## Overview

This architecture supports an Early Warning System for food security monitoring by ingesting and processing real-time and batch data (e.g., satellite imagery, economic indicators, conflict data, weather forecasts). The design emphasizes decoupled, event-driven processing, high availability, and secure multi-region deployment, with flexibility to integrate additional data sources over time.

## System Architecture Diagram

A diagram of a network

AI-generated content may be incorrect.

## Architecture Components

### 1. Data Ingestion Layer

**Real-Time Data Ingestion:**

* **AWS Kinesis Data Streams:**  
  Captures high-velocity streams (e.g., weather updates, conflict alerts) in real time. Data records are continuously pushed into Kinesis for immediate downstream processing.
* **Event Decoupling (EventBridge/SNS/SQS):**  
  Serves as an intermediary to decouple data producers from processing components. Events from Kinesis or S3 batch uploads are published to EventBridge, which routes them to the appropriate processing service.

**Batch Data Ingestion:**

* **Amazon S3:**  
  Acts as a durable repository for bulk data such as daily economic indicators or historical datasets. This raw data is later processed in batch jobs.

### 2. Data Processing Layer

**Real-Time Processing:**

* **AWS Lambda:**  
  Processes incoming real-time data with low latency. Triggered by events from EventBridge, Lambda functions perform data cleaning, transformation, and compute composite indicators (e.g., cfii), generating alerts instantly when thresholds (e.g., cfii > 1) are exceeded.

**Batch Processing:**

* **AWS Glue:**  
  Runs scheduled ETL jobs on S3 data, aggregating historical data, enhancing data quality, and producing comprehensive analytical reports essential for long-term trend analysis.

**Containerized Workloads for Complex Analytics:**

* **Amazon ECS/EKS:**  
  Supports heavier, stateful processing tasks that exceed Lambda’s capabilities. This enables deep analytics and machine learning inference, providing the necessary compute flexibility and control.

### 3. Data Storage & Caching Layer

**Structured Data Storage:**

* **Amazon RDS/Aurora:**  
  Stores relational data (processed records, metadata, aggregated metrics) and supports complex queries for reporting. Aurora ensures high availability and robust performance.

**NoSQL Storage for Rapid Access:**

* **Amazon DynamoDB:**  
  Offers low-latency access to frequently queried data like alert statuses and time-series metrics, ideal for unpredictable workloads.

**Long-Term and Unstructured Data Storage:**

* **Amazon S3:**  
  Archives both raw and processed datasets, supporting historical analyses and compliance requirements.

**Caching:**

* **Amazon ElastiCache:**  
  Caches frequently accessed data to reduce database load and improve API response times, especially for repetitive queries.

### 4. API & Frontend Layer

**API Exposure:**

* **AWS API Gateway:**  
  Provides a secure, RESTful interface to backend services. It handles request routing, throttling, and authentication while exposing endpoints (e.g., /alerts, /data/{country}) for client applications.

**User Dashboard:**

* **Frontend Hosting (S3 + CloudFront or AWS Amplify):**  
  Delivers an interactive dashboard for food security analysts. Users can select specific countries (e.g., Yemen, Syria) to view time-series data, alerts, and statistics. CloudFront ensures low latency globally, while Amplify facilitates continuous integration and deployment.

### 5. Security, Observability, and Deployment

**Security:**

* **AWS IAM & KMS:**  
  Enforces fine-grained access control and data encryption. All services operate with IAM roles, and data at rest is encrypted with KMS-managed keys. Deployment within a secured VPC adds network isolation.
* **API Security:**  
  API Gateway incorporates authentication (e.g., JWT tokens) and rate limiting to mitigate abuse.

**Observability:**

* **Amazon CloudWatch & AWS X-Ray:**  
  Monitor system performance, log activity, and trace requests across the architecture. These tools, alongside potential third-party integrations, provide real-time insights and troubleshooting capabilities.

**Deployment Strategy:**

* **Infrastructure as Code (CloudFormation/Terraform):**  
  Automates infrastructure setup and documentation, ensuring consistency and simplifying updates.
* **CI/CD Pipelines (AWS CodePipeline/CodeBuild):**  
  Streamline development, testing, and deployment processes, reducing downtime and allowing rapid iterations.
* **Multi-Region Deployment:**  
  Enhances disaster recovery and global performance by supporting regional redundancy, thereby reducing latency for international users.