\*Test Plan and Procedures

\*Introduction

In order to avoid ending up with a buggy system, building a test plan is more than important. A well thought out and organized test plan helps one to take a methodical approach to spot and correct errors. In this chapter, the objectives and scope testing with different test strategies will be discussed. Unit, integration, system, security, performance, usability, cross browser testing etc. are some of these test strategies.

\*Objectives

The objectives of creating a test plan are :-

* To confront the challenges and focusing on the major issues
* To describe the strategies for creating a system test case
* To identify any bugs and verify all functions
* To validate the system against its principles
* To manage change and adapt to any adverse situation
* To prevent unnecessary expenses by developing a buggy system.

\*Scope

The specific part of a system where testing is required is known as scope such as all the functions of a specific product, its existing interfaces, integration of all functions etc. Test scope is influenced by various factors, e.g. size, complexity, budget, time and so on. All of these aspects will be thoroughly checked and revised.

\*Functions to be tested

The functions that require to go through the testing procedures are :-

* Security of the login system
* Validity and usability of user inputs
* Search
* Manipulation of information in the database
* Confirmation of user actions

\*Functions not to be tested

The functions that are not to be tested:

* Every code of SQL commands
* Outputs for every SQL commands individually
* Outputs for each pages

Being a rather elaborate project, this project is comprised of a lot of functions. So, it’s not possible to test all of these functions separately. Which is why, some specific areas of the system were chosen.

\*Test Strategy

This includes the aforementioned types of tests that are needed to be carried out to mark and correct errors in the system. These are explained below:

\*\*Unit Testing

Unit testing is a method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use. (Adam Kolawa; Dorota Huizinga; 2007). This will enable us to verify that our functions work and also help to identify the issues of our logic and algorithms that are comprised of certain functions. Unit testing will be done each time when a new block of code is added. Samples are as follows :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | |  |  | |
|  | |  |  | |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | |  |  | |
|  | |  |  | |
|  |  |  |  |  |
|  |  |  |  |  |

\*\*Integration Testing

Integration testing, also known as I&T, is a phase of the testing process where the program units are combined and tested as groups through various processes. Integration testing can expose problems with the interfaces among program components before trouble occurs in real-world program execution. It’s carried out in two major ways. These are :

1. Bottom-up Integration :  begins with unit testing, followed by tests of progressively higher-level combinations of units called builds.

2. Top-down Integration :  the highest-level modules are tested first and progressively lower-level builds are tested after that.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | |  |  | |
|  | |  |  | |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | |  |  | |
|  | |  |  | |
|  |  |  |  |  |
|  |  |  |  |  |

\*\*System Testing

System testing is conducted on a complete, integrated system to evaluate whether it has met necessary requirements and specifications. System testing is a limited type of testing, which falls under the Black box testing category of software testing; it seeks to detect defects within the system as a whole. Both functional and non-functional requirements are to be tested here.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | |  |  | |
|  | |  |  | |
|  |  |  |  |  |
|  |  |  |  |  |

\*\*Security Testing

Security testing is the phase of a test plan where vulnerabilities of a system are detected in order to prevent massive loss of information. A slight flaw in this part might lead to unauthorized access, thus the confidentiality and integrity of the system might be breached. So, security testing is crucial. It is a kind of non-functional testing.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | |  |  | |
|  | |  |  | |
|  |  |  |  |  |
|  |  |  |  |  |

\*\*Cross Browser Testing

Cross browser testing refers to the ability to test a web application across different browsers to check how the application functionality behaves across all the tested browsers. This is done in order to find out any discrepancies in the system. Because of the differences in the way web browsers interpret HTML and JavaScript, this test is necessary. The system will be run on various web browsers so that errors can be found and dealt with.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | |  |  | |
|  | |  |  | |
|  |  |  |  |  |
|  |  |  |  |  |

\*\*Usability Testing

Usability testing refers to the evaluation of the system through real users, where they are asked to complete tasks, to see whether they encounter problems and confusions etc. It is required to check whether it meets its intended purpose. This is a very important phase in testing where necessary plans are made to make changes regarding the ease of accessibility and improvement in performance of the user.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | |  |  | |
|  | |  |  | |
|  |  |  |  |  |
|  |  |  |  |  |

\*\*Performance Testing

Performance testing is done to determine the speed and effectiveness of a system under a specific amount of workload. It is a kind of non-functional testing. The attributes of performance testing are : Speed, Scalability, Stability and Reliability.

Performance testing is necessary for Improving user experience on sites and web apps, increasing revenue generated from websites, gathering metrics useful for tuning the system, identifying bottlenecks such as database configuration, determining if a new release is ready for production and finally providing reporting to business stakeholders regarding performance against expectations.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | |  |  | |
|  | |  |  | |
|  |  |  |  |  |
|  |  |  |  |  |

\*\*Test Environment

A testing environment is a setup of software and hardware on which the testing team is going to perform the testing of the newly built software product. This setup consists of the physical setup which includes hardware, and logical setup that includes Server Operating system, client operating system, database server, front end running environment, browser (if web application), IIS (version on server side) or any other software components required to run this software product. This testing setup is to be built on both the ends – i.e. the server and client.

\*\*Test Schedule

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

\*\*Summary

The aforementioned test strategies are sufficient to check the basics of the system and resolving errors if found. In the end of this chapter, further analysis on the test environment and the schedule of the test are also provided.