**Mern Stack**

**Name:** C Akil Shanmuga Sriram

**Unique ID:** E0222023

**Project Title:** Car Pooling Platform

**MongoDB Data Modeling**

**1. Users Collection**

**Purpose**: Store user profiles, authentication details, and preferences  
 **Modeling Approach**: Embedded documents for user preferences and vehicle info

{

"\_id": "ObjectId",

"email": "string",

"password": "string (hashed)",

"profile": {

"firstName": "string",

"lastName": "string",

"phone": "string",

"dateOfBirth": "date",

"profilePicture": "string (URL)",

"verified": "boolean"

},

"preferences": {

"role": "string (enum: 'driver', 'rider', 'both')",

"smokingAllowed": "boolean",

"musicPreference": "string",

"chattiness": "string (enum: 'quiet', 'moderate', 'chatty')",

"maxDetour": "number (minutes)"

},

"vehicle": {

"make": "string",

"model": "string",

"year": "number",

"color": "string",

"licensePlate": "string",

"seats": "number"

},

"emergencyContact": {

"name": "string",

"phone": "string",

"relationship": "string"

},

"location": {

"address": "string",

"coordinates": {

"type": "Point",

"coordinates": [ "number (longitude)", "number (latitude)" ]

}

},

"rating": {

"average": "number",

"totalRatings": "number"

},

"createdAt": "date",

"updatedAt": "date",

"isActive": "boolean"

}

**2. Rides Collection**

**Purpose**: Store ride offers from drivers **Modeling Approach**: Referenced documents for user relationships, embedded for route details

json

{  
 "\_id": ObjectId,  
 "driverId": "ObjectId (ref: Users)",  
 "route": {  
 "startLocation": {  
 "address": "string",  
 "coordinates": {  
 "type": "Point",  
 "coordinates": [longitude, latitude]  
 }  
 },  
 "endLocation": {  
 "address": "string",  
 "coordinates": {  
 "type": "Point",  
 "coordinates": [longitude, latitude]  
 }  
 },  
 "waypoints": [  
 {  
 "address": "string",  
 "coordinates": {  
 "type": "Point",  
 "coordinates": [longitude, latitude]  
 }  
 }  
 ],  
 "estimatedDistance": "number (km)",  
 "estimatedDuration": "number (minutes)"  
 },  
 "schedule": {  
 "departureTime": "date",  
 "isRecurring": "boolean",  
 "recurringDays": ["enum [monday, tuesday, ...]"],  
 "recurringEndDate": "date"  
 },  
 "capacity": {  
 "totalSeats": "number",  
 "availableSeats": "number"  
 },  
 "pricing": {  
 "costPerSeat": "number",  
 "currency": "string"  
 },  
 "preferences": {  
 "smokingAllowed": "boolean",  
 "musicAllowed": "boolean",  
 "maxDetour": "number (minutes)",  
 "genderPreference": "enum [any, male, female]"  
 },  
 "status": "enum [active, completed, cancelled]",  
 "createdAt": "date",  
 "updatedAt": "date"  
}

**3. Bookings Collection**

**Purpose**: Track ride bookings and their status **Modeling Approach**: Referenced documents for scalability and data integrity

json

{  
 "\_id": ObjectId,  
 "rideId": "ObjectId (ref: Rides)",  
 "riderId": "ObjectId (ref: Users)",  
 "driverId": "ObjectId (ref: Users)",  
 "status": "enum [requested, confirmed, cancelled, completed]",  
 "seatsBooked": "number",  
 "pickupLocation": {  
 "address": "string",  
 "coordinates": {  
 "type": "Point",  
 "coordinates": [longitude, latitude]  
 }  
 },  
 "dropoffLocation": {  
 "address": "string",  
 "coordinates": {  
 "type": "Point",  
 "coordinates": [longitude, latitude]  
 }  
 },  
 "estimatedPickupTime": "date",  
 "estimatedDropoffTime": "date",  
 "actualPickupTime": "date",  
 "actualDropoffTime": "date",  
 "totalCost": "number",  
 "paymentStatus": "enum [pending, completed, refunded]",  
 "requestedAt": "date",  
 "confirmedAt": "date",  
 "completedAt": "date",  
 "cancellationReason": "string",  
 "specialRequests": "string"  
}

**4. Messages Collection**

**Purpose**: Handle communication between drivers and riders **Modeling Approach**: Referenced documents for user relationships

json

{  
 "\_id": ObjectId,  
 "conversationId": "string",  
 "senderId": "ObjectId (ref: Users)",  
 "receiverId": "ObjectId (ref: Users)",  
 "rideId": "ObjectId (ref: Rides)",  
 "bookingId": "ObjectId (ref: Bookings)",  
 "content": "string",  
 "messageType": "enum [text, system, location]",  
 "isRead": "boolean",  
 "sentAt": "date",  
 "readAt": "date"  
}

**5. Ratings Collection**

**Purpose**: Store user ratings and reviews **Modeling Approach**: Referenced documents for independent rating management

json

{  
 "\_id": ObjectId,  
 "bookingId": "ObjectId (ref: Bookings)",  
 "raterId": "ObjectId (ref: Users)",  
 "ratedUserId": "ObjectId (ref: Users)",  
 "ratingType": "enum [driver, rider]",  
 "rating": "number (1-5)",  
 "review": "string",  
 "categories": {  
 "punctuality": "number (1-5)",  
 "cleanliness": "number (1-5)",  
 "communication": "number (1-5)",  
 "safety": "number (1-5)"  
 },  
 "isAnonymous": "boolean",  
 "createdAt": "date"  
}

**6. Payments Collection**

**Purpose**: Track payment transactions (simplified for project/academic use) **Modeling Approach**: Referenced documents for audit trails and transaction history

json

{  
 "\_id": ObjectId,  
 "bookingId": "ObjectId (ref: Bookings)",  
 "payerId": "ObjectId (ref: Users)",  
 "receiverId": "ObjectId (ref: Users)",  
 "amount": "number",  
 "currency": "string",  
 "paymentMethod": "enum [credit\_card, digital\_wallet, upi]",  
 "transactionId": "string (simple sequential ID)",  
 "status": "enum [pending, completed, failed, refunded]",  
 "paymentGateway": "string (dummy gateway names)",  
 "gatewayTransactionId": "string (dummy gateway IDs)",  
 "refundAmount": "number",  
 "refundReason": "string",  
 "processedAt": "date",  
 "createdAt": "date"  
}

**Note**: This is a simplified payment system designed for project demonstration purposes. In a real-world implementation, this would include proper payment gateway integration, security tokens, encryption, and compliance with financial regulations.

**7. Notifications Collection**

**Purpose**: Handle push notifications and alerts **Modeling Approach**: Referenced documents for scalable notification system

json

{  
 "\_id": ObjectId,  
 "userId": "ObjectId (ref: Users)",  
 "type": "enum [booking\_request, booking\_confirmed, ride\_cancelled, payment\_received, safety\_alert]",  
 "title": "string",  
 "message": "string",  
 "relatedEntityId": "ObjectId",  
 "relatedEntityType": "enum [ride, booking, payment]",  
 "isRead": "boolean",  
 "isPush": "boolean",  
 "isEmail": "boolean",  
 "isSMS": "boolean",  
 "sentAt": "date",  
 "readAt": "date"  
}

**8. Safety\_Reports Collection**

**Purpose**: Handle safety incidents and emergency reports **Modeling Approach**: Referenced documents for security and audit purposes

json

{  
 "\_id": ObjectId,  
 "reporterId": "ObjectId (ref: Users)",  
 "reportedUserId": "ObjectId (ref: Users)",  
 "rideId": "ObjectId (ref: Rides)",  
 "bookingId": "ObjectId (ref: Bookings)",  
 "incidentType": "enum [harassment, unsafe\_driving, route\_deviation, emergency]",  
 "description": "string",  
 "location": {  
 "coordinates": {  
 "type": "Point",  
 "coordinates": [longitude, latitude]  
 },  
 "address": "string"  
 },  
 "evidence": ["string (URLs)"],  
 "status": "enum [open, investigating, resolved, closed]",  
 "priority": "enum [low, medium, high, critical]",  
 "assignedTo": "ObjectId (ref: Users)",  
 "resolution": "string",  
 "createdAt": "date",  
 "resolvedAt": "date"  
}

**Data Modeling Decisions**

**Embedded vs Referenced Documents**

**Embedded Documents Used For:**

1. **User Preferences & Vehicle Info**: Rarely updated independently, always queried together
2. **Route Details**: Tightly coupled with ride data, no need for separate queries
3. **Location Coordinates**: Atomic data that belongs to its parent entity
4. **Rating Categories**: Sub-components of the main rating

**Referenced Documents Used For:**

1. **User Relationships**: Users exist independently and are referenced across collections
2. **Ride-Booking Relationships**: Many-to-many relationship requiring separate management
3. **Messages**: High volume data that needs independent querying and pagination
4. **Payments**: Require audit trails and independent transaction management

**Indexing Strategy**

**Essential Indexes:**

1. **Users Collection**:
   1. { "email": 1 } - Unique index for authentication
   2. { "location.coordinates": "2dsphere" } - Geospatial queries
2. **Rides Collection**:
   1. { "driverId": 1, "status": 1 } - Driver's active rides
   2. { "route.startLocation.coordinates": "2dsphere" } - Location-based search
   3. { "schedule.departureTime": 1, "status": 1 } - Time-based queries
3. **Bookings Collection**:
   1. { "riderId": 1, "status": 1 } - User's booking history
   2. { "rideId": 1, "status": 1 } - Ride bookings
   3. { "status": 1, "requestedAt": 1 } - Admin monitoring

**Schema Benefits**

1. **Performance**: Embedded documents reduce joins for frequently accessed data
2. **Scalability**: Referenced documents allow independent scaling of high-volume collections
3. **Flexibility**: Hybrid approach supports both transactional and analytical queries
4. **Consistency**: Referenced relationships maintain data integrity
5. **Security**: Sensitive data (payments, safety reports) are isolated in separate collections

**Considerations for Scale**

1. **Sharding Strategy**: Shard rides collection by geographical region
2. **Read Replicas**: Use read replicas for search and analytics queries
3. **Data Archiving**: Move completed rides older than 6 months to archive collections
4. **Caching**: Implement application-level caching for user profiles and active rides

This data model provides a solid foundation for the car pooling platform while maintaining flexibility for future enhancements and scale requirements.