Contents

On-Call Runbook

Overview

This runbook provides guidance for on-call engineers supporting the RADIANT platform.

On-Call Responsibilities

1. **Monitor** alerts and dashboards
2. **Respond** to incidents within SLA
3. **Escalate** when needed
4. **Document** all actions taken
5. **Handoff** to next on-call

Shift Schedule

- Primary on-call: 24/7 coverage
- Secondary on-call: Backup for escalation
- Shifts rotate weekly (Monday 9am)

Alert Sources

Source	Type	Priority
PagerDuty	Alerts	High
Slack #alerts	Warnings	Medium
Email	Informational	Low

First Response

1. Acknowledge Alert

```
# Via PagerDuty app or CLI
pd incident acknowledge <incident-id>
```

2. Initial Assessment (5 minutes)

- □ What is the alert?
- □ What service is affected?
- □ What's the impact?
- □ When did it start?

3. Check Dashboards

```
# Open CloudWatch dashboard
open "https://console.aws.amazon.com/cloudwatch/home?region=us-east-1#dashboards:name=radiant-p

# Or use AWS CLI
aws cloudwatch get-metric-data \
  --metric-data-queries file://quick-metrics.json \
  --start-time $(date -d '1 hour ago' -Iseconds) \
  --end-time $(date -Iseconds)
```

4. Check Service Health

```
# API health
curl -s https://api.radiant.example.com/v2/health | jq

# Dashboard health
curl -s -o /dev/null -w "%{http_code}" https://admin.radiant.example.com

# Database (via admin API)
curl -s -H "Authorization: Bearer $TOKEN" \
  https://api.radiant.example.com/v2/admin/health/database | jq
```

Common Alerts

API Error Rate High

Alert: `radiant-production-api-5xx-errors`

Quick Check:

```
# Recent Lambda errors
aws logs filter-log-events \
  --log-group-name /aws/lambda/radiant-production-router \
  --filter-pattern "ERROR" \
  --start-time $(date -d '30 minutes ago' +%s000) \
  --limit 20
```

Actions: 1. Check if it's a single endpoint or widespread 2. Check recent deployments 3. Check database connectivity 4. Escalate if > 5 minutes

API Latency High

Alert: `radiant-production-api-latency`

Quick Check:

```
# Lambda duration
aws cloudwatch get-metric-statistics \
  --namespace AWS/Lambda \
  --metric-name Duration \
  --dimensions Name=FunctionName,Value=radiant-production-router \
  --statistics p99 \
  --period 60 \
  --start-time $(date -d '30 minutes ago' -Iseconds) \
  --end-time $(date -Iseconds)
```

Actions: 1. Check if cold starts are high 2. Check database query times 3. Check AI provider latency 4. Consider scaling up

Database CPU High

Alert: `radiant-production-db-cpu`

Quick Check:

```
# DB metrics
aws cloudwatch get-metric-statistics \
  --namespace AWS/RDS \
  --metric-name CPUUtilization \
  --dimensions Name=DBClusterIdentifier,Value=radiant-production \
  --statistics Average \
  --period 60 \
  --start-time $(date -d '30 minutes ago' -Iseconds) \
  --end-time $(date -Iseconds)
```

Actions: 1. Check for long-running queries 2. Check connection count 3. Consider read replica 4. Escalate to database team

Lambda Throttling

Alert: Lambda concurrent execution limit

Quick Check:

```
aws cloudwatch get-metric-statistics \
  --namespace AWS/Lambda \
  --metric-name Throttles \
  --dimensions Name=FunctionName,Value=radiant-production-router \
  --statistics Sum \
  --period 60 \
  --start-time $(date -d '30 minutes ago' -Iseconds) \
  --end-time $(date -Iseconds)
```

Actions: 1. Request concurrency limit increase 2. Check for retry storms 3. Consider provisioned concurrency

Escalation

When to Escalate

- SEV1/SEV2 incidents
- Unable to resolve within 30 minutes
- Security incidents
- Data loss potential
- Need additional expertise

Escalation Path

1. **Secondary On-Call** - First escalation
2. **Engineering Lead** - Major incidents
3. **Security Team** - Security issues
4. **Executive** - Business-critical

How to Escalate

```
# Via PagerDuty
pd incident escalate <incident-id> --escalation-policy "Engineering Lead"

# Via Slack
/page @engineering-lead SEV2 - API errors > 5%
```

Useful Commands

Log Analysis

```
# Search logs for errors
aws logs filter-log-events \
  --log-group-name /aws/lambda/radiant-production-router \
  --filter-pattern "ERROR" \
  --start-time $(date -d '1 hour ago' +%s000)

# Search for specific request
aws logs filter-log-events \
  --log-group-name /aws/lambda/radiant-production-router \
  --filter-pattern '"requestId":"abc123"'
```

Quick Metrics

```
# API request count
aws cloudwatch get-metric-statistics \
  --namespace AWS/ApiGateway \
  --metric-name Count \
  --dimensions Name=ApiName,Value=radiant-production-api \
  --statistics Sum \
  --period 300 \
  --start-time $(date -d '1 hour ago' -Iseconds) \
  --end-time $(date -Iseconds)
```

Service Status

```
# All Lambda functions
aws lambda list-functions \
  --query "Functions[?starts_with(FunctionName, 'radiant-production')].[FunctionName,LastModifi
  --output table

# All RDS clusters
aws rds describe-db-clusters \
  --query "DBClusters[?starts_with(DBClusterIdentifier, 'radiant')].[DBClusterIdentifier,Status
  --output table
```

Handoff Procedure

End of Shift

1. **Document** any ongoing issues
2. **Update** incident tickets
3. **Brief** incoming on-call
4. **Transfer** PagerDuty responsibility

Handoff Template

```
## On-Call Handoff

**Date:** YYYY-MM-DD
**Outgoing:** @name
**Incoming:** @name

### Active Incidents
- None / [Incident links]

### Recent Issues
- [Brief description of any issues in past 24h]
```

```
### Upcoming Changes
- [Any scheduled deployments or maintenance]

### Notes
- [Anything the incoming on-call should know]
```

Resources

- Incident Response Runbook
- Deployment Runbook
- CloudWatch Dashboard
- Admin Dashboard
- PagerDuty