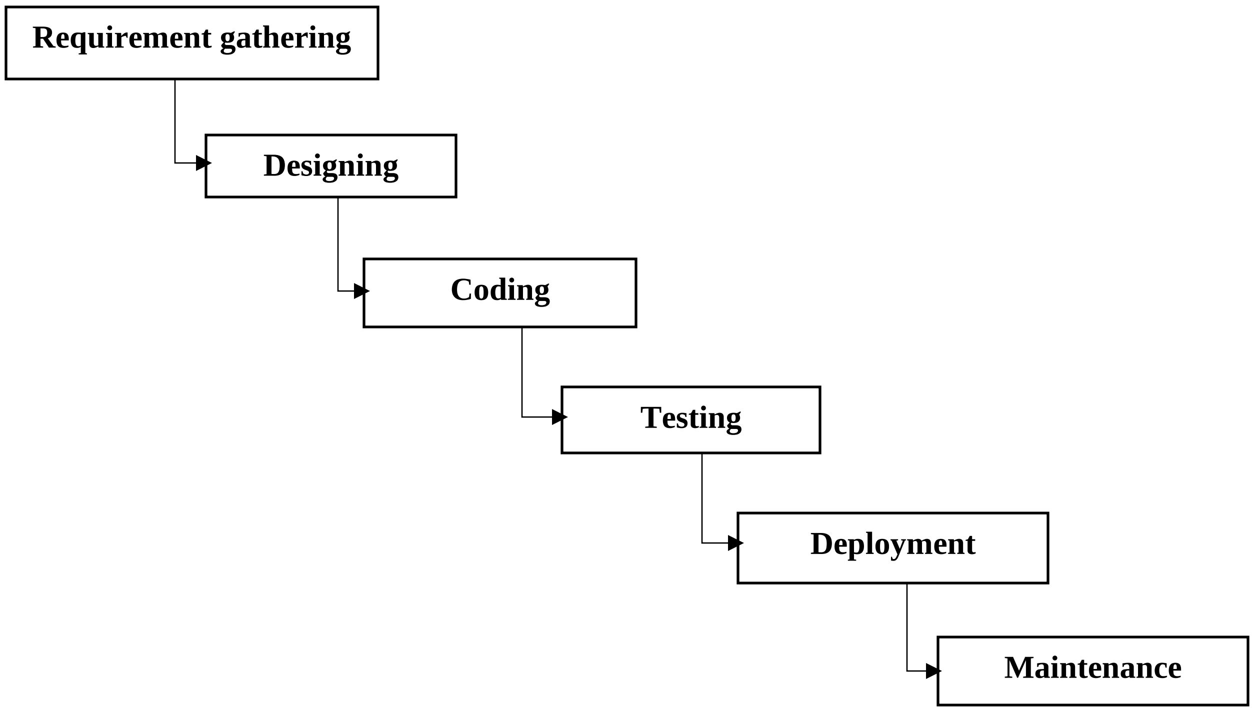
Assignment 1:

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

Work flow of the Software development life cycle



1. Requirements Gathering:

- ❖ This is where the developers talk with the people who need the software (like clients or users) to understand what the software should do.
- ❖ They ask questions, make lists, and create documents to capture all the important details.

2. Design:

- ❖ Once they know what the software needs to do, they start planning how to build it.
- ❖ They design the structure of the software, like deciding what pages or screens it will have, what buttons will do, and how everything will fit together.

3. Implementation (Coding):

- ❖ This is where the actual building of the software happens.
- ❖ Developers write the code, which is like giving instructions to the computer on what to do.
- ❖ They take the designs and turn them into a working software program.

4. **Testing:**
    - ❖ After the software is built, it's time to make sure it works correctly.
    - ❖ Testers try out the software to find any mistakes or things that don't work as expected.
    - ❖ They run different tests to check if everything functions smoothly.

5. **Deployment:**
    - ❖ Once the software is tested and working well, it's ready to be released to users.
    - ❖ It's like launching a new product into the world.
    - ❖ Developers may also provide training and support to help users understand and use the software effectively.

6. **Maintenance:**
    - ❖ Even after the software is released, the work isn't over.
    - ❖ Developers continue to monitor the software for any issues that may arise.
    - ❖ They may also make updates or improvements based on feedback from users or changes in technology.

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.

- ✓ **Introduction:**

  APNA Bank wanted to make banking easier for its customers, so they decided to create a mobile app. They followed a step-by-step process to make sure the app worked well and was safe to use.

1. **Requirement Gathering:**

   First, they talked to a lot of people to find out what they needed in the app. This included bank bosses, tech experts, and everyday customers. They figured out what features were most important, like checking balances, sending money, paying bills, and keeping everything safe.

2. **Design:**

   Next, they drew pictures and made plans for how the app would look and work. They made sure it would be easy to use and could handle lots of people using it at once. They also made sure it would be safe from hackers.

3. **<u>Implementation:</u>**
   Then, they started building the app. They wrote the computer code that made the app do what they wanted. They did this step by step, fixing problems as they went along. They used a method called Agile, which means they made small improvements often.

4. **<u>Testing:</u>**
   They checked the app a lot to make sure it worked right. They tested it in different ways, like seeing if it worked on different phones or if it could handle many people using it at once. They also checked to make sure it was safe from hackers

5. **<u>Deployment:</u>**
   Once they were sure the app worked well, they put it out for people to use. They did this carefully to make sure it didn't cause any problems for the bank or the customers. They also taught people how to use it.

6. **<u>Maintenance:</u>**
   Even after the app was out, they kept working on it. They fixed any problems that came up and made it better based on what people said. They also kept it safe from hackers by updating it regularly.

- ✓ **Project Outcomes:**
  Because they followed all these steps carefully, the app was a big hit. People liked how easy it was to use, and it made banking faster and safer. This helped the bank stay ahead of the competition and attract more customers.

- ✓ **Conclusion:**
  By taking their time and following each step, APNA Bank made a great app that made banking easier for everyone. This shows that when you plan things well and work hard, you can make something awesome that people love to use.

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.

1. **Waterfall Model:**
   Advantages:
   - ➢ Simple and easy to understand.
   - ➢ It follows a step-by-step process (like making a recipe), which makes it easy to measure progress.

- Works well when the project requirements are clear and won't change much.
- Each step has specific goals, so it's easy to know when each part is done.

Disadvantages:

- It's not very flexible; once you start, it's hard to change things.
- Customers don't get much say until the end, so they might not like the final product.
- If the initial plan is wrong, the whole project can fail.
- Testing happens at the end, so it's hard to catch and fix problems early.

2. **Agile Model:**

Advantages:

- It's flexible and can change as needed.
- Customers are involved all along, so they get what they want.
- It builds the project in small steps, so you can test and fix things early.
- It focuses on teamwork and communication.

Disadvantages:

- It needs the customer's input a lot, which can be hard to manage.
- There might not be much documentation, so it's harder for new team members to join or for future maintenance.
- It might not work well for projects with strict rules or laws.
- It's hard to predict how long or how much it will cost since things can change.

3. **Spiral Model:**

Advantages:

- It focuses on finding and fixing problems early, like testing the waters before diving in.
- It can change and improve based on feedback from each step.
- It's good for projects with lots of unknowns or big risks.
- It mixes parts of both the Waterfall and Agile methods.

Disadvantages:

- It's complex and needs careful planning and management.
- If risks aren't managed well, it can take a long time.
- It needs a skilled team to work through each step.
- It might not be the best for simple projects.

4. **Spiral Model:**

Advantages:

- It focuses on finding and fixing problems early, like testing the waters before diving in.
- It can change and improve based on feedback from each step.
- It's good for projects with lots of unknowns or big risks.
- It mixes parts of both the Waterfall and Agile methods.

Disadvantages:

- It's complex and needs careful planning and management.
- If risks aren't managed well, it can take a long time.
- It needs a skilled team to work through each step.
- It might not be the best for simple projects.

✓ Use of SDLC model

Each SDLC model has its own set of advantages and disadvantages, and the choice of model depends on factors such as project requirements, complexity, risk tolerance, and stakeholder preferences. By understanding the characteristics of each model, project teams can select the most appropriate approach to ensure successful project outcomes