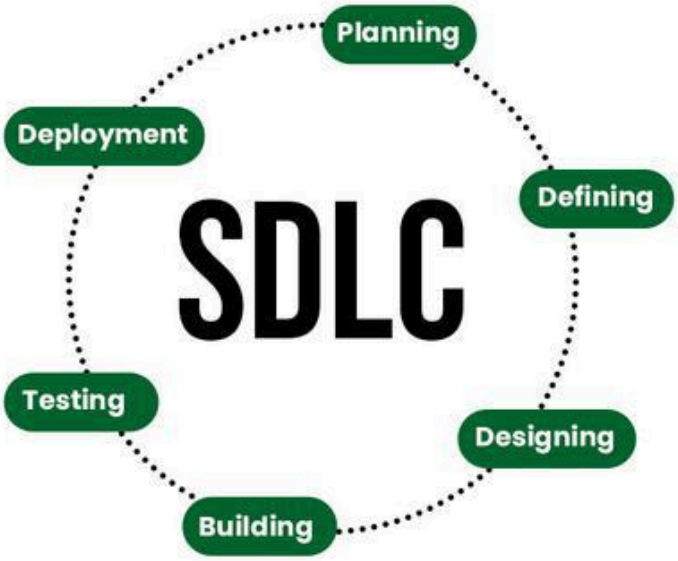
Software Development Life Cycle (SDLC)

Last Updated : 25 Mar, 2025

Software development life cycle (SDLC) is a structured process that is used to design, develop, and test good-quality software. SDLC, or software development life cycle, is a methodology that defines the entire procedure of software development step-by-step. The **goal of the SDLC life cycle model** is to deliver high-quality, maintainable software that meets the user's requirements. SDLC in software engineering models outlines the plan for each stage so that each stage of the software development model can perform its task efficiently to deliver the software at a low cost within a given time frame that meets users requirements. In this article we will see Software Development Life Cycle (SDLC) in detail.

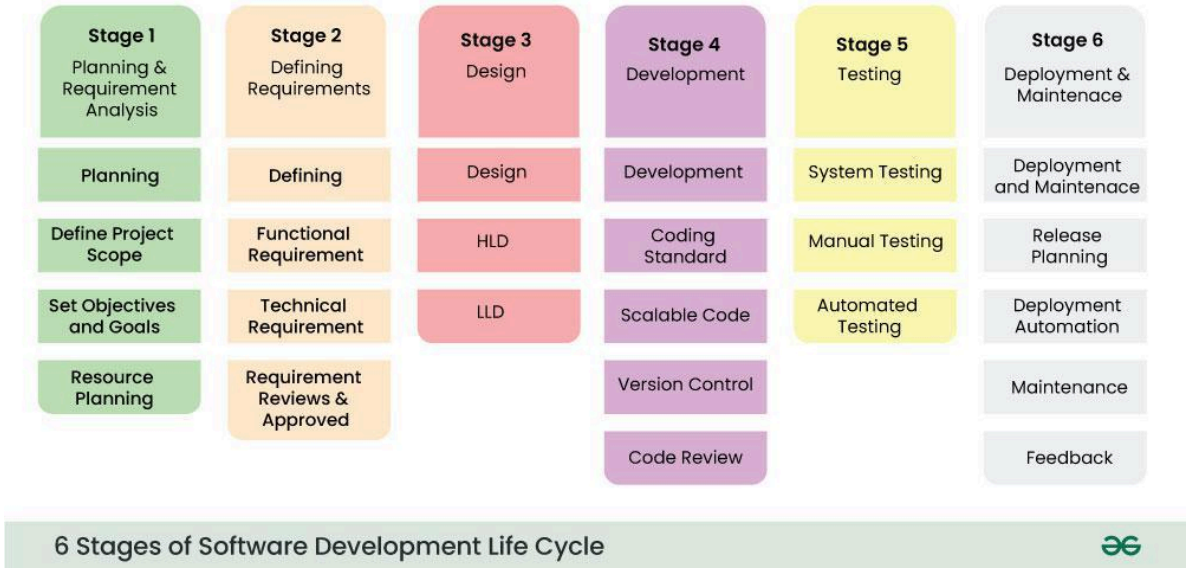
What is the Software Development Life Cycle (SDLC)?

SDLC is a process followed for software building within a software organization. SDLC consists of a precise plan that describes how to develop, maintain, replace, and enhance specific software. The life cycle defines a method for improving the quality of software and the all-around development process.



Stages of the Software Development Life Cycle

SDLC specifies the task(s) to be performed at various stages by a software engineer or developer. It ensures that the end product is able to meet the customer's expectations and fits within the overall budget. Hence, it's vital for a software developer to have prior knowledge of this software development process. SDLC is a collection of these six stages, and the stages of SDLC are as follows:



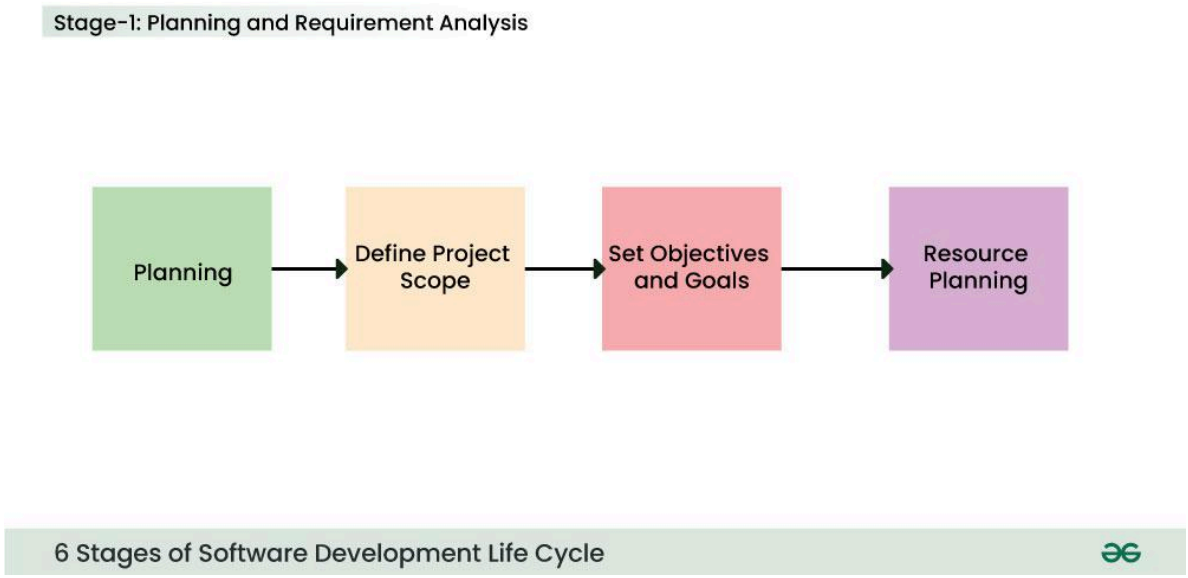
Software Development Life Cycle Model SDLC Stages

The **SDLC Model involves six phases or stages** while developing any software.

Stage-1: Planning and Requirement Analysis

Planning is a crucial step in everything, just as in software development. In this same stage, requirement analysis is also performed by the developers of the organization. This is attained from customer inputs, and sales department/market surveys.

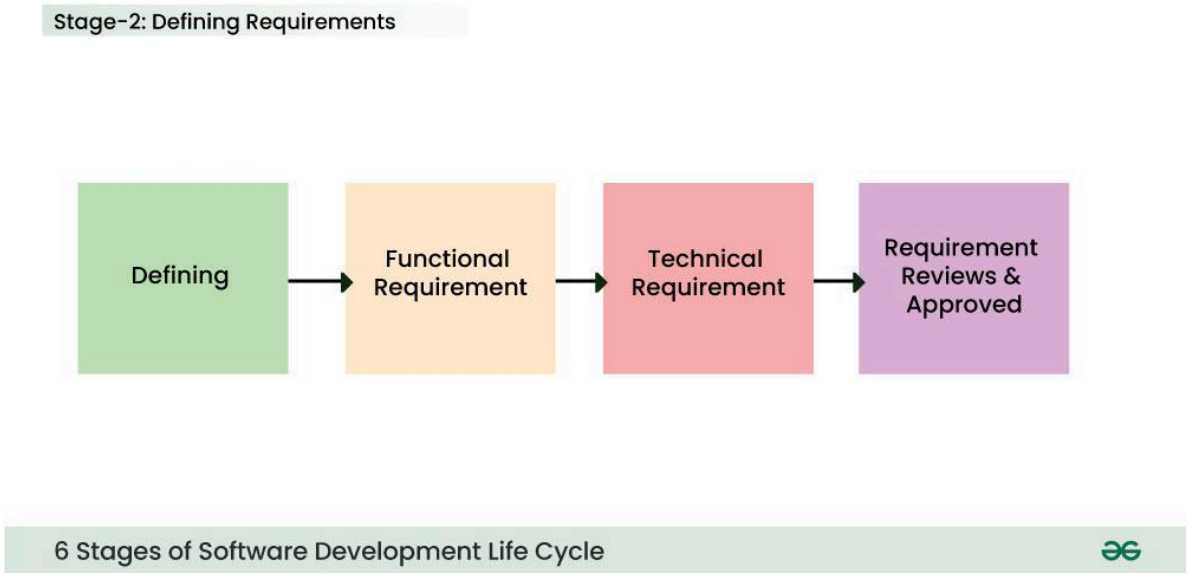
The information from this analysis forms the building blocks of a basic project. The quality of the project is a result of planning. Thus, in this stage, the basic project is designed with all the available information.



Stage-1 : Planning and Requirement Analysis

Stage-2: Defining Requirements

In this stage, all the requirements for the target software are specified. These requirements get approval from customers, market analysts, and stakeholders.
This is fulfilled by utilizing SRS (Software Requirement Specification). This is a sort of document that specifies all those things that need to be defined and created during the entire project cycle.

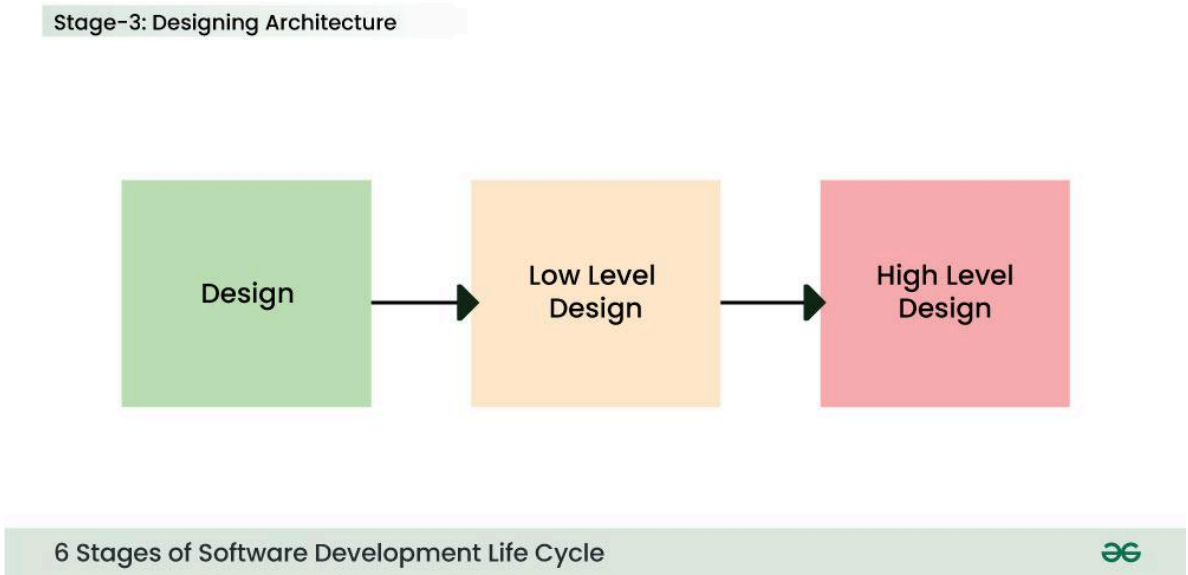


Stage-2 : Defining Requirements

Stage-3: Designing Architecture

SRS is a reference for software designers to come up with the best architecture for the software. Hence, with the requirements defined in SRS, multiple designs for the product architecture are present in the Design Document Specification (DDS).

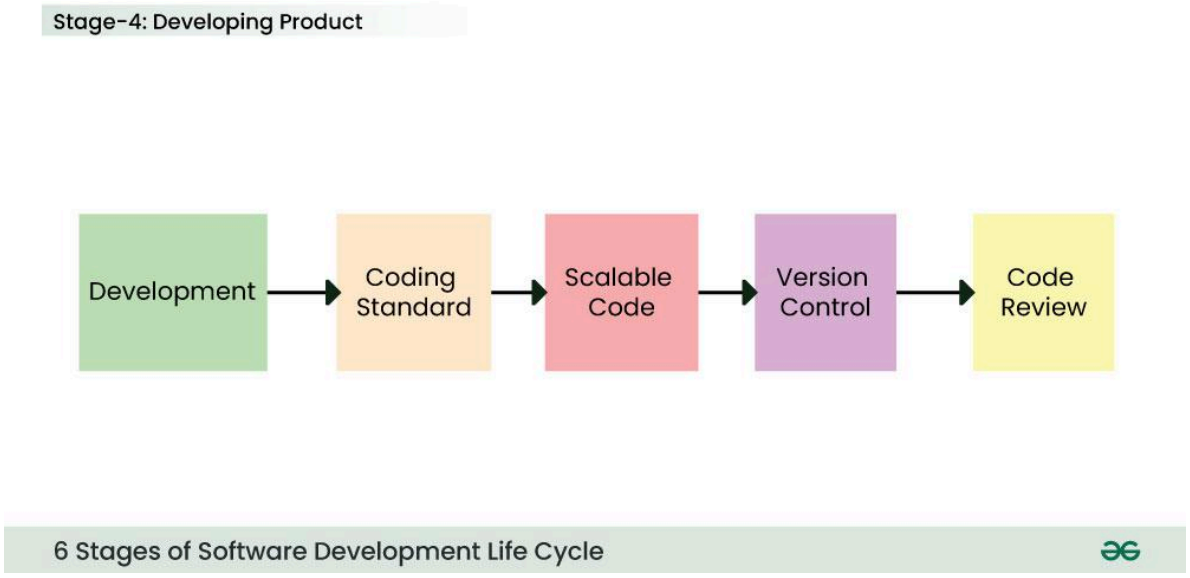
This DDS is assessed by market analysts and stakeholders. After evaluating all the possible factors, the most practical and logical design is chosen for development.



Stage 3: Design

Stage-4: Developing Product

At this stage, the fundamental development of the product starts. For this, developers use a specific programming code as per the design in the DDS. Hence, it is important for the coders to follow the protocols set by the association. Conventional programming tools like compilers, interpreters, debuggers, etc. are also put into use at this stage. Some popular languages like C/C++, Python, Java, etc. are put into use as per the software regulations.

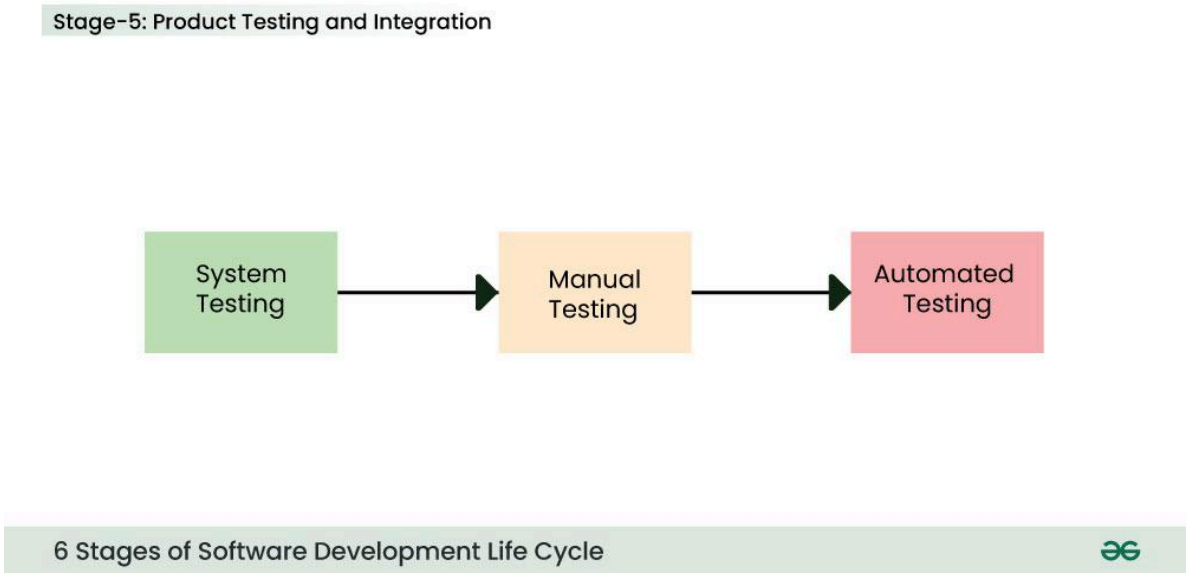


Stage 4: Development

Stage-5: Product Testing and Integration

After the development of the product, testing of the software is necessary to ensure its smooth execution. Although, minimal testing is conducted at every stage of SDLC. Therefore, at this stage, all the probable flaws are tracked, fixed, and retested. This ensures that the product confronts the quality requirements of SRS.

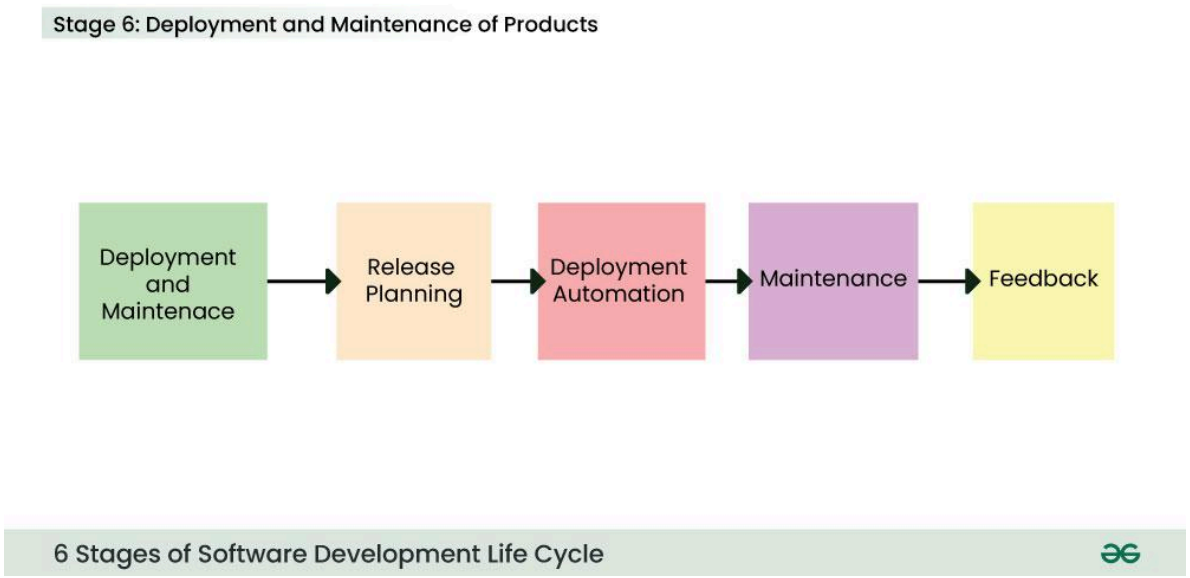
Documentation, Training, and Support: Software documentation is an essential part of the software development life cycle. A well-written document acts as a tool and means to information repository necessary to know about software processes, functions, and maintenance. Documentation also provides information about how to use the product. Training in an attempt to improve the current or future employee performance by increasing an employee's ability to work through learning, usually by changing his attitude and developing his skills and understanding.



Stage 5: Testing

Stage-6: Deployment and Maintenance of Products

After detailed testing, the conclusive product is released in phases as per the organization's strategy. Then it is tested in a real industrial environment. It is important to ensure its smooth performance. If it performs well, the organization sends out the product as a whole. After retrieving beneficial feedback, the company releases it as it is or with auxiliary improvements to make it further helpful for the customers. However, this alone is not enough. Therefore, along with the deployment, the product's supervision.



Stage 6: Deployment and Maintenance

Software Development Life Cycle Models

To this day, we have **more than 50** recognized **SDLC models** in use. But **None of them is perfect**, and each brings its favorable aspects and disadvantages for a specific software development project or a team.

Here, we have listed the **top five most popular SDLC models**:

1. Waterfall Model

It is the fundamental model of the software development life cycle. This is a very simple model. The **waterfall model** is not in practice anymore, but it is the basis for all other SDLC models. Because of its simple structure, the waterfall model is easier to use and provides a tangible output. In the waterfall model, once a phase seems to be completed, it cannot be changed, and due to this less flexible nature, the waterfall model is not in practice anymore.

2. Agile Model

The agile model in SDLC was mainly designed to adapt to changing requests quickly. The main goal of the **Agile model** is to facilitate quick project completion. The agile model refers to a group of development processes. These processes have some similar characteristics but also possess certain subtle differences among themselves.

3. Iterative Model

In the **Iterative model in SDLC**, each cycle results in a semi-developed but deployable version; with each cycle, some requirements are added to the software, and the final cycle results in the software with the complete requirement specification.

4. Spiral Model

The spiral model in SDLC is one of the most crucial SDLC models that provides support for risk handling. It has various spirals in its diagrammatic representation; the number of spirals depends upon the

type of project. Each loop in the spiral structure indicates the Phases of the **Spiral model**.

5. V-Shaped Model

The **V-shaped model in SDLC** is executed in a sequential manner in V-shape. Each stage or phase of this model is integrated with a testing phase. After every development phase, a testing phase is associated with it, and the next phase will start once the previous phase is completed, i.e., development & testing. It is also known as the verification or validation model.

6. Big Bang Model

The Big Bang model in SDLC is a term used to describe an informal and unstructured approach to software development, where there is no specific planning, documentation, or well-defined phases.

What is the need for SDLC?

SDLC is a method, approach, or process that is followed by a software development organization while developing any software. SDLC models were introduced to follow a disciplined and systematic method while designing software. With the software development life cycle, the process of software design is divided into small parts, which makes the problem more understandable and easier to solve. SDLC comprises a detailed description or step-by-step plan for designing, developing, testing, and maintaining the software.

> *Follow the project **Library Management System** or **E Portfolio Website** to see the use of Software Development Life Cycle in a Software Projects.*

How does SDLC Address Security?

A frequent issue in software development is the delay of security-related tasks until the testing phase, which occurs late in the software development life cycle (SDLC) and occurs after the majority of crucial design and implementation has been finished. During the testing phase,

security checks may be minimal and restricted to scanning and penetration testing, which may fail to identify more complicated security flaws.

Security issue can be address in SDLC by following DevOps. Security is integrated throughout the whole SDLC, from build to production, through the use of DevSecOps. Everyone involved in the DevOps value chain have responsibility for security under DevSecOps.

Real Life Example of SDLC

Developing a banking application using SDLC:

- **Planning and Analysis:** During this stage, business stakeholders' requirements about the functionality and features of banking application will be gathered by program managers and business analysts. Detailed SRS (Software Requirement Specification) documentation will be produced by them. Together with business stakeholders, business analysts will analyse and approve the SRS document.
- **Design:** Developers will receive SRS documentation. Developers will read over the documentation and comprehend the specifications. Web pages will be designed by designers. High level system architecture will be prepared by developers.
- **Development:** During this stage, development will code. They will create the web pages and APIs needed to put the feature into practice.
- **Testing:** Comprehensive functional testing will be carried out. They will guarantee that the banking platform is glitch-free and operating properly.
- **Deployment and Maintenance:** The code will be made available to customers and deployed. Following this deployment, the customer can access the online banking. The same methodology will be used to create any additional features.

How to Choose an SDLC Model?

Choosing the right SDLC (Software Development Life Cycle) model is essential for project success. Here are the key factors to consider:

1. **Project Requirements:**
   - **Clear Requirements:** Use **Waterfall** or **V-Model** if requirements are well-defined and unlikely to change.
   - **Changing Requirements:** Use **Agile** or **Iterative** models if requirements are unclear or likely to evolve.

2. **Project Size and Complexity:**
   - **Small Projects:** Use **Waterfall** or **RAD** for small, simple projects.
   - **Large Projects:** Use **Agile**, **Spiral**, or **DevOps** for large, complex projects that need flexibility.

3. **Team Expertise:**
   - **Experienced Teams:** Use **Agile** or **Scrum** if the team is familiar with iterative development.
   - **Less Experienced Teams:** Use **Waterfall** or **V-Model** for teams needing structured guidance.

4. **Client Involvement:**
   - **Frequent Client Feedback:** Use **Agile**, **Scrum**, or **RAD** if regular client interaction is needed.
   - **Minimal Client Involvement:** Use **Waterfall** or **V-Model** if client involvement is low after initial planning.

5. **Time and Budget Constraints:**
   - **Fixed Time and Budget:** Use **Waterfall** or **V-Model** if you have strict time and budget limits.
   - **Flexible Time and Budget:** Use **Agile** or **Spiral** if you can adjust time and budget as needed.

6. **Risk Management:**
   - **High-Risk Projects:** Use **Spiral** for projects with significant risks and uncertainties.
   - **Low-Risk Projects:** Use **Waterfall** for projects with minimal risks.

7. **Product Release Timeline:**
   - **Quick Release Needed:** Use **Agile** or **RAD** to deliver products quickly.

- **Longer Development Time:** Use **Waterfall** or **V-Model** for projects with no urgent deadlines.

8. **Maintenance and Support:**
   - **Long-Term Maintenance:** Use **Agile** or **DevOps** for projects needing continuous updates and support.
   - **Minimal Maintenance:** Use **Waterfall** or **V-Model** if little future maintenance is expected.

9. **Stakeholder Expectations:**
   - **High Stakeholder Engagement:** Use **Agile** or **Scrum** if stakeholders want ongoing involvement.
   - **Low Stakeholder Engagement:** Use **Waterfall** or **V-Model** if stakeholders prefer involvement only at major milestones.

Note:

- ***Waterfall****: Best for clear, stable projects with minimal changes.*
- ***V-Model****: Good for projects with clear requirements and a strong focus on testing.*
- ***Agile/Scrum****: Ideal for projects with changing requirements and frequent client interaction.*
- ***Spiral****: Suitable for high-risk projects with evolving requirements.*
- ***RAD****: Useful for projects needing rapid development.*
- ***DevOps****: Best for continuous integration and ongoing support*

Conclusion

In conclusion, we now know that the **Software Development Life Cycle (SDLC) in software engineering is an important framework for the better and more structured development of optimized software programs.** In a world full of rapid evolution in technology, SDLC phases plays a crucial role in enabling some good and innovative solutions for helping users and organizations. Also, it's better to adapt SDLC principles to achieve software development goals effectively.

Important Questions on Software Development Life Cycle (SDLC)

1. Which of the following is not a life cycle model?

(A) Spiral model

(B) Prototyping model

(C) Waterfall model

(D) Capability maturity model

Solution: *The correct Answer is* ***(D)****.*

2. What is the appropriate pairing of items in the two columns listing various activities encountered in a software life cycle?

P. Requirements Capture	**2.** Domain Analysis
Q. Design	**3.** Structural and Behavioral Modeling
R. Implementation	**1.** Module Development and Integration
S. Maintenance	**4.** Performance Tuning

(A) P-3, Q-2, R-4, S-1

(B) P-2, Q-3, R-1, S-4

(C) P-3, Q-2, R-1, S-4

(D) P-2, Q-3, R-4, S-1

Solution: *The correct Answer is* ***(B)****.*

Comment More info

Next Article