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.

Title: The Software Development Life Cycle (SDLC)

1. Requirements

Description:

1. Collect and analyse business needs.
2. Define clear, concise, and measurable requirements.

Importance:

1. Establishes the foundation and scope of the project.
2. Ensures all stakeholders have a common understanding.

Interconnection:

1. Inputs: Stakeholder needs, business objectives
2. Outputs: Requirements specification documents for the Design phase.

2. Design

Description:

1. Create architecture and detailed design for the system.
2. Plan system components, interfaces, and data models.

Importance:

1. Provides a blueprint for the development team.
2. Reduces complexity by breaking down the system into manageable parts.

Interconnection:

1. Inputs: Requirements documents.

2. Outputs: Design documents, architecture diagrams for the Implementation phase.

3. Implementation

Description:

1. Code the software based on design specifications.
2. Integrate system components and ensure they function together.

Importance:

1. Transforms design into a working software system.
2. Core phase where actual product is built.

Interconnection:

1. Inputs: Design documents.
2. Outputs: Source code, executables for the Testing phase.

4. Testing

Description:

1. Verify that the software meets requirements.
2. Identify and fix defects and issues.

Importance:

1. Ensures software quality and functionality.
2. Prevents costly post-deployment fixes.

Interconnection:

1. Inputs: Source code, test plans.
2. Outputs: Tested software, bug reports, feedback for the Implementation phase (if issues are found).

5. Deployment

Description:

1. Release the software to users.
2. Set up production environment and train users.

Importance:

1. Delivers the final product to the end-users.
2. Marks the transition from development to maintenance.

Interconnection:

1. Inputs: Tested software, deployment plans.
2. Outputs: Operational software, user feedback for future requirements.

Interconnection of Phases:

Each phase relies on the outputs of the previous phase.

Continuous feedback loops ensure refinement and improvement.

Effective communication and documentation are critical throughout.