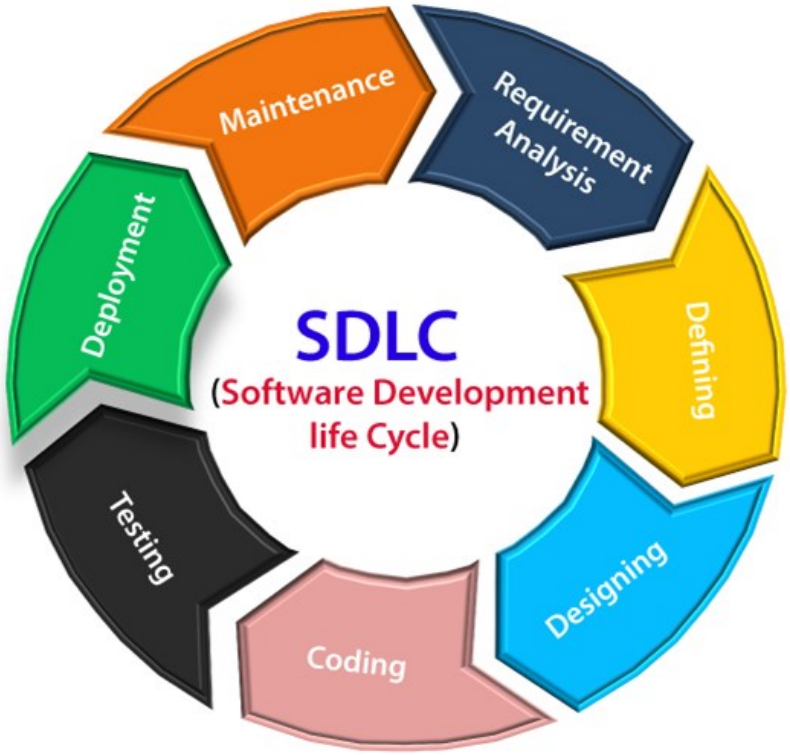
Assignment 1: SDLC Overview - Create a one-page infographic that outlines the SDLC phases (Requirements, Design, Implementation, Testing, Deployment), highlighting the importance of each phase and how they interconnect.



Software development life cycle (SDLC) is a process/step-by-step procedure used by the software industry to design, develop, and test high-quality software. It aims to produce high-quality software that meets customer expectations.

Requirement Analysis: The development team collects requirements from several stakeholders such as customers, internal and external experts, and managers to create a software requirement specification document.

- Importance - This information helps to plan the basic project approach. The requirement includes how the product will be used and who will use the product to determine the load of operations.
- Interconnections - Set the stage for all subsequent phases.

Defining: Defining the architecture, components, interfaces, and data for a system to satisfy specified requirements. In this stage, all the requirements for the target software are specified. These requirements get approval from customers, market analysts, and stakeholders.

The document sets expectations and defines common goals that aid in project planning. The team estimates costs, creates a schedule, and has a detailed plan to achieve their goals.

- Importance - Provides a clear and comprehensive understanding of what needs to be achieved, ensuring alignment among all stakeholders and it is feasible within budget and time constraints.

- Interconnections - Lays the groundwork for design and development by setting clear objectives and requirements.

Designing: The design phase includes a detailed analysis of new software according to the requirement phase. This is the high-priority phase in the development life cycle.

High-Level Design and Low-Level Design are two essential milestones in these phases. While low-level language handles on a microscopic level, high-level design engages on a massive level.

- Importance - A well-thought-out design serves as a blueprint for the development team, leading to fewer issues during implementation.
- Interconnections - Translates requirements into a structured solution that can be coded in the next phase.

Coding: The actual coding and conversion of the design documentation into the actual software within the development environment.

- Importance - This is where the software is built and where most of the developer's work is concentrated.
- Interconnections - Uses the design documents to create functional software.

Testing: Evaluating the software to ensure it meets all specified requirements and is free of defects.

- Importance - Detects and fixes bugs, ensuring the software is reliable and performs as expected.
- Interconnections - Validates the implementation against the requirements, feeding back into the implementation phase if issues are found.

Deployment: Releasing the finished product to users and deploying it into a production environment.

- Importance - This phase puts the product into use, where it can deliver value to the users.
- Interconnections - This phase that makes use of the tested software, completing the initial development cycle.

Maintenance: Ongoing support and refinement of the software post-deployment.

- Importance - Ensures the software continues to function correctly and evolves to meet changing user needs.

- Interconnections - Feeds back into planning for future updates and improvements, creating a cycle of continuous improvement.