**Getting Started with Heroku using Node.js and React**

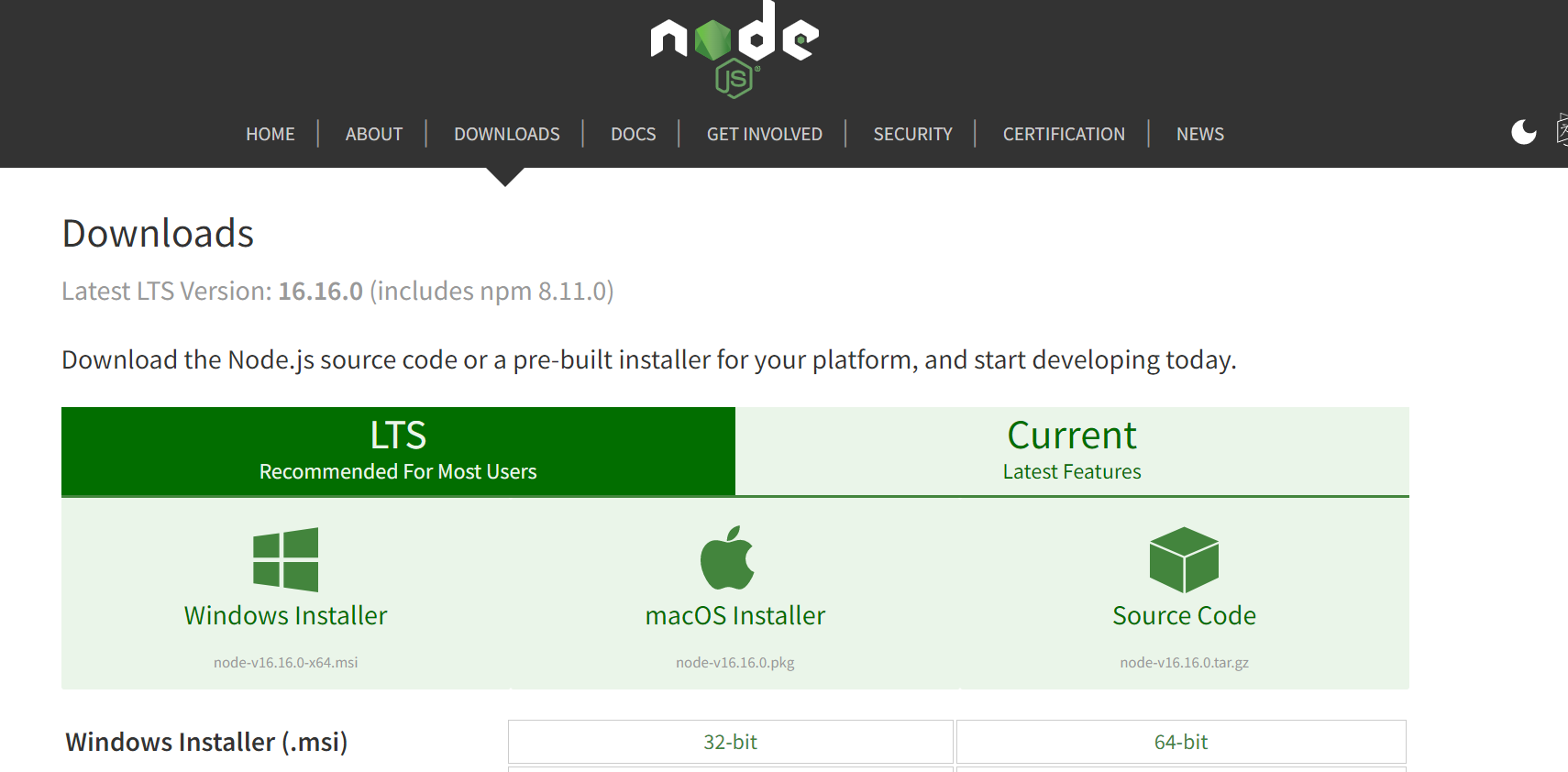
**Bradley Dinger & Benjamin Rittenhouse**

Code: <https://github.com/Braddinger13/CIS4282_ReactNav>

This document will go through the steps of creating a basic React Navigation project, how to run your project locally, and how to publish your app using Heroku and Github.

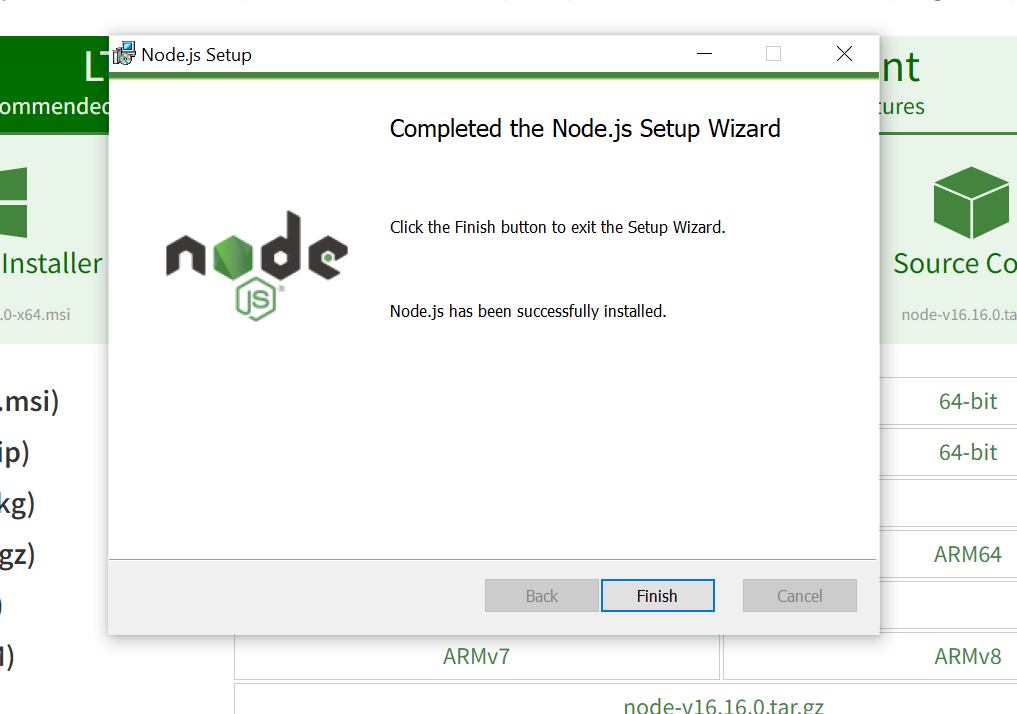
1. **Download Node.js**

To start developing a Node.js backend, we first need to install Node.js.



The Node.js official download page located at <https://nodejs.org/en/download/>.

Just click whichever green box under LTS that applies to your Operating System.

Once you are done with the installation wizard (all the presets should be fine), you should see the box below. 

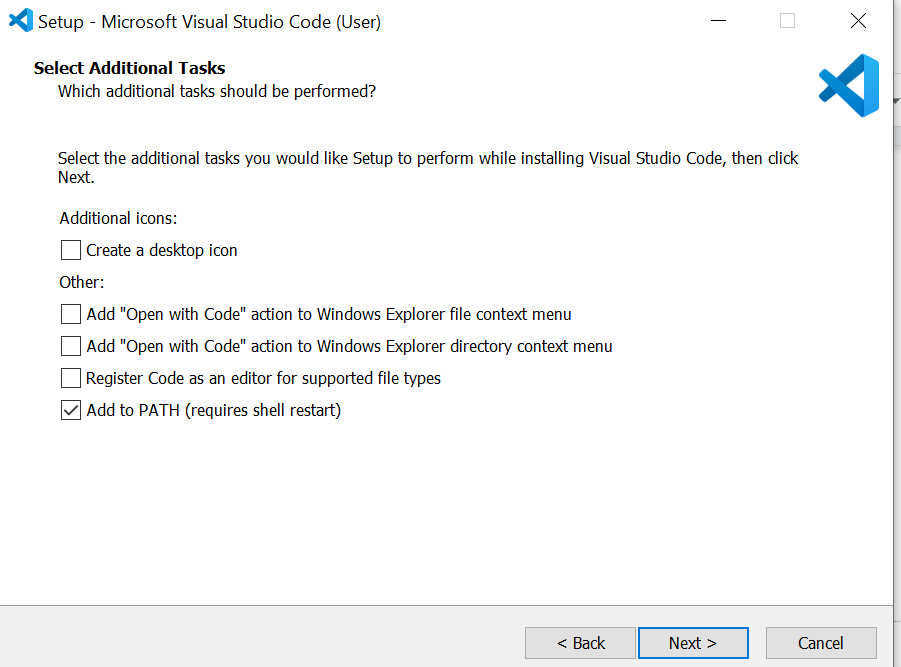
Just hit finish and you should now have **Node.js** and the **Node Package Manager** **(npm)** installed. NPM will be used to help set up our development environment.

1. **Create Project Folders/Files**

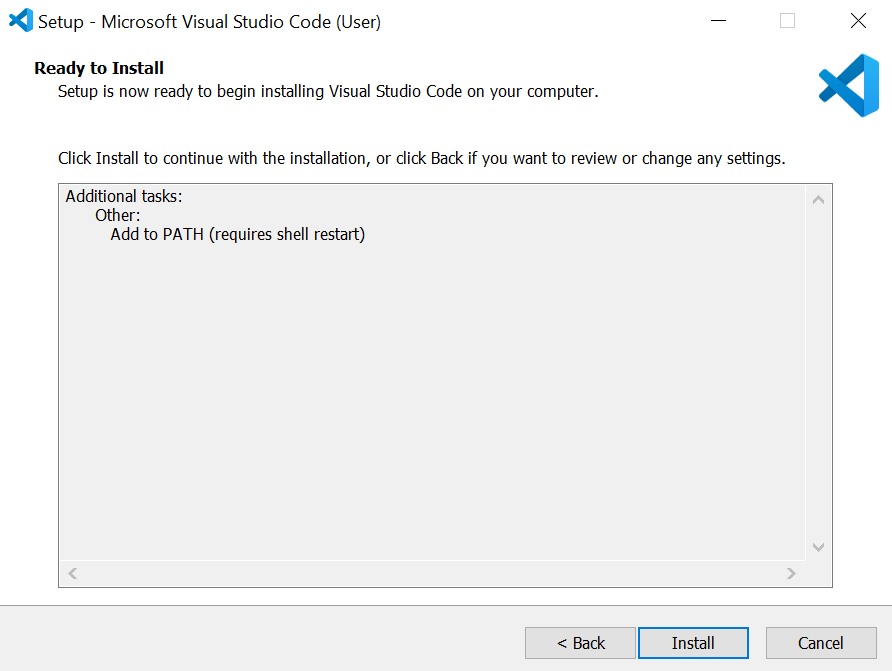
I will be using Visual Studio Code as my IDE.

The official download page for VS Code is located here: <https://code.visualstudio.com/download>

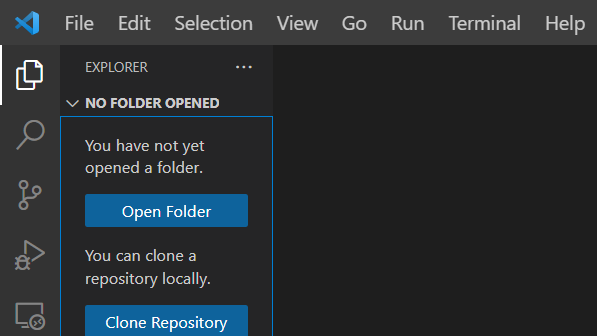
Follow the setup, most of the presets should be fine. Check off the **Create a Desktop Icon** box if you wish to have the desktop icon.



Next, click install and wait for the installation to finish.



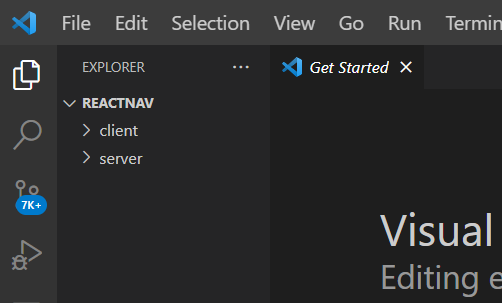
Now, open up VS Code. To start a new project, you can open up the explorer tab on the left-side of the screen and select **Open Folder** as shown below**.** If you have not yet made a folder for your project, you can do this in your file explorer, select it, then click **Select Folder** to open it in VSCode**.**



**File Structure-**

Once a folder is selected, you will see it appear in the explorer along with its contents, as of right now you will probably see nothing on the folder.

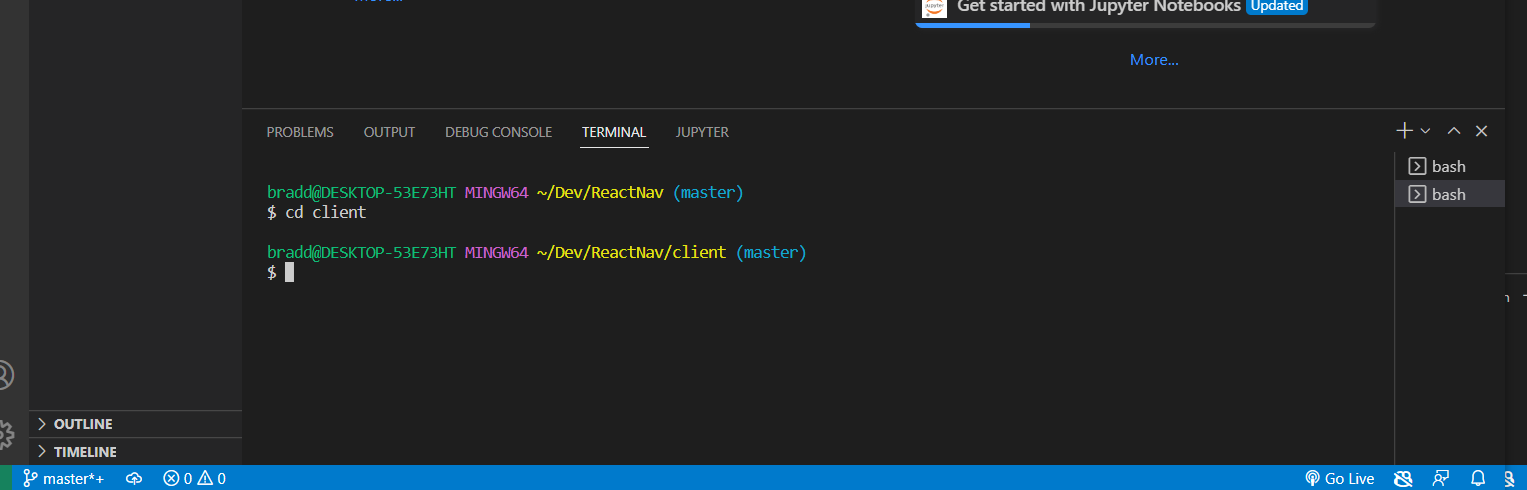
To get started with file structure, I like to make two folders, one for your client(front-end) code and one for your server(back-end) code. You can create these folders by pressing the new folder button in your top-level folder. In my case, the top level folder is called ReactNav. It will look like this.

****

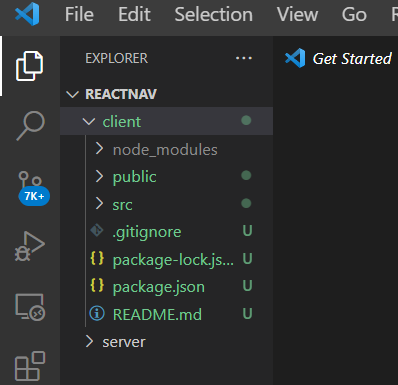
1. **Starting a React Application**

You can easily start a React application using your built-in command line in VSCode. At the top of your screen, you will see a **Terminal** tab. In that tab, click the new terminal button which will open a terminal window at the bottom of your screen.

This should open to your top-level folder directory. Since we are focusing on React right now, you should switch your directory to your client folder by typing “**cd client**” into your terminal.



Now we can set up a basic react app by typing “**npx create-react-app .**” with the period included to use the current directory. It may take some time for everything to be set up. The explorer will look like this afterwards:

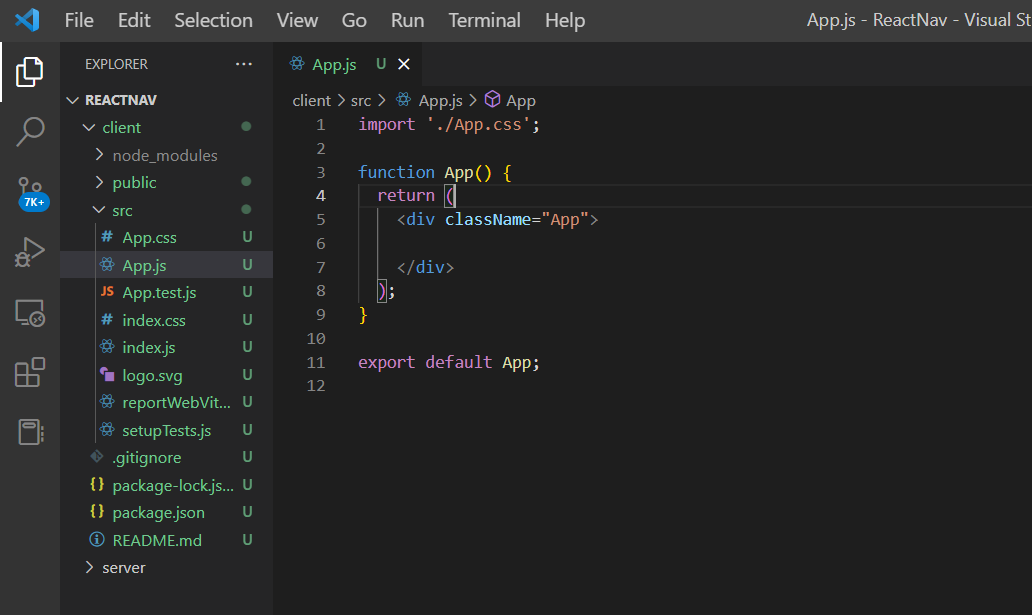


In your **src** folder, there is a file called App.js. This file, by default, is the file that you will run. Your other react components and styling are added on top of this.

To test that everything is working with react, type “**npm start**” into your terminal. This should run the default react page in your browser. Looks like this:



Now delete everything inside the parent div in App.js since we will not be using it in our code.

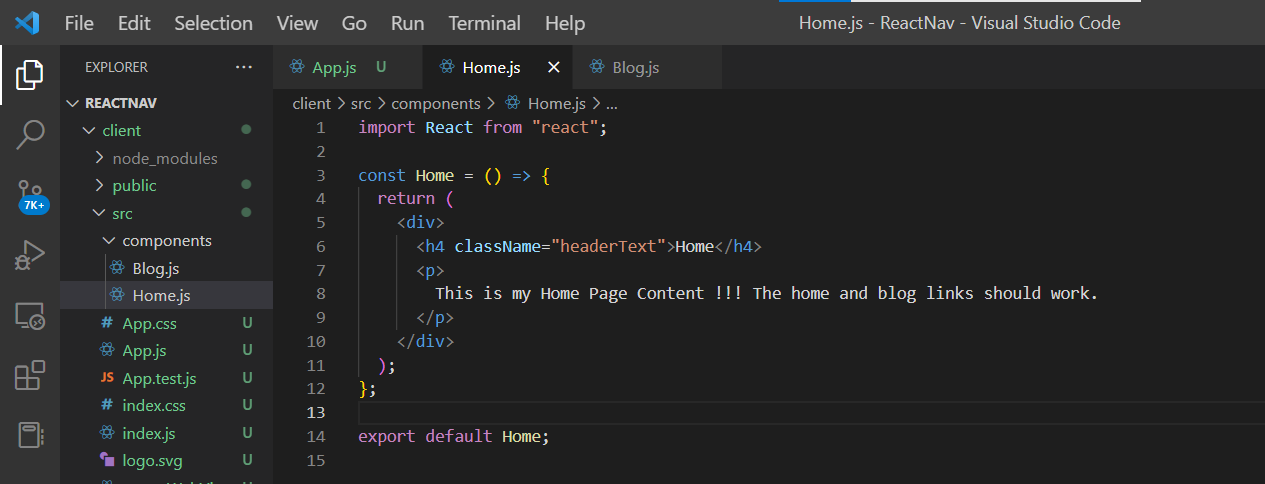


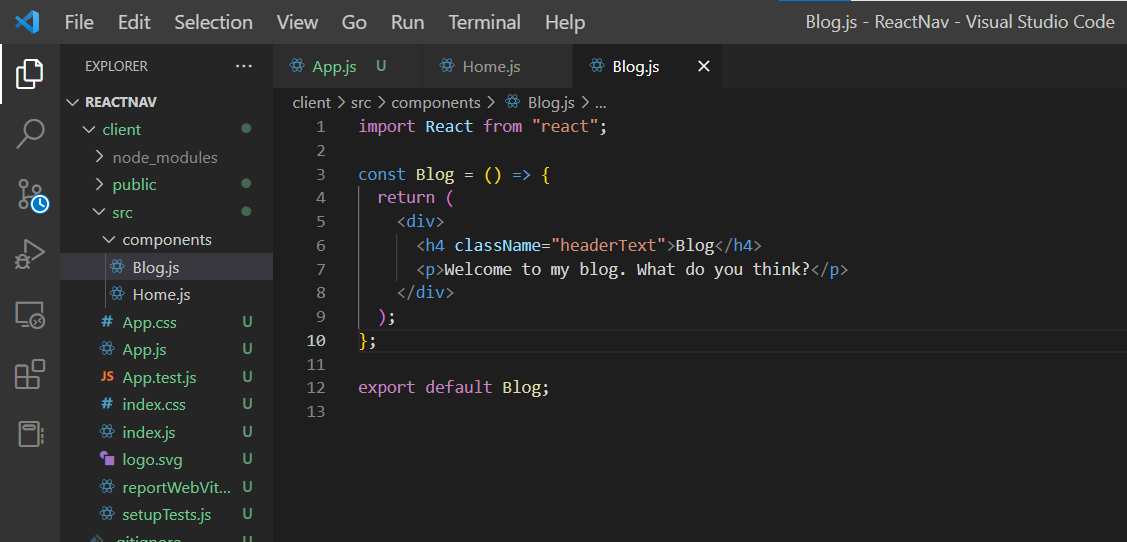
Upon saving the above code you can see the page in your browser do a live refresh to display your newest code. Of course it will be a blank page at this point.

1. **Adding Components and Routing**

For this project, we will create two basic React components in which we will use for routing in our App.js file.

Here is an example of two very simple components, Home.js and Blog.js:

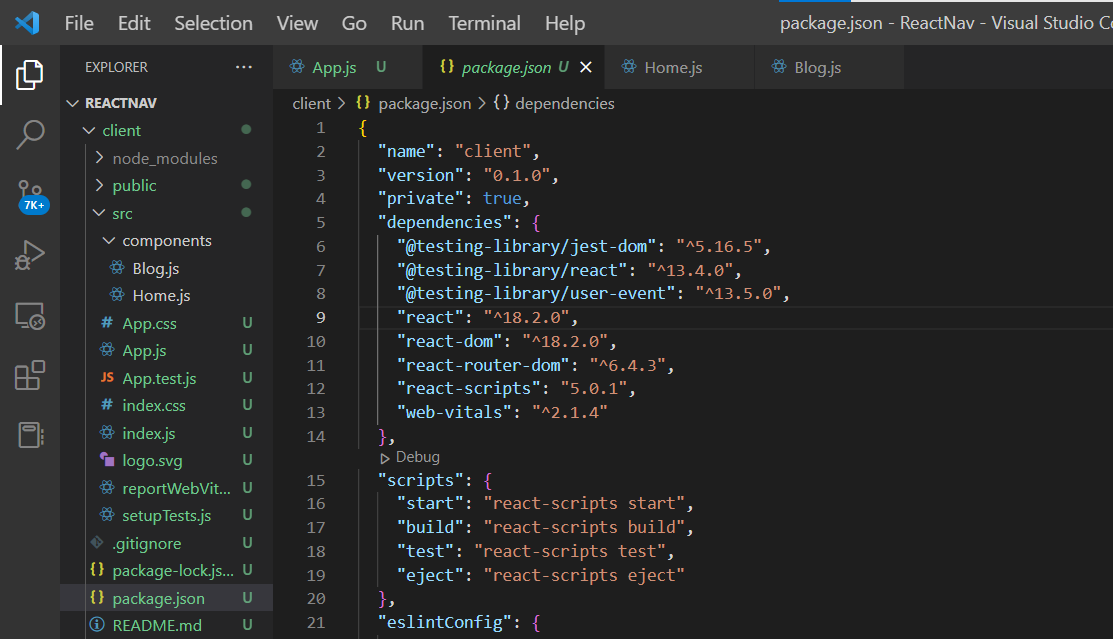




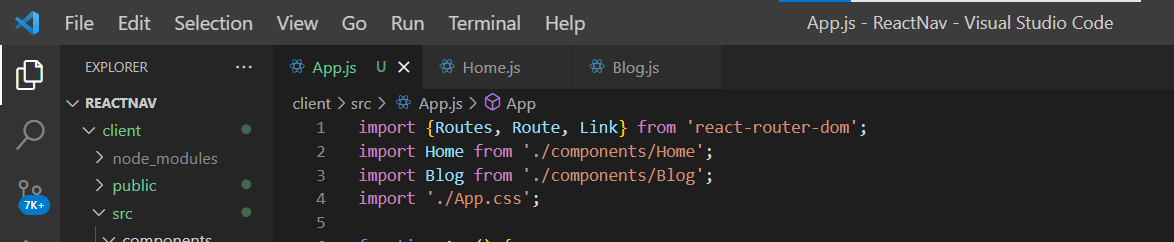
Since we have our two components set up, we can **navigate back to App.js** and use them in our routing.

Before we start to set up client-side routing with react, we need to install **react-router-dom** in our package.json file. Run a new terminal aside from your currently running react app, and cd into your client folder.

Install the react-router-dom dependency from npm. This will assist us in setting up client side routing. You should type in “**npm install react-router-dom**” into your command line. You should now see it in your package.json:

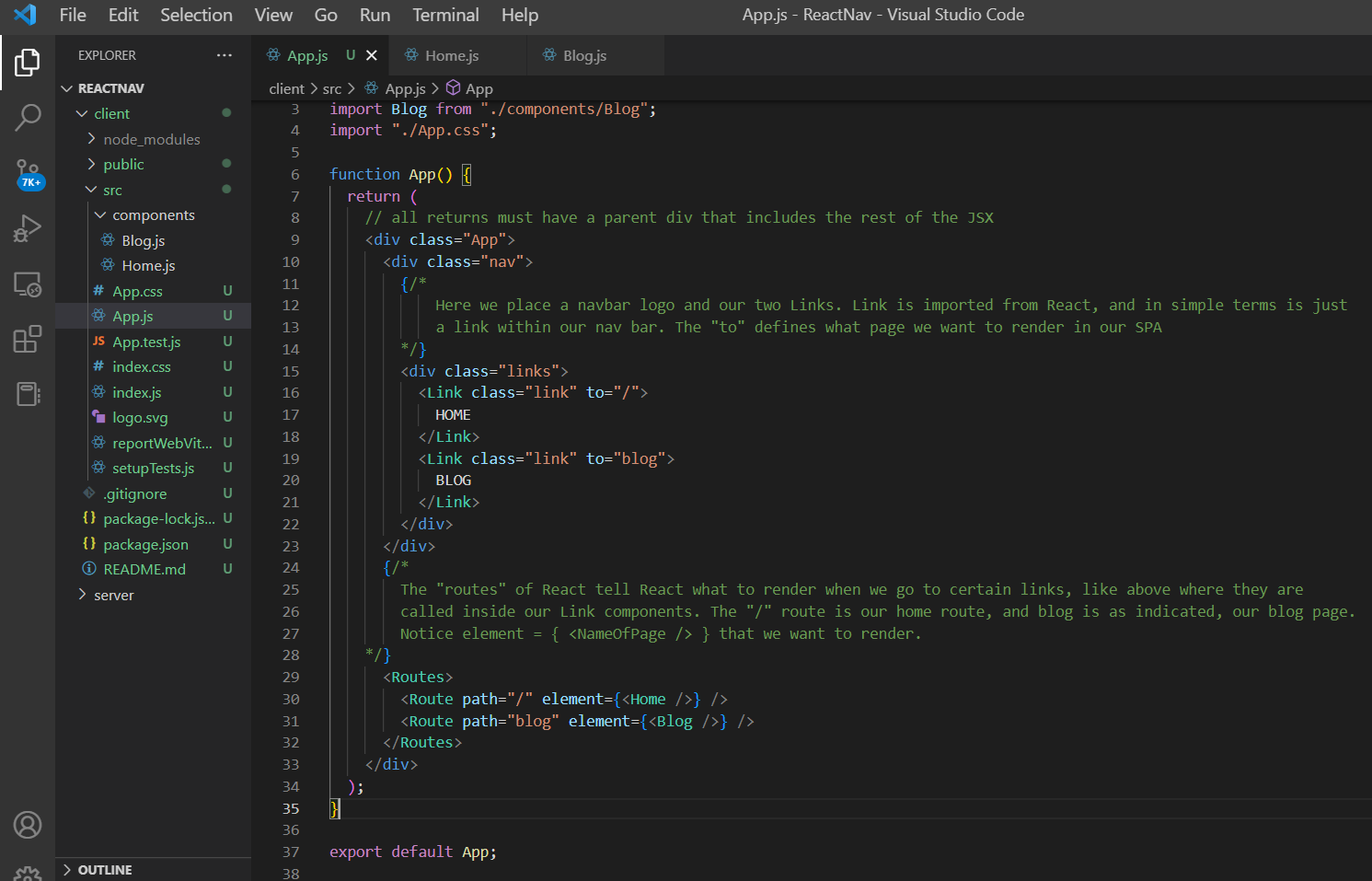


At the top of App.js, you must include the following imports:



This will allow you to use react-router-dom as well as the two components you created.

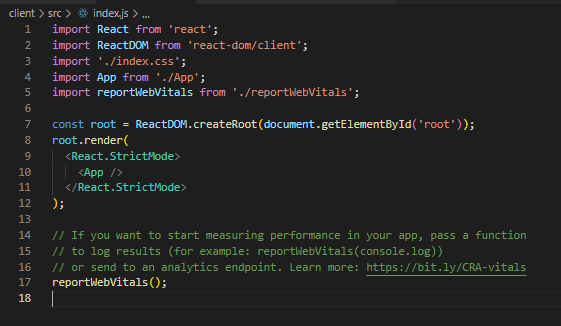
**Routing-**

****

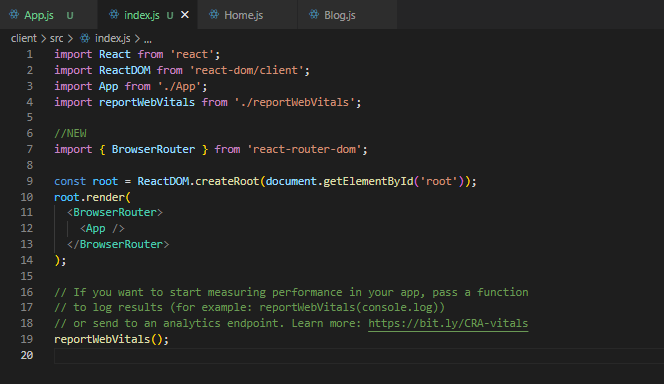
Here is a basic setup for routing in react from App.js.

You may notice at this point that your page is not displaying anything, if this is the case try changing this block of code in **index.js:**

**Before-**

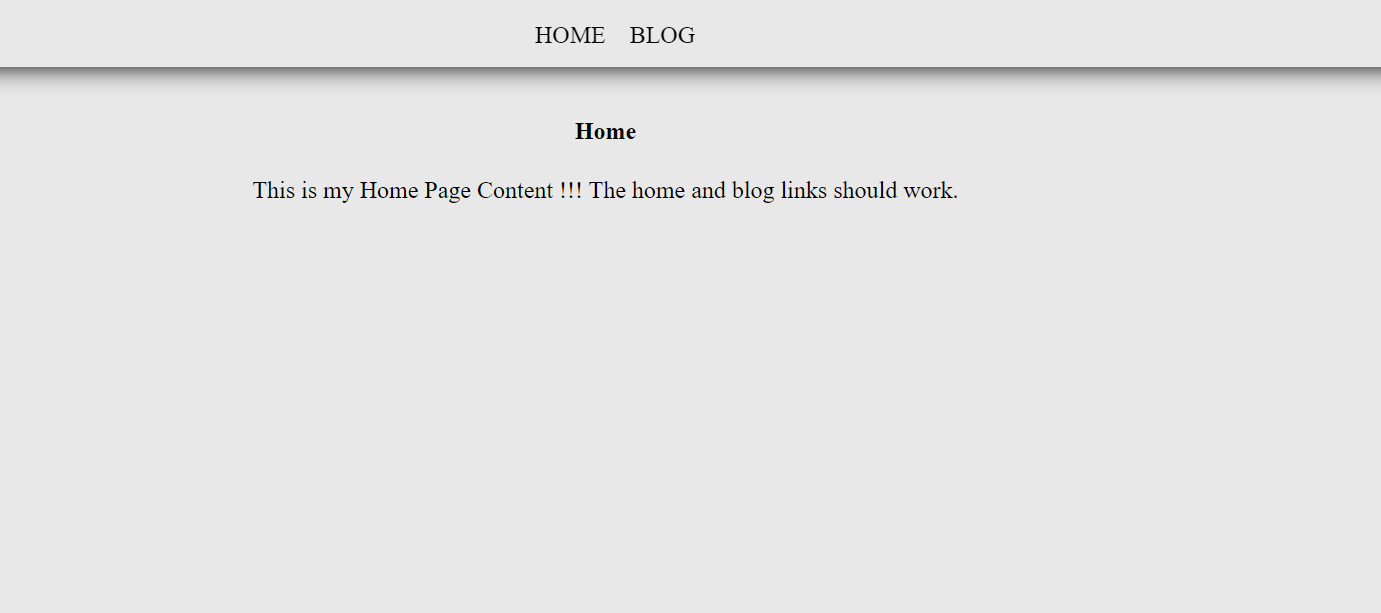


**After-**

****

All I did in this is import the BrowserRouter from react-router-dom and change the component that wraps <App /> from strict mode to the BrowserRouter.

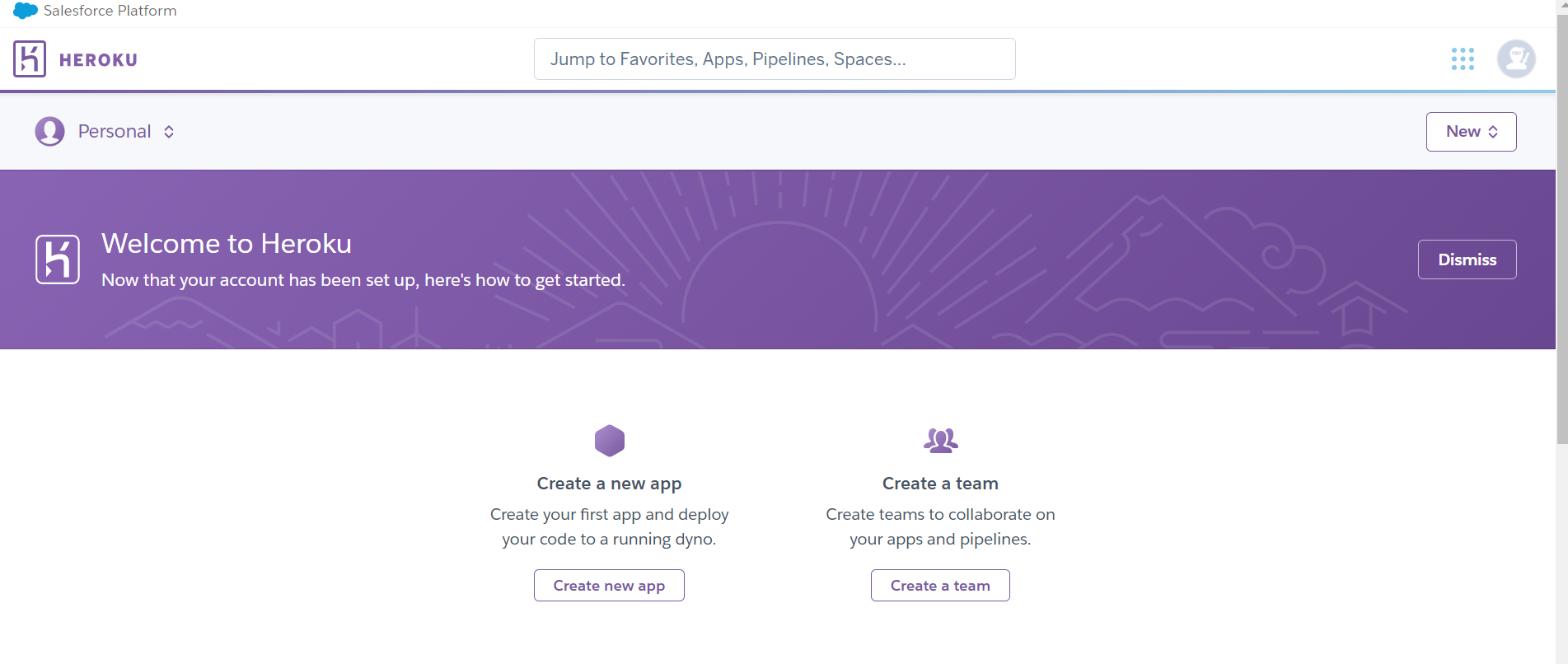
At this point you should have a very basic application with navigation. This won’t look great without some styling, so adding some basic styling can organize this page a lot better.



Of course this is extremely basic, so I suggest changing up the style in the best way you see fit.

1. **Create Heroku Account**

For our backend hosting, we will use Heroku (<https://www.heroku.com/>)

First, create a Heroku account. Once you have your account set up, you will see the heroku homepage: 

Since this project is frontend only, we do not need to add any database and can host this for free.

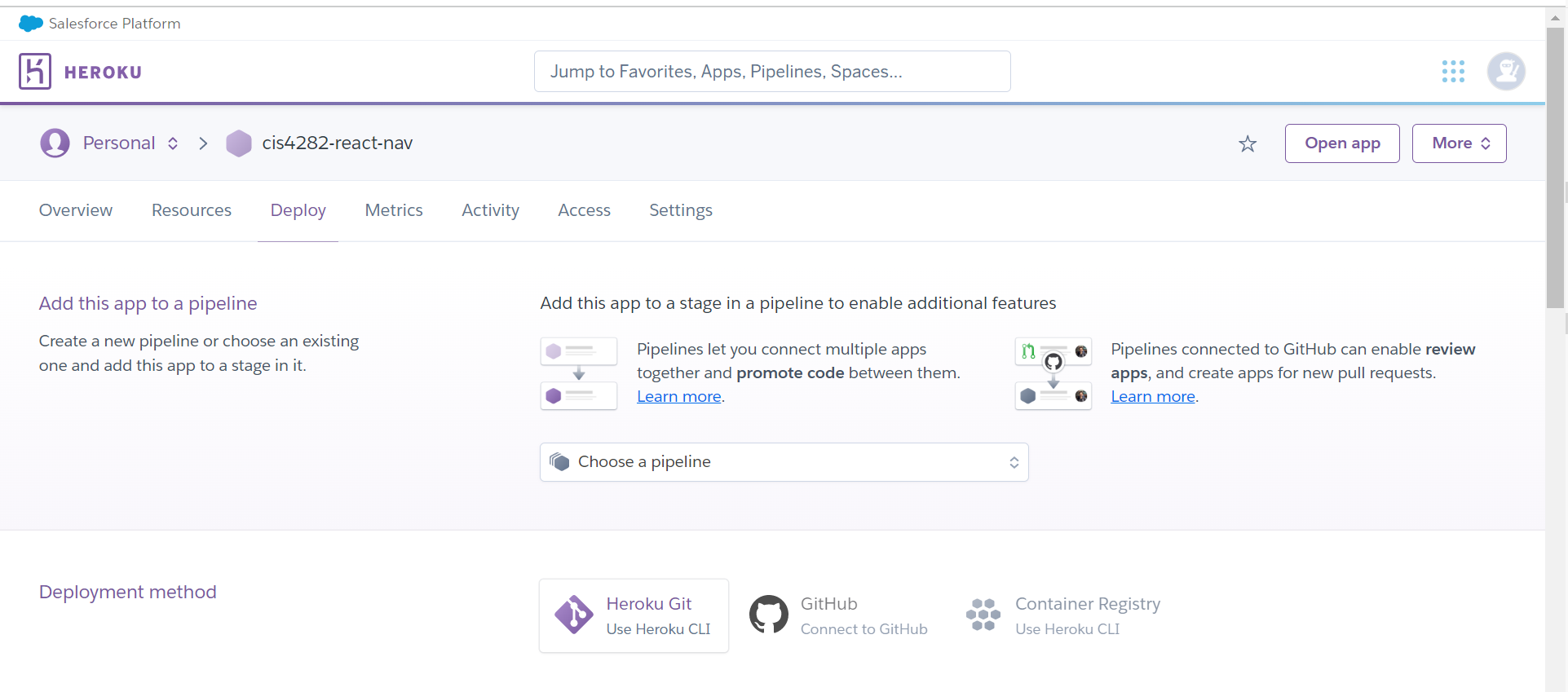
1. **Heroku App Setup**

Now we are ready to create a new app on Heroku, so if you haven’t already, navigate back to the homepage.

Towards the upper right you will see the  icon. Click that and select **“create new app”.**

Submit your app name and select your region.

You should see this screen after creating your app:



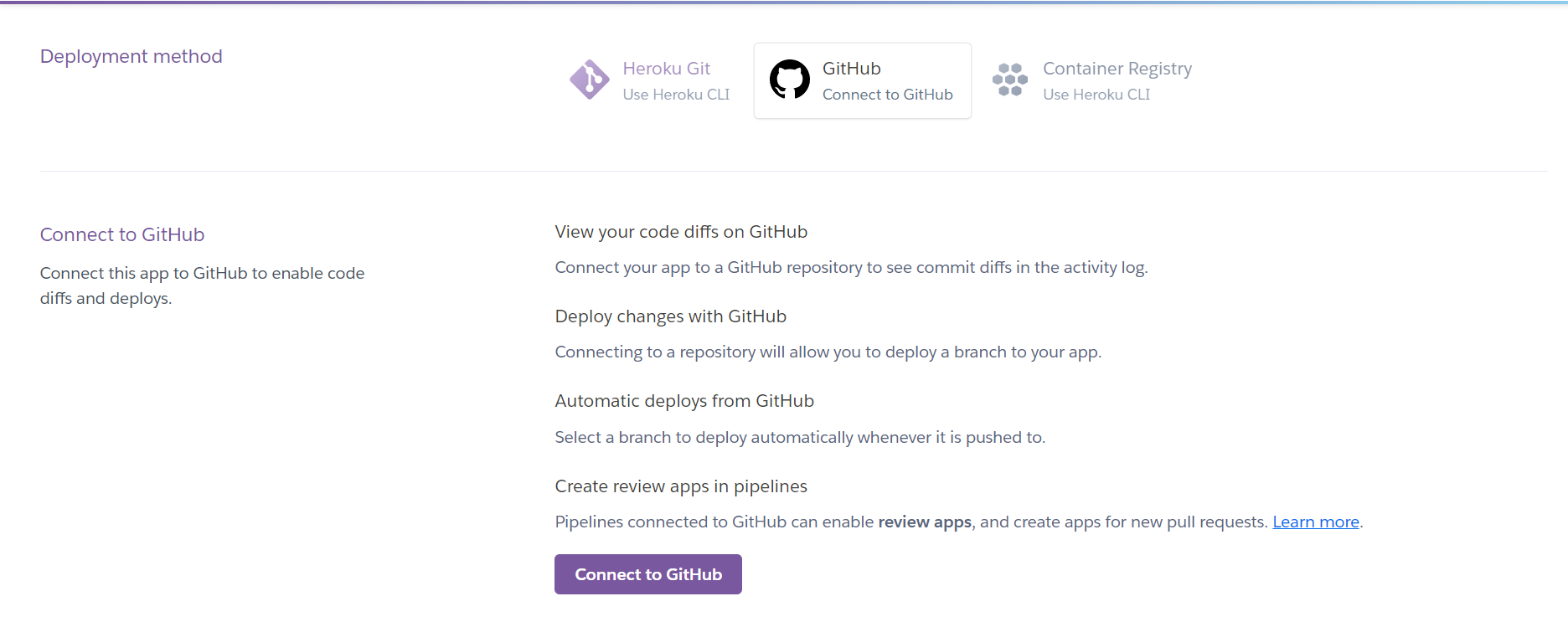
1. **Deployment**

One way we can deploy our app on Heroku is through **Github.** To do this, create a github repository and upload the contents of your project.

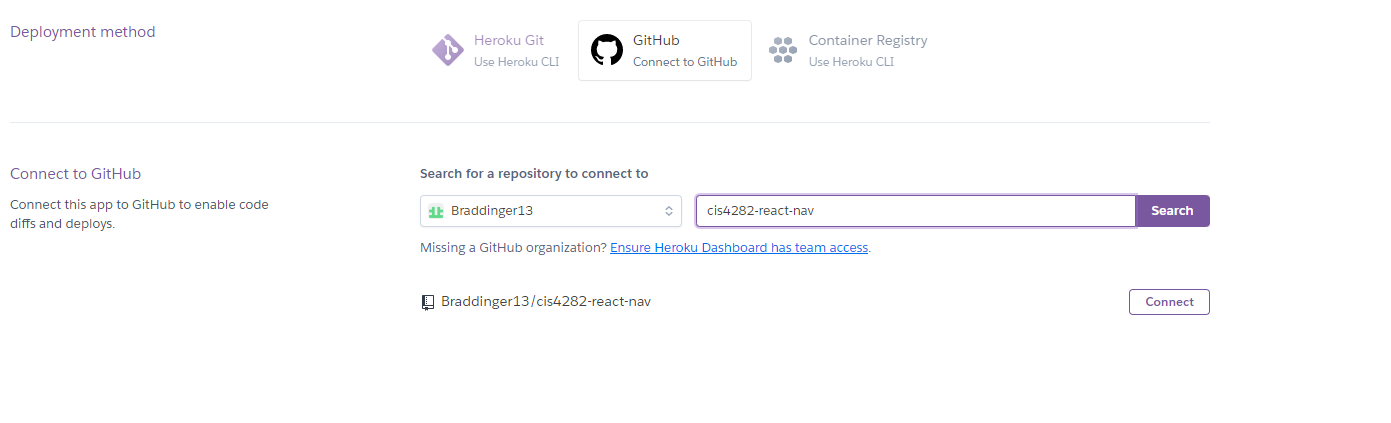
Here is a useful document for using git and setting up a github account/repository.

[GitHub Doc.pdf](https://drive.google.com/file/d/1dtDsmBL3reOUokJ--U7DwwnkVLjKhyEn/view)

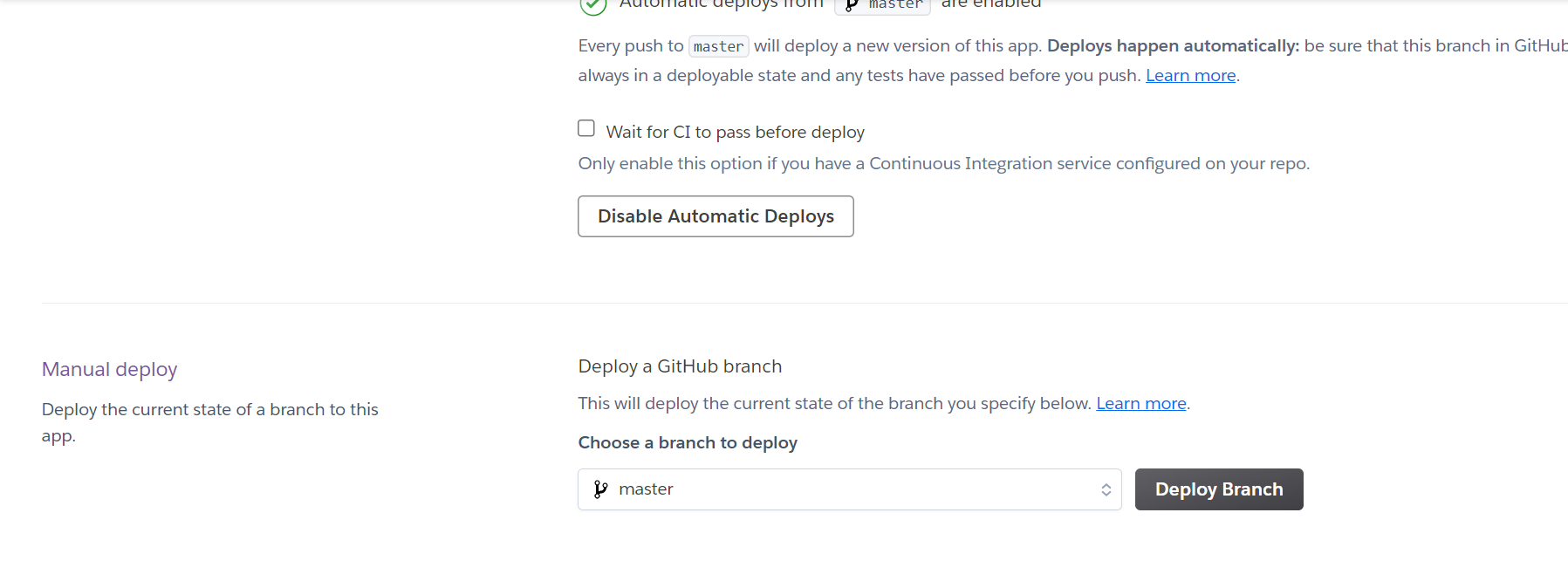
Once you have everything uploaded to github, navigate back to your Heroku app page and click the **Deploy** tab. You should see this in the deployment method section.



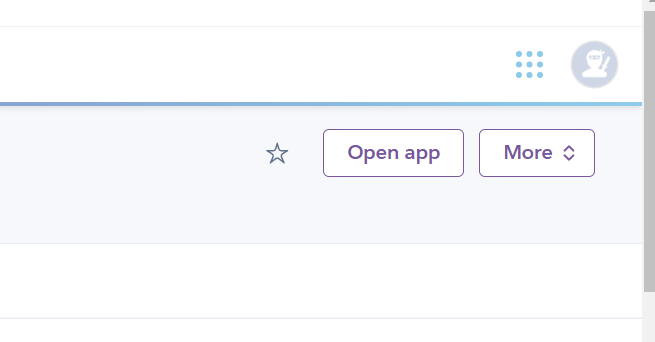
Once you are connected through github, you will be able to search for your repository.



Click connect. Then I would recommend enabling automatic deployment so whenever you push your project to your main/master repository branch, Heroku will redeploy your site with new changes.



After clicking deploy, it will take some time to complete, but you should be able to access your app and database through the web!



In the upper right corner of this page, you can click **Open app**, and it will bring you to your newly created page with your app’s name as its prefix.

