College of Engineering

Computer Science & Eng. Dept.

Course: COE420

Software Engineering Lab

Date: 10th June 2020

Course Instructor: Dr. Khaled El Fakih

Email: kelfakih@aus.edu

Lab Instructor: Ms Hend ElGhazaly

Email: helghazaly@aus.edu

Lab 1: Version Control

- **Objectives:**
  - Introduction to version control methodology in Git and GitHub.
  - Creating GitHub accounts, organization, repository.
  - Pushing and pulling versions to the remote directory using Git.

- **Before you start:**
  - Agree on who will be *member_A, member_B, member_C, member_D,* in your team.
  - Download the report template from iLearn. Paste the screenshots as you are doing the lab exercises to avoid missing any required screenshots.

You should work **individually** on the lab but **submit only one group report** with all the screenshots.

Hand in: ***One team member needs to*** upload **one group** report on iLearn (following the template provided) that includes:

- Cover page with names and IDs of each team member.
- The link to the GitHub repository created.
- Screenshots from Git and GitHub in each exercise.

Due Date: Thursday 11th July, 11L59pm (5% per day will be applied on late submissions).

- **Useful resources:**
  - **Git documentation:** https://git-scm.com/doc
  - **About GitHub:** https://guides.github.com/
  - **Pro Git Book:** https://git-scm.com/book/en/v2

- **Unix Commands:**

```
ls -> List all files in the present working directory

cd path -> Change director to path

cd .. Move up by one directory

pwd -> Show present working directory
```

Exercise 1: Create accounts, organization and repository on GitHub (all team members)

- ➢ Each team member needs to create an account on *GitHub*.
- ➢ Only one team member needs to create an organization and a repository.

member_A is the owner of the repository and needs to invite his/her team members. ***member_B, member_C and member_D*** *need* to join the organization to work on the project.

The following is an example of a project details that will be used in the steps below:

Sample organization name: COE420-Hend-S20
Sample Repository Name: COE420Lab1
Team Members: Hend and Mohamed
Users accounts names on GitHub:
Hend: helghazaly
Mohamed: melghazaly

1. Create an account on GitHub (**all team members**):
    a. Browse [*www.github.com/join*](http://www.github.com/join) and create an account.

    Note that the username and organization's name have to be unique (not used before).

2. Create a new Organization (**member_A only**)

    While creating an account, check the box that asks *if you would like to create an organization account* and fill the details (e.g: organization name: COE420-Hend-S20). If you have not checked the option to create an organization account at sign up:

    Click on the + icon at the top right and select *New Organization*. Select the "***Team for Open Source***" plan and fill the details.

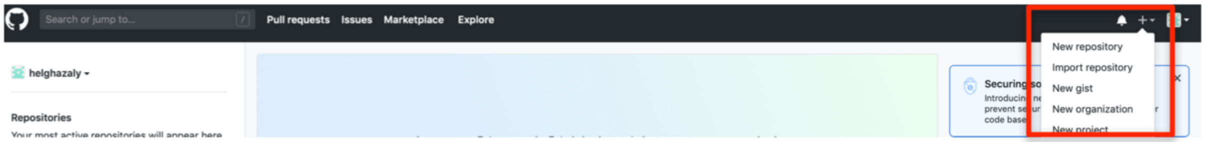
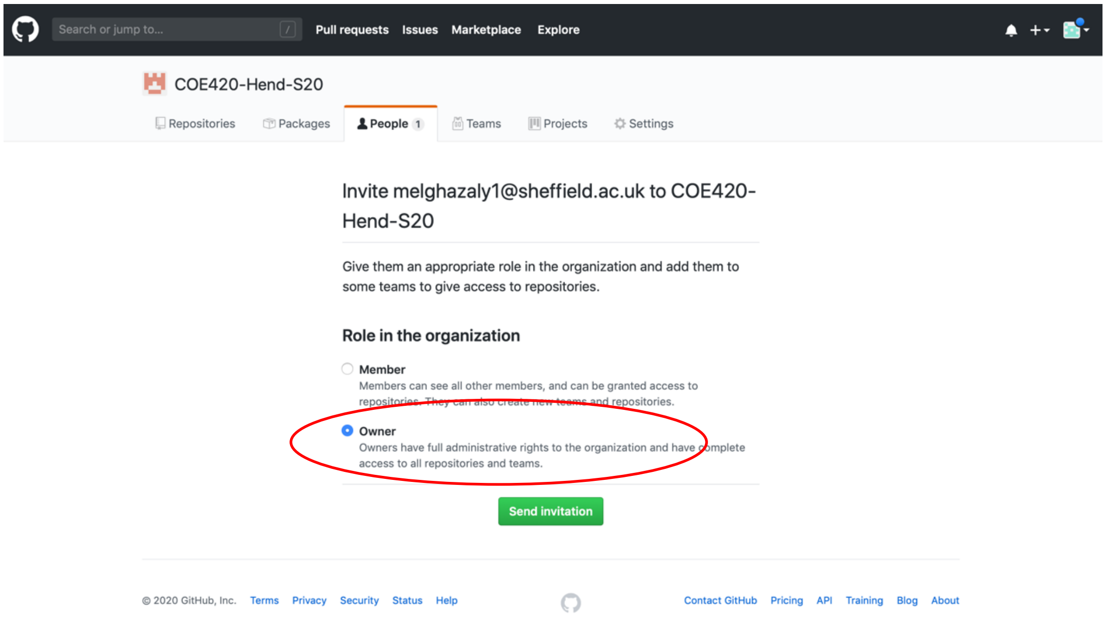


Figure 1: Creating an organization

3. Invite organization members **(member_A only)**

    a. Enter the username/Full name/email/ name of their GitHub account.
    b. Make the other members also **owners** of the organization (e.g. *member_B: Mohamed* also owner *COE420- Hend-S20*) (**Figure 2).**

An invitation will then be sent to the team members. Each team member has to accept the invitation by opening their email and accepting the invitation to join the organization.



Invite melghazaly1@sheffield.ac.uk to COE420-Hend-S20

Give them an appropriate role in the organization and add them to some teams to give access to repositories.

Role in the organization

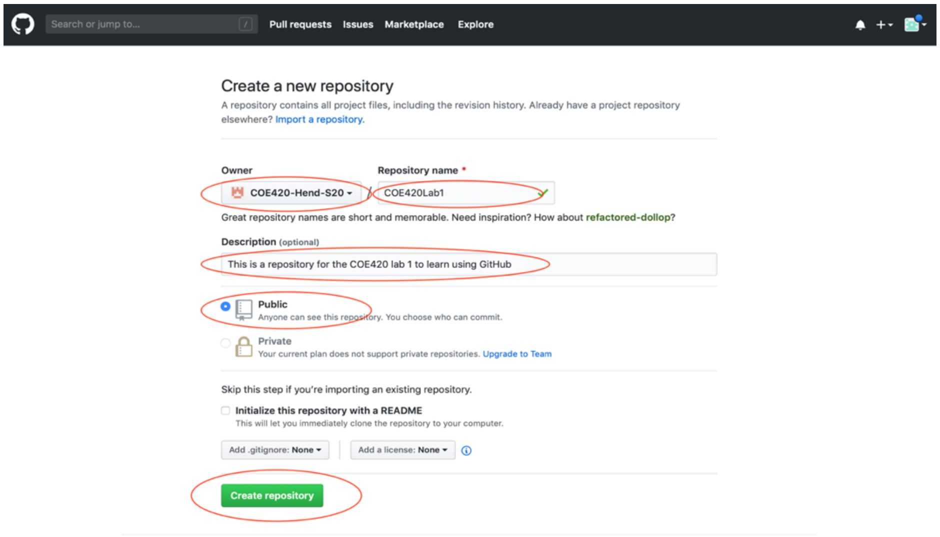
○ Member
Members can see all other members, and can be granted access to repositories. They can also create new teams and repositories.

● Owner
Owners have full administrative rights to the organization and have complete access to all repositories and teams.

Figure 2: Invite team members to join the organization and make them owners

4. Create a new repository within the organization created ***(member_A only)***

1. Go to your organization and click on "***Create a new repository***" or in the upper right corner click + and then select ***New repository***
2. Ensure that the owner selected is the name of the organization (*e.g. COE420-Hend-S20*)
3. Name your repository ***COE420Lab1*** and write *a* ***short Description***. The name should be unique to that particular member's account.
4. Set it to ***Public***
5. Click on *"**Create repository**"* button.



Create a new repository
A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner: COE420-Hend-S20 / Repository name *: COE420Lab1 ✓
Great repository names are short and memorable. Need inspiration? How about refactored-dollop?

Description (optional): This is a repository for the COE420 lab 1 to learn using GitHub

● Public
Anyone can see this repository. You choose who can commit.

○ Private
Your current plan does not support private repositories. Upgrade to Team

Skip this step if you're importing an existing repository.

□ Initialize this repository with a README
This will let you immediately clone the repository to your computer.

Figure 3: Create a repository within the organization

Exercise 2: Install Git (all team members)

1. Check if you have Git installed on your computer by running in the terminal:

   ```
   git --version
   ```

2. If you do not have it, download Git on your computer via this link: https://git-scm.com/downloads
   a. Click on your operating system and follow the installation instructions.
3. Run `git --version` in the terminal to verify that it is installed.

If you have windows you can also use Git Bash: https://gitforwindows.org/

Exercise 3: Configure your details (all team members)

Create your identity by setting your name and email. This helps in knowing who did what changes to the code when you view all the commits.

1. To configure your details run in the terminal:

   ```
   git config --global user.name yourName
   git config --global user.email yourEmail
   ```

   Modify the commands with your details

- Add in the report screenshot of configuring your details

2. To check the details saved, you can run:

   ```
   git config --list
   ```

Exercise 4: Initialize a Git repository (member_A)

1. Create a folder (Lab1) on your computer.
2. Download **Calculations.java** file on iLearn and save it in that folder.
3. In the terminal, navigate to the folder where you saved **Calculations.java**
4. Initialize a git repository by running in the terminal:

   ```
   git init
   ```

   This command will generate a hidden **.git** directory for your project, where Git stores all internal tracking data for the current repository.

Exercise 5: Initial Commit (stage, commit code) (member_A)

In this exercise, you will stage, commit the *Calculations.java* file and push it to the remote repository. You should do these steps every time you want to commit.

1. In the terminal, run the following command to get the status of the repository and observe the output.

   ```
   git status
   ```

   You should see *Calculations.java* in red under the Untracked files listed.

2. Place the Calculations.java file in the *staging* area to commit it.

   ```
   git add Calculations.java
   ```

   You can also use `git add .` *to add all the files in the project folder to the staging area instead of adding multiple files individually.*

3. Check the status again

   ```
   git status
   ```

   You should see Calculations.java in green under Changes to be committed.

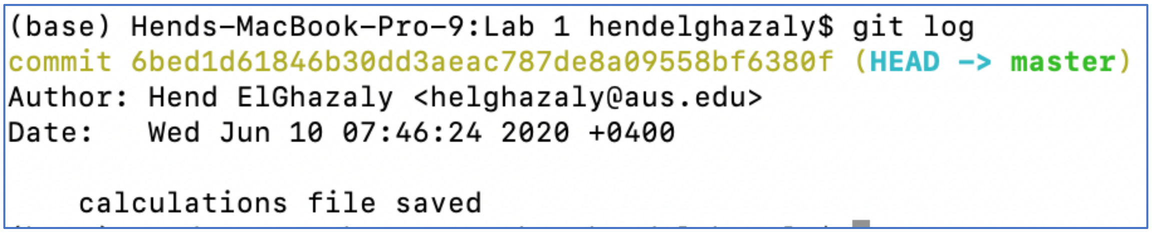
4. Commit the file from the staging area to save it in the local repository.

   ```
   git commit -m “initial commit”
   ```

5. Check the commit details by running:

   ```
   git log
   ```

   You should see a similar output:



```
(base) Hends-MacBook-Pro-9:Lab 1 hendelghazaly$ git log
commit 6bed1d61846b30dd3aeac787de8a09558bf6380f (HEAD -> master)
Author: Hend ElGhazaly <helghazaly@aus.edu>
Date:   Wed Jun 10 07:46:24 2020 +0400

    calculations file saved
```

Note: The above steps (staging and commit) need to be done every time you want to commit changes to the local repository.

6. **6.** Push the local repository content to the remote repository for the other team members to see it.

   a. Run the following to point your local repository to the remote repository:

   ```
   git remote add origin https://github.com/COE420-Hend-
   S20/COE420Lab1.git
   ```

   Modify the link to the name of your organization and repository

   b. Push the local repository to the remote repository.

   ```
   git push origin master
   ```

   Note: Since you are pushing this to remote repository, a *sign-up* window pops up and you need to login to your account.

- ➢ Add in the report screenshots showing the commands were run and screenshot from GitHub showing the commit made (you should see it in the repository in GitHub under commits)

Exercise 6: Pushing changes via Git (member_B, member_C, member_D)

The following steps need to be done by the rest of the team members:

1. Each member needs to clone the repository *COE420Lab1*. Navigate to the project folder (e.g. `cd U:/`) then run:

```
git clone https://github.com/COE420-Hend-S20/COE420Lab1.git
```

Modify the link to the name of your organization and repository

You should then be able to see Calculations.java file in your project folder.

2. Complete the code in *Calcualtions.java* according to your member number (e.g *member_B* implement the *subtraction*) and push changes made back to the repository:

a. Open Eclipse or any editor.

b. Add the feature (e.g. s*ubtraction*) to the Java program.

c. Save the file.

3. Push the local repository after adding the feature (e.g *subtraction by member_b*)

```
cd COE420Lab1
git add .
git commit –m "Subtraction implemented"
git push –u origin master
```

Sign in to your GitHub account if the window pops up

Note: You may need to pull changes (Exercise 7) before pushing the changes.

- ➢ Add in the report screenshots by each team member showing that they made the commits and pushed the changes.

Exercise 7: Getting Updated Version (all team members)

Each team member should do this step before pushing their changes. *Member_A* should also update his/her local repository with the features added by the other team members:

1. pull changes to get the updated version:

   ```
   git pull origin master
   ```

2. Open Calculations.java file and check that the features are added by the team members

- ➢ Add in the report screenshots by each team member showing that they got the most updated version by running the pull command.