**CHAPTER 7**

**SYSTEM TESTING**

System Testing is a level of the software testing where complete and integrated software is tested. The purpose of this test is to evaluate the system's compliance with the specified requirements. By definition of ISTQB system testing is the process of testing an integrated system to verify that it meets specified.

**7.1 TESTING METHODS**

Software Testing Type is a classification of different testing activities into categories, each having, a defined test objective, test strategy, and test deliverables. The goal of having a testing type is to validate the Application under Test for the defined Test Objective.

For instance, the goal of Accessibility testing is to validate the AUT to be accessible by disabled people. So, if your Software solution must be disabled friendly, you check it against Accessibility Test Cases.

**7.2 TYPES OF TESTING**

**7.2.1 Unit Testing**

In computer programming, unit testing is a software testing 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.

In this banking bot, every units of code is been tested and the correctness of every module is been ensured.

**7.2.2 Integration Testing**

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

In this bot, the units are been tested as a whole and the testing was successful.

**7.2.3 Functional Testing**

Functional testing is a quality assurance (QA) process and a type of black-box testing that bases its test cases on the specifications of the software component under test. Functions are tested by feeding them input and examining the output, and internal program structure is rarely considered (unlike white-box testing). Functional testing usually describes what the system does. Functional testing does not imply that you are testing a function (method) of your module or class. Functional testing tests a slice of functionality of the whole system.

Functional testing has many types:

* Smoke testing
* Sanity testing
* Regression testing
* Usability testing

**7.2.4 Stress Testing**

Stress testing a Non-Functional testing technique that is performed as part of performance testing. During stress testing, the system is monitored after subjecting the system to overload to ensure that the system can sustain the stress.

Reasons can include:

* to determine breaking points or safe usage limits
* to confirm mathematical model is accurate enough in predicting breaking points or safe usage limits
* to confirm intended specifications are being met
* to determine modes of failure (how exactly a system fails)
* to test stable operation of a part or system outside standard usage

The recovery of the system from such phase (after stress) is very critical as it is highly likely to happen in production environment.

In this banking bot, whole of the modules are been tested and it has the safe usage measures.

**7.2.5 Acceptance Testing**

Acceptance Testing is a level of the software testing where a system is tested for acceptability. The purpose of this test is to evaluate the system’s compliance with the business requirements and assess whether it is acceptable for delivery.

Formal testing with respect to user needs, requirements, and business processes conducted to determine whether or not a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system.

In this bot, the customer’s acceptance is been monitored and it is been put into usage.

**7.2.6 White Box Testing**

White Box Testing is the testing of a software solution's internal coding and infrastructure. It focuses primarily on strengthening security, the flow of inputs and outputs through the application, and improving design and usability. White box testing is also known as Clear Box testing, Open Box testing, Structural testing, Transparent Box testing, Code-Based testing, and Glass Box are testing. It is one of two parts of the "box testing" approach of software testing. Its counter-part, black box testing, involves testing from an external or end-user type perspective. On the other hand, White box testing is based on the inner workings of an application and revolves around internal testing.

The term "white box" was used because of the see-through box concept. The clear box or white box name symbolizes the ability to see through the software's outer shell (or "box") into its inner workings. Likewise, the "black box" in "black box testing" symbolizes not being able to see the inner workings of the software so that only the end-user experience can be tested.

In this banking bot, all the inner functionality is been tested and it is been correctly implemented.

**7.2.7 Black Box Testing**

Black box testing is a software testing techniques in which functionality of the software under test (SUT) is tested without looking at the internal code structure, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on the software requirements and specifications.

In this banking bot, the implementation part is been checked for its correctness.

**7.2.7.1 Methods of Black Box Testing**

There are many types of Black Box Testing but following are the prominent ones -

* Functional testing - This black box testing type is related to functional requirements of a system; it is done by software testers.
* Non-functional testing - This type of black box testing is not related to testing of a specific functionality, but non-functional requirements such as performance, scalability, usability.
* Regression testing - Regression testing is done after code fixes, upgrades or any other system maintenance to check the new code has not affected the existing code.

**7.3 TESTING STRATEGY**

Test Strategy is also known as test approach defines how testing would be carried out. Test approach has two techniques:

* Proactive - An approach in which the test design process is initiated as early as possible in order to find and fix the defects before the build is created.
* Reactive - An approach in which the testing is not started until after design and coding are completed.

Test strategy calls for implementing two entirely different methodologies for testing this project. The college management includes a fair amount of manual UI-based testing.

In this college management, proactive approach is been used for testing. Since proactive approach is efficient it is been used in this bot.