Module: ERP Base Module

Type

Infrastructure module for ingesting, structuring, and initial aggregation of raw data on self-hosted infrastructure

Purpose

Continuously collect and store vehicle, driver, event, financial, and maintenance data. Serve as the single source of truth for all raw streams, feeding downstream analytics, access control, and reporting modules—while keeping infrastructure spend under $1 000/month for a ~500-vehicle fleet.

Subsystem Structure

Subsystem	Function	Key Data
transport_registry	Vehicle onboarding, park inventory, profile & status	VIN, make/model, registration docs, mileage
telemetry_engine	Sensor data collection (CAN/OBD/GPS)	speed, GPS coords, fuel rate, error codes
driver_module	Driver profiles, documents, ratings, feedback	driver_id, insurance, peer feedback, access status
event_log	Trip, incident & violation logging	trip_id, timestamp, fines, repairs, violations
maintenance_layer	Maintenance scheduling, history & cost tracking	service_date, parts_list, cost, next_due
finance_block	Financial flows management	deposit_id, deductions, operator & investor payouts
supply_core	Inventory management, write-offs & procurement	stock levels, purchase orders, supply analytics
investor_interface	Investor reporting & payout schedules	asset_id, valuation, payout_schedule, contract terms
access_control	Vehicle/driver access rules, lockouts & remote immobilization	rental_status, lockout_log, reason_code, kill_cmd
personnel_tasks	Park staff tasks & KPI tracking	task_id, role, assigned_to, deadline, status
interface_layer	Web & mobile UIs for all user roles	See **Interface Components** below
analytics_kpi_engine	Compute & aggregate `driver_*` & `fleet_*` metrics	sleep_index, penalty_rate, fleet_size, active_count, avg_rev_vehicle, repair_rate, idle_rate

Interface Components

- **operator_portal** Dispatcher web app for task assignment, live fleet monitoring, alerts, and emergency immobilization controls.
- **driver_mobile_app** Smartphone app for drivers: receives routes, reports status, uploads documents.
- **investor_dashboard** Web portal for investors: views asset performance, payout schedules, contract terms.
- **admin_console** Administrative interface: system configuration, user management, health checks, and “Remote Immobilize” button to send Vialon relay commands.

Operation Mode & Data Flow

1. **Ingestion**
   - Real-time streams (sampled at 5–15 sec): telemetry & driver events
   - Batch jobs (every 5–15 min): finance, maintenance, inventory updates
2. **Storage**
   - TimescaleDB/PostgreSQL on 2× self-hosted VPS (4 vCPU, 8 GB RAM, 200 GB SSD)
   - Retention: raw data ≤ 7 days, aggregates ≤ 90 days
3. **Feature Pipeline**

   text

   ```
   [ERP Core System]
       | raw_events, telemetry, finance, maintenance
       ↓
   [Analytics/KPI Engine]
       ↓
   [platform_features Feature Store]
       ↓
   [Consumer Modules]
   ```

4. **Consumers** Normality Corridor, Risk Engine, Coach Engine, FSM Access, IMS, Investor Interface

Output Data (Feature Store)

Field	Description	Source	Update Frequency
driver_sleep_index	Driver sleep quality index	telemetry_engine	real-time / batch
driver_penalty_rate	Count of fines per period	event_log	batch
fleet_size	Total number of vehicles in the fleet	transport_registry	batch
fleet_active_count	Vehicles currently in operation	event_log	batch
avg_rev_vehicle	Average revenue per vehicle	finance_block	batch
repair_rate	Proportion of vehicles under repair	maintenance_layer	batch
idle_rate	Proportion of vehicles idle (no driver/route)	event_log + transport_registry	batch

Field	Description	Source	Update Frequency
…	…	…	…

API Scenarios

Version: v1.0 (backward-compatible) **Authorization:** OAuth2 (JWT Bearer)

1. **Get driver events for a date**
2. `GET /api/v1/event_log/driver/{driver_id}?date=YYYY-MM-DD`
3. **Get recent vehicle telemetry**
4. `GET /api/v1/telemetry/{vehicle_id}?limit=10`
5. **Check driver access status**
6. `GET /api/v1/access/{driver_id}/status`
7. **Fetch fleet financial report**
8. `GET /api/v1/finance/park/{park_id}?period=2025-Q2`
9. **Remote immobilize vehicle**

```
POST /api/v1/access/{vehicle_id}/immobilize
Content-Type: application/json
{
  "method": "vialon_relay",
  "reason": "safety_lockdown"
}
```

• Response 202 Accepted → `{ job_id: "XYZ", status: "pending" }` • GET `/api/v1/access/{vehicle_id}/immobilize/{job_id}` → `{ status: "completed", result: "success" }`

SLA, Performance & Batch Jobs

- **Real-time ingestion latency:** ≤ 30 s
- **API response (p95):** ≤ 200 ms
- **Throughput:** ≥ 1 000 RPS
- **Batch SLA:** each job processed ≤ 5 min; retry/back-off; DLQ for failures
- **Immobilization command latency:** ≤ 15 s end-to-end; retry up to 3 times on failure
- **Scaling:** vertical VPS upgrades; Docker-Compose replicas; scheduled batch windows

Data Quality

Subsystem	Validation	Deduplication
telemetry_engine	Sanity checks, threshold filters	by timestamp
event_log	Schema & business-rule validation	by event hash
driver_module	PII document integrity, uniqueness	by driver_id
finance_block	Reconciliation, balance checks	by transaction_id
maintenance_layer	Schedule & status consistency checks	by maintenance_id

Security & Data Protection

- **Authentication:** OAuth2 (JWT Bearer)
- **Authorization:** RBAC (operator, driver, investor, admin); only **admin** or **specialized operator** can issue `immobilize`
- **Encryption:** TLS 1.2+ (in-transit), AES-256 (at-rest)
- **Vulnerability Management:** regular SAST/DAST; dependency scans; bi-annual pentests
- **Incident Response:** playbooks; IR procedures; automated patch pipelines
- **Audit:** all immobilize commands logged immutably (`user_id`, `timestamp`, `vehicle_id`, `reason`)
- **GDPR / PII:** data classification; pseudonymization; "right to be forgotten" deletion

Observability & CI/CD

- **Logs & Tracing:** Fluentd → Elasticsearch → Kibana; OpenTelemetry
- **Metrics & Dashboards:** Prometheus + Grafana; SLO/SLA alerts
- **CI/CD:** GitHub Actions / Jenkins (lint → test → build → deploy)
- **Orchestration:** Docker Compose for services; cron for batch ETL

Compliance, Audit & Data Governance

- **Audit Logs:** immutable record of all ops & transforms (Postgres WAL + file audit)
- **Retention:** logs & PII stored ≥ 7 years; secure-erase capability
- **Basel III / ESG / SDG:** documented compliance; SHAP explainability reports
- **Data Governance:**
  - **Data Catalog:** https://confluence.company.com/erp-data-catalog
  - **Lineage Documentation:** https://datahub.company.com/lineage
  - **Data Owners:**

Backup & Disaster Recovery

- **Backups:** full daily `pg_dump`; incremental WAL every 4 h
- **Retention:** backups ≥ 30 days
- **RPO:** ≤ 24 h; **RTO:** ≤ 1 h (manual restore)
- **Replication:** optional standby VPS + cold snapshots in object storage
- **DR Drills:** quarterly failover tests with post-mortem reports

Key Use Cases

1. **Onboard New Vehicle**
   - POST `/api/v1/transport_registry` with VIN, model, initial data → **201 Created**
2. **Alert on High Idle Rate**
   - Trigger when `idle_rate > 0.3` for 15 min → Grafana alert → PagerDuty
3. **Generate Daily Fleet Report**
   - Cron job GET `/api/v1/finance/park/{park_id}?period=today` → PDF emailed
4. **Remote Immobilization**

- Operator clicks **Remote Immobilize** in `admin_console` → POST `/immobilize` → Vialon relay command → status polled until **success**

Health Checks & Alerts

- **Endpoint:** GET `/healthz` →

  json

  ```
  { "status": "ok", "uptime": "72h", "lag_ms": 120 }
  ```

- **Grafana Alert Rules:**
  - `avg(api_p95) > 200 ms for 5 min` → Severity P1 → PagerDuty
  - `node_cpu_usage > 80% for 10 min` → Slack notification

Data Consistency Model

- **Real-time writes:** eventual consistency; idempotent keys prevent duplicates
- **Batch jobs:** exactly-once semantics via idempotent processing and DLQ
- **Reads:** always from latest aggregated state in `platform_features`

ETL Orchestration

- **Config Location:** Git repo `infra/etl/dags/*.py`
- **Execution:** cron scheduler on secondary VPS
- **Monitoring:** each DAG posts status to `/api/v1/etl/status`; logs to ELK

Infrastructure & Cost-Saving Summary

- **Monthly spend:** $300–400 for ~500 vehicles
- **Key levers:** self-hosted VPS; TimescaleDB; sampled ingestion; cron-driven ETL; Docker Compose
- **Scales to 10 k vehicles** under $1 000/month; beyond that adopt k8s, Kafka, geo-sharding

Versioning & Roadmap

Version	Goal	Acceptance Criteria	Success Metrics
v1.0	Core ingestion & API	All endpoints operational; p95 < 200 ms SLA met	99.9% uptime
v1.1	Enhanced UI & basic dashboards	React/Streamlit UI live; + 5 new fleet metrics	+ 30% dashboard adoption
v2.0	Advanced KPI & sampling tuning	+ 10 new metrics; ingestion at 10 sec intervals	– 20% data volume
v3.0	Pre-k8s scaling & clustering	Deploy on spot-node k8s; support 100 k vehicles	Sub-5 min batch SLA at scale

Scaling Path & Costs

Phase	Fleet Size	Infrastructure	Monthly Cost
Phase 1	up to 10 k	Vertical VPS upgrade; TimescaleDB shards & replicas	$500–800
Phase 2	up to 100 k	Lightweight k8s (spot nodes); Kafka; Redis cache	$5 000–7 000
Phase 3	up to 1 M	Geo-sharded clusters; serverless peaks; data lake	$30 000–50 000

This comprehensive specification now fully integrates remote immobilization support via Vialon relay, alongside all subsystems, interfaces, data flows, SLAs, security, observability, compliance, DR, use cases, health checks, consistency model, ETL orchestration, cost-saving measures, versioning, and scaling strategy.

Rough Development Cost Estimate

Below are two scenarios—MVP and Full—updated to include remote-immobilization (Vialon relay) integration.

1. MVP Scope (~5 months)

Includes

- All core subsystems (transport_registry, telemetry_engine, event_log, maintenance, finance, supply, investor_interface, access_control w/ immobilization)
- Analytics/KPI Engine + Feature Store
- Interface components (operator_portal, driver_mobile_app, investor_dashboard, admin_console w/ “Remote Immobilize”)
- API, security (OAuth2/JWT, HMAC), monitoring, basic CI/CD, DR

Role	Effort (h)	Rate ($/h)	Cost ($)
Product Manager	160	100	16 000
Solution Architect	120	120	14 400
Backend Developers (2)	2×480 = 960	80	76 800
IoT/Telematics Integrator	160	90	14 400
Data Engineer	320	90	28 800
DevOps Engineer	200	100	20 000
QA Engineer	200	60	12 000
Frontend Developer	200	70	14 000
Subtotal			**196 400**
+20% Contingency			**39 280**
Total MVP			**235 680**

Approx. **$240 K** for 5 months.

2. Full-Featured Product (~9 months)

Adds

- BI dashboards, advanced analytics
- Real-time streaming (Kafka), geo-sharding prep
- Serverless hooks, plug-in extensibility
- Enhanced UI, multi-region DR

Role	Effort (h)	Rate ($/h)	Cost ($)
Product Manager	320	100	32 000
Solution Architect	240	120	28 800
Backend Developers (4)	4×800 = 3 200	80	256 000
IoT/Telematics Integrator	240	90	21 600
Data Engineers (2)	2×640 = 1 280	90	115 200
DevOps Engineer	320	100	32 000
QA Engineers (2)	2×400 = 800	60	48 000
Frontend Developers (2)	2×400 = 800	70	56 000
BI/Analytics Expert	200	100	20 000
Subtotal			**609 600**
+20% Contingency			**121 920**
Total Full			**731 520**

Approx. **$730 K** for 9 months.

Factors Affecting Final Budget

- Seniority and location of team members
- Exact integration complexity with Vialon and telematics hardware
- Extent of real-time vs. batch processing
- Depth of QA (automated tests, load testing, security audits)
- Choice of tools (managed vs. self-hosted)