8.1 Introduction

Purpose

Testing is an investigation conducted to provide stakeholders with information about

the quality of the product or service under test. Software testing also provides an objective,

independent view of the software to allow the business to appreciate and understand the risks

of software implementation. Test techniques include, but are not limited to, the process of

executing a program or application with the intent of finding software bugs. Software testing

can also be stated as the process of validating and verifying that a software program or

application or product:

1. Meets the business and technical requirements that guided its design and development;

2. Works as expected; and

3. Can be implemented with the same characteristics

**Scope of Testing**

Software testing, depending on the testing method employed, can be implemented at any time in the development process. However, most of the test effort occurs after the requirements have been defined and the coding process has been completed. As such, the methodology of the test is governed by the software development methodology adopted. Different software development models will focus the test effort at different points in the development process. Newer development models, such as Agile, often employ test driven development and place an increased portion of the testing in the hands of the developer, before it reaches a formal team of testers. In a more traditional model, most of the test execution occurs after the requirements have been defined and the coding process has been completed.

**8.1.1 Test Plan**

To test this application we are going with proper sequencing of testing like unit, integration,

validation, GUI, Low level and High level test cases, major scenarios likewise. We will go with the GUI testing first and then integration testing. After integration testing performs the high level test cases and major scenarios which can affect the working on the application. We will perform the testing on the data transmitted using the various inputs and outputs and validate the results. It also intends to cover any deviations that the project might take from the initially agreed Test Strategy in terms of scope, testing methodology, tools, etc.. This test plan covers details of testing activities for this project and scope.

**8.1.2 Software to be tested**

1. Edraw Max:

It enables students, teachers and business professional store liable create and publish various kinds of diagram store present any ideas. With this application users can easily create professional- looking flow charts, organizational charts, network diagrams, business presentations, building plans, mind maps, science illustration, fashion designs, UML diagrams and much more.

**2. Star UML:**

Star UML is a fully fledged, open source, UML modeling tool thats supports the ability to create software designs, from basic concepts, through to the coded solution. The user should be aware that this tool is more complex than a simple UML diagram editing tool, in that, through the use of the model Drive Architecture (MDA) standard, the tool supports complex modeling which is realizable in code.

**8.2 Test Cases**

* GUI Testing

Graphical User Interface (GUI) testing is the one of the mechanism in which user interface developed System Under some graphical rules. GUI testing includes checking various controls- menus, buttons, icons, dialog boxes and windows etc. Proposed system is tested for user inputs against different modules, validations are done. GUI is tested for appearance of different controls, visibility graphs is tested. GUI testing involves following actions:

1. Check all elements for size, position, width, length and acceptance of characters or numbers. For instance, you must be able to provide inputs to the input fields.

2. Overall functionality related with performance of users graphical interface are checked.

3. Check Error Messages are displayed correctly

4. Check the font, layout details, style and display warning messages if it is false.

5. check the positioning of GUI elements.

* 8.2.1 Unit Testing

It is the testing of individual software units of the application .it is done after the complexion of an individual unit before integration. Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. This is a structural testing, that relies on knowledge of its construction and is invasive.

Unit tests perform basic tests at component level and test a specific business process,

application, and/or system configuration. Unit tests ensure that each unique path of a

business process performs accurately to the documented specifications and contains

clearly defined inputs and expected results.

 8.2.2 Integration Testing

Integration tests are designed to test integrated software components to determine if they

actually run as one program. Testing is event driven and is more concerned with the

basic outcome of screens or fields. Integration tests demonstrate that although the

components were individually satisfaction, as shown by successfully unit testing, the

combination of components is correct and consistent. Integration testing is specifically

aimed at exposing the problems that arise from the combination of components.

 Testing Strategy

Software testing methods are traditionally divided into white- and black-box testing. These

two approaches are used to describe the point of view that a test engineer takes when

designing test cases.

1. White-box testing

In white-box testing an internal perspective of the system, as well as programming skills, are

used to design test cases.

2. Black-box testing

Black-box testing treats the software as a &quot;black box&quot;, examining functionality without any

knowledge of internal implementation. The testers are only aware of what the software is

supposed to do, not how it does it.

3. Grey-box testing

Grey-box testing involves having knowledge of internal data structures and algorithms for

purposes of designing tests, while executing those tests at the user, or black-box level. The

tester is not required to have full access to the software.