**Git Assignment Answers**

### **Basic Git Answers:**

1. Git is a tool where it is used for source code management. It is a free and open source version control system. Git is used to track changes in code and it will help multiple developers together.
2. Git is a Version control system that manages the tracking of code history , where as the GitHub is a cloud server based platform that you can store the code, share work together to write the code with others.
3. Go to search engine , search for git install for specified windows or mac. Download the gif file, open the file run the installation file with administration. Choose the location where you need to save and install the required default components Git,Git Gui,Git Bash.

Add to git to the windows, choose next extra options to enable system caching, click finish to complete the installation and start using the Git.

1. First we need to open git cmd and we need to check git version, then git config --global –list, press enter and it will read the config file it shows the user and email.
2. A Git repository is a virtual storage of the project which will manage and track the files and directory.
3. First we need to create github account and open profile in that we need to click on your repositories and click on new repo give the repo name and create a Readme file , next step keep it as private or public as per your requirement , keep the authenticate key , click on create repository it will redirect to the commands.
4. To clone a repository from GitHub – git clone and < https: paste the url>.
5. The purpose of .Git ignore file is to ensure certain files that Git remain un tracked.
6. To check the status of your working directory in Git is – git status.
7. To add files to the staging area in Git – git add <file name>.

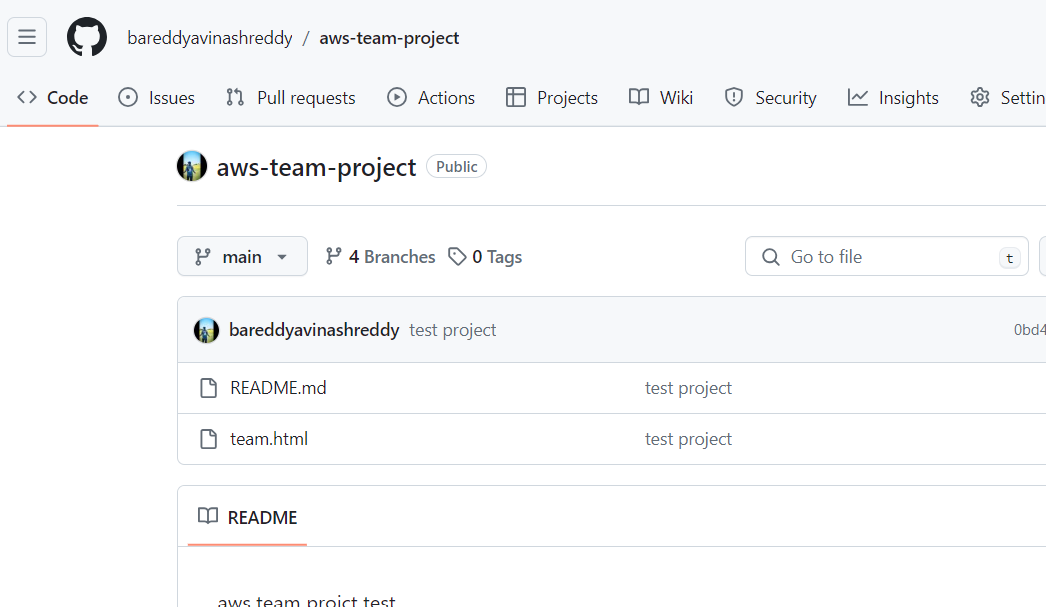
### **Intermediate Git Questions:**

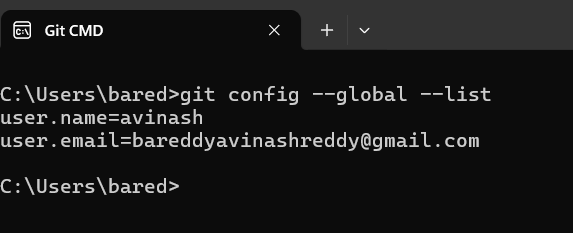
1. The concepts of git commit is basically when you commit the changes you are telling git to record the current state of your project inclusing modifications since the last commit you have change.
2. To create a new commit in Git – git commit “msg”.
3. The purpose of the git log command is used to displays all of the commits in a repository's history.
4. To view the history of commits in a repository we can use - git log command with the branch name <git log branch name>.
5. To view the changes made in a commit - git show <commit-hash>
6. The branch in git is used to allow work on different parts of projects without impacting the main branch. after the work is completed we can merge the all branches with the main branch.
7. To create a new branch in Git we use , git branch new\_branch . Once it is created we need to use git checkout new\_branch to switch to that branch.
8. To switch between branches in Git we use , git switch <branch name >.
9. difference between git merge and git rebase is Git merge is used to merge different branches where as git rebase is used for  integrate the changes from one branch into another. <git merge feature main> , <git rebase main>.
10. To resolve merge conflicts in Git open a file and make any necessary changes After editing the file, we can use the git add a command to stage the new merged content, and finally is to create a new commit with the help of the git commit command.

### **Advanced Git Questions:**

1. Git stash is used to record the current state of the working directory and the index, but want to go back to a clean working directory and that command saves your local modifications away and revert working directory to match the Head commit.
2. To apply stashed changes in Git, git stash apply STASH-NAME applies the changes and leaves a copy in the stash.
3. The purpose of the git tag command is used to create tags in repository branches.
4. To create and push tags to a remote repository (git remote –v), (git tag –a v1.0 –m “msg”), (git push origin v1.0).
5. The concept of remote repositories in Git is , Git repository that's hosted on the Internet or another network.
6. To you add a remote repository in Git use the git remote add command on the terminal, in the directory your repository is stored at.
7. To push changes to a remote repository , (git push -u remote-name <branch-name>).
8. To pull changes from a remote repository we need to use git fetch.
9. The command git fetch Is used for downloads objects to the local machine without overwriting existing local code in the current branch. The command pulls a record of remote repository changes, allowing insight into progress history before adjustments.
10. To delete a branch in git ( git delete –d “branch name”).

### **Git Exercises**

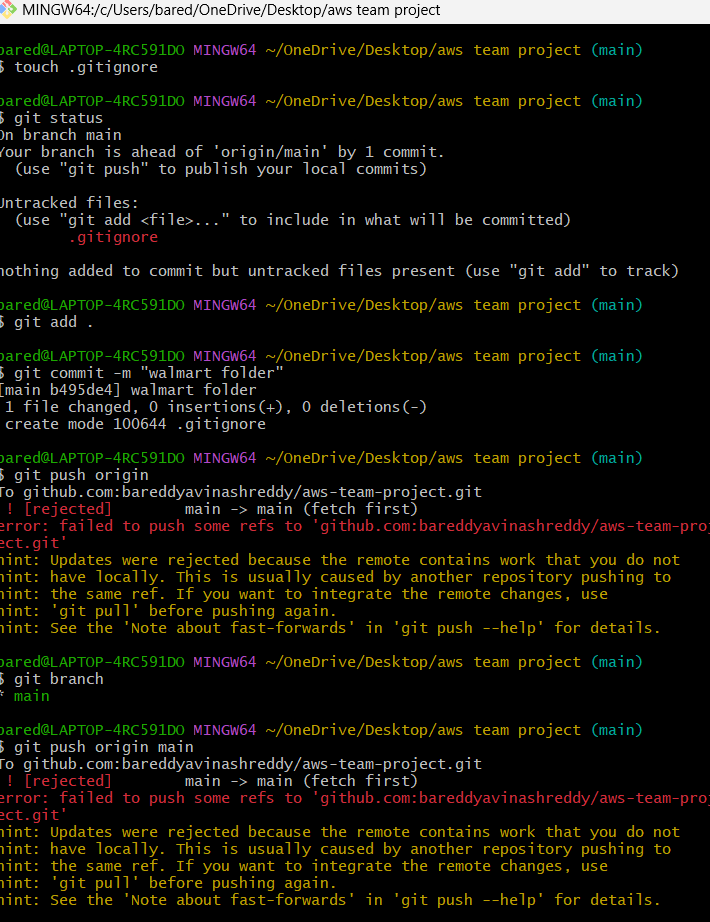
1. Create a new Git repository and configure your username and email 



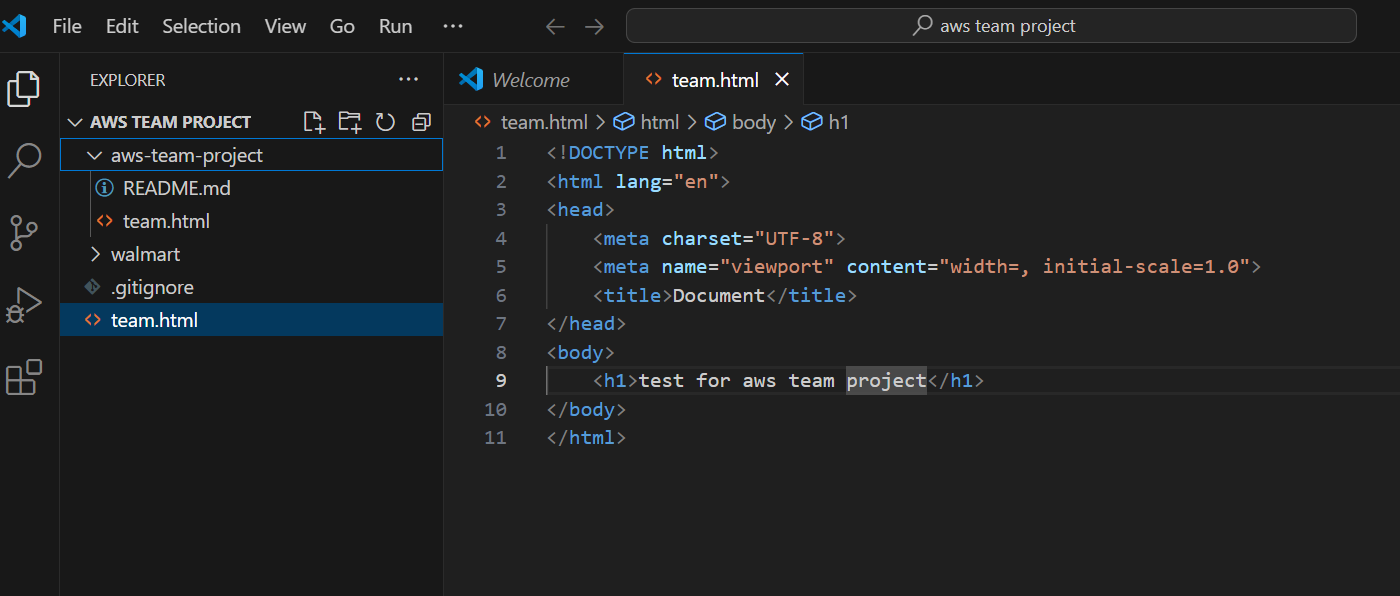
1. Create a file, add some content to it, and commit the changes



1. Create a .gitignore file and add rules to ignore specific files and directories



1. Clone an existing repository from GitHub and make some changes



1. Create a new branch, make some changes, and switch back to the main branch

* git checkout -b new-branch
* git add .
* git commit -m "Made changes on new-branch"
* git checkout main

1. Merge changes from a feature branch into the main branch

* git checkout main
* git pull origin main
* git merge feature-branch
* git commit -m "Merge changes from feature-branch into main"
* git push origin main

1. Resolve a merge conflict between two branches

* git checkout branchA
* git merge branchB
* content from branch A , content from branch B
* Combined content from branchA and branchB
* git add <file\_with\_conflict>
* git add .
* git commit -m "Resolved merge conflict between branchA and branchB"
* git push origin branchA

1. Use git stash to save your work and then apply the stashed changes

* git stash
* git checkout branchA
* git merge branchB
* <<<<<<< HEAD
* Content from branchA
* Content from branchB
* >>>>>>> branchB
* Combined content from branchA and branchB
* git add file.txt
* git commit -m "Resolved merge conflict between branchA and branchB"
* git stash pop
* git add .
* git commit -m "Applied stashed changes after resolving merge conflict"

1. Create a tag for a specific commit and push the tag to the remote repository

* git log
* git tag -a v1.0 abc123 -m "Tagging version 1.0"
* git push origin v1.0

1. Add a remote repository, push your local changes, and pull changes from the remote repository

* git init
* git remote add origin <link>
* echo "do some Some changes" > file.txt
* git add file.txt
* git commit -m "Add file.txt with some changes"
* git push -u origin main
* git pull origin main

1. Create a Git alias for a commonly used command

git config --global alias.st status

git config --global alias.lg "log --oneline --graph --decorate --all"

git config --global alias.cm "commit -m"

git config --global alias.co checkout

git config --global alias.plr "pull --rebase"

1. View the commit history with a graph representation



1. Revert a commit and understand the differences between git revert and git reset

* The git revert preserves the commit history by creating a new commit that undoes the change.
* Reset rewrites the commit history by undoing the operations it is a safer option when working with other developers because it doesn't affect others' work.

1. Squash multiple commits into a single commit using git rebase

* git rebase -i HEAD~1
* git checkout new\_branch\_name
* git merge --squash <Branch>
* git rebase -i HEAD~2
* pick <commit\_hash\_1> <commit\_message\_1>
* squash <commit\_hash\_2> <commit\_message\_2>

1. Use git bisect to find the commit that introduced a bug

* git bisect start
* git bisect bad
* git bisect good HEAD~10
* git bisect bad
* git bisect good
* git bisect reset

1. Set up a Git hook to run a script before or after certain Git events
2. Use git cherry-pick to apply a commit from one branch to another
3. Learn how to use git blame to find out who made specific changes to a file
4. Configure and use a Git GUI client
5. Create a fork of a repository on GitHub, make changes, and open a pull request

* git clone <https://>
* cd <repository-name>
* git remote add upstream [https://](NULL)
* git checkout -b <my-feature-branch>
* need to Make changes using a text editor then
* git add .
* git commit -m "Add new feature"
* git push origin <my-feature-branch>
* Go to your forked repository on GitHub.
* Click "Compare & pull request".
* Fill out the form and submit the pull request