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.

• SDLC is step by step implementation of software.

• Software Development Life Cycle is a systematic approach to develop software. It is a Process followed by Software Developers and Software Tester to provide quality software.

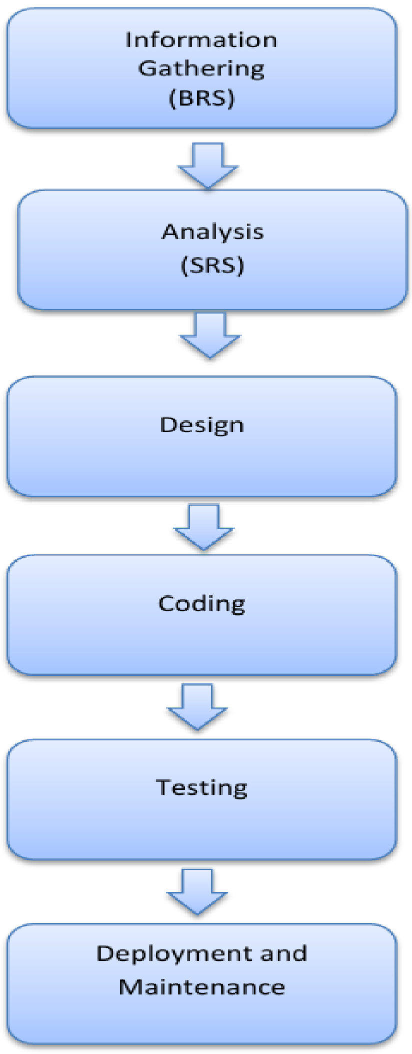
• The SDLC aims to produce high-quality software that meets customer expectations, completion within times and cost estimates.

• ISO/IEC 12207 is an international standard for software life-cycle processes. It defines all the tasks required for developing and maintaining software.

• ISO (the International Organization for Standardization).

• IEC (the International Electro-technical Commission).

Software Development Life Cycle (SDLC):



i) Requirement Gathering

Requirement Gathering is the most important phase in software development life cycle.

“Business requirements are gathered in this phase.

“ Meeting held with client/customer in order to determine the requirements like; -

- Who is going to use the system?

- How will they use the system/ domain of the system

- What data should be input into the system?

- What data should be output by the system?

Business Analyst collects the requirements from the Customer/Client as per the client’s business needs and documents the requirements in the Business Requirement Specification (BRS). Hence BRS is a bridge between Client (Customer) and Business analyst (Company).

Roles Involved: Business Analyst (BA).

Outcome: Business Requirement Specification (BRS)

ii) Analysis

Once the BRS is get ready the next step is to do the analysis of BRS and prepare a SRS document accordingly. SRS documents consist of detail technical information and specification of software. SRS consists of all the product requirements to be designed and developed during the project life cycle. SRS is detailed technical description of BRS.

Key people involved in this phase are Project Manager, Business Analyst and Senior members of the development and tester Team.

Roles Involved: Business analyst, Dev. & QA team head, Design engineer, Project Managers

Outcome: Software requirement specification (SRS)

iii) Design

In Design phase Senior Developers and Architects, they give the architecture of the software product to the developer. It has two steps one is HLD (High Level Design) and another is LLD (Low Level Design).

> High Level Design (HLD) is the overall system design, covers the system main module.

Ex: Sign up module is one of the main module in Gmail

> Low Level Design (LLD) is the detailed system design, covers sub modules which comes under main module.

Ex: Under sign up module we have different field like First name, last name, Mob no. etc..

> The outcome of this phase is High Level Document and Low Level Document which works as an input to the next phase Coding.

Roles Involved: UI and UX developer.

Outcome: Technical Design Document (TDD)

IV) Coding / Development

> Developers (seniors, juniors and fresher) involved in this phase, this is the phase where we start building the software and start writing the code for the product. The work is divided in modules/units and actual coding is started in this coding phase and it is the main focus for developer. Coding is one of the longest phases of SDLC.

>Coding means programming, one line code is called instruction and multiple line of instruction is called Program. And multiple program forms software.

> The outcome of this phase is Source Code Document (SCD) and the developed product.

> Generally following type of developers are used for coding stage

> The developer who has knowledge of both front end and back end is called, full stack developer. Roles involved: Developers (All type)

Outcome: Programs or Application or Module

v) Testing

> Once the software is complete then it is deployed in the testing environment. The testing team starts testing (either test the software manually or using automated test tool)

> Testing is done to verify that the entire application works according to the customer requirement.

> During this phase, testing team may find defects which they communicate to developers; the development team fixes the defect and sends back to Testing for a re-test. This process continues until the software is working according to the business needs of that system.

Roles Involved: Testers.

Outcome: Defects List, Test Summary Report, Test Plan, Test Case document

vi) Deployment & Maintenance

> After successful testing, the application is transfer to production (deployed to the customer for their use), Deployment is done by the Developer and Once when the customers start using the developed system then the actual problems will come up and needs to be solved from time to time.

> Fixing the issues found by the customer comes in the maintenance phase. 100% testing is not possible – because, the way testers test the product is different from the way customers use the product. Maintenance should be done as per SLA (Service Level Agreement)

Roles Involved: Testers, Developers, Customer, Business team, Architects, Project Manager, and Delivery Manager

Outcome: Quality Product, Enhancements & Production Issues (Maintenance)

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.

Case Study: Building a Library Management System (LMS)

This case study analyzes the implementation of SDLC phases in the development of a Library Management System (LMS) for a university library. It evaluates how each phase contributes to the project outcomes.

Project Scope:

The LMS aims to automate core library functions like book borrowing and returning, member management, book search and reservation, and overdue notifications.

SDLC Phases and their Contribution:

1. Requirement Gathering:

- **Activities:** Interviews with librarians, students, and faculty to understand their needs and pain points.
- **Outcomes:** A well-defined list of functional and non-functional requirements, including system features, user roles, performance expectations, and security needs. This clear understanding minimizes development rework later.

2. Design:

- **Activities:** Creating system architecture diagrams, user interface (UI) mockups, and data flow diagrams. Defining database schema and functionalities like search algorithms.
- **Outcomes:** A blueprint for the system, ensuring a cohesive and user-friendly experience. Design flaws identified and addressed early prevent costly coding errors later.

3. Implementation:

- **Activities:** Developers write code based on the design documents. Unit testing of individual code modules is performed to ensure their functionality.
- **Outcomes:** A functional LMS application ready for integration testing. Early unit testing helps identify and fix bugs early in the development cycle.

4. Testing:

- **Activities:** Integration testing to ensure different modules work together seamlessly. System testing to validate the entire application against the requirements. User Acceptance Testing (UAT) with librarians, students, and faculty to gather feedback and identify usability issues.
- **Outcomes:** A system free of critical bugs and meets user expectations. Early detection of issues through rigorous testing minimizes post-deployment problems.

5. Deployment:

- **Activities:** Careful planning and execution to migrate data from the old system (if any) and deploy the LMS to the production environment. User training and documentation are provided.
- **Outcomes:** A smooth transition to the new system with minimal disruption to library operations. Training ensures users can leverage the system's functionalities effectively.

6. Maintenance:

- **Activities:** Addressing bug reports and feature requests submitted by users. Regular system updates and security patches are implemented. Performance monitoring and optimization are conducted.
- **Outcomes:** A continuously improving LMS that remains reliable, secure, and meets evolving user needs. Maintenance ensures the system stays relevant and adapts to changing requirements.

Evaluation of Project Outcomes:

- **Reduced Manual Work:** Automating tasks like borrowing and returning books frees up librarian time for other tasks.
- **Improved User Experience:** Online search and reservation capabilities enhance the user experience for students and faculty.
- **Increased Efficiency:** Streamlined workflows improve overall library operations and resource management.
- **Scalability:** The system can be scaled to accommodate a growing user base and book collection.
- **Reduced Costs:** Automation can lead to cost savings in the long run due to reduced manual labor and improved efficiency.

Conclusion:

By following a structured SDLC approach, the LMS project achieved its goals of automating core library functions, improving user experience, and enhancing overall library operations. Each phase of the SDLC played a vital role in this success. Requirement gathering ensured the system addressed real needs. Design provided a roadmap for development. Implementation translated the design into a working system. Testing identified and corrected issues before deployment. Deployment delivered the system to users with proper training. Finally, maintenance ensures the system remains functional, secure, and adaptable over time.