**EXPERIMENT NO 4.3**

## 🎯 Aim

To design and implement an E-commerce product catalog using nested document structures in MongoDB, demonstrating the use of embedded documents for efficient data modeling, retrieval, and querying.

## 📖 Theory

* MongoDB is a NoSQL, document-oriented database that stores data in flexible JSON-like documents.
* Nested Document Structure (or embedding) allows storing related data inside a single document instead of separate collections.
* In an E-commerce Catalog, a product usually has:
  + General details (name, category, price, description).
  + Variants (size, color, stock availability).
  + Customer reviews (rating, comments, user info).
* Using nested documents, all these can be stored inside one product document. This reduces the need for joins and improves query performance.

Example Benefits:

* One product = One document → Easier reads.
* Related data is grouped together.
* Faster aggregation for common queries (e.g., average rating).

CODE:

// Switch / create database

use ecommerceDB;

// Create collection

db.createCollection("products");

// Insert product with nested documents

db.products.insertOne({

product\_id: "P101",

name: "Smartphone XYZ",

category: "Electronics",

price: 25999,

description: "Latest 5G smartphone with AMOLED display",

variants: [

{ color: "Black", storage: "128GB", stock: 50 },

{ color: "Blue", storage: "256GB", stock: 30 }

],

reviews: [

{ user: "Amit", rating: 5, comment: "Excellent phone!" },

{ user: "Riya", rating: 4, comment: "Good but a bit pricey." }

]

});

// Query: Fetch all products in Electronics category

db.products.find({ category: "Electronics" }).pretty();

// Query: Fetch only product name and variants

db.products.find(

{ product\_id: "P101" },

{ name: 1, variants: 1, \_id: 0 }

);

// Update: Add new review

db.products.updateOne(

{ product\_id: "P101" },

{ $push: { reviews: { user: "Neha", rating: 3, comment: "Average battery life." } } }

);

// Aggregation: Calculate average rating of a product

db.products.aggregate([

{ $match: { product\_id: "P101" } },

{ $unwind: "$reviews" },

{ $group: { \_id: "$product\_id", avgRating: { $avg: "$reviews.rating" } } }

]);

## 📌 Learning Outcomes

After completing this experiment, you will be able to:

1. Understand the concept of nested documents in MongoDB.
2. Model an E-commerce catalog efficiently using embedded structures.
3. Perform CRUD operations (Insert, Query, Update) on nested fields.
4. Use aggregation pipelines to compute metrics like average rating.
5. Appreciate the difference between embedding vs referencing in MongoDB schema design.

