**Add React App files in the GitHub repository**

1. Create a new directory and change into it.
2. Use the **init** command to create a Git repository in that directory. Observe that there is now a .git directory.
3. Create a README file. Look at the output of the **status** command; the README you created should appear as an untracked file.
4. Use the **add** command to add the new file to the staging area. Again, look at the output of the **status** command.
5. Now use the **commit** command to commit the contents of the staging area.
6. Create a src directory and add a couple of files to it.
7. Use the **add** command, but name the directory, not the individual files. Use the **status** command. See how both files have been staged. Commit them.
8. Make a change to one of the files. Use the **diff** command to view the details of the change.
9. Next, add the changed file, and notice how it moves to the staging area in the **status** output. Also observe that the **diff** command you did before using add now gives no output.
10. Now – without committing – make another change to the same file you changed in step 10. Look at the status output, and the **diff** output.
11. Notice how you can have both staged and unstaged changes, even when you’re talking about a single file.
12. Observe the difference when you use the add command to stage the latest round of changes. Finally, commit them.
13. Use the **log** command in order to see all of the commits you made so far.
14. Use the **show** command to look at an individual commit. How many characters of the commit identifier can you get away with typing at a minimum?
15. Make a couple more commits, at least one of which should add an extra file.

git config http.proxy <http://192.168.222.1:3128>

git config https.proxy <http://192.168.222.1:3128>

**Add Collaborators for the React App**

For this task, you will work in a small group. Between 2 and 4 people is about right.

Main Task

1. First, one person in the group should **create a public repository** using their GitHub account.
2. This same person should then follow the instructions from GitHub to **add a remote**, and then **push** their repository. Do not forget the –u flag, as suggested by GitHub!
3. All of the other members of the group should then be added as **collaborators**, so they can commit to the repository also.
4. Next, everyone else in the group should **clone** the repository from GitHub. Verify that the context of the repository is what is expected.
5. One of the group members who just cloned should now make a local **commit**, then **push it**. Everyone should verify that when they pull, that commit is added to their local repository (use git log to check for it).
6. Look at each other’s **git log** output
7. Two members of the group should now **make a commit** locally, and **race to push it**. To keep things simple, be sure to edit different files.
8. Repeat the last two steps a couple of times, to practice.

Stretch Task

1. Now create a situation where two group members both edit the same line in the same file and commit it locally. Race to push.
2. When the runner-up does a pull, they should get a merge conflict.
3. Look as a group at the file in conflict, and resolve it.
4. Use the add command to stage the fix, and then use commit to make the merge commit.
5. Notice how this procedure is exactly the one you got used to when resolving conflicts in branches.