USK00 - Let's Git!

Last edited on : Thu Mar 12 14:45:49 +06 2020 by @yaakovazat

Author: Yaakov Azat

UID: 0f40b3be-643e-11ea-bc55-0242ac130003

Branch: Alpha

License

Preface

Let me tell you a story

I know that you have alot of questions about this material (I suppose that you are a curius super girl~), one of them maybe, "What is Git and why we use it?". Before I tell you more about Git and Github, I wanna talk you a recent story of mine.

Before I tell you more about Git and Github, I wanna talk to you about a recent story of mine. The time was last year, I was preparing my master thesis. I did a lot of research and written almost a hundred pages of the article, then all of a sudden, my computer shut down, I was focusing too much on the work and hadn't found that the computer is already out of battery! Then, it's not hard to image, I lost my last changes to the article, not all of them but almost latest 10 pages. After that, I opened my computer, open the "Autosave" feature of the Microsft Office, but that does not solve all of my problems:

1. the ability to save all my changes to che document that I am editing.
2. the alibility to editing the same document with my friends.
3. the ability to label a change
4. the ability to give a detailed explanation of why a change was made
5. the ability to move between different versions of the same document
6. the ability to undo change A, make edit B, then get back change A without affecting edit B

The version control tool, Git, can do all of those things - *and more*!!! (bet you didn't see *that* coming!) So have I sold you yet on the awesomeness that is Git? I hope so, cause we're about to dive into it in the next section.

Quick Start

About Git

Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is easy to learn and has a tiny footprint with lightning fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows. [1]

Download and Install Git

You can check your git version by typing the command below on your terminal (You can check wheather you have installed Git or not by this command too):

```
git --version
```

If you see something like:

`git version 2.21.1 (Apple Git-122.3)` on Mac OS or `git version 2.17.1` on your linux computer, then it means `git` has already been installed, if not, please refer the installation instructions below:

Linux

Note : We recommand you to use Ubuntu, Raspbian OS, Kali Linux or Other Debian Based Linux Distributions for this class.

Open your favorite terminal and type:

```
sudo apt install git
```

Mac OS

There are several ways to install `git` on Mac:

1. **GUI installation** :
    1. Open https://git-scm.com/ and click `Download 2.23.0 for Mac' :

2. Open your downloads folder and double click to open the `git-2.23.0-intel-universal-mavericks.dmg` file you just downloaded.
3. The install process is the same as other programms and ignored

2. **Terminal installation with brew**
    1. Install `brew`: paste these commands to your terminal and click `Enter`

    ```
    /bin/bash -c "$(curl -fsSL
    https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"
    ```

    2. Install git with `brew`, type `brew install git` in your terminal and press `Enter`

Windows

To install Git on windows:

1. open https://git-scm.com/download/win on your browser and git should downloaded automatically, if not, click `Click here to download manually` as below:

2. When you've successfully started the installer, you should see the **Git Setup** wizard screen. Follow the **Next** and **Finish** prompts to complete the installation. The default options are pretty sensible for most users.
3. **Select Git Bash during installation**

That's all for installing git.

Configuring Git

Configure user information for all local repositories:

1. To set the email you want attached to your commit transactions:

```
git config --global user.name "[name]"
```

for example I will do this on my computer:

```
git config --global user.name "Yaakov Azat"
```

2. To set the email you want attached to your commit transactions:

```
git config --global user.email "[email address]"
```

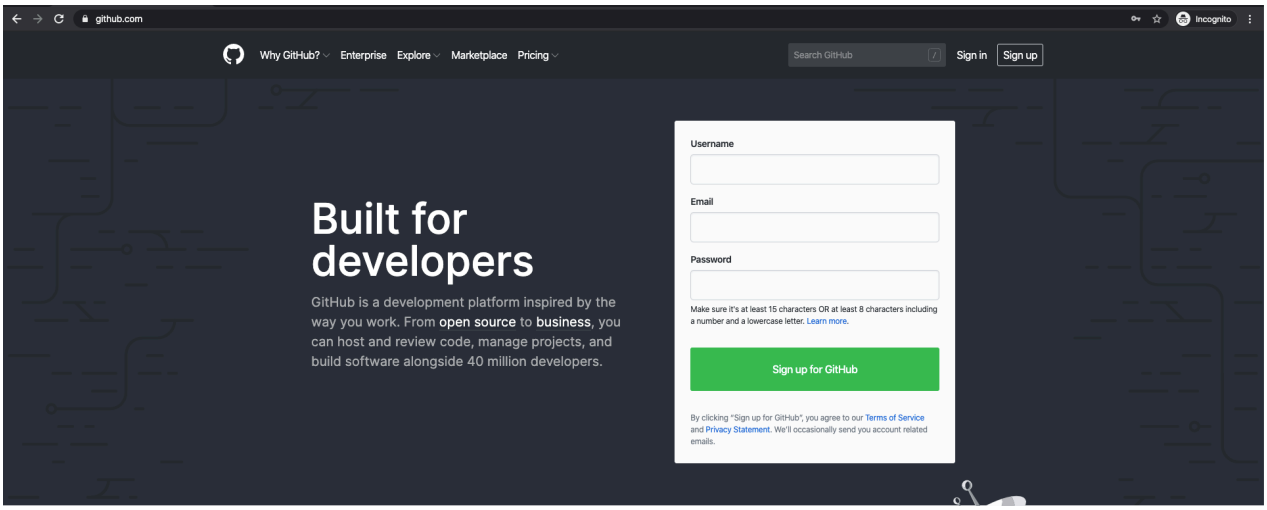
for example, I will do this for myself:

```
git comfig --global user.email "yaakovazat@gmail.com"
```

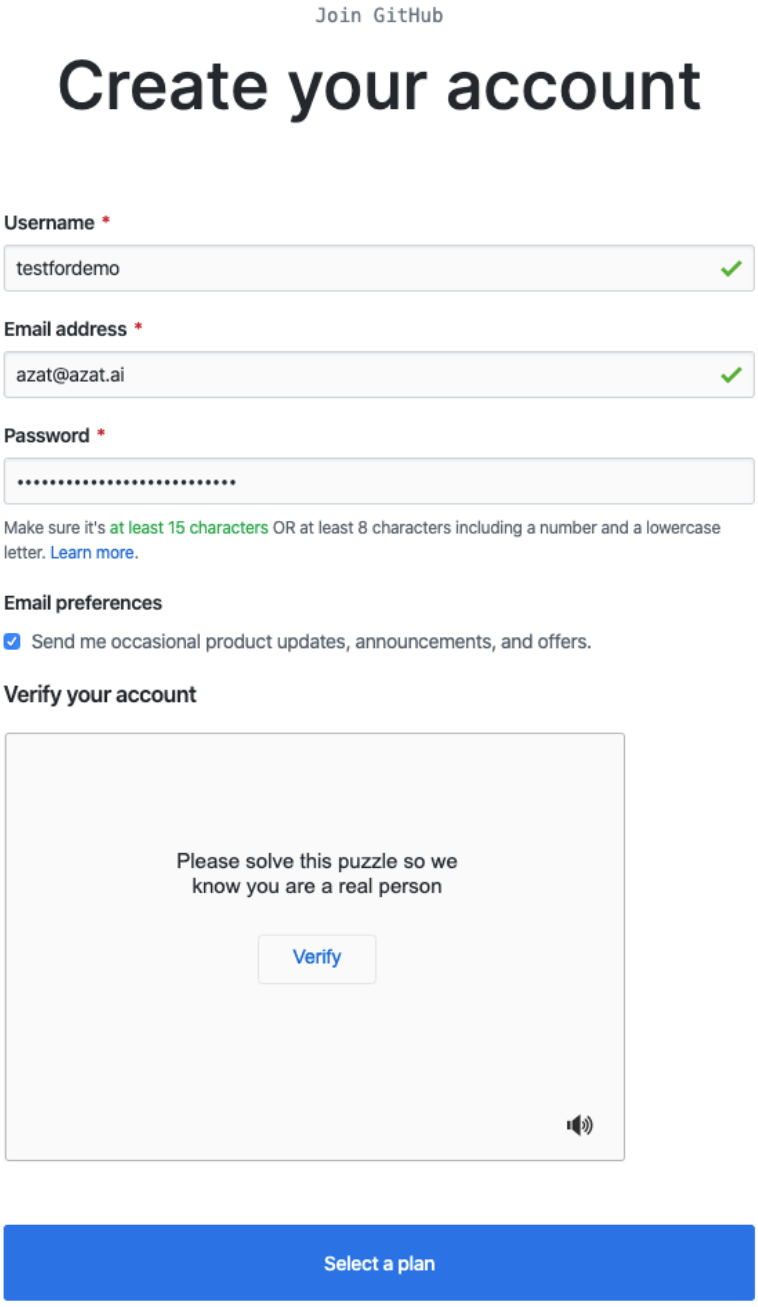
Registering on Github

Github is a platform that provides hosting for software development version control using Git (We will talk about this very in detail later on the course.

- To register your Github account, first you should open Github homepage (https://github.com/) on your browser.



- Click `sign up` on the right corner for registration.



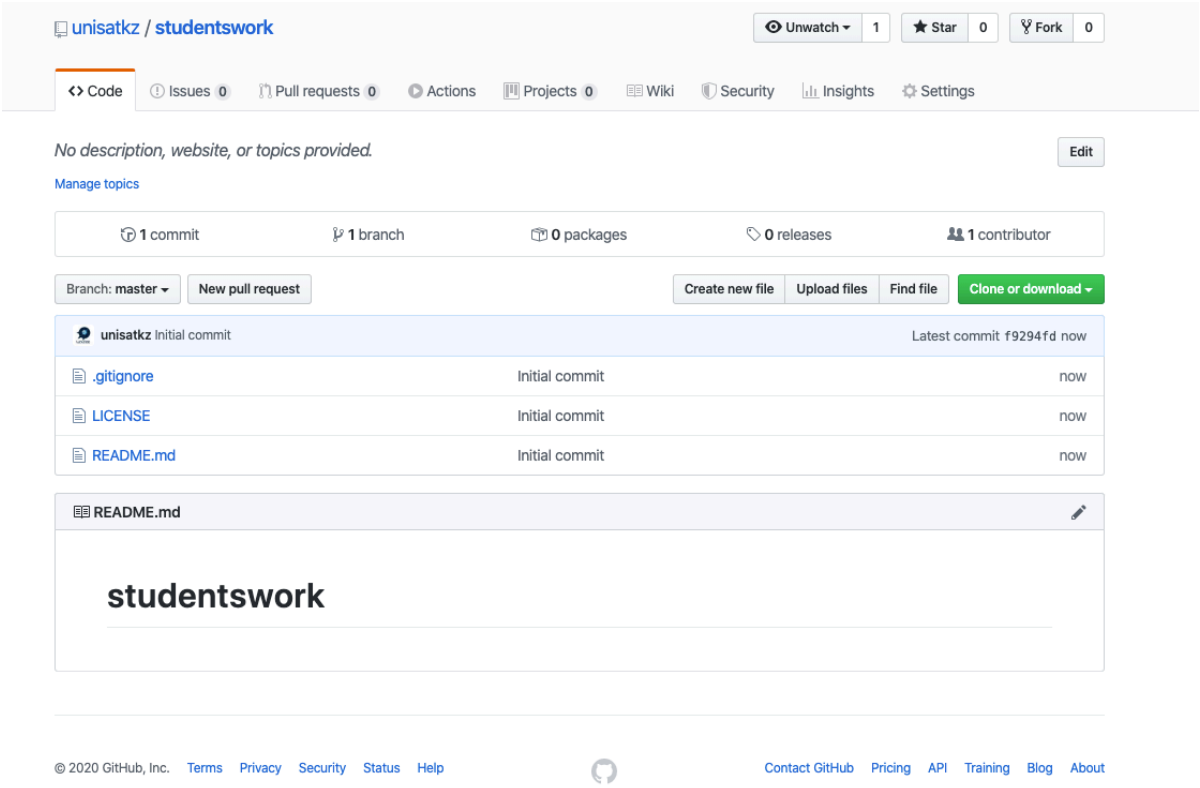
- Pick a nice `username`, enter your email and fill the password.

- Next, you need to verify your identity.
- Then, finally, you can click `Select a plan`
- Select the **free** account and that's all.

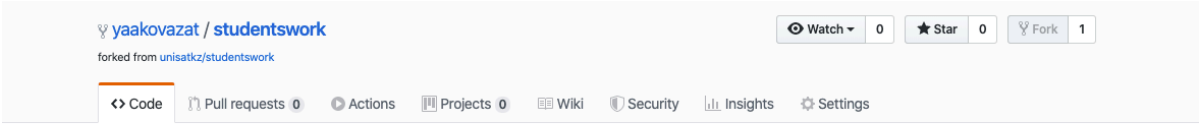
Using Git

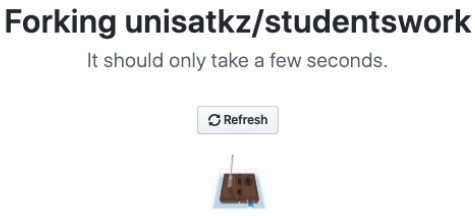
Fork the `UniSat` repo:

- Open https://github.com/unisatkz/studentswork on your browser.



- Click the star icon and then click `fork` .





Clone the `UniSat` Repo:

To submit your content to the repository, you should first clone the repository to your local computer, to do this, open your terminal (or git bash) then type:

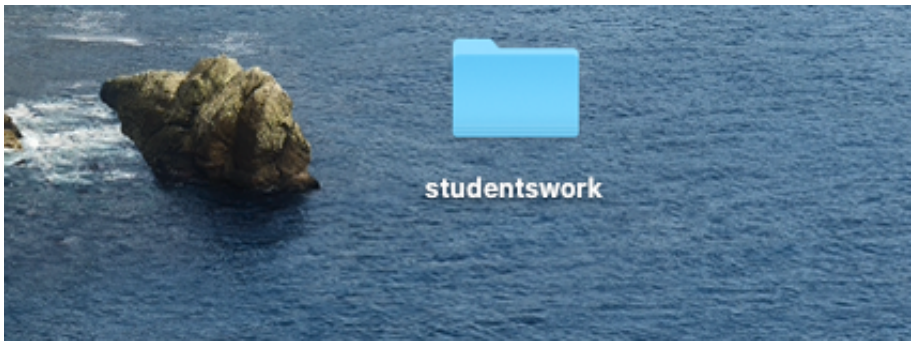
```
git clone https://github.com/[yourgithubusername]/studentswork
```

!!! YOU SHOULD REPLACE [yourgithubusername] ABOVE WITH YOURS !!!

for me, I will do this as below:

```
git clone https://github.com/yaakovazat/studentswork
```

This will clone the repository to your desktop:



Open the repository folder, drag or copy your documents in to the `studentswork` folder,

Add all of your documents that should be added

Add the changes.

After putting all the staff inside the repository folder `studentswork`, now you are able to push it to your Github, to do so, **first enter the `studentswork` folder in your terminal** (seach how to enter a folder in Windows/Linux/Mac terminal if you don't know) , and then type this command:

```
git add .
```

Comment the changes

You can comment your changes, that means your changes now meaningful with some labels so that you can easily understand whats done for your documents/codes later on.

To do this, type this:

```
git commit -m 'Any thing to comment for the change'
```

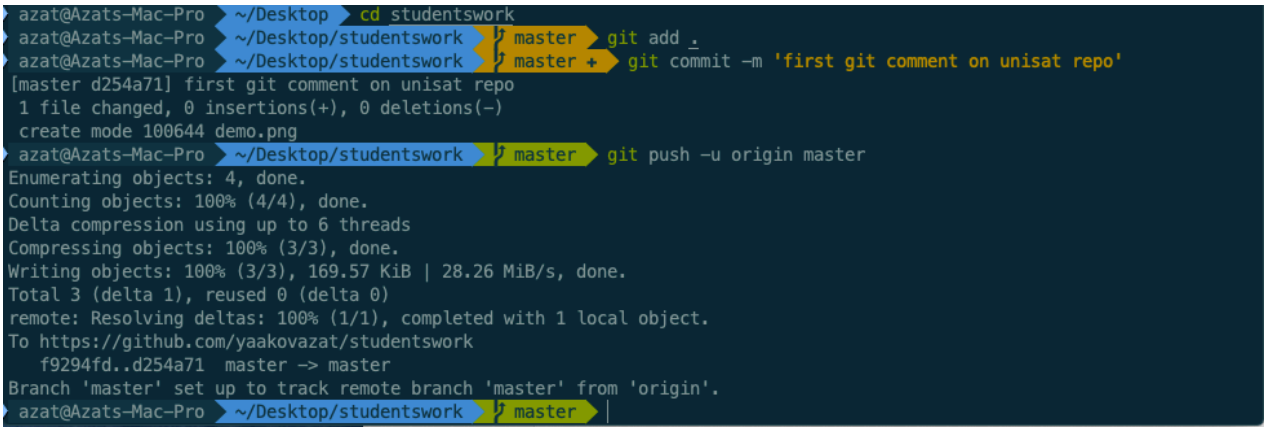
I will write this for example:

```
git commit -m 'first commit using github'
```

Push the changes

To save the changes on your Github account, you should push the repository with this command:

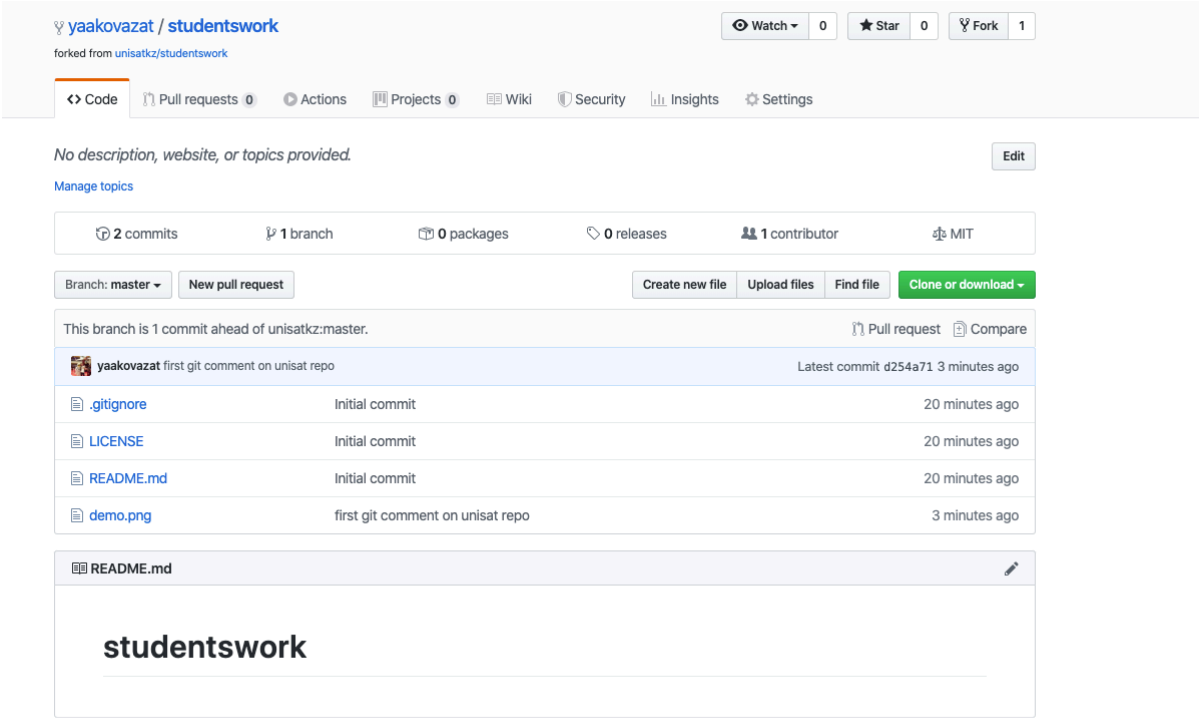
```
git push -u origin master
```



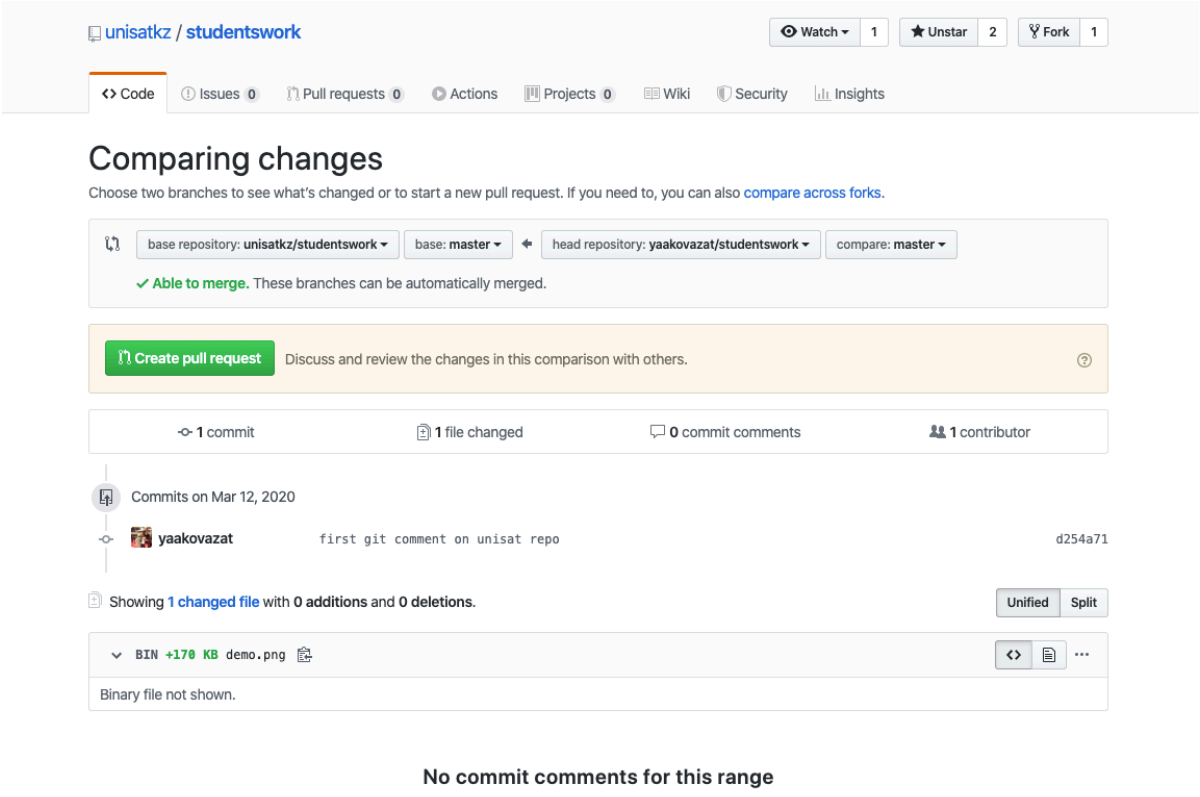
Pull Request

After finishing the steps above, you should able to upload all of your publishing documents to the Github repository. In the next step, we are going to contribute and share our changes with other `UniSat` team members, to do so, we should create a `pull request`

- open yout Github account and open the repository `studentswork`



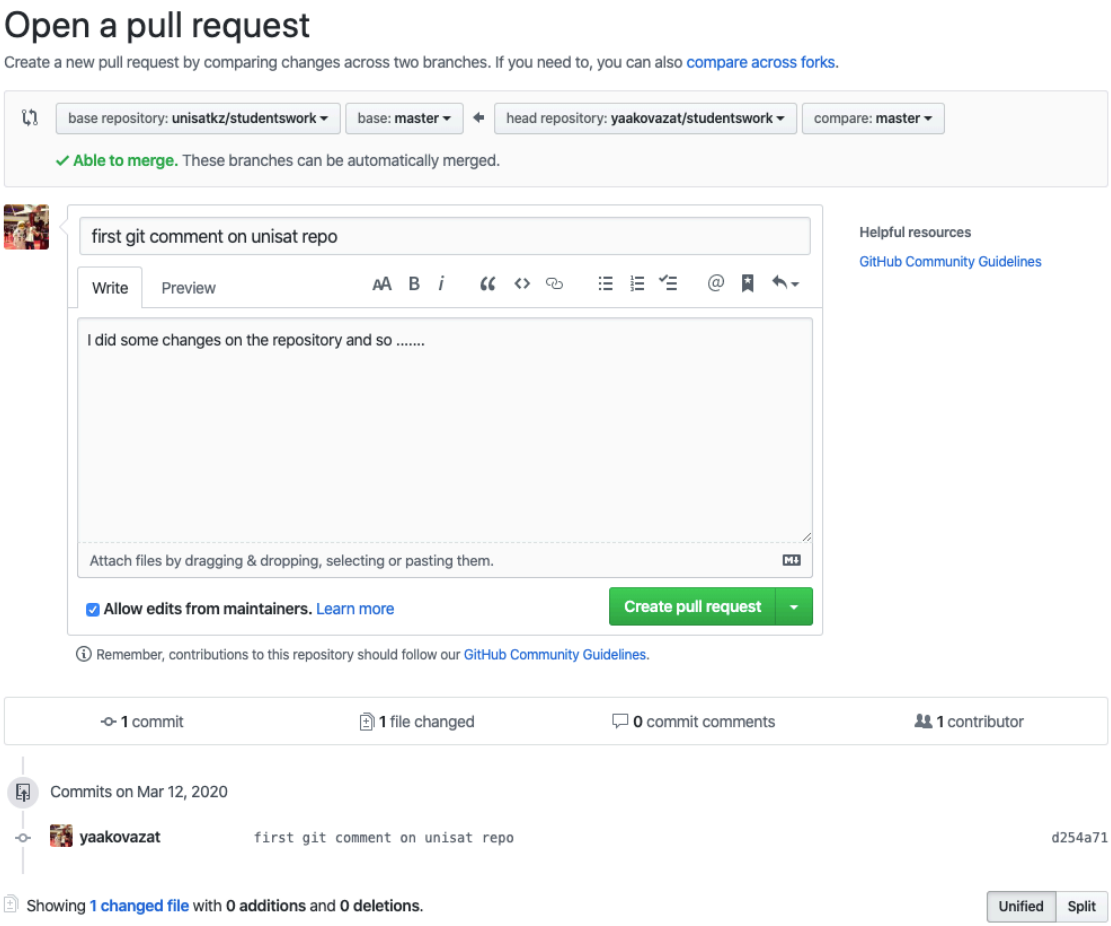
- click on `New pull request` button:



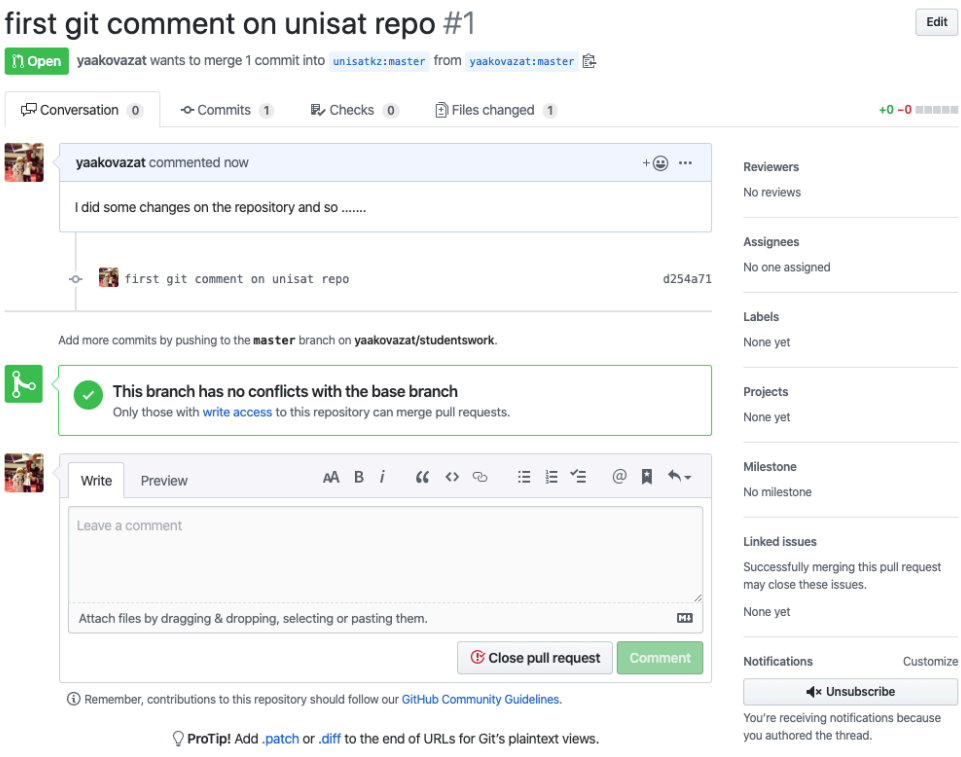
No commit comments for this range

This will send our changes to the main repository.

- Click the green 'Create pull request' button:



Write some comments if you want, after finish typing, just click `Create pull request` again:



That's all!

If you are lucky, the UniSat Github repository admin will soon reply to your comments and merge your work to the main repository.

NOTICE: THIS IS ONLY THE PART OF THE WHOLE WORK AND PUBLISHED AS A BETA. YOU DO NOT HAVE TO WORRY IF YOU were NOT ABLE TO FINISH THESE TASKS, YOU WILL BE taught VERY CONCRETEly ON THE CLASS.