DOCUMENT ON MANUAL TESTING DAY 3

SMOKE TESTING

**Smoke Testing** is a type of software testing performed after a build is created to ensure that the most critical functionalities of the application are working correctly. It acts as a preliminary check to determine if the software is stable enough to proceed with further testing.

Why its called as Smoke Testing?

Generally hardware engineers while testing the tool if they got smoke it is called as incorrect likewise in software also while testing the code it get errors(smoke) so this is the reason behind the name of smoke testing

In logging system the smoke system is involved ,Ensuring the login page loads correctly. Entering valid credentials to confirm successful login Verifying the logout feature works .If these fail, the build is sent back for fixes without further testing

SANITY TESTING

Verifies specific functionality or fixes, Narrow and deep testing. It ensures that recent changes or fixes in the codebase do not introduce new bugs.

It’s a retesting process of smoke testing

Regression Testing : (re running if the code changes)

With Regression testing we can rerun the code and fixes the bugs even after re writing the code, Every time a new module is added leads to changes in the Program. The regression testing look after the code to make bug free even after adding new components to the code.

Dynamic Testing : A testing which is done after code development.

The main purpose of dynamic testing is to test software behaviour with dynamic

variables

dynamic testing requires code to be executed

Alpha Testing : Its is final testing in development

Advantage is immediate solution is possible

Beta Testing : it is 1st testing in client side . it is also called user acceptance testing UAT

Disadvantage is no immediate solution, if defect is found

Installation Testing : providing required resources at client location

It is type of testing in which test engineer check deployment process is successful as

per user guideline

Deployment document is also called as user manual : it is document prepared by project manager

Usability Testing : checking application for user friendliness

Monkey Testing : used for game testing, used for random input To check the application or system will crash

Portability testing : Developed application Should support multiple environment

Exploratory Testing : When test engineer does not have idea of functional testing then he is learning through

exploring application

End to End Testing : We can check all internal component for successful response

Internal component like Client , Network, Server Database etc are working fine

Means Testing internal component

Security Testing : Checking Security of application

Reliability Testing : The Developed application Should Support Longer Duration

Here we discuss Different type of SDLC models

1.Waterfall Model

2. V-Shaped Model

3.Incremental Life Cycle Model

4.Spiral Model

In the SDLC Models each model has same phase

V shaped model also called as verification and validation model It is an enhancement of the traditional **Waterfall model** and is often referred to as the **Verification and Validation model** because it pairs development phases with their respective testing phases. In the V model testing we have two sides and one Right and one is Left

Right side model is verification Left side model is Validation

VERIFICATION PHASE(Left side)

Requirements Analysis:  
Gather and document system requirements. *Corresponding Test Phase*: User Acceptance Testing (UAT).

System Design:  
Define the system architecture and high-level design.  
*Corresponding Test Phase*: System Testing.

High-Level Design (HLD):  
Break down the system design into modules and define their functionality.  
*Corresponding Test Phase*: Integration Testing.

Low-Level Design (LLD):  
Detailed design of individual components, including logic and interfaces. *Corresponding Test Phase*: Unit Testing.

Coding/Implementation:  
Actual development of software code.

**VALIDATION** (Right side)

Unit Testing:  
Tests individual components/modules for functionality.

Integration Testing:  
Tests the interaction between integrated modules.

System Testing:  
Tests the entire system for compliance with system requirements.

User Acceptance Testing (UAT):  
Validates the system against business needs and user requirements.

SPIRAL MODEL

The Spiral Model works by developing software in cycles, called spirals. Each cycle includes planning, analysing risks, building and testing, then reviewing with the customer. After feedback, the process repeats, improving the product with each round. This approach handles risks early and allows changes during developmentits also overcome the waterfall Model

Incremental Model

The Incremental Model is a software development approach where the product is built and delivered in small, manageable sections called increments. Each increment adds functionality to the previous one, allowing partial versions of the system to be released and tested. In incremental testing we required models to test the process of the code

TESTCASE DESIGN TECHNIQUE

Test Case Design Techniques are methods used to create effective test cases that thoroughly evaluate the functionality, performance, and reliability of software. These techniques ensure systematic and efficient testing by covering as many scenarios as possible while minimizing redundancy.

1. ECP Equivalence Class Partition

2. BVA value analysis

3. Design Table

4. State transition

5. Error guessing

Equivalence Class Partitioning is a testing technique where input data is divided into groups (equivalence classes) that are expected to behave the same. You then test one value from each group instead of testing every possible input. This reduces the number of test cases while still covering all scenarios.

Example:

For a number input between 1 and 100:

Valid: Any number from 1 to 100 (e.g., 50).

Invalid: Numbers less than 1 (e.g., 0) and greater than 100 (e.g., 101).

Boundary Value Analysis (BVA) is a testing technique that focuses on testing the extreme values (boundaries) of input ranges, as errors often occur at these points.

Example:

For an age input between 18 and 60:

Test values like 17, 18, 60, and 61