New chat
Search chats
Library
Sora
GPTs

Chats
Node Express SQL Setup
File Upload Preview Guide
Image upload with preview
Image Upload with Preview
React Video Animation
Create table with popover
Dynamic NavItem Icons
Setting Multiple Routes
Property Management Dashbo...
Document Upload Integration
Add View Icon
Form Validation and Enhancem...
Table UI Enhancements
Admin User Form Creation
Reusable Table Component
User List Table MUI

create a new demo project from node + exprees js in creat get, post, put, delete api and profrsnal code formet all step by step creat and databes sql set all setup profesnaly and code formeting

Updated saved memory

Great! Let's create a **professional Node.js + Express.js demo project** with **GET, POST, PUT, DELETE** APIs and an **SQL (MySQL)** database integration. I'll guide you step-by-step with clean and scalable folder structure, proper formatting, and best practices.

✅ Step 1: Project Setup

1. Create Project Folder

bash Copy Edit

```bash
mkdir express-sql-demo
cd express-sql-demo
```

2. Initialize `package.json`

bash Copy Edit

```bash
npm init -y
```

3. Install Required Packages

Ask anything

```bash
npm install express mysql2 dotenv body-parser cors
npm install --save-dev nodemon
```

4. Create Folder Structure

pgsql Copy Edit

```
express-sql-demo/
│
├── config/
│   └── db.config.js
│
├── controllers/
│   └── user.controller.js
│
├── routes/
│   └── user.routes.js
│
├── models/
│   └── user.model.js
│
├── .env
├── server.js
├── package.json
└── README.md
```

✅ Step 2: Database Setup (MySQL)

Ask anything

sql Copy Edit

```sql
CREATE DATABASE express_demo;

USE express_demo;

CREATE TABLE users (
  id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(100),
  email VARCHAR(100) UNIQUE,
  age INT
);
```

✅ Step 3: `.env` File

ini Copy Edit

```ini
PORT=5000
DB_HOST=localhost
DB_USER=root
DB_PASSWORD=yourpassword
DB_NAME=express_demo
```

✅ Step 4: Create Config File

Ask anything

```js
const mysql = require("mysql2");
require("dotenv").config();

const db = mysql.createConnection({
  host: process.env.DB_HOST,
  user: process.env.DB_USER,
  password: process.env.DB_PASSWORD,
  database: process.env.DB_NAME,
});

db.connect((err) => {
  if (err) throw err;
  console.log("✅ Connected to MySQL database.");
});

module.exports = db;
```

✅ Step 5: Model

`models/user.model.js`

```js
const db = require("../config/db.config");

const User = {
  },

  getById: (id, callback) => {
    db.query("SELECT * FROM users WHERE id = ?", [id], callback);
  },

  create: (data, callback) => {
    db.query("INSERT INTO users SET ?", data, callback);
  },

  update: (id, data, callback) => {
    db.query("UPDATE users SET ? WHERE id = ?", [data, id], callback);
  },

  delete: (id, callback) => {
    db.query("DELETE FROM users WHERE id = ?", [id], callback);
  },
};

module.exports = User;
```

✅ Step 6: Controller

`controllers/user.controller.js`

```js
const User = require("../models/user.model");

    if (err) return res.status(500).send(err);
    res.send(data);
  });
};

exports.getUserById = (req, res) => {
  User.getById(req.params.id, (err, data) => {
    if (err) return res.status(500).send(err);
    res.send(data[0]);
  });
};

exports.createUser = (req, res) => {
  User.create(req.body, (err, result) => {
    if (err) return res.status(500).send(err);
    res.status(201).send({ id: result.insertId, ...req.body });
  });
};

exports.updateUser = (req, res) => {
  User.update(req.params.id, req.body, (err) => {
    if (err) return res.status(500).send(err);
    res.send({ id: req.params.id, ...req.body });
  });
};

exports.deleteUser = (req, res) => {
  User.delete(req.params.id, (err) => {
    if (err) return res.status(500).send(err);
    res.send({ message: "User deleted successf" });
  });
```

✅ Step 7: Routes

`routes/user.routes.js`

```js
const express = require("express");
const router = express.Router();
const userController = require("../controllers/user.controller");

router.get("/", userController.getAllUsers);
router.get("/:id", userController.getUserById);
router.post("/", userController.createUser);
router.put("/:id", userController.updateUser);
router.delete("/:id", userController.deleteUser);

module.exports = router;
```

✅ Step 8: Entry Point

`server.js`

```js
const express = require("express");
require("dotenv").config();

const app = express();

app.use(cors());
app.use(bodyParser.json());

app.use("/api/users", userRoutes);

const PORT = process.env.PORT || 5000;
app.listen(PORT, () => console.log(`🚀 Server is running on port ${PORT}`));
```

✅ Step 9: `package.json` script

Update `scripts`:

```json
"scripts": {
  "start": "node server.js",
  "dev": "nodemon server.js"
}
```

✅ Step 10: Run Your Project

```bash
npm run dev
```

✅ Sample API Endpoints

Method	Endpoint	Description
GET	`/api/users`	Get all users
GET	`/api/users/:id`	Get user by ID
POST	`/api/users`	Create new user
PUT	`/api/users/:id`	Update user
DELETE	`/api/users/:id`	Delete user

Do you want me to generate this as a downloadable ZIP file for you or need Postman collection for testing APIs?

Ask anything

Tools

Upgrade plan
More access to the best models