Name	Akshansh Kaundal
Sec	622 - B
UID	23BCS13369

EXPERIMENT 5.1

Title

CRUD Operations for Product Database Using Mongoose

Objective

Learn how to implement basic Create, Read, Update, and Delete (CRUD) operations on a MongoDB collection using Mongoose in Node.js. This task helps you understand schema design, database connectivity, and handling data in a structured, real-world manner.

Task Description

Create a Node.js application that connects to a MongoDB database using Mongoose. Define a

Product model with properties such as name, price, and category. Implement routes or functions to perform CRUD operations: add a new product, retrieve all products, update a product by its ID, and delete a product by its ID. Use appropriate Mongoose methods for each operation and ensure that all data validations and error handling are included.

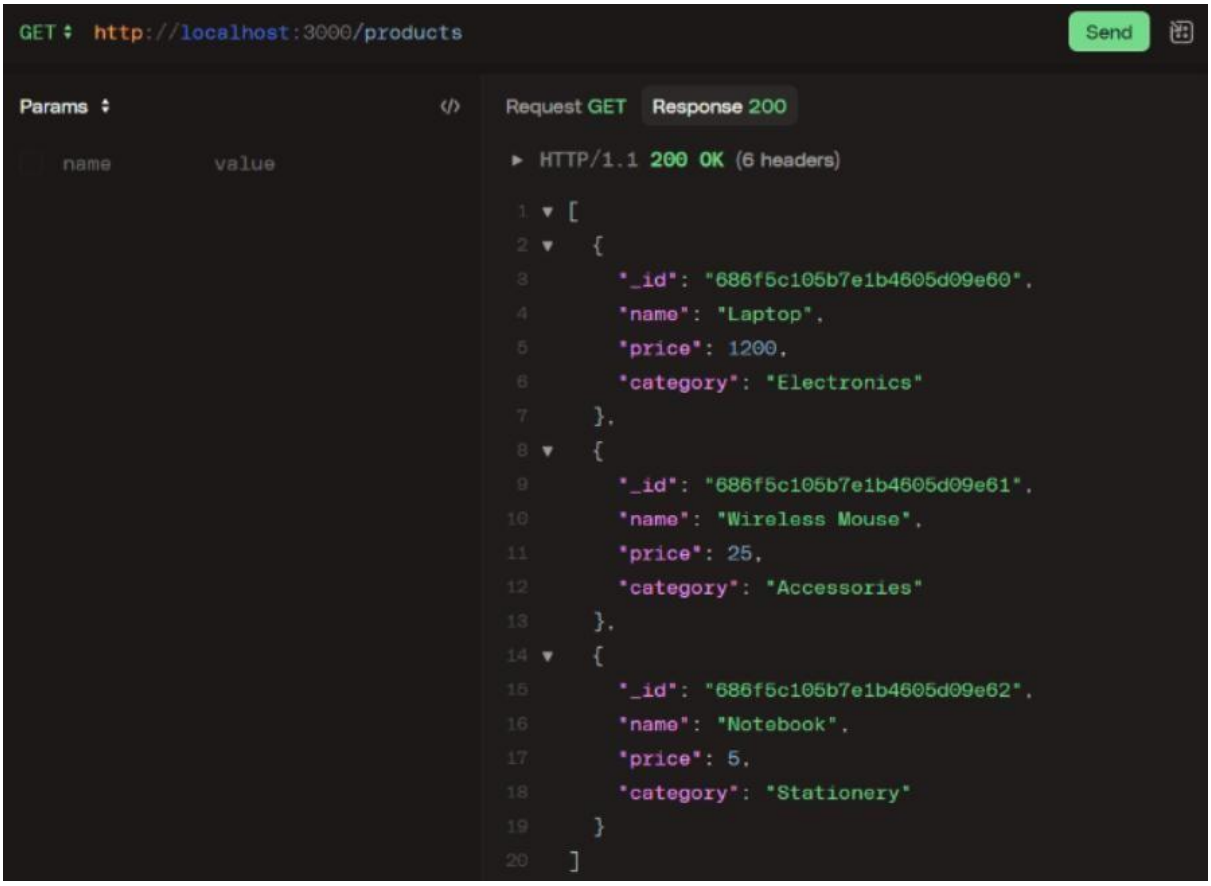
Code

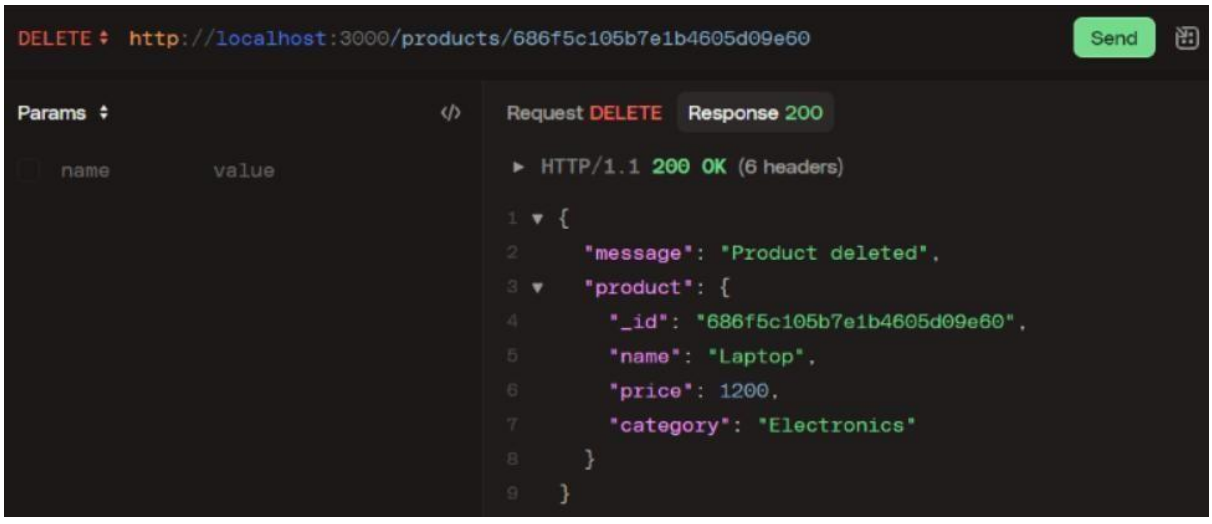
```
// Import dependencies const
mongoose = require('mongoose'); //
Connect to MongoDB
mongoose.connect('mongodb://localho
```

```
st:27017/productDB',                    {
useNewUrlParser:                    true,
useUnifiedTopology: true
});

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

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

// CREATE: Add a new product async function
addProduct(name, price, category) {   const product = new
Product({ name, price, category });   await product.save();
console.log('Product added:', product);
}

// READ: Retrieve all products async
function getAllProducts() {   const products
= await Product.find();   console.log('All
Products:', products); }  // UPDATE:
Update a product by ID async function
updateProduct(id, updateData)
```

```
{   const product = await Product.findByIdAndUpdate(id, updateData, { new: true });
console.log('Updated Product:', product);
}

// DELETE: Delete a product by ID async function
deleteProduct(id) {   await
Product.findByIdAndDelete(id);
console.log('Product deleted with ID:', id);
}

// Example usage (async () => {   await
addProduct('Laptop', 1200, 'Electronics');   await
getAllProducts();

  const products = await Product.find();   if
(products.length > 0) {     const productId =
products[0]._id;     await updateProduct(productId,
{ price: 1300 });     await deleteProduct(productId);
  }

  mongoose.connection.close();
})();
```

OUTPUT :


GET http://localhost:3000/products
Send
Params
name value
Request GET
Response 200
HTTP/1.1 200 OK (6 headers)
[
{
"_id": "686f5c105b7e1b4605d09e60",
"name": "Laptop",
"price": 1200,
"category": "Electronics"
},
{
"_id": "686f5c105b7e1b4605d09e61",
"name": "Wireless Mouse",
"price": 25,
"category": "Accessories"
},
{
"_id": "686f5c105b7e1b4605d09e62",
"name": "Notebook",
"price": 5,
"category": "Stationery"
}
]


DELETE http://localhost:3000/products/686f5c105b7e1b4605d09e60
Send
Params
name value
Request DELETE
Response 200
HTTP/1.1 200 OK (6 headers)
{
"message": "Product deleted",
"product": {
"_id": "686f5c105b7e1b4605d09e60",
"name": "Laptop",
"price": 1200,
"category": "Electronics"
}
}