GIT – Project management in the cloud

|  |
| --- |
| **Learning Outcomes** |
| At the end of this practical lesson, students will be able to:   * Setup the Python Flask Web Application environment. * Setup/Configuration * Working locally * Going remote |

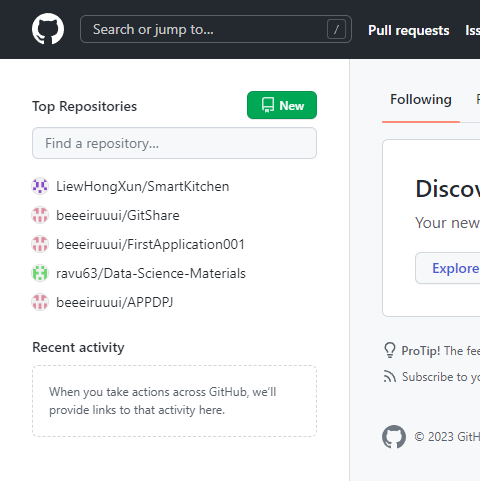
1. Set Up Environment
   1. Tools/Software Used

Below are the tools/software used to push and pull from Git:

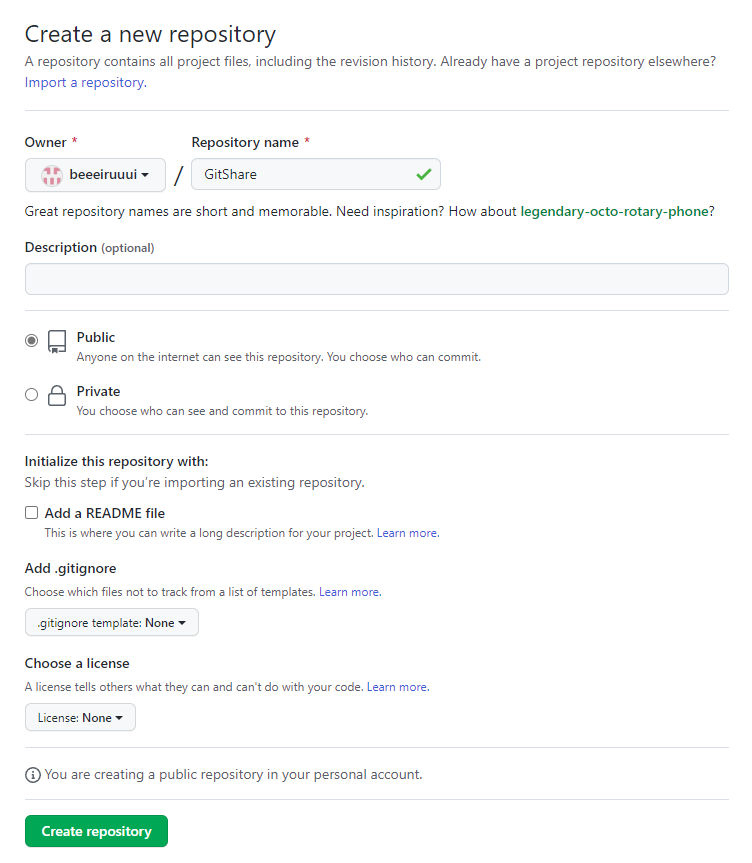
* + - PyCharm Edu 2021.3
      * [https://www.jetbrains.com/education/download/#section=pycharm-edu](https://www.jetbrains.com/education/download/#section%3Dpycharm-edu)
      * This version includes Python 3.10
* GitHub Desktop
  + <https://desktop.github.com/>
* GitHub Website
  + <https://github.com/>
  1. Create a Simple Application By the Creator of desired Repository

Step 1: sign up or sign in to your github account

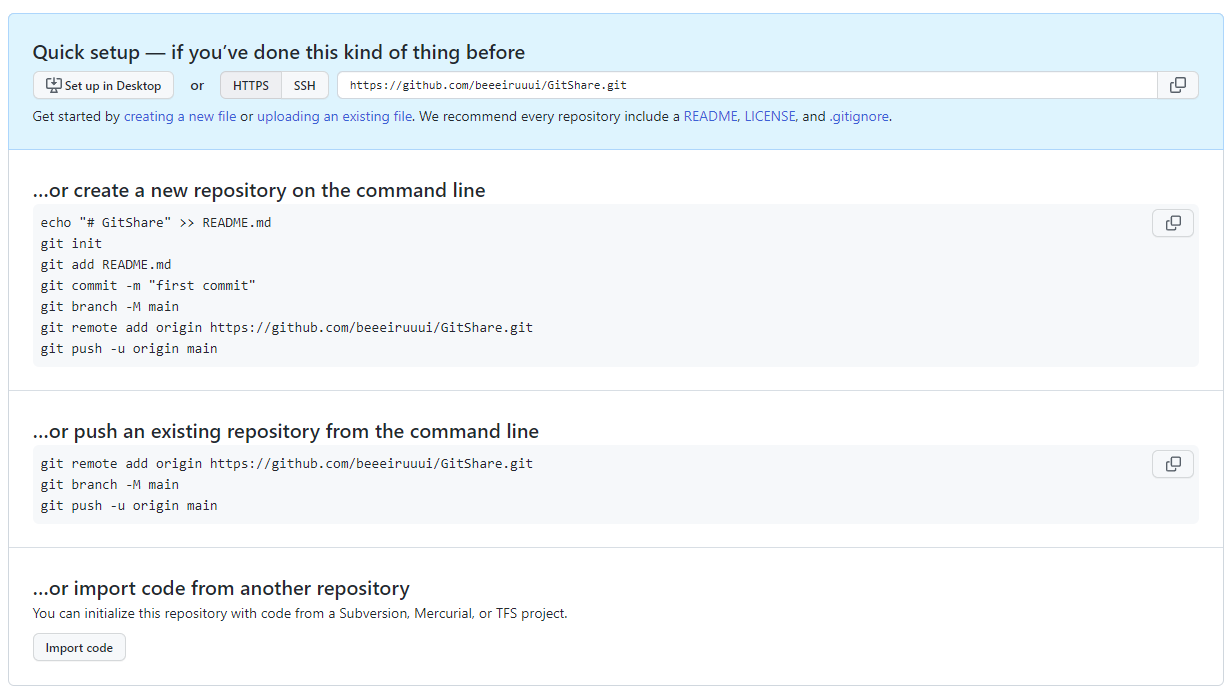
Step 2: Click New to create a repository



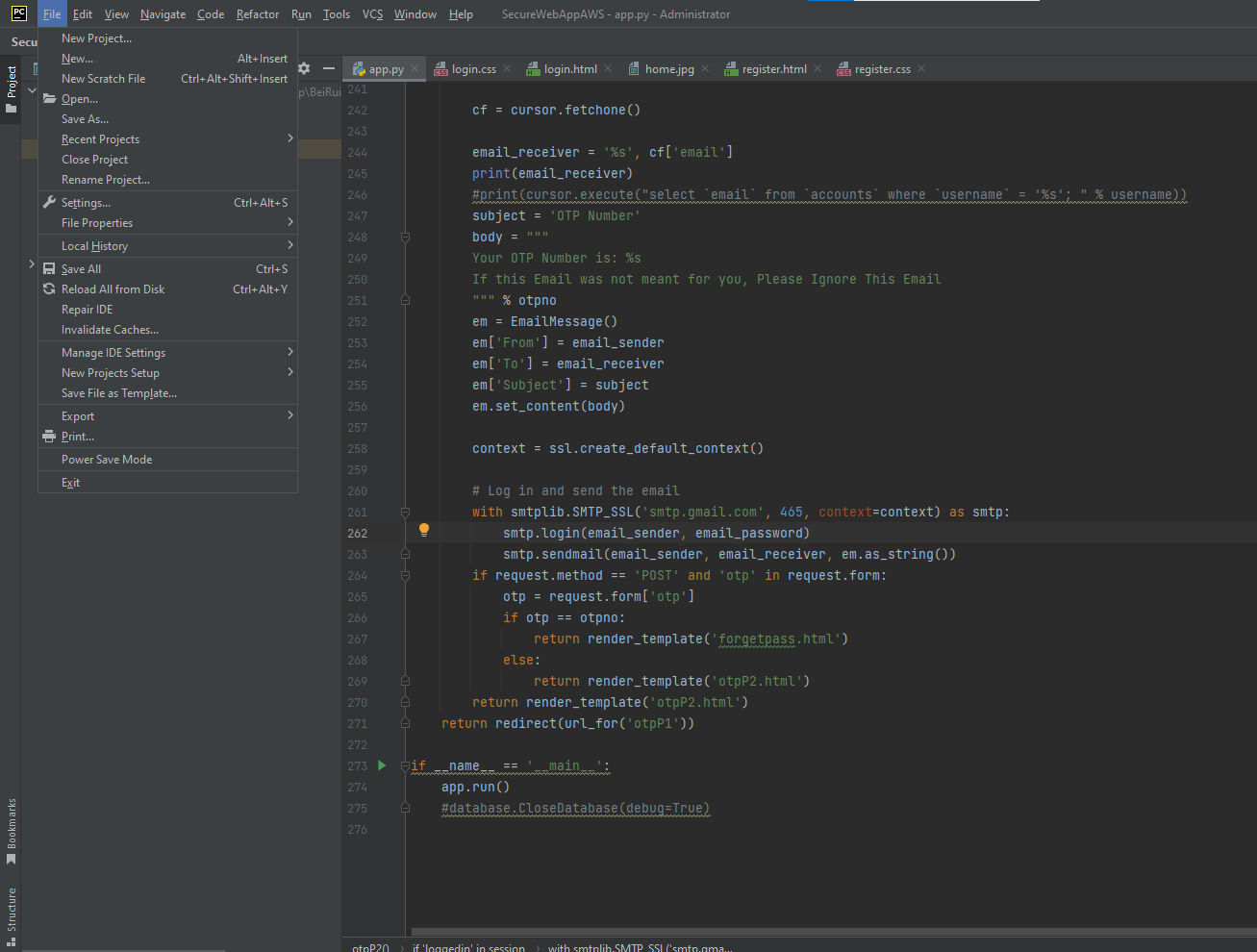
Step 3: Create a new repository. Under “Repository name” type in GitShare. And click “create repository”.



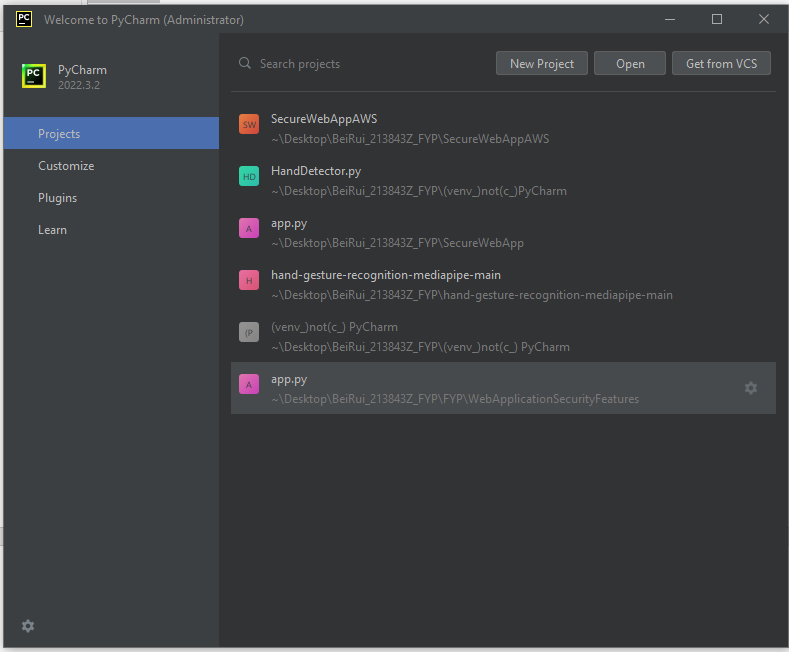
Step 4: See that a link has been created. We will need the link in Step 8.



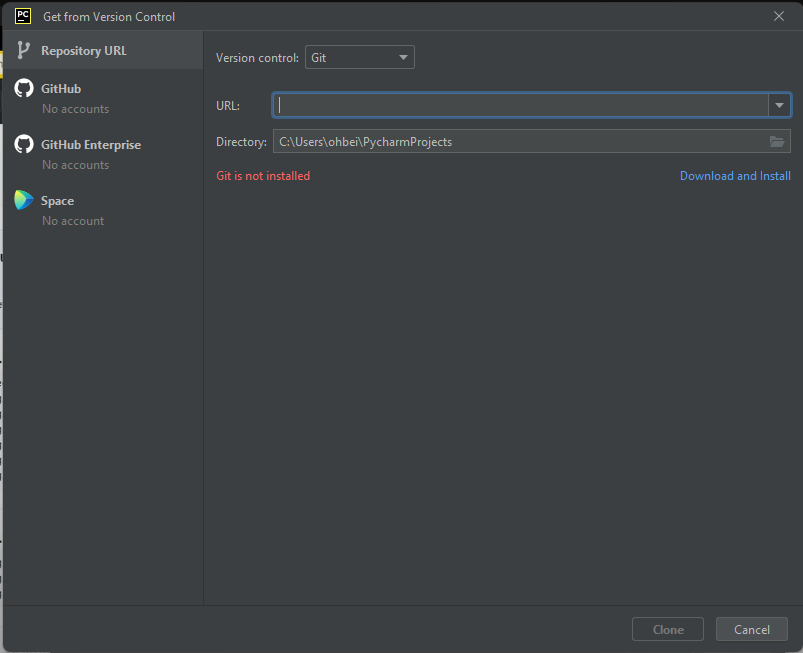
Step 5: Assuming you have an already opened a project. Select “file” and click “close project”. If not skip to step 6.



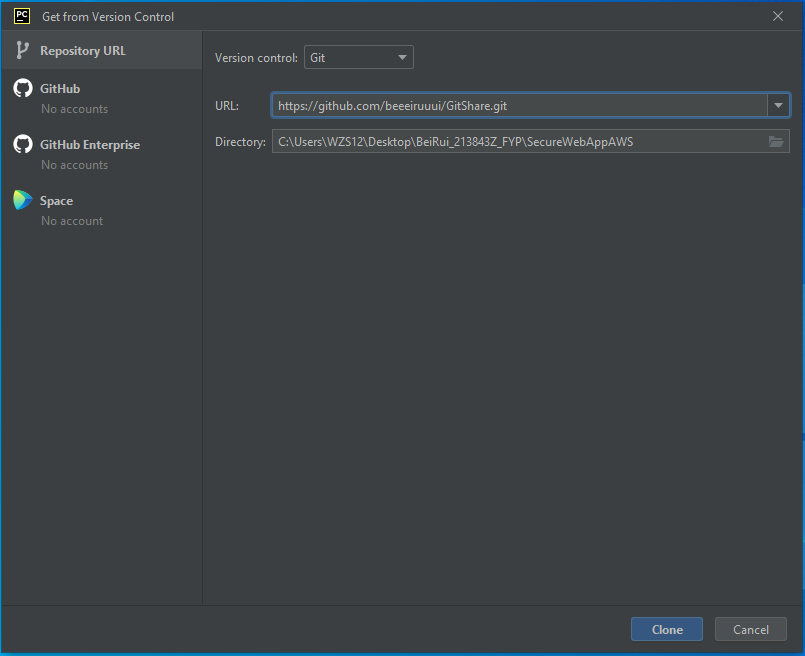
Step 6: After step 5, a pop up appears. Or after opening PyCharm Edu software, a windows appears. Select “Get from VCS”.



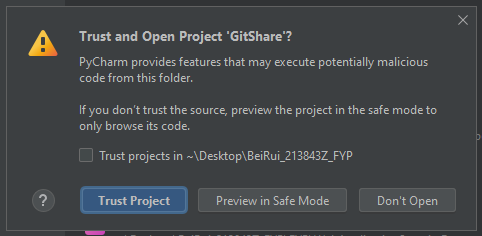
Step 7: Click “download and install” if any other pop up appears, just use default selected settings and click okay



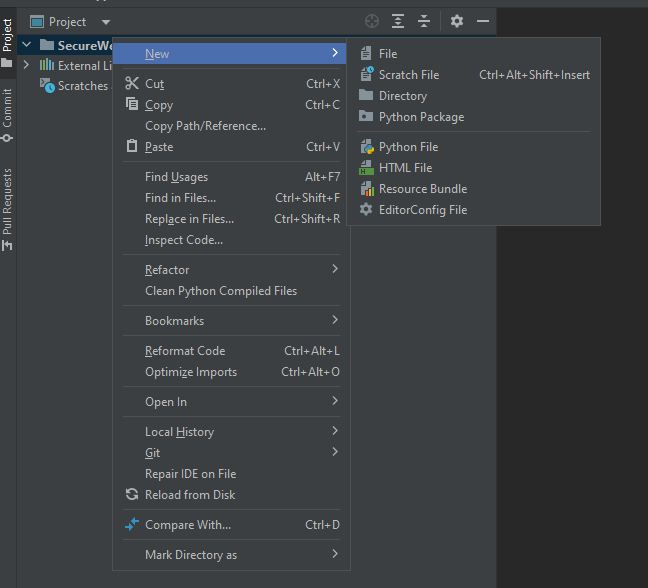
Step 8: in “URL:” textbox, copy the GitHub link in step 4 and paste it in. In “Directory” textbox, put in your desired path so that you know where your project that works with GitHub is being stored. And click “Clone”



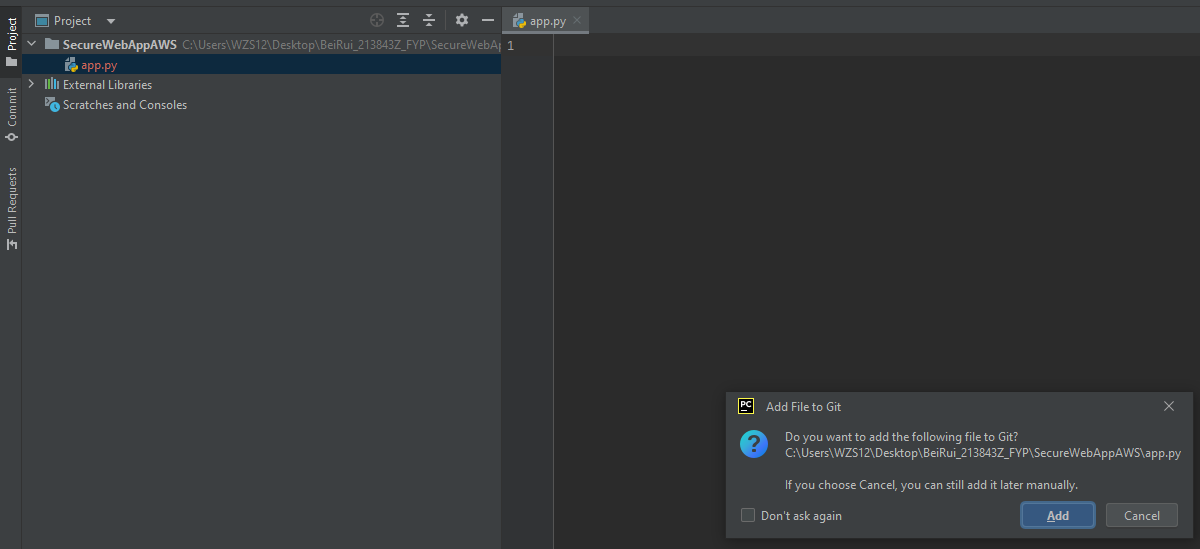
Step 9: a pop up appears. Click “Trust Project”



Step 10: Right click on “SecureWebAppAWS” > new > Python File. Create a new Python File called “app.py”.

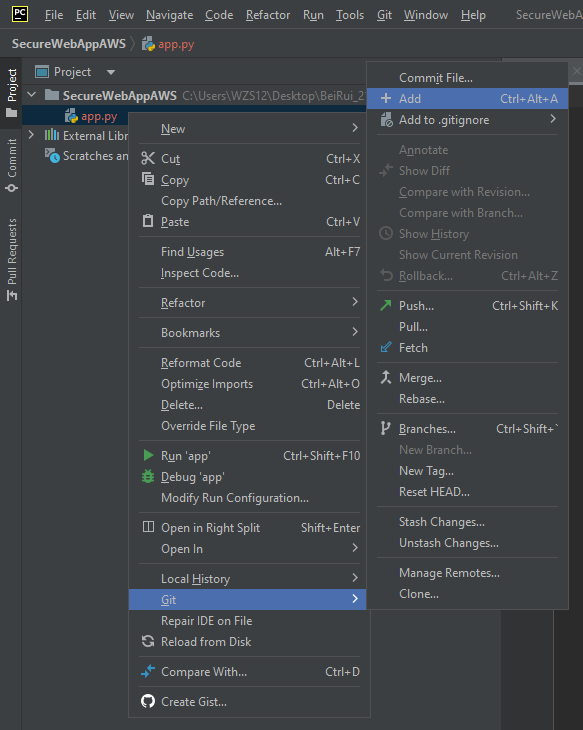


Step 11: a pop up will appear at the bottom right of your screen. Click “cancel” to prevent confusion or uncertainty as you continue forward in on your project without the help of this guide. To add manually, follow Step 12.

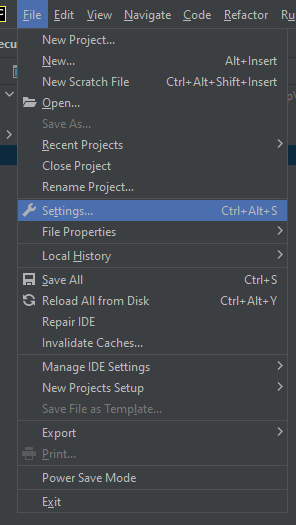
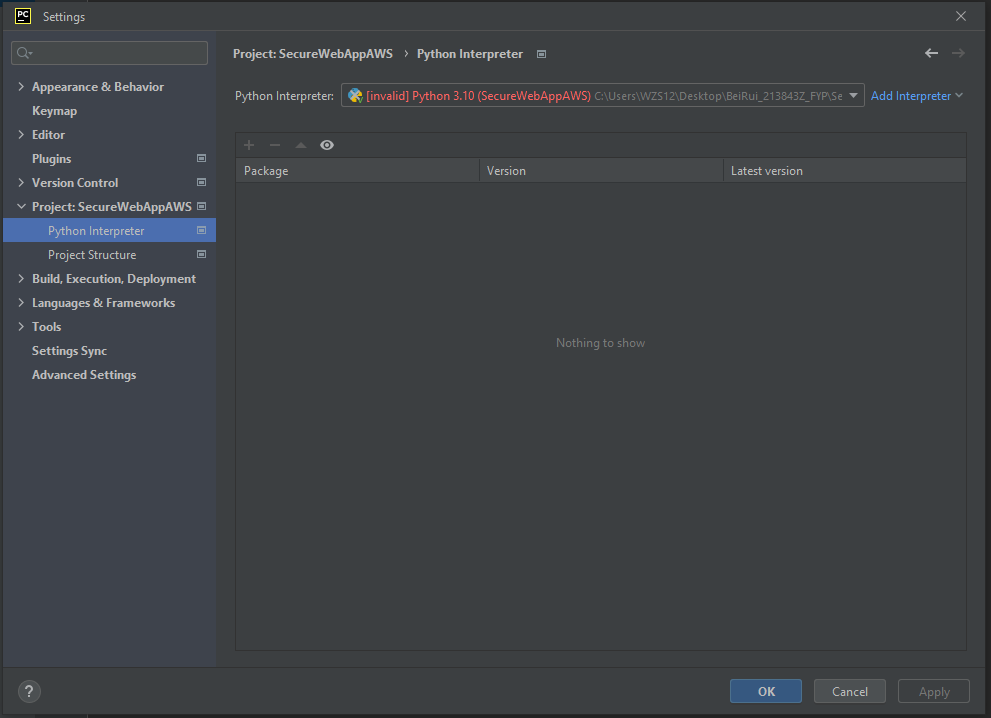


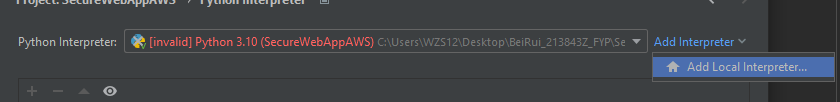
Click cancel to prevent confusion or uncertainty as you continue forward in on your project without the help of this guide

Step 12: to add manually right click app.py > Git > Add. As we do not want PyCharm to add venv automatically after the folder is being created.

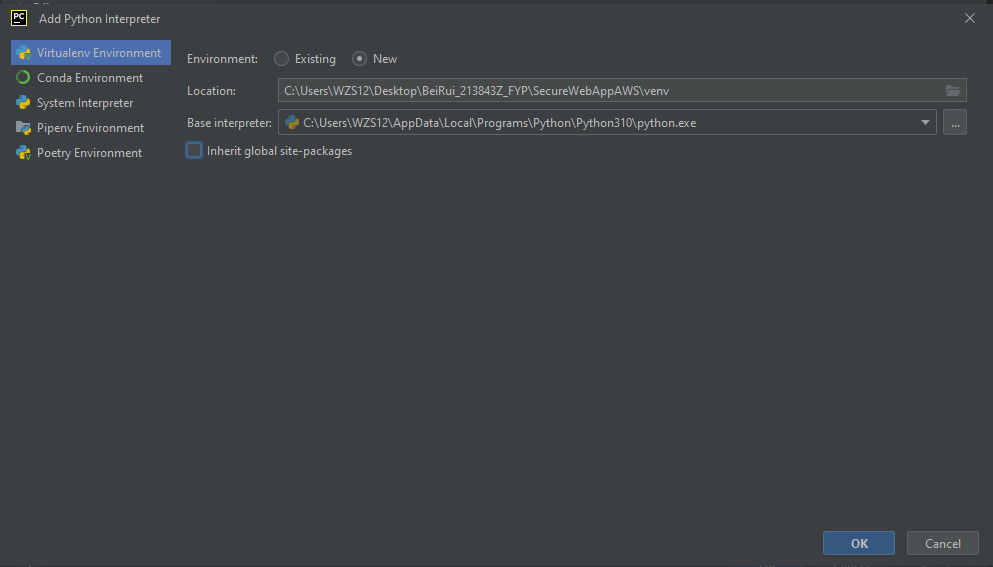
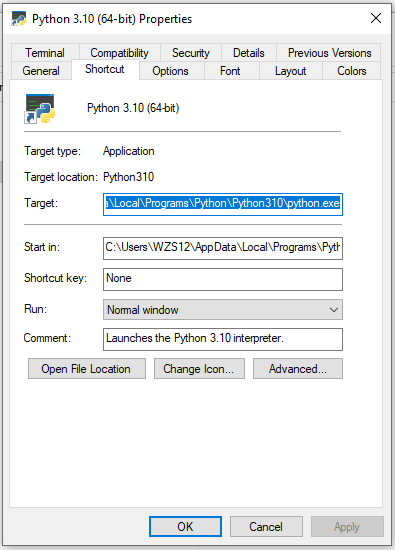


Step 13: add venv folder. Click file > Settings > Project: SecureWebAppAWS > Python Interpreter > add interpreter > add local interpreter. **Do not** add venv into Git.

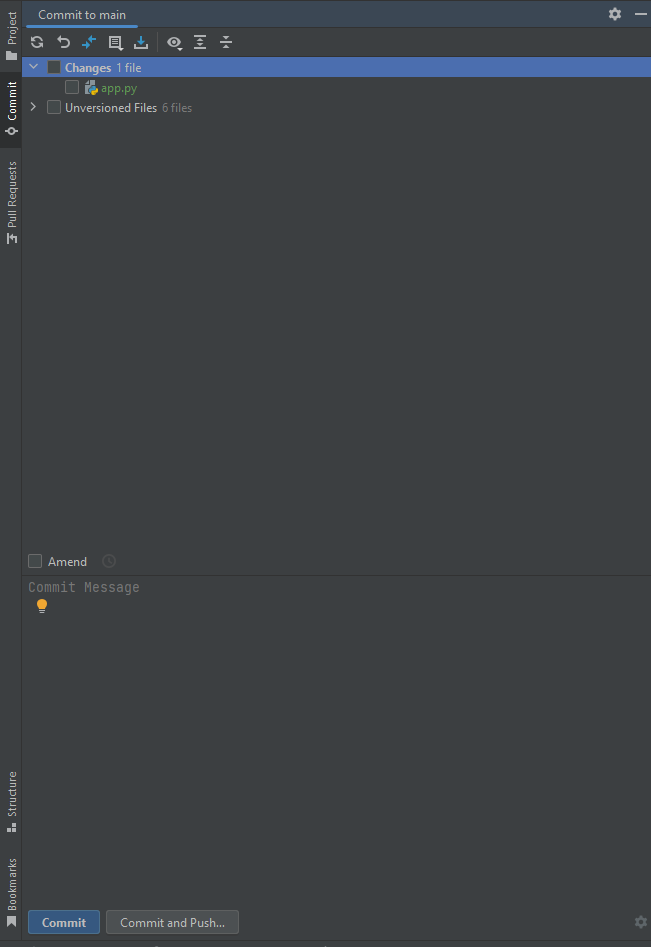
 



Step 14: in “Location:” textbox put venv inside “SecureWebAppAWS” folder. In “Base interpreter:“ textbox copy and paste the “Target:” location of python 3.10 into “Base Interpreter” textbox.

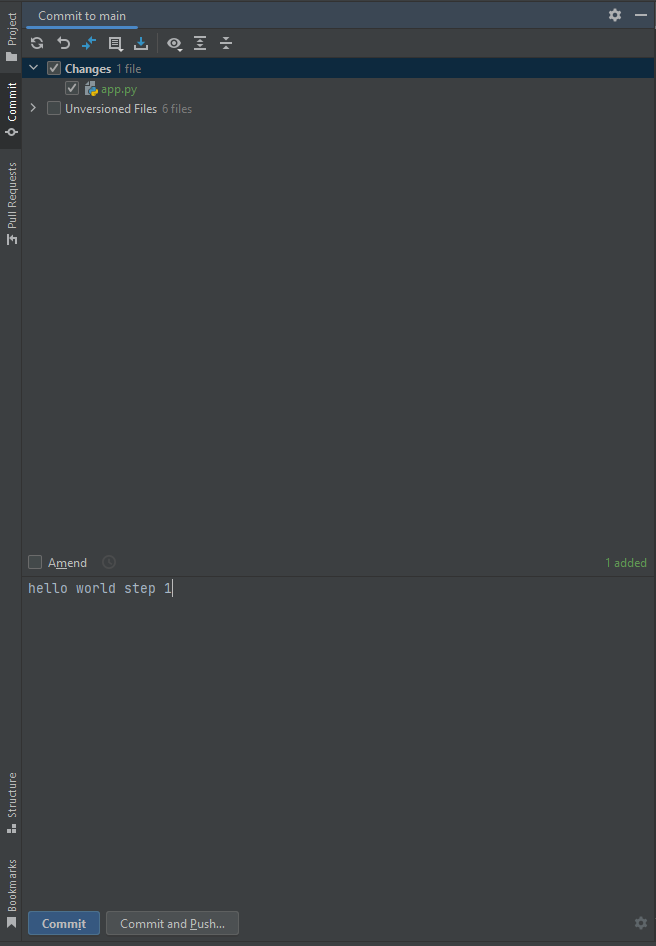
 

Step 15: Click commit. At the left of the windows, you will see app.py in Blue means that it is not committed. if app.py is in Green means it is committed. To make it turn Green from Blue, follow the next few steps. If you skip Step 12 you will realize that no Changes have been made in the commit tab

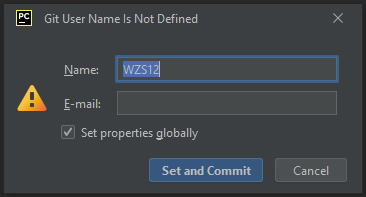
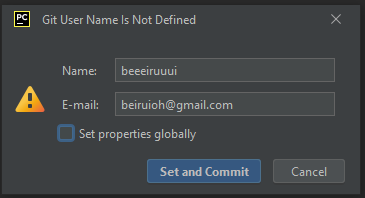


Step 16: to commit and send(Push) it to GitHub:

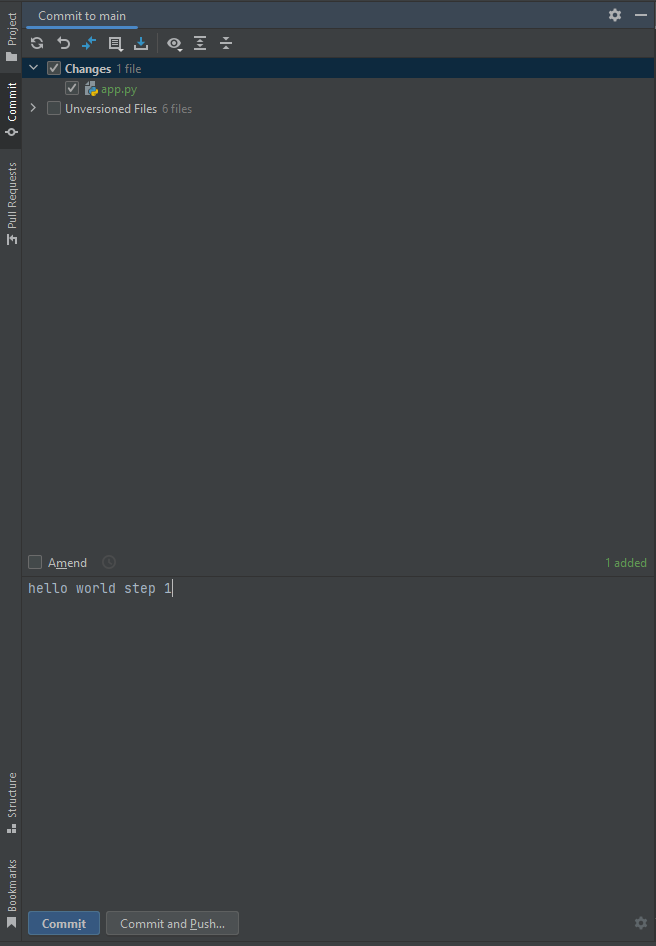
1. Click refresh to keep changes always up to date (boxed in red)
2. Tick “Changes” checkbox (boxed in yellow). **Do not** tick “Unversioned Files” checkbox as we will not use those files
3. Under “Amend” in “Commit Message” type any message to show what individuals did to the project (boxed in orange)
4. Lastly press commit (boxed in green)



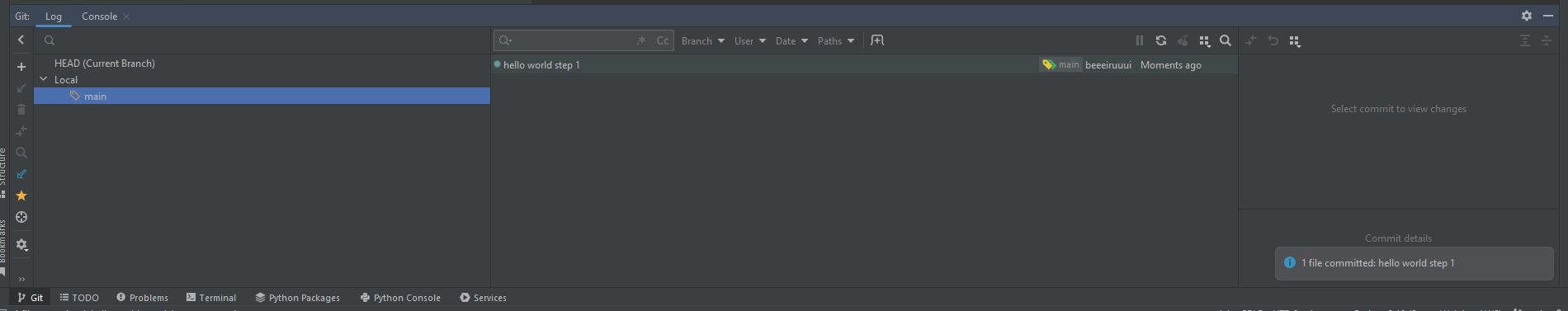
Step 17: a pop up will appear saying Git username is not defined. In “Name:” textbox, type your GitHub username. In “E-mail:” textbox, type your email that you used to create your GitHub account. Uncheck “Set properties globally” checkbox. Lastly press set and commit.

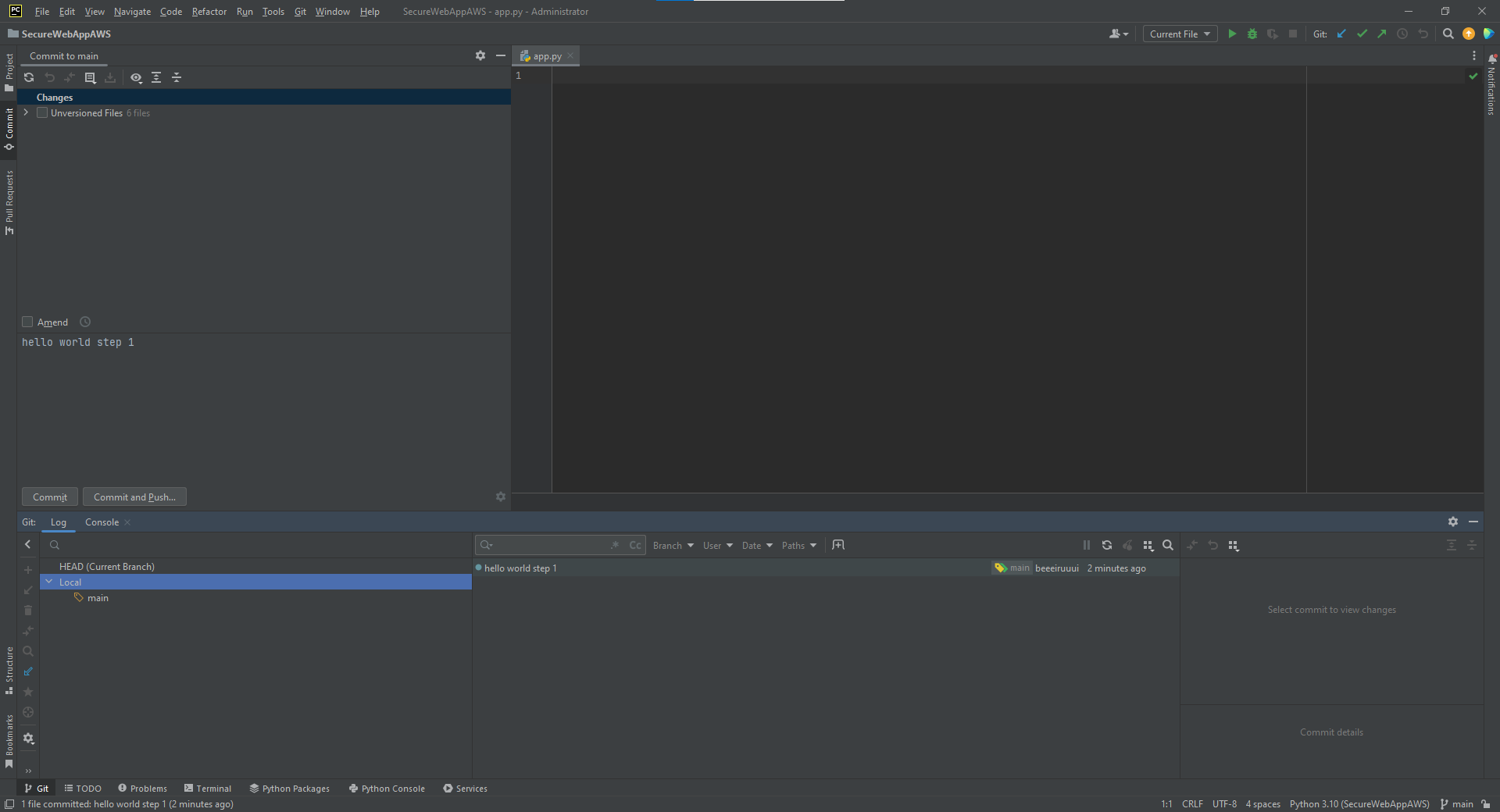
Step 18: press Commit



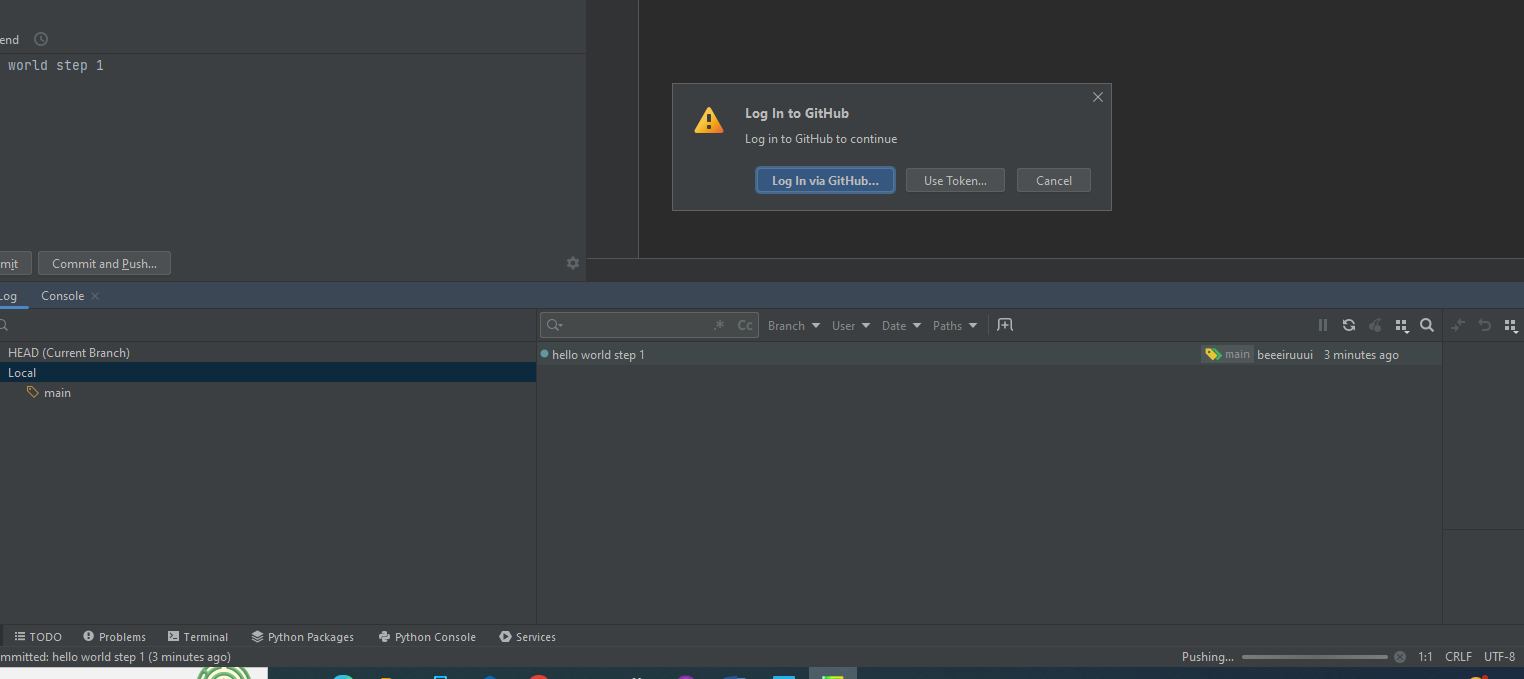
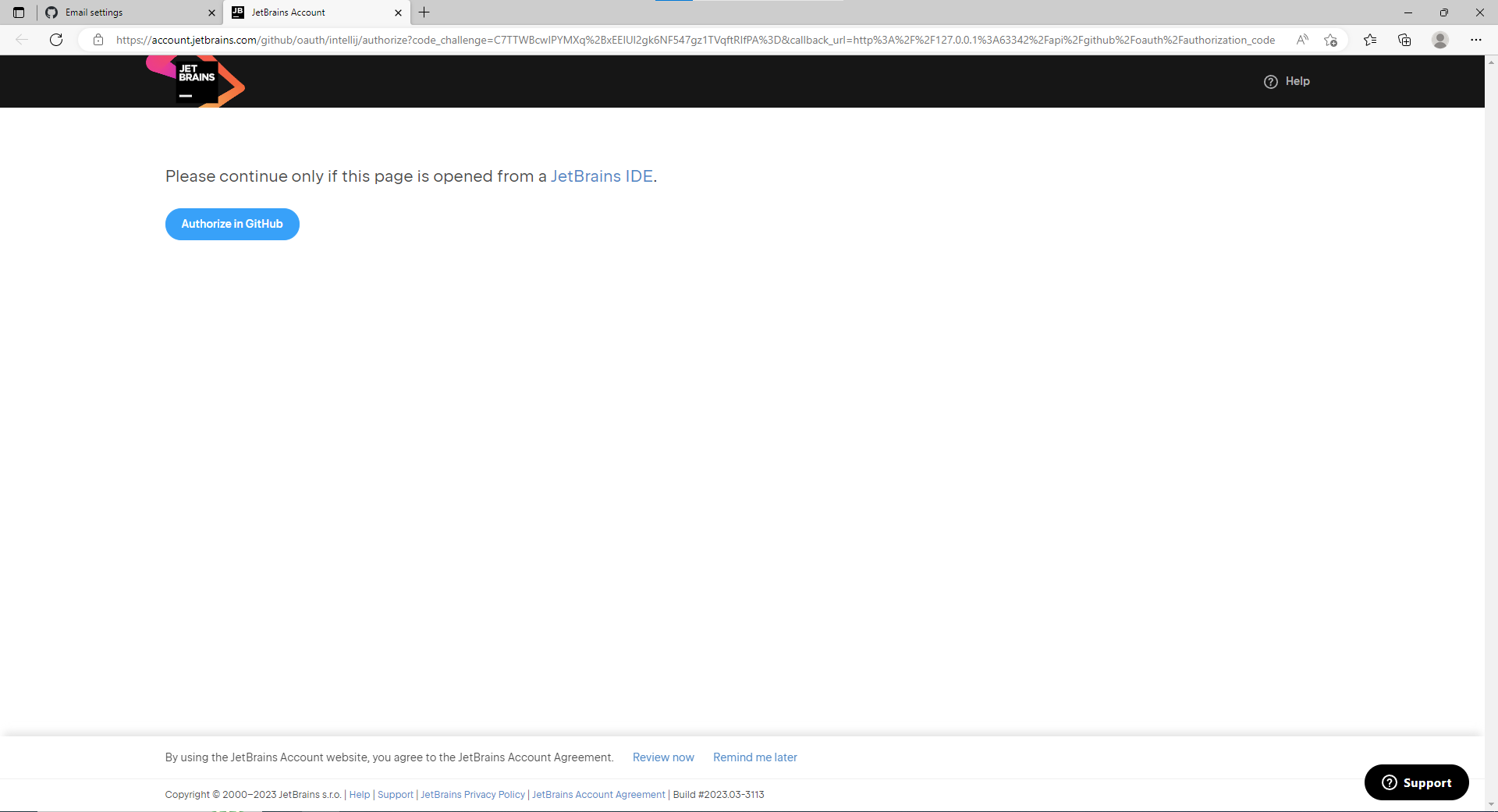
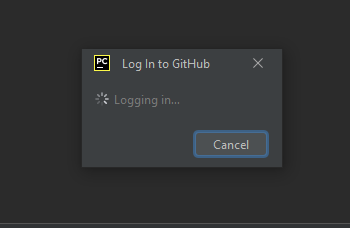
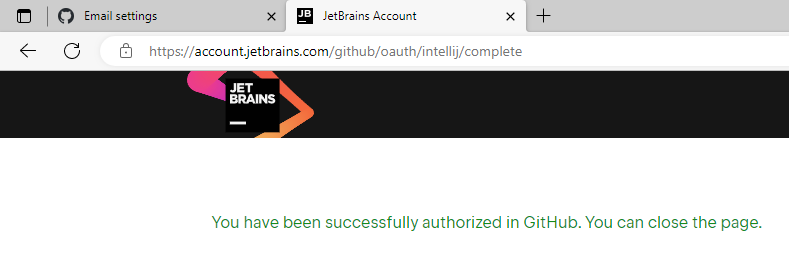
Step 19: press “Git” tab at the bottom left of the window notice that “hello world step 1” have been shown in the bottom middle screen. This means your commit is a success. However, it is not committed to GitHub Website or GitHub Desktop. In order the commit to GitHub, follow the next few steps.



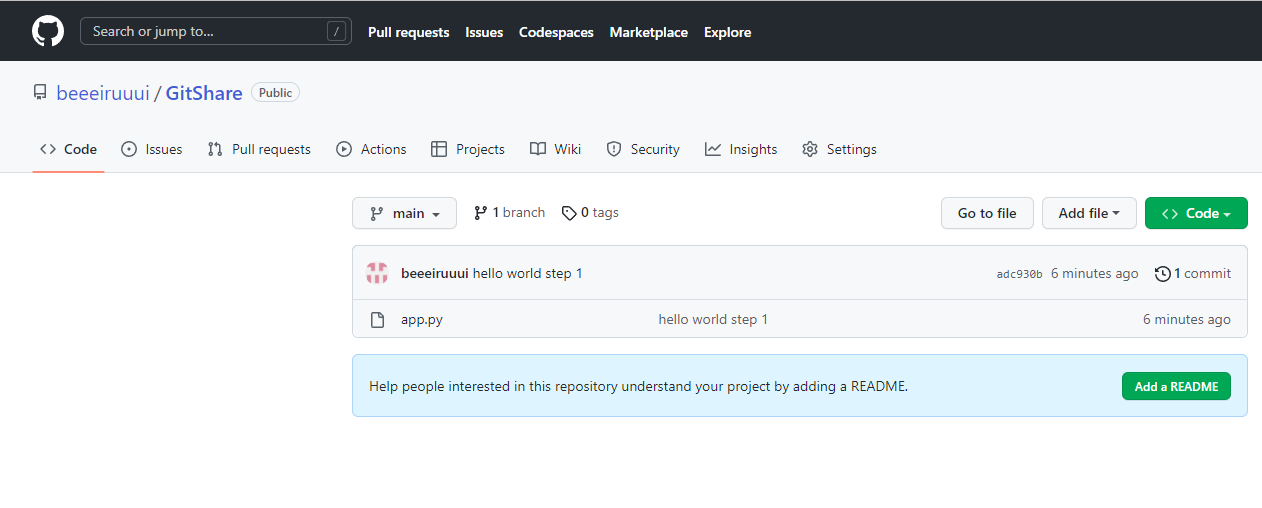
Step 20: press on the green arrow pointing to the top right(represent “push”) at the top right of the screen to push your edits onto GitHub.



Step 21: a pop up will appear to ask you to login to GitHub. Press “Log In via GitHub…” it will lead you to your default web browser to ask you to login. If you have done it long ago in step 1 it will show you a button “Authorize in GitHub”. Click “Authorize in GitHub”. Wait for the webpage to prompt you “You can close the page”.

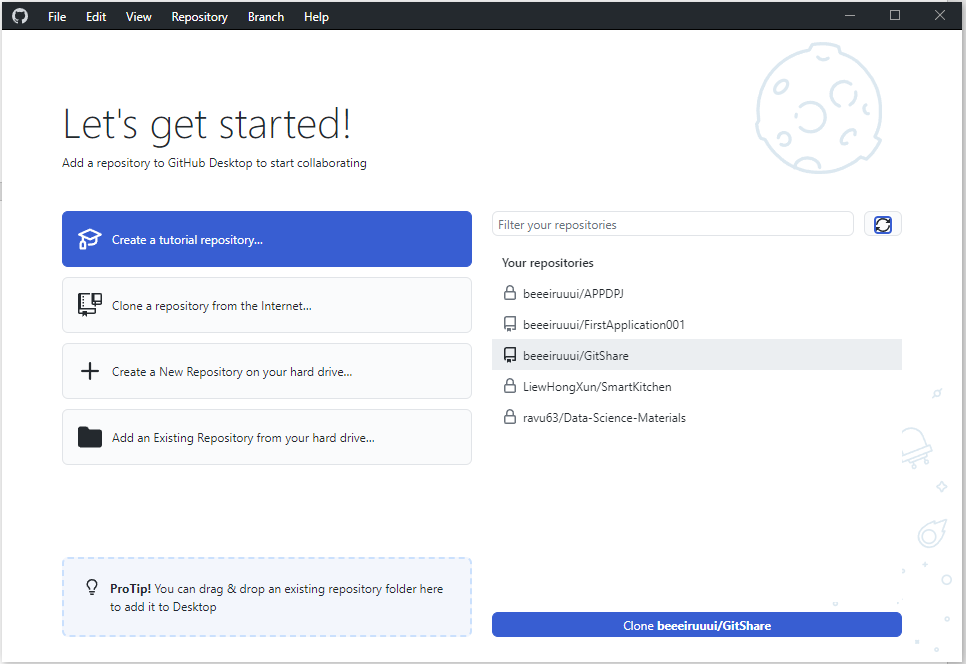
  

Step 22: go to GitHub Webpage. Notice that you have committed it successfully to GitShare repository.

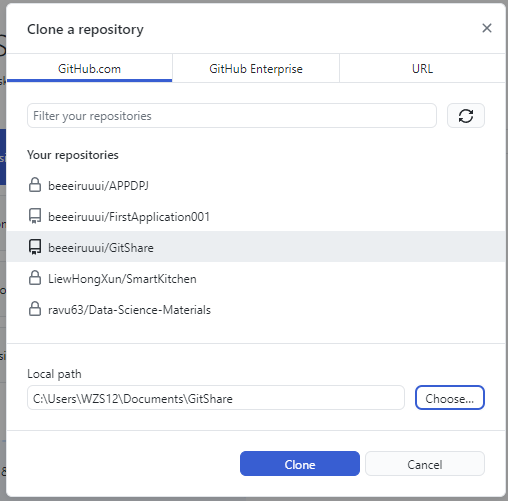


* 1. Creators and collaborators of Repository

Step 1: open GitHub Desktop and select “Clone a repository from the Internet…”

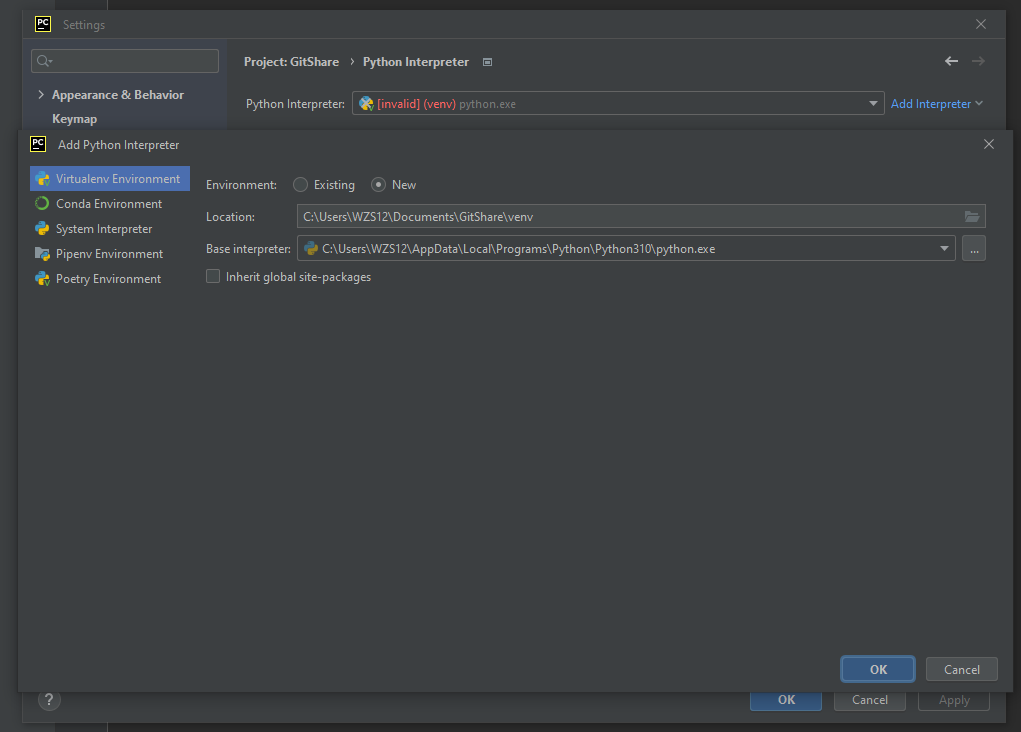
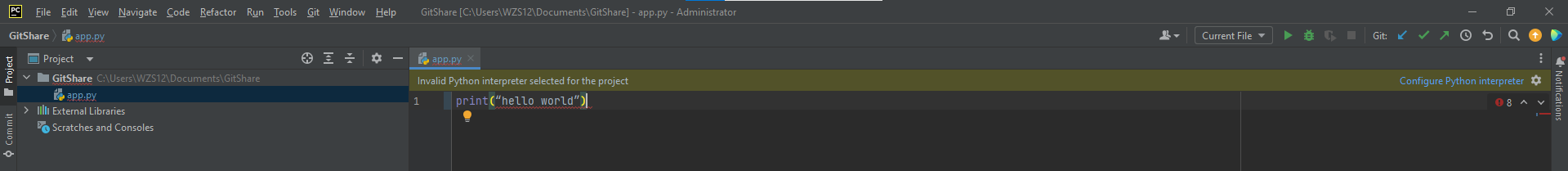


Step 2: click “Refresh” and select the Repository you are supposed to work on. Next either pin point the Local path to your already created Folder that is working with GitHub or choose a new Local path that you want it to be stored. Lastly press “Clone”.

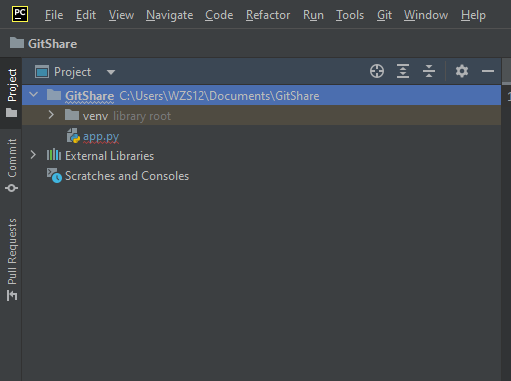
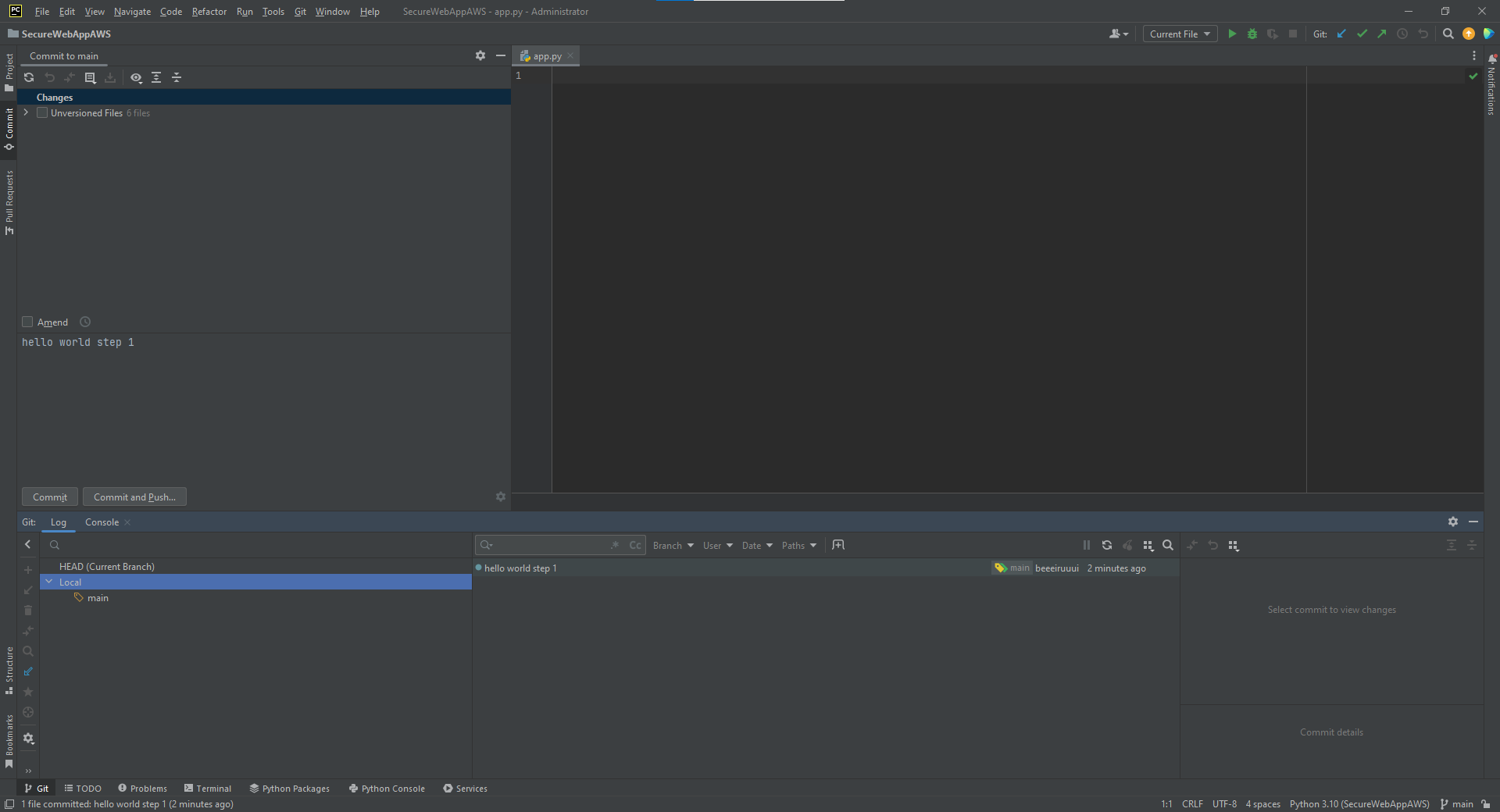
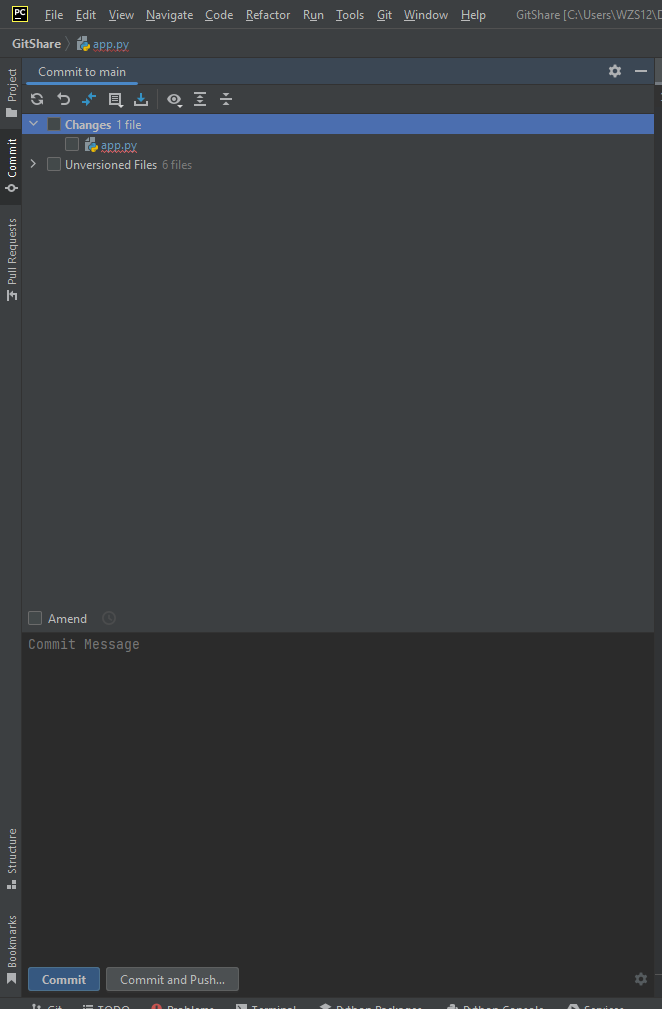
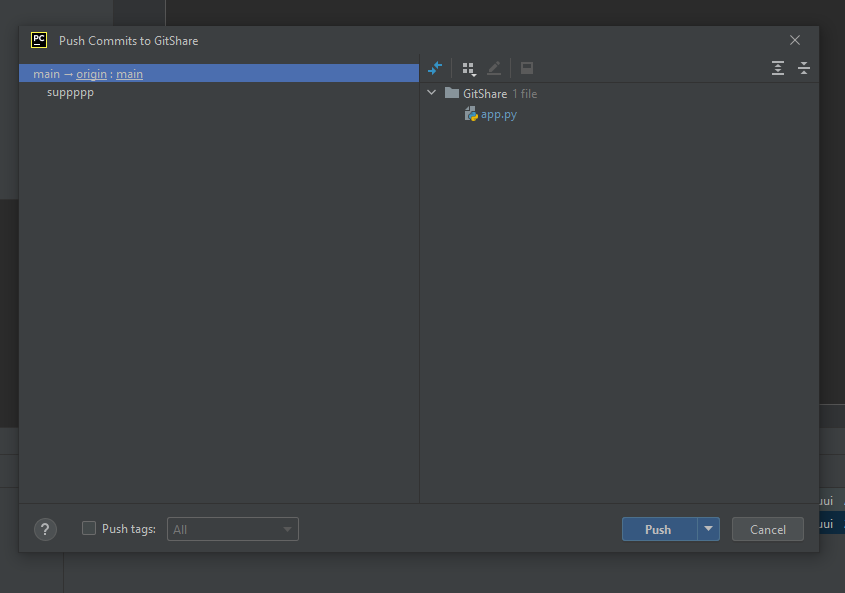
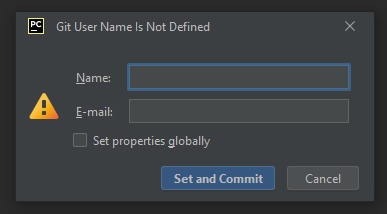


* 1. Collaborators Pulling and Pushing the Simple Application Repository created by Creators.

Step 1: Open your cloned project. In app.py, type in “print(“hello world”)”(type it out yourself because the close and open inverted commas design in word document is different from the open and close inverted commas design in python. If you copy and paste, good luck). Notice that after a while a yellow notification pop up appears telling you to configure a python interpreter. Repeat 1.2 Step 13 to 14

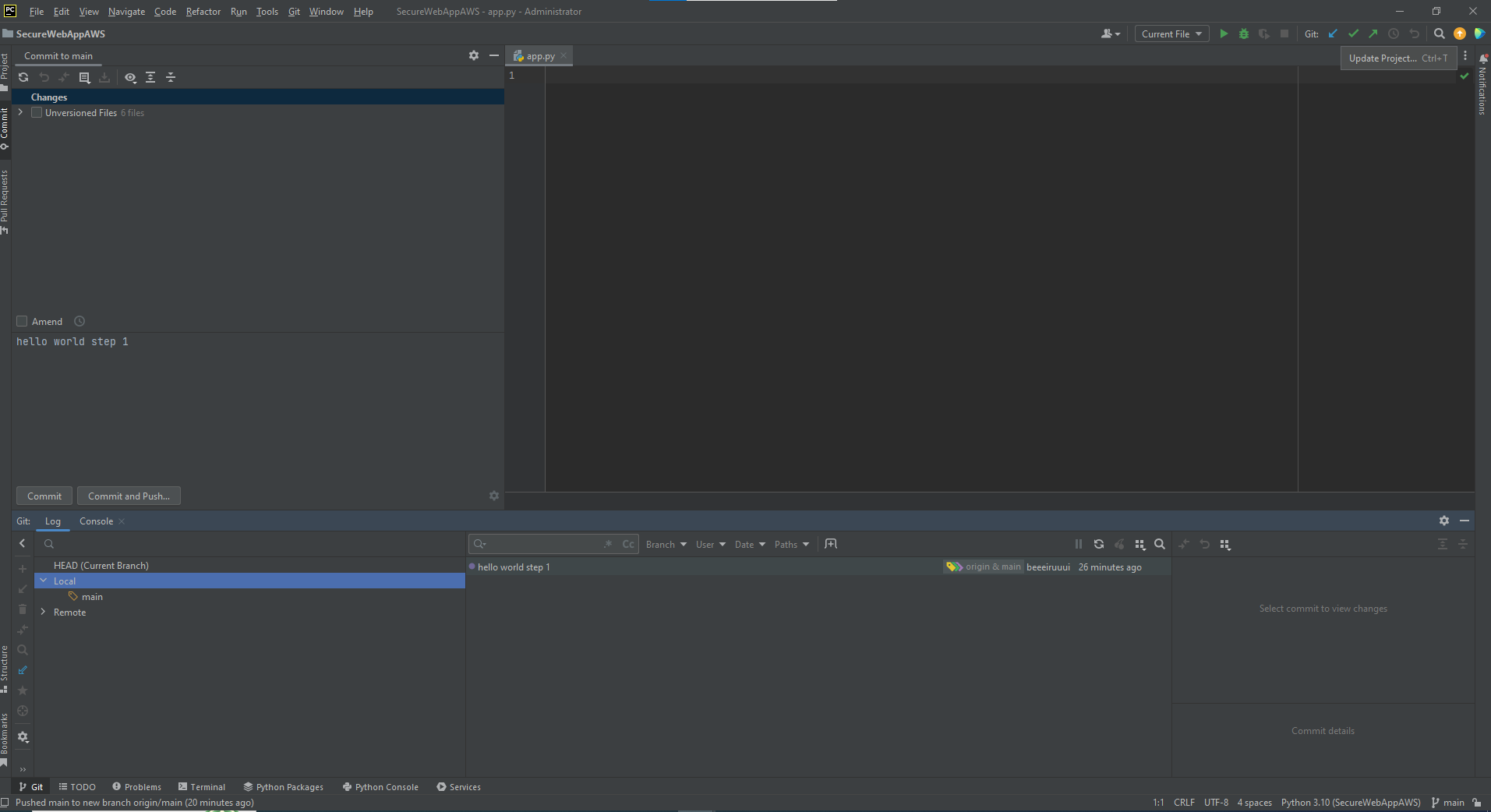


Step 2: Repeat 1.2 Step 15 to 22

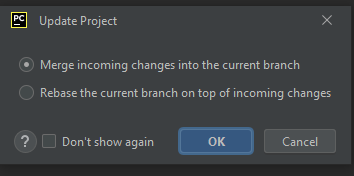
  

* 1. Creators Pulling and Pushing the Simple Application Repository edited by Collaborators.

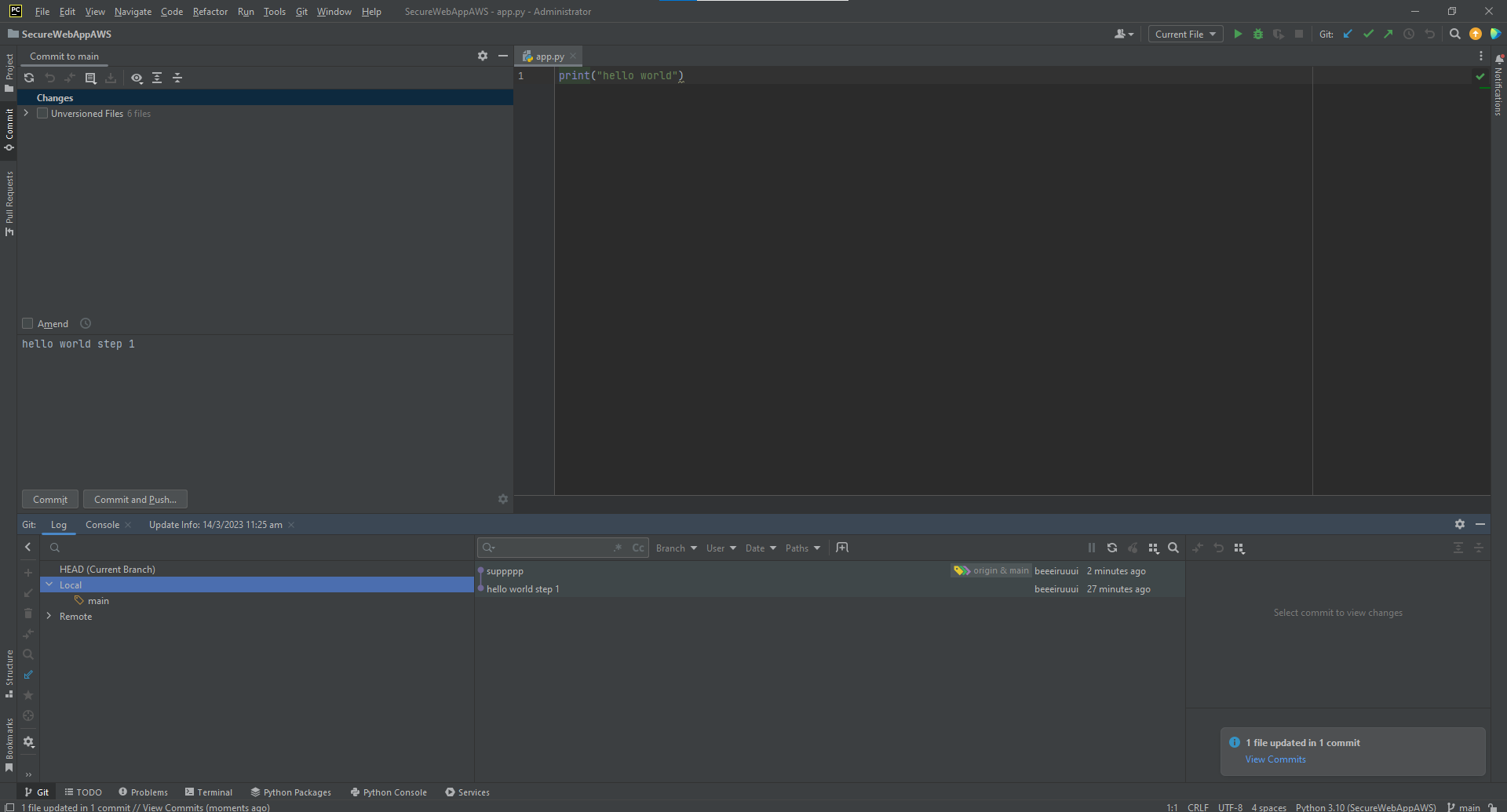
Step 1: for groupmates who wants to update their project to the latest edits, press on the blue arrow pointing to the bottom right(represent “pull”) at the top right of the screen to pull groupmates edits from GitHub into your project.



Step 2: a pop up will appear asking how do you want to update your project. Select “Merge incoming changes into the current branch”. Click okay.



Step 3: all different groups, notice that same changes are shown on each of your screen? If not, please pull again and click all the refresh buttons you see in the picture below.



Step 4: pull and push from you to others or others to you, to keep your project up to date.

* 1. THE END explore further at your own risk

Now you are good to go! Notice how venv Folder is undisturbed?

Best practices:

1. do a schedule planner so that each of you will know when to integrate at different timing to avoid overlapping.
2. each groupmate in different groups should use same python version for example, if one person use 3.9.3, the other should use 3.9.13. as long as it is in the range of 3.9.x, you are good to go.
3. However, if you want to use 3.10.x while your group mates uses 3.9.x, it doesn’t matter for those using 3.10.x. But matters for those using 3.9.x. As the higher the version, new things maybe added which the old versions do not support or have.