**Node**

1)make folder and open by vs code

Ctrl ~(to open terminal)

2)make api folder (we are gonna make 2 folder api and client)

Go in terminal

Cd api

Npm init -y

Npm i express mysql cors

* npm init -y: Quickly sets up a new Node.js project with default settings.
* npm i express mysql cors: Installs three important packages:
  + express: Helps build web servers and APIs easily.
  + mysql: Allows Node.js to interact with MySQL databases.
  + cors: Ensures secure communication between different domains in your app.

CORS (Cross-Origin Resource Sharing) is used to handle security when your frontend, hosted on one domain, wants to make requests to your backend, hosted on a different domain.

bodyParser is used to parse and handle data sent in the request body, like form submissions or JSON data.(no need to install this as already in express .json

2)new file index.js

**Index.js**

const express = require("express");

const app = express();

const cors = require("cors");

1)import express ,cors

2)initalize express (to start express application) with app

app.use(express.json());

app.use(

cors({ //dont forget ({})

origin: "\*",

})

);

const PORT=8080;

app.listen(PORT, (req, res) => {

console.log(`Server running on port ${PORT}`);

});

Not “ “ or ‘ ‘ but ` ` i,e backtilt

This code sets up a server:

using Express with JSON parsing and CORS configuration, and then starts the server on port 8080.

- `app.use(express.json());`: parse incoming JSON data.

- `app.use(cors({ origin: "\*" }));`: Allows the server to accept requests from any origin (CORS).

- `const PORT=8080;`: Specifies that the server will run on port 8080.

- `app.listen(PORT, (req, res) => {...});`: Starts the server and prints a message saying it's running on port 8080.

In simple terms, it's preparing a server to handle data in a specific format (JSON), allowing requests from anywhere, and starting the server on port 8080.

Run

Node index.js

o/p

Server running on port 8080

**Db.js**

1)import mysql package

const mysql = require("mysql");

* Const mysql: declare a variable
* require("mysql"): The require built-in to import external modules or libraries.. The string "mysql" is the name of the module being imported.

2)const mysqlConnection = mysql.createConnection({ //remenber ({

host: "localhost",

user: "root",

password: "root123",

database: "product",

port: 3306, //no “3306”

});

Dont forget to write values in “ “

Create var->mysql module call createConnection method->({})->hupdp->:” ”,

3)mysqlConnection.connect((err) => { //remenber connect() then (err)

if (err) {

console.log("Error - " + err.message);

}

console.log("Connected to the database!");

});

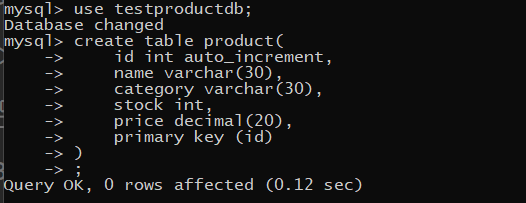
module.exports = mysqlConnection;

Call createConnection method on above variable=>param(err)=>if(err) log error message with delatils=>else print connected in console,log

Exports for to be used by other files

4) in sql





create **sqlCommand.sql** to write sql commands in it

create table product(

id int auto\_increment,//not autoincrement

name varchar(30),

category varchar(30),

stock int,

price decimal(20),

primary key (id)

);

5) **create services** folder and make **productService.js**

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

const add=(req,res)=>{

const {name,price}=req.body;

console.log(req.body);

const str='insert into product(name,price) values("${name}",${price})';

connection.query(str,(err,data,field)=>{

if(err){

console.log(err.message); //alwayys write err.message to show short err msd in terminal otherwise big stack trace

res.status(400).json({success:false});

}

else{

res.status(201).json({success:true,message:"added"});

}

});

};

module.exports={

addProduct

};

adds a product to a database based on the received data, and it responds with success or failure messages.

1)define function to add with parsm req resp

2)const { name, category, price, stock } = req.body;

Extract Req k body se jo files aye h

3)console.log(req.body);:

Logs the received data to the console.

4)insert extrected value in db

const query='insert into product(name,price) values("${name}",${price})';

Write in ‘ ‘

Insert into tb(c1,c2) values(i1,i2)

${}

For varchar “$()”

No ‘ ‘ but `` (backticks)

5)connection.query(query, (err, data, field) => {...});: Executes a SQL query on a database using the query method.

* query: The SQL query to be executed.
* (err, data, field) => {...}: A callback function that handles the results of the query.
  + err: Represents any error that might occur during the query execution.
  + data: Contains the result data returned from the query.
  + field: Contains information about the fields in the result set.

6)if (err) {...}: Checks for errors during the query execution.

* If there's an error, it logs the error message, sends a response with status 400 (bad request), and a JSON object indicating failure.
* If there's no error, it sends a response with status 201 (created) and a JSON object indicating success.

7)module.exports={

addProduct

};

If not written this error in index.js that service required method not found

const {add}=REQUIRE("./service/productService");

ReferenceError: REQUIRE is not defined

**Index.js**

This code defines a function named `getProducts` that fetches data from a MySQL database table named "Product" using a connection established in the `MysqlConnect` module. The retrieved data is then sent as a JSON response with a success message.

- `const connection = require("../DatabaseConnection/MysqlConnect");`: Imports a module that establishes a connection to a MySQL database.

- `const getProducts = (req, res) => {...};`: Defines a function named `getProducts` that takes a request (`req`) and a response (`res`) as parameters.

- `const query = `SELECT \* FROM Product`;`: Defines a SQL query to select all rows from the "Product" table.

- `connection.query(query, (err, data, field) => {...});`: Executes the SQL query using the established database connection. The callback function handles the results of the query.

- `if (err) console.log(err.message);`: Checks for any errors during the query execution and logs the error message if there is an error.

- `res.status(200).json({ message: "success", productData: data });`: If the query is successful, it sends a JSON response with a success message and the retrieved data from the "Product" table.

In summary, this code fetches all products from a MySQL database and sends them as a JSON response with a success message.

In **index.js**

const {add}=require("./service/productService");

app.post("/api/products/add",add);

No ../service but ./service or else module not found error

* 1)Imports the add function from the productService module.

2)call post method of map in that (1param=”uri/” ,2param=func)

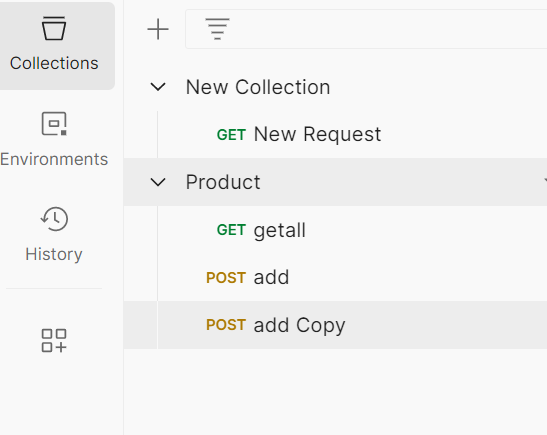
the add function will be called when a POST request is made to this

Nodeman index.js (o/p-Server is running on 8080

coonected to db

Open postman

collections=>+=>force close all=>new collection=>



Put methide name(get.post)

Enter url <http://localhost:8080/api/products/add>

Go in body

Go in raw

{

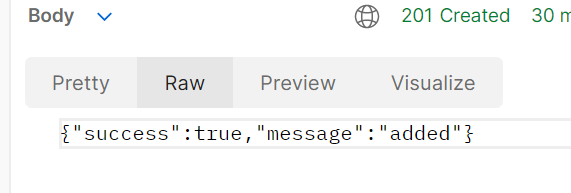
"name":"dsojvh",

"price":547

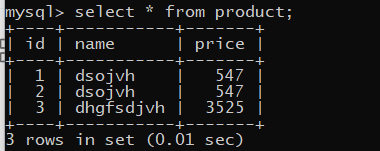
}

Key also in “”

send



CHECK IN DB



**Getall**

const getProduct=(req,res)=>{

const str=`select \* from product`;

connection.query(str,(err,data,field)=>{

if(err){

console.log(err.message);

res.status(400).json({success:false});

}

else{

res.status(200).json({success:true,data});

}

});

};

module.exports={

add,

getProduct

};

Index.js

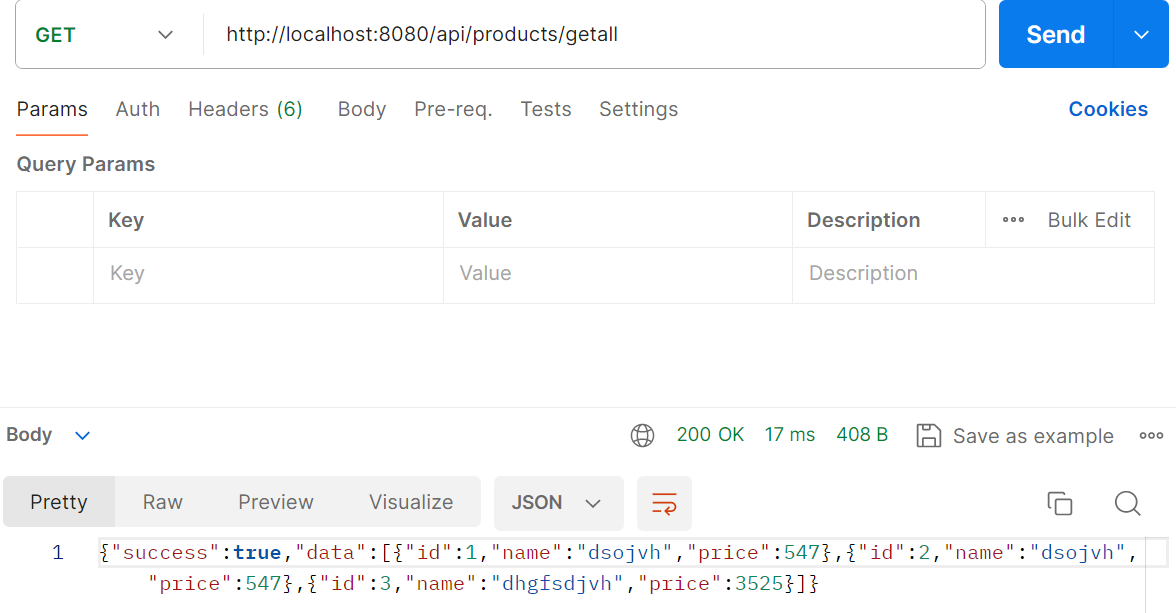
const {add,getProduct}=require("./service/productService");

app.get("/api/products/getall",getProduct)

Dont forget to change to post or else error (cannot get url)

Dont forget / in “/api… or else error in postman cannot get /api

As we give it in url



Dont froget to change method to get

**getAllById**

**const getById=(req,res)=>{**

**const str=`select \* from product where id=${req.params.id}`;**

**connection.query(str,(err,data,field)=>{**

**if(err){**

**console.log(err.message);**

**res.status(400).json({success:false});**

**}**

**res.status(200).json({success:true,data});**

**});**

**};**

**Index.js**

**const {add,getProduct,getById}=require("./service/productService");**

**app.get("/api/products/getById/:id",getById);**

select all columns from the "product" table where the "id" matches the value provided in the request parameters (req.params.id).

**Update**

**const update=(req,res)=>{**

**const{id,name,price}=req.body;**

**console.log(req.body);**

**const str= `update product set name="${name}",price=${price} where id=${id}`;**

**//no req.paramsid here as we take from user**

**connection.query(str,(err,data,field)=>{**

**if(err){**

**console.log(err.message);**

**res.status(400).json({success:false});**

**}**

**res.status(200).json({success:true,message:"updated"});**

**});**

**};**

**module.exports={**

**add,**

**getProduct,**

**getById,**

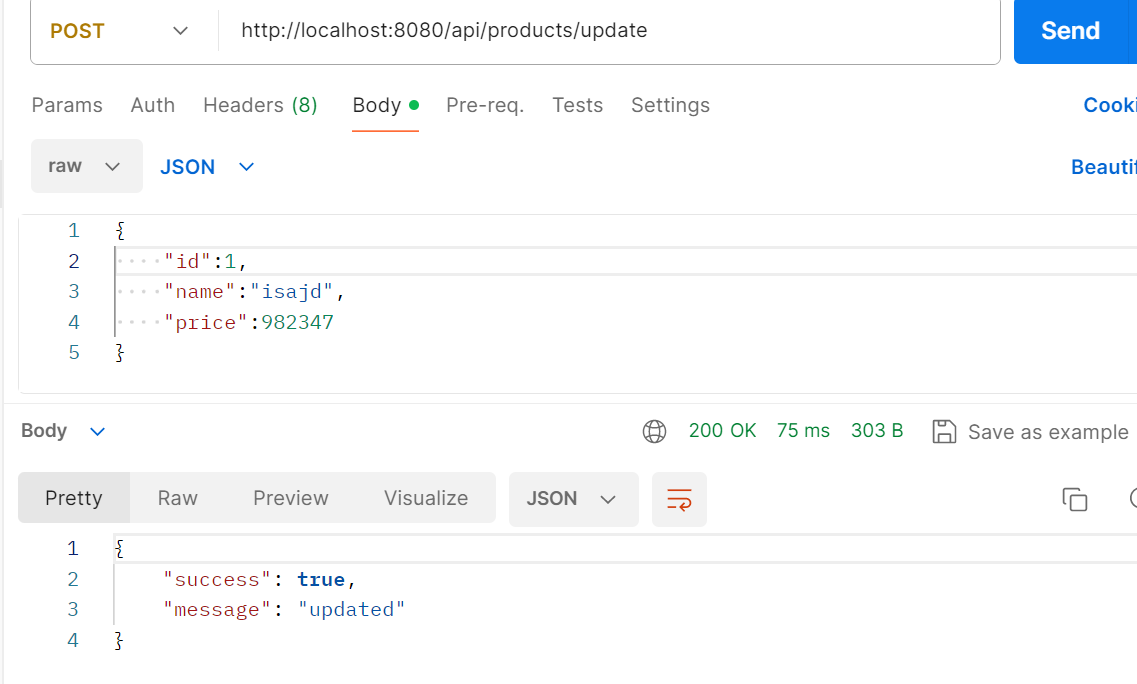
**update**

**};**

**Index.js**

const {add,getProduct,getById,update}=require("./service/productService");

app.post("/api/products/update",update);



**deleteById**

const deleteById=(req,res)=>{

const str=`delete from product where id=${req.params.id}`;

connection.query(str,(err,data,field)=>{

if(err){

console.log(err.message);

res.status(400).json({success:false});

}

res.status(200).json({success:true,message:"deleted"});

});

};

module.exports={

add,

getProduct,

getById,

update,

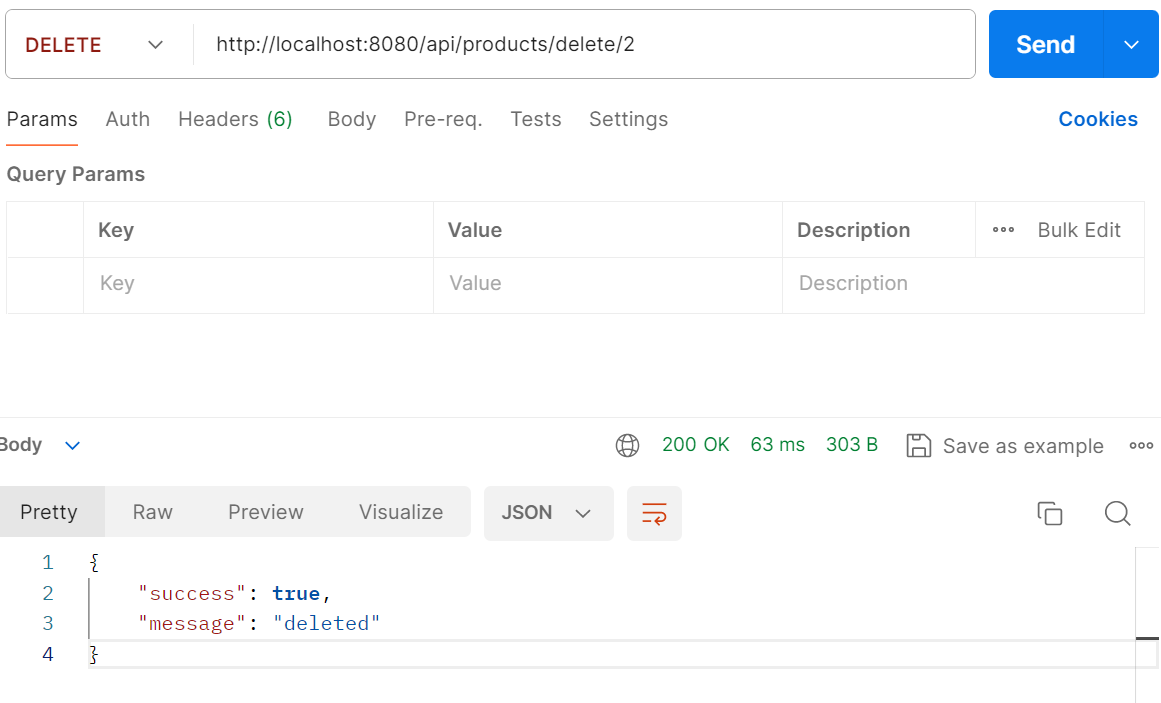
deleteById

};

Index.js

const {add,getProduct,getById,update,deleteById}=require("./service/productService");

app.delete("/api/products/delete/:id",deleteById);



In index.js add new method

Give uri

Open postman

Select method

Type uri and id

Update method wite update query

In index.js update product

App.post give uri

Open postman

Put copy name

Mysql create db and trable

Db connect

Nodemon

Service folder

Add

Index.js call

Postman

**React**

In terminal cd ..

Npx create-react-app client

Cd client/

Npm i react-router-dom

Npm i axios

npm start

Delete

Clear app.css

In app.js import browserRouter

In function app return browserRouter routes

import { BrowserRouter, Route, Routes } from "react-router-dom";

export default function App() {

return (

<BrowserRouter>

<Header />

<Routes>

<Route path={"/"} element={<Home />} />

<Route path={"/create-product"} element={<CreateProduct />} />

<Route path={"/update-product"} element={<UpdateProduct />} />

</Routes>

</BrowserRouter>

);

import { BrowserRouter, Route, Routes } from "react-router-dom";

import Home from "./pages/Home";

import CreateProduct from "./pages/CreateProduct";

import EditProduct from "./pages/EditProduct";

function App() {

return (

<BrowserRouter>

<Routes>

<Route path="/" element={<Home />} />

<Route path="/create-product" element={<CreateProduct />} />

<Route path="/edit-product/:id" element={<EditProduct />} />

</Routes>

</BrowserRouter>

);

}

export default App;

In pages folder

In home.jsx

Add links

In pages->createProduct

In home.jsx

useEffect

In terminal

Npm i axios

In useEffect fetchdata res console log

Const[roducts ,setProducts]

In terminal

Cd api

Nodemon

Add table list then button

Add co nst handleedit,handledelelte

Postman