**Dynamic Testing** is a type of software testing where you actually **run** the program or system to check if it behaves as expected. In simple terms, dynamic testing is about **testing the software while it's running** to see if everything works correctly.

**Analogies to Understand Dynamic Testing:**

1. **Driving a Car**: Imagine you’ve built a car, and now you need to test if it drives correctly. You get inside the car, turn on the engine, and start driving to see if the brakes work, the engine runs smoothly, and the steering is fine. This is like dynamic testing, where you “run” the software to see if everything works as expected.
2. **Testing a Recipe**: Suppose you’ve written a recipe. **Static testing** would be like reading the recipe to check for spelling mistakes or missing ingredients, while **dynamic testing** would be like actually cooking the dish to see if it turns out the way you expect.

**Key Features of Dynamic Testing:**

* **Execution of the Program**: Unlike static testing (where you don’t run the code), dynamic testing involves **running** the program to see if it behaves correctly under different conditions.
* **Finding Bugs While Running**: It helps find **runtime issues**—errors that only appear when the program is actually running, like crashes, incorrect outputs, or poor performance.
* **Types of Dynamic Testing**: It includes both functional and non-functional testing, such as:
  + **Functional Testing**: Checking if the software performs its expected tasks correctly (e.g., does the login page work as expected?).
  + **Non-Functional Testing**: Checking aspects like performance (e.g., how fast the software works under load), security (e.g., is the software safe from attacks), and usability (e.g., is it easy for users to navigate?).

**Steps in Dynamic Testing:**

1. **Prepare Test Cases**: Based on the requirements or design of the software, testers create **test cases** that describe the inputs and expected outputs.
2. **Run the Software**: The software is executed, and the testers provide the inputs described in the test cases. The goal is to check if the software produces the correct output or behaves as expected.
3. **Compare Actual vs Expected Results**: After running the software, you compare the **actual results** (what the software did) with the **expected results** (what you thought it should do).
   * If the software behaves as expected, the test is successful.
   * If the software does not behave as expected, it's a **bug**, and the testers need to report it.
4. **Log and Track Defects**: Any bugs or issues found during dynamic testing are logged in a **bug tracking system** so that developers can fix them.
5. **Repeat the Process**: The tests may be repeated for different scenarios, including edge cases, performance testing, and security testing.
6. **Types of Dynamic Testing:**
7. **Unit Testing**: Testing individual parts (or units) of the code, such as a function or method, to make sure each part works as expected.
8. **Integration Testing**: Testing how different parts of the software work together when combined ( does the login system work with the database?).
9. **System Testing**: Testing the entire software system as a whole to ensure it meets the requirements (does the entire web application work correctly from start to finish?).
10. **Acceptance Testing**: Testing to ensure that the software meets the end-user requirements and is ready for release.

**Benefits of Dynamic Testing:**

* **Real-World Testing**: Since the software is actually run during dynamic testing, you get a clear understanding of how it will perform in the real world.
* **Bug Detection**: It helps find issues that only appear when the software is in use, such as memory leaks, crashes, or logic errors.
* **Improves Software Quality**: Dynamic testing ensures that the software works as intended and that any defects are identified and fixed before release.

**Example of Dynamic Testing:**

Let’s say you are testing an **e-commerce website** where users can log in, browse products, and place orders.

1. **Test Case**: You create a test case that checks if a user can log in successfully using a valid username and password.
2. **Run the Test**: You input the correct login details and see if the website allows you to log in and navigate to the homepage.
3. **Expected Result**: You expect the website to log you in and show the homepage.
4. **Actual Result**: If the website logs you in correctly, the test is a success. If not, there’s a bug (e.g., login might not be working).
5. **Fixing the Bug**: If a bug is found, the development team fixes the issue, and the test is run again to verify the fix.

**In Summary:**

**Dynamic Testing** is testing the software by actually **running** it. It helps identify **runtime issues** (like crashes or incorrect behaviour) and ensures the software works as expected under real conditions. This type of testing is essential because it verifies that the program does what it’s supposed to do in a live environment, based on real inputs and actions.