Node

Angular

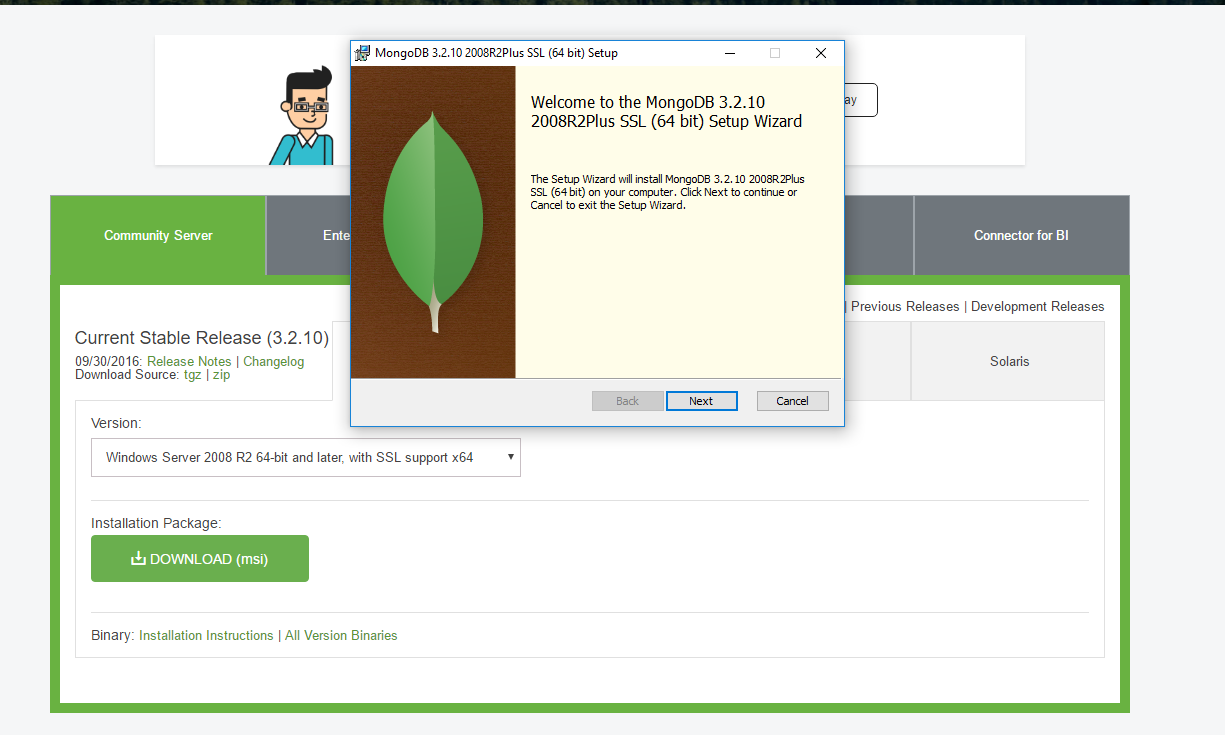
Dev work

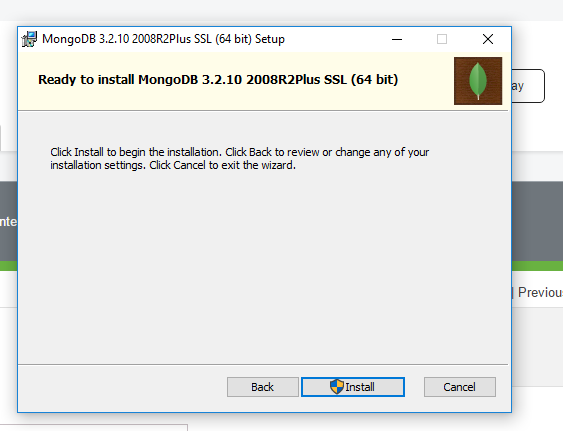
<https://www.npmjs.com/package/nodemon>

Database

MongoDB mongoose

Install with <https://docs.mongodb.com/getting-started/shell/tutorial/install-mongodb-on-windows/>



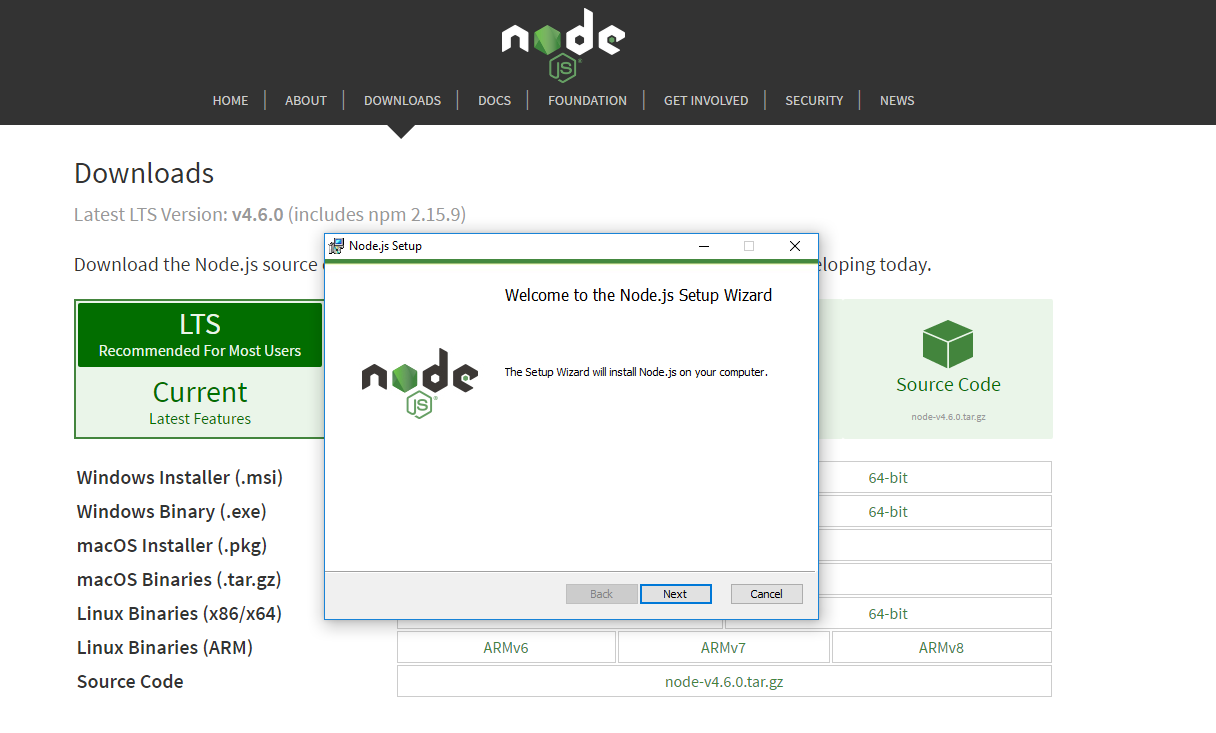


Node

First off install node.js. From this we can use NPM (Node Package Manager) and set up a server to run our app off.

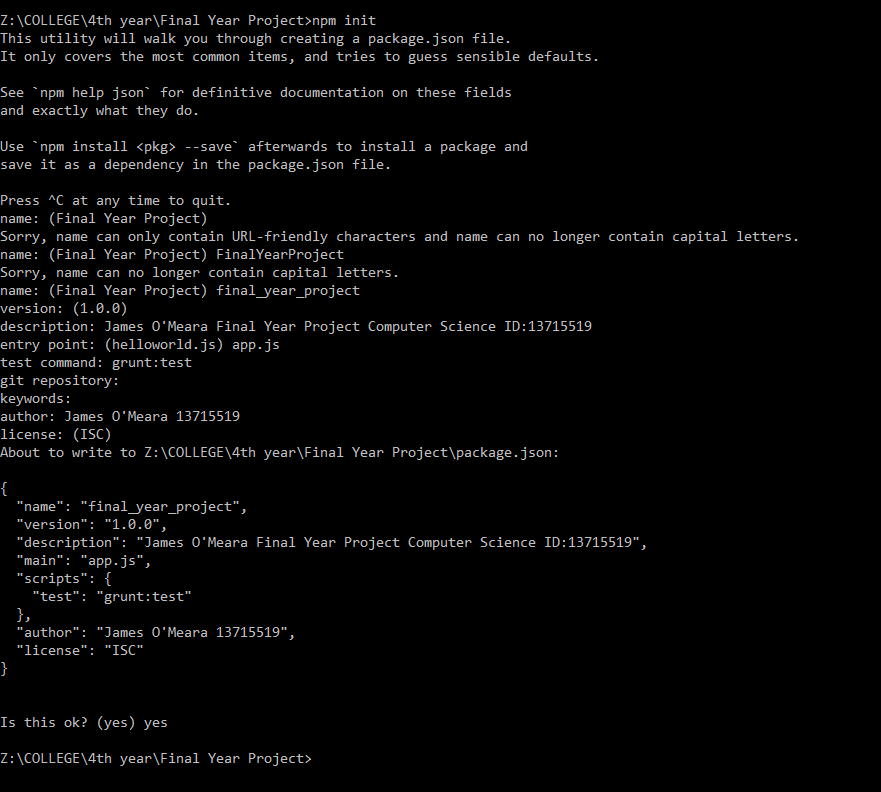
Download link for Node can be easily found on the site: <https://nodejs.org/en/download/>

From here pick your Operating System and version.

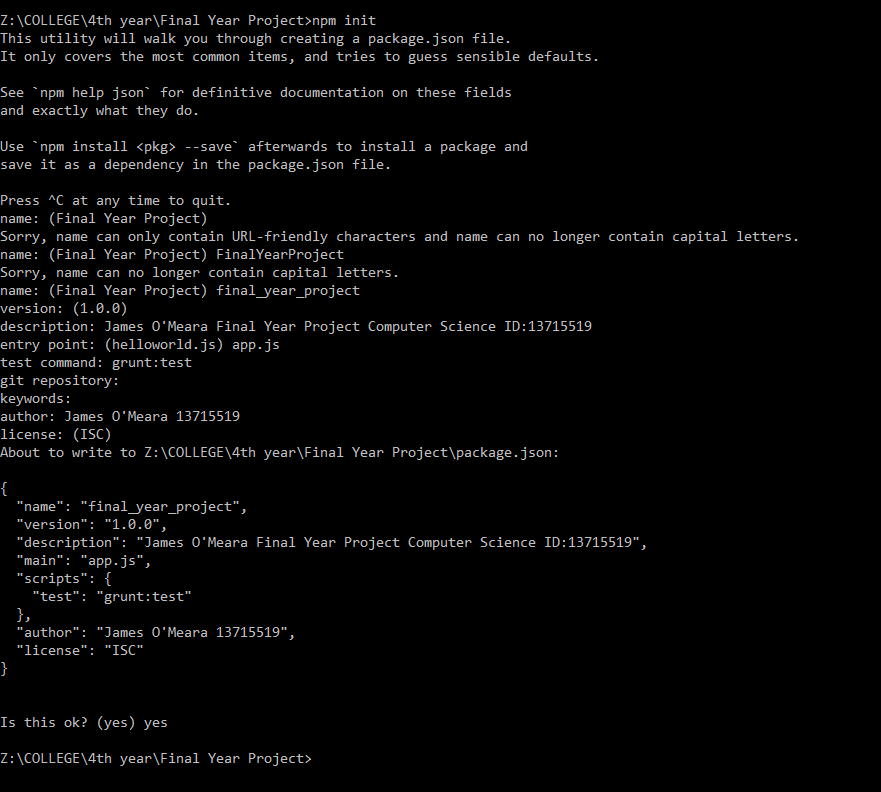


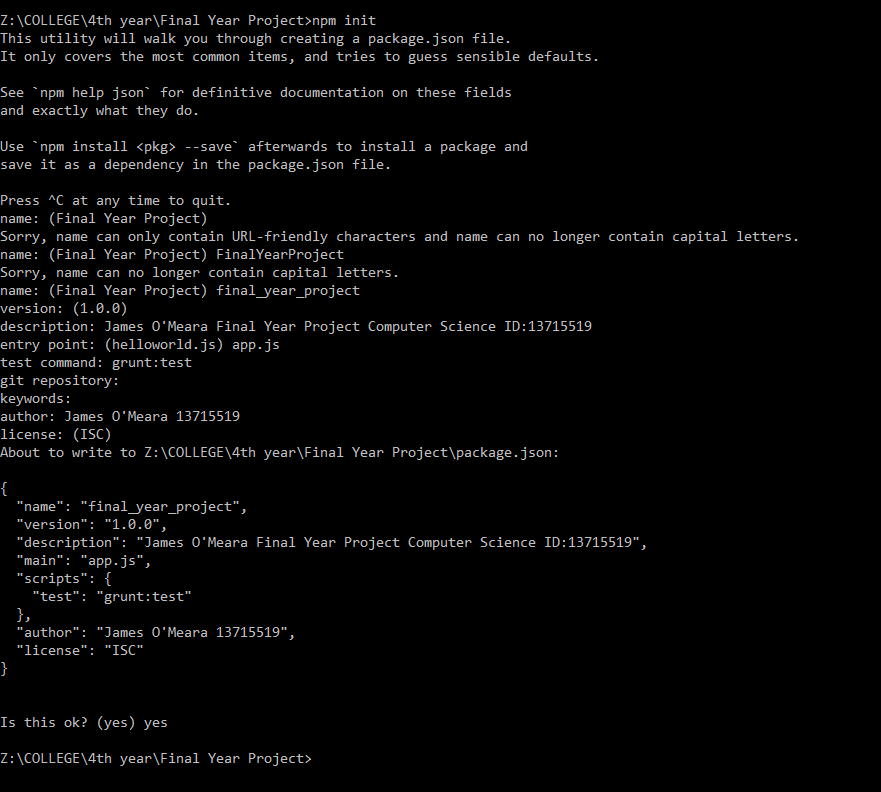
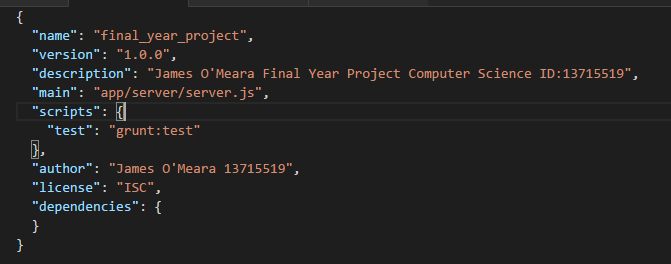
Next we can create our App folder. Here I have called mine “FinalYearProject”. From here we can go through the steps of initializing a node project. This will result in node creating a “package.json” file which is used for node/project settings and list of dependencies.

1. Open a command window in this new folder you just created
2. Type “npm init”



1. Next you will go through the setup, (leaving field blank will use default) you can copy the same input I have entered here if you wish.



1. When you have finished, you will be prompted with the output of the package.json
2. If happy with this, type “yes” and press enter. Remember that these can be changed at any time in the future.
3. Finally you will now have this “package.json” file in your folder. 

NPM

Installing a dependency

NPM (Node Package Manager) Hosts a very large range of open source libraries/degeneracies and they can be easily downloaded/installed into your project.

This can be done by:

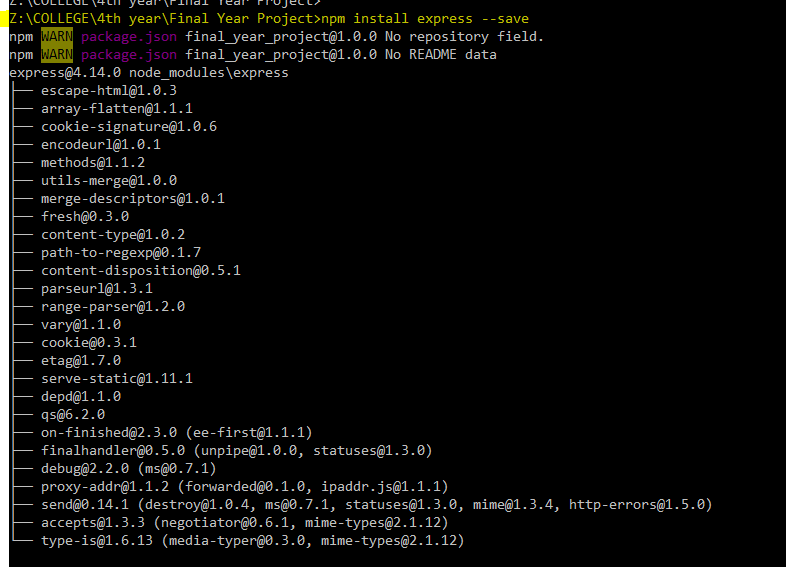
1. opening a command window inside your project, as I have one open below, (you will find the package.json at this level).
2. Then type the command “Npm install (followed by the package you wish)”

E.g.: “Npm install express”

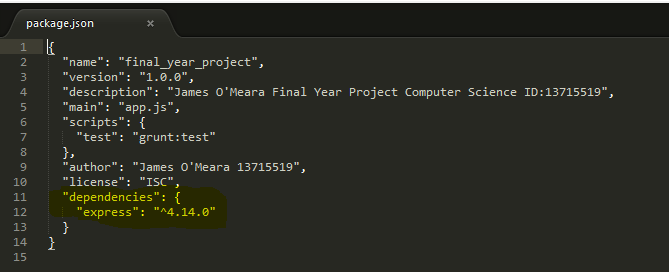
1. If you wish to save this to your package.json as a dependency you can do so by adding “—save” to the end of it.

E.g.: “Npm install express –save”

1. One other option for installing dependencies is by adding “—save-dev” to the end of the command instead of “—save”, as save-dev will mean this dependency is only used for development. And will not be bundled into the release of the app.



1. Finally you will see this dependency reflected in your package.json.

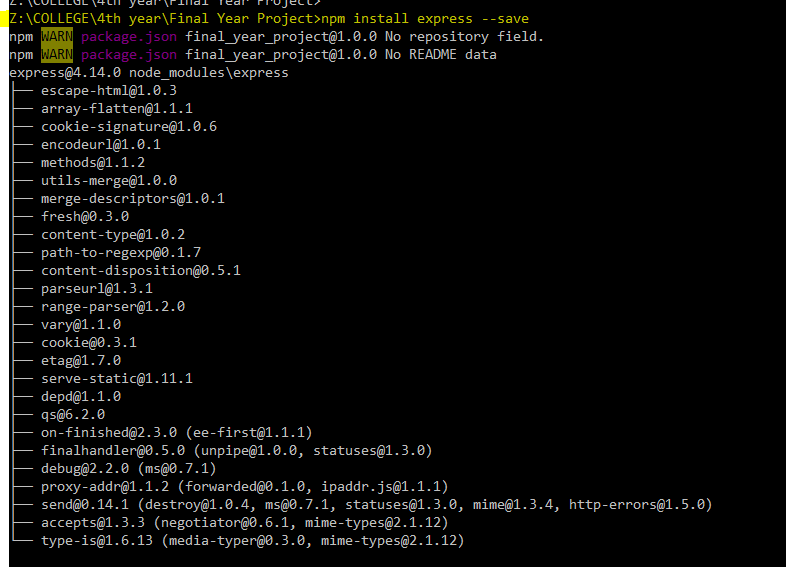


Setting up node server

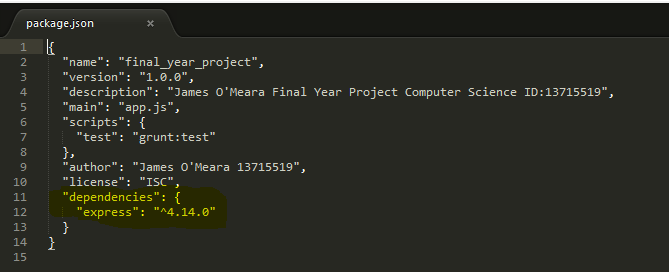
Our app will require a server side, node does this easily and can be set up very quickly.

We are going to use an express server, First off install express via Npm command:

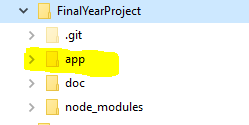
“npm install express –save”



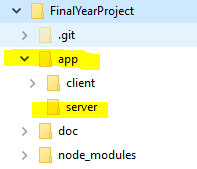
After installing the package your package.json should have updated to reflect the new dependacy since you stated that you wanted to “—save” it.



Once we have this done we can no create our app folder to store all our source code. Create a folder “app” inside the folder we previously created (this one is called “FinalYearProject”)

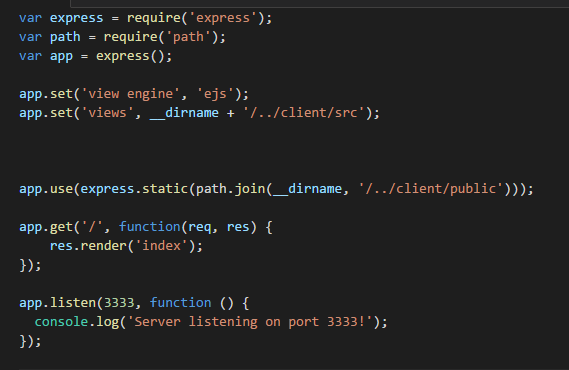


Then inside this create a folder named “server” any code here dealing with the server will be put in here.

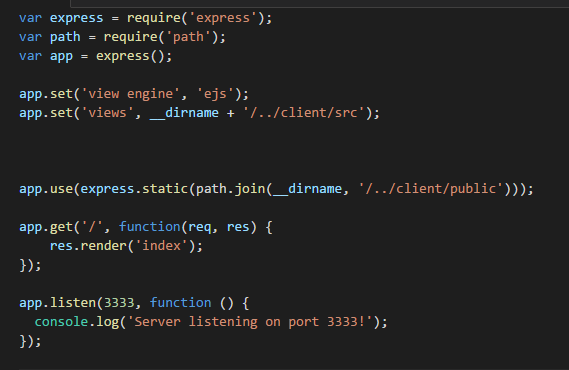


Create a file Server.js

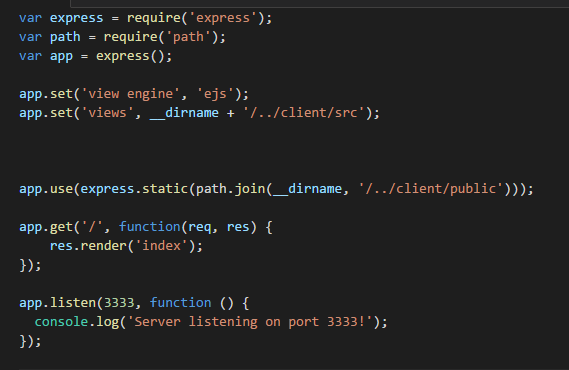
We will run this file when we want to start our server, this will also deal with the routing along with many other things.



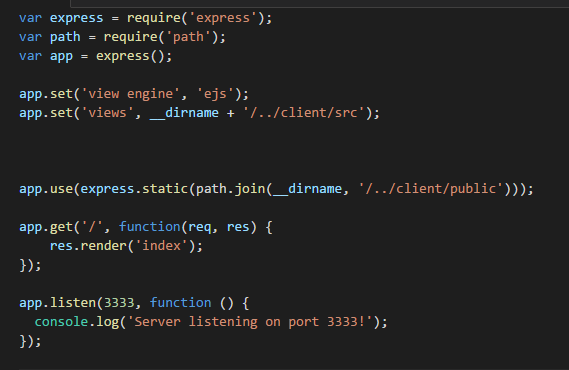
First off we must require any dependencies for our server, here we need express and path, Express to run an express server and path to tell our server what location to return files from.



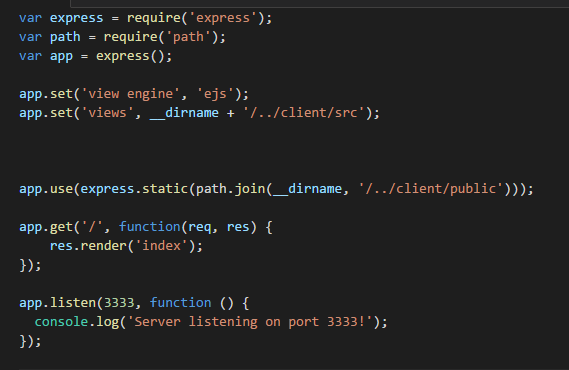
Next we must set our view engine, in this app we are using .ejs files not html files. So we set our view engine to .egs and tell it where it will find the files from.



Another way to tell our server where to find all the files located in joining our local folder to the express static location, this will ensure that it will look for files in the directory you specify here.



Now we tell our server what file to render when we enter the particular url into the browser. Here we are just specifying the home url.



Then finally we want to start our server, specify the port here and the functional code inside. Here we are just printing out a simple message to know that we are up and running.

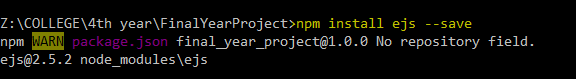
Run the server by

Node server.js



EJS

Creating index.egs.



Create a file called index.ejs in app/client/ and create a simple webpage to display a hello world message. This will be the entry point for the app in which the server will render to the user.

