

API Contracts v1.0 - Fleet Management Service

This document defines the data contracts and endpoints for the **Fleet Management API** of the TraEnSys project. This service is responsible for managing fleets, vehicles, drivers, tracking trips, and handling geofencing.

1. Physical Data Model (Tables)

This section describes the database tables that support the service, based on the class diagram and incorporating the required corrections.

1.1. Core Tables

Table: `fleets` Stores information about each vehicle fleet.

Column	Data Type	Constraints	Description
<code>id</code>	UUID	PRIMARY KEY	Unique identifier for the fleet.
<code>name</code>	VARCHAR(255)	NOT NULL	The public name of the fleet.
<code>creationDate</code>	DATE	NOT NULL	Date when the fleet was created.
<code>managerUserId</code>	UUID	FOREIGN KEY	References the <code>id</code> of the user (Fleet Manager) in the authentication service.

Table: `vehicles` Stores all data related to a single vehicle.

Column	Data Type	Constraints	Description
<code>id</code>	UUID	PRIMARY KEY	Unique identifier for the vehicle.
<code>fleetId</code>	UUID	FOREIGN KEY, NOT NULL	The fleet to which the vehicle belongs (<code>fleets</code>).
<code>licensePlate</code>	VARCHAR(50)	NOT NULL, UNIQUE	The vehicle's unique license plate.
<code>brand</code>	VARCHAR(100)		Brand of the vehicle (e.g., Toyota).
<code>model</code>	VARCHAR(100)		Model of the vehicle (e.g., Yaris).
<code>manufacturingYear</code>	INT		The year the vehicle was manufactured.
<code>type</code>	VEHICLE_TYPE (Enum)		The type of vehicle (CAR, TRUCK, VAN, BIKE).
<code>color</code>	VARCHAR(50)		Color of the vehicle.

Table: `drivers` Contains information specific to the driver role.

Column	Data Type	Constraints	Description
userId	UUID	PRIMARY KEY, FOREIGN KEY	References the corresponding user in the authentication service.
assignedVehicleId	UUID	FOREIGN KEY, UNIQUE	The vehicle currently assigned to the driver (vehicles). Can be NULL.
licenceNumber	VARCHAR(100)	NOT NULL, UNIQUE	The driver's license number.
status	BOOLEAN	NOT NULL	Indicates if the driver is active (true) or inactive (false).

Table: **trips** Logs every trip made by a vehicle.

Column	Data Type	Constraints	Description
id	UUID	PRIMARY KEY	Unique identifier for the trip.
vehicleId	UUID	FOREIGN KEY, NOT NULL	The vehicle that made the trip (vehicles).
driverId	UUID	FOREIGN KEY, NOT NULL	The driver who made the trip (drivers).
startDate	DATE	NOT NULL	The date the trip started.
endDate	DATE		The date the trip ended.
startTime	TIME	NOT NULL	The time the trip started.
endTime	TIME		The time the trip ended.
type	VEHICLE_TYPE (Enum)		Type of vehicle used (redundant but useful for history).
color	VARCHAR(50)		Color of the vehicle (redundant but useful for history).

1.2. Parameter Tables (1-to-1 Relationship with **vehicles**)

Table: **financial_parameters**

Column	Data Type	Constraints	Description
id	UUID	PRIMARY KEY	Unique identifier for the record.
vehicleId	UUID	FOREIGN KEY, UNIQUE, NOT NULL	Linked to a single vehicle (vehicles).
insuranceNumber	VARCHAR(100)		Insurance policy number.
insuranceExpiryDate	DATE		Expiry date of the insurance.

Column	Data Type	Constraints	Description
registrationDate	DATE		The vehicle's registration date.
purchaseDate	DATE		Date the vehicle was purchased.
depreciationRate	INT		Annual depreciation rate (as a %).
costPerKm	FLOAT		Estimated operational cost per kilometer.

Table: maintenance_parameters

Column	Data Type	Constraints	Description
id	UUID	PRIMARY KEY	Unique identifier for the record.
vehicleId	UUID	FOREIGN KEY, UNIQUE, NOT NULL	Linked to a single vehicle (vehicles).
lastMaintenanceDate	DATE		Date of the last maintenance.
nextMaintenanceDue	DATE		Date when the next maintenance is due.
engineStatus	ENGINE_STATUS (Enum)		Status of the engine (OK, NEEDS_SERVICE...).
batteryHealth	INT		The health of the battery (as a %).
maintenanceStatus	MAINTENANCE_STATUS (Enum)		Overall maintenance status (UP_TO_DATE, PENDING...).

Table: operational_parameters Stores real-time data for a vehicle.

Column	Data Type	Constraints	Description
id	UUID	PRIMARY KEY	Unique identifier for the record.
vehicleId	UUID	FOREIGN KEY, UNIQUE, NOT NULL	Linked to a single vehicle (vehicles).
currentTripId	UUID	FOREIGN KEY	The current trip for this vehicle (trips). Can be NULL.

Column	Data Type	Constraints	Description
status	BOOLEAN	NOT NULL	Operational status (e.g., <code>true</code> for in-service).
currentLocationPointId	UUID	FOREIGN KEY	The last known GPS coordinate (<code>geofence_points</code>).
currentSpeed	FLOAT		The vehicle's current speed in km/h.
fuelLevel	VARCHAR(50)		Current fuel level (e.g., "75%", "12/16").
mileage	FLOAT		Total mileage of the vehicle.
odometerReading	FLOAT		The reading from the odometer.
bearing	FLOAT		The vehicle's direction of travel in degrees (0-360).
timestamp	DATETIME		Timestamp of the last data update.

1.3. Geofencing and Route Tables

Table: `geofence_zones` Defines a geographical zone.

Column	Data Type	Constraints	Description
id	UUID	PRIMARY KEY	Unique identifier for the zone.
name	VARCHAR(255)	NOT NULL	Name of the zone (e.g., "Mokolo Market Area").
surfaceArea	INT		Area of the zone.
perimeter	INT		Perimeter of the zone.

Table: `geofence_points` Stores the coordinates that define the vertices of geofence zones or points on a route.

Column	Data Type	Constraints	Description
id	UUID	PRIMARY KEY	Unique identifier for the point.
latitude	FLOAT	NOT NULL	Latitude coordinate.
longitude	FLOAT	NOT NULL	Longitude coordinate.

Table: `geofence_zone_vertices` (Pivot Table) Associates points with a zone to form a polygon.

Column	Data Type	Constraints	Description
zoneId	UUID	FOREIGN KEY, PK	References the zone (<code>geofence_zones</code>).

Column	Data Type	Constraints	Description
pointId	UUID	FOREIGN KEY, PK	References the point (geofence_points).
vertexOrder	INT	NOT NULL	The order of the point in the polygon sequence.

Table: routes Defines a route with a start and end point.

Column	Data Type	Constraints	Description
id	UUID	PRIMARY KEY	Unique identifier for the route.
startPointId	UUID	FOREIGN KEY	The starting point (geofence_points).
endPointId	UUID	FOREIGN KEY	The ending point (geofence_points).

Table: trip_routes (Pivot Table) Junction table for the N-N relationship between trips and routes.

Column	Data Type	Constraints	Description
tripId	UUID	FOREIGN KEY, PK	References the trip (trips).
routeId	UUID	FOREIGN KEY, PK	References the route (routes).

Table: geofence_events Logs every time a vehicle enters or exits a zone.

Column	Data Type	Constraints	Description
id	UUID	PRIMARY KEY	Unique identifier for the event.
zoneId	UUID	FOREIGN KEY, NOT NULL	The zone involved (geofence_zones).
vehicleId	UUID	FOREIGN KEY, NOT NULL	The vehicle involved (vehicles).
timestamp	DATETIME	NOT NULL	The exact time of the event.
type	EVENT_TYPE (Enum)	NOT NULL	Type of event (ENTRY or EXIT).

2. Data Representation

2.1. Fleet Model

Represents a collection of vehicles managed by a Fleet Manager.

```
{
  "id": "f1a2b3c4-fleet-0001-d5e6f7g8h9i0",
  "name": "Yaoundé Express Voyage",
  "creationDate": "2025-01-15",
  "manager": {
    "userId": "7b2e3f4a-1c9d-4b8a-8e6f-2d3c4b5a6d7e",
```

```
    "name": "Gabriel Nomo"
  },
  "vehicleCount": 15
}
```

2.2. Vehicle Model (Detailed)

Represents a single vehicle with all its associated parameters.

```
{
  "id": "v1e2h3c4-c5a6-4b7c-8d9e-f0g1h2i3j4k5",
  "fleetId": "f1a2b3c4-fleet-0001-d5e6f7g8h9i0",
  "licensePlate": "CE 123 AB",
  "brand": "Toyota",
  "model": "Yaris",
  "manufacturingYear": 2022,
  "type": "CAR",
  "color": "White",
  "assignedDriver": {
    "userId": "d1r2i3v4-e5f6-4a7b-8c9d-e0f1g2h3i4j5",
    "name": "Aissatou Bello"
  },
  "financialParameters": {
    "insuranceNumber": "INS-YDE-2025-8843",
    "insuranceExpiryDate": "2026-06-30",
    "registrationDate": "2022-07-15",
    "purchaseDate": "2022-07-01",
    "depreciationRate": 15,
    "costPerKm": 125.5
  },
  "maintenanceParameters": {
    "lastMaintenanceDate": "2025-09-10",
    "nextMaintenanceDue": "2026-03-10",
    "engineStatus": "OK",
    "batteryHealth": 92,
    "maintenanceStatus": "UP_TO_DATE"
  },
  "operationalParameters": {
    "status": true,
    "currentSpeed": 45.5,
    "fuelLevel": "65%",
    "mileage": 45012.8,
    "odometerReading": 45012.8,
    "bearing": 182.5,
    "timestamp": "2025-10-30T14:22:10.000Z",
    "currentLocation": {
      "latitude": 3.8667,
      "longitude": 11.5167
    }
  }
}
```

2.3. Driver Model

Represents the fleet-specific information of a driver.

```
{
  "userId": "d1r2i3v4-e5f6-4a7b-8c9d-e0f1g2h3i4j5",
  "licenceNumber": "YDE/DR/2018/98765",
  "status": true,
  "assignedVehicle": {
    "vehicleId": "v1e2h3c4-c5a6-4b7c-8d9e-f0g1h2i3j4k5",
    "licensePlate": "CE 123 AB"
  },
  "userProfile": {
    "name": "Aissatou Bello",
    "phone": "+237699112233"
  }
}
```

2.4. Trip Model

```
{
  "id": "t1r2i3p4-a5b6-4c7d-8e9f-a0b1c2d3e4f5",
  "vehicleId": "v1e2h3c4-c5a6-4b7c-8d9e-f0g1h2i3j4k5",
  "driverId": "d1r2i3v4-e5f6-4a7b-8c9d-e0f1g2h3i4j5",
  "startDate": "2025-10-30",
  "startTime": "08:15:00",
  "endDate": "2025-10-30",
  "endTime": "09:05:00",
  "routes": [
    {
      "routeId": "r1o2u3t4e-001",
      "startPoint": { "latitude": 3.8721, "longitude": 11.5173 },
      "endPoint": { "latitude": 3.8472, "longitude": 11.5015 }
    }
  ]
}
```

2.5. Geofence Zone Model

```
{
  "id": "z1o2n3e4-a5b6-4c7d-8e9f-a0b1c2d3e4f5",
  "name": "Mokolo Market Area",
  "surfaceArea": 1.5,
  "perimeter": 5.2,
  "vertices": [
    { "latitude": 3.8750, "longitude": 11.5000, "order": 1 },
  ]
}
```

```
{ "latitude": 3.8790, "longitude": 11.5050, "order": 2 },
{ "latitude": 3.8760, "longitude": 11.5080, "order": 3 },
{ "latitude": 3.8710, "longitude": 11.5030, "order": 4 }
]
```

3. General API Rules

- **Base URL:** `/api/v1`
- **Data Format:** `application/json`
- **Authentication:** Header `Authorization: Bearer <JWT_TOKEN>`
- **Error Handling:**

```
{
  "timestamp": "2025-10-30T10:30:00.000Z",
  "status": 404,
  "error": "Not Found",
  "message": "Vehicle with ID 'v-invalid-id' not found.",
  "path": "/api/v1/vehicles/v-invalid-id"
}
```

4. Fleet Management API Endpoints

4.1. Fleets (`/fleets`)

POST `/fleets`

- **Description:** Creates a new fleet.
- **Access:** Authenticated (`fleet: fleet:create`).
- **Request (Body):** `{"name": "Douala City Transports", "managerUserId": "a9b8c7d6..."}`
- **Response (201 Created):** The newly created Fleet object.

GET `/fleets`

- **Description:** Retrieves a list of fleets. For a Fleet Manager, returns only their fleets. For an Admin, returns all fleets.
- **Access:** Authenticated (`fleet: fleet:read`).
- **Response (200 OK):** An array of Fleet objects.

GET `/fleets/{fleetId}`

- **Description:** Retrieves detailed information for a specific fleet.
- **Access:** Authenticated (`fleet: fleet:read`).
- **Response (200 OK):** A single detailed Fleet object.

PUT /fleets/{fleetId}

- **Description:** Updates the details of an existing fleet.
- **Access:** Authenticated (`fleet:fleet:update`).
- **Request (Body):** `{"name": "Douala Premium Transports", "managerUserId": "a9b8c7d6..."}`
- **Response (200 OK):** The updated Fleet object.

DELETE /fleets/{fleetId}

- **Description:** Deletes a fleet. Business logic should prevent deletion if the fleet still contains vehicles.
- **Access:** Authenticated (`fleet:fleet:delete`).
- **Response (204 No Content).**

4.2. Vehicles (/vehicles)

POST /fleets/{fleetId}/vehicles

- **Description:** Adds a new vehicle to a specified fleet.
- **Access:** Authenticated (`fleet:vehicle:create`).
- **Request (Body):** `{ "licensePlate": "CE 789 EF", "brand": "Toyota", "model": "RAV4", ... }`
- **Response (201 Created):** The full detailed Vehicle Model.

GET /vehicles

- **Description:** Retrieves a paginated list of all vehicles accessible to the user.
- **Access:** Authenticated (`fleet:vehicle:read`).
- **Query Parameters:** `?fleetId={id}&type=CAR&status=true`
- **Response (200 OK):** An array of Vehicle objects.

GET /vehicles/{vehicleId}

- **Description:** Retrieves the complete details of a single vehicle.
- **Access:** Authenticated (`fleet:vehicle:read`).
- **Response (200 OK):** The full Vehicle Model JSON.

PUT /vehicles/{vehicleId}

- **Description:** Updates the core details of a vehicle (brand, model, color, etc.).
- **Access:** Authenticated (`fleet:vehicle:update`).
- **Request (Body):** `{ "model": "Corolla", "color": "Blue", "manufacturingYear": 2021 }`
- **Response (200 OK):** The updated full Vehicle Model.

DELETE /vehicles/{vehicleId}

- **Description:** Deletes a vehicle from the system.
- **Access:** Authenticated (`fleet:vehicle:delete`).

- **Response (204 No Content).**

PUT /vehicles/{vehicleId}/financial-parameters

- **Description:** Updates the financial parameters for a specific vehicle.
- **Access:** Authenticated (fleet:vehicle:update).
- **Request (Body):** { "insuranceNumber": "INS-NEW-2025-1111", "costPerKm": 130.0 }
- **Response (200 OK):** The updated Financial Parameters object.

PUT /vehicles/{vehicleId}/maintenance-parameters

- **Description:** Updates the maintenance parameters for a specific vehicle.
- **Access:** Authenticated (fleet:vehicle:update).
- **Request (Body):** { "lastMaintenanceDate": "2025-10-28", "engineStatus": "NEEDS_SERVICE" }
- **Response (200 OK):** The updated Maintenance Parameters object.

4.3. Drivers (/drivers)

POST /drivers

- **Description:** Registers an existing system user as a driver.
- **Access:** Authenticated (fleet:driver:create).
- **Request (Body):** { "userId": "u1s2e3r4...", "licenceNumber": "LT/DR/2020/11223" }
- **Response (201 Created):** The full Driver Model.

GET /drivers

- **Description:** Retrieves a list of all drivers.
- **Access:** Authenticated (fleet:driver:read).
- **Query Parameters:** ?status=true
- **Response (200 OK):** An array of Driver objects.

GET /drivers/{driverUserId}

- **Description:** Retrieves the profile for a specific driver.
- **Access:** Authenticated (fleet:driver:read).
- **Response (200 OK):** A single Driver Model object.

PUT /drivers/{driverUserId}

- **Description:** Updates a driver's fleet-specific information.
- **Access:** Authenticated (fleet:driver:update).
- **Request (Body):** { "licenceNumber": "YDE/DR/2022/55443", "status": false }
- **Response (200 OK):** The updated Driver Model.

DELETE /drivers/{driverUserId}

- **Description:** De-registers a user as a driver (does not delete the user account).
- **Access:** Authenticated (`fleet:driver:delete`).
- **Response (204 No Content).**

POST /drivers/{driverUserId}/assign-vehicle

- **Description:** Assigns a vehicle to a driver.
- **Access:** Authenticated (`fleet:driver:assign`).
- **Request (Body):** { "vehicleId": "v1e2h3c4..." }
- **Response (200 OK):** { "message": "Vehicle assigned successfully." }

POST /drivers/{driverUserId}/unassign-vehicle

- **Description:** Unassigns the current vehicle from a driver.
- **Access:** Authenticated (`fleet:driver:assign`).
- **Response (200 OK):** { "message": "Vehicle unassigned successfully." }

4.4. Trips (/trips)

POST /trips

- **Description:** Starts a new trip for a vehicle.
- **Access:** Authenticated (`fleet:trip:create`).
- **Request (Body):** { "vehicleId": "v1e2h3c4...", "driverId": "d1r2i3v4...", "startDate": "2025-10-31", "startTime": "09:00:00" }
- **Response (201 Created):** The new Trip object.

GET /trips

- **Description:** Retrieves a list of trips, with filtering options.
- **Access:** Authenticated (`fleet:trip:read`).
- **Query Parameters:** ?vehicleId={id}&driverId={id}&startDate=YYYY-MM-DD&endDate=YYYY-MM-DD
- **Response (200 OK):** An array of Trip objects.

GET /trips/{tripId}

- **Description:** Retrieves the details of a single trip.
- **Access:** Authenticated (`fleet:trip:read`).
- **Response (200 OK):** A single Trip Model object.

POST /trips/{tripId}/end

- **Description:** Ends an active trip.
- **Access:** Authenticated (`fleet:trip:update`).
- **Request (Body):** { "endDate": "2025-10-31", "endTime": "11:30:00" }
- **Response (200 OK):** The updated Trip object.

4.5. Geofencing (/geofence)

POST /geofence/zones

- **Description:** Creates a new geofence zone.
- **Access:** Authenticated (`fleet:geofence:create`).
- **Request (Body):** { "name": "Bastos Residential Area", "vertices": [...] }
The request body is a JSON object with a "name" string and a "vertices" array of coordinate pairs.
- **Response (201 Created):** The full Geofence Zone Model.

GET /geofence/zones

- **Description:** Retrieves a list of all defined geofence zones.
- **Access:** Authenticated (`fleet:geofence:read`).
- **Response (200 OK):** An array of Geofence Zone objects.

GET /geofence/zones/{zoneId}

- **Description:** Retrieves the details of a single geofence zone.
- **Access:** Authenticated (`fleet:geofence:read`).
- **Response (200 OK):** A single Geofence Zone Model object.

PUT /geofence/zones/{zoneId}

- **Description:** Updates a geofence zone's name or vertices.
- **Access:** Authenticated (`fleet:geofence:update`).
- **Request (Body):** { "name": "Bastos VIP Area", "vertices": [...] }
The request body is a JSON object with a "name" string and a "vertices" array of coordinate pairs.
- **Response (200 OK):** The updated Geofence Zone Model.

DELETE /geofence/zones/{zoneId}

- **Description:** Deletes a geofence zone.
- **Access:** Authenticated (`fleet:geofence:delete`).
- **Response (204 No Content):**

GET /geofence/events

- **Description:** Retrieves a log of geofence events, with optional filters.
- **Access:** Authenticated (`fleet:geofence:read`).
- **Query Parameters:** `?vehicleId={id}&zoneId={id}&type=ENTRY&startDate=YYYY-MM-DD`
Optional query parameters for filtering events by vehicle, zone, type, and start date.
- **Response (200 OK):** An array of geofence event objects.