ASSIGNMENT-4

Name: Mummalaneni Amrutha

Roll no:20NN1A1238

E-Mail: amruthamummalaneni@gmail.com

College: Vignan’s Nirula Institute of Science and Technology for Women’s

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"
    }
```

```
}
```

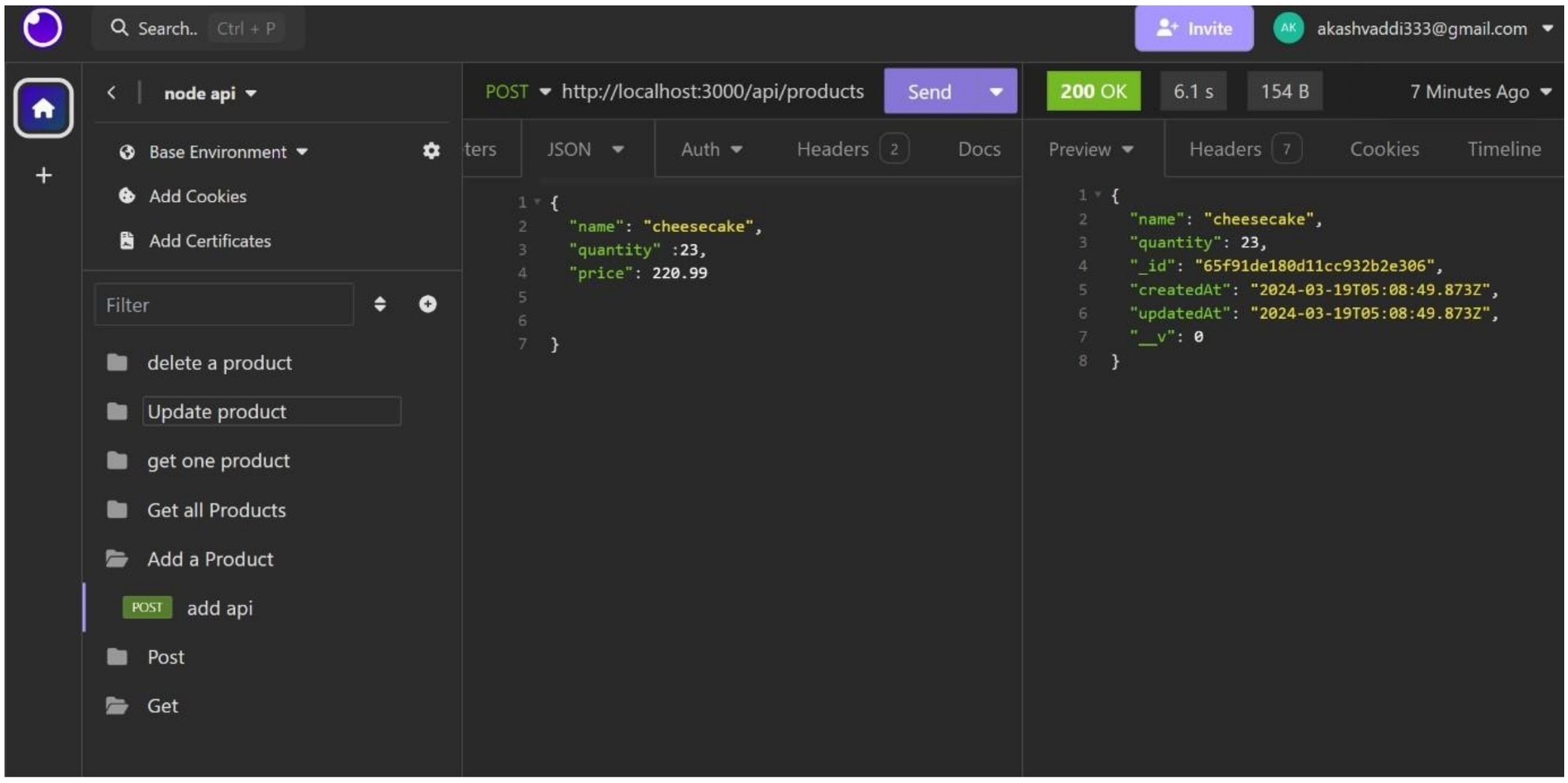
Product.model.js

```
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
                },


    },
```

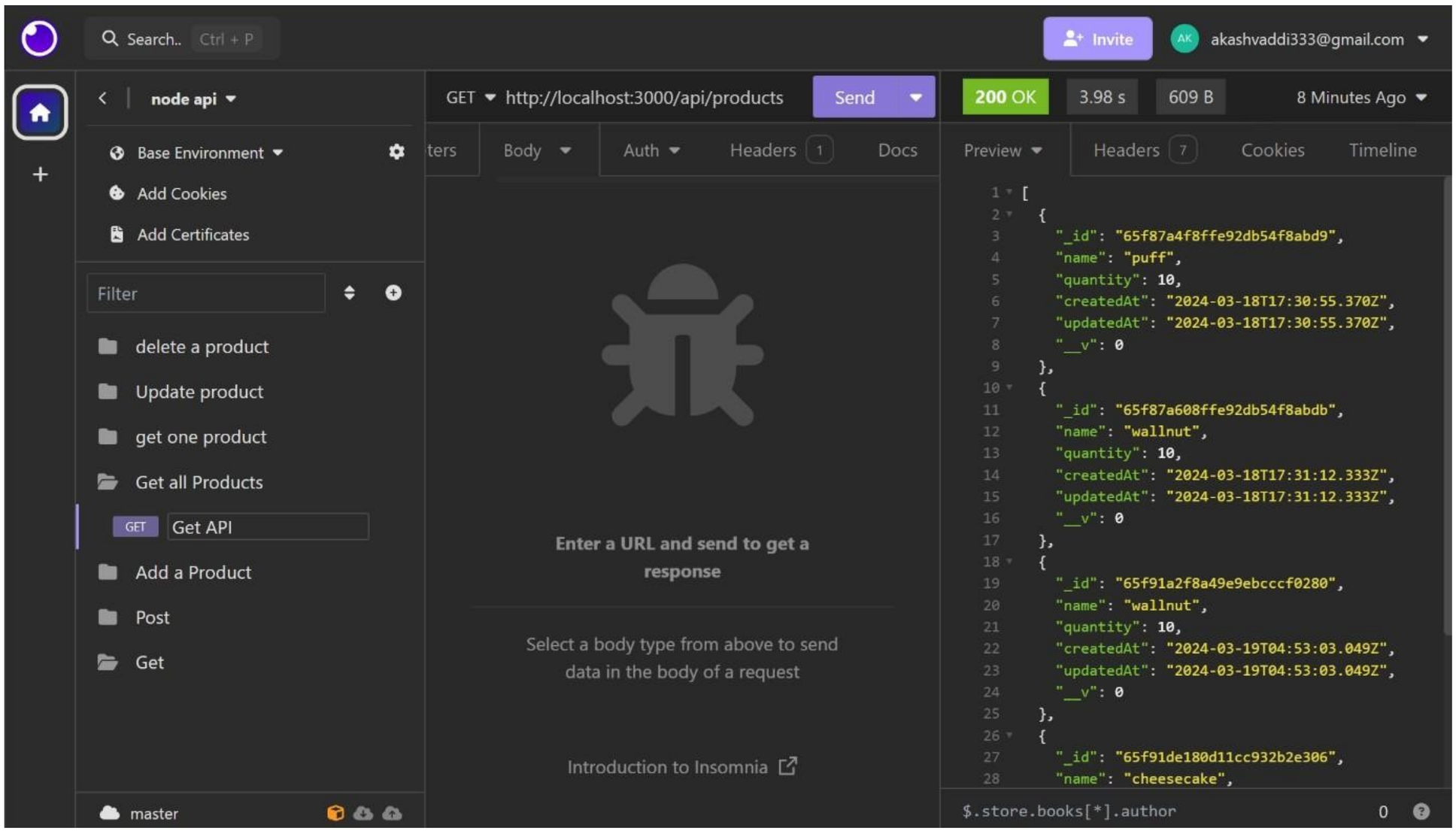
```
        {timestamps: true,
        }
);


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

- ***CRUD operations***

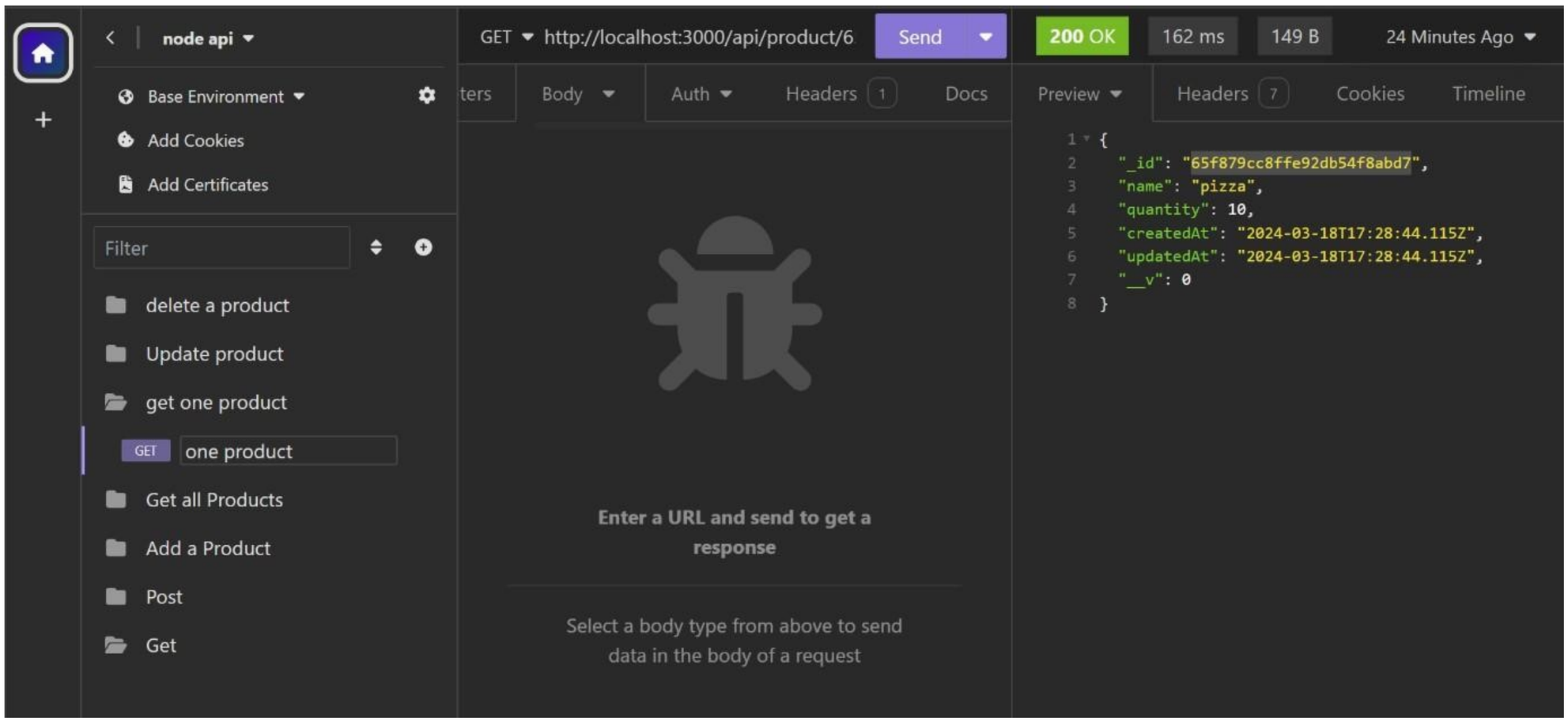
Create api:



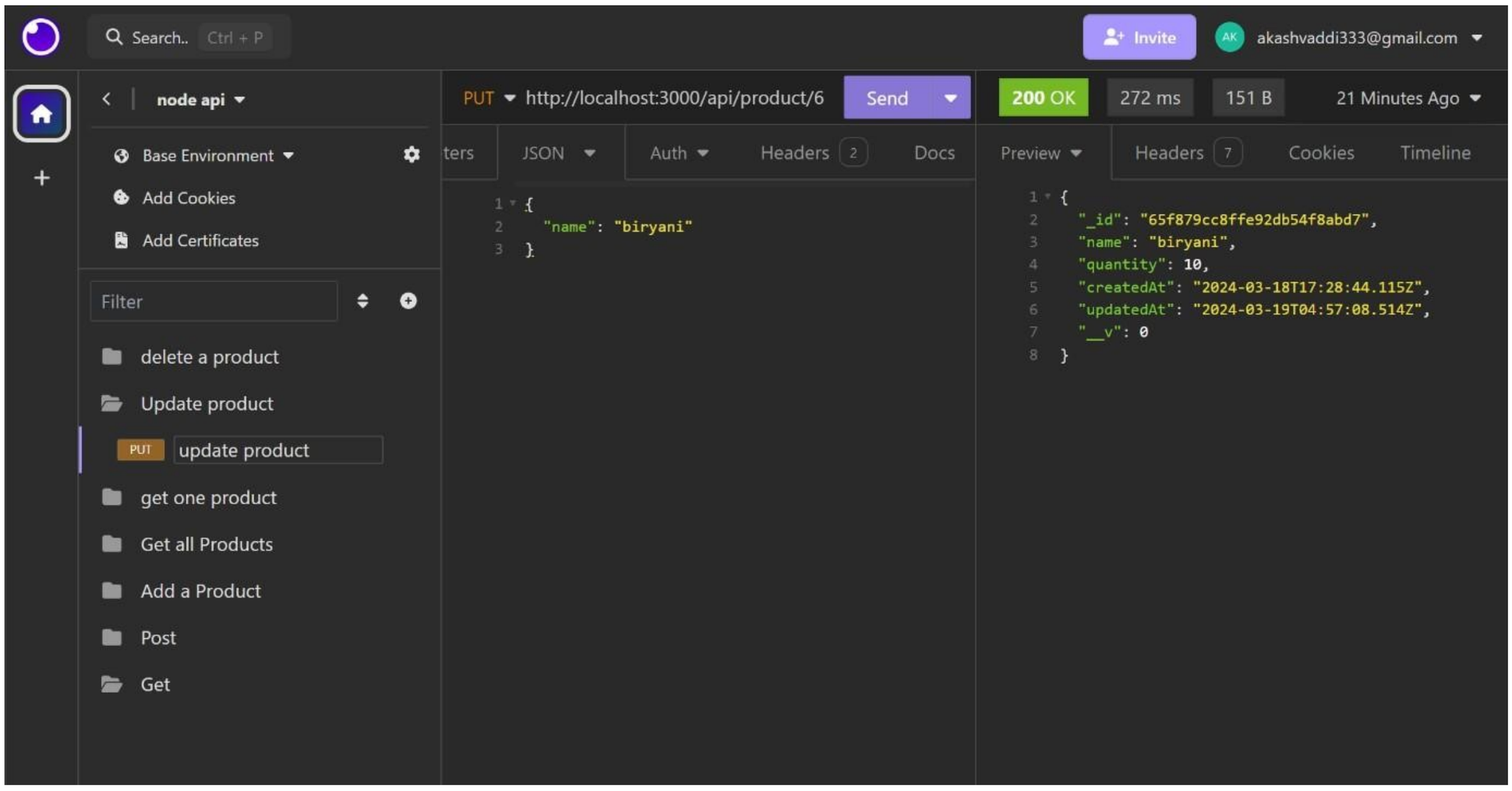
Read Api:



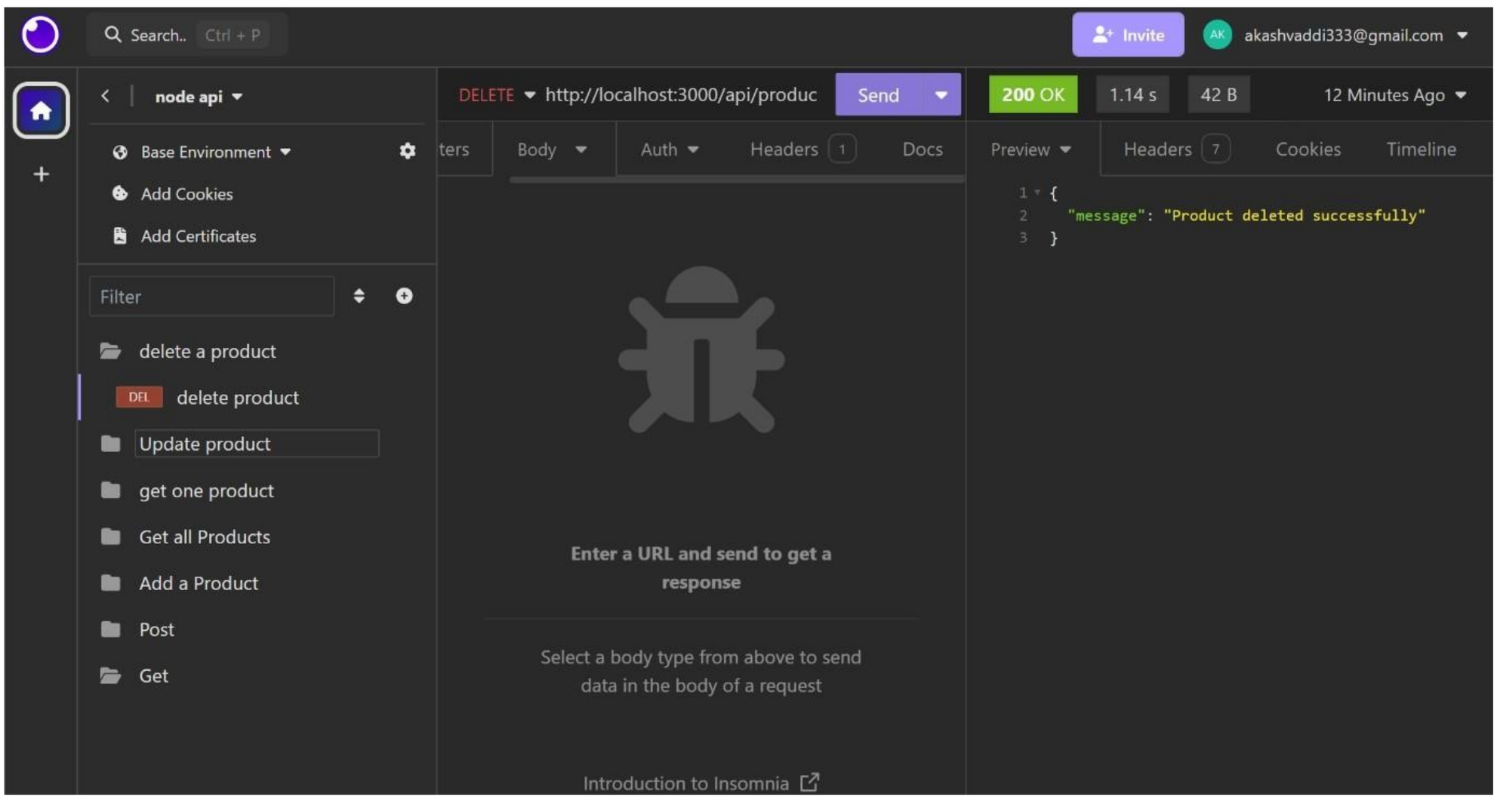
Read one Api:



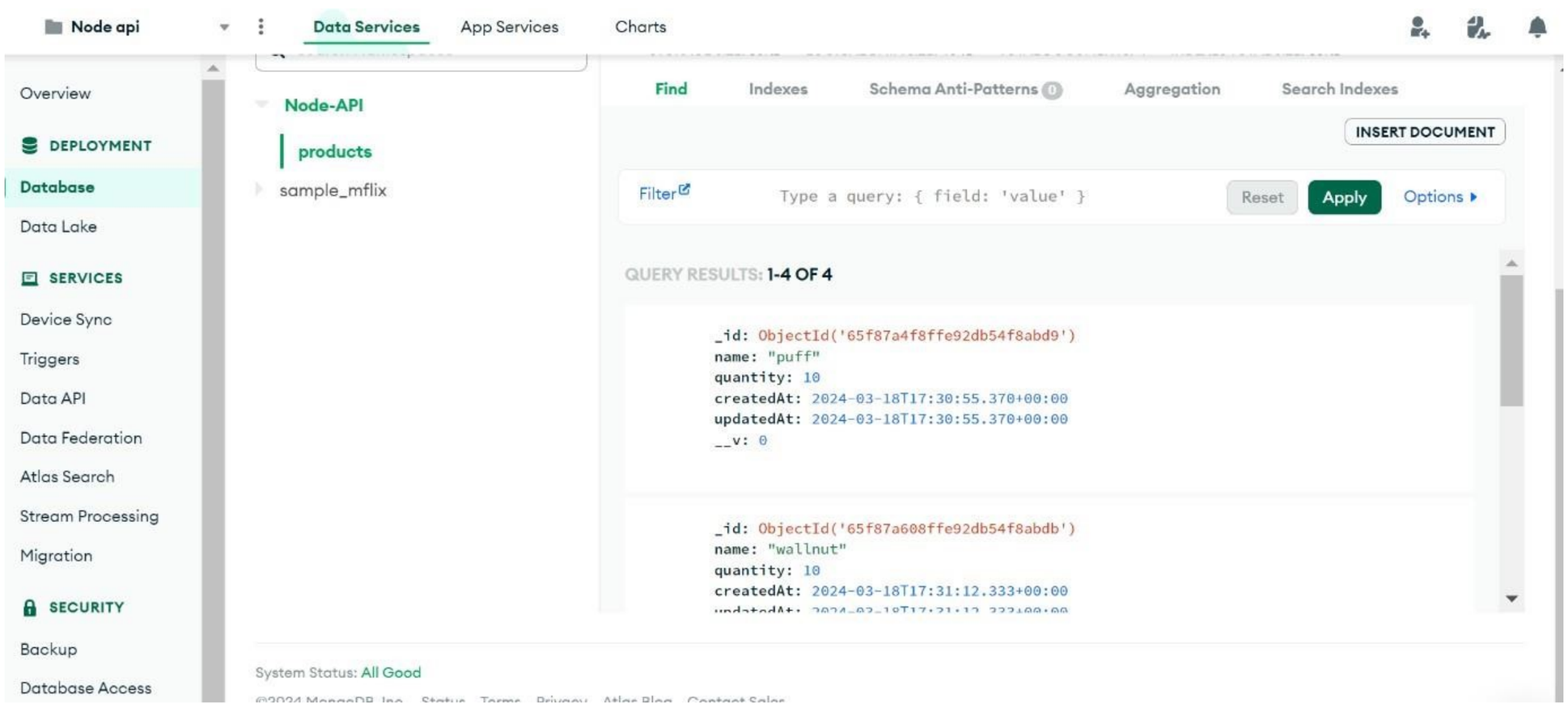
Update Api:



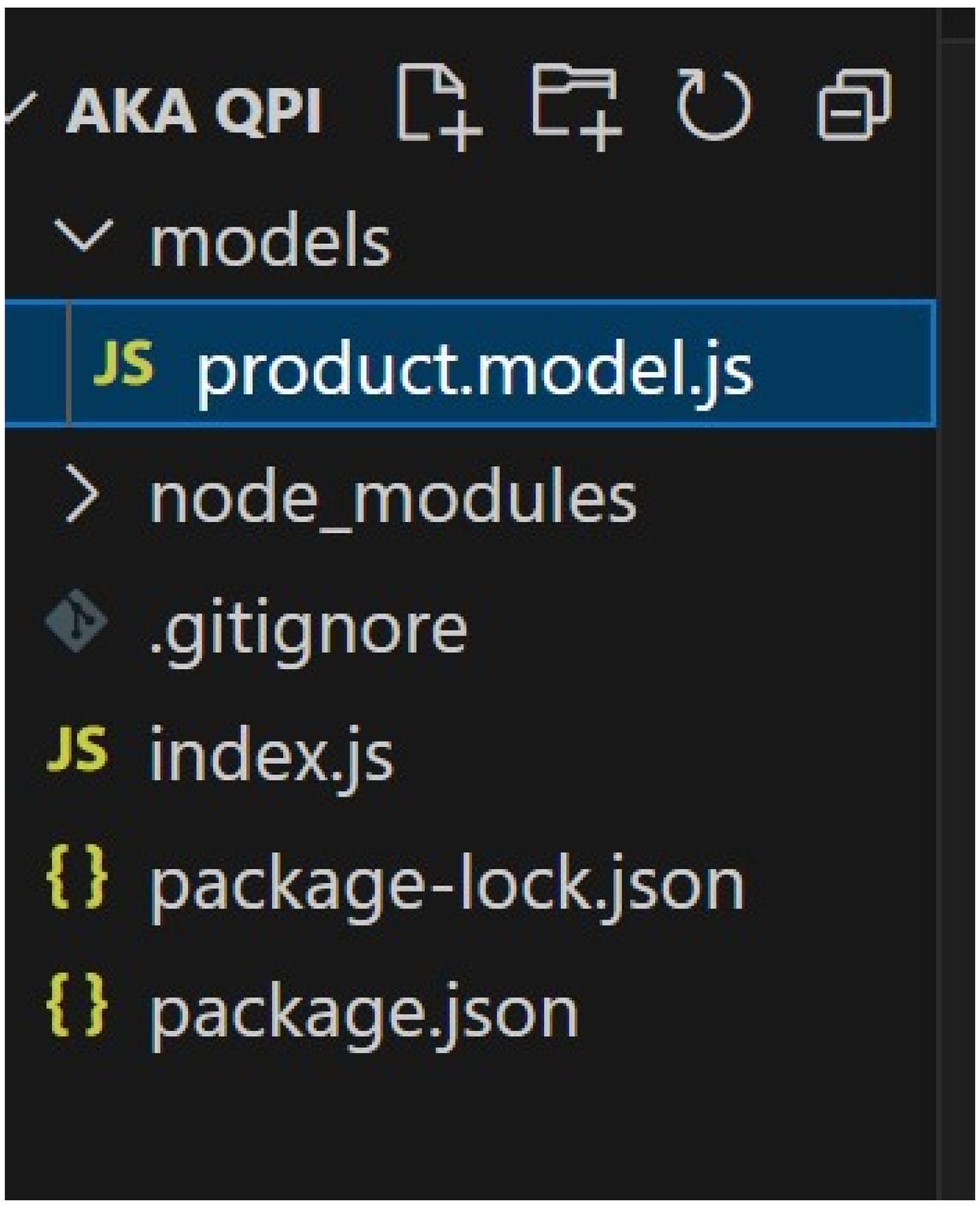
Delete Api:



MongoDB : (final view)


Node api
Data Services
App Services
Charts
Overview
DEPLOYMENT
Database
Data Lake
SERVICES
Device Sync
Triggers
Data API
Data Federation
Atlas Search
Stream Processing
Migration
SECURITY
Backup
Database Access
Node-API
products
sample_mflix
Find
Indexes
Schema Anti-Patterns 0
Aggregation
Search Indexes
INSERT DOCUMENT
Filter
Type a query: { field: 'value' }
Reset
Apply
Options
QUERY RESULTS: 1-4 OF 4
_id: ObjectId('65f87a4f8ffe92db54f8abd9')
name: "puff"
quantity: 10
createdAt: 2024-03-18T17:30:55.370+00:00
updatedAt: 2024-03-18T17:30:55.370+00:00
__v: 0
_id: ObjectId('65f87a608ffe92db54f8abdb')
name: "wallnut"
quantity: 10
createdAt: 2024-03-18T17:31:12.333+00:00
System Status: All Good

AKA QPI:


AKA QPI
models
product.model.js
node_modules
.gitignore
index.js
package-lock.json
package.json