Akasa Air - Data Engineering Assignment

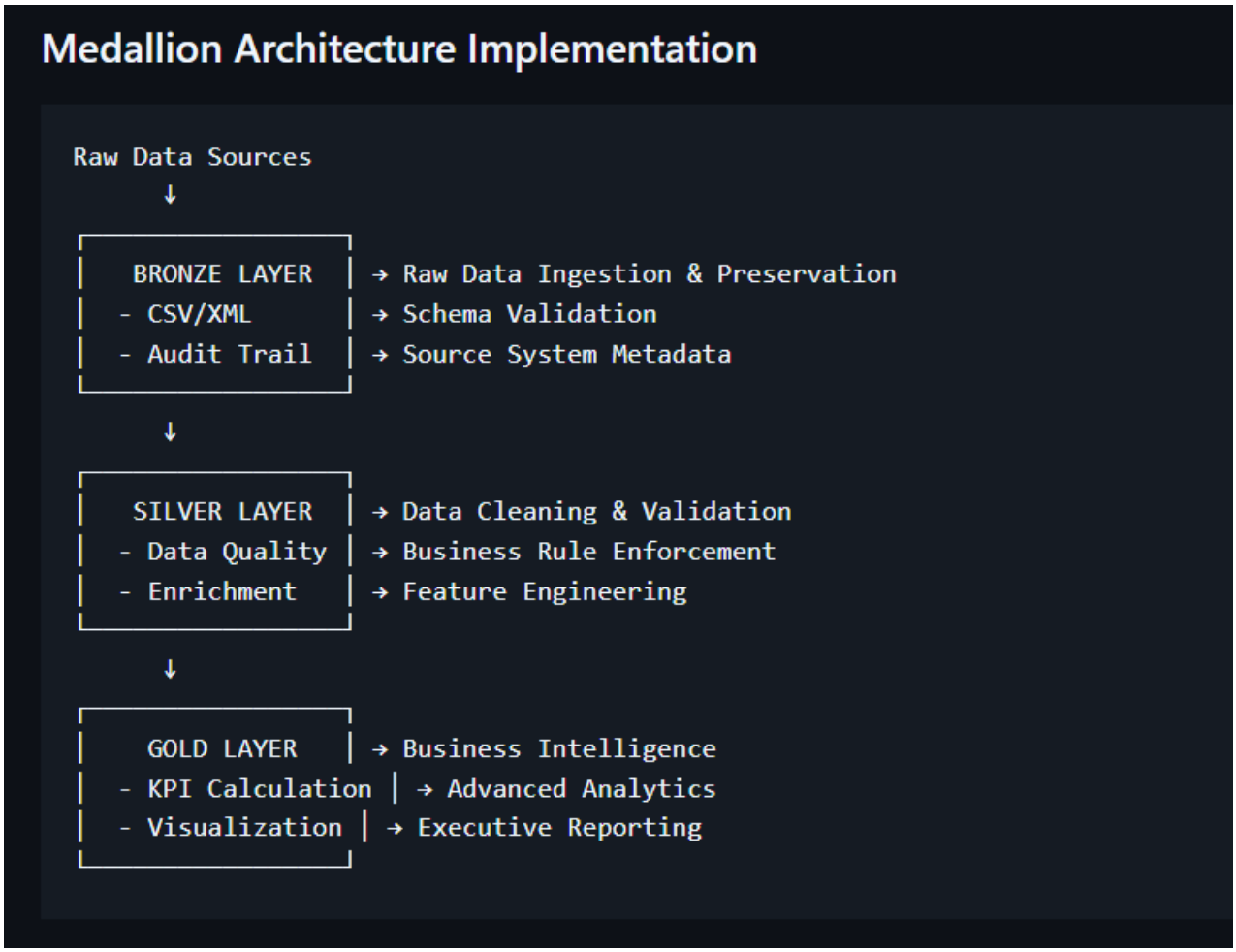
This is customer and order analytics, developed using the Medallion Architecture. The platform ingests data in multiple formats-CSV and XML, performs cleaning over time, and disseminates useful business KPIs through Gold-layer dashboards. Solutions are provided in both in-memory (Pandas) and database (MySQL) environments for more versatility.

- **Table-Based (MySQL) Approach:** SQL-driven processing to ensure scalability and high performance.
- **In-Memory (Pandas) Approach:** Makes use of Python for lightweight analytics, enabling rapid prototyping and ease of integration.

Data Sources

Source	Format	Fields
Customer Data	CSV	customer_id, customer_name, mobile_number, region
Order Data	XML	order_id, mobile_number, order_date_time, sku_id, sku_count, total_amount

Technical Architecture



Project Structure

```
akasa-data-engineer/
│
├── main.py                          # Main pipeline execution script
├── requirements.txt                  # Python dependencies
├── .env.example                     # Environment variables template
├── .gitignore                       # Git ignore rules
│
├── assets/                          # Static assets and outputs
│   ├── task_DE_new_customers.csv    # Sample customer data
│   ├── task_DE_new_orders.xml       # Sample order data
│   ├── analytics_dashboards/        # Generated visualizations
│   │   ├── dashboard_20251107_073641.png
│   │   ├── revenue_20251107_073641.png
│   │   └── regional_20251107_073641.png
│   └── reports/                     # Business reports
│       ├── business_report_20251107_043727.txt
│       └── business_report_20251107_043727.json
│
├── data/                            # Processed data storage
│   ├── bronze/                      # Raw ingested data
│   │   ├── customers_bronze_20251106_230722.parquet
│   │   └── orders_bronze_20251106_230722.parquet
│   ├── silver/                      # Cleaned and enriched data
│   │   ├── customers_silver_20251106_230722.parquet
│   │   ├── orders_silver_20251106_230722.parquet
│   │   └── processing_metadata_20251106_230722.json
│   └── gold/                        # Business KPIs and metrics
│       ├── repeat_customers_20251106_230722.parquet
│       ├── monthly_trends_20251106_230722.parquet
│       ├── regional_revenue_20251106_230722.parquet
│       ├── top_customers_30d_20251106_230722.parquet
│       └── gold_metadata_20251106_230722.json
│
├── logs/                            # Application logs
│   └── akasaair_processing.log      # Main processing log file
│
├── src/                             # Source code
│   ├── __init__.py
│   │
│   ├── config/                      # Configuration management
│   │   ├── __init__.py
│   │   └── config_manager.py        # Paths, logging, environment config
│   │
│   ├── bronze/                      # Bronze Layer - Raw Data Ingestion
│   │   ├── __init__.py
│   │   ├── data_loader.py           # Main bronze data loader
│   │   ├── csv_ingestor.py          # CSV customer data ingestion
│   │   └── xml_ingestor.py          # XML order data ingestion
│   │
│   ├── silver/                      # Silver Layer - Data Cleaning & Enrichment
│   │   ├── __init__.py
│   │   ├── silver_processor.py      # Main silver processor
│   │   ├── data_cleaner.py          # Data cleaning operations
│   │   ├── data_validator.py        # Data validation & quality checks
│   │   └── data_enricher.py         # Feature engineering & enrichment
│   │
│   ├── gold/                        # Gold Layer - Business Intelligence
│   │   ├── __init__.py
│   │   ├── gold_processor.py        # Main gold processor
│   │   ├── kpi_calculator.py        # Core KPI calculations
│   │   └── business_metrics.py      # Additional business metrics
│   │
│   ├── presentation/                # Visualization & Reporting
│   │   ├── __init__.py
│   │   ├── visualizer.py            # Business dashboard generation
│   │   └── report_generator.py      # Text and JSON report generation
│   │
│   └── utils/                       # Utility functions
│       ├── __init__.py
│       ├── database.py              # Database connection & operations
│       └── helpers.py               # Common helper functions
│
└── docs/                            # Documentation
```

Setup and Installation

Prerequisites

- Python 3.8+ (core data pipeline engine).
- MySQL 8.0+ (if choosing database approach).
- Dependency management via `pip install -r requirements.txt`.

Steps

1. Clone the project repository.
2. Install all dependencies.
3. If using database, configure `.env` with DB credentials.
4. Place source CSV/XML files in `./assets` directory.

Run `python main.py` for pipeline execution

- The script:
    - Loads customer (CSV) and order (XML) data.
    - Runs the Table-Based Approach (if MySQL credentials provided).
    - Runs the In-Memory Approach (Pandas).
    - Displays KPI results and logs the workflow to `data_processing.log`.

Sample Output

```
REPEAT CUSTOMERS:
--------------------------------------------------
customer_name  number_of_orders
  Aarav Mehta                2

MONTHLY TRENDS:
--------------------------------------------------
  month  total_orders  total_revenue
2025-09            1          35720
2025-10            1          22350
2025-11            1          15897

REGIONAL REVENUE:
--------------------------------------------------
region  regional_revenue
  West            38247
 North            35720

TOP CUSTOMERS 30D:
--------------------------------------------------
customer_name  recent_spend
  Aarav Mehta         38247

VISUALIZATIONS GENERATED:
--------------------------------------------------
• Comprehensive Dashboard: assets\analytics_dashboards\dashboard_20251107_073951.png
• Revenue: assets\analytics_dashboards\revenue_20251107_073951.png
• Regional: assets\analytics_dashboards\regional_20251107_073951.png

======================================================================
KPI CALCULATION & VISUALIZATION COMPLETE
======================================================================
INFO: Gold layer completed: All business KPIs calculated
INFO: ======================================================================
INFO: TABLE-BASED APPROACH: SQL Database Processing
INFO: ======================================================================
INFO: All KPIs calculated securely using parameterized SQL queries
```

Core Components

Dual Processing Engine

Approach	Use Case	Advantages
In-Memory (Pandas)	Rapid development & testing	Zero dependencies, instant execution
Database (MySQL)	Production workloads	Scalability, ACID compliance, SQL power

Bronze Layer - Raw Data Ingestion

- **CSVIngestor**: Customer data processing with schema validation
- **XMLIngestor**: Order data extraction with structure preservation
- **DataLoader**: Coordination of multi-source data ingestion

Silver Layer - Data Refinement

- **DataCleaner**: Data quality enforcement & standardization
- **DataValidator**: Comprehensive validation rules & integrity checks
- **DataEnricher**: Feature engineering & business context addition

Gold Layer - Business Intelligence

- **KPICalculator**: Core KPI computation engine
- **BusinessMetrics**: Advanced analytics & customer segmentation
- **Visualization Engine**: Professional dashboard generation
- **ReportGenerator**: Multi-format business reporting

Security and Best Practices

- All SQL queries use parameterization through SQLAlchemy, protecting against injection issues.
- Credentials are never exposed in code or logs.
- All input data is validated and sanitized before processing.
- Operational events are logged in data_processing.log for auditability.

Conclusion

This project effectively delivers a strong, dual-approach data processing pipeline: database and in-memory computation for creating business-critical KPIs. The architecture promotes modularity, scalability, and strict data
security.