Day 4: - Assignment

What is Git?

Git is a popular version control system. It was created by Linus Torvalds in 2005 and has been maintained by Junio Hamano since then.

It is used for:

- Tracking code changes
- Tracking who made changes.
- Coding collaboration

What does Git do?

- Manage projects with **Repositories.**
- **Clone** a project to work on a local copy.
- Control and track changes with **Staging** and **Committing**
- **Branch** and **Merge** to allow for work on different parts and versions of a project.
- **Pull** the latest version of the project to a local copy.
- **Push** local updates to the main project.

Why Git?

- Over 70% of developers use Git!
- Developers can work together from anywhere in the world.
- Developers can see the full history of the project.
- Developers can revert to earlier versions of a project.

What is GitHub?

- Git is not the same as GitHub.
- GitHub makes tools that use Git.
- GitHub is the largest host of source code in the world and has been owned by Microsoft since 2018.

- In this tutorial, we will focus on using Git with GitHub.

What are differences between GitHub and Git?

1. GitHub requires Git, but Git doesn't need GitHub.
2. Git is managed by the Linux Foundation while GitHub is owned by Microsoft.
3. Git is open source and GitHub is proprietary.
4. Git is installed locally while GitHub is cloud-based.
5. Git is free; GitHub isn't.
6. Git's feature set is stable while GitHub offerings are expanding.
7. Git predates GitHub by three years.
8. Git and GitHub have different competitors.
9. Git has no security features while GitHub has many.
10. GitHub has a web interface while Git doesn't.

What is deployment?

In the realm of technology, "deployment" refers to the process of making a software application or system available for use. It involves transferring the developed software from a testing environment or a development environment to a live environment where users can interact with it. Deployment encompasses various tasks, including configuring servers, setting up databases, installing necessary software dependencies, and ensuring that the application runs smoothly in its production environment.

Ex: Mobile deployment, desktop application development, web development

What are the differences between google drive and git hub?

GitHub and Google Drive are both popular tools used in the context of collaboration and file management, but they serve different purposes and have distinct features. Here are some key differences between GitHub and Google Drive:

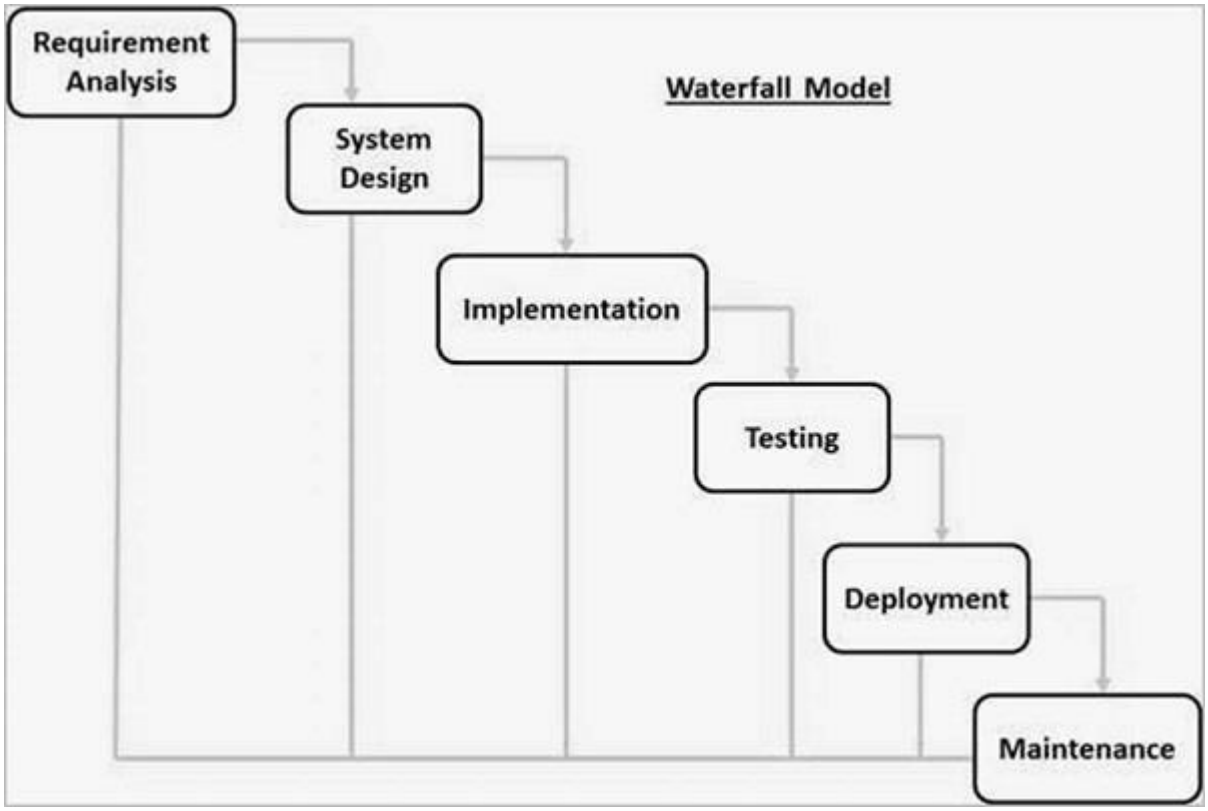
1. **Purpose**:
   - **GitHub**: GitHub is primarily a platform for version control and collaboration on software development projects. It provides features such as version history, branching, merging, and issue tracking, making it ideal for teams working on code together.
   - **Google Drive**: Google Drive is a cloud-based file storage and synchronization service that allows users to store, share, and collaborate on various types of files, including documents, spreadsheets, presentations, and more. While it can be used for collaboration, it's not specifically tailored for software development like GitHub.
2. **Version Control**:
   - **GitHub**: GitHub uses Git, a distributed version control system, to track changes to files in a project. It allows developers to create branches, commit changes, merge branches, and revert to previous versions easily.
   - **Google Drive**: Google Drive does not have built-in version control like Git. While it does keep a revision history of files, it's more basic compared to the branching and merging capabilities of Git/GitHub.
3. **Collaboration Features**:
   - **GitHub**: GitHub offers features such as pull requests, code reviews, issue tracking, and project boards, which are specifically designed to facilitate collaboration among developers and teams working on software projects.
   - **Google Drive**: Google Drive provides collaboration features like real-time editing, commenting, and sharing permissions, making it suitable for collaboration on documents, spreadsheets, and other files. However, it lacks features tailored for software development collaboration like pull requests and code reviews.
4. **File Types**:
   - **GitHub**: GitHub is primarily used for managing code files, such as source code, configuration files, documentation, etc. While it can handle other file types, its main focus is on software development-related files.
   - **Google Drive**: Google Drive supports a wide range of file types beyond just code files, including documents, spreadsheets, presentations,

images, videos, etc. It's a more general-purpose file storage and collaboration platform.

5. **Integration**:
    - **GitHub**: GitHub integrates with various third-party services and development tools, such as continuous integration/delivery (CI/CD) platforms, project management tools, code analysis tools, etc., through its extensive API and marketplace.
    - **Google Drive**: Google Drive integrates seamlessly with other Google Workspace (formerly G Suite) apps like Google Docs, Google Sheets, and Google Slides. It also offers integration with third-party apps through Google Workspace Marketplace.

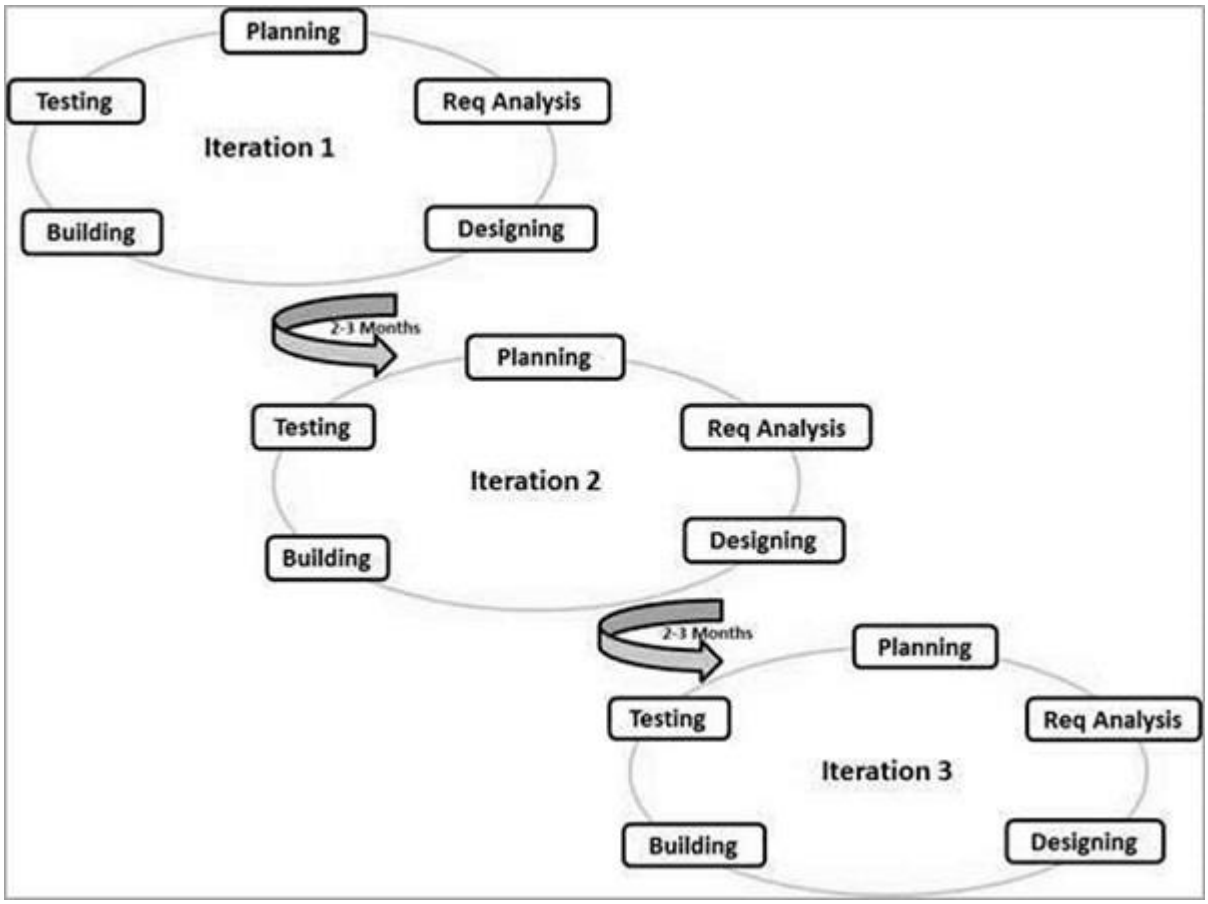
What is waterfall methodology?

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.



What is Agile methodology?

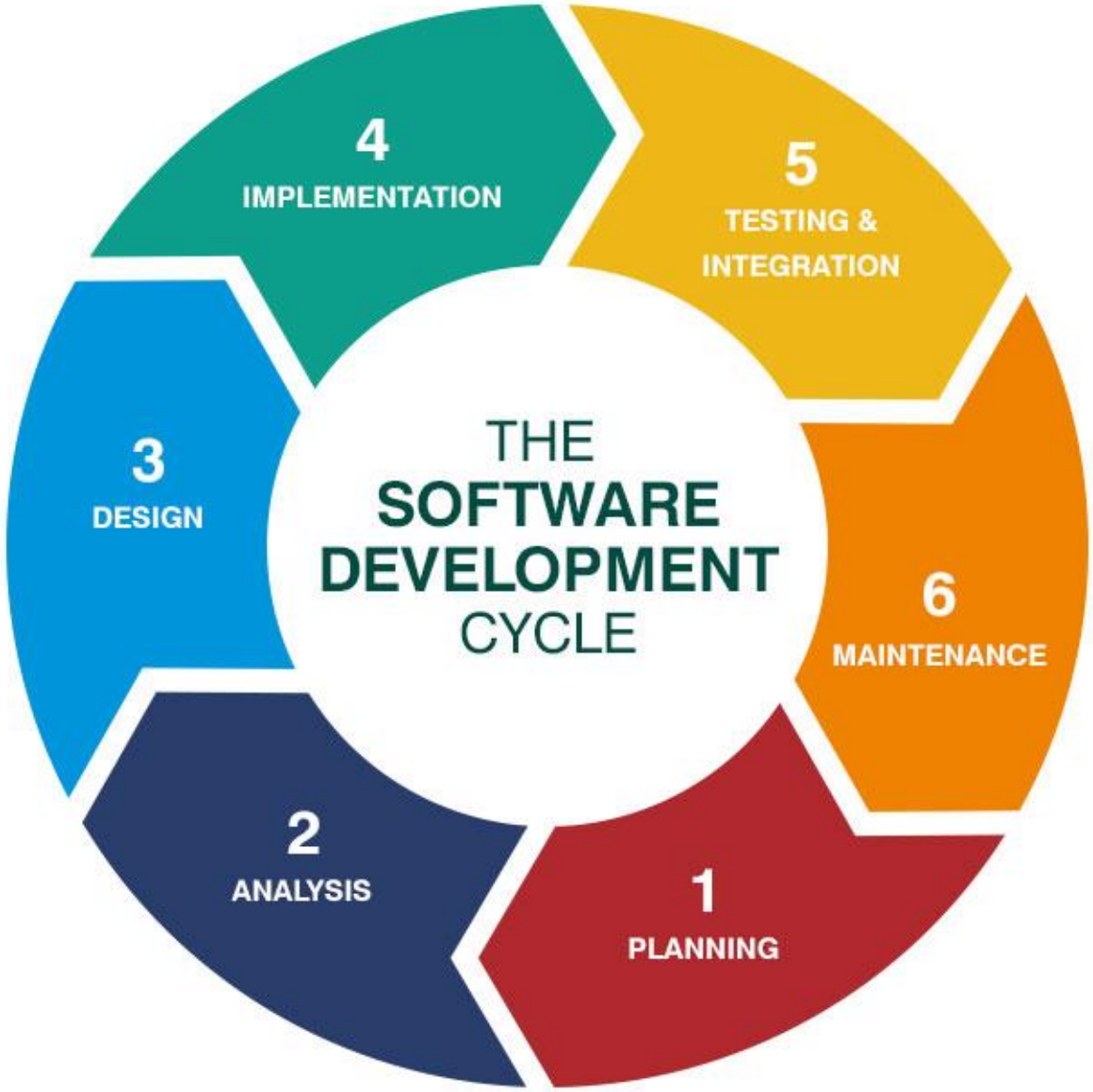
The agile model was mainly designed to adapt to changing requests quickly. The main goal of the **Agile model** is to facilitate quick project completion. The agile model refers to a group of development processes. These processes have some similar characteristics but also possess certain subtle differences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



What is SDLC?

SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace, and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.

The following figure is a graphical representation of the various stages of a typical SDLC.

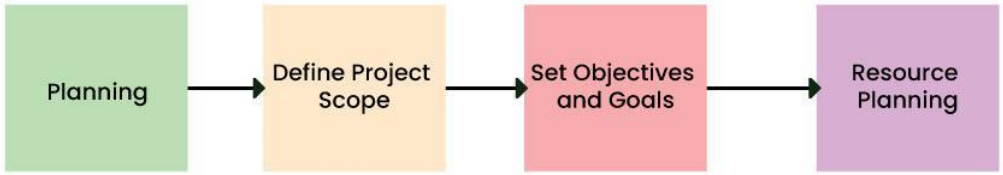


A typical Software Development Life Cycle consists of the following stages –

PLAN

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.

Stage-1: Planning and Requirement Analysis



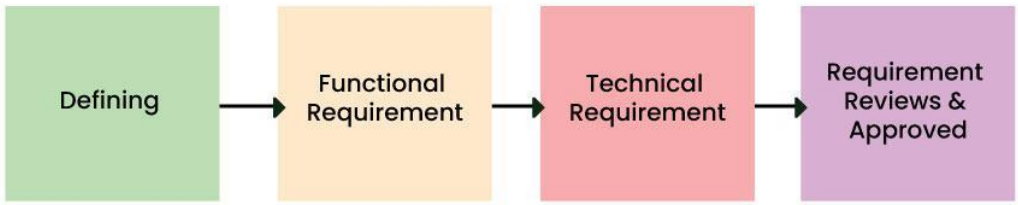
6 Stages of Software Development Life Cycle

Defining

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

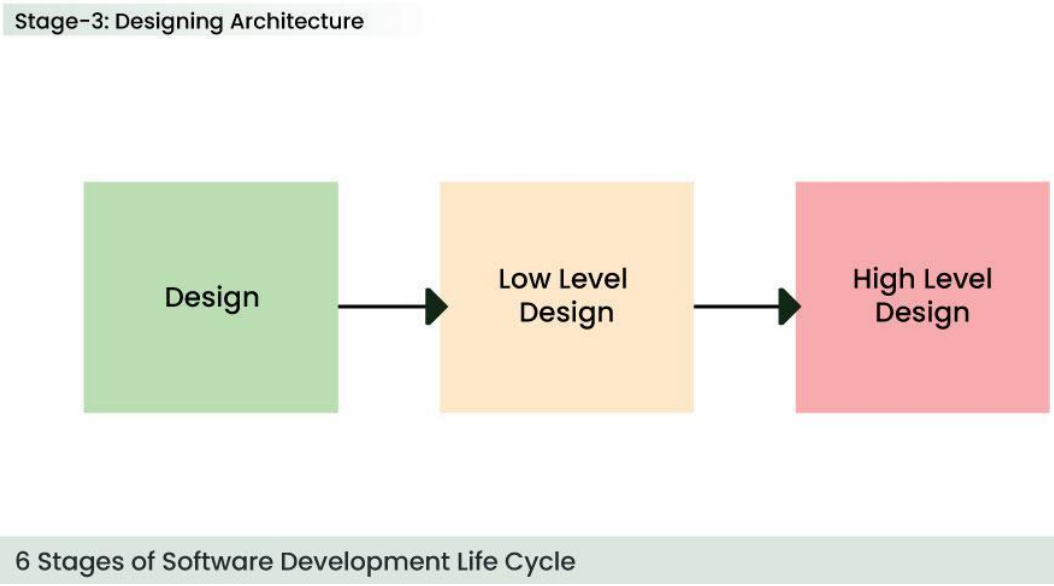


6 Stages of Software Development Life Cycle

Design

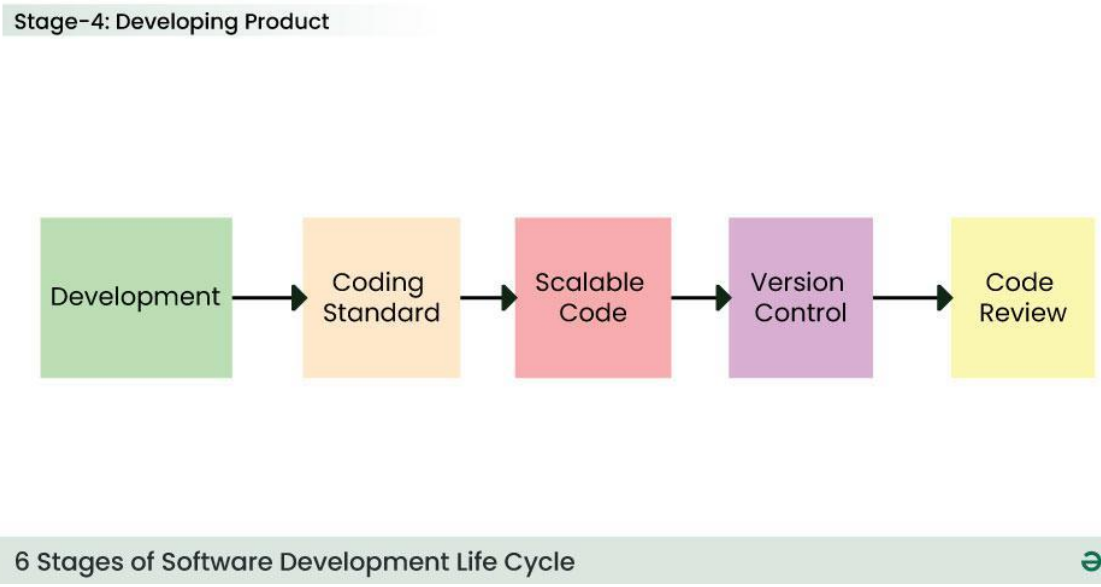
In the design phase, software engineers analyze requirements and identify the best solutions to create the software. For example, they may consider integrating pre-existing modules, making technology choices, and

identifying development tools. They will look at how to best integrate the new software into any existing IT infrastructure the organization may have.



Implement

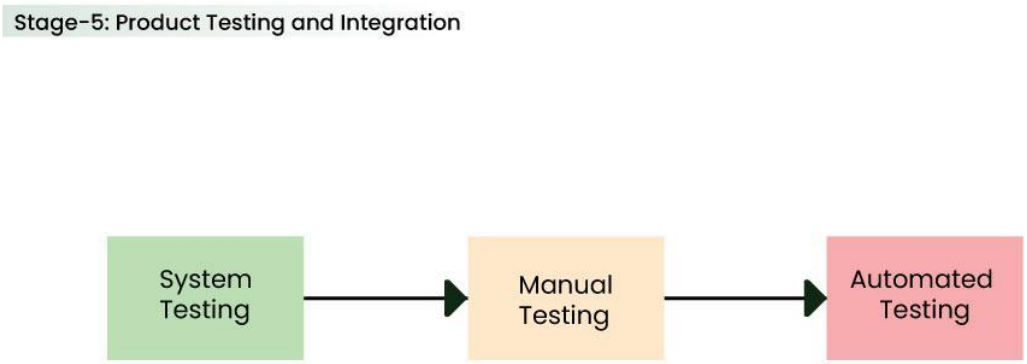
In the implementation phase, the development team codes the product. They analyze the requirements to identify smaller coding tasks they can do daily to achieve the final result.



Test

The development team combines automation and manual testing to check the software for bugs. Quality analysis includes testing the software for

errors and checking if it meets customer requirements. Because many teams immediately test the code they write, the testing phase often runs parallel to the development phase.



Maintain

In the maintenance phase, among other tasks, the team fixes bugs, resolves customer issues, and manages software changes. In addition, the team monitors overall system performance, security, and user experience to identify new ways to improve the existing software.