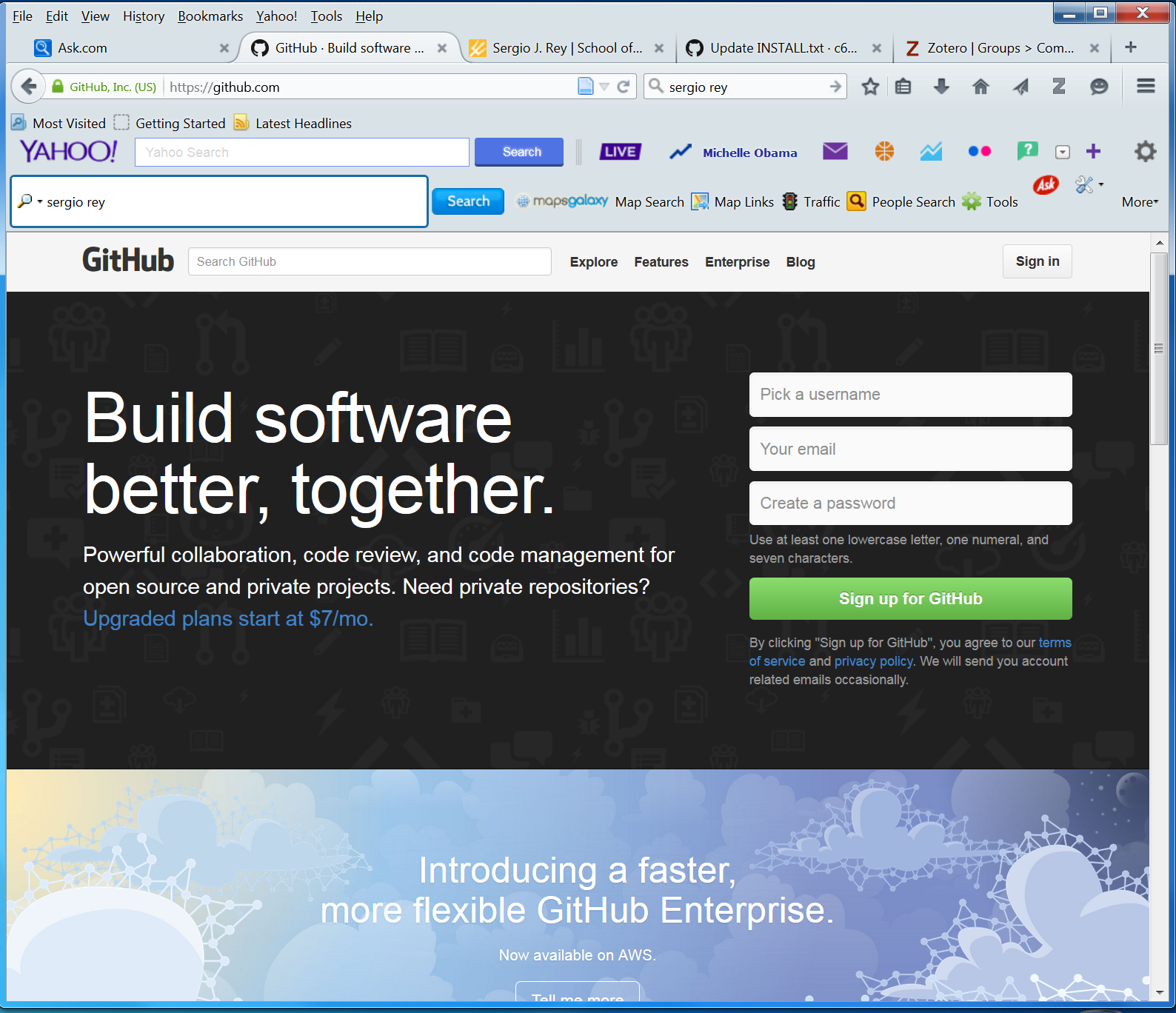
**Tips on how to set up a GitHub account: Note: First we introduce web- & command line-based GitHub (Pages 1~8). If you are interested in GitHub Windows, go to Page 9.**

1. Go to the website <https://github.com/>, you will see the following page:



**Figure 1: The GitHub main webpage** (before you create an account and sign in)

Then choose your username, enter your email address, and create a password. Then click the green button Sign up for GitHub.

1. Choose your GitHub plan.

Once you complete Step 1, you will be prompted to choose a plan. Below is an extract from the GitHub website that may help you make the decision:

“GitHub provides two types of plans: free plans and paid plans.

Both plans have the exact same features. They can have any number of public repositories, with unlimited collaborators.

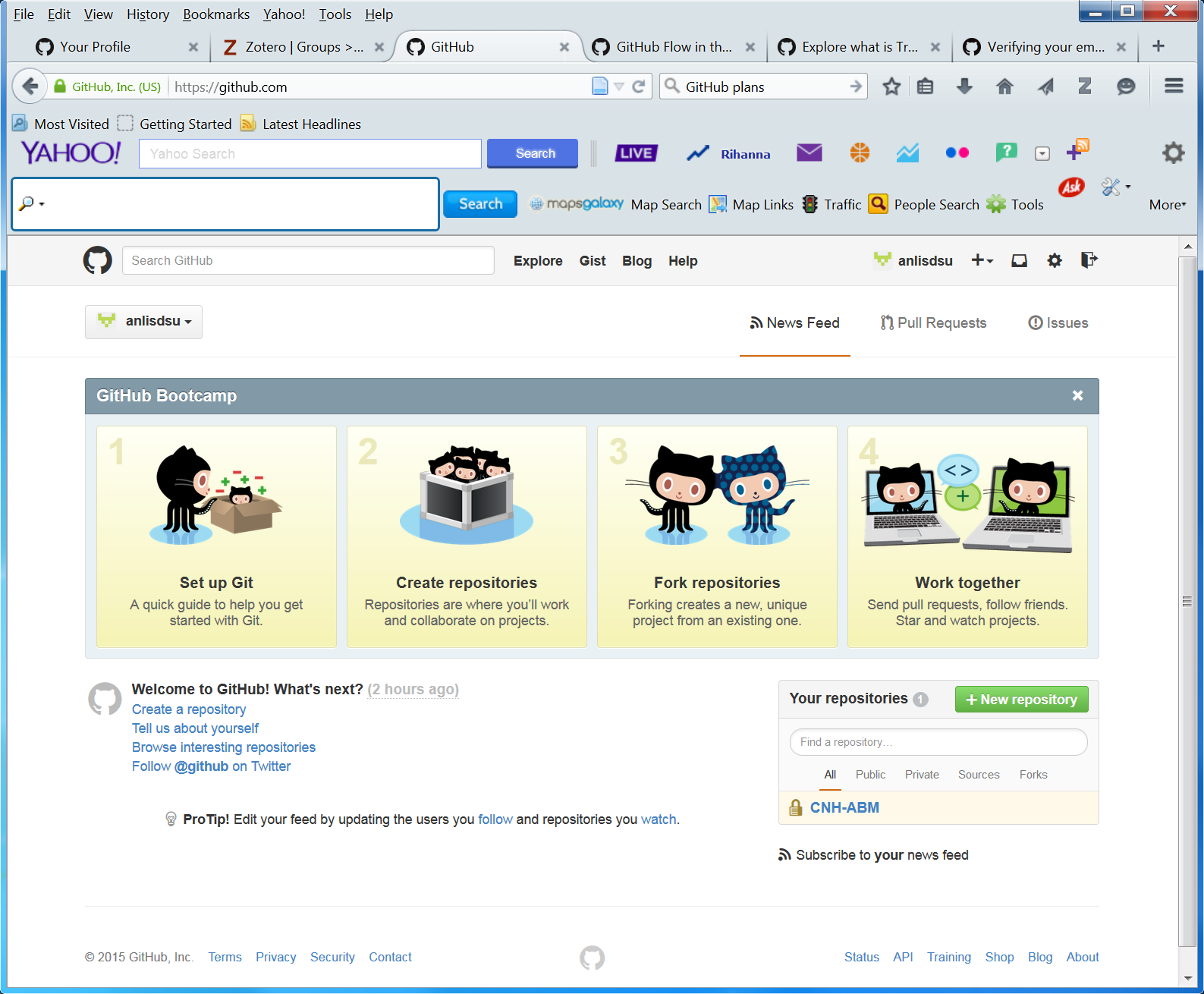
However, paid accounts can create private repositories. Public repositories are viewable by anyone, while private repositories have their visibility limited to just you and your collaborators.

The number of private repositories available is determined by specific paid plans. You can see all of our plans and pricing at github.com/plans. You can upgrade or downgrade your plan at any time.”

Therefore beginners may choose a Free plan. Later you can upgrade to any other plan(s).

1. Start use of GitHub.

You can go back to the GitHub website (note to use GitHub, all you need is a web browser) at <https://github.com/>. Click the Sign in Button on the upper-right corner of the page (when prompted, enter your email and password). Then you will be able to access the GitHub main page (Figure 2).

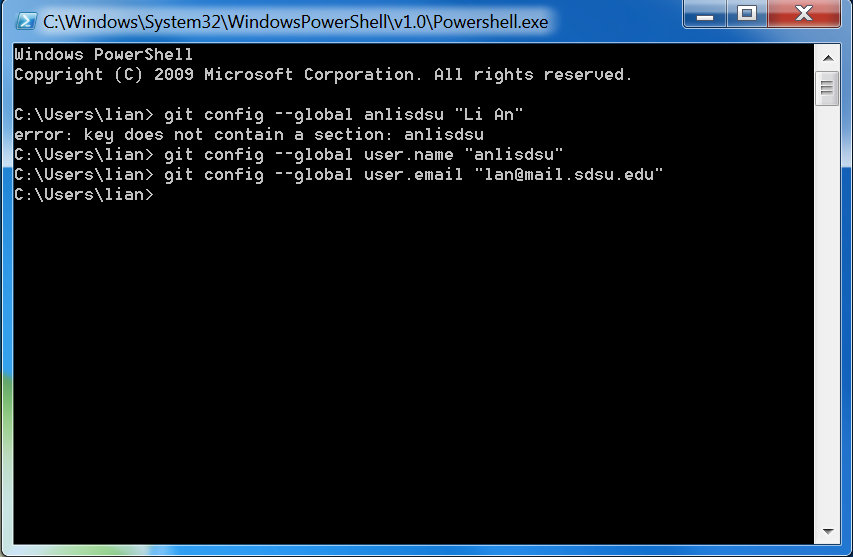


**Figure 2: The GitHub webpage** (after sign-in)

Before we start using GitHub, I recommend you click on the four steps on the above page, i.e., 1. Set up Git, 2. Create repositories, 3. Fork repositories, and 4. Work together. Then read these instructions and understand basic concepts such as repository (a place to store, save, edit, and retrieve your code and work with other people) and fork repositories (a fork is a copy of a repository for proposing changes to someone else’s project).

* 1. Sign up Git.

Click the Set up Git page (see Figure 2), you will get to a page with four steps. Simply follow the instructions for installing Git, open Git Shell, and telling the system your information.

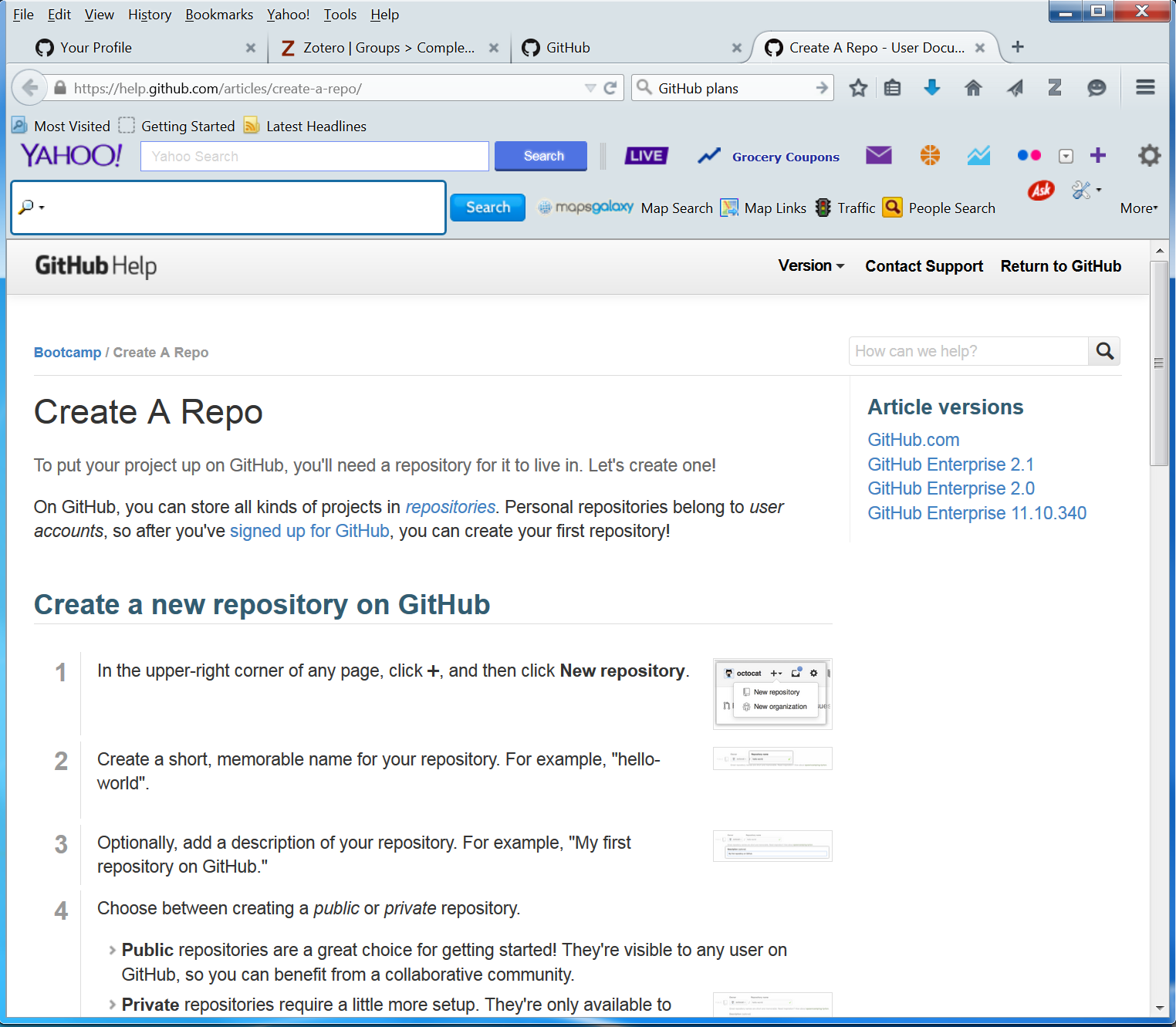


**Figure 3: The Git Shell window after installation of GitHub for Windows.**

Once the installation is done, two icons (one for Git Shell and one for GutHub) will be placed on your desktop (also a log file icon is placed, but I moved it to a folder inside). The above graph (Figure 3) shows Steps 3 and 4 after installation of GitHub for Windows. Note the GitHub username and email address should be in quotation marks.

* 1. Create repositories.

Click the button for 2. Create repositories (Figure 2), and you will see a page for instructions like the page below (Figure 4):



**Figure 4: Webpage for instructions on how to create a repository.**

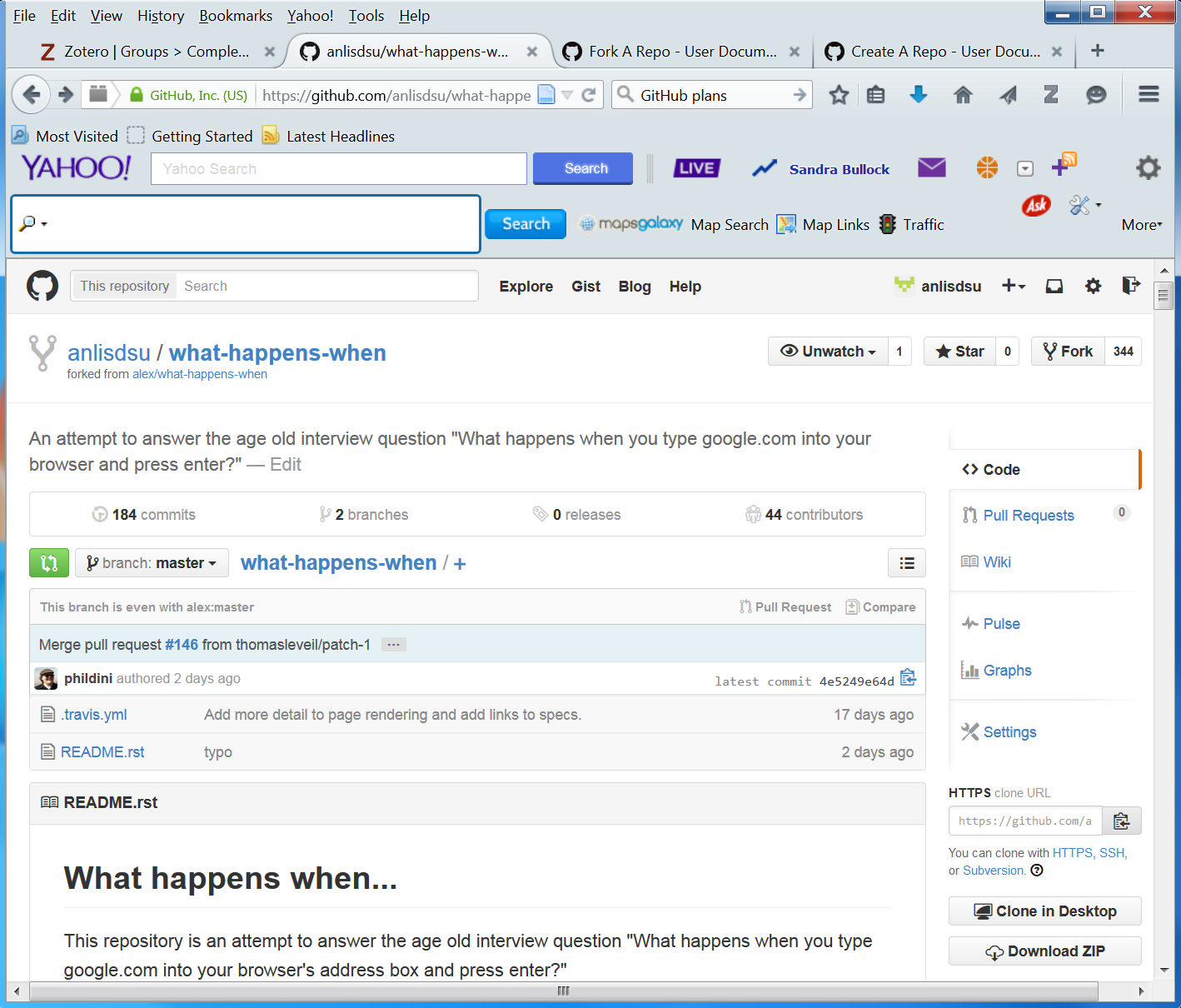
But be aware that the instructions are given on the basis on the main webpage <https://github.com/> (the same as Figure 2). Follow the instructions on this page. Here I created a repository named **CNH-ABM**, which you can see from near the lower-right bottom of Figure 2. This was created to be private (only shared with people within the group); let us create one that is public, which is named **ABM-Modules**.

The next thing to do during this step is to commit a change. Automatically, GitHub creates a README.md file. Click on this file, and write something on the box. Then scroll down to the end of the page, click the green button “Commit changes”. Then all we have written will be saved to the file README.md. There are other types of files you can change and commit.

* 1. Fork repositories.

The idea of fork repositories is to either propose changes to someone else's project or to use someone else's project as a starting point for your own idea. Therefore for practice purpose, let us go to the main page and click on Browse interesting repositories (near the lower-left corner). Among the many repositories, let us hit alex/what-happens-when as an example.

In the top-right corner of the page under alex/what-happens-when, click Fork. Then a pop-up window shows the progress of downloading the repo and everything inside it. When this process is done, you will see a page like below (Figure 5):



**Figure 5: Outcome of a fork opertation: Another person’s repository downloaded to my repository**

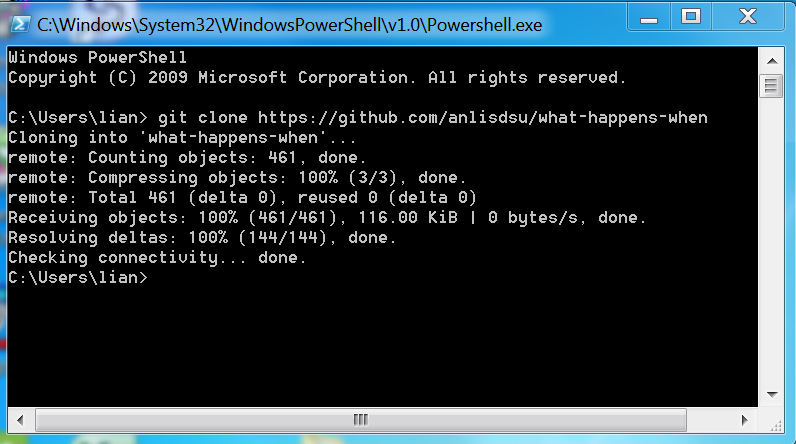
From here we can open a file (such as README.md file), click the pen icon (for editing purpose), add a few words, and then hit the green button “Commit changes”.

So far we have a fork of the alex/what-happens-when repository, but we do not have the files in that repository on our local computer. Below we clone a version to our local computer following the steps below:

1. On GitHub, navigate to your fork of the alex/what-happens-when repository.
2. In the right sidebar of your fork's repository page, click  to copy the clone URL for your fork.
3. Open the command line (i.e., double click the icon  Git Shell that is often on your Desktop for Windows users).
4. Type the following command to clone:

git clone https://github.com/anlisdsu/what-happens-when

Then you will see the following progress (Figure 6):

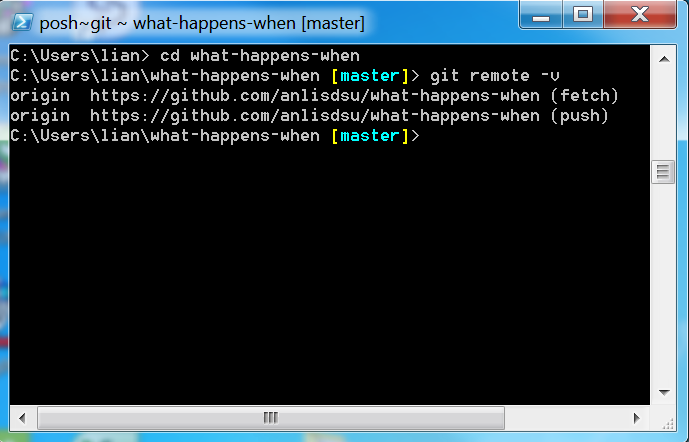


**Figure 6: The clone process (downloading a repository to a local computer)**

After doing this, let us verify it by going to C:\Users\lian at my computer: Yes, there is a folder what-happens-when there, which contains two files: one is .travis.yml, and the other is the readme file.

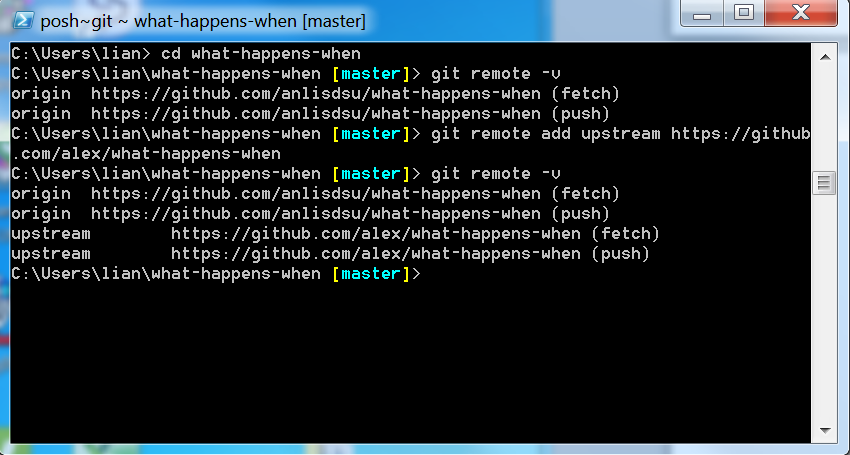
1. Next sync our fork with the original what-happens-when repository:

[Same as Steps a-c above: On GitHub, navigate to our fork of what-happens-when repository. Then on the right sidebar of this fork's repository page, click to copy the clone URL for your fork. Open the the command line]. Go to the directory for what-happens-when (now C:\Users\lian\what-happens-when), and type the command: git remove –v, you will see the following:



**Figure 7: The current configured remote repository for our fork**

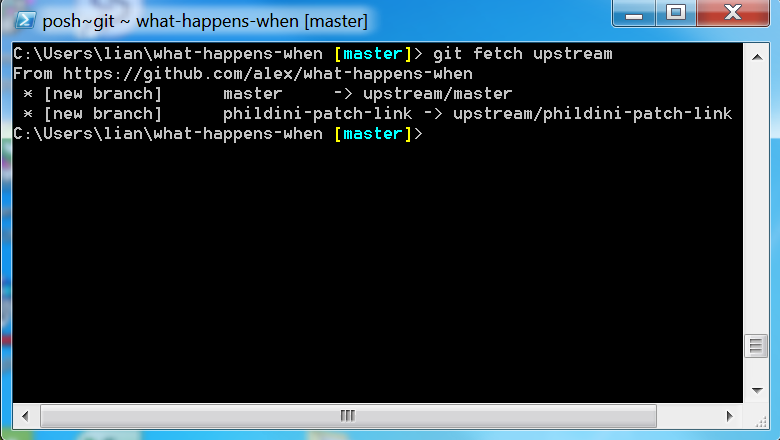
Then type git remote add upstream/ https://github.com/alex/ what-happens-when, then all should be done to set up the upstream source. To make sure this, type again git remote –v, you will see the changes made thus far (Figure 8):



**Figure 8: The outcome of setting up the new upstream repository**

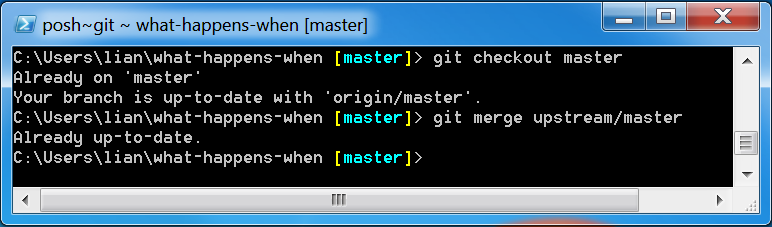
1. Then we can do the sync work

Still go to the terminal (for Windows users), stay at the same local directory (here C:\Users\lian\what-happens-when), and type git fetch upstream to fetch the branches and their respective commits from the upstream repository. We will see the following outcome after some time (Figure 9):



**Figure 9: The command and outcome for sync.**

Then we should type git checkout master to check out our fork's local master branch and git merge upstream/master to merge the changes from upstream/master into your local master branch:



**Figure 10: The outcome for merging upstream changes to local branch.**

* 1. Work with other people

Click 4. Work together (see Figure 2), there are a bunch of things we can do, such as Follow People on GitHub (thus get notification about what s/he does), Watch a Project (say up-to-date with a project), Pull Requests (letting the original authors know the changes we add or make), and Issues (often telling the HitHub community some problems that need to be fixed).

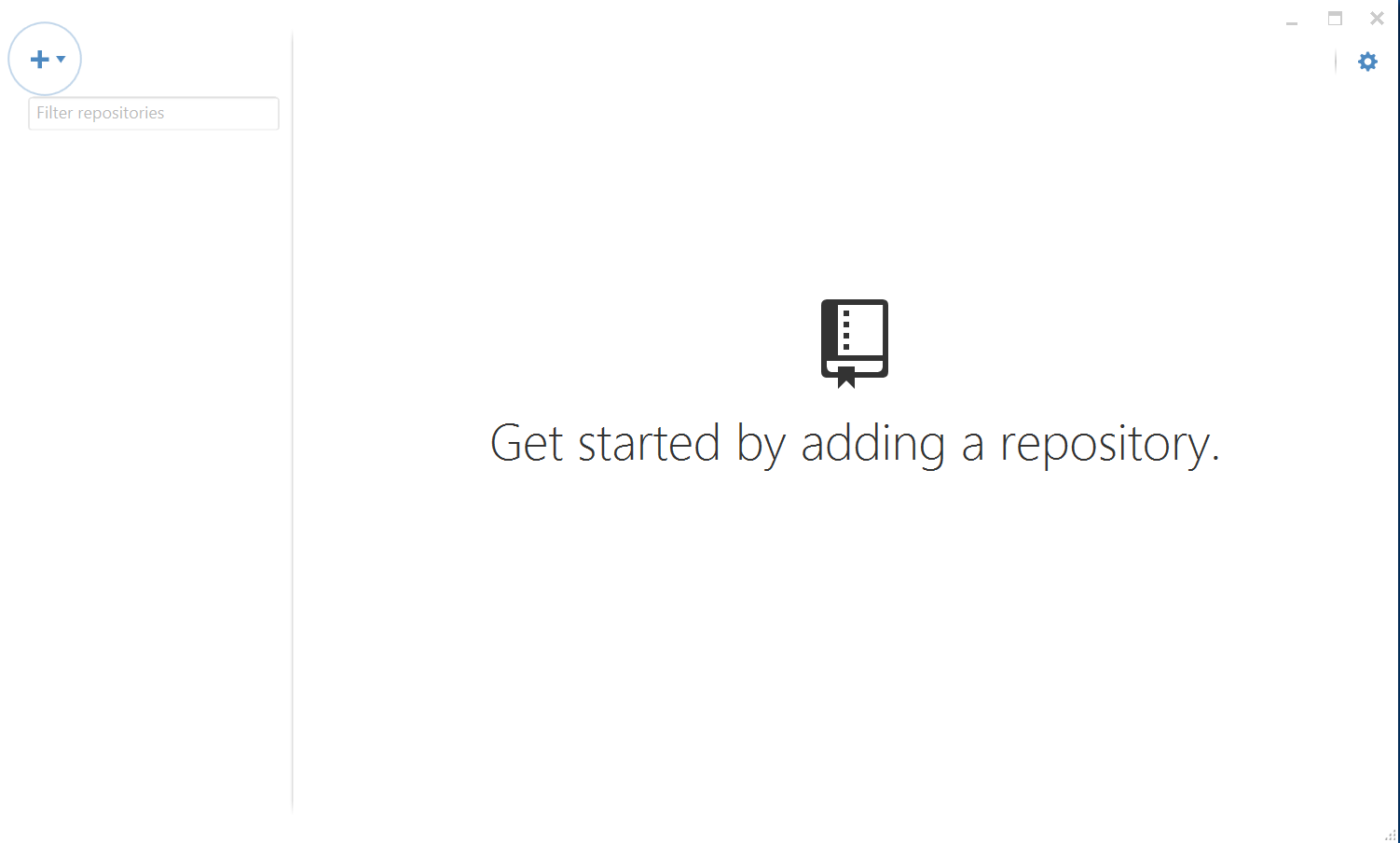
**Below we introduce installation and use of GitHub for Windows.**

1. Download software

Go to <https://windows.github.com/>, click the button Download GitHub for Windows. When prompted, hit Save File. If you are a Windows user, the GitHubSetup.exe file is often saved under your Downloads folder (for me it is C:\Users\lian\Downloads). Double click this exe file and follow instructions to install the software.

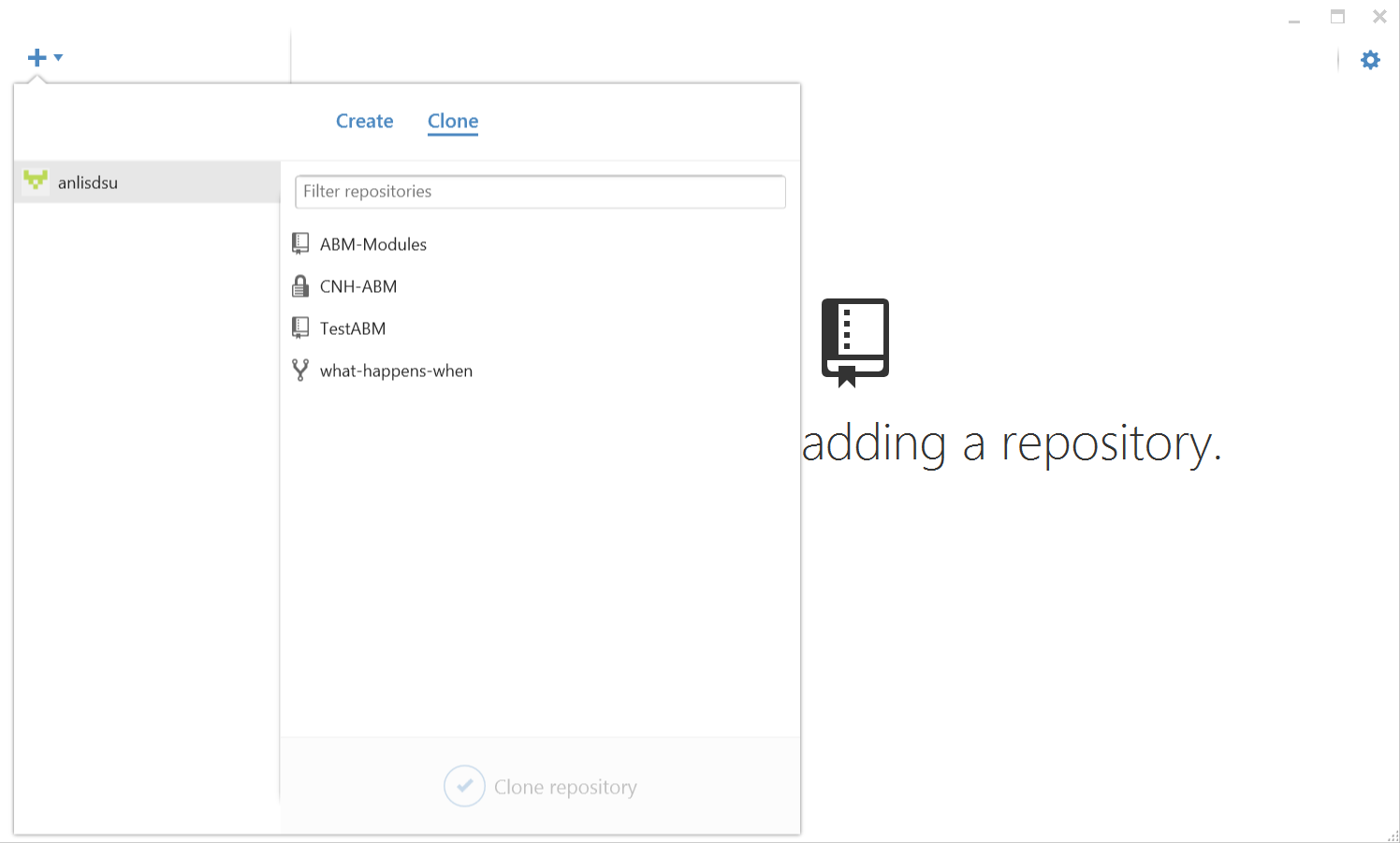
1. Use of GitHub (for Windows).

Once the download and installation is done, you should see an icon  on your computer’s desktop. Double click it, you will see a window pop-up like this (Figure 11):



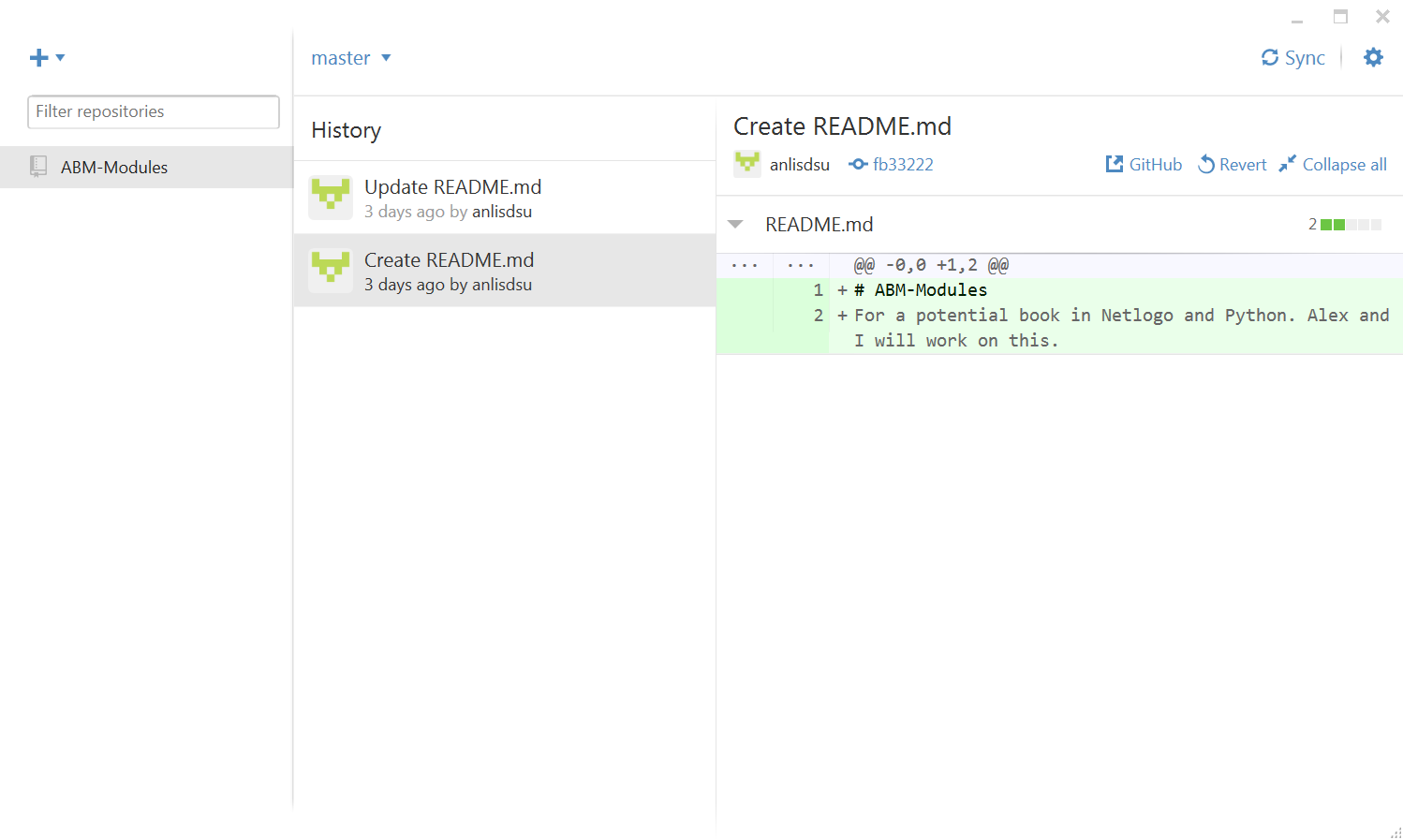
**Figure 11: The window of GitHub for Windows.**

Click on the top-left corner (the + sign within a circle), you will see two options there: Create or Clone. Because we have created a few repositories earlier, let us choose Clone first. Once the Clone button is clicked, you will see all the repositories that have been created in the past



**Figure 12: Clone an earlier produced repository.**

Choose ABM-Modules, for instance, from the list, and then click the check sign (within a circle) near the bottom of the window. Then you will be asked to pick up a directory to clone the repository into. Then you will see what is inside the repository (Figure 13):

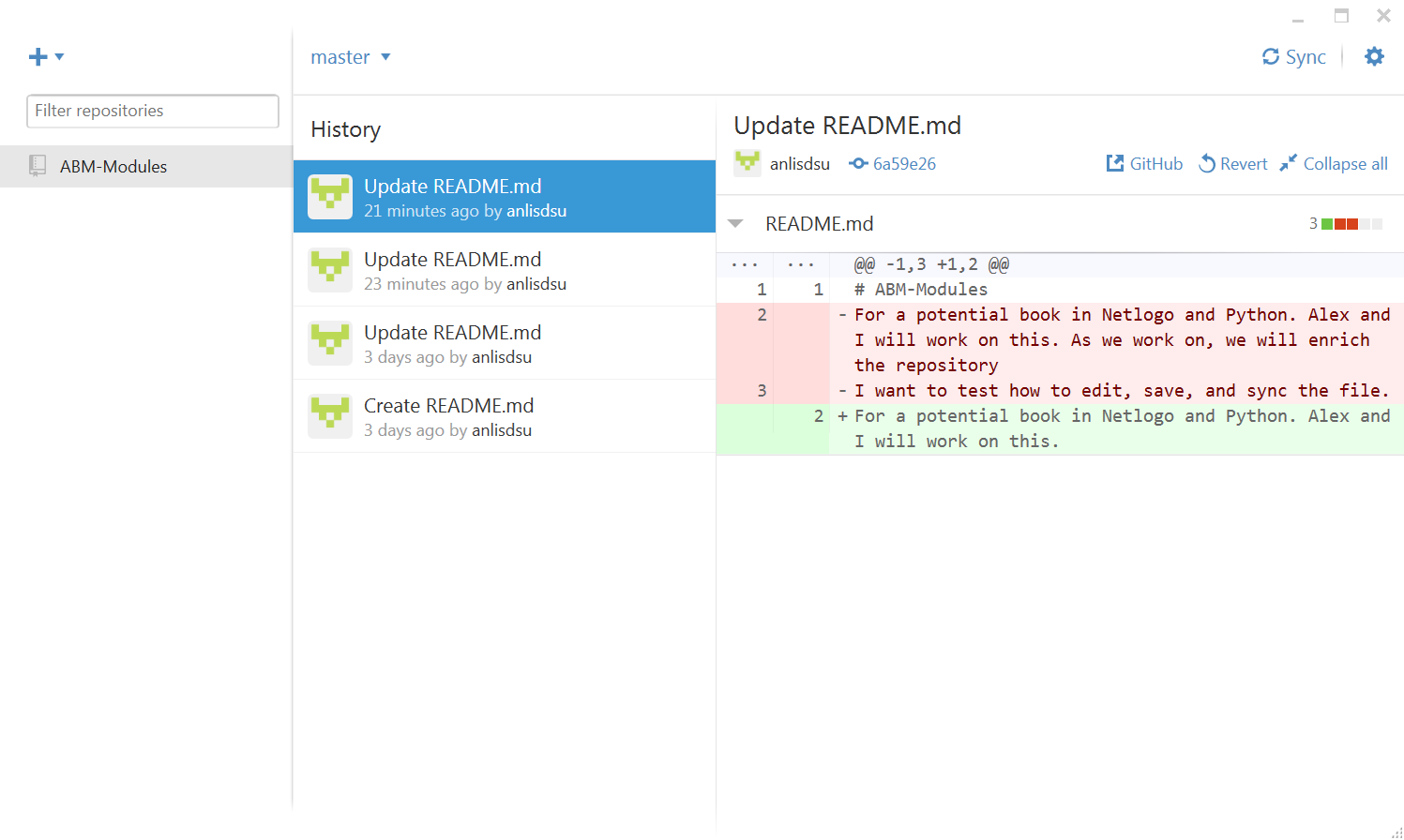


**Figure 13: The cloned repository and its content**

From here we can see the history of the README.md file on the right of the panel with different colors representing changes over time, where times of changes are recorded. If we want to edit the md (or whatever) file, simply go to the top-right button (the one to the left of the Sync button), you will see to drop-down menu. Check the View on GitHub, then a GitHub webpage about the repository (anlisdsu/ABM-Modules) will come out.

To make changes on README.md, click on the file name, and then hit the pen. Then in the editing panel, add a sentence “I want to test how to edit, save, and sync the file”, scroll down the page, and hit the green button named Commit changes. This way the change will be saved. Then delete it and commit the change again.

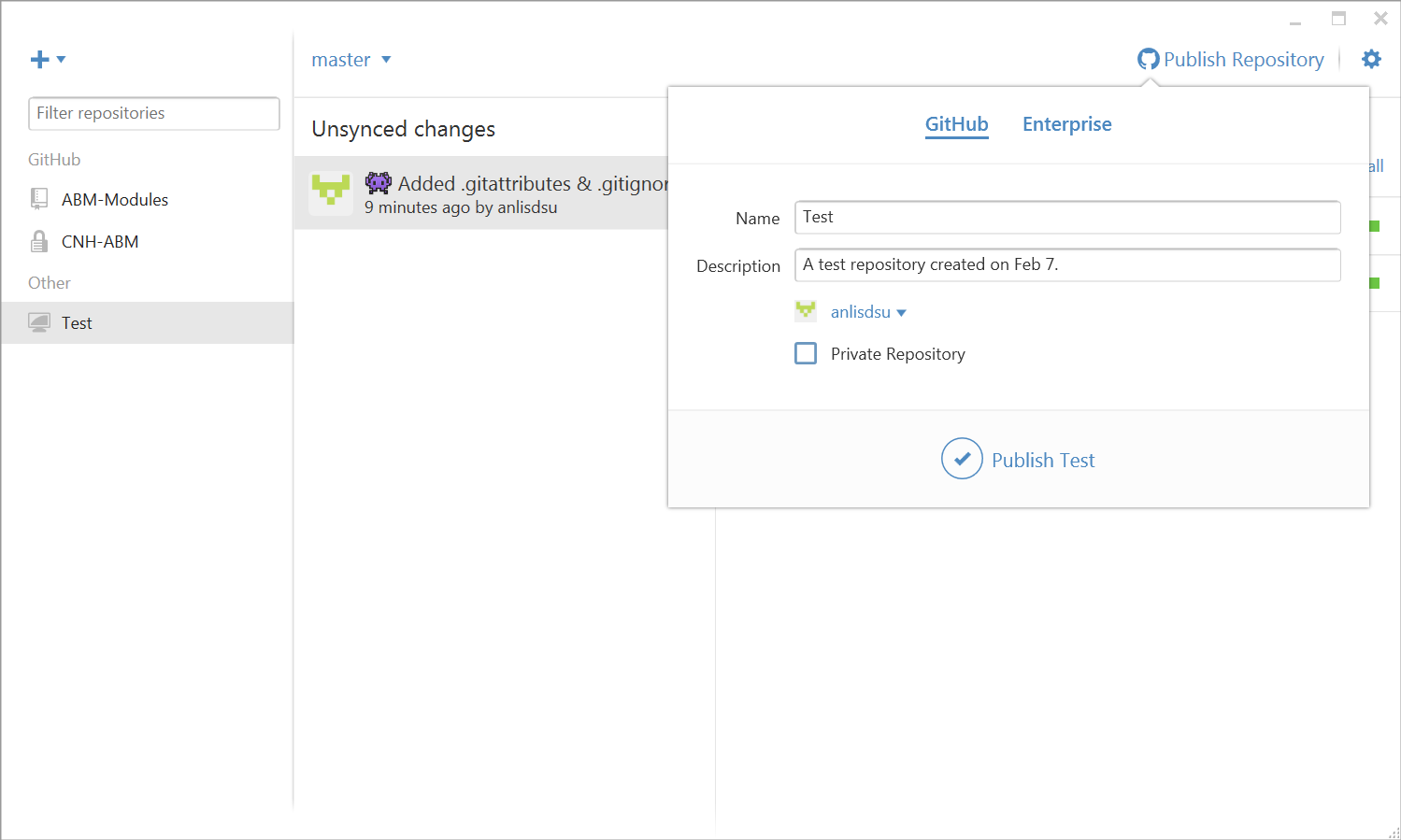
To see these changes from your desktop GitHub, hit the Sync button, and you will see all the changes made (Figure 14):



**Figure 13: The outcome of Sync with the online GitHub contents.**

Similarly we can clone other repositories from the GitHub page, for instance CNH-ABM (a private one we created last time) that was created earlier.

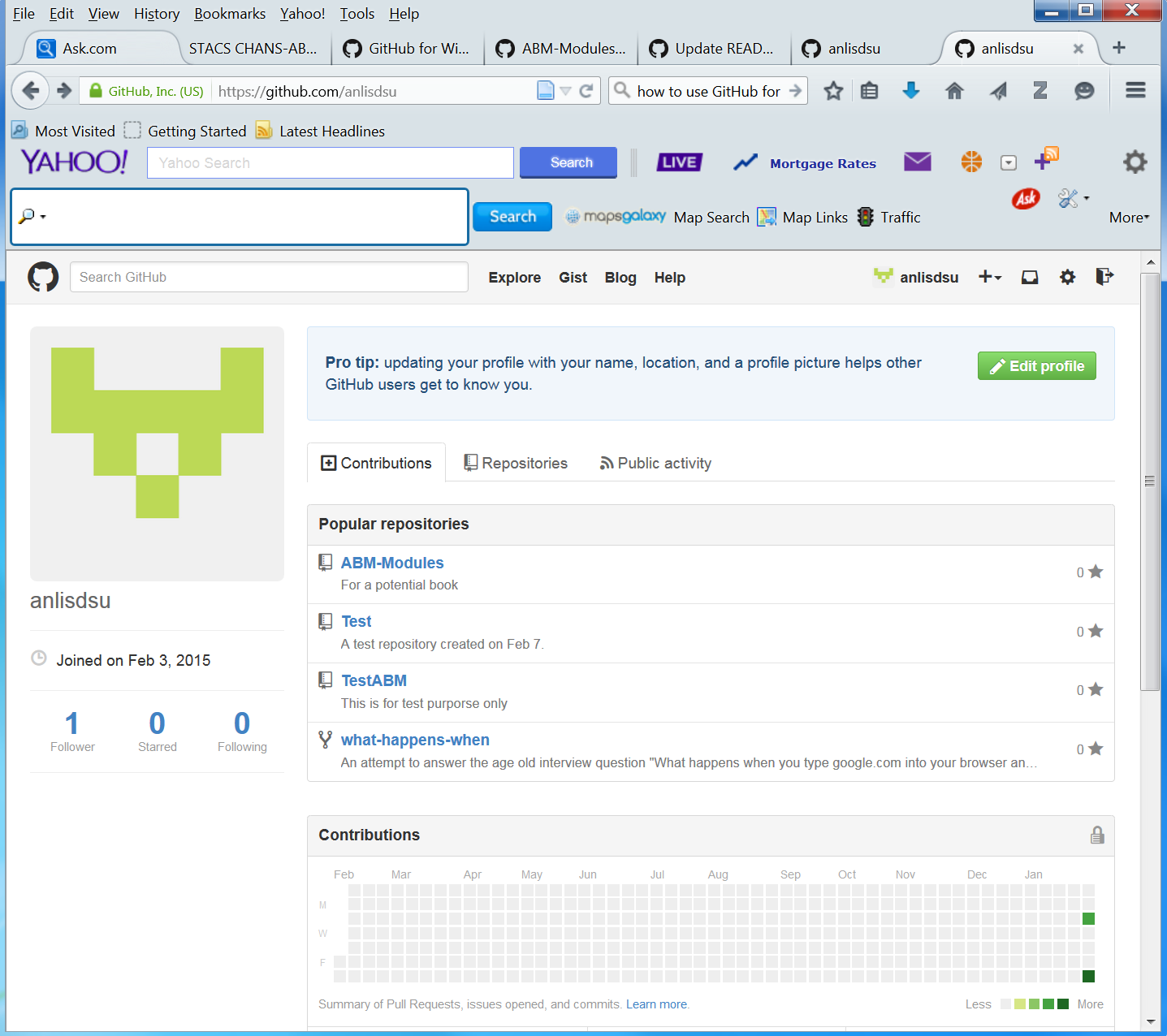
Alternatively you can choose the Create option (next to Clone). Give a name for the repository (say Test), and you can choose the local path unless you want to use the default (C:\Users\lian\Documents\GitHub) path. Hit the button Publish Repository, and from the drop-down men choose, put a description, specify the GitHub username (here anlisdsu next to the green icon), you will see the following panel (Figure 14). Then click Publish Test.



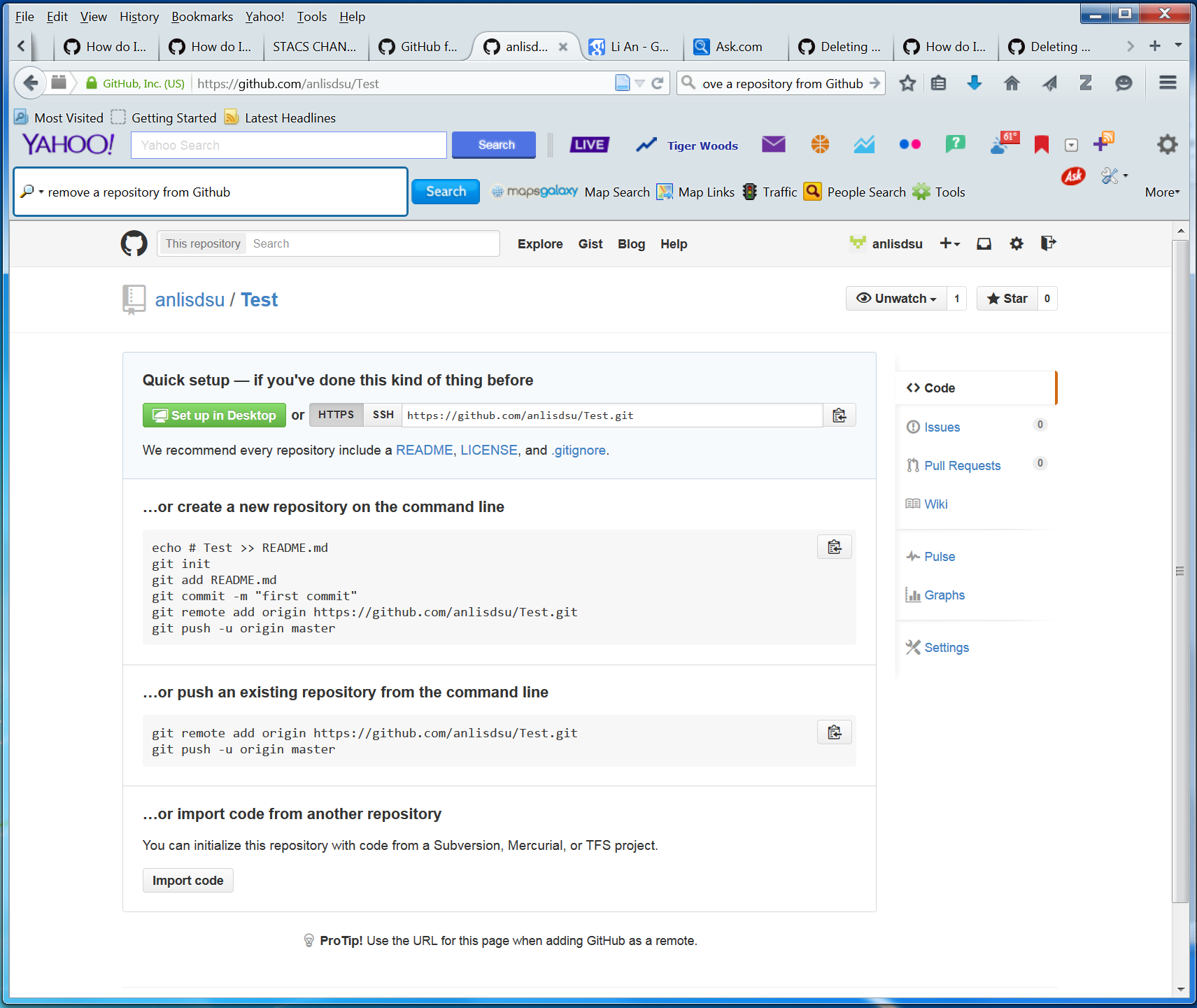
**Figure 13: Create a repository from GitHub Windows.**

To make sure this repository is uploaded, we can go to the GitHub web <https://github.com/anlisdsu> and check whether this is successfully uploaded. The repository named Test should be there (Figure 14). Up to this step, we should be able to make use of the power of GitHub and develop models together with other modelers.

Just one little thing: If you want to **delete a repository**, say the above Test repository, simply go to the repository (i.e., <https://github.com/anlisdsu/Test>), in the repository action bar, click Settings (see the right side of Figure 15). Under Danger Zone, click Delete this repository. Then after verifying what you want to do, you can delete it.



**Figure 14: The updated GitHub page after publishing a repository created from GitHub Windows.**



**Figure 15: Deleting a repository**

Now let us practice how to use GitHub to write and contribute to a program.

1. Write a program.

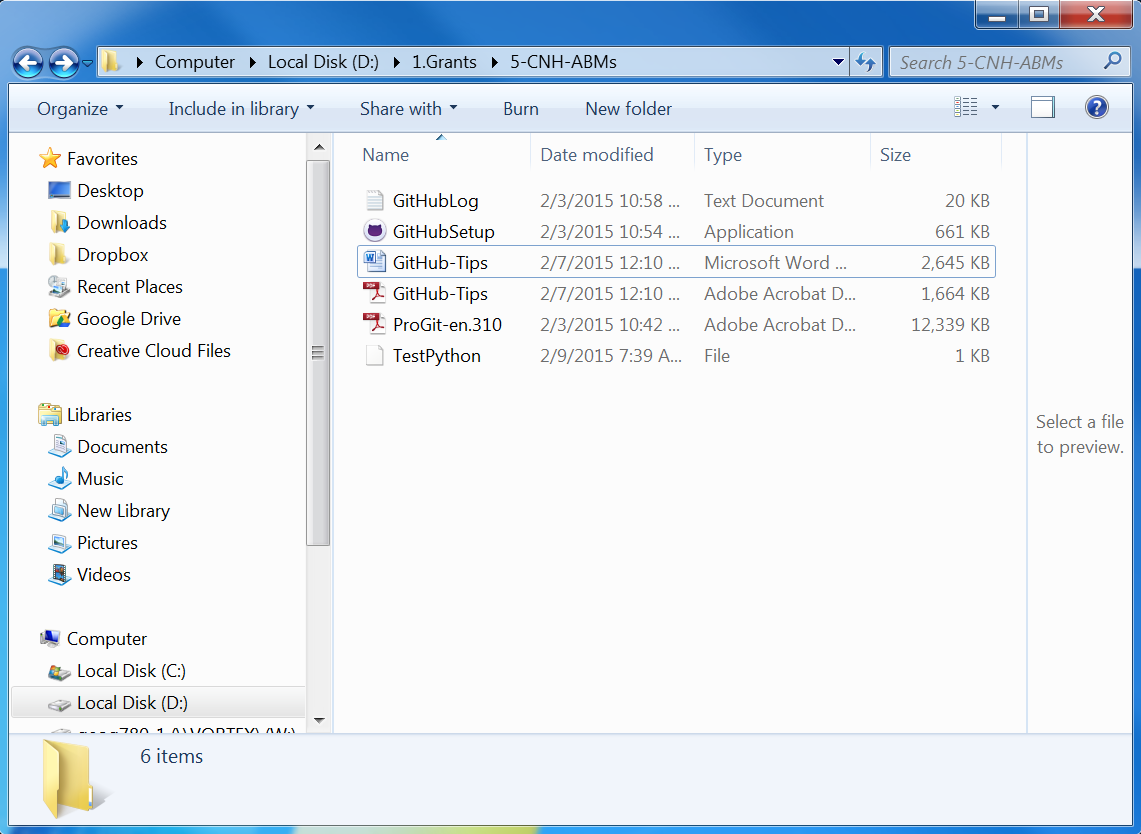
Let us write an extremely easy one is to write the following under Python IDLE or Python GUI as below:

print “Hello, world! My name is Li and I am testing Python and GitHub. Do they communicate well?”

and save it as a file (e.g., TestPython.py) under a certain directory of your computer, say, D:\1.Grants\5-CNH-ABMs\TestABM\.

1. Put this program under a repository

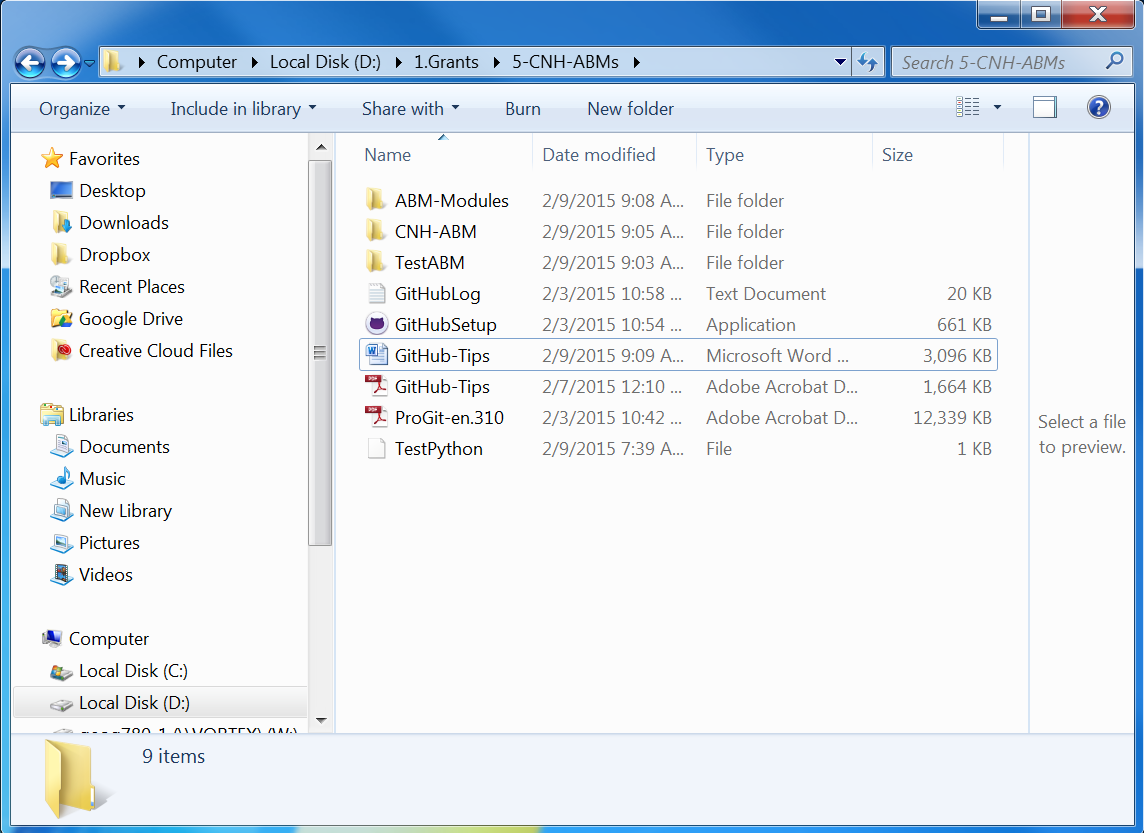
Open GitHub (the Windows version). Depending on where you want to place the program file TestPython.py, we can do different things. First we take a look at my local computer setting as below (Figure 16), where the above program file TestPython is saved under D:\1.Grants\5-CNH-ABMs\:



**Figure 16: The local directory in my personal computer, which is to be linked to my GitHub repositories.**

1. Clone your repository to local folder

You should have practiced clone earlier (see Pages 10-11). But here we want to refresh the relationships between my local computer directories and my Github repositories. Simply go to Windows GitHub (click the GitHub icon and get a page like Figure 12). Click the plus sign (+), you should see all the four repositories I have on GitHub (note I already deleted the Test repository; see Pages 12-13). When prompted for a local directory, use D:\1.Grants\5-CNH-ABMs\. You will see that a local folder named TestABM will also be created under D:\1.Grants\5-CNH-ABMs\. [Here you may also clone your other repositories, here ABM-Modules and CNH-ABM, into your local folder].

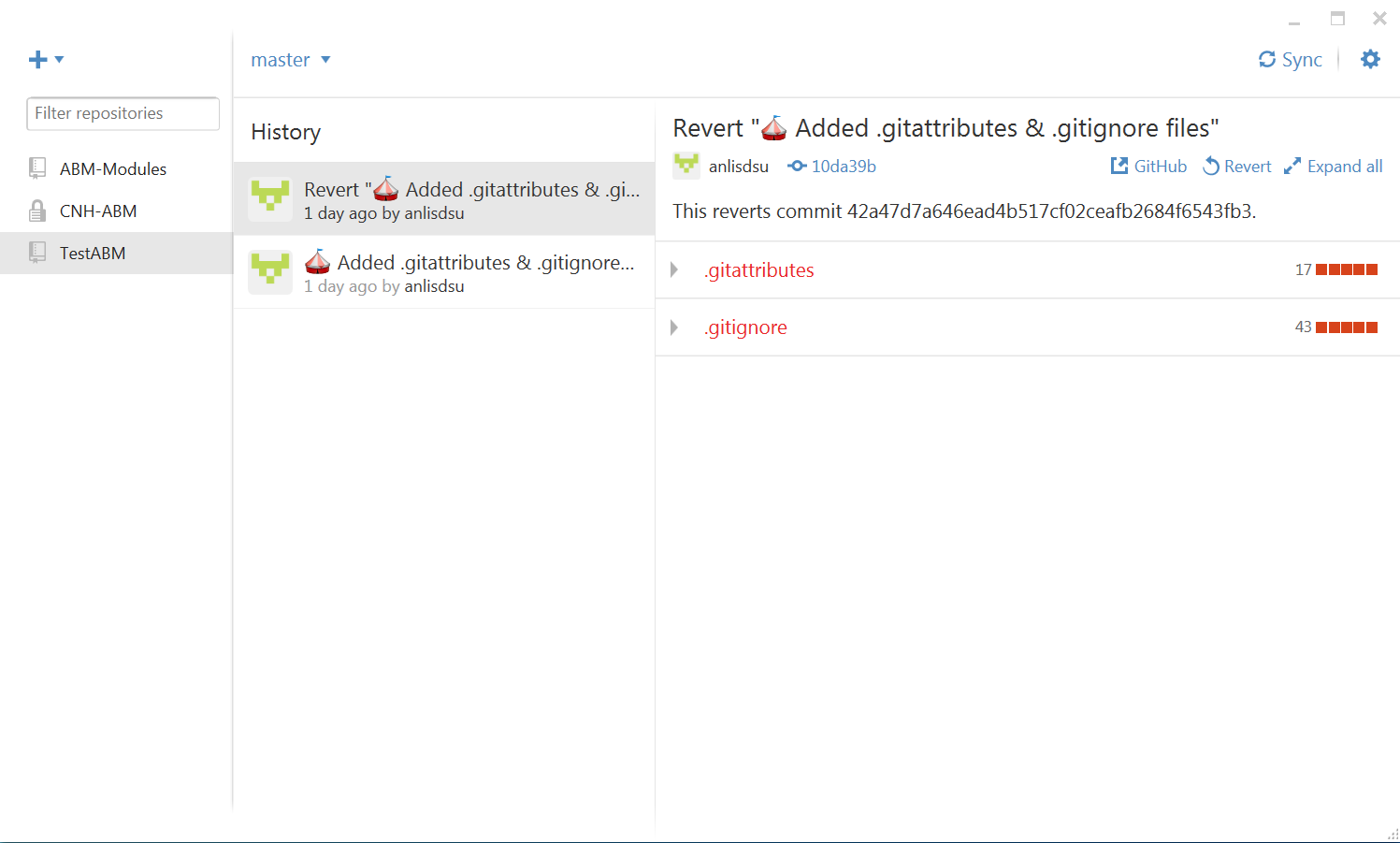


**Figure 17: The local directory in my personal computer, after linking to my GitHub repositories (compare the difference between this one and Figure 16).**

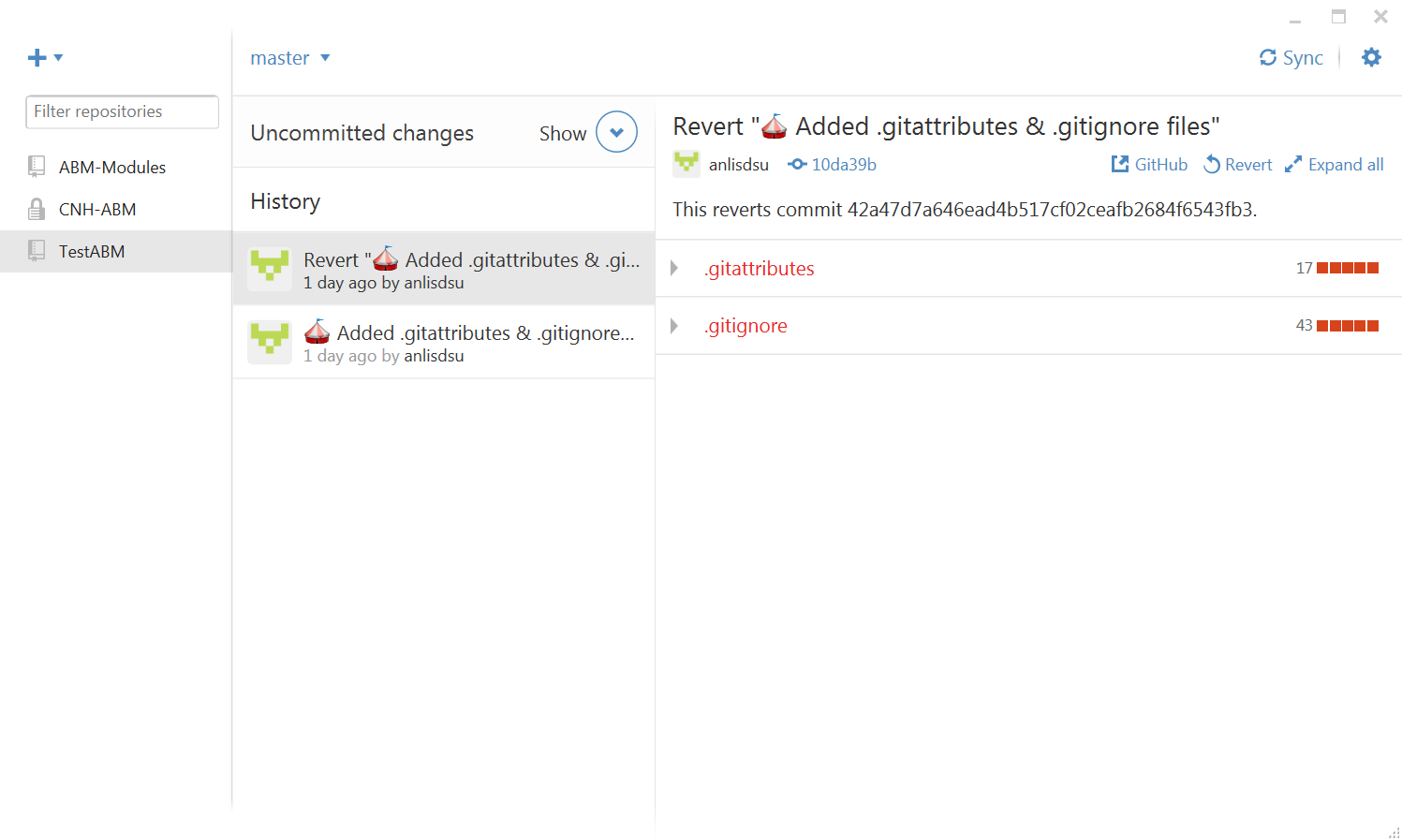
But up to this point, if you click TestABM repository on GitHub, you will not see the program file TestPython.py yet. See next step for why.

1. Save your file into a repository

Move (or save) your program file TestPython.py into any of the three subdirectories, i.e., ABM-Modules, CNH-ABM, and TestABM. Here we choose to save it into TestABM. Now if you go to GitHub, click TestABM, you will see a line for Uncommitted changes (Figure 18; compare it to Figure 17).



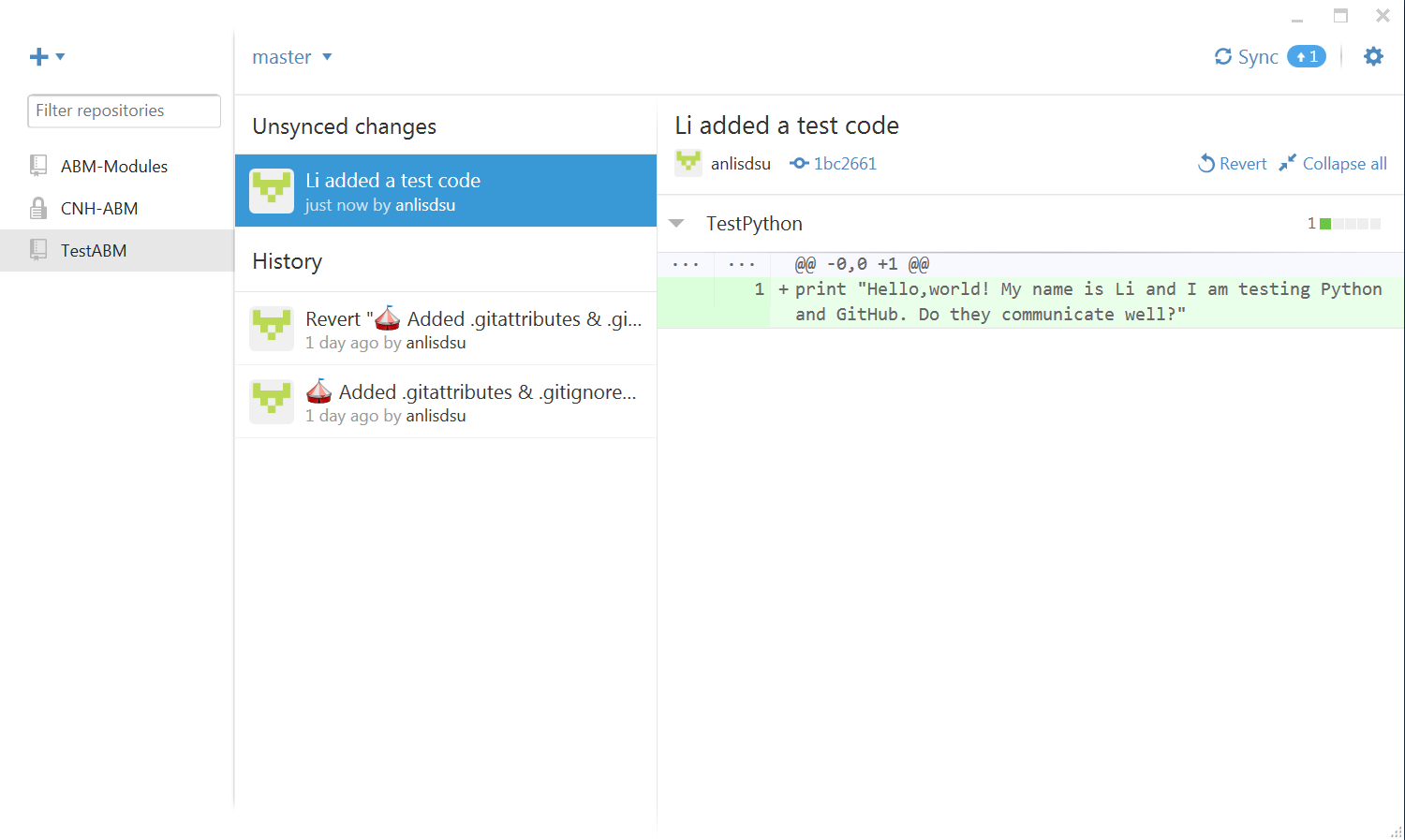
**Figure 17: What the TestABM repository page looks like before saving a program file in.**



**Figure 18: What the TestABM repository page looks like after saving a program file in**

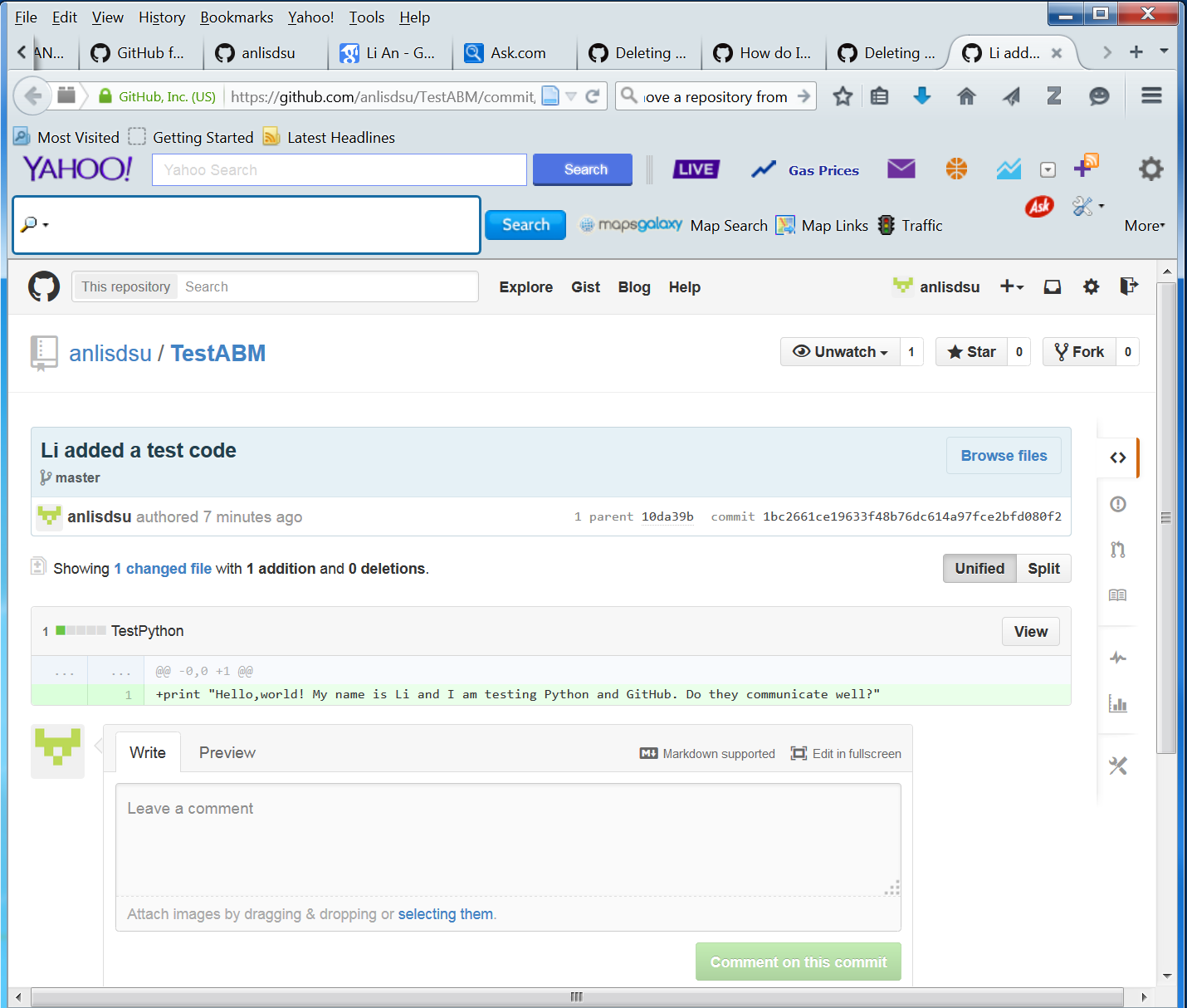
1. Commit the change

Click the Show button (a circle), and in the drop-down boxes, write down some words (e.g., “Li added a test code” or in general who did what at what time), and click the button Commit to master. All the changes will be uploaded and shown as (Figure 19). Note the top item under the Unsynced changes.



**Figure 19: What the TestABM repository page looks like after committing a change.**

Then you might want to click the sync button near the top right of the panel (e.g., Figure 19) to synchronize the changes to your GitHub page. To verify if this is successful, go to the GitHub under the same repository (here TestABM), you will see the same program there (Figure 20).



**Figure 20: What the GitHub looks like after synchronizing our changes.**

1. Continue making changes.

Assume we have closed all the programs (here TestPython.py) and now want to work on it after rest for some time. Go and open IDLE Python🡪File🡪 Open, browse to D:\1.Grants\5-CNH-ABMs\TestABM\, open TestPython.py, and add a few words “Changes are made later” as:

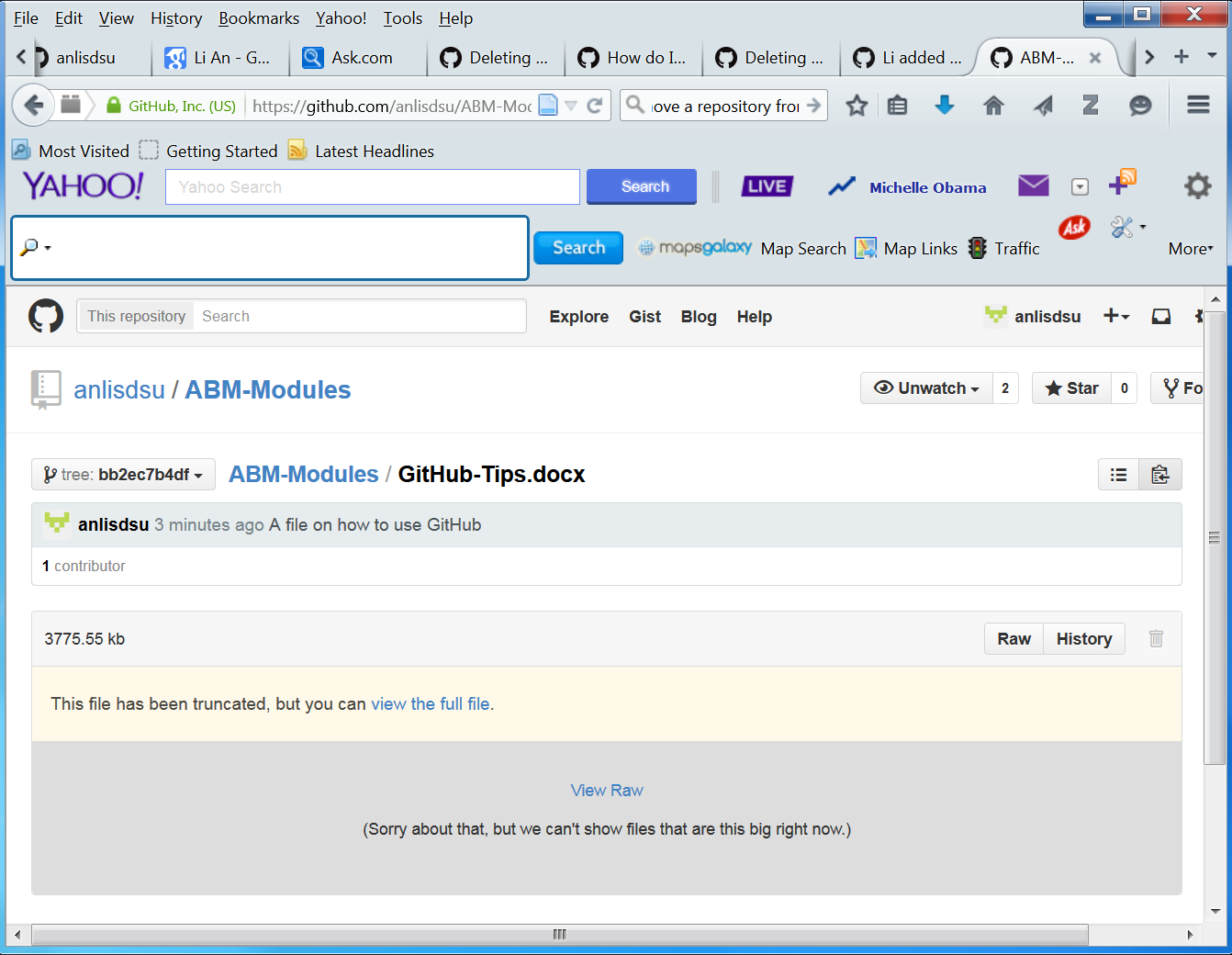
print "Hello,world! My name is Li and I am testing Python and GitHub.

Do they communicate well? Changes are made later"

Save the file. Open GitHub (for Windows), you will see Uncommitted changes near the top of the panel, and repeat Step 5) to commit the change. You will see the change. And you can go and delete some words from the file TestPython.py, save the file under IDLE. Go back to GitHub and commit the change.

So up to this point, we should be able to use GitHub to make changes and upload to GitHub for public contribution. Here I also put this documentation file (on how to use GitHub) online under the public repository ABM-Modules. Note the CNH-ABM is a private repository and only invited users can access it.

After we commit the change and sync it, the GitHub looks like this:



**Figure 21: The Github page under the ABM-Modules repository after the change is committed and synced.**

The file is so big, it cannot be displayed on the GitHub web, but you can click view the full file or View Raw button to open the entire Word file.