Introduction to Software Systems

Lab 1: Understanding Git and Github

Date: March 29, 2022

Official docs

Repository creation

- Config your local machine, add user.name, user.email, github.user, github.token (generate access token from github), with `git config user.name "Jon Doe"` ..etc. Use `--global` for personal PCs.
- Make a directory named `ISS_Lab-1`
- Init the directory with `git init .`

Creating and adding files

- Make a file named `name.txt` with your name and roll number in it like the format below:

```
Kshitijaa Jaglan
2019115005
```

- `git add` your file
- Commit the file with a meaningful message
- `git push` the changes to the main repository
- Check online if your changes have been pushed. You should see the file `name.txt` with the details entered

Pull and updation

- Open the Github repo in your browser and add a second file called `name2.txt` with the same content
- Do `git pull` and check if the changes have been pulled to your local machine

Branching

- Use `git branch` to create a new branch called `other_details`
- Create a file called `other_details.txt` with content in the following format (Add your details instead) (in your main/master branch only):

```
Roll: 2019115005
Branch: CHD
```

- Follow the general process to `add`, `commit` and `push` the file *Verify online if your changes have been pushed*
- `checkout` to the branch `other_details` created earlier and create a file with the same name `other_details.txt` and add content in the following format (please not that this is in an order different from that in the main/master branch):

```
Branch: CHD
Roll: 2019115005
```

- Follow the general process to `add`, `commit` and `push` the file *Verify online if your changes have been pushed*

Pull requests and Merge conflicts

- Go to your browser and create a Pull Request to merge the branch `other_details` into `main` / `master`
- You will notice that there is a merge conflict
- Resolve the merge conflict to get content in any format you like and merge the pull request

Logs

- `git log` enables you to see the previous commits
- `git log --oneline` gives a better minimalistic view, `--graph` flag shows you the graph of branches across the timeline ..etc