**ASSIGNMENT-4**

**Name: Lingala Bhagya lakshmi**

**Roll no:**20NN1A1230

**E-Mail: bhagyalakshmi7207@gmail.com**

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

|  |
| --- |
| *Index.js*  express = require('express')constmongoose = require ('mongoose');constProduct = require('./models/product.model.js'); constapp = 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 {constproducts = 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;constproduct = await Product.findById(id); res.status(200).json( product );  } catch(error){res.status(500).json({message: error.message});}  });  //creatapi  app.post('/api/products',async(req,res)=>{ try{constproduct = await Product.create(req.body); res.status(200).json(product); |
| }catch(error){res.status(500).json({message: error.message});  }  });  //update a productapp.put('/api/product/:id', async (req,res) =>{ try {const{id} = req.params;  constproduct = await Product.findByIdAndUpdate(id ,  req.body);  if(!product){return res.status(404).json({message:"Product not found"});  }  constupdatedProduct= 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;  constproduct = 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

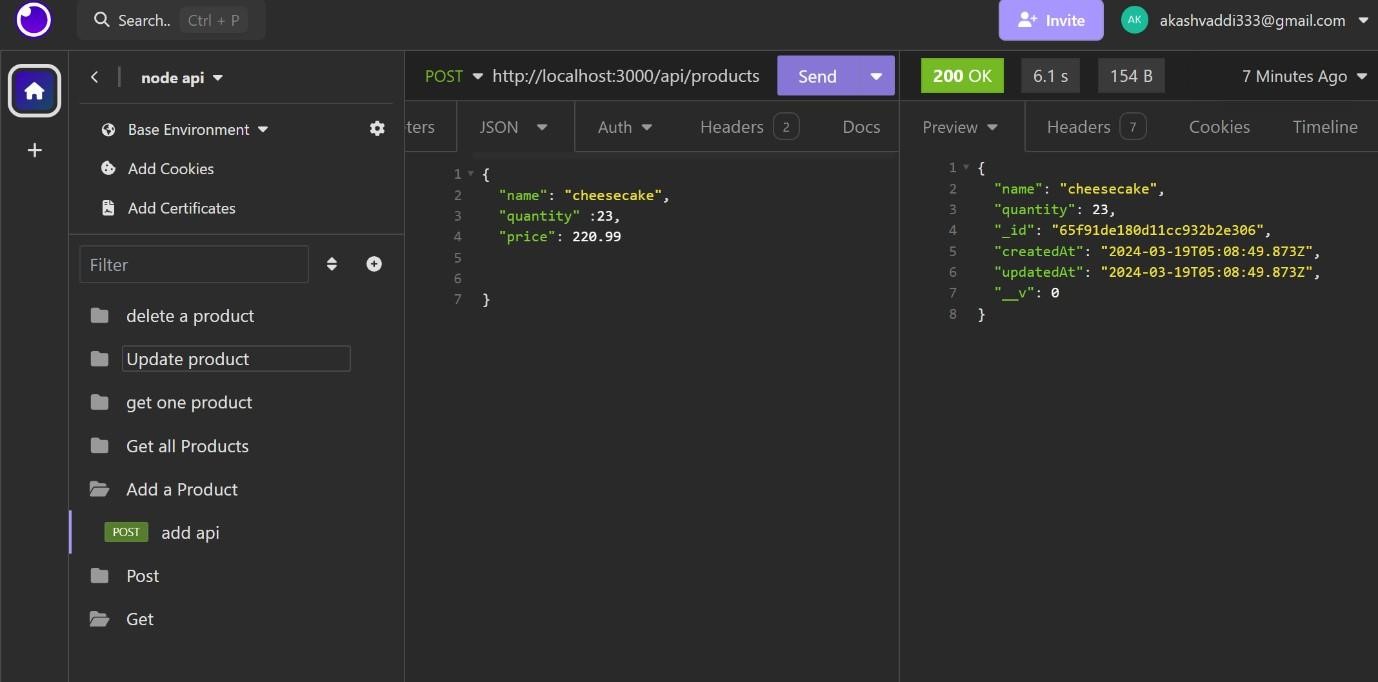
|  |
| --- |
| {  "name": "aka-qpi",  "version": "1.0.0",  "description": "",  "main": "index.js",  "scripts": {  "test": "echo \"Error: no test specified\" && exit 1",  "serve": "node index.js",  "dev": "nodemon index.js"  },  "keywords": [],  "author": "",  "license": "ISC",  "dependencies": {  "express":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

Product.model.js

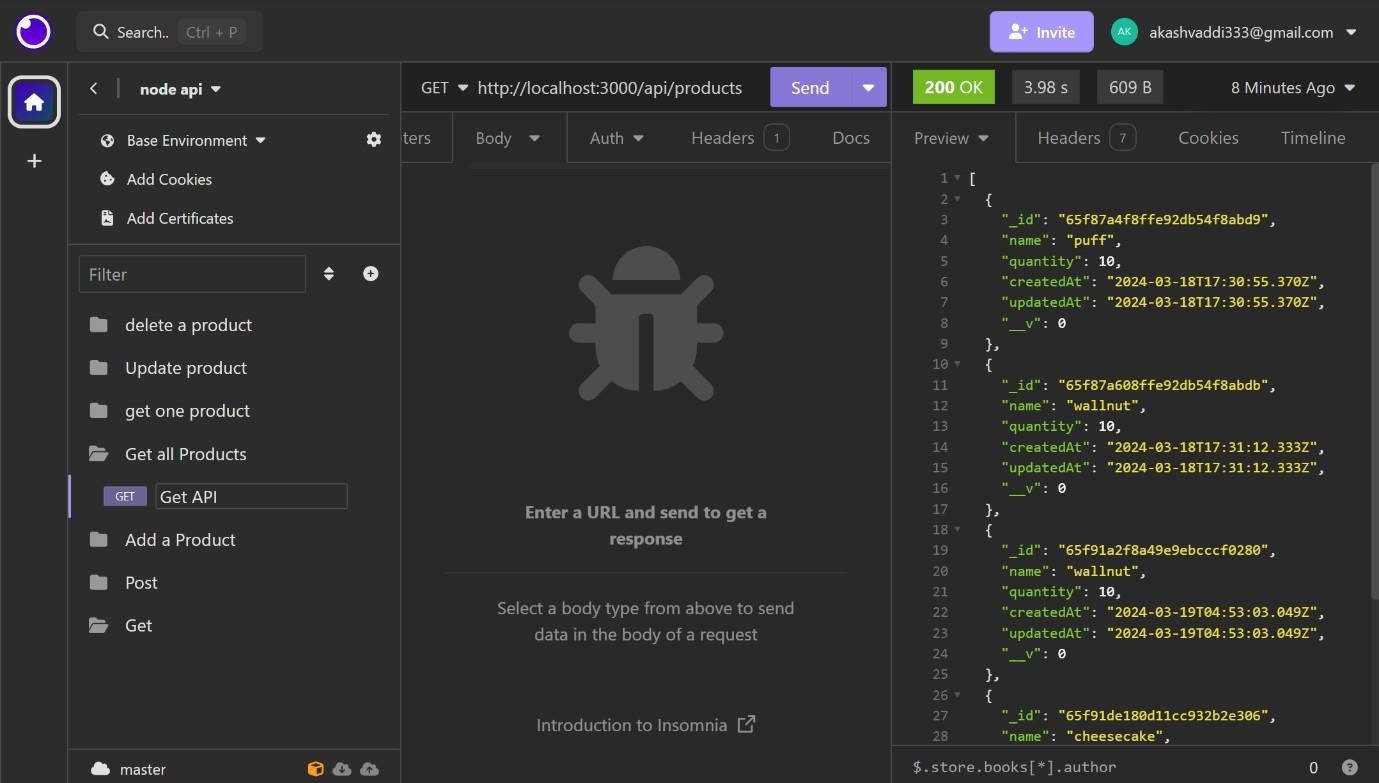
|  |
| --- |
| constmongoose = require ('mongoose');  constProductSchema= mongoose.Schema(  {name: {type:String,  required: [true,"proto"],  },quantity:{type:Number, required:true, default:0  },  image:{type:String, required: false  },  } ,  {timestamps: true,  }  );  constProduct = mongoose.model("Product",ProductSchema); module.exports= Product; |

## CRUD operations

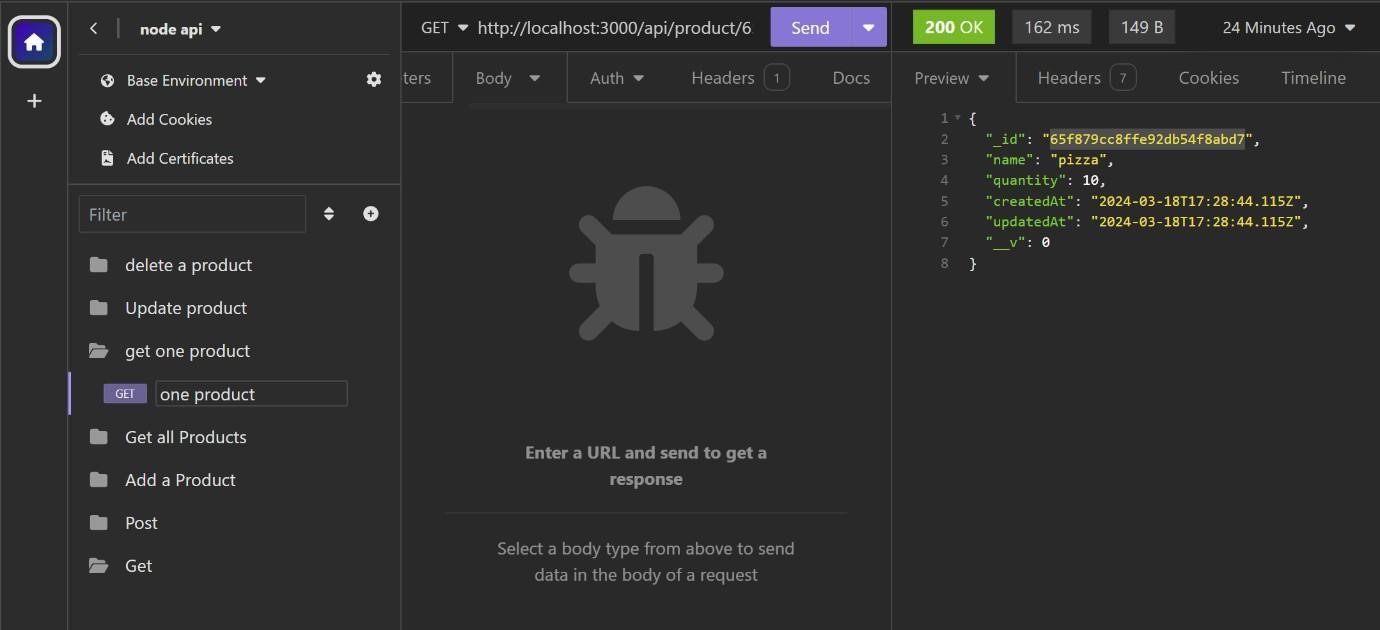
Create api:



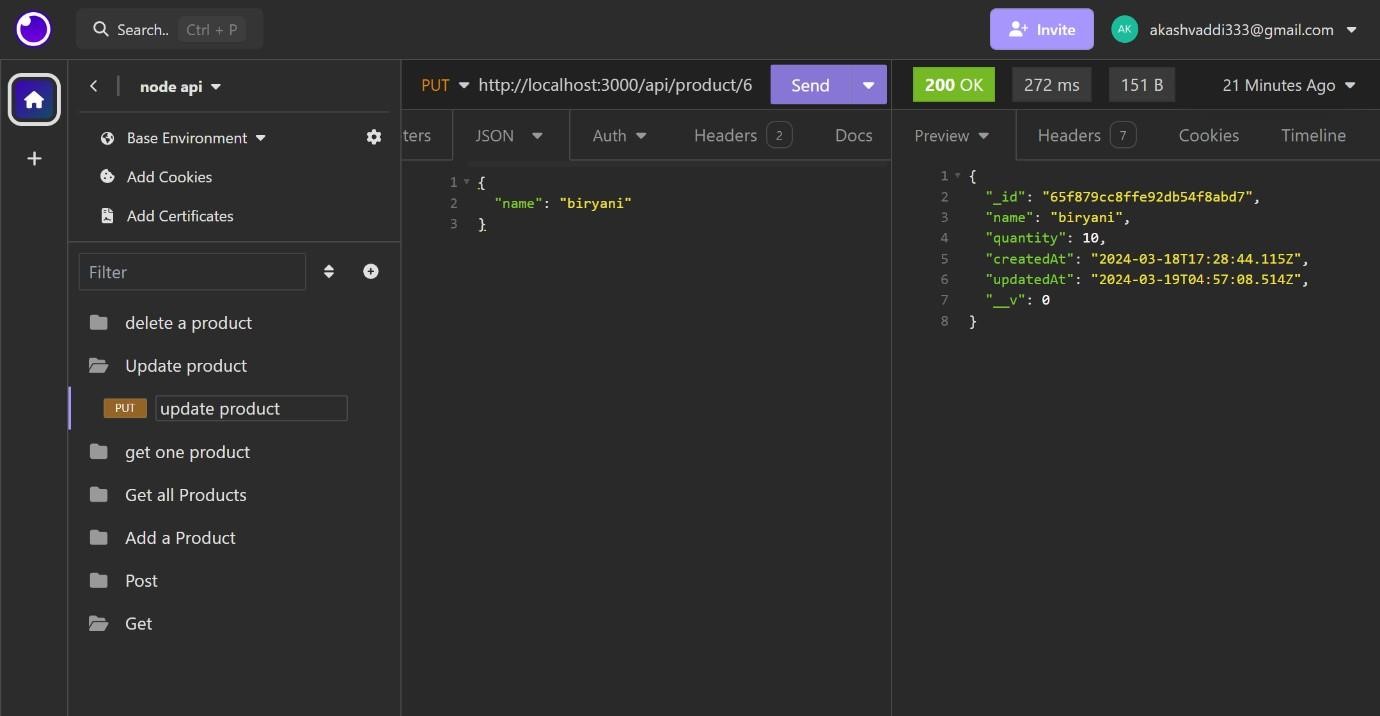
Read Api:



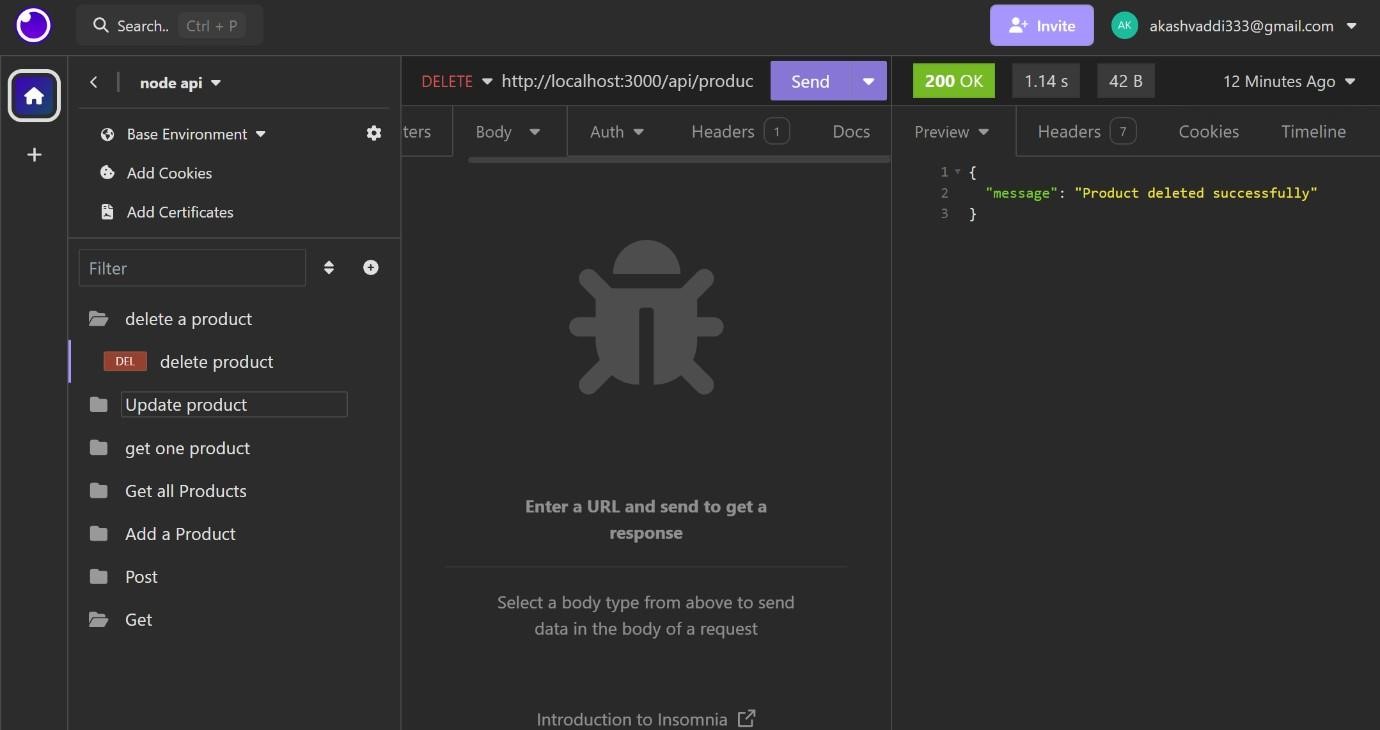
Read one Api:



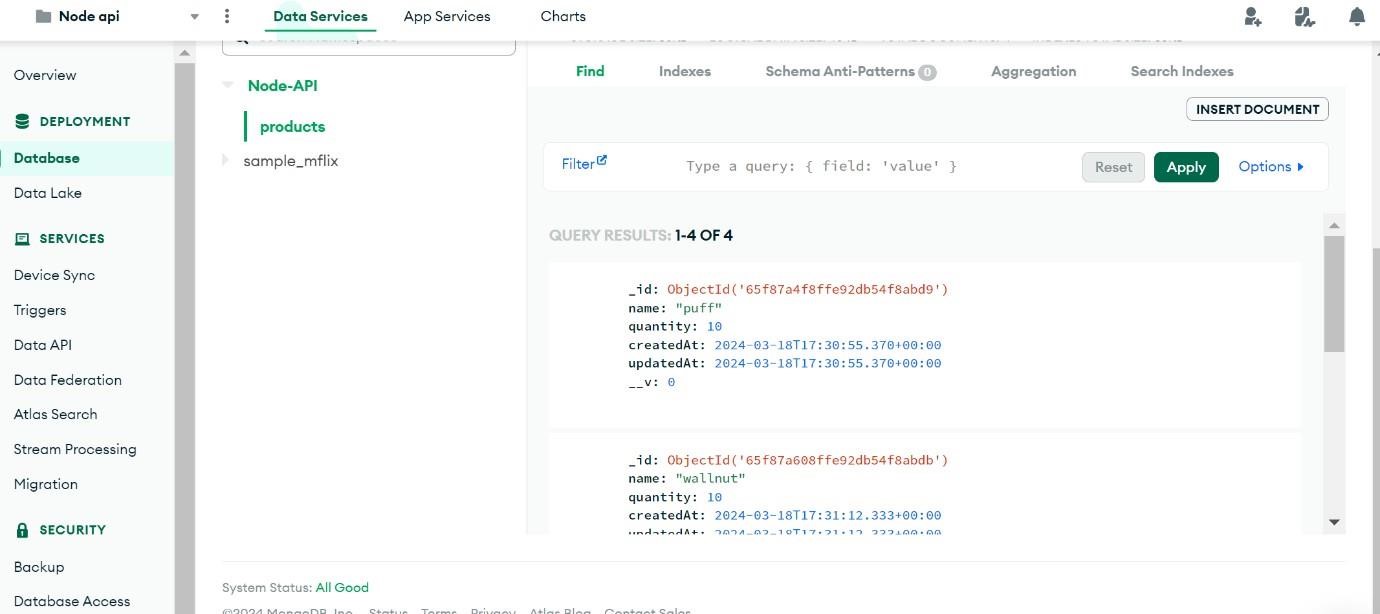
Update Api:



Delete Api:



MongoDB : (final view)



AKA QPI:

