Testing

1. Functional Testing

This type of testing verifies how well the software performs its **stated functions** according to the requirements.

- **Unit Testing:** Checks the smallest, isolated parts of the code (modules, functions) for correctness. Developers usually perform this.
- **Integration Testing:** Verifies the interactions between different modules or components of the system to ensure they work together correctly.
- **System Testing:** A comprehensive check of the entire integrated system against its specified requirements.
- **Acceptance Testing:** The final check of the system from the user's or client's perspective to ensure it meets their expectations and business needs. This can be **Alpha Testing** (internal testing by the client's team) or **Beta Testing**(external testing by real users).
- **Regression Testing:** Ensures that new code changes (bug fixes, new features) haven't broken existing functionality that previously worked correctly.

2. Non-Functional Testing

This type of testing evaluates the **quality of the system's performance**, rather than its functionality. It answers questions like "how well" or "how fast" something works.

- **Performance Testing:** Assesses the speed, responsiveness, stability, and scalability of the system under a particular load.
  - **Load Testing:** Checks system behavior under expected (normal) load.
  - **Stress Testing:** Verifies system behavior under extreme conditions or very high load (exceeding expected), to find the breaking point.
  - **Endurance/Stability Testing:** Checks the system's ability to sustain continuous load over a long period.
  - **Scalability Testing:** Determines the system's ability to increase its performance or user capacity by adding resources.
- **Reliability Testing:** Checks the system's ability to perform its functions without failures over a specified period.
- **Usability Testing:** Evaluates how easy and user-friendly the system is for end-users.
- **Security Testing:** Identifies vulnerabilities and weaknesses in the system that could lead to unauthorized access, data breaches, or other threats.
- **Compatibility Testing:** Verifies the software's operation across different environments (operating systems, browsers, devices, database versions).
- **Localization Testing:** Checks the correctness of software translation and adaptation for a specific region or language.
- **Maintainability Testing:** Assesses the ease of making changes, fixes, or additions to the system.

3. Change and Maintenance Related Testing

These types of testing focus on aspects related to changes within the system or its upkeep.

- **Smoke Testing:** A quick test of the software's basic functions after a build or deployment, to ensure there are no critical issues preventing further testing.
- **Sanity Testing:** A more focused test conducted after small changes or bug fixes to ensure the specific function works as expected and changes haven't introduced obvious problems.