Building a Scalable Financial Data Pipeline for Customer Analytics

Bootcamp - AWS Data Engineering Project – 1

Objective

The project addresses key challenges in retail banking by providing **data-driven insights** into customer transactions and loan behaviors. The pipeline will support:

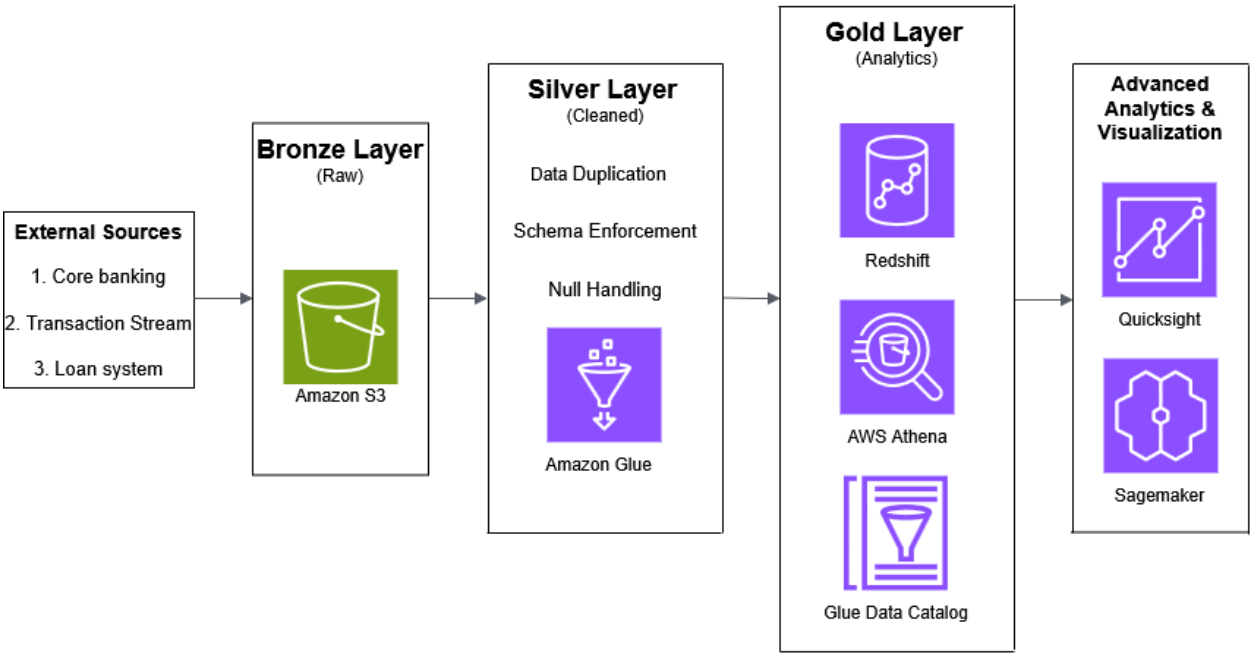
- **Risk management** – Early detection of potential loan defaults
- **Customer personalization** – Tailored financial product recommendations
- **Operational efficiency** – Automated fraud detection and reporting
- **Regulatory compliance** – Audit-ready data lineage and quality controls

The solution must balance **scalability, cost-efficiency, and business agility** while handling sensitive financial data.

System Architecture:

The solution is structured into multi-layered zones based on the medallion architecture:

1. Bronze Layer: Raw data ingestion from multiple sources.
2. Silver Layer: Processed and standardized data.
3. Gold Layer: Business-ready analytics and dashboards.



Prerequisite:

1. Install Visual Studio code application.
2. Install Terraform
3. Prepare raw data source for analysis.

How to install Terraform: Install Terraform.docx

Component Breakdown:

Component	Description
Terraform + S3	Used to provision the data lake structure. Created Bronze, Silver, Gold folders
AWS Glue	Used for ETL jobs across layers; performs cleaning, joining, aggregation
Athena	Used to query curated data from the Gold layer stored in Parquet format
Glue Crawler	Crawls Silver and Gold data to register them as tables in Glue Data Catalog
Databricks	Runs machine learning pipeline for loan default prediction using PySpark
Logistic Regression	ML model used to calculate probability of loan default per customer

Design Decisions:

Topic	Choice	Why
Architecture	Medallion (Bronze → Silver → Gold)	Scalable, modular, and supports clean separation of raw and refined data
ETL Tool	AWS Glue (PySpark)	Serverless, supports schema enforcement and integrates well with S3
ML Platform	Databricks	Simplified distributed training and model deployment with Spark
Model Type	Logistic Regression	Interpretable binary classification model for risk prediction
Storage Format	Parquet	Optimized columnar format for analytics and Athena
Query Layer	Athena with Glue Catalog	Serverless SQL engine, easy to visualize in QuickSight or alerts

Data Flow:

```
Raw CSVs (Bronze Layer in S3)
      ↓
AWS Glue Job (cleaning, schema enforcement)
      ↓
Cleaned Parquet files (Silver Layer in S3)
      ↓
AWS Glue Job (joins, aggregations)
      ↓
Customer-level summaries (Gold Layer in S3)
      ↓
Glue Crawler → Glue Catalog → Athena Tables
      ↓
Athena SQL queries + Databricks ML + Dashboards
```

Security and Compliance:

- IAM Roles: IAM roles used for Glue and Athena to access only necessary S3 buckets
- Data Separation: Layered architecture separates raw vs refined data to control access
- S3 Bucket Policies: Applied through Terraform to control data access and encryption
- Auditability: Data lineage is clear across Bronze → Silver → Gold transformations
- PII Handling: Sensitive columns like customer IDs are preserved but never duplicated or exposed unnecessarily

Monitoring & Quality:

Aspect	Approach
Data Quality	Null filters, duplicate removal, and type enforcement in the Silver layer
Job Monitoring	AWS Glue job logs and step-by-step PySpark transformations
Query Validity	Athena used to validate output datasets (Gold) using SQL LIMIT and filters
Model Evaluation	AUC and ROC curve plotted in Databricks to assess prediction quality
Alerting (Future Scope)	Lambda + Athena query + SNS planned to detect high-risk customers automatically