Test Project

Module Architecting 2

Cloud Computing

Submitted by:
Amazon Web Service, Independent Test Project Designer

Contents

This Test Project consists of the following documentation/files:

1. WSC2026_TP53_WellArchitect_2 (this document)
2. Architecture diagram
3. Game Assets (provided in S3)

Use of AI Tools and Models

You may not use any AI tool or service to produce, debug, or explain your work at any point. This includes external services, personal accounts, and any model self-hosted on competition infrastructure.

Overview

Today's Test Project for Apex Capital consists of two independent workstreams. Module A is a lightweight, fully serverless market data dashboard requested by the analysis team. Module B is deploying the firm's five-service investment management platform onto AWS. The two modules are delivered and scored independently; you may tackle them in any order.

CAUTION

Please use **us-east-1** region.

Module A: Market Data Dashboard

Scenario

Apex Capital's analysis team requires a web-based market data dashboard that displays daily stock quotes to analysts and portfolio managers. This dashboard is delivered as a standalone, fully serverless web application. Your task is to deploy it so that it is publicly accessible, returns correct quote data, and remains available throughout the competition.

Game Asset

This module ships with its own game asset — a separate archive (.zip) in the game assets S3 bucket. The archive contains the dashboard's frontend, market data, and a README with the detailed instructions. The summary below describes the intended architecture and requirements; follow the README in the archive for the exact resources, schema, and configuration values you must use.

s3://wsc-tp53-2026-game-asset-day2/module-a

Requirements (Summary)

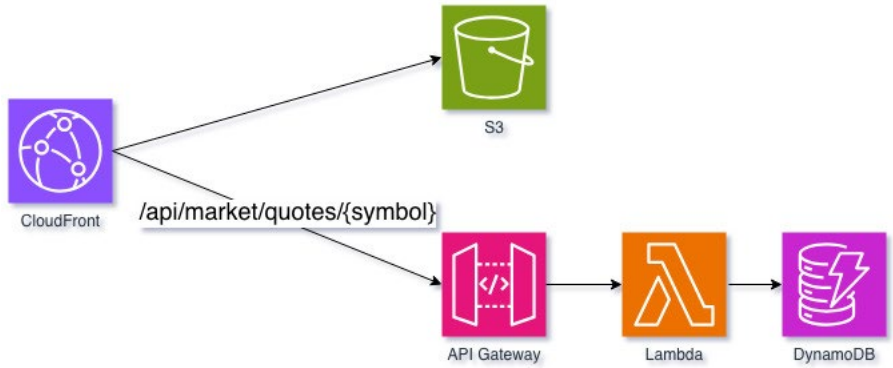
1. Data store: Load the provided daily stock-quote dataset into a DynamoDB table keyed by stock symbol, following the schema defined in the README.
2. API backend: Expose a REST API (API Gateway) backed by a Lambda function that returns the quote data for a requested symbol from DynamoDB, and returns the appropriate error response when a symbol is not found.
3. Frontend hosting: Host the provided dashboard frontend on Amazon S3 and serve it through an Amazon CloudFront distribution.

© WorldSkills International

4. CORS: Configure the API so that the dashboard, served from CloudFront, can call it successfully from the browser.

The README in the game asset provides the exact table schema, endpoint path, expected request/response format, and file names. Refer to it for all implementation details.

Architecture



Module B: Investment Management Platform

Task Description

Apex Capital has built a comprehensive investment management platform comprising five microservices — trade execution, order management, portfolio analytics, risk assessment, and accounting reconciliation. Each service is packaged as a production-ready binary with configuration documentation and health-check definitions.

As the DevOps engineer, you are responsible for:

- Deploying the entire platform onto AWS infrastructure
- Configuring networking, service discovery, data stores, and message brokers
- Establishing monitoring, logging, and alerting for operational visibility
- Designing a multi-AZ, high-availability topology with auto-scaling
- Setting up CI/CD pipelines for automated deployments
- Implementing disaster recovery strategies that minimize RTO and data loss
- Ensuring all critical data stores have backup and retention policies (minimum 8-day retention) to comply with financial regulatory requirements

Background

Apex Capital serves over two hundred institutional clients across equity, fixed-income, and derivative strategies, processing a high volume of trades daily with burst patterns that can spike tenfold within minutes.

The platform operates under the following constraints:

- Availability: Sub-second failover during volatile market conditions; zero-downtime deployments
- Network: All inter-service traffic must remain within private boundaries, with private connectivity to AWS services; network traffic must be logged for audit
- Security: All data encrypted at rest and in transit using customer-managed keys; security groups enforce least-privilege access with no overly permissive inbound rules
- Observability: Centralized logging, distributed tracing, and metric dashboards for rapid incident diagnosis

© WorldSkills International

Initial State

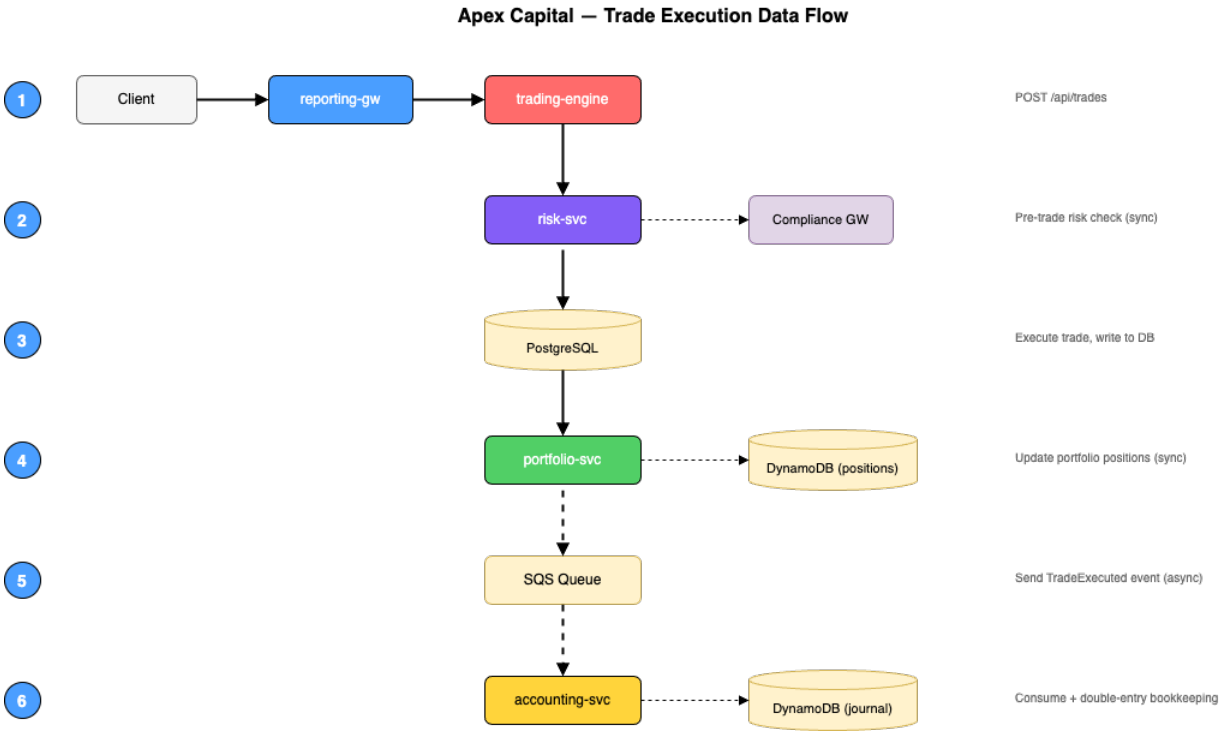
All game assets are available in a pre-provisioned S3 bucket in your account:

S3 Bucket: s3://wsc-tp53-2026-game-asset-day2/module-b
The bucket contains:

PATH	CONTENTS
bin/	Five pre-compiled binary files (Linux amd64)
docs/	Configuration documentation for each service
docs/known-issues.md	Known issues document describing identified problems
docs/architecture.md	Architecture overview document

The binaries are black-box executables. You cannot view or modify the source code. All configuration is done through environment variables.

System Data Flow



© WorldSkills International

Services

SERVICE	ROLE
reporting-gw	API gateway, rate limiting, routing
trading-engine	Trade execution, order management
portfolio-svc	Position management, portfolio valuation
accounting-svc	Journal entries, double-entry bookkeeping
risk-svc	Pre-trade risk checks, compliance

Layers

SERVICE	DATA STORE
reporting-gw	ElastiCache Redis (rate limiting)
trading-engine	RDS PostgreSQL (trades)
portfolio-svc	DynamoDB (positions) + Redis (cache)
accounting-svc	DynamoDB (journal entries)
risk-svc	DynamoDB (check history)

Tasks

1. Access the Cloud Raiser platform, and study the service documentation files in the game assets S3 bucket.
2. Deploy all five microservices on AWS using appropriate container orchestration services.
3. Configure all required infrastructure: networking, databases, caching, message queues, and event buses.
4. Establish correct service-to-service communication and verify end-to-end trade flow.
5. Identify and resolve known issues documented in the known-issues file.
6. Implement high availability: multi-AZ deployment, graceful degradation, automatic recovery, and safe deployment rollback mechanisms to prevent faulty releases from impacting production traffic.
7. Implement a CI/CD pipeline capable of automatically detecting new binary versions and deploying them to production.
8. Earn points based on system functionality as outlined in Score Events and Scoreboard.

Technical Details

1. The server applications are precompiled binaries that must remain unmodified. Using any altered or unofficial binary will result in disqualification.
2. The server applications are designed for x86_64 architecture on Linux. Consider computing services that minimize infrastructure management (e.g., AWS Fargate).
3. All service configuration is performed through environment variables. Refer to the individual service documentation for required variables.
4. The scoring system will periodically send trade requests to your API endpoint and verify end-to-end processing. Points may be awarded for areas including, but not limited to: successful deployment, end-to-end

© WorldSkills International

trade flow correctness, risk compliance enforcement, high availability, and concurrent request handling. Refer to the Scoreboard for score events.

5. Some services have known issues documented in known-issues.md. Identifying and resolving these issues is part of the challenge.
6. Data Integrity: Manual database modifications or direct data manipulation will result in point deductions. All production data stores must be protected against accidental deletion.
7. Elasticity: The system must support variable and unpredictable traffic patterns without manual capacity provisioning.
8. Auditability: All inbound traffic through the load balancer must be fully traceable for post-incident analysis.
9. Concurrency & Scaling: The scoring system will send concurrent trade requests to simulate peak trading hours. A single trading-engine instance has limited processing capacity — ensure your architecture scales horizontally to handle sustained load.
10. Continuous Deployment: During the competition, new versions of the trading-engine binary will be published to the game assets S3 bucket at unannounced intervals (no less than 20 minutes apart). Each new version overwrites the previous file in place — once a new version is published, the old version is no longer available in the bucket. Your system should detect new binary versions and deploy them automatically. Points are awarded each time the scoring system confirms a new version is running.
11. You must copy all required binary files and configuration documents from the game assets S3 bucket into a dedicated S3 bucket within your account. This bucket will serve as the centralized artifact repository for your CI/CD pipeline and deployment workflows. As this bucket holds production-critical assets, ensure appropriate versioning, lifecycle policies, encryption at rest, and access controls are in place to maintain data integrity and cost efficiency over time.
12. As new binary versions are published throughout the competition, your image repository will accumulate multiple builds. Ensure your repository remains clean, consistent, and appropriately secured to support reliable automated deployments.

Service Details

Database

1. A PostgreSQL database must be deployed for the trading-engine service. The required table schema will be output to stderr if the service starts without the correct tables. Table name: trades.
2. Three DynamoDB tables are required:

    - Positions table (for portfolio-svc): PK=portfolio_id, SK=asset_symbol, Stream enabled
    - Journal entries table (for accounting-svc): PK=entry_id, SK=trade_id, with GSIs
    - Risk checks table (for risk-svc): PK=check_id, SK=trade_id, with GSI

Caching

An ElastiCache Redis is required for:

- reporting-gw: API rate limiting
- portfolio-svc: Position query caching

Message Queue

An SQS queue is required for asynchronous communication between trading-engine and accounting-svc.

Event Bus

An EventBridge custom event bus is required for portfolio concentration alerts.

© WorldSkills International

Applications

Binary Files

The development team has provided the following pre-compiled binary files. Refer to **docs/architecture.md** in the game assets for detailed service communication flows and network topology.

Configuration

Refer to the service documentation files in the S3 bucket for required variables and startup instructions.

Preconfigured IAM Roles

Preconfigured IAM roles are available for your use:

1. game-ecs-task-execution-role
2. game-codepipeline-role
3. game-codebuild-role
4. game-lambda-role
5. game-scheduler-role

Scoring

Monitor your Score Events and Scoreboard for progress.

For Module A, the scoring system periodically accesses your CloudFront URL (which must return a successful response), queries your API with symbols from the dataset, and verifies that the returned values match the source data. Points are awarded continuously for as long as the dashboard is reachable and the API returns correct data. The scoring system will begin sending traffic between 30 and 90 minutes after the competition starts.

For Module B, the scoring system periodically submits trade requests to your platform endpoint and verifies correct end-to-end processing, high availability, and successful automated deployment of new binary versions. Points accrue continuously as these checks pass. The scoring system will begin sending traffic between 60 and 120 minutes after the competition starts.

Reference Documents

https://docs.aws.amazon.com/
https://docs.aws.amazon.com/AmazonECS/latest/developerguide/
https://docs.aws.amazon.com/elasticache/latest/dg/
https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/
https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/
https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/
https://docs.aws.amazon.com/wellarchitected/latest/framework/welcome.html
https://docs.aws.amazon.com/lambda/latest/dg/
https://docs.aws.amazon.com/apigateway/latest/developerguide/
https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/
https://docs.aws.amazon.com/AmazonS3/latest/userguide/WebsiteHosting.html

© WorldSkills International