**# Next.js + Mongoose + MongoDB**

**## Folder Structure**

```code

database

    |-connection

        |-connection.js

    |-controller

        |-productController.js

        |-userController.js

    |-model

        |-productModel.js

        |-userModel.js

pages

    |-api

        |-products

            |-[productId].js

            |-index.js

        |-users

            |-[userId].js

            |-index.js

```

**## database/connection/connection.js**

```js

const {default: mongoose} = require("mongoose");

mongoose.set('strictQuery', true);

const connectMongo = async () => {

    try {

        const {connection} = await mongoose.connect(process.env.MONGO\_URI);

        if(connection.readyState === 1) {

            console.log("Database Connected!");

        }

    } catch(error) {

        console.log(error.message);

        console.log("Database Not Connected!");

    }

};

export default connectMongo;

```

**## database/controller/productController.js**

```js

import Products from "../model/productModel";

// http://localhost:3000/api/products

export const getProducts = async (req, res) => {

    try {

        const products = await Products.find({});

        if(!products) {

            return res.json({error: 'Data Not Found!'});

        }

        res.json(products);

    } catch(error) {

        res.json({error: 'Error While Fetching Data!'});

    }

};

// Same as userController.js

```

**## database/controller/userController.js**

```js

import Users from "../model/userModel";

// http://localhost:3000/api/users

export const getUsers = async (req, res) => {

    try {

        const users = await Users.find({});

        if(!users) {

            return res.json({error: 'Data Not Found!'});

        }

        return res.json(users);

    } catch(error) {

        return res.json({error: 'Error While Fetching Data!'});

    }

};

// http://localhost:3000/api/users/\_id

export const getUser = async (req, res) => {

    try {

        const {userId} = req.query;

        if(!userId) {

            return res.json({error: 'User Id Not Found!'});

        }

        const user = await Users.findById(userId);

        return res.json(user);

    } catch(error) {

        return res.json({error: 'Error While Fetching Data!'});

    }

};

// http://localhost:3000/api/users

export const postUser = async (req, res) => {

    try {

        const formData = req.body;

        if(!formData) {

            return res.json({error: "Form Data Not Provided!"});

        }

        Users.create(formData, (err, data) => {

            if(!err) {

                return res.json(data);

            }

        });

    } catch(error) {

        return res.json(error);

    }

};

// http://localhost:3000/api/users/?userId=\_id

export const putUser = async (req, res) => {

    try {

        const {userId} = req.query;

        const formData = req.body;

        if(userId && formData) {

            const user = await Users.findByIdAndUpdate(userId, formData);

            return res.json(user);

        }

        return res.json({error: "User Not Selected."});

    } catch(error) {

        return res.json({error: "Error While Updating the Data."});

    }

};

// http://localhost:3000/api/users/?userId=\_id

export const deleteUser = async (req, res) => {

    try {

        const {userId} = req.query;

        if(userId) {

            const user = await Users.findByIdAndDelete(userId);

            return res.json(user);

        }

        return res.json({error: "Error While Deleting Data!"});

    } catch(error) {

        return res.json({error: "Error While Updating the Data."});

    }

};

```

**## database/model/productModel.js**

```js

import {model, models, Schema} from "mongoose";

const productSchema = new Schema(

    {

        name: String,

        price: String

    }

);

const Products = models.product || model('product', productSchema);

export default Products;

```

**## database/model/userModel.js**

```js

import {Schema, models, model} from "mongoose";

const userSchema = new Schema(

    {

        name: String,

        email: String,

        password: String

    }

);

const Users = models.user || model('user', userSchema);

export default Users;

```

**## pages/api/products/index.js**

```js

import connectMongo from "@/database/connection/connection";

import {getProducts} from "@/database/controller/productController";

const handler = async (req, res) => {

    connectMongo().catch(() => res.json({error: "Error in the connection"}));

    const {method} = req;

    switch(method) {

        case 'GET':

            getProducts(req, res);

            break;

        default:

            break;

    }

};

export default handler;

```

**## pages/api/products/[productId].js**

```js

// Same as [userId].js

```

**## pages/api/users/index.js**

```js

import connectMongo from "@/database/connection/connection";

import {deleteUser, getUsers, postUser, putUser} from "@/database/controller/userController";

const handler = async (req, res) => {

    connectMongo().catch(() => res.json({error: "Error in the connection"}));

    const {method} = req;

    switch(method) {

        case 'GET':

            getUsers(req, res);

            break;

        case 'POST':

            postUser(req, res);

            break;

        case 'PUT':

            putUser(req, res);

            break;

        case 'DELETE':

            deleteUser(req, res);

            break;

        default:

            res.json({error: "Notice the method!"});

            break;

    }

};

export default handler;

```

**## pages/api/users/[userId].js**

```js

import connectMongo from "@/database/connection/connection";

import {getUser} from "@/database/controller/userController";

const handler = async (req, res) => {

    connectMongo().catch(() => res.json({error: "Error in the connection"}));

    const {method} = req;

    switch(method) {

        case 'GET':

            getUser(req, res);

            break;

        default:

            break;

    }

};

export default handler;

```