WEEK 1 (DAY-3)

1) Why Do We Need a Version Control System?

A version control system (VCS) is a software tool that helps manage changes to files, documents, or any other type of content over time. It provides a systematic approach to track, organize, and control revisions made to a project.

2) What is Git?

Git is a distributed, open-source version control system. It is a revision control system used to track changes in computer files. It is a tool to manage your code & file history while co-ordinating work remotely on those files with others.



3) Architecture of Git. / Workflow of Git.



4) Git installation steps and setup.

5) Basic Local Git Operations. (On GitBash)

To check the present directory, we use pwd command

$pwd

To check the version, we use

$ git –-version

To remove files from both staging and working directory

$git rm a.txt

To remove files only from staging area

$git rm –-cached a.txt

To remove files only from working directory

$rm a.txt

To know how many files created in working directory

$ls

To know the staging area files

$git ls-files

If you are commiting first time, then then configure using

$git config --global user.email “[keertidesai2318@gmail.com](mailto:keertidesai2318@gmail.com)”

$git config –-global user.name “keerti”

▪ creating a repository,

To create folder in cmd, we use mkdir command.

$mkdir folder name

To change the directory

$cd folder name

To create a local repository

$git init

* “git init” creates an empty Git repository in the current directory.

▪ cloning a repository,

It is used to make a copy of the target repository or clone it. If I want a local copy of my repository from GitHub, this tool allows creating a local copy of that repository on your local directory from the repository URL.

$ git clone path\_to\_source-directory

▪ making and recording changes

To create a file, we use

$vim filename or touch filename

To Check the status of your repository

$git status

* “git status” command provides information about the modified files, untracked files, and staged changes.
* Note: Red color files are untracked files, Green color files are staged files.

▪ staging and committing changes

Before committing the changes, we need to stage them (adding files into staging area)

$git add filename

To stage multiple file at once, we use

$ git add file1name file2name

To stage all files at once, we use

$git add .

Run “git status” again to ensure that the desired changes are staged and ready to be committed. The staged files are in Green color.

To commit the staged files(storing all the files in local repository), we use

$git commit –m “write some message”

▪ viewing the history of all the changes

To view the history of changes, we use

$git log

* This command displays a list of commits, including the commit message, author, date, and a unique commit hash.

To know how many commits are done in one line, we use

$git log –-oneline

▪ undoing changes

6) Git Branching and merging.

***Git Branching***: In a GitHub, Master is the main branch. All the source code will be stored in this branch after reviewing by the team members.

Branching is a fundamental and powerful feature of Git that allows for parallel development, isolating changes, and managing different lines of development within a repository. Each branch in Git represents an independent line of development, enabling multiple contributors to work on different features or bug fixes simultaneously.

* Important features of branching :--
* When we create a new branch, the new branch will inherit the properties of master branch.
* The branches are isolated from each other, means the developer modify the contents from one branch will not effect to another branch

▪ Creating and switching to new branches

* Open a default directory, create a folder like GITPROJECT.
* Open git cmd, type $git init .

This will create a local repository (ex:C:/gitproject/.git)

* To create a file, type $touch a.txt
* To know file is created or not,type $ls
* To add file into staging area, we use $git add a.txt
* To commit a file, we use $git commit -m “this is my file”
* To view the branch OR see the currently active branch, we use $git branch
* To create a new branch, we use $git branch branch name
* ***WHY TO SWITCH BRANCH? To know the files and folders from the particular branch***
* To switch to the newly created branch, we use $git checkout branch name
* Alternatively, we can combine branch creation and switching into a single command, we can use $git checkout -b branch name

▪ Switching between branches

* To view all the branches in repository, we use $git branch
* To switch to an existing branch, we use $git checkout branch name
* If we want to switch to a branch named "feature-branch", we type $git checkout feature
* It will switch working directory and current branch to specified branch
* After running this command, we can check that is switched to another branch, type $git branch

▪ Merging local branches together

***Merging***: is used to combine changes from different branches into the current branch. It allows you to integrate the changes made in one branch into another branch, incorporating new features, bug fixes, or updates from one branch to another

Ones we complete the feature branch(new branch), we can merge new branch with master branch

There Are Two Ways of Merging

* Fast Forward Merging
* Three Way Merging

WEEK 1 (DAY 4)

1)GitHub : It is a hosting service for git repositories. Git is a tool, while GitHub is the service to use Git.

It is also a web-based platform designed to help developers collaborate on software projects. It primarily focuses on version control, allowing multiple people to work on the same codebase simultaneously and keep track of changes over time.

- Basics of distributed git

1. Repository

2. Branch

3. Commit

4. Pull request

5. Merge

6. Push

- Account creation and configuration

1. Open Google Chrome, type GitHub and Sign in to github account.

2. Click on Sign up to create a new account.

3. Specify username,email address and password. After specifying email, a confirmation mail will be sent to your email account.

4. Open your email account & confirmation mail. Confirm it by clicking on a link present in mail.

5. After creating an account, Click sign in and provide username & password. Click on sign in button.

6. You are signed in to the GitHub account

- Create and push to repositories

1. To create a new repository, we have two options:-

i. Click on NEW button

ii. Click on [+] symbol at right side, Select option as new repository.

2. Mention the repository name as testrepo.

3. If everyone want to access the file , select option as public.

4. Click on create repository button.

5. We get repository url, by sharing this url anybody can access the repository.

6. To create a new file in remote repository, click on creating a new file link.

7. Specify the filename and smention the content.

8. After writing the content, Click on commit new file.

* CLONING

9. To clone the repository into a local system, Open the default folder, Right click on folder and select GitBash.

$git clone '<https://github.com/keerti/testrepo.git>'

10. To enter into cloned repository

$cd repositoryname(ex: $cd testrepo)

11. To know the existing file from the cloned repository

$ls

12. To know the content of the file

$ cat filename (ex: cat myfile)

13. To modify the content of the file,

$vim myfile

14. To save and exit from the file press esacpe button and :wq! enter

15. To create a new file

$vim myfile2

press i (Insert)for insertion, Type content & save it (esc :wq!)

16. Add these file to staging area & commit mode

$git add .

$git commit -m "updated files......"

$git log --oneline

* PUSH

17. To push the files from local to remote repository

$git push origin main

18. In remote repository(GitHub). Open a file and edit the content.

19. After writing the content click on ***commit changes.***

* PULL

20. To Update the content in local repository

$git pull origin main

- Versioning

- Collaboration

Collaboration in GitHub involves multiple people working together on a shared project, often using version control to manage changes, track progress, and ensure a seamless development process. GitHub provides a platform and tools that facilitate collaboration among team members.

Using this concept for single repository(project) we can invite number of users , so that we can work together

* Create a repository&add files
* Click on settings, select option collaborator.
* Click on add people button to add users by specifying username or email id. Click on Select a collaborator above.
* Click on add, to add the user into a repository Invited user will open the GiHub account . click on profile dropdown list and select the option (your organization) click on join.
* click on accept invitation button.
* The repository will display in the invited user to copy repo from the remote to local, click on code dropdown list & copy the url.
* Open the git cmd, create a folder . right click on folder and select GitBash option
* To copy the repository $git clone <https://github.com/keerti/pizza-receipe.git>
* To enter into a folder using cmd $cd pizza-receipe/
* To add content of file, we can use vim & specify filename
* The file content will display. press i ti insert data- after modifying data, save file by pressing esc :wq! <==enter
* add all files to staging area $git add .
* To commit a file $git commit -m "modify file..."
* push the content to remote repository $git push origin main

- Migration

Migration in GitHub generally refers to the process of moving repositories, projects, or data from one location or platform to another. This can include migrating code from one version control system to Git, transferring repositories between GitHub accounts, or moving from another version control platform to GitHub.