

## Index.js

```
express = require('express')
const mongoose = require ('mongoose');
const Product = require('./models/product.model.js');
const app = express()

app.use(express.json());

//reading all products
app.get('/', function (req, res) {
    res.send("hello from the node api update");
});

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

    try {
        const products = await Product.find({});
        res.status(200).json(products);

    }catch(error){
        res.status(500).json({message: error.message});
    }

});

//read api but by only one product
app.get('/api/product/:id', async (req,res) =>{
    try{
        const {id} = req.params;
        const product = await Product.findById(id);
        res.status(200).json( product );

    } catch(error){
        res.status(500).json({message: error.message});
    }
});

//creat api
app.post('/api/products',async (req,res)=>{
    try{
        const product = await Product.create(req.body);
        res.status(200).json(product);
    }
});
```

```

    }catch (error){
      res.status(500).json({message: error.message });
    }
  });

  //update a product
  app.put('/api/product/:id', async (req,res) => {
    try {
      const{id} = req.params;

      const product = await Product.findByIdAndUpdate(id ,
req.body);

      if(!product){
        return res.status(404).json({message:"Product not found"});
      }

      const updatedProduct = await Product.findById(id);
      res.status(200).json(updatedProduct);

    }catch(error){
      res.status(500).json({message: error.message });
    }
  });

  //delete a product

  app.delete("/api/product/:id", async(req,res)=>{
    try{
      const{id}= req.params;
      const product = await Product.findByIdAndDelete(id);
      if (!product){
        return res.status(404).json({message: "Product not found"});
      }

      res.status(200).json({message:"Product deleted successfully"});

    }catch(error){
      res.status(500).json({message: error.message });
    }
  } )

```

```

    //here first i connected db and then listened to the port

    mongoose.connect("mongodb+srv://akashvaddi333:K5m18vy6fB6aU7K2@cluster0.h
p9gamr.mongodb.net/Node-API?retryWrites=true&w=majority&appName=Cluster0")
    .then(() => {console.log('Connected!');
    app.listen(3000, () =>{
        console.log('server is running on port 3000')
    });
});
});

```

## Package. Json

```

{
  "name": "aka-qpi",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "serve": "node index.js",
    "dev": "nodemon index.js"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "express": "^4.18.3",
    "mongodb": "^6.5.0",
    "mongoose": "^8.2.2"
  },
  "devDependencies": {
    "nodemon": "^3.1.0"
  }
}

```

## Product.model.js

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

const ProductSchema = mongoose.Schema(
  {
    name: {
      type:String,
      required: [true,"proto"],

    },
    quantity:{
      type:Number,
      required:true,
      default:0
    },
    image:{
      type:String,
      required: false
    },

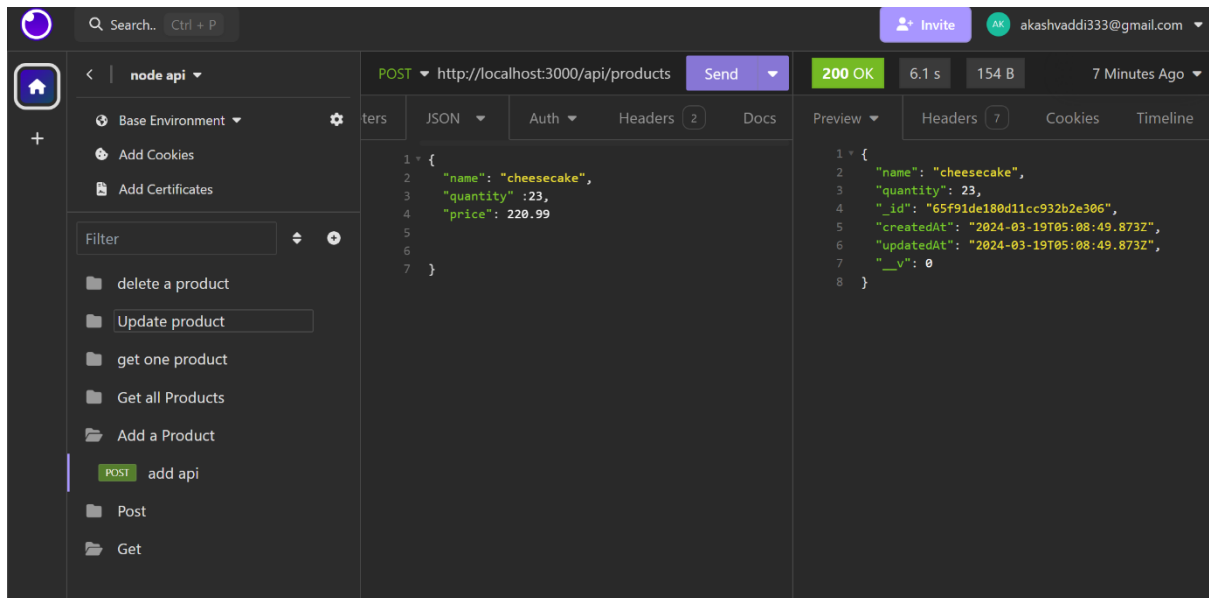
  } ,

  {
    timestamps: true,
  }
);

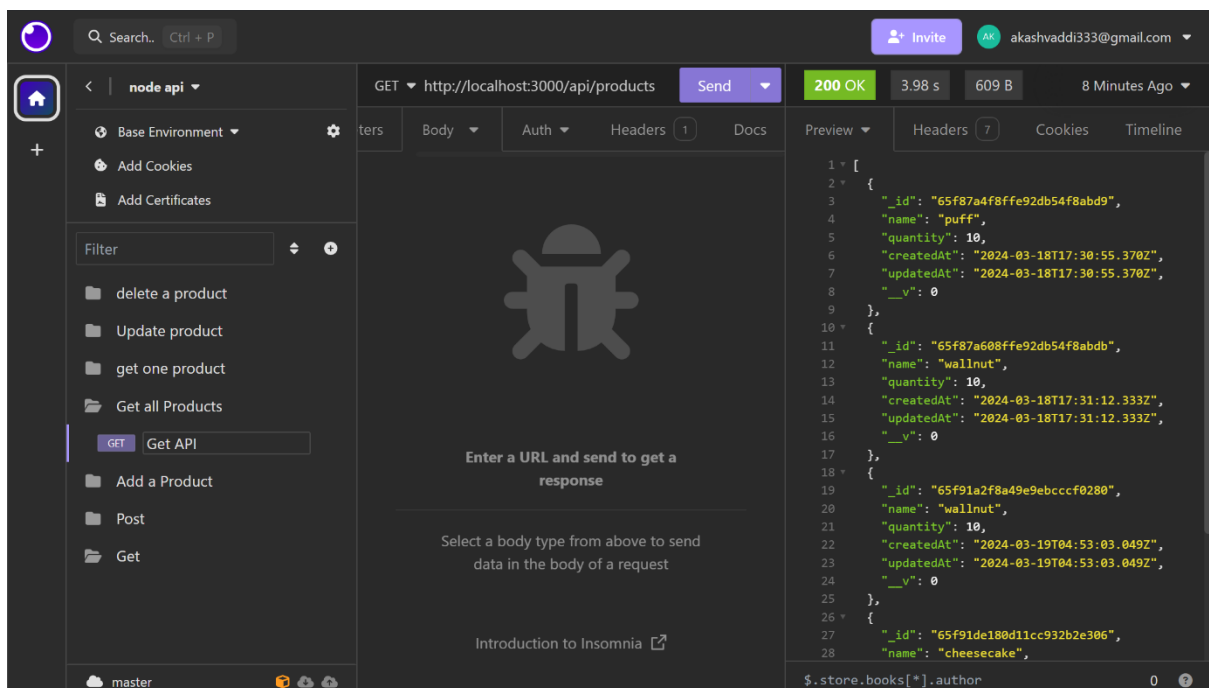
const Product = mongoose.model("Product",ProductSchema);
module.exports= Product;
```

- **CRUD operations**

Create api:



Read Api:



## Read one Api:

The screenshot shows the Node API client interface. On the left, a sidebar lists various API endpoints under the 'node api' folder. The 'GET one product' endpoint is selected. The main area displays the request details: a GET request to `http://localhost:3000/api/product/6`. The response is a 200 OK status with a response time of 162 ms and a body size of 149 B. The response body is a JSON object:

```
1 {
2   "_id": "65f879cc8ffe92db54f8abd7",
3   "name": "pizza",
4   "quantity": 10,
5   "createdAt": "2024-03-18T17:28:44.115Z",
6   "updatedAt": "2024-03-18T17:28:44.115Z",
7   "__v": 0
8 }
```

## Update Api:

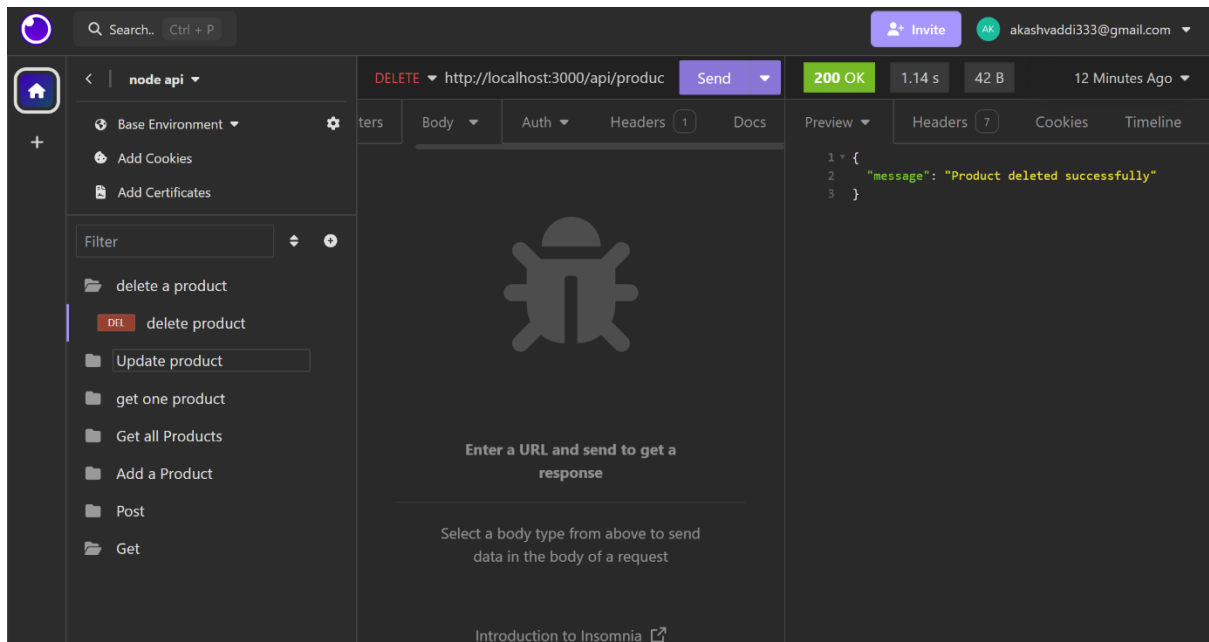
The screenshot shows the Node API client interface. On the left, the 'PUT update product' endpoint is selected. The main area displays the request details: a PUT request to `http://localhost:3000/api/product/6` with a JSON body:

```
1 {
2   "name": "biryani"
3 }
```

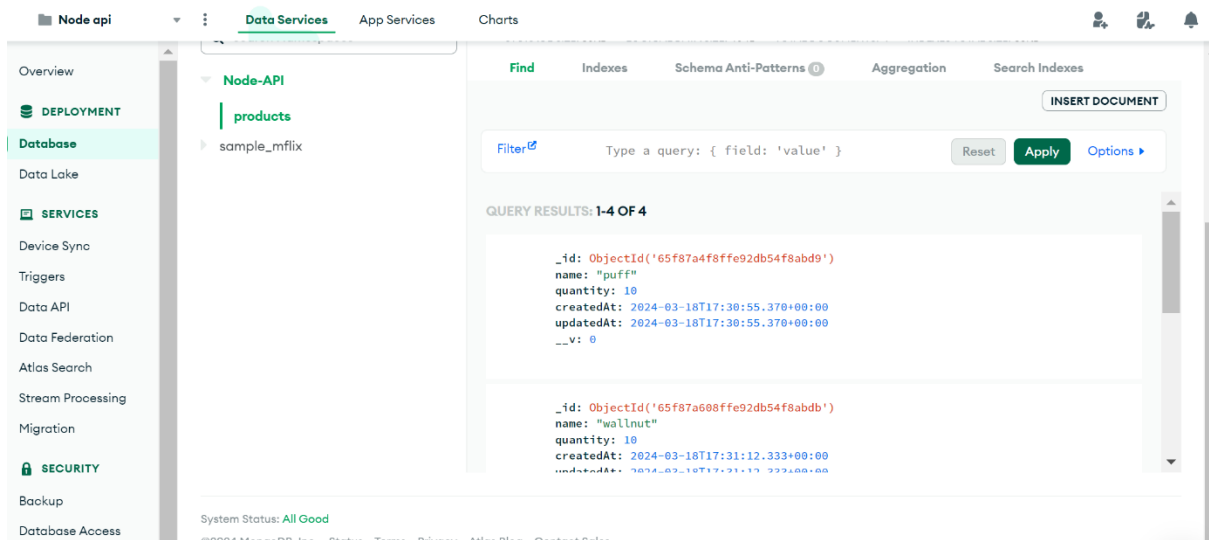
The response is a 200 OK status with a response time of 272 ms and a body size of 151 B. The response body is a JSON object:

```
1 {
2   "_id": "65f879cc8ffe92db54f8abd7",
3   "name": "biryani",
4   "quantity": 10,
5   "createdAt": "2024-03-18T17:28:44.115Z",
6   "updatedAt": "2024-03-19T04:57:08.514Z",
7   "__v": 0
8 }
```

## Delete Api:



## MongoDB : (final view)



AKA QPI:

