**Que : What is Github? When was it created? Why? By Who? What similar platforms exist? Why would you use such a platform?**

GitHub is a code hosting platform for version control and collaboration. It lets us and others work together on projects from anywhere. It was launched in April 2008 by Tom Preston-Werner, Chris Wanstrath, and PJ Hyett . Similar platforms as github are bitbucket, gitlab, Redmine, sourceforge.

We would use such a platform because it offers all of the distributed version control and source code management (SCM) functionality of Git with its own features. Its own features include:

 Documentation, including automatically-rendered README files

 Issue tracking

 Commits history.

 Pull requests with code review and comments.

 Wikis.

 Integrations Directory.

 Email notifications.

**Que : Work of Git tutorial:**

1. In the tutorial, first step is to initialize a git repository for which the command is: ‘git init’
2. To know the git status: ‘git status’

Git creates a new file: octacat.txt

1. Now, to start tracking changes made to octacat.txt, we have to add it to staging area by using ‘git add’
2. After running ‘git add’ command, it started tracking octacat.txt file.
3. Commiting: It means to make a change. We are storing the file with a message that what we have changed by running command- git commit -m “Add cute octocat story”
4. We can use wildcards to add many files of the same type. Now, we are adding all the files with the help of command- git add ‘\*.txt’ and then we stored it with a message ‘Add all the txt files’
5. We can see the changes we have committed in the order we committed them by running command- ‘git log’
6. A new empty github repository is created to use with Try git at <https://github.com/try-git/try_git.git>. We are pushing the local repo of git to the github server by running command- git remote add origin <https://github.com/try-git/try_git.git>
7. With the help of push command, we can tell Git where to put our commits when we are ready. We are now going to run a command “git push –u origin master”. The name of our remote is origin and the default local branch is master. The –u tells Git to remember the parameters.
8. We can check for changes on our github repository and can pull down any new changes by running command: ”git pull origin master”
9. We can see the change from our last commit by running command: “git diff HEAD”. We use HEAD when we want to see the diff of our most recent commit.
10. Staged files are the files which are ready to be committed.
11. After this, we are going to run command “git-diff --staged” to see the changes we staged.
12. By running the command “git reset octofamily/octodog.txt”, we can remove the file octofamily/octodog.txt
13. We can undo all the changes we have made by running command “git checkout--<target>”
14. According to the tutorial, now we have made a new branch “clean\_up” by running command “git branch clean\_up”
15. We can see the branches by running “git branch” and we can switch branches using command “git checkout <branch>”
16. We can remove the files from disk by running command “git rm”. Ex- if we want to remove the txt files, we will run command “git rm’\*.txt’”
17. Now, according to the tutorial, we will commit changes by running “git commit –m”Remove all the cats””
18. Now, we need to merge the changes we did in clean up branch to master branch. For that, first we will have to go to master branch and then we will run command “git merge clean\_up”.
19. To clean a branch, we can run command ”git branch -d <branch name>”.
20. To push everything we have been working on to our remote repository, run command “git push”

**Que : Define the following terms:**

**Repository**: A **repository** is usually used to organize a single project. We can insert folders and files, images, videos, spreadsheets, and data sets in the repositories – anything our project needs.

**Commit**: On GitHub, saved changes are called commits. Each commit has an associated commit message, which explains why a particular change is made. Commit messages caontains the history of our changes, so other contributors can understand what we have done and why.

**Push**: Push means to update the file. For ex-Pushing a file in Github means to load the file in github repository which we are using.

**Branch**: **Branching** is the way to work on different versions of a repository at one time. By default, our repository has one branch named master which is considered to be the definitive branch. We use branches to experiment and make edits before saving the changes to master.

**Fork**: In git, we can merge a branch to another branch. As an example, if we want to merge branch A to branch B, we can go to branch B and run command “git merge A”.

**Clone**: The git clone command copies an existing Git repository.

**Pull**: Git pull runs git fetch with the given parameters and calls git merge to merge the retrieved branch heads into the current branch.

**Pull Request**: Pull Requests are the heart of collaboration on GitHub. When we open a pull request, we are proposing our changes and requesting that someone review and pull in our contribution and merge them into their branch. Pull requests show diffs, or differences, of the content from both branches.

As soon as we make a commit, we can open a pull request and start a discussion, even before the code is finished.