Prepared by Group 1

Waterfall Methodology

Team Members

Daniel Karume
SCT211-0072/2022

Javan Odhiambo
SCT211-0098/2022

Ian Dancun
SCT211-0004/2022

Peaches Wagithi
SCT211-0005/2022

Sandrah Lewa
SCT211-0090/2022

Preface


PROJECT MANAGER: *describes the team's agile process*
SENIOR ENGINEER:
THIS IS THE TALLEST WATERFALL ON THE PLANET
imgflip.com

Introduction

The Waterfall model follows a step-by-step process where each phase is dependent on the completion of the previous one. This method works well when project requirements are clear from the start and unlikely to change. However, it is not ideal for projects that require flexibility or continuous adjustments.

Quick Summary

A linear and sequential software development approach.

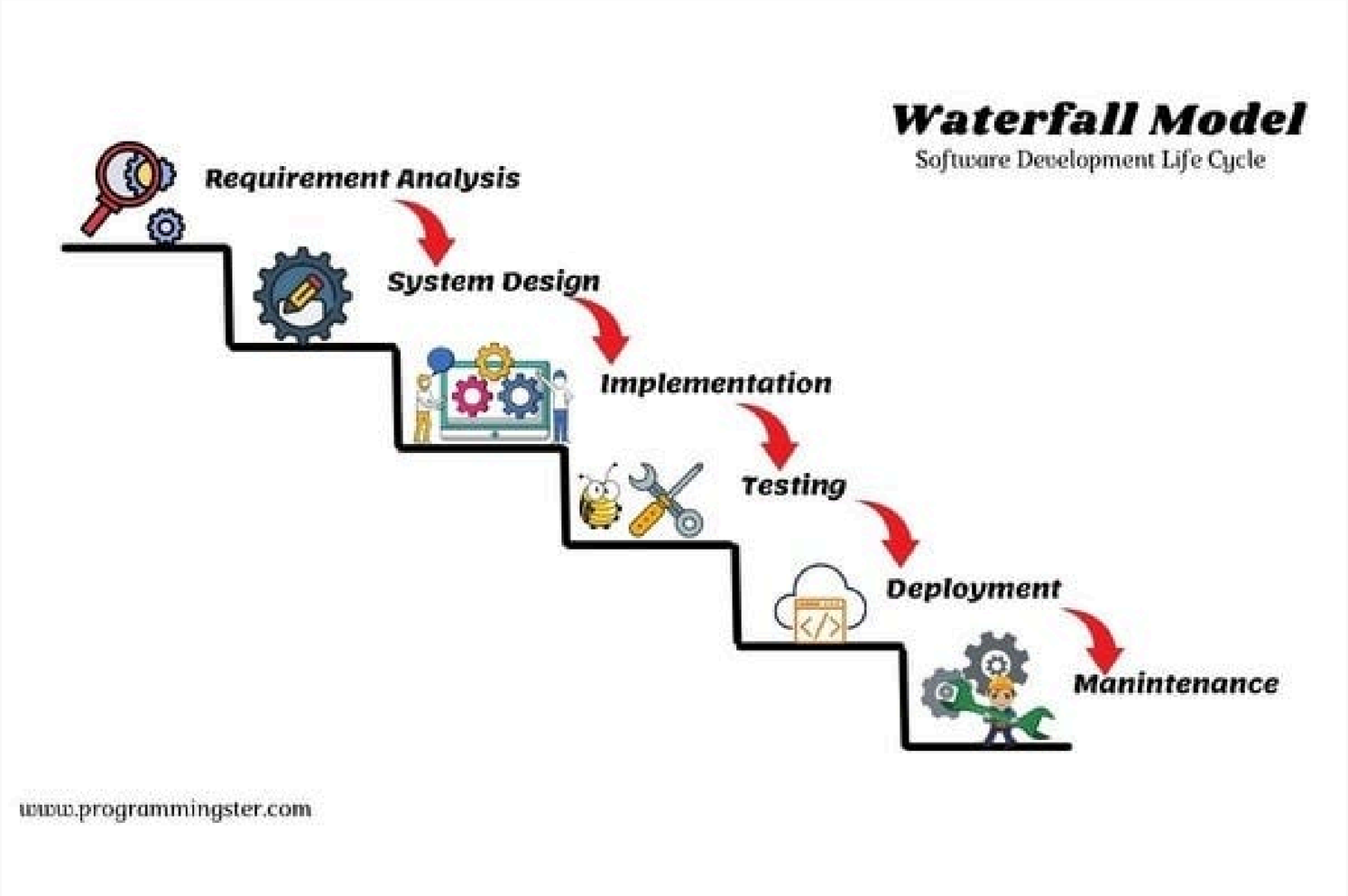
Each phase must be fully completed before moving to the next.

No overlapping or iterative steps—once a phase is done, you can’t go back.

Best suited for well-defined projects with fixed requirements and minimal changes.

Phases of the Waterfall Model

Each phase plays a critical role in the project's success. Since Waterfall is strictly sequential, moving backward is not allowed. Mistakes or changes late in the process can be costly, which is why detailed planning in the first two phases is crucial.



Phase 1: Requirement Analysis

- Gather and analyze client requirements to define project scope.
- Conduct meetings with stakeholders to document functional and non-functional needs.
- Ensure all requirements are clear, complete, and agreed upon before proceeding.
- Once finalized, changes are difficult to implement.

Outputs: Requirement Specification Document (RSD), feasibility reports, and project plan.

Phase 2: System Design

Translate requirements into a blueprint for the system.

System Architecture

We determine how different system components will interact. This includes decisions on client-server models, APIs, and third-party integrations.

Database Design

The database schema is designed to ensure efficient data storage, retrieval, and security. We define tables, relationships, and indexing strategies.

UI Design

Wireframes and prototypes are created to map out user flows and interface elements, ensuring a seamless user experience.

Outputs:

Design documents, wireframes, database schema.

Phase 3: Implementation

Development begins with writing actual code

- The project is divided into modules or components to allow parallel development and easier debugging.
- Developers follow best practices, including code documentation, version control (e.g., Git), and coding standards to maintain consistency.
- Depending on the project, front-end and back-end development may be handled separately, with APIs facilitating communication between them.

Outputs: Source Code, Compiled programs

Phase 4: Testing

The system undergoes rigorous testing to identify and fix bugs or errors before deployment

This ensures the software meets functional and performance expectations.

Unit Testing
Individual components

Intergration Testing
Interaction between components

System Testing
Entire system validation

Acceptance Testing
Meets user requirements

Outputs: Test Reports, Bug fixes

Phase 5: Deployment

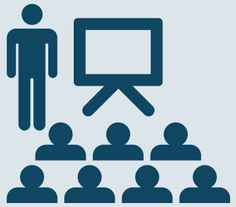
The fully developed and tested software is released to the production environment. This stage ensures that the software is accessible to end-users and operates as intended in a live setting.

Production Environment Setup

Configuring the infrastructure

Software Installation

Deploying the software



User Training

Train users on how to use the system

Go-Live & Monitoring

Monitoring for any issues.

Outputs: Fully deployed software, User training materials, Initial performance metrics

Phase 6: Maintenance

Necessary updates and modifications are made to ensure continued functionality, security, and efficiency. This phase is crucial for addressing real-world challenges and improving user experience.

Performance Enhancements

Make the system more efficient

Security Updates

Fix cyber security risks

Feature Upgrades

Update features to fit the environment it is used in

Bug Fixes

Fix bugs that might have slipped.

Outputs:

- Updated software versions, Performance reports, Security patches

Advantages

Well-suited for smaller projects:

- Works best when requirements are known upfront.

Good for regulatory compliance

- Used in industries like healthcare and finance.

Thorough documentation

- Beneficial for future maintenance and training.

Predictable Timelines & Costs

- Since all planning is done upfront, timelines and budgets are more predictable.

Industry Suitability

- Commonly used in industries like healthcare, finance, and aerospace, where detailed documentation and strict processes are required.

Disadvantages

Rigid and inflexible

- Difficult to accommodate changes after development starts.

Late-stage testing

- Errors discovered late can be costly.

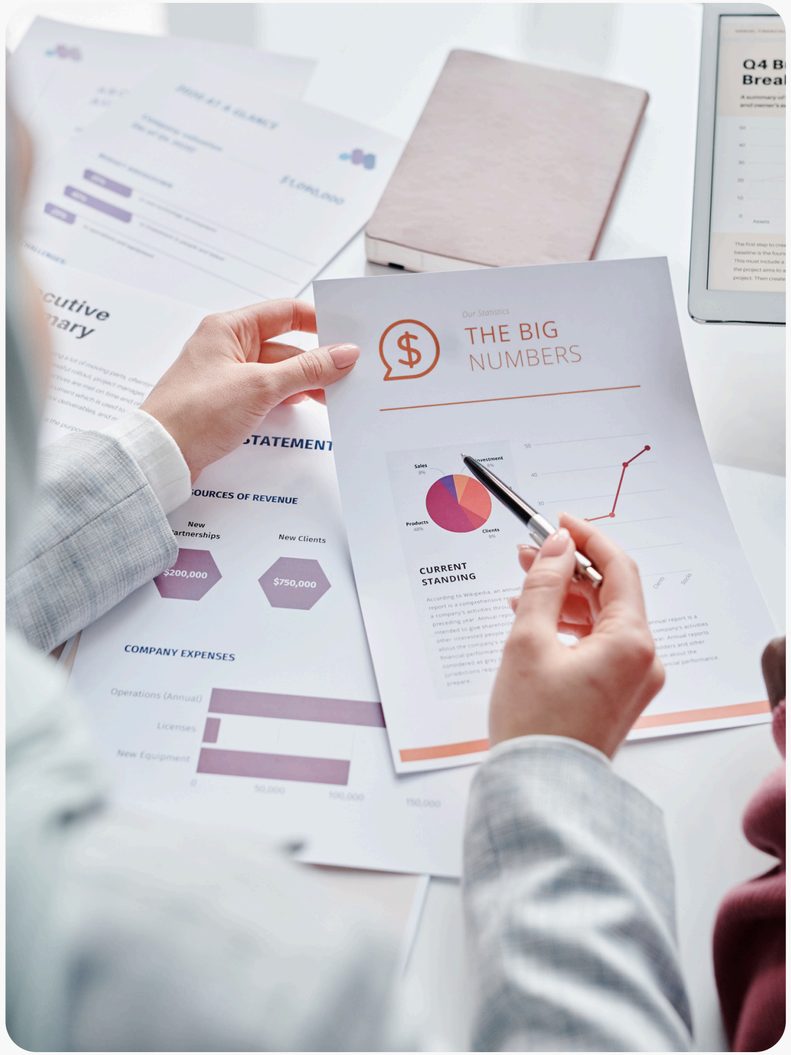
Slow development process

- Not ideal for rapidly changing environments

High risk for large projects

- Incorrect initial requirements affect the entire project

When to Use the Waterfall Model ?



Government and military projects

- Require strict documentation and sequential phases

Banking and financial software

- Security and compliance require structured development.

Manufacturing and construction

- Well-defined processes with little room for change.

Projects with fixed scope and budget

- Well-defined processes with little room for change.

Educational and research software

- Requires well-documented processes

Waterfall vs. Agile

FEATURES	WATERFALL	AGILE
APPROACH	SEQUENTIAL	ITERATIVE
FLEXIBILITY	LOW	HIGH
TESTING PHASE	AT THE END	CONTINUOUS
BEST FOR	FIXED REQUIREMENTS	EVOLVING PROJECTS

Real-Life Example of Waterfall Model

Developing a banking system.

- A banking system requires high security, regulatory compliance, and well-defined processes.

Key Characteristics in Action

- Each phase (Requirement → Design → Development → Testing → Deployment) is fully completed before moving to the next.
- No changes mid-development to avoid risks and ensure a structured process.

Why Waterfall?

- It ensures all requirements are finalized before development to prevent security loopholes.

Conclusion

- Waterfall is a structured, sequential model.
- Best for projects with clear, fixed requirements.
- Not ideal for flexible, evolving projects.
- Choosing the right methodology depends on project needs.

Thank you