Prepared by Team OSOC

Basic Git Workflow with GitHub

Getting Started with GitHub

1. Navigate to https://www.github.com
2. Create an account or sign in if you already have an account.

Creating a new repository in GitHub

1. After signing in, click on the GitHub Logo and click on the `New Repository` button.
2. Name of the repository should be meaningful and short with no spaces, can use '-' symbol between words.
3. For the time being name it as `github-demo` .
4. Give the description according to yourself.
5. Check the `public` radio button, so that your repository will be publicly seen.
6. Click on the "**Initialize this repository with a README**"
7. Click on the `Create Repository` button.

Preparing our local system

Set up the project folder

1. Open the command line (For Linux Users) or Git Bash if you are on windows.
2. Navigate to a directory of your choice using the `cd` command.

3. After navigating to a directory of your choice, create a `projects` directory by using the following command: `mkdir projects`
4. Navigate to the projects directory: `cd projects`

Git Configuration

1. Setting up user name: `git config --global user.name "<Insert you name here>"`.
2. Setup your email address used to sign up for GitHub: `git config --global user.email "email@example.com"`.
3. Confirm your credentials using: `git config --global --list`.

Copy the repository from GitHub to your local machine

Cloning a Repository to local machine

1. Locate the cloning options on the repository you just created on github and copy the URL for your repository.
2. It will be something like "https://www.github.com/username/repository-name.git
3. Now come back to the command line on your local machine and type the following command: `git clone <repository-link>`.
4. It will make a full copy of your repository on GitHub to your local machine.
5. Cloning will automatically create a folder with the repository name inside your projects folder.You can confirm it by typing `ls` inside your projects folder.
6. Navigate to the repository using the `cd` command.
7. `cd github-demo`
8. After navigating to the folder type: `ls` and then you will see a file named `README.md`

The First Commit

1. Create a file named `start.txt` and add some content to it.
2. Use the following command `echo "This is a simple text file" >> start.txt`

3. You can see the file in your working directory by using `ls` .
4. To see the contents of the file that you just created, we can use the `cat` command as: `cat start.txt`
5. Now to see the status of your git repository you can use the `git status` command and you will see that you have an untracked file “start.txt” that you just created, which means that this file is not yet included in your git repository.
6. To **add** it to your git repo, first of all you will have to bring it to the staging area and it can be done using: `git add start.txt`
7. Again if you check the status using `git status` then you can see that the file now is in the staging area ready to be **committed**.
8. To commit the file to the repository we use the `git commit` command as: `git commit -m "Added start text file"` .
9. In the above command `-m` means the commit message which is to identify each commit uniquely.Commit message should be very concise and should be clearly indicating the nature of the commit.
   10.Now check the status using the `git status` command and you will see that there is nothing to commit and the working directory is clean.This is how a commit is done.

Publishing Changes back to GitHub

After you have done your first commit, you can publish the changes to your GitHub repository,since the commit you have done is still local and the remote repository on GitHub will not be reflecting the changes that you have done on your local machine at this point of time.

1. We can do this by using the `git push` command. This command takes two arguments, the first one being the name of the GitHub copy of the repository (usually named ‘origin’)and the second being the local branch of the repository on your machine (usually named ‘main’).
2. The command is: `git push origin main` ,where origin refers to the GitHub copy of our repository and main refers to our default and only branch in our local repository.
3. As you type this command and press Enter, you will be prompted to enter the username and password of your GitHub account.
4. If everything went correctly then after refreshing your browser where you have opened the GitHub repository you will be able to see that the file you created is now being added to your GitHub repository.

Git Branching and Merging

Branches are meant to work on different aspects of the same projects simultaneously without affecting the current workflow.Suppose you are working on a project and you need to test out an experimental feature but do not want to add it to your project just yet. So what you can do is create a branch of your project, work on the experimental feature in this branch and when the feature is thoroughly tested, then merge this branch to your main branch of the project.

1. To create a new branch(say “experimental”) and switch to it at the same time run `git checkout -b experimental` .This is shorthand for:

```
git branch experimental
git checkout experimental
```

2. Let us create a new file in this branch:

```
echo "This is an experimental text file" >> new-file.txt
```

3. Add this file to the staging area and commit:

```
git add new-file.txt
git commit -m "new file added"
```

After this if you switch to the main branch you can see that the “new-file.txt” is not present in this branch.
4. Now, to merge the experimental branch with the main branch we need to use the `git merge` command after checking out the main branch.

```
git checkout main
git merge experimental
```

5. Now since we no longer require the experimental branch we can delete it:

```
git branch -d experimental
```

6. To see the branches in your repository you can run : `git branch` command.