**Mern Stack**

**Name:** C Akil Shanmuga Sriram

**Unique ID:** E0222023

**Project Title:** Car Pooling Platform

**MongoDB CRUD Operations for Car Pooling Platform**

**Table of Contents**

1. **Users Collection CRUD**
2. **Rides Collection CRUD**
3. **Bookings Collection CRUD**
4. **Messages Collection CRUD**
5. **Ratings Collection CRUD**
6. **Payments Collection CRUD**
7. **Notifications Collection CRUD**
8. **Safety Reports Collection CRUD**

**Users Collection CRUD**

**CREATE Operations**

**Create New User Registration**

db.users.insertOne({

email: "newuser@email.com",

password: "$2b$10$hashedpassword",

profile: {

firstName: "Jane",

lastName: "Smith",

phone: "+1234567890",

dateOfBirth: new Date("1995-03-15"),

profilePicture: "https://example.com/profiles/jane.jpg",

verified: false

},

preferences: {

role: "rider",

smokingAllowed: false,

musicPreference: "pop",

chattiness: "moderate",

maxDetour: 10

},

vehicle: null,

emergencyContact: {

name: "John Smith",

phone: "+1234567891",

relationship: "Father"

},

location: {

address: "123 New Street, Coimbatore, Tamil Nadu",

coordinates: {

type: "Point",

coordinates: [76.9600, 11.0200]

}

},

rating: {

average: 0,

totalRatings: 0

},

createdAt: new Date(),

updatedAt: new Date(),

isActive: true

});

**READ Operations**

**Find User by Email (Login)**

javascript

db.users.findOne(  
 { email: "[john.doe@email.com](mailto:john.doe@email.com)" },  
 { password: 1, profile: 1, isActive: 1 }  
);

**Get User Profile**

javascript

db.users.findOne(  
 { \_id: ObjectId("64f8a1b2c3d4e5f6a7b8c9d0") },  
 { password: 0 }  
);

**Find Drivers Near Location**

javascript

db.users.find({  
 "location.coordinates": {  
 $near: {  
 $geometry: {  
 type: "Point",  
 coordinates: [76.9558, 11.0168]  
 },  
 $maxDistance: 5000  
 }  
 },  
 "preferences.role": { $in: ["driver", "both"] },  
 isActive: true,  
 vehicle: { $ne: null }  
});

**UPDATE Operations**

**Update User Profile**

javascript

db.users.updateOne(  
 { \_id: ObjectId("64f8a1b2c3d4e5f6a7b8c9d0") },  
 {  
 $set: {  
 "profile.phone": "+9876543210",  
 "preferences.musicPreference": "rock",  
 updatedAt: new Date()  
 }  
 }  
);

**Update User Rating**

javascript

db.users.updateOne(  
 { \_id: ObjectId("64f8a1b2c3d4e5f6a7b8c9d0") },  
 {  
 $set: {  
 "rating.average": 4.3,  
 "rating.totalRatings": 25,  
 updatedAt: new Date()  
 }  
 }  
);

**DELETE Operations**

**Soft Delete User (Deactivate)**

javascript

db.users.updateOne(  
 { \_id: ObjectId("64f8a1b2c3d4e5f6a7b8c9d0") },  
 {  
 $set: {  
 isActive: false,  
 updatedAt: new Date()  
 }  
 }  
);

**Rides Collection CRUD**

**CREATE Operations**

**Create New Ride Offer**

javascript

db.rides.insertOne({  
 driverId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d0"),  
 route: {  
 startLocation: {  
 address: "123 Main St, Coimbatore, Tamil Nadu",  
 coordinates: {  
 type: "Point",  
 coordinates: [76.9558, 11.0168]  
 }  
 },  
 endLocation: {  
 address: "Tech Park, Saravanampatti, Coimbatore",  
 coordinates: {  
 type: "Point",  
 coordinates: [77.0073, 11.0677]  
 }  
 },  
 estimatedDistance: 15.2,  
 estimatedDuration: 35  
 },  
 schedule: {  
 departureTime: new Date("2025-07-25T08:00:00Z"),  
 isRecurring: false,  
 recurringDays: [],  
 recurringEndDate: null  
 },  
 capacity: {  
 totalSeats: 3,  
 availableSeats: 3  
 },  
 pricing: {  
 costPerSeat: 150,  
 currency: "INR"  
 },  
 preferences: {  
 smokingAllowed: false,  
 musicAllowed: true,  
 maxDetour: 15,  
 genderPreference: "any"  
 },  
 status: "active",  
 createdAt: new Date(),  
 updatedAt: new Date()  
});

**READ Operations**

**Find Available Rides by Route**

javascript

db.rides.find({  
 "route.startLocation.coordinates": {  
 $near: {  
 $geometry: {  
 type: "Point",  
 coordinates: [76.9558, 11.0168]  
 },  
 $maxDistance: 2000  
 }  
 },  
 "capacity.availableSeats": { $gt: 0 },  
 status: "active",  
 "schedule.departureTime": { $gte: new Date() }  
}).sort({ "schedule.departureTime": 1 });

**Get Driver's Active Rides**

javascript

db.rides.find({  
 driverId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d0"),  
 status: "active"  
}).sort({ "schedule.departureTime": 1 });

**UPDATE Operations**

**Update Available Seats (After Booking)**

javascript

db.rides.updateOne(  
 { \_id: ObjectId("65f8a1b2c3d4e5f6a7b8c9e0") },  
 {  
 $inc: { "capacity.availableSeats": -1 },  
 $set: { updatedAt: new Date() }  
 }  
);

**Update Ride Status**

javascript

db.rides.updateOne(  
 { \_id: ObjectId("65f8a1b2c3d4e5f6a7b8c9e0") },  
 {  
 $set: {  
 status: "completed",  
 updatedAt: new Date()  
 }  
 }  
);

**DELETE Operations**

**Cancel Ride (Soft Delete)**

javascript

db.rides.updateOne(  
 { \_id: ObjectId("65f8a1b2c3d4e5f6a7b8c9e0") },  
 {  
 $set: {  
 status: "cancelled",  
 updatedAt: new Date()  
 }  
 }  
);

**Bookings Collection CRUD**

**CREATE Operations**

**Create New Booking Request**

javascript

db.bookings.insertOne({  
 rideId: ObjectId("65f8a1b2c3d4e5f6a7b8c9e0"),  
 riderId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d1"),  
 driverId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d0"),  
 status: "requested",  
 seatsBooked: 1,  
 pickupLocation: {  
 address: "456 Oak Ave, Coimbatore, Tamil Nadu",  
 coordinates: {  
 type: "Point",  
 coordinates: [76.9650, 11.0200]  
 }  
 },  
 dropoffLocation: {  
 address: "Tech Park, Saravanampatti, Coimbatore",  
 coordinates: {  
 type: "Point",  
 coordinates: [77.0073, 11.0677]  
 }  
 },  
 estimatedPickupTime: new Date("2025-07-24T08:10:00Z"),  
 estimatedDropoffTime: new Date("2025-07-24T08:45:00Z"),  
 totalCost: 150,  
 paymentStatus: "pending",  
 requestedAt: new Date(),  
 specialRequests: "Please call when you arrive"  
});

**READ Operations**

**Get User's Booking History**

javascript

db.bookings.find({  
 riderId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d1")  
}).sort({ requestedAt: -1 });

**Get Driver's Pending Booking Requests**

javascript

db.bookings.find({  
 driverId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d0"),  
 status: "requested"  
}).sort({ requestedAt: 1 });

**UPDATE Operations**

**Confirm Booking**

javascript

db.bookings.updateOne(  
 { \_id: ObjectId("66f8a1b2c3d4e5f6a7b8c9f0") },  
 {  
 $set: {  
 status: "confirmed",  
 confirmedAt: new Date()  
 }  
 }  
);

**Complete Booking**

javascript

db.bookings.updateOne(  
 { \_id: ObjectId("66f8a1b2c3d4e5f6a7b8c9f0") },  
 {  
 $set: {  
 status: "completed",  
 actualDropoffTime: new Date(),  
 completedAt: new Date()  
 }  
 }  
);

**DELETE Operations**

**Cancel Booking**

javascript

db.bookings.updateOne(  
 { \_id: ObjectId("66f8a1b2c3d4e5f6a7b8c9f0") },  
 {  
 $set: {  
 status: "cancelled",  
 cancellationReason: "User cancelled"  
 }  
 }  
);

**Messages Collection CRUD**

**CREATE Operations**

**Send New Message**

javascript

db.messages.insertOne({  
 conversationId: "conv\_64f8a1b2c3d4e5f6a7b8c9d0\_64f8a1b2c3d4e5f6a7b8c9d1",  
 senderId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d1"),  
 receiverId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d0"),  
 rideId: ObjectId("65f8a1b2c3d4e5f6a7b8c9e0"),  
 bookingId: ObjectId("66f8a1b2c3d4e5f6a7b8c9f0"),  
 content: "Hi! I'm running 5 minutes late. Please wait for me.",  
 messageType: "text",  
 isRead: false,  
 sentAt: new Date()  
});

**READ Operations**

**Get Conversation Messages**

javascript

db.messages.find({  
 conversationId: "conv\_64f8a1b2c3d4e5f6a7b8c9d0\_64f8a1b2c3d4e5f6a7b8c9d1"  
}).sort({ sentAt: 1 });

**Get Unread Messages for User**

javascript

db.messages.find({  
 receiverId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d0"),  
 isRead: false  
}).sort({ sentAt: -1 });

**UPDATE Operations**

**Mark Message as Read**

javascript

db.messages.updateOne(  
 { \_id: ObjectId("67f8a1b2c3d4e5f6a7b8c900") },  
 {  
 $set: {  
 isRead: true,  
 readAt: new Date()  
 }  
 }  
);

**Ratings Collection CRUD**

**CREATE Operations**

**Create New Rating**

javascript

db.ratings.insertOne({  
 bookingId: ObjectId("66f8a1b2c3d4e5f6a7b8c9f0"),  
 raterId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d1"),  
 ratedUserId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d0"),  
 ratingType: "driver",  
 rating: 5,  
 review: "Excellent driver! Very punctual and safe.",  
 categories: {  
 punctuality: 5,  
 cleanliness: 5,  
 communication: 4,  
 safety: 5  
 },  
 isAnonymous: false,  
 createdAt: new Date()  
});

**READ Operations**

**Get User's Ratings**

javascript

db.ratings.find({  
 ratedUserId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d0")  
}).sort({ createdAt: -1 });

**Get Average Rating for User**

javascript

db.ratings.aggregate([  
 {  
 $match: { ratedUserId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d0") }  
 },  
 {  
 $group: {  
 \_id: "$ratedUserId",  
 averageRating: { $avg: "$rating" },  
 totalRatings: { $sum: 1 },  
 averagePunctuality: { $avg: "$categories.punctuality" },  
 averageCleanliness: { $avg: "$categories.cleanliness" }  
 }  
 }  
]);

**UPDATE Operations**

**Update Rating**

javascript

db.ratings.updateOne(  
 {   
 \_id: ObjectId("68f8a1b2c3d4e5f6a7b8c910"),  
 createdAt: { $gte: new Date(Date.now() - 24 \* 60 \* 60 \* 1000) }  
 },  
 {  
 $set: {  
 rating: 4,  
 review: "Good driver, minor delay but overall positive experience."  
 }  
 }  
);

**Payments Collection CRUD**

**CREATE Operations**

**Create New Payment Transaction**

javascript

db.payments.insertOne({  
 bookingId: ObjectId("66f8a1b2c3d4e5f6a7b8c9f0"),  
 payerId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d1"),  
 receiverId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d0"),  
 amount: 150,  
 currency: "INR",  
 paymentMethod: "digital\_wallet",  
 transactionId: "TXN\_" + Date.now(),  
 status: "pending",  
 paymentGateway: "DummyPay",  
 createdAt: new Date()  
});

**READ Operations**

**Get Payment History for User**

javascript

db.payments.find({  
 $or: [  
 { payerId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d1") },  
 { receiverId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d1") }  
 ]  
}).sort({ createdAt: -1 });

**Get Pending Payments**

javascript

db.payments.find({  
 status: "pending"  
}).sort({ createdAt: 1 });

**UPDATE Operations**

**Update Payment Status to Completed**

javascript

db.payments.updateOne(  
 { \_id: ObjectId("69f8a1b2c3d4e5f6a7b8c920") },  
 {  
 $set: {  
 status: "completed",  
 gatewayTransactionId: "dummy\_txn\_001",  
 processedAt: new Date()  
 }  
 }  
);

**Process Refund**

javascript

db.payments.updateOne(  
 { \_id: ObjectId("69f8a1b2c3d4e5f6a7b8c925") },  
 {  
 $set: {  
 status: "refunded",  
 refundAmount: 100,  
 refundReason: "Driver cancelled - car breakdown",  
 processedAt: new Date()  
 }  
 }  
);

**Notifications Collection CRUD**

**CREATE Operations**

**Create New Notification**

javascript

db.notifications.insertOne({  
 userId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d0"),  
 type: "booking\_request",  
 title: "New Booking Request",  
 message: "Sarah Wilson has requested to book your ride to Tech Park",  
 relatedEntityId: ObjectId("66f8a1b2c3d4e5f6a7b8c9f0"),  
 relatedEntityType: "booking",  
 isRead: false,  
 isPush: true,  
 isEmail: true,  
 isSMS: false,  
 sentAt: new Date()  
});

**READ Operations**

**Get Unread Notifications for User**

javascript

db.notifications.find({  
 userId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d0"),  
 isRead: false  
}).sort({ sentAt: -1 });

**UPDATE Operations**

**Mark Notification as Read**

javascript

db.notifications.updateOne(  
 { \_id: ObjectId("70f8a1b2c3d4e5f6a7b8c930") },  
 {  
 $set: {  
 isRead: true,  
 readAt: new Date()  
 }  
 }  
);

**Safety Reports Collection CRUD**

**CREATE Operations**

**Create New Safety Report**

javascript

db.safety\_reports.insertOne({  
 reporterId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d1"),  
 reportedUserId: ObjectId("64f8a1b2c3d4e5f6a7b8c9d8"),  
 rideId: ObjectId("65f8a1b2c3d4e5f6a7b8c9e4"),  
 bookingId: ObjectId("66f8a1b2c3d4e5f6a7b8c9f0"),  
 incidentType: "unsafe\_driving",  
 description: "Driver was using phone while driving and not following traffic rules",  
 location: {  
 coordinates: {  
 type: "Point",  
 coordinates: [76.9735, 11.0161]  
 },  
 address: "Gandhipuram, Coimbatore"  
 },  
 evidence: ["<https://example.com/evidence_photo.jpg>"],  
 status: "open",  
 priority: "high",  
 createdAt: new Date()  
});

**READ Operations**

**Get All Open Safety Reports**

javascript

db.safety\_reports.find({  
 status: { $in: ["open", "investigating"] }  
}).sort({ priority: 1, createdAt: 1 });

**Get High Priority Reports**

javascript

db.safety\_reports.find({  
 priority: { $in: ["high", "critical"] },  
 status: { $ne: "closed" }  
}).sort({ createdAt: 1 });

**UPDATE Operations**

**Resolve Safety Report**

javascript

db.safety\_reports.updateOne(  
 { \_id: ObjectId("71f8a1b2c3d4e5f6a7b8c940") },  
 {  
 $set: {  
 status: "resolved",  
 resolution: "Driver was given safety training and warning. Account monitored.",  
 resolvedAt: new Date()  
 }  
 }  
);

**General CRUD Best Practices**

**Indexing for Performance**

javascript

*// Essential Indexes*  
db.users.createIndex({ "email": 1 }, { unique: true });  
db.users.createIndex({ "location.coordinates": "2dsphere" });  
db.rides.createIndex({ "driverId": 1, "status": 1 });  
db.bookings.createIndex({ "riderId": 1, "status": 1 });  
db.messages.createIndex({ "conversationId": 1, "sentAt": 1 });  
db.ratings.createIndex({ "ratedUserId": 1, "createdAt": -1 });  
db.payments.createIndex({ "bookingId": 1 }, { unique: true });  
db.notifications.createIndex({ "userId": 1, "isRead": 1 });  
db.safety\_reports.createIndex({ "status": 1, "priority": 1 });

**Error Handling Examples**

javascript

*// Example with error handling for user creation*  
try {  
 const result = await db.users.insertOne(newUserDocument);  
 if (result.acknowledged) {  
 console.log("User created successfully:", result.insertedId);  
 }  
} catch (error) {  
 if (error.code === 11000) {  
 console.error("Email already exists");  
 } else {  
 console.error("Error creating user:", error.message);  
 }  
}