# Documentation Technique

Pipeline ETL & Architecture Medallion (Azure/Databricks)

De l'Ingestion Brute à la Valorisation "Gold"

**Yoann LEHONG CHEFFSON**

23 décembre 2025

# Table des matières

# 1 Contexte et Objectifs

Ce document détaille l'implémentation d'un pipeline de données (ETL) moderne conçu pour traiter des flux de données financiers (Cryptomonnaies).

L'objectif principal est de démontrer la maîtrise des concepts clés du Data Engineering :

— Structuration des données selon l'architecture **Medallion** (Bronze/Silver/Gold).

— Optimisation du stockage via le format **Parquet**.

— Automatisation et orchestration des flux de données via Python.

# 2 Architecture Technique

## 2.1 Le Modèle "Medallion" (Multi-Hop)

Le projet repose sur une architecture en couches successives, standard de l'industrie (Databricks / Azure Synapse) :

**Bronze (Raw)**
Zone d'atterrissage des données brutes (format CSV). Les données sont immuables et fidèles à la source.

**Silver (Cleansed)**
Zone de qualité. Les données sont nettoyées, dédoublonnées, typées et converties en format colonnaire (Parquet).

**Gold (Curated)**
Zone de valeur métier. Les données sont agrégées (KPIs) pour alimenter les outils de Reporting (Power BI).

## 2.2 Structure du Projet

L'organisation des fichiers reflète cette architecture logique :

```
crypto_ingestion/
          data/                    # Data Lake local
                bronze/              # CSV bruts
                silver/              # Fichiers Parquet optimis s
                gold/                # KPIs agr g s
          bronze_to_silver.py      # Script de nettoyage
          silver_to_gold.py        # Script d'agr gation
          pipeline.py              # Orchestrateur
          requirements.txt         # D pendances (pandas, pyarrow...)
          .env                     # S curit  (Variables d'env)
```

Arborescence des fichiers

# 3 Implémentation des Flux ETL

## 3.1 Étape 1 : Bronze vers Silver (Nettoyage)

Ce processus transforme le CSV brut en format Parquet. **Pourquoi Parquet ?** Contrairement au CSV (ligne par ligne), le Parquet est un format colonnaire binaire qui permet une compression élevée (jusqu'à -70% de stockage) et des lectures sélectives ultra-rapides.

```
# Typage fort des donn es pour garantir le sch ma
df['extraction_date'] = pd.to_datetime(df['date_extraction'], errors='coerce')
df['price_usd'] = pd.to_numeric(df['price_usd'], errors='coerce')

# Sauvegarde avec compression Snappy
```

```
df.to_parquet(output_parquet, engine='pyarrow', compression='snappy')
```

Listing 1 – Extrait de bronze_to_silver.py

### 3.2 Étape 2 : Silver vers Gold (Agrégation)

Cette étape calcule les indicateurs clés de performance (KPIs) métier.

```
# Calcul des m triques Business
avg_price = df['price_usd'].mean()
total_cap = df['market_cap'].sum()

# Cr ation du Dataset Gold
kpi_df = pd.DataFrame({
    'avg_price_usd': [avg_price],
    'total_market_cap': [total_cap],
    'calculation_date': [pd.Timestamp.now()]
})
```

Listing 2 – Extrait de silver_to_gold.py

## 4 Automatisation (Orchestration)

Pour éviter l'exécution manuelle des scripts, un orchestrateur (`pipeline.py`) a été développé. Il agit comme un "Azure Data Factory" local.

### 4.1 Logique d'exécution

L'orchestrateur garantit l'intégrité du pipeline en gérant les dépendances : l'étape Gold ne se lance **que si** l'étape Silver a réussi.

```
def run_step(script_name):
    print(f"Lancement de {script_name}...")
    exit_code = os.system(f"python {script_name}")
    if exit_code != 0:
        return False  # Arr t en cas d'erreur
    return True

# Ex cution s quentielle
if run_step("bronze_to_silver.py"):
    run_step("silver_to_gold.py")
```

Listing 3 – Logique de dépendance dans pipeline.py

## 5 Conclusion et Perspectives

Ce projet valide la capacité à mettre en place une chaîne de traitement de données complète, robuste et évolutive.

**Prochaines étapes :**

1. Migration du stockage local vers **Azure Data Lake Gen2**.
2. Déploiement de l'orchestration sur **Azure Data Factory**.
3. Visualisation des données Gold via **Power BI**.