**💳 Demo Banking App by** [**PerfTesters.com**](https://perftesters.com/)

A full-stack demo banking web application designed for developers and performance testers to practice core backend functionality such as user authentication, payments, transaction handling, and even direct SQL debugging.

**🚀 Features**

* 🔐 **User Authentication** with JWT (Login & Register)
* 💰 **Check Account Balance**
* 🔁 **Send Payments** to other users (by User ID)
* 📜 **View Transaction History**
* 🐞 **SQL Debugger** – Admin tool for running raw SELECT queries
* 🧩 **Lightweight SQLite Database** (auto-generated)
* 💻 **Responsive Frontend UI** with Tailwind CSS

**🛠️ Tech Stack**

* **Frontend**: HTML + Tailwind CSS + Vanilla JavaScript
* **Backend**: Node.js (ES Modules) + Express.js + JWT
* **Database**: SQLite (banking.db auto-created on first run)

**📁 Project Structure**

demo-banking-app/

├── public/

│ └── index.html # Frontend UI

├── middleware/

│ └── verifyToken.js # JWT verification middleware

├── models/

│ └── transactionModel.js# DB methods for transactions

├── server.mjs # Main Express server

├── banking.db # SQLite DB (auto-generated)

├── package.json

└── README.md

**📦 Setup Guide**

**1. Install Dependencies (PowerShell)**

npm install express dotenv jsonwebtoken sqlite sqlite3

**2. Start the Server**

node server.mjs

**3. Open in Browser**

Visit: <http://localhost:3000/index.html>

**🧪 How to Use**

**✅ Register**

Create a new account using a unique username and password.

**🔐 Login**

Get authenticated and receive a token (used automatically in frontend).

**💼 Check Balance**

View your current account balance.

**💸 Make a Payment**

Send money to another user using their **User ID**.

**📃 Transaction History**

Review your transaction records – both incoming and outgoing.

**🐞 SQL Debugger (Admin Only)**

Allows execution of raw **SELECT** SQL queries for debugging.

⚠️ Only SELECT queries are permitted for security.

**Example Queries:**

SELECT \* FROM users;

SELECT \* FROM transactions WHERE from\_user\_id = 1 OR to\_user\_id = 1;

**🔐 Authentication Info**

All secure endpoints (/balance, /payment, /transactions, /execute-query) require a **JWT token**.

Include this token in the request header:

Authorization: Bearer <your\_token>

**⚙️ Developer Notes**

* banking.db is automatically created on first server start.
* Replace the hardcoded JWT secret in server.mjs for production environments.
* UI and routes can be extended for additional banking features.

Happy testing!  
🧪 Made for learning and simulation by [PerfTesters.com](https://perftesters.com/)