Task 2 – Full ETL & Analysis on SpaceX

Objective

Design and implement a reproducible ETL pipeline on SpaceX API datasets, integrating multiple endpoints, enforcing a data contract, and producing defensible analysis. This is **individual work**. Submit a complete package at the end.

Scope (offline JSON only)

Use locally saved JSON files from these endpoints:

- v4/launches/past, v4/rockets, v4/launchpads, v4/payloads, v4/cores

Constraints

- **No** pandas.json_normalize or third-party ETL frameworks. Write custom flatteners.
- Work **offline** from JSON files (no new web calls).
- All transforms must be **idempotent** (same inputs → same outputs).
- Timezone: **Europe/Lisbon** everywhere.

Deliverables

1. **Codebase** (Windows-friendly) under src/spacex_etl/ with type hints.
2. **Clean CSVs** in data/clean/, **rejects** in data/out/rejects/, and a data/out/quality_report.json.
3. **Notebook** analysis/eda.ipynb (reads only clean CSVs) with figures + a 1–2 page narrative (findings, caveats, assumptions).
4. **Docs**: README.md, DATA_CONTRACT.md, Decisions.md, ASCII pipeline diagram, and **column-level lineage**.
5. **Signature**: include assignment_signature = SPX-ETL-2025-IdrA in quality_report.json.

Required Features

Pipeline Design

- Star schema:
    - fact_launches (one row per launch)
    - dim_rocket, dim_launchpad, dim_payload, dim_core

    - Bridge tables: bridge_launch_payload, bridge_launch_core (include landing fields)
- Custom flatteners (no json_normalize):
    - flatten_launch(...), flatten_rocket(...), flatten_launchpad(...), flatten_payload(...), flatten_core(...)
- Deterministic runs; include a simple content-hash cache to skip unchanged inputs.

Data Contract (DATA_CONTRACT.md)

- Column names, types, nullability, enums/ranges, and **semantic rules**.
- Definitions for Unknown vs Failure.
- Timezone policy (**Europe/Lisbon**).
- SLAs: completeness thresholds, referential integrity expectations.

Quality Gates

- Schema/type checks; referential integrity across all FKs.
- Rule checks:
    - No future date_utc.
    - Non-negative, sensible payload.mass_kg.
    - If success=False but all cores landing_success=True, flag contradiction.
- Anomaly detection: monthly success rate; flag months **> 3σ** from a trailing 24-month mean.
- Route failures to data/out/rejects/{table}.csv with reason.
- Summarize to data/out/quality_report.json (include assignment_signature).

Analysis (analysis/eda.ipynb)

- Reads **only** from data/clean/.
- Produce:
    - Launches per quarter + cumulative successes.
    - Success rates by **rocket family** (derive families via your explicit regex mapping; document it).

    - Launchpad reliability with **Wilson 95% CI (manual formula)**; compare to normal approximation.
    - Payload mass vs outcome; discuss missingness/confounding.
- Conclude with a 1–2 page narrative (assumptions, limits, what to validate with domain experts).

Tests & Docs

- Unit tests for flatteners, integrity rules, and anomaly logic.
- Decisions.md: key choices + **≥3 rejected alternatives** per major decision.
- ASCII pipeline & dataflow diagram in README.md.
- Column-level lineage mapping each final column to raw fields and transforms.

Submission

- **Preferred**: a **Git repository** link with all deliverables.
- **If the repository isn't ready by the deadline**: send the **zipped project folder** (including data/clean/, data/out/, and the notebook) **via WhatsApp**—same channel as Task 1.