**TESTING**

Software testing is critical element of software quality assurance and represents the ultimate review of specifications, design and code generation. System testing is the stage of implementation, it is aimed for ensuring that the system works accurately and efficiently before live operations commences.

Testing is a purpose of executing a programmed with intend of finding errors.

1. Preparing a test case that has high probability of finding undiscovered errors.
2. Testing to erase out all kinds of bucks from the program.

Before going for testing, first we have to decide the type of test. For this impact system, unit testing is carried out. And the following things are taken to consideration.

1. To ensure that information properly places in and out of the program.
2. To ensure that the module operates properly at boundaries established to limit or restrict processing.
3. To find out whether all statements in module have been executed at least once.
4. To find out whether error handling paths are working correctly.

**7.1 TESTING STRATEGIES**

A strategy for software testing integrates software test case design methods in to a well-planned series of steps that results in the successful construction of the software. The strategy provides a road map that describes the step to be conducted as part of testing, when these steps are planned and undertaken, and how much effort, time and resources will be required. Therefore any testing strategy must incorporate test planning, test case, design, test execution and resultant data collection and evaluation. A software testing strategy should be flexible enough to promote customized testing approach. At the same time, it must be rigid enough to promote reasonable planning and management tracking as the project processes. The project manager, software engineer and testing specialists develop a strategy for software testing.

The general characteristics of software testing strategy are:

1. Testing begins at the component level and works “outward” toward the integration of the entire computer system.
2. Different testing techniques are appropriate at different point in time.

A strategy for software testing must accommodate low-level testis that are necessary to verify a small source code segment has been correctly implemented as well as high level testing that validate major system function against customer requirements.

**7.2 UNIT TESTING**

Unit test comprises of a set test performed by an individual programmer prior to the integration of the unit into large system. Program unit is usually small enough that the programmers who developed and can it in great detail and certainly in greater than will possible when the unit is integrated into evolving software project. Unit testing should be an exhaustive as possible. In this system, each module was tested individually to ensure that every representation in the module meets the requirements.

In unit testing I have tested each module according to its working there was some error in the tilt detection module after some reworking of connection, corrected its working properly. The next module is of moisture content it was also working accordingly. And the next module, vibration it is also working properly and gives all values in ThingSpeak.

**7.3 INTEGRATION TESTING**

Integration testing is a system technique for constructing the program structure while at the same time conducting test to uncover errors associated with interfacing. The objective is to take unit testing modules and build a program structure that has been dictated by design. Bottom-up integration is the traditional strategy used to integrate the components of a software system into functioning whole. In integration testing I combined code of each module and connection between each module and tested together for correction there was no error in the project working.

Bottom-up integration consists of a unit test followed by testing of the entire system. Subsystem consists of several modules that communicated with other defined interface. The errors were isolated and corrected to produce a fully functional system. Top-down integration method is an incremental approach to the construction of the program structure. The project was tested to ensure that every representation meets the requirements.

## **7.5TESTING RESULTS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.No** | **Test Case** | **Input** | **Expected Output** | **Pass or Fail** |
| 1 | Vibration sensor | Check for vibrations in the bridge and in the ground | Vibration range | Pass |
| 2 | Moisture sensor | Check for moisture content below the bridge | Moisture range | Pass |
| 3 | Tilt Sensor | Check for tilt range of the bridge | Tilt axis | Pass |

**USER ACCEPTANCE TESTING**

This testing is generally performed when the project is nearing its end. This test mainly qualifies the project and decides if it will be accepted by the users of the system. The users of the project are responsible for the test.

They test weather the project will work effectively and accurately.

**DATA VALIDATION TESTING**

Data validation is the process of testing the accuracy of data; a set of rules you can apply to a control to specify the type and range of data that can enter. It can be used to display error alert when users enter incorrect values into a form. In this project data validation testing carried out on all input from pages to test the accuracy.