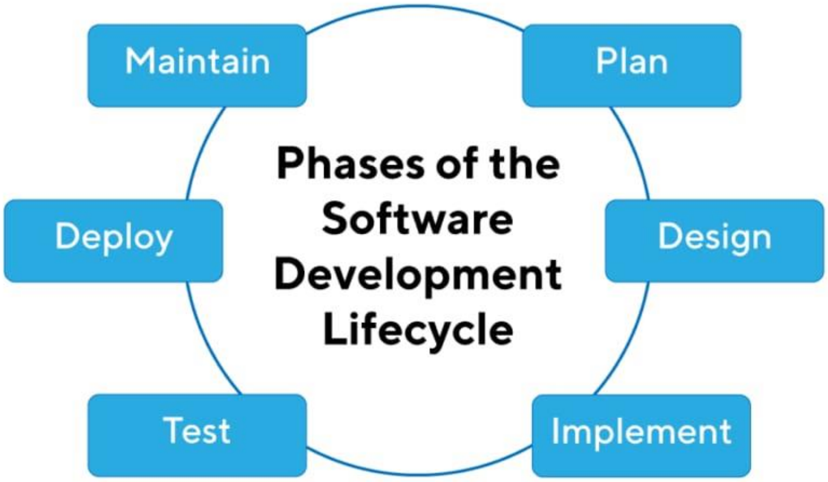
DAY 2:

Assignment 1:

Q) SDLC Overview - Create a one-page infographic that outlines the SDLC phases (Requirements, Design, Implementation, Testing, Deployment), highlighting the importance of each phase and how they interconnect.



1. Requirements Phase: This phase involves gathering and documenting the requirements for the software product. It's crucial because it sets the foundation for what needs to be built and ensures alignment between stakeholders and development teams.

2. Design Phase: In this phase, the software architecture and design are created based on the gathered requirements. It's important for ensuring that the system is scalable, maintainable, and meets the functional and non-functional requirements.

3. Implementation Phase: This phase involves actual coding and development of the software. Developers write code based on the design specifications. It's essential for turning the design into a functional product.

4. Testing Phase: Here, the software is rigorously tested to identify and fix any defects or bugs. Different types of testing, such as unit testing, integration testing, and system testing, are conducted to ensure the quality and reliability of the software.

5. Deployment Phase: This phase involves releasing the software to production and making it available to users. It's crucial for ensuring that the software is successfully installed and configured in the production environment.

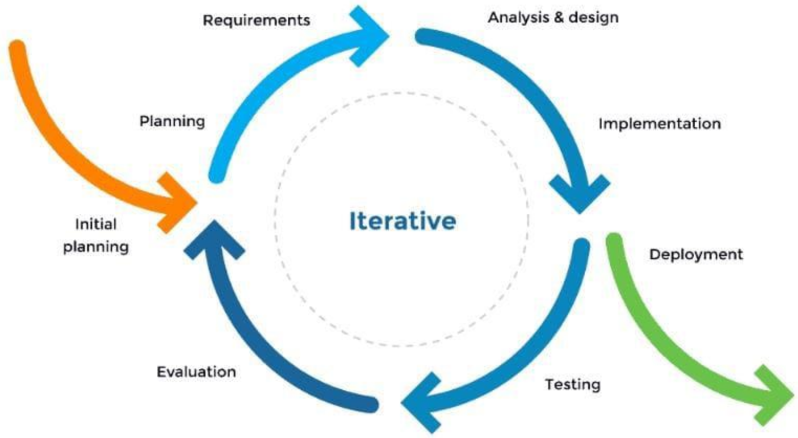
Assignment 2:-

Q) Develop a case study analyzing the implementation of SDLC phases in a real-world engineering project. Evaluate how Requirement Gathering, Design, Implementation, Testing, Deployment, and Maintenance contribute to project outcomes.

Case Study: Implementation of SDLC Phases in a Real-World Engineering Project

Project: Development of a new software system for a healthcare organization

SDLC Phases:



1. Requirement Gathering:

The project team met with stakeholders from the healthcare organization to understand their needs and requirements for the new software system. This included identifying the system's functionality, features, and performance requirements.

2. Design:

The project team developed a detailed design for the software system, including its architecture, components, and interfaces. The design was reviewed and approved by the stakeholders.

3. Implementation:

The project team developed the software system according to the approved design. This included writing code, creating unit tests, and integrating the different components of the system.

4. Testing:

The project team conducted extensive testing of the software system to ensure that it met all of the requirements. This included unit testing, integration testing, system testing, and user acceptance testing.

5. Deployment:

The project team deployed the software system to the healthcare organization's production environment. This included installing the software on the organization's servers and configuring it for use.

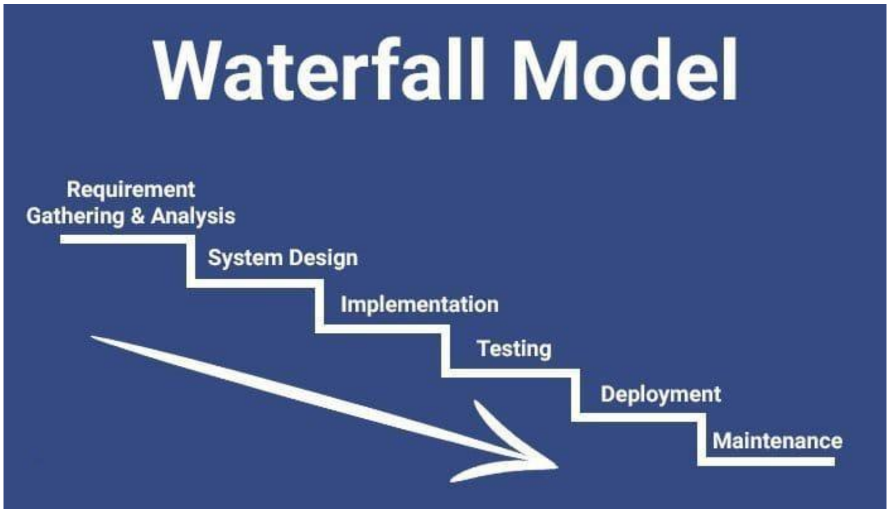
6. Maintenance:

The project team is providing ongoing maintenance and support for the software system. This includes fixing bugs, adding new features, and responding to user requests.

Assignment 3:-

Q) Research and compare SDLC models suitable for engineering projects. Present findings on Waterfall, Agile, Spiral, and V-Model approaches, emphasizing their advantages, disadvantages, and applicability in different engineering contexts.

A:- Waterfall Model:



Advantages:

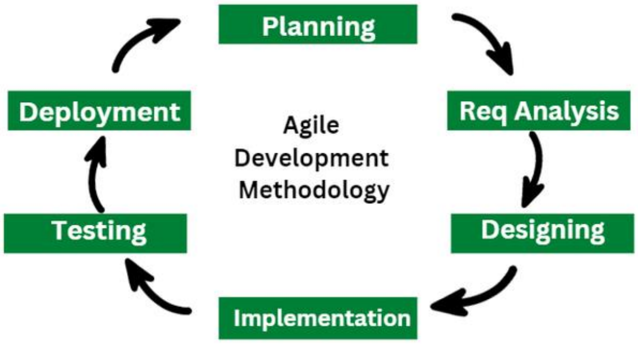
- Sequential and easy to understand.
- Well-suited for projects with stable requirements.
- Each phase has specific deliverables, making it easy to measure progress.

Disadvantages:

- Lack of flexibility, difficult to accommodate changes.
- Testing occurs late in the cycle, leading to potential issues with identifying defects early.
- Client feedback is limited until the end of the project.

Applicability:

- Ideal for projects with well-defined requirements and where the technology is stable



Agile Model:

Advantages:

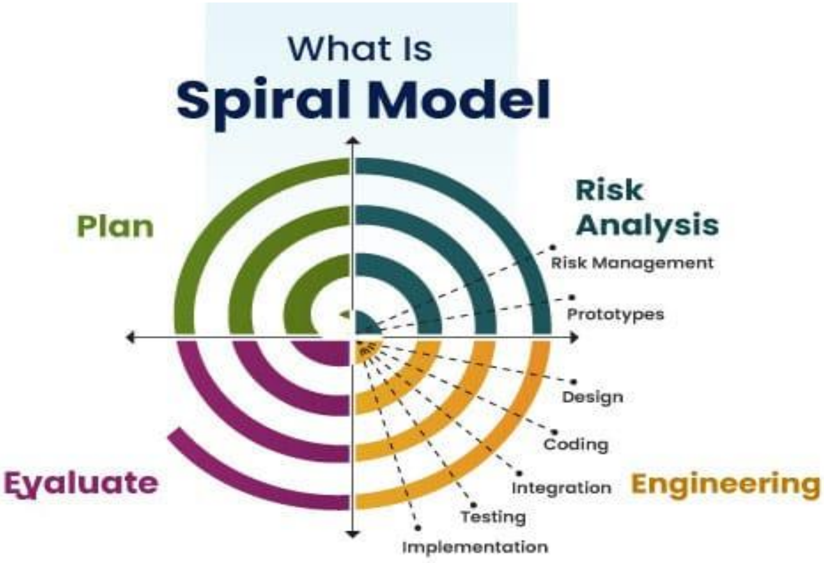
- Highly flexible and adaptive to changes.
- Promotes customer collaboration and feedback through iterative development.
- Allows for quick responses to market changes and customer needs.

Disadvantages:

- Requires active customer involvement throughout the project.
- May lead to scope creep if not managed properly.
- Challenges with scaling for large and distributed teams.

Applicability:

- Best suited for projects with evolving requirements and where customer involvement is crucial.



Spiral Model:

Advantages:

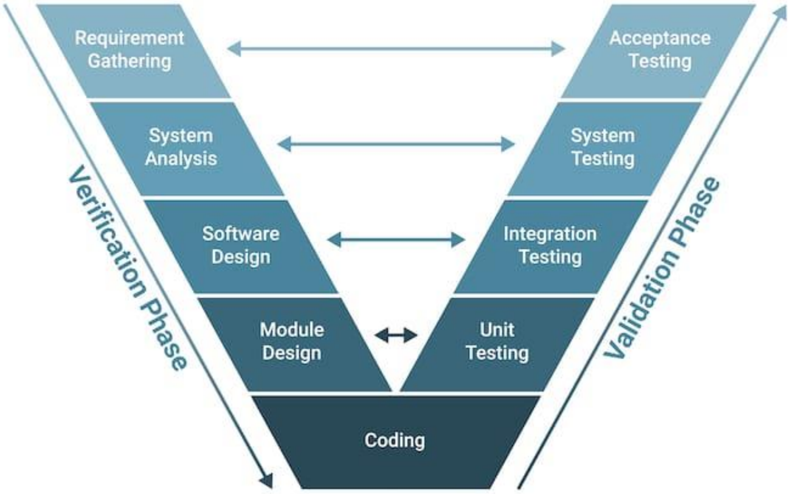
- Incorporates risk management throughout the development process.
- Allows for early identification and mitigation of risks through iterative development.
- Flexible and accommodates changes during the development cycle.

Disadvantages:

- Can be complex and costly to implement.
- Requires experienced personnel for risk assessment and management.
- May result in schedule slippage if risks are not managed effectively.

Applicability:

- Suitable for large-scale projects with high risks and uncertainties.



V-Model:

Advantages:

- Emphasizes the importance of testing throughout the development lifecycle.
- Ensures that each development phase has a corresponding testing phase.
- Well-suited for projects with stringent regulatory requirements.

Disadvantages:

- Can be rigid and difficult to accommodate changes once the process has started.
- Testing can be time-consuming and expensive.
- Limited flexibility compared to Agile or Spiral models.

Applicability:

- Suitable for projects with well-defined requirements and where thorough testing is critical.