🏗 Final Project Roadmap

Step 1 – Networking (`networking.yaml`)

- Build **VPCs** (`prod-app`, `prod-data`)
- Create **Public & Private Subnets** across AZs
- Deploy **Internet Gateway + NAT Gateway**
- Add **VPC Endpoints** (S3, DynamoDB, Logs, STS, KMS, Secrets)
- Configure **Security Groups** for API Gateway, Lambda, EC2

Step 2 – Storage Foundations (`data-lake.yaml`)

- Create **S3 Raw Bucket** (gzip JSON)
- Create **S3 Processed Bucket** (Parquet partitions)
- Enable **Versioning + Encryption (KMS CMK)**
- Add **Lifecycle Policies** → IA/Glacier
- Configure **Cross-Region Replication (CRR)**

Step 3 – Ingestion & Streaming (`streaming.yaml`)

- Deploy **API Gateway (REST/HTTP)** + Auth (Cognito/WAF)
- Set up **EventBridge Scheduler** (API polling)
- Create **Kinesis Data Stream** (partition by symbol/city)
- Add **SQS Dead-Letter Queues**

Step 4 – Processing & Enrichment (`compute.yaml`)

- **Lambda Validator/Enricher** → writes to DynamoDB + S3
- **Lambda Aggregator** → calculates rolling averages
- Configure **DLQs + Retry Policies**

Step 5 – Datastores (`datastores.yaml`)

- Create **DynamoDB Global Table: MetricsTable**
- Create **DynamoDB Global Table: LatestTable**
- Optional: Deploy **DAX Cluster** for caching

Step 6 – Analytics Layer (`analytics.yaml`)

- Deploy **Glue Crawler** for S3 processed data
- Set up **Athena Query Results Bucket**

Step 7 – Dashboards (`dashboards.yaml`)

- Create **QuickSight Datasets** (Athena + DynamoDB)
- Build **QuickSight Dashboards** (stocks, weather)
- Add **CloudWatch Dashboards** (pipeline health)

Step 8 – Alerting & Anomaly Detection (`compute.yaml` + `security.yaml`)

- Deploy **Rules Engine Lambda** (10% stock drop trigger)
- Configure **CloudWatch Alarms** (Lambda errors, Kinesis lag, DLQ depth)
- Set up **SNS Topics**: OpsAlerts, MarketAlerts, WeatherAlerts, FinOpsAlerts

---

Step 9 – Security & Governance (`security.yaml`)

- Define **IAM Roles** (least privilege, cross-account assume-role)
- Create **KMS CMKs** for S3, DynamoDB, Secrets
- Store secrets in **Secrets Manager** (API keys, rotation)
- Enable **CloudTrail + Config Rules** (block public S3, enforce CMKs)
- Turn on **GuardDuty + Security Hub + WAF** for API Gateway

Step 10 – Resilience & DR (`networking.yaml` + `datastores.yaml`)

- Configure **S3 CRR** → secondary region
- Enable **DynamoDB Global Tables (multi-region)**
- Add **Route 53 Failover Routing**
- Configure **AWS Backup Plans** (S3 + DynamoDB)

Step 11 – CI/CD & IaC (`root.yaml` + nested stacks)

- Organize **Root Stack + Nested Stacks**
- Version control with **Git**
- Use **Parameters** (symbols, cities, thresholds)
- Define **Mappings** (region-specific ARNs)
- Reuse templates for networking, compute, analytics

Demo Script

1. Inject stock/weather data into API Gateway
2. Data flows: **API Gateway → Kinesis → Lambda**
3. Processed data stored in **DynamoDB + S3 Parquet**
4. Analytics: run **Athena queries + QuickSight refresh**
5. Trigger an alert (≥10% stock drop) → **SNS email/Slack notification**
6. Monitoring: show **CloudWatch dashboard + Budgets alerts**

🏗️ Final Project Roadmap (Stock/Weather Real-Time Analytics)

Roadmap Table

Step	File	Key Tasks	Outputs
1. Networking	`networking.yaml`	Build VPCs (`prod-app`, `prod-data`), create subnets (public/private), IGW + NAT Gateway, VPC Endpoints (S3, DynamoDB, Logs, STS, KMS, Secrets), configure Security Groups.	Isolated, secure networking environment for all workloads.
2. Storage Foundations	`data-lake.yaml`	Create S3 buckets (Raw & Processed), enable versioning & encryption (KMS), lifecycle policies, cross-region replication.	Secure data lake for raw and processed datasets.
3. Ingestion & Streaming	`streaming.yaml`	Deploy API Gateway (REST/HTTP) with Cognito/WAF auth, EventBridge Scheduler for API polling, Kinesis Data Stream for events, SQS DLQs.	Scalable event ingestion pipeline.
4. Processing & Enrichment	`compute.yaml`	Lambda Validator/Enricher (normalize, enrich data), Lambda Aggregator (rolling averages), configure DLQs.	Clean, enriched data stored in DynamoDB + S3 processed.
5. Datastores	`datastores.yaml`	Deploy DynamoDB Global Tables (`MetricsTable`, `LatestTable`), optional DAX cluster for caching.	Highly available, multi-region datastore for hot metrics.
6. Analytics Layer	`analytics.yaml`	Glue Crawler catalogs S3 data, Athena query results bucket.	Queryable datasets for analytics.
7. Dashboards	`dashboards.yaml`	QuickSight datasets (Athena + DynamoDB), build dashboards (stocks, weather), CloudWatch dashboards (pipeline health).	Visual insights for users and operators.
8. Alerting & Anomaly Detection	`compute.yaml`, `security.yaml`	Rules Engine Lambda (detect 10% stock drop), CloudWatch Alarms (Lambda errors, Kinesis lag, DLQ depth), SNS topics (OpsAlerts, MarketAlerts, WeatherAlerts, FinOpsAlerts).	Real-time alerts for anomalies and ops issues.

Step	File	Key Tasks	Outputs
9. Security & Governance	`security.yaml`	IAM roles (least privilege, cross-account assume-role), KMS CMKs, Secrets Manager, CloudTrail, Config Rules, GuardDuty, Security Hub, WAF.	Security, compliance, and access control across the environment.
10. Resilience & DR	`networking.yaml`, `datastores.yaml`	S3 CRR, DynamoDB Global Tables (multi-region), Route 53 failover routing, AWS Backup Plans.	Disaster recovery and high availability.
11. CI/CD & IaC	`root.yaml` (parent)	Root stack calls nested stacks, Git version control, use Parameters (symbols, cities, thresholds), Mappings (region-specific ARNs).	Automated, reproducible deployments.

Demo Script

1. Inject stock/weather data into API Gateway.
2. Data flows: **API Gateway → Kinesis → Lambda**.
3. Processed data stored in **DynamoDB + S3 Parquet**.
4. Analytics: run **Athena queries + QuickSight refresh**.
5. Trigger an alert (≥10% stock drop) → **SNS email/Slack notification**.
6. Monitoring: show **CloudWatch dashboards + Budgets alerts**.

🕮 Full Explanation: What This Project Will Do

This project builds a **real-time, serverless analytics system** on AWS that ingests live stock and weather data, processes it, stores it securely, and provides dashboards and alerts.

- **Data Ingestion:** External data sources push events through API Gateway or are polled by EventBridge. Data flows into Kinesis for scalable streaming.
- **Processing:** Lambda functions validate and enrich the raw data, calculate rolling averages, and store results in DynamoDB for fast queries and S3 for long-term analytics.
- **Storage:** S3 acts as a **data lake** with raw and processed zones, while DynamoDB Global Tables provide fast, multi-region access to current metrics.
- **Analytics & Visualization:** Glue + Athena catalog and query the processed data. QuickSight dashboards provide near real-time visualizations of stock trends and weather patterns.
- **Alerting:** A Rules Engine Lambda detects anomalies (like a 10% stock drop) and sends alerts via SNS. CloudWatch alarms monitor pipeline health (Lambda errors, Kinesis lag, DLQs).
- **Security & Governance:** IAM roles enforce least privilege, KMS encrypts all sensitive data, Secrets Manager manages API keys, CloudTrail + Config track compliance, and WAF protects the API Gateway. GuardDuty and Security Hub add continuous threat detection.
- **Resilience & DR:** S3 is replicated across regions, DynamoDB uses Global Tables, Route 53 handles failover, and AWS Backup ensures recovery.
- **Automation:** All resources are deployed via **CloudFormation nested stacks**, ensuring a reproducible and version-controlled IaC approach.

👉 In short: **it’s a complete, production-like AWS system** that demonstrates ingestion, processing, analytics, monitoring, alerting, cost controls, and security — touching nearly every topic from your syllabus.