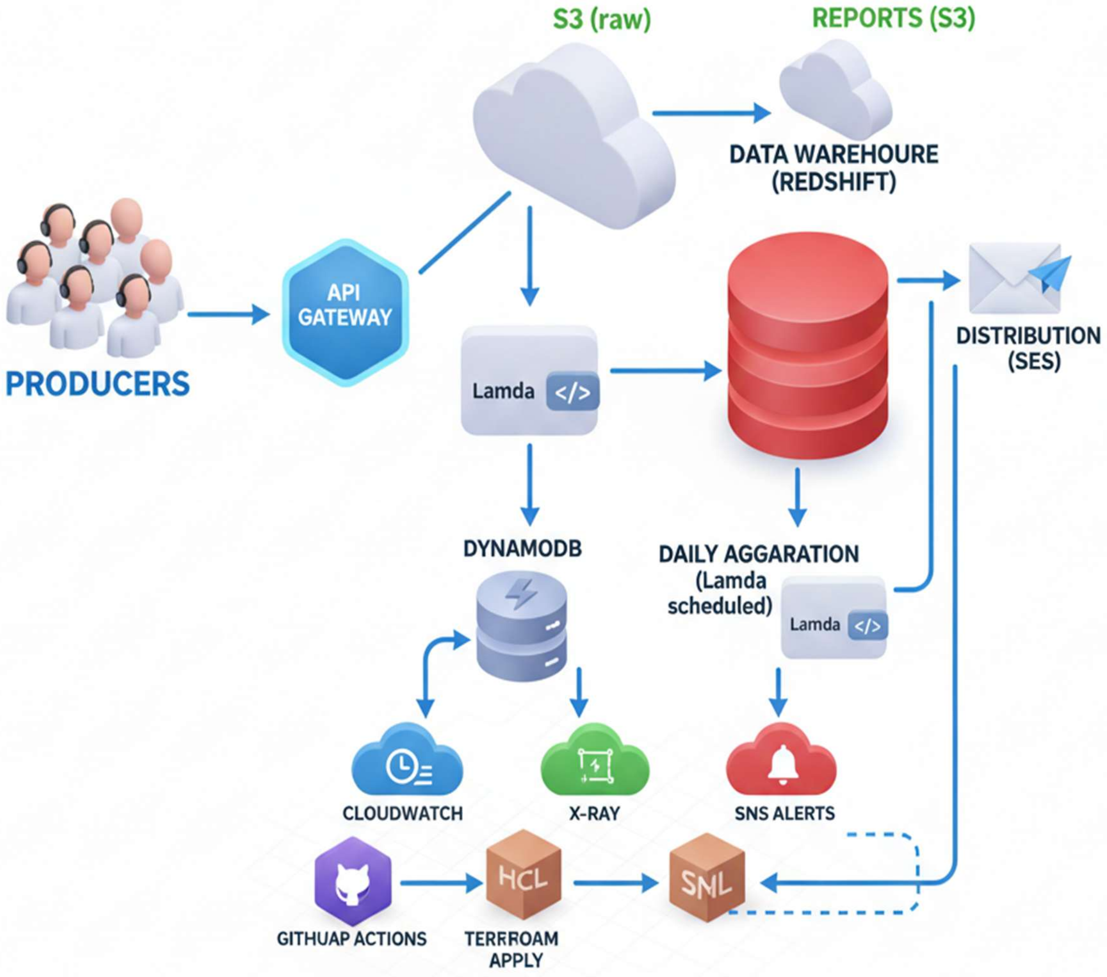
Architecture & Justification

Choose either **AWS** or **Azure**. Below is a robust, production-ready AWS example (can be mapped to Azure equivalents).

1. High-level architecture (AWS)



2. Justification of design choices

- For data producers, API Gateway offers a safe entry point that handles request validation and scaling on its own.

- Lightweight ETL (Extract, Transform, Load) processes are performed by AWS Lambda, which transforms raw event data before storing it in DynamoDB and S3 for analytics and persistence.

- High-performance querying and reporting are made possible by Amazon Redshift's function as a consolidated data warehouse.

- Email-based delivery is made simpler by Amazon SES, which manages automatic report distribution.

- SNS Alerts (real-time notifications), AWS X-Ray (tracing), and CloudWatch (logs) all guarantee observability.

- In addition to guaranteeing dependability, real-time responsiveness, and simple interaction with downstream reporting tools, this approach reduces the amount of human management.

3. Explanation of automation, deployment, and reporting flow

- Terraforms Infrastructure as Code (IaC) enables automation by uniformly provisioning Redshift, API Gateway, Lambda, and DynamoDB resources across environments.

- GitHub Actions streamlines CI/CD processes by automatically pushing updates, running Terraform plan/apply for deployments, and running Lambda tests.

- Producers submit events through the API Gateway to start the data flow. This data is processed by lambda functions and stored in DynamoDB and S3 (raw).

- Daily data is compiled by a scheduled Lambda and loaded into Redshift for analysis. Aggregated reports from Redshift are sent via SES (email automation) after being saved back to S3.

4. Brief description of fault-tolerance and scalability considerations

- Fault-tolerance is achieved through Lambda’s built-in retries, error handling, and Dead-Letter Queues (DLQ) for failed events.

- CloudWatch and SNS alerts notify engineers of failures, enabling proactive response and recovery.

- Scalability is handled automatically by AWS managed services—Lambda scales concurrently based on event volume, while Redshift and DynamoDB handle large-scale data workloads elastically.

- Data redundancy in S3 ensures durability, and Terraform automation ensures quick recovery or replication across regions if needed.