Building a Modern Banking Data Platform on Azure

A comprehensive 5-day engineering journey to create an enterprise-grade, real-time data platform for fraud detection, customer analytics, and regulatory compliance.

Made with GAMMA

The Banking Data Challenge

The Problem

Banks generate massive volumes of data from ATMs, UPI transactions, mobile banking, branches, and core systems. Legacy infrastructure struggles with real-time processing, security, and scalability requirements.

- Fragmented data sources creating silos
- Delayed fraud detection
- Incomplete customer profiles
- Manual deployment processes

The Solution

Build a modern, cloud-native data platform on Azure that delivers unified analytics, real-time insights, and automated operations.

- Real-time fraud detection
- Unified Customer 360 profiles
- Secure, scalable storage
- Automated CI/CD pipelines
- Business and regulatory reporting

Made with GAMMA

Project Objectives and Technical Stack

Data Ingestion

Collect ATM, UPI, customer, and core banking data in real-time and batch modes using Azure Functions and Event Grid.

Transformation

Clean, validate, merge, deduplicate, and standardize data using PySpark in Azure Databricks and Synapse.

Operational Store

Build Cosmos DB containers for real-time transaction lookups with millisecond latency and global distribution.

Data Warehouse

Create dimensional models with fact and dimension tables in Azure SQL Database for analytics and reporting.

Real-Time Alerts

Trigger instant fraud alerts using Event Grid subscriptions and serverless Azure Functions for immediate response.

Security & CI/CD

Protect resources with VNet integration and Azure Firewall. Automate deployments using Jenkins and GitHub for continuous delivery.

Made with GAMMA

Day 1

Data Architecture and Ingestion Layer

Day 1 establishes the foundational architecture with event-driven ingestion. We create a multi-layered data lake structure (raw → bronze → silver → gold) and implement real-time file detection using Azure Event Grid.

01

Create ADLS Gen2 Storage

Enable Hierarchical Namespace and create containers for raw/atm, raw/upi, raw/customer, and raw/master data sources.

02

Configure Event Grid

Set up Event Grid subscription to detect "Blob Created" events and route them to Azure Function endpoints for processing.

03

Build Event Grid Trigger Function

Develop Python Function to capture file metadata, validate file type, and push messages to Storage Queue for scalable processing.

04

Deploy Queue Infrastructure

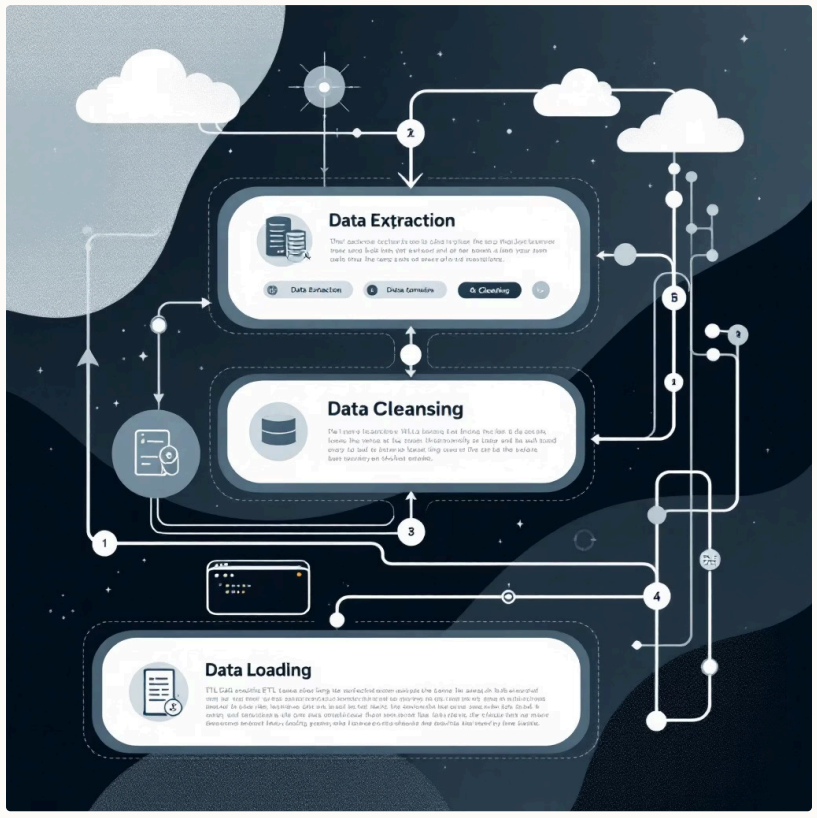
Create ingestion-queue for decoupling event detection from processing, enabling horizontal scaling and fault tolerance.

Made with GAMMA

Day 2

Queue Processing and Data Cleaning Pipeline

Day 2 focuses on consuming queued messages, reading files from Blob Storage, performing comprehensive data validation and cleaning, then loading the processed data into Cosmos DB operational containers.



Data Cleaning Operations

- Remove empty and duplicate rows
- Standardize date/time formats across sources
- Convert amount fields to numeric types
- Normalize column names and remove special characters
- Validate required columns and detect missing values
- Classify transactions as UPI or ATM type

Queue Trigger

Function automatically wakes when message arrives

File Retrieval

Download CSV from Blob using message metadata

Clean with Pandas

Apply validation rules and transformations

Convert to JSON

Transform rows to JSON documents for Cosmos DB

Insert to Cosmos

Load into ATMTransactions or UPIEvents containers

Made with GAMMA

Day 3

Data Warehouse and PySpark ETL

Day 3 transforms cleaned operational data into an analytics-ready SQL warehouse. Using PySpark in Databricks or Synapse, we read from Cosmos DB, apply dimensional modeling techniques, and create fact and dimension tables.

Extract from Cosmos DB

Read cleaned ATM and UPI transaction data using PySpark connector with optimized partition strategies.

Transform in Bronze/Silver Layers

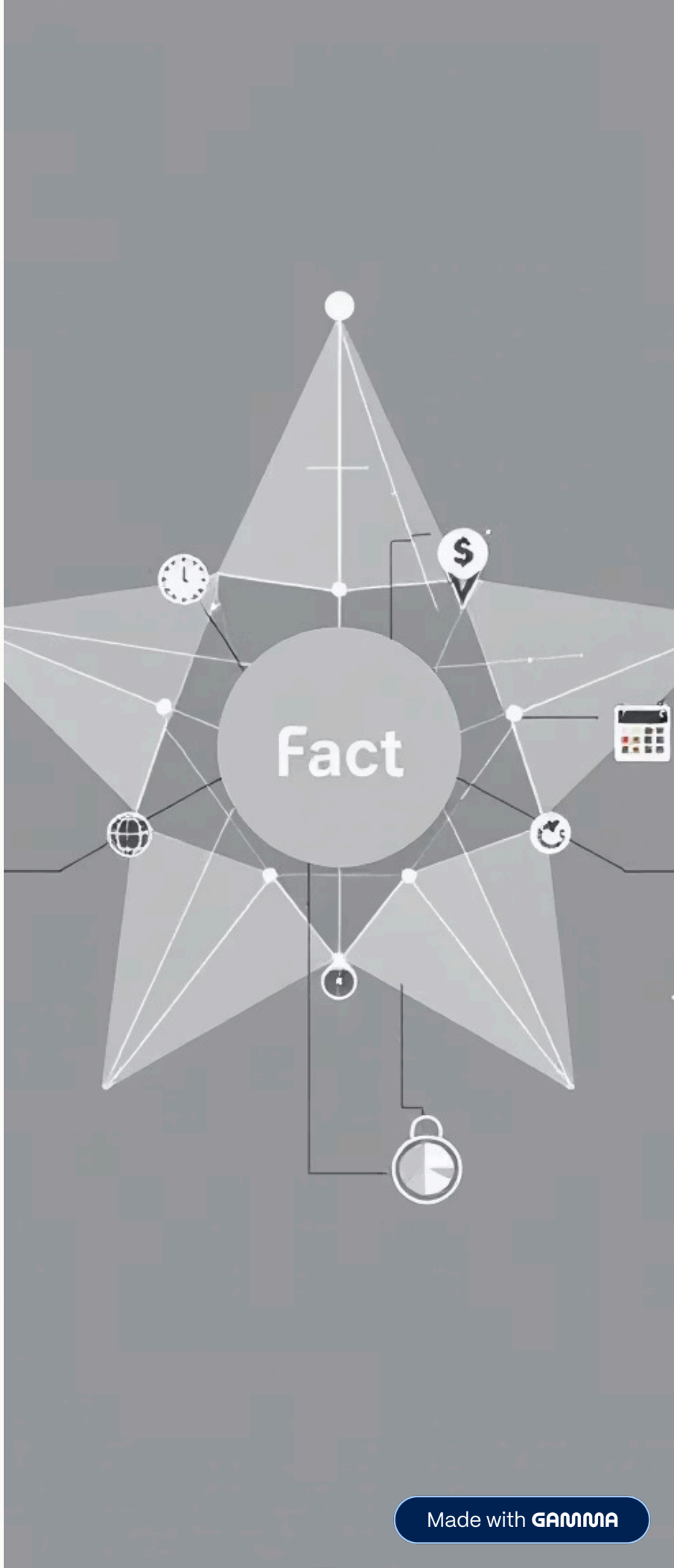
Combine data sources, add calculated fields, fix data types, handle nulls, and apply business logic transformations.

Build Dimensional Model

Create dimension tables (DimCustomer, DimProduct, DimBranch, DimDate) and fact tables (FactTransactions, FactFraudDetection).

Load to Azure SQL

Write gold layer data to Azure SQL Database using bulk insert operations for optimal performance and query speed.



Made with GAMMA

Day 4

Real-Time Fraud Detection and Security

Fraud Detection Architecture

Implement event-driven fraud detection using Azure Function with Event Grid trigger. The function analyzes transaction patterns in real-time and writes alerts to Cosmos DB FraudAlerts container.

- Pattern matching for suspicious activity
- Threshold-based amount detection
- Velocity checks for rapid transactions
- Geo-location anomaly detection
- Real-time customer risk scoring

CI/CD Pipeline with Jenkins

Automate the entire deployment workflow from code commit to production deployment using Jenkins, GitHub, and Azure CLI.

1

Push to GitHub

Commit Python functions and infrastructure code to version control repository

2

Jenkins Webhook

Trigger automated build pipeline on commit to main branch

3

Run Tests

Execute unit tests, integration tests, and code quality checks

4

Deploy to Azure

Package and deploy Functions, ETL jobs, and SQL scripts to Azure environment

Made with GAMMA

Day 5

Power BI Reporting and Analytics

Day 5 delivers business value through comprehensive reporting. We build two main dashboards using Power BI connected to Azure SQL: Fraud Analytics for security teams and Customer 360 for relationship managers.

Fraud Analytics Dashboard

Key Metrics: 501 accounts monitored, 255M total transaction volume, 14 fraud events detected

- Suspicious transaction identification
- Customer risk behavior analysis
- Fraud trend visualization over time
- Alert distribution by channel and location

Customer 360 Dashboard

Unified View: Complete customer profile integrating all data sources for relationship management

- Personal and account details
- Transaction history and patterns
- Product usage and preferences
- Fraud and risk profile scores
- Channel preferences (UPI/ATM/Branch)

Technical Architecture Overview

Event Grid

Event-driven architecture for real-time file detection and fraud alerts with publish-subscribe pattern.

Azure Functions

Serverless compute for data ingestion, processing, and fraud detection with automatic scaling.

ADLS Gen2

Hierarchical data lake with bronze, silver, and gold layers for structured data processing.

Cosmos DB

Globally distributed NoSQL database for operational transactions with sub-10ms latency.

Azure SQL

Relational data warehouse with dimensional modeling for analytics and reporting workloads.

PySpark

Distributed data processing engine for ETL transformations and large-scale data engineering.

Made with GAMMA

Project Outcomes and Next Steps

5	100%	<10ms	255M
Days to Production	**Automated**	**Query Latency**	**Transaction Volume**
Complete platform deployed from architecture to reporting	End-to-end CI/CD pipeline with zero manual deployment steps	Real-time transaction lookups in Cosmos DB operational store	Platform handles massive scale across ATM and UPI channels

Key Achievements

- Event-driven architecture for real-time processing
- Multi-layered data lake (bronze/silver/gold)
- Operational and analytical data stores
- Automated fraud detection pipeline
- Comprehensive data quality framework
- Dimensional modeling for analytics
- CI/CD automation with Jenkins
- Business intelligence dashboards

> This platform establishes the foundation for modern banking operations with scalable architecture, real-time insights, and automated workflows ready for production deployment.

Made with GAMMA

Day-wise Proper Problems + How They Were Solved

Day-1 Problems (Architecture + Ingestion Layer)

Problem 1 — Event Grid was not triggering the Function

Reason: Missing permission + wrong endpoint during subscription.
Solution:

- Enabled Storage Event Subscription
- Re-created Event Grid with correct Function endpoint
- Added correct RBAC roles for Event Grid → Storage → Function.

Problem 2 — File not detected in raw container

Reason: Raw folder path mismatch (raw/ATM vs raw/atm).
Solution: Standardized folder names and updated function config.

Problem 3 — Function App failed to start

Reason: Python version mismatch in Azure (local 3.12, Azure supports 3.10).
Solution: Re-created Function App with Python 3.10 runtime

Made with GAMMA

Day-2 Problems (Queue Trigger + Data Cleaning + Cosmos DB)

Problem 1 — Queue Trigger not firing

Reason: The queue name in function.json didn't match actual queue "ingestion-queue".
Solution: Updated queue binding → restart Function → successful trigger.

Problem 2 — CSV schema mismatch (missing/extra columns)

Reason: ATM and UPI files had unequal column names.
Solution: Added schema validation + normalization logic in pandas.

Problem 3 — Duplicate data kept going to Cosmos DB

Reason: Raw files contained repeating transactions.
Solution: Applied:

- `drop_duplicates()`
- created unique TransactionID key
- validated before Cosmos insert.

Problem 4 — Cosmos DB insert failure

Reason: Wrong partition key used (/TransactionId instead of /TransactionID).
Solution: Recreated containers with correct partition key.

Made with GAMMA

Day-3 Problems (ETL + Data Warehouse + PySpark)

Problem 1 — Cosmos DB → PySpark connector error

Reason: Missing connection JAR + wrong account key.
Solution: Added correct Maven coordinates + updated config keys.

Problem 2 — Bronze → Silver transformations failing

Reason: Inconsistent date formats across ATM/UPI.
Solution: Applied PySpark date parsing:
`to_timestamp()` with multiple fallback formats.

Problem 3 — Fact table creation error in SQL

Reason: Null CustomerID causing join failures.
Solution:

- Cleaned null rows in Silver
- Implemented surrogate key for unknown customers.

Problem 4 — Large file processing taking long time

Reason: Small cluster size.
Solution: Increased cluster to 4-node, optimized transformations.

Made with GAMMA

Day-4 Problems (Fraud Detection + CI/CD + Security)

Problem 1 — FraudAlertFunction not deploying

Reason: Missing Cosmos DB connection string in Application Settings.
Solution: Added:

- COSMOS_URL
- COSMOS_KEY
- DB_NAME
- CONTAINER
  Then redeployed successfully.

Problem 2 — Event Grid alert not reaching fraud function

Reason: Subscriber endpoint was blocked by Function App firewall.
Solution: Allowed Azure service access + adjusted network restrictions.

Problem 3 — Jenkins pipeline failing

Reason: Missing Python dependencies in requirements.txt.
Solution: Updated repo → triggered successful CI/CD run.

Problem 4 — Permission denied to push to GitHub

Reason: Missing PAT token for Jenkins Git operations.
Solution: Added GitHub Personal Access Token to Jenkins credentials.

Made with GAMMA

Day-5 Problems (Power BI + Customer 360 + Dashboard)

Problem 1 — Power BI relationship errors

Reason: Many-to-many relationships between FactTransactions & DimCustomer.
Solution:

- Fixed by adding bridge table
- Used surrogate keys
- Managed relationships properly.

Problem 2 — DAX measures showing wrong numbers

Reason: Data type mismatch (Amount stored as text).
Solution: Converted to numeric type in Gold layer + refreshed Power BI.

Problem 3 — Customer 360 not showing complete profile

Reason: Missing joins between FactFraudDetection & DimCustomer.
Solution: Re-created model using CustomerKey as the primary join.

Problem 4 — Dashboard refresh was slow

Reason: Large fact tables without indexing.
Solution: Partitioned FactTransactions and optimized SQL.

Made with GAMMA

Project Summary: Modern Banking Data Platform

Our project successfully built an end-to-end modern banking data platform on Azure, enabling real-time fraud detection, comprehensive Customer 360 insights, and advanced analytics capabilities.

Azure Ecosystem

Leveraged Azure for a scalable and secure cloud infrastructure, integrating diverse services for data ingestion, processing, storage, and analytics.

Real-time Processing

Implemented event-driven architecture with Event Grid and Azure Functions for immediate data ingestion and transformation from various banking sources.

Unified Data Views

Established a multi-layered data lake (ADLS Gen2) and Azure SQL Data Warehouse, creating clean, governed data for operational and analytical needs.

Enhanced Security & CI/CD

Integrated robust security measures (VNet, Firewall, RBAC) and automated deployments with Jenkins for continuous integration and delivery.

Actionable Insights

Developed Power BI dashboards for critical business metrics, including real-time fraud detection alerts and a holistic Customer 360 profile.

Faster Fraud Detection

Real-time alerts significantly reduce financial risk.

Accurate Customer Insights

Comprehensive 360-degree views drive personalized services.

Scalable & Secure

Enterprise-ready foundation for future growth.

Reduced Deployment Time

CI/CD automation streamlines development cycles.

Made with GAMMA