Software Development Life Cycle (SDLC)

The SDLC is a structured approach to software development that outlines the phases involved in creating software applications. It ensures that the development process is systematic, efficient & meets customer requirements. Various SDLC models such as Waterfall & Agile, provide different methodologies for implementing these phases.

Phases of SDLC

1) Requirement Analysis

- Objectives & requirements of project are identified & documented
- involves gathering input from stakeholders to understand their needs & expectations.

Importance:

- Clarity
- Foundation
- Avoids Miscommunication

2) System Design

- involves creating blueprint for the software architecture.
- includes both high level design & detailed design.

Importance:

- Guidance
- Efficiency
- Error reduction

3) Implementation (Coding)

- actual code is written based on the design documents.
- involves development of software components, functionalities & modules.

Importance:

- Execution
- Modularity
- Quality

4) Testing

- evaluating the software to identify & fix defects.
- uses various methods such as unit testing, integration testing, system testing etc.

Importance:

- Quality Assurance
- Bug identification
- Validation

5) Deployment

- involves releasing the software to the production environment where it becomes accessible to the end users.

Importance:

- Availability
- Configuration
- User Training

6) Maintenance

- involves making updates & improvements to the software after deployment.
- includes fixing bugs, adding new features & enhancing performance.

Importance:-

- Longevity
- Support
- Enhancements

-) SDLC Models

→ Waterfall Model

- linear & sequential approach to software development
- Each phase must be completed before the next phase begins, with no overlap.

Advantages:

- Simplicity
  - Structured
  - Documentation

Disadvantages :-

- Inflexibility
- Risk
- Delayed Testing

→ Spiral Model

- combines the iterative ~~approach~~ nature of Agile with the systematic aspects of waterfall Model,
- focuses on Risk assessment & reduction by iteratively refining requirements & solutions.

Advantages:

- Risk Management
- Flexibility
- Customer feedback

Disadvantages:

- Complexity
- Costly
- Time consuming

→ Agile Model

- iterative & incremental approach to software development
- focuses on flexibility, collaboration & customer feedback with work divided into small, manageable iterations called sprints.

Advantages:-

- Flexibility
- Customer Collaboration
- Early Detection

Disadvantages

- Scope Creep
- Documentation
- Dependency