



Beginning Frontend Development with React

Lesson 1

Introducing React and UI Design

Lesson Objectives

In this lesson, we will cover the following topics:

- Describe what React is and where it fits in the development of your applications
- Set up the infrastructure of a React-based application
- Design the UI of your application, optimizing it for use in React

Topic A: What is React?

- React is a JavaScript library for building composable user interfaces
- A *component* is an element that contributes to building the user interface
- React components have the ability to present data that changes over time
- React implements the *View* part of the most common presentational design patterns

Topic B: How to Set up a React-Based Application

Setting up a modern JavaScript development environment requires the installation and configuration of a few tools:

- Package managers
- Transpilers
- Module bundlers
- Task runners
- Scaffolding tools
- Test runners



Installing create-react-app

create-react-app is a command-line interface (CLI) that allows us to set up a React-based application without needing to configure any of the aforementioned tools:

```
npm install -g create-react-app
```

```
create-react-app --version
```

Creating Your First React Application

Set up all the stuff for a React-based application named **hello-react**:

```
create-react-app hello-react
```

Activity A: Creating an Application with create-react-app

Aim

The aim of the activity is to start becoming familiar with create-react-app and the content it creates.

Scenario

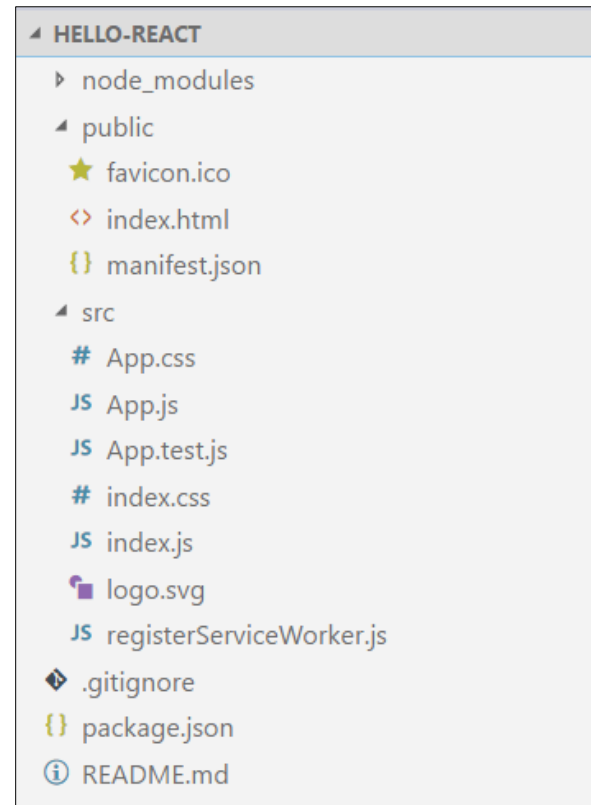
We need to set up a development environment in order to create a product catalog application built with React.

Step for Completion

Use **create-react-app** to create the development environment, and give the name **my-shop** to the sample application.

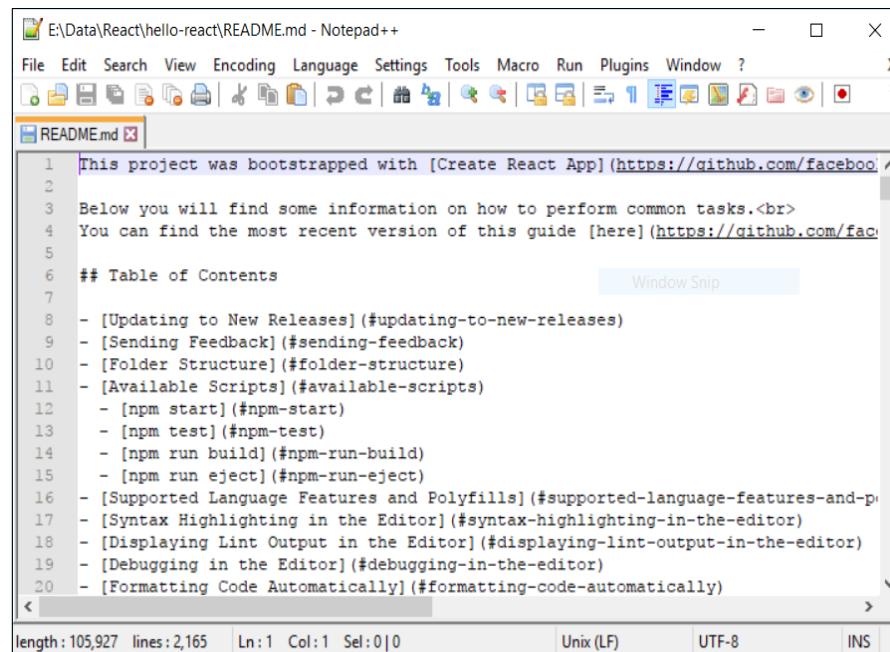
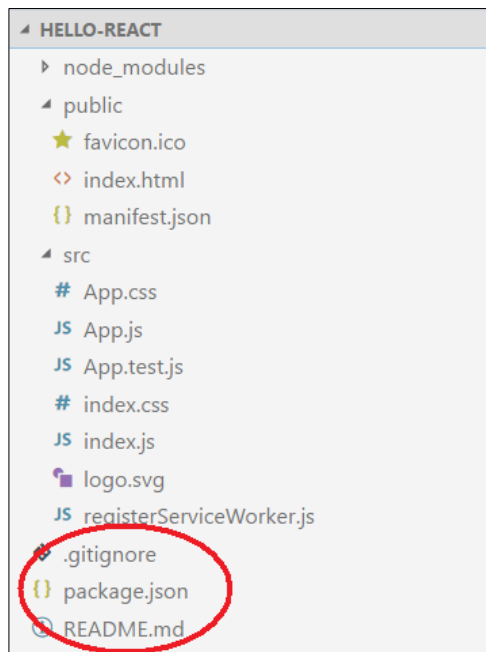
Exploring the Generated Content

Let's take a look at the files generated by create-react-app, so that we can get an understanding of the structure of a React-based application.



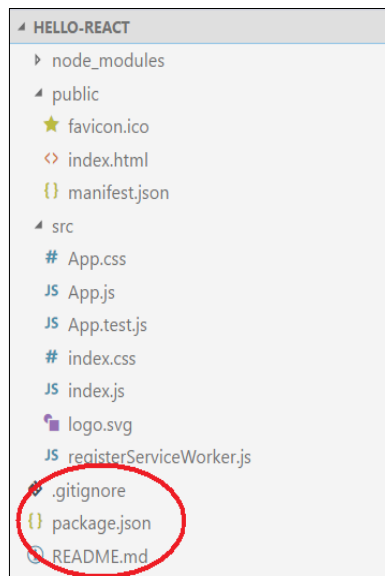
Exploring the Generated Content

In the root folder, we can see a **README.md** file, the **package.json** file, and the **.gitignore** file:



Exploring the Generated Content

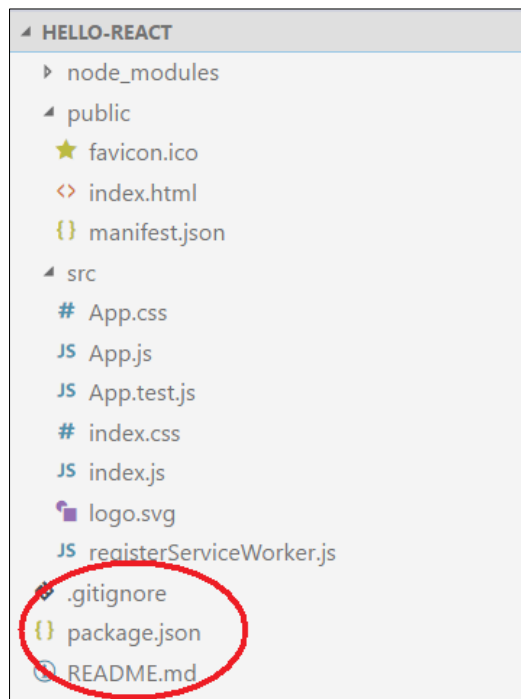
The **package.json** file contains information about the project, such as the name, the version, and so on, and references to all the npm packages used by the current project



```
{
  "name": "hello-react",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "react": "^16.0.0",
    "react-dom": "^16.0.0",
    "react-scripts": "1.0.14"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test --env=jsdom",
    "eject": "react-scripts eject"
  }
}
```

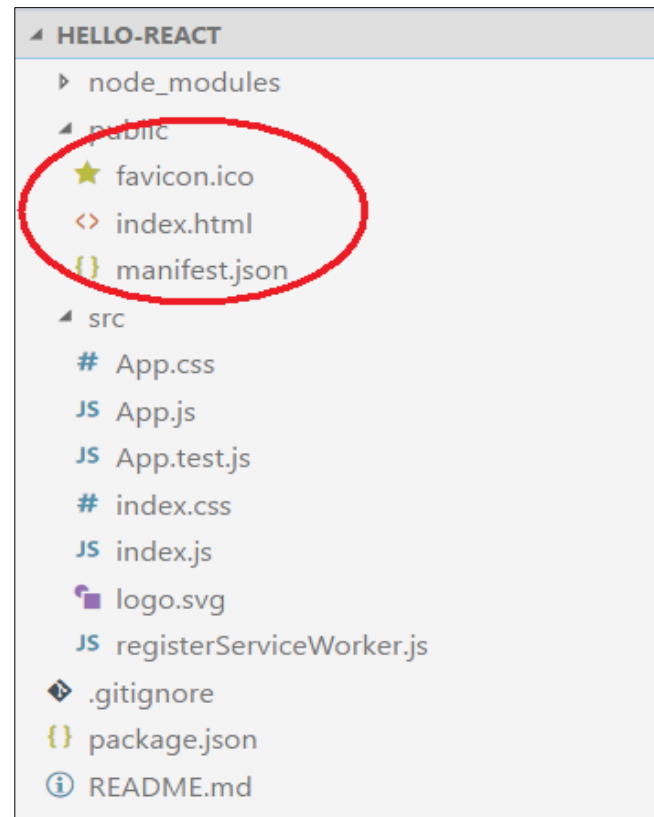
Exploring the Generated Content

Nowadays, it is essential to have a project under version control:



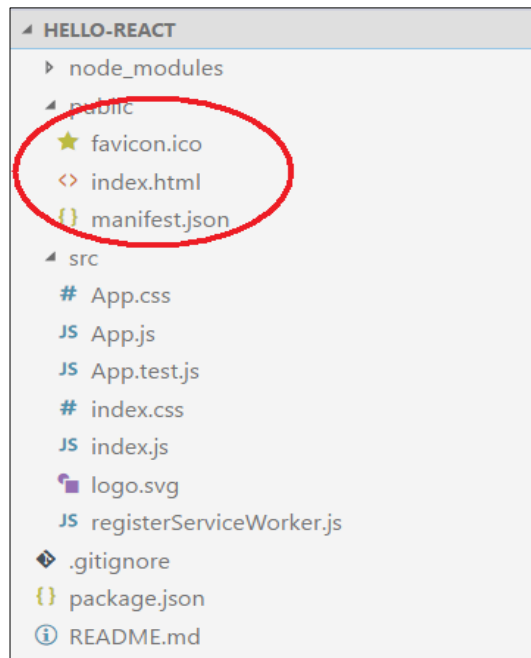
Exploring the Generated Content

These are the static parts of our application



Exploring the Generated Content

- Keras has two methods for evaluating a model: either using parameters during training:

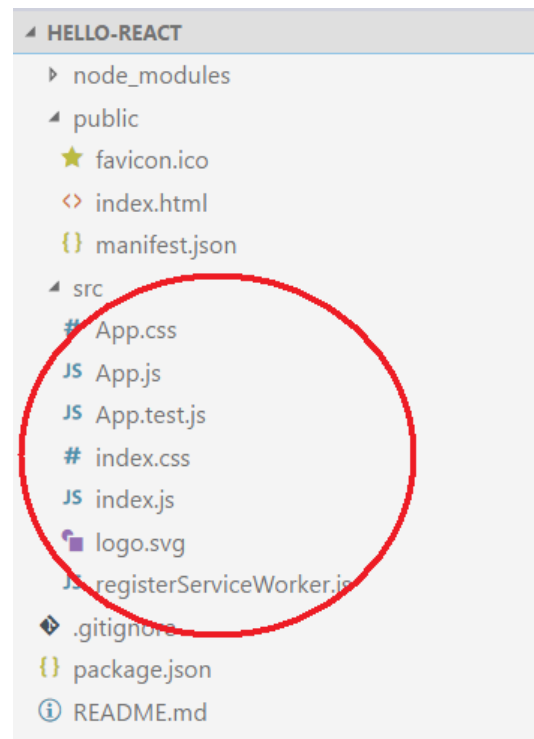


index.html

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1, shrink-to-fit=no">
    <meta name="theme-color" content="#000000">
    <link rel="manifest"
                                href="%PUBLIC_URL%/manifest.json">
    <link rel="shortcut icon"
                                href="%PUBLIC_URL%/favicon.ico">
    <title>React App</title>
  </head>
  <body>
    <noscript>
      You need to enable JavaScript to run this app.
    </noscript>
    <div id="root"></div>
  </body>
</html>
```

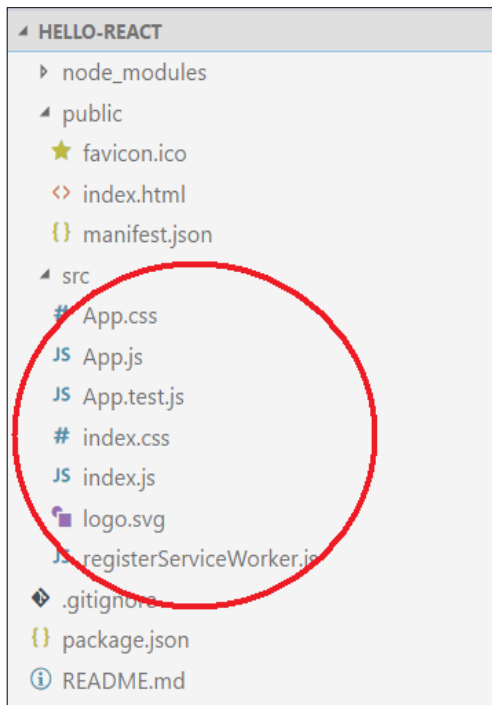
Exploring the Generated Content

The source folder contains the basic files, with the code that we can modify for our purposes:



Exploring the Generated Content

Let's understand the code of the **index.js** file:

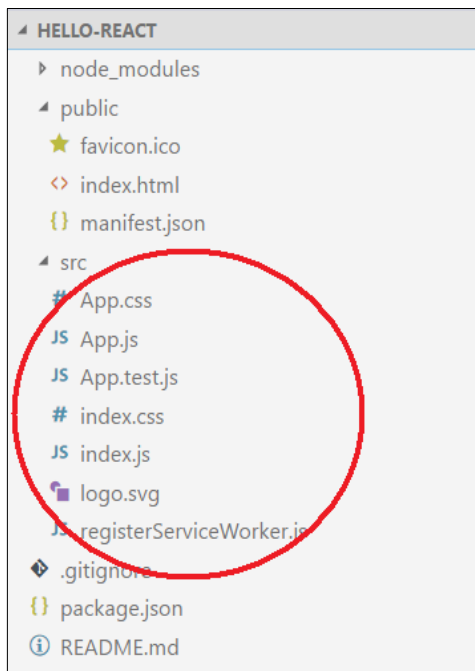


index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
import registerServiceWorker from './registerServiceWorker';

ReactDOM.render(<App />, document.getElementById('root'));
registerServiceWorker();
```


Exploring the Generated Content



App.js

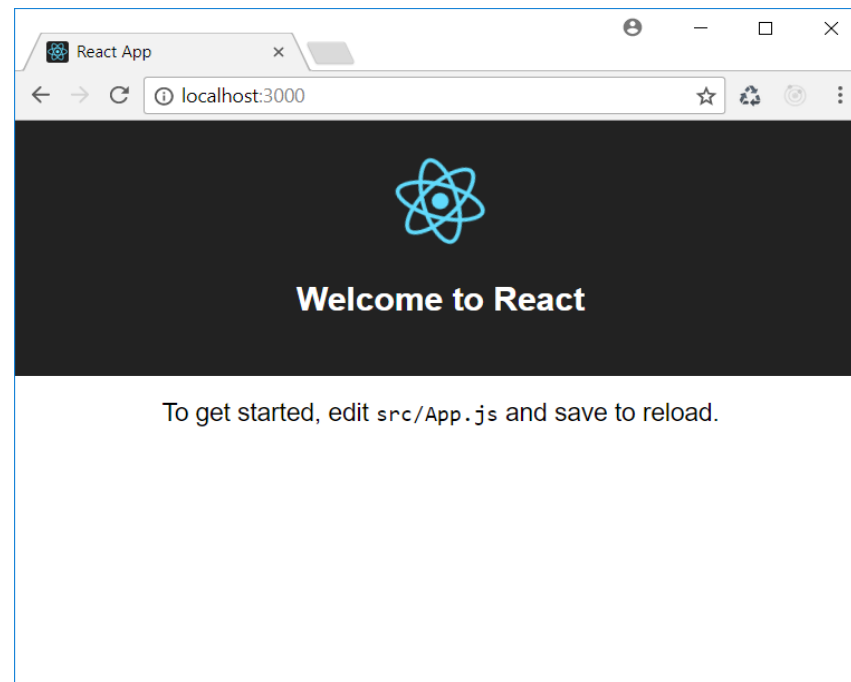
```
import React, { Component } from 'react';
import logo from './logo.svg';
import './App.css';

class App extends Component {
  render() {
    return (
      <div className="App">
        <header className="App-header">
          <img src={logo} className="App-logo" alt="logo" />
          <h1 className="App-title">Welcome to React</h1>
        </header>
        <p className="App-intro">
          To get started, edit <code>src/App.js</code> and save to reload.
        </p>
      </div>
    );
  }
}

export default App;
```

Topic B: How to Set up a React-based Application

```
npm start
```



Activity B: Starting and Changing the Application

Aim

The aim of the activity is to become familiar with launching an application and appreciating the hot reloading feature.

Scenario

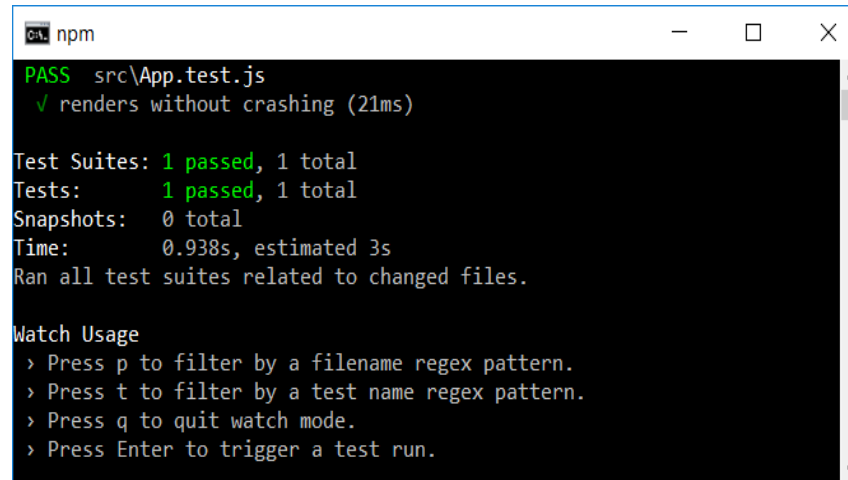
We want to change the title of the application that was created in the previous activity.

Step for Completion

Start the application so that you can see it in a browser, then edit the `App.js` file and set the title to My Shop.

Activity B: Starting and Changing the Application

npm test

A screenshot of a terminal window titled 'npm'. The window shows the output of the 'npm test' command. The output indicates that the test suite 'src\App.test.js' passed, specifically the test 'renders without crashing' which took 21ms. It also shows summary statistics: 1 test suite passed out of 1 total, 1 test passed out of 1 total, and 0 snapshots. The total time taken was 0.938s, with an estimated 3s for future runs. The terminal also displays 'Watch Usage' instructions, including pressing 'p' for filename regex, 't' for test name regex, 'q' to quit watch mode, and 'Enter' to trigger a test run.

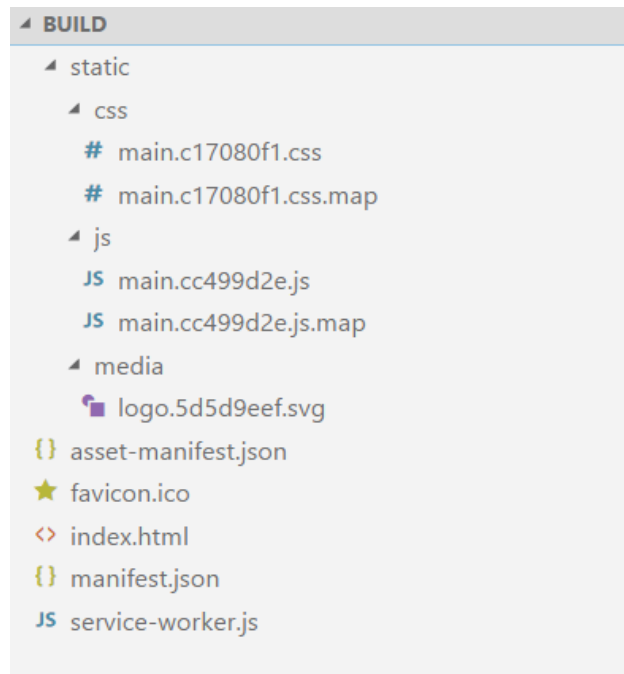
```
npm
PASS src\App.test.js
  ✓ renders without crashing (21ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        0.938s, estimated 3s
Ran all test suites related to changed files.

Watch Usage
> Press p to filter by a filename regex pattern.
> Press t to filter by a test name regex pattern.
> Press q to quit watch mode.
> Press Enter to trigger a test run.
```

Activity B: Starting and Changing the Application

```
npm run build
```



Activity B: Starting and Changing the Application

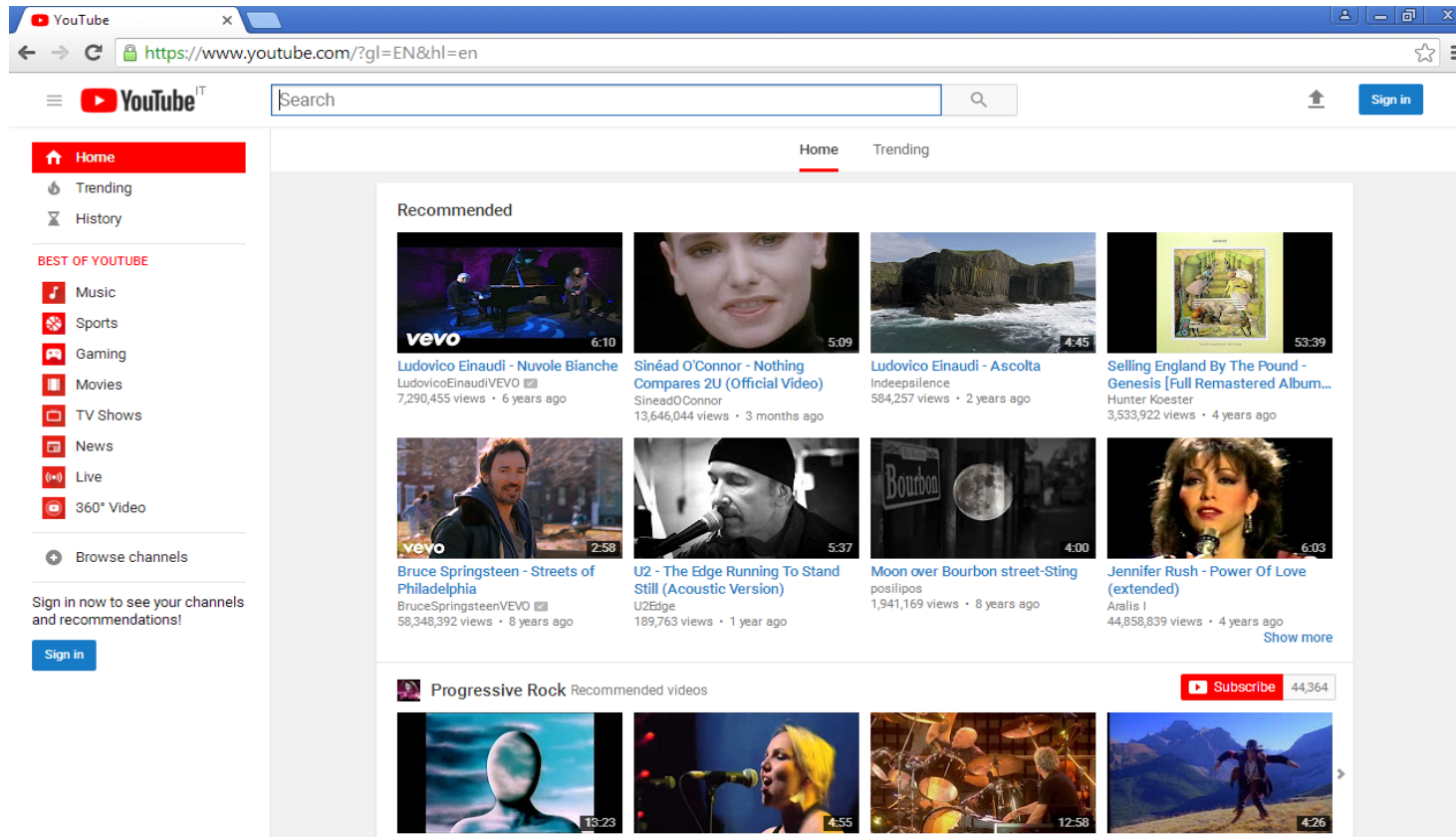
```
npm run eject
```

- It put our application out of the CLI context
- It gives us the power and the responsibility to manage it
- This is a one-way process

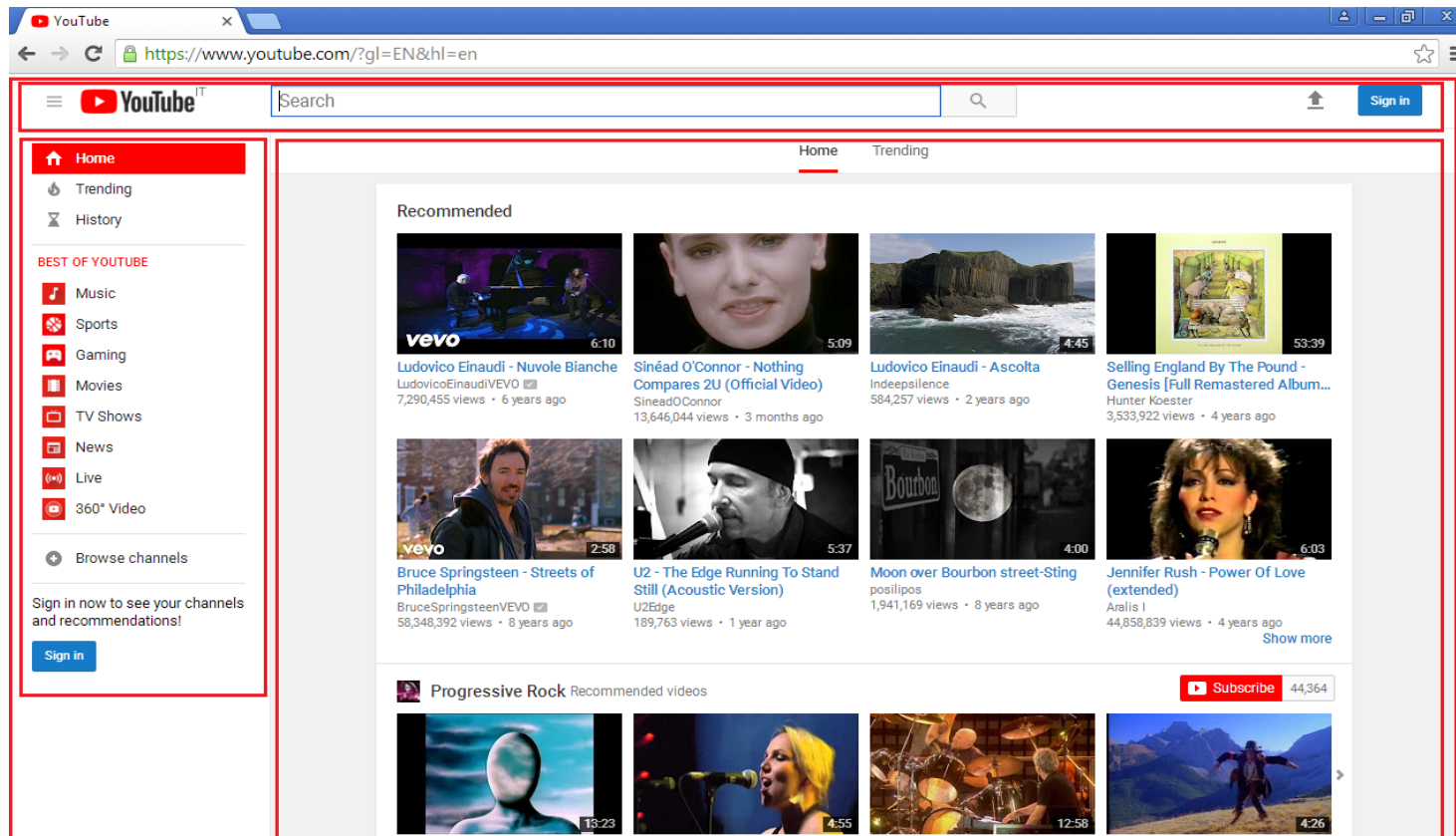
Topic C: How to Design a UI

- A user interface is an aggregate of components
- The whole React application is an aggregate of components
- From a design point of view, we can say that a **component** is a part of the user interface, having a specific role
- A hierarchy of components is often called a **component tree**

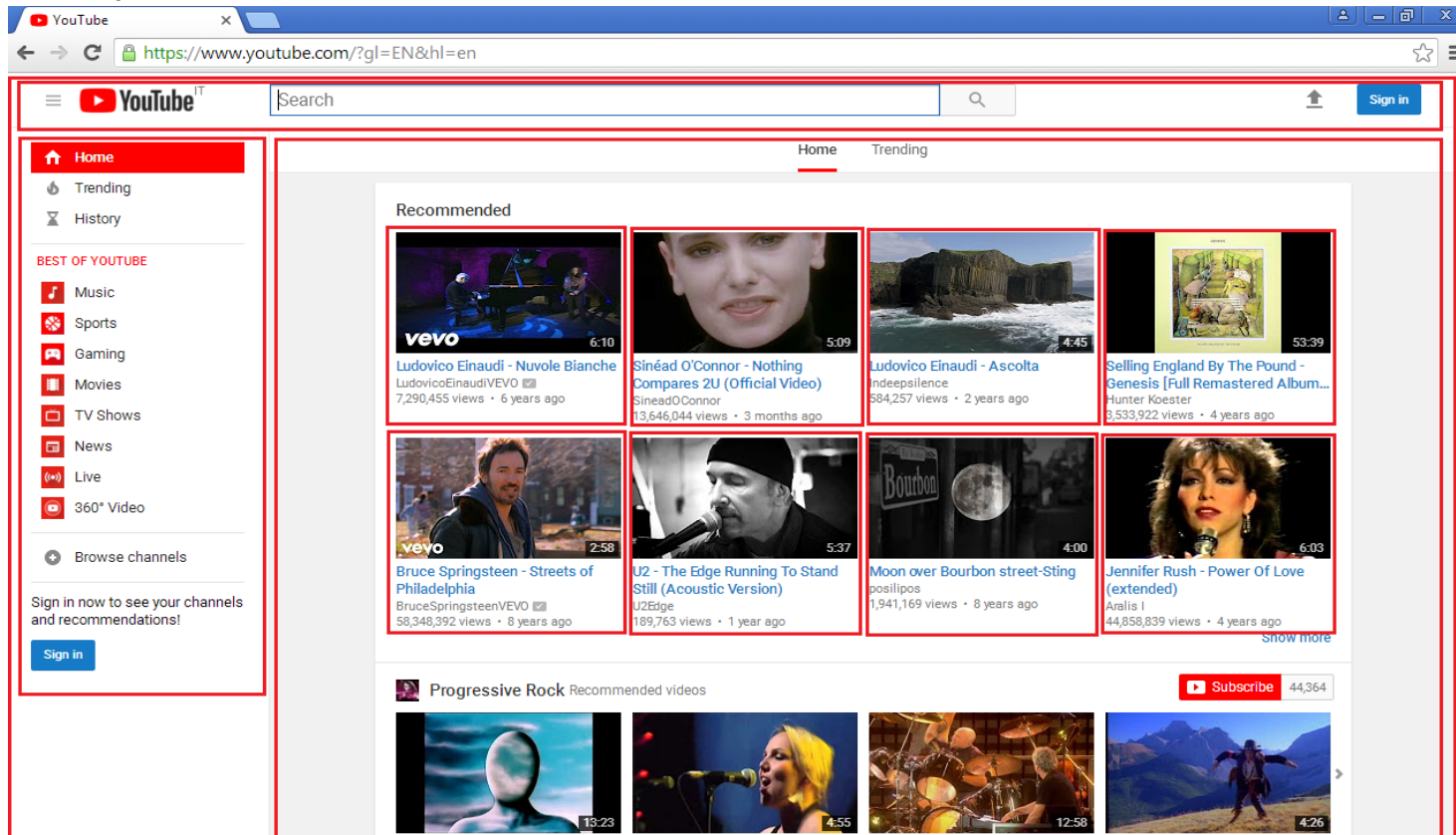
Decompose a User Interface



Decompose a User Interface



Decompose a User Interface



Container and Presentational Components

- **Container** components are components that have not a relevant visual effect.
Their role is mainly to group together other components.
- **Presentational** components are components that display data in a graphical form.

Activity C: Detecting Components in a Web User Interface

Aim

The aim of the activity is to address the design process when implementing React-based user interfaces.

Scenario

We need to convert the Wikipedia website's user interface (<https://en.wikipedia.org>) into React components.

Steps for Completion

1. Analyze its current structure and detect the items you can implement as components.
2. Indicate which would be container and which would be presentational components.

Summary

In this lesson, we started to explore the React world. In particular, we

- Established that React is a user interface library, used to implement the View part of various MV* design patterns
- Introduced the **create-react-app** tool, which helps us to set up a development environment to build React-based applications
- Explored the items comprising a typical React-based application
- Analyzed the approach to design user interfaces that best suits the React world