Waterfall Model

By : Gehad Safwat

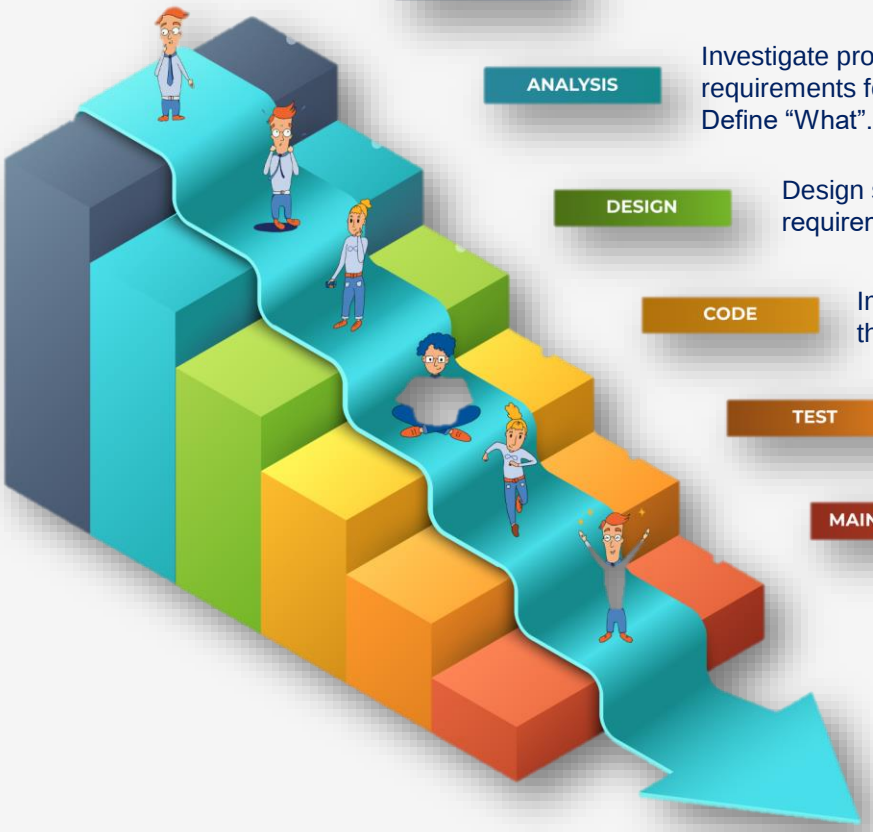
Introduction

- The Waterfall methodology **was once the dominant approach** that informed the project life cycle and processes that software development teams followed .

- A **linear development model** that unfolds in strict sequence, each phase of the Waterfall methodology's life cycle begins when the previous ends.

- The model's name is a nod to the fact that traditional Waterfall model **does not allow a software development team to return to previous phases** of the life cycle **once they are completed** – the project, like a waterfall, **flows in one direction only**.

02 Phases



REQUIREMENTS — Define the problem & identify its scope. Develop plan to solve the problem.

ANALYSIS — Investigate problem to define requirements for solving it . Define “What”.

DESIGN — Design solution based upon requirements . Define “How”.

CODE — Implement (program) the design.

TEST — Test program to ensure the requirements are satisfied.

MAINTENANCE — Fix or improve Apps

1. Requirements analysis

- All requirements of the final software product's utility and features have to be gathered here.
- Once the stage is completed the software development team should have all the information needed to complete the project without any, or minimal, further involvement from anyone .

2. Design

- We first have the logical design phase that involves putting all possible solutions on the table and analyzing their strengths and weaknesses within the context.
- Once theoretical ideas have been assessed and decisions taken on which to go with, the physical or detailed design phase is when they are documented and detailed as concrete specifications.

3. Implementation/Coding

- That's when software developers write and assemble the actual code that turns the specifications detailed in the design phase into a functioning software system.

4. Testing

- Software testers (who might be supported by automated testing tools) have to make sure every component of the software system works as intended, both autonomously and across any dependencies.
- Testers will use documentation created at the design phase, user personas and user journey scenarios to run as many test cases as possible in the attempt to uncover any bugs that need to be fixed before deployment.

5. Deployment

- When the software system has been tested and approved for release, a copy must be transferred from the software development environment and released in the live staging environment from which users will be able to access and use it. This stage is called deployment.

6. Maintenance

- In the maintenance phase, the software is in use and the primary job is now to keep it available and running smoothly as well as fixing any bugs reported by users that may have been missed during the testing phase.

Waterfall's strengths are a good fit for software development projects where:

- Requirements are clear, fixed and free of any ambiguity
- Product definition is stable
- Where the tools and technology used is consistent and is not changing
- Financial and expertise resources are made fully available
- The project's timeline is short

Waterfall's weaknesses mean it is not a good fit for software development projects because :

- There is more than a very low level of risk requirements and scope may change as assumptions become better informed
- This model cannot accept the changes in requirements during development.
- It becomes tough to go back to the phase. For example, if the application has now shifted to the coding phase, and there is a change in requirement, It becomes tough to go back and change it.
- Since the testing done at a later stage, it does not allow identifying the challenges and risks in the earlier phase, so the risk reduction strategy is difficult to prepare.

Thank You!