مشروع E-commerce — Implementation (Source Code & Execution)

الملخص: هذا مستودع تعليمي/نمطي لمشروع متجر إلكتروني بسيط (Fullstack) مكوّن من: - Backend: Node.js + Express + SQLite (ملف `database.sqlite`) — واجهة REST API للمنتجات والسلة. - Frontend: React (Vite) — عرض المنتجات، صفحة منتج، عربة تسوّق، الدفع الوهمي.

الهدف: كود جاهز للتشغيل محليًا، مع ملفات كاملة وتعليمات تشغيل ودليل لرفع المشروع على GitHub.

بنية المشروع (Tree)

```
ecommerce-project/
├─ backend/
│  ├─ package.json
│  ├─ server.js
│  ├─ db.js
│  ├─ routes/
│  │  └─ products.js
│  ├─ seed.js
│  └─ README.md
├─ frontend/
│  ├─ package.json
│  ├─ index.html
│  └─ src/
│     ├─ main.jsx
│     ├─ App.jsx
│     ├─ api.js
│     ├─ components/
│     │  ├─ ProductList.jsx
│     │  ├─ ProductCard.jsx
│     │  ├─ ProductPage.jsx
│     │  └─ Cart.jsx
│     └─ styles.css
└─ README.md
```

---

1) Backend — الملفات

backend/package.json

```
{
 "name": "ecommerce-backend",
 "version": "1.0.0",
```

```
  "main": "server.js",
  "scripts": {
    "start": "node server.js",
    "seed": "node seed.js"
  },
  "dependencies": {
    "cors": "^2.8.5",
    "express": "^4.18.2",
    "sqlite3": "^5.1.6"
  }
}
```

---

backend/db.js

```
const sqlite3 = require('sqlite3').verbose();
const path = require('path');
const dbPath = path.join(__dirname, 'database.sqlite');

const db = new sqlite3.Database(dbPath);

// Initialize tables if not exist
db.serialize(() => {
  db.run(`
    CREATE TABLE IF NOT EXISTS products (
      id INTEGER PRIMARY KEY AUTOINCREMENT,
      title TEXT,
      description TEXT,
      price REAL,
      image TEXT,
      stock INTEGER
    )
  `);
});

module.exports = db;
```

---

backend/seed.js

```
const db = require('./db');

const products = [
  { title: 'T-Shirt Cotton', description: 'Comfortable cotton T-Shirt', price:
19.99, image: 'https://picsum.photos/seed/1/400/300', stock: 50 },
  { title: 'Sneakers', description: 'Stylish running shoes', price: 79.9, image:
'https://picsum.photos/seed/2/400/300', stock: 20 },
  { title: 'Jeans', description: 'Slim fit jeans', price: 49.5, image: 'https://
```

```
picsum.photos/seed/3/400/300', stock: 30 },
 { title: 'Leather Wallet', description: 'Genuine leather wallet', price: 29.0,
image: 'https://picsum.photos/seed/4/400/300', stock: 100 }
];

db.serialize(() => {
 const stmt = db.prepare('INSERT INTO products (title, description, price,
image, stock) VALUES (?,?,?,?,?)');
 products.forEach(p => stmt.run(p.title, p.description, p.price, p.image,
p.stock));
 stmt.finalize(() => {
  console.log('Seeding completed.');
  db.close();
 });
});
```

---

backend/routes/products.js

```
const express = require('express');
const router = express.Router();
const db = require('../db');

// GET /api/products
router.get('/', (req, res) => {
 db.all('SELECT * FROM products', (err, rows) => {
  if (err) return res.status(500).json({ error: err.message });
  res.json(rows);
 });
});

// GET /api/products/:id
router.get('/:id', (req, res) => {
 const id = req.params.id;
 db.get('SELECT * FROM products WHERE id = ?', [id], (err, row) => {
  if (err) return res.status(500).json({ error: err.message });
  if (!row) return res.status(404).json({ error: 'Product not found' });
  res.json(row);
 });
});

module.exports = router;
```

---

backend/server.js

```
const express = require('express');
const cors = require('cors');
```

```
const productsRouter = require('./routes/products');

const app = express();
app.use(cors());
app.use(express.json());

app.use('/api/products', productsRouter);

const PORT = process.env.PORT || 4000;
app.listen(PORT, () => console.log(`Backend running on http://localhost:${PORT}
`));
```

---

backend/README.md

```
cd backend
npm install
npm run seed    # creates and seeds database.sqlite
npm start       # runs server on port 4000
```

---

2) Frontend — الملفات

frontend/package.json

```
{
 "name": "ecommerce-frontend",
 "version": "1.0.0",
 "private": true,
 "scripts": {
  "dev": "vite",
  "build": "vite build",
  "preview": "vite preview"
 },
 "dependencies": {
  "react": "^18.2.0",
  "react-dom": "^18.2.0",
  "react-router-dom": "^6.14.1"
 },
 "devDependencies": {
  "vite": "^5.0.0"
 }
}
```

---

frontend/index.html

```
<!doctype html>
<html>
 <head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>E-commerce Demo</title>
 </head>
 <body>
  <div id="root"></div>
  <script type="module" src="/src/main.jsx"></script>
 </body>
</html>
```

---

frontend/src/main.jsx

```
import React from 'react'
import { createRoot } from 'react-dom/client'
import { BrowserRouter, Routes, Route } from 'react-router-dom'
import App from './App'
import './styles.css'

createRoot(document.getElementById('root')).render(
 <React.StrictMode>
  <BrowserRouter>
   <Routes>
    <Route path="/*" element={<App/>} />
   </Routes>
  </BrowserRouter>
 </React.StrictMode>
)
```

---

frontend/src/api.js

```
const API_BASE = import.meta.env.VITE_API_BASE || 'http://localhost:4000';

export async function fetchProducts() {
 const res = await fetch(`${API_BASE}/api/products`);
 return res.json();
}

export async function fetchProduct(id) {
 const res = await fetch(`${API_BASE}/api/products/${id}`);
```

```
  return res.json();
}
```

---

frontend/src/App.jsx

```
import React, { useState } from 'react'
import { Routes, Route, Link } from 'react-router-dom'
import ProductList from './components/ProductList'
import ProductPage from './components/ProductPage'
import Cart from './components/Cart'

export default function App(){
 const [cart, setCart] = useState([]);

 const addToCart = (product) => {
  setCart(prev => {
   const found = prev.find(p=>p.id === product.id);
   if(found) return prev.map(p=> p.id===product.id ? {...p, qty: p.qty+1} : p);
   return [...prev, {...product, qty:1}];
  })
 }

 const updateQty = (id, qty) => {
  setCart(prev => prev.map(p=> p.id===id ? {...p, qty} : p));
 }

 return (
  <div className="app">
   <header>
    <Link to="/"><h1>Mini E-commerce</h1></Link>
    <Link to="/cart">Cart ({cart.reduce((s,c)=>s+c.qty,0)})</Link>
   </header>

   <main>
    <Routes>
     <Route path="/" element={<ProductList addToCart={addToCart}/> } />
     <Route path="/product/:id" element={<ProductPage addToCart={addToCart}/
> } />
     <Route path="/cart" element={<Cart cart={cart} updateQty={updateQty}/> } /
>
    </Routes>
   </main>
  </div>
 )
}
```

---

frontend/src/components/ProductList.jsx

```
import React, { useEffect, useState } from 'react'
import { fetchProducts } from '../api'
import ProductCard from './ProductCard'

export default function ProductList({ addToCart }){
 const [products, setProducts] = useState([]);
 useEffect(()=>{ fetchProducts().then(setProducts) },[]);

 return (
  <div className="product-list">
   {products.map(p => (
    <ProductCard key={p.id} p={p} addToCart={addToCart} />
   ))}
  </div>
 )
}
```

---

frontend/src/components/ProductCard.jsx

```
import React from 'react'
import { Link } from 'react-router-dom'

export default function ProductCard({ p, addToCart }){
 return (
  <div className="card">
   <img src={p.image} alt={p.title} />
   <h3>{p.title}</h3>
   <p>{p.description}</p>
   <div className="card-foot">
    <strong>${p.price.toFixed(2)}</strong>
    <div>
     <button onClick={()=>addToCart(p)}>Add</button>
     <Link to={`/product/${p.id}`}>View</Link>
    </div>
   </div>
  </div>
 )
}
```

---

frontend/src/components/ProductPage.jsx

```
import React, { useEffect, useState } from 'react'
import { useParams } from 'react-router-dom'
```

```
import { fetchProduct } from '../api'

export default function ProductPage({ addToCart }){
 const { id } = useParams();
 const [product, setProduct] = useState(null);

 useEffect(()=>{ fetchProduct(id).then(setProduct) },[id]);
 if(!product) return <div>Loading...</div>

 return (
  <div className="product-page">
   <img src={product.image} alt=""/>
   <div>
    <h2>{product.title}</h2>
    <p>{product.description}</p>
    <h3>${product.price.toFixed(2)}</h3>
    <button onClick={()=>addToCart(product)}>Add to cart</button>
   </div>
  </div>
 )
}
```

---

frontend/src/components/Cart.jsx

```
import React from 'react'

export default function Cart({ cart, updateQty }){
 const total = cart.reduce((s,p)=> s + p.price * p.qty, 0);

 return (
  <div>
   <h2>Your Cart</h2>
   {cart.length===0 && <p>Cart is empty</p>}
   {cart.map(p=> (
    <div key={p.id} className="cart-row">
     <img src={p.image} alt="" />
     <div>
      <h4>{p.title}</h4>
      <p>${p.price.toFixed(2)}</p>
      <input type="number" value={p.qty} min={1} onChange={(e)=>
updateQty(p.id, Number(e.target.value))} />
     </div>
    </div>
   ))}

   <h3>Total: ${total.toFixed(2)}</h3>
   <button onClick={()=> alert('Checkout dummy — integrate payment gateway')}
>Checkout</button>
```

```
    </div>
  )
}
```

---

frontend/src/styles.css

```
body{ font-family: Arial, sans-serif; margin:0; padding:0; }
header{ display:flex; justify-content:space-between; padding:16px 24px;
background:#111; color:white }
header a{ color:white; text-decoration:none }
.app main{ padding:24px }
.product-list{ display:grid; grid-template-columns: repeat(auto-fit,
minmax(220px,1fr)); gap:16px }
.card{ border:1px solid #eee; padding:12px; border-radius:8px }
.card img{ width:100%; height:140px; object-fit:cover; border-radius:6px }
.card-foot{ display:flex; justify-content:space-between; align-items:center;
margin-top:8px }
.product-page{ display:flex; gap:20px }
.product-page img{ width:400px; height:300px; object-fit:cover }
.cart-row{ display:flex; gap:12px; align-items:center; border-bottom:1px solid
#eee; padding:8px 0 }
.cart-row img{ width:80px; height:60px; object-fit:cover }
```

---

3) Root README.md (شرح التشغيل & رفع على GitHub)

````
# Mini E-commerce — Demo

## تشغيل محلي

1. Backend
```bash
cd backend
npm install
npm run seed    # seeds database.sqlite
npm start
````

سيعمل السيرفر على http://localhost:4000

1. Frontend

```
cd frontend
npm install
npm run dev
```
```

سيعمل الواجهة على http://localhost:5173

تأكد أن `VITE_API_BASE` إذا أردت تغييره، ضعه في ملف .env في مجلد frontend.

رفع للمستودع على GitHub

1. أنشئ مستودعًا جديدًا في حسابك على GitHub، ثم في جذر المشروع شغّل:

```
git init
git add .
git commit -m "Initial commit: mini ecommerce"
git branch -M main
git remote add origin https://github.com/<YOUR_USERNAME>/<REPO_NAME>.git
git push -u origin main
```

2. اضف ملف (اختياري) .gitignore وادراج node_modules و database.sqlite

```
node_modules
frontend/node_modules
backend/node_modules
backend/database.sqlite
```

---

4) ملاحظات وامتدادات مستقبلية (اقتراحات)

- ربط بقاعدة MongoDB أو Postgres للبيئات الحقيقية.
- إضافة نظام مستخدمين (Auth) وطلبات (orders).
- ربط بوابة دفع حقيقية (Stripe/PayPal).
- رفع الواجهة على Vercel والـ backend على Render/Heroku.

انتهى — هذا المشروع قابل للتنفيذ محليًا. لو عايز أعمل لك: رفع فعلي على GitHub (أو (ارشدك خطوة بخطوة) جاهز للتحميل، قولّي وانا أجهزّه أزوّدك بملف zip.