EXP-4.1:

CRUD Operations for Product Database Using Mongoose:

```
// productApp.js
const express = require("express");
const mongoose = require("mongoose");

const app = express();
app.use(express.json());

// 1. Connect to MongoDB
mongoose.connect("mongodb://127.0.0.1:27017/productDB", {
  useNewUrlParser: true,
  useUnifiedTopology: true,
})
.then(() => console.log("✅ MongoDB connected"))
.catch(err => console.error("❌ Connection error:", err));

// 2. Define Product Schema
const productSchema = new mongoose.Schema({
  name: { type: String, required: true },
  price: { type: Number, required: true },
  category: String,
  stock: { type: Number, default: 0 }
});

// 3. Create Model
const Product = mongoose.model("Product", productSchema);

// ---------- CRUD Operations ----------

// CREATE - Add new product
app.post("/products", async (req, res) => {
  try {
    const product = new Product(req.body);
    await product.save();
    res.status(201).json(product);
  } catch (err) {
    res.status(400).json({ error: err.message });
```

```
  }
});

// READ - Get all products
app.get("/products", async (req, res) => {
  try {
    const products = await Product.find();
    res.json(products);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});

// READ - Get product by ID
app.get("/products/:id", async (req, res) => {
  try {
    const product = await Product.findById(req.params.id);
    if (!product) return res.status(404).json({ error: "Product not found" });
    res.json(product);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});

// UPDATE - Update product by ID
app.put("/products/:id", async (req, res) => {
  try {
    const product = await Product.findByIdAndUpdate(
      req.params.id,
      req.body,
      { new: true, runValidators: true }
    );
    if (!product) return res.status(404).json({ error: "Product not found" });
    res.json(product);
  } catch (err) {
    res.status(400).json({ error: err.message });
  }
});

// DELETE - Delete product by ID
```

```
app.delete("/products/:id", async (req, res) => {
  try {
    const product = await Product.findByIdAndDelete(req.params.id);
    if (!product) return res.status(404).json({ error: "Product not found" });
    res.json({ message: "Product deleted" });
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});

// ------------------------------------------------------

// Start server
app.listen(3000, () => {
  console.log("🚀 Server running on http://localhost:3000");
});
```

OUTPUTS:

ID	Name	Price	Category	Stock
64a1	Laptop	80000	Electronics	10
64a2	Phone	50000	Electronics	25
64a3	Shoes	3000	Fashion	50