ChatGTP: Would you like me to give you a complete working example project (folder structure + files) you can copy-paste and run? Or help add purchases or categories next?

Yes, please!

Awesome! 🙌 Here's a **complete beginner-friendly starter project** that connects **Node.js** with **PostgreSQL** using **pg**, builds a **REST API**, and lets you test it easily using **REST Client** or **Postman**.

**📁 Project Structure**

web-template-store/

├── db.js

├── index.js

├── .env

├── create\_tables.sql

├── seed.sql

├── test.rest

├── package.json

└── README.md

**🔨 Step-by-Step Setup**

**1. Create the Project Directory**

mkdir web-template-store

cd web-template-store

npm init -y

npm install express pg dotenv

**2. Create .env File**

DATABASE\_URL=postgres://your\_user:your\_password@localhost:5432/your\_database

Replace with your real PostgreSQL user, password, and database name.

**3. Create db.js**

// db.js

const { Pool } = require("pg");

require("dotenv").config();

const pool = new Pool({

connectionString: process.env.DATABASE\_URL,

});

module.exports = pool;

**4. Create create\_tables.sql**

CREATE TABLE IF NOT EXISTS users (

id SERIAL PRIMARY KEY,

username VARCHAR(100) NOT NULL,

email VARCHAR(255) UNIQUE NOT NULL,

created\_at TIMESTAMP DEFAULT NOW()

);

CREATE TABLE IF NOT EXISTS templates (

id SERIAL PRIMARY KEY,

name VARCHAR(150) NOT NULL,

description TEXT,

price NUMERIC(10,2),

created\_at TIMESTAMP DEFAULT NOW()

);

Run this using psql:

psql -U your\_user -d your\_database -f create\_tables.sql

**5. Create seed.sql**

INSERT INTO users (username, email)

VALUES ('johndoe', 'john@example.com');

INSERT INTO templates (name, description, price)

VALUES

('Portfolio Template', 'Clean layout for portfolios', 19.99),

('Business Template', 'Corporate responsive design', 24.99);

Run it:

psql -U your\_user -d your\_database -f seed.sql

**6. Create index.js (Express App)**

// index.js

const express = require("express");

const app = express();

const db = require("./db");

app.use(express.json());

// GET all templates

app.get("/api/templates", async (req, res) => {

try {

const result = await db.query("SELECT \* FROM templates");

res.json(result.rows);

} catch (err) {

res.status(500).json({ error: "Internal server error" });

}

});

// POST a new template

app.post("/api/templates", async (req, res) => {

const { name, description, price } = req.body;

try {

const result = await db.query(

"INSERT INTO templates (name, description, price) VALUES ($1, $2, $3) RETURNING \*",

[name, description, price]

);

res.status(201).json(result.rows[0]);

} catch (err) {

res.status(500).json({ error: "Failed to create template" });

}

});

app.listen(3000, () => {

console.log("🚀 Server running at http://localhost:3000");

});

**7. Create test.rest (For VS Code REST Client)**

### Get all templates

GET http://localhost:3000/api/templates

###

### Create a new template

POST http://localhost:3000/api/templates

Content-Type: application/json

{

"name": "Blog Template",

"description": "Minimal clean blog layout",

"price": 14.99

}

**8. Add Scripts to package.json (optional)**

"scripts": {

"start": "node index.js"

}

Run it with:

npm start

**✅ You’re Ready!**

Now your app is working, your REST API is live at http://localhost:3000, and you can:

* View all templates
* Add a new template
* Expand later with users, purchases, reviews

**💡 Next Steps (Optional)**

Let me know if you'd like help with any of these next:

* Add a **purchases** table and endpoint
* Add **users registration/login**
* Add **categories** or **template previews**
* Add **validation**
* Deploy to **Render** or **Railway**

Would you like me to zip this up into a GitHub starter project or generate more features?

Great question! 😄 Here's how you can pick up right where we left off next time:

✅ Just say something like:

“Hey, can we continue from the web templates