Experiment-10

CRUD Operations for Product Database Using Mongoose

Code Implementation

This file contains a Node.js script that demonstrates the four basic **CRUD** (Create, Read, Update, Delete) operations on a MongoDB database using the Mongoose ODM (Object Data Modeling) library.

```
// Import the mongoose library
const mongoose = require('mongoose');


// Define the MongoDB connection URI.
// This connects to a local database named 'productDB'.
const mongoURI = 'mongodb://127.0.0.1:27017/productDB';


// Define the schema for our Product model
const productSchema = new mongoose.Schema({
  name: {
    type: String,
    required: true,
    trim: true
  },
  price: {
    type: Number,
    required: true,
    min: 0
  },
  description: {
    type: String,
    required: false
  },
  inStock: {
    type: Boolean,
    default: true
  }
});

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

// Main function to run the CRUD operations
async function runCRUDOperations() {
  try {
    // Connect to the MongoDB database
    await mongoose.connect(mongoURI);
    console.log('Successfully connected to MongoDB.');
    // --- CREATE ---
```

```
console.log('\n--- CREATING new product... ---');
const newProduct = new Product({
  name: 'Laptop Pro',
  price: 1200,
  description: 'A high-performance laptop for professionals.'
});
const savedProduct = await newProduct.save();
console.log('Product created successfully:', savedProduct);
const productId = savedProduct._id; // Save the ID for later operations


// --- READ ---
console.log('\n--- READING all products... ---');
const allProducts = await Product.find();
console.log('Found products:', allProducts);


// --- UPDATE ---
console.log(`\n--- UPDATING product with ID: ${productId}... ---`);
const updatedProduct = await Product.findByIdAndUpdate(
  productId,
  { price: 1150, inStock: false },
  { new: true } // This option returns the modified document
);
console.log('Product updated successfully:', updatedProduct);
// --- DELETE ---
```

```
    console.log(`\n--- DELETING product with ID: ${productId}... ---`);
    const deletedProduct = await Product.findByIdAndDelete(productId);
    console.log('Product deleted successfully:', deletedProduct);

    // Verify deletion by trying to read all products again
    console.log('\n--- Verifying deletion, reading all products... ---');
    const productsAfterDelete = await Product.find();
    console.log('Products remaining:', productsAfterDelete);

  } catch (error) {
    console.error('An error occurred:', error);
  } finally {
    // Disconnect from the database
    await mongoose.disconnect();
    console.log('\nDisconnected from MongoDB.');
  }
}

// Run the main function
runCRUDOperations();
```