DR Runbook

Drop — Disaster Recovery Runbook

Infrastructure Overview

Production Environment

- **Service:** AWS App Runner
- **Region:** eu-west-1 (Ireland)
- **Service ARN:** `arn:aws:apprunner:eu-west-1:324480209768:service/drop-web/8e45b0d335304487a1880f4e32d6aeec`
- **Service URL:** https://9ef3szvvsb.eu-west-1.awsapprunner.com
- **ECR Repository:** `324480209768.dkr.ecr.eu-west-1.amazonaws.com/drop-web`

Database

- **RDS Instance:** drop-db
- **Endpoint:** `drop-db.czu2qe4quy4v.eu-west-1.rds.amazonaws.com:5432`
- **Database Name:** dropapp
- **Username:** dropuser
- **Backup Strategy:** Automated snapshots, 7-day retention
- **Backup Window:** 23:24-23:54 UTC daily

Staging Environment

- **Platform:** Fly.io
- **App Name:** drop-staging
- **Region:** arn (Stockholm)
- **Database:** PostgreSQL 16 (RDS, eu-north-1, or Docker in CI)

Domain

- **Production:** getdrop.no (future)
- **Current:** App Runner subdomain

Backup Strategy

RDS PostgreSQL (Production)

- **Automated Snapshots:** Daily at 23:24 UTC
- **Retention Period:** 7 days
- **Point-in-Time Recovery:** Enabled (5-minute granularity)
- **Manual Snapshots:** Created before major changes
- **Storage:** Same region (eu-west-1)

Staging PostgreSQL (RDS)

- **Automated Snapshots:** Daily, 7-day retention (same config as production)
- **Backup Method:** Manual export via `flyctl ssh console` and `pg_dump` (PostgreSQL 16 — `sqlite3` no longer applies; see ADR-014)
- **Recommended:** Export before major changes

Recovery Procedures

Scenario 1: App Runner Service Down

Symptoms

- Service health checks failing
- 5xx errors from App Runner URL
- CloudWatch alarms triggered

Investigation Steps

```
# 1. Check service status
aws apprunner describe-service \
  --service-arn arn:aws:apprunner:eu-west-1:324480209768:service/drop-
web/8e45b0d335304487a1880f4e32d6aeec \
  --region eu-west-1
```

```
# 2. View recent logs (last 10 minutes)
aws logs tail /aws/apprunner/drop-web/8e45b0d335304487a1880f4e32d6aeec/application \
  --follow \
  --since 10m \
  --region eu-west-1

# 3. Check deployment history
aws apprunner list-operations \
  --service-arn arn:aws:apprunner:eu-west-1:324480209768:service/drop-
web/8e45b0d335304487a1880f4e32d6aeec \
  --region eu-west-1
```

Recovery Actions

Option A: Restart Service

```
# Trigger new deployment (no code change)
aws apprunner start-deployment \
  --service-arn arn:aws:apprunner:eu-west-1:324480209768:service/drop-
web/8e45b0d335304487a1880f4e32d6aeec \
  --region eu-west-1

# Monitor deployment status
aws apprunner describe-service \
  --service-arn arn:aws:apprunner:eu-west-1:324480209768:service/drop-
web/8e45b0d335304487a1880f4e32d6aeec \
  --query 'Service.Status' \
  --region eu-west-1
```

Option B: Rollback to Previous Image

```
# 1. List recent ECR images
aws ecr describe-images \
  --repository-name drop-web \
  --region eu-west-1 \
  --query 'sort_by(imageDetails,& imagePushedAt)[-5:]'

# 2. Update service to use previous image tag
# (Manual step: Update .github/workflows/deploy-aws.yml with previous tag and push)
```

```
# 3. Or update directly via App Runner console (rollback to previous deployment)
```

RTO: 5-10 minutes (restart) / 15-20 minutes (rollback)

Scenario 2: RDS Database Failure

Symptoms

- Connection timeouts to `drop-db.czu2qe4quy4v.eu-west-1.rds.amazonaws.com`
- Database errors in App Runner logs
- RDS CloudWatch metrics show instance down

Investigation Steps

```
# 1. Check RDS instance status
aws rds describe-db-instances \
  --db-instance-identifier drop-db \
  --region eu-west-1 \
  --query 'DBInstances[0].DBInstanceStatus'

# 2. Check for automated snapshots
aws rds describe-db-snapshots \
  --db-instance-identifier drop-db \
  --region eu-west-1 \
  --query 'DBSnapshots[?SnapshotType==`automated`] | sort_by(@, &SnapshotCreateTime)[-5:]'

# 3. Review recent events
aws rds describe-events \
  --source-identifier drop-db \
  --source-type db-instance \
  --region eu-west-1 \
  --duration 60
```

Recovery Actions

Option A: Restore from Latest Automated Snapshot

```
# 1. Identify latest snapshot
LATEST_SNAPSHOT=$(aws rds describe-db-snapshots \
  --db-instance-identifier drop-db \
  --region eu-west-1 \
```

```
  --query 'DBSnapshots[?SnapshotType==`automated`] | sort_by(@, &SnapshotCreateTime)[-
1].DBSnapshotIdentifier' \
  --output text)

echo "Latest snapshot: $LATEST_SNAPSHOT"

# 2. Restore to new instance
aws rds restore-db-instance-from-db-snapshot \
  --db-instance-identifier drop-db-restored \
  --db-snapshot-identifier $LATEST_SNAPSHOT \
  --db-instance-class db.t4g.micro \
  --vpc-security-group-ids sg-XXXXX \
  --db-subnet-group-name default \
  --region eu-west-1

# 3. Wait for restore to complete (10-20 minutes)
aws rds wait db-instance-available \
  --db-instance-identifier drop-db-restored \
  --region eu-west-1

# 4. Update DATABASE_URL in App Runner
# (Manual step: Update environment variable via AWS Console or CLI)

# 5. Verify connection
NEW_ENDPOINT=$(aws rds describe-db-instances \
  --db-instance-identifier drop-db-restored \
  --query 'DBInstances[0].Endpoint.Address' \
  --output text \
  --region eu-west-1)

echo "New endpoint: $NEW_ENDPOINT"
```

Option B: Point-in-Time Recovery

```
# Restore to specific timestamp (e.g., 1 hour ago)
aws rds restore-db-instance-to-point-in-time \
  --source-db-instance-identifier drop-db \
  --target-db-instance-identifier drop-db-pitr \
  --restore-time $(date -u -d '1 hour ago' '+%Y-%m-%dT%H:%M:%SZ') \
  --db-instance-class db.t4g.micro \
```

```
  --region eu-west-1

# Wait for restore
aws rds wait db-instance-available \
  --db-instance-identifier drop-db-pitr \
  --region eu-west-1
```

RPO: 24 hours (snapshot) / 5 minutes (PITR) **RTO:** 30 minutes (snapshot) / 30 minutes (PITR)

Scenario 3: Data Corruption

Symptoms

- Application reports data inconsistencies
- Missing or incorrect records in database
- User reports of lost data

Investigation Steps

```
# 1. Connect to RDS and inspect data
psql -h drop-db.czu2qe4quy4v.eu-west-1.rds.amazonaws.com \
     -U dropuser \
     -d dropapp \
     -c "SELECT COUNT(*) FROM users WHERE deleted_at IS NOT NULL;"

# 2. Check audit_log table for suspicious activity
psql -h drop-db.czu2qe4quy4v.eu-west-1.rds.amazonaws.com \
     -U dropuser \
     -d dropapp \
     -c "SELECT * FROM audit_log WHERE action IN ('DELETE', 'UPDATE') ORDER BY timestamp DESC
LIMIT 50;"

# 3. Identify time of corruption
# Review application logs and database query logs
```

Recovery Actions

Option A: Selective Data Restore (if corruption is isolated)

```
# 1. Create temporary snapshot of current state
aws rds create-db-snapshot \
```

```
  --db-instance-identifier drop-db \
  --db-snapshot-identifier drop-db-before-restore-$(date +%Y%m%d-%H%M) \
  --region eu-west-1

# 2. Restore clean snapshot to temporary instance
CLEAN_SNAPSHOT=<snapshot-before-corruption>

aws rds restore-db-instance-from-db-snapshot \
  --db-instance-identifier drop-db-temp \
  --db-snapshot-identifier $CLEAN_SNAPSHOT \
  --db-instance-class db.t4g.micro \
  --region eu-west-1

# 3. Export affected tables from clean instance
pg_dump -h <temp-endpoint> \
        -U dropuser \
        -d dropapp \
        -t users \
        -t transactions \
        --data-only \
        > clean_data.sql

# 4. Selectively import into production (after verification)
psql -h drop-db.czu2qe4quy4v.eu-west-1.rds.amazonaws.com \
     -U dropuser \
     -d dropapp \
     < clean_data.sql

# 5. Terminate temporary instance
aws rds delete-db-instance \
  --db-instance-identifier drop-db-temp \
  --skip-final-snapshot \
  --region eu-west-1
```

Option B: Full Database Restore (see Scenario 2)

RTO: 1-2 hours (selective) / 30 minutes (full restore) **RPO:** Depends on snapshot age

---

Scenario 4: Full Region Outage (eu-west-1)

Current State

- **No automated cross-region failover**
- **No replica in secondary region**
- **Manual failover required**

Investigation Steps

```
# 1. Check AWS Service Health Dashboard
# https://health.aws.amazon.com/health/status

# 2. Verify RDS snapshots are accessible
aws rds describe-db-snapshots \
  --db-instance-identifier drop-db \
  --region eu-west-1

# 3. Check ECR images (may need to copy to secondary region)
aws ecr describe-images \
  --repository-name drop-web \
  --region eu-west-1
```

Recovery Actions (Manual Failover to eu-north-1)

```
# 1. Copy latest RDS snapshot to eu-north-1
LATEST_SNAPSHOT=$(aws rds describe-db-snapshots \
  --db-instance-identifier drop-db \
  --region eu-west-1 \
  --query 'DBSnapshots[?SnapshotType==`automated`] | sort_by(@, &SnapshotCreateTime)[-
1].DBSnapshotIdentifier' \
  --output text)

aws rds copy-db-snapshot \
  --source-db-snapshot-identifier arn:aws:rds:eu-west-1:324480209768:snapshot:$LATEST_SNAPSHOT
\
  --target-db-snapshot-identifier drop-db-failover-$(date +%Y%m%d) \
  --region eu-north-1

# 2. Restore RDS in eu-north-1
aws rds restore-db-instance-from-db-snapshot \
  --db-instance-identifier drop-db-failover \
  --db-snapshot-identifier drop-db-failover-$(date +%Y%m%d) \
```

```
  --db-instance-class db.t4g.micro \
  --region eu-north-1

# 3. Copy ECR image to eu-north-1
# (Manual: create ECR repo in eu-north-1, retag and push latest image)

# 4. Deploy App Runner in eu-north-1
# (Manual: create new App Runner service via console with failover database endpoint)

# 5. Update DNS (when getdrop.no is active)
# Point getdrop.no to new App Runner URL
```

RTO: 2-4 hours (manual process) **RPO:** Last snapshot before outage (24 hours worst case, 5 minutes with PITR if available)

Scenario 5: Security Incident

Symptoms

- Suspicious database activity
- Unauthorized access attempts
- AML alerts triggered
- STR report filed

Investigation Steps

```
# 1. Check audit logs for suspicious activity
psql -h drop-db.czu2qe4quy4v.eu-west-1.rds.amazonaws.com \
     -U dropuser \
     -d dropapp \
     -c "SELECT * FROM audit_log WHERE timestamp > NOW() - INTERVAL '24 hours' ORDER BY
timestamp DESC;"

# 2. Review AML alerts
psql -h drop-db.czu2qe4quy4v.eu-west-1.rds.amazonaws.com \
     -U dropuser \
     -d dropapp \
     -c "SELECT * FROM aml_alerts WHERE status = 'open' OR created_at > NOW() - INTERVAL '24
hours';"
```

```
# 3. Check AWS CloudTrail for API activity
aws cloudtrail lookup-events \
  --lookup-attributes AttributeKey=ResourceName,AttributeValue=drop-db \
  --region eu-west-1 \
  --max-results 50

# 4. Review App Runner access logs
aws logs filter-log-events \
  --log-group-name /aws/apprunner/drop-web/8e45b0d335304487a1880f4e32d6aeec/application \
  --start-time $(date -u -d '24 hours ago' +%s)000 \
  --region eu-west-1
```

Containment Actions

```
# 1. Revoke compromised sessions
psql -h drop-db.czu2qe4quy4v.eu-west-1.rds.amazonaws.com \
     -U dropuser \
     -d dropapp \
     -c "UPDATE sessions SET revoked = 1 WHERE user_id IN (SELECT user_id FROM aml_alerts
WHERE status = 'open');"

# 2. Temporarily disable affected users
psql -h drop-db.czu2qe4quy4v.eu-west-1.rds.amazonaws.com \
     -U dropuser \
     -d dropapp \
     -c "UPDATE users SET kyc_status = 'rejected' WHERE id IN (SELECT user_id FROM aml_alerts
WHERE severity = 'critical');"

# 3. Rotate database credentials
aws rds modify-db-instance \
  --db-instance-identifier drop-db \
  --master-user-password <new-password> \
  --apply-immediately \
  --region eu-west-1

# Update DATABASE_URL in App Runner with new password

# 4. Enable enhanced monitoring
aws rds modify-db-instance \
  --db-instance-identifier drop-db \
```

```
  --monitoring-interval 1 \
  --monitoring-role-arn arn:aws:iam::324480209768:role/rds-monitoring-role \
  --region eu-west-1

# 5. Take forensic snapshot
aws rds create-db-snapshot \
  --db-instance-identifier drop-db \
  --db-snapshot-identifier drop-db-incident-$(date +%Y%m%d-%H%M) \
  --region eu-west-1
```

Investigation & Remediation

1. **Analyze audit logs** — identify scope of breach
2. **File STR reports** — if financial crime suspected (via `str_reports` table)
3. **Notify Finanstilsynet** — if user data compromised (GDPR requirement)
4. **Update security policies** — patch vulnerabilities
5. **User communication** — notify affected users if required by GDPR

RTO: Immediate containment (revoke sessions) / 24-48 hours full investigation

---

RTO/RPO Targets

Scenario	RTO	RPO
App Runner restart	5-10 minutes	0 (no data loss)
App Runner rollback	15-20 minutes	0 (no data loss)
RDS snapshot restore	30 minutes	24 hours (last snapshot)
RDS PITR restore	30 minutes	5 minutes (PITR granularity)
Full region failover	2-4 hours	24 hours (manual process)
Security incident containment	Immediate	0 (logs preserved)

Contacts

Primary

- **Alem Bašić (CEO):** +47 40 47 42 51
- **Email:** alem@alai.no

AI Operations

- **John (AI Director):** Slack #drop-alerts channel

External Support

- **AWS Support:** Premium support via AWS Console
- **Fly.io Support:** Email support@fly.io

Runbook Maintenance

Review Schedule

- **Quarterly review** — verify all ARNs, endpoints, and procedures
- **After incidents** — update based on lessons learned
- **Before major releases** — verify backup and rollback procedures

Test Schedule

- **Annually** — full DR drill (restore from snapshot to temporary instance)
- **Quarterly** — App Runner restart and rollback tests
- **Monthly** — verify snapshot creation and retention

Change Log

Date	Change	Author
2026-02-18	Initial version created	Builder 3 (AI)

Appendix: Useful Commands

Quick Health Check

```
# Check App Runner status
aws apprunner describe-service \
```

```
  --service-arn arn:aws:apprunner:eu-west-1:324480209768:service/drop-
web/8e45b0d335304487a1880f4e32d6aeec \
  --query 'Service.Status' \
  --output text \
  --region eu-west-1

# Check RDS status
aws rds describe-db-instances \
  --db-instance-identifier drop-db \
  --query 'DBInstances[0].DBInstanceStatus' \
  --output text \
  --region eu-west-1

# Check latest snapshot age
aws rds describe-db-snapshots \
  --db-instance-identifier drop-db \
  --region eu-west-1 \
  --query 'DBSnapshots[?SnapshotType==`automated`] | sort_by(@, &SnapshotCreateTime)[-
1].SnapshotCreateTime' \
  --output text
```

Database Connection Test

```
# Test connection from local machine
psql -h drop-db.czu2qe4quy4v.eu-west-1.rds.amazonaws.com \
     -U dropuser \
     -d dropapp \
     -c "SELECT 1;"
```

Log Streaming

```
# Stream App Runner application logs
aws logs tail /aws/apprunner/drop-web/8e45b0d335304487a1880f4e32d6aeec/application \
  --follow \
  --region eu-west-1

# Stream RDS error logs
aws rds download-db-log-file-portion \
  --db-instance-identifier drop-db \
```

```
  --log-file-name error/postgresql.log \
  --region eu-west-1
```

Revision #7
Created 2026-02-23 11:28:56 UTC by John
Updated 2026-05-23 10:58:16 UTC by John