**Taxi2o – Azure Lakehouse Architecture**

**Table of Contents**

## TOC — Architecture Article

* [Mission](#1-mission)
* [Objectives](#2-objectives)
* [Cloud Architecture (Final)](#3-high-level-architecture-final)
* [Why these Azure services (and trade-offs)](#4-why-these-azure-services-and-trade-of)
  + [4.1 Ingestion](#41-ingestion)
  + [4.2 Storage & File Format](#42-storage--file-format)
  + [4.3 Transform & Serve](#43-transform--serve)
  + [4.4 BI & Consumption](#44-bi--consumption)
* [Pipelines & Orchestration](#6-pipelines--orchestration)
  + [6.1 Event-Based Approach (Master & Sub-Pipelines)](" \l "61-event-based-approach-master--sub-pip)
  + [6.2 Failure Handling & Retries (what happens if things break)](#62-failure-handling--retries-what-happe)
  + [6.3 Parallelism & Dependencies](#63-parallelism--dependencies)
  + [6.4 Timing & SLAs](#64-timing--slas)
* [7) Access Patterns & BI Modes](#7-access-patterns--bi-modes)
* [8) Security & Governance](#8-security--governance)
* [9) Data Quality Playbook (Silver)](#9-data-quality-playbook-silver)
* [10) Cost Management](#10-cost-management)
* [11) What We Improved Across Phases](#11-what-we-improved-across-phases)
* [12) Example KPIs & Dashboards](#12-example-kpis--dashboards)
* [13) DevOps & Folder Structure](#13-devops--folder-structure)
* [14) Implementation Steps (walkthrough)](#14-implementation-steps-walkthrough)

# Mission

Design and implement a secure, scalable cloud-based data platform that unifies Booking, Driver, Vehicle, GPS, and Customer Feedback data to deliver near real-time operational insights and reliable curated reporting using a modern Azure lakehouse.

# Objectives

* Unify core Taxi2o data sources in a single governed platform.
* Ingest both streaming and batch: GPS/app events in real time; systems of record on schedule.
* Adopt Medallion (Bronze → Silver → Gold) with Delta for ACID, time travel, and upserts.
* Serve analytics with Synapse SQL and Power BI using clear access patterns.
* Operational excellence: monitoring, alerts, retries, and fault tolerance by design.
* Security & cost control: private endpoints, Key Vault, RBAC, and right-sized compute.

# Cloud Architecture (Final)

**A diagram of a software flow

AI-generated content may be incorrect.**

1. **Source Systems**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Domain | Examples | Cadence | Key Fields | Notes |
| Customer Info | profiles, segments, contact prefs | Daily/Hourly | customer\_id, segment, status | From CRM/SaaS or SQL |
| Driver Info | onboarding, license, payouts | Daily/Hourly | driver\_id, license\_status, rating | HR/operations system |
| Vehicle Info | VIN/plate, insurance, capacity | Weekly/Daily | vehicle\_id, make/model, capacity | Registry/ERP |
| Booking (events) | requested/accepted/pickup/dropoff/cancel | **Streaming** | trip\_id, state, timestamps | App telemetry |
| GPS Tracking | lat/lon/speed/heading | **Streaming** | device\_id, trip\_id, event\_ts, lat, lon | High volume, low latency |

## 2) Ingestion Layer

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Component | What it does | Inputs | Outputs | Why we chose it |
| **Azure Data Factory** | Orchestrates batch copies & schedules | Customer/Driver/Vehicle DBs & APIs | **Bronze** Delta files (by ingest\_date) | Mature connectors, retries, parameters, cost-efficient for batch |
| **Azure Event Hubs** | Ingests real-time app telemetry | Booking events, GPS pings | Stream to **Bronze** via Spark | High-throughput partitions + checkpoints; analytics-first; simple vs IoT Hub |

## 3) Lakehouse on ADLS Gen2 (Delta)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Layer | Purpose | Typical Partitions | Data Actions | Guarantees |
| **Bronze (Raw)** | Land data **as-is** (replayable) | ingest\_date (batch), event\_date (stream) | Append-only; store bad rows in quarantine | Full fidelity; reproducible backfills |
| **Silver (Clean & Conformed)** | Standardize, dedupe, validate business rules | event\_date/domain | Type casts, UTC, units (km/CAD), dedup, geo & time checks | Analytics-ready, stable schemas |
| **Gold (Star/Marts)** | BI-friendly facts & dims, aggregates | by date/zone/driver | MERGE from Silver; build FactTrip & dims | Fast, governed serving layer |

## 4) Processing (Synapse Analytics)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Engine | Workload | Reads → Writes | Key Logic | Notes |
| **Synapse Spark (Structured Streaming)** | Real-time ingestion & enrichment | Event Hubs → Bronze | JSON parse, schema, checkpoints | 1–5 min freshness targets |
| **Synapse Spark (Batch)** | Bronze → Silver curation | Bronze → Silver | DQ, dedupe, UTC/units, geo sanity, conformance | Quarantine + metrics output |
| **Synapse Spark/SQL** | Silver → Gold modeling | Silver → Gold | MERGE late data, build FactTrip/Dim\* | Idempotent loads |
| **Synapse SQL Serverless** | Ad-hoc queries over lake | Gold Delta | External queries, views | Low cost exploration |
| **Synapse Dedicated (opt.)** | Heavy BI concurrency | Gold → Warehouse | Materialized marts, SLAs | Add only if needed |

## 6) Analytics (Power BI)

|  |  |  |  |
| --- | --- | --- | --- |
| Dataset Mode | When to use | Pros | Cons |
| **Direct Lake** | Delta Gold in the lake, near-real-time | Fast, minimal refresh overhead | Requires Delta model hygiene |
| **Import** | Small/medium curated marts | Sub-second visuals | Scheduled refresh windows |
| **DirectQuery** | Data must remain at source | No duplication | Query-time latency |

## 7) End-User Applications

|  |  |  |  |
| --- | --- | --- | --- |
| Consumer | Examples of Pages | Backed by | Typical Metrics |
| **Driver Dashboard** | Earnings, acceptance/cancel, on-time | Gold facts/dims | On-time %, utilization, ratings |
| **Taxi2o Ops App** | Live heatmaps, ETA accuracy, SLA alerts | Silver (streamed agg) / Gold | Active drivers, wait times |
| **Executives** | Financial & growth KPI | Gold aggregates | Revenue, trips, churn, zone trends |

## 8) Data Flow

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Step | From → To | Mechanism | Trigger | Resulting Table/Asset |
| 1 | Batch systems → **ADF** | JDBC/REST/SaaS connectors | Schedules (hourly/daily) | Raw files staged |
| 2 | **ADF** → **Bronze** | Copy activity (partitioned) | Step 1 success | bronze/<domain>/ingest\_date=... |
| 3 | Apps → **Event Hubs** | SDK/HTTP AMQP | Continuous | Event stream persisted |
| 4 | **Event Hubs** → **Bronze** | Spark Structured Streaming | Continuous + checkpoints | bronze/gps, bronze/bookings (by event\_date) |
| 5 | **Bronze** → **Silver** | Spark batch jobs | After ingest windows | Cleaned, conformed Delta |
| 6 | **Silver** → **Gold** | Spark/SQL MERGE | Hourly/nightly | FactTrip + Dim\* + aggregates |
| 7 | **Gold** → **Power BI / Synapse SQL** | Direct Lake/Import/Serverless | On publish/refresh | Dashboards & ad-hoc queries |

## 9) Event Hubs vs IoT Hub (selection guide)

|  |  |  |
| --- | --- | --- |
| Criterion | Event Hubs | IoT Hub |
| Primary use | Analytics/event ingestion | **Device management** + telemetry |
| Per-device twin/commands | No | **Yes** |
| Typical producers | Mobile apps, services | Managed hardware devices |
| Why Taxi2o chose | App telemetry (Bookings/GPS) without device mgmt | Add later if in-car devices need mgmt |

## 12) SLOs, Costs, and Ops

|  |  |  |
| --- | --- | --- |
| Topic | Target/Practice | Rationale |
| Freshness (stream) | 1–5 minutes | Live ops decisions |
| Freshness (batch marts) | Hourly (ops), nightly (exec) | Cost vs value |
| Retry policy | Exponential backoff ×3, then alert | Resilience without runaway jobs |
| Replay | Keep Event Hubs retention + Bronze history | Deterministic backfills |
| Performance | Delta OPTIMIZE/VACUUM; partition pruning | Consistent BI speeds |
| Cost levers | Autoscale Spark, Serverless SQL for ad-hoc | Avoid over-provisioning |

## 13) What each layer produces (quick map)

|  |  |
| --- | --- |
| Layer | Concrete Outputs |
| **Bronze** | booking\_raw, gps\_raw, customer\_raw, driver\_raw, vehicle\_raw, \*\_quarantine |
| **Silver** | trips, gps, drivers, vehicles, customers (+ DQ metrics) |
| **Gold** | fact\_trip, dim\_date, dim\_driver (SCD2), dim\_vehicle, dim\_customer, dim\_geo, aggregated views |

## 14) Minimal glossary (used in diagram)

|  |  |
| --- | --- |
| Term | Meaning |
| **Delta Lake** | Table format with ACID and MERGE support on ADLS |
| **Bronze/Silver/Gold** | Raw → Clean/Conformed → BI Serving |
| **Structured Streaming** | Spark streaming API with checkpoints |
| **Direct Lake** | Power BI reads Delta files directly (low-latency BI) |

# 6) Pipelines & Orchestration

## 6.1 Event-Based Approach (Master & Sub-Pipelines)

* Master Pipeline (ADF): orchestrates Ingestion → Silver → Gold → Serving refresh.
* Sub-Pipelines: Batch Ingestion; Streaming Ingestion; Curation (Silver + DQ); Gold Build (facts/dims/aggregates).

## 6.2 Failure Handling & Retries

* Retry policy: Exponential backoff (e.g., 1m → 5m → 15m), max 3 attempts per activity.
* After 3 failures: mark run failed, raise alert/ticket with context; quarantine offending batch; skip downstream dependents (fail-fast).
* Partial failures: continue processing unaffected domains; ensure isolation.
* Audit & logging: capture run IDs, row counts (source vs landed vs conformed), and lineage.
* Escalation: alert not acknowledged in 15 minutes auto-escalates to on-call.
* Backfill: replay from Bronze with parameters (date range, source) when fixed.

A diagram of a process

AI-generated content may be incorrect.

## Timing

* Streaming KPIs: 1–5 minute freshness for operational dashboards.
* Batch marts: hourly for operations; nightly for executives.
* Timeouts: per-activity limits aligned to volume (e.g., booking hourly ingest ≤ 10 min).

# Access Patterns & BI Modes

* Direct Lake (preferred for Delta in lake): near real-time without heavy refresh.
* Import: sub-second visuals for curated, modest-sized datasets.
* DirectQuery: keep data at source; accept query-time latency; cache where possible.
* Modeling: star schema, role-playing dates, and semantic measures (DAX)

# Data Quality Playbook (Silver)

* Schema & partition validation per table.
* Required fields: enforce non-null business keys and timestamps; correct types.
* Deduplication rules for Trips and GPS (business keys).
* Geo sanity: coordinates within service area; plausible speed/distance.
* Temporal integrity: pickup before dropoff; driver status valid at trip time.
* Quarantine: store failed rows with reason codes; publish DQ dashboard (null %, dup %, out-of-range %).
* Contracts: Silver is stable; schema changes versioned and flagged.

# Cost Management

* Compute scaling: autoscale Spark, use spot where safe; shut down idle pools.
* Storage hygiene: Delta OPTIMIZE and VACUUM; prune old checkpoints.
* Serving choice: prefer Serverless for ad-hoc; add Dedicated only for strict SLAs.
* Event retention: right-size Event Hubs retention for replay/testing without waste.

# Improvements Across Phases

* Phase 1: Dashboards read raw/curated layers directly; no Serving tier; instability.
* Phase 2: Compute mixed inside storage area; Serving paths not explicit.
* Final: Clear Medallion with Delta; separated Transform vs Serve; documented access patterns and SLOs.

# KPIs & Dashboards

* On-time Pickup % (within X minutes of scheduled time).
* Driver Utilization (time with passengers / total shift).
* Average Wait Time, Cancellation Rate (by zone/driver/hour).
* Revenue per km, Trip success rate, GPS signal quality.
* Heatmaps: active drivers, demand hotspots, SLA breaches.

# Implementation Steps (Walkthrough)

1. Provision: RG, VNet, Key Vault, ADLS Gen2, Event Hubs, Synapse, Purview, Monitor.
2. Secure: Private endpoints; disable public access; RBAC groups; secrets in Key Vault; MSI for services.
3. Ingest (Batch): ADF linked services & datasets; incremental copy to Bronze.
4. Ingest (Stream): Event Hubs + Spark Structured Streaming to Bronze with checkpoints.
5. Curation (Silver): Spark jobs for schema enforcement, DQ, dedupe, business rules; publish metrics.
6. Gold Build: MERGE into Fact/Dim; create aggregates/materialized views.
7. Serve: Expose with Synapse Serverless/Dedicated; model in Power BI (Direct Lake/Import/DirectQuery).
8. Ops: Alert rules, retry policies, backfill/replay runbooks; cost dashboards.
9. Governance: Purview scan + glossary; dataset ownership & SLAs documented.