

## Group Members:

* Smile Smile(200563908)
* Dev Dev(200562142)
* Robin Robin(200571744)

**Assignment Instructions:**

You will create an express application:

1. Install the full Express required and recommended dependencies:
2. Install Express
3. Install Nodemon
4. Add a folder named “data” inside the project folder. Inside the “data” folder add a JSON file with at least 6 items (objects)
5. Adding a "public" folder for a full well-designed HTML website (static website):
6. 4 HTML pages at least
7. images/media and professional (nice and elegant) CSS
8. One of the HTML pages will be a demo for using API (Application Programming Interface) request to a back-end side (server-side) to load the JSON data (using Express with Node for sure)
9. This page should include a button to communicate with Express URL Route to call (fetch) JSON data
10. The JSON data that you will grab/load from the server-side should be displayed in a nicely formatted and professional layout to the end user (clients) in this page
11. you will need a JavaScript file to handle the front-end interactions with the user where you will attach an event listener to the button of the page. When the user clicks the button you will call function with Async/Await\* to fetch the data according to the URL that you specified (or will specify) in your application JS file.
12. \* Async/Await is the syntactic sugar of using promises
13. For sure, you will have to add the JavaScript file for running Express using the local virtual server "localhost"
14. Use any name you prefer to this Express application JavaScript file (by default, programmers use app.js or you can use index.js)
15. Write the full template/boilerplate of Express from you repos code examples or the express documentations
16. https://expressjs.com/en/starter/hello-world.html
17. Import or require the JSON file to be saved into a variable of any name of your choice
18. Add the .get() method with a specific route to send the JSON Data which the variable that you have created that contains the JSON content. Notice that the URL and the end-point of this get() method will be used in your client-side JavaScript file.

**Assignment Submission Materials:**

**1. The PDF file that contains the screenshots explained below:**

* Use **MS-Word** to collect all the assignment images (screenshots), put and arrange them all in one professional document then convert it to a PDF file to be uploaded/submitted. Please consider the following screenshots (images):
* The folder structure inside IntelliJ IDEA/Visual Studio Code or any other Editor that you are using that shows the 2 listed files above
* The VS Code embedded terminal window that shows the current working directory (the path of your folder)
* The PDF file should prepared in a professional way with a cover page

**2. The link to your GitHub repository where you have your assignment (project) uploaded**

**1. Introduction**

The **Sneaker Store Web Application** is a simple and dynamic project designed to showcase a collection of sneakers by fetching data from a JSON API. The application displays the sneaker details, including images, descriptions, and prices, in a visually appealing layout. It is built using **HTML**, **CSS**, and **JavaScript**, with a focus on modern web development practices such as fetching data from external sources and dynamically rendering it on the webpage.

This project demonstrates the use of essential web technologies and offers hands-on experience in integrating APIs, handling asynchronous data requests, and ensuring a responsive, user-friendly design.

**2. Technologies Used**

* **HTML (Hypertext Markup Language)**
* Used for structuring the content of the web pages. It provides the skeleton of the website by defining the layout, forms, and other essential elements.
* **CSS (Cascading Style Sheets)**
* Used for styling the web pages, making them visually appealing. CSS controls the layout, colors, fonts, spacing, and other design aspects. In this project, responsive design principles are implemented to ensure the site looks good on different screen sizes.
* **JavaScript**
* Used for dynamic content manipulation and making the webpage interactive. In this project, JavaScript is used to fetch data from an external JSON API and render it dynamically on the web page.
* **JSON (JavaScript Object Notation)**
* JSON is used as the data format for storing and transmitting sneaker information. It enables easy parsing and rendering of the sneaker data on the webpage.
* **API (Application Programming Interface)**
* An external API is used to fetch sneaker details like name, price, image, and description. JavaScript makes asynchronous requests to this API to retrieve and display the data in real time.

**Step1 :** **Install Node.js**

Node.js provides the runtime environment for running JavaScript code server-side. This is required to build the backend for our Sneaker Store application.

**Step2: Create Project Folder**

We will create a folder named as sneaker store and after that we will Initialize Node.js Project to create a package.json file with default configurations.

**Step 3: Install Required Packages**

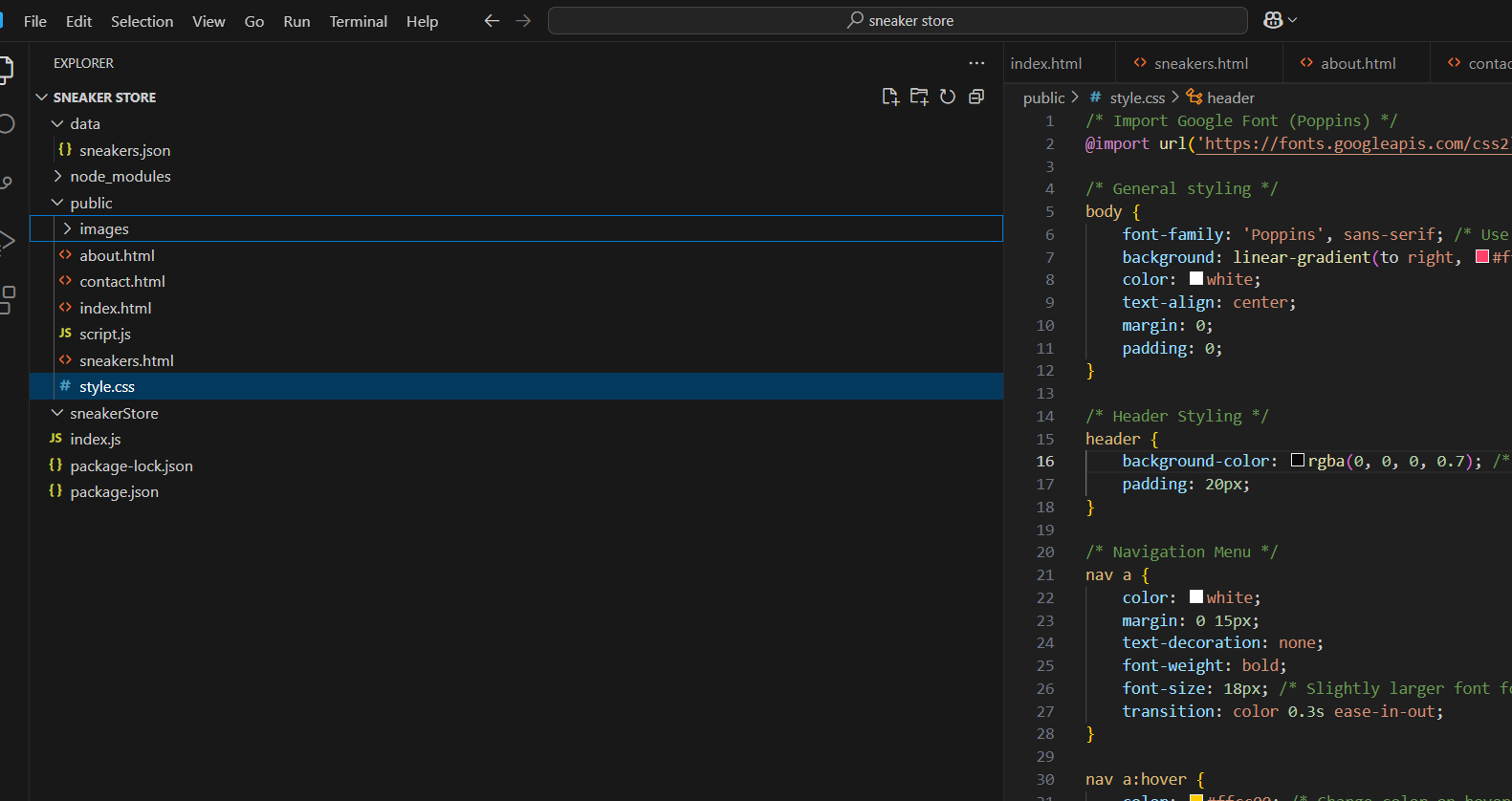
In this step we will Install Express.js and Install Nodemon (for automatic server restart)

* npm install express
* npm install --save-dev nodemon

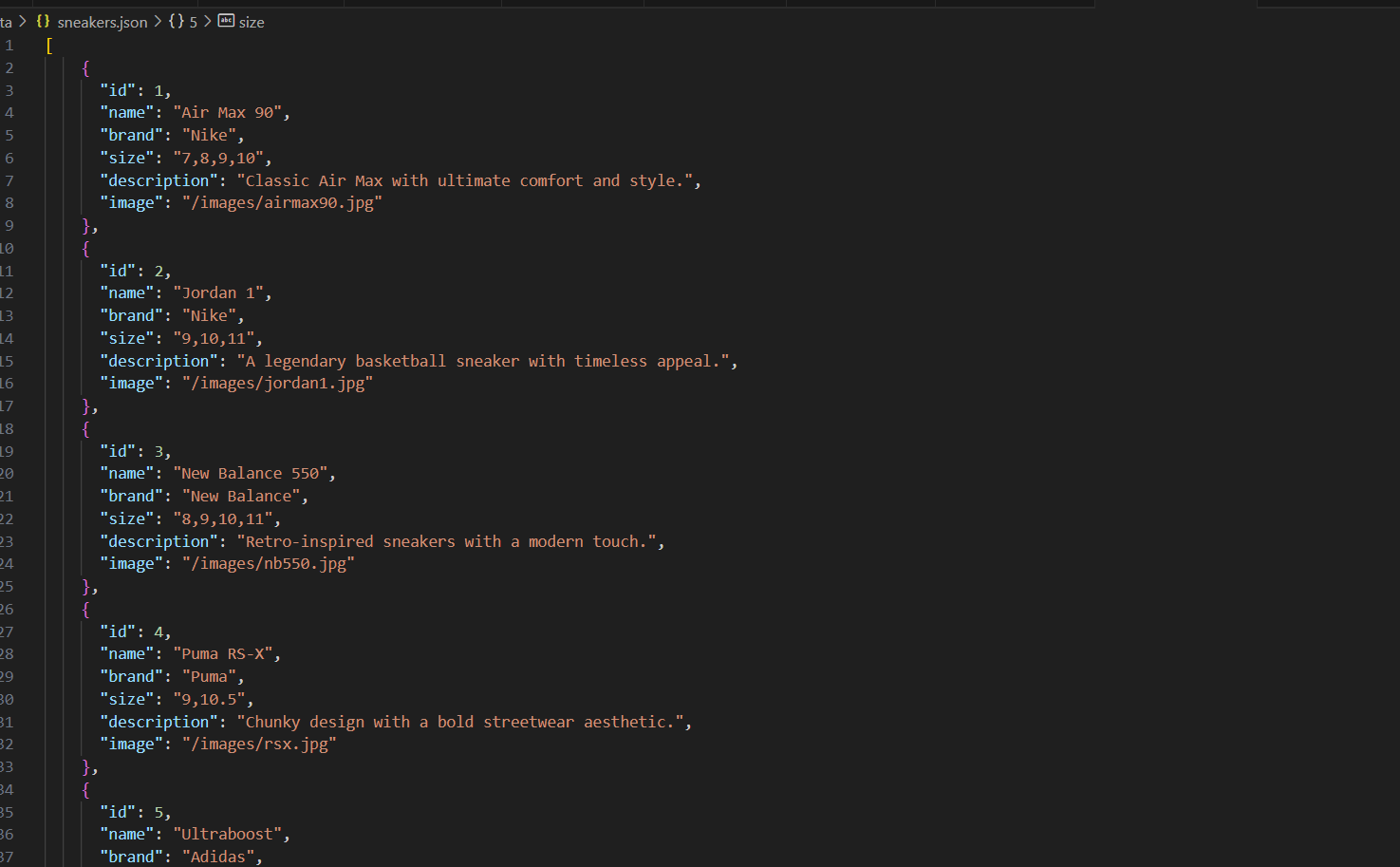
**Step 4: Create Project Files**

In step 4 ,we will create all the essential files to get the output and the project structure is as following

* **Create necessary files and folders like:**
  + **data/sneakers.json (to store sneaker data in JSON format)**
  + **public/ (to store static files like images, HTML, CSS, and JS files)**
  + **sneakerStore/ (for backend code and Express setup)**
  + **package.json (for dependency management)**



**Step 5: Create sneakers.json in data/ Folder**

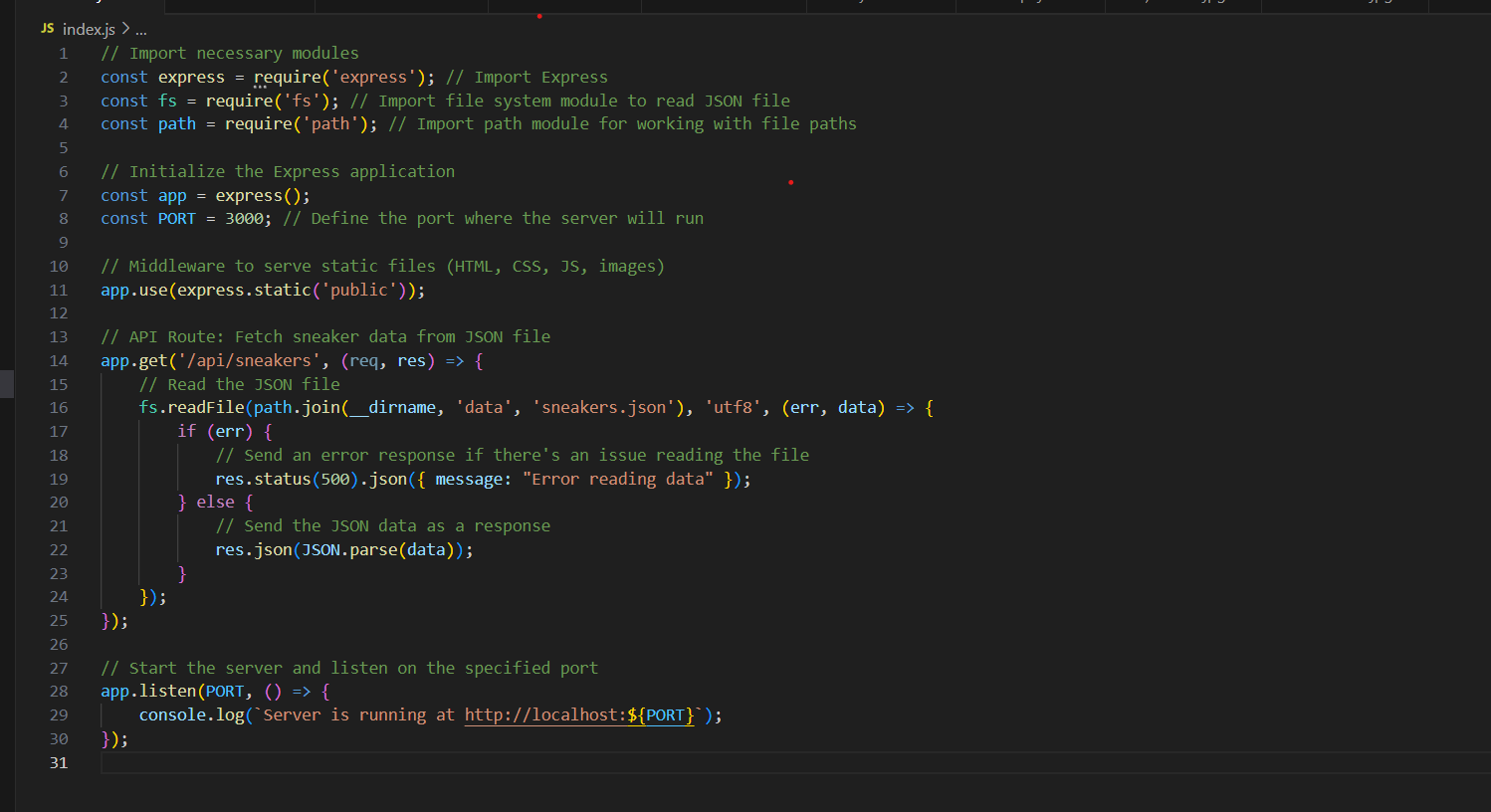
****

**Step 6: Set up Express.js CRUD application:**

This is where we set up the backend using Express to serve routes and static files.

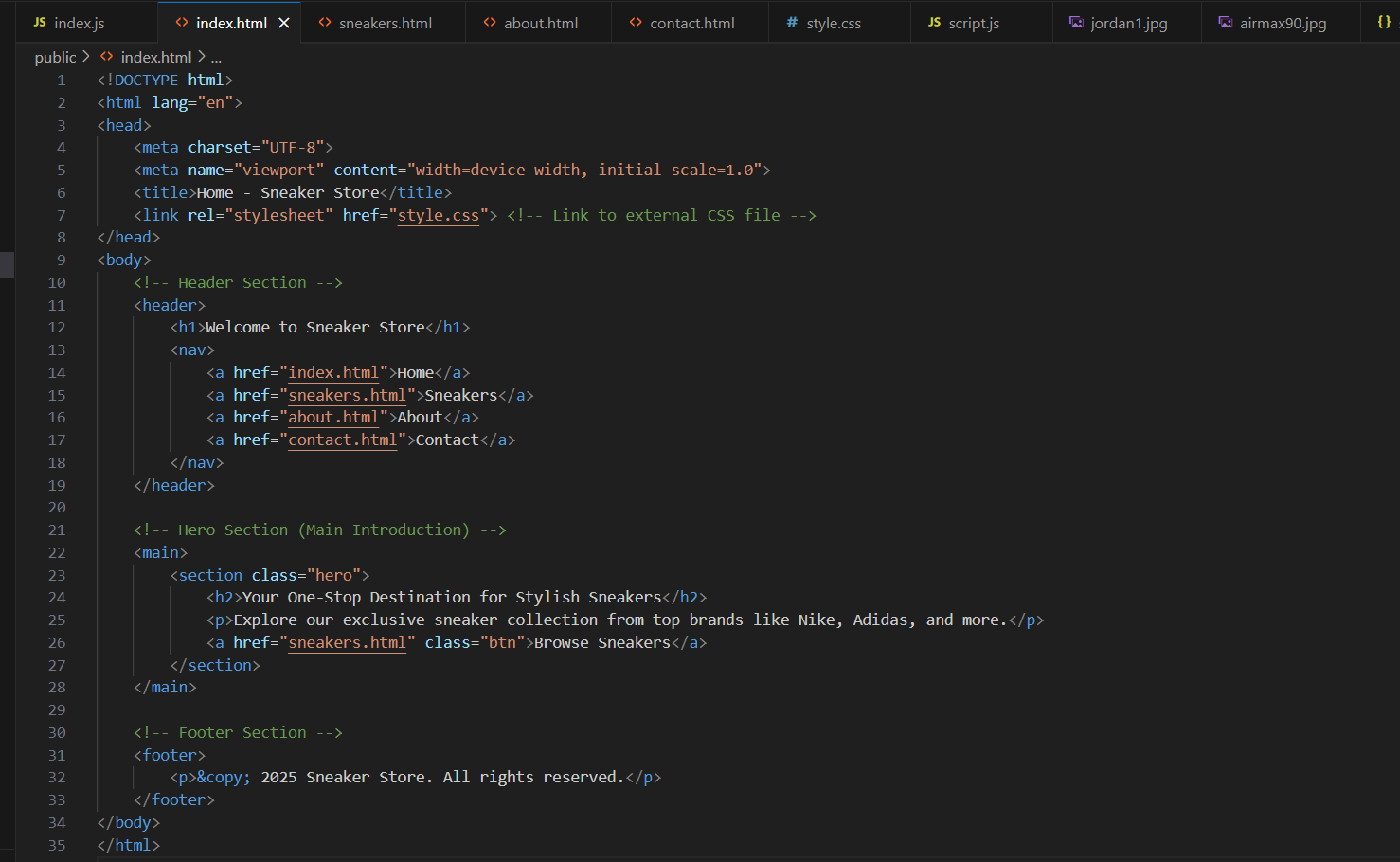
**Action**:

* Inside the sneaker-store folder, create INDEX.JS and add the following content:

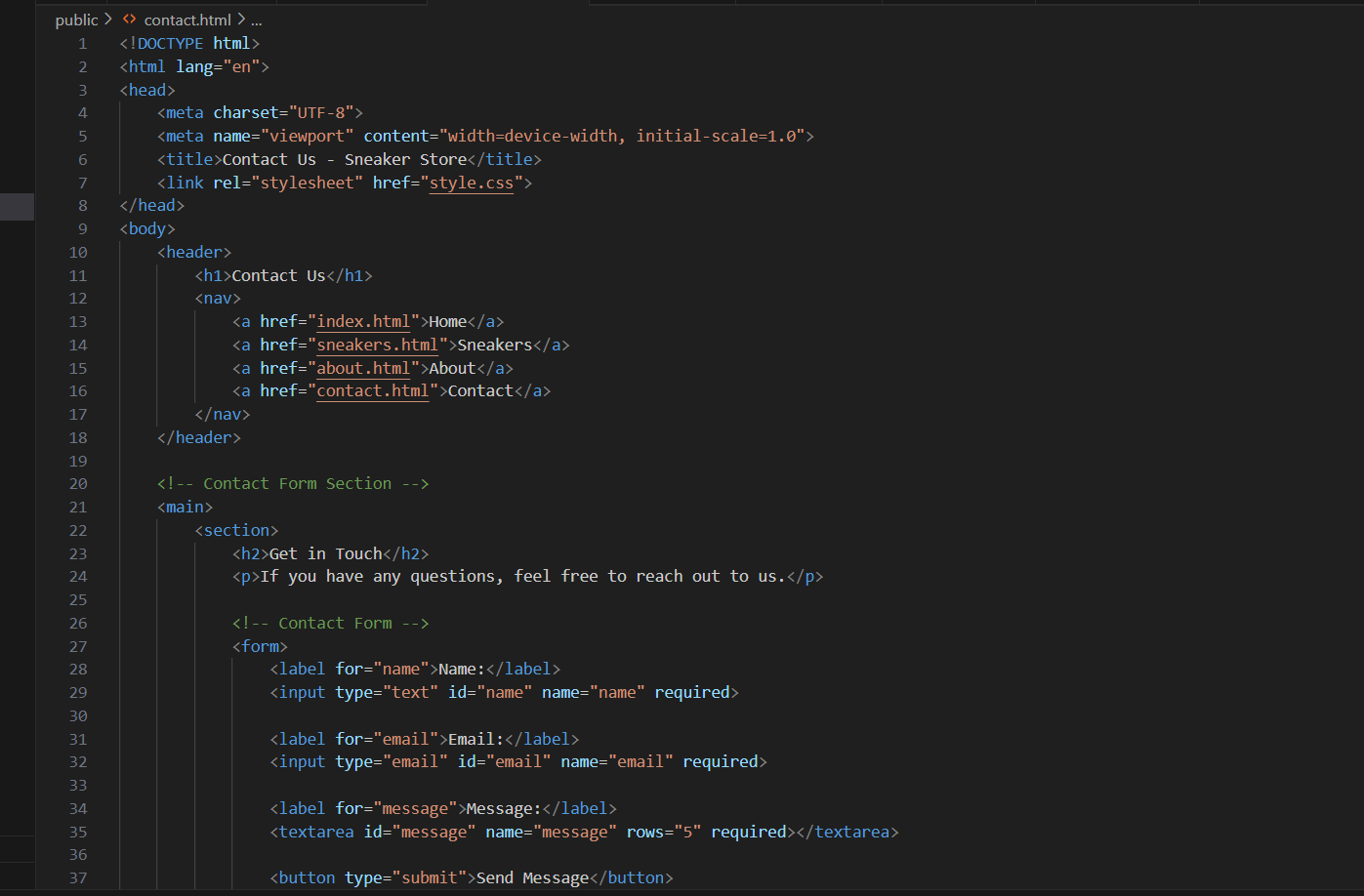


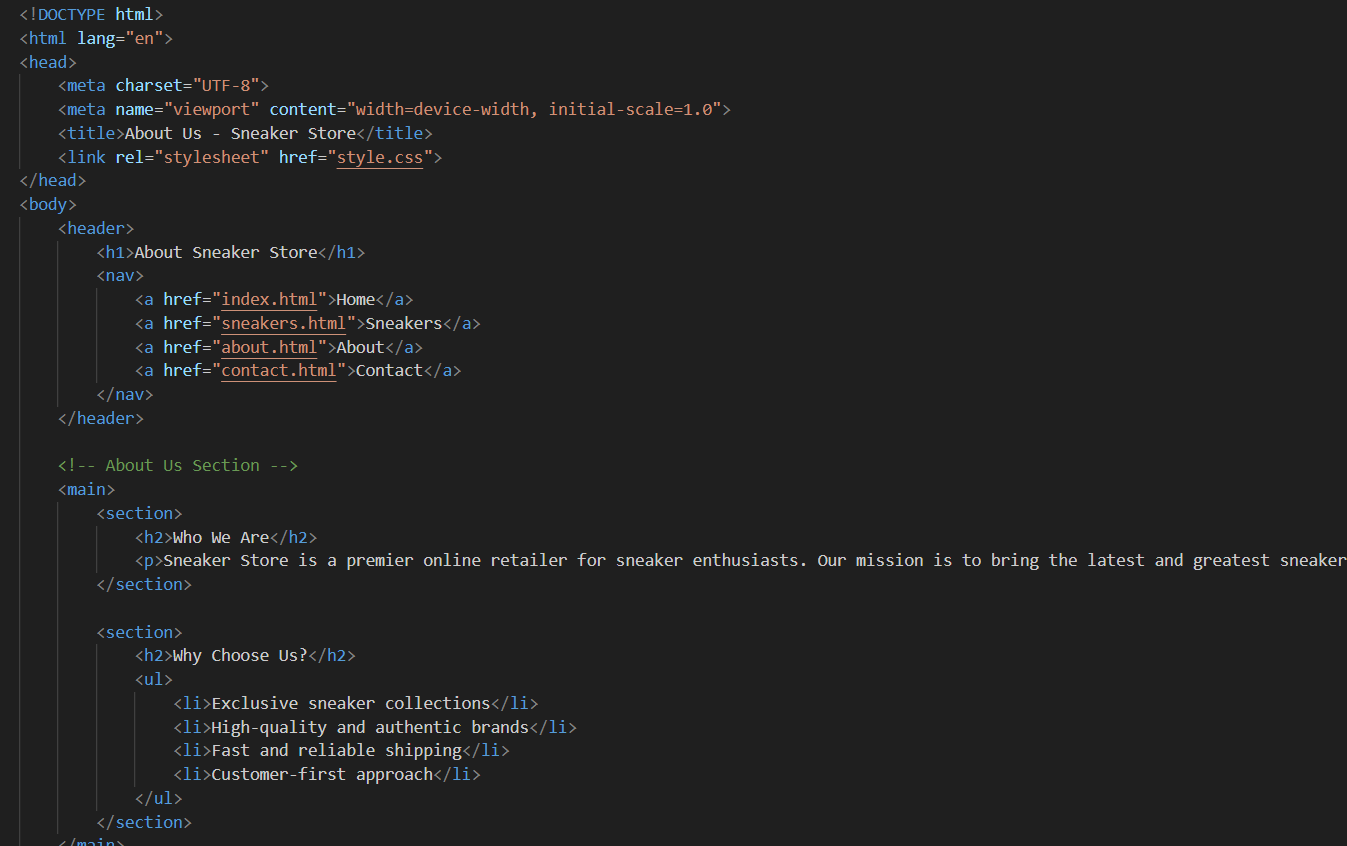
**Step 7: Create Views (HTML Pages)**

* **index.html** (Home Page):
  + Displays sneaker listings fetched from the backend.
  + Links to **CSS** and **JavaScript** files.
* **about.html** and **contact.html**:
  + These pages provide additional information about the store and contact details.
* **Index.html**



* **Contact.html**



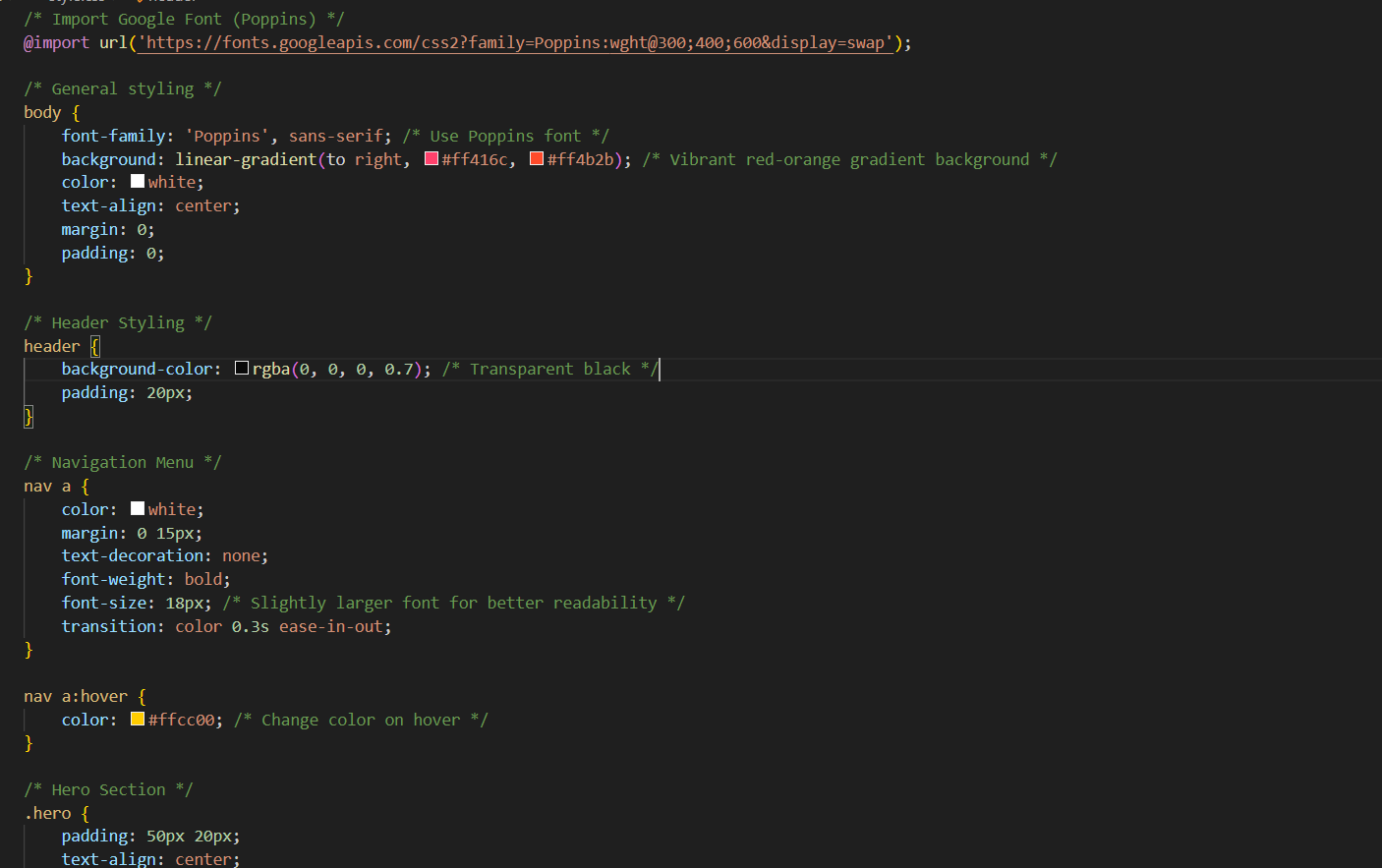
* **About.html** ****

**Step 8: Style the Website (CSS)**

The CSS will define how the pages and elements look.

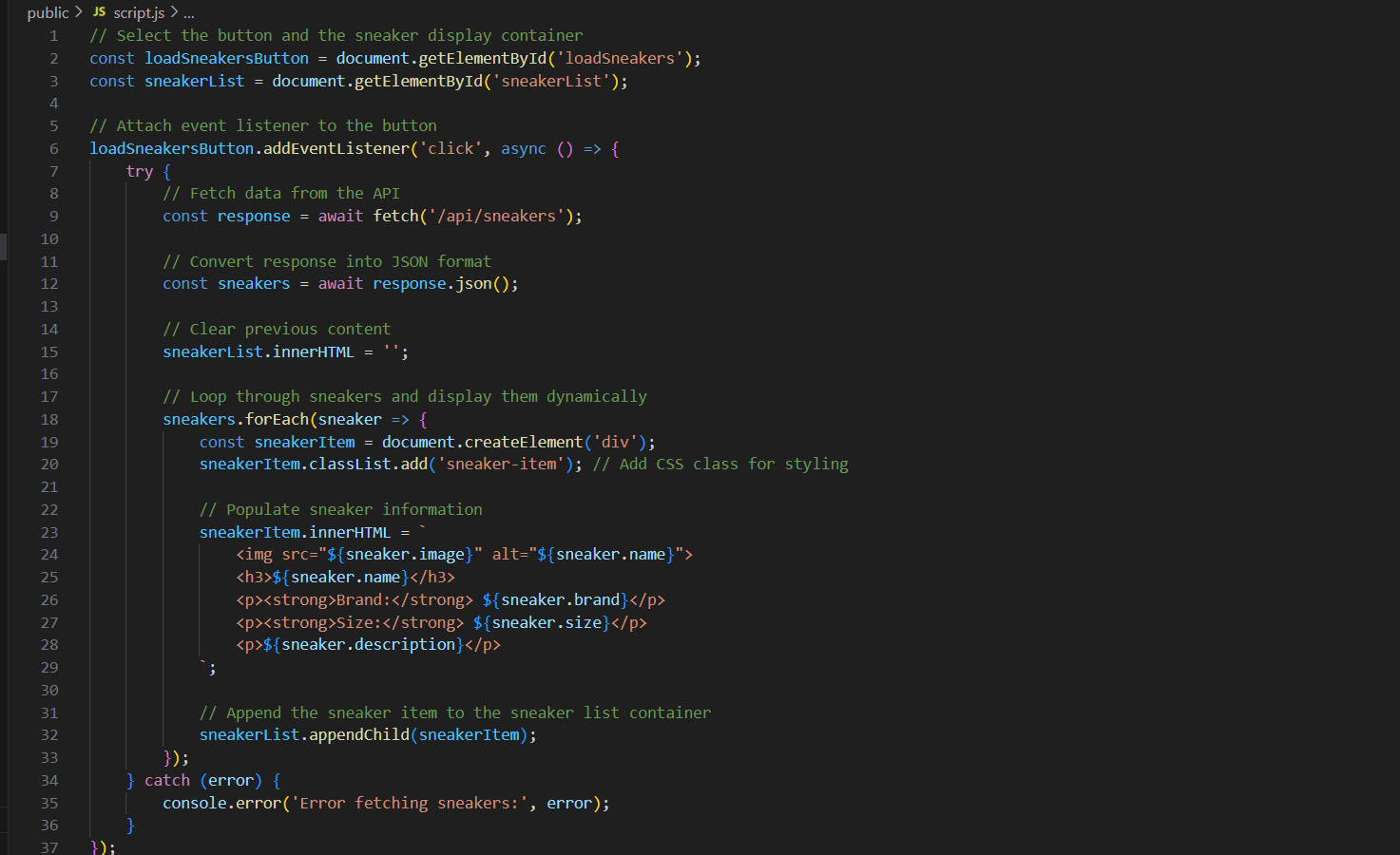
**Action**:

* Inside the public/css/ folder, create a style.css file with the following:



**Step 9: Add Dynamic Sneaker Data in script.js**

In public/script.js, fetch the sneaker data from the backend and dynamically display it on the homepage:

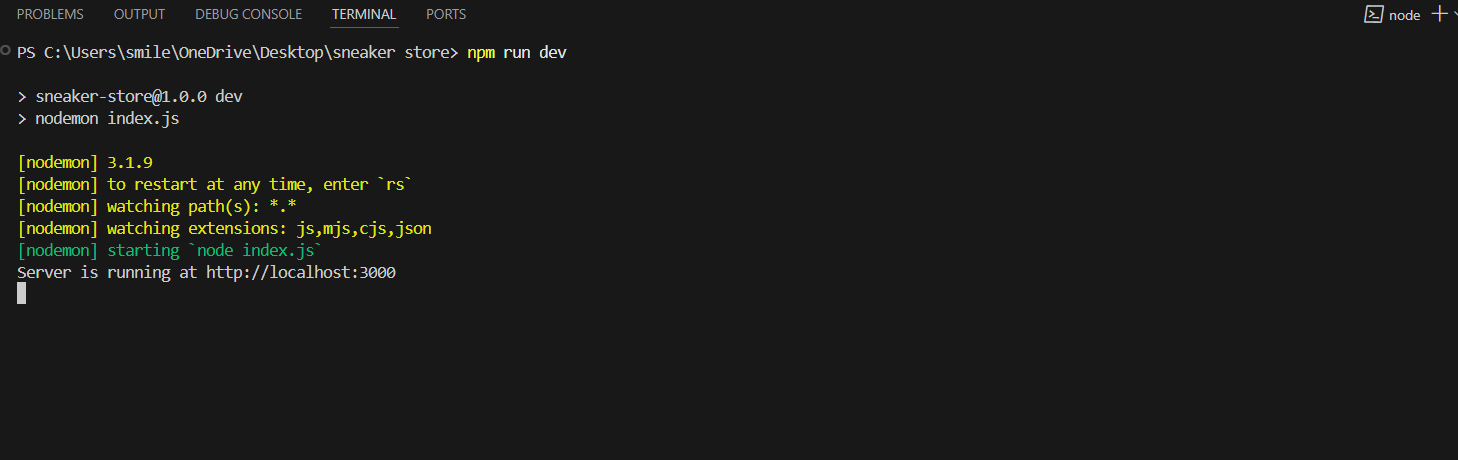


**Step 10: Run the project**

Start the server using:

**npm run dev**

* **Open your browser and go to http://localhost:3000 to see the app running.**

****

Step 11: **Output Images:** 