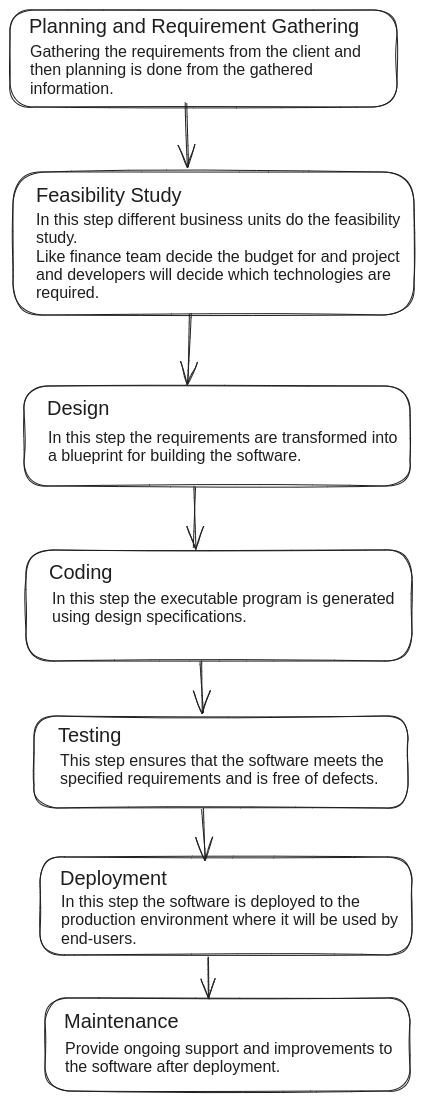
**Assignment 1: SDLC Overview - Create a one-page infographic that outlines the SDLC phases (Requirements, Design, Implementation, Testing, Deployment), highlighting the importance of each phase and how they interconnect.**

****

**Assignment 2: Develop a case study analysing the implementation of SDLC phases in a real-world engineering project. Evaluate how Planning and Requirement Gathering, Feasibility study, Design, Implementation, Testing, Deployment, and Maintenance contribute to project outcomes.**

**Case** **study-** The Android chat app where users can send text, multimedia messages, and engage in real-time communication.

**SDLC phases:**

1. **Planning and Requirement gathering:** In this step define scope of the chat app and gather user requirements. Comprehensive project plan outlining timelines , resource allocation, and deliverables. Clear understanding of user expectations to guide subsequent phases effectively.
2. **Feasibility study:** In this step evaluate technical requirements, analyse potential risks (data security and scalability), and estimate project cost and timelines. Informed decision-making on project feasibility, identification of potential challenges, and mitigation strategies to ensure project success.
3. **Design:** This step deals with different designing aspects of the app, like-

* Architectural design- Define the app architecture for scalability and performance. Scalable architecture ensures seamless real-time messaging, robust server-client interaction, and efficient handling of multimedia content.
* User Interface design- Design an intuitive and visually appealing interface for Android devices. User-friendly interface promoting easy navigation, efficient message management, and customisable settings.
* Database design- Design an efficient database schema for storing user profiles, chat histories, and multimedia content. Optimised database structure facilitating quick access to messages, efficient storage management, scalability for future enhancements.

1. **Coding:** Translate design specifications into functional code. In this step the features are developed which are aligning with design specifications, adherence to Android coding standards, and implementation of Firebase security rules for data protection.
2. **Testing:** Ensures app functionality, performance, and security. In this step there is identification and resolution of bugs during development phases, assurance of app stability, and compliance with performance benchmarks and security standards.
3. **Deployment:** App is deployed for public availability. Successful deployment reaching a wide audience of Android users, ensuring app discoverability, installation reliability, and seamless updates through the Play store.
4. **Maintenance:** Provide ongoing support, updates, and enhancements to the Android chat app after the deployment. Continued app reliability, improved user satisfaction through continuous improvements, adaptation to evolving Android platform updates and user expectations.

The structured implementation of SDLC phases in developing the Android chat app ensured a robust, user-friendly, and scalable solution that met both technical requirements and user expectations. By following SDLC principles from planning and requirement gathering through maintenance, the project can achieve successful outcomes in terms of functionality, performance, security, and user satisfaction.

**Assignment 3: Research and compare SDLC models suitable for engineering projects. Present findings on Waterfall, Agile, Spiral, and V-Model approaches, emphasising their advantages, disadvantages, and applicability in different engineering contexts.**

Software Development Life Cycle (SDLC) models provide structured approaches to software development, each with its own strengths and weaknesses.

1. **Waterfall Model**

The Waterfall model is a linear and sequential approach where each phase must be completed before the next begins. It flows a top-down methodology, starting from planning and requirement gathering to maintenance.

**Advantages:**

* **Structured approach:** Clear, defined stages with specific deliverables and review processes.
* **Easy to understand:** Simple and straightforward, making it easy to understand and manage.
* **Documentation:** Extensive documentation provides a clear path for maintenance and future update.

**Disadvantages:**

* **Inflexibility:** Difficult to go back and make changes once a phase is completed.
* **Late testing:** Testing phase comes late in the development cycle, making it hard to address issues.
* **Risk management:** Poor at handling risks and changes during the development process.

**Applicability:**

Best suited for projects with well-defined requirements that are

that are unlikely to change, such as a Calculator app where

stages are clear and changes are minimal.

1. **Agile Model**

The Agile model emphasises iterative development, where requirements evolve through collaboration between self-organising and cross-functional teams. It promotes adaptive planning, evolutionary development, early delivery, and continuous improvement.

**Advantages:**

* **Flexibility:** Easily accommodates changes and new requirements.
* **Customer Involvement:** Continuous feedback from stakeholders ensures the product meets their needs.
* **Early and Frequent Delivery:** Regular releases allow for early product delivery and faster time to market.

**Disadvantages:**

* **Unpredictable:** Due to evolving requirements, it is difficult to predict the end product, timelines, and cost.
* **Dependency on team dynamics:** Agile model requires a team to collaborate and make decisions together. Inadequate communication within the team or lack of experience or expertise among team members can negatively affect the output quality and productivity.
* **Documentation:** Less emphasis on documentation can lead to knowledge gaps.

**Applicability**

Ideal for projects with rapidly changing requirements.

1. **Spiral Model**

This is a SDLC model that provides a systematic and iterative approach to software development. It emphasises risk analysis and is represented as a spiral with each loop representing s phase in the development process.

**Advantages:**

* **Risk Management:** Focuses on identifying and mitigating risks early in each iteration.
* **Flexibility:** Allows for changes and refinements throughout the development process.
* **Customer Feedback:** Continuous customer involvement ensures alignment with user needs.
* **Early prototyping:** Help visualise and refine requirements, reducing ambiguity.
* **Iterative Development:** Enables incremental improvements and learning in each cycle.
* **Resource Management:** Facilitates better planning and allocation based on priorities and risks.

**Disadvantages:**

* **Complexity:** The model can be quite complex for small projects where there is a small team as it requires expertise in risk assessment and management, which can be challenging for small teams.
* **Cost:** Iterative processes and risk assessments can increase cost.
* **Time-Consuming:** Extensive planning and prototyping can extend project timelines.
* **Requires expertise:** Needs skilled project managers and developers for successful implementation.
* **Difficulty in Milestone Definition:** Challenging to define clear milestones due to iterative nature.

**Applicability:**

The spiral model is best for large, complex projects with evolving requirements and high risks, where continuous customer feedback and prototyping is necessary.

1. **V-Model**

The V-Model (Verification and Validation Model) extends the waterfall model by emphasising testing phases that correspond to each development stage, forming a V shape.

**Advantages:**

* **Structured approach:** Clearly defined development and testing phases.
* **Early Defect Detection:** Testing starts early, catching issues sooner.
* **Clear Milestones:** Defined stages helps in project tracking.
* **Comprehensive Verification and Validation:** Ensures high- quality output aligned with requirements.
* **Reduced Post-Release Defects:** Thorough testing minimises issues after deployment.

**Disadvantages:**

* **Inflexible to changes:** Difficult to accommodate changes in once a stage is completed.
* **Not suitable for Complex Projects:** Limited adaptability for evolving requirements.
* **High Dependence on Initial Requirements:** Relies heavily on upfront clarity of requirements.
* **Increased Time and Cost:** Sequential process may prolong timelines and increase costs, especially if changes are required later in the process.
* **Limited Stakeholder Involvement:** Stakeholders may have less engagement until later stages, which could lead to misunderstandings or discrepancies between expectations and deliverables.

**Applicability:**

The V-Model is best suited for projects that prioritise structured development, early testing, and adherence to predefined requirements and standards.