**Ride Management System**

**Final Report**

**Member1 : Praveen Aluru**

**Member2 : Niharika Balusu**

**Member3 : Yashasri Syamala**

**Member4 : Mary Lavanya Thumma**

**Project Overview**

The Ride Management System is a dynamic and user-centric platform designed to enhance the experience of ride-sharing services while ensuring efficiency and accountability. It empowers three distinct user roles: **Admins**, **Drivers**, and **Riders**, each with access to features tailored to their specific responsibilities.

**Core Purpose:** The system streamlines ridesharing by enabling efficient ride management, seamless bookings, and transparent revenue sharing among Admins, Drivers, and Riders. It fosters collaboration and ensures a smooth experience for all stakeholders.

**Key Stakeholders and Their Functions:**

1. **Admins**
   * Oversight of all ride-related activities and revenue tracking
   * Approval and management of driver registrations
   * Monitoring ride commissions and revenue distribution
2. **Drivers**
   * Registration and profile management
   * Creation, modification, and cancellation of rides
   * Management of ride bookings and schedules
   * Earning and tracking of ride revenue
3. **Riders**
   * Discovery of available rides based on preferences.
   * Booking and cancellation of rides
   * Managing personal booking history and schedules
   * Access to real-time ride availability and updates

**Primary System Features:**

1. **Ride Management**
   * Creation of rides by drivers, including pickup/drop-off points, start time, price per seat, available seats, and vehicle information
   * Centralized monitoring of all rides by administrators
   * Cancellation of rides by drivers with automated email notifications sent to booked riders
2. **Booking System**
   * Search functionality for riders to find available rides within a specified radius
   * Real-time ride availability updates for riders
   * Flexible payment options during booking
   * Personal booking history accessible to riders
3. **Revenue Tracking**
   * Automatic commission calculation for rides
   * Admin dashboard to view total earnings and commission details per ride
   * Earnings tracking for drivers, including a breakdown of completed rides
4. **User Management**
   * Driver registration with detailed profile setup, including license number and vehicle information
   * Admin approval system for driver accounts before they can create rides
   * Role-based access and functionalities for Admins, Drivers, and Riders
5. **Notification System**
   * Automated email notifications for ride cancellations by drivers, sent to affected riders
   * Booking confirmations and updates for riders

**Expected Benefits:**

1. **Operational Efficiency**
   * Streamlined ride management for admins
   * Automated commission tracking and revenue distribution
   * Reduced manual intervention in driver approval and ride creation
2. **Enhanced User Experience**
   * Intuitive, role-specific interfaces for Admins, Drivers, and Riders
   * Seamless ride booking and payment processes for riders
   * Notifications to keep users informed of ride cancellations or updates
3. **Transparency and Accountability**
   * Clear revenue-sharing model ensuring trust between admins and drivers
   * Accurate ride and booking history tracking for drivers and riders
   * Reliable communication via automated email notifications

This system aims to create a seamless, reliable, and user-friendly ride-sharing experience while maintaining operational control and fostering transparency among all stakeholders.

**Database Description**

The **Ride Management System** is designed to facilitate the organization, booking, and management of rides, ensuring efficient and user-friendly ridesharing services. The system uses MongoDB as the database.

**Key Tables and Relationships**

### Admin Table

* **Represents:** System administrators
* **Attributes:**
  + **admin\_id (PK):** Unique identifier (ObjectId or Auto-incremented ID)
  + **name:** Full name of the admin (String)
  + **email:** Unique email address for the admin (String)
  + **password:** Hashed password for security (String)
  + **phone\_number:** Contact phone number for the admin (String)
* **Relationships:**
  + Can manage multiple drivers, riders, and rides (Implied relationships with Driver, Rider, and Rides tables)
  + Can have multiple ride commission records (Implied relationship with Booking or Payment tables)

### Driver

* **Represents**: Information about the drivers registered in the system.
* **Attributes**:
  + **id (PK):** Unique identifier (ObjectId)
  + **userName**: Unique username for the driver
  + **password**: Hashed password
  + **email**: Unique email address
  + **licenseNumber**: License information for the driver
  + **phoneNumber**: Contact number
  + **vehicleInfo**: Information about the driver's vehicles
  + **createdAt**: Account creation timestamp
* **Relationships**:
  + Can have multiple **rides** (Rides table)
  + Can have multiple **bookings** (Booking table) as the driver
  + Can have multiple **earnings** (Booking table)

### Rider

* **Represents**: Information about the riders using the ride services.
* **Attributes**:
  + **id (PK):** Unique identifier (ObjectId)
  + **userName**: Unique username for the rider
  + **password**: Hashed password
  + **email**: Unique email address
  + **phoneNumber**: Contact number
  + **createdAt**: Account creation timestamp
* **Relationships**:
  + Can have multiple **bookings** (Booking table) as the rider
  + Can have multiple **payments** (Payment table)
  + Can have multiple **refunds** (Refund table)

### Booking

* **Represents**: Information about the bookings made by a rider for a particular ride and the earnings for both the driver and admin.
* **Attributes**:
  + **id (PK):** Unique identifier (ObjectId)
  + **adminCommission**: Commission earned by the admin
  + **driverEarnings**: Earnings for the driver
  + **driverId**: Reference to the driver (Driver table)
  + **rideId**: Reference to the ride (Rides table)
  + **riderId**: Reference to the rider (Rider table)
  + **paymentId**: Reference to the payment (Payment table)
* **Relationships**:
  + Associated with one **driver** (Driver table)
  + Associated with one **ride** (Rides table)
  + Associated with one **rider** (Rider table)
  + Associated with one **payment** (Payment table)

### Payment

* **Represents**: Payment records made by riders.
* **Attributes**:
  + **id (PK):** Unique identifier (ObjectId)
  + **paymentDate**: Date of the payment
  + **paymentMethod**: Method used for payment
  + **paymentStatus**: Payment status (Success/Failed)
  + **riderId**: Reference to the rider (Rider table)
* **Relationships**:
  + Associated with one **rider** (Rider table)
  + Can have multiple **bookings** (Booking table)

### Refund

* **Represents**: Refund information in case a ride is canceled.
* **Attributes**:
  + **id (PK):** Unique identifier (ObjectId)
  + **amountRefunded**: Amount refunded to the rider
  + **paymentId**: Reference to the payment (Payment table)
  + **bookingId**: Reference to the booking (Booking table)
  + **refundDate**: Date the refund was processed
  + **riderId**: Reference to the rider (Rider table)
  + **refundStatus**: Status of the refund (Processed/Failed)
* **Relationships**:
  + Associated with one **payment** (Payment table)
  + Associated with one **booking** (Booking table)
  + Associated with one **rider** (Rider table)

### Rides

* **Represents**: Information about the available rides for booking.
* **Attributes**:
  + **id (PK):** Unique identifier (ObjectId)
  + **availableseats**: Number of seats available for the ride
  + **createdAt**: Timestamp when the ride was created
  + **driverId**: Reference to the driver (Driver table)
  + **dropLocation**: Ride drop-off location
  + **pickupLocation**: Ride pickup location
  + **pricePerSeat**: Price for each seat
  + **startTime**: Time the ride is scheduled to start
  + **status**: Status of the ride (Scheduled/Completed/Cancelled)
  + **vehicleInfo**: Details about the vehicle for the ride
  + **vehicleId**: Reference to the vehicle (can be used if there's a separate table for vehicles)
* **Relationships**:
  + Associated with one **driver** (Driver table)
  + Can have multiple **bookings** (Booking table)
  + Can have multiple **riders** (Riders table, via the Booking table)

|  |  |
| --- | --- |
| Table Name | Description |
| Admin | Stores admin information and credentials for overseeing the system |
| Driver | Stores driver information and credentials including license number and vehicles’ details for overseeing the system. |
| Rider | Stores rider information and credentials for overseeing the system. |
| Booking | Holds information about bookings made by rider for a particular ride and the earnings of driver and admin |
| Payment | Records rider payments for riders, including the amount and method used. |
| Refund | Holds information regarding any refunds made in case of ride cancellation. |
| Rides | Stores all the rides information , ride details , list of riders |

**Admin Functions**

1. **Ride and Driver Management**
   * View and monitor all active rides, including ride details such as pickup/drop-off locations, times, prices, and available seats
   * Approve or reject driver registrations and manage driver profiles (vehicle details, license information)
   * Track driver performance, including completed rides and revenue
2. **Commission Tracking and Revenue Management**
   * Monitor commissions earned from completed rides (20% for admin, 80% for drivers)
   * Access detailed reports on total earnings, ride commissions, and payouts for both admin and drivers

**Driver Functions**

1. **Ride Creation**
   * Register with the system, add vehicle details, and provide license information
   * Create new rides by selecting pickup/drop-off points, start time, price per seat, and available seats
   * Edit or cancel existing rides
   * View and manage personal ride history and bookings
2. **Booking Management**
   * Monitor and manage rider bookings for each ride
   * View and track ride revenue, with a breakdown of commissions
   * Send automated cancellation notifications to riders in case of ride cancellations

**Rider Functions**

1. **Ride Discovery and Booking**
   * Search for available rides within a set radius and filter by pickup location
   * Book rides, select payment options, and confirm reservations
   * View personal booking history and ride details
2. **Booking Management**
   * Cancel bookings if necessary and receive automated notifications of ride changes
   * Make payments for booked rides and track payment history

**General System Functions**

* **User Authentication**
  + Secure login and user authentication for all roles (Admin, Driver, Rider)
  + Password reset functionality
  + Role-based access control, ensuring appropriate permissions for each user type
* **Profile Management**
  + Update personal information, such as contact details, username, and password
  + Driver profile management, including vehicle and license information
* **Payment Processing**
  + Process ride payments, including the calculation of commissions for admin and driver
  + Refund to riders when ride is cancelled by driver
* **Ride Scheduling System**
  + Automatically check for scheduling conflicts for ride timings
  + Ensure ride availability based on driver schedules and seat capacity

**User Roles**

**1. Administrator (Admin)**

* **Primary Role**: Overall system management and administration
* **Access Level**: Full system access
* **Responsibilities**:
  + Approving or rejecting driver registrations
  + Monitoring all active rides and system usage
  + Overseeing commission tracking and revenue distribution
  + Managing and monitoring all ride activities and cancellations
* **Special Privileges**:
  + Can approve/reject driver registrations and view all driver details
  + Full access to all system reports, including earnings and commission data

**2. Driver**

* **Primary Role**: Ride creation and management, interacting with riders
* **Access Level**: Limited to managing their own rides and profile
* **Responsibilities**:
  + Registering with the system and adding vehicle details
  + Creating new rides, including selecting pickup/drop-off locations, times, and prices
  + Managing ride bookings, cancellations, and notifications for riders
  + Tracking personal earnings and commission breakdown
* **Special Privileges**:
  + Can create and cancel rides
  + Can view bookings and revenue details for each ride
  + Can update personal and vehicle information

**3. Rider**

* **Primary Role**: Searching for, booking, and managing rides
* **Access Level**: Basic user access
* **Responsibilities**:
  + Searching for available rides within a set radius
  + Booking rides, selecting payment options, and confirming reservations
  + Viewing and managing their booking history
  + Cancelling bookings if necessary and tracking payments
* **Special Privileges**:
  + Can search and book available rides
  + Can cancel bookings and receive notifications about cancellations
  + Can view personal ride history

**Key Role Interactions**:

* Admins oversee both drivers and riders, managing their activities and ensuring system functionality.
* Drivers directly interact with riders through ride creation, booking, and cancellations.
* Riders interact with the system primarily to search, book, and manage their rides, with notifications about cancellations or updates.

### **Customer Audience for the Ride Management System**

**Primary Target Audiences:**

1. **Ride Service Providers (Drivers)**
   * Professional drivers and vehicle owners
   * Small to medium-sized vehicle fleets
   * Independent drivers looking for flexible work opportunities
   * Commercial transportation companies offering rideshare services
2. **Riders (Passengers)**
   * Commuters seeking affordable, convenient travel options
   * Individuals looking for a flexible transportation service
   * Students and professionals seeking shared rides
3. **Corporate Clients**
   * Businesses needing group travel options for employees
   * Companies providing ride-sharing as a benefit to employees
   * Event organizers offering ride services for attendees
   * Corporate fleets seeking ride management systems

**Target Market Characteristics:**

1. **Size and Scale**
   * Small to medium-sized fleets (1-50 drivers)
   * Large ride-sharing platforms with hundreds of drivers
   * Independent drivers or small businesses offering carpooling
   * Corporate clients managing employee transportation
2. **Technical Requirements**
   * Basic mobile device and internet connectivity
   * User-friendly web app for booking and ride management
   * Secure payment systems for booking rides
   * Location-based services for accurate ride matching and route planning
3. **Industry Sectors**
   * Transportation and logistics
   * Shared economy and ride-hailing services
   * Corporate transportation management
   * Tourism and travel industry for long-distance rides
   * Urban mobility and smart city initiatives
4. **Geographical Scope**
   * Local to regional ride services (within a city or metropolitan area)

**Key Benefits for Each Audience:**

1. **For Ride Service Providers (Drivers)**
   * Flexible work schedule with ride creation and management
   * Opportunity to earn revenue based on rides completed
   * Automated notifications for cancellations, and updates
   * Commission tracking and payout management
   * Easy profile management (vehicle details, license, payment info)
2. **For Riders (Passengers)**
   * Convenient search and booking for available rides
   * Simple and secure payment options
   * Easy cancellation and notification system
   * Access to ride history and tracking
   * Enhanced convenience through ride-sharing for cost savings
3. **For Platform Operators (Admin)**
   * Complete oversight of driver registrations and ride management
   * Monitoring of commissions and system analytics
   * Automated conflict detection and booking management
   * Centralized control of ride approval and driver tracking
   * User management and detailed reporting features

**Market Positioning:**

* **Target Segment**: Focus on mid to high-end users, including both individual drivers/riders and larger corporate clients.
* **User Experience**: Emphasis on a seamless, user-friendly web app for both drivers and riders, designed for efficiency and ease of use.
* **Scalability**: Platform is scalable, from individual rides to large fleet operations and corporate integrations.
* **Cost-Effectiveness**: Offer a cost-effective ride-sharing solution, benefiting both drivers and riders by reducing travel costs.

**Summary**

The Ride Management System is an innovative digital platform designed to streamline the coordination of ride services. The system caters to a diverse set of users, including Administrators, Drivers, and Riders, with tailored functionality for each role. Admins have full control over the platform, managing driver registrations, monitoring active rides, and tracking commissions. Drivers can register, create and manage rides, and track earnings, while Riders can search for available rides, book seats, and manage their bookings. Key features include automated ride creation, real-time booking management, secure payment options, and automated cancellation notifications. The platform’s architecture ensures scalability, allowing it to support both small-scale ride-sharing and large fleet operations. With its intuitive interface and comprehensive administrative tools, the system significantly reduces manual oversight while enhancing the ride experience for both drivers and riders.

**Screenshots**

**A screenshot of a login page

Description automatically generated**

**A screenshot of a car with a phone and a phone

Description automatically generated**

**A screenshot of a login page

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a screenshot of a ride

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a chat

Description automatically generated**

**A screenshot of a credit card payment

Description automatically generated**

**A screenshot of a personal details

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**Document Data Model:**

**Admin**

admin\_id

name

email

password

phone\_number

**Rides**

ride\_id

driver\_id

pickup\_location

drop\_location

ride\_status

seats\_available

price\_per\_seat

ride\_time

riders\_id

**Drivers**

driver\_id

name

email

password

phone\_number

vehicle\_info

license\_number

created\_at

updated\_at

**Bookings**

booking\_id

driver\_id

ride\_id

rider\_id

pickup\_location

drop\_location

admin\_commission

driver\_earnings

payment\_id

**Riders**

rider\_id

name

email

password

phone\_number

created\_at

updated\_at

**Refunds**

refund\_id

booking\_id

rider\_id

amount\_refunded

refund\_status

refund\_date

payment\_id

**Payment**

payment\_id

rider\_id

payment\_method

payment\_status

payment\_date

**Sample Data**

{

"Driver": [

{

"id": "driver1",

"userName": "john\_doe\_driver",

"password": "hashedpassword123",

"email": "john.doe@example.com",

"licenseNumber": "D1234567",

"phoneNumber": "+1234567890",

"vehicleInfo": "Toyota Camry - Blue - 2019",

"createdAt": "2024-01-01T10:00:00Z"

}

],

"Rider": [

{

"id": "rider1",

"userName": "jane\_doe\_rider",

"password": "hashedpassword456",

"email": "jane.doe@example.com",

"phoneNumber": "+0987654321",

"createdAt": "2024-01-02T12:00:00Z"

}

],

"Booking": [

{

"id": "booking1",

"adminCommission": 10.0,

"driverEarnings": 40.0,

"driverId": "driver1",

"rideId": "ride1",

"riderId": "rider1",

"paymentId": "payment1"

}

],

"Payment": [

{

"id": "payment1",

"paymentDate": "2024-01-03T14:00:00Z",

"paymentMethod": "CREDIT\_CARD",

"paymentStatus": "COMPLETED",

"riderId": "rider1"

}

],

"Refund": [

{

"id": "refund1",

"amountRefunded": 10.0,

"paymentId": "payment1",

"bookingId": "booking1",

"refundDate": "2024-01-04T09:00:00Z",

"riderId": "rider1",

"refundStatus": "REFUNDED"

}

],

"Rides": [

{

"id": "ride1",

"availableSeats": 3,

"createdAt": "2024-01-01T08:00:00Z",

"driverId": "driver1",

"dropLocation": "456 Elm Street, Springfield",

"pickupLocation": "123 Maple Avenue, Springfield",

"pricePerSeat": 15.0,

"startTime": "2024-01-05T10:00:00Z",

"status": "SCHEDULED",

"vehicleInfo": "Toyota Camry - Blue - 2019",

"vehicleId": "vehicle1"

}

]

}

**MongoDB Shell Commands**

// Insert Driver data

db.Driver.insertOne({

"id": "driver1",

"userName": "john\_doe\_driver",

"password": "hashedpassword123",

"email": "john.doe@example.com",

"licenseNumber": "D1234567",

"phoneNumber": "+1234567890",

"vehicleInfo": {

{

"model": "Toyota Camry",

"make": "Toyota",

"capacity": 5,

"licensePlate": "ABC1234"

},

{

"model": "Honda Civic",

"make": "Honda",

"capacity": 4,

"licensePlate": "XYZ5678"

}

},

"createdAt": ISODate("2024-01-01T10:00:00Z")

});

// Insert Rider data

db.Rider.insertOne({

"id": "rider1",

"userName": "jane\_doe\_rider",

"password": "hashedpassword456",

"email": "jane.doe@example.com",

"phoneNumber": "+0987654321",

"createdAt": ISODate("2024-01-02T12:00:00Z")

});

// Insert Booking data

db.Booking.insertOne({

"id": "booking1",

"adminCommission": 10.0,

"driverEarnings": 40.0,

"driverId": "driver1",

"rideId": "ride1",

"riderId": "rider1",

"paymentId": "payment1"

});

// Insert Payment data

db.Payment.insertOne({

"id": "payment1",

"paymentDate": ISODate("2024-01-03T14:00:00Z"),

"paymentMethod": "CREDIT\_CARD",

"paymentStatus": "COMPLETED",

"riderId": "rider1"

});

// Insert Refund data

db.Refund.insertOne({

"id": "refund1",

"amountRefunded": 10.0,

"paymentId": "payment1",

"bookingId": "booking1",

"refundDate": ISODate("2024-01-04T09:00:00Z"),

"riderId": "rider1",

"refundStatus": "REFUNDED"

});

// Insert Ride data

db.Rides.insertOne({

"id": "ride1",

"availableSeats": 3,

"createdAt": ISODate("2024-01-01T08:00:00Z"),

"driverId": "driver1",

"dropLocation": "456 Elm Street, Springfield",

"pickupLocation": "123 Maple Avenue, Springfield",

"pricePerSeat": 15.0,

"startTime": ISODate("2024-01-05T10:00:00Z"),

"status": "SCHEDULED",

"vehicleInfo": {

"model": "Toyota Camry",

"make": "Toyota",

"capacity": 5,

"licensePlate": "ABC1234"

}

});

// Read all data from Driver collection

db.Driver.find()

// Read all data from Rider collection

db.Rider.find()

// Read specific booking details by riderId

db.Booking.find({"riderId": "rider1"})

// Read specific payment details by riderId

db.Payment.find({"riderId": "rider1"})

// Read ride details for a specific ride

db.Rides.find({"id": "ride1"})

//Update vehicle info add new vehicle

db.Driver.updateOne( { "id": "driver1" }, { $push: { "vehicleInfo": { "model": "Honda Accord", "make": "Honda", "capacity": 5, "licensePlate": "NEW1234" } } } )

//Update available seats each time ride is booked

db.Rides.updateOne( { "id": "ride1" }, { $inc: { "availableSeats": -1 } }