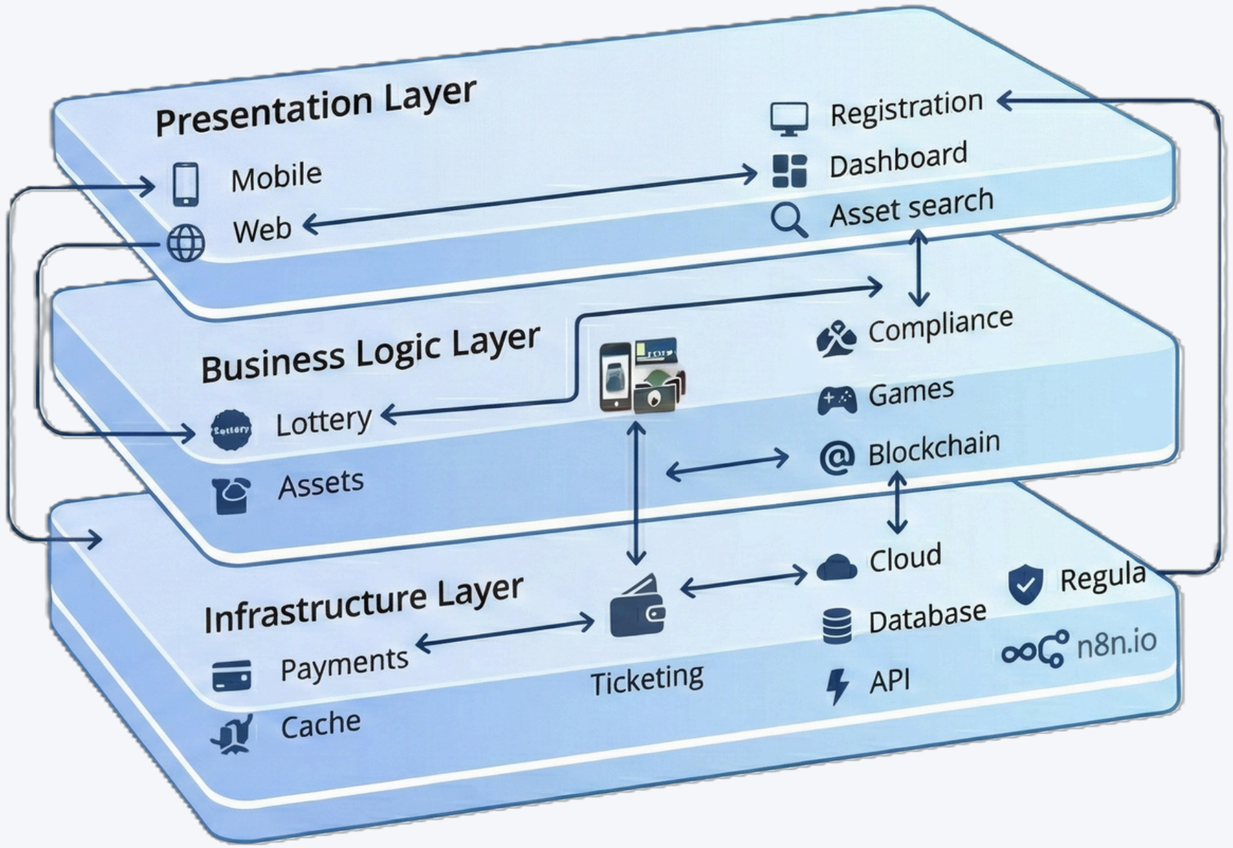
Technology Architecture

AWS-aligned platform blueprint for a regulated digital gaming/lottery ecosystem

Target state • scalable • secure • auditable



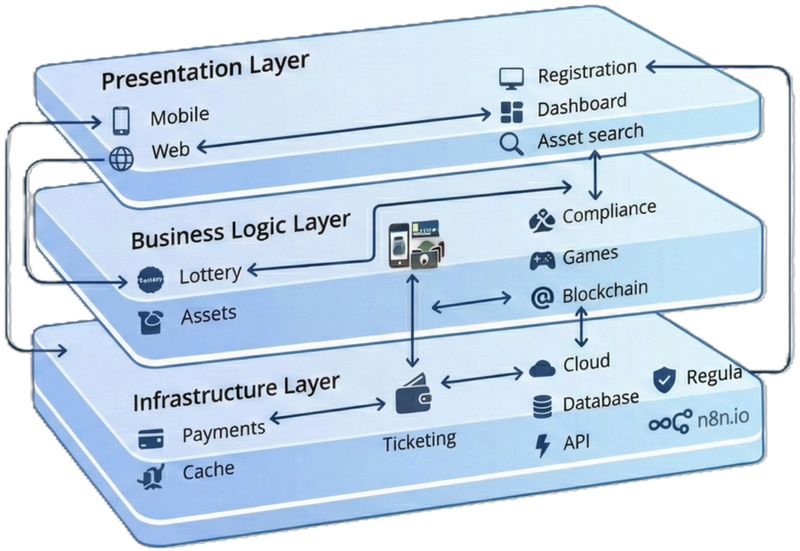
Current-state architecture inputs
Three-layer design

Current Platform Context

The AWS target design preserves the existing three-layer separation while replacing generic platform blocks with AWS-native building blocks

CURRENT STATE

Three Layer Design



Architecture details

Presentation, business logic, and infrastructure layers are a strong starting point for AWS account, networking, compute, and data segmentation

Core services

Registration & KYC
Identity verification and compliance

Ticketing Engine
Lottery ticket sales and management

Regulatory Compliance
Full audit trail and reporting

Skill Mechanics
Game design and fairness systems

Asset Acquisition
Real estate due diligence pipeline

Draw Management
Blockchain-verified lottery execution

Payment Orchestration
Multi-currency payout processing

Microservice-friendly

Each service is mapped to independently or collectively deploy

Architecture Principles for the AWS Target State

These principles drive service selection, decomposition boundaries, and non-functional requirements

DESIGN DRIVERS

ES Elastic scale

Autoscale stateless services independently to handle registration peaks, ticket surges, search spikes, and payout cycles.

SD Secure by default

Embed identity, network segmentation, secrets management, encryption, and audit controls into every layer

ED Event-driven domains

Use asynchronous messaging for ticket issuance, draw execution, payment updates, notifications, and compliance workflows

DF Data fit-for-purpose

Match each workload with the right store: transactional SQL, key-value, cache, search, archive, and immutable evidence

OR Operational resilience

Design for multi-AZ continuity, graceful degradation, replayable workflows, backup, and disaster recovery

CV Compliance visibility

Maintain traceability for user identity, payment actions, draw history, content changes, and operator interventions

Target AWS Reference Architecture

Illustrative target state showing how the current platform layers map into AWS edge, application, data, security, and operations capabilities

REFERENCE VIEW

Edge & access

Route 53 | CloudFront | AWS WAF | Cognito | API Gateway | ALB | S3 static UI | OpenSearch | CloudWatch RUM

Business services platform

Primary pattern: Amazon ECS on Fargate for stateless APIs and workers; Lambda for short-running event handlers; Cloud Map for service discovery

Registration & KYC | Ticketing Engine | Regulatory Compliance | Skill Mechanics | Asset Acquisition | Draw Mgmt | Payment Orchestration

Eventing & workflow

EventBridge + SNS/SQS fanout decouple domains. Step Functions governs KYC, payout, compliance, and document-driven workflows

Data layer

Aurora PostgreSQL for core transactions; DynamoDB for high-throughput states; ElastiCache for low-latency reads; S3 for evidence and documents

Security & governance

IAM • KMS • Secrets Manager • CloudTrail • Config • Security Hub

Observability & ops

CloudWatch • X-Ray • OpenSearch • Systems Manager • Backup

Delivery foundation

CodePipeline/GitHub Actions • ECR • IaC (CDK/Terraform)

Channel, Edge, and Access Architecture

Front-end access paths for mobile, web, registration journeys, dashboards, and asset search

USER ACCESS

Client channels

- Mobile app
- Web portal
- Operator back office
- Partner portals

E AWS edge

Route 53 resolves domains and supports failover routing. CloudFront accelerates UI delivery. AWS WAF + Shield protect public endpoints and reduce attack surface.

I Identity & API ingress

Amazon Cognito handles user sign-up, MFA, federation, and token issuance. API Gateway fronts public APIs; ALB exposes private microservices behind the edge layer

X Experience services

Registration/KYC journeys call workflow APIs, dashboard data is aggregated through domain APIs, and asset search reads from OpenSearch with cache acceleration

Recommended implementation detail

Host web assets on S3 + CloudFront; keep mobile and operator clients on the same identity plane; terminate TLS at the edge; propagate JWT claims to downstream services for policy enforcement and audit correlation

Business Logic Layer on AWS

Microservices are organized as domain APIs, workers, and orchestration workflows running on a shared but segmented platform foundation

COMPUTE

Primary runtime pattern

Amazon ECS on Fargate

One service per business capability

Independent autoscaling policies

Separate task roles and secrets

Blue/green releases via CodeDeploy

Private subnets behind internal ALBs

Domain API services

Registration & KYC

Ticketing Engine

Skill Mechanics

Asset Acquisition

Draw Management

Payment Orchestration

Compliance API

Supporting execution patterns

Lambda for light event handlers and scheduled jobs

Step Functions for long-running workflows

SQS-backed workers for retries, throttling, and partner isolation

Optional EKS path if the team already operates Kubernetes at scale

Boundary rule

Keep synchronous APIs for user-facing reads/writes; shift cross-domain side effects to EventBridge or queues so ticketing, payouts, compliance, and notifications do not block each other

Data Architecture by Workload Type

Using specialized AWS data services rather than forcing all domains into a single database pattern

DATA LAYER

Operational transaction stores

Amazon Aurora PostgreSQL

- Account records
- Ticket purchase transactions
- Draw states
- Payout ledger

Amazon DynamoDB

- High-volume session/entitlement lookups
- Idempotency keys
- Rules/feature flags

Performance & search layer

Amazon ElastiCache

- Hot dashboard counters
- Profile/session cache
- Frequently accessed game data

Amazon OpenSearch Service

- Asset search
- Operator search
- Operational analytics and trace exploration

Documents, Evidence & Archives

Amazon S3

- KYC files
- Compliance evidence
- Reports/exports
- Immutable audit packages

AWS Backup + Glacier tiering

- Long-term retention
- Policy-driven recovery

Optional QLDB / blockchain anchoring

- Tamper-evident draw evidence

Practical rule

Using domain-owned schemas and storage accounts per service boundary. Replicate read models to OpenSearch or analytics stores instead of sharing transactional tables across domains.

Security, Identity & Compliance Controls

A regulated platform needs visible controls at identity, network, data, and governance layers - not only at the application layer

CONTROLS

Identity & access

Amazon Cognito for users and operators
IAM roles per service
MFA for privileged access
AWS IAM Identity Center for admin federation

Data protection

KMS-managed encryption for S3, RDS, EBS, and secrets
Secrets Manager for partner credentials
TLS everywhere, including internal service paths

Governance & audit

CloudTrail, Config, Audit Manager, and Security Hub provide evidence of change, policy drift, control status, and security findings

Threat detection

GuardDuty, Inspector, Detective, WAF logs, and centralized SIEM/search provide actionable detection across the platform

Compliance workflow controls

Step Functions can enforce approval checkpoints, evidence capture, escalation, and SLA timers for KYC review, suspicious activity investigation, payout exceptions, and operator override workflows

Network segmentation

Use separate AWS accounts for production, non-production, shared services, and security tooling.
Within each VPC, keep public exposure limited to edge components and place services/data in private subnets with VPC endpoints where practical

External Integration Architecture

Payments, compliance, and blockchain dependencies should be isolated so partner issues do not directly destabilize customer-facing services

INTEGRATIONS

Payments

Payouts, refunds, wallet, settlement

Compliance

KYC/AML, sanctions, case mgmt

Draw & fairness

Draw notarization, evidence, proof

AWS integration hub

EventBridge
SQS / SNS
Step Functions
Lambda connectors
Secrets Manager
API destinations / private integration adapters

Purpose:
- retries
- throttling
- partner abstraction
- replayability
- audit visibility

PSP / banking

External payment processors and banking rails

KYC/AML vendors

Identity verification, AML scoring, watchlists

Blockchain service

Managed node or external anchoring provider

Implementation note

Run n8n in ECS with the same secrets, logging, and network controls as any other integration workload

AWS Account, Network, and Environment Topology

Separate blast radius by account and keep public ingress narrow; keep services and data private by default

TOPOLOGY

Shared edge

Route 53
CloudFront
WAF/Shield

Production account

Public subnets: ALB only
Private app subnets: ECS/Lambda
Private data subnets: Aurora/Redis
Separate security groups per service

Non-production account

Dev/QA/UAT
Lower-cost scale profile
Synthetic test data
Same IaC modules as production

Security/logging account

Centralized CloudTrail
Security Hub/GuardDuty
Immutable log archive
Cross-account dashboards

Network practices

Using VPC endpoints for S3, Secrets Manager, CloudWatch, and other managed services when appropriate
Favoring private service-to-service traffic
Restricting outbound access via controlled egress and NAT boundaries
Using Transit Gateway only if multiple VPCs or shared services justify it

Availability & DR

Multi-AZ for core data stores and runtimes
Cross-region backup/replication for evidence and critical databases
Warm standby is a pragmatic phase-2 target for regulated operations that need regional continuity
Runbooks and failover tests belong in the operating model, not only on paper

DevSecOps, Observability & Resilience

The platform should be operable as a product: deployable, measurable, reversible, and recoverable

OPERATE

Delivery pipeline

Source control

→ build/test

→ container scan

→ image publish to ECR

→ deploy with blue/green or canary

→ automated rollback on alarms

Using:

CodePipeline/CodeBuild/CodeDeploy or

GitHub Actions + AWS deploy actions

Observability stack

CloudWatch metrics, logs, alarms

X-Ray tracing for request paths

OpenSearch dashboards for

operations/security

Synthetics and RUM for customer

journeys

Central correlation IDs across ticket,

draw, payout, and compliance flows

Operational controls

Systems Manager runbooks

Patch/session control

Feature flags and AppConfig

Automated backup policies

Queue DLQs and replay utilities

Resilience focus

Graceful degradation if partners fail

Circuit breakers on external calls

Idempotent commands

Replayable events

Chaos / failover drills in non-

production

Microservice-to-AWS Mapping

Illustrative landing zone for each current platform capability. Final service selection can be tuned during detailed solution design

MAPPING

Current capability	Primary AWS services	Implementation note
Registration & KYC	Cognito, API Gateway, ECS/Lambda, Step Functions, S3, OpenSearch	Identity onboarding, document handling, review workflow, and searchable evidence
Ticketing Engine	ECS Fargate, Aurora PostgreSQL, DynamoDB, EventBridge, SQS, ElastiCache	Transaction integrity plus burst handling and asynchronous downstream events
Regulatory Compliance	Step Functions, CloudTrail, Config, Audit Manager, S3, OpenSearch	Case management, policy checks, evidence retention, and operator auditability
Skill Mechanics	ECS/Lambda, DynamoDB, ElastiCache, AppConfig	Low-latency rules, session state, fairness/configuration controls
Asset Acquisition	ECS, Aurora, S3, OpenSearch, Step Functions	Document-centric due diligence and searchable asset metadata
Draw Management	ECS/Lambda, Aurora, QLDB or blockchain integration, EventBridge	Deterministic draw control with tamper-evident evidence trails
Payment Orchestration	ECS/Lambda, Step Functions, SQS, Aurora, Secrets Manager	Retry-safe payout / settlement flows isolated from partner variability

Sizing decision

By having strong Kubernetes operations, the ECS-based boxes above can be replaced with EKS while keeping the same edge, data, security, and eventing architecture

Delivery Roadmap

A phased program reduces migration risk while establishing the platform foundation early

ROADMAP

Phase 1

Landing zone
IAM/security baseline
Network + CI/CD
Observability foundation

Phase 2

Identity + edge
Web hosting
API ingress
Registration/KYC workflows

Phase 3

Ticketing/draw domains
Payments integration
Search + cache
Partner decoupling

Phase 4

DR uplift
Advanced analytics
Automation hardening
Cost optimization

Execution guidance plan

Start with the platform foundation before migrating domain services. Move customer-facing identity and edge paths early, then bring over the most operationally critical domains (ticketing, draw, payout, compliance) with full observability and rollback support. Keep data migration, evidence retention, and regulatory sign-off as explicit workstreams rather than implied technical tasks.