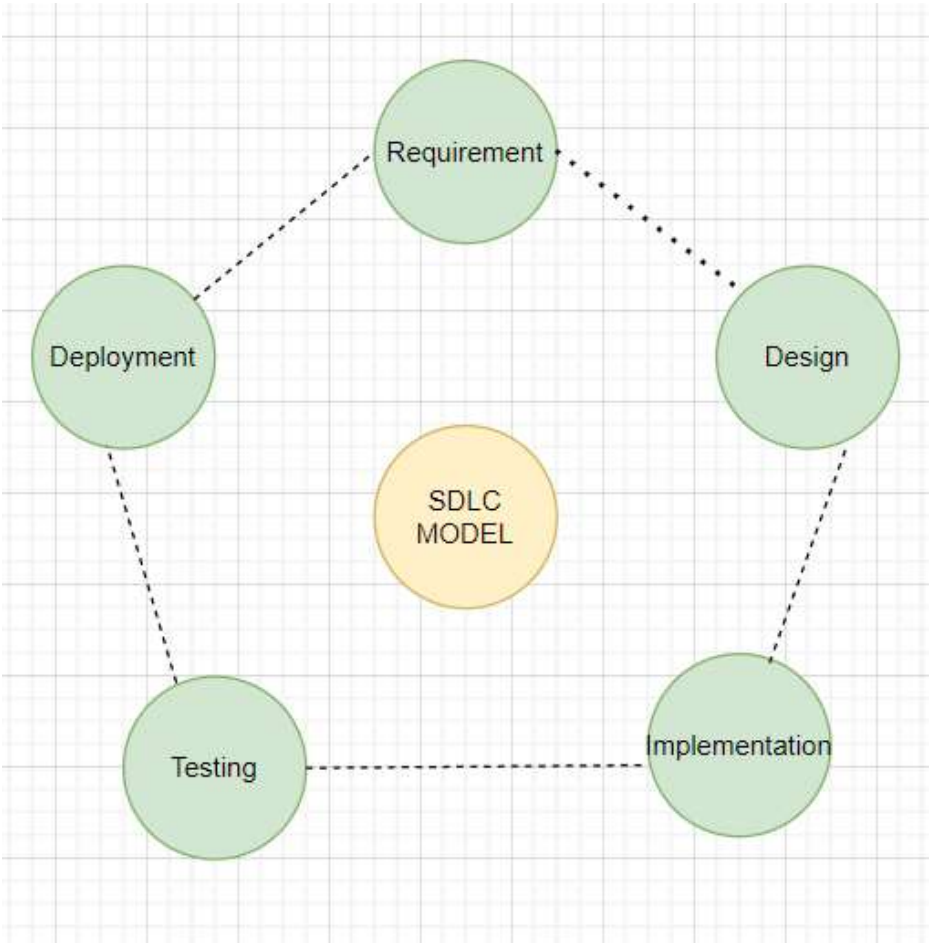
**Assignment 1 : - SDLC Overview - Create a one - page infographic that outlines the SDLC phases(Requirements , design, Implementation, testing , Deployment), highlighting the importance of each phase and how they interconnect.**
**Answer:**




**Importance of each phases :**

**1.Requirement :** This stage Involves gathering, analyzing, and documenting user requirements.For the consistency of the criteria, there will be several review meetings. All findings of the review should be logged and tracked. They suggest that both formal & informal interviews should be held with the applicant's right stakeholders. This will help developers to get a true view of what the application is expected to do.Clear requirements ensure that the final product meets user expectations.

**2.Design:** Translates requirements into a detailed blueprint for the software. Good design ensures scalability, maintainability.

**3.Development :** Involves actual coding based on the design specifications. It's where the software begins to take shape based on the earlier phases.It is important for the coders to follow the protocols set by the Association.

**4.Testing :** After the development of the product, testing of the software is necessary to ensure its smooth execution. Although , minimal testing is conducted at every stage of SDLC. Therefore at this stage all the probable flaws are tracked, fixed, and retested. This ensures that the product confronts the quality requirements of SRS.

**5.Deployment :** After detailed testing, the conclusive product is released in phases as per the organization's strategy. Then it is tested in a real industrial environment. It is important to ensure its smooth performance. If it performs well, the Organization sends out the product as a whole. After retrieving beneficial feedback, the company releases it as it is or with auxiliary improvements to make it further helpful for the customers.

## 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.

### Answer:

**Case Study: Developing a Smart Irrigation System with SDLC**

This case study analyzes the implementation of the Software Development Life Cycle (SDLC) phases in the development of a smart irrigation system. It will evaluate how each phase contributes to the project's overall success.

**1. Requirement Gathering:**

- **Activities:** Interviews with farmers, data analysis on water usage patterns, research on existing irrigation systems.
- **Outcomes:** Clear understanding of farmer needs (water efficiency, cost reduction, remote monitoring), desired features (soil moisture sensors, automated watering schedules), and technical requirements (compatibility with existing infrastructure).
- **Importance:** A solid foundation for the project. Omitting crucial requirements can lead to costly rework later.

**2.Design:**

- **Activities:** System architecture design, user interface (UI) design, choosing appropriate sensors and communication protocols.
- **Outcomes:** Detailed technical specifications, system architecture diagrams, UI mockups, and communication protocols defined.
- **Importance:** Ensures all components work together seamlessly and the UI is user-friendly. Poor design can lead to usability issues and integration problems.

**3. Implementation:**

- **Activities:** Coding the software for the irrigation controller, integrating sensors, developing mobile app for user control**.**
- **Outcomes:** Functional prototype of the smart irrigation system.
- **Importance:** Transforms design into a working system. Bugs and errors can lead to system malfunctions or unexpected behavior.

**4. Testing:**

- **Activities:** Unit testing of individual software modules, integration testing of the complete system, user acceptance testing (UAT) with farmers.
- **Outcomes:** Identification and rectification of bugs, verification of functionalities against requirements.
- **Importance:** Ensures the system functions as intended and meets user expectations. Untested systems can be unreliable and frustrating to use.

**5. Deployment:**

- **Activities:** Installation of the smart irrigation system on farms, user training on system operation and mobile app usage.
- **Outcomes:** Delivery of the functioning system to end users.
- **Importance:** Ensures smooth transition from development to real-world use. Inadequate training can lead to user frustration and misuse of the system.

**6. Maintenance:**

- **Activities:** Bug fixes based on user feedback, software updates with new features, remote monitoring and system health checks.
- **Outcomes:** Ensures continued system reliability, addresses user feedback, and improves functionality over time.
- **Importance:** Prevents system degradation, enhances user experience, and keeps the system competitive in the market.

Effective implementation of all SDLC phases is crucial for the success of the smart irrigation system project. Each phase contributes significantly to meeting user needs, delivering a well-functioning system, and ensuring its long-term adoption by farmers. Neglecting any phase can lead to costly rework, delays, and user dissatisfaction.

## 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.

## Answer : Comparison of SDLC Models for Engineering Projects

### 1. Waterfall Model

- **Description:** A sequential model where each phase (requirements gathering, design, development, testing, deployment, maintenance) is completed in a linear fashion before moving to the next.
- **Advantages:**
    - Clear documentation and defined phases
    - Easier project management and resource allocation
    - Suitable for well-defined projects with clear requirements
- **Disadvantages:**
    - Inflexible, difficult to adapt to changing requirements
    - High risk of rework due to late detection of errors
    - Limited user involvement until late stages
- **Applicability:** Engineering projects with well-defined requirements upfront, such as building a bridge with established specifications.

## 2. Agile Model

- **Description:** An iterative and incremental approach where requirements are broken down into user stories, developed in short sprints, and reviewed continuously.
- **Advantages:**
    - Adaptable to changing requirements
    - Faster feedback loop and early error detection
    - Increased user involvement throughout the development process
- **Disadvantages:**
    - Requires strong team communication and collaboration
    - May not be suitable for projects with strict deadlines or complex dependencies
    - More documentation overhead compared to Waterfall
- **Applicability:** Engineering projects with evolving requirements or those requiring user feedback at each stage, such as developing a new robotics controller.

### 3. Spiral Model

- **Description:** A risk-driven model that combines elements of Waterfall and Agile. Each iteration involves risk identification, mitigation, prototyping, and evaluation before proceeding to the next iteration.
- **Advantages:**
    - Explicitly addresses project risks early on
    - Iterative approach allows for adjustments based on findings
    - Suitable for complex projects with high uncertainty
- **Disadvantages:**
    - Can be more complex to manage than Waterfall or Agile
    - Requires strong risk assessment and mitigation skills
    - May lead to longer development cycles compared to Agile
- **Applicability:** Engineering projects with high levels of uncertainty or significant risks, such as developing a new medical device.

### 4. V-Model

- **Description:** An extension of the Waterfall model that emphasizes verification and validation activities alongside each development phase. Verification ensures the system is built according to specifications, while validation ensures it meets user needs.
- **Advantages:**
    - Strong emphasis on quality assurance and testing
    - Early detection of defects
    - Suitable for safety-critical engineering projects
- **Disadvantages:**
    - Shares the inflexibility of Waterfall
    - May require additional resources for extensive testing
    - Less adaptable to changing requirements
- **Applicability:** Engineering projects where safety and rigorous testing are paramount, such as developing an aircraft control system.