

SDLC & WATERFALL MODEL

Sumia Akter Ria



CONTENTS

- Introduction of SDLC
- Phase of SDLC
- Why SDLC
- Waterfall Model
- Pros & Cons of Waterfall Model

INTRODUCTION

SDLC (Software Development Life Cycle): **Structured process** for building high-quality software.

INVOLVES ACTIVITIES LIKE



PLANNING



Planning is key! This is where software development takes shape.



Senior team leads the **planning**, using **customer needs**, **sales insights**, **market research**, and **industry expertise**.



This info becomes the **foundation** for the project.



Better planning = Better software! The quality is built from the ground up.

REQUIREMENTS



This stage captures everything the **software needs to do**



SRS (Software Requirement Specification) is a key document used here.



SRS details what needs to be built and gets approval from key players.



Think of it as a **blueprint** outlining all the features and functionalities.

DESIGN

High-Level Design (HLD):

Overall system architecture, module functions, and interactions.

Low-Level Design (LLD):


Detailed breakdown of each module's functionality, data structures, and interfaces.

DEVELOPMENT

Coding begins! Developers write code based on the design documents.

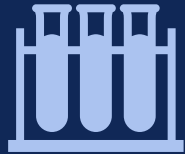


Devs follow **coding rules** (by managers) and **tools** to write clean, consistent code.



The aim is for development teams to **produce working software as quickly as possible**

TESTING



Testers verify if it works as planned (according to requirements).



Bugs are found and reported to developers.



Developers fix bugs and send it back for retesting.



This loop continues until the software is stable and bug-free.

DEPLOYMENT

After testing **finds no bugs**,
deployment
begins.

Project manager
reviews and approves final
software.

Deployment is
checked for **any final issues**.

MAINTENANCE

1

Fix bugs reported
by users.

2

Upgrade the
software to **newer**
versions.

3

End of the
beginning !

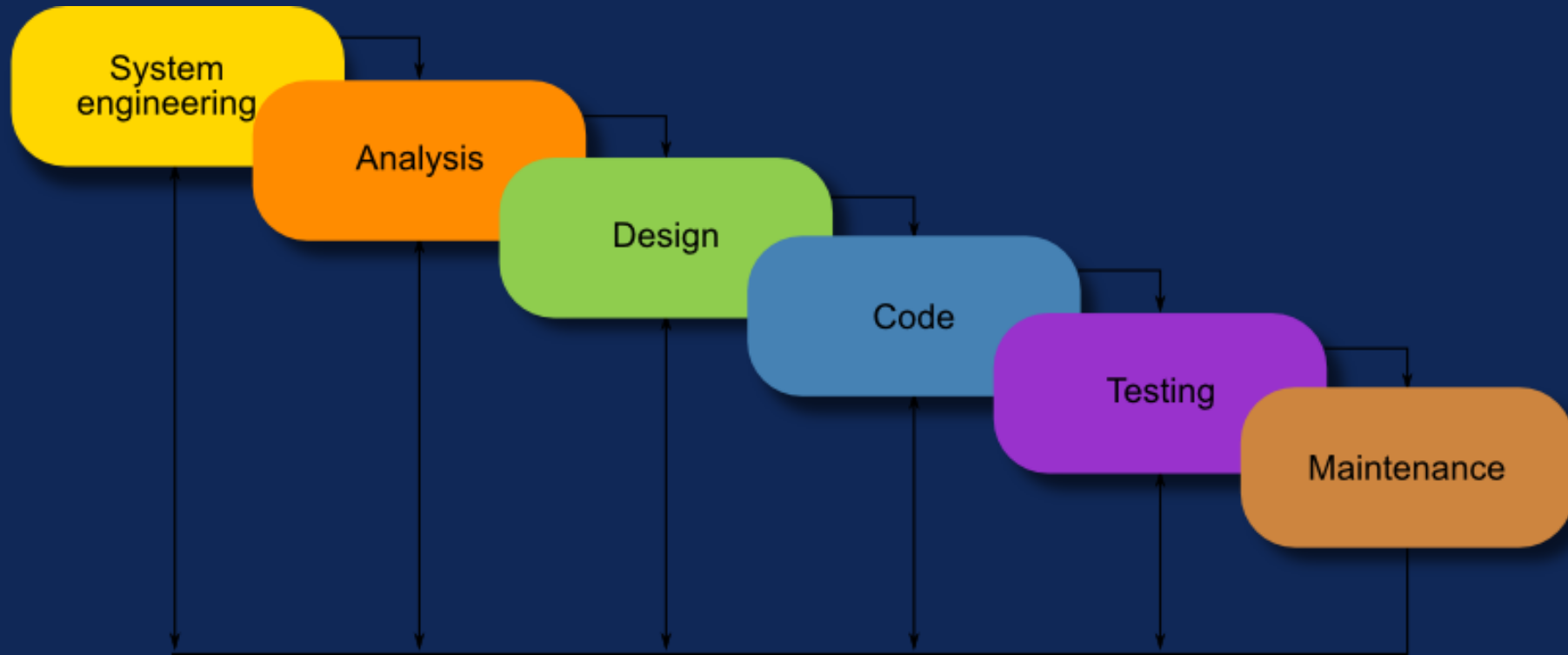


WHY SDLC?

- Provides a clear roadmap for all.
- Improves planning, reduces risk, and keeps costs in check.
- Delivers high-quality software that customers love.

WATERFALL MODEL

The Waterfall Model is a **straightforward** software development **approach** where you complete each step (planning, design, building, testing, etc.) entirely before moving on to the next. It's like a **one-way waterfall** - no going back!



WATERFALL MODEL

Requirement Gathering & Analysis: Define what the software needs to do. (Output: SRS Document)

System Design: Plan the software's architecture based on requirements. (Input: SRS Document)

Implementation & Coding: Write the code to bring the design to life. (Input: System Design Documents)

Testing: Thoroughly test the software to find and fix bugs. (Input: Developed Code)

Deployment: Release the software to users. (Input: Tested and Approved Software)

Maintenance: Fix any issues that arise after deployment.

PROS & CONS

Pros:

Waterfall model: A simple software development approach.

Step-by-step process: Each phase follows the one before, like a waterfall.

Clear deliverables: Each phase has defined outputs, making projects easier to manage.

Cons:

Waterfall model is slow: Each stage must finish completely before moving on, making it time-consuming.

Not ideal for short projects: This rigid structure might not fit the tight timelines of short sprints.

Unfriendly to changing needs: Waterfall assumes clear requirements upfront. If needs change later, fixing them across all stages becomes expensive.



THANK YOU

Sumia Akter Ria

sumiaria70@gmail.com