

Programming in Python II

GitHub

**NGEE ANN**
POLYTECHNIC

All rights reserved. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of the author.

Trademarked names may appear in this document. Rather than use a trademark symbol with every occurrence of a trademarked name, the names are used only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The information in this document is distributed on an "as is" basis, without warranty. Although every precaution has been taken in the preparation of this document, the author shall not have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this document.

Table of Contents

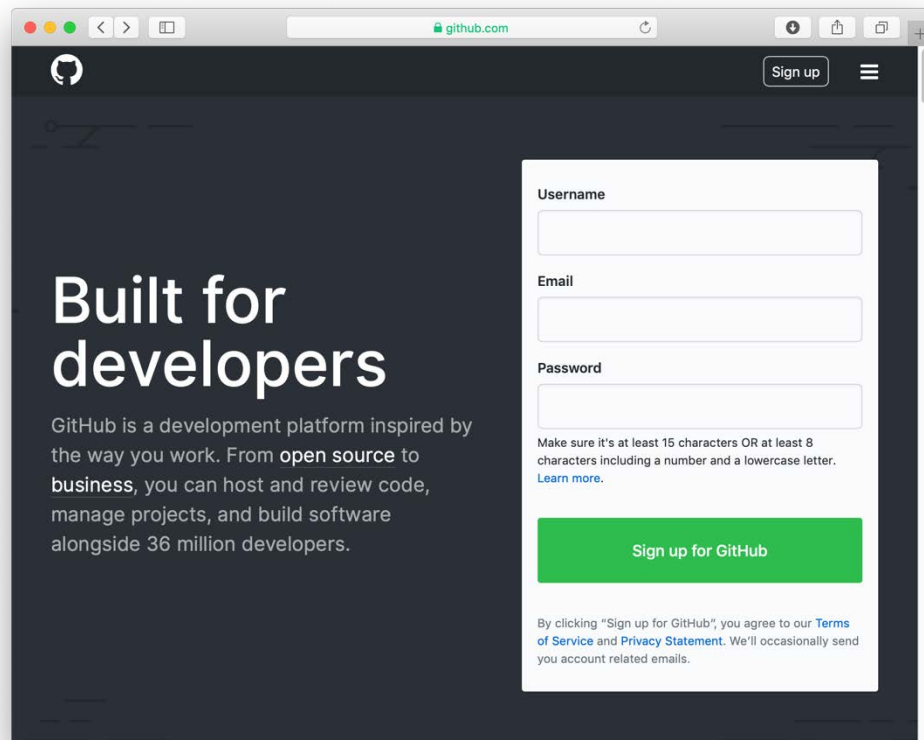
Lab 1. Getting Started with GitHub.....	3
Lab 2. Resolving Merge Conflicts.....	15

Lab 1. Getting Started with GitHub

Description	In this lab, you will learn how to get started with GitHub
What You Will Learn	<ul style="list-style-type: none"> • How to create an account on GitHub • How to create a repository • How to add file(s) to a repository • How to create a branch in a repository • How to create pull request • How to merge pull requests
Duration	60 minutes

Creating an Account

1. Using the Web browser, go to <http://github.com>. Enter your username, email, and password to sign up for a new account.



If you already have a GitHub account, simply sign-in to your existing account

Creating a Repository

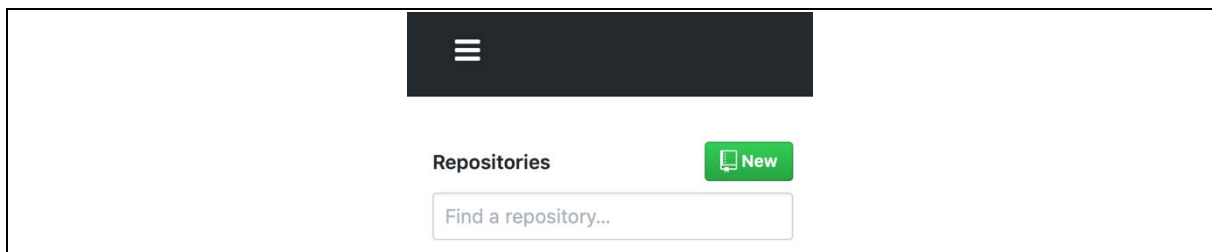


A *repository* is usually used to organize a single project

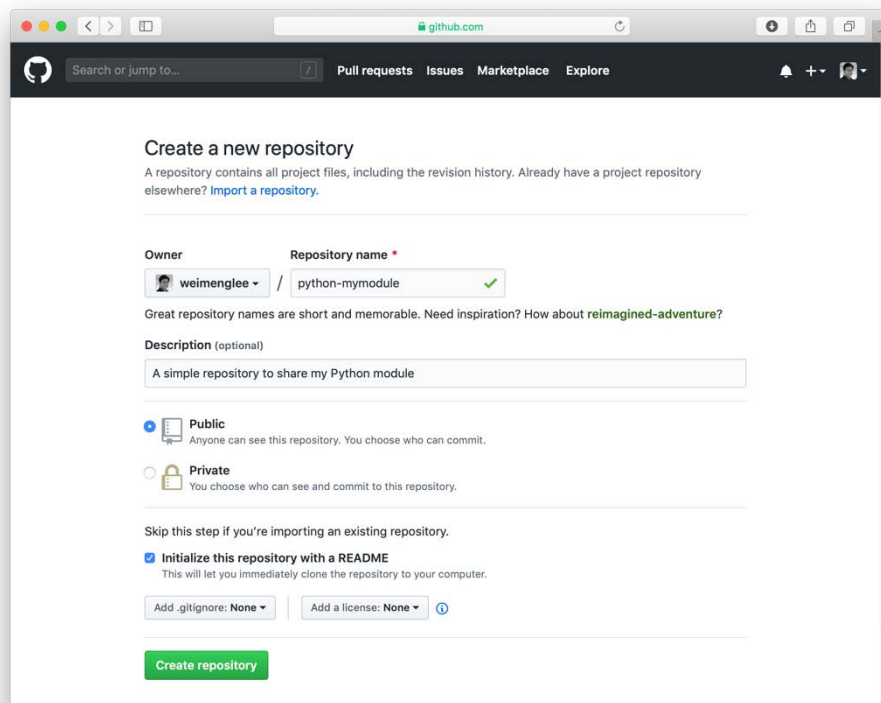
Repositories can contain folders and files, images, videos, spreadsheets, and data sets – anything your project needs

You should include a README, or a file with information about your project

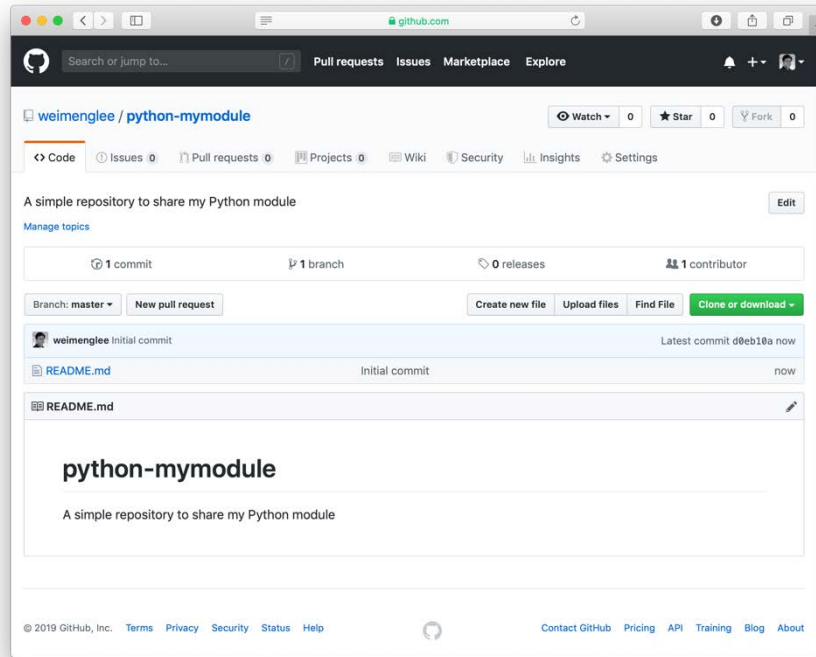
1. Click on the New button to create a new repository:



2. Enter the following information and click the Create repository button at the bottom of the screen:



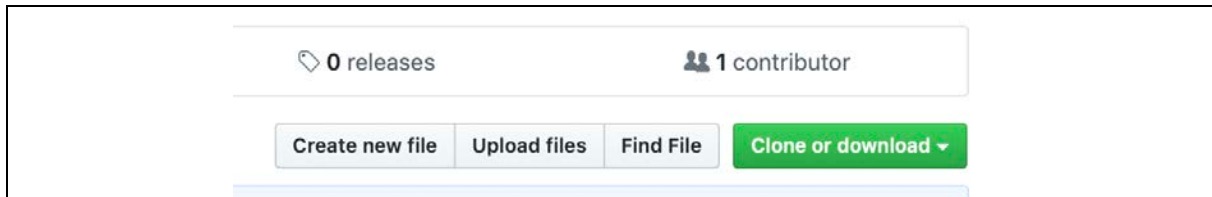
3. Your repository would now be created:



At this moment, your repository only has a single file – README.md

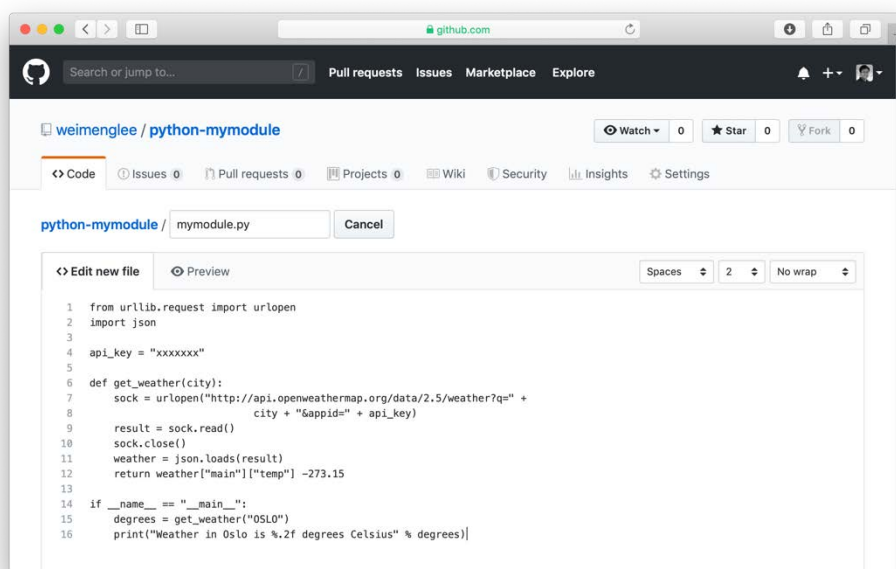
Adding Files to your Repository

1. Click on the Create new file button to create a new file in your repository:



If you already have created your file(s) that you want to upload to GitHub, it is easier to click the Upload files button to upload them to your repository

2. Name the file as mymodule.py:



3. Add in the following statements in bold:

```
from urllib.request import urlopen
import json

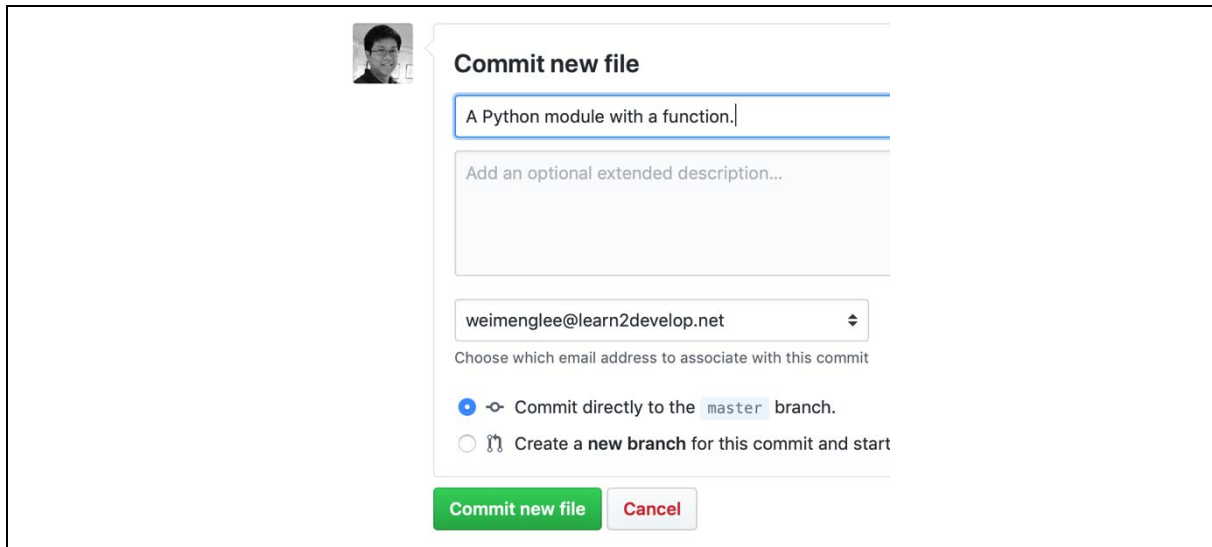
api_key = "xxxxxxx"

def get_weather(city):
    sock = urlopen("http://api.openweathermap.org/data/2.5/weather?q=" +
                  city + "&appid=" + api_key)
    result = sock.read()
    sock.close()
    weather = json.loads(result)
    return weather["main"]["temp"] - 273.15

if __name__ == "__main__":
    degrees = get_weather("OSLO")
```

```
print("Weather in Oslo is %.2f degrees Celsius" % degrees)
```

4. Give a description to the file and click Commit new file:



Commit new file

A Python module with a function.

Add an optional extended description...

weimenglee@learn2develop.net

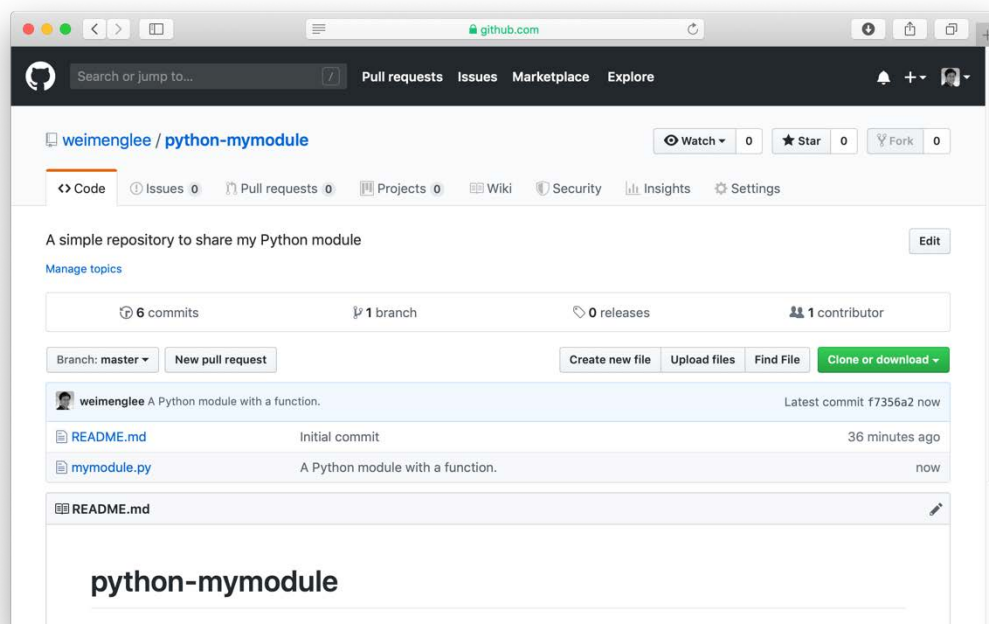
Choose which email address to associate with this commit

☒ Commit directly to the `master` branch.

☐ Create a new branch for this commit and start

Commit new file **Cancel**

5. You should now see the mymodule.py file in your repository:



Creating a Branch

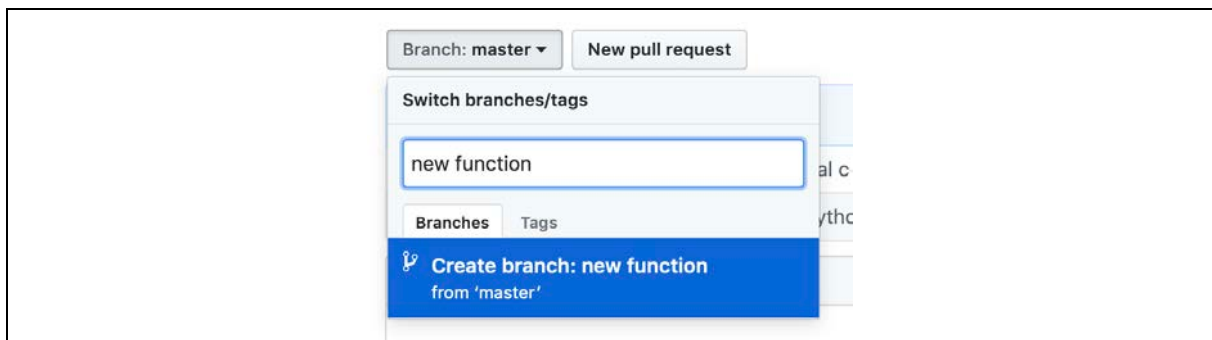


Branching is the way to work on different versions of a repository at one time

By default your repository has one branch named master which is considered to be the definitive branch

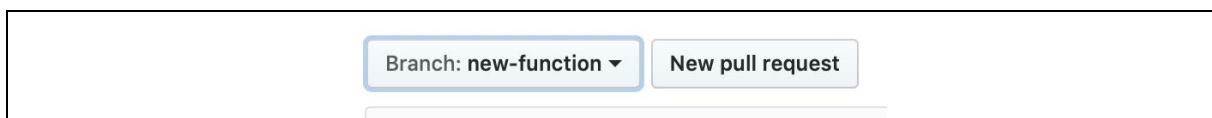
You use branches to experiment and make edits before committing them to master

1. Click the Branch:master drop-down list and type in a new branch name – “new function”:



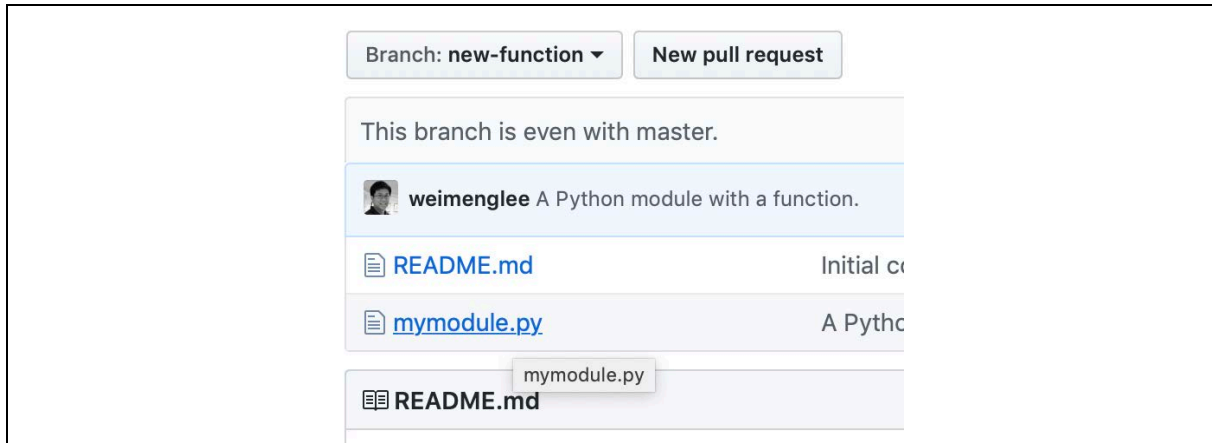
Then, click the button Create branch: new function.

2. You will now see your new branch created:

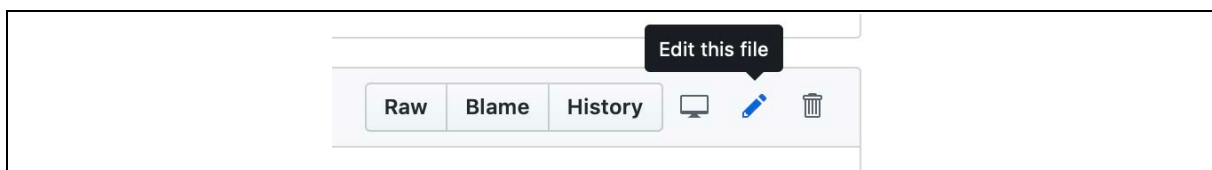


The new-function branch is a copy of master

3. Click on the mymodule.py file:



4. You will now see the content of mymodule.py. Click the pencil icon located on the right of the page to edit the page:



5. Add in the following statements in bold:

```
from urllib.request import urlopen
import json


api_key = "xxxxxxx"

def get_weather(city):
    sock = urlopen("http://api.openweathermap.org/data/2.5/weather?q=" +
                    city + "&appid=" + api_key)
    result = sock.read()
    sock.close()
    weather = json.loads(result)
    return weather["main"]["temp"] - 273.15

def postal_lookup(postal_code):
    sock = urlopen("http://api.postcodes.io/postcodes/" + postal_code)
    result = sock.read()
    sock.close()
    details = json.loads(result)
    return (details["result"]["latitude"], details["result"]["longitude"])

if __name__ == "__main__":
    degrees = get_weather("OSLO")
    print("Weather in Oslo is %.2f degrees Celsius" % degrees)
    location = postal_lookup("B323PP")
    print(location)
```

6. Enter the following details and click Commit changes:



Commit changes

Added another function called postal_lookup()

The postal_lookup() takes in a postal code and returns the latitude and longitude of locations bearing that postal code.

weimenglee@learn2develop.net

Choose which email address to associate with this commit

☒ Commit directly to the `new-function` branch.
 ☐ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

Commit changes Cancel

7. Click on the python-module link:

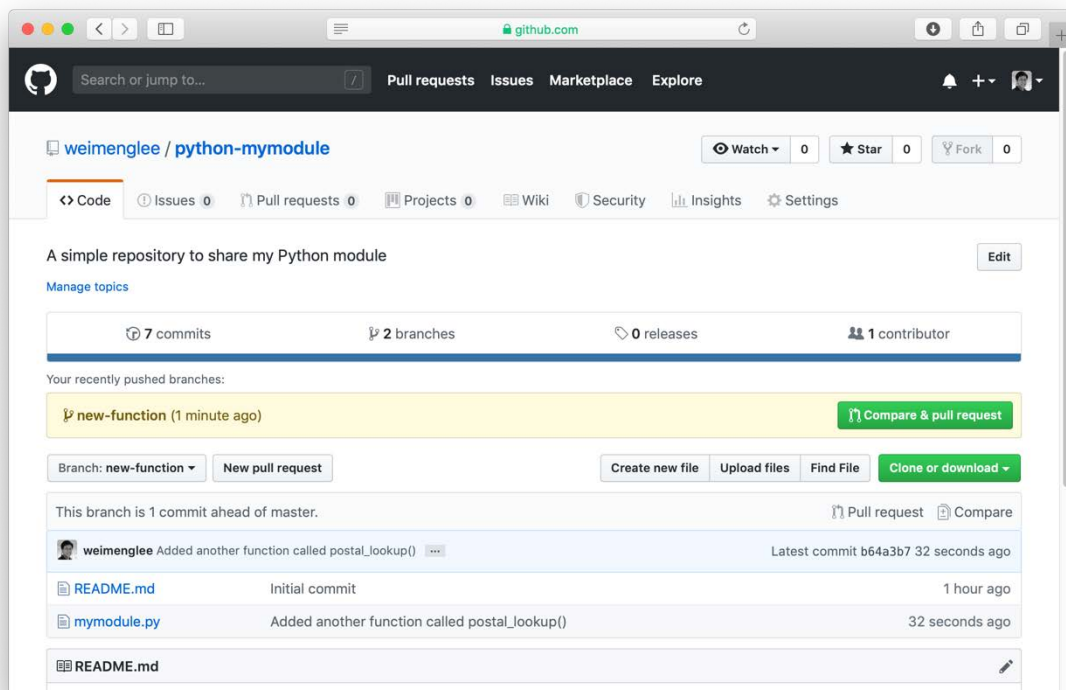
Branch: new-function

python-mymodule / mymodule.py


 weimenglee A Python Module with two functions.

1 contributor

8. You should now see the following:



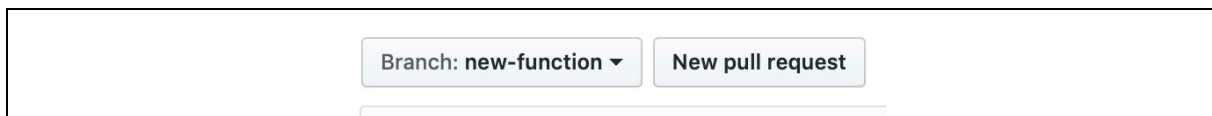
The screenshot shows the GitHub repository page for 'weimenglee / python-mymodule'. The repository is described as 'A simple repository to share my Python module'. It has 7 commits, 2 branches, 0 releases, and 1 contributor. The 'new-function' branch is selected, which is 1 commit ahead of master. The commit history shows two commits: 'Initial commit' for 'README.md' and 'Added another function called postal_lookup()' for 'mymodule.py'. The 'mymodule.py' file is highlighted in the file list.

Pull Request

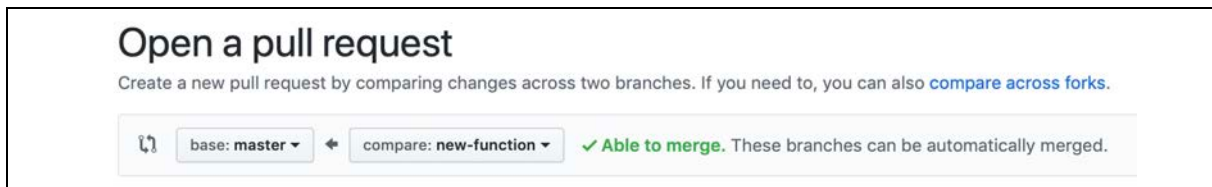


A pull request (or PR) is a way to alert a repository's owners that you want to make some changes to their code. It allows them to review the code and make sure it looks good before putting your changes on the master branch.

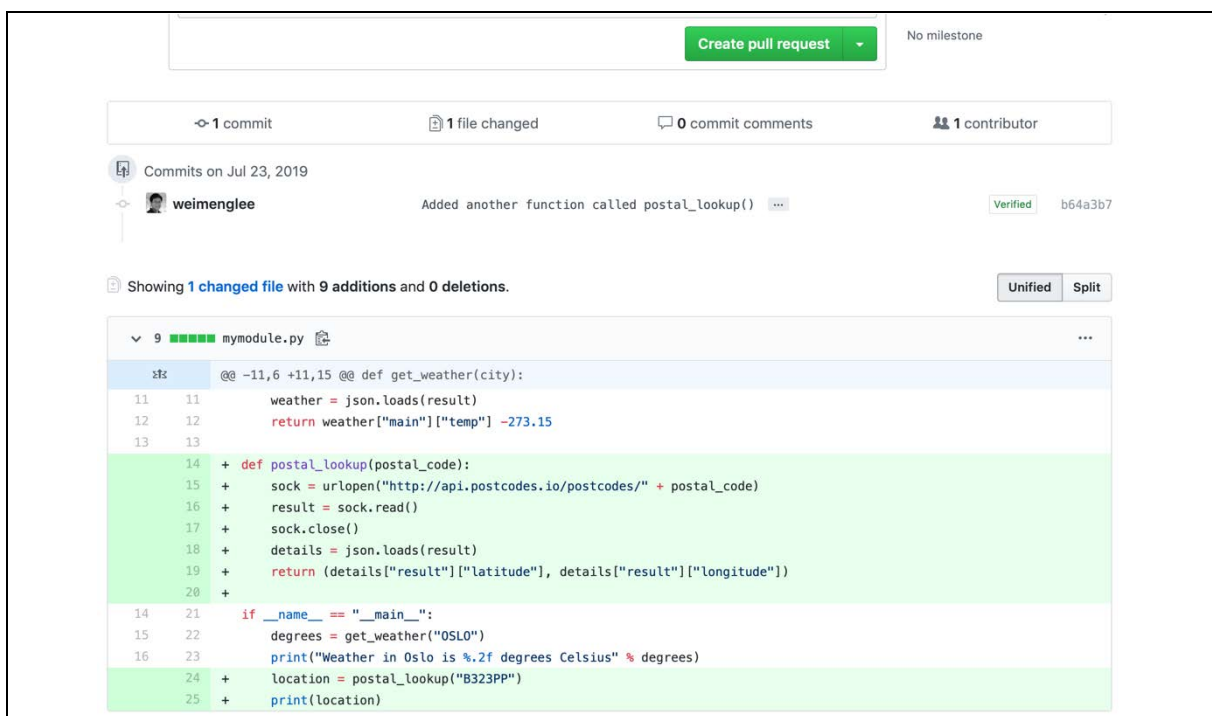
1. Click the New pull request button:




2. You will be able to compare the master with the new-function branch:



3. Look at the bottom of the page and see if the changes are what you want. Click the Create pull request button:




4. Give a description to your pull request and click Comment:





WritePreview

AA B i “ > ↺ ⋮ ⋮ ⋮ @ 📎 ↶

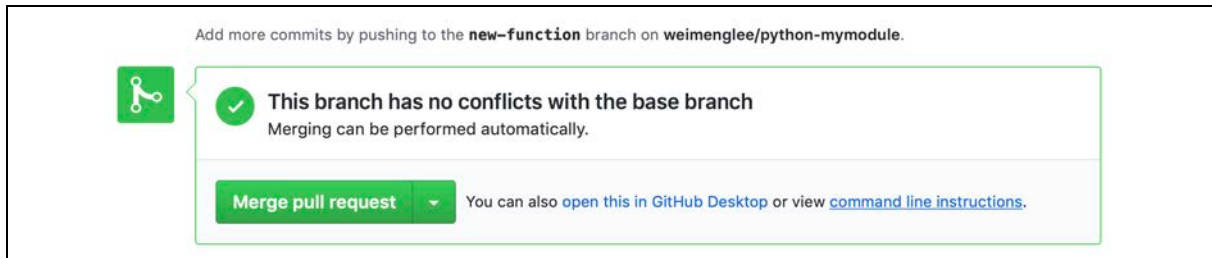
Added postal_lookup()

Attach files by dragging & dropping, selecting or pasting them. 

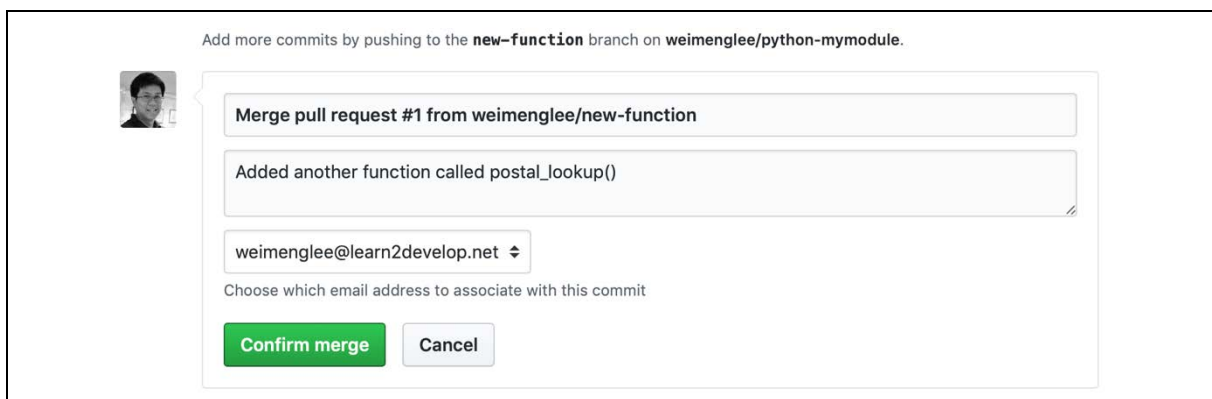
Merging Pull Request

1. Click Merge pull request:

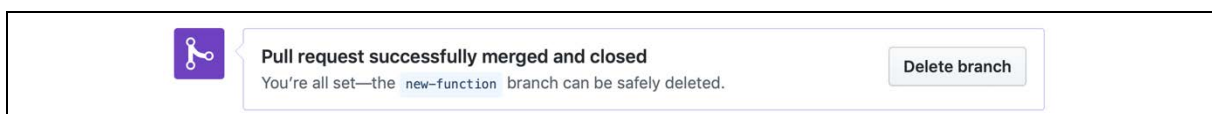


Merging pull request will merge the changes to the master branch

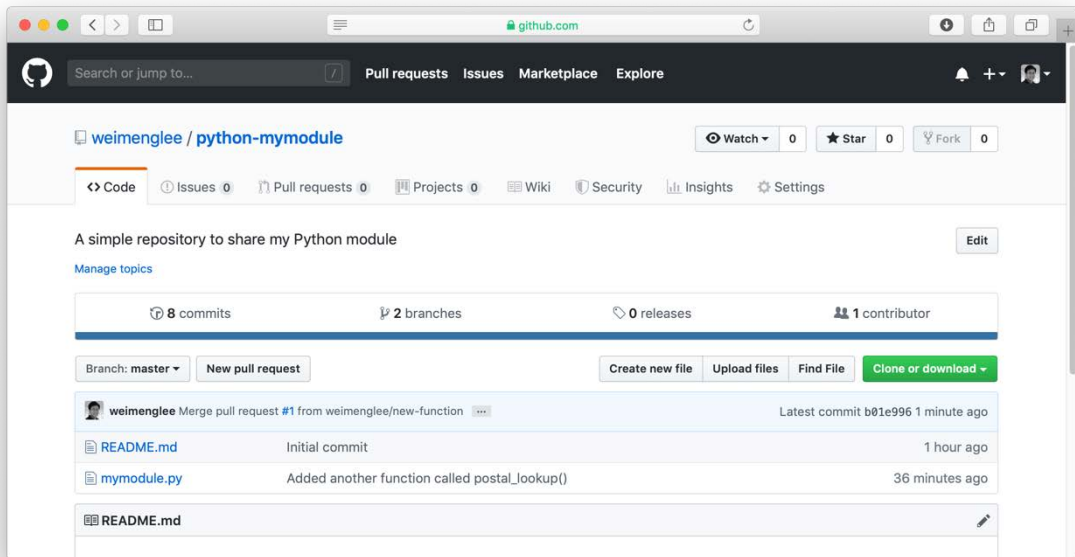
2. Click Confirm merge:



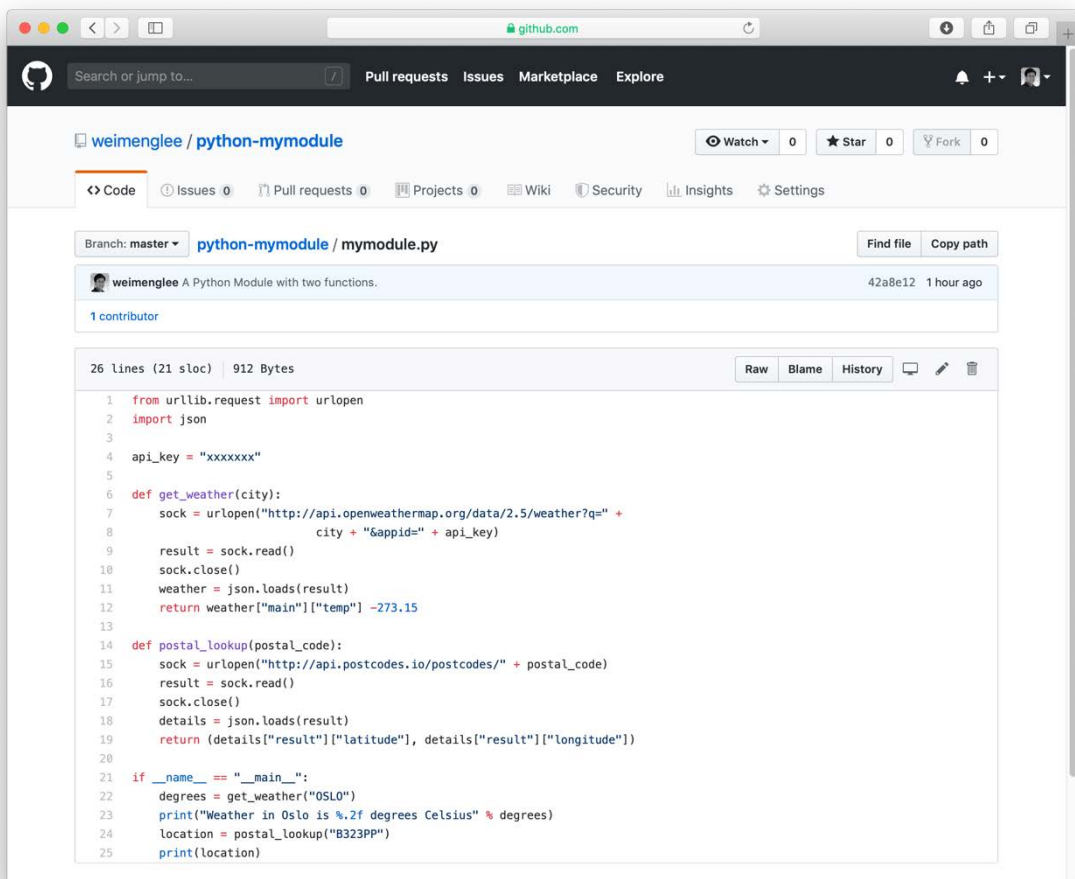
3. You can now delete the branch. Click Delete branch:



4. Go back to the main page for your repository. Click the mymodule.py file:

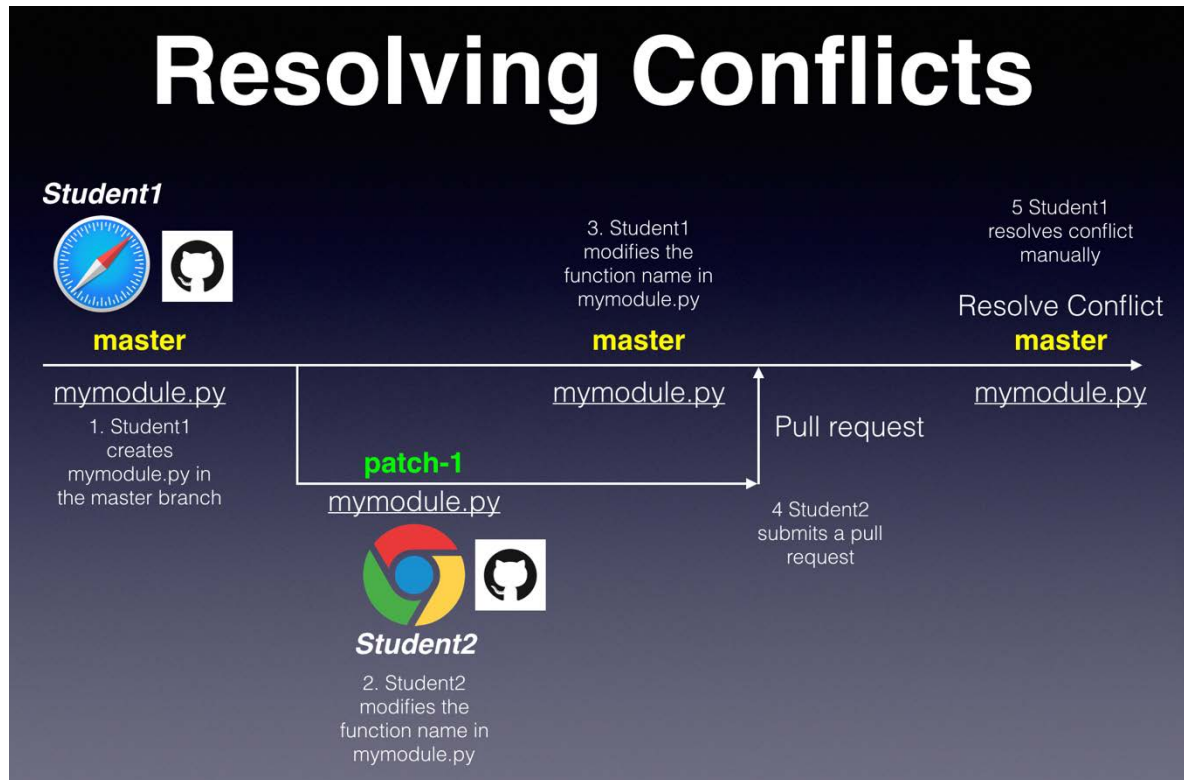


5. You should now see that your module has two functions:



Lab 2. Resolving Merge Conflicts

Description	<i>In this lab, you will learn how to resolve merge conflicts in pull requests.</i>
What You Will Learn	<ul style="list-style-type: none"> • How to fork a repo • How to merge conflicting pull requests
Duration	30 minutes



For this lab, we will make use of two GitHub Accounts:

- Student 1 - student1.7896
- Student 2 - student2.7896

For illustration, we shall use two Web browsers to demonstrate:

- Student 1 will login using Safari
- Student 2 will login using Chrome

Student 1




1. Using Safari, login to GitHub for student1.7896.
2. Create a new repository:

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner

Repository name *

 student17896

/

My Project

Great repository names are s Your new repository will be created as My-Project it upgraded-parakeet?

Description (optional)

☒
Public

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

☐
Private

You choose who can see and commit to this repository.

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.

Add .gitignore: None

Add a license: None

Create repository

3. Click Create new file to add a new file to your repository:

0 releases

1 contributor

Create new file

Upload files

Find File

Clone or download

4. Name the file mymodule.py and add the following statements in bold:

My-Project / mymodule.py Cancel

<> Edit new file Preview

```

1  from urllib.request import urlopen
2  import json
3
4  api_key = "xxxxxxx"
5
6  def get_weather(city):
7      sock = urlopen("http://api.openweathermap.org/data/2.5/weather?q=" +
8                      city + "&appid=" + api_key)
9      result = sock.read()
10     sock.close()
11     weather = json.loads(result)
12     return weather["main"]["temp"] - 273.15
13
14  if __name__ == "__main__":
15     degrees = get_weather("OSLO")
16     print("Weather in Oslo is %.2f degrees Celsius" % degrees)
17

```

```

from urllib.request import urlopen
import json


api_key = "xxxxxxx"

def get_weather(city):
    sock = urlopen("http://api.openweathermap.org/data/2.5/weather?q=" +
                    city + "&appid=" + api_key)
    result = sock.read()
    sock.close()
    weather = json.loads(result)
    return weather["main"]["temp"] - 273.15

if __name__ == "__main__":
    degrees = get_weather("OSLO")
    print("Weather in Oslo is %.2f degrees Celsius" % degrees)

```

- Click Commit new file to save the file:



Commit new file

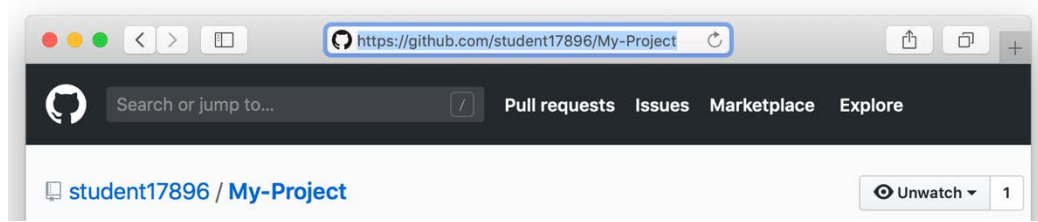
Create mymodule.py

Add an optional extended description...

☒ Commit directly to the `master` branch.
 ☐ Create a new branch for this commit and st

Commit new file Cancel

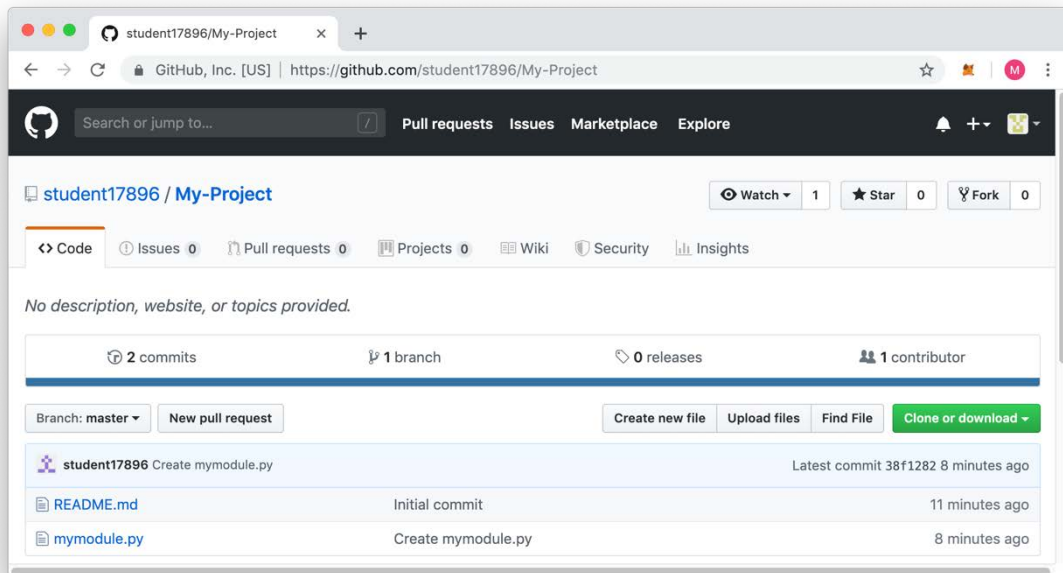
- Take note of the URL for the repository:



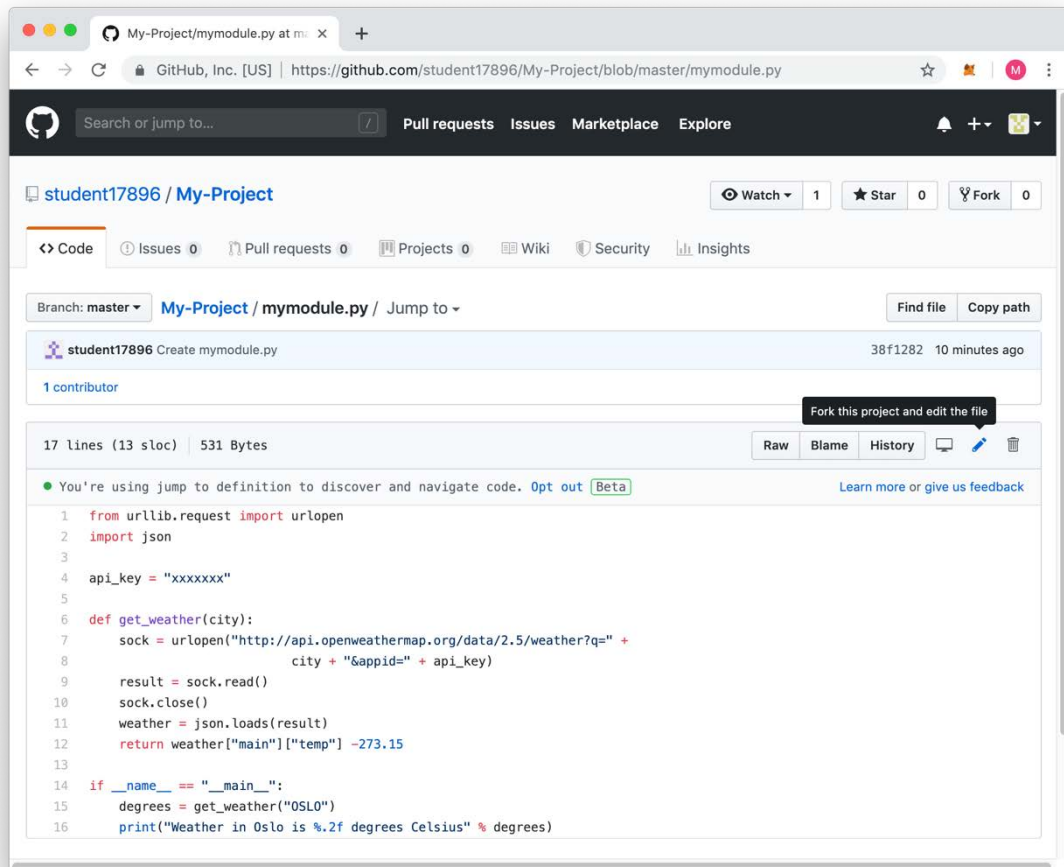
Student 2



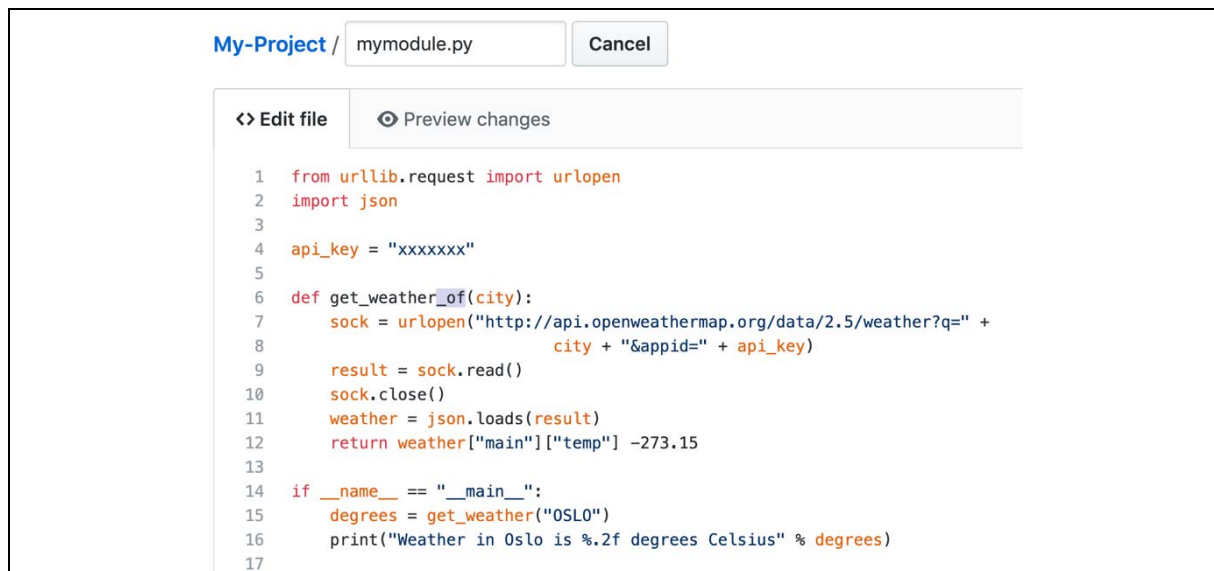
1. Using Chrome, login to GitHub for student2.7896.
2. Go to the repository for Student1:




3. Click on the mymodule.py file and click the pencil icon to edit it:



4. Change the name of the `get_weather()` function to `get_weather_of`:



5. At the bottom of the page, type the description and click **Propose file change**:



Propose file change

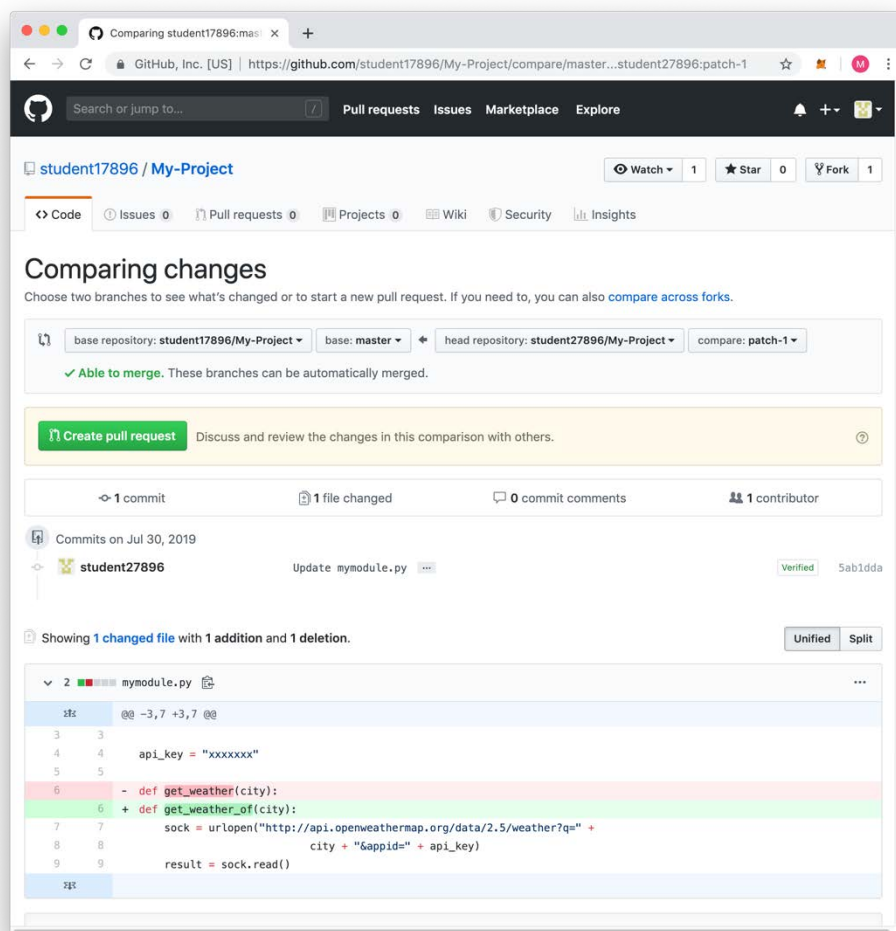
Update mymodule.py

Changed function name of get_weather() to get_weather_of().

Propose file change

Cancel

6. You should see the following:



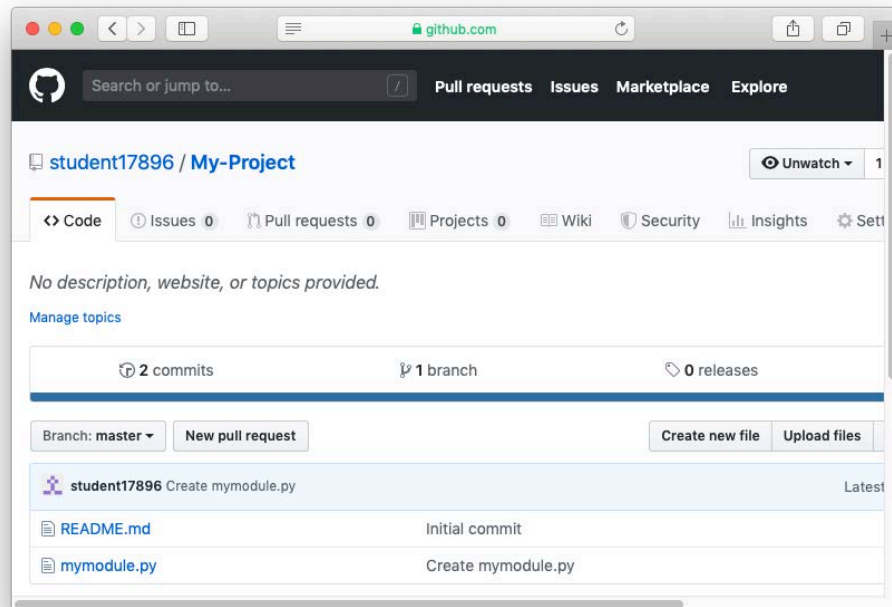
Essentially, at this moment you have forked the repo

7. You will come back to this page shortly.

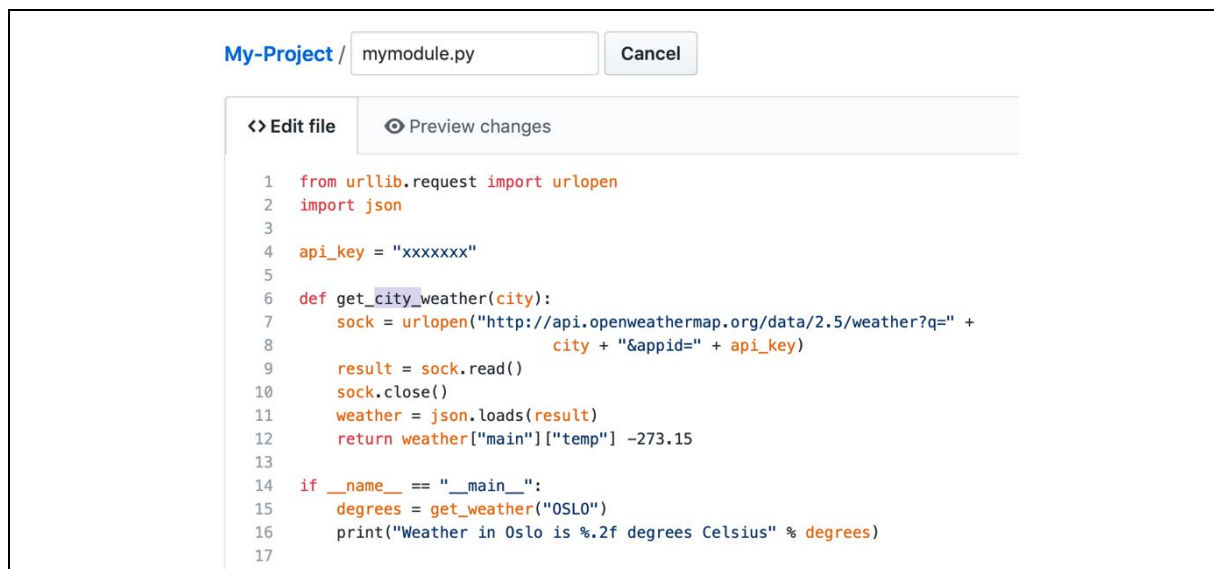
Student 1



1. Back in Safari, click the mymodule.py file and then the pencil icon to edit it:



2. Change the name of the get_weather() function to get_city_weather:



3. At the bottom of the page, type the following description and click Commit changes:



Commit changes

Update mymodule.py

Changed the function name of get_weather() to get_city_weather().

- ☒ Commit directly to the `master` branch.
- ☐ Create a new branch for this commit and start a pull request. [Learn more](#)

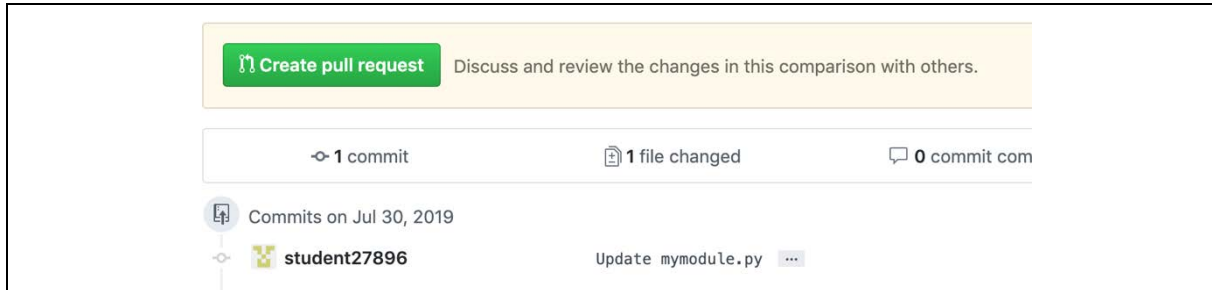
Commit changes

Cancel

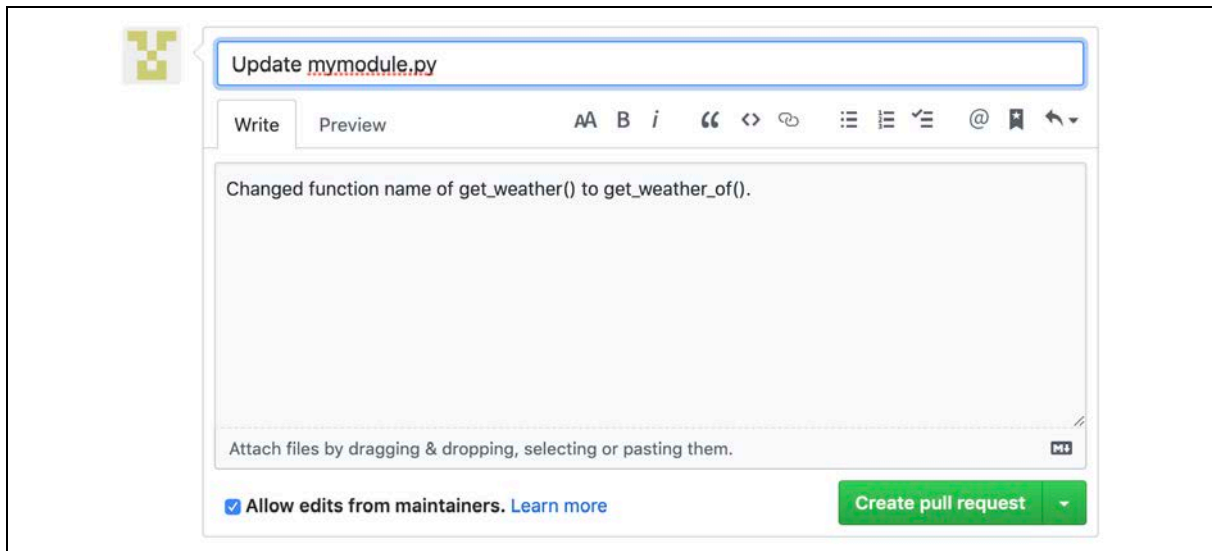
Student 2



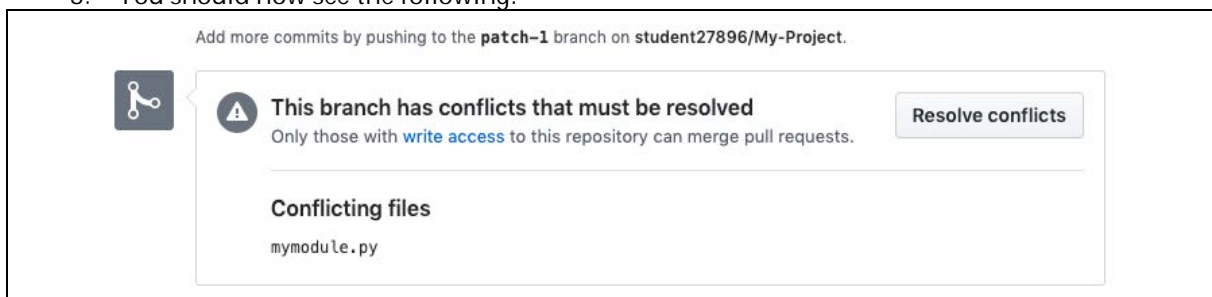
1. Back in Chrome, click the Create pull request button:



2. Type the description as follows and click the Create pull request button:



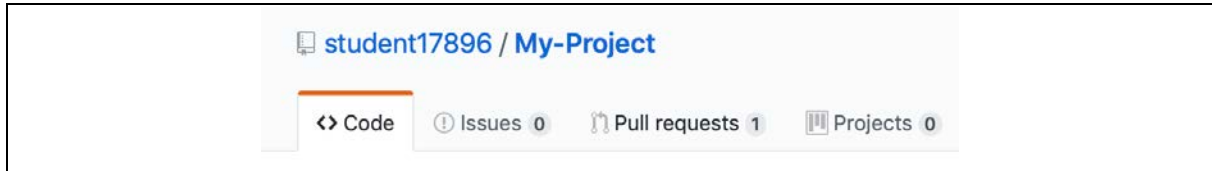
3. You should now see the following:



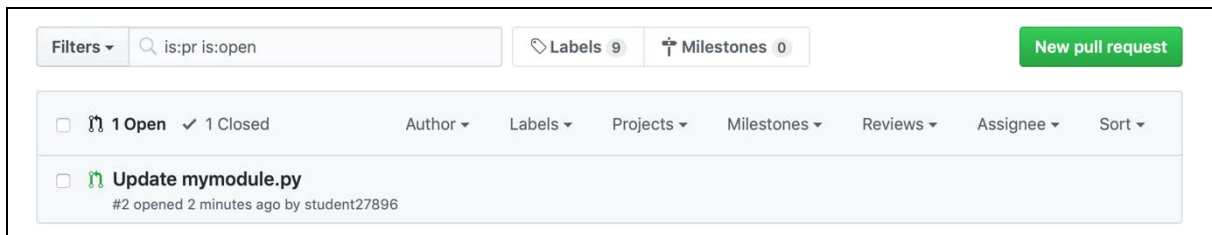
Student 1



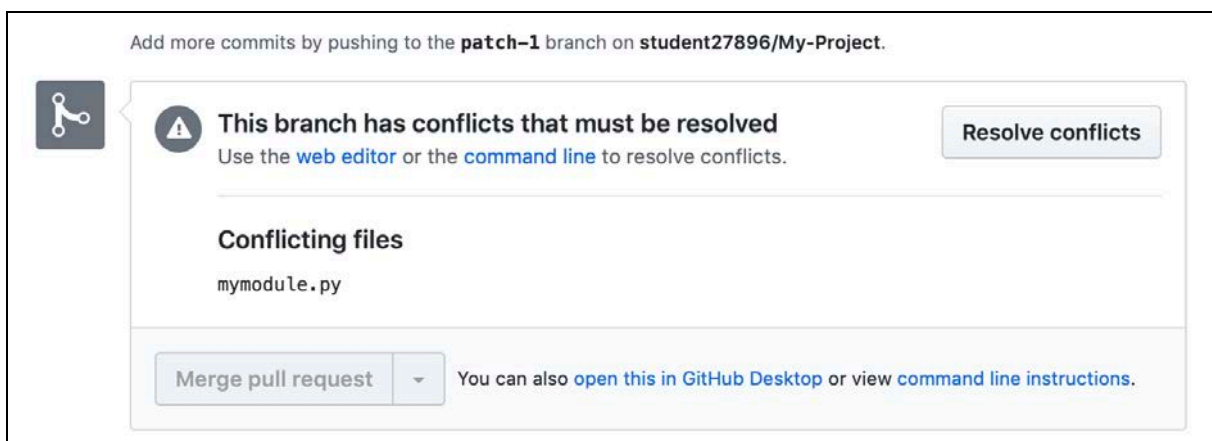
1. Back in Safari, observe that there is now a Pull requests. Click on it.



2. Click on the Update mymodule.py pull request:



3. Click the Resolve conflicts button:



4. You will see the following:

mymodule.py

1 conflict

```

1  from urllib.request import urlopen
2  import json
3
4  api_key = "xxxxxxx"
5
6  <<<<<<< patch-1
7  def get_weather_of(city):|
8  =====
9  def get_city_weather(city):
10 >>>>>>> master
11     sock = urlopen("http://api.openweathermap.org/data/2.5/weather?q=" +
12                    city + "&appid=" + api_key)
13     result = sock.read()
14     sock.close()
15     weather = json.loads(result)
16     return weather["main"]["temp"] -273.15
17
18 if __name__ == "__main__":
19     degrees = get_weather("OSLO")
20     print("Weather in Oslo is %.2f degrees Celsius" % degrees)
21

```

5. Change the name of the function to the following:

mymodule.py

1 conflict

```

1  from urllib.request import urlopen
2  import json
3
4  api_key = "xxxxxxx"
5
6  def get_weather_of_city(city):
7      sock = urlopen("http://api.openweathermap.org/data/2.5/weather?q=" +
8                     city + "&appid=" + api_key)
9      result = sock.read()
10     sock.close()
11     weather = json.loads(result)
12     return weather["main"]["temp"] -273.15
13
14 if __name__ == "__main__":
15     degrees = get_weather("OSLO")
16     print("Weather in Oslo is %.2f degrees Celsius" % degrees)
17

```

6. Click Mark as resolved:

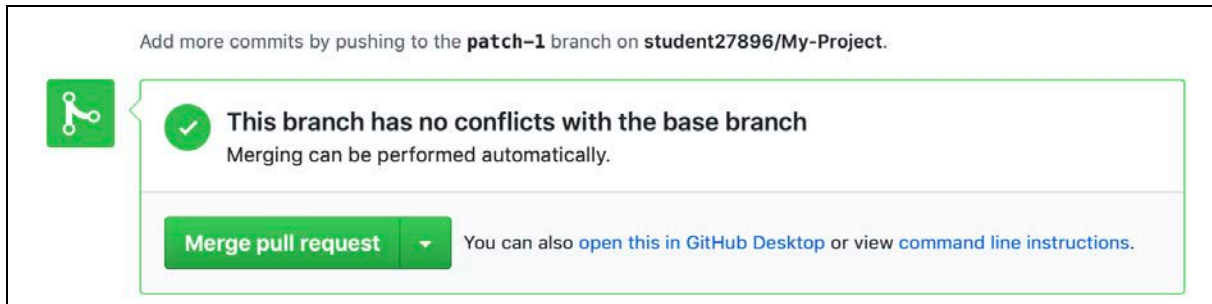
Mark as resolved

7. Then, click the Commit merge button:

Commit merge

✓ Resolved

8. Click Merge pull request:



9. Click Confirm merge:



10. The content of mymodule.py would now look like this:

```
from urllib.request import urlopen
import json

api_key = "xxxxxxx"

def get_weather_of_city(city):
    sock = urlopen("http://api.openweathermap.org/data/2.5/weather?q=" +
                    city + "&appid=" + api_key)
    result = sock.read()
    sock.close()
    weather = json.loads(result)
    return weather["main"]["temp"] - 273.15

if __name__ == "__main__":
    degrees = get_weather("OSLO")
    print("Weather in Oslo is %.2f degrees Celsius" % degrees)
```