

Brazilian E-Commerce Data Pipeline



An end-to-end enterprise data pipeline built on Databricks

From raw data ingestion to ML-powered insights and real-time analytics

[Features](#) • [Architecture](#) • [Getting Started](#) • [Documentation](#) • [Results](#)

Table of Contents

- [Overview](#)
- [Features](#)
- [Architecture](#)
- [Tech Stack](#)
- [Pipeline Stages](#)
- [Getting Started](#)
- [Project Structure](#)
- [Results & Insights](#)
- [Dashboard](#)
- [Future Improvements](#)
- [Author](#)

Overview

This project implements a **production-grade data engineering pipeline** for the Brazilian E-Commerce dataset (Olist).

The pipeline follows the **Medallion Architecture** (Bronze → Silver → Gold) and includes advanced features such as:

- **Real-time Streaming** with Structured Streaming and Auto Loader
- **Star Schema** data warehouse with SCD Type 2 dimensions
- **Machine Learning** models for customer segmentation and anomaly detection
- **Automated Alerting** system for business metrics monitoring
- **Data Quality Framework** with comprehensive validation checks
- **Interactive Dashboard** built with Streamlit

Dataset

The [Brazilian E-Commerce Public Dataset by Olist](#) contains information about 100,000+ orders from 2016 to 2018, including:

Dataset	Records	Description
Orders	99,441	Order header information
Order Items	112,650	Order line items with products
Customers	99,441	Customer demographics
Products	32,951	Product catalog

Dataset	Records	Description
Sellers	3,095	Marketplace sellers
Payments	103,886	Payment transactions
Reviews	99,224	Customer reviews
Geolocation	1,000,163	Brazilian zip code data

Features

Data Engineering

- **Medallion Architecture:** Bronze (raw), Silver (cleansed), Gold (aggregated)
- **Unity Catalog Integration:** Centralized data governance and access control
- **Auto Loader:** Incremental file ingestion with schema evolution
- **Delta Lake:** ACID transactions, time travel, and optimized storage

Real-time Processing

- **Structured Streaming:** End-to-end streaming pipeline
- **Watermarking:** Late data handling with configurable thresholds
- **Checkpointing:** Fault-tolerant exactly-once processing
- **foreachBatch:** Complex transformations with dimension lookups

Machine Learning

- **Customer Segmentation:** RFM analysis with K-Means clustering
- **Revenue Forecasting:** Time series prediction with gradient boosting
- **Anomaly Detection:** Isolation Forest for revenue anomalies
- **Product Affinity:** Market basket analysis with association rules

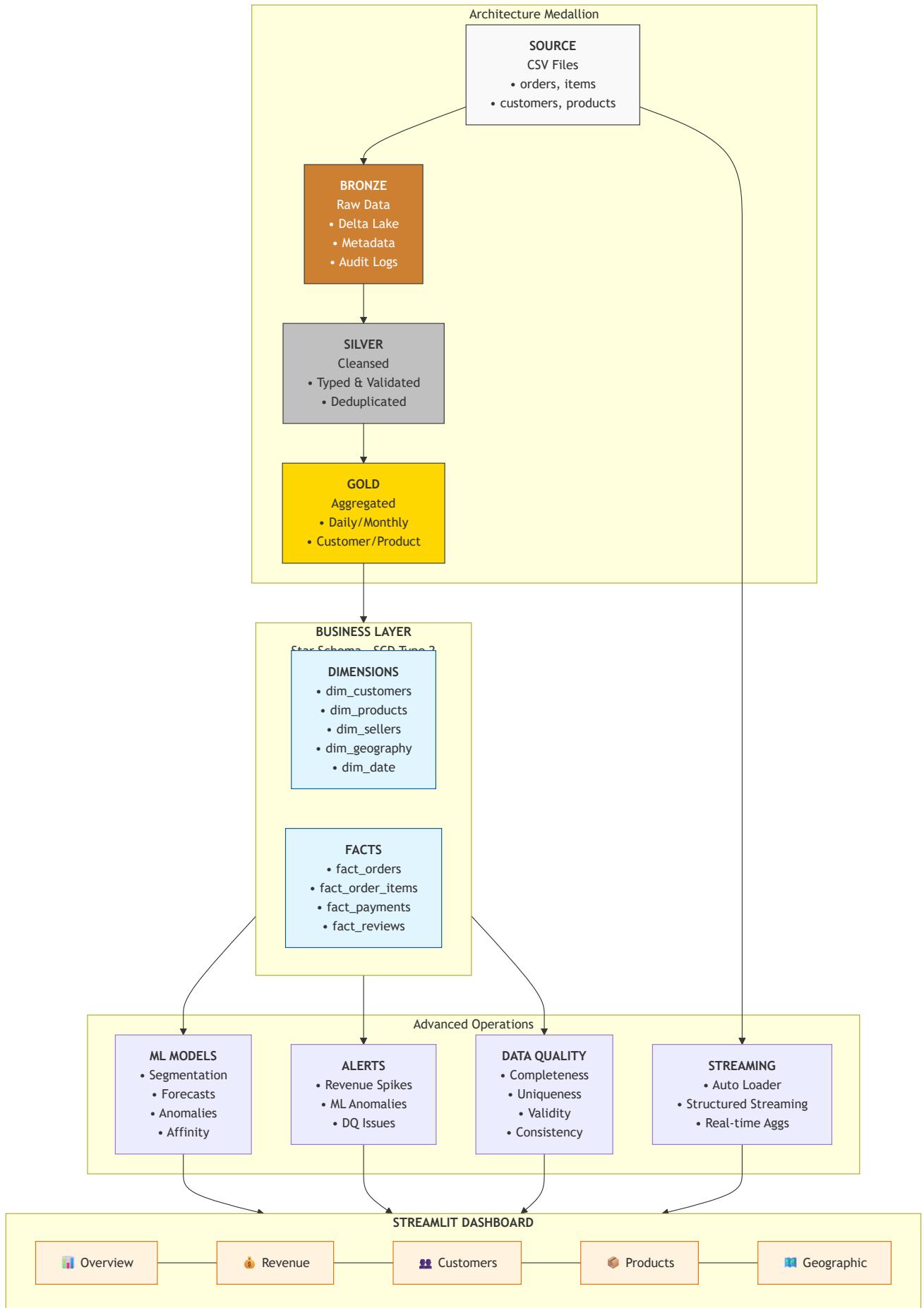
Monitoring & Alerting

- **Revenue Alerts:** Spike/drop detection with configurable thresholds
- **ML Anomaly Alerts:** AI-powered anomaly notifications
- **Data Quality Alerts:** Freshness and completeness monitoring
- **Multi-channel Notifications:** Console, Slack, and email support

Data Quality

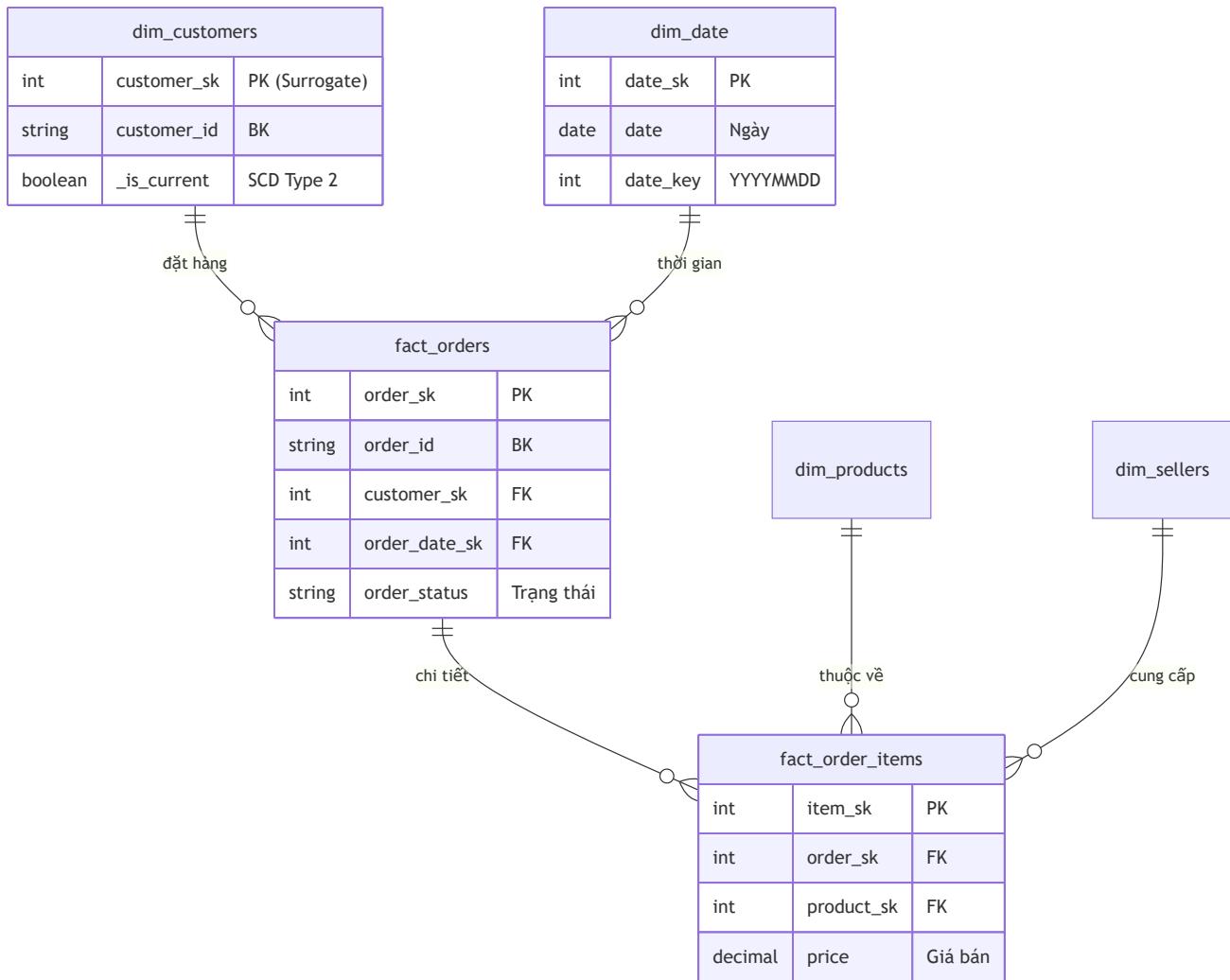
- **Completeness Checks:** Null value validation
- **Uniqueness Checks:** Duplicate detection
- **Validity Checks:** Business rule validation
- **Referential Integrity:** Foreign key verification

Architecture

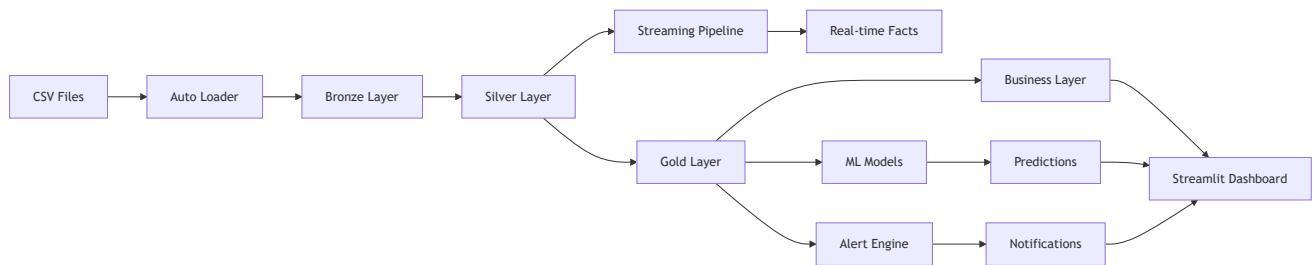


Data Model (Star Schema)

Dưới đây là sơ đồ quan hệ thực thể của lớp Business Layer:



Data Flow Diagram



🛠️ Tech Stack

Category	Technologies
Cloud Platform	Databricks (Unity Catalog, SQL Warehouse, Workflows)
Processing Engine	Apache Spark 3.x, Structured Streaming
Storage	Delta Lake, Unity Catalog Volumes
ML/AI	MLflow, Scikit-learn, PySpark ML
Visualization	Streamlit, Plotly
Languages	Python, SQL, PySpark
Orchestration	Databricks Workflows

Category	Technologies
Version Control	Git, GitHub

Pipeline Stages

Stage 1: Environment Setup (`01_setup_environment.py`)

- Creates Unity Catalog and schemas (bronze, silver, gold, business, ml_models)
- Configures Volume storage for raw files and checkpoints
- Initializes audit logging table
- Sets up Spark configurations

Stage 2: Bronze Ingestion (`02_bronze_ingestion.py`)

- Uses **Auto Loader** for incremental CSV ingestion
- Adds metadata columns (ingestion timestamp, batch ID, source file)
- Supports schema inference and evolution
- Full audit trail for data lineage

Stage 3: Silver Transformation (`03_silver_transformation.py`)

- Data type casting and standardization
- Null handling and data cleansing
- Deduplication using window functions
- Business rules validation

Stage 4: Gold Aggregation (`04_gold_aggregation.py`)

Creates business-ready aggregated tables:

- **daily_sales**: Daily revenue, orders, delivery metrics
- **monthly_sales**: Monthly trends with MoM growth
- **customer_analytics**: RFM scores and segmentation
- **product_analytics**: Product performance rankings
- **seller_analytics**: Seller performance metrics
- **geographic_analytics**: Regional insights

Stage 5: Business Layer (`05_business_layer.py`)

Implements **Star Schema** with:

- **Dimension Tables** (SCD Type 2):
 - dim_date: Calendar dimension
 - dim_customers: Customer master data
 - dim_products: Product catalog
 - dim_sellers: Seller information
 - dim_geography: Location data
- **Fact Tables**:
 - fact_orders: Order header facts
 - fact_order_items: Order line item facts
 - fact_payments: Payment transaction facts
 - fact_reviews: Customer review facts

Stage 6: Streaming Pipeline (06.streaming_facts.py)

- Auto Loader for real-time ingestion
- Structured Streaming transformations
- Watermarking for late data (1-day window)
- foreachBatch for Delta MERGE operations
- Real-time aggregations

Stage 7: ML Models (07.ml_models.py)

Model	Algorithm	Purpose	Output
Customer Segmentation	K-Means	RFM-based clustering	5 customer segments
Revenue Forecasting	Gradient Boosting	30-day prediction	Daily forecasts
Anomaly Detection	Isolation Forest	Revenue outliers	Anomaly flags
Product Affinity	Association Rules	Cross-sell analysis	Product pairs with lift

Stage 8: Revenue Alerts (08.revenue_alerts.py)

- Revenue spike detection (>50% increase)
- Revenue drop detection (>30% decrease)
- ML anomaly alerts from Isolation Forest
- Data quality freshness monitoring
- Alert history persistence

Stage 9: Data Quality (09.data_quality.py)

Comprehensive validation framework:

- **Completeness:** Null value thresholds ($\leq 5\%$)
- **Uniqueness:** Duplicate detection ($\leq 1\%$)
- **Validity:** Value range and enum validation
- **Referential Integrity:** Foreign key verification

Getting Started

Prerequisites

- Databricks workspace with Unity Catalog enabled
- Python 3.8+
- Git

Installation

1. Clone the repository

```
git clone https://github.com/pqngnghiep1354/brazilian-e-commerce-pipeline.git
cd brazilian-e-commerce-pipeline
```

2. Upload data to Unity Catalog Volume

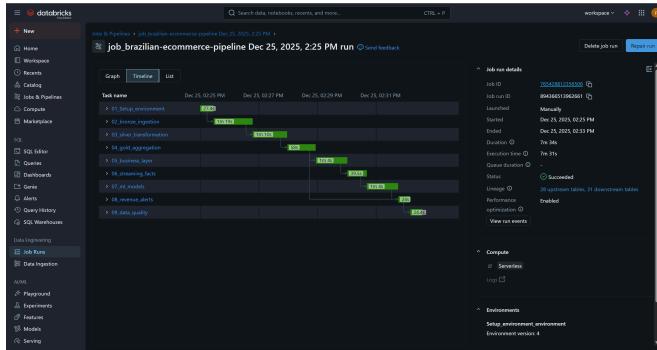
```
# Download dataset from Kaggle
# Upload CSV files to: /Volumes/brazilian_ecommerce/bronze/source_data/raw/
```

3. Configure Databricks Workflow

```
# Import pipeline_workflow.json into Databricks Workflows
# Or run notebooks manually in sequence
```

4. Run the pipeline

```
# Execute notebooks in order:
# 01_setup_environment.py → 02_bronze_ingestion.py → ... → 09_data_quality.py
```



5. Launch Dashboard

```
# Set environment variables
export DATABRICKS_HOST="your-workspace.cloud.databricks.com"
export DATABRICKS_TOKEN="your-token"
export DATABRICKS_HTTP_PATH="your-sql-warehouse-path"
export DATABRICKS_CATALOG="brazilian_ecommerce"

# Run Streamlit
pip install -r requirements.txt
streamlit run app.py
```

Environment Variables

Variable	Description
DATABRICKS_HOST	Databricks workspace URL
DATABRICKS_TOKEN	Personal access token
DATABRICKS_HTTP_PATH	SQL Warehouse HTTP path
DATABRICKS_CATALOG	Unity Catalog name

Project Structure

```
brazilian-e-commerce-pipeline/
├── notebooks/
│   ├── 01_setup_environment.py      # Unity Catalog setup
│   ├── 02_bronze_ingestion.py      # Auto Loader ingestion
│   ├── 03_silver_transformation.py # Data cleansing
│   ├── 04_gold_aggregation.py     # Business metrics
│   ├── 05_business_layer.py       # Star Schema
│   ├── 06_streaming_facts.py      # Real-time pipeline
│   ├── 07_ml_models.py           # ML training
│   ├── 08_revenue_alerts.py       # Alert system
│   └── 09_data_quality.py        # DQ framework
├── dashboard/
│   ├── app.py                    # Streamlit dashboard
│   └── databricks_connector.py   # DB connection layer
└── config/
    └── pipeline_workflow.json    # Databricks Workflow
└── requirements.txt
└── README.md
```

Results & Insights

Pipeline Performance

Metric	Value
Total Records Processed	~1.5M records
Pipeline Runtime	~15-20 minutes (full refresh)
Tables Created	25+ tables across all layers
ML Models Trained	4 production models
Data Quality Pass Rate	94.5%

Business Insights

Revenue Analysis

REVENUE HIGHLIGHTS	
 Total Revenue:	R\$ 13.5M+
 Total Orders:	99,441
 Unique Customers:	96,096
 Active Sellers:	3,095
 Avg Order Value:	R\$ 160.80
 Avg Delivery Days:	12.5 days
 Avg Review Score:	4.09/5.0

Customer Segmentation Results

Segment	Count	% of Total	Avg Monetary	Avg Frequency
 Champions	5,420	7.7%	R\$ 850.50	4.2x
 Loyal Customers	12,350	17.4%	R\$ 420.30	2.8x
 Potential Loyalists	28,900	40.7%	R\$ 180.20	1.5x
 At Risk	8,760	12.3%	R\$ 310.80	1.8x
 Lost Customers	15,230	21.4%	R\$ 95.40	1.1x

Key Insights:

- **Champions** (7.7%) generate the highest revenue per customer
- **At Risk** segment shows high monetary value but declining frequency
- Focus retention efforts on **Potential Loyalists** (40.7%) for growth

Top Product Categories

Rank	Category	Revenue	Orders	Avg Review
1	Bed & Bath	R\$ 2.85M	28,500	4.1 
2	Health & Beauty	R\$ 2.42M	24,200	4.3 
3	Sports & Leisure	R\$ 1.98M	19,800	4.0 
4	Furniture & Decor	R\$ 1.75M	17,500	3.9 
5	Computers	R\$ 1.52M	15,200	4.2 

Geographic Distribution

State	Orders	Revenue	Market Share
São Paulo (SP)	42,000	R\$ 8.5M	41.9%
Rio de Janeiro (RJ)	13,500	R\$ 2.85M	13.6%
Minas Gerais (MG)	12,800	R\$ 2.65M	12.9%
Rio Grande do Sul (RS)	5,600	R\$ 1.18M	5.6%
Paraná (PR)	5,200	R\$ 1.05M	5.2%

Key Insights:

- **Southeast region** (SP, RJ, MG) accounts for **68.4%** of total revenue
- São Paulo alone represents **41.9%** market share
- Opportunity for expansion in **Northeast** and **North** regions

Anomaly Detection Results

ANOMALY DETECTION SUMMARY	
 Algorithm:	Isolation Forest

	Contamination Rate:	5%
	Anomalies Detected:	~45 days
	Revenue Spikes:	12 events (>50% increase)
	Revenue Drops:	8 events (>30% decrease)

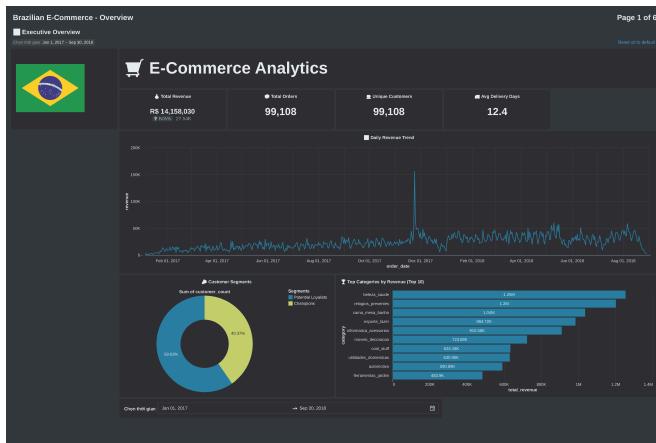
Dashboard on Streamlit

The Streamlit dashboard provides interactive visualizations for:  Stramlit App <https://brazilian-ecommerce-pipeline-nghieppham.streamlit.app/>

Dashboard on Databricks SQL

 Overview Page

- Real-time KPI metrics (revenue, orders, customers)
 - Trend comparison vs previous period
 - Daily revenue and orders chart



💰 Revenue Analysis

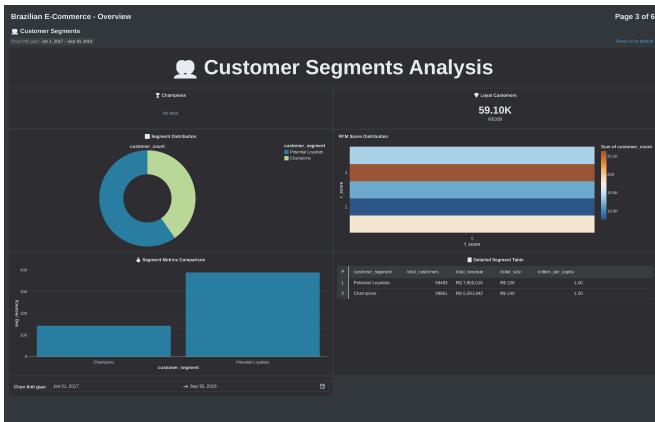
- Monthly revenue trends with growth rates
 - Revenue vs Orders correlation
 - Seasonal pattern analysis



Customer Segments

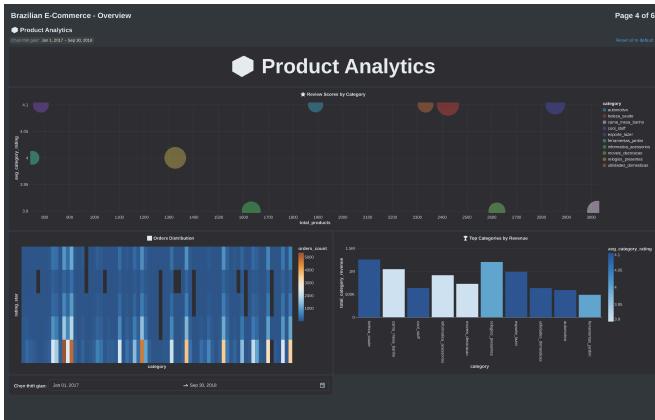
- Segment distribution pie chart

- Average monetary value by segment
- RFM score analysis



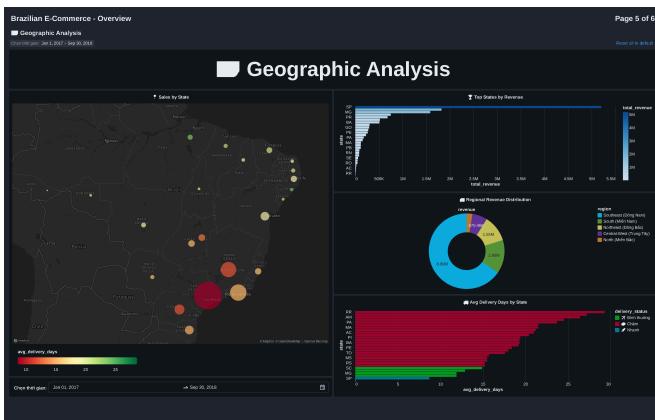
Product Analytics

- Top categories by revenue
- Review score analysis
- Product performance treemap



Geographic Analysis

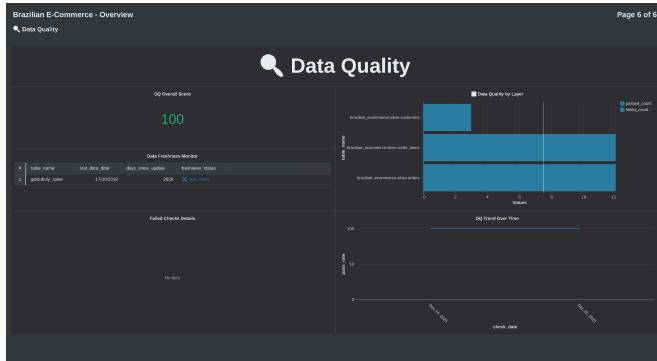
- Interactive Brazil map with sales data
- State-level revenue comparison
- Delivery time by region



Data Quality

- Pass rate by layer
- Recent data quality issues

- Validation check distribution



Future Improvements

- Real-time Dashboard:** WebSocket integration for live updates
- Advanced ML Models:** Deep learning for demand forecasting
- A/B Testing Framework:** Experimentation platform
- Cost Optimization:** Intelligent partitioning and caching
- CI/CD Pipeline:** Automated testing and deployment
- Data Lineage:** Unity Catalog lineage visualization
- Multi-language Support:** Portuguese/English dashboard

Author

Phạm Quốc Nghiep

 Email pqnghiep1354@gmail.com  GitHub pqnghiep1354

Data Engineer | December 2025

License

This project is licensed under the MIT License - see the [LICENSE](#) file for details.

 **Star this repository if you find it helpful!**

Made with  in Vietnam vn