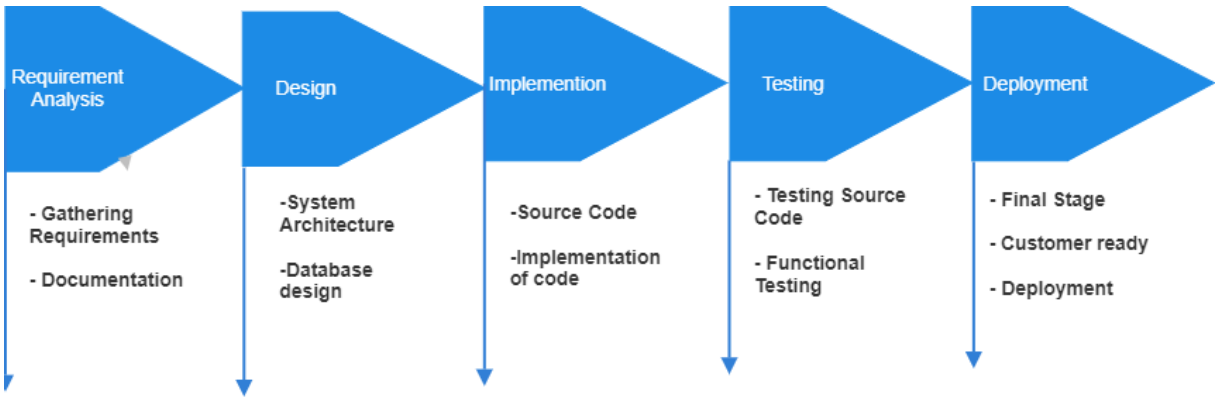
Que 1.

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.



Que 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.

Ans: Case Study for Banking System:

Case Study: Implementation of SDLC Phases in the Development of a Banking System:

Project Overview: A leading bank sought to develop a new online banking system to improve customer experience, enhance security, and offer new services. The project aimed to integrate functionalities such as account management, fund transfers, bill payments, and customer support.

Phase 1: Requirement Gathering

Activities:

- **Stakeholder Interviews:** Met with bank executives, IT staff, and customer service representatives.
- **Customer Surveys:** Gathered feedback from bank customers on desired features.
- **Regulatory Review:** Analyzed compliance requirements for financial systems.

Outcomes:

- **Functional Requirements:** Secure login, account overview, transaction history, fund transfers.
- **Non-functional Requirements:** High security, reliability, and user-friendly interface.
- **Constraints:** Regulatory compliance, existing IT infrastructure, and budget.

Evaluation: Comprehensive requirement gathering ensured alignment with business goals and customer needs, forming a robust project basis.

Phase 2: Design

Activities:

- **System Architecture Design:** Developed high-level system architecture.
- **Detailed Design:** Created detailed specifications for each module (e.g., authentication, transaction processing).
- **Prototyping:** Built UI prototypes for user feedback.

Outcomes:

- **Blueprints:** Detailed design documents.
- **Validated Prototypes:** Early user feedback incorporated into design.
- **Risk Management Plan:** Identified and planned for potential risks.

Evaluation: Thorough design phase facilitated clear understanding and alignment, minimizing design flaws.

Phase 3: Implementation

Activities:

- **Coding:** Developed software modules.
- **Integration:** Integrated new system with existing banking infrastructure.
- **Testing Environment Setup:** Prepared environments for upcoming testing phases.

Outcomes:

- **Operational Software:** Functional software modules.
- **Integration Milestones:** Successfully integrated key components.
- **Documentation:** Comprehensive system documentation.

Evaluation: Successful implementation through coordinated efforts, though integration required addressing unforeseen challenges.

Phase 4: Testing

Activities:

- **Unit Testing:** Tested individual software modules.
- **System Testing:** Verified end-to-end functionality.
- **User Acceptance Testing (UAT):** Ensured system met user expectations.

Outcomes:

- **Resolved Issues:** Identified and fixed bugs.
- **Performance Metrics:** Assessed system performance under various conditions.
- **Stakeholder Approval:** Secured approval based on UAT results.

Evaluation: Extensive testing ensured the system's robustness and reliability, with thorough issue resolution.

Phase 5: Deployment

Activities:

- **Pilot Deployment:** Limited rollout to test in real-world conditions.
- **Full Deployment:** Gradual rollout across all users.
- **Training:** Provided training for bank staff.

Outcomes:

- **Operational System:** Full deployment completed.
- **Real-world Feedback:** Gathered data for further optimization.
- **Trained Personnel:** Staff trained and ready to assist customers.

Evaluation: Phased deployment minimized risks and ensured a smooth transition.

Phase 6: Maintenance

Activities:

- **Monitoring:** Ongoing performance monitoring.
- **Updates:** Regular software updates and security patches.
- **Support:** Established customer support channels.

Outcomes:

- **System Reliability:** Maintained high availability and performance.
- **Continuous Improvement:** Implemented updates based on feedback.
- **Efficient Support:** Effective issue resolution and customer assistance.

Evaluation: Proactive maintenance and support ensured long-term system reliability and customer satisfaction.

Que.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.

Ans:

All models for Case Study "Banking System":

1. Waterfall Model

Overview:

A linear and sequential approach where each phase must be completed before the next begins.

Advantages:

- **Predictability**: Clear stages and timelines.
- **Documentation**: Extensive documentation ensures regulatory compliance.
- **Simplicity**: Easy to manage and understand.

Disadvantages:

- **Inflexibility**: Hard to accommodate changes after the project starts.
- **Late Testing**: Issues might be discovered late in the process.

Applicability:

Suitable for projects with well-defined, stable requirements and a strong emphasis on documentation and regulatory compliance.

2. Agile Model

Overview:

An iterative approach focusing on collaboration and customer feedback, with continuous planning, development, testing, and integration.

Advantages:

- **Flexibility**: Adapts to changing requirements.
- **Customer Involvement**: Continuous feedback improves alignment with user needs.
- **Frequent Releases**: Allows for early detection and resolution of issues.

Disadvantages:

- **Less Predictable**: Timelines and costs can be less predictable.
- **Documentation**: Often less detailed, which can be a challenge for compliance.

Applicability:
Ideal for projects where requirements may evolve, such as online banking features or mobile banking apps, where rapid iteration and customer feedback are crucial.

3. Spiral Model
Overview:
Combines design and prototyping in stages, with a focus on risk assessment and mitigation through iterative refinement.
Advantages:

- **Risk Management**: Proactive identification and reduction of risks.
- **Flexibility**: Iterative process allows for adjustments based on feedback.
- **Customer Feedback**: Regular input ensures product meets user needs.

Disadvantages:

- **Complexity**: Managing iterations can be complicated.
- **Cost**: Higher costs due to multiple iterations.
- **Expertise**: Requires skilled risk management.

Applicability:
Best for large-scale, high-risk projects like core banking systems where risk management and iterative refinement are crucial.

4. V-Model (Verification and Validation Model)
Overview:
An extension of the Waterfall model, emphasizing verification and validation at each development stage.
Advantages:

- **Early Detection of Defects**: Systematic testing at each phase.
- **Structured**: Clear stages with corresponding testing.
- **Parallel Processes**: Development and testing occur simultaneously.

Disadvantages:

- **Inflexibility**: Less adaptable to changes.
- **Cost and Time**: Heavy testing requirements can be resource-intensive.

Applicability:
Suitable for projects requiring rigorous validation and verification, such as transaction processing systems, where reliability and security are paramount.