MongoDB Schema Design — Online Retail Platform

1. Product Catalog Design

Collection Name: products

Schema Design:

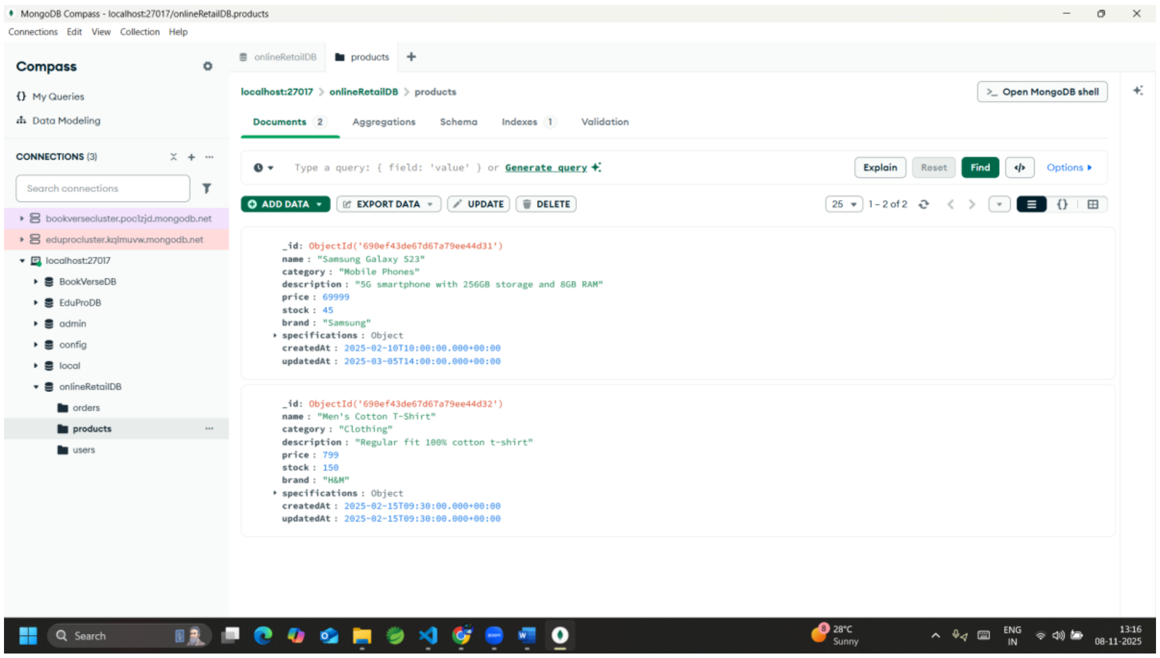
```
{

  _id: ObjectId,
  name: String,
  category: String,
  description: String,
  price: Number,
  stock: Number,
  brand: String,
  specifications: Object,
  createdAt: ISODate,
  updatedAt: ISODate
}
```

Sample Documents:

```
// Example 1: Electronics Product
{

        name: "Samsung Galaxy S23",
        category: "Mobile Phones",
        description: "5G smartphone with 256GB storage and 8GB RAM",
        price: 69999,
        stock: 45,
        brand: "Samsung",
        specifications: {
            color: "Phantom Black",
            screenSize: "6.1 inches",
            battery: "3900 mAh"
        },
        createdAt: new Date("2025-02-10T10:00:00Z"),
        updatedAt: new Date("2025-03-05T14:00:00Z")
    }
```

```
// Example 2: Fashion Product
{

        name: "Men's Cotton T-Shirt",
        category: "Clothing",
        description: "Regular fit 100% cotton t-shirt",
        price: 799,
        stock: 150,
        brand: "H&M",
        specifications: {
            size: ["S", "M", "L", "XL"],
            color: "Navy Blue"
        },
        createdAt: new Date("2025-02-15T09:30:00Z"),
        updatedAt: new Date("2025-02-15T09:30:00Z")
    }
```

Sample Output:



```
_id: ObjectId('690ef43de67d67a79ee44d31')
name : "Samsung Galaxy S23"
category : "Mobile Phones"
description : "5G smartphone with 256GB storage and 8GB RAM"
price : 69999
stock : 45
brand : "Samsung"
specifications : Object
createdAt : 2025-02-10T10:00:00.000+00:00
updatedAt : 2025-03-05T14:00:00.000+00:00

_id: ObjectId('690ef43de67d67a79ee44d32')
name : "Men's Cotton T-Shirt"
category : "Clothing"
description : "Regular fit 100% cotton t-shirt"
price : 799
stock : 150
brand : "H&M"
specifications : Object
createdAt : 2025-02-15T09:30:00.000+00:00
updatedAt : 2025-02-15T09:30:00.000+00:00
```

2. Customer Orders Design

Collection Name: orders

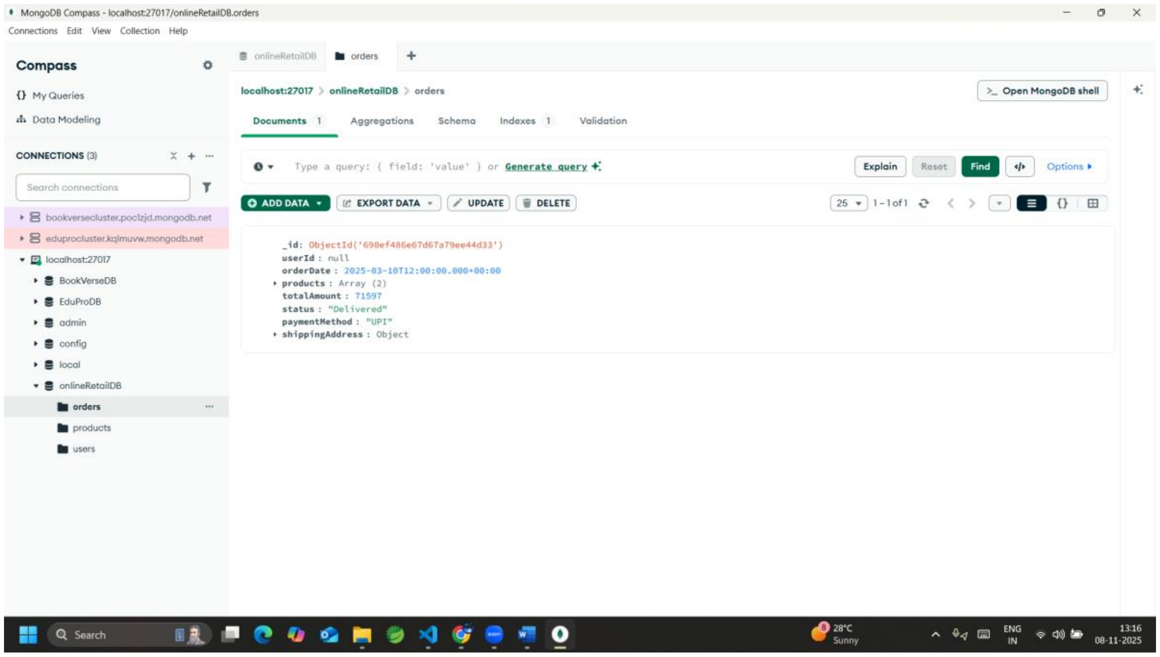
Schema Design:

```
{
  _id: ObjectId,
  userId: ObjectId,
  orderDate: ISODate,
  products: [
    { productId: ObjectId, quantity: Number, priceAtPurchase: Number }
  ],
  totalAmount: Number,
  status: String,
  paymentMethod: String,
  shippingAddress: {
    street: String,
    city: String,
    state: String,
    postalCode: String,
    country: String
  }
}
```

Sample Document:

```
{

  _id: ObjectId("6749f3b92b8a3c11a1d4b301"),
  userId: ObjectId("6749f2f72b8a3c11a1d4b501"),
  orderDate: ISODate("2025-03-10T12:00:00Z"),
  products: [
    { productId: ObjectId("6749f2a12b8a3c11a1d4b201"), quantity: 1,
priceAtPurchase: 69999 },
    { productId: ObjectId("6749f2a12b8a3c11a1d4b202"), quantity: 2,
priceAtPurchase: 799 }
  ],
  totalAmount: 71597,
  status: "Delivered",
  paymentMethod: "UPI",
  shippingAddress: {
    street: "221B Baker Street",
    city: "Mumbai",
    state: "Maharashtra",
    postalCode: "400001",
    country: "India"
  }
}
```

Sample Output:



```
_id: ObjectId('690ef486e67d67a79ee44d33')
userId : null
orderDate : 2025-03-10T12:00:00.000+00:00
▸ products : Array (2)
totalAmount : 71597
status : "Delivered"
paymentMethod : "UPI"
▸ shippingAddress : Object
```

3. User Authentication Design

Collection Name: users

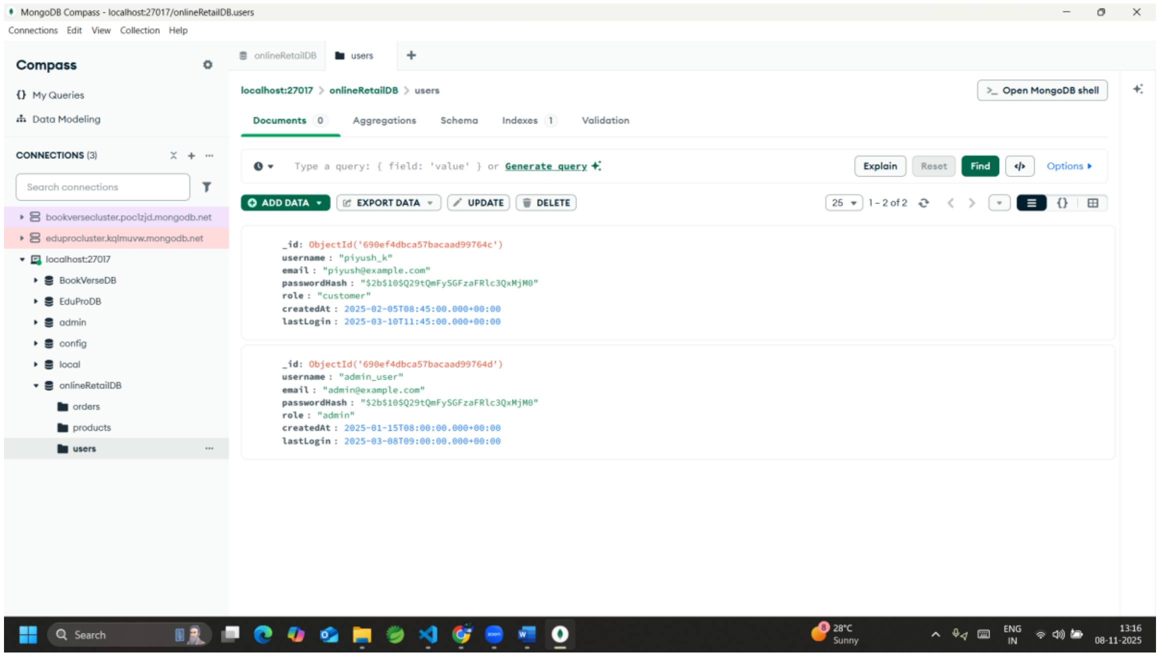
Schema Design:

```
{

  _id: ObjectId,
  username: String,
  email: String,
  passwordHash: String,
  role: String,
  createdAt: ISODate,
  lastLogin: ISODate
}
```

Sample Document:

```
{

  _id: ObjectId("6749f2f72b8a3c11a1d4b501"),
  username: "piyush_k",
  email: "piyush@example.com",
  passwordHash: "$2b$10$Q29tQmFySGFzaFRlc3QxMjM0",
  role: "customer",
  createdAt: ISODate("2025-02-05T08:45:00Z"),
  lastLogin: ISODate("2025-03-10T11:45:00Z")
}
```

Sample Document:



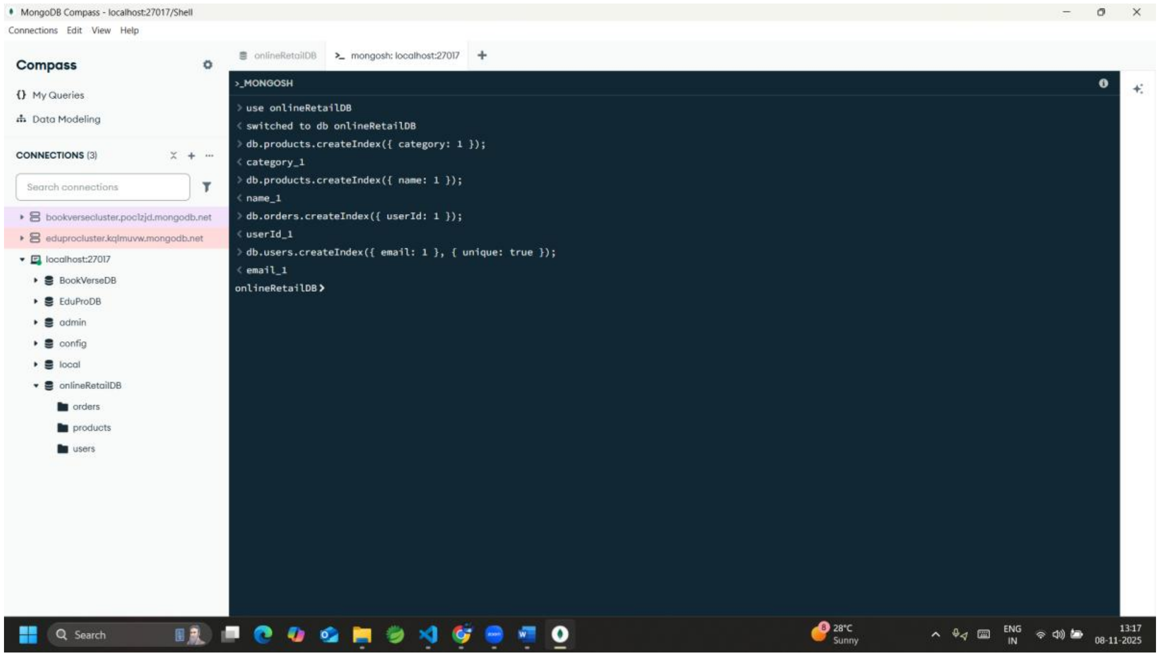
```
_id: ObjectId('690ef4dbca57bacaad99764c')
username : "piyush_k"
email : "piyush@example.com"
passwordHash : "$2b$10$Q29tQmFySGFzaFRlc3QxMjM0"
role : "customer"
createdAt : 2025-02-05T08:45:00.000+00:00
lastLogin : 2025-03-10T11:45:00.000+00:00

_id: ObjectId('690ef4dbca57bacaad99764d')
username : "admin_user"
email : "admin@example.com"
passwordHash : "$2b$10$Q29tQmFySGFzaFRlc3QxMjM0"
role : "admin"
createdAt : 2025-01-15T08:00:00.000+00:00
lastLogin : 2025-03-08T09:00:00.000+00:00
```

4. Querying and Indexing

Indexes:

```

db.products.createIndex({ category: 1 });

db.products.createIndex({ name: 1 });

db.orders.createIndex({ userId: 1 });
db.users.createIndex({ email: 1 }, { unique: true });
```

Sample Document:

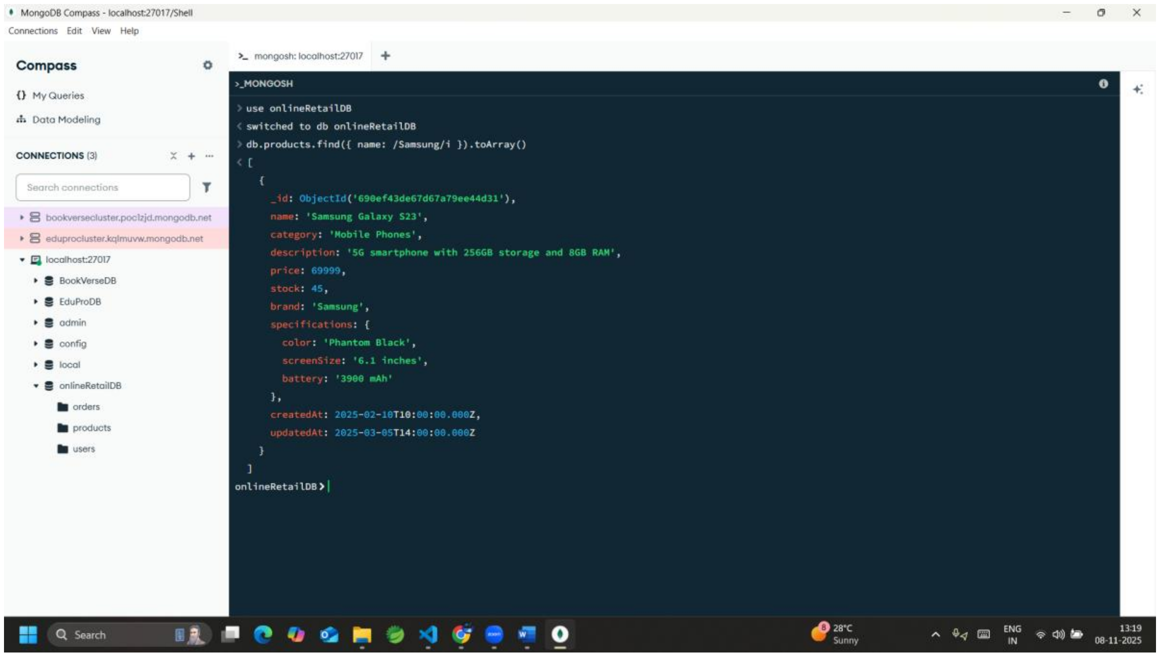


```
> use onlineRetailDB
< switched to db onlineRetailDB
> db.products.createIndex({ category: 1 });
< category_1
> db.products.createIndex({ name: 1 });
< name_1
> db.orders.createIndex({ userId: 1 });
< userId_1
> db.users.createIndex({ email: 1 }, { unique: true });
< email_1
onlineRetailDB >
```

5. Sample Queries

```
//1. Retrieve all products from a category:

db.products.find({ category: "Mobile Phones" }).pretty();
```

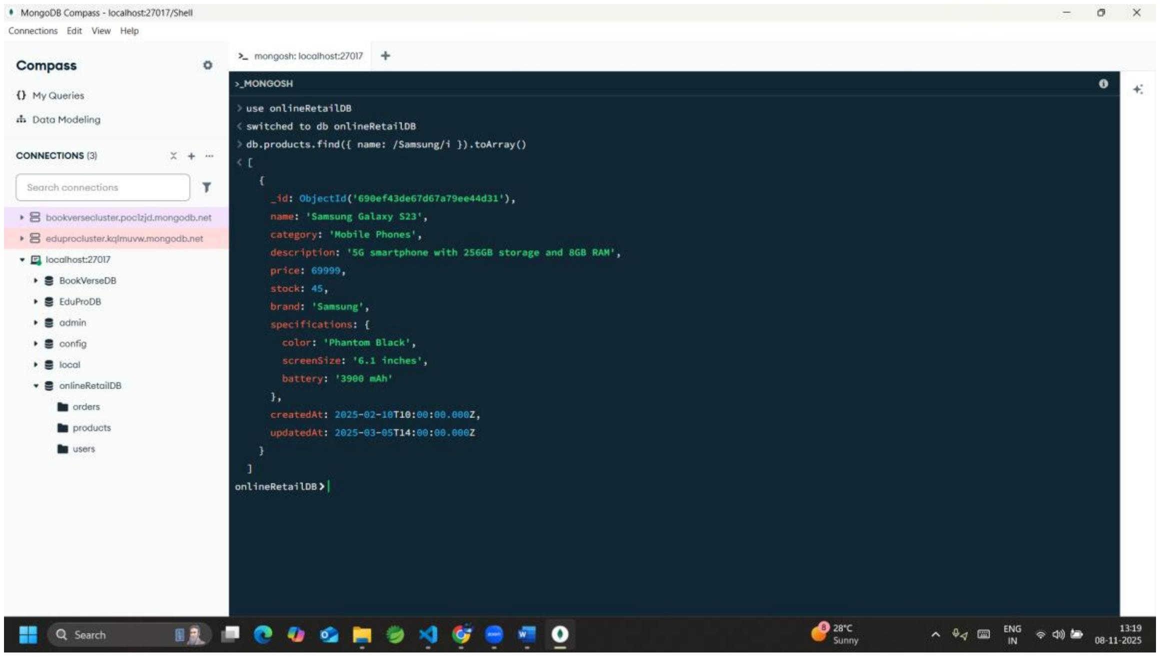
Sample Output:



```
> use onlineRetailDB
< switched to db onlineRetailDB
> db.products.find({ name: /Samsung/i }).toArray()
< [
    {
      _id: ObjectId('690ef43de67d67a79ee44d31'),
      name: 'Samsung Galaxy S23',
      category: 'Mobile Phones',
      description: '5G smartphone with 256GB storage and 8GB RAM',
      price: 69999,
      stock: 45,
      brand: 'Samsung',
      specifications: {
        color: 'Phantom Black',
        screenSize: '6.1 inches',
        battery: '3900 mAh'
      },
      createdAt: 2025-02-10T10:00:00.000Z,
      updatedAt: 2025-03-05T14:00:00.000Z
    }
  ]
onlineRetailDB >
```

```
//2. Find a specific product by name:
db.products.find({ name: /Samsung/i }, { name: 1, price: 1, stock: 1 });
```

Sample Output:



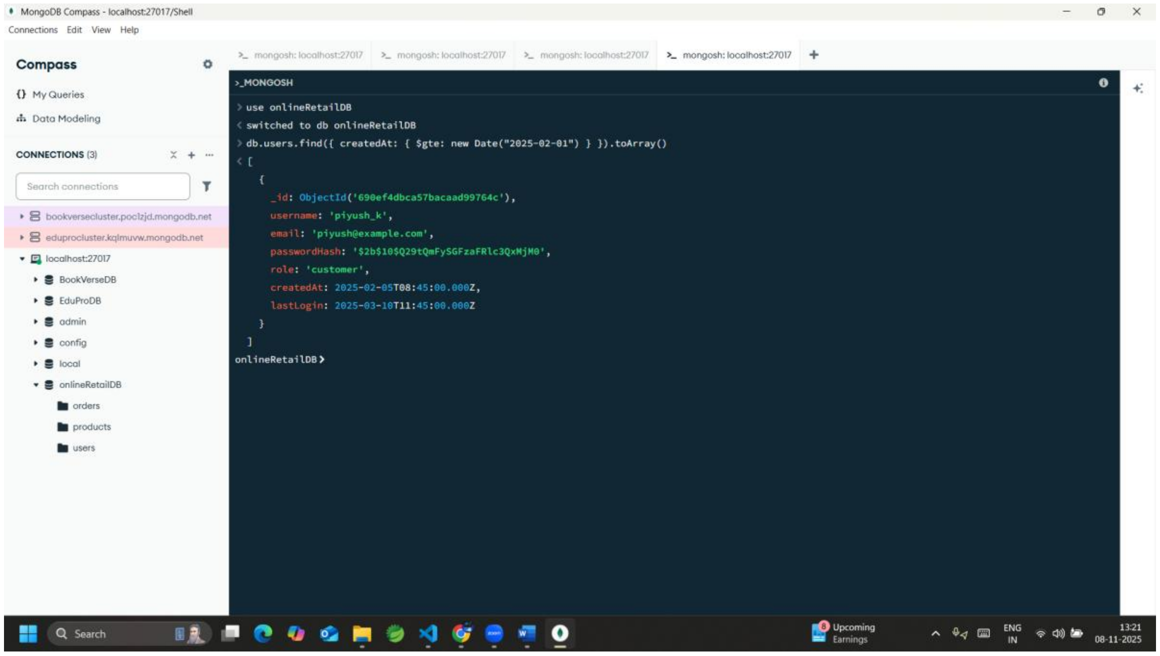
```
> use onlineRetailDB
< switched to db onlineRetailDB
> db.products.find({ name: /Samsung/i }).toArray()
< [
    {
      _id: ObjectId('690ef43de67d67a79ee44d31'),
      name: 'Samsung Galaxy S23',
      category: 'Mobile Phones',
      description: '5G smartphone with 256GB storage and 8GB RAM',
      price: 69999,
      stock: 45,
      brand: 'Samsung',
      specifications: {
        color: 'Phantom Black',
        screenSize: '6.1 inches',
        battery: '3900 mAh'
      },
      createdAt: 2025-02-10T10:00:00.000Z,
      updatedAt: 2025-03-05T14:00:00.000Z
    }
  ]
onlineRetailDB >
```

```
//3. Retrieve all orders for a specific user:
db.orders.find({ userId: ObjectId("690ef4dbca57bacaad99764c") });
```

Sample Output:

```
//4. Find all users registered after a date:
db.users.find({ createdAt: { $gte: ISODate("2025-02-01") } });
```

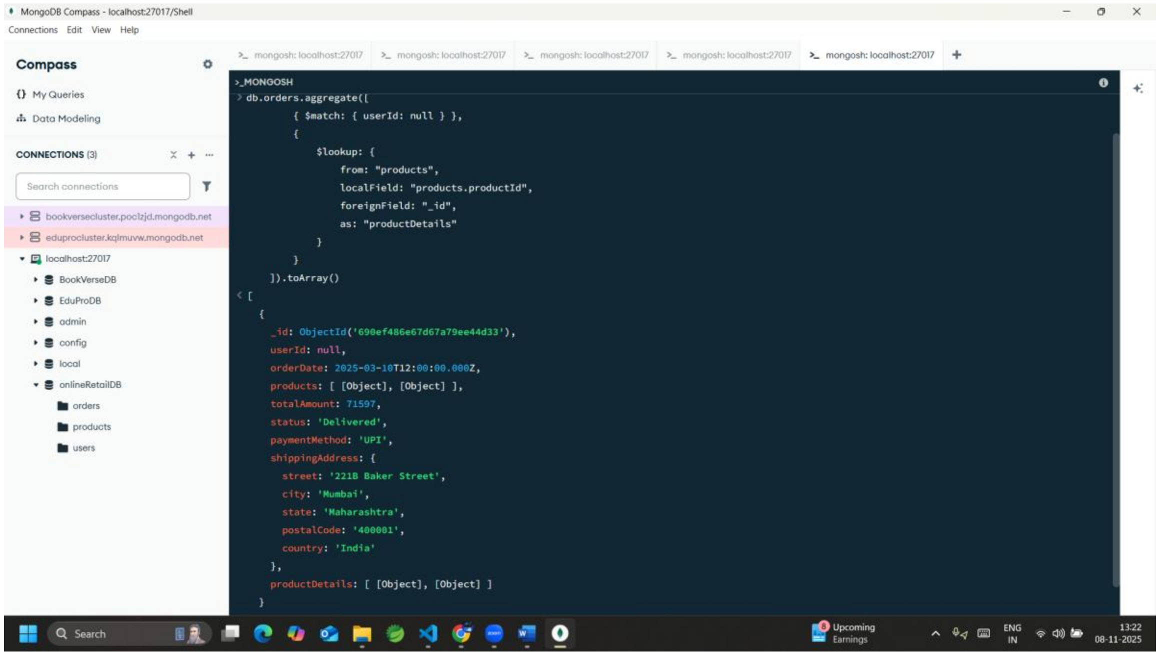
Sample Output:



```
> use onlineRetailDB
< switched to db onlineRetailDB
> db.users.find({ createdAt: { $gte: new Date("2025-02-01") } }).toArray()
< [
    {
      _id: ObjectId('690ef4dbca57bacaad99764c'),
      username: 'piyush_k',
      email: 'piyush@example.com',
      passwordHash: '$2b$10$Q29tQmFySGFzaFRlc3QxMjM0',
      role: 'customer',
      createdAt: 2025-02-05T08:45:00.000Z,
      lastLogin: 2025-03-10T11:45:00.000Z
    }
  ]
onlineRetailDB >
```

```
//5. Retrieve orders with product details(Aggregation):
db.orders.aggregate([
    { $match: { userId: ObjectId("6749f2f72b8a3c11a1d4b501") } },
    {
        $lookup: {
            from: "products",
            localField: "products.productId",
            foreignField: "_id",
            as: "productDetails"
        }
    }
]);
```

Sample Output:



```
> db.orders.aggregate([
        { $match: { userId: null } },
        {
            $lookup: {
                from: "products",
                localField: "products.productId",
                foreignField: "_id",
                as: "productDetails"
            }
        }
    ]).toArray()
< [
    {
      _id: ObjectId('690ef486e67d67a79ee44d33'),
      userId: null,
      orderDate: 2025-03-10T12:00:00.000Z,
      products: [ [Object], [Object] ],
      totalAmount: 71597,
      status: 'Delivered',
      paymentMethod: 'UPI',
      shippingAddress: {
        street: '221B Baker Street',
        city: 'Mumbai',
        state: 'Maharashtra',
        postalCode: '400001',
        country: 'India'
      },
      productDetails: [ [Object], [Object] ]
    }
```

6. Benefits of MongoDB for This Use Case

- Scalability: Horizontal scaling using sharding for millions of products.
- Flexibility: Supports dynamic attributes for different product types.
- Performance: Indexing optimizes search and retrieval.
- Aggregation: Efficient analytics and order summaries.
- Integration: Works seamlessly with Node.js and Mongoose ORM.