



# React Admin Dashboard - API Integration Guide

## (Production)

**Target Audience:** Web Frontend Team

**Base URL (Production):** `https://driversklub-backend.onrender.com/`

**Base URL (Development):** `http://localhost:5000`

**Auth:** Requires `Authorization: Bearer <TOKEN>` with Role `SUPER_ADMIN` or `OPERATIONS`

**Version:** 3.1.0

**Last Updated:** January 9, 2026



## Table of Contents

- [1. Authentication](#)
- [2. Dispatch & Trip Operations](#)
- [3. Fleet & Asset Management](#)
- [4. Operations & Assignments](#)
- [5. User Management](#)
- [6. Pricing Calculator](#)
- [7. Payment & Financial Management](#)
- [8. Frontend Implementation Notes](#)
- [9. Rapido Operational Monitoring](#)

## 1. Authentication

### 1.1 Admin Login Flow

Same as driver authentication but requires `SUPER_ADMIN` or `OPERATIONS` role.

**Endpoint:** `POST /auth/verify-otp`

**Response:**

```
{
  "success": true,
  "data": {
    "accessToken": "eyJ...",
    "refreshToken": "def...",
    "user": {
      "id": "uuid",
      "role": "SUPER_ADMIN"
    }
  }
}
```

**Action:** Check `user.role`. Redirect to dashboard only if `SUPER_ADMIN` or `OPERATIONS`.

## 2. Dispatch & Trip Operations

---

### 2.1 Create New Trip

**Endpoint:** POST /trips

**Roles:** SUPER\_ADMIN , OPERATIONS

**Request Body:**

```
{
  "tripType": "AIRPORT",
  "originCity": "Delhi",
  "destinationCity": "Gurgaon",
  "pickupLocation": "T3 Terminal, Gate 4",
  "pickupLat": 28.5562,
  "pickupLng": 77.1000,
  "dropLocation": "Cyber Hub, Gurgaon",
  "pickupTime": "2025-12-25T10:00:00Z",
  "vehicleSku": "EV_SEDAN",
  "distanceKm": 45
}
```

[!IMPORTANT] **Strict Trip Constraints:**

- **Start Window:** Driver can only start trip **2.5 Hours** before pickup
- **Geofence:** `pickupLat` & `pickupLng` are **MANDATORY** for the app to allow "Arrived" status (500m radius)
- **T-1 Constraint:** `pickupTime` must be > 24 hours from now

**Response (201):**

```
{
  "success": true,
  "data": {
    "id": "uuid",
    "status": "CREATED",
    "price": 1200,
    "tripType": "AIRPORT",
    "pickupTime": "2025-12-25T10:00:00Z"
  }
}
```

### 2.2 List All Trips (Grid View)

**Endpoint:** GET /admin/trips

**Role:** SUPER\_ADMIN

**Query Params:**

- `page` (default: 1)

- `limit` (default: 10)
- `status` (optional): Filter by status (e.g., `CREATED` , `DRIVER_ASSIGNED` , `STARTED` )

**Response (200):**

```
{
  "success": true,
  "data": {
    "trips": [
      {
        "id": "uuid",
        "tripType": "AIRPORT",
        "pickupLocation": "T3 Terminal",
        "dropLocation": "Cyber Hub",
        "pickupTime": "2025-12-25T10:00:00Z",
        "status": "CREATED",
        "price": 1200,
        "driver": null,
        "customerPhone": "9876543210",
        "providerMapping": {
          "providerType": "MMT",
          "externalBookingId": "MMT-123"
        }
      }
    ],
    "total": 150,
    "page": 1,
    "limit": 10
  }
}
```

## 2.3 Assign Driver (Dispatch)

**Endpoint:** `POST /admin/trips/assign`

**Role:** `SUPER_ADMIN`

**Description:** The core action of the dashboard. Logic: "Select Trip → Select Driver → Assign"

**Request Body:**

```
{
  "tripId": "uuid-trip-id",
  "driverId": "uuid-driver-id"
}
```

**Response (200):**

```
{
  "success": true,
```

```
    "message": "Driver assigned successfully"
  }
```

**Side Effects:**

1. Updates Trip Status → `DRIVER_ASSIGNED`
  2. Creates `TripAssignment` record (transactional)
  3. Pushes Notification to Driver App
  4. If MMT Trip, pushes Webhook to MMT ( `/driver-assigned` )
- 

## 2.4 Unassign Driver

**Endpoint:** `POST /admin/trips/unassign`

**Role:** `SUPER_ADMIN`

**Description:** Force cancel/detach driver from trip

**Request Body:**

```
{
  "tripId": "uuid"
}
```

**Response (200):**

```
{
  "success": true,
  "message": "Driver unassigned successfully"
}
```

**Side Effects:**

- Status: `DRIVER_ASSIGNED` → `CREATED`
  - If MMT Trip, triggers `/detach-trip` webhook
- 

## 2.5 Reassign Driver

**Endpoint:** `POST /admin/trips/reassign`

**Role:** `SUPER_ADMIN`

**Description:** Change assigned driver (e.g., when driver cancels or car breaks down)

**Request Body:**

```
{
  "tripId": "uuid",
  "driverId": "uuid-new-driver"
}
```

#### Response (200):

```
{
  "success": true,
  "message": "Driver reassigned successfully"
}
```

#### Side Effects:

- If MMT Trip, triggers `/reassign-chauffeur` webhook
- 

## 3. Fleet & Asset Management

---

### 3.1 Fleets (Operators)

**[!TIP] Rapido Integration:** Rapido Captains are managed as a Fleet. Their status is synced automatically.

#### Create Fleet

**Endpoint:** `POST /fleets`

**Role:** `SUPER_ADMIN`

#### Request Body:

```
{
  "name": "Delhi Cabs Pvt Ltd",
  "mobile": "9999988888",
  "city": "DELHI",
  "fleetType": "COMPANY",
  "panNumber": "ABCDE1234F"
}
```

#### Response (201):

```
{
  "success": true,
  "data": {
    "id": "uuid",
    "name": "Delhi Cabs Pvt Ltd",
    "city": "DELHI"
  }
}
```

#### List Fleets

**Endpoint:** `GET /fleets`

**Roles:** `SUPER_ADMIN` , `OPERATIONS`

**Response (200):**

```
{
  "success": true,
  "data": [
    {
      "id": "uuid",
      "name": "Delhi Cabs Pvt Ltd",
      "city": "DELHI",
      "fleetType": "COMPANY",
      "status": "ACTIVE"
    }
  ]
}
```

**Get Fleet Details**

---

## 3. Fleet Management

---

### 3.1 Fleets & Hubs

**Create Fleet**

**Endpoint:** POST /fleets **Role:** SUPER\_ADMIN

**Manage Hubs**

**Create Hub:** POST /fleets/:id/hubs **List Hubs:** GET /fleets/:id/hubs **Get Hub Details:** GET /fleets/hubs/:id

**Manage Hub Managers**

**Create Manager:** POST /fleets/:id/hub-managers **List Managers:** GET /fleets/:id/hub-managers **Assign**

**Manager:** POST /fleets/hubs/:hubId/assign-manager

**Fleet Resources (Hub Context)**

**Add Vehicle to Hub:** POST /fleets/hubs/:id/add-vehicle **Remove Vehicle from Hub:** POST

/fleets/hubs/:id/remove-vehicle **Add Driver to Hub:** POST /fleets/hubs/:id/add-driver **Remove Driver from**

**Hub:** POST /fleets/hubs/:id/remove-driver

---

### 3.2 Vehicles (Cars)

**Add Vehicle**

**Endpoint:** POST /vehicles

**Roles:** SUPER\_ADMIN, OPERATIONS

**Request Body:**

```
{
  "fleetId": "uuid-fleet-id",
```

```
"vehicleNumber": "DL10CA1234",
"vehicleName": "Tata Tigor EV",
"fuelType": "ELECTRIC",
"ownership": "OWNED"
}
```

#### Response (201):

```
{
  "success": true,
  "data": {
    "id": "uuid",
    "vehicleNumber": "DL10CA1234",
    "vehicleName": "Tata Tigor EV"
  }
}
```

#### List Vehicles by Fleet

**Endpoint:** GET /vehicles/fleet/:fleetId

**Roles:** SUPER\_ADMIN , OPERATIONS , MANAGER

#### Response (200):

```
{
  "success": true,
  "data": [
    {
      "id": "uuid",
      "vehicleNumber": "DL10CA1234",
      "vehicleName": "Tata Tigor EV",
      "fuelType": "ELECTRIC",
      "status": "ACTIVE"
    }
  ]
}
```

#### Update Vehicle Documents

**Endpoint:** PATCH /vehicles/:id/docs

**Roles:** SUPER\_ADMIN , OPERATIONS

#### Request Body:

```
{
  "rcUrl": "https://s3.aws.com/rc.pdf",
  "insuranceUrl": "https://s3.aws.com/insurance.pdf"
}
```

### Update Vehicle Status

**Endpoint:** PATCH /vehicles/:id/status

**Roles:** SUPER\_ADMIN , OPERATIONS

#### Request Body:

```
{
  "status": "ACTIVE"
}
```

#### Status Values:

- ACTIVE - Vehicle is operational
- INACTIVE - Vehicle temporarily unavailable
- MAINTENANCE - Vehicle under maintenance

### Deactivate Vehicle

**Endpoint:** PATCH /vehicles/:id/deactivate

**Roles:** SUPER\_ADMIN , OPERATIONS

---

## 3.3 Drivers (Profiles)

### Onboard Driver

**Endpoint:** POST /drivers

**Roles:** SUPER\_ADMIN , OPERATIONS

#### Request Body:

```
{
  "fleetId": "uuid-fleet-id",
  "firstName": "Raj",
  "lastName": "Kumar",
  "mobile": "9812345678",
  "licenseNumber": "DL-12345-67890",
  "email": "raj@example.com"
}
```

#### Response (201):

```
{
  "success": true,
  "data": {
    "id": "uuid",
    "firstName": "Raj",
    "lastName": "Kumar",
    "mobile": "9812345678"
  }
}
```



```
}  
}
```

### List Drivers by Fleet

**Endpoint:** GET /drivers/fleet/:fleetId

**Roles:** SUPER\_ADMIN , OPERATIONS , MANAGER

**Response (200):**

```
{  
  "success": true,  
  "data": [  
    {  
      "id": "uuid",  
      "firstName": "Raj",  
      "lastName": "Kumar",  
      "mobile": "9812345678",  
      "status": "ACTIVE",  
      "kycStatus": "APPROVED"  
    }  
  ]  
}
```

### Get Driver Details

**Endpoint:** GET /drivers/:id

**Roles:** SUPER\_ADMIN , OPERATIONS , MANAGER

---

## 4. Operations & Assignments

### 3.4 Update Driver Status & Availability

#### Update Status

**Endpoint:** PATCH /drivers/:id/status **Roles:** SUPER\_ADMIN , OPERATIONS

**Request Body:**

```
{  
  "status": "SUSPENDED"  
}
```

#### Update Availability

**Endpoint:** PATCH /drivers/:id/availability **Roles:** SUPER\_ADMIN , OPERATIONS , MANAGER

**Request Body:**

```
{  
  "isAvailable": false  
}
```

---

## 4. Operations & Assignments

---

### 4.1 Attendance Management

#### Approve Attendance

**Endpoint:** POST /attendance/:id/approve **Roles:** SUPER\_ADMIN , MANAGER

**Request Body:**

```
{  
  "remarks": "Approved by Ops"  
}
```

#### Reject Attendance

**Endpoint:** POST /attendance/:id/reject **Roles:** SUPER\_ADMIN , MANAGER

#### Get Attendance Details

**Endpoint:** GET /attendance/:id **Roles:** SUPER\_ADMIN , OPERATIONS , MANAGER

---

### 4.2 Daily Vehicle Assignment (Roster)

**Endpoint:** POST /assignments

**Roles:** SUPER\_ADMIN , OPERATIONS , MANAGER

**Description:** Link a driver to a car for the day

**Request Body:**

```
{  
  "driverId": "uuid-driver",  
  "vehicleId": "uuid-vehicle",  
  "fleetId": "uuid-fleet"  
}
```

**Response (201):**

```
{  
  "success": true,  
}
```

```
"data": {
  "id": "uuid",
  "driverId": "uuid",
  "vehicleId": "uuid",
  "startDate": "2025-12-25T00:00:00Z"
}
```

**Goal:** Driver cannot receive trips without this active link.

---

### 4.3 Get Assignments by Fleet

**Endpoint:** GET /assignments/fleet/:fleetId

**Roles:** SUPER\_ADMIN , OPERATIONS , MANAGER

**Response (200):**

```
{
  "success": true,
  "data": [
    {
      "id": "uuid",
      "driver": {
        "firstName": "Raj",
        "lastName": "Kumar"
      },
      "vehicle": {
        "vehicleNumber": "DL10CA1234"
      },
      "startDate": "2025-12-25T00:00:00Z",
      "status": "ACTIVE"
    }
  ]
}
```

---

### 4.4 End Assignment

**Endpoint:** PATCH /assignments/:id/end

**Roles:** SUPER\_ADMIN , OPERATIONS , MANAGER

---

### 4.5 Driver Preference Management

#### Get Pending Requests

**Endpoint:** GET /drivers/preference/pending-requests

**Roles:** SUPER\_ADMIN , OPERATIONS

**Response (200):**

```
{
  "success": true,
  "statusCode": 200,
  "message": "Pending preference requests retrieved successfully",
  "data": [
    {
      "id": "bd3c2df9-d58d-4b5a-8d20-2ecd8db1b63e",
      "driverId": "ad8324ca-2dea-4618-ba5e-3095fa123d06",
      "currentPreference": {
        "accept_rentals": false,
        "prefer_airport_rides": false
      },
      "requestedPreference": {
        "accept_rentals": true,
        "prefer_airport_rides": true
      },
      "status": "PENDING",
      "requestAt": "2026-01-08T04:38:38.415Z"
    }
  ]
}
```

### Update Request Status

**Endpoint:** POST /drivers/preference/update-status

**Roles:** SUPER\_ADMIN, OPERATIONS

### Request Body (Approve):

```
{
  "id": "bd3c2df9-d58d-4b5a-8d20-2ecd8db1b63e",
  "status": "APPROVED"
}
```

### Request Body (Reject):

```
{
  "id": "bd3c2df9-d58d-4b5a-8d20-2ecd8db1b63e",
  "status": "REJECTED",
  "rejection_reason": "demo test"
}
```

## 5. User Management

### 5.1 Create User

**Endpoint:** POST /users

**Role:** SUPER\_ADMIN

**Request Body:**

```
{
  "phone": "9876543210",
  "role": "DRIVER",
  "name": "Raj Kumar"
}
```

**Roles:** SUPER\_ADMIN , OPERATIONS , MANAGER , DRIVER

---

## 5.2 List All Users

**Endpoint:** GET /users

**Roles:** SUPER\_ADMIN , OPERATIONS

---

## 5.3 Deactivate User

**Endpoint:** PATCH /users/:id/deactivate

**Role:** SUPER\_ADMIN

---

# 6. Pricing Calculator

---

## 6.1 Preview Pricing

**Endpoint:** POST /pricing/preview

**Auth Required:** No

**Description:** "Get Estimate" button on Create Trip form

**Request Body:**

```
{
  "distanceKm": 45,
  "tripType": "AIRPORT"
}
```

**Response (200):**

```
{
  "success": true,
  "data": {
    "baseFare": 800,
    "distanceCharge": 400,
  }
}
```

```
    "totalFare": 1200,
    "breakdown": {
      "minBillableKm": 40,
      "ratePerKm": 20
    }
  }
}
```

**Use Case:** Show fare before booking to set customer expectations.

---

## 7. Payment & Financial Management

---

### 7.1 Create Rental Plan

**Endpoint:** POST /payment/admin/rental-plans

**Roles:** SUPER\_ADMIN , OPERATIONS

**Request Body:**

```
{
  "fleetId": "uuid",
  "name": "Weekly Plan",
  "rentalAmount": 3500,
  "depositAmount": 5000,
  "validityDays": 7
}
```

**Response (201):**

```
{
  "id": "uuid",
  "fleetId": "uuid",
  "name": "Weekly Plan",
  "rentalAmount": 3500,
  "depositAmount": 5000,
  "validityDays": 7,
  "isActive": true
}
```

**Use Case:**

- **Onboarding:** Create standard plans (e.g., "Weekly Gold") for new drivers to choose from during registration.
- 

### 7.2 Get Rental Plans

**Endpoint:** GET /payment/admin/rental-plans/:fleetId

**Roles:** SUPER\_ADMIN , OPERATIONS , MANAGER

**Query Parameters:**

- activeOnly (boolean, default: true)
- 

### 7.3 Create Penalty

**Endpoint:** POST /payment/admin/penalty

**Roles:** SUPER\_ADMIN , OPERATIONS

**Request Body:**

```
{
  "driverId": "uuid",
  "type": "MONETARY",
  "amount": 500,
  "reason": "Customer complaint",
  "category": "BEHAVIOR"
}
```

**Penalty Types:**

- MONETARY - Financial penalty (auto-deducted from deposit for rental model)
- WARNING - Verbal/written warning
- SUSPENSION - Temporary suspension (requires suspensionStartDate and suspensionEndDate )
- BLACKLIST - Permanent ban

**Response (201):**

```
{
  "id": "uuid",
  "driverId": "uuid",
  "type": "MONETARY",
  "amount": 500,
  "reason": "Customer complaint",
  "isPaid": true,
  "deductedFromDeposit": true
}
```

**Use Case:**

- **Quality Control:** Penalize drivers for "No Shows" or poor behavior reported by customers.
  - **Deterrence:** Deduct from deposit immediately to enforce compliance.
- 

### 7.4 Waive Penalty

**Endpoint:** POST /payment/admin/penalty/:id/waive

**Roles:** SUPER\_ADMIN , OPERATIONS

**Request Body:**

```
{
  "waiverReason": "First-time offense, driver apologized"
}
```

**Response (200):**

```
{
  "success": true,
  "message": "Penalty waived successfully"
}
```

**Side Effects:**

- Refunds deposit if already deducted
- Reverses suspension/blacklist status

**Use Case:**

- **Dispute Resolution:** If a driver provides valid proof (e.g., car breakdown), Ops can waive the penalty.
- 

## 7.5 Create Incentive

**Endpoint:** POST /payment/admin/incentive

**Roles:** SUPER\_ADMIN, OPERATIONS

**Request Body:**

```
{
  "driverId": "uuid",
  "amount": 500,
  "reason": "Completed 50 trips this month",
  "category": "MILESTONE"
}
```

**Response (201):**

```
{
  "id": "uuid",
  "driverId": "uuid",
  "amount": 500,
  "reason": "Completed 50 trips this month",
  "isPaid": false
}
```

---



## 7.6 Process Incentive Payout

**Endpoint:** POST /payment/admin/incentive/:id/payout

**Roles:** SUPER\_ADMIN , OPERATIONS

**Response (200):**

```
{
  "success": true,
  "txnId": "TXN_1735123456_PAY123",
  "status": "PENDING",
  "utr": "UTR123456789"
}
```

**Note:** Sends money to driver's bank account via Easebuzz

**Use Case:**

- **Reward Distribution:** Operation team processes the approved incentive to credit the driver's bank account.
- 

## 7.7 Reconcile Daily Collection

**Endpoint:** POST /payment/admin/collection/:id/reconcile

**Roles:** SUPER\_ADMIN , OPERATIONS , MANAGER

**Request Body:**

```
{
  "expectedRevenue": 5000,
  "reconciliationNotes": "All collections verified"
}
```

**Response (200):**

```
{
  "success": true,
  "message": "Collection reconciled successfully"
}
```

**Side Effects:**

- Calculates revenue share
- Applies incentives and penalties
- Prepares for payout

**Use Case:**

- **End-of-Day Ops:** Manager verifies the physical cash collected matches the system's `expectedRevenue` before closing the shift.
-

## 7.8 Process Daily Payout

**POST** `/payment/admin/collection/:id/payout`

**DEPRECATED:** Use Bulk Payout instead. **Roles:** `SUPER_ADMIN`, `OPERATIONS`

---

## 7.9 Bulk Payout (Manual)

**Endpoint:** `POST /payment/admin/bulk-payout` **Roles:** `SUPER_ADMIN`, `OPERATIONS`

**Request:** `multipart/form-data`

- `file` : CSV File ( `phone, amount` or `accountNumber, amount` )

**Response:**

```
{
  "total": 10,
  "success": 9,
  "failed": 1,
  "amountDisbursed": 45000
}
```

**Use Case:**

- **Weekly Settlements:** Accountant uploads a CSV of all driver payouts on Monday morning to process them in one batch.
- 

## 7.10 Get Pending Reconciliations

**Endpoint:** `GET /payment/admin/reconciliations/pending`

**Roles:** `SUPER_ADMIN`, `OPERATIONS`, `MANAGER`

**Response (200):**

```
{
  "reconciliations": [
    {
      "id": "uuid",
      "driver": {
        "firstName": "Raj",
        "lastName": "Kumar"
      },
      "date": "2025-12-29T00:00:00.000Z",
      "totalCollection": 5000,
      "isReconciled": false
    }
  ]
}
```

**Use Case:**

- **Manager Dashboard:** Show a list of drivers whose collections are yet to be verified for the previous day.

---

## 7.11 Get Pending Payouts

**Endpoint:** GET /payment/admin/payouts/pending

**Roles:** SUPER\_ADMIN , OPERATIONS

**Response (200):**

```
{
  "payouts": [
    {
      "id": "uuid",
      "driver": {
        "firstName": "Raj",
        "lastName": "Kumar",
        "bankAccountNumber": "1234567890"
      },
      "date": "2025-12-29T00:00:00.000Z",
      "netPayout": 3800,
      "isPaid": false
    }
  ]
}
```

**Use Case:**

- **Finance Review:** Finance team reviews all verified collections that are ready for payout before initiating the bank transfer.

---

## 7.12 Generate Vehicle QR Code

**Endpoint:** POST /payment/admin/vehicle/:id/qr

**Roles:** SUPER\_ADMIN , OPERATIONS

**Response (201):**

```
{
  "id": "uuid",
  "vehicleId": "uuid",
  "virtualAccountId": "VA123456",
  "virtualAccountNumber": "1234567890123456",
  "ifscCode": "HDFC0000001",
  "qrCodeBase64": "data:image/png;base64,...",
  "upiId": "driversklub.va123456@easebuzz",
  "isActive": true
}
```

**Use Case:**

- **New Car Setup:** Generate a unique QR code sticker for a new vehicle so passengers can pay via UPI directly to the vehicle's virtual account.
-

### 7.13 Get Vehicle QR Code

**Endpoint:** GET /payment/admin/vehicle/:id/qr

**Roles:** SUPER\_ADMIN , OPERATIONS , MANAGER

**Response (200):**

```
{
  "id": "uuid",
  "vehicleId": "uuid",
  "qrCodeBase64": "data:image/png;base64,...",
  "upiId": "driversklub.va123456@easebuzz",
  "isActive": true
}
```

**Use Case:**

- **Reprinting:** Manager retrieves the existing QR code if the physical sticker is damaged or lost.

---

### 7.14 InstaCollect Orders (Dynamic QR)

**Create Order (Generate Dynamic QR)**

**Endpoint:** POST /payment/orders **Roles:** SUPER\_ADMIN , OPERATIONS , MANAGER

**Request Body:**

```
{
  "customerName": "John Doe",
  "customerPhone": "9876543210",
  "amount": 2500,
  "description": "Advance Payment for Trip #123"
}
```

**Response (201):**

```
{
  "success": true,
  "data": {
    "id": "uuid-order-id",
    "totalAmount": 2500,
    "collectedAmount": 0,
    "remainingAmount": 2500,
    "status": "PENDING",
    "virtualAccountId": "VA_ORDER_123",
    "qrCodeBase64": "...",
    "upiId": "driversklub.order123@easebuzz"
  }
}
```

### Use Case:

- **Ad-Hoc Payments:** Driver or Admin enters an amount on the app/dashboard to generate a **one-time QR code** for a passenger to scan and pay instantly.

### Get Order Details

**Endpoint:** `GET /payment/orders/:id` **Roles:** `SUPER_ADMIN` , `OPERATIONS` , `MANAGER`

### Response (200):

```
{
  "success": true,
  "data": {
    "id": "uuid-order-id",
    "customerName": "John Doe",
    "customerPhone": "9876543210",
    "description": "Advance Payment",
    "totalAmount": 2500,
    "collectedAmount": 1000,
    "remainingAmount": 1500,
    "status": "PARTIAL",
    "virtualAccountId": "VA_ORDER_123",
    "qrCodeBase64": "data:image/png;base64...",
    "transactions": [
      {
        "id": "txn-uuid",
        "amount": 1000,
        "status": "SUCCESS",
        "date": "2025-12-25T10:00:00Z"
      }
    ],
    "createdAt": "2025-12-25T09:00:00Z"
  }
}
```

### List Orders

**Endpoint:** `GET /payment/orders` **Roles:** `SUPER_ADMIN` , `OPERATIONS` , `MANAGER`

### Query Params:

- `page` (default: 1)
- `limit` (default: 10)
- `status` (optional): `PENDING` , `PARTIAL` , `COMPLETED`
- `search` (optional): Filter by Customer Name or Phone

### Response (200):

```
{
  "success": true,
  "data": [
    {
      "id": "uuid-order-1",
      "customerName": "Alice Smith",
```

```

    "totalAmount": 5000,
    "collectedAmount": 5000,
    "status": "COMPLETED",
    "createdAt": "2025-12-24T10:00:00Z"
  },
  {
    "id": "uuid-order-2",
    "customerName": "Bob Jones",
    "totalAmount": 2000,
    "collectedAmount": 0,
    "status": "PENDING",
    "createdAt": "2025-12-25T11:00:00Z"
  }
],
"pagination": {
  "page": 1,
  "limit": 10,
  "total": 50,
  "totalPages": 5
}
}

```

## 8. Frontend Implementation Notes

### 8.1 CORS

- **Current:** Configured to allow all origins ( \* )
- **Production:** Whitelist specific domains

### 8.2 Date Handling

- Use `date-fns` or `moment` to parse UTC ISO strings from API
- **Always display in User's Local Time**
- Store in UTC, display in local

```

import { format, parseISO } from 'date-fns';

const displayTime = format(parseISO(trip.pickupTime), 'PPpp');
// Output: "Dec 25, 2025, 10:00 AM"

```

### 8.3 State Management

#### Recommendations:

- Cache `Fleets` and `Drivers` lists (TanStack Query recommended) as they change infrequently
- Poll `Trips` list (every 30s) or use a "Refresh" button for operations
- Handle `401 Unauthorized` by redirecting to Login
- Implement optimistic updates for better UX

#### Example with TanStack Query:

```
const { data: trips } = useQuery({
  queryKey: ['trips', { status, page }],
  queryFn: () => fetchTrips({ status, page }),
  refetchInterval: 30000, // 30 seconds
});
```

### 8.4 Error Handling

```
try {
  await assignDriver(tripId, driverId);
  toast.success('Driver assigned successfully');
} catch (error) {
  if (error.response?.status === 401) {
    // Redirect to login
    router.push('/login');
  } else {
    toast.error(error.response?.data?.message || 'Failed to assign driver');
  }
}
```

### 8.5 Role-Based UI

```
const canCreateTrip = ['SUPER_ADMIN', 'OPERATIONS'].includes(user.role);
const canApproveAttendance = ['SUPER_ADMIN', 'MANAGER'].includes(user.role);

{canCreateTrip && <Button onClick={openCreateTripModal}>Create Trip</Button>}
```

### 8.6 Pagination Component

```
<Pagination
  currentPage={page}
  totalPages={Math.ceil(total / limit)}
  onPageChange={setPage}
/>
```

### 8.7 Status Badge Component

```
const getStatusColor = (status) => {
  switch (status) {
    case 'CREATED': return 'gray';
    case 'DRIVER_ASSIGNED': return 'blue';
    case 'STARTED': return 'yellow';
    case 'COMPLETED': return 'green';
    case 'CANCELLED': return 'red';
    default: return 'gray';
  }
};

<Badge color={getStatusColor(trip.status)}>{trip.status}</Badge>
```

## 8.8 Real-time Updates (Optional)

Consider implementing WebSocket connection for real-time trip status updates:

```
const socket = io('wss://driversklub-backend.onrender.com');

socket.on('trip:updated', (trip) => {
  queryClient.setQueryData(['trip', trip.id], trip);
});
```

## Checklist for Production

- ☐ Implement token refresh logic
- ☐ Add role-based access control to UI
- ☐ Implement pagination for all list views
- ☐ Add loading states for all API calls
- ☐ Add error handling with user-friendly messages
- ☐ Implement date/time formatting (UTC → Local)
- ☐ Add confirmation dialogs for destructive actions
- ☐ Implement search/filter functionality
- ☐ Add export to CSV functionality
- ☐ Test all edge cases (empty states, errors, etc.)

## 9. Rapido Operational Monitoring

### 9.1 Conflict Resolution Logic

The dashboard does not have a manual "Sync Rapido Status" button because the process is fully automated.

- Logic:** The backend runs a worker every 5 minutes.



- **Conflicts:** If a driver is found ONLINE on Rapido but BUSY internally, the system auto-corrects this.
- **Logs:** All auto-corrections are logged in the backend logs (viewable via server logs).

## 9.2 Manual Override Alert

If a driver manually forces themselves ONLINE in the Rapido app:

1. System detects `status: online` webhook.
2. System checks assignments.
3. If conflict exists, system forces OFFLINE immediately.