Post-Read: Understanding GIT

Session Recap:

We started the session with a brief introduction to GitHub, we discussed its importance, advantages and implementation in real-life Software Development Environment. We learned about the Version Control System, also discussed why and how it has become so mandatory in real-life software development. We discussed the challenges developers faced before GitHub and how GitHub has helped them to overcome them.

Then we learned how to install the GitHub, let's quickly recall the steps involved -

1. Create an account on github.com and login.
2. Install git for your computer system (Windows/Mac/Linux).
3. Add SSH key into the GitHub account.
4. Create a repository on GitHub.
5. Link the repository with your local Project Folder.

We have also learned about the SSH Key configuration, and learned a few git commands. Please refer to the Git-Command Guide shared in the pre-read documents to learn more. Here are few commands that you may need for your usual GIT-operations -

Git Commands for reference:

$ git init	Initializes a new Git repository. If you want to place a project under revision control.
$ git add .	Moves changes from the working directory to the staging area.
$ git commit -m “<message>”	Takes the staged snapshot and commits it to the project history. Combined with git add, this defines the basic workflow for all Git users
$ git status	Displays the state of the working directory and the staged snapshot.
$ git log	Lets you explore the previous revisions of a project. It provides several formatting options for displaying committed snapshots.
$ git branch	Will show the list of branches within a repository

© 2023 HeroX Private Limited. All rights reserved.

$ git branch <newBranchName>	This command is your general-purpose branch administration tool. It lets you create isolated development environments within a single repository.
$ git checkout	checking out old commits and old file revisions
$ git checkout <branchName>	In addition to checking out old commits and old file revisions, git checkout is also the means to navigate existing branches.
$ git branch -M <new name>	Renames the branch
$ git clone <repository-URL>	Creates a copy of an existing Git repository. Cloning is the most common way for developers to obtain a working copy of a central repository
$ git reset	Undoes changes to files in the working directory. Resetting lets you clean up or completely remove changes that have not been pushed to a public repository
$ git pull <repository-URL> <branch-name>	Pulling is the automated version of git fetch. It downloads a branch from a remote repository, then immediately merges it into the current branch.
$ git push <repository-URL> <branch-name>	It lets you move a local branch to another repository, which serves as a convenient way to publish contributions.

© 2023 HeroX Private Limited. All rights reserved.