Microservice_And_Articeture > JS script.js > loadProducts

```
// For Fedora: use the local JSON Server endpoint
// Make sure you've started JSON Server with:
// npx json-server --watch db.json --port 3000
const apiUrl = 'http://localhost:3000/products';

// If using a cloud API (AWS, Azure, Heroku), replace the above line with:
// const apiUrl = 'https://your-cloud-api-endpoint.com/products';

async function loadProducts() {
  try {
    // Fetch data from the backend microservice
    const response = await fetch(apiUrl);

    // Check if the response is okay
    if (!response.ok) throw new Error('Network response not ok');

    // Convert the fetched data to JSON
    const products = await response.json();

    // Get the container div in index.html
    const container = document.getElementById('product-container');
    container.innerHTML = ''; // Clear old content before adding new

    // Loop through products and create HTML cards
    products.forEach(product => {
      const card = document.createElement('div');
      card.className = 'product-card';
      card.innerHTML = `
        <h3>${product.title}</h3>
        <p>Price: $${product.price}</p>
      `;
      container.appendChild(card);
    });-

  } catch (error) {
    // Log any errors (e.g., server not running, wrong URL)
    console.error('Error loading products:', error);
  }
}

// Run this function automatically when the page loads
window.onload = loadProducts;
```

croservice_And_Articeture > # style.css > ...

```css
body {
  font-family: Arial, sans-serif;
  margin: 20px;
}

#product-container {
  display: flex;
  gap: 15px;
  flex-wrap: wrap;
}

.product-card {
  background-color: #f0f0f0;
  border-radius: 5px;
  padding: 10px;
  width: 200px;
  box-shadow: 1px 1px 5px #ccc;
}
```

Microservice_And_Architecture > <> index.html > ...

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <title>Microservice Client</title>
  <link rel="stylesheet" href="style.css" />
</head>
<body>
  <h1>Product List</h1>
  <div id="product-container"></div>
  <script src="script.js"></script>
</body>
</html>
```

```
bash: json: command not found...
shubham@fedora:~/microservice-client$ nano db.json
shubham@fedora:~/microservice-client$ json-server --watch db.json --port 3000
--watch/-w can be omitted, JSON Server 1+ watches for file changes by default
node:events:496
      throw er; // Unhandled 'error' event
      ^

Error: listen EADDRINUSE: address already in use :::3000
    at Server.setupListenHandle [as _listen2] (node:net:1940:16)
    at listenInCluster (node:net:1997:12)
    at Server.listen (node:net:2102:7)
    at App.listen (file:///usr/local/lib/node_modules/json-server/node_modules/@tinyhttp/app/dist/app.js
310:58)
    at file:///usr/local/lib/node_modules/json-server/lib/bin.js:132:5
Emitted 'error' event on Server instance at:
    at emitErrorNT (node:net:1976:8)
    at process.processTicksAndRejections (node:internal/process/task_queues:90:21) {
  code: 'EADDRINUSE',
  errno: -98,
  syscall: 'listen',
  address: '::',
  port: 3000
}

Node.js v22.17.1
shubham@fedora:~/microservice-client$ ^C
shubham@fedora:~/microservice-client$ sudo lsof -i :3000
COMMAND  PID    USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
code    6236 shubham   79u  IPv4  67616      0t0  TCP localhost:hbci (LISTEN)
shubham@fedora:~/microservice-client$ kill -9 6236
shubham@fedora:~/microservice-client$ json-server --watch db.json --port 3001
--watch/-w can be omitted, JSON Server 1+ watches for file changes by default
JSON Server started on PORT :3001
Press CTRL-C to stop
Watching db.json...

♡( ◡‿◡ )

Index:
http://localhost:3001/

Static files:
Serving ./public directory if it exists

Endpoints:
http://localhost:3001/users
http://localhost:3001/orders
```