Git and Github

Assignment Questions

Assignment Questions

1. What is Git?
2. What do you understand by the term 'Version Control System'?
3. What is GitHub?
4. Mention some popular Git hosting services.
5. Different types of version control systems
6. What benefits come with using GIT?
7. What is a Git repository?
8. How can you initialize a repository in Git?

Ques1: What is a git ?

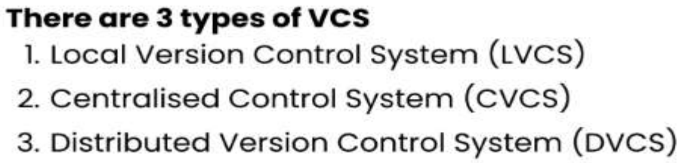
Ans 1: Git is a distributed version control system for software development that allows multiple developers to collaborate on a project. It was created by Linus Torvalds in 2005 and has become one of the most popular version control systems in use today.

Git works by allowing developers to track changes to source code, documentation, and other files in a project, creating a complete history of all changes made to the project over time. Each developer can create their own "branch" of the project, which is a separate copy of the code that they can modify without affecting the main codebase. Once changes have been made, developers can merge their branch with the main codebase, resolving any conflicts that may arise.

Ques2: what do you understand by the term Version Control System?

Ans2: Version Control System (VCS) is a software tool that helps software developers to manage changes to the source code and other artifacts that make up software development projects. The purpose of a VCS is to keep track of the history of changes made to files over time and to enable multiple developers to work on the same codebase simultaneously.

1) A VCS tracks changes to files through a series of snapshots, allowing developers to view and restore previous versions of files, track who made changes and when, and collaborate with other developers on the same codebase. This makes it easier to manage and track changes to the codebase, debug issues, and collaborate on a project.
2) There are two main types of version control systems: centralized and distributed. In a centralized VCS, all changes to files are made on a central server, while in a distributed VCS, developers work on their own local copy of the codebase and changes are synchronized across multiple copies.



There are 3 types of VCS

1. Local Version Control System (LVCS)
2. Centralised Control System (CVCS)
3. Distributed Version Control System (DVCS)

3) VCSs are widely used in software development, allowing teams to manage and track changes to large and complex software projects, collaborate effectively, and ensure that the final product is of high quality.

In localized VCS, all changes are made on the local machine and no network is required. In centralized VCS, multiple users work on the same codebase on a central server, with limited collaboration and security. In distributed VCS, each user has their own local copy of the codebase and changes are synchronized across multiple copies, allowing for high collaboration and security. Distributed VCS also offers advanced branching and merging capabilities, making it highly scalable and flexible for large and complex software projects.

Ques3: What is GitHub?

Ans3:

- GitHub is a cloud-based platform that provides a server infrastructure to host Git repositories and provides users with an easy-to-use web interface to manage those repositories or to version control those repositories using the Git system.
- It provides a platform for software developers to manage, collaborate on, and contribute to code repositories.
- GitHub allows developers to create their own Git repositories, contribute to other open-source projects, and collaborate with other developers.
- It provides a range of tools for issue tracking, code review, and project management.

- GitHub has become a key platform for open-source software development and is widely used by developers and development teams around the world.
- It also supports integration with other development tools, such as continuous integration services and project management software.
- GitHub is owned by Microsoft, which acquired the platform in 2018.

Ques4: Mention some Popular Git hosting services?

Ans4:

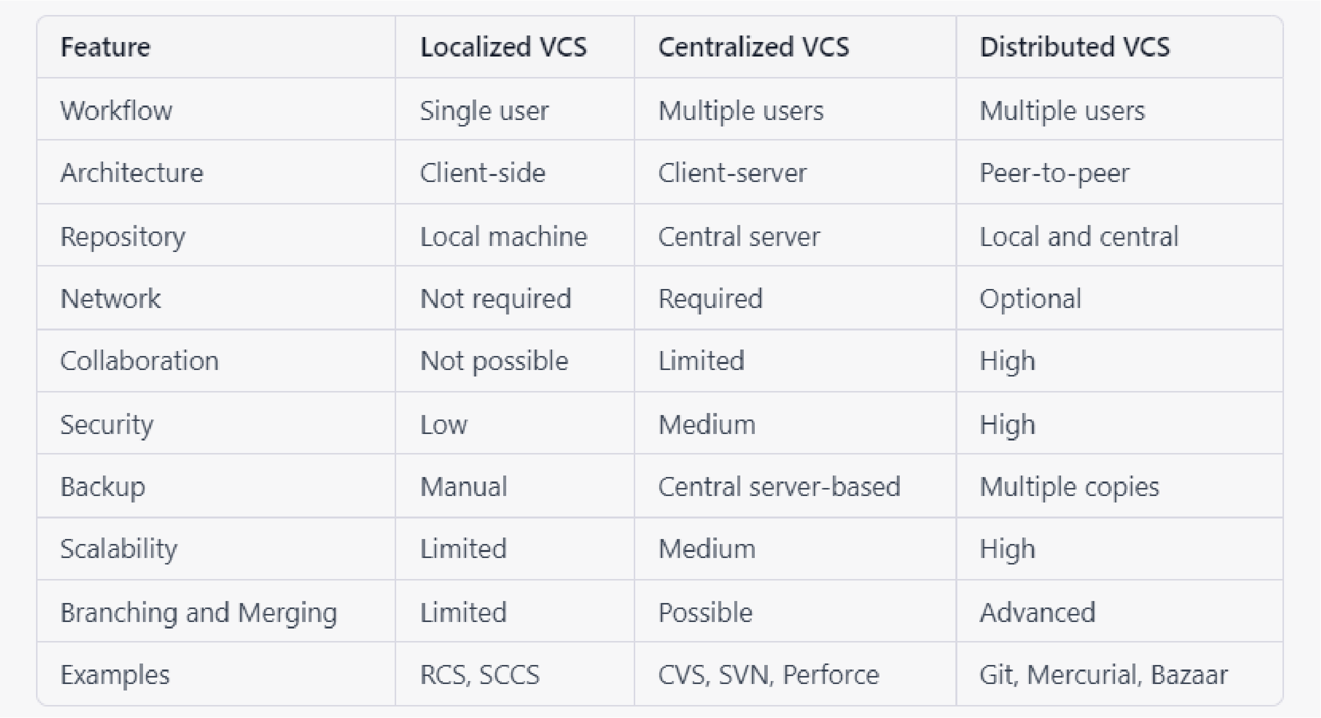
1. **GitHub:** The most popular Git hosting service, with a wide range of features and tools for software development.
2. **GitLab**: An open-source Git hosting service that provides a platform for managing Git repositories, issue tracking, and continuous integration.
3. **Bitbucket**: A Git hosting service that offers both free and paid plans, with features such as issue tracking, pull requests, and build automation.
4. **SourceForge:** A web-based platform that provides hosting for software development projects, including Git repositories.
5. **CodePlex**: A Git hosting service that is primarily focused on hosting open-source projects, with features such as issue tracking, source code browsing, and wiki pages.
6. **Launchpad**: A Git hosting service that provides a platform for software development, including hosting for Git repositories, bug tracking, and code reviews.
7. **Assembla**: A Git hosting service that provides a platform for software development, including hosting for Git repositories, issue tracking, and code reviews.

There are many other Git hosting services available, but these are some of the most popular and widely used platforms.

Ques5: Different types of Version Control System?

Ans 5: There are three types of version control systems (VCS):

1. A local version control system (VCS) is a VCS that runs on a single computer and stores the version history of files in a local database. This type of VCS does not require a central server or internet connection, and is typically used by individual developers or small teams.
2. Centralized Version Control System (CVCS): This type of VCS uses a central server to store the repository, and users can check out and modify files from that central repository. Examples of CVCS include CVS and Subversion (SVN).
3. Distributed Version Control System (DVCS): This type of VCS allows users to create a local repository on their own machines, and then push changes to a remote repository, or pull changes from other team members. Examples of DVCS include Git and Mercurial



Feature	Localized VCS	Centralized VCS	Distributed VCS
Workflow	Single user	Multiple users	Multiple users
Architecture	Client-side	Client-server	Peer-to-peer
Repository	Local machine	Central server	Local and central
Network	Not required	Required	Optional
Collaboration	Not possible	Limited	High
Security	Low	Medium	High
Backup	Manual	Central server-based	Multiple copies
Scalability	Limited	Medium	High
Branching and Merging	Limited	Possible	Advanced
Examples	RCS, SCCS	CVS, SVN, Perforce	Git, Mercurial, Bazaar

Ques6: What are the benefits come with using Git?

Ans6: There are several benefits to using Git as a version control system:

1. **Distributed architecture**: Git is a distributed version control system, which means that every user has a full copy of the repository on their local machine. This allows for offline work, faster cloning, and reduces the reliance on a central server.
2. **Branching and merging:** Git makes it easy to create branches, experiment with new features, and merge changes back into the main branch when they are ready. This helps keep the codebase organized and reduces the risk of conflicts.
3. **Speed and performance:** Git is designed to be fast, even when dealing with large codebases or repositories with a long history.
4. **Collaboration**: Git allows multiple developers to work on the same codebase simultaneously, with each developer able to work on their own copy of the repository and merge changes back in when they are ready.
5. **Open-source and community-driven:** Git is open-source software and has a large and active community of users and contributors, which means that it is constantly evolving and improving.
6. **Integration:** Git can be easily integrated with other tools and services, such as continuous integration and deployment systems, code review tools, and issue trackers.

Ques7: What is a Git repoistory?

Ans7: A Git repository is a collection of files and directories, along with the version history of those files and directories, that is managed by the Git version control system.

When you create a new Git repository, Git creates a .git directory in the root directory of the repository. This directory contains all of the metadata and version history for the repository, including information about the files and directories that are tracked, the commit history, and the branches and tags that have been created.

The repository can be hosted on a local machine or a remote server, and can be accessed by multiple developers who can collaborate on the codebase. Developers can clone a copy of the repository to their local machines, make changes to the files, and then push their changes back to the remote repository, where they can be merged with changes made by other developers.

In short, a Git repository is a central location where code is stored and versioned, and where multiple developers can collaborate on the same codebase.

Ques8:How can you initialize the repository in the Git?

Ans8: To initialize a new Git repository, you can use the "git init" command in the terminal or command prompt.

Here are the steps to initialize a new Git repository:

1. Open a terminal or command prompt and navigate to the directory where you want to create the repository.
2. Run the command "git init". This will create a new .git directory in the current directory, which contains all of the metadata and version history for the repository.
3. Optionally, you can create a new file in the repository directory and add it to the repository using the "git add" command. For example, if you create a new file called "README.md", you can add it to the repository by running the command "git add README.md".
4. Once you have added any files to the repository, you can make an initial commit using the "git commit" command. For example, you can run the command "git commit -m 'Initial commit'" to make an initial commit with a commit message of "Initial commit".
   After these steps, you have successfully initialized a new Git repository and made an initial commit. You can continue to add files, make changes, and commit your changes as you work on your project.