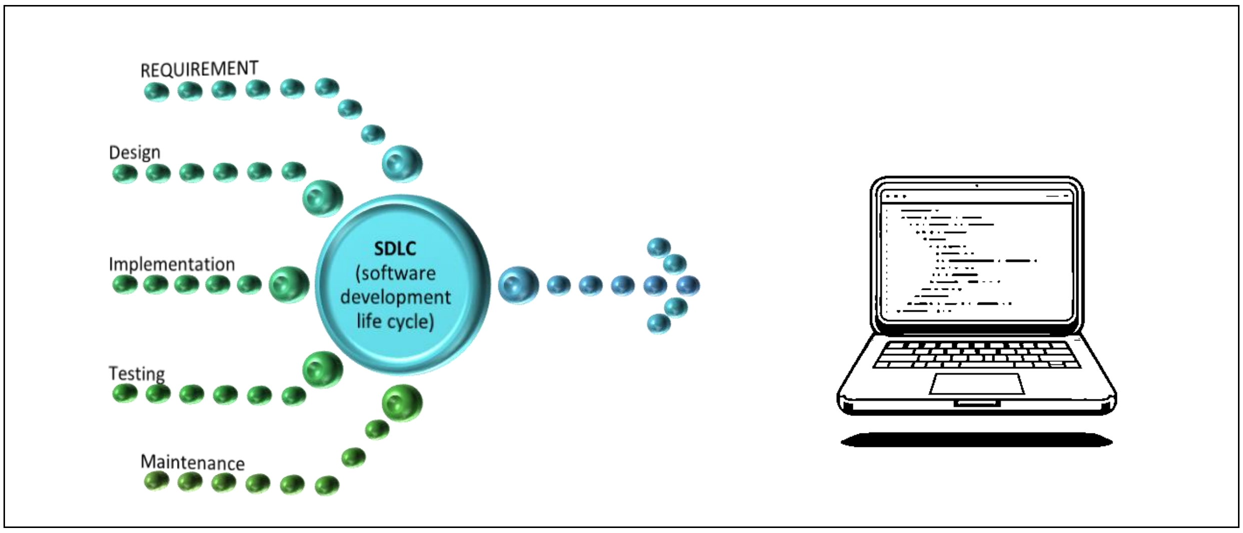
Assignment 1: SDLC Overview - Create a one-page infographic that outlines the SDLC phases (Requirements, Design, Implementation, Testing, Deployment), highlighting the importance of each phase and how they interconnect.

SDLC phases:



A simple representation of SDLC.

Phases of SDLC (Software Development Life Cycle):

- ***Requirement***
- ***Design***
- ***Implementation***
- ***Testing***
- ***Maintenance or Deployment***

- ***Requirement:*** Here BA (Business Analyst) gathers the client's needs, understands their business, and ensures that enough resources are available for design and planning to move forward. An SRS (software requirement specification) with all requirements is also created.

  - ***Importance:***
    - **This ensures that the right product is built by Team.**
    - **Clear understanding of objectives and goals.**
    - **Better Resources utilization.**
    - **Clear understanding of client's needs.**

  - ***Interconnection with design phase:***
    - **If requirements are unclear that leads to flawed designs.**

- ***Design:*** After gathering the detailed requirements (SRS), the design phase begins with creating low-level designs for initial client confirmation. Once approved, a more detailed high-level design is developed using tools like Figma. This design takes into account all factors, and the most practical and logical approach is chosen for development.

  - ***Importance:***
    - **Prototype or modelling helps stakeholder to review.**
    - **Bridges the gap between requirements and implementation ensuring the project aligns with client needs.**
    - **Architectural designs are created.**

  - ***Interconnection with development phase:***
    - **A well-thought-out design leads to more efficient and effective development.**
    - **Clear understanding of working flow reducing ambiguity during implementation.**

- ***Implementation /development:*** This is the phase where most of the time and resources are utilized to create a compact working functional software. In this phase all coding tasks and application building with version controlling and unit testing is performed.

  - ***Importance:***
    - **Core functionalities are created.**
    - **Working developed software are being created.**
    - **Decides major cost and profits.**
  - ***Interconnection with Testing phase:***
    - **High quality code reduces the task and need of testing**
    - **Highly impact cycles of testing.**

- ***Testing:*** After finishing coding development this phase ensures the quality of work is achieved or not as per SRS. Tester with their testcases performs operation also a end user test their scenarios in this phase.

  - ***Importance:***
    - **Ensures Better quality and no defects.**
    - **Enhance software reliability and user satisfactions.**
    - **Identify bugs early so reduce the cost of fixes.**
  - ***Interconnection with Maintenance and implementation phase:***
    - ***A well tested software with no bugs can be sent to development and to end user.***
    - ***This phase shows the quality of coder/programmer as less defects shows high quality code.***

- ***Maintenance:*** In this phase, the well-tested software is delivered to end users. It also involves addressing any bugs or issues identified by users and applying fixes accordingly. The phase ensures optimal utilization of servers and other external resources for smooth functioning.
  - ***Importance:***
    - **Ensures the product reaches end users and delivers its intended value.**
    - **Allows the client to start generating revenue from the product.**
    - **Maintains product quality by applying hotfixes and updates.**
    - **Includes ongoing monitoring to ensure the software functions as expected.**

  - ***Interconnection with requirement (back to start):***
    - **User feedback from this phase often drives updates and new features, initiating the SDLC process anew.**

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.

Project Case study:

The ADMS (Accidental data management system) software is used to track and analysis data related to accidents in states that include crashes, fatalities, injuries and other damages. The Primary goal is to increase safety measures, get real time accurate data, better road infra, better policy making. For this project SDLC methodology is adhered for better structure.

Requirement Gathering:

- **Activity:**
  - **Collaborated with state government agencies, law enforcement**, and transportation departments to collect relevant accident data.
  - Conducted **stakeholder interviews** to gather user stories and understand key reporting requirements.
  - Identified **critical tracking parameters** such as crash details, injury severity, fatalities, locations, and contributing factors.
  - **Defined visualization needs** for data, including heatmaps, trend analysis, and prioritized accident hotspots.
- **Outcome**: A detailed SRS document with:
  - A clear understanding of stakeholder requirements.
  - Precise data fields and parameters for system functionality.
- ***Evaluation***: Comprehensive requirement gathering ensured:
  - Alignment with real-world needs and stakeholder expectations.
  - A robust foundation for system design.

Design:

- **Activity:**
  - Create **system architectural** design.
  - Create **Database schema**
  - **Multiple datasets** should be created for older data to get necessary idea.
  - **Ui/Ux designs** with proper functionality, heatmaps, dashboard design etc must be prepared in this phase.
- **Outcome**:
  - A scalable dataset and user-friendly design.
  - System functionality and future growth idea.

- ***Evaluation***: clear design will make further development easier that will lead to better functional product.

Implementation:

- **Activity:**
  - Database Is quite vast so proper **sharding, partitioning** of data can be used also a multiple server database schema can be followed as per location.
  - **Dashboard, heatmap generation, prone accident location info and other formula**e to describe those properties can be quite complex, so a quality optimized code would be needed.
  - **Parallel tasks must be implemented** so data processing and analysis of data can be smooth.
- **Outcome:**
  - A functional system that captured and processed accident data in real time.
- **Evaluation:**
  - Incremental development allowed early testing and ensured the system met user expectations.

Testing:

- **Activity:**
  - Project consist large data hence a **comprehensive testing** is required.
  - **Unit testing** for data entry modules is needed.
  - **System testing with larger dataset** must be performed for valid data & reports.
- **Outcome:**
  - Identified and resolved discrepancies in data handling and display, ensuring accuracy and reliability.
- **Evaluation:** Thorough testing ensured the system's reliability and readiness for deployment.

Deployment:

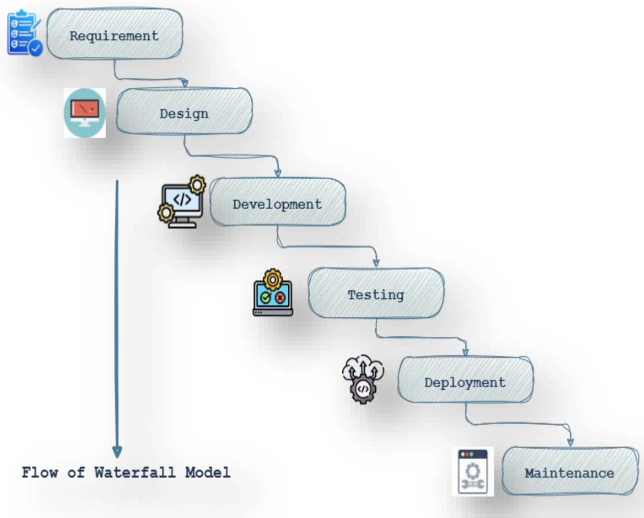
- **Activity:**
  - Deploy system to **controlled environment** and test it.
  - **Trained users like transport data experts, law enforcement personnel must analysis result created**.
  - **Roll out Software to global or specified location** and test system working correctly.
  - Continuously ask for feedbacks for accuracy of system and further improvements.
- **Outcome:**
  - **Revenue generation**.
  - Higher scope for update system with newer requirements.
- **Evaluation:** A phased roll-out strategy minimized risks and enabled real-time troubleshooting.

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.

Research and comparison of SDLC Models:

Waterfall Model:

- **Linear and sequential** model.
- One way **Top to bottom approach**, next phase starts after completion of previous phase.
- **6 phases**: **Requirement, Design, Development, Testing, Deployment and Maintenance**
- **Requirement:** Getting business requirement and gether SRS, CRS,use cases, user stories etc and a feasibility analysis also done. **For waterfall model it is the most important phase** as there will be no changes SRS after completion of this phase.
- **Design:** Ui/Ux design, Software architecture, Servers, OS and Low-level design etc are created from requirements.
- **Development:** Source code, version control and build version with almost all functionalities are created in this phase.



- **Testing:** After development software will be tested with multiple environments and found out defects in codes or functionalities, developers will fix them and make software ready for next phase.
- **Deployment:** Software will be deployed in real world environment remotely by release engg on the server.
- **Maintenance:** If any changes are required like addition, removal or updation of any functionality is done in this phase.

Advantages:

- **Clear goals and requirements.**
- **Easier project management due to structured phases.**
- **Stable requirements enable faster development and testing**
- **Timelines and deadlines are very realistic.**

Disadvantages:

- Minimal client involvement after the 1st phase, leads to unmatched expectations.
- Highly inflexible and changes are not accommodated.
- Lacks prototype or early builds.
- Late discovery of defects so higher cost.

Applicability:

- Should be used when requirement very clear, stable and should not change until whole development is completed.

- Best for small sized projects as their requirement might be very clear and it provide faster development.
- If there are limited resources.

Best Use cases:

- Small projects like School Attendance management.
- Civil Engineering projects.
- Constructing simple mechanical device projects.

Spiral Model:

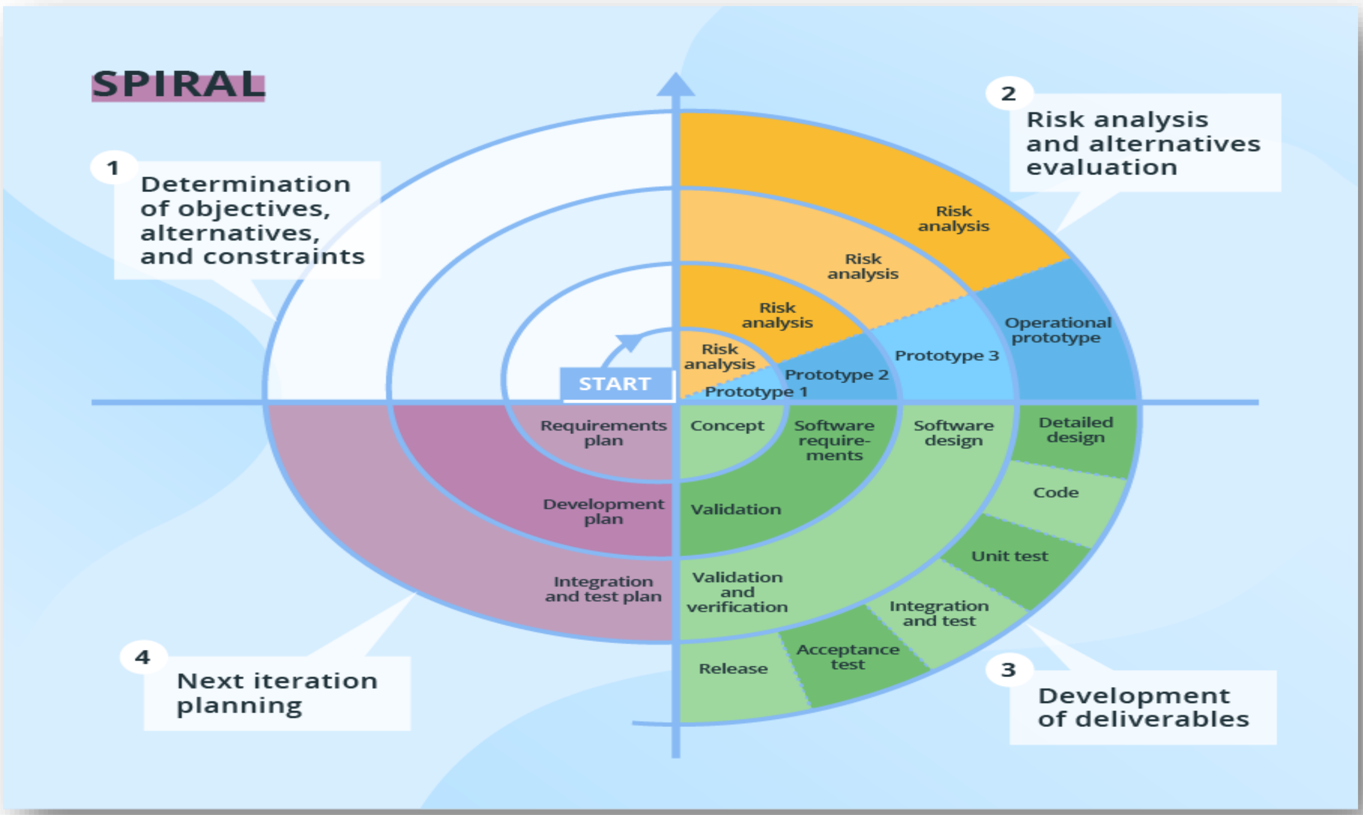


Image ref: https://www.scnsoft.com/blog-pictures/custom-software-development/4-spiral.png

- **iterative development with risk analysis at each phase.**
- **Each spiral includes all waterfall phases with risk management.**
- **A prototype is created after every iteration/spiral.**
- Requirement can be very after a prototype is created.

***Advantages**:*

- Highly **Flexible**.
- Very less chances of late findings of defects as there is risk analysis at every step.
- Client's involvement needed at each step so final software align with their expectation.
- **High quality product**.

***Disadvantages**:*

- Expertise needed in risk analysis.
- Time consuming.
- Increased complexity may decrease productivity.
- Very Expensive.

Applicability:

- Suitable for Large project.
- Projects where very detailing is needed and are too ambitious.
- Safety related projects.
- High uncertainty is required in projects.

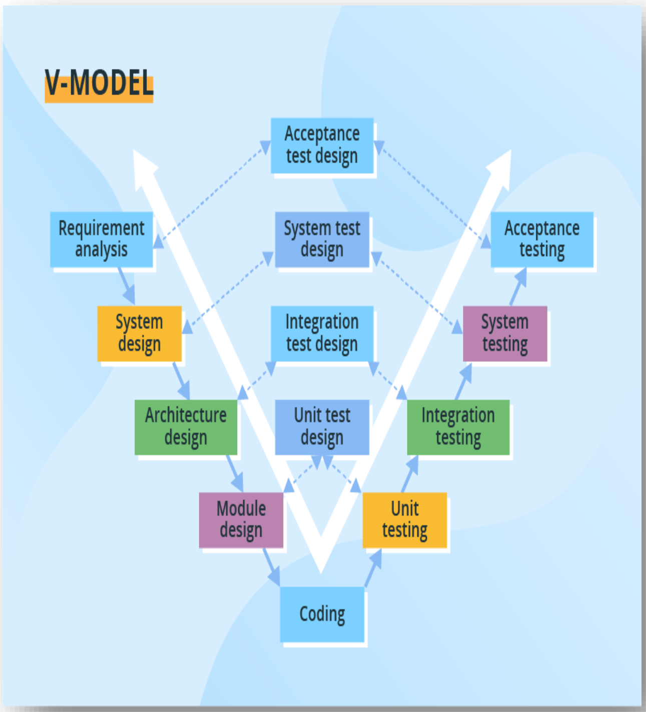
Best Use cases:

- Infrastructural projects.
- Cutting edge research and development.

.

V-Model or verification-validation model:

- **Linear model with corresponding testing activity.**
- Parallel testing and development.
- An extension of waterfall having testing(validations) at each phase.



***Advantages**:*

- Errors/defects are detected early.
- High quality delivery.
- Documentation makes clear goals and understanding.
- Better Resource management.

***Disadvantages**:*

- Time consuming.
- Inflexible.
- High cost in dynamic projects.

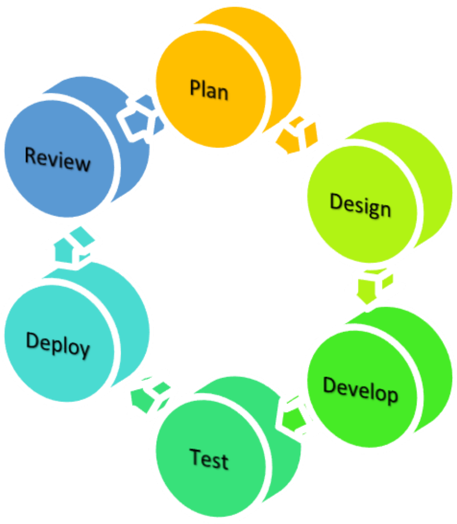
Applicability:

- Projects where quality is very important.
- When Failure is unacceptable.
- Project where Safety is needed.

Best Use cases:

- Medical projects.
- Military applications.
- Automotive and safety critical projects.

Agile model:



- Agile model is **iterative and incremental.**
- focusing on **flexibility and customer collaboration**
- Stakeholder feedback is integrated continuously
- Several Agile methodologies, including Scrum, Kanban, and Extreme Programming (XP), have been developed to implement these principles.

Advantages:

- Highly flexible and adaptive to changing requirements
- Encourages active stakeholder involvement.
- Focuses on user satisfaction and value-driven outcomes.
- Delivers working software quickly through iterative cycles.

Disadvantages:

- Documentation can be insufficient due to rapid development cycles.
- Difficult to predict project timelines and costs accurately
- Requires active user and client involvement.

Applicability:

- Software projects & computer applications.
- Iot projects.
- Dynamic engineering projects.

Best Use cases:

- E-comm apps.
- Games.