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.

1. **Requirements Gathering:** This phase involves understanding the needs of the stakeholders, such as the users and the business. It includes activities such as conducting interviews, workshops, and user research to define the features and functionalities of the software.

**Importance:** A solid foundation is crucial. By clearly understanding the requirements upfront, you can avoid costly rework later in the process.

1. **Design:** In this phase, the software architecture is created based on the requirements gathered in the previous phase. It includes activities such as defining the user interface (UI), data flow, and system architecture.

**Importance:** The design phase serves as a blueprint for development. A well-designed system is easier to develop, test, and maintain.

1. **Implementation:** This phase involves coding the software based on the design specifications. Developers write code, unit test individual components, and integrate them into a working system.

**Importance:** Here's where the coding magic happens! Implementation brings the design to life.

1. **Testing:** In this phase, the software is thoroughly tested to ensure it meets the requirements and functions as expected. It includes activities such as unit testing, integration testing, system testing, and user acceptance testing (UAT).

**Importance:** Testing is essential for quality assurance. It helps identify and fix bugs before the software is deployed to users.

1. **Deployment:** This phase involves releasing the software to the users. It includes activities such as training users, installing the software, and monitoring its performance.

**Importance:** Deployment is the moment your software reaches the world! A smooth deployment ensures a positive user experience.

**How the Phases Interconnect**

The SDLC phases are not linear but rather iterative. Information and feedback from each phase can influence the previous phases. For example, issues identified during testing may require changes to be made in the design or implementation phases.

By following a well-defined SDLC, software development teams can create high-quality software that meets user needs, is delivered on time and within budget, and is easy to maintain.