**Code-1:**

package connection;

import com.mongodb.client.MongoClient;

import com.mongodb.client.MongoClients;

import com.mongodb.client.MongoCollection;

import com.mongodb.client.MongoDatabase;

import com.mongodb.client.FindIterable;

import org.bson.Document;

import com.mongodb.client.model.Filters;

public class MongoDBConnection8 {

public static void main(String[] args) {

// Creating a Mongo client

MongoClient mongoClient = MongoClients.create("mongodb://localhost:27017");

MongoDatabase database = mongoClient.getDatabase("vit");

// Get the collection

MongoCollection<Document> collection = database.getCollection("sales");

// Find all documents

//collection.deleteOne(Filters.eq("",""));

collection.deleteMany(Filters.eq("item", "Cappuccino"));

System.out.println("Document deleted.");

System.out.println("\*\*\*Documents\*\*\*");

// Select a particular document (This line seems to be intended to find all documents after deletion)

FindIterable<Document> documents = collection.find();

for (Document document : documents) {

System.out.println(document);

}

}

}

**Code-2:**

package connection;

import com.mongodb.client.MongoClient;

import com.mongodb.client.MongoClients;

import com.mongodb.client.MongoCollection;

import com.mongodb.client.MongoDatabase;

import com.mongodb.client.FindIterable;

import org.bson.Document;

import com.mongodb.client.model.Filters;

public class MongoDBConnection9 {

public static void main(String[] args) {

MongoClient mongoClient = MongoClients.create("mongodb://localhost:27017");

MongoDatabase database = mongoClient.getDatabase("vit");

MongoCollection<Document> collection = database.getCollection("sales");

System.out.println("--- Documents (Skipping 1, Limiting 2) ---");

FindIterable<Document> limitedDocuments = collection.find().skip(1).limit(2);

for (Document document : limitedDocuments) {

System.out.println(document);

}

mongoClient.close();

System.out.println("--- Connection Closed ---");

}

}

**Code-3:**

package connection;

import com.mongodb.client.MongoClient;

import com.mongodb.client.MongoClients;

import com.mongodb.client.MongoCollection;

import com.mongodb.client.MongoDatabase;

import com.mongodb.client.FindIterable;

import org.bson.Document;

public class MongoDBEvenDocumentPrinter {

public static void main(String[] args) {

MongoClient mongoClient = MongoClients.create("mongodb://localhost:27017");

System.out.println("Connected to MongoDB successfully!");

MongoDatabase database = mongoClient.getDatabase("vit");

System.out.println("Accessed database: vit");

MongoCollection<Document> collection = database.getCollection("sales");

System.out.println("Accessed collection: sales");

FindIterable<Document> allDocuments = collection.find();

System.out.println("\n--- Processing Documents (Printing Every Other Document) ---");

int index = 0;

for (Document doc : allDocuments) {

if (index % 2 == 0) {

System.out.println("Document at index " + index + ": " + doc.toJson());

}

index++;

}

System.out.println("\n--- Document processing complete ---");

mongoClient.close();

System.out.println("MongoDB connection closed.");

}

}

**Code-4:**

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

public class SortAsc {

public static void main(String[] args) {

MongoClient mongoClient = MongoClients.create("mongodb://localhost:27017");

MongoDatabase database = mongoClient.getDatabase("vit");

// Get the collection

MongoCollection<Document> collection = database.getCollection("sales");

// Retrieve the documents and store them in a list

List<Document> documentList = new ArrayList<>();

for (Document doc : collection.find()) {

documentList.add(doc);

}

// Sort documents in descending order by pice

Collections.sort(documentList, (doc1, doc2) -> {

Double price1 = doc1.getDouble("price");

Double price2 = doc2.getDouble("price");

return price2.compareTo(price1); // Descending order

// for ascending order: return price1.compareTo(price2);

});

// Printing

for (Document doc : documentList) {

System.out.println("Sorted Document: " + doc);

}

}

}