**MongoDB Schema Design Challenge**

1. E-Commerce Store – Product & Orders

Ans:

const users = new mongoose.Schema(

    {

        name:{type:String, required:true},

        email:{type:String, required:true, unique:true},

        password:{type:String, required:true},

    }

);

const products = new mongoose.Schema(

    {

        title:{type:String, required:true},

        description:{type:String, required:true},

        price:{type:Number, required:true},

        category:{type:String, required:true},

        stock:{type:Number, required:true},

    }

);

const orders = new mongoose.Schema(

    {

        userid:{type: mongoose.Schema.ObjectId, ref:'users', required:true},

        productids:{type:String, required:true},

        total\_amount:{type:Number, required:true},

        orderDate:{type:Date, required:true},

    }

);

const reviews = new mongoose.Schema(

    {

        userid:{type: mongoose.Schema.ObjectId, ref:'users', required:true},

        productid:{type: mongoose.Schema.ObjectId, ref:'products', required:true},

        rating:{type:Number, min:1,max:5},

        comment:{type:String, required:true},

    }

);

2. Online Course Platform – Instructors & Students

Ans: const users = new mongoose.Schema(

    {

        name:{ type:String, required: true},

        email:{ type:String, unique:true,required: true},

        role:{ type:String,enum:['Student','Instructor'] ,required: true}

    }

);

const courses = new mongoose.Schema(

    {

        title:{ type:String, required: true},

        instructorid:{ type:String, required: true, unique:true},

        category:{ type:String, required: true},

        price:{ type:Number, required: true, min:0},

        creatredAt:{ type:Date, required: true}

    }

);

const enrollments = new mongoose.Schema(

    {

        title:{ type:String, required: true},

        videoURL:{ type:String, required: true},

        duration:{ type:Number, required: true}

    }

);

const lessons = new mongoose.Scema(

    {

        duration:{type:Number,min:1}

    }

);

3. Event Booking System – Organizers & Attendees

Ans:

const users = new mongoose.Schema(

    {

        name: { type:String, required: true},

        email: { type:String, required: true, unique:true},

        role: { type:String,enum:['organizer','attendee'], required: true},

    }

);

const events = new mongoose.Schema(

    {

        title: { type:String, required: true},

        organizerId: { type:String, required: true},

        location: { type:String, required: true},

        startTime: { type:Date, required: true},

        endTime: { type:Date, required: true},

        capacity: { type:Number, required: true, min:0},

    }

);

const bookings = new mongoose.Schema(

    {

        eventId: { type:String, required: true},

        attendeeId: { type:String, required: true},

        bookingDate: { type:Date, required: true}

    }

);

4. Blogging Platform – Authors & Articles

Ans:

const authors = new mongoose.Schema(

    {

        name:{type:String, required:true},

        email:{type:String, required:true, unique:true},

        bio:{type:String, required:true},

    }

);

const articles = new mongoose.Schema(

    {

        title:{type:String, required:true},

        content:{type:String, required:true},

        authorId:{type:String, required:true},

        tags:{type:Array, enum:['AI','nature','health'],required:true},

        published:{type:Boolean, required:true},

        createdAt:{type:Date, required:true},

    }

);

const comments = new mongoose.schema(

    {

        articleId:{type:String, required:true},

        userName:{type:String, required:true},

        commentText:{type:String, required:true},

        postedAt:{type:Date, required:true},

    }

);

5. Subscription App – Users & Plans

Ans:

const user = new mongoose.Schema(

    {

    name: { type:String, required: true},

    email: { type:String, unique:true},

    signupDate: { type:Date,required:true}

    }

);

const plans = new mongoose.Schema(

    {

        name:{ type:String, required: true},

        price:{ type:Number, required: true, min:0},

        features:{ type:Array, required: true},

        billingCycle:{ type:String,enum:['monthly','yearly'], required: true},

    }

);

const subscriptions = new mongoose.Schema(

    {

        userId:{ type:String, required: true},

        planId:{ type:String, required: true},

        startDate:{ type:Date, required: true},

        isActive:{ type:Boolean, required: true},

    }

);