Software Testing Analysis

Team 5 - KRB Energy

1. Acceptance Testing

Acceptance Testing -: A level of the software testing process 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.

It is the last phase of software testing performed after System Testing and before making the system available for actual use.

BLACK BOX TESTING is defined as a testing technique in which functionality 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 software requirements and specifications. In BlackBox Testing we just focus on inputs and output of the software system without bothering about internal knowledge of the software program.Mainly applicable to higher levels of testing:- System Testing and Acceptance Testing.

This method attempts to find errors in the following categories:

- Incorrect or missing functions
- Interface errors
- Errors in data structures or external database access
- Behavior or performance errors
- Initialization and termination errors

How the black box testing method is used

- Initially, the requirements and specifications of the system are examined.
- Tester chooses valid inputs (positive test scenario) to check whether SUT (System Under Test) processes them correctly. Also, some invalid inputs (negative test scenario) are chosen to verify that the SUT is able to detect them.
- Tester determines expected outputs for all those inputs.
- Software tester constructs test cases with the selected inputs.
- The test cases are executed.
- Software tester compares the actual outputs with the expected outputs.
- Defects if any are fixed and re-tested.

Example

We invited a tester, without knowledge of the internal structures of a website,and he tested the web pages by using a browser; providing inputs (clicks, keystrokes) and verifying the outputs against the expected outcome.

Several use-cases such as Controlling the AC , Viewing the AC temperature (both based on selection of a particular AC in KRB's building and floor-wise map) , Several User-Authentication features (login , logout , account information) were tested by the tester and expected outputs actually came.

Also the Admin feature was tested by the same tester . In the Admin page , he tested the Controlling of all ACs , and Sending Email Alerts (she checked this manually checking her mail-box).

All the implemented features worked correctly in this level.

2. System Testing

A level of the software testing process where a complete, integrated system is tested. The purpose of this test is to evaluate the system's compliance with the specified requirements.

System Testing is the third level of software testing performed after Integration testing and before Acceptance testing.

Example

In this level along with the Black-Box testing we also tested the use-cases by writing test scripts. One such example is that of Displaying of Hobolink Graphs : The script checks whether the data collected and displayed is the same as the original Hobolink Data.

Also following tests were done multiple times by various team members and the client herself :

UI (clicking buttons , proper rendering and the design of the application)

Performance of the Hobolink data displayed and the Controlling AC part

Security of various users (space user, admin) and end-to-end integration of all features.

3. Integration Testing

Integration testing is the process of testing the interface between two software units or modules. It's focused on determining the correctness of the interface. The purpose of the integration testing is to expose faults in the interaction between integrated units. Once all the modules have been unit tested, integration testing is performed.

1. **Bottom-Up Integration –** In bottom-up testing, each module at lower levels is tested with higher modules until all modules are tested. The primary purpose of this integration testing is, each subsystem is to test the interfaces among various modules making up the subsystem. This integration testing uses test drivers to drive and pass appropriate data to the lower level modules.

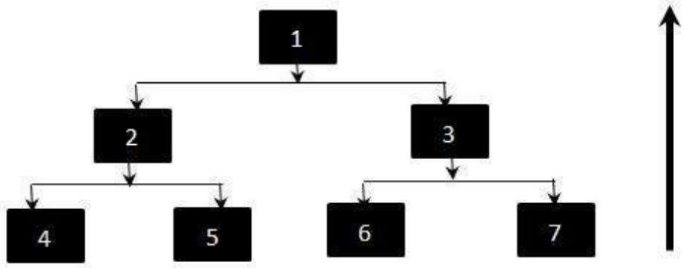
2. **Top-Down Integration –** Top-down integration testing technique used in order to simulate the behaviour of the lower-level modules that are not yet integrated.In this integration testing, testing takes place from top to bottom. First high-level modules are tested and then low-level modules and finally integrating the low-level modules to a high level to ensure the system is working as intended.

3. **Big-Bang Integration –** It is the simplest integration testing approach, where all the modules are combining and verifying the functionality after the completion of individual module testing. In simple words, all the modules of the system are simply put together and tested. This approach is practicable only for very small systems. If once an error is found during the integration testing, it is very difficult to localize the error as the error may potentially belong to any of the modules being integrated. So, debugging errors reported during big bang integration testing are very expensive to fix.

4. **Sandwich Integration –** This testing follows a combination of top down and bottom-up testing approaches. In top-down approach, testing can start only after the top-level module have been coded and unit tested. In a bottom-up approach, testing can start only after the bottom level modules are ready. This sandwich approach overcomes this shortcoming of the top-down and bottom-up approaches.

Example

Most beneficial and suitable approach was Bottom-Up approach



The order of Integration by Bottom-down approach will be:

```
4,2
5,2
6,3
7,3
2,1
3,1
```

In our application we proceeded as follows :

First we tested the control and monitor part together, then the Hobolink with the Dashboard together, and finally the above combined parts together with login/logout module.

This approach was beneficial for our application because we build our application that way incrementally and also our high level modules (the dashboard) invoked specific lower level modules , hence this approach suited our application.

4. Unit Testing

UNIT TESTING is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed.

Unit testing is the first level of testing done before integration testing.

WHITE BOX TESTING is a software testing method in which the internal structure/design/implementation of the item being tested is known to the tester. The tester chooses inputs to exercise paths through the code and determines the appropriate outputs.

How do we perform White Box Testing?

1. **UNDERSTAND THE SOURCE CODE**

   The first thing a tester will often do is learn and understand the source code of the application.The tester should be able to find security issues and fix them.

2. **CREATE TEST CASES AND EXECUTE**

   The second basic step involves testing the application's source code for proper flow and structure.The tester will develop little tests for each process or series of processes in the application.

Example

Here we (all the team members) studied the implementation code of a certain field on the webpage, determined all legal (valid and invalid) AND illegal inputs and verified the outputs against the expected outcomes, which is also determined by studying the implementation code.

Particular examples are :

Signup/Login/Logout Module - Here firstly there were two types of users Space Users and Admin , so depending on the user the authentication was done. The authentication was done by testing APIs on the Postman software.
Also the testing of valid credentials and input fields was performed.

Controlling the AC / Viewing the AC temperature - Here when we change the set point temperature of an AC , we can check both manually and by the test scripts we made. The test scripts confirmed that temperature/status is changed. We also manually verified this particular feature.