

Creating First React App

System Setup

Before really starting with React, we first need to make sure that the basic system setup is available. basic system nodejs ,npm(with npx) react

The first prerequisite which needs to be installed on the development system is **Node.js** and **NPM** (Node.js Package Manager). **NPM** comes check nodejs suceesfully installed or not bundled with **Node.js**, so you only need to make sure that you have node --version , npm -v , npx -v Node.js **installed**.

You can quickly check if an **up-to-date** version of Node.js is already installed on your system by executing the following command:

```
$ node --version
```

If not, you can go to the Node.js website at <https://nodejs.org/> and **download** the installer which is needed for your specific operating system.

In addition, you should install a **code editor** for React development. My recommendation here is to use **Visual Studio Code**, which can be downloaded for free at <https://code.visualstudio.com>. Visual Studio Code is an **all-in-one** solution that provides you with a code editor, an extensible plugin system, and an integrated terminal.

Creating A First React Project From Scratch

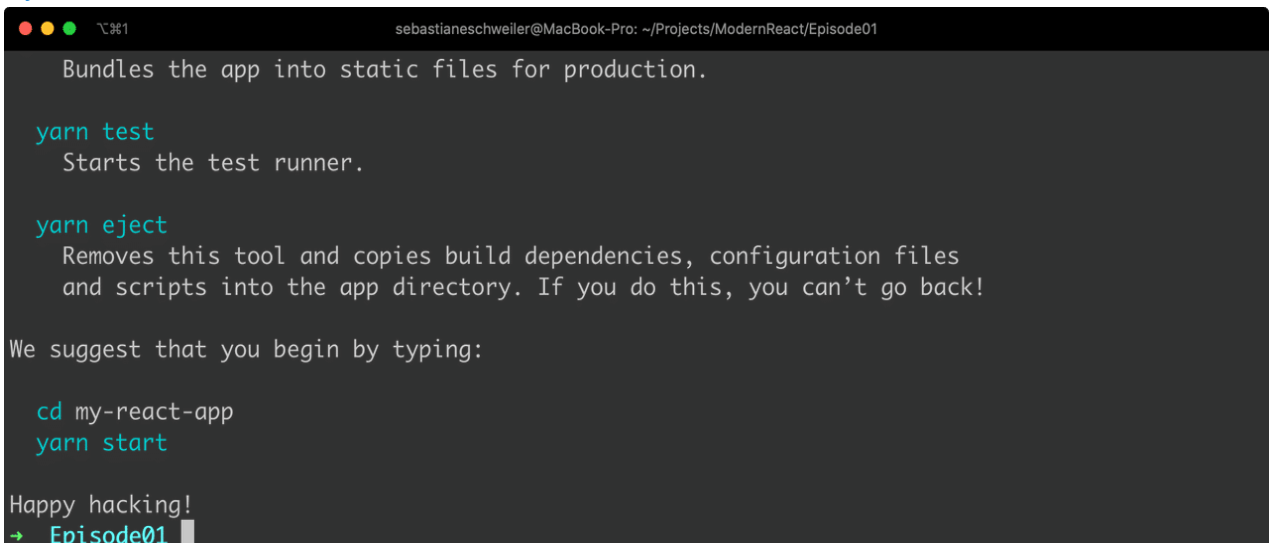
One of the **easiest** ways of setting up a new **React** project is to use the **create-react-app** script. To execute this **script**, we're able to simply use the **npx** command, which makes it possible to execute the script without prior downloading the package in a separate step:

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

`npx create-react-app myfirstreactapp`
following command for create react app

As a command-line parameter, we need to pass over the name of the new **project folder**, e.g., **my-react-app**. After having run the command, you should see something similar to:

`cd myfirstreactapp`
`yarn start`

A terminal window screenshot showing the output of the 'create-react-app' command. The window title is 'sebastianeschweiler@MacBook-Pro: ~/Projects/ModernReact/Episode01'. The output text is: 'Bundles the app into static files for production.' followed by 'yarn test' (Starts the test runner.), 'yarn eject' (Removes this tool and copies build dependencies, configuration files and scripts into the app directory. If you do this, you can't go back!), 'We suggest that you begin by typing:', 'cd my-react-app', 'yarn start', 'Happy hacking!', and '→ Episode01' with a cursor.

```
sebastianeschweiler@MacBook-Pro: ~/Projects/ModernReact/Episode01
Bundles the app into static files for production.

yarn test
  Starts the test runner.

yarn eject
  Removes this tool and copies build dependencies, configuration files
  and scripts into the app directory. If you do this, you can't go back!

We suggest that you begin by typing:

  cd my-react-app
  yarn start

Happy hacking!
→ Episode01
```

Now we're ready to enter the newly created project folder by typing in:

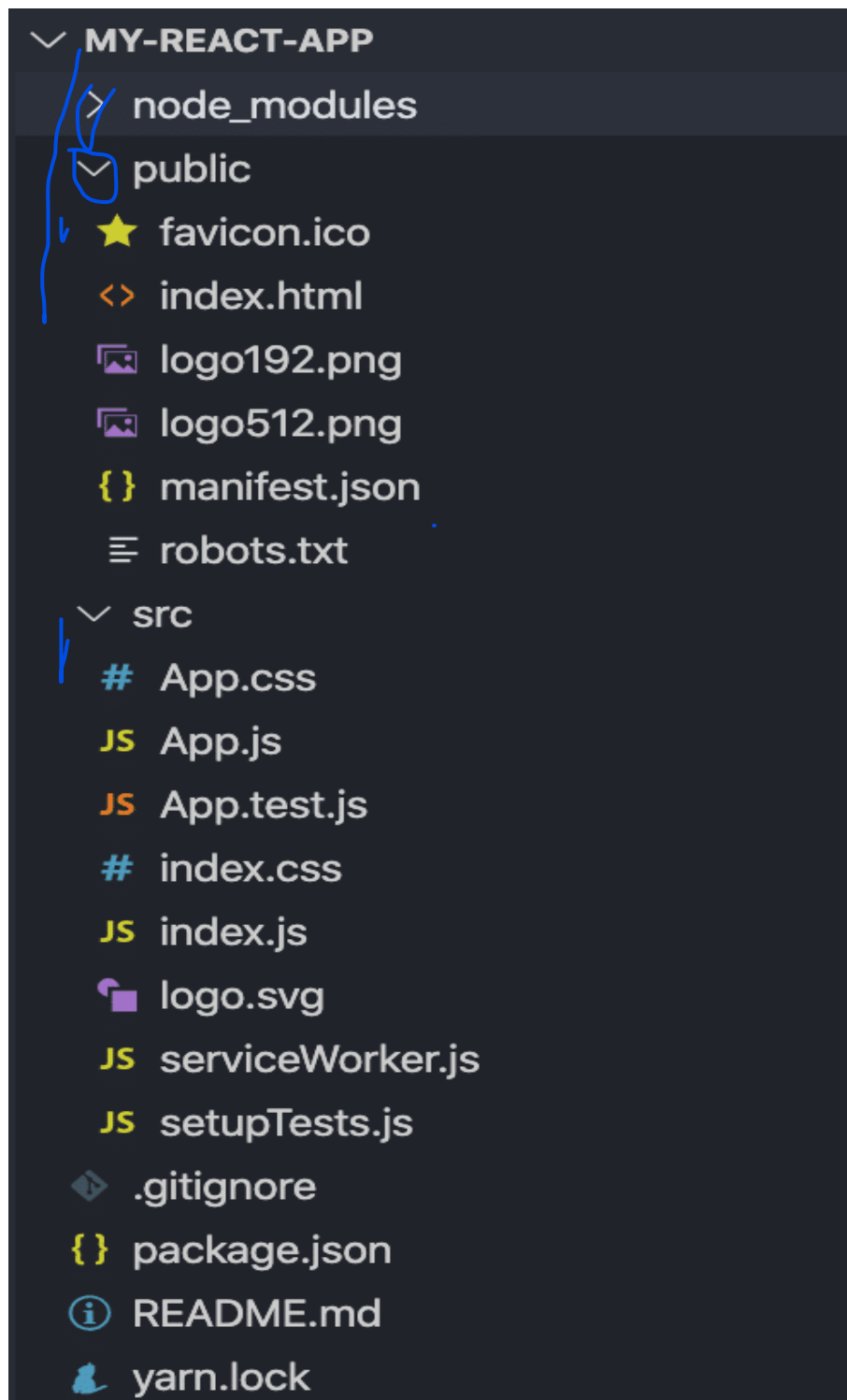
\$ cd my-react-app

And from within the project folder, start Visual Studio Code via:

\$ code .

Exploring The Initial Project Structure

Now that we've opened the new project folder in **Visual Studio Code**, we're ready to start exploring the initial project structure:



- **node_modules/**: Folder which contains all project dependencies, e.g., packages that have been downloaded and installed by using the Node.js Package Manager (NPM).
- **Public/**: Folder contains static assets of the web application like **index.html**.
- **Src/**: This is the folder where you can find the **JavaScript implementation** of your React application. E.g., by default, you can find the App component implementation in **App.js**. The corresponding initial unit test case implementations can be found in **App.test.js**, and **index.js** contains the code, which is the starting point of your React application.

Taking A Look Into Package.json

Now that you have a first overview of the project structure, we need to take a look at the **package.json** file to get an overview of defined scripts. Scripts can be executed via **npm**, e.g., to start the **development web server**, build for **production**, etc.

The scripts section of the initial package.json file looks like the following:

```
"scripts": {  
  "start": "react-scripts start",  
  "build": "react-scripts build",  
  "test": "react-scripts test",  
  "eject": "react-scripts eject"  
},
```

Running The **Development Web Server**

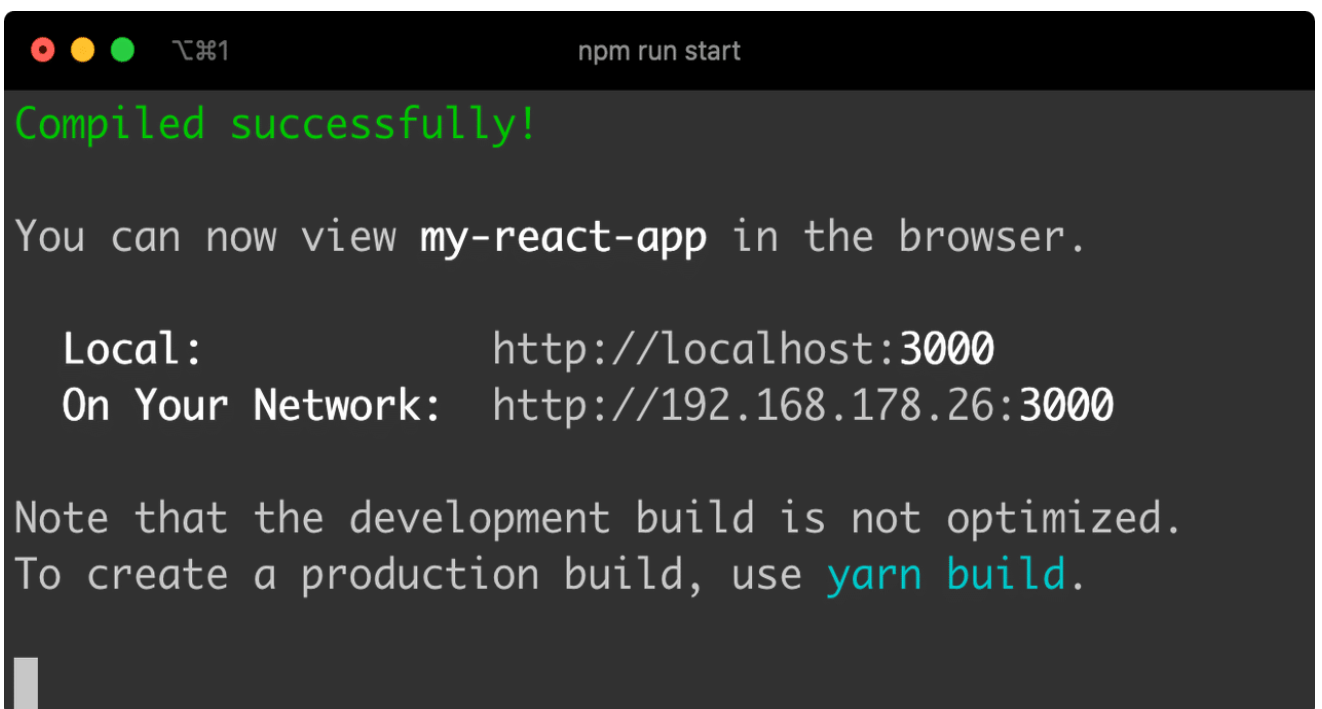
Running the development web server is done by **executing** the start **script** via:

\$ npm run start

or

\$ npm start

This should give you the following output:



```
npm run start

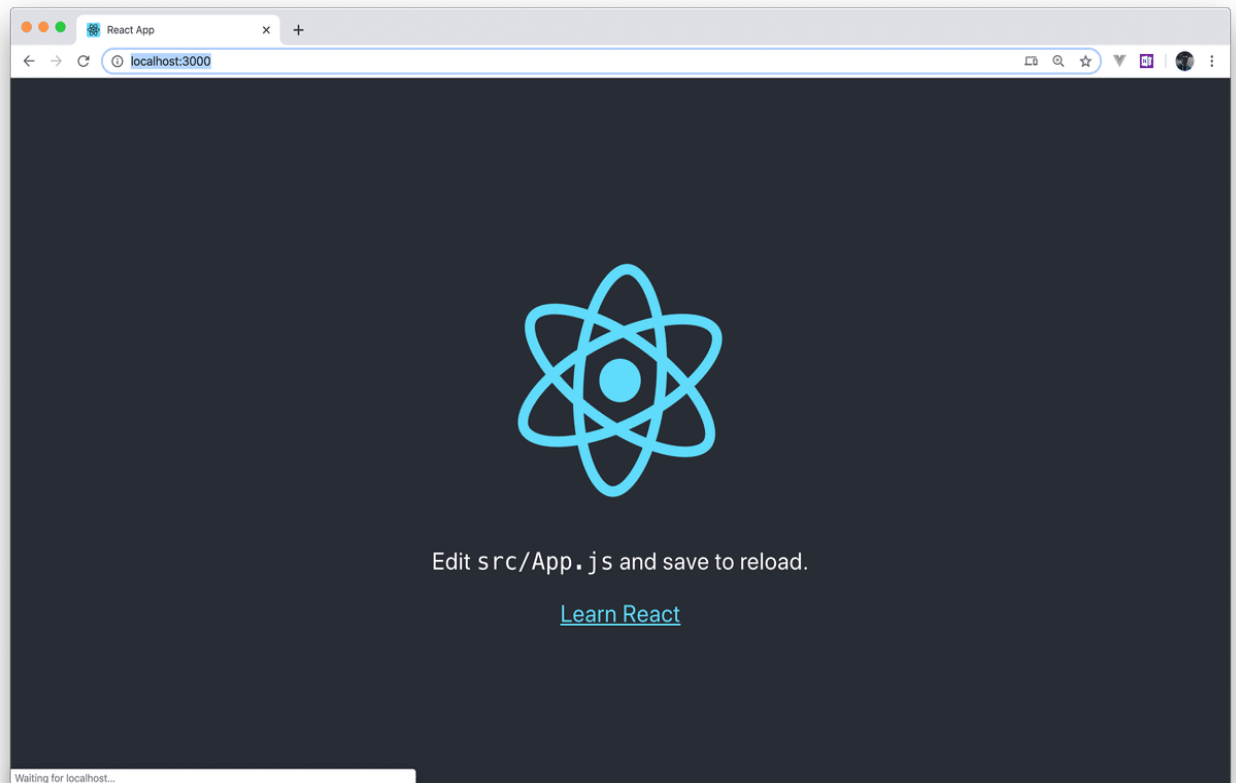
Compiled successfully!

You can now view my-react-app in the browser.

Local:            http://localhost:3000
On Your Network:  http://192.168.178.26:3000

Note that the development build is not optimized.
To create a production build, use yarn build.
```

Here you can see that the development web server has been started on **port 3000**. The browser is opened automatically, and **URL `http://localhost:3000`** has opened automatically so that you now should be able to see the following output:



(Congratulations, You have successfully Made your First React App).