SAMFMS

Service Contracts

Team Firewall Five:

Team Member	Student number
Herman Engelbrecht	u22512374
Morné van Heerden	u21482153
Laird Glanville	u22541332
Stefan Jansen van Rensburg	u22550055
Nicolaas Johan Jansen van Rensburg	u22592732

Contents

1. Conventions

- Transport: REST over HTTP. JSON bodies (multipart/form-data for file uploads where stated).
- Auth: Bearer JWT in Authorization header. Role/permission enforcement per service.
- IDs: UUIDv4 strings (e.g., vehicle_id, driver_id, trip_id).
- Timestamps: ISO 8601 UTC (e.g., 2025-08-18T14:25:00Z).
- Numbers: Decimal for currency/quantities; integers for counts.
- Pagination (lists): page (default 1), page_size (default 25, max 200). Responses include items, total, page, page_size.
- Filtering (common): q (free text), status, start_time, end_time, sort (field[:asc|:desc]).
- Response envelope (success): { data: <object|array>, meta?: { page, page_size, total } }.
- Response envelope (error): { error: { code: string, message: string, details?: any } }.
- Error codes: 400 validation, 401 unauthenticated, 403 forbidden, 404 not found, 409 conflict, 422 semantic validation, 500 server error, 502/503/504 upstream/unavailable/timeout.
- Resilience: Circuit breakers, retries with backoff, request de-duplication (content TTL 60s; correlation TTL 300s).
- Messaging (AMQP): Exchange 'service_requests' → '<service>.requests'; responses on 'service_responses'. Messages JSON with correlation_id.

2. Core Gateway

Routes /management/*, /maintenance/*, /gps/*, /trips/* to respective services. Also exposes health and service registry.

Health

- GET /health/
  - Input: no input
  - Output: { data: { status: 'ok' } }.
- GET /health/ready
  - Input: none
  - Output: readiness booleans per dependency.
- GET /health/live
  - Input: none
  - Output: { data: { alive: true } }.
- GET /health/detailed
  - Input: none
  - Output: { data: { db, mq, discovery, circuits } }.
- GET /health/metrics
  - Input: optional ?limit
  - Output: basic counters/gauges.
- GET /services
  - Input: none
  - Output: { data: [{ name, base_path, version }] }.

Routing

- ANY /{service}/{path}

Auth Proxy

- All /auth/* paths are forwarded to Security.

3. Management Service

Base: /management/*

- Vehicle: { vehicle_id, plate, vin?, make, model, year?, status:'active'|'inactive'|'maintenance', mileage? }
- Assignment: { assignment_id, driver_id, vehicle_id, start_time, end_time?, start_mileage?, end_mileage? }
- Driver: { driver_id, name, email, phone?, license_no?, status }
- MileageRecord: { mileage_record_id, vehicle_id, odometer, reading_time }
- FuelRecord: { fuel_record_id, vehicle_id, liters, cost, currency, odometer?, time }
- Notification: { notification_id, type:'system'|'assignment'|'maintenance', message, recipients?, status:'unread'|'read'|'archived', created_at }

Vehicles (/vehicles)

- GET /vehicles
  - Input: paging, q, status, driver_id
  - Output: { items:[Vehicle], total,... }.
- POST /vehicles
  - Input: { plate, vin?, make, model, year?, status? }
  - Output: Vehicle.
- GET /vehicles/{vehicle_id}
  - Input: path id
  - Output: Vehicle.
- PUT /vehicles/{vehicle_id}
  - Input: partial Vehicle fields
  - Output: Vehicle.
- DELETE /vehicles/{vehicle_id}
  - Input: path id
  - Output: { data: { deleted: true } }.
- GET /vehicles/{vehicle_id}/assignments
  - Input: paging/date filters
  - Output: [Assignment].

- GET /vehicles/{vehicle_id}/usage
    - Input: start_time?, end_time?
    - Output: { mileage, trips, fuel }.

Drivers (/drivers)

- GET /drivers
    - Input: paging, q, status
    - Output: [Driver].
- POST /drivers
    - Input: { name, email, phone?, license_no?, status? }
    - Output: Driver.
- GET /drivers/{driver_id}
    - Input: none
    - Output: [Driver]
- PUT /drivers/{driver_id}
    - Input: partial Driver
    - Output: Driver.
- DELETE /drivers/{driver_id}
    - Input: None
    - Output: { data: { deactivated: true } }.
- POST /drivers/{driver_id}/assign-vehicle?vehicle_id=
    - Input: None
    - Output: Assignment.
- POST /drivers/{driver_id}/unassign-vehicle
    - Input: None
    - Output: { data: { unassigned: true } }.

Assignments (/assignments)

- POST /assignments
    - Input: { driver_id, vehicle_id, start_time?, start_mileage? }
    - Output: Assignment.
- DELETE /assignments/{assignment_id}?end_mileage=
    - Input: None
    - Output: { data: { closed: true } }.
- GET /assignments/active
    - Input: paging

    - Output: [Assignment].

Mileage (/mileage)

- POST /mileage/update
    - Input: { vehicle_id, odometer, reading_time? }
    - Output: { data: { updated: true }, vehicle_mileage }
- GET /mileage/records/{mileage_record_id}
    - Input: None
    - Output: MileageRecord

Fuel (/fuel)

- POST /fuel/records
    - Input: { vehicle_id, liters, cost, currency, odometer?, time? }
    - Output: FuelRecord.
- PUT /fuel/records/{fuel_record_id}
    - Input: partial FuelRecord
    - Output: FuelRecord.
- GET /fuel/records/{fuel_record_id}
    - Input: None
    - Output: FuelRecord.

Notifications (/notifications)

- POST /notifications
    - Input: { type, message, recipients?:[user_id]|'broadcast' }
    - Output: Notification.
- PUT /notifications/{notification_id}/read
    - Input: None
    - Output: { data: { read: true } }.
- GET /notifications/recipient/{recipient_id}
    - Input: status?
    - Output: [Notification].

Analytics (/analytics)

- GET /analytics/dashboard
    - Input: range?
    - Output: { kpis:{ utilization, active_drivers, mileage }, highlights:[...] }
- GET /analytics/driver-performance/{driver_id}
    - Input: range?
    - Output: { scores, incidents, mileage }.

4. Maintenance Service

Base: /maintenance/*

- MaintenanceRecord: { record_id, vehicle_id, category, service_date, cost?, currency?, next_due_date?, notes? }.
- License: { record_id, entity_type, entity_id, number, expiry_date, status:'active'|'expired'|'revoked' }.

Records (/records)

- POST /records
    - Input: { vehicle_id, category, description?, service_date, cost?, currency?, next_due_date? }
    - Output: MaintenanceRecord.
- GET /records
    - Input: paging, q, vehicle_id?, date range
    - Output: [MaintenanceRecord].
- PUT /records/{record_id}
    - Input: partial MaintenanceRecord
    - Output: MaintenanceRecord.

Licenses (/licenses)

- POST /licenses
    - Input: { entity_type:'vehicle'|'driver'|'company', entity_id, number, issued_date?, expiry_date, status? }
    - Output: License.

- GET /licenses/status/expiring?days=
    - Input: None
    - Output: [License].
- POST /licenses/{record_id}/renew
    - Input: { expiry_date, notes? }
    - Output: License.
    -

Notifications & Analytics

- GET /notifications/pending
    - Input: None
    - Output: [Notification].
- POST /notifications/process
    - Input: None
    - Output: { data: { processed: count } }.
- GET /analytics/summary/vehicle/{vehicle_id}
    - Input: range?
    - Output: { last_service, upcoming, costs }.

5. GPS Service

Base: /gps/*

- Location: { vehicle_id, lat, lon, time, speed?, heading? }.
- Place: { place_id, name, lat, lon, radius?, category?, tags?[] }.
- Geofence: { geofence_id, type, geometry, name, created_at }.

Locations (/locations)

- POST /locations/update
    - Input: { vehicle_id, lat, lon, time?, speed?, heading? }
    - Output: Location (current) and appended to history.
- GET /locations/{vehicle_id}
    - Input: path id
    - Output: Location.

- GET /locations?vehicle_ids=a,b,c
    - Input: query
    - Output: [Location].
- GET /locations/{vehicle_id}/history?start_time&end_time&limit
    - Input: None
    - Output: [Location].

Places (/places)

- POST /places
    - Input: { name, lat, lon, radius?, category?, tags?[] }
    - Output: Place.
- GET /places
    - Input: paging, q, category?
    - Output: [Place].
- PUT /places/{place_id}
    - Input: partial Place
    - Output: Place.

Geofences (/geofences)

- POST /geofences
    - Input: { type:'circle'|'polygon'|'rectangle', geometry, name, description? }
    - Output: Geofence.
- GET /geofences
    - Input: paging, q
    - Output: [Geofence].

6. Trip Planning Service

Base: /trips/*

- Trip: { trip_id, status:'planned'|'in_progress'|'paused'|'completed'|'cancelled', origin, destination, waypoints?[], driver_assignment?, vehicle_assignment?, scheduled_start_time?, scheduled_end_time?, actual_start_time?, actual_end_time?, estimated_duration?, estimated_distance?, priority? }

Trips (/trips)

- POST /trips
  - Input: CreateTripRequest
  - Output: { data: { trip } }

- GET /trips
  - Input: filters: status[], priority[], driver_id?, vehicle_id?, start_date?, end_date?, paging, sort_by, sort_order
  - Output: { items:[trip], total, skip, limit }

- GET /trips/{trip_id}
  - Input: path id
  - Output: { data: { trip } }

- PUT /trips/{trip_id}
  - Input: UpdateTripRequest
  - Output: { data: { trip } }

- DELETE /trips/{trip_id}
  - Input: path id
  - Output: { data: { deleted: true } }

- POST /trips/{trip_id}/start
  - Input: none
  - Output: { data: { trip }, message }

- POST /trips/{trip_id}/pause
  - Input: none
  - Output: { data: { trip }, message }

- POST /trips/{trip_id}/resume

    - Input: none
    - Output: { data: { trip }, message }

- POST /trips/{trip_id}/complete
    - Input: none
    - Output: { data: { trip }, message }

- POST /trips/{trip_id}/cancel
    - Input: reason?
    - Output: { data: { trip }, message }

- PUT /trips/{trip_id}/progress
    - Input: TripProgressRequest
    - Output: { data: { progress }, message }

- POST /trips/{trip_id}/optimize-route
    - Input: RouteOptimizationRequest
    - Output: { data: { optimized_route, time_saved, distance_saved }, message }

Drivers & Availability (/drivers)

- GET /drivers
    - Input: status?, department?, paging
    - Output: { data: { drivers } }

- POST /drivers/{trip_id}/assign
    - Input: AssignDriverRequest
    - Output: { data: { assignment }, message }

- DELETE /drivers/{trip_id}/unassign
    - Input: none
    - Output: { data: { unassigned: true }, message }

- GET /drivers/{driver_id}/availability?start_time&end_time
    - Input: path id, start_time, end_time
    - Output: { data: { driver_id, is_available, start_time, end_time }, message }

- GET /drivers/availability?start_time&end_time&driver_ids=
    - Input: start_time, end_time, driver_ids?
    - Output: { data: { availability: [...] }, message }
    -

- GET /drivers/available?start_time&end_time
    - Input: start_time, end_time
    - Output: { data: { available_drivers, total_available, total_checked, timeframe }, message }

Notifications (/notifications)

- GET /notifications/
    - Input: unread_only?, limit, skip
    - Output: { data: { notifications, total, unread_count }, message }

- POST /notifications/
    - Input: NotificationRequest (admin or fleet_manager only)
    - Output: { data: { sent_count, notification_ids }, message }

- PUT /notifications/{notification_id}/read
    - Input: path id
    - Output: { data: { marked_read }, message }

- GET /notifications/unread/count
    - Input: none
    - Output: { data: { unread_count }, message }

- GET /notifications/preferences
    - Input: none
    - Output: { data: preferences, message }

- PUT /notifications/preferences
    - Input: UpdateNotificationPreferencesRequest
    - Output: { data: preferences, message }

Phone usage monitoring (/driver/ping)

- POST /driver/ping
    - Input: DriverPingRequest
    - Output: StandardResponse with { status, message, ping_received_at, next_ping_expected_at, session_active, violations_count }

- GET /driver/ping/violations/{trip_id}
    - Input: path id
    - Output: { data: { trip_id, violations[], total_violations }, message }

- GET /driver/ping/session/{trip_id}
    - Input: path id
    - Output: { data: { status, statistics }, message }

Safety violations

- POST /speed-violations/
    - Input: CreateSpeedViolationRequest
    - Output: { data: { violation_id, trip_id, driver_id, speed, speed_limit, speed_over_limit, location, time, created_at }, message }

- GET /speed-violations/trip/{trip_id}
    - Input: path id
    - Output: { data: { violations[], total }, message }

- POST /excessive-braking-violations/
    - Input: CreateExcessiveBrakingViolationRequest
    - Output: { data: { violation_id, trip_id, driver_id, deceleration, threshold, deceleration_over_threshold, location, time, created_at }, message }

- GET /excessive-braking-violations/trip/{trip_id}
    - Input: path id
    - Output: { data: { violations[], total }, message }

- POST /excessive-acceleration-violations/
    - Input: CreateExcessiveAccelerationViolationRequest
    - Output: { data: { violation_id, trip_id, driver_id, acceleration, threshold, acceleration_over_threshold, location, time, created_at }, message }

- GET /excessive-acceleration-violations/trip/{trip_id}
    - Input: path id
    - Output: { data: { violations[], total }, message }

- Driver history analytics (/driver-history)
    - GET /driver-history/{driver_id}
    - Input: path id
    - Output: { data: stats, message }

- GET /driver-history
    - Input: skip, limit, risk_level?, search?
    - Output: { data: { histories, pagination, filters }, message }

- GET /driver-history/{driver_id}/summary
    - Input: path id
    - Output: { data: { driver_id, driver_name, safety_score, risk_level, completion_rate, total_trips, total_violations, last_updated }, message }

- POST /driver-history/recalculate
    - Input: none
    - Output: { data: results, message }

- POST /driver-history/{driver_id}/update
    - Input: path id
    - Output: { data: updated_stats, message }

- GET /driver-history/analytics/risk-distribution
    - Input: none
    - Output: { data: { total_drivers, distribution:{low,medium,high} }, message }

- GET /driver-history/{driver_id}/trips
    - Input: driver_id, skip, limit, status?
    - Output: { data: { trips[], pagination, filters }, message }

7. Security Service

Base: /auth/* and /admin/*

- User: { user_id, name, email, role, created_at }.
- Role: { name, permissions:[string] }.

Authentication (/auth)

- POST /auth/signup
    - Input: { name, email, password }
    - Output: { token, user }.
- POST /auth/login
    - Input: { email, password }
    - Output: { token, user }.
- POST /auth/logout
    - Input: none (header token)
    - Output: { data:{ logged_out:true } }.
- GET /auth/me
    - Input: none

    - Output: user profile.
- POST /auth/verify-token
    - Input: { token? } or header
    - Output: { valid, user }.
- GET /auth/roles
    - Input: None
    - Output: [Role].

Users & Admin (/users, /admin)

- GET /users
    - Input: paging, q, role?
    - Output: [User].
- PUT /users/{user_id}/permissions
    - Input: { role, permissions?[] }
    - Output: User.
- POST /admin/invite-user
    - Input: { email, role, expires_at? }
    - Output: { invitation_id, otp_masked }.
- POST /admin/verify-otp
    - Input: { invitation_id, otp }
    - Output: { verified:true }.
- POST /admin/activate-user/{user_id}
    - Input: None
    - Output: { data:{ activated:true } }.

8. Messaging Contracts

- Request message: { correlation_id, method, endpoint, user_context:{ user_id, roles[], tenant_id? }, data? }.
- Response message (success): { correlation_id, status:'success', data, timestamp }.
- Response message (error): { correlation_id, status:'error', error:{ code, message }, timestamp }.
- Timeouts (seconds): management 30–60; maintenance 30–60; gps 20–30; trips 45–60; security 30.