Vite.config.js

```
import { defineConfig } from 'vite'
import react from '@vitejs/plugin-react'


// https://vite.dev/config/
export default defineConfig({
  server:{
    proxy:{
      '/api' : 'http://localhost:8000/'
    }
  },
  plugins: [react()],
})
```

App.jsx

```
import axios from "axios";
import { useEffect, useState } from "react";


const App = () => {
  const [product, setProduct] = useState([]);
  const [name, setName] = useState();
  const [price, setPrice] = useState();
  const [search, setSearch] = useState('');
  // Data fetch (GET) request
  useEffect(() => {
    const ani = async () => {
      const response = await axios.get('/api/user');
      setProduct(response.data);
    };
    ani();
  }, []);


  // Add product
  const aniket = async (e) => {
    e.preventDefault();
```

```
    const pro = {
      name: name,
      price: price,
    };
    const response = await axios.post('/api/user', pro);
    console.log('Response:', response.data);
    setName('');
    setPrice('');
    setProduct([...product, response.data]);
  };

  // Delete product
  const deleteProduct = async (id) => {
    const response = await axios.delete(`/api/user/${id}`);
    console.log('Deleted:', response.data);

    setProduct(product.filter((item) => item._id !== id));
  };

  // Update product
```

```
  const updateProduct = async (id, updatedData) => {
    const response = await axios.put(`/api/user/${id}`,
updatedData);
    console.log('Updated:', response.data);

    // Update the state with the modified product
    setProduct(
      product.map((item) =>
        item._id === id ? { ...item, ...updatedData } : item
      )
    );
  };

  // Filter products based on search query
  const filteredProducts = product.filter((item) =>
    item.name.toLowerCase().includes(search.toLowerCase()) //
Case-insensitive search
  );

  return (
```

```
    <>
      <h1>Jay Mata di {filteredProducts.length}</h1>

      {/* Search Box */}
      <input
        type="text"
        placeholder="Search Products..."
        value={search}
        onChange={(e) => setSearch(e.target.value)} // Update
the search state
      />

      {/* Add Product Form */}
      <form onSubmit={aniket}>
        <input
          type="text"
          placeholder="Enter Your Name"
          value={name}
          onChange={(e) => setName(e.currentTarget.value)}
        />
```

```
        <input
          type="number"
          placeholder="Enter Your Price "
          value={price}
          onChange={(e) => setPrice(e.currentTarget.value)}
        />
        <button type="submit">Click</button>
      </form>

      {/* Display filtered products */}
      {filteredProducts.map((item) => (
        <div key={item._id}>
          <span>
            <strong>Id</strong>: {item._id}
<strong>Name</strong>: {item.name}{" "}
            <strong>Price</strong>: {item.price}
          </span>

          {/* Edit and Delete buttons */}
```

```
          <button onClick={() =>
deleteProduct(item._id)}>Del</button>


          {/* Update Product */}
          <button
            onClick={() => {
              // For demonstration, we'll update the name and
price
              const updatedData = { name: "Updated Name",
price: 100 };
              updateProduct(item._id, updatedData);
            }}
          >
            Update
          </button>
        </div>
      ))}
    </>
  );
};
```

```
export default App;
```

Server.js

```
import express from 'express';
import mongoose from 'mongoose';
import cors from 'cors';

const app = express();

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

mongoose.connect('mongodb://localhost:27017/e-comm', {
  useNewUrlParser: true,
  useUnifiedTopology: true
});

const productSchema = new mongoose.Schema({
```

```
  name: String,
  price: Number,
  brand: String,
  category: String
});

const Product = mongoose.model('Product', productSchema);

app.get("/", (req, res) => {
  res.send("Jay Mata Di");
});

app.get("/api/user", async (req, res) => {
  const data = await Product.find();
  res.json(data);
});

app.post('/api/user', async (req, res) => {
```

```
    const data = new Product(req.body);
    const result = await data.save();
    res.status(201).json(result);
});

app.delete('/api/user/:id', async (req, res) => {
    const { id } = req.params;
      const deletedProduct = await
Product.findByIdAndDelete(id);
      res.status(200).json({ message: "Product deleted
successfully" });
  });

  app.put('/api/user/:id', async (req, res) => {
    const { id } = req.params;
    const updatedProduct = await
Product.findByIdAndUpdate(id, req.body, { new: true });
```

```
    res.status(200).json(updatedProduct);
  });



app.listen(8000, () => {
  console.log("Server is running on http://localhost:8000");
});
```

Package.json

```
{
  "name": "server",
  "version": "1.0.0",
  "main": "server.js",
  "type": "module",
  "scripts": {
    "start": "nodemon server.js",
    "test": "echo \"Error: no test specified\" && exit 1"
```

```
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": "",
  "dependencies": {
    "body-parser": "^1.20.3",
    "cars": "^1.1.6",
    "cors": "^2.8.5",
    "express": "^4.21.2",
    "mongoose": "^8.8.4"
  }
}
```