When a browser makes a get request to the index route of our application, it receives HTML.

But as we've seen, there's no file called index.html in any of the folders in our application.

When the app receives the request it has to assemble and render the HTML from templates and data.

This allows the application to serve customized web pages to different clients. As you know, browsers use HTML, CSS, and JavaScript to display a web page. The CSS and client-side JavaScript files will always be the same, no matter who requests them, or what the application state is.

In fact, they're literally our complete files sitting on the web server in one or more directories.

Express doesn't need to build them, like it does the HTML in our application.

For this reason, these files are called static assets.

We've been writing our Express application in JavaScript.

This is different from the client-side JavaScript our web page might use.

Our Express app and our browser app are two different environments and

play different roles in what the end-user sees.

In fact, the static JavaScript files don't even look like JavaScript to Express.

They might as well be long strings of nonsense as far as Express is concerned.

These files are just handed straight over to the browser without even being attempted to be interpreted by Express.

The browser knows what to do with these files and turns them into all kinds of magical user interactions.

To send images plain CCS and JavaScript files from the server's file system to the client. Express uses something called a Static Server.

Because these files are prebuilt and won't change on the server, passing them through all of Express's routing and template rendering would be a waste of time.

The Static Server just sends them straight to the client saving Express time and effort.

The name ‘public’ is commonly used for the folder containing static assets.