**CHAPTER 7**

**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.

* Preparing a test case that has high probability of finding undiscovered errors.
* 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:

* Testing begins at the component level and works “outward” toward the integration of the entire computer system.
* 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.

These testing strategy checks the correctness of the newly engineered system. Errors found through the testing is rectified at the initial stages itself for better final product. Following are the series tests carried out: -

**Unit Testing**: In this each unit or the component is tested to see that the desired output is obtained.

**Integration Testing**: The objective here is to take the unit tested modules and see their performance where integrated with other modules of the system.

**System Testing**: In system testing, the software and the other system elements are tested as a whole. System testing is actually a series of different tests whose primary purpose is to fully exercise the whole system for its final desired result.

**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.

Prior to integration of the unit in to a large system, it is necessary to perform a set of tests to each unit of the system. Accordingly, functional efficiency of all sensors used, alert by Blynk application and correct storing of patient data etc. by the android app were tested repeatedly. Details of unit testing are appended in the table below for clarity.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SL.NO** | **Procedures** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| 1 | Testing of temperature sensor for accurate result. | Temperature of patient measured must be accurate. | Measuring accurately. | Pass. |
| 2 | Testing accuracy in measuring pulse rate by the sensor. | Patients pulse rate must be measured accurately every time. | Pulse rate measured accurately every time. | Successful. |
| 3 | Measuring of oxygen level of patient. | Measurement of correct oxygen level is a must. | Measured accurately. | Accurate result. |
| 4 | Alert by Blynk application (Notification). | System must send notification to alert patient/family. | System did send alert signal on sensing up normal result. | Pass |
| 5 | Storing of data from sensors measuring temperature, pulse rate & oxygen level. | Save the data of patient accurately in database. | Same as expected. | Pass |
| 6 | All the captured readings of the patient are sent to the respective patient android application (Blynk app). | Patients reading to be accurately stored in the android application. | Same as expected. | Pass |

Table 7.2: Unit test cases and results

**7.3 INTEGRATION TESTING**

It is the systematic technique for constructing the program structure to uncover errors associated with the interface. The objective is to take unit tested module and built the program structure that has been dictated by design. Body sensors are connected to the NodeMCU and their output is fed to the mobile app. The entire thing is tested as a whole to see the functioning of integrated system. Test results of the integrated system is appended below for clarity of information.

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.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SL.NO** | **Procedures** | **Expected Result** | **Actual Result** | **Pass or Fail** |
| 1 | Checking integrated output of body sensors | The values stored be correctly integrated. | Correct integration notified. | Pass |
| 2 | Accessibility of wrist band by sensors. | grant or denial of output. | Same as expected. | Pass |
| 3 | Correct passage &storing of body sensors NodeMCU output by mobile application. | Medical data to be correctly integrated by sensors. | Same as expected. | Pass |

Table 7.3: Integration test cases and results

**7.4 SYSTEM TESTING**

In system testing, the software and other system elements are tested as a whole. System testing is actually a series of different tests whose primary purpose is to fully exercise the IOT based BSN healthcare system. A table showing the details of various system tests and their results are appended below: -

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SL.NO** | **Procedures** | **Input** | **Expected Output** | **Pass or Fail** |
| 1 | Temperature determination of patient. | Check temperature input of patient. | Actual temperature of patient. | Is reading temperature correctly. |
| 2 | Determination of pulse rate and oxygen level. | Patient pulse rate and oxygen level. | Actual measure of patients pulses rate & oxygen level. | Reading correctly |
| 3 | Blynk operation accurately. | Patients medical parameters from sensors. | Correct interpretation of patient’s readings and detection of abnormalities and notifying the same. | Pass |
| 4 | Adding new patient. | New patient’s data. | Registered successfully. | Pass |
| 5 | View patient data. | Patients data | Shows the list of user patient. | Pass |
| 6 | View history of any patient. | Patient’s history data. | Shows the history of any user patient as desired. | Pass |

Table 7.4: System test cases and results

**7.5 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 or the customers of the project are responsible for the test.

**7.6 DATA VALIDATION TESTING**

Data validation is the process of testing the accuracy of data; a set of rule 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.

**7.7 OUTPUT TESTING**

After performing the validation testing, the next step is output testing of the proposed system since no system could be useful if it does not produce the required output generated or considered into two ways; one is on screen and another is printed format. The output format on the screen is found to be correct as the format was designed in the system design phase according to the user needs. For the hard copy also, the output comes out as the specified requirements by the user. Hence output testing does not result in any correction in the system.

**7.8 TESTING RESULT**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SL.NO** | **Procedures** | **Input** | **Expected Output** | **Pass or Fail** |
| 1 | Temperature determination of patient. | Temperature input of patient. | Actual temperature of patient. | Is reading temperature correctly. |
| 2 | Determination of pulse rate and oxygen level. | Patient pulse rate and oxygen level. | Actual measure of patients pulses rate & oxygen level. | Reading correctly |
| 3 | Blynk operation accurately. | Patients medical parameters from sensors. | Correct interpretation of patient’s readings and detection of abnormalities and notifying the same. | Pass |
| 4 | Adding new patient. | New patient’s data. | Registered successfully. | Pass |
| 5 | View patient data. | Patients data | Shows the list of user patient. | Pass |
| 6 | View history of any patient. | Patient’s history data. | Shows the history of any user patient as desired. | Pass |

Table 7.5: Testing and results

Use case diagram