

EXP-11

- **AIM:-** To implement Create, Read, Update, and Delete (CRUD) operations for a product database using Mongoose (an ODM for MongoDB) in a Node.js environment.
- **THEORY:-**
- Mongoose: An Object Data Modeling (ODM) library for MongoDB and Node.js.
 - Provides schema-based data modeling.
 - Simplifies validation, type casting, and query building.
- CRUD Operations:
 - Create: Add new documents (products) into the database.
 - Read: Retrieve product details from the database.
 - Update: Modify existing product details.
 - Delete: Remove product data from the database.
- Schema & Model in Mongoose:
 - Schema defines the structure of documents.
 - Model provides an interface to interact with MongoDB collections.
- **CODE:-**

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

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

```
const bodyParser = require('body-parser');
```

```
const app = express();
```

```
app.use(bodyParser.json());
```

```
// MongoDB connection
```

```
mongoose.connect("mongodb://127.0.0.1:27017/productDB", {
```

```
  useNewUrlParser: true,
```

```
  useUnifiedTopology: true
```

```
}).then(() => console.log("✅ Connected to MongoDB"))
```

```
.catch(err => console.log(err));
```

```
// Schema definition

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

// Model

const Product = mongoose.model("Product", productSchema);

// --- CRUD APIs ---

// Create (POST)

app.post('/products', async (req, res) => {
  const product = new Product(req.body);
  await product.save();
  res.send("Product added successfully!");
});

// Read (GET)

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

// Update (PUT)
```

```
app.put('/products/:id', async (req, res) => {
  await Product.findByIdAndUpdate(req.params.id, req.body);
  res.send("Product updated successfully!");
});
```

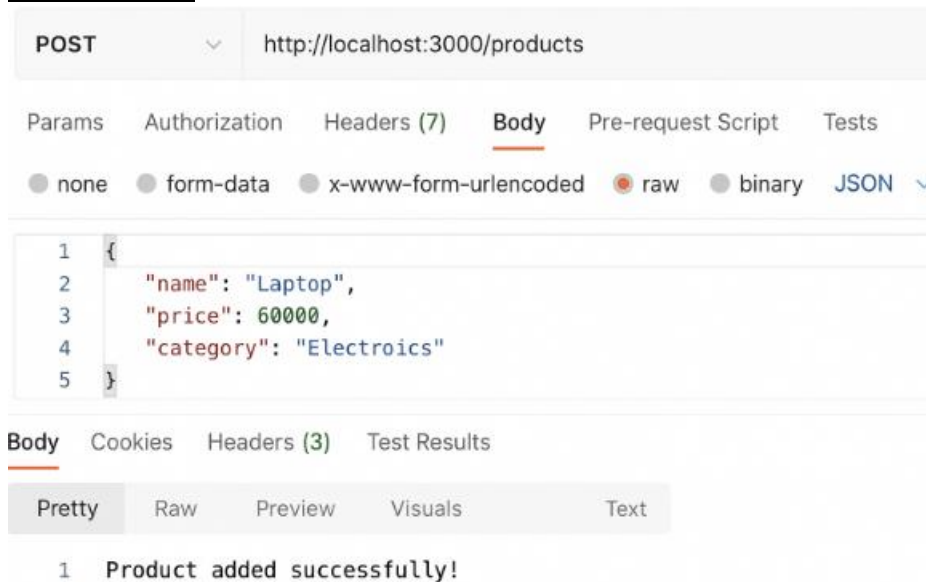
// Delete (DELETE)

```
app.delete('/products/:id', async (req, res) => {
  await Product.findByIdAndDelete(req.params.id);
  res.send("Product deleted successfully!");
});
```

// Server listen

```
app.listen(3000, () => console.log("🚀 Server running on port 3000"));
```

- **OUTPUT: -**



- **LEARNING OUTCOMES:-**

- ✓ Understood CRUD operations in a NoSQL database using Mongoose.
- ✓ Learned how to define a Mongoose schema and model.
- ✓ Practiced building a RESTful API with Express.js.
- ✓ Learned how to interact with MongoDB using Mongoose queries.

- ✓ Gained hands-on experience in server-side programming for database operations.