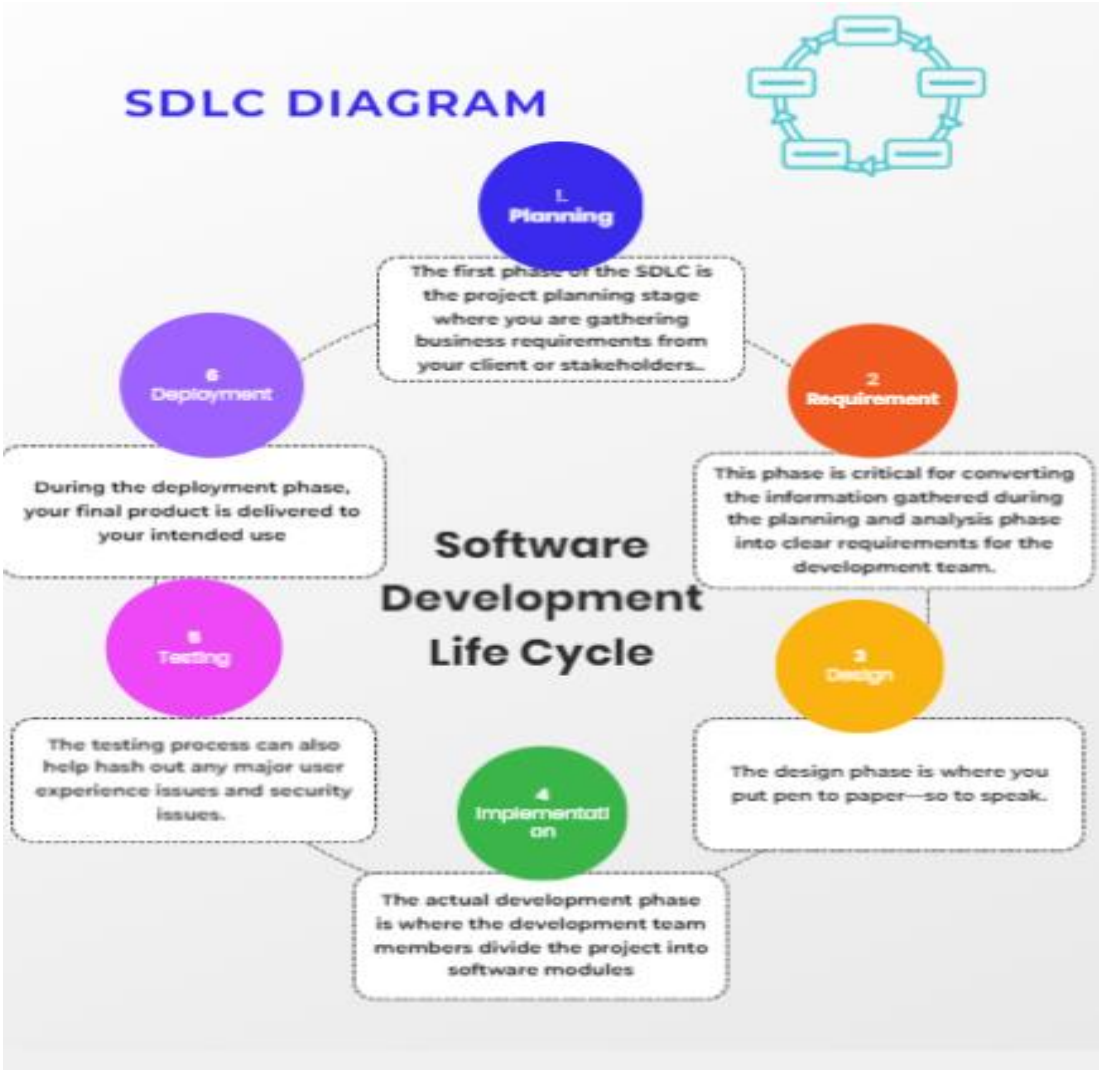
Assignment 1: SDLC Overview - Create a one-page infographic that outlines the SDLC phases (Requirements, Design, Implementation, Testing, Deployment), highlighting the importance of each phase and how they interconnect.

What Is the Software Development Life Cycle (SDLC) :-

The software development life cycle (SDLC) creates an environment where businesses and IT departments can collaborate to produce high-quality software.

The software development life cycle (SDLC) is a set of stages, activities, and tasks that software projects go through.

There are different software development life cycle models specified and designed, which are followed during the software development phase. These models are also called "Software Development Process Models."



Why is SDLC important?

Software development can be challenging to manage due to changing requirements, technology upgrades, and cross-functional collaboration.

The software development lifecycle (SDLC) methodology provides a systematic management framework with specific deliverables at every stage of the software development process.

Here are some benefits of SDLC:

- Increased visibility of the development process for all stakeholders involved
- Efficient estimation, planning, and scheduling
- Improved risk management and cost estimation
- Systematic software delivery and better customer satisfaction

Phases of the software development life cycle:

All software development life cycle models involve various stages. Although these strategies can vary from model to model, below are the following SDLC sequence:

1. Planning
2. Requirement
3. Designing
4. Implementation
5. Testing
6. Deployment

1. Planning:-

The planning phase begins the cycle. It's the time to define the project goals and milestones, attaching a timeline, personnel or skillset, and budget for each activity. While each company may have a unique approach to development planning, this phase typically involves understanding the client's expectations, listing project objectives, identifying the target audience, and highlighting project requirements.

For example, let's say a company wants to create an online ecommerce store. The planning phase would involve deciding on the target audience and suitable features, like live inventory tracking, video product catalogues, and integrated social media. The project lead can conduct surveys and analyses to know the exact expectations of the end users.

Once the team knows customer requirements and the product's expected features, the design team can collect the client's input to determine the project's feasibility and set timelines, milestones, and budgets.

- **Importance:** The Planning Phase sets the project's direction, scope, and objectives. It involves defining project goals, estimating resources, creating schedules, and identifying potential risks.
- **Interconnectivity:** Effective planning is crucial for the success of the entire SDLC. It ensures alignment with business objectives, guides decision-making in subsequent phases, and mitigates risks early in the project lifecycle. Additionally, planning provides a roadmap for efficient execution, facilitating smoother transitions between phases and enhancing overall project management.

2. Requirement :-

Requirement gathering involves collecting all relevant information from the client and using it to create the product, ensuring their expectations are met. To do so, business analysts and project managers meet with the client to discuss software requirements in detail.

The purpose of this meeting is to more thoroughly understand the client's wants and needs, including a description of the software, who the end user will be, and its overall purpose. This information is then written into the software requirement specification (SRS) document.

Once the team creates the SRS document, they pass it to the client for approval. The SRS document can then serve as a guide throughout the designing and development processes.

- **Importance:** This phase involves gathering, analyzing, and documenting requirements from stakeholders. It lays the foundation for the entire project.
- **Interconnectivity:** Clear requirements ensure alignment between stakeholders and developers, guiding subsequent phases effectively.

3. Design:-

In the design phase, software engineers analyze requirements and identify the best solutions to create the software. This helps to guide the developers to create a working representation of the client's expectations while considering user-friendliness and multiscreen compatibility.

For example, if a team is developing an online shop, the design phase might consider the back-end framework, shopping cart features, payment features, and user experience parameters. They might also consider how the checkout page should display to the customer.

- **Importance:** Design phase transforms requirements into a blueprint for the software solution. It defines system architecture, modules, interfaces, and data structures.

- **Interconnectivity:** Designs derived from requirements ensure that the developed software meets user needs and technical specifications.

4. Implementation:-

The implementation phase is where the design created in the design phase is implemented into the necessary application programming interfaces (APIs) and project components. The result of this phase is a fully functional product.

After implementation, the final product requires further testing to address all bugs before release. The team measures the software against the specifications in the SRS document and sends a test version to reviewers. The feedback gathered during this phase allows developers to make necessary product changes before full implementation.

- **Importance:** Implementation involves coding based on the design specifications. It's where the actual development work takes place.
- **Interconnectivity:** A solid design facilitates efficient coding, reducing rework and ensuring the software aligns with the intended functionality.

5. Testing:-

In the testing phase, developers—especially DevOps professionals—verify the software meets the predetermined requirements, designs, and other quality standards. Without proper software testing, the system may contain bugs or vulnerabilities that can go unnoticed and potentially lead to serious problems when put into use.

An example of this is deploying an application to a live environment like the Google Play Store. The application must be tested on various screens beforehand to ensure it works as intended. This includes

validating data input, measuring application performance, and confirming security features are up to parameters.

- **Importance:** Testing verifies that the software meets quality standards and functions as intended. It includes unit, integration, system, and acceptance testing.
- **Interconnectivity:** Testing validates whether the implementation aligns with the requirements and design, identifying and rectifying defects early in the process.

7. Deployment:-

The deployment phase typically consists of putting the software into a production environment so it’s accessible to users. After successful deployment, customers can use the newly developed

software. In some cases, the client may request user acceptance testing before deployment to ensure the software meets expectations.

User acceptance testing involves testing the newly created software and ensuring it performs correctly and meets specific requirements.

- **Importance:** Deployment involves releasing the software for use by end-users. It requires careful planning to ensure a smooth transition from development to production.
- **Interconnectivity:** Successful deployment relies on accurate implementation and rigorous testing, ensuring that the final product meets user expectations and business objectives.