# Hoops Tracker deployment plan

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Component** | **Configuration** | **Auto-Scaling Parameters** | **Sticky Sessions** | **Backup Node** | **Load Handling** | **Cost (Hourly)** | **Description** |
| **VPC** | us-east-1, 2 private subnets (us-east-1a, us-east-1b), 2 public subnets | N/A | N/A | N/A | Supports ~50,200 messages/second, ~50,000–60,000 GETs/second | ~$0.00 | Secure network, restricts access (ports 443, 8080, 6379, 5432). |
| **ECS Fargate (Stats-API Service)** | **Spring Boot REST**, 1 vCPU, ~2 GB/task, min 3 tasks, port 8080, rate limiting (100 calls/second/client), **access to TimescaleDB** | CPU ~70%, memory ~80%, ~10,000 connections/task, Redis latency ~1.0 ms P99, DB latency ~1.0 ms P99, scale 1 task | Enabled (ALB target group) | 1 task (min 3 tasks) | 50,000–60,000 GETs/second (20,000/task), **read/write TimescaleDB** | ~$0.14 (3 tasks), ~$0.23 (5 tasks) | Handles REST API calls (/players/{id}/stats), **direct DB & Redis accerss**, auto-scales, sticky sessions for affinity. |
| **ECS Fargate (Ingest Service)** | **Spring Boot STOMP**, 1 vCPU, ~2 GB/task, min 3 tasks, port 8080, rate limiting (100 messages/second/client) | CPU ~70%, memory ~80%, ~10,000 connections/task, Redis latency ~1.0 ms P99, DB latency ~1.0 ms P99, scale 1 task | Enabled (ALB target group) | 1 task (min 3 tasks) | 50,200 messages/second (20,000/task), updates Redis/TimescaleDB | ~$0.14 (3 tasks), ~$0.23 (5 tasks) | Processes WebSocket messages (/topic/live),  **Redis access,**  rate limits clients, auto-scales, sticky sessions for affinity. |
| **ECS Fargate (DB-Redis Service)** | **Spring Boot microservice**, 1 vCPU, ~2 GB/task, min 2 tasks, **only connects to Redis & TimescaleDB**, no API exposure | CPU ~50%, memory ~70%, ~5,000 connections/task | N/A | 1 task (min 2 tasks) | ~10,000–12,000 queries/second | ~$0.12 (2 tasks), ~$0.18 (3 tasks) | Dedicated service for **direct DB & Redis queries** (e.g., batch updates, analytics). |
| **ALB** | Internet-facing, HTTPS (TLS 1.3, wss://), port 443, health checks (/health), AWS WAF rate limiting (~100 requests/messages/second/client) | Auto-scales nodes (large to 4xlarge) | Duration-based (~1 hour) | N/A | ~60 Mbps WebSocket, ~432–444 Mbps REST | ~$0.04 | Routes REST (/api/) to stats-api, WebSocket (/ws/) to ingest service, rate limits clients, ensures session continuity. |
| **ElastiCache Redis** | 2-node master-slave (r7g.medium, ~1.5 GB), Redis 7.1, 3 Sentinels | N/A (2 nodes sufficient) | N/A | 1 replica (HA, Sentinel failover ~10–30s) | 50,440–60,440 ops/second (~0.5 ms read/DEL) | ~$0.30 | Caches ~2400 entries, handles increments, sets, deletes. |
| **RDS/TimescaleDB** | db.m6g.large (~2 vCPUs, ~8 GB, ~12 GB gp3 storage, ~6 GB primary, ~6 GB replica), primary + Multi-AZ replica, port 5432 | N/A (scales vertically if needed) | N/A | 1 replica (Multi-AZ, failover ~1–2 min) | ~50,200 ops/second, ~10,000–12,240 queries/second | ~$0.53 | Stores ~3.7 GB (2 sessions), supports time-series, **direct DB access enabled** for REST API, Ingest Service & DB-Redis Service. |
| **Route 53** | DNS to ALB (stats.nba.example.com) | N/A | N/A | N/A | Low-latency routing | ~$0.0004 | Directs traffic to ALB, ensures availability. |
| **CloudWatch** | Metrics (CPU, memory, connections, Redis/DB latency), alarms for auto-scaling | Triggers ECS scaling (CPU > 70%, memory > 80%, connections > 10,000, Redis/DB latency > 1.0 ms) | N/A | N/A | Monitors ~50,200 messages/second, ~50,440–60,440 ops/second | ~$0.01 | Tracks performance, enables dynamic scaling. |

### **Updated Cost & Scalability Estimates**

* **Base Cost**: ~$1.39/hour (with three ECS services)
* **Scaled Cost**: ~$1.57/hour (if max ECS scaling reached)
* **Monthly Cost**: ~$1000–1200/month depending on traffic
* **Performance Boost**: Handles **direct DB & Redis access for DB-Redis Service**, reducing API overhead.
* **Security Considerations**: Ensure DB authentication is restricted with TLS, IAM roles.

This setup ensures **efficient DB & Redis access** while keeping **API exposure minimal** for the third service. Let me know if you need further refinements! 🚀