AWS Microservices Platform Deployment

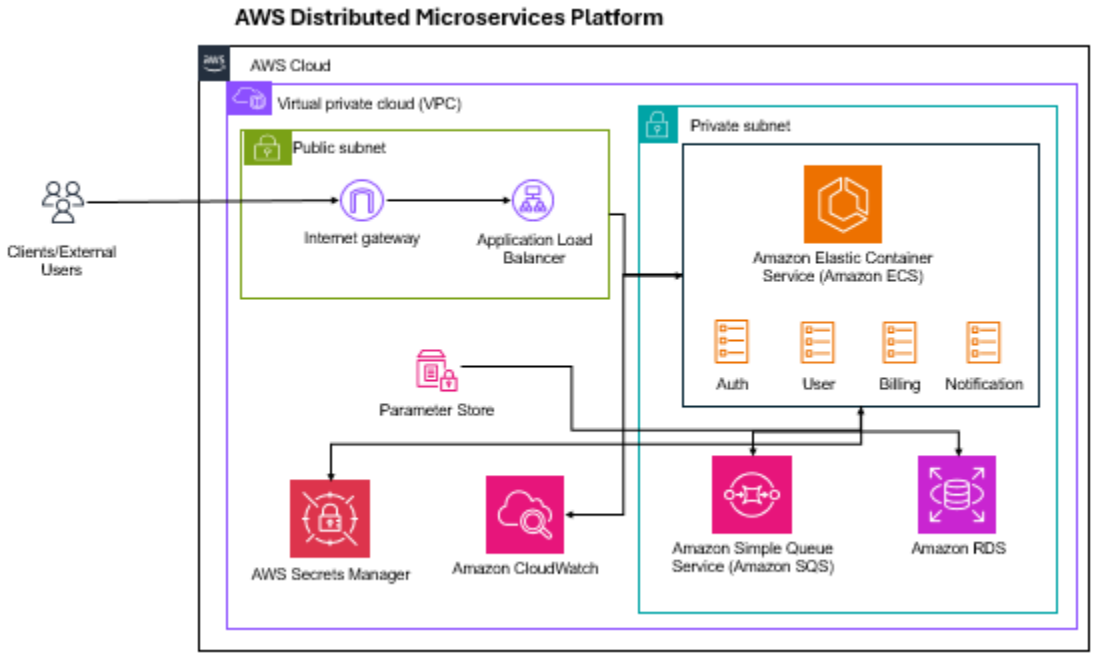
Project: Distributed Microservices Platform
Prepared by: Software Development Team
Date: December 16th, 2025

1. **Objective**

   The objective of this deliverable is to document the coordinated installation, configuration and technical validation of an AWS hosted distributed microservices platform. The task focused on deploying containerized services, integrating AWS managed dependencies, enforcing security controls and verifying systems readiness for downstream integration and testing.

2. **Target Architecture Summary**

   The platform is deployed within an AWS Virtual Private Cloud (VPC) spanning two Availability Zones to ensure fault tolerance. Public subnets host the Application Load Balancer (ALB), while private subnets host ECS Fargate tasks and database resources. All inter service communication occurs within private networks and is governed by IAM and security group policies.



Core AWS services:

- Amazon ECS (Fargate)
- Application Load Balancer
- Amazon ECR
- Amazon RDS (PostgreSQL)
- Amazon SQS
- AWS Systems Manager – Parameter Store
- AWS Secrets Manager
- Amazon CloudWatch
- IAM
- VPC Networking Components

3. **Pre-Deployment Validation**
   Prior to application deployment, the following technical validations were performed;

   a. VPC configuration was reviewed to confirm subnet segmentation, route tables, NAT gateway availability and network access controls.
   b. IAM roles were validated to ensure ECS task execution roles and task roles had least privilege access to required AWS services including CloudWatch Logs, SQS and Parameter Store.
   c. Security groups were configured to allow inbound traffic only from the ALB to ECS services with all other ingress restricted.
   d. Required AWS resources including RDS instances, SQS queues and parameter namespaces were provisioned and verified as available.

4. **Application Deployment and Configuration**
   Microservices were deployed as containerized workloads using Amazon ECS on Fargate. Each service was defined using ECS task definitions specifying container images stored in Amazon ECR, CPU and memory allocations, port mappings and health check configurations.

   Environment specific configuration values were injected at runtime using AWS Systems Manager Parameter Store and AWS Secrets Manager to avoid hardcoded

credentials.

Service deployment was coordinated through the CI/CD pipeline, which executed container builds, image pushes to ECR, ECS service updates and rolling deployments with zero downtime configurations.

Application Load Balancer listeners and target groups were configured to route traffic to ECS services based on path based routing rules.

5. **CloudFormation Template**
   microservicesplatform.yaml file attached

6. **Dependency Integration**
   The following integrations were validated during installation:

   a. Database connectivity between ECS services and Amazon RDS using private endpoints and encrypted connections.
   b. Asynchronous messaging workflows via Amazon SQS, including queue access permissions, message visibility timeouts and retry behavior.
   c. Centralized logging via CloudWatch Logs with log streams created per service and environment.
   d. CloudWatch metrics collection for ECS service CPU utilization, memory usage, task health and ALB request metrics.

7. **Post Deployment Verification**
   Post Deployment Verification included both infrastructure and application-level checks:
   a. ECS service reached desired task counts without deployment failures
   b. ALB target groups reported healthy targets across all Availability Zones
   c. Service health endpoints returned successful HTTP responses under load
   d. Inter-service communication was confirmed by publishing and consuming messages from SQS queues
   e. CloudWatch dashboards displayed real time metrics, and no critical alarms were triggered during validation

8. **Validation Checklist and Sign-Off**
    - ☒ ECS clusters and services operational
    - ☒ ALB routing and health checks passing
    - ☒ Secure configuration and secrets management validated
    - ☒ Logging and monitoring enabled and verified
    - ☒ No critical errors observed post deployment

Deployment Status: Successful
Technical Readiness: Approved for integration and QA testing