Member-only story

How to Build an App with React + Flask

Concise and Simple

Ran (Reine) · Follow
Published in The Startup
4 min read · Dec 26, 2020

Listen Share

Image from link

Note: If you'd like to read about building + deploying your React-Flask project, do check out this tutorial.

I mostly did AI related stuff back in my undergrad years but haven't had the chance to go into web development until my final semester of school. Even though AI was really interesting, I figured that the only way I can show others what I'm doing with it is to put it on the web.

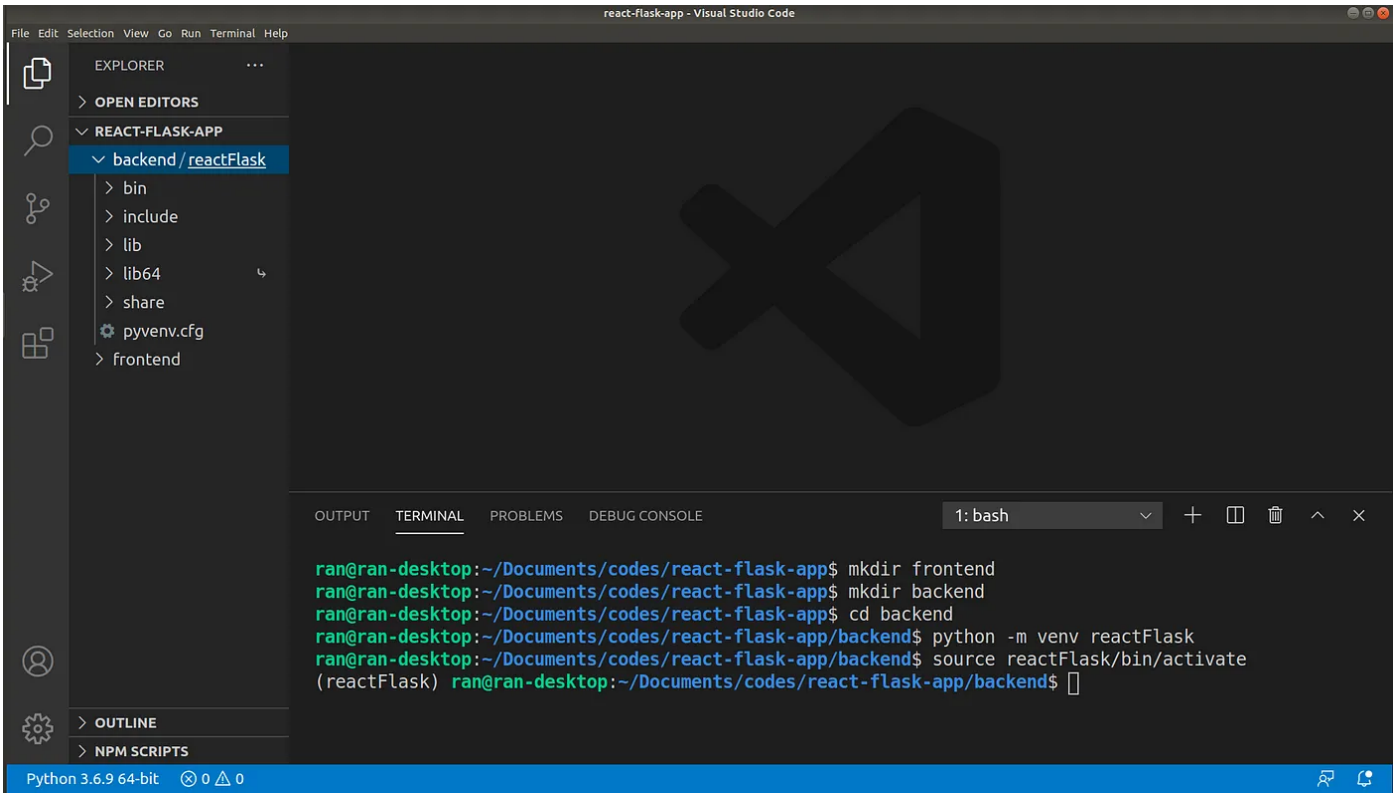
With that thought, I learned Flask and React separately and these days I wanted to figure out how to use both Flask in conjunction with React so I can create applications with better looking interfaces.

The problem is, the first few tutorials I looked at were way too complicated (like, I just wanna make an API call from React front-end to Flask back-end that's it, do I really have to run eject, use another UI framework, etc?).

After a few tries, here's one that (I think) is really concise (probably will only take you 10 mins to complete).

First things first, here's the GitHub Repo for easy reference — link.

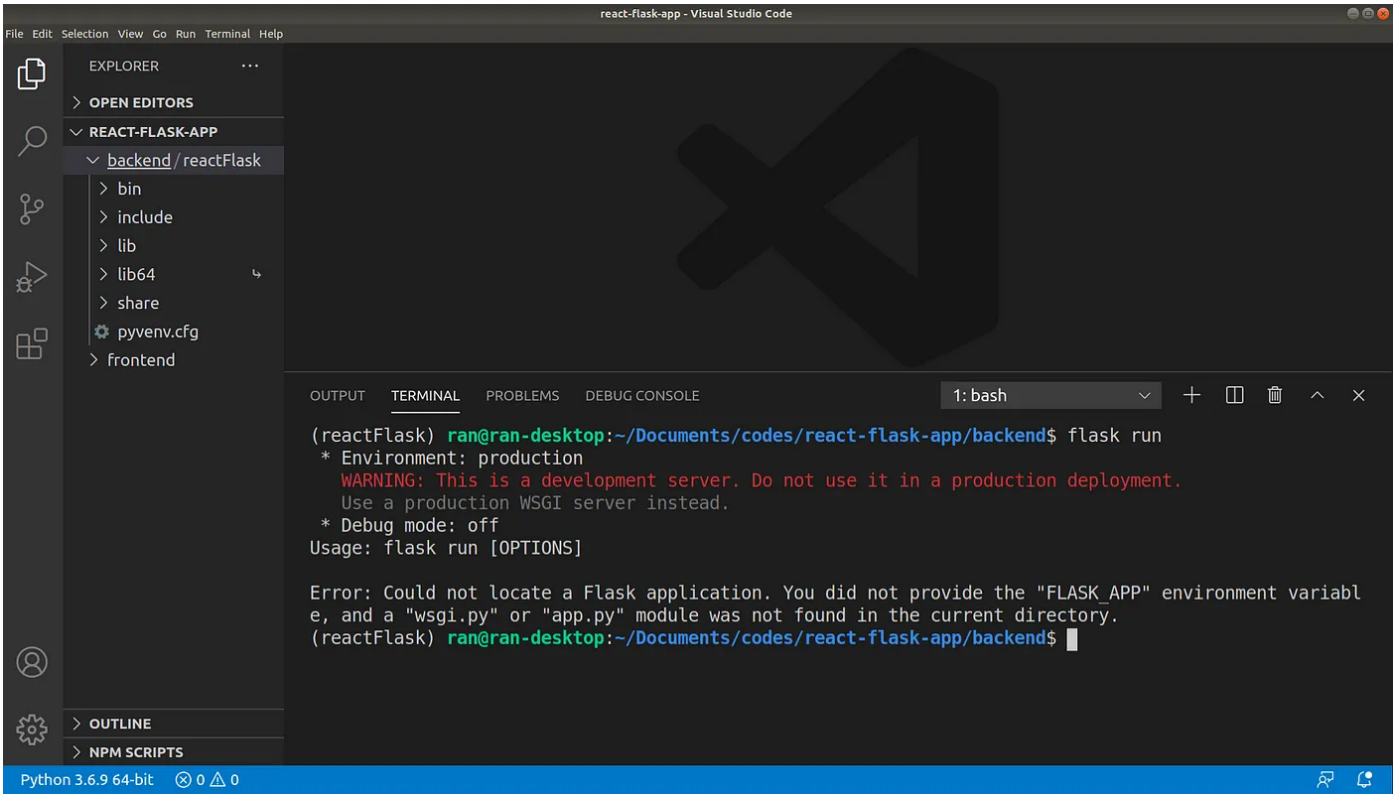
Back-end



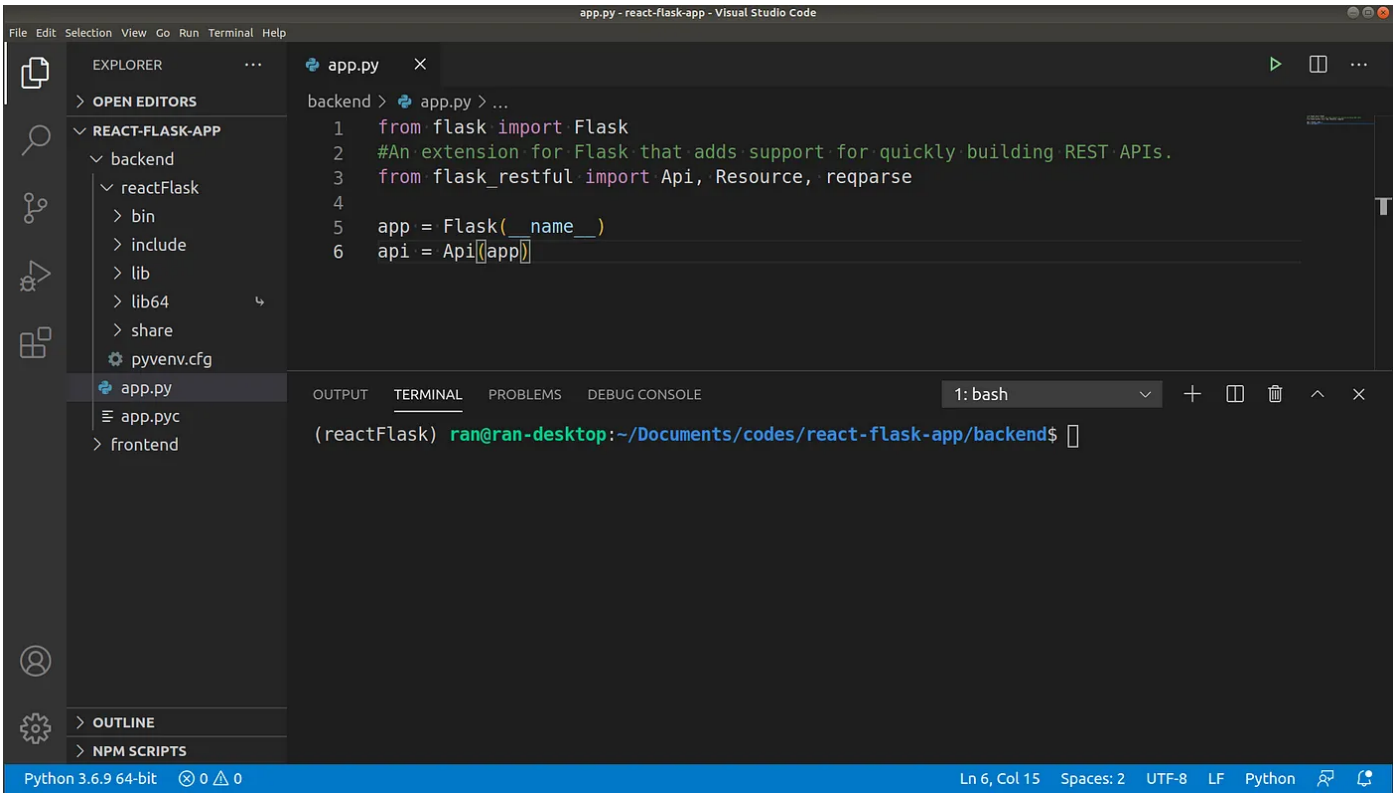
Project Set Up

```
conda create -n reactFlask
conda activate reactFlask
pip install Flask
pip install  flask-restful
```

We'll start off by creating two directories (one for back-end and one for front-end). Although not strictly necessary, you're highly encouraged to use s virtual environment so that the versions of the python libraries and packages don't conflict with your other projects.

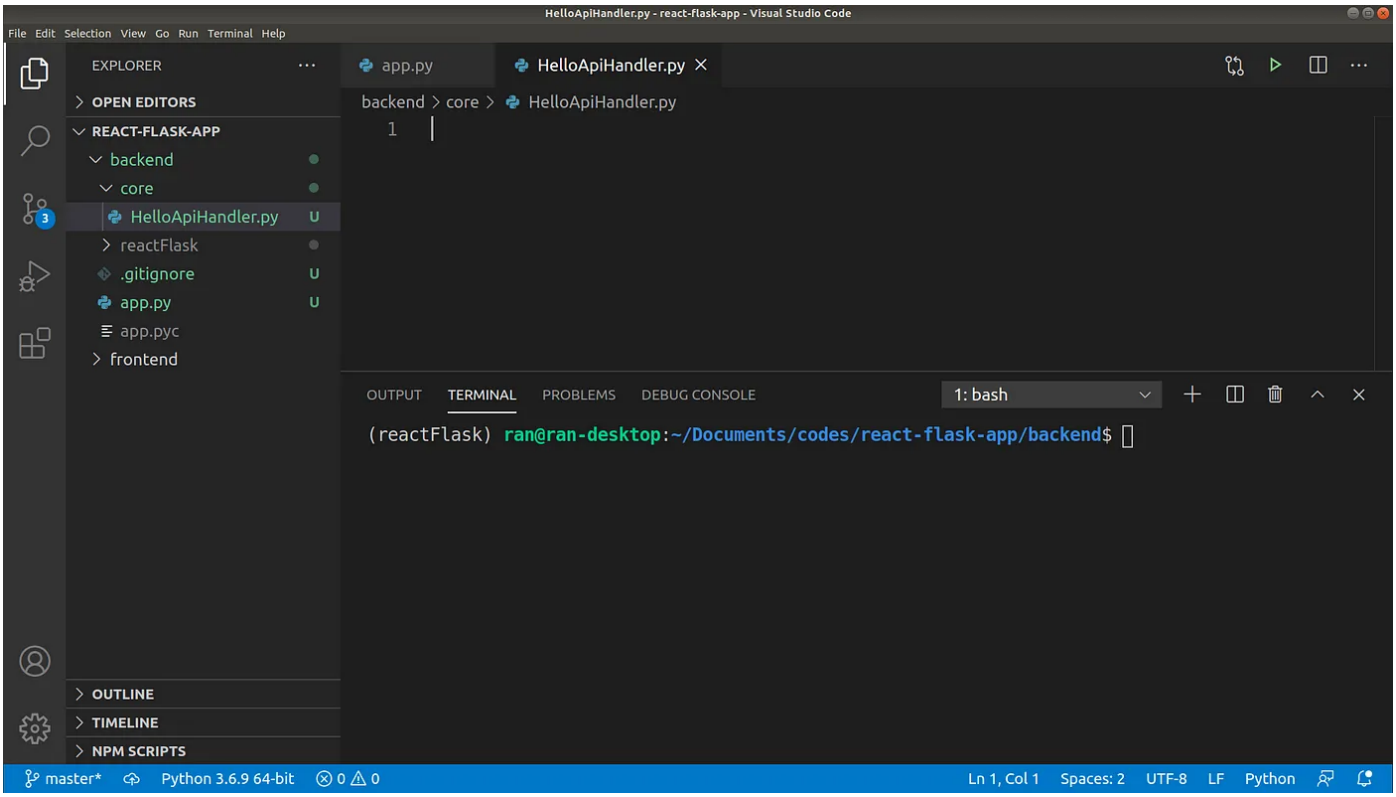


Here's what happens when you run flask without app.py



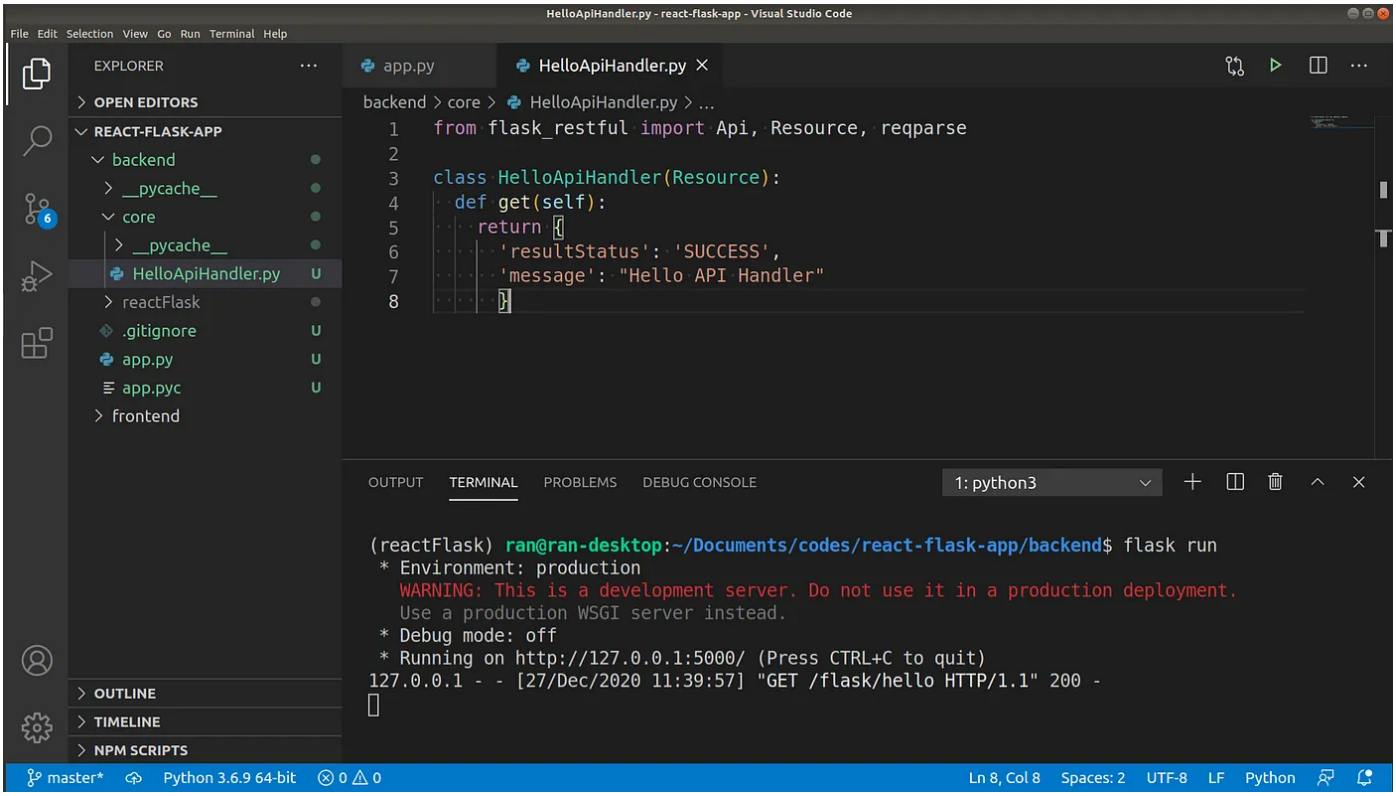
So we create a file app.py in the backend directory

We'll be using flask_restful (an extension for Flask that provides additional support for building REST APIs). pip install flask-restful (did above)

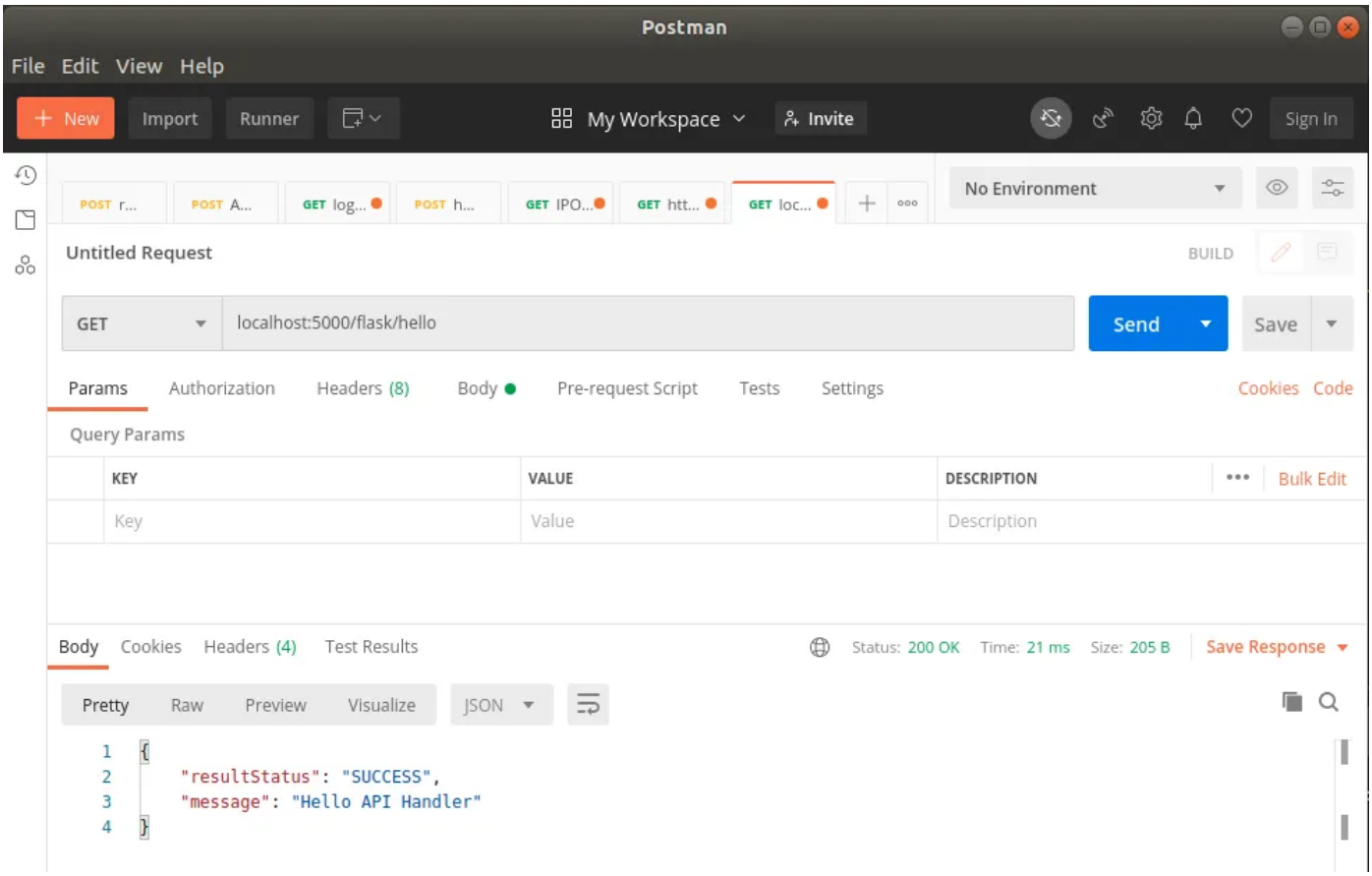


Writing our Hello Handler

In backend directory, create a core directory. This is where all our API handlers will reside in — we technically can just put this in the app.py file itself too but I'm just putting it here for a clean structure.



We'll just write a very simple return when this API endpoint is called

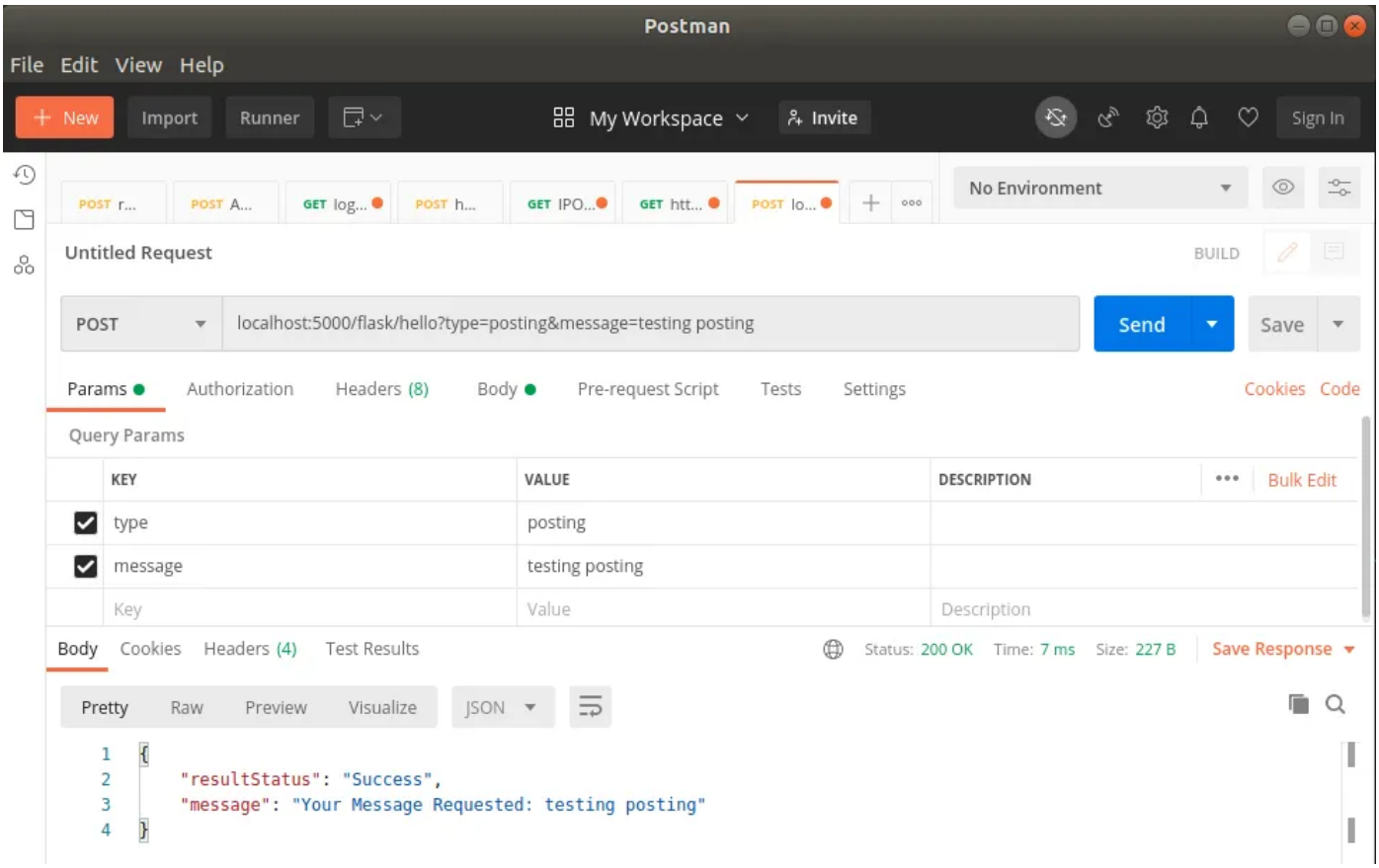


It works!

Okay now let's write something for a POST request in the same HelloAPIHandler.py file

```python
def post(self):
    parser = reqparse.RequestParser()
    parser.add_argument('type', type=str)
    parser.add_argument('message', type=str)

    args = parser.parse_args()

    #note, the post req from frontend needs to match the strings here (e.g. 'type and 'mes

    request_type = args['type']
    request_json = args['message']
    # process request_type and request_json here
    # currently just returning directly for demo purposes
    ret_status = request_type
    ret_msg = request_json

    if ret_msg:
      message = "Your Message Requested: {}".format(ret_msg)
    else:
      message = "No Msg"

    final_ret = {"resultStatus": "Success", "message": message}

    return final_ret
```

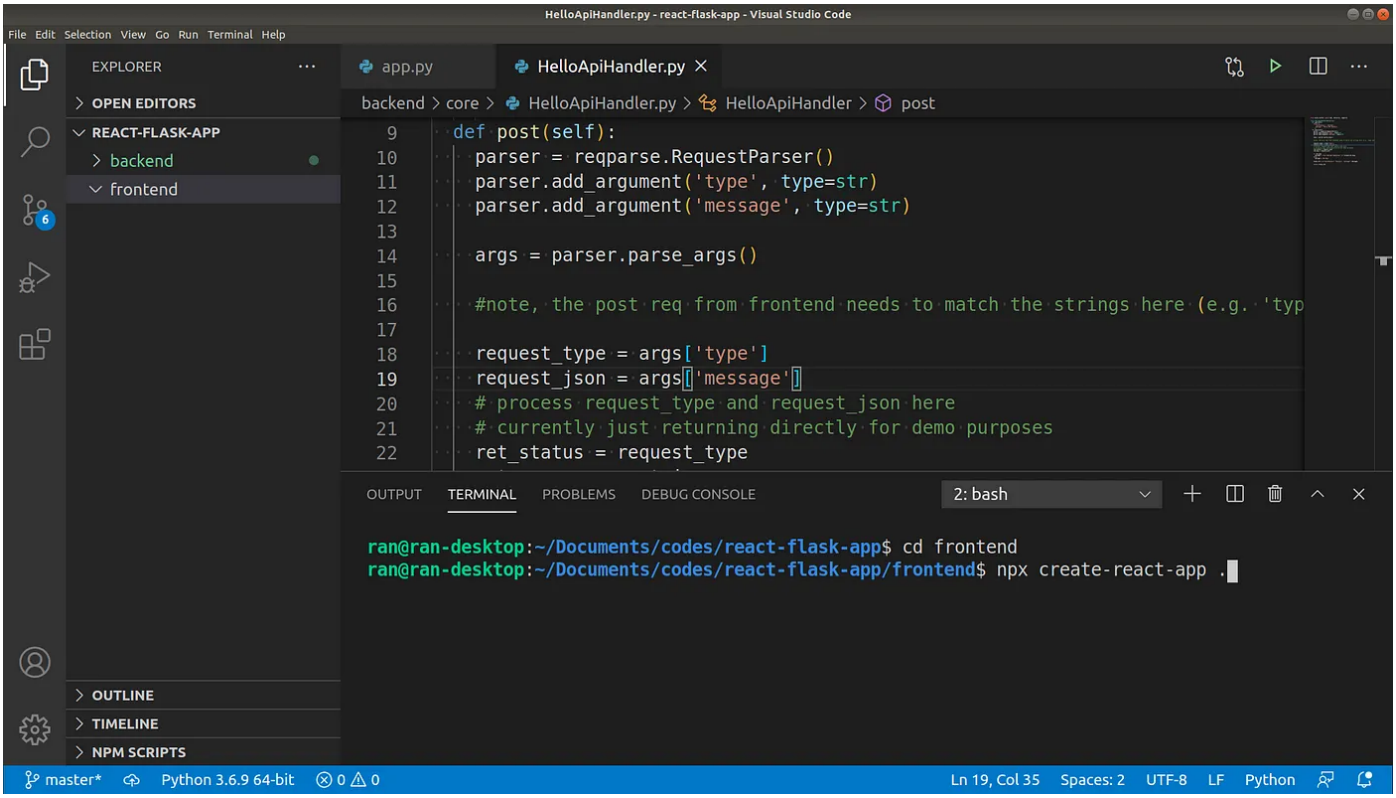
HelloApiHandler.py hosted with ❤️ by GitHub view raw



Works for POST too! :)

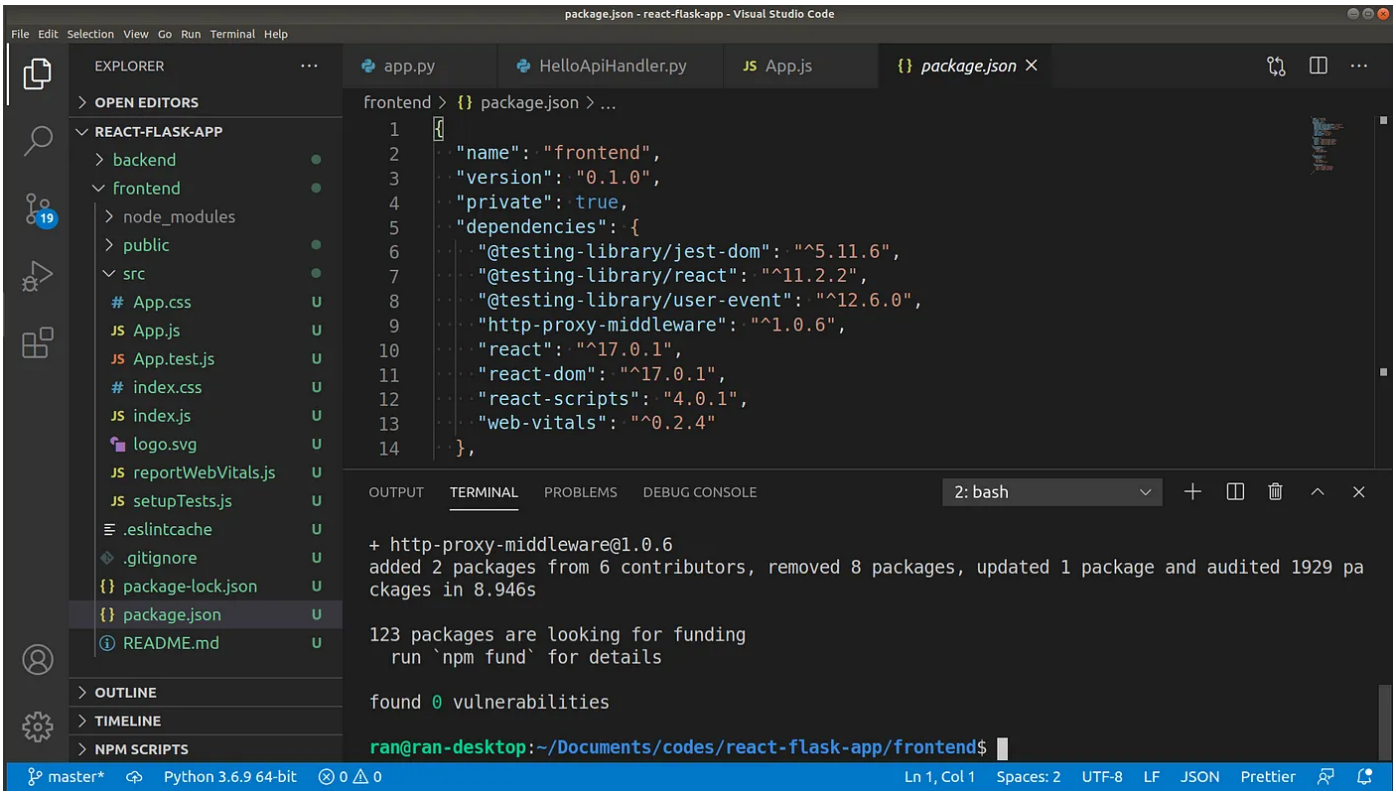
Okay so now we're all done with the back-end codes!

Front-end

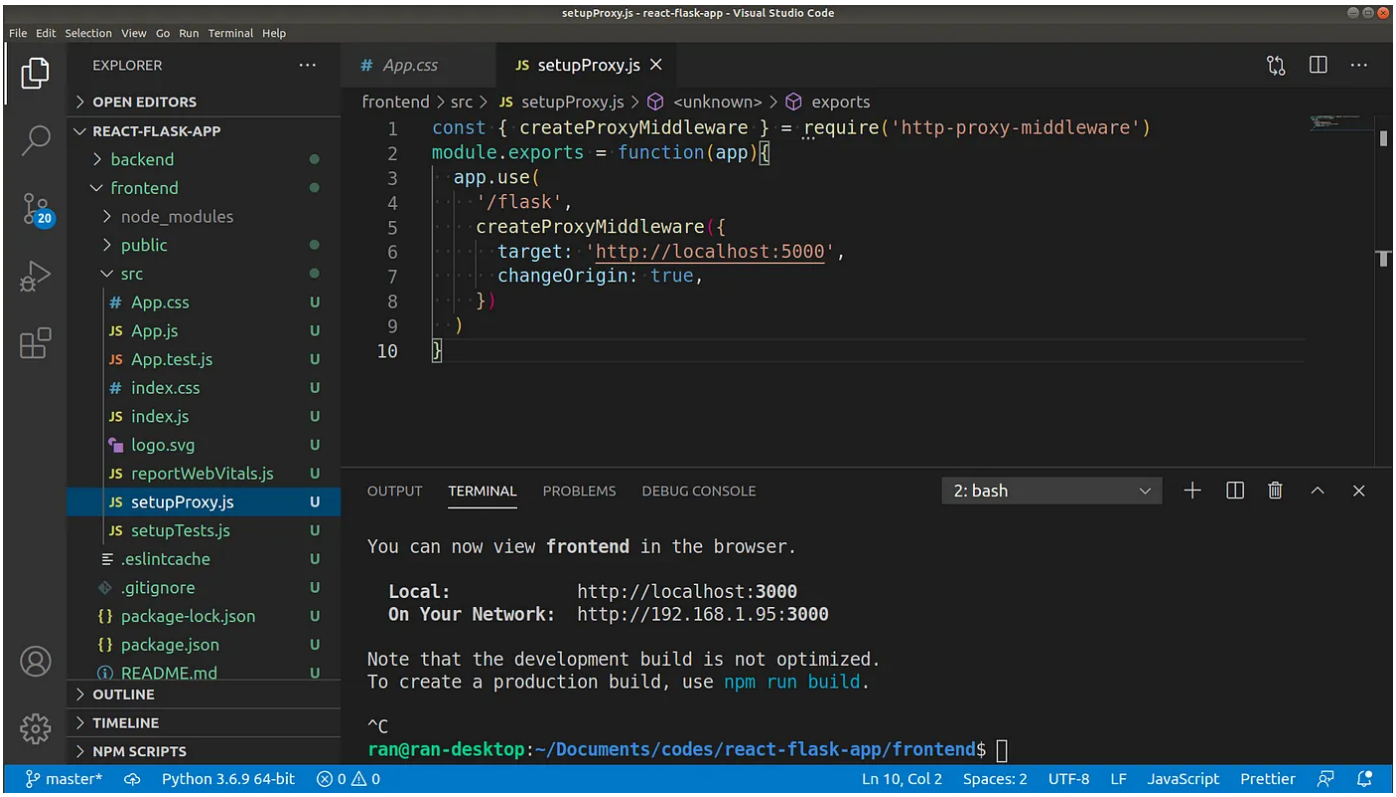


run create-react-app

Since we're using React, let's start by creating a React app. In order not to over-complicate this tutorial, we'll just display our GET and POST messages in the App.js page.

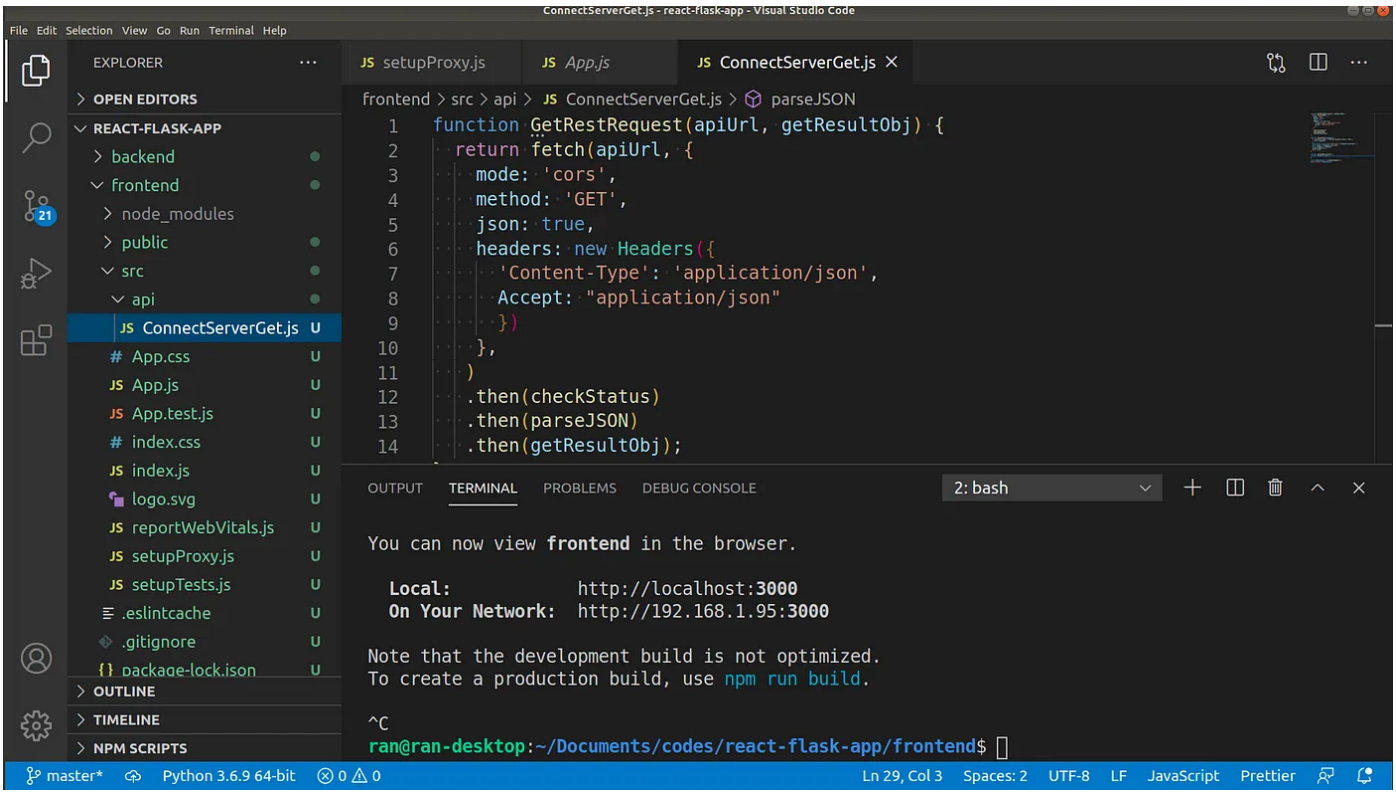


Install http-proxy-middleware



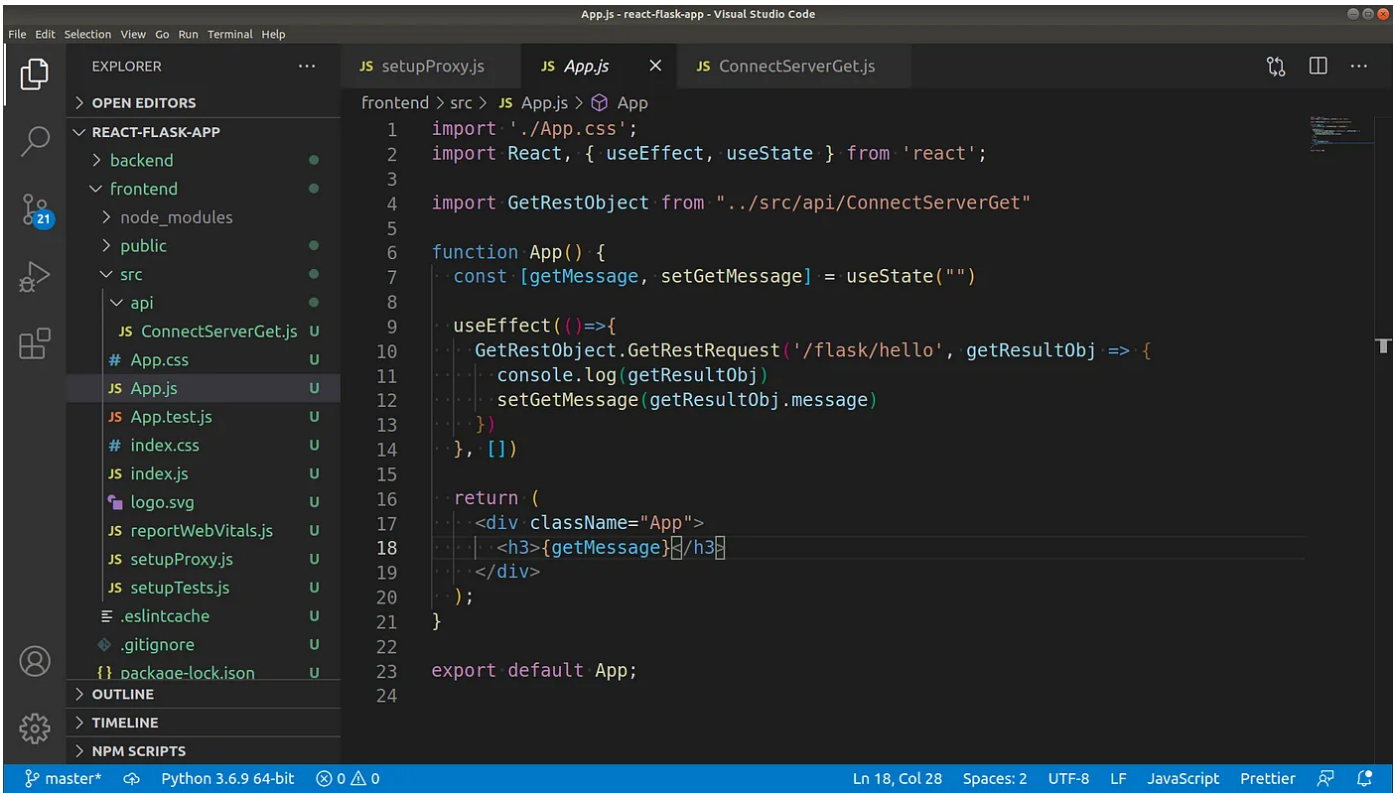
setupProxy.js

Create the setupProxy.js file in the src directory. What this piece of code above means is that all API calls from the front-end will be made to http://localhost:5000/flask/{something}.

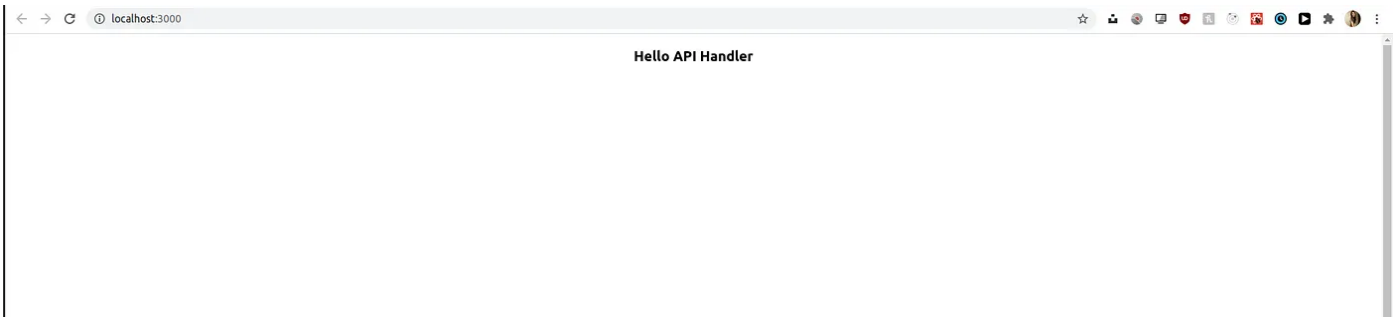


Get Function — click this to get the codes

This is one way of writing the GET request. Alternatively, you can also just write a simple axios GET request. If you're building a larger application, place this inside your Redux actions.

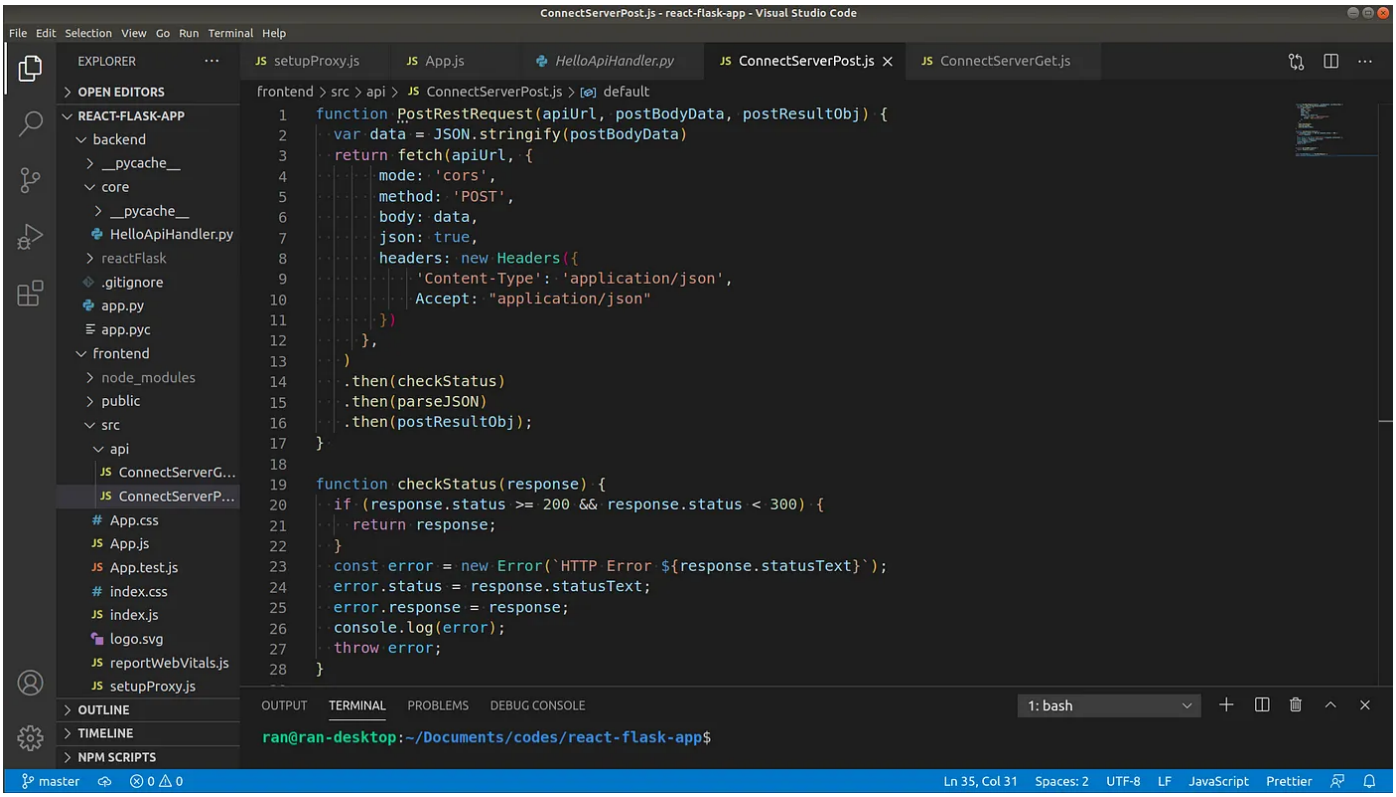


Call the GET request function in a useEffect (with square bracket) so it gets called once onLoad

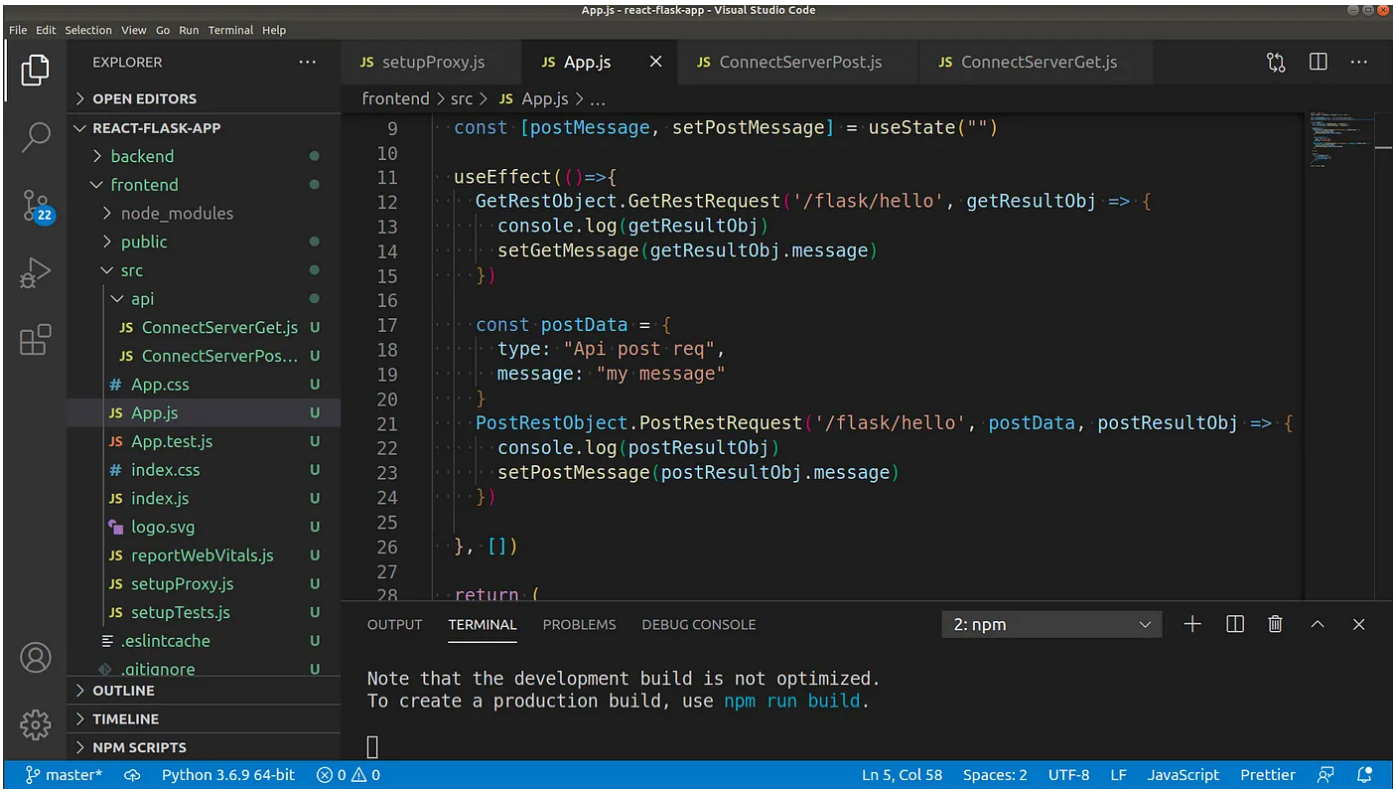


It works! ❤

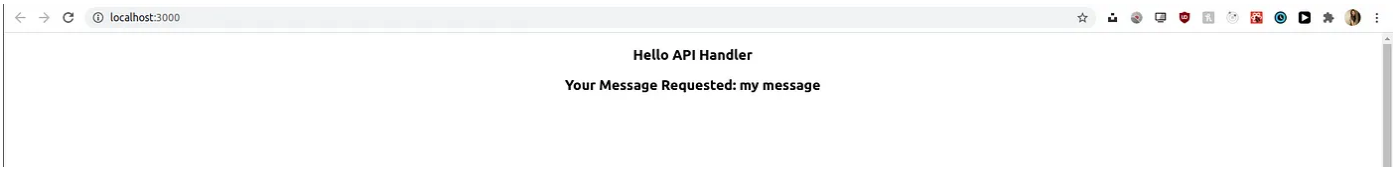
As for the POST request function,



Full codes please see — link



We'll test with these two hard-coded messages



It returns it correctly! ❤

Note that it's returning the same message we sent it because I assigned it that way (ret_msg = args['message']) in the back-end. I'm doing this just for demo purposes and if you're writing an actual application, remember to process the requests first before returning it :)

That's all for today's article! Thank you for reading this and please feel free to reach out by commenting or messaging me if you run into any issues/have any questions or comments! Referenced from — link.

Happy ReFlasking! ʕ•ᴥ•ʔっ♡

React | Python | Programming | Software Development | Flask

Follow

Written by Ran (Reine)

433 Followers · Writer for The Startup

I live for days when I can watch skies of blue, while enjoying the view. Most other days I'm a city rat who scuttles between Art and Coding. SG NTU CS Grad.

More from Ran (Reine) and The Startup

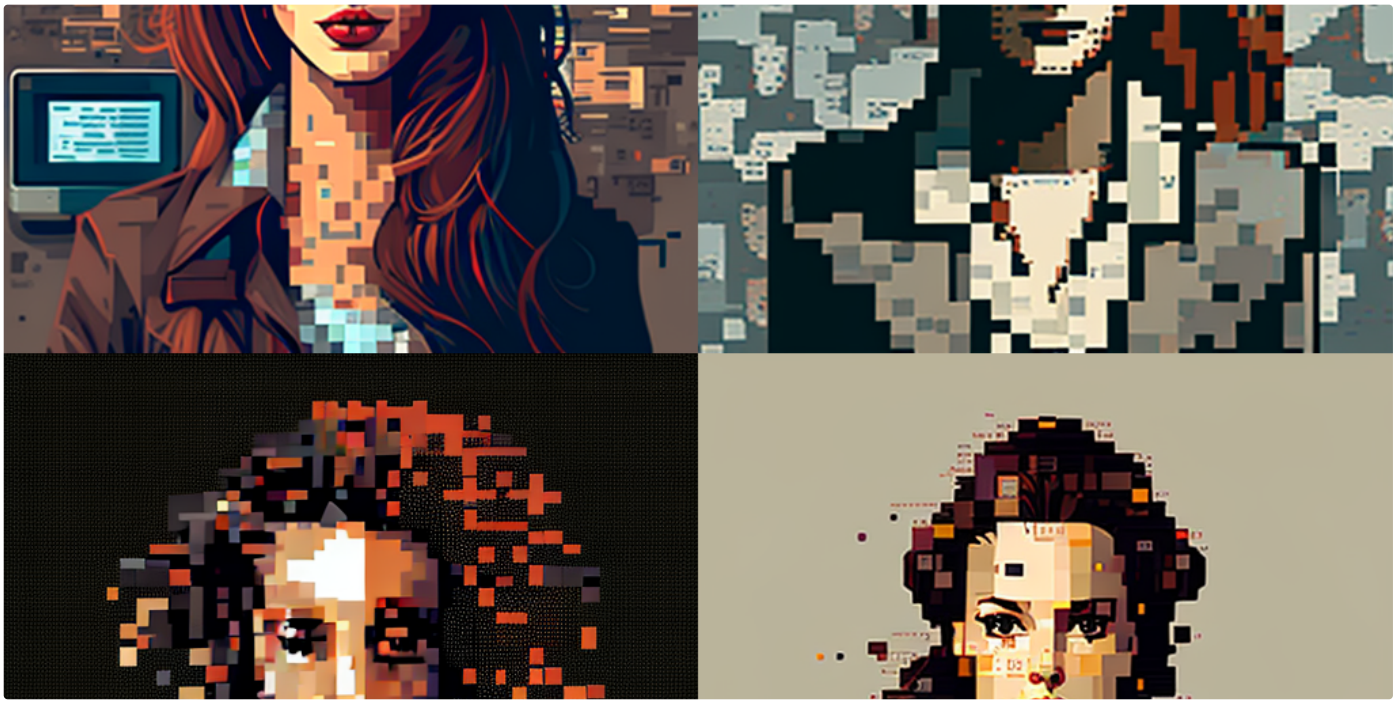
Ran (Reine) in Towards Data Science

Build & Deploy a React + Flask App

Probably the most simple and concise guide

· 5 min read · Jan 15, 2021

405 14



Zulie Rane in The Startup

If You Want to Be a Creator, Delete All (But Two) Social Media Platforms

In October 2022, during the whole Elon Musk debacle, I finally deleted Twitter from my phone. Around the same time, I also logged out of...

✨ · 8 min read · Apr 18

25K 502

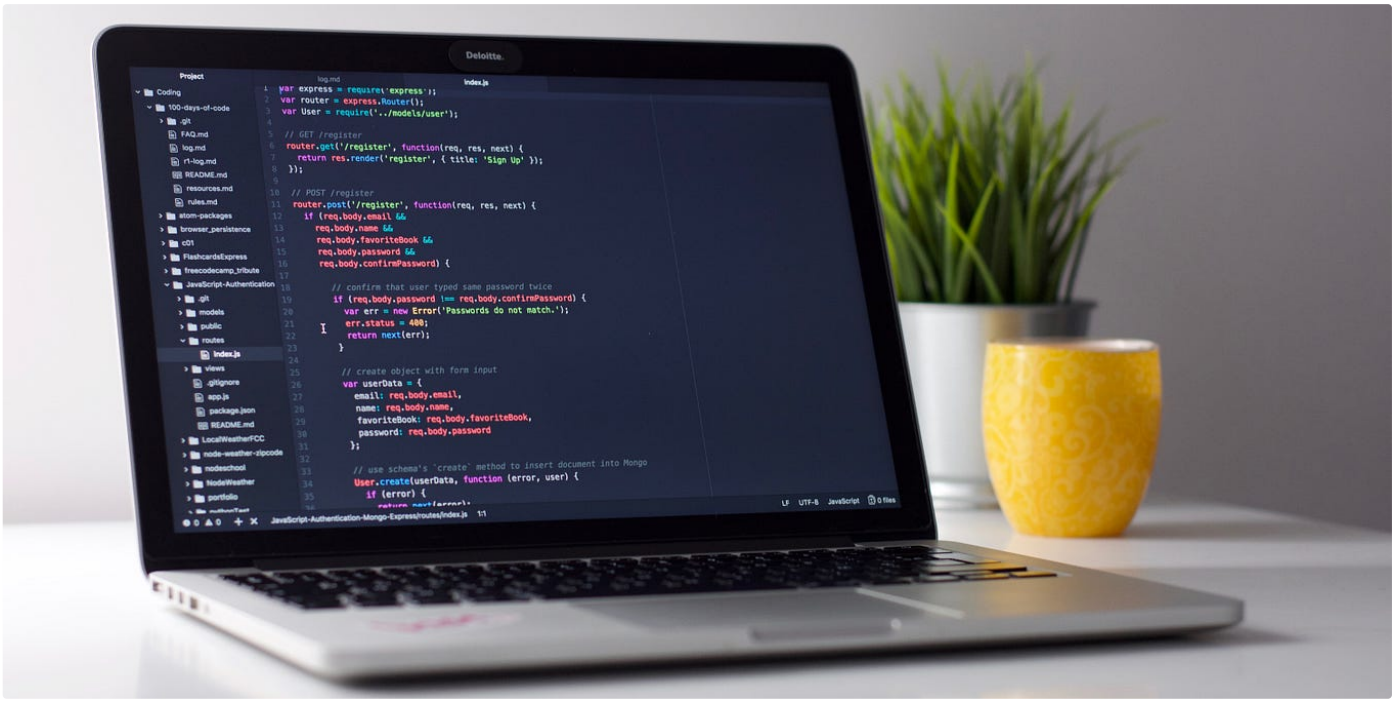
Nitin Sharma in The Startup

Goodbye ChatGPT: These (New) AI Tools Will Leave You Speechless

I bet that 99% of the readers are not familiar with any of these tools.

· 7 min read · May 6

2.6K 41



Ran (Reine) in Analytics Vidhya

How to install OpenCV.js

(With detailed instructions for compiling from source)

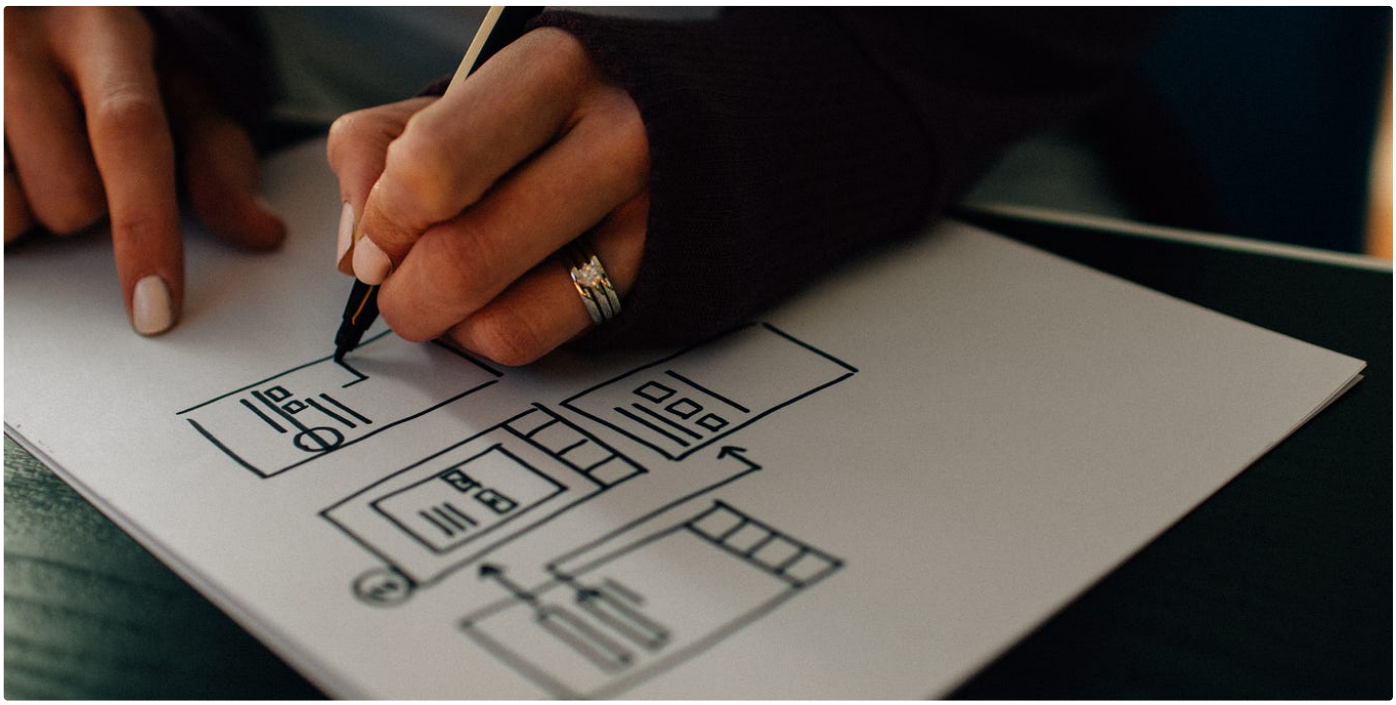
· 3 min read · Mar 15, 2020

53 4

See all from Ran (Reine)

See all from The Startup

Recommended from Medium



Audhi Aprilliant in Geek Culture

Part 2 — End to End Machine Learning Model Deployment Using Flask

How to build a user interface for loan approval prediction app and set up the backend using flask

· 10 min read · Nov 30, 2022

178 1

Yujian Tang in Plain Simple Software

Create an API with User Management using FastAPI and Firebase

A quick start guide to creating an API with user authentication

· 12 min read · Dec 13, 2022

Kumar Shubham in Towards Data Science

Build a Blog Website using Django Rest Framework — Users App (Part 2)

In the second part, we will deal with building user-related models and views and will test the user-related APIs.

· 15 min read · Dec 14, 2022

211

Lynn Kwong in Towards Data Science

Build a WebSocket Application with FastAPI and Angular

Learn to build a two-way interactive communication application with the WebSocket protocol

· 8 min read · Jan 30

77

Reed Barger in Web Dev Hero

7 React Projects for Beginners in 2023 (+ Code)

You're ready to start making simple projects with React, but you don't know what to make. Where should you start?

· 6 min read · Jan 11

269 5

Viktor Nagornyy in Powered by Django

How To Create An API with Django REST Framework Quickly

Learn how to integrate Django Rest Framework API into your Django project with this beginner-friendly tutorial, including two example APIs.

✨ · 6 min read · Apr 4

31 1