Software: **Software** consists of computer programs that instruct the execution of a computer.

SDLC - Software Development Life Cycle

⇒ The structured process/ phases of the development process to achieve better results in an organised manner.

⇒ It gives a shape/ a process to the software so that it evolves from time to time.

PHASES:

⇒ Requirements: Understanding all the requirements required for the product / the user.

⇒ Design: Designing the architecture and the planning of the software.

⇒ Implementation: Development of the software.

⇒ Verification: Tests the developed software.

⇒ Maintenance: Deploying the software and taking care of the user requirements.

TYPES:

⇒ Waterfall model

⇒ Iterative model

⇒ Prototype model

⇒ Spiral model

⇒ V model

⇒ Agile model

WATERFALL MODEL:

⇒ In this model, we can enter the next phase only after the completion of the earlier phase. We can’t go back to earlier phases even if needed.

⇒ Easy to implement, but it is not suitable for projects which have changing requirements.

⇒ The Requirements and analysis is the longest phase in this model.

ITERATIVE MODEL:

⇒ In this model, we go through one cycle of development and if there are extra requirements we go to the next cycle to meet the requirement.

⇒ We go through a number of cycles until we meet all the requirements.

PROTOTYPE MODEL:

⇒ In this model, we usually build a prototype of the project just to have an idea of how the project should run.

⇒ After building the prototype we usually go through the phases of SDLC and develop the product.

SPIRAL MODEL:

⇒ Spiral model works exactly how Iterative model works. ⇒ The only difference will be that Spiral model projects will be long term projects and will have a very long-time requirement change.

V MODEL:

⇒ In this model, testing is done for each and every phase so that the results are accurate.

⇒ But it can be a hassle to test the product every time.

AGILE MODEL:

⇒ Agile is the most used SDLC model these days because of its ease and flexibility to the changes.

⇒ In an agile model, first a model is developed based on the initial requirements and will be released and maintained as well.

⇒ If there are any other requirements needed the developed product is developed further to meet the requirements and the updated version will be released and maintained.

⇒ This way the release of the product won’t be delayed and also the requirements will be satisfied.

SCRUM:

⇒ Scrum is the complete team coming together to discuss their progress and give insights on each other’s work.

⇒ Scrum enhances the Retrospective view of the project, and it will lead to better results.

GIT:

⇒ Git is the version control which helps multiple developers to work simultaneously.

⇒ It helps in saving the complete version history of the code and also saves the past version of the code to be safe from the bugs of the latest version. Only after the complete testing we can change the version.

* **ls -lrt:** will list files and directories in the long format, sorted by modification time with the oldest files shown first.
* **cd:** change directory command.
* **git branch <branch\_name>:** to create a new branch.
* **git checkout <branch\_name>:** to change into that given branch.
* **git status:** gives the status of the files present in the branch.
* **git branch -a:** lists out all the branches present in the repository.
* **git branch -d <branch\_name>:** to delete the given branch.
* **git add <file\_name>:** stages the given file for commit.
* **git commit -m “some\_message”:** commits the staged files along with a message to have a clear understanding of the commit.
* **git push:** used to push the commits into the version control.
* **git push —set-upstream origin <branch\_name>:** to push the files into a new branch from a remote branch for the first time.
* **history:** lists out all the commands used.
* **vi <filename>:** to open a file in vi editor.
* **git clone <repo\_link>:** used to clone the existing repository into the local system.

⇒ Whenever a branch is created it will be a copy of the existing branch you createdthe branch from.

⇒ The new branch will be considered and shown only when there is a change in one or other file, otherwise it is not shown.

⇒ After all the required changes are done with files, it will be recorded as a branch and then we need to pull a request to merge the branch with the main branch.

⇒ So then we need to merge the new branch with the main branch.