**4. Testing**

**Introduction**

Testing is the process of finding differences between the expected behavior specified by system models and the observed behavior implemented system. From modeling point of view, testing is the attempt of falsification of the system with respect to the system models. The goal of testing is to design tests that exercise defects in the system and to reveal problems. The process of executing a program with intent of finding errors is called testing.

**4.1 Test Plan**

Testing process starts with a test plan. This plan identifies all the testing related activities that must be performed and specifies the schedules, allocates the resources, and specified guidelines for testing. During the testing of the unit the specified test cases are executed and the actual result compared with expected output. The final output of the testing phase is the test report and the error report.

**Test Data**

Here all test cases that are used for the system testing are specified. The goal is to test the different functional requirements specified in Software Requirements Specifications (SRS) document.

**Unit Testing**

Each individual module has been tested against the requirement with some test data.

**Test Report**

The module is working properly provided the user has to enter information. All data entry forms have tested with specified test cases and all data entry forms are working properly.

**Error Report**

If the user does not enter data in specified order then the user will be prompted with error messages. Error handling was done to handle the expected and unexpected errors.

Software Testing Techniques

Software testing is the critical element of the software quality assurance and represents the ultimate review of specification, designing and coding.

Testing Objectives

* Testing is the process of executing a program with the intent of finding an error.
* A good test case design is one that has a probability of finding an as yet undiscovered error.
* A successful test is one that uncovers as yet undiscovered error.
* Testing cannot show the absence of defects, it can only show that software errors are present.

**4.2 Testing Methodology**

1. Black box or functional testing
2. White box testing or structural testing

**Black box testing**

This method is used when knowledge of the specified function that a product has been designed to perform is known. The concept of black box is used to represent a system whose inside workings are not available to inspection. In a black box the test item is a "Black" , since its logic is unknown , all that is known is what goes in and what comes out , or the input and output. Black box testing attempts to find errors in the following categories:

.

As shown in the following figure of Black box testing, we are not thinking of the internal workings, just we think about

What is the output to our system?

What is the output for given input to our system?

**?**

Input

Output

**Figure 4.2.1:** The Black box is an imaginary box that hides its internal workings.

**White box testing**

White box testing is concerned with testing the implementation of the program. The intent of structural is not to exercise all the inputs or outputs but to exercise the different programming and data structure used in the program. Thus structural testing aims to achieve test cases that will force the desire coverage of different structures.

**INTERNAL WORKING**

Input

Output

**Figure 4.2.2:** The White Box testing strategy, the internal workings.

**4.3 Software Testing Strategies**

A software testing strategy provides a road map for the software developer. Testing is a set of activities that can be planned in advance and conducted systematically. For this reason, a template for software testing a set of steps into which we can place specific test case design methods should be defined in software engineering process. Any software testing strategy should have the following characteristics.

**Unit Testing**

Unit testing focuses on the building blocks of the software system, that is, objects and sub system. There are three motivations behind focusing on components. First, unit testing reduces the complexity of the overall tests activities, allowing us to focus on smaller units of the system. Second, unit testing makes it easier to pinpoint and correct faults given that few components are involved in this test. Hence the goal is to test the internal logic of the module.

**Integration Testing**

In the integration testing, many test modules are combined into sub systems, which are then tested. The goal here is to see if the modules can be integrated properly, the emphasis being on testing module interaction. After structural testing and functional testing we get error free modules. These modules are to be integrated to get the required results of the system. After checking a module, another module is tested and is integrated with the previous module.

**System Testing**

In system testing the entire software is tested . The reference document for this process is the requirement document and the goal is to see whether the software meets its requirements. The system was tested for various test cases with various inputs.

**Acceptance Testing**

Acceptance testing is sometimes performed with realistic data of the client to demonstrate that the software is working satisfactory. Testing here focus on the external behavior of the system, the internal logic of the program is not emphasized. In acceptance testing the system is tested for various inputs.

**4.4 Test Cases**

A test case is a set of input data and an expected result that exercises the component with the purpose of causing failures and detecting faults. Test cases are classified into black box test and white box test. Black box test focus on input/output behavior of the component. White box test focus on internal structure of the components.

**Test case for admin Login:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test.no** | **Input** | **Expected o/p** | **Observed o/p** | **Status** |
| 1 | user name, password | Displaying respected home page | Displayed requested home page | Pass |
| 2 | On entering wrong user name and password | Error should be displayed in the login screen and display login screen again | Displayed login page again | Pass |

**Test case for user registration**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test.no** | **Input** | | **Expected o/p** | **Observed o/p** | **Status** |
| 1 | Enter the values to all the mandatory fields | | Create user when no such id exists, otherwise display user already exists | User registered successfully | Pass |
| 2 | | On entering the wrong details or left the fields | Display message user not created | User not created | Pass |

4.5 System Security

The protection of computer-based resources that include hardware, software, data, procedures and people against unauthorized use or natural Disaster is known as System Security.

System Security can be divided into four related issues:

* Security
* Integrity
* Privacy
* Confidentiality

**System Security**

System Security refers to the technical innovations and procedures applied to the hardware and operation systems to protect against deliberate or accidental damage from a defined threat.

**Data Security**

Data Security is the protection of data from loss, disclosure, modification and destruction.

**System Integrity**

System Integrity refers to the power functioning of hardware and programs, appropriate physical security and safety against external threats such as eavesdropping and wiretapping.

**Privacy**

Privacy defines the rights of the user or organizations to determine what information they are willing to share with or accept from others and how the organization can be protected against unwelcome, unfair or excessive dissemination of information about it.

**Confidentiality**

Confidentiality is a special status given to sensitive information in a database to minimize the possible invasion of privacy. It is an attribute of information that characterizes its need for protection.

**Secured Software**

It is the technique used for the purpose of converting communication. It transfers message secretly by embedding it into a cover medium with the use of information hiding techniques. .NET has two kinds of security:

* Role Based Security.
* Code Access Security.

The Common Language Runtime (CLR) allows code to perform only those operations that the code has permission to perform. So, CAS is the CLR's security system that enforces security policies by preventing unauthorized access to protected resources and operations.