

# Product CRUD API with Node.js & Mongoose

## Project Overview

This project is a simple RESTful API built with Node.js, Express, and MongoDB (Mongoose) that allows you to perform CRUD operations on a Product database.

- **Create a new product**
- **Read all products or a single product**
- **Update a product**
- **Delete a product**

## Technologies Used

- Node.js
- Express.js
- MongoDB
- Mongoose
- Postman (for API testing)

## Folder Structure

CRUD\_Operations/

├─ index.js

├─ models/

| └─ Product.js

## Initialization

1. Install dependencies:

**npm install express mongoose**

2. Make sure MongoDB is installed and running locally, or use a MongoDB Atlas URI.

3. Start the server (Server will run at: <http://localhost:3000>)

**node index.js**

# Code

## 1. Product.js

```
const mongoose = require('mongoose');

const productSchema = new mongoose.Schema({
  name: { type: String, required: true },
  price: { type: Number, required: true },
  category: { type: String, required: true }
});

module.exports = mongoose.model('Product', productSchema);
```

## 2. index.js

```
const express = require('express');
const mongoose = require('mongoose');
const app = express();
app.use(express.json());

mongoose.connect('mongodb://127.0.0.1:27017/productDB');

const Product = mongoose.model('Product', new mongoose.Schema({
  name: String,
  price: Number,
  category: String
}));

app.get('/products', async (req, res) => res.json(await Product.find()));

app.get('/products/:id', async (req, res) => {
  const p = await Product.findById(req.params.id);
  res.json(p || { message: 'Not found' });
});

app.post('/products', async (req, res) => res.status(201).json(await new
Product(req.body).save()));

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

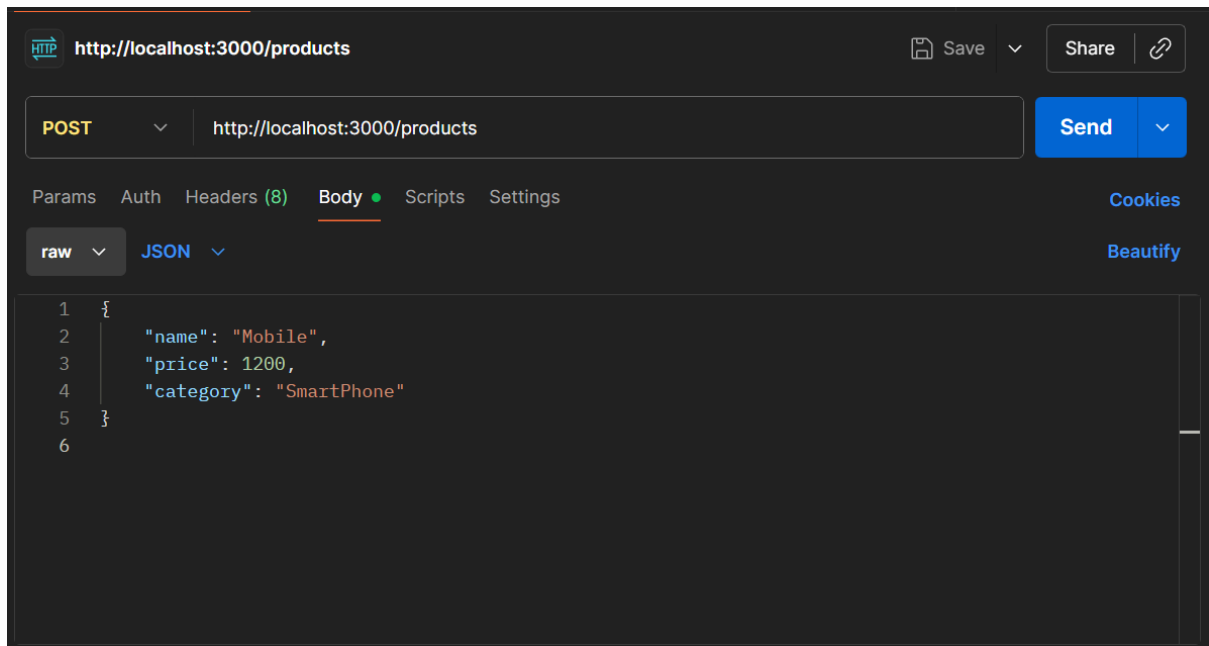
```
res.json(p || { message: 'Not found' });  
});  
app.delete('/products/:id', async (req, res) => {  
  const p = await Product.findByIdAndDelete(req.params.id);  
  res.json(p ? { message: 'Deleted', product: p } : { message: 'Not found' });  
});  
app.listen(3000, () => console.log('Server running on http://localhost:3000'));
```

## API Endpoints

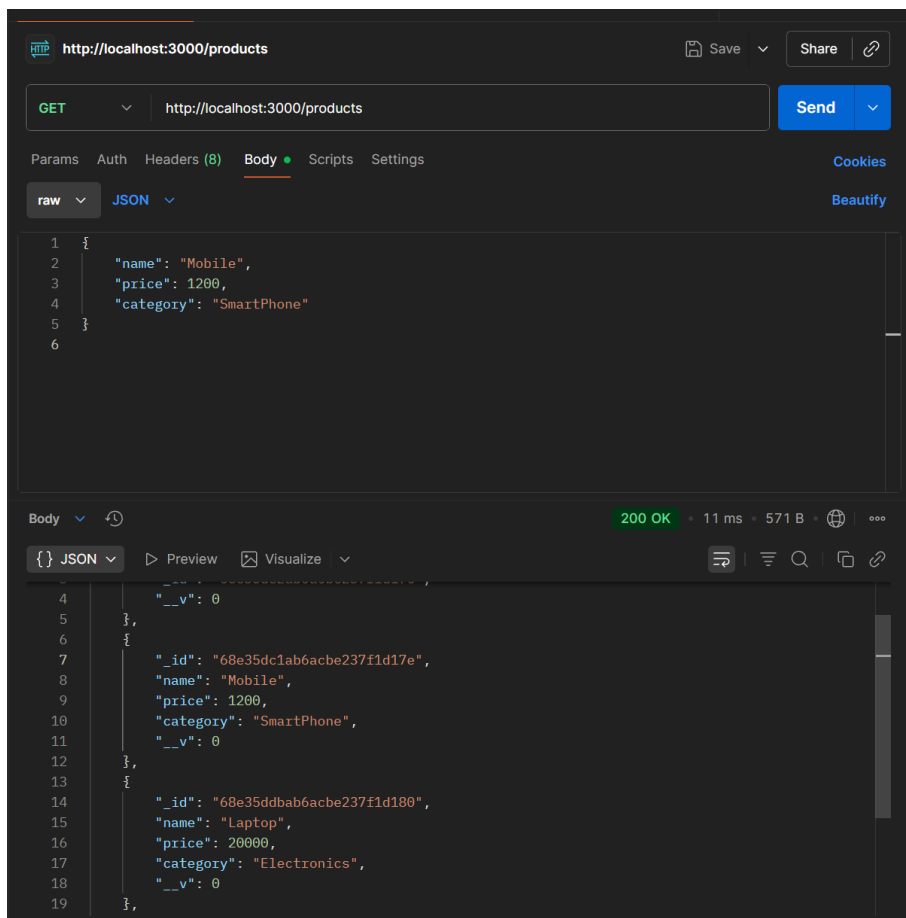
Method	Endpoint	Description
GET	/products	Get all products
GET	/products/:id	Get a single product by ID
POST	/products	Add a new product
PUT	/products/:id	Update a product by ID
DELETE	/products/:id	Delete a product by ID

# Screen Shots

## 1. Add a new product (POST)



## 2. Get all products



### 3. PUT (UPDATE ENTRIES)

The screenshot shows a REST client interface with the following details:

- URL:** `http://localhost:3000/products/68e35dc1ab6acbe237f1d17e`
- Method:** `PUT`
- Body (raw JSON):**

```
1 {  
2   "price": 5000  
3 }
```
- Response (JSON):**

```
1 {  
2   "_id": "68e35dc1ab6acbe237f1d17e",  
3   "name": "Mobile",  
4   "price": 5000,  
5   "category": "SmartPhone",  
6   "__v": 0  
7 }
```
- Status:** `200 OK`, `10 ms`, `330 B`

### 5. DELETE DATA

The screenshot shows a REST client interface with the following details:

- URL:** `http://localhost:3000/products/68e35decab6acbe237f1d182`
- Method:** `DELETE`
- Body:** Empty
- Response (JSON):**

```
1 {  
2   "message": "Deleted",  
3   "product": {  
4     "_id": "68e35decab6acbe237f1d182",  
5     "name": "Mobile",  
6     "price": 1200,  
7     "category": "SmartPhone",  
8     "__v": 0  
9   }  
10 }
```
- Status:** `200 OK`, `7 ms`, `363 B`

**NAME : PRAJJWAL KANDPAL**  
**UID : 23BIS70052**