# Set Up a React Development Environment: Exercise 3

You will need Node.js and npm.

### Create a new Project

* 1. Open a command prompt.
  2. Create a React project using create-react-app:

npx create-react-app react-dev-env

1. The directory this creates, react-dev-env, will be referred to as the project root. This process will take some minutes to complete.
2. Go into the project root directory that was just created with cd
   1. Open the package.json file in an editor.
   2. Look at available scripts:

"scripts": {

"start": "react-scripts start", "build": "react-scripts build", "test": "react-scripts test", "eject": "react-scripts eject"

},

}

* + 1. The 'start' script runs the development server.
    2. The 'build' script builds the app for production and places it in the 'build' folder.
    3. The 'test' script launches the test runner.
    4. The 'eject' script externalizes the project's configuration (a one-way process).

1. Look in the project's "\src" directory:

App.css // css styles to go with the main application

App.js // the main application file

App.test.js // unit test spec for the main application index.css // global css styles loaded in index.js index.js // loads the App component

logo.svg // logo used by the default App component reportWebVitals.js //

1. Take a look in the project's "\public" directory:

favicon.ico // applicatin favicon

index.html // application entry point html file logo192.png // image used by default App component logo512.png // image used by default App component manifest.json // app info for mobile device homescreens robots.txt // tells search engine which files to crawl

Run the development server:

1. npm start
2. The command will also open a browser tab pointing to the application at:

http://localhost:3000/

1. Let’s see what makes the default app.
2. Open src\index.js in an editor. You should see the following:

ReactDOM.render(

<React.StrictMode>

<App />

</React.StrictMode>, document.getElementById('root')

);

1. The above code loads the App component into this div:

<div id="root"></div>

1. Which appears in the public\index.html file.
2. Close the index.js file.
3. Open the src\App.js file in an editor. You should see the following:

import logo from './logo.svg'; import './App.css';

function App() { return (

<div className="App">

<header className="App-header">

<img src={logo} className="App-logo" alt="logo" />

<p>

Edit <code>src/App.js</code> and save to reload.

</p>

<a

* + 1. className="App-link" href="https://reactjs.org" target="\_blank" rel="noopener noreferrer"

>

Learn React

</a>

</header>

</div>

);

}

export default App;

1. The above code is what produces the default output we see in the browser. We are going to replace it with our own implementation.
2. Replace the 'App()' function in src\App.js with the following:

function App() { return (

<div id='main' >

* 1. <h1>My App</h1>

</div>

);

}

1. Remove the following import (at the top of App.js):

import logo from './logo.svg';

1. Save the App.js file
2. Open src\App.css in an editor.
3. Delete the entire contents of App.css and replace it with the following:

h1{

font-family: sans-serif;

}

#main{

padding: 15px;

}

1. Save the App.css file.
2. Look at the command prompt where you started the development server. The development server process takes care of re-compiling and re-deploying the updated application. You should see the following output:

Compiled Successfully!

You can now view react-dev-env in the browser.

Local: http://localhost:3000

1. Note that the development build is not optimized. To create a production build, use yarn build.
2. Go back to the browser. The application should have updated