**PROJECT NAME:**

API for ShoppingCart

**INTRODUCTION:**

REST API for ShoppingCart to add ,delete and update products from the database (MONGODB).

**REQUIREMENTS:**

* + - Visual studio code
    - Postman
    - MongoDB
    - Node js

**INSTALLATION:**

* Install visual studio code from this link -><https://code.visualstudio.com/download>.
* Install postman from this link -><https://www.postman.com/downloads>
* Install mongoDB from this link -><https://www.mongodb.com/try/download/community>
* Install node js from this link -><https://nodejs.org/en/download>

**CONFIGURATION:**

**NPM:**

* After installing node js , initiate npm by ***npminit*** in VS code terminal.

**Visual studio code:**

* After installing the VS code, install required packages.
* For our project the required packages are,
  + **Express** ( *npmi express* to install the package).
  + **Mongoose** ( *npmi mongoos*e to install the package).

**MongoDB:**

* After installing mongoDB , Create a database using cmd or mongoDB compass.
* Create a database by giving the required informations (in mongoDB compass).

**Program:**

Create required schema:

* customerSchema

const mongoose = require('mongoose');

const Schema = mongoose.Schema;

const LineSchema = new Schema({

       price:{

            type:String,

            required:true

        },

        quantity:{

            type:String,

            required:true

        },

        tax\_lines:{

            type:Array,

            price:{

            type:String,

            required:true

        },

        rate:{

            type:String,

            required:true

        },

        title:{

            type:String,

            required:true

        }

    }

});

const customerSchema = new Schema({

     created\_at:{

        type:Date,

        default:Date.now()

    },

    customer\_id:{

        type:String,

        required:true

    },

    Line\_items:{

        type:[LineSchema],

        default:[]

    }

    });

module.exports = Customer = mongoose.model('customers',customerSchema);

**Creating Routers**

* **Add new customer**

const express = require('express');

const { findByIdAndDelete } = require('../model/customer');

const router = express.Router();

const Customer = require('../model/customer');

// // Add new customers

router.post('/', async (req, res) => {

    const customer = new Customer(req.body);

    try {

        await customer.save();

        res.status(201).send(customer);

    } catch (error) {

        res.status(500).send();

    }

});

* **Add new product**

router.post('/:cid', async (req, res) => {

    try {

        const customer = await Customer.findOne({customer\_id:req.params.cid});

        if (!customer) {

            return res.status(404).send({ error: 'customer not found' });

        }

           const newProduct = req.body;

           console.log(newProduct);

           console.log(typeof customer.Line\_items);

           customer.Line\_items.push(newProduct)

           customer.save();

        res.status(200).send(customer);

    } catch (error) {

        res.status(500).send({ error: 'Internal server error' });

    }

});

* **Get OrderdDetails using OrderId**

router.get('/:id', async (req, res) => {

    try {

        const order = await Customer.findById(req.params.id);

        if (!order) {

            return res.status(404).send({ error: 'order not found' });

        }

        res.send(order);

    } catch (error) {

        res.status(500).send({ error: 'Internal server error' });

       }

});

* **Get details using CustomerId**

router.get('/product/:cid', async (req, res) => {

    try {

        const customer = await Customer.findOne({customer\_id:req.params.cid});

        if (!customer) {

            return res.status(404).send({ error: 'customer not found' });

        }

         res.send(customer.Line\_items);

    } catch (error) {

        res.status(500).send({ error: 'Internal server error' });

       }

});

* **Get total number of products and total price by customerId**

router.get('/products/:cid', async (req, res) => {

    try {

        const customer = await Customer.findOne({customer\_id:req.params.cid});

        if (!customer) {

            return res.status(404).send({ error: 'customer not found' });

        }

         const products=customer.Line\_items.length;

         const price = customer.Line\_items;

         var y = 0;

         price.forEach((e)=>{

             y += parseFloat(e.price);

            });

        console.log(y);

        const total = [{"total products":products,"totalprice":y}]

        total.push();

        // console.log(total);

        res.send(total);

    } catch (error) {

        res.status(500).send({ error: 'Internal server error' });

       }

});

* **Delete a product in Line\_items**

router.delete('/:id/delete/:pid', async (req, res) => {

    const orderId = req.params.id;

    const product\_id = req.params.pid;

    try {

        const customer = await Customer.findById(orderId);

        if (!customer) {

            return res.status(404).send({ error: 'customer not found' });

        }

        customer.Line\_items.map((e)=>{

             if(e.\_id == product\_id)

              return e.remove()

            });

         customer.save()

        res.send(customer);

        } catch (error) {

        res.status(500).send({ error: 'Internal server error' });

    }

});