Modern JS: Webpack Dev Server Configuration

Open webpack_dev_server.html for the markup/code | Images are taken from: JS Course by Jonas Schmedtmann

In order to make our lives a bit easier when writing JavaScript code, we are going to add the **Webpack Dev Server** to our setup. Because of adding Webpack Dev Server, we will have automatic page reload when we save our code. Therefore, besides all the amazing functionalities included in Webpack, it also provides us with a **Development Server** which will automatically bundle all our JavaScript files and then reload the app in the browser whenever we change a file. This will save a lot of time when writing our JavaScript app. And so, we install the Webpack Dev Server using NPM (as it is an NPM Package) as shown below (command is: `npm install webpack-dev-server --save-dev`).

```
C:\Users\srira\Desktop\JavaScript\Modern-JS-ES6-NPM-Babel-Webpack\forkify_project>npm install webpack-dev-server --save-dev
npm WARN rollback Rolling back readable-stream@2.3.6 failed (this is probably harmless): EPERM: operation not permitted, lstat 'C:\Users
ode_modules\fsevents\node_modules'
npm WARN forkify@1.0.0 No repository field.
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.11 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.11: wanted {"os":"darwin","arch":"any"} (current: {'

+ webpack-dev-server@3.10.3
added 184 packages from 164 contributors and audited 8843 packages in 35.192s

17 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities


C:\Users\srira\Desktop\JavaScript\Modern-JS-ES6-NPM-Babel-Webpack\forkify_project>
```

Now we check the `package.json` file to see whether **Webpack Dev Server** is mentioned inside `"devDependencies"` object. `package.json` is shown below.

package.json 1-Babel-Webpack\forkify_project\package.json\

You, a few seconds ago | 1 author (You)

```
{
  "name": "forkify",
  "version": "1.0.0",
  "description": "Forkify Project",
  "main": "index.js",
  "scripts": {
    "dev": "webpack --mode development",
    "build": "webpack --mode production"
  },
  "author": "Sriram Chandrabhatta",
  "license": "ISC",
  "devDependencies": {
    "webpack": "^4.41.6",
    "webpack-cli": "^3.3.11",
    "webpack-dev-server": "^3.10.3"
  },
  "dependencies": {}
}
```

Now, in order to configure our dev server, we will add another field to `webpack.config.js` file (which is our configuration file). The field that we add in the `module.exports` object is `devServer`. The `devServer` field is an object, in which we first add the field `contentBase` where we specify the folder from where Webpack can serve the files. In this case, that's the `dist` folder in our app. Therefore, we set `contentBase` as, `contentBase: './dist'`. We put `./dist` as our `contentBase` because all of our code inside the `./dist` folder is the code that will be shipped to the client and therefore, all of the final production code is inside the `./dist` folder. `./src` folder is only there for development purposes, not for distribution/deployment purposes. Therefore, all our source code for the app goes in `./src` folder, which then gets compiled and bundled into the `./dist` folder as `bundle.js`, and so, what we really want to serve to the client/end-user is the resource (code, html) inside the `./dist` folder. After we make the changes, the configuration file, i.e., `webpack.config.js` file should look as shown below.

webpack.config.js roject\webpack.config.js\ <unknown>\

```
You, a few seconds ago | 1 author (You)
const path = require('path');

module.exports = {
    entry: './src/js/index.js',
    output: {
        path: path.resolve(__dirname, 'dist/js'),
        filename: 'bundle.js',
    },
    mode: 'development',
    devServer: {        You, a few seconds ago • Unc
        contentBase: './dist'
    }
};
```

To make the Webpack Dev Server work, we add an **npm script** to `package.json`. Inside `package.json`, in the `"scripts"` field, we add a new field which is the standard name for dev servers and it is called `"start"`. Therefore, the script `"start"` is a script that will always keep running in the background and updates the browser as soon as we change our code. When we run the `"start"` script, what we want `npm` to do is to run the webpack dev server in development mode and then open the app in the browser automatically. So for that, we give our script as: "start": "webpack-dev-server --mode development --open" as shown below.

package.json Modern-JS-ES6-NPM-Babel-Webpack\forkify_project\package.json\ {} s

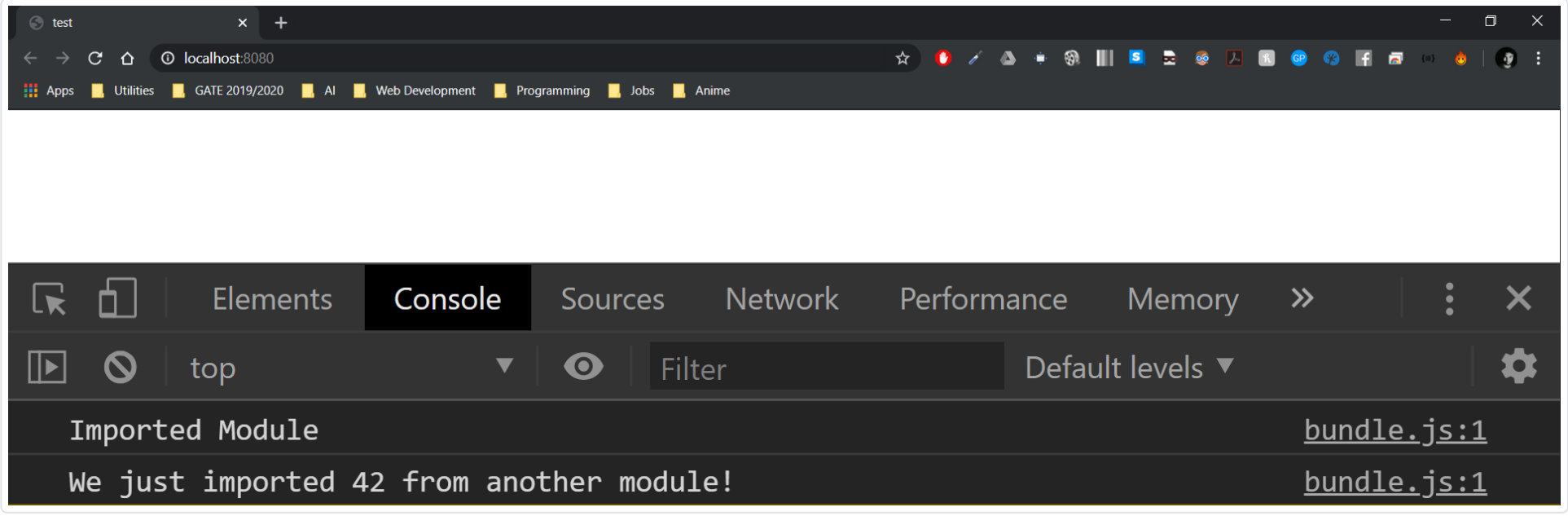
You, a few seconds ago | 1 author (You)

```
{
  "name": "forkify",
  "version": "1.0.0",
  "description": "Forkify Project",
  "main": "index.js",
  "scripts": {
    "dev": "webpack --mode development",
    "build": "webpack --mode production",
    "start": "webpack-dev-server --mode development --open"
  },
  "author": "Sriram Chandrabhatta",
  "license": "ISC",
  "devDependencies": {
    "webpack": "^4.41.6",
    "webpack-cli": "^3.3.11",
    "webpack-dev-server": "^3.10.3"
  },
  "dependencies": {}
}
```

We will run the **npm script** with the command - `npm run start` as shown below.

```
C:\Users\srira\Desktop\JavaScript\Modern-JS-ES6-NPM-Babel-Webpack\forkify_project>npm run start

> forkify@1.0.0 start C:\Users\srira\Desktop\JavaScript\Modern-JS-ES6-NPM-Babel-Webpack\forkify_project
> webpack-dev-server --mode development --open

i ｢wds｣: Project is running at http://localhost:8080/
i ｢wds｣: webpack output is served from /
i ｢wds｣: Content not from webpack is served from ./dist
i ｢wdm｣: Hash: 18598f11232ec4818b3d
Version: webpack 4.41.6
Time: 553ms
Built at: 02/18/2020 7:10:33 PM
    Asset      Size  Chunks             Chunk Names
bundle.js  361 KiB    main  [emitted]  main
Entrypoint main = bundle.js
[0] multi (webpack)-dev-server/client?http://localhost:8080 ./src/js/index.js 40 bytes {main} [built]
[./node_modules/ansi-html/index.js] 4.16 KiB {main} [built]
[./node_modules/html-entities/index.js] 231 bytes {main} [built]
[./node_modules/loglevel/lib/loglevel.js] 8.41 KiB {main} [built]
[./node_modules/webpack-dev-server/client/index.js?http://localhost:8080] (webpack)-dev-server/client?http://localhost:8080 4.29 KiB {main} [built
]
[./node_modules/webpack-dev-server/client/overlay.js] (webpack)-dev-server/client/overlay.js 3.51 KiB {main} [built]
[./node_modules/webpack-dev-server/client/socket.js] (webpack)-dev-server/client/socket.js 1.53 KiB {main} [built]
[./node_modules/webpack-dev-server/client/utils/createSocketUrl.js] (webpack)-dev-server/client/utils/createSocketUrl.js 2.91 KiB {main} [built]
[./node_modules/webpack-dev-server/client/utils/log.js] (webpack)-dev-server/client/utils/log.js 964 bytes {main} [built]
[./node_modules/webpack-dev-server/client/utils/reloadApp.js] (webpack)-dev-server/client/utils/reloadApp.js 1.59 KiB {main} [built]
[./node_modules/webpack-dev-server/client/utils/sendMessage.js] (webpack)-dev-server/client/utils/sendMessage.js 402 bytes {main} [built]
[./node_modules/webpack-dev-server/node_modules/strip-ansi/index.js] (webpack)-dev-server/node_modules/strip-ansi/index.js 161 bytes {main} [built
]
[./node_modules/webpack/hot sync ^\.\/log$] (webpack)/hot sync nonrecursive ^\.\/log$ 170 bytes {main} [built]
[./src/js/index.js] 187 bytes {main} [built]
[./src/js/test.js] 51 bytes {main} [built]
    + 19 hidden modules
i ｢wdm｣: Compiled successfully.
```

And the **npm script** opens the browser with `./dist/index.html` and we get the output in the console as shown below.



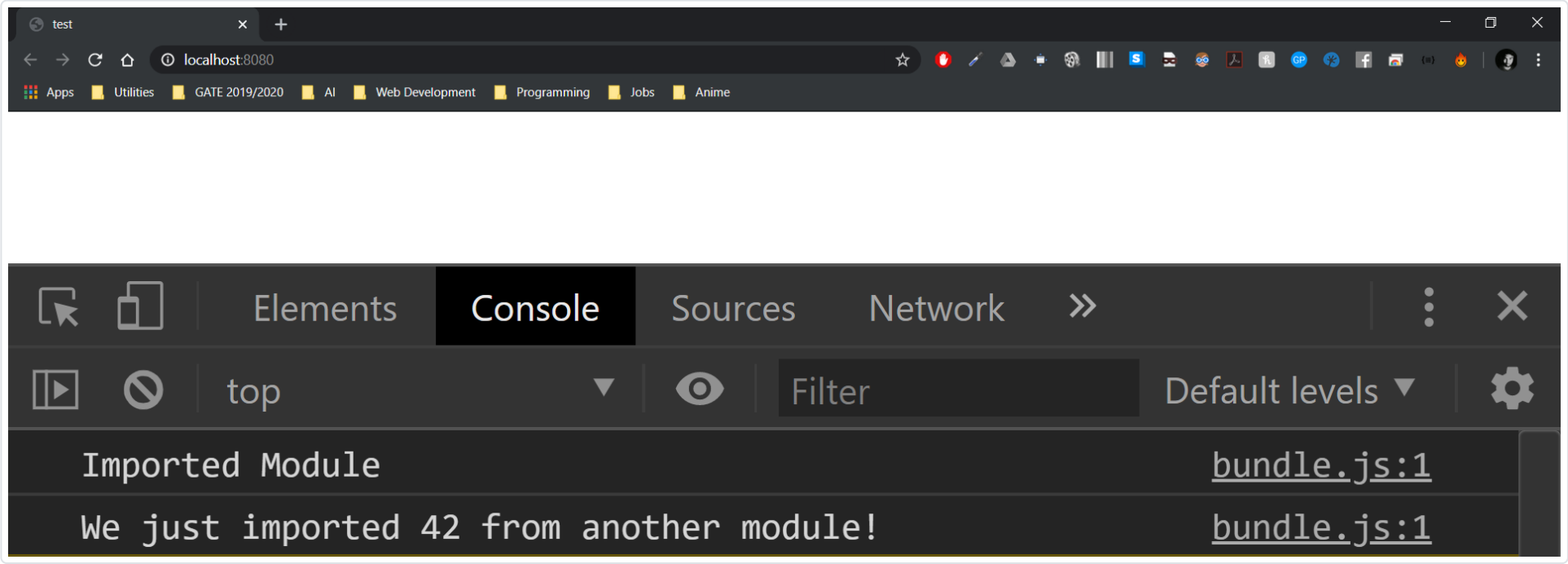
The only difference when opening `./dist/index.html` normally, opening it like this is, now `./dist/index.html` is running on a local webserver (as we can see the address above which says: `localhost:8080`) where it is simulating a real HTTP server while when we open it normally, it is just opening as a file (using the `file://...` protocol).

Now, we can see changes in the app as and when we change the code in the app. We will be able to see the output being reflected instantly in webpack dev server (which is `localhost:8080` in our browser). So now, if we change the code in the `./src/js/test.js` and `./src/js/index.js` files, we will see those changes being reflected @`localhost:8080`. The changes we made in the files is shown below.

Now, the output we should see in the console should be:

```
Imported Module
We just imported 12345 from another module called test
```

But the output we get is shown below.



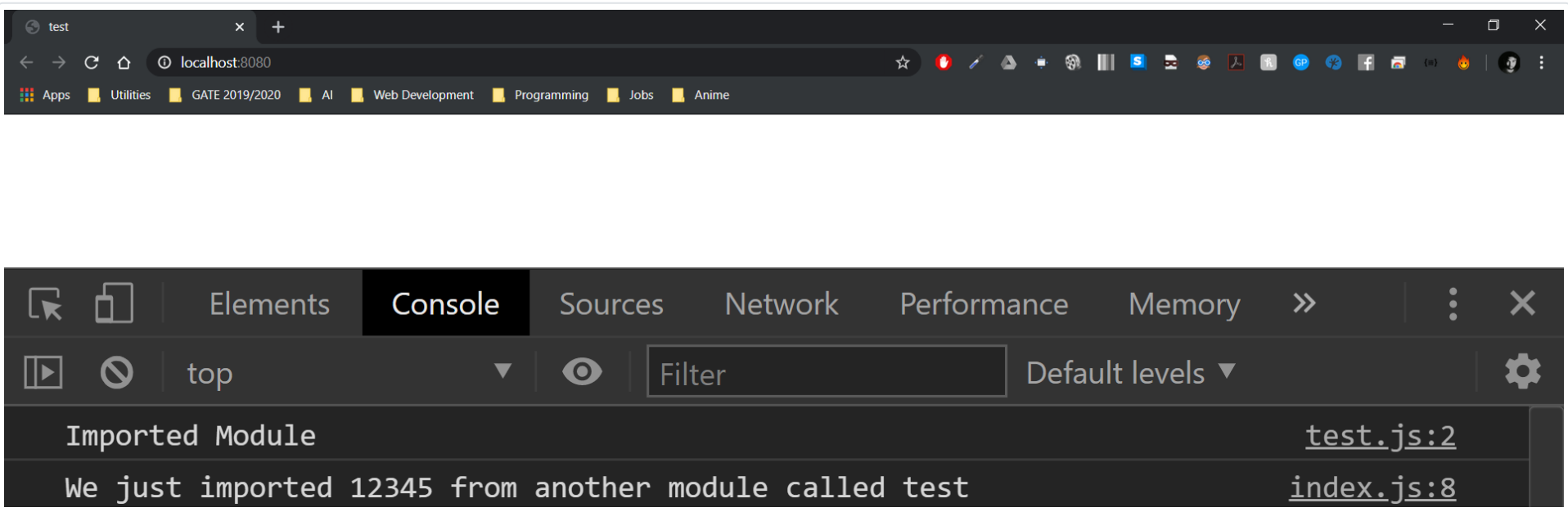
The output we get is the same as the previous output and that's because, when we run the development server, webpack will bundle our modules together but it will actually not write it to a file on disk, instead webpack will automatically inject it into the html file (in our case, it is `./dist/index.html`). Therefore, we can go ahead and delete the `./dist/js/bundle.js` file, because whenever we run >`npm run start`, it is still reading JavaScript from `./dist/js/bundle.js` file, which we do not want and not from the one we are generating on the fly through the webpack dev server and that's also because there's a little problem in the webpack configuration file also. In our `webpack.config.js` file, in the `module.exports` object, the concerned field, which `ouput` field, has a property called

`path`, which resolved as the current absolute path along with `'dist/js'`. And because of `'dist/js'`, every time we run >`npm run start`, the webpack dev server will try to inject the dynamically made `bundle.js` file into an html file which it is trying to find inside the `.../dist/js` folder. But the required html file (which is `.../dist/index.html`) is present in the parent folder, which is `dist`. Therefore, we change and resolve the path to current absolute path along with it, we concatenate `'dist'`. Then, for `/dist/index.html` to find the Bundled JavaScript file, we change the `filename` property from `'bundle.js'` to `'js/bundle.js'`. After those changes, all we have managed to do is, we have changed the output folder to the entire `dist` folder. The changes made in `webpack.config.js` file is shown below.

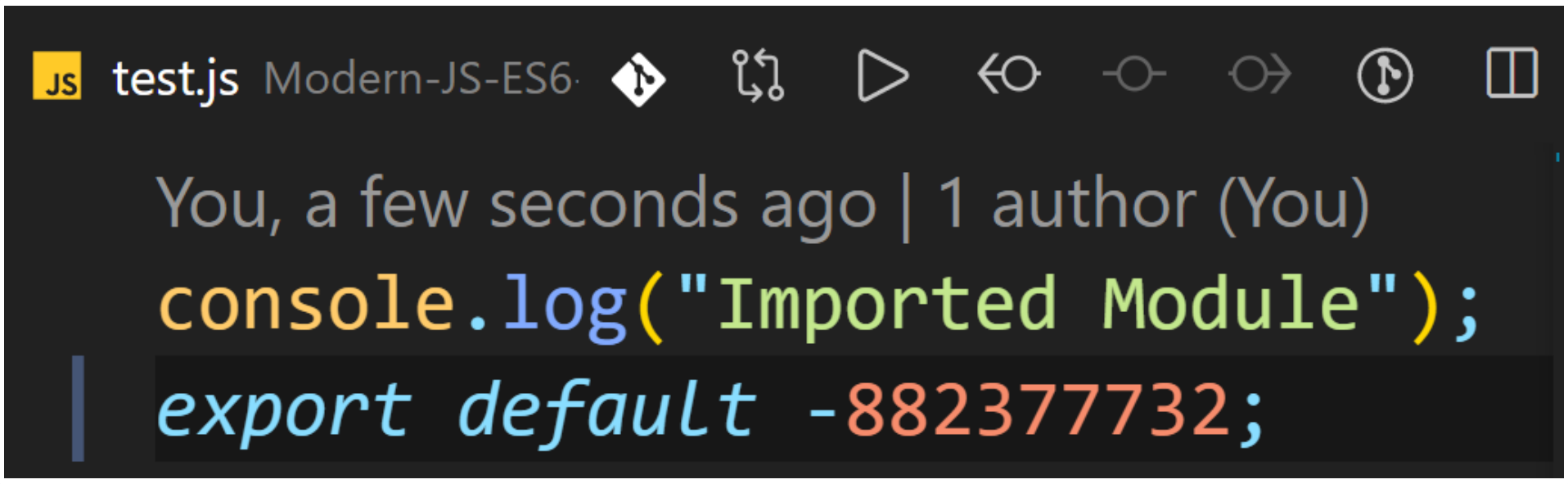
webpack.config.js Modern-JS-ES6-NPM-Babel-Webpack\for

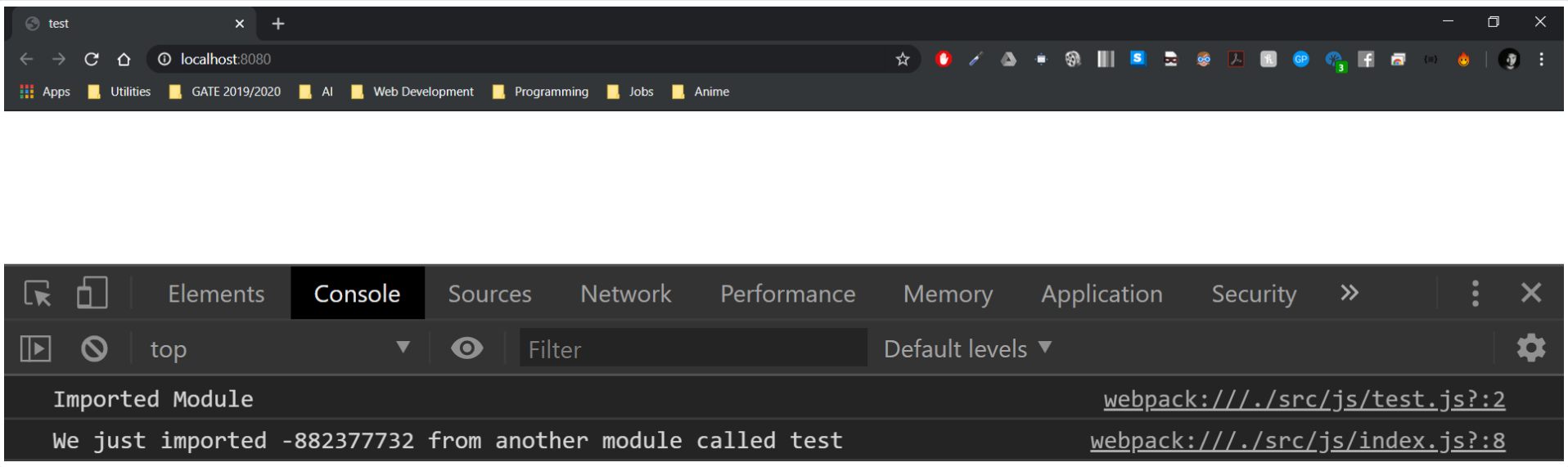
```
You, a few seconds ago | 1 author (You)
const path = require('path');

module.exports = {
    entry: './src/js/index.js',
    output: {
        path: path.resolve(__dirname, 'dist'),
        filename: 'js/bundle.js',
    },
    mode: 'development',
    devServer: {
        contentBase: './dist'
    }
};
```

Now we again run the **npm script** -- `start` using the command: >`npm run start`, and we see the output in the browser @`localhost:8080` as shown below.

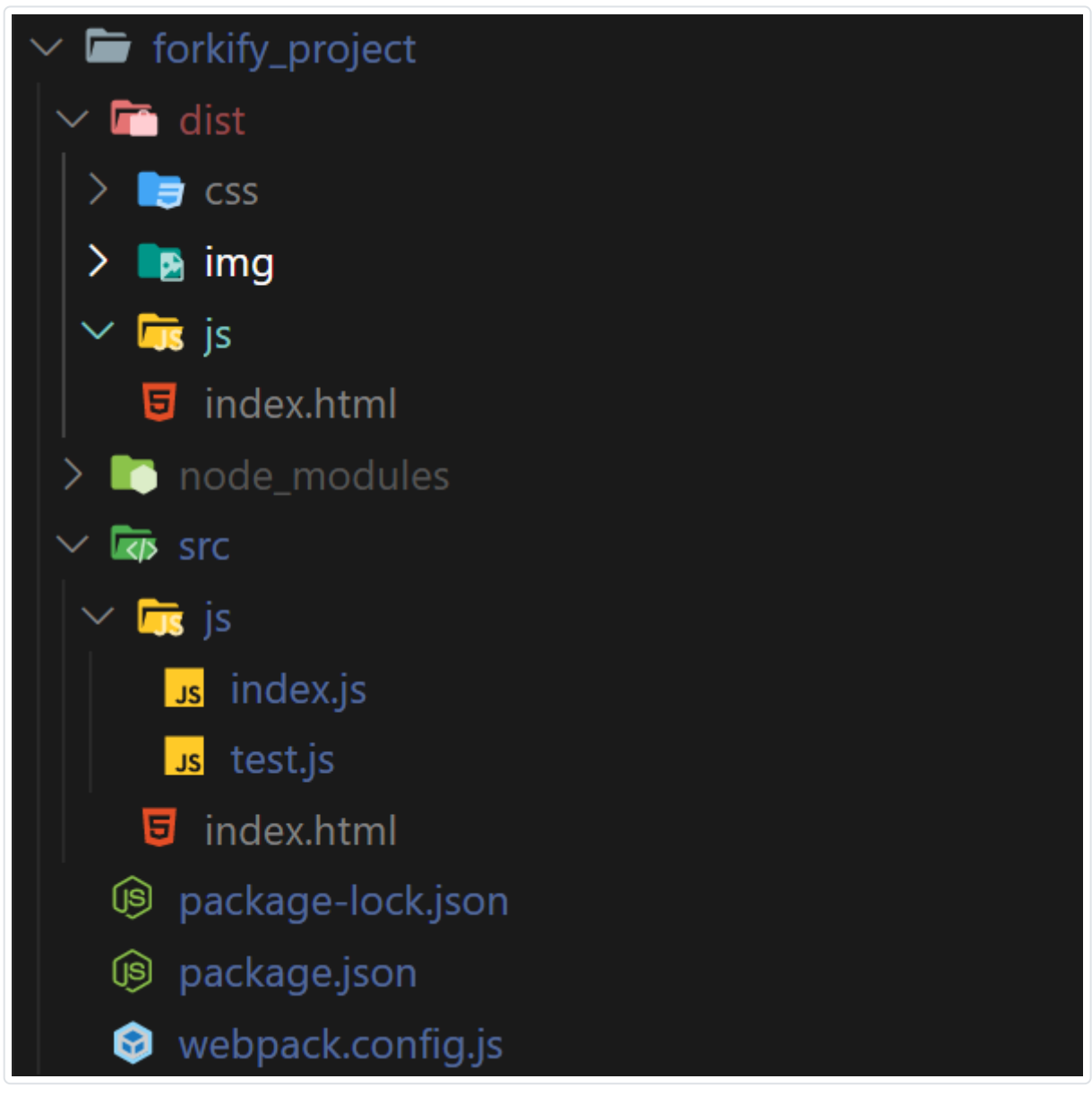


Now we change the code inside `./src/js/test.js` or `./src/js/index.js`, we will see the changes being reflected instantly in the browser @`localhost:8080` as shown below.

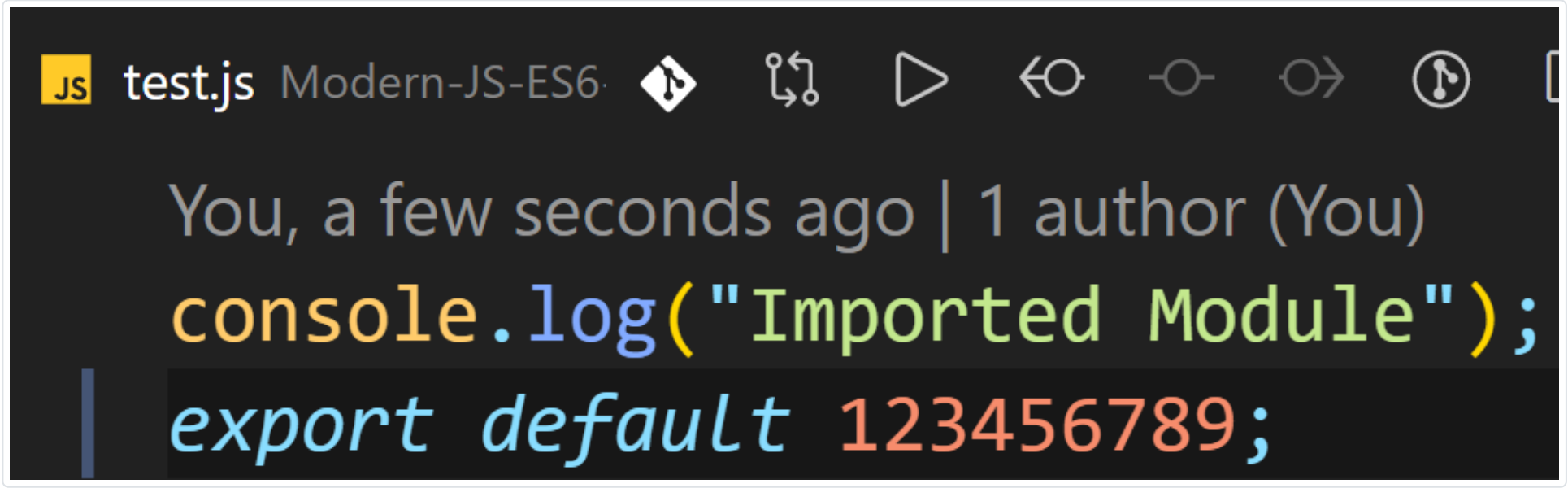


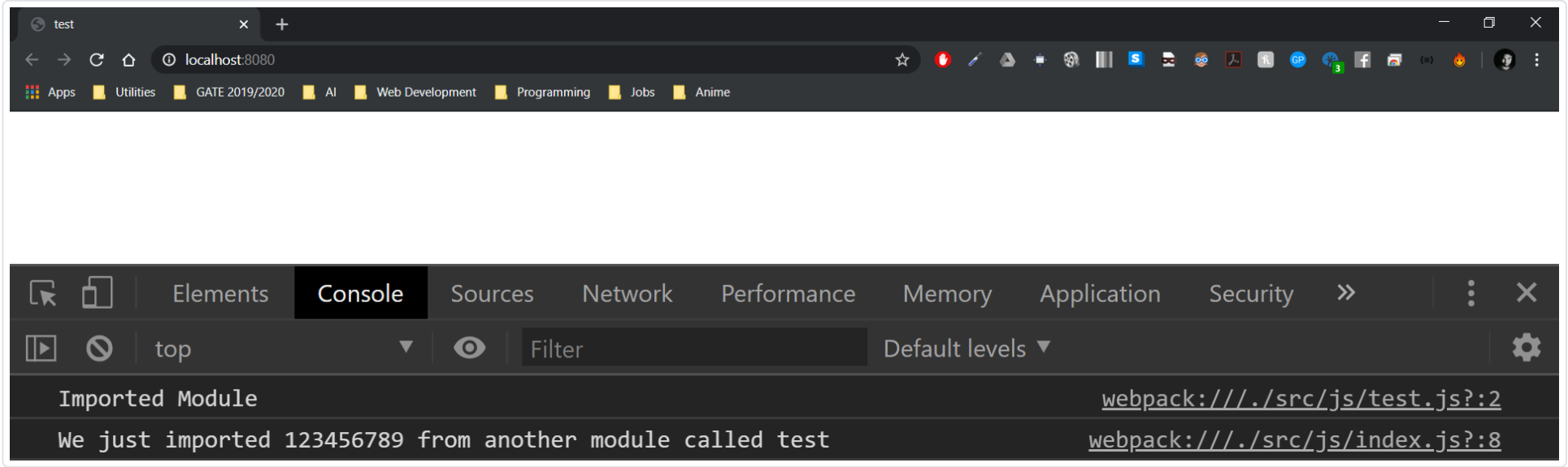


Note that we didn't re-run the >`npm run start` command or anything, everything happened dynamically. Now, we can actually delete the distribution code, which is `bundle.js` located at `./dist/js` and still the code will execute in the server because **we are running in development mode** and therefore, the `bundle.js` generated is directly being injected into `./dist/index.html` file rather than being written on the disk.
We can see below that there's no `bundle.js` file inside `./dist/js` folder.



And now we again make changes to `./src/js/test.js` file and we can see the changes being registered in the browser @`localhost:8080` as shown below.





And so, the `bundle.js` file from `./dist/js` is actually gone, but our webpack dev server running @`localhost:8080` in the browser, is still actually registering the changes we make inside the app. That's great!

One final thing we would do (apart from making the webpack dev server work for us) is to copy the `index.html` file from `./src` folder, into our distribution folder, i.e., `./dist`. Now, in `index.html` from `./src`, we can go ahead and delete the `<script src="bundle.js"></script>` tag, or just comment it. Now, what we want to do is that, we want to copy the markup of `./src/index.html` into `./dist/index.html` and inject the `<script src="bundle.js"><script>` tag into it automatically. This can be done using webpack very easily. Now, in order to do that, we use something called **plugins**.

Remember the **four core concepts** of webpack? **entry points**, **output**, **loaders** and **plugins**. We will now talk about **plugins**. **Plugins** allow us to do complex processing of our input files, and in this case our input file is `./src/index.html`. Therefore we will use a plugin called **html webpack plugin** and in order to use it, we have to install it using `npm`. Command to install it: `npm install html-webpack-plugin --save-dev`. It is also shown below (along with changes that have been automatically made in `package.json`).

```
C:\Users\srira\Desktop\JavaScript\Modern-JS-ES6-NPM-Babel-Webpack\forkify_project>npm install html-webpack-plugin --save-dev
npm WARN forkify@1.0.0 No repository field.
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.11 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.11: wanted {"os":"darwin","arch":"any"} (c
urrent: {"os":"win32","arch":"x64"})

+ html-webpack-plugin@3.2.0
added 40 packages from 59 contributors and audited 8938 packages in 16.247s

18 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities


C:\Users\srira\Desktop\JavaScript\Modern-JS-ES6-NPM-Babel-Webpack\forkify_project>
```

package.json Modern-JS-ES6-NPM-Babel-Webpack\forkify_project\package.json\ {} dev

You, 2 minutes ago | 1 author (You)

```
{
  "name": "forkify",
  "version": "1.0.0",
  "description": "Forkify Project",
  "main": "index.js",
  "scripts": {
    "dev": "webpack --mode development",
    "build": "webpack --mode production",
    "start": "webpack-dev-server --mode development --open"
  },
  "author": "Sriram Chandrabhatta",
  "license": "ISC",
  "devDependencies": {
    "html-webpack-plugin": "^3.2.0",        You, 2 minutes ago
    "webpack": "^4.41.6",
    "webpack-cli": "^3.3.11",
    "webpack-dev-server": "^3.10.3"
  },
  "dependencies": {}
}
```

Now we have to make changes in the webpack configuration, i.e., in `webpack.config.js` file. We have to `require()` the file, i.e., `import` the file and save it into a variable inside the `webpack.config.json` file. The convention for the name of the variable used for importing the file(s) for `html-webpack-plugin` is `HtmlWebpackPlugin`.

Now, inside the `module.exports` object, we can specify a new property called `plugins`, which receives an array of all the plugins that we are using. And the first plugin (and the only one in our case) is the `html-webpack-plugin`. Therefore, we make an instance of HtmlWebpackPlugin and to its constructor, we send in an object (In Modern JS, we generally pass options/parameters to the constructor as an object) containing two properties which are `filename` and `template` where, `filename` is the target file onto which we want to copy the content of file mentioned in the `template` field. *Note that we only mention the name of the target file i.e., `'index.html'` in the `filename` field, because the path is taken care by webpack by looking at the `path` field mentioned inside the `output`*

object of `module.exports` *object of the* `webpack.config.js` *file. And also, the* `template` *field needs to have the source file mentioned along with the relative path i.e.,* `'./src/index.html'`.
The modified webpack configuration file is shown below.

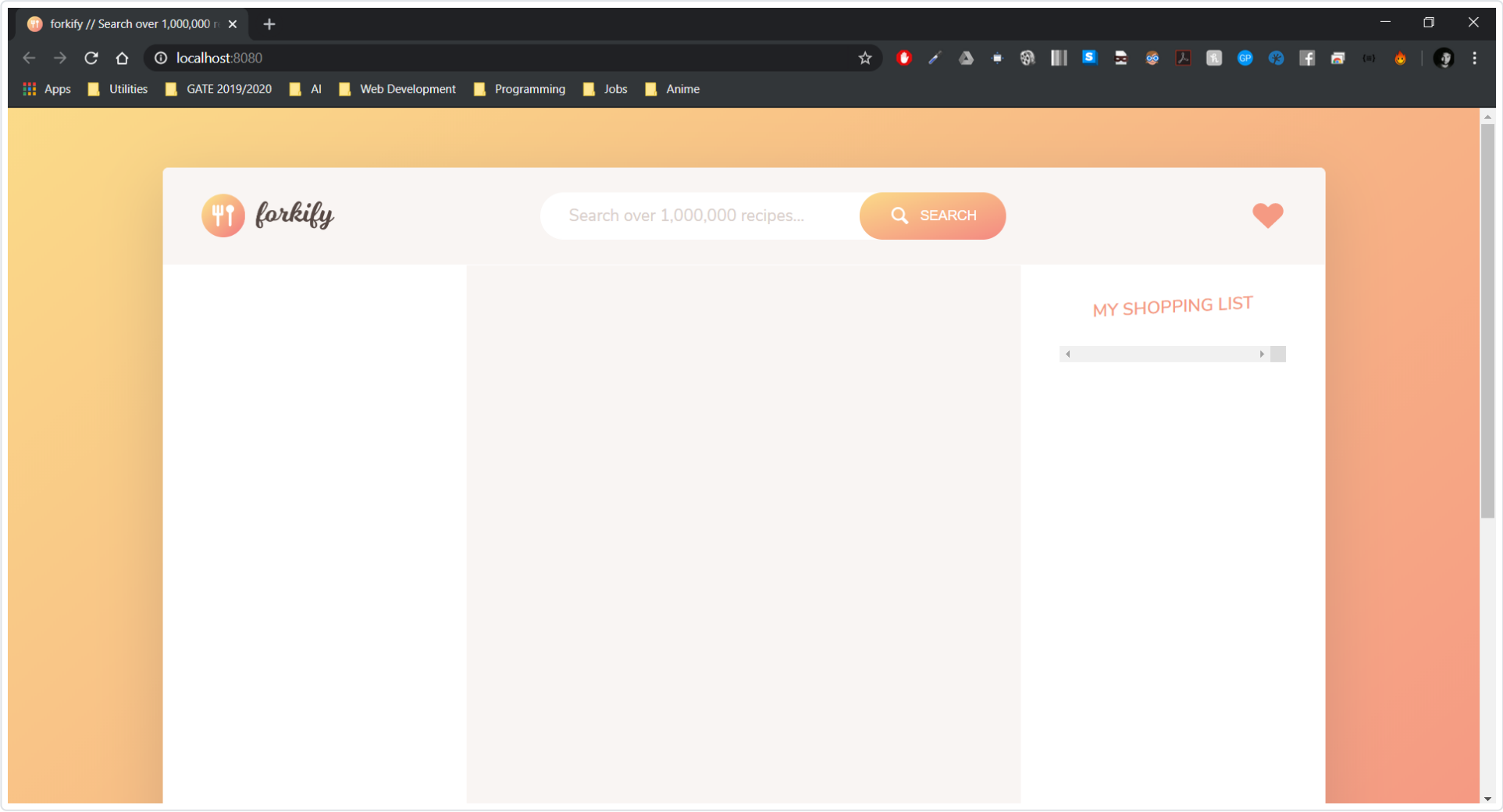
webpack.config.js Modern-JS-ES6-NPM-Babel-Webpack\forkify_project\webpack.config.js\ <unknown>\ plugins

You, a few seconds ago | 1 author (You)

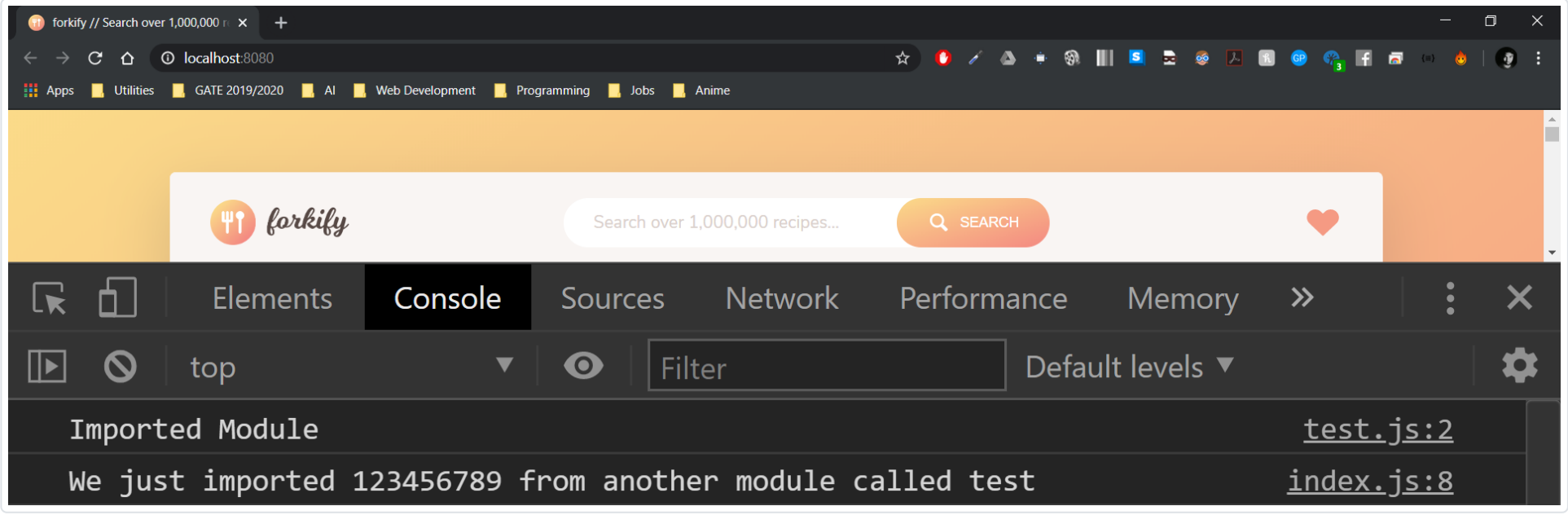
```
const path = require('path');
const HtmlWebpackPlugin = require('html-webpack-plugin');

module.exports = {
    entry: './src/js/index.js',
    output: {
        path: path.resolve(__dirname, 'dist'),
        filename: 'js/bundle.js',
    },
    mode: 'development',
    devServer: {
        contentBase: './dist'
    },
    plugins: [      You, a few seconds ago • Uncommitted changes
        new HtmlWebpackPlugin({
            filename: 'index.html',  // filename of the the target file
            template: './src/index.html' // filename along with path of the source
        })
    ]
};
```

We will start the webpack dev server again now using the `npm` command: `npm run start` as shown below, and it will open `./dist/index.html` (if `index.html` is not available at `./dist`, webpack will automatically create `index.html` inside `./dict`. As a matter of fact, we can delete `index.html` from `./dist` and check whether webpack automatically creates `index.html` for us inside `./dist` folder) in the browser @`localhost:8080`, with the code copied from `./src/index.html` as shown below.

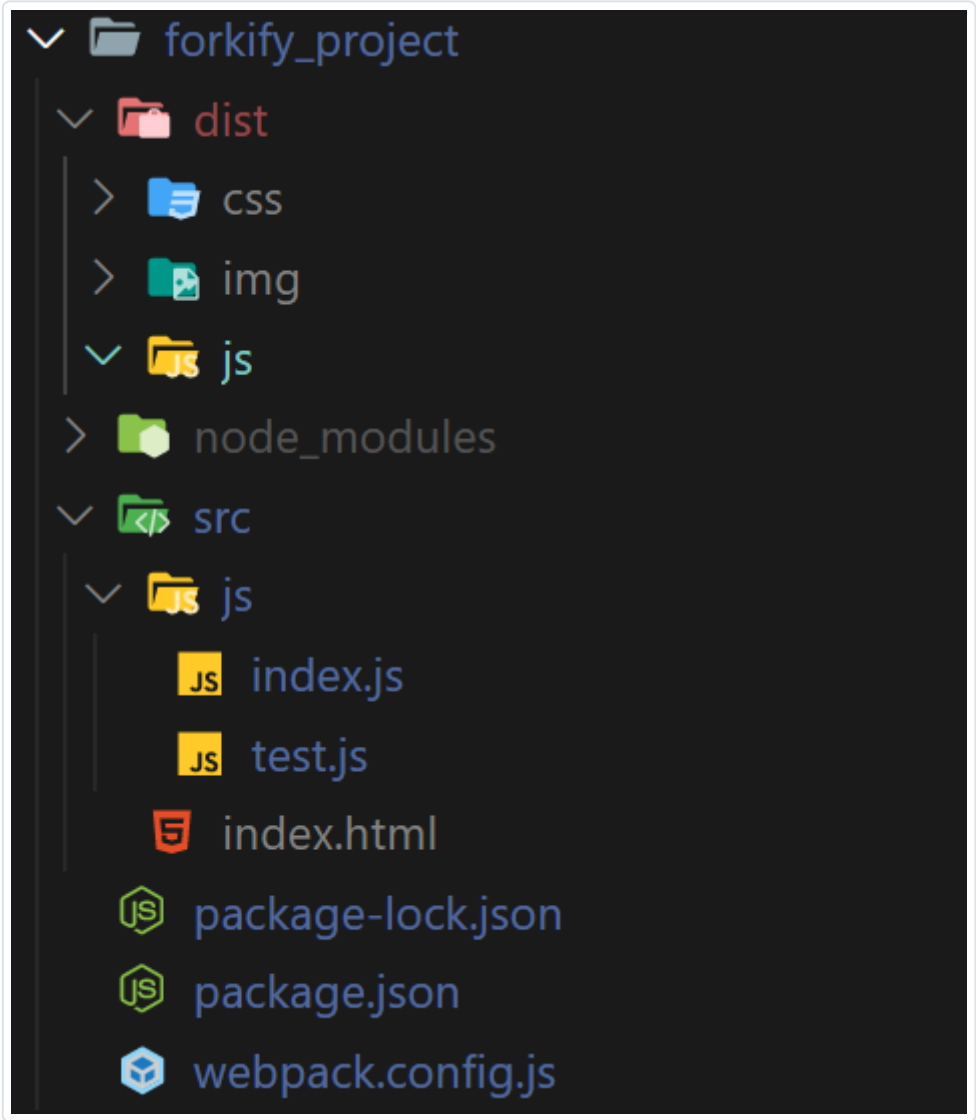
```
C:\Users\srira\Desktop\JavaScript\Modern-JS-ES6-NPM-Babel-Webpack\forkify_project>npm run start

> forkify@1.0.0 start C:\Users\srira\Desktop\JavaScript\Modern-JS-ES6-NPM-Babel-Webpack\forkify_project
> webpack-dev-server --mode development --open

i ｢wds｣: Project is running at http://localhost:8080/
i ｢wds｣: webpack output is served from /
i ｢wds｣: Content not from webpack is served from ./dist
i ｢wdm｣: wait until bundle finished: /
i ｢wdm｣: Hash: d9e2a0befbb932c2d188
Version: webpack 4.41.6
Time: 855ms
Built at: 02/18/2020 9:00:00 PM
       Asset      Size  Chunks             Chunk Names
  index.html  17.7 KiB          [emitted]
js/bundle.js   361 KiB    main  [emitted]  main
Entrypoint main = js/bundle.js
[0] multi (webpack)-dev-server/client?http://localhost:8080 ./src/js/index.js 40 bytes {main} [built]
[./node_modules/ansi-html/index.js] 4.16 KiB {main} [built]
[./node_modules/html-entities/index.js] 231 bytes {main} [built]
[./node_modules/loglevel/lib/loglevel.js] 8.41 KiB {main} [built]
[./node_modules/webpack-dev-server/client/index.js?http://localhost:8080] (webpack)-dev-server/client?http://localhost:8080 4.29 KiB {main} [built]
[./node_modules/webpack-dev-server/client/overlay.js] (webpack)-dev-server/client/overlay.js 3.51 KiB {main} [built]
[./node_modules/webpack-dev-server/client/socket.js] (webpack)-dev-server/client/socket.js 1.53 KiB {main} [built]
[./node_modules/webpack-dev-server/client/utils/createSocketUrl.js] (webpack)-dev-server/client/utils/createSocketUrl.js 2.91 KiB {main} [built]
[./node_modules/webpack-dev-server/client/utils/log.js] (webpack)-dev-server/client/utils/log.js 964 bytes {main} [built]
[./node_modules/webpack-dev-server/client/utils/reloadApp.js] (webpack)-dev-server/client/utils/reloadApp.js 1.59 KiB {main} [built]
[./node_modules/webpack-dev-server/client/utils/sendMessage.js] (webpack)-dev-server/client/utils/sendMessage.js 402 bytes {main} [built]
[./node_modules/webpack-dev-server/node_modules/strip-ansi/index.js] (webpack)-dev-server/node_modules/strip-ansi/index.js 161 bytes {main} [built]
[./node_modules/webpack/hot sync ^\.\/log$] (webpack)/hot sync nonrecursive ^\.\/log$ 170 bytes {main} [built]
[./src/js/index.js] 198 bytes {main} [built]
[./src/js/test.js] 58 bytes {main} [built]
    + 19 hidden modules
Child html-webpack-plugin for "index.html":
     1 asset
    Entrypoint undefined = index.html
    [./node_modules/html-webpack-plugin/lib/loader.js!./src/index.html] 18.3 KiB {0} [built]
    [./node_modules/lodash/lodash.js] 528 KiB {0} [built]
    [./node_modules/webpack/buildin/global.js] (webpack)/buildin/global.js 472 bytes {0} [built]
    [./node_modules/webpack/buildin/module.js] (webpack)/buildin/module.js 497 bytes {0} [built]
i ｢wdm｣: Compiled successfully.
```

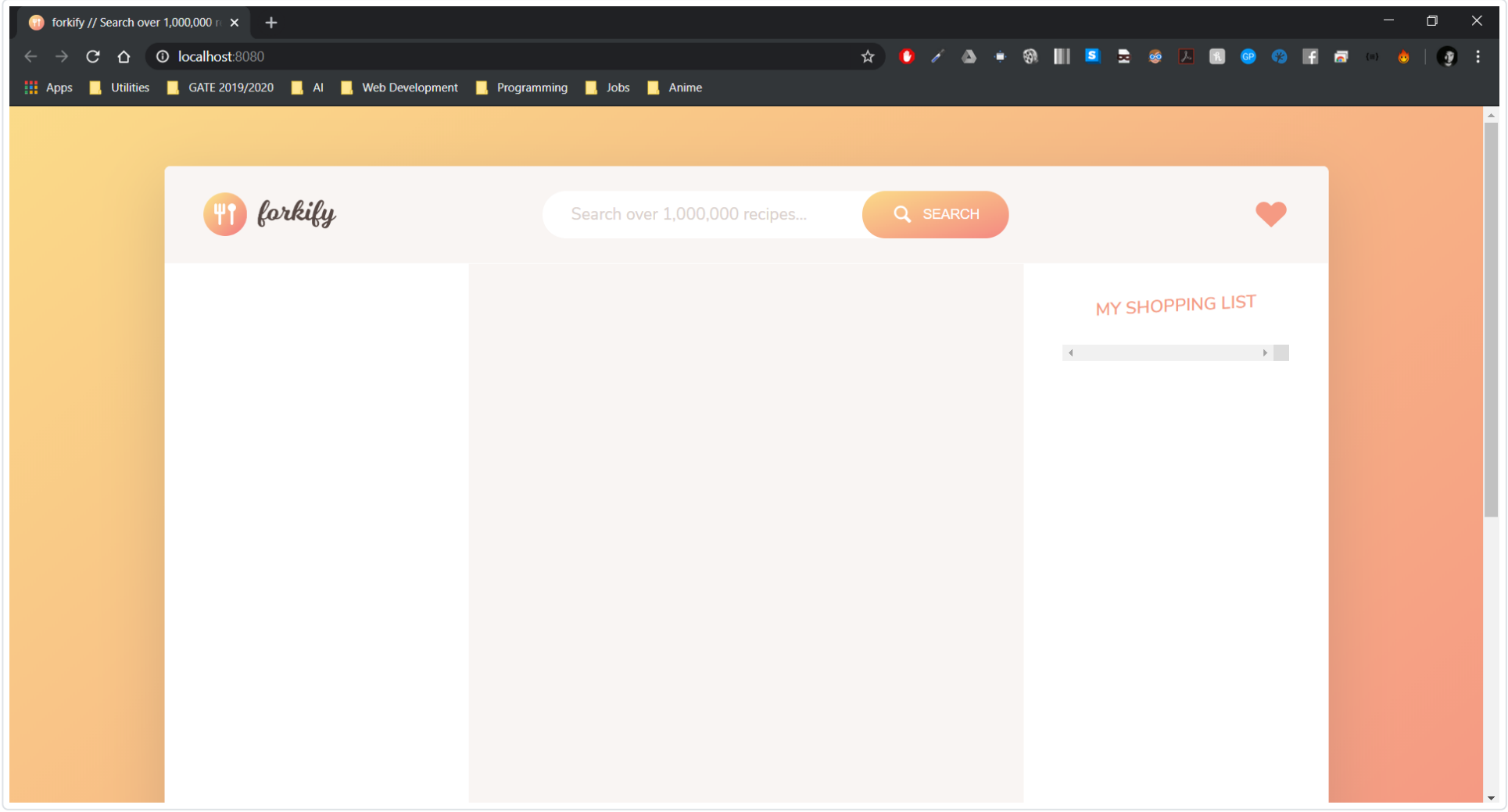


And when we check the console, it is still logged output from `./src/js/index.js` as shown below.



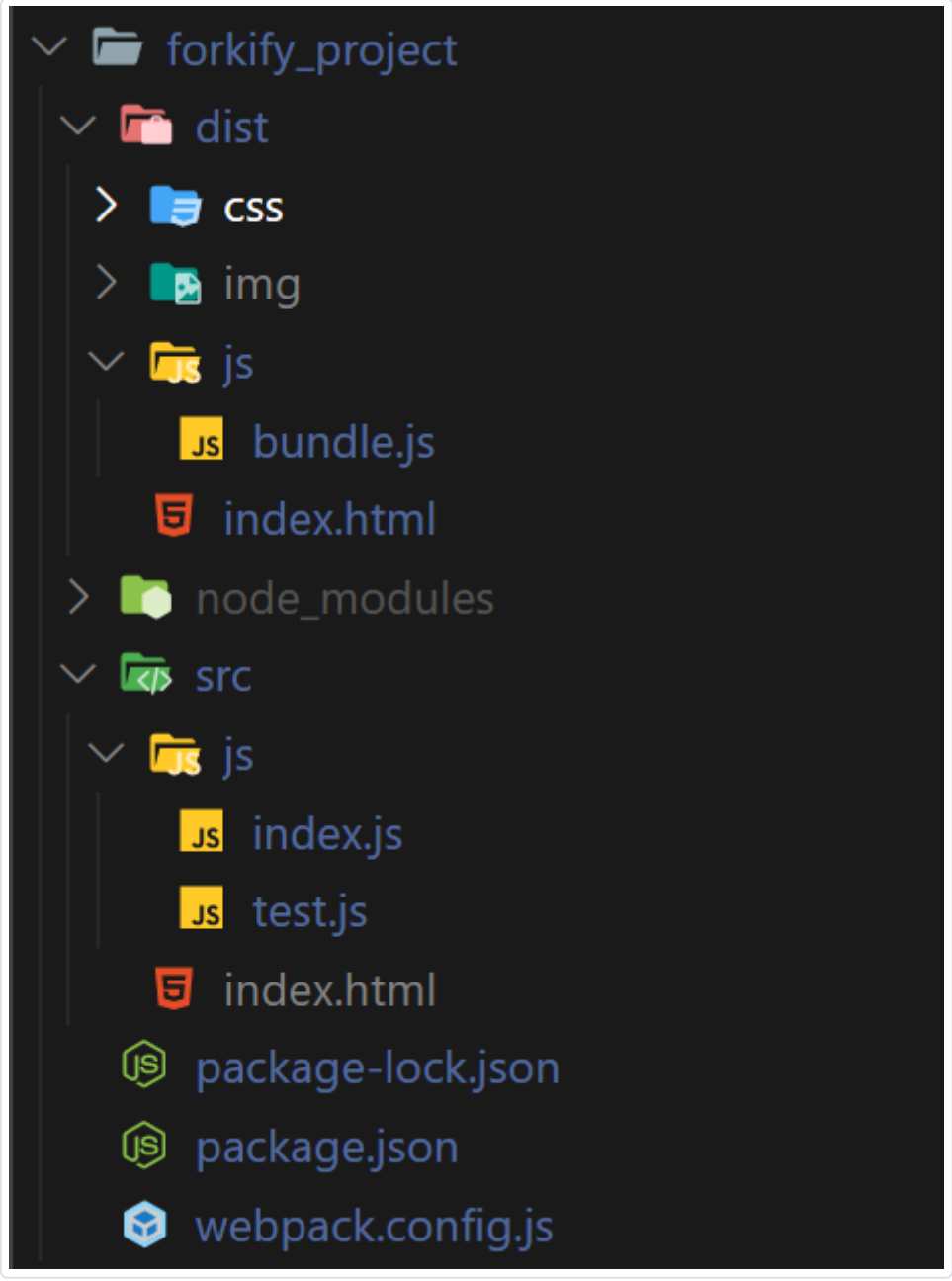
We can delete the existing `./dist/index.html` file and the server wouldn't crash, because we are running the app in the webserver in **development mode**. We can see below that `./dist/index.html` is deleted, but the webserver @`localhost:8080` didn't crash, as shown below.





And also, we can see from the image above that `bundle.js` is still not created inside `./dist/js` folder. The app is still running well because the app is running still in **development mode**. To actually write the `index.html` and `bundle.js` into the `./dist` and `./dist/js` folders respectively, we simply run either of the **npm script** we created earlier which were `build` and `dev` by simply running the command: `npm run build` or `npm run dev`, where `build` script generates the minified and optimized `bundle.js` whereas `dev` script generates un-optimized `bundle.js`, inside `./dist/js` folder. Both scripts generate `index.html` inside the `./dist` folder. The generated files along with the working of the command (dev version) is shown below.

```
C:\Users\srira\Desktop\JavaScript\Modern-JS-ES6-NPM-Babel-Webpack\forkify_project>npm run dev

> forkify@1.0.0 dev C:\Users\srira\Desktop\JavaScript\Modern-JS-ES6-NPM-Babel-Webpack\forkify_project
> webpack --mode development

Hash: b04709950cc9c1f3c42e
Version: webpack 4.41.6
Time: 337ms
Built at: 02/18/2020 9:20:39 PM
       Asset      Size  Chunks             Chunk Names
  index.html  17.7 KiB          [emitted]
js/bundle.js  4.66 KiB    main  [emitted]  main
Entrypoint main = js/bundle.js
[./src/js/index.js] 198 bytes {main} [built]
[./src/js/test.js] 58 bytes {main} [built]
Child html-webpack-plugin for "index.html":
     1 asset
    Entrypoint undefined = index.html
    [./node_modules/html-webpack-plugin/lib/loader.js!./src/index.html] 18.3 KiB {0} [built]
    [./node_modules/webpack/buildin/global.js] (webpack)/buildin/global.js 472 bytes {0} [built]
    [./node_modules/webpack/buildin/module.js] (webpack)/buildin/module.js 497 bytes {0} [built]
        + 1 hidden module

C:\Users\srira\Desktop\JavaScript\Modern-JS-ES6-NPM-Babel-Webpack\forkify_project>
```



We can also see that inside `./dist/index.html`, `<script src="js/bundle.js"></script>` is automatically copied from the webpack configuration file where we gave the `filename` to be `'js/bundle.js'` as shown below.

index.html

```
                        svg#icon-circle-with-cross"></
                        use>
                    </svg>
                </button>
            </li>
            -->
        </ul>

        <div class="copyright">
            &copy; by Jonas Schmedtmann. Powered by
            <a href="http://food2fork.com"
            target="_blank" class="link">Food2Fork.
            com</a>.
        </div>

    </div>
</div>

    <!-- <script  src="js/bundle.js"></script> -->
<script type="text/javascript" src="js/bundle.js"></
script></body>

</html>
```

webpack.config.js

```
You, 38 minutes ago | 1 author (You)
const path = require('path');
const HtmlWebpackPlugin = require('html-webpack-plugin');

module.exports = {
    entry: './src/js/index.js',
    output: {
        path: path.resolve(__dirname, 'dist'),
        filename: 'js/bundle.js',       You, 40 minutes ag
    },
    mode: 'development',
    devServer: {
        contentBase: './dist'
    },
    plugins: [
        new HtmlWebpackPlugin({
            filename: 'index.html',  // filename of the
            the target file
            template: './src/index.html' // filename
            along with path of the source
        })
    ]
};
```

The only thing that's missing now is **Babel** which converts ESNext JavaScript Code to ES5 JavaScript Code for compatibility across all the web browsers out there.