**1. Create the Project**

mkdir my-api

cd my-api

npm init -y

npm install express

**2. Create db.json**

Put your existing data in a file named db.json:

{

"products": [

{

"id": "1",

"name": "React Book",

"description": "Great boo about React.",

"price": "$120",

"stock": 10

},

{

"id": "2",

"name": "ES6 Book",

"description": "The basis of React!",

"price": "$80",

"stock": 20

},

{

"id": "3229",

"name": "AJ",

"description": "as",

"price": "1",

"stock": "1"

}

]

}

**3. Create server.js**

const express = require('express');

const fs = require('fs');

const app = express();

const PORT = 3000;

app.use(express.json());

// Helper to read db.json

function readDB() {

const data = fs.readFileSync('db.json');

return JSON.parse(data);

}

// GET all products

app.get('/products', (req, res) => {

const db = readDB();

res.json(db.products);

});

// GET a product by ID

app.get('/products/:id', (req, res) => {

const db = readDB();

const product = db.products.find(p => p.id === req.params.id);

if (product) res.json(product);

else res.status(404).json({ error: 'Product not found' });

});

// POST a new product

app.post('/products', (req, res) => {

const db = readDB();

const newProduct = req.body;

db.products.push(newProduct);

fs.writeFileSync('db.json', JSON.stringify(db, null, 2));

res.status(201).json(newProduct);

});

// PUT update a product

app.put('/products/:id', (req, res) => {

const db = readDB();

const index = db.products.findIndex(p => p.id === req.params.id);

if (index === -1) return res.status(404).json({ error: 'Not found' });

db.products[index] = { ...db.products[index], ...req.body };

fs.writeFileSync('db.json', JSON.stringify(db, null, 2));

res.json(db.products[index]);

});

// DELETE a product

app.delete('/products/:id', (req, res) => {

const db = readDB();

const newProducts = db.products.filter(p => p.id !== req.params.id);

db.products = newProducts;

fs.writeFileSync('db.json', JSON.stringify(db, null, 2));

res.status(204).send();

});

app.listen(PORT, () => {

console.log(`Server running at http://localhost:${PORT}`);

});

**4. Run It**

node server.js

Visit: <http://localhost:3000/products>

Once your Express server is running with the /products endpoints, you can fetch data in your React app like this:

**Use /products Endpoint in React**

**1. Start the Server**

First, ensure your Express server is running:

node server.js

It will be available at:  
http://localhost:3000/products

**2. In React – Install Axios (optional but recommended)**

npm install axios

**3. Fetch Data in a React Component**

**Example: Products.jsx**

import React, { useEffect, useState } from 'react';

import axios from 'axios';

function Products() {

const [products, setProducts] = useState([]);

useEffect(() => {

// Make GET request to your local Express endpoint

axios.get('http://localhost:3000/products')

.then(response => {

setProducts(response.data);

})

.catch(error => {

console.error('Error fetching products:', error);

});

}, []);

return (

<div>

<h1>Product List</h1>

{products.length === 0 ? (

<p>Loading...</p>

) : (

<ul>

{products.map(product => (

<li key={product.id}>

<strong>{product.name}</strong>: {product.description} – {product.price}

</li>

))}

</ul>

)}

</div>

);

}

export default Products;

**4. Use the Component in App**

// App.jsx

import React from 'react';

import Products from './Products';

function App() {

return (

<div>

<Products />

</div>

);

}

export default App;

**CORS Note**

If you're getting a CORS error (common in local dev), add this to your server.js:

npm install cors

Then update the server code:

const cors = require('cors');

app.use(cors());