Data Engineering Technical Assessment: Healthcare Eligibility Pipeline

1. Overview

The goal of this assessment is to build a scalable data pipeline that ingests member eligibility files from multiple healthcare partners, each utilizing different file formats. Your solution should output a unified, standardized dataset ready for downstream consumption.

- **Expected Time Commitment:** 1–2 hours
- **Primary Objective:** Demonstrate the ability to build a **configuration-driven** ingestion process that minimizes code changes when adding new data sources.

2. The Scenario

Our platform receives eligibility data from various partners. Currently, we need to onboard two partners: **Acme Health** and **Better Care**. Their file structures differ significantly. Your pipeline must:

1. Ingest both file formats using a central configuration (avoid hardcoded logic).
2. Transform the raw data into a standardized schema.
3. Output a single, unified dataset.

Sample Data

Partner A: "Acme Health" (acme.txt)

- **Format:** Pipe-delimited (|)
- **Data:**

```None
MBI|FNAME|LNAME|DOB|EMAIL|PHONE
1234567890A|John|Doe|03/15/1955|JOHN.DOE@EMAIL.COM|5551234567
9876543210B|Jane|Smith|07/22/1948|jane.smith@email.com|5559876543
```

Partner B: "Better Care" (bettercare.csv)

- **Format:** Comma-delimited (,)
- **Data:**

```None
subscriber_id,first_name,last_name,date_of_birth,email,phone
BC-001,Alice,Johnson,1965-08-10,alice.j@test.com,555-222-3333
```

```
BC-002,Charlie,Brown,1972-03-25,charlie.b@test.com,5554445555
```

3. Requirements

A. Configuration-Driven Ingestion

Define partner configurations (e.g., JSON, YAML, or a Dictionary) that specify:

- File delimiters.
- Column mappings (**Partner Column** $\rightarrow$ **Standard Field**).

> **Success Metric:** Adding a third partner with a different delimiter and column names should only require a configuration update, not a change to the core processing logic.

B. Standardized Output Schema

The final output must adhere to the following transformations:

Field	**Transformation**
external_id	Map from the partner's unique ID field
first_name	Convert to Title Case
last_name	Convert to Title Case
dob	Format as ISO-8601 (`YYYY-MM-DD`)
email	Convert to lowercase
phone	Format as `XXX-XXX-XXXX`

partner_code	Add a hardcoded identifier for the partner

4. Deliverables

Please provide the following:

1. **Source Code:** A working script/notebook that processes both sample files.
2. **Configuration:** The config files or objects used for both partners.
3. **README:** A brief document explaining:
    - Instructions on how to run the pipeline.
    - A short explanation of how to add a new partner.

Technical Choice

You are free to use the tools you are most comfortable with. We use **Databricks/PySpark** internally, solutions with other tools are perfectly acceptable.

5. Evaluation Rubric

Criteria	**What We're Looking For**
Functionality	Does the code run and produce the correct standardized output?
Scalability	Is the ingestion truly configuration-driven?
Code Quality	Is the code readable, organized, and maintainable?

Bonus Points (Optional)

- **Validation:** Ensure external_id is present; flag or drop rows where it is missing.
- **Error Handling:** Gracefully handle malformed rows or incorrect date formats.