Full Stack Foundations

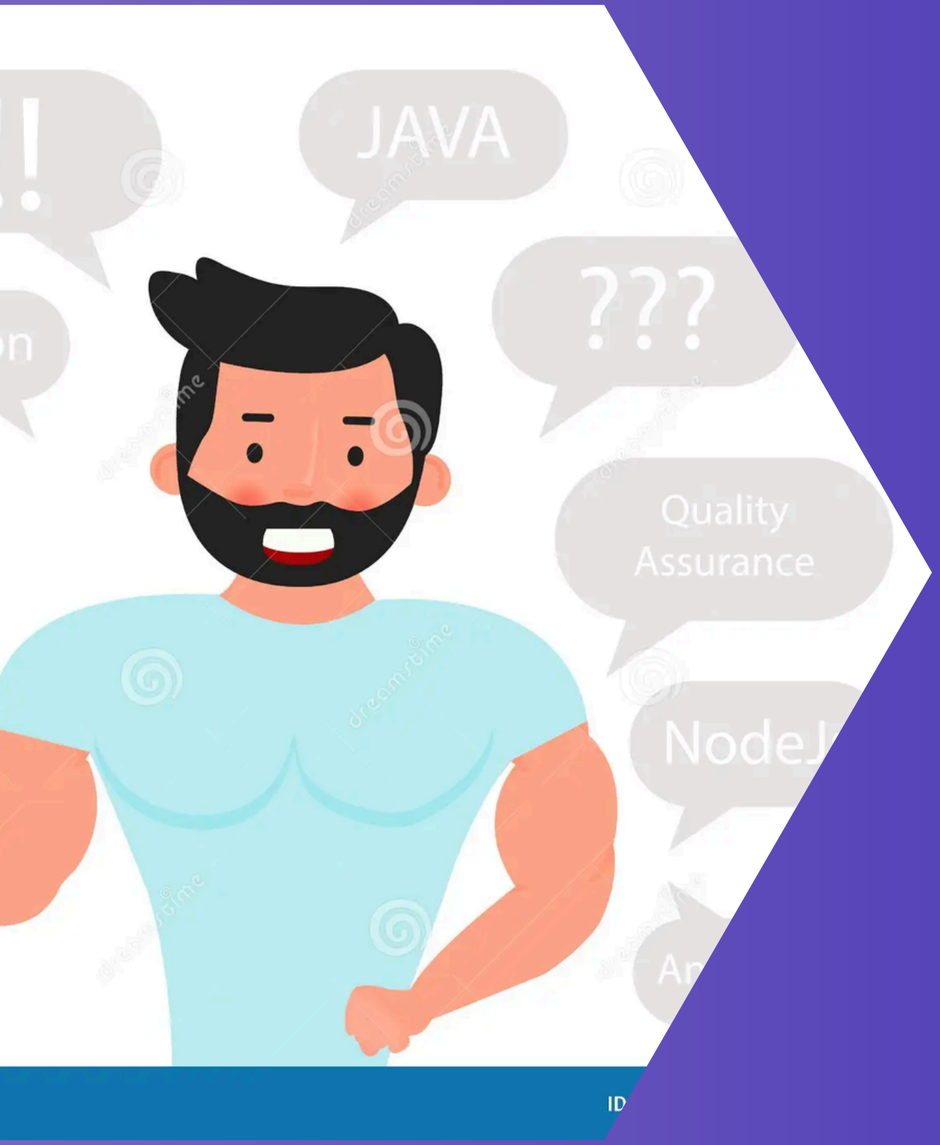
Frontend using React and Tailwind, APIs using Flask!

PRESENTED BY
Jake Esperson

04/29/2025

Agenda

1. THINKING LIKE A DEVELOPER
2. CHOOSING A STACK
3. FRONTEND DEMO
4. API DEMO



Thinking like a Developer

uhh, what?

But I’m already a developer, of course I think like one!

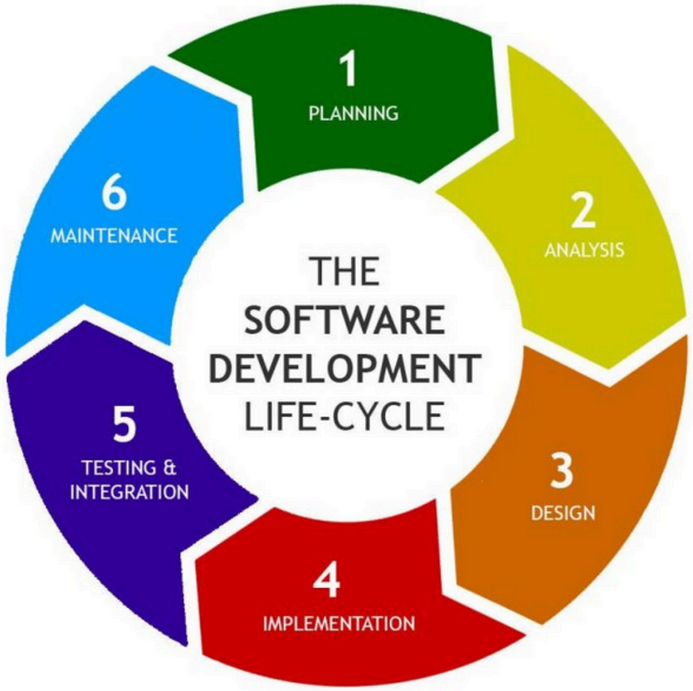
In this case, I’m talking about needing to plan out your app

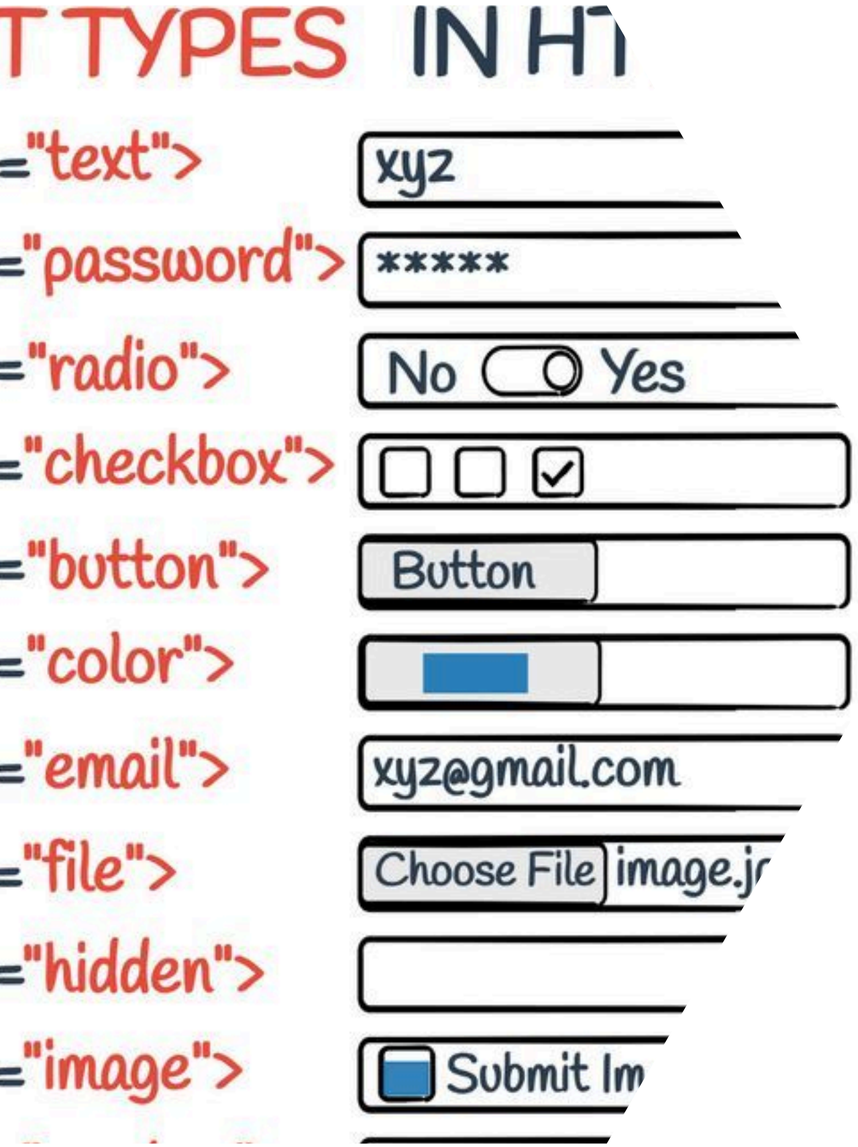
Full stack development is roughly 50/50 between planning and coding

The more you plan, the less time you spend fixing bugs

What goes into planning?

- **Inputs and Outputs**
- **Tech Stack**
- UI/UX
- Who the users are
- Role based access control
- Analytics
- Testing
- DevOps
- and more...





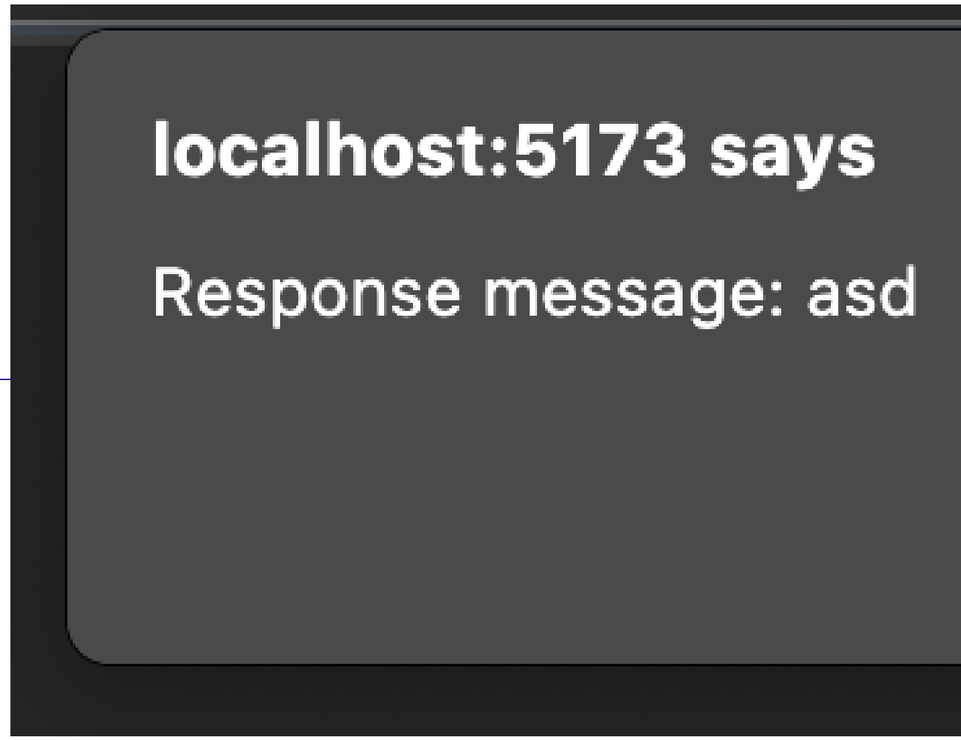
Inputs

What the user provides and what the app provides to the backend for some task

Ex: forms, images, user data

Outputs

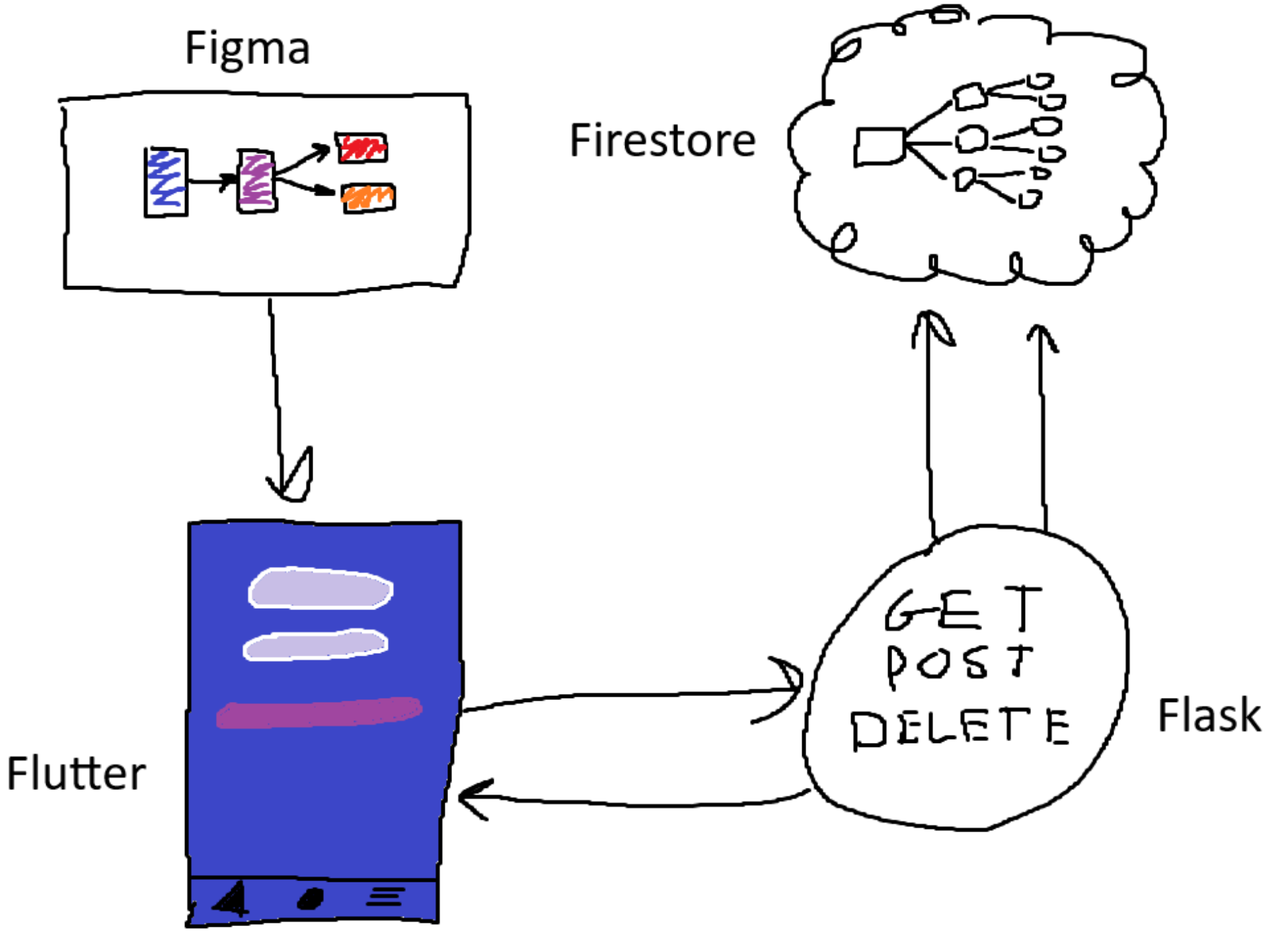
What the user is getting back after doing something with the data
Ex: confirm messages, database items

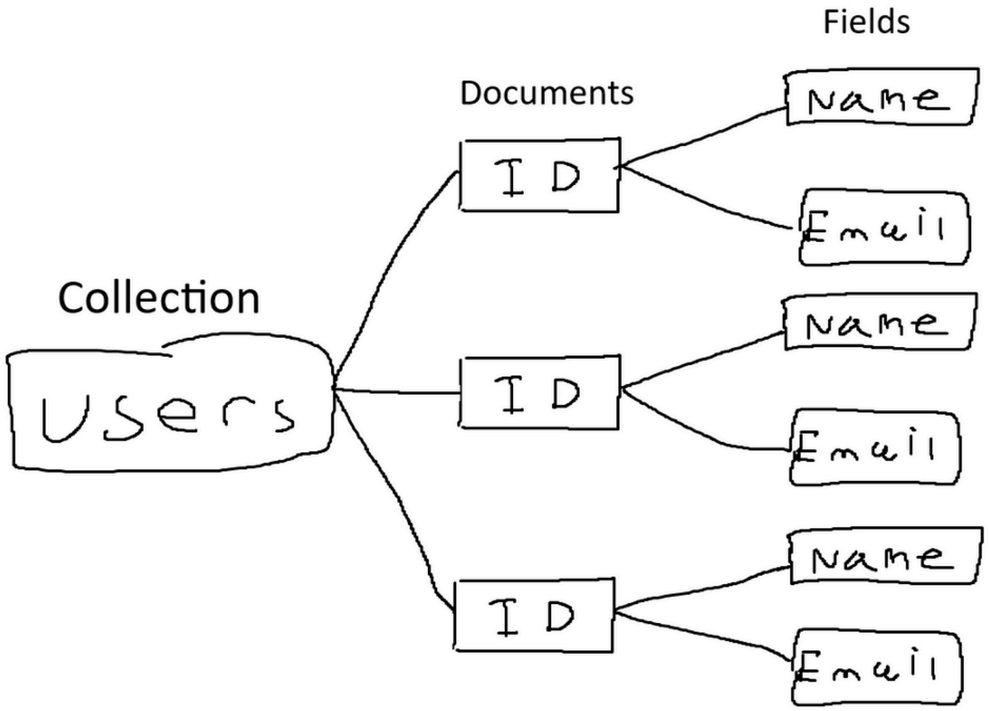


Formatting

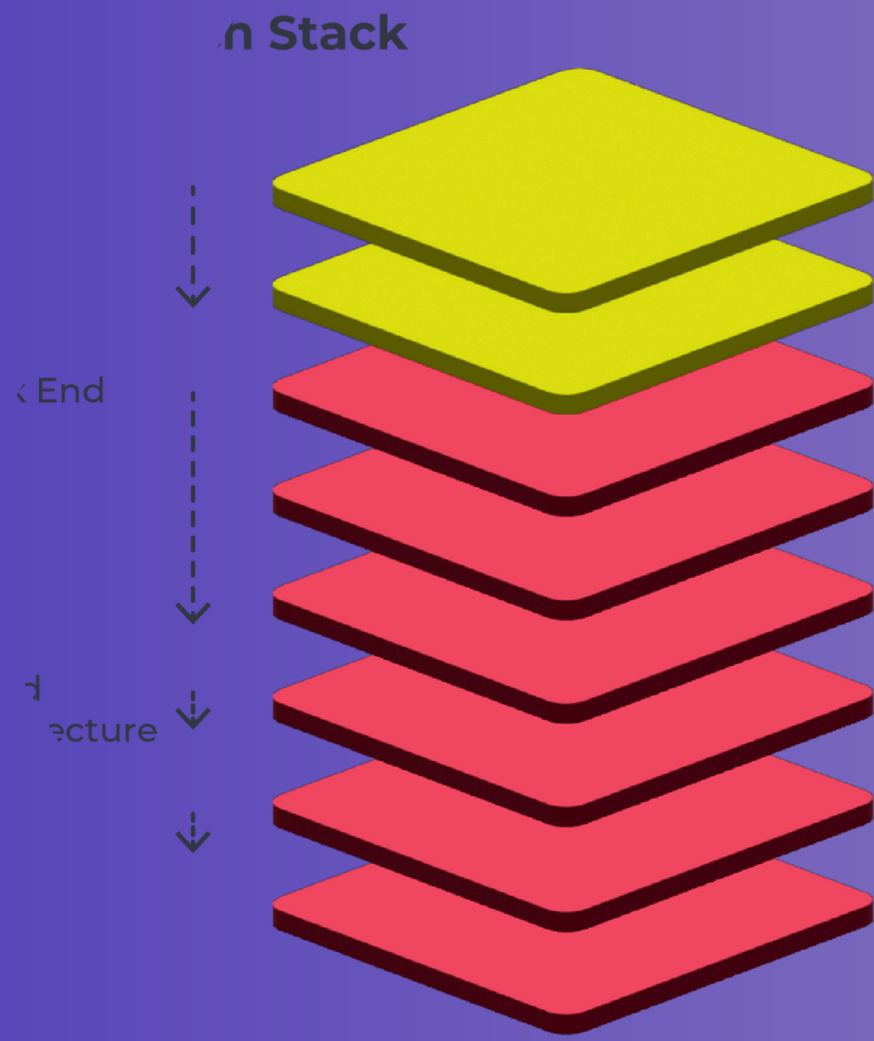
- JSON = JavaScript Object Notation
  - Series of named values
  - Values can have subvalues
- Universal standard for data
- How data is transferred on the internet, from backend to frontend

Use diagrams!





Tech Stacks



Don’t overcomplicate it

Our stack for this workshop

FRONTEND: REACT

Widely used, great documentation, and always relevant

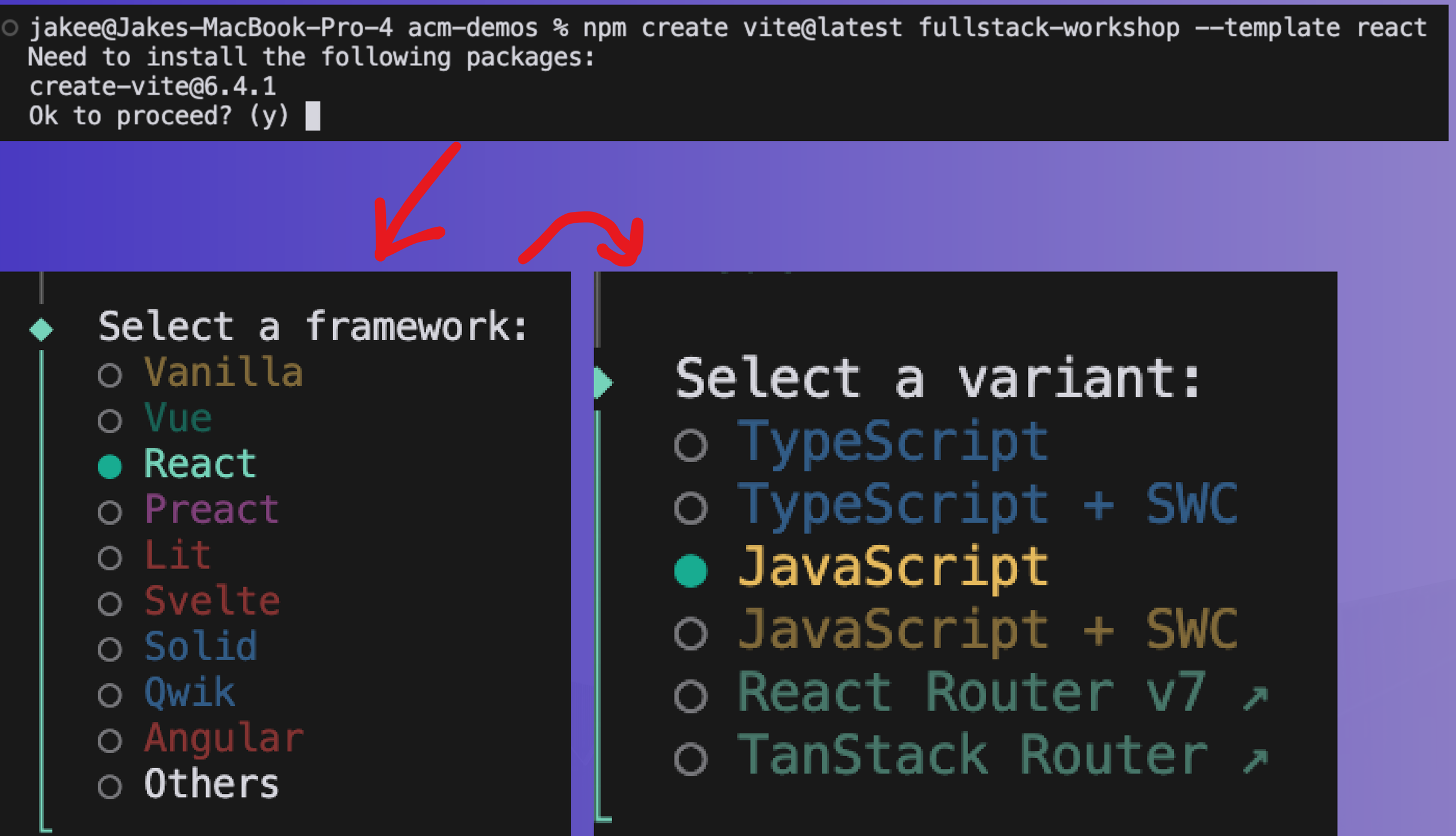
STYLING: TAILWIND

Great with React, simplifies CSS classes, and centralizes everything

PROXY API: FLASK

Python for ease of use, great libraries, and works well with all technoloigies (AI!)

Demo Time!
You'll need Nodejs and Python for this


jakee@Jakes-MacBook-Pro-4 acm-demos % npm create vite@latest fullstack-workshop --template react
Need to install the following packages:
create-vite@6.4.1
Ok to proceed? (y)
Select a framework:
Vanilla
Vue
React
Preact
Lit
Svelte
Solid
Qwik
Angular
Others
Select a variant:
TypeScript
TypeScript + SWC
JavaScript
JavaScript + SWC
React Router v7 ↗
TanStack Router ↗

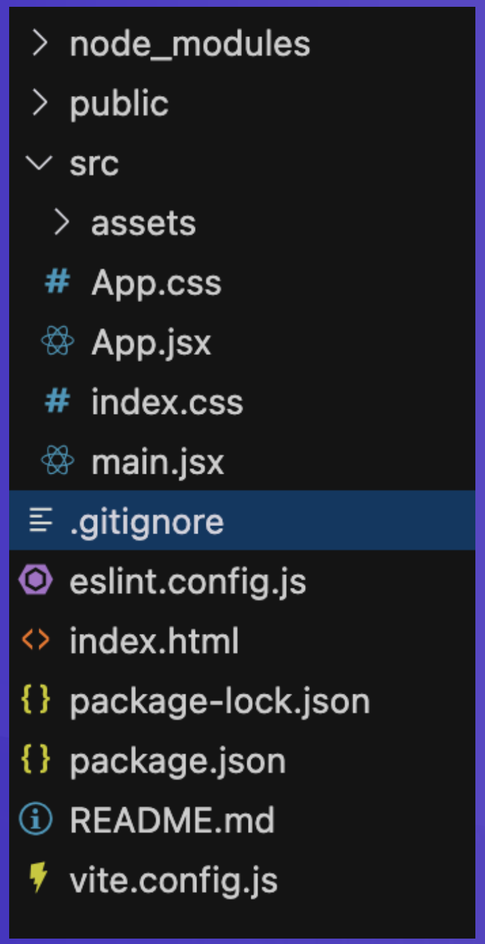
```
jakee@Jakes-MacBook-Pro-4 fullstack-workshop % npm i

added 152 packages, and audited 153 packages in 32s

32 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

```
jakee@Jakes-MacBook-Pro-4 fullstack-workshop % npm i react-router-dom
```

1. create app w/ Vite

```
jakee@Jakes-MacBook-Pro-4 fullstack-workshop % npm run dev

> fullstack-workshop@0.0.0 dev
> vite


  VITE v6.3.3  ready in 226 ms

  ➜  Local:   http://localhost:5173/
  ➜  Network: use --host to expose
  ➜  press h + enter to show help
^C
```

File Structure



Setup for any React page

```jsx
import React from "react"; // Importing React

function Page() {
  function func() {
    // do something
  }

  return (
    <div>
      {/* some html */}
    </div>
  );
}

export default Page; // Exporting the Page component
```

1. create app w/ Vite
2. npm i, react-router-dom, npm run dev

Replace App.jsx with the following:

```
import React from "react";
import { BrowserRouter, Routes, Route } from "react-router-dom";
import Home from "./Home";
// import About from "./About";

function App() {
  return (
    <div>
      <BrowserRouter>
        <Routes>
          <Route path="/" element={<Home />}></Route>
          {/* <Route path="/about" element={<About />}></Route> */}
        </Routes>
      </BrowserRouter>
    </div>
  );
}

export default App;
```

1. create app w/ Vite
2. npm i, react-router-dom, npm run dev

Create a new file Home.jsx

```
import React, { useState } from "react";

function Home() {
  const [counter, setCounter] = useState(0);

  function incrementCounter() {
    setCounter(counter + 1);
  }

  return (
    <div>
      <h1>Home</h1>
      <p>Counter = {counter}</p>
      <button onClick={incrementCounter}>Increment</button>
      <button onClick={() => setCounter(counter - 1)}>Decrement</button>
    </div>
  );
}

export default Home;
```

1. create app w/ Vite
2. npm i, react-router-dom, npm run dev
3. modify App.jsx

Delete App.css (the whole file, not just what's in it)

Delete everything in index.css
except the following

```css
:root {
  font-family: system-ui, Avenir, Helvetica, Arial, sans-serif;
  line-height: 1.5;
  font-weight: 400;

  color-scheme: light dark;
  color: rgba(255, 255, 255, 0.87);
  background-color: #242424;

  font-synthesis: none;
  text-rendering: optimizeLegibility;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
}
```

1. create app w/ Vite
2. npm i, react-router-dom, npm run dev
3. modify App.jsx
4. create Home.jsx

Put the command in the terminal

```
jakee@Jakes-MacBook-Pro-4 fullstack-workshop % npm install tailwindcss @tailwindcss/vite

added 12 packages, and audited 171 packages in 4s

34 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

1. create app w/ Vite
2. npm i, react-router-dom, npm run dev
3. modify App.jsx
4. create Home.jsx
5. delete App.css, edit index.css

edit vite.config.js

```
import { defineConfig } from 'vite'
import react from '@vitejs/plugin-react'
import tailwindcss from '@tailwindcss/vite'

// https://vite.dev/config/
export default defineConfig({
  plugins: [react(), tailwindcss()],
})
```

at the top of index.css

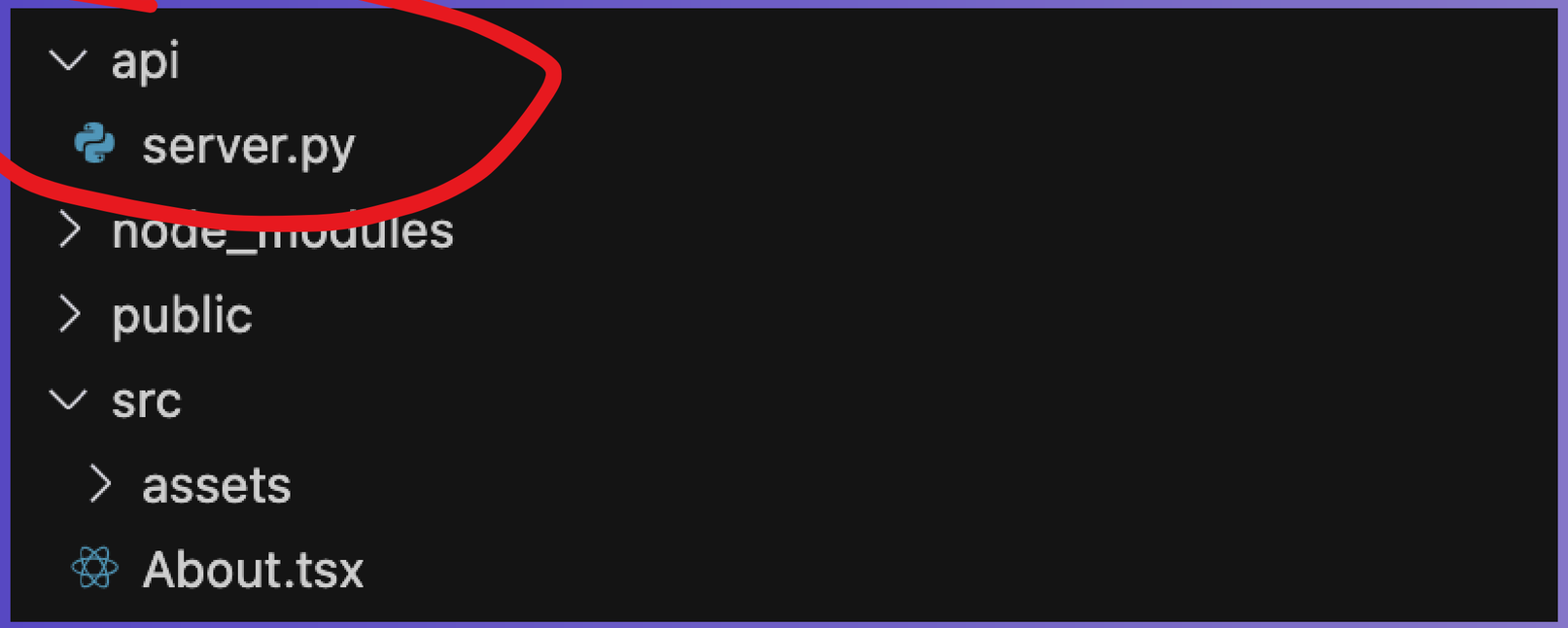
```
@import "tailwindcss";

:root {
```

change the HTML inside the return statement of Home.jsx

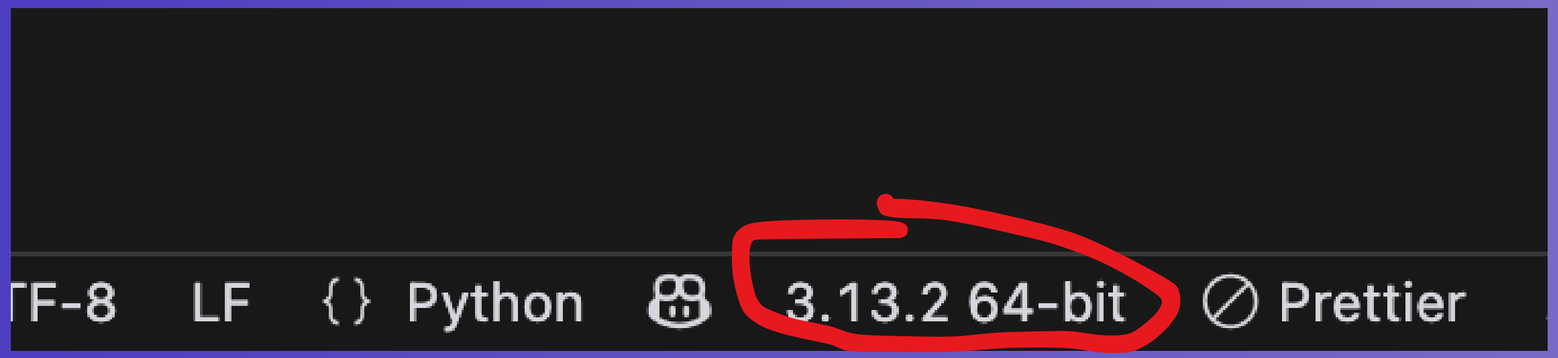
```
return (
  <div className="flex flex-col items-center justify-center min-h-screen">
    <h1 className="text-5xl">Home</h1>
    <p>Counter = {counter}</p>
    <button
      className="bg-blue-500 hover:bg-blue-700 text-white font-bold py-2 px-4 rounded"
      onClick={incrementCounter}
    >
      Increment
    </button>
    <button
      className="bg-red-500 hover:bg-red-700 text-white font-bold py-2 px-4 rounded ml-2"
      onClick={() => setCounter(counter - 1)}
    >
      Decrement
    </button>
  </div>
```

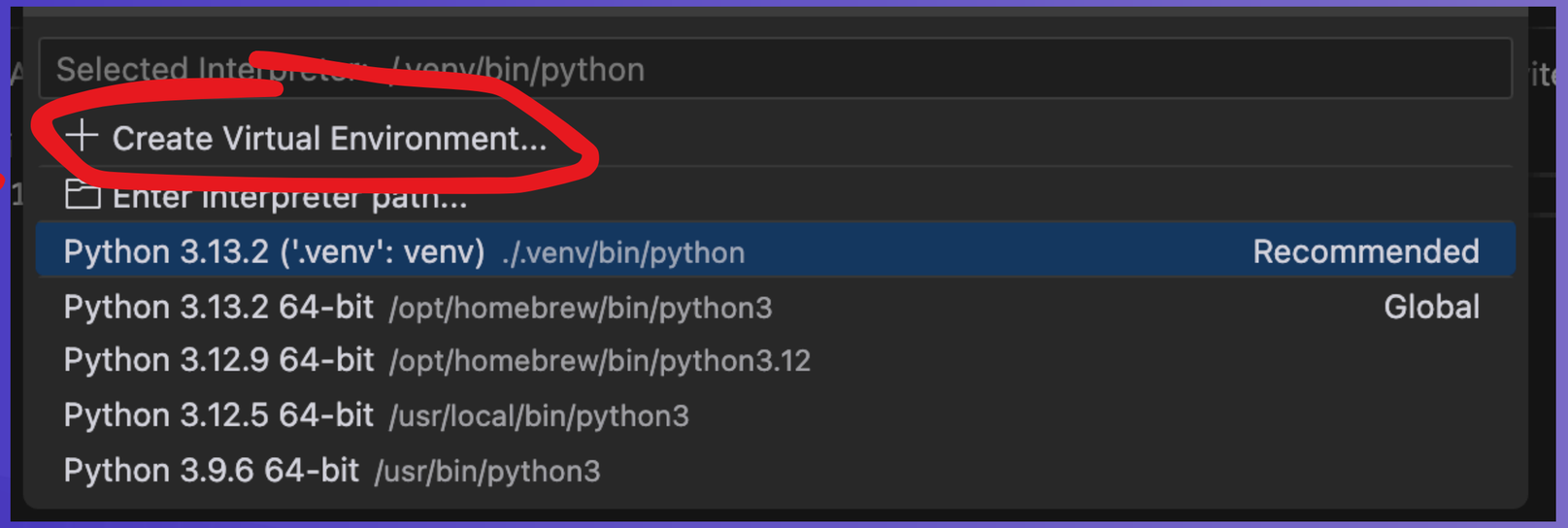
1. create app w/ Vite
2. npm i, react-router-dom, npm run dev
3. replace App.jsx
4. write Home.jsx
5. delete App.css, change index.css
6. configure Tailwind

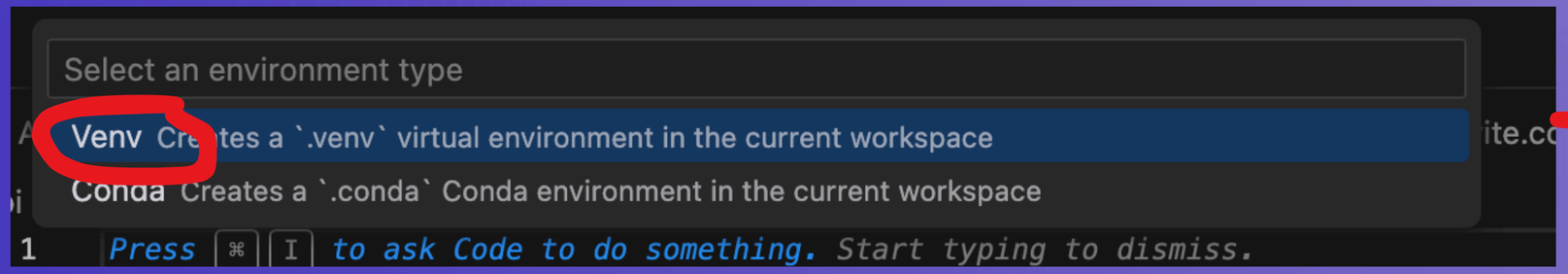
Create the api folder
and make server.py

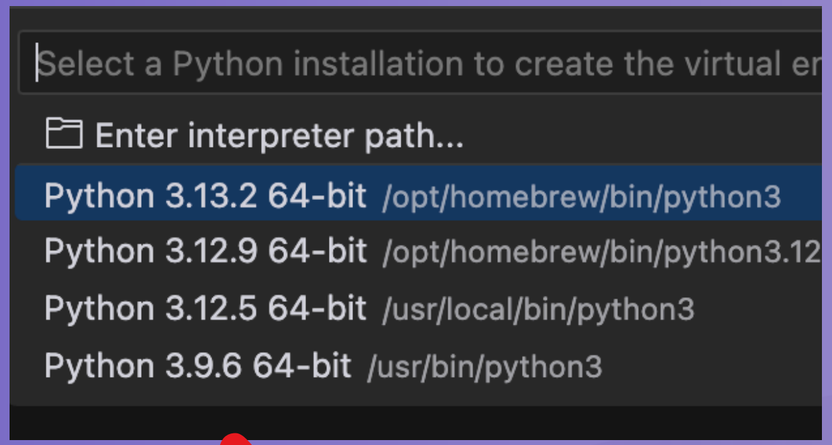


Make sure the python file is open in VSCode







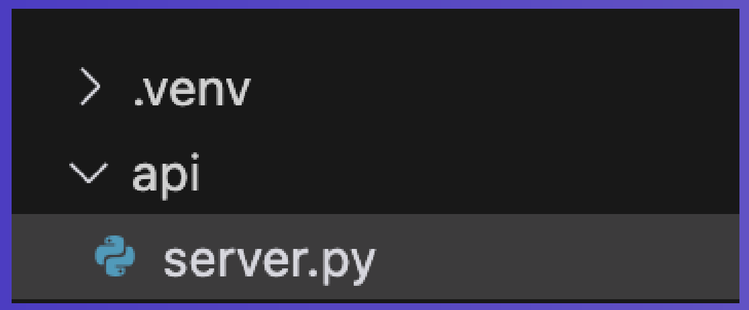


Do “cd api” first

```
jakee@Jakes-MacBook-Pro-4 api % source ../.venv/bin/activate
(.venv) jakee@Jakes-MacBook-Pro-4 api %
```

```
(.venv) jakee@Jakes-MacBook-Pro-4 api % pip install flask flask-cors
Collecting flask
```

1. set up Python Virtual Environment



server.py

```
from flask import Flask, request
from flask_cors import CORS

app = Flask(__name__)
CORS(app)

# Functions here

if __name__ == "__main__":
    app.run(debug=True, host='0.0.0.0', port=3000)
```

1. set up Python Virtual Environment
2. install flask and flask-cors

Replace
 # Functions here
with the following

```
@app.route('/api', methods=['GET'])
def api():
    return {"message": "Hello, World!"}

@app.route('/api/message', methods=['GET'])
def message():
    message = request.args.get('message')
    if message:
        return {"message": f"{message}"}
    else:
        return {"error": "No message provided"}, 400
```

1. set up Python Virtual Environment
2. install flask and flask-cors
3. boilerplate Flask app

Run in the terminal

```
(.venv) jakee@Jakes-MacBook-Pro-4 api % py server.py
 * Serving Flask app 'server'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:3000
 * Running on http://192.168.1.117:3000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 118-670-377
```

Web Browser

localhost:3000/api

```
{
  "message": "Hello, World!"
}
```

1. set up Python Virtual Environment
2. install flask and flask-cors
3. boilerplate Flask app
4. create API endpoints

Replace with your name! (%20 is a space)

localhost:3000/api/message?message=Hi%20Jake

```
{
  "message": "Hi Jake"
}
```

Replace Home.jsx OR Create a new page and change the function name and export to a different name (don’t forget to add it to the router in App.jsx!)

```
import React, { useState } from "react";

function Home() {
  const [message, setMessage] = useState("");

  function sendMessage () {
    const apiURL = "http://localhost:3000/api/message?message=" + message;

    fetch(apiURL, {
      method: "GET",
      headers: {
        "Content-Type": "application/json",
      },
    })
      .then((response) => {
        if (!response.ok) {
          throw new Error("Network response was not ok");
        }
        return response.json();
      })
      .then((data) => {
        alert("Response: " + JSON.stringify(data));
        alert("Response message: " + data.message);
      })
      .catch((error) => {
        alert("Error: " + error.message);
      });
  };
```

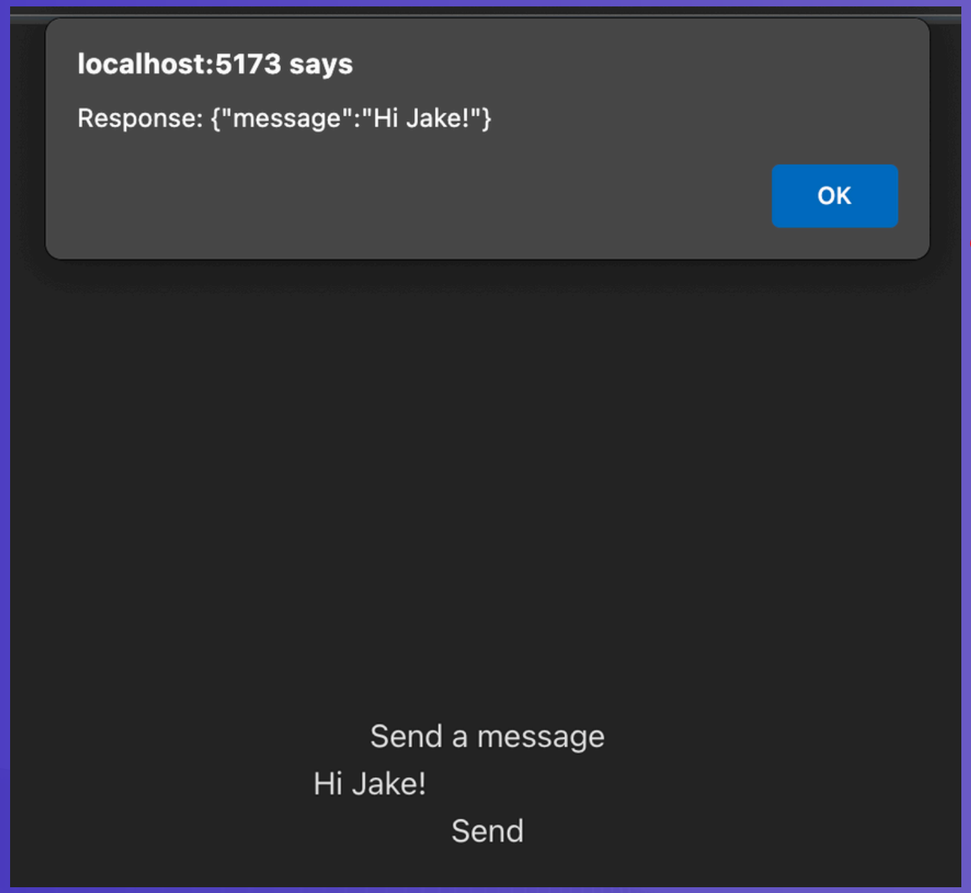
This is part 1 of the file, part 2 on the next slide

Replace Home.jsx OR Create a new page and change the function name and export to a different name (don’t forget to add it to the router in App.jsx!)

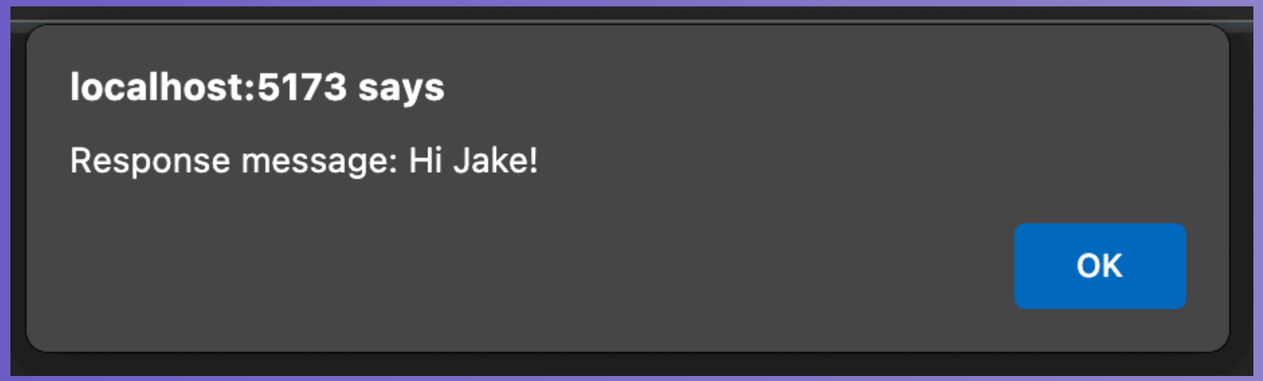
```
return (
   <div className="flex flex-col items-center justify-center min-h-screen">
    <h1>Send a message</h1>
    <input
     type="text"
     value={message}
     onChange={(e) => setMessage(e.target.value)}
     placeholder="Type your message here"
    />
    <button onClick={sendMessage}>Send</button>
   </div>
  );
}

export default Home;
```

part 2, just put it right after the first part

Back to the browser! This is the raw JSON output



And the formatted (or parsed) JSON output



Next Steps

Connecting to External Services

- Your proxy API calls other APIs (CORS)
- This may be relevant for Intel Hack 👀

Databases

- Storing your data is crucial
- Short answer: use Firebase

Announcements

ACM BEACH DAY!
NEXT SATURDAY
5/10 11AM-4PM
SANTA CRUZ

Thank you!

Make sure to scan for attendance!

Intel Hack Website: https://intel.scuacm.com