IoT Data Analytics Architecture – Portfolio Case Study

Project Overview

Designed and documented a scalable, secure, and cost-effective IoT data processing architecture on AWS for a **transport and payments solutions use case**.

The solution ingests real-time data from **point-of-sale (POS) devices and ferry terminals**, processes and transforms the data, and provides actionable insights for executive decision-making, operations optimisation, and machine learning analytics.

Problem Statement

The organisation needed to:

- Capture real-time data streams from IoT POS and ferry devices.
- Store raw data for compliance and ML training.
- Transform data into analytics-ready formats.
- Provide dashboards to operations, finance, and executive teams for quick data-driven decisions.
- Maintain a scalable, highly available, and secure architecture while optimising cost.

Tools & Technologies

- **AWS API Gateway** – Secure ingestion of POST requests from IoT devices.
- **AWS Lambda** – Serverless data processing and routing logic.
- **AWS Glue & Glue Crawler** – Data transformation and schema cataloguing.
- **Amazon S3** – Storage for raw and transformed data.
- **Amazon DynamoDB** – Metadata and index storage for quick retrieval and tracking.
- **Amazon QuickSight** – Dashboards and business intelligence.
- **Amazon SageMaker** – Machine learning model training using historical data.
- **IAM Roles & Policies** – Fine-grained access controls and security.

⚙ Solution Architecture

1. **Data Ingestion**
   IoT devices send POST requests ➔ API Gateway ➔ triggers Lambda.

2. **Processing & Storage**
   Lambda stores raw data in S3 raw bucket, writes metadata to DynamoDB, and triggers Glue jobs.

3. **Data Transformation**
   Glue Jobs process data ➔ store cleaned/transformed output in S3 transformed bucket ➔ Glue Crawler updates schema in Data Catalog.

4. **Analytics & Machine Learning**
   QuickSight connects to DynamoDB and transformed S3 data for dashboards.
   SageMaker accesses raw data in S3 for ML training, referencing DynamoDB for dataset tracking.

5. **Security & Scalability**
   IAM roles with least privilege for Lambda, Glue, QuickSight, and SageMaker.
   Serverless and multi-AZ services ensure high availability and cost efficiency.

Challenges Faced

- Designing for high throughput ingestion with low latency.
- Managing schema evolution and continuous cataloguing.
- Ensuring secure access controls across multiple services.
- Architecting for cost optimisation while maintaining performance and scalability.

Solutions Implemented

- Used **AWS Glue Crawler** to dynamically catalog evolving datasets.
- Designed Lambda functions with efficient error handling and retry logic for ingestion stability.
- Integrated **DynamoDB** as a lightweight metadata index for rapid dataset tracking and referencing.
- Configured **IAM roles and policies** based on least privilege for security compliance.
- Proposed **serverless-first architecture** (API Gateway + Lambda + Glue) to eliminate idle resource costs.

□ Impact & Results

- Enabled real-time data availability for executive and operations dashboards.
- Reduced operational complexity through automation with Glue and Lambda.
- Improved data governance via Glue Data Catalog and DynamoDB metadata indexing.
- Laid the foundation for advanced machine learning use cases with SageMaker.
- Designed with multi-zonal high availability, ensuring zero downtime for critical ingestion pipelines.

□□□ My Role

- Architected and documented the end-to-end solution.
- Designed IAM policies, data flows, and security considerations.
- Proposed implementation patterns for Lambda and Glue integration.
- Provided scalability and cost optimisation recommendations.
- Developed detailed technical documentation for stakeholder presentations.

□ Key Takeaways

□ Serverless architectures simplify scaling and reduce costs.
□ DynamoDB is effective for lightweight metadata and indexing.
□ Glue Crawlers automate schema management, enabling seamless analytics.
□ Clear IAM role boundaries are critical for security and audit compliance.

□ Cost Considerations

This architecture is **documented without live AWS resource deployment** in this repository to avoid incurring costs. It serves as a detailed reference and portfolio showcase for implementation in production environments.
