Assignment – 2 (Github)   - Dixit Patel

(1) One of the biggest advantages of Git is its branching capabilities. Unlike centralized version control systems, Git branches are cheap and easy to merge.
There is some more advantages of Git are as under:
(1) It keeps track if changes to the code.
(2) Revert back to the old version of the code.
(3) Test changes to code without losing the original.
(4) It synchronize code between different people.

(2) C, TCL ,Perl, Python, Shell.

(3) The staging area is a simple file, generally contained in your Git directory ,that stores information about what will go into your next commit. It's sometimes referred to as the index, but it's becoming standard to refer to it as the staging area.
The Git index is used as a staging area between your working directory and your repository. You can use the index to build up a set of changes that you want to commit together.

(4) - In the upper-right corner of any page, click , and then click New repository.
- Type a short, memorable name for your repository. …
- Optionally, add a description of your repository. ...
- Choose to make the repository either public or private. ...
- Select initialize this repository with a README.
- Click Create repository.

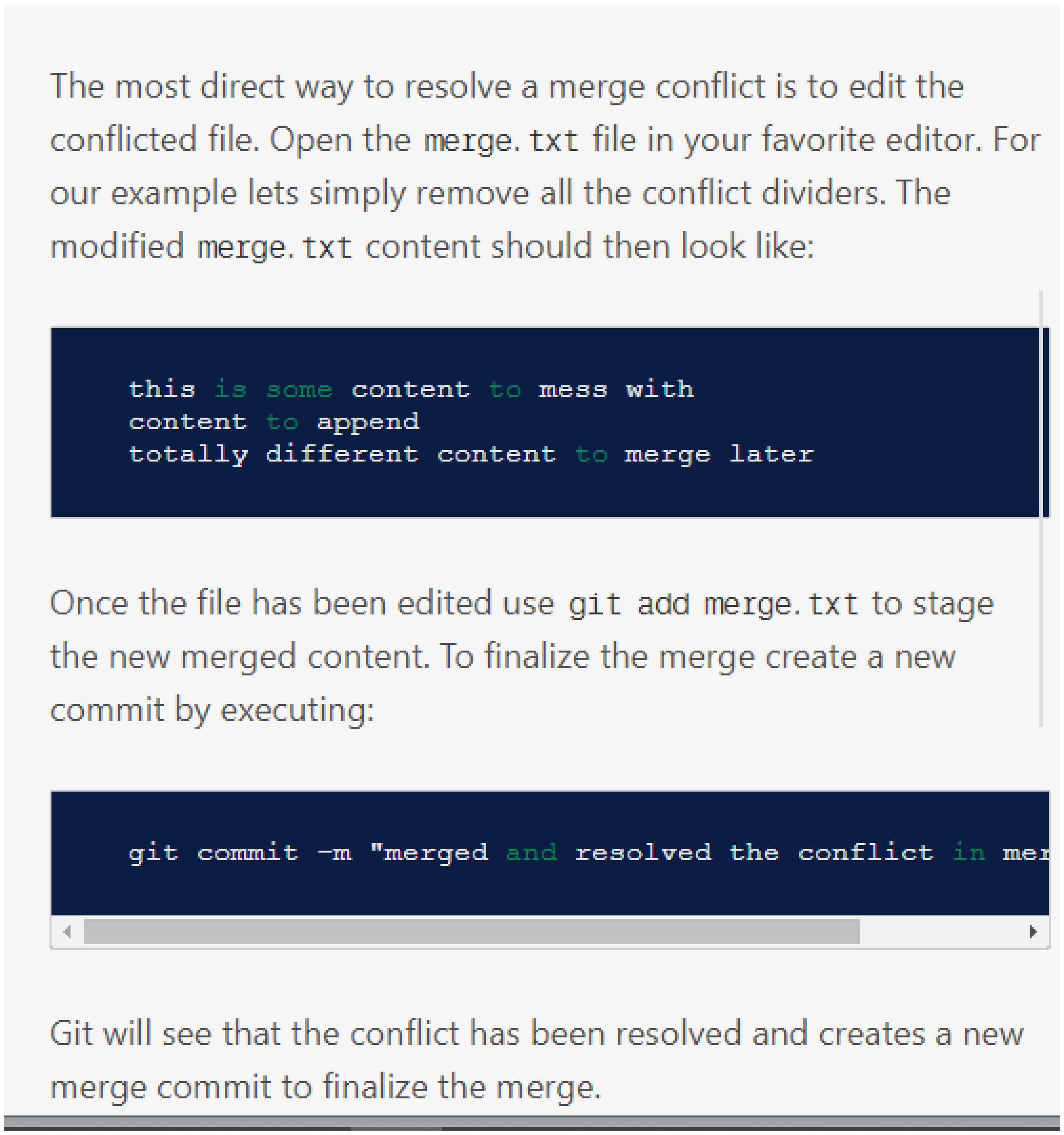
(5) HEAD is a reference to the last commit in the currently check-out branch. You can think of the HEAD as the "current branch". When you switch branches with git checkout, the HEAD revision changes to point to the tip of the new branch. You can see what HEAD points to by doing: cat .git/HEAD.
There can be any number of heads in a GIT repository. By default there is one head known as HEAD in each repository in GIT.

(6) Branching is a feature available in most modern version control systems. ... Instead of copying files from directory to directory, Git stores a branch as a reference to a commit. In this sense, a branch represents the tip of a series of commits—it's not a container for commits.

(7) - From your terminal window, list the branches on your repository. ...
- Create a new feature branch in the repository.
- Switch to the feature branch to work on it.
- Commit the change to the feature branch.
- Switch back to the master branch.
- Push the feature branch to Bitbucket.

(8) A conflict arises when two separate branches have made edits to the same line in a file, or when a file has been deleted in one branch but edited in the other. Conflicts will most likely happen when working in a team environment

(9)



The most direct way to resolve a merge conflict is to edit the conflicted file. Open the `merge.txt` file in your favorite editor. For our example lets simply remove all the conflict dividers. The modified `merge.txt` content should then look like:

```
this is some content to mess with
content to append
totally different content to merge later
```

Once the file has been edited use `git add merge.txt` to stage the new merged content. To finalize the merge create a new commit by executing:

```
git commit -m "merged and resolved the conflict in mer
```

Git will see that the conflict has been resolved and creates a new merge commit to finalize the merge.

.

(10) The git config command is a convenience function that is used to set Git configuration values on a global or local project level. These configuration levels correspond to .gitconfig text files. Executing git config will modify a configuration text file.

(11) A fork is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project. Mac Windows Linux All. Most commonly, forks are used to either propose changes to someone else's project or to use someone else's project as a starting point for your own idea.

(12) A fork is really a Github (not Git) construct to store a clone of the repo in your user account. As a clone, it will contain all the branches in the main repo at the time you made the fork. ... These commits are pulled from either my fork or my branch to the main repo. A commit is a set of changes to the code.

(13) Pull requests let you tell others about changes you've pushed to a GitHub repository. Once a pull request is sent, interested parties can review the set of changes, discuss potential modifications, and even push follow-up commits if necessary.
A separate version of the code is BRANCH.

(14) When downloading content from a remote repository, git pull and git fetch commands are available to accomplish the task. You can consider git fetch the 'safe' version of the two commands. It will download the remote content, but not update your local repository's working state, leaving your current work intact. git pull is the more aggressive alternative, it will download the remote content for the active local branch and immediately execute git merge to create a merge commit for the new remote content. If you have pending changes in progress this will cause conflicts and kickoff the merge conflict resolution flow.

(15) If you want to revert the last commit just do git revert (unwanted commit hash); then you can push this new commit, which undid your previous commit. To fix the detached head do git checkout (current branch).

(16) The main advantage of the Forking Workflow is that contributions can be integrated without the need for everybody to push to a single central repository. Developers push to their own server-side repositories, and only the project maintainer can push to the official repository. This allows the maintainer to accept commits from any developer without giving them write access to the official codebase.

(17) HEAD is a reference to the last commit in the currently check-out branch. You can think of the HEAD as the "current branch".
Working tree are the files that you are currently working on.
Index is not the working directory and does not have to include the working directory. Index is just a file within the git repository that stores info what you want to commit.

(18) (1) Find last commit hash on master branch
(2) Find last commit hash on a "branch"
(3) Run command git merge-base <commit-hash-step1> <commit-hash-step2>.
(4) If output of step 3 is same as output of step 2, then a branch has been already merged into master.

(19) Git clone is primarily used to point to an existing repo and make a clone or copy of that repo at in a new directory, at another location. The original repository can be located on the local filesystem or on remote machine accessible supported protocols. The git clone command copies an existing Git repository.

(20) Git stash temporarily shelves (or stashes) changes you've made to your working copy so you can work on something else, and then come back and re-apply them later on.

(21) Use git stash when you want to record the current state of the working directory and the index, but want to go back to a clean working directory. The command saves your local modifications away and reverts the working directory to match the HEAD commit.

(22) In case we do not need a specific stash, we use git stash drop command to remove it from the list of stashes. By default, this command removes to latest added stash.
To remove a specific stash we specify as argument in the git stash drop <stashname> command.

(23) The git stash saves command takes your uncommitted changes (both staged and unstaged), saves them away for later use, and then reverts them from your working copy.

(24) A README.MD file contains information about other files in a directory or archive of computer software. A form of documentation, it is usually a simple plain text file called README.MD. README.md is used to generate the html summary you see at the bottom of projects. Github has their own flavor of Markdown. Order of Preference: If you have two files named README and README.md, the file named README.md is preferred, and it will be used to generate github's html summary.
MD stands for markdown.

(25) ##Create a new repository on the command line :

```
touch README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin git@github.com:alexpchin/<reponame>.git
git push -u origin master
```

(26) Git checkout , when used on branches, alters the target of the HEAD ref. It can be used to create branches, switch branches, and checkout remote branches. The git checkout command is an essential tool for standard Git operation. It is a counterpart to git merge.

(27) **git push -u origin new-feature** - This command pushes new-feature to the central repository (origin), and the -u flag adds it as a remote tracking branch. After setting up the tracking branch, git push can be invoked without any parameters to automatically push the new-feature branch to the central repository.

(28) The git rm command can be used to remove individual files or a collection of files. The primary function of git rm is to remove tracked files from the Git index. Additionally, git rm can be used to remove files from both the staging index and the working directory.

(29) you can reapply the changes to your working copy and keep them in your stash with git stash apply.

(30) Git log - Formatting Log Output Filtering the Commit History Summary. The purpose of any version control system is to record changes to your code.

(31) The git add command adds a change in the working directory to the staging area. It tells Git that you want to include updates to a particular file in the next commit.

(32) git diff is a multi-use Git command that when executed runs a diff function on Git data sources. These data sources can be commits, branches, files and more.

(33) The git status command displays the state of the working directory and the staging area. It lets you see which changes have been staged, which haven't, and which files aren't being tracked by Git. Status output does not show you any information regarding the committed project history.

(34) No

(35) The -d option stands for --delete , which would delete the local branch, only if you have already pushed and merged it with your remote branches. The -D option stands for --delete --force , which deletes the branch regardless of its push and merge status, so be careful using this one!

(36) In Git, this simplest form of integration is called a "fast-forward" merge. Both branches then share the exact same history. In a lot of cases, however, both branches moved forward individually. To make an integration, Git will have to create a new commit that contains the differences between them - the merge commit.

(37) For removing a single file : git rm --cached mylogfile.log

(38) when looking to incorporate changes from one Git branch into another.Use merge in cases where you want a set of commits to be clearly grouped together in history.Use rebase when you want to keep a linear commit history.

(39) Git repository is just a file location where you are storing all the files related to your project. ... When you git commit your code, a version/snapshot is created in your local repo.
Remote repository: A remote repository generally lies somewhere outside your system, on a remote machine.

(40) git commit –m

(41) A commit object contains the reference to another tree object and some other information(author, committer etc.).

(42) .

(43) Bitbucket , SVN, Perforce , Mercurial.

(44) Every gist is a Git repository, which means that it can be forked and cloned. If you are signed in to GitHub when you create a gist, the gist will be associated with your account and you will see it in your list of gists when you navigate to your gist home page. Gists can be public or secret.

(45) .

(46) GitLab , Bitbucket

.