SDLC ASSIGNMENT

Assignment 1: 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.

Infographic Title: SDLC Phases Overview

1. Requirements Phase

- **Description:** Gather and document all necessary requirements from stakeholders.
- **Importance:** Ensures all stakeholders have a clear understanding of what the project should achieve.
- **Key Activities:** Stakeholder interviews, requirement workshops, creating requirement documents.
- **Output:** Requirements Specification Document.

2. Design Phase

- **Description:** Develop the architecture and design of the system based on the requirements.
- **Importance:** Provides a blueprint for the system, ensuring all requirements are addressed.
- **Key Activities:** System architecture design, detailed design, creating design documents.
- **Output:** Design Specification Document.

3. Implementation Phase

- **Description:** Actual coding and development of the system based on design documents.
- **Importance:** Translates design into a functional system.
- **Key Activities:** Writing code, unit testing, code reviews.
- **Output:** Source code, build versions.

4. Testing Phase

- **Description:** Verify that the system works as intended and meets all requirements.
- **Importance:** Identifies defects and ensures the system is reliable and performs well.
- **Key Activities:** Writing test cases, executing tests, logging defects, regression testing.

- **Output:** Test Reports, Defect Logs.

5. Deployment Phase

- **Description:** Release the system to a live environment where it can be used by end-users.
- **Importance:** Makes the system available for use, transitioning from development to operation.
- **Key Activities:** Deployment planning, system configuration, user training, deployment.
- **Output:** Deployed System, Deployment Documentation.

Interconnections:

- **Requirements ↔ Design:** Clear requirements guide the design process.
- **Design ↔ Implementation:** Design documents are used to develop the actual system.
- **Implementation ↔ Testing:** Developed code is tested to ensure it meets requirements.
- **Testing ↔ Deployment:** Only thoroughly tested code is deployed to the live environment.
- **Deployment ↔ Maintenance:** Post-deployment issues are handled during the maintenance phase.

Assignment 2: 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 a New Public Transportation System

1. Requirement Gathering

- **Activities:** Extensive stakeholder meetings including city officials, engineers, and public representatives to understand needs.
- **Outcome:** Comprehensive requirement document detailing routes, schedules, and system features.

2. Design Phase

- **Activities:** Architectural design of routes, bus stations, and software system for scheduling and real-time tracking.
- **Outcome:** Detailed blueprints and design documents for both physical infrastructure and software.

3. Implementation Phase

- **Activities:** Construction of bus stations, purchase and setup of buses, and development of the scheduling and tracking software.
- **Outcome:** Physical infrastructure in place, and software developed and installed.

4. Testing Phase

- **Activities:** Pilot testing with a limited number of buses, software testing including stress tests, and feedback collection from users.
- **Outcome:** Identification and resolution of defects, ensuring reliability and performance.

5. Deployment Phase

- **Activities:** Full-scale launch of the system, public awareness campaigns, and training for drivers and support staff.
- **Outcome:** System is live and operational, used by the public as intended.

6. Maintenance Phase

- **Activities:** Continuous monitoring, regular updates, and addressing any emerging issues.
- **Outcome:** Sustained performance and user satisfaction, with ongoing improvements.

Assignment 3: 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.

Comparison of SDLC Models

1. Waterfall Model

- **Advantages:**
    - Simple and easy to understand.

    - Well-defined stages with clear objectives.
    - Easy to manage due to rigidity.
- **Disadvantages:**
    - Difficult to accommodate changes once the project starts.
    - Late discovery of issues due to late testing phase.
- **Applicability:** Best for projects with well-defined requirements and minimal changes expected.

2. Agile Model

- **Advantages:**
    - Flexibility to changes and iterative improvements.
    - Continuous delivery of small, functional segments.
    - Active stakeholder involvement and feedback.
- **Disadvantages:**
    - Can be challenging to predict time and budget.
    - Requires strong collaboration and communication.
- **Applicability:** Suitable for projects with evolving requirements and where quick delivery of components is beneficial.

3. Spiral Model

- **Advantages:**
    - Emphasizes risk management.
    - Iterative approach with continuous refinement.
    - Can handle changes better than Waterfall.
- **Disadvantages:**
    - Complex to manage.
    - Requires expertise in risk assessment.
- **Applicability:** Ideal for large, complex projects with high risk and where requirements may evolve.

4. V-Model (Verification and Validation Model)

- **Advantages:**
    - Each development stage has a corresponding testing phase.

    - Emphasizes validation and verification.
    - Defects are detected early.
- **Disadvantages:**
    - Rigid and less flexible to changes.
    - High risk of missing details if initial requirements are not thorough.
- **Applicability:** Best for projects where quality is critical, and requirements are well-understood upfront.

Summary

Waterfall is best for projects with clear requirements and minimal expected changes. **Agile** is suited for projects requiring flexibility and iterative progress. **Spiral** is beneficial for complex projects with high risk and evolving requirements. **V-Model** is ideal for projects where validation and verification are crucial throughout the development process. Each model has its unique strengths and weaknesses, and the choice depends on project specifics and requirements.