Mobiles [Android, iOS, BlackBerry OS, Symbian, Palm OS] are out of scope as we are not supporting our applications on mobiles/hand held devices.

### Automated Testing –

Under automated testing below are phases evolved and executed.

**Feasibility analysis** – It determines what to automate and what cannot be automated. Selenium WebDriver can automate only browser. So, any feature beyond the browser actions cannot be automated. In such scenarios, using third party tools we can handle features like Manipulating test data file [Apache POI], handling windows pop up [Auto IT/Robot class], Reading and comparing images [Sikuli]. But few feature like Sending/Receiving or Reading an email cannot be automated, reading a captcha no one cannot automate

**Framework analysis** – Choosing a framework which will suffice all required automated actions. In case additional support required then maintaining the existing framework

**Object Repository creation** – All web elements across the application which are to be automated should be stored in non-source file. This will have all the locators for given web element as per the precedence i.e. [Id/Name/Link/Partial Link/Class name/Tag name/CSS Selector/X Path]

**Modular Design** – As per test steps defined in manual test cases, different actions on the web pages are broken down in smaller methods. Those methods to be called from respective test cases.

**Test Scripting** – Actual coding for browser actions and calling from related methods. Grouping all functional test cases in final test suite

**Test Execution and Reporting** – Executing the test suite and maintaining the test log against the execution. Final test result of individual test case/test method pass/fail/skipped to be broadcasted within team

## Testing Levels

### Unit Testing –

In Unit Testing individual units/components of a software/system are tested. White Box Testing method is used for executing the unit test. Unit testing should be done by the programmers either manually or using automated test cases for all possible test items [client side code, sever side code, database objects] and executed in Development environment.

### Integration Testing –

In integration testing individual modules are combined and tested as a group. Once the modules are unit tested, they are integrated one by one, till all the modules are integrated, to check the behavior, and validate whether the requirements are implemented correctly or not. This involves functional testing to ensure data and control flow within the application system from one module to others.

### System Testing –

System testing involves functional as well as non-functional testing as mentioned above. The testing of an entire integrated system is done to verify that it meets the specified requirements. The responsibility lies with testing team and performed on test environment.

### Acceptance testing –

Acceptance testing is basically done to ensure that the requirements of the specification are met. Acceptance testing involves two phase. First phase is Alpha testing. Alpha testing is done by the testing team in testing environment prior to release to stakeholder. Second phase is Beta testing and it is done by the stakeholder on production environment.

Beside these levels of testing, **Regression testing** must be carried at **bug level** and **sprint level** when development/fixing is frozen to ensure the changes in code does not have any ill-impact on existing working and tested features.

## Testing Tools and Technology

Below are the tools and technology used during test life cycle:

### Manual Testing

* **Manual Test Management Tool** – Team Foundation Server 2015

All artefacts are maintained in Team Foundation Server with the hierarchy as

Project – Phase/Sprint – User Story – Test Cases - Defects

### Automated Testing

For Automated testing below are tools and technologies used.

* **Programming language** – Java, Third party jars like Apache POI, Log4j etc.
* **Integrated Development Environment** – Eclipse Neon 2
* **Version Control** – Team Foundation Server 2015
* **Build Tools** – Maven, ANT
* **Testing Tools/API** – Selenium WebDriver 2.53, TestNG, XSLT and Extent Report, other third party jars
* **C. I. C. D. Tool** – Jenkins [Jenkins as a Windows Service]
* **Automated Test Execution** – Jenkins [Jenkins as WAR]

## Test Environments

### Manual Testing

#### For manual test execution

As per current scope we require system installed with Win 7-64 bit, RAM 8 GB

All required browsers [IE Version 11, Google Chrome latest Version, Mozilla Firefox latest Version]

System registered under Domain capita.co.in

Commercial Off the shelf product – Microsoft Office 2016

Valid Domain credentials under Domain capita.co.in

### Automated Testing

#### For automated test development

System with above Manual Testing prerequisites

Java Platform – JDK Windows 64-bit Version 1.7 or above

All required browser’s driver for Selenium

All testing tools mentioned under Testing Tools and Technology

#### For automated test execution

System with above Manual Testing prerequisites

Java Platform – JRE Windows 64-bit Version 1.7 or above

All required browser’s driver for Selenium

Project Build [target folder from Automation team] against which automated suite to be executed

## Defect Management

All defect found to be logged in Test Management Tool. For defect reporting/logging we follow below Defect Life Cycle.

### New –

This is initial or default status for the bug. When the defect is identified, tester will log the defect in Test Management Tool and will have this status. All defects now been assigned to Developer’s Technical Lead or auto assigned to respective Module Lead/Owner. Next possible status will be “In Progress”.

New 🡪 In Progress

Reasons: Implementation Started

### In Progress –

Based on prioritization, Developer’s Technical Lead will assign the bug to available developer. When developer starts fixing the same he/she should update the status to “In Progress”.

If developer holds the current issue or need to divert to other high prior task, then he will update the status as “New” again. Next possible status be New or will be for QA

In Progress 🡪 New

Reasons: Stopped Progress, On Hold

In Progress 🡪 Won’t Fix

Reasons: Not feasible, Technical Limitation, Technical Overhead

In Progress 🡪 QA for Invalid defect

Reasons: Invalid environment, Invalid test scenario or data, Lack of Product Knowledge

In Progress 🡪 QA for Fixed defect

Reasons: Code fixed

In Progress 🡪 QA for Duplicate

Reasons: Known Issue -

### QA –

After marking “In Progress”, developer will analyze the defect and respective action will be taken. Based on the analysis and corrective action for the defect the defect will be tagged with Resolution as:

#### Won’t Fix –

If it’s a technical limitation and cannot be addressed or fixed. Next possible status should not be available.

#### Invalid –

In case the said bug’s corrective action violating the intended functionality of the module/system or the bug is reported against invalid test scenario/test case/test data. Next possible status will be Closed.

QA for Invalid 🡪 Closed

Reasons: Agreed by QA

#### Fixed –

In case the defect was valid, its corrective action taken. Next possible status will be Resolved or Reopened.

QA 🡪 Resolved

Reasons: Working Ok

QA 🡪 Reopened

Reasons: Issue still exists, Bad fix

#### Duplicate –

In case if its already known issue and logged in Test Management Tool. Assignee should find the original defect and mentioning against current defect must be mandatory. Next possible status will be Closed.

Duplicate 🡪 Closed

Reasons: Agreed by QA

Once defect is updated with status as “QA”, the corrected version should be available on test environment

### Resolved –

Tester will retest the bug once the bug is found corrected and no conflict in expected and actual behavior then bug updated as Resolved. Next possible status will be Closed.

Resolved 🡪 Closed

Reasons: Sprint Released

### Reopened –

If in case during retesting bug still exists or there is still conflict between actual and expected, then bug should be marked as Reopened. After Re-opened it will undergo the next status change as In-Progress and cycle continues.

Reopened 🡪 In Progress

Reasons: Implementation Re-Started

### Closed –

Once all the defects in sprint are Resolved and when its shipped then all bugs will be marked as Closed. This will be last status in the bug life.

During Regression cycle, even Closed defect also can be Reopened which can be from same sprint or previous sprints