**EXERCISE-1**

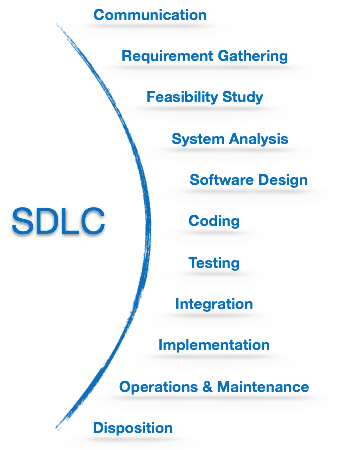
**Get an understanding of the stages in software development lifecycle, the process models, values and principles of agility and the need for agile software development. This will enable you to work in projects following an agile approach to software development**.

**Software:**

Software is a set of instructions, data or programs used to operate computers and execute specific tasks.

**Software Development Life Cycle:**

Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality software. The SDLC aims to produce a high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.



**Waterfall Model:**

The Waterfall Model was the first Process Model to be introduced. It is also referred to as a **linear-sequential life cycle model**. It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases.

The Waterfall model is the earliest SDLC approach that was used for software development.

The waterfall Model illustrates the software development process in a linear sequential flow. This means that any phase in the development process begins only if the previous phase is complete. In this waterfall model, the phases do not overlap.

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.

The following illustration is a representation of the different phases of the Waterfall Model.



## Waterfall Model – Application:

Every software developed is different and requires a suitable SDLC approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are

* Requirements are very well documented, clear and fixed.
* Product definition is stable.
* Technology is understood and is not dynamic.
* There are no ambiguous requirements.
* Ample resources with required expertise are available to support the product.
* The project is short.

**Advantages of Waterfall Model:**

The advantages of waterfall development are that it allows for departmentalization and control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one.

Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order.

Some of the major advantages of the Waterfall Model are as follows −

* Simple and easy to understand and use
* Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
* Phases are processed and completed one at a time.
* Works well for smaller projects where requirements are very well understood.
* Clearly defined stages.
* Well understood milestones.
* Easy to arrange tasks.
* Process and results are well documented.

AD

**Disadvantages of Waterfall Model:**

The disadvantage of waterfall development is that it does not allow much reflection or revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-documented or thought upon in the concept stage.

The major disadvantages of the Waterfall Model are as follows −

* No working software is produced until late during the life cycle.
* High amounts of risk and uncertainty.
* Not a good model for complex and object-oriented projects.
* Poor model for long and ongoing projects.
* Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
* It is difficult to measure progress within stages.
* Cannot accommodate changing requirements.
* Adjusting scope during the life cycle can end a project.
* Integration is done as a "big-bang. at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.

**EXERCISE-2**

**Get a working knowledge of using extreme automation through XP programming practices of test first development, refactoring and automating test case writing**.

**Agile Model**

Agile model believes that every project needs to be handled differently and the existing methods need to be tailored to best suit the project requirements. In Agile, the tasks are divided to time boxes (small time frames) to deliver specific features for a release.

Iterative approach is taken and working software build is delivered after each iteration. Each build is incremental in terms of features; the final build holds all the features required by the customer.

**Advantages of Agile Model:**

* Is a very realistic approach to software development.
* Promotes teamwork and cross training.
* Functionality can be developed rapidly and demonstrated.
* Resource requirements are minimum.
* Suitable for fixed or changing requirements
* Delivers early partial working solutions.
* Good model for environments that change steadily.
* Minimal rules, documentation easily employed.
* Enables concurrent development and delivery within an overall planned context.
* Little or no planning required.
* Easy to manage.
* Gives flexibility to developers.

**Disadvantages of Agile Model:**

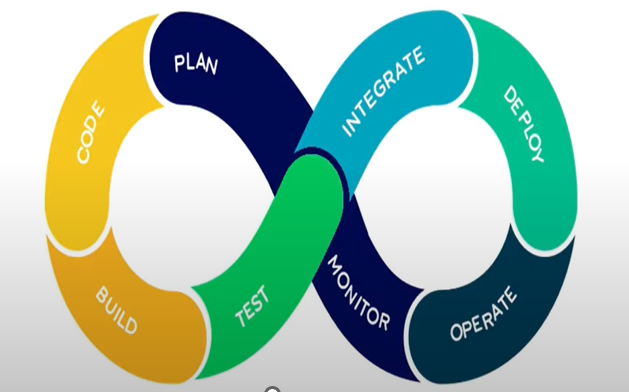
* Not suitable for handling complex dependencies.
* More risk of sustainability, maintainability and extensibility.
* An overall plan, an agile leader and agile PM practice is a must without which it will not work.
* Strict delivery management dictates the scope, functionality to be delivered, and adjustments to meet the deadlines.
* Depends heavily on customer interaction, so if customer is not clear, team can be driven in the wrong direction.
* There is a very high individual dependency, since there is minimum documentation generated.
* Transfer of technology to new team members may be quite challenging due to lack of documentation.

**EXERCISE-3**

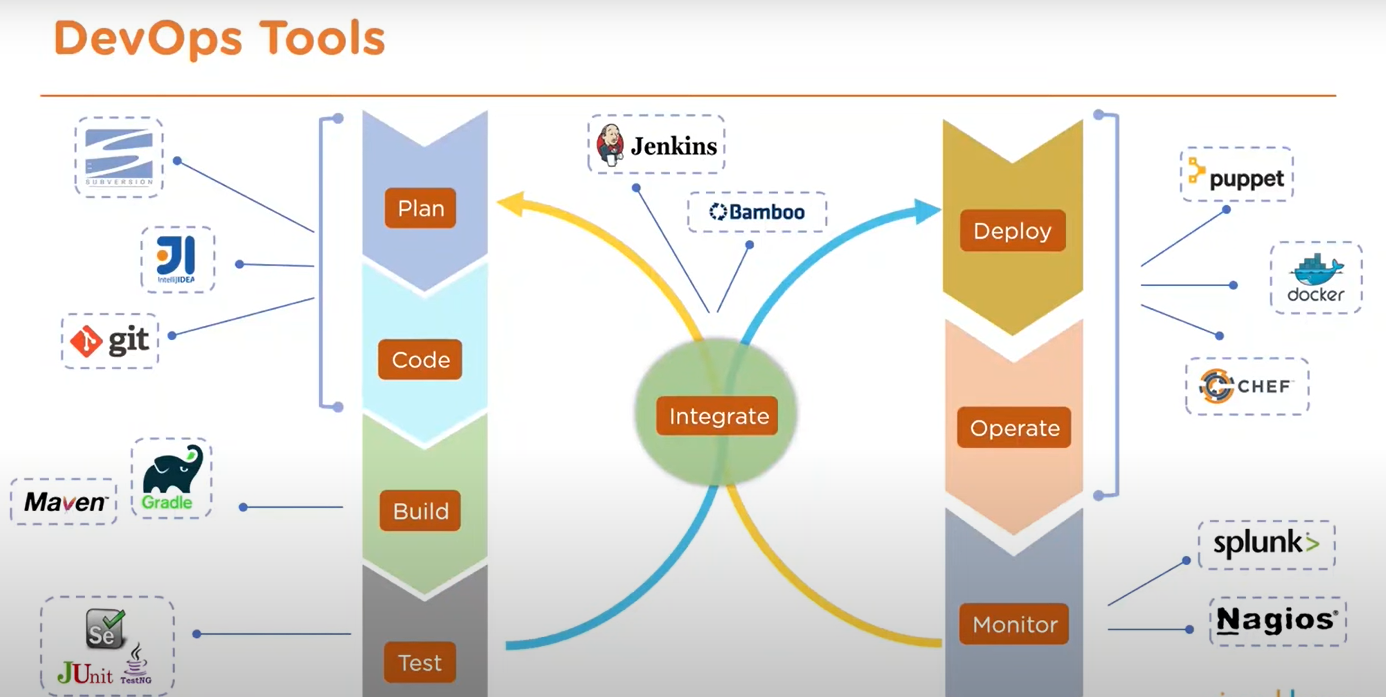
**It is important to comprehend the need to automate the software development lifecycle stages through DevOps. Gain an understanding of the capabilities required to implement DevOps, continuous integration and continuous delivery practices.**

**What is DevOps?**

The DevOps is a combination of two words, one is software Development, and second is Operations. This allows a single team to handle the entire application lifecycle, from development to **testing, deployment**, and **operations**. DevOps helps you to reduce the disconnection between software developers, quality assurance (QA) engineers, and system administrators. DevOps promotes collaboration between Development and Operations team to deploy code to production faster in an automated & repeatable way. DevOps helps to increase organization speed to deliver applications and services. It also allows organizations to serve their customers better and compete more strongly in the market. DevOps can also be defined as a sequence of development and IT operations with better communication and collaboration. DevOps has become one of the most valuable business disciplines for enterprises or organizations. With the help of DevOps, **quality**, and **speed** of the application delivery has improved to a great extent.



* In plan stage, business owners and software development team discuss project goals and create a plan.
* Programmers then design and code the application and use tools like Git store application code.
* Build tools like Maven and Gradle, take code from different repositories and combine them to build the complete application.
* Application is tested using automation testing tools like Selenium and Junit to ensure software quality.
* When testing is compile, new features are integrated automatically to the already existing codebase.
* Application is packaged after release and deployed from development server to production server.
* Once software is deployed, operations team perform activities such as configuring servers and provisioning them with the required resources.
* Once software is deployed, operations team perform activities such as configuring servers and provisioning them with the required resources.



**Git:** **Git** is an **open-source distributed version control system**. It is designed to handle minor to major projects with high speed and efficiency. It is developed to co-ordinate the work among the developers. The version control allows us to track and work together with our team members at the same workspace.

**Maven:** Maven, a [Yiddish word](https://en.wikipedia.org/wiki/Maven) meaning accumulator of knowledge, began as an attempt to simplify the build processes in the Jakarta Turbine project. There were several projects, each with their own Ant build files, that were all slightly different. JARs were checked into CVS. We wanted a standard way to build the projects, a clear definition of what the project consisted of, an easy way to publish project information, and a way to share JARs across several projects.

**Jenkins:** Jenkins Tutorial is designed for both beginners and professionals. Our Tutorial provides all the basic and advanced concepts of Jenkins, such as Jenkins installation, Jenkins Configuration, Jenkins Pipeline, etc. Jenkins is an open source automation tool written in Java programming language that allows continuous integration. Jenkins builds and tests our software projects, which continuously making it easier for developers to integrate changes to the project, and making it easier for users to obtain a fresh build.

**Advantages:**

* DevOps is an excellent approach for quick development and deployment of applications.
* It responds faster to the market changes to improve business growth.
* DevOps escalate business profit by decreasing software delivery time and transportation costs.
* DevOps clears the descriptive process, which gives clarity on product development and delivery.
* It improves customer experience and satisfaction.
* DevOps simplifies collaboration and places all tools in the cloud for customers to access.
* DevOps means collective responsibility, which leads to better team engagement and productivity.

**Disadvantages:**

* DevOps professional or expert's developers are less available.
* Developing with DevOps is so expensive.
* Adopting new DevOps technology into the industries is hard to manage in short time.
* Lack of DevOps knowledge can be a problem in the continuous integration of automation projects.

