Chapter 6

TESTING

Software Testing is the process used to help identify the correctness, completeness, security and quality of the developed computer software. Testing is the process of technical investigation and includes the process of executing a program or application with the intent of finding errors. It is an investigation conducted to provide stakeholders with information about the quality of the software product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include the process of executing a program or application with the intent of finding software bugs (errors or other defects), and verifying that the software product is fit for use.

6.1 Introduction

Software testing involves the execution of a software component or system component to evaluate one or more properties of interest. In general, these properties indicate the extent to which the component or system under test:

- Meets the requirements that guided in its design and development.
- Responds correctly to all kinds of inputs.
- Performs its functions within an acceptable time.
- It is sufficiently usable.
- Can be installed and run in its intended environments.
- Achieves the general result its stakeholders desire.

As the number of possible tests for even simple software components is practically infinite, all software testing uses some strategy to select tests that are feasible for the available time and resources.

As a result, software testing typically (but not exclusively) attempts to execute a program or application with the intent of finding software bugs (errors or other defects). The job of testing is an iterative process as when one bug is fixed, it can illuminate other, deeper bugs, or can even create new ones.

Software testing can provide objective, independent information about the quality of software and risk of its failure to users or sponsors. Software testing can be conducted as soon as executable software (even if partially complete) exists. The overall approach to software development often determines when and how testing is conducted. For example, in a phased process, most testing occurs after system requirements have been defined and then implemented in testable programs. In contrast, under an agile approach, requirements, programming, and testing are often done concurrently.

Software Testing can be broadly classified into two types:

- Manual Testing - Manual testing includes testing a software manually, i.e., without using any automated tool or any script. In this type, the tester takes over the role of an end-user and tests the software to identify any unexpected behavior or bug. There are different stages for manual testing such as unit testing, integration testing, system testing, and user acceptance testing.

- Automation Testing - Automation testing, which is also known as Test Automation, is when the tester writes scripts and uses another software to test the product. This process involves automation of a manual process. Automation Testing is used to re-run the test scenarios that were performed manually, quickly, and repeatedly.

Software level testing can be majorly classified into 4 levels shown in figure 6.1:

- Unit Testing - A level of the software testing process where individual units/components of a software/system are tested. The purpose is to validate that each unit of the software performs as designed.

- Integration Testing - A level of the software testing process where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units.

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

- 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.

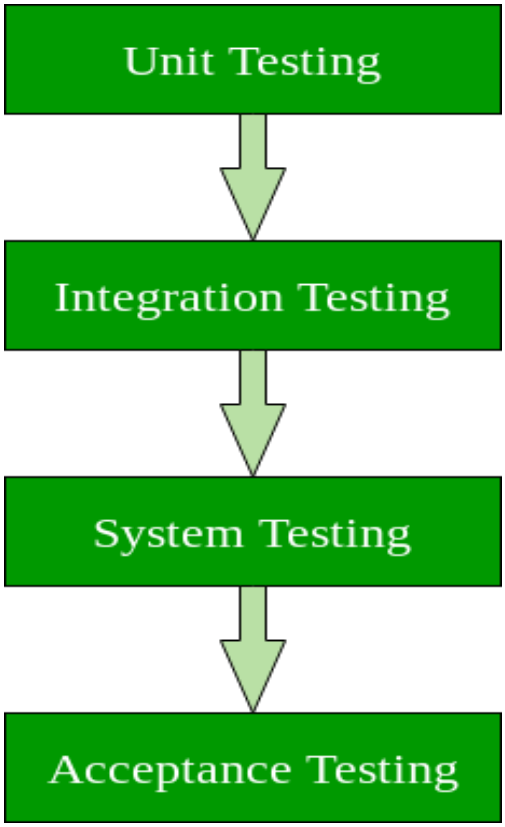

Figure 6.1 Levels of Testing

6.2 Unit Testing

It is a level of software testing where individual units or components of a software are tested. The purpose is to validate that each unit of software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output. In procedural programming, a unit may be an individual program, function, procedure, etc.

In object-oriented programming, the smallest unit is the method, which may belong to base or super class, abstract class or derived or child class. Unit testing frameworks, drivers, stubs, and mock/fake objects are used to assist in unit testing. It is performed by using white box testing method.

Unit testing is the first level of software testing and is performed prior to integration testing. It is normally performed by software developers themselves or their peers. In rare cases, it may be performed by independent software testers.

Table 6.1 Unit Testing

Type of test	Has it been performed	Explanations	Software components
Requirement testing	Yes	To check if all requirements are met	Check it manuallyto see if all the software are properly used
Unit	Yes	To check the modules for mistakes of developer	Requires checking manually
Integration	No	To check if all the modules are combines properly	If code runs in continuity it automatically checks
Performance	Yes	To check how the algorithm is performing	Various performance measures such as accuracy ,speed etc.
Stress	No	No need	NA
Compliance	No	No need	na
Security	No	No need	No issues in security

Load	No	No need	No load

6.3 Integration Testing

Integration testing is also taken as integration and testing this is the major testing process where the units are combined and tested. Its main objective is to verify whether the major parts of the program is working fine or not. This testing can be done by choosing the options in the program and by giving suitable inputs it is tested.

Table 6.2 Integration Testing

Test Case ID	List of various modules that required testing	Type of testing required	Techniques for writing test cases
1	Keras	Requirement, unit performance	White box
2	Wide Resnet	Requirement, unit performance	White box
3	Binary Classification	Requirement, unit performance	White box

6.4 System Testing

System testing is defined as testing of a complete and fully integrated software product. This testing falls in black –box testing where in knowledge of the inner design of the code is not a pre-requisite and is done by the testing team. System testing is done after integration testing is complete. System testing should test functional and non-functional requirements of the software. It is the third level of software testing performed after integration tesing and before acceptance testing. It is the process of testing an integrated systems to verify that it meets specified requirements. Usually, black box testing method is used.

Table 6.3 System Testing

Test case id	Input	Expected output	Status
1	Single Face from live web camera	Recognize gender and estimate age of single face	Pass
2	Two face from live web camera	Recognize gender and estimate age of two face	Pass
3	Three face from live web camera	Recognize gender and estimate age of three face	Pass

6.5 Validation Testing

The process of evaluating software during the development process or at the end of the development process to determine whether it satisfies specified business requirements.

Validation Testing ensures that the product actually meets the client's needs. It can also be defined as to demonstrate that the product fulfills its intended use when deployed on appropriate environment.

Table 6.4 Validation Testing

Test case id	Test case for components	Debugging technique	Status
1	Feature Extraction	Print debug	Pass
2	Gender Recognition	Print debug	Pass

3	Age Estimation	Print debug	Pass

6.6 User Acceptance Testing

User Acceptance Testing (UAT) is a type of testing performed by the end user or the client to verify/accept the software system before moving the software application to the production environment. UAT is done in the final phase of testing after functional, integration and system testing is done. The main purpose of UAT is to validate the end to end business flow. It does NOT focus on Cosmetic errors, Spelling mistakes or System testing. User Acceptance Testing is carried out in a separate testing environment with production-like data setup. It is a kind of black box testing where two or more end-users will be involved.

Table 6.5 User Acceptance Testing

Test case ID	Input	Output	Status
1.	Face of female from live web camer	Recognizes as Female(F)	Pass
2.	Face of male from live web camera	Recognizes as Male(M)	Pass