https://medium.com/@vandal777/portfolio-hecho-en-reactjs-publicado-con-github-pages-921a57a717ea

https://www.styleshout.com/free-templates/ceevee/

https://transform.tools/html-to-jsx

Portfolio hecho en ReactJs publicado con GitHub Pages

Captura de mí Portfolio creado con Create-React-App

Github a parte de ser un increíble gestor de repositorios que nos permite crear nuestros repositorios públicos, desde finales del año 2018 nos permite también crearlos como repositorios privados de forma gratuita. Pero a parte de permitirnos alojar nuestros repositorios, también nos brinda la oportunidad de publicar

nuestros proyectos en la web, proporcionándonos de esta manera un alojamiento web gratuito, para poder hacer nuestras pruebas o incluso poder crearnos un Portfolio.

Eso es lo que vamos en este Post **¿ A que es genial?**

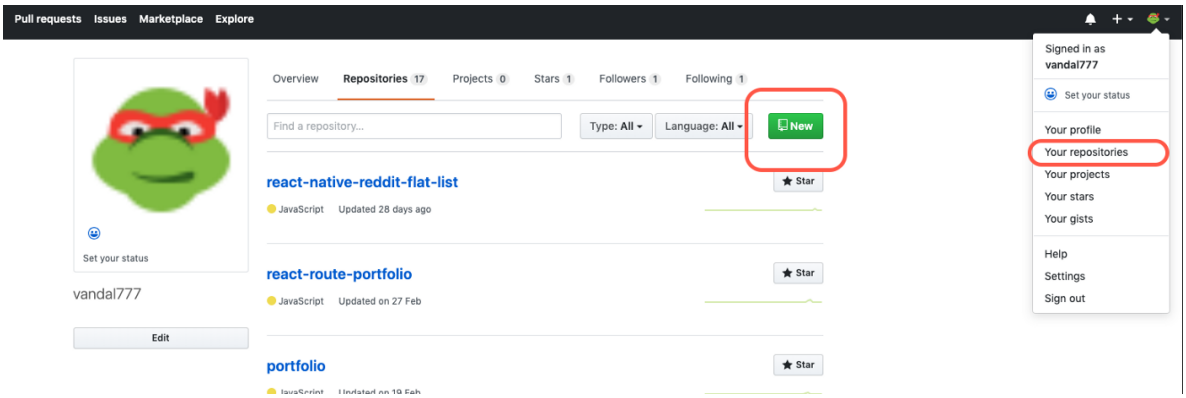
Pre-requisitos: Para poder desarrollar esta aplicación asumo que tenemos instalado NodeJS. Si no para instalarlo podemos ir aquí.

Node ≥ 6 npm ≥ 5.2

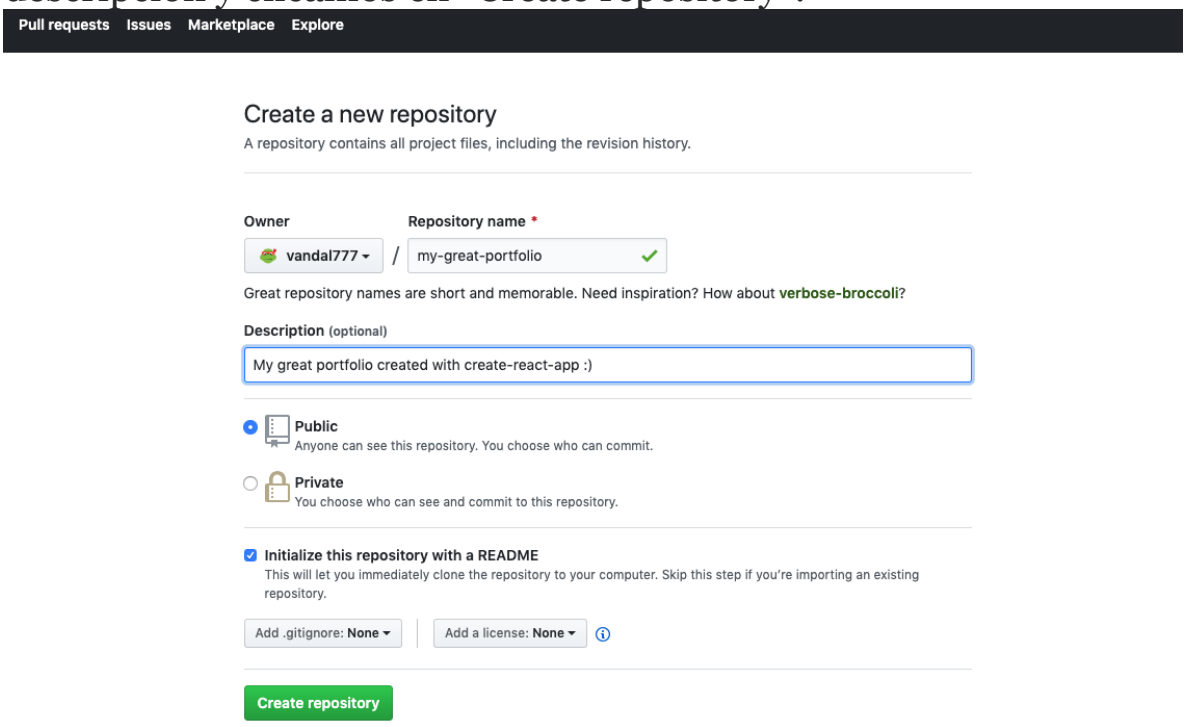
1. Creación de un repositorio de GitHub

Para empezar lo primero que necesitaremos es tener una cuenta de **GitHub** para poder crear nuestro repositorio vacío**.**

Una vez hemos creado nuestra cuenta de GitHub, procedemos a crear un repositorio vacío que utilizaremos para subir nuestra aplicación, desde la pestaña de nuestro usuario, iremos a “Your repositories” y después hacemos clic en el botón “New” que he resaltado en la siguiente imagen.



Después le damos un nombre a nuestro repositorio, una descripción y clicamos en “Create repository”.

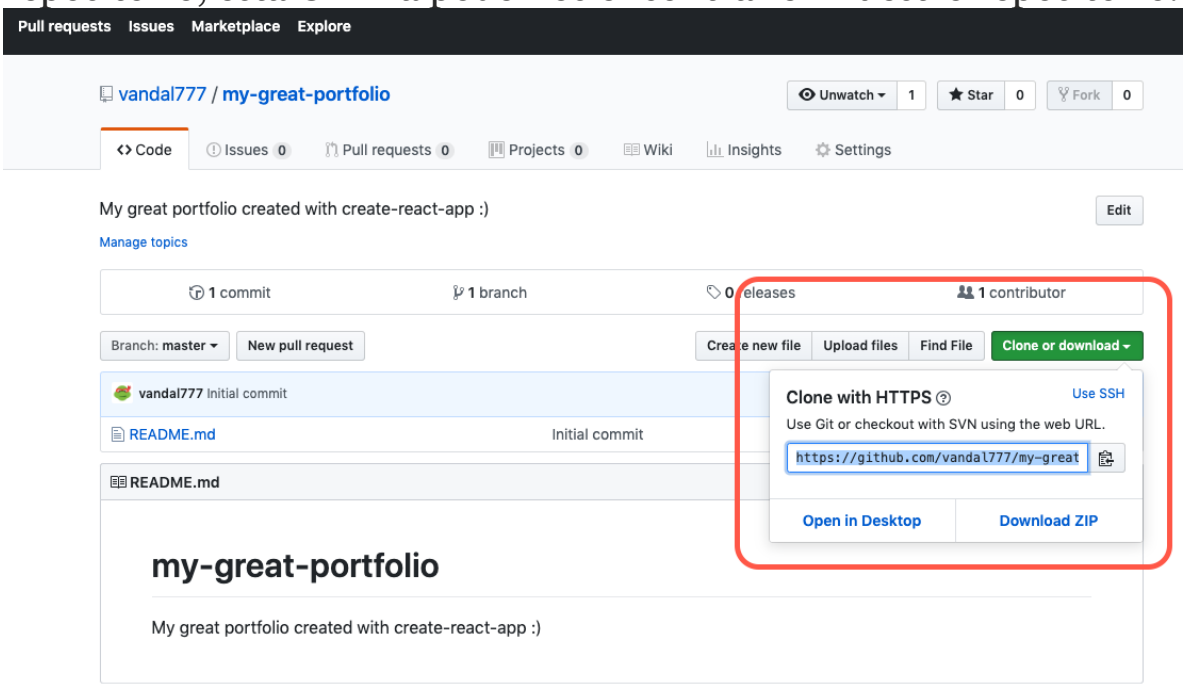


2. Clone de nuestro repositorio de GitHub en nuestra maquina local

Lo siguiente que haremos después de haber creado nuestro repositorio es clonar nuestro repositorio en nuestra maquina local.

Iremos a nuestro IDE, yo en este caso estoy utilizando Visual Studio Code y nos haremos un clone de nuestro repositorio vacío.

Para poder hacer él clone necesitamos la URL de nuestro repositorio, esta URL la podemos encontrar en nuestro repositorio.



URL para clonar nuestro repositorio

Con esta URL nos vamos al directorio donde queremos clonar nuestro respositorio y hacemos el clone.

Después nos movemos a la carpeta de nuestro directorio y ya podemos ver como ahí tenemos nuestro README que creamos en el repositorio.

```
git clone https://github.com/vandal777/my-great-portfolio.git
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

users-Mac-mini:Tests developer1$ git clone https://github.com/vandal777/my-great-portfolio.git
Cloning into 'my-great-portfolio'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
users-Mac-mini:Tests developer1$ cd my-great-portfolio/
users-Mac-mini:my-great-portfolio developer1$ ls
README.md
```

Captura del clone de nuestro repositorio

3. Creacion de nuestra aplicación con ReactJS

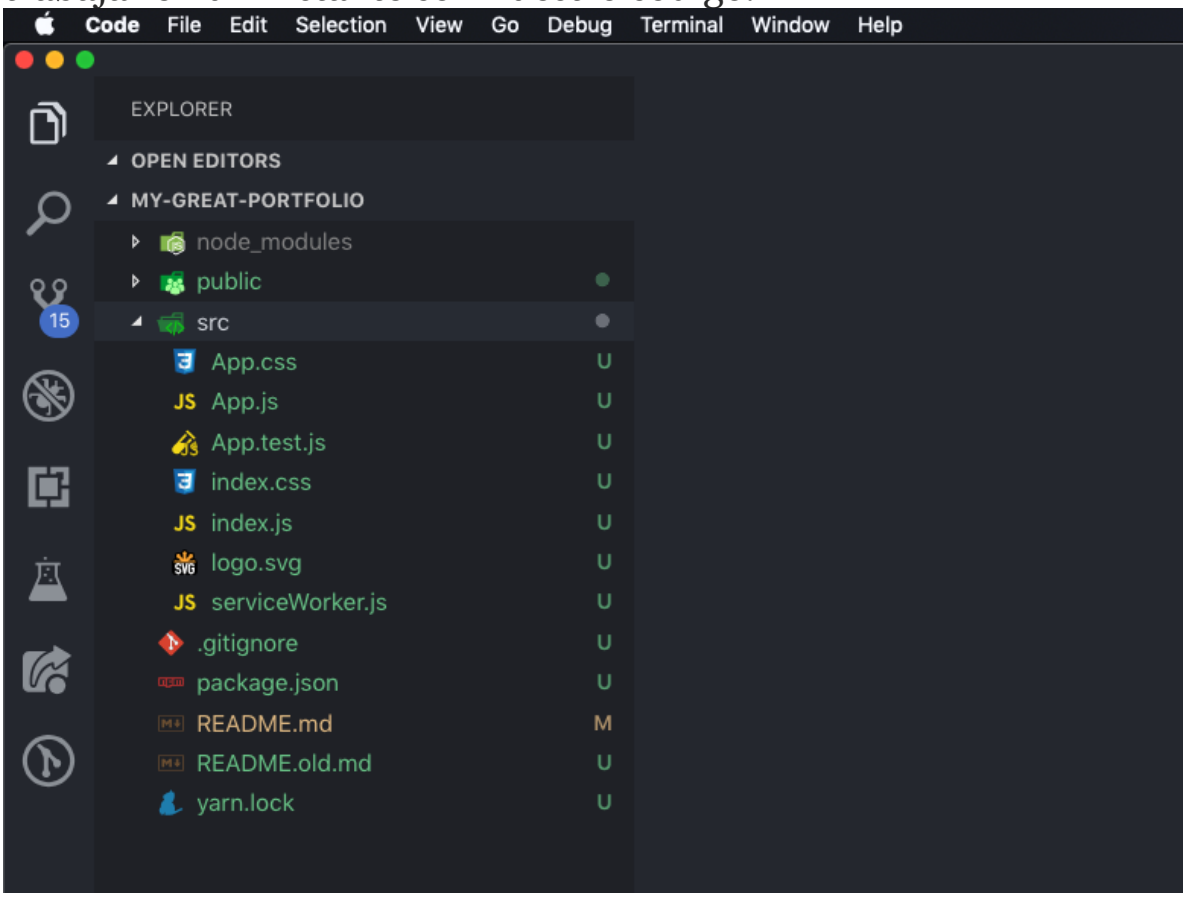
Una vez tenemos creado y clonado nuestro repositorio ahora vamos a crear nuestra aplicación con create-react-app

Mi primer create-react-app… wiiiiiiiiiiiiii

En la **misma carpeta** donde hemos hecho él clone de nuestra aplicacion haremos la creacion de nuestra aplicacion con create-react-app, para ello ejecutaremos el siguiente comando.

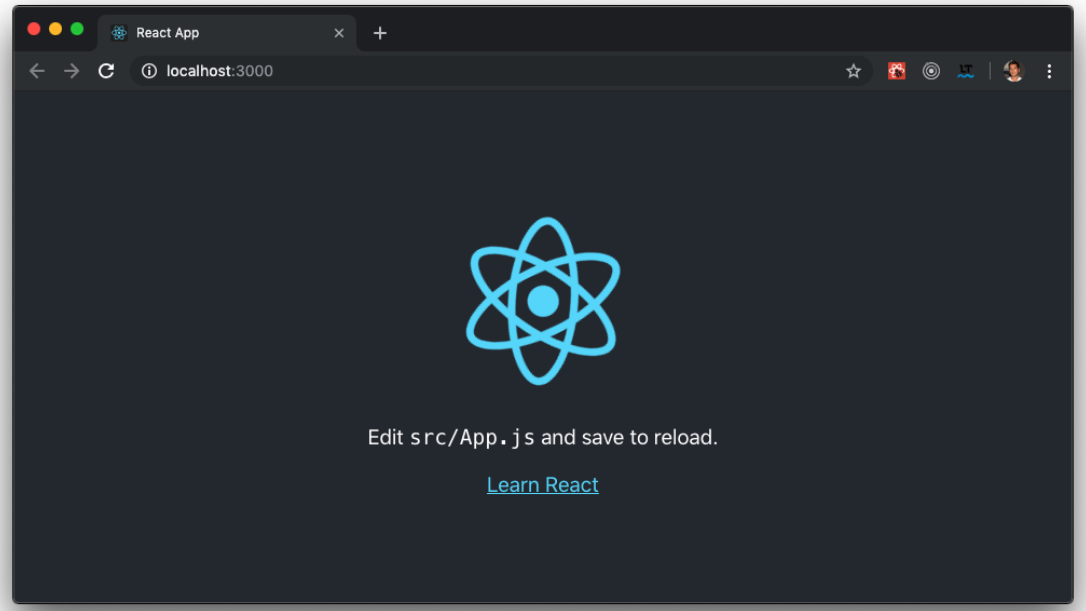
```
npx create-react-app my-react-app
```

Como podemos observar, **create-react-app** nos ha creado una serie de archivos y de directorios, que nos permiten comenzar a trabajar en un instante con nuestro código.



Ahora ya podemos iniciar nuestra aplicación con el comando. Se nos abrirá una ventana de Chrome con nuestra primera aplicación :D !!!

```
npm start
```



4. Creación de nuestro portfolio

Una vez ya tenemos nuestra aplicacion creado, vamos a proceder a crear nuestro propio Portfolio.

En mi caso y para hacer esta demostracion, he utilizado un Template ya existente en internet, el cual podemos descargar de forma gratuita y amoldarlo a nuestras necesidades. Simplemente tenemos que saber un poco de CSS y de la logica de componentes de ReactJs.

1. Lo primero que necesitamos entonces es descargar el Template de esta URL y descomprimirlo en un lugar conocido para despues acceder a él.

https://www.styleshout.com/free-templates/ceevee/

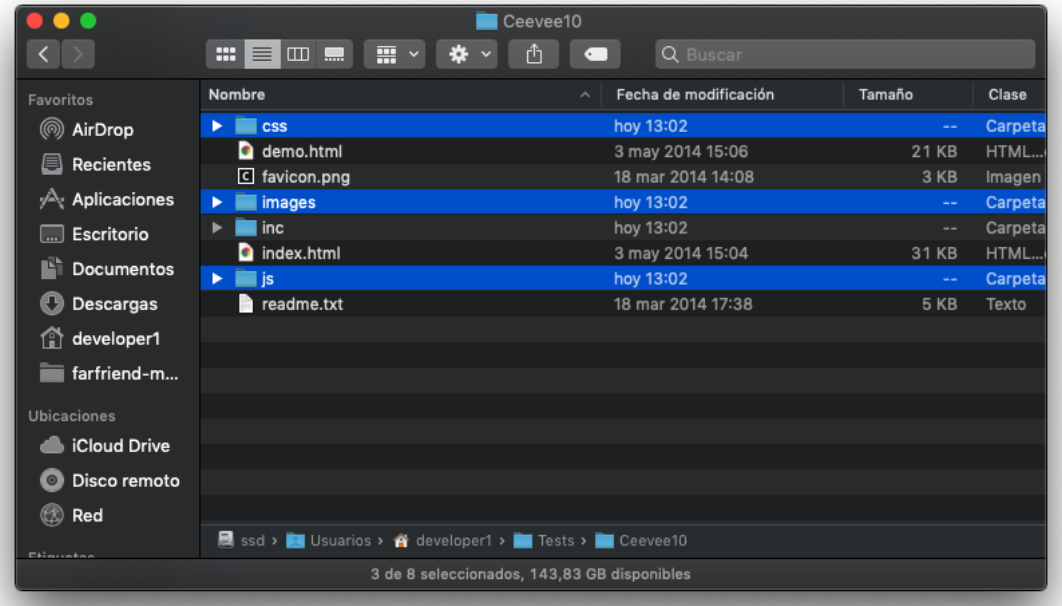
2. Ahora que ya tenemos el Template.

De la carpeta del template copiamos las carpetas **js, images y css la carpeta public que está en** raiz de nuestro proyecto

Ceevee10/js → my-great-portfolio/public

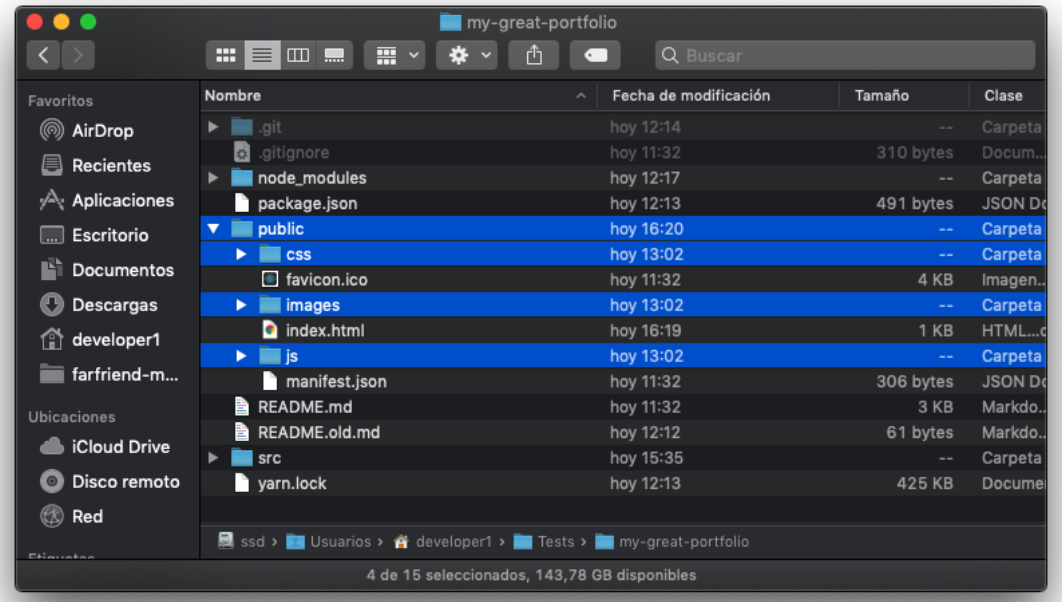
Ceevee10/images → my-great-portfolio/public

Ceevee10/css→ my-great-portfolio/public



Estas 3 carpetas las copiamos dentro de nuestro proyecto

Nos quedaría de esta manera:



3. Ahora vamos a editar el archivo **index.html** que esta dentro de nuestro proyecto en la ruta:

```
my-great-portfolio/public/index.html
```

Le añadimos estas etiquetas dentro de la etiqueta **<head></head>** del archivo **index.html**

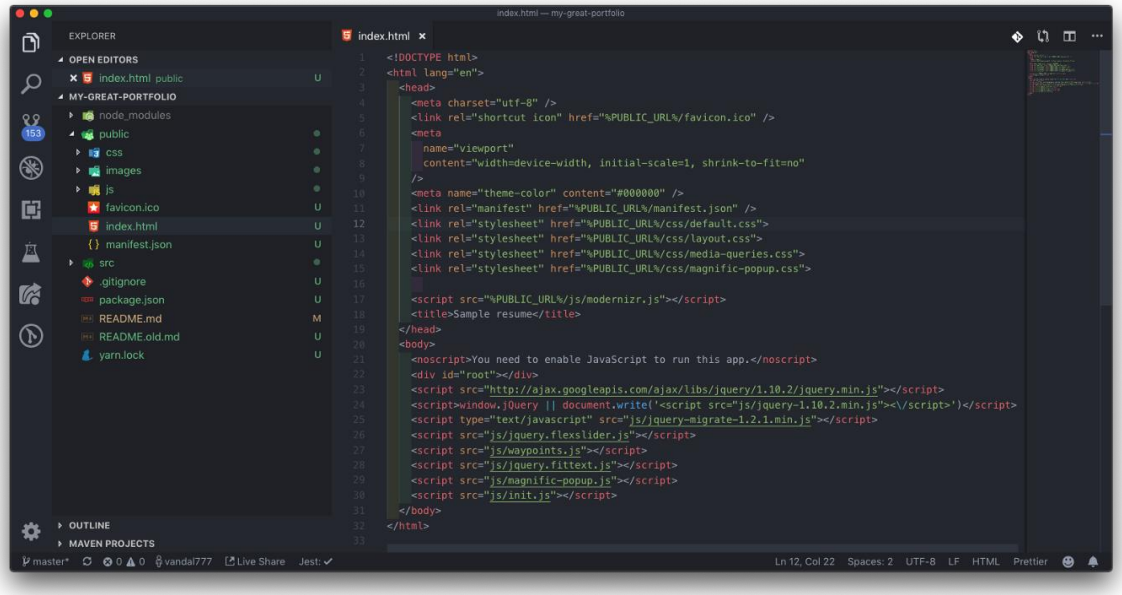
```
<link rel="stylesheet" href="%PUBLIC_URL%/css/default.css">
<link rel="stylesheet" href="%PUBLIC_URL%/css/layout.css">
<link rel="stylesheet" href="%PUBLIC_URL%/css/media-
queries.css">
<link rel="stylesheet" href="%PUBLIC_URL%/css/magnific-
popup.css">

<script src="%PUBLIC_URL%/js/modernizr.js"></script>
<title>Sample resume</title>
```

Dentro del mismo archivo **index.html** añadimos lo siguiente debajo de este div**:**


```
<script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.
min.js"></script>
    <script>window.jQuery || document.write('<script
src="js/jquery-1.10.2.min.js"><\/script>')</script>
    <script type="text/javascript" src="js/jquery-migrate-
1.2.1.min.js"></script><script
src="js/jquery.flexslider.js"></script>
    <script src="js/waypoints.js"></script>
    <script src="js/jquery.fittext.js"></script>
    <script src="js/magnific-popup.js"></script>
    <script src="js/init.js"></script>
```

Quedaría de esta manera:



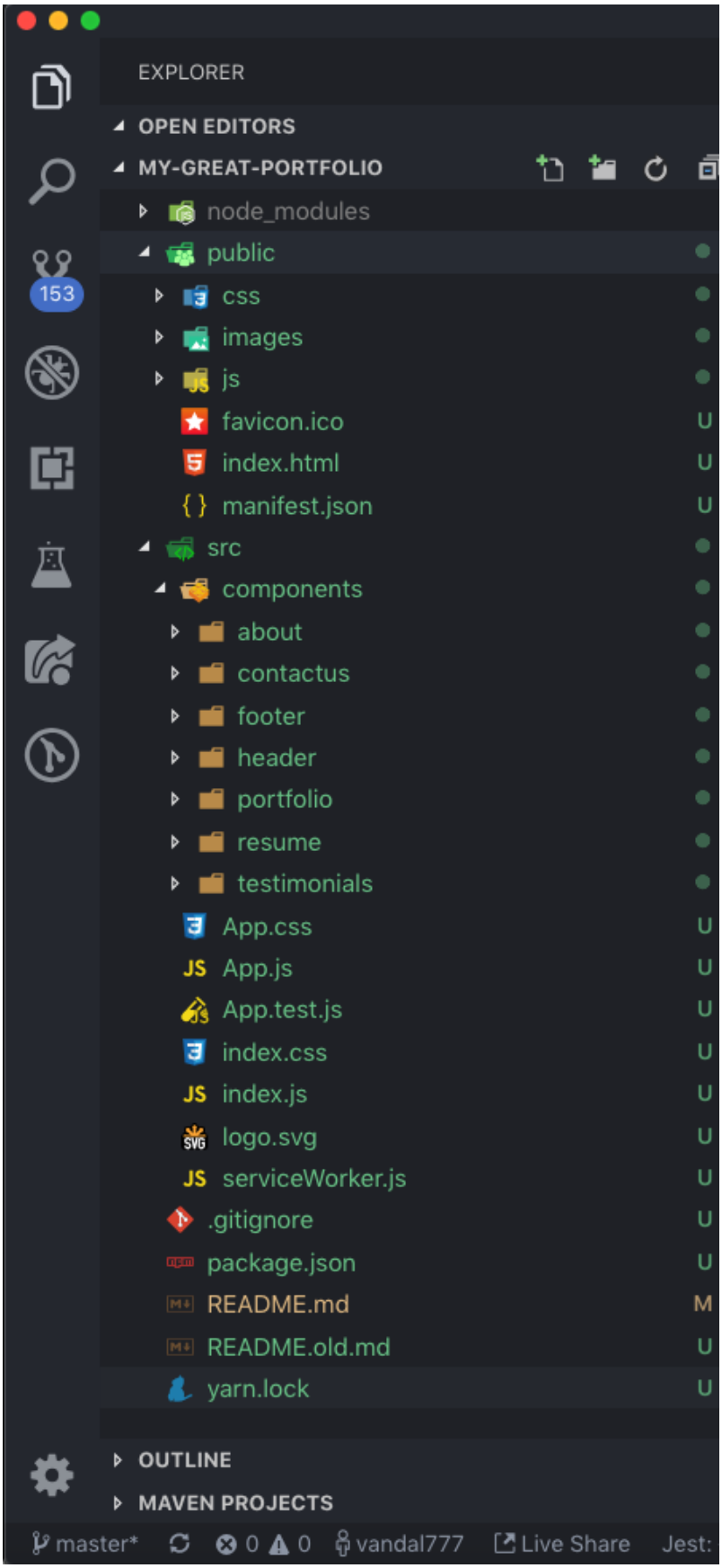
4. Ahora vamos a convertir las pantallas definidas en el Template, en diferentes componentes. Para ellos vamos a crear un directorio de carpetas para los diferentes componentes, con su respectivo contenido.

Toda esta estructura la crearemos dentro de la carpeta **/src/components**

Desde la raiz de nuestro proyecto vamos a ejecutar los siguientes comandos.

```
cd src
mkdir components
cd components
mkdir header
mkdir about
mkdir resume
mkdir portfolio
mkdir testimonials
mkdir contactus
mkdir footer
```

Para conseguir esta estructura de carpetas:


EXPLORER
OPEN EDITORS
MY-GREAT-PORTFOLIO
node_modules
public
css
images
js
favicon.ico
index.html
manifest.json
src
components
about
contactus
footer
header
portfolio
resume
testimonials
App.css
App.js
App.test.js
index.css
index.js
logo.svg
serviceWorker.js
.gitignore
package.json
README.md
README.old.md
yarn.lock
OUTLINE
MAVEN PROJECTS
master*
0 0
vandal777
Live Share
Jest:

5. Una vez tengamos la estructura de carpetas vamos a crear los diferentes componentes a base de cortarlos de la estructura del Template.

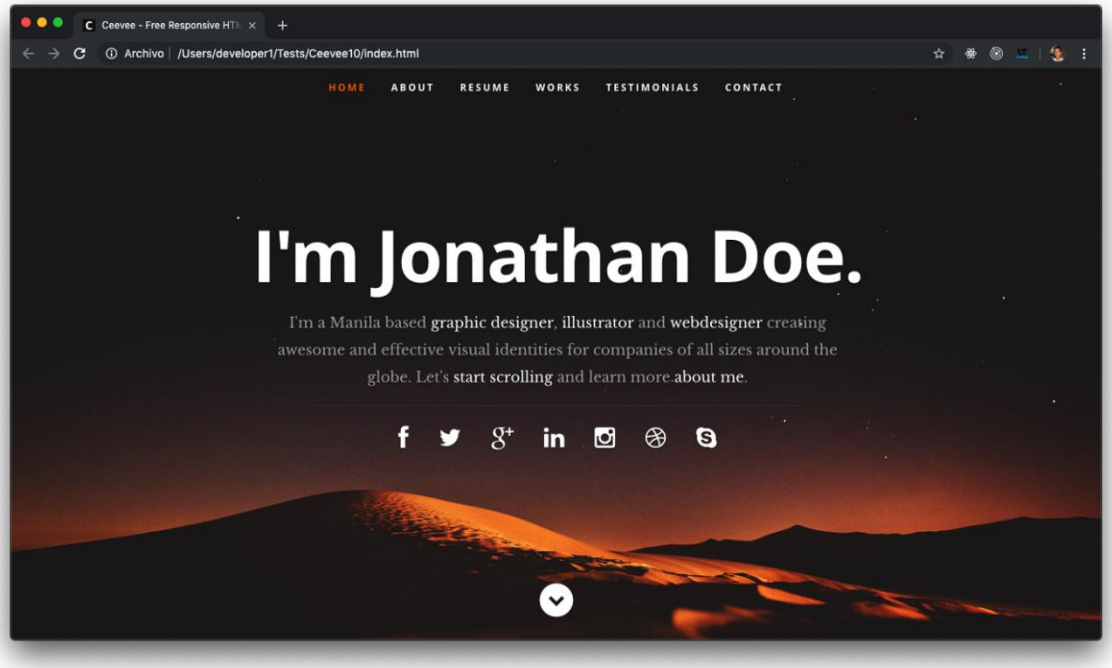
Os voy a poner un ejemplo para que os quede más claro:

El Template es una página estática HTML la cual toda la estructura de la propia página esta creada en una sola, en este caso seria el archivo **index.html** que esta dentro de la carpeta raíz del Template

/Ceevee10/index.html

Si abrimos esta página con cualquier editor de texto veremos que esta formada, por todas las partes de la propia página.

Para poder ver el Template, solo hace falta abrir ese mismo **index.html** con un explorador web, como seria Google Chrome y esto seria lo que veríamos:



Index.html del Template

Entonces para conseguir la separación por componentes que nos ofrece ReactJS vamos a empezar por partes.

Yo voy a hacerlo solo para el primer componente que es el **Header**, pero vosotros tenéis que hacerlo con todos los componentes, que serian estos:

header, about, resume, portfolio, testimonials, contactus, footer

El componente que vamos a coger es el **Header**, ya que es primero que vemos en la pantalla del Template.

Header del template

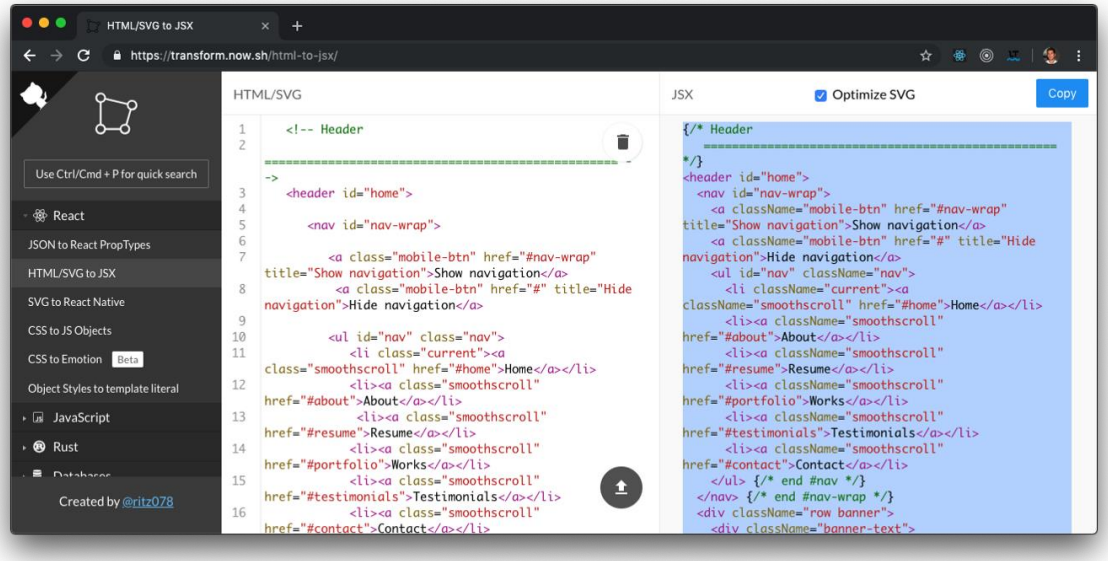
Para conseguir extraer el **Header** abriremos al archivo **index.html** del Template con nuestro editor de texto y buscaremos dentro del **body** estas 2 etiquetas, que delimitan el inicio y el fin de una sección, que será lo que nosotros convertiremos en componentes:

```
<body><!-- Header
   ================================================ -->
   <header id="home">   <---Copy from this section
   ...
   ...
   </header> <!-- Header End -->...
</body>
```

Una vez copiado todo ese texto desde el inicio del **Header** hasta su final, lo vamos a pegar en una web que nos convierte código **HTML** en código **JS.**

https://transform.now.sh/html-to-jsx/

HTML/SVG to JSX
An online REPL for converting HTML to JSX with proper support for SVG.
transform.now.sh



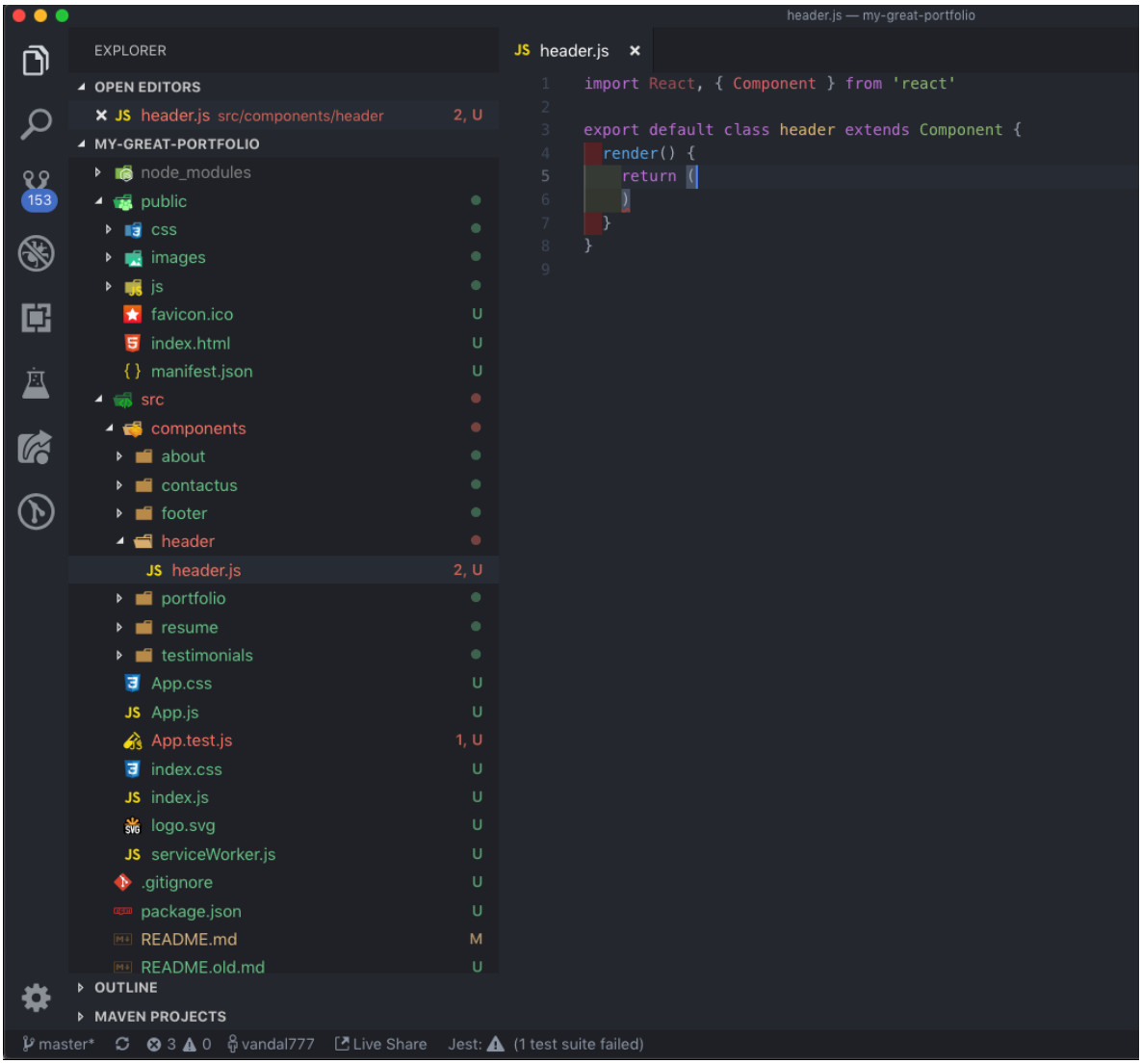
ahora nos vamos al directorio que hemos creado para nuestro header:

my-great-portfolio/src/header

Ahí dentro creamos un archivo llamado **header.js** y le añadimos el siguiente contenido:

```
import React, { Component } from 'react'export default class
header extends Component {
  render() {
    return (
    )
  }
}
```

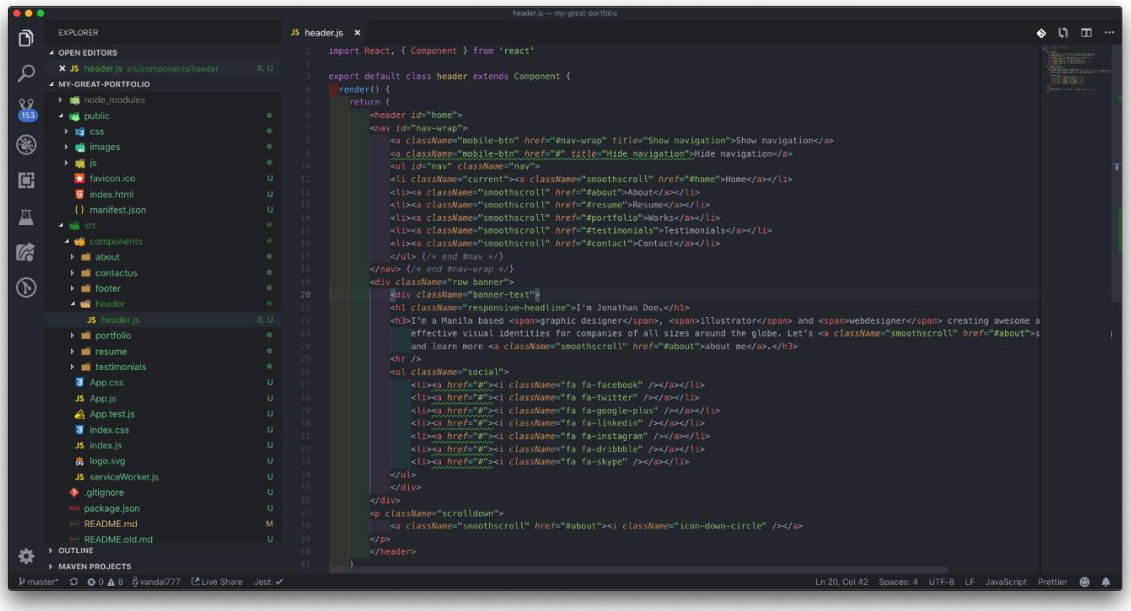
Quedando el archivo de esta manera:



header.js

Ahora añadimos dentro de los paréntesis del **return** el resultado que nos ha devuelto la web anterior al convertir el código HTML a JSX.

Como resultado tendremos nuestro componente de React llamado **header.js**



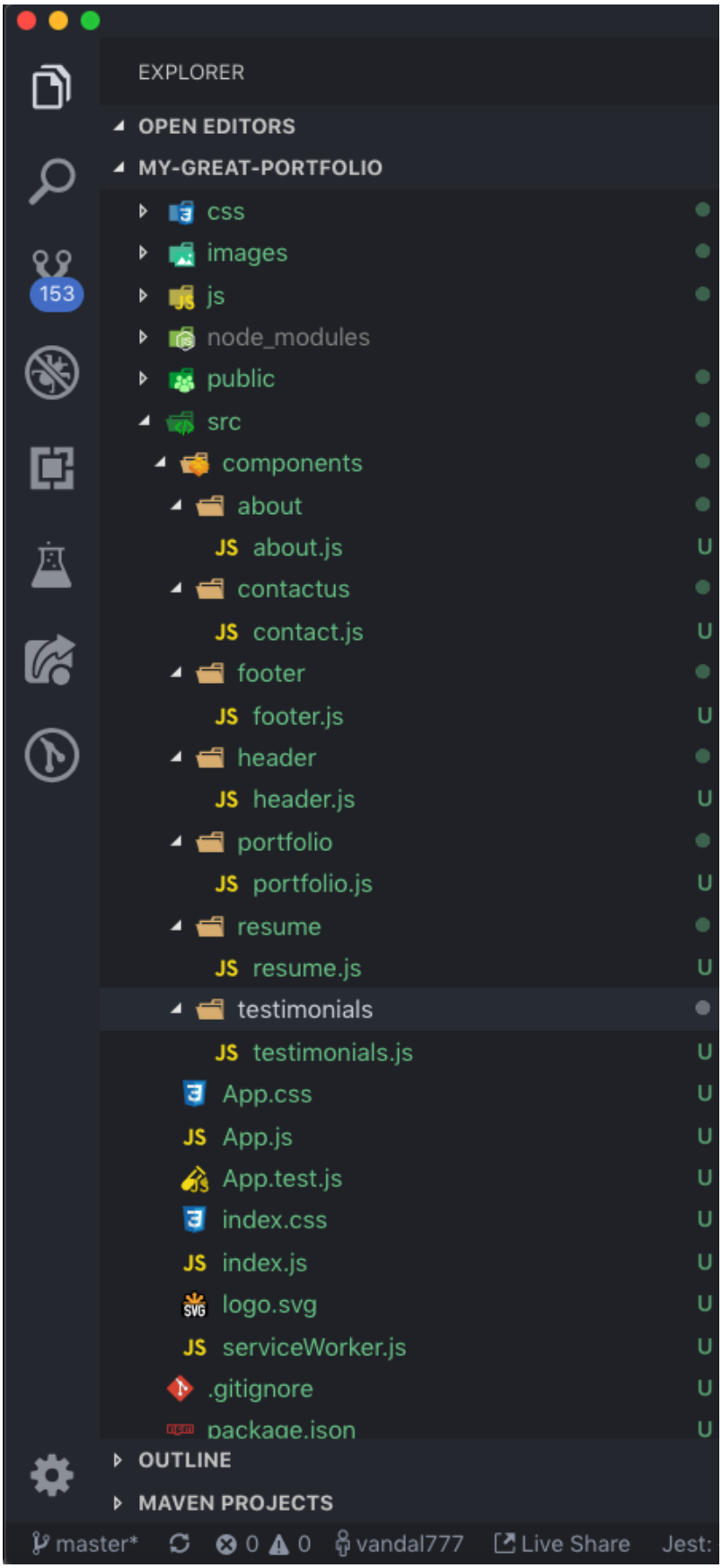
header.js

Ahora repetimos estos mismos pasos para todos los componentes. Yo como soy buena persona y esto ya lo tengo hecho, podéis acceder a mi repositorio y copiar todos los archivos que necesitáis.

https://github.com/vandal777/my-great-portfolio

vandal777/my-great-portfolio
My great portfolio created with ReactJs. Contribute to vandal777/my-great-portfolio development by creating an account…
github.com

Una vez creados todos los archivos, tenemos que tener esta estructura de archivos como esta:


EXPLORER
OPEN EDITORS
MY-GREAT-PORTFOLIO
css
images
js
node_modules
public
src
components
about
about.js U
contactus
contact.js U
footer
footer.js U
header
header.js U
portfolio
portfolio.js U
resume
resume.js U
testimonials
testimonials.js U
App.css U
App.js U
App.test.js U
index.css U
index.js U
logo.svg U
serviceWorker.js U
.gitignore U
package.json U
OUTLINE
MAVEN PROJECTS
master*
0 0
vandal777
Live Share
Jest:

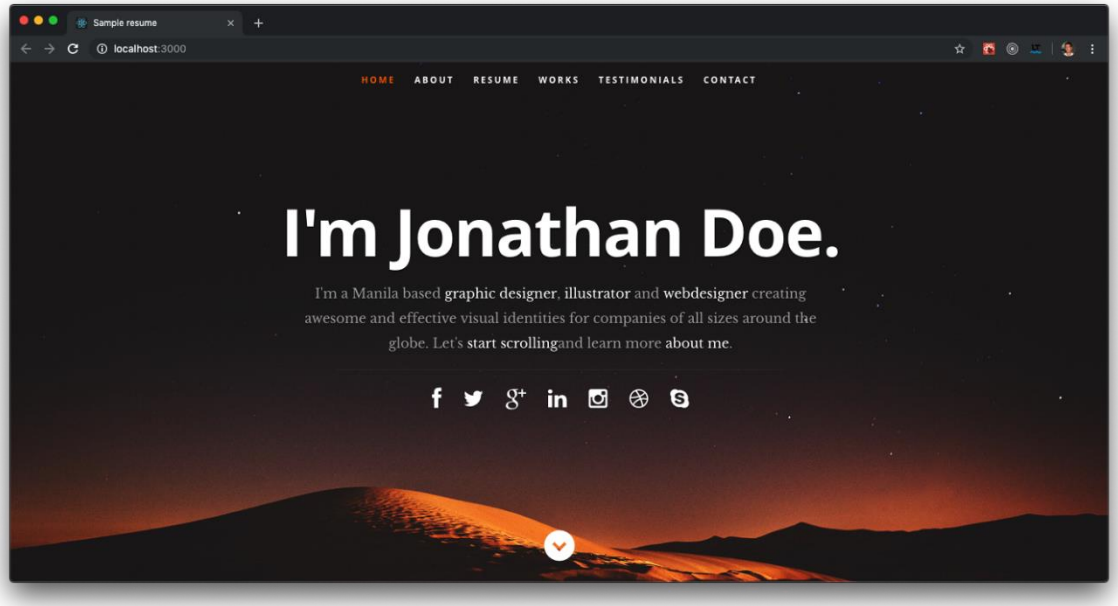
Estructura de archivos final del proyecto

6) Lo siguiente que haremos sera en el archivo **my-great-portfolio/src/App.js** sustituir el código actual por el siguiente codigo:

```
import React, { Component } from 'react';
import Header from './components/header/header';
import About from './components/about/about';
import Resume from './components/resume/resume';
import Portfolio from './components/portfolio/portfolio';
import Testimonials from
'./components/testimonials/testimonials';
import ContactUs from './components/contactus/contactus';
import Footer from './components/footer/footer';
class App extends Component {
  render() {
    return (
      <div className="App">
        <Header />
        <About />
        <Resume />
        <Portfolio />
        <Testimonials />
        <ContactUs />
        <Footer />
      </div>
    );
  }
}export default App;
```

Una vez hecho esto, ya podemos arrancar nuestra aplicación con:

```
npm start
```



Nuestro proyecto con ReactJS porfin !! :D !!

Y ya podemos ver nuestra aplicación corriendo

5. Deploy de nuestra app a GitHub Pages

Por ultimo falta lo más importante... Hacernos visibles al resto del mundo. Para ello vamos a utilizar Github Pages. En el principio del articulo ya explique lo que era, pero en resumen es un alojamiento web gratis que nos ofrece GitHub.

1. Para poder hacer el deploy de nuestra app en github pages, primero necesitamos tener instalada la librería, a si que la instalamos con:

```
npm install --save-dev gh-pages
```

2. Una vez instalada la librería, vamos a modificar nuestro archivo **package.json** que se encuentra en la raíz del proyecto y

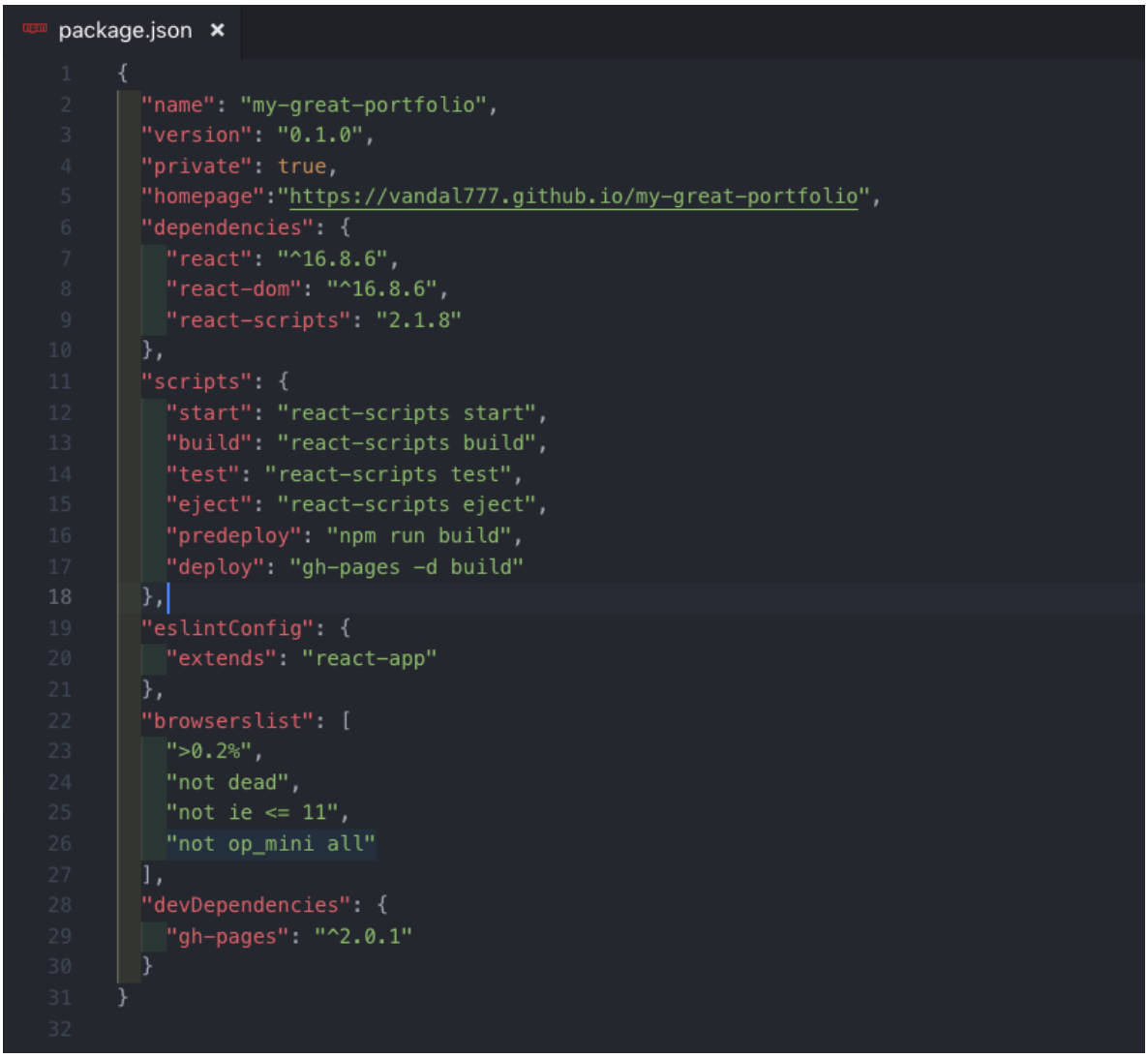
añadimos esta línea dentro del archivo. Claramente esta línea será diferente, dependiendo de vuestro nombre de usuario y el nombre de vuestro repositorio.

```
 "homepage":"https://vandal777.github.io/my-great-portfolio",
```

3. Ahora una vez hecho esto, tenemos que añadir un par de lineas más dentro del mismo archivo package.json dentro de “scripts” con el siguiente contenido:

```
"predeploy": "npm run build",
"deploy": "gh-pages -d build"
```

Quedando finalmente nuestro archivo package.json de esta manera:



package.json

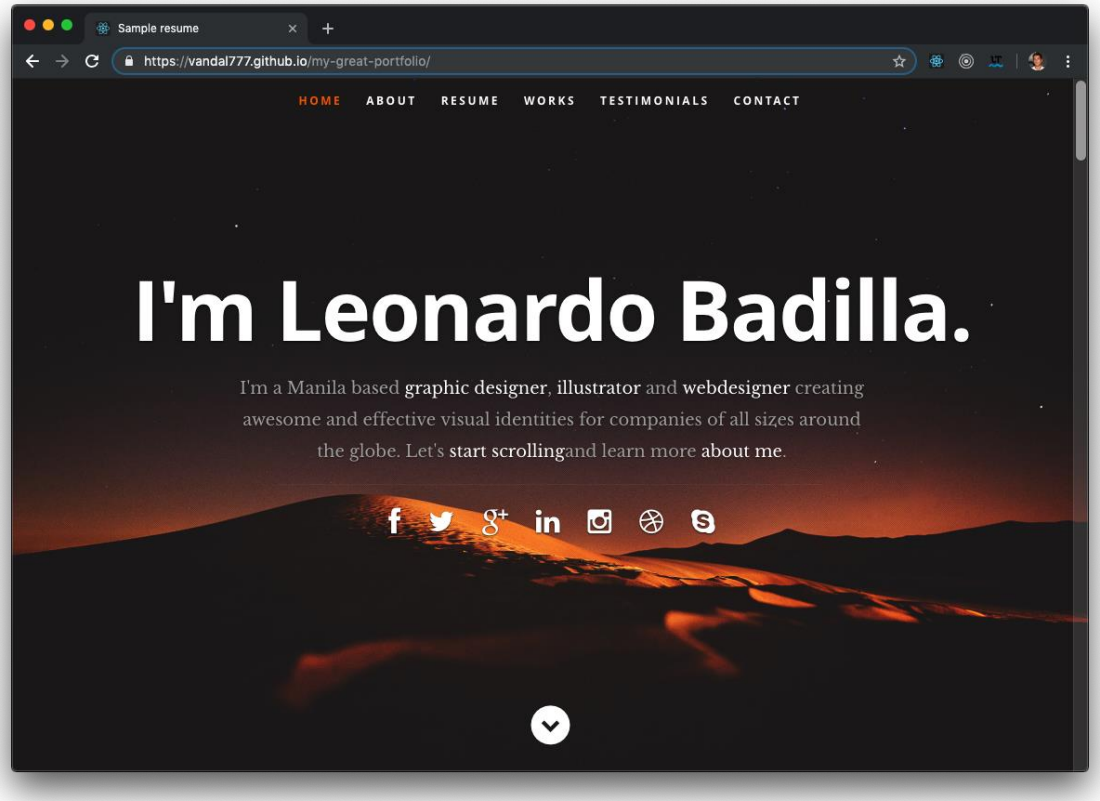
4. Finalmente ya lo tenemos todo para poder publicar nuestra aplicación. Solo tenemos que ejecutar el siguiente comando.

```
npm run deploy
```

Una vez hecho esto, nuestra aplicación se subirá automáticamente a la URL

https://vandal777.github.io/my-great-portfolio/

La cual alojara nuestro genial Portfolio.



Resultado final de nuestro portfolio y nuestro index.html