ASSIGNMENT 5 NAME :S NAVEENA NITHYASHRI

CSE-2YEAR

## DEVELOP STOCK MANAGEMENT APPLICATION USING MERN STACK.

### SOURCE CODE:

### BACKEND:

const express = require("express");

const mongoose = require("mongoose");

const cors = require("cors");

const bodyParser = require("body-parser");

const Product = require("./models/Product");

const app = express();

app.use(cors());

app.use(bodyParser.json());

// MongoDB Connection

mongoose.connect("mongodb://localhost:27017/stockDB", {

useNewUrlParser: true,

useUnifiedTopology: true,

})

.then(() => console.log("MongoDB connected"))

.catch((err) => console.log(err));

// Get all products

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

try {

const products = await Product.find();

res.json(products);

} catch (err) {

res.status(400).send(err);

}

});

// Add a new product

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

const newProduct = new Product(req.body);

try {

await newProduct.save();

res.status(201).send("Product added successfully!");

} catch (err) {

res.status(400).send(err);

}

});

// Update a product

app.put("/api/products/:id", async (req, res) => {

try {

const updatedProduct = await Product.findByIdAndUpdate(req.params.id, req.body, { new: true });

res.json(updatedProduct);

} catch (err) {

res.status(400).send(err);

}

});

// Delete a product

app.delete("/api/products/:id", async (req, res) => {

try {

await Product.findByIdAndDelete(req.params.id);

res.send("Product deleted successfully");

} catch (err) {

res.status(400).send(err);

}

});

// Start the server

app.listen(5000, () => {

console.log("Server is running on http://localhost:5000");

});

**//models/Product.js**

//This file defines the MongoDB schema for a product.

//javascript

const mongoose = require("mongoose");

const productSchema = new mongoose.Schema({

name: {

type: String,

required: true,

},

quantity: {

type: Number,

required: true,

},

price: {

type: Number,

required: true,

},

});

.exports = mongoose.model("Product", productSchema);

### FRONTEND

import React, { useState, useEffect } from "react";

import axios from "axios";

function App() {

const [products, setProducts] = useState([]);

const [newProduct, setNewProduct] = useState({ name: "", quantity: 0, price: 0 });

const [updatedProduct, setUpdatedProduct] = useState({ name: "", quantity: 0, price: 0 });

useEffect(() => {

axios.get("http://localhost:5000/api/products")

.then(response => setProducts(response.data))

.catch(error => console.error("Error fetching products", error));

}, [products]);

const handleAddProduct = async () => {

await axios.post("http://localhost:5000/api/products", newProduct);

setNewProduct({ name: "", quantity: 0, price: 0 });

};

const handleDeleteProduct = async (id) => {

await axios.delete(`http://localhost:5000/api/products/${id}`);

};

const handleUpdateProduct = async (id) => {

await axios.put(`http://localhost:5000/api/products/${id}`, updatedProduct);

setUpdatedProduct({ name: "", quantity: 0, price: 0 });

};

return (

<div className="App">

<h1>Stock Management System</h1>

<h2>Add Product</h2>

<input

type="text"

placeholder="Product Name"

value={newProduct.name}

onChange={(e) => setNewProduct({ ...newProduct, name: e.target.value })}

/>

<input

type="number"

placeholder="Quantity"

value={newProduct.quantity}

onChange={(e) => setNewProduct({ ...newProduct, quantity: e.target.value })}

/>

<input

type="number"

placeholder="Price"

value={newProduct.price}

onChange={(e) => setNewProduct({ ...newProduct, price: e.target.value })}

/>

<button onClick={handleAddProduct}>Add Product</button>

<h2>Product List</h2>

<ul>

{products.map((product) => (

<li key={product.\_id}>

{product.name} - Quantity: {product.quantity} - Price: ${product.price}

<button onClick={() => handleDeleteProduct(product.\_id)}>Delete</button>

<button onClick={() => setUpdatedProduct({ name: product.name, quantity: product.quantity, price: product.price })}>Update</button>

</li>

))}

</ul

{updatedProduct.name && (

<div>

<h3>Update Product</h3>

<input

type="text"

value={updatedProduct.name}

onChange={(e) => setUpdatedProduct({ ...updatedProduct, name: e.target.value })}

/>

<input

type="number"

value={updatedProduct.quantity}

onChange={(e) => setUpdatedProduct({ ...updatedProduct, quantity: e.target.value })}

/>

<input

type="number"

value={updatedProduct.price}

onChange={(e) => setUpdatedProduct({ ...updatedProduct, price: e.target.value })}

/>

<button onClick={() => handleUpdateProduct(updatedProduct.\_id)}>Update Product</button>

</div>

)}

</div>

);

}

export default App;

### OUTPUT:

UPDATING A PRODUCT:

---------------------------------------------

Stock Management System

---------------------------------------------

Add Product:

---------------------------------------------

Product Name: [\_\_\_\_\_\_\_\_\_\_\_]

Quantity: [\_\_\_\_\_\_\_\_\_\_\_]

Price: [\_\_\_\_\_\_\_\_\_\_\_]

[ Add Product ]

---------------------------------------------

---------------------------------------------

Product List:

---------------------------------------------

| Name | Quantity | Price | Action |

---------------------------------------------

| Banana | 50 | $0.3 | [Delete] [Update] |

| Orange | 250 | $0.65 | [Delete] [Update] |

---------------------------------------------

Product updated successfully.

---------------------------------------------