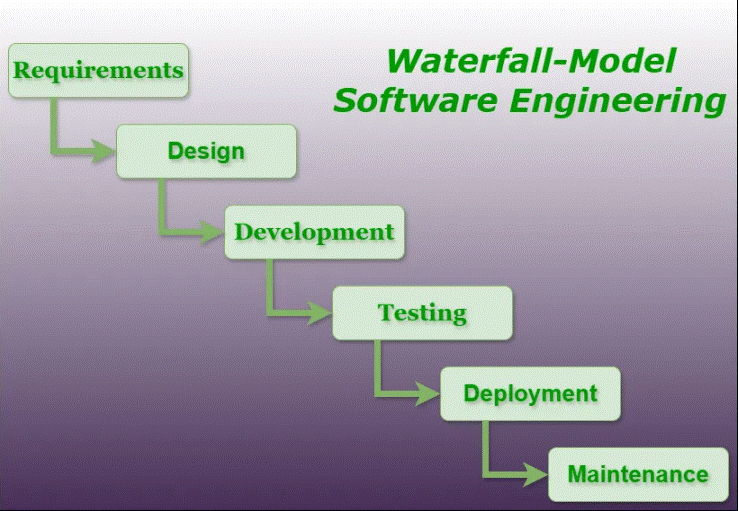
The waterfall model is a [software development model](https://www.geeksforgeeks.org/top-8-software-development-models-used-in-industry/) used in the context of large, complex projects, typically in the field of information technology. It is characterized by a structured, sequential approach to [project management](https://www.geeksforgeeks.org/software-engineering-software-project-management-plan-spmp/) and [software development](https://www.geeksforgeeks.org/software-development/).

The waterfall model is useful in situations where the project requirements are well-defined and the project goals are clear. It is often used for large-scale projects with long timelines, where there is little room for error and the project stakeholders need to have a high level of confidence in the outcome.

1. **Requirements:** The first phase involves gathering requirements from stakeholders and analyzing them to understand the scope and objectives of the project.
2. **Design:** Once the requirements are understood, the design phase begins. This involves creating a detailed design document that outlines the software architecture, user interface, and system components.
3. **Development:** The Development phase include implementation involves coding the software based on the design specifications. This phase also includes unit testing to ensure that each component of the software is working as expected.
4. **Testing:** In the testing phase, the software is tested as a whole to ensure that it meets the requirements and is free from defects.
5. **Deployment:** Once the software has been tested and approved, it is deployed to the production environment.
6. **Maintenance:** The final phase of the Waterfall Model is maintenance, which involves fixing any issues that arise after the software has been deployed and ensuring that it continues to meet the requirements over time.



The V-model is a type of [SDLC model](https://www.geeksforgeeks.org/top-8-software-development-models-used-in-industry/?ref=lbp) where the process executes sequentially in a V-shape. It is also known as the Verification and Validation model. It is based on the association of a testing phase for each corresponding development stage. The development of each step is directly associated with the testing phase. The next phase starts only after completion of the previous phase i.e., for each development activity, there is a testing activity corresponding to it.

**Verification Phases:**

It involves a static analysis technique (review) done without executing code. It is the process of evaluation of the product development phase to find whether specified requirements are met.

There are several Verification phases in the V-Model:

**Business Requirement Analysis:**

This is the first step of the designation of the development cycle where product requirement needs to be cured from the customer’s perspective. in these phases include proper communication with the customer to understand the requirements of the customers. these are the very important activities that need to be handled properly, as most of the time customers do not know exactly what they want, and they are not sure about it at that time then we use an **acceptance test design**planning which is done at the time of business requirement it will be used as an **input**for acceptance testing.

**System Design:**

Design of the system will start when the overall we are clear with the product requirements, and then need to design the system completely. This understanding will be at the beginning of complete under the product development process. these will be beneficial for the future execution of test cases.

**Architectural Design:**

In this stage, architectural specifications are comprehended and designed. Usually, several technical approaches are put out, and the ultimate choice is made after considering both the technical and financial viability. The system architecture is further divided into modules that each handle a distinct function. Another name for this is High-Level Design (HLD).

At this point, the exchange of data and communication between the internal modules and external systems are well understood and defined. During this phase, integration tests can be created and documented using the information provided.

**Module Design:**

This phase, known as Low-Level Design (LLD), specifies the comprehensive internal design for every system module. Compatibility between the design and other external systems as well as other modules in the system architecture is crucial. Unit tests are a crucial component of any development process since they assist in identifying and eradicating the majority of mistakes and flaws at an early stage. Based on the internal module designs, these unit tests may now be created.

**Coding Phase:**

The Coding step involves writing the code for the system modules that were created during the Design phase. The system and architectural requirements are used to determine which programming language is most appropriate.

The coding standards and principles are followed when performing the coding. Before the final build is checked into the repository, the code undergoes many code reviews and is optimized for optimal performance.

**Validation Phases:**

It involves dynamic analysis techniques (functional, and non-functional), and testing done by executing code. Validation is the process of evaluating the software after the completion of the development phase to determine whether the software meets the customer’s expectations and requirements.

So, V-Model contains Verification phases on one side of the Validation phases on the other side. The verification and Validation phases are joined by the coding phase in a V-shape. Thus, it is called V-Model.   
There are several **Validation** phases in the V-Model:

**Unit Testing:**

Unit Test Plans are developed during the module design phase. These Unit Test Plans are executed to eliminate bugs in code or unit level.

**Integration testing:**

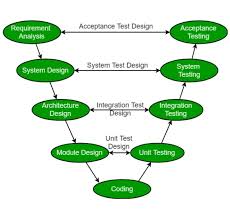
After completion of unit testing Integration testing is performed. In integration testing, the modules are integrated and the system is tested. Integration testing is performed in the Architecture design phase. This test verifies the communication of modules among themselves.

**System Testing:**

System testing tests the complete application with its functionality, inter-dependency, and communication. It tests the functional and non-functional requirements of the developed application.

**User Acceptance Testing (UAT):**

UAT is performed in a user environment that resembles the production environment. UAT verifies that the delivered system meets the user’s requirement and the system is ready for use in the real world.



**What is Agile?**

Agile is a [project management](https://www.geeksforgeeks.org/project-management-tutorial/) and software development approach that aims to be more effective.

1. It focuses on delivering smaller pieces of work regularly instead of one big launch.
2. This allows teams to adapt to changes quickly and provide customer value faster.

*Agile is not just a methodology; it’s a mindset. Agile isn’t about following specific rituals or techniques. Instead, it’s a bunch of methods that show a dedication to quick feedback and always getting better.*

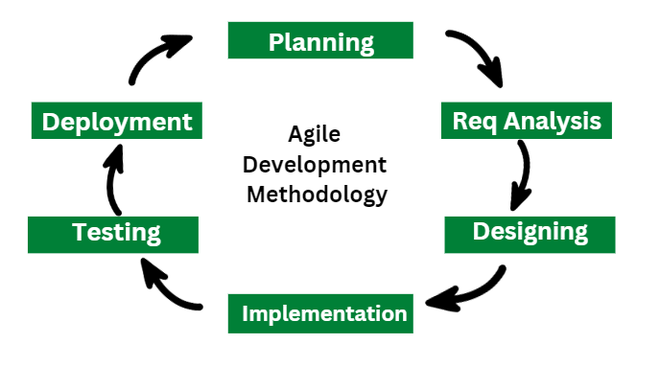
**What is the Agile Methodology?**

Agile methodologies are iterative and incremental, which means it’s known for breaking a project into smaller parts and adjusting to changing requirements.

1. They prioritize flexibility, collaboration, and customer satisfaction.
2. Major companies like Facebook, Google, and Amazon use Agile because of its adaptability and customer-focused approach.

**Life cycle of Agile Methodology**

The [Agile software development life cycle](https://www.geeksforgeeks.org/agile-sdlc-software-development-life-cycle/) helps you break down each project you take on into six simple stages:



**1. Requirement Gathering**

* In this stage, the project team identifies and documents the needs and expectations of various stakeholders, including clients, users, and subject matter experts.
* It involves defining the [project’s scope](https://www.geeksforgeeks.org/what-is-scope-in-project-management/), objectives, and requirements.
* Establishing a budget and schedule.
* Creating a project plan and allocating resources.

**2. Design**

* Developing a high-level system architecture.
* Creating detailed specifications, which include data structures, algorithms, and interfaces.
* Planning for the software’s user interface.

**3. Development (Coding)**

Writing the actual code for the software. Conducting unit testing to verify the functionality of individual components.

**4. Testing**

This phase involves several types of testing:

* **Integration Testing:** Ensuring that different components work together.
* **System Testing:** Testing the entire system as a whole.
* [**User Acceptance Testing:**](https://www.geeksforgeeks.org/user-acceptance-testing-uat/) Confirming that the software meets user requirements.
* [**Performance Testing**](https://www.geeksforgeeks.org/performance-testing-software-testing/)**:**Assessing the system’s speed, scalability, and stability.

**5. Deployment**

* Deploying the software to a production environment.
* Put the software into the real world where people can use it.
* Make sure it works smoothly in the real world.
* Providing training and support for end-users.

**6. Review (Maintenance)**

* Addressing and resolving any issues that may arise after deployment.
* Releasing updates and patches to enhance the software and address problems.

