Interview Questions from scratch:

1. **Software testing:**

   Software testing is the process of finding bugs & defects in software applications which make sure that software is Bug & Defect free & is able to fullfill customer requirements.

2. **Bug Life cycle:**
   1. New
   2. Assigned
   3. Opened
   4. Fixed
   5. Retest
   6. Verified
   7. Test closure

3. **SDLC:** Stands for software development Life cycle Which is used during software development.

   Phases of SDLC are:

   1. Requirements gathering.
   2. Planning
   3. Designing
   4. Coding
   5. Testing
   6. Deployment
   7. Maintenance

4. **STLC:** stands for Software Testing Life cycle, This is used by tester while testing,

   Phases of software testing life cycle are:

   1. Requirement Analysis
   2. Test planning
   3. Test case writing
   4. Test environment setup
   5. Test Execution
   6. Defect Tracking & reporting
   7. Test Closure.

5. **Functional Testing (Correct Definition):**

   Functional Testing is a type of software testing in which the **software system is tested against the functional requirements/specifications**.
   The main objective is to ensure that **each feature of the software application works as expected**.

📌 **Example (E-commerce Website):**

- Suppose there is a **Search functionality** in an e-commerce website.

- When the user searches for *"mobile"*, the system should return **all available mobile products**.
- If the search returns incorrect results or no results despite having products, then it means **functional testing has failed**.

6. **Non-functional testing:**

Non-Functional Testing (NFT) is the type of software testing that checks the **non-functional aspects** of a system — such as **performance, usability, reliability, scalability, security, and compatibility**.
It ensures that the software not only works functionally but also meets the expected **quality attributes**.

📌 **Examples:**

Performance Testing – Checking how fast an e-commerce site loads when **1,000 users search for "mobile" at the same time**.
Usability Testing – Ensuring the checkout process is **simple and user-friendly**.
Reliability Testing – Verifying the system runs **continuously for 24 hours** without crashing.
Security Testing – Making sure only authorized users can access their accounts.

Verification & Validation:

Verification:

It is the process of checking whether the software is being **built correctly** as per requirements, without executing the code.

Validation:

It is the process of checking whether the **developed software meets the business needs and requirements**, by executing the code.

Types of Software testing:

1. **Smoke Testing (Build Verification Testing):**

Definition: Smoke Testing is a type of software testing that verifies whether the basic and critical functionalities of a build are working. It ensures that the build is stable enough for further detailed testing.

📌 Key Point: It is a shallow and wide testing, covering critical features only, not detailed testing.

📌 Example:

- In an e-commerce website, after receiving a new build, testers check:
  - Can the user login?
  - Can the user search for products?
  - Can the user add products to the cart?
- If these basic features fail, the build is rejected for further testing.

2.Sanity Testing

Definition: Sanity Testing is a type of **brief, focused testing** to verify that a **specific functionality or bug fix is working correctly** after changes in the build.

📌 **Key Points:**

- It is **narrow and deep** (tests only the affected part of the application).
- Done when a **small change or bug fix** is made.

📌 **Example:**

- If a bug in the **search feature** of an e-commerce site is fixed, sanity testing will check **only the search functionality** to ensure it works correctly, not the entire application.

3.Regression Testing:

Regression testing is type of software testing where new features & functionality should not have any impact on the existing features & functionality.

Example: Suppose checkout features is added after Add to cart feature then checkout feature should not have any impact on the add to cart feature.

4. Retesting:

Retesting is the process of testing the features and functionalities again after developer's fixes to make sure that the bug has been resolved successfully and working as expected.

5.White Box testing:

Definition: White Box Testing is a type of software testing in which the tester **has knowledge of the internal code, logic, and structure** of the application and tests the software based on that knowledge.

📌 **Key Points:**

- Also called **Clear Box, Glass Box, or Structural Testing**.
- Focuses on **code paths, conditions, loops, and branches**.
- Testers usually **have programming knowledge**.

📌 **Example:**

- In an e-commerce website, a tester examines the **search algorithm** in the code to verify that it correctly filters products based on user input.

4. Black box testing:

Black Box Testing is a type of software testing in which the tester **does not need any knowledge of the internal code or logic** and tests the application based on **requirements and functionality**.

📌 **Key Points:**

- Also called **Behavioural or Functional Testing**.
- Focuses on **inputs and expected outputs**, not internal implementation.
- Testers **don't need programming knowledge** and can be outsiders.

📌 **Example:**

- In an e-commerce website, testers check if **searching for a product** shows the correct results without knowing how the search algorithm works.

Unit Testing:

Unit Testing is a type of software testing in which **each individual module or component of the software is tested independently** to ensure it works correctly.

📌 **Key Points:**

- Usually done by **developers** during the coding phase.
- Focuses on **smallest testable parts of the application**.
- Helps in **early detection of bugs**.

📌 **Example:**

- In an e-commerce website, testing the **"Add to Cart" function** separately to ensure it adds the correct product and quantity before integrating with the checkout system.

Integration testing:

Integration Testing is a type of software testing in which **two or more modules/components are tested together** to ensure they **work correctly and are compatible** when integrated.

📌 **Key Points:**

- Focuses on **interaction between modules**.
- Can be done using **Top-Down, Bottom-Up, or Big Bang approaches**.

📌 **Example:**

- In an e-commerce website, testing the **"Add to Cart" module with the "Checkout" module** to ensure products added to the cart are correctly processed during checkout.

System testing:

System Testing is the process of testing the complete application as a whole to verify that it meets the specified requirements and works as expected.

📌 **Key Points:**

- Done after integration testing.

- Focuses on end-to-end functionality of the software.
- Performed by testers, not developers.

📌 **Example:**

- In an e-commerce website, testing login, search, add to cart, payment, and order confirmation together to ensure the entire system works correctly.

User Acceptance Testing (UAT)

UAT is the process of testing to verify whether the software meets the user's or business requirements before it goes live.

📌 **Key Points:**

- Usually done by end users or clients, not developers.
- Focuses on real-world usage scenarios.
- Ensures the software is ready for production.

📌 **Example:**

- A client tests an e-commerce website to ensure search, add to cart, and checkout workflows function exactly as expected.

Alpha Testing

Alpha Testing is **internal testing performed by the development or internal QA team** before releasing the software to real users. It ensures that the software is **almost ready for release**.

📌 **Key Points:**

- Performed **in-house** by testers or developers.
- Focuses on **finding bugs before beta testing**.
- Simulates **real-world usage** but in a controlled environment.

📌 **Example:**

- Testing an e-commerce website internally to check **login, search, cart, and payment functionality** before sending it to selected real users for Beta Testing.

Beta Testing

Beta Testing is **external testing performed by real users or clients** after the software has passed alpha testing, to validate it in a **real-world environment** before full release.

📌 **Key Points:**

- Performed **by selected real users** outside the organization.
- Helps identify **bugs or usability issues** not found during alpha testing.
- Feedback is used to **improve the software** before the official launch.

📌 **Example:**

- An e-commerce website is released to a group of **real customers** to test **search, product browsing, checkout, and payment**, and their feedback is used to fix remaining issues.

End-to-End Testing (E2E)

End-to-End Testing is the process of testing the **complete workflow of an application** from start to finish to ensure all components and integrations work together correctly.

📌 **Key Points:**

- Verifies **complete system flow**, including external interfaces, databases, and network communication.
- Ensures the **entire application works as expected in real-world scenarios**.
- Usually performed by **testers**.

📌 **Example:**

- In an e-commerce website, testing a **user login → product search → add to cart → checkout → payment → order confirmation** workflow to ensure the entire process works seamlessly.

Exploratory Testing

Exploratory Testing is a type of testing where the tester **explores the application without predefined test cases**, learning the system while testing to find defects.

📌 **Key Points:**

- Focuses on **discovery and investigation** rather than following scripts.
- Testers **simultaneously learn, design, and execute tests**.
- Useful when **requirements are incomplete or unclear**.

📌 **Example:**

- In an e-commerce website, a tester navigates randomly through **search, filters, product details, and checkout** to discover unexpected bugs.

Ad Hoc Testing

Ad Hoc Testing is an **informal type of testing** where the tester **checks the application randomly without any planning or test cases** to find defects.

📌 **Key Points:**

- No formal documentation or process is followed.
- Testers rely on **experience and intuition**.
- Useful for **quick bug discovery**.

📌 **Example:**

- In an e-commerce website, a tester randomly clicks buttons, navigates pages, and performs unexpected actions to see if the site crashes or behaves incorrectly.

Installation Testing

Installation Testing is a type of testing that ensures the software **installs, uninstalls, and updates correctly** in the target environment.

📌 **Key Points:**

- Checks **installation steps, configurations, and dependencies**.
- Ensures the software **works properly after installation**.
- Also called **Implementation Testing**.

📌 **Example:**

- For an e-commerce desktop application, testing whether the **setup file installs the application properly**, creates required shortcuts, and allows successful uninstallation.

Recovery Testing

Recovery Testing is a type of testing that verifies whether the software **can recover gracefully from crashes, failures, or unexpected conditions**.

📌 **Key Points:**

- Checks the system's **reliability and fault tolerance**.
- Ensures that **data is not lost** and the system resumes normal operation.

📌 **Example:**

- In an e-commerce website, simulating a **server crash during checkout** and checking if the system **recovers without losing the user's cart or order details**.

Maintenance Testing

Maintenance Testing is performed on a **software application after it has been released**, to ensure that **updates, enhancements, or bug fixes do not break existing functionality**.

📌 **Key Points:**

- Focuses on **post-release changes**.
- Ensures the software remains **stable and reliable** after modifications.

📌 **Example:**

- After adding a **new payment gateway** in an e-commerce website, maintenance testing checks that **existing features like login, search, cart, and checkout** still work correctly.

1. Performance Testing

Performance Testing checks how the software performs in terms of **speed, responsiveness, and stability** under expected workloads.

📌 **Key Points:**

- Measures **response time, throughput, and resource usage**.
- Ensures software **meets performance requirements**.

📌 **Example:**

- Checking how fast an e-commerce website loads when **1000 users search products simultaneously**.

2. Load Testing

Load Testing verifies how the system behaves under **expected peak load conditions**.

📌 **Key Points:**

- Focuses on **system behavior under normal and peak load**.
- Helps identify **performance bottlenecks**.

📌 **Example:**

- Testing if an e-commerce website can handle **5000 simultaneous users placing orders**.

3. Stress Testing

Stress Testing checks how the system behaves under **extreme or beyond-peak load conditions**.

📌 **Key Points:**

- Determines **system limits and failure points**.
- Ensures **graceful recovery under stress**.

📌 **Example:**

- Testing a website with **10,000 users at once** to see if it crashes or slows down.

4. Security Testing

Security Testing ensures the application is **protected from unauthorized access, data breaches, and vulnerabilities**.

📌 **Key Points:**

- Verifies **authentication, authorization, data protection, and encryption**.
- Helps identify **security weaknesses**.

📌 **Example:**

- Checking if **user accounts and payment data** on an e-commerce website are secure from hacking.

---

5. Usability Testing

Usability Testing checks how **user-friendly and easy to navigate** the application is.
📌 **Key Points:**

- Focuses on **interface, navigation, and user experience**.
- Ensures software is **intuitive and simple to use**.

📌 **Example:**

- Verifying that a user can easily **search, add products to the cart, and checkout** without confusion.

6. Compatibility Testing

Compatibility Testing ensures the application works correctly across **different devices, browsers, operating systems, and screen sizes**.
📌 **Key Points:**

- Checks **cross-browser, cross-device, and cross-platform compatibility**.
- Detects issues related to **environment differences**.

📌 **Example:**

- Testing if an e-commerce website works properly on **Chrome, Firefox, mobile, and tablet**.

7. Reliability Testing

Reliability Testing verifies that the system can **run continuously without failure** under normal conditions.
📌 **Key Points:**

- Ensures **system stability and dependability**.
- Checks that software performs correctly for **long durations**.

📌 **Example:**

- Ensuring an e-commerce website functions reliably for **24 hours without crashing or losing data**.

Difference between Bug, Defect & error.

1. Error

An Error is a **mistake or flaw in the code or logic** made by the developer during development.
📌 **Key Points:**

- Usually identified by **developers**.
- Causes **incorrect behavior in the software** if not fixed.

📌 **Example:**

- Developer writes wrong logic: if(userAge > 18) instead of if(userAge >= 18).

2. Defect

A Defect is a **deviation from the expected requirement or specification** found during testing.

📌 **Key Points:**

- Usually identified by **testers**.
- Occurs when the software **doesn't meet the requirement**.

📌 **Example:**

- The login button doesn't work as per requirement.

3. Bug

A Bug is a **problem in the software that causes it to produce incorrect or unexpected results**

📌 **Key Points:**

- Can be found by **users, testers, or QA**.
- Impacts **software functionality or user experience**.

📌 **Example:**

- Clicking "Add to Cart" adds 2 items instead of 1.

Quick Way to Remember

- **Error** → Developer's mistake in code.
- **Defect** → Tester identifies mismatch with requirements.
- **Bug** → User experiences unexpected behavior.