Session-5

Regression Testing

- Testing conducts on modified build to make sure there will not be impact on existing functionality because of changes like adding/deleting/modifying features.
- Unit regression testing
  - Testing only the changes/modifications done by the developer.
- Regional Regression Testing
  - Testing the modified module along with the impacted modules
  - Impact Analysis meeting conducts to identify impacted modules with QA & Dev.
- Full Regression
  - Testing the main feature & remaining part of the application.
  - Ex: Dev has done changes in many modules, instead of identifying impacted modules, we perform one round of full regression.

Re-Testing

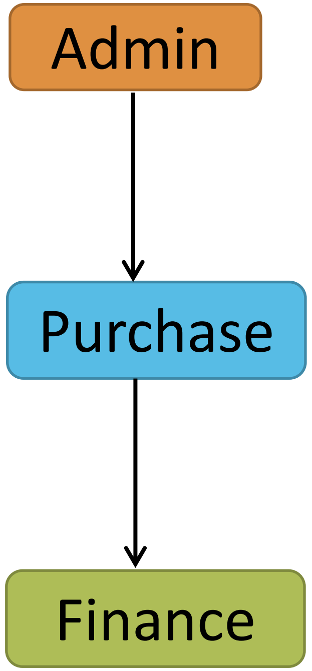
- Whenever the developer fixed a bug, tester will test the bug fix is called Re-testing.
- Tester close the bug if it worked otherwise re-open and send to developer.

- To ensure that the defects which were found and posted in the earlier build were fixed or not in the current build.

- Example
  - Build 1.0 was released. Test team found some defects (Defect Id 1.0.1, 1.0.2) and posted.
  - Build 1.1 was released, now testing the defects 1.0.1 and 1.0.2 in this build is retesting.

Example: Re-Testing Vs Regression Testing

- An Application Under Test has three modules namely **Admin**, **Purchase** and **Finance**.
- Finance module depends on Purchase module.
- If a tester found a bug on Purchase module and posted. Once the bug is fixed, the tester needs to do **Retesting** to verify whether the bug related to the Purchase is fixed or not and also tester needs to do **Regression Testing** to test the Finance module which depends on the Purchase module.

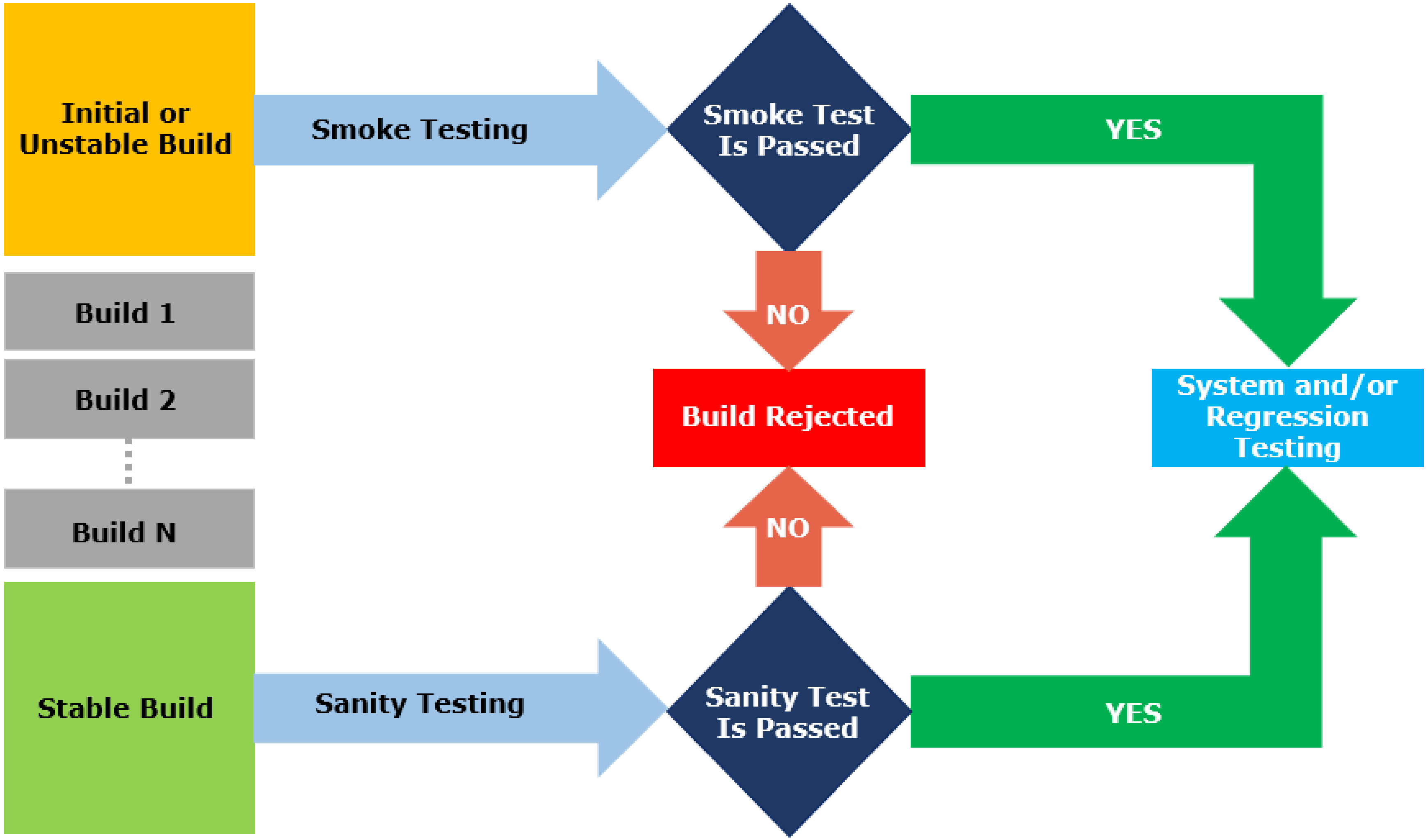


Smoke Vs Sanity Testing

- Smoke and Sanity Testing come into the picture after build release.

Smoke Testing	Sanity Testing
Smoke Test is done to make sure the build we received from the development team is testable/stable or not	Sanity Test is done during the release phase to check for the main functionalities of the application without going deeper.
Smoke Testing is performed by both Developers and Testers	Sanity Testing is performed by Testers alone
Smoke Testing, build may be either stable or unstable	Sanity Testing, build is relatively stable
It is done on initial builds.	It is done on stable builds.
It is a part of basic testing.	It is a part of regression testing.
Usually it is done every time there is a new build release.	It is planned when there is no enough time to do in-depth testing.

Smoke Testing Vs Sanity Testing



Exploratory Testing

- We have to explore the application ,understand completely and test it.
- Understand the application , identify all possible scenarios , document it then use it for testing.
- We do exploratory testing when the Application ready but there is no requirement.
- Test Engineer will do exploratory testing when there is no requirement.
- Drawbacks:
- You might misunderstand any feature as a bug (or) any bug as a feature since you do not have requirement.
- Time consuming
- If there is any bug in application , you will never know about it.

Adhoc Testing

- Testing application randomly without any test cases or any business requirement document.
- Adhoc testing is an informal testing type with an aim to break the system.
- Tester should have knowledge of application even thou he doesn't have requirements/test cases.
- This testing is usually an unplanned activity.

Ad Hoc Testing
No Documentation
No Test Design
No Test Case

Monkey/Gorilla Testing

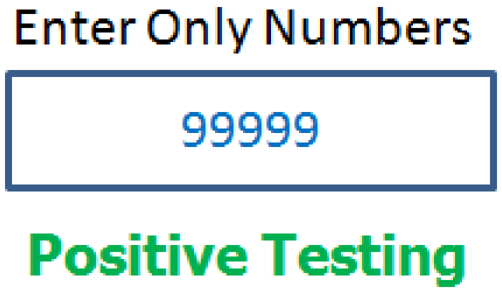
- Testing application randomly without any test cases or any business requirement document.
- Adhoc testing is an informal testing type with an aim to break the system.
- Tester do not have knowledge of application
- Suitable for gaming applications.

Adhoc Testing Vs Monkey Testing Vs Exploratory Testing

Adhoc Testing	Monkey Testing	Exploratory Testing
No Documentation	No Documentation	No Documentation
No Plan	No Plan	No Plan
Informal testing	Informal testing	Informal testing
Tester should know Application functionality	Testers doesn't know Application functionality	Testers doesn't know Application functionality
Random Testing	Random Testing	Random Testing
Intension is to break the application/find out corner defects	Intension is to break the application/find out corner defects	Intension is to learn or explore functionality of application
Any Applications	Gaming Applications	Any Applications which is new to tester

Positive Testing

- Testing the application with valid inputs is called as Positive Testing.
- It checks whether an application behaves as expected with positive inputs.
- For example -



Enter Only Numbers

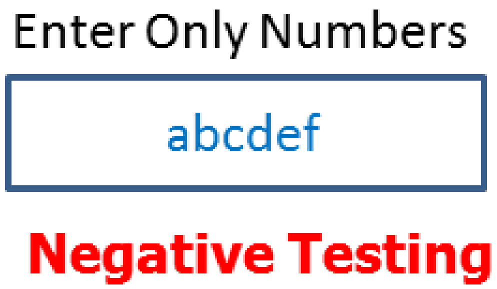
99999

Positive Testing

There is a text box in an application which can accept only numbers. Entering values up to **99999** will be acceptable by the system and any other values apart from this should not be acceptable. To do positive testing, set the valid input values from 0 to 99999 and check whether the system is accepting the values.

Negative testing

- Testing the application with invalid inputs is called as Negative Testing.
- It checks whether an application behaves as expected with the negative inputs.
- For example -



Enter Only Numbers

abcdef

Negative Testing

Negative testing can be performed by entering characters A to Z or from a to z. Either software system should not accept the values or else it should throw an error message for these invalid data inputs.

Positive V/s Negative Test Cases

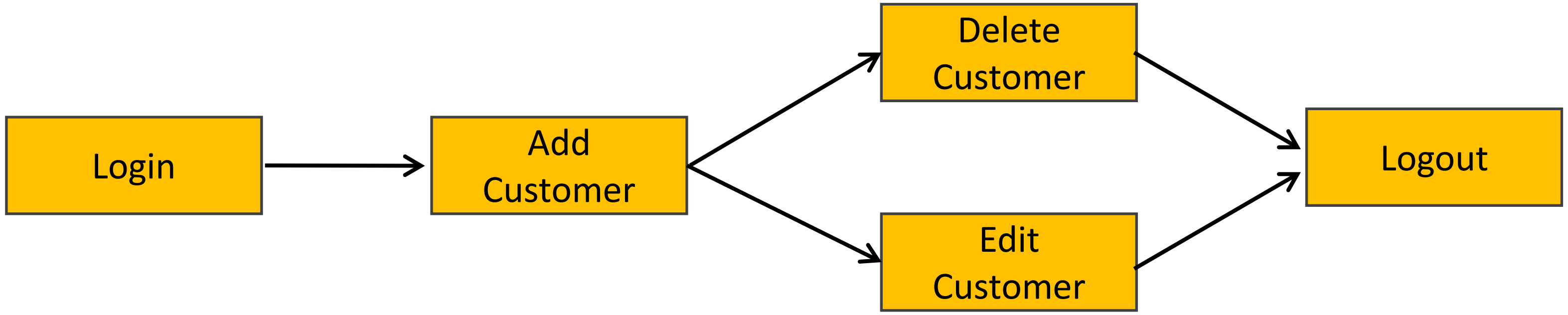
- **Requirement:**
  - For Example if a text box is listed as a feature and in FRS it is mentioned as Text box accepts 6 - 20 characters and only alphabets.

- **Positive Test Cases:**
  - Textbox accepts 6 characters.
  - Textbox accepts upto 20 chars length.
  - Textbox accepts any value in between 6-20 chars length.
  - Textbox accepts all alphabets.

- **Negative Test Cases:**
  - Textbox should not accept less than 6 chars.
  - Textbox should not accept chars more than 20 chars.
  - Textbox should not accept special characters.
  - Textbox should not accept numerical.

END-To-END Testing

- Testing the overall functionalities of the system including the data integration among all the modules is called end-to-end testing.



End-To-End Test

1) Login
2) ADD New Customer
3) Edit customer
4) Delete Customer
5) Logout

Globalization and Localization Testing

- **Globalization Testing:**
- Performed to ensure the system or software application can run in any cultural or local environment.
- Different aspects of the software application are tested to ensure that it supports every language and different attributes.
- It tests the different currency formats, mobile number formats and address formats are supported by the application.
- For example, Facebook.com supports many of the languages and it can be accessed by people of different countries. Hence it is a globalized product.

- **Localization Testing:**
- Performed to check system or software application for a specific geographical and cultural environment.
- Localized product only supports the specific kind of language and is usable only in specific region.
- It testes the specific currency format, mobile number format and address format is working properly or not.
- For example, Baidu.com supports only the Chinese language and can be accessed only by people of few countries. Hence it is a localized product.