Docker Quickstart

Infrastructure setup and service validation

PSScript Manager Docs

Docker Quick Start Guide

This guide will help you quickly get started with the enhanced Docker infrastructure including connection pooling, high availability, and automated backups.

Prerequisites

- Docker Engine 20.10+
- Docker Compose 2.0+
- At least 4GB RAM available
- At least 20GB disk space

Quick Start (5 Minutes)

1. Clone and Configure

```
# Copy environment file
cp .env.example .env

# Edit .env if needed (optional for development)
# nano .env
```

2. Create Required Directories

```
# Create backup directories
mkdir -p backups/postgres backups/redis backups/logs
```

3. Start All Services

```
# Using docker-compose
docker-compose up -d

# OR using the management script
chmod +x docker-manage.sh
./docker-manage.sh start
```

4. Verify Services

```
# Check all services are running
docker-compose ps

# Check health status
./docker-manage.sh health
```

5. Access Services

- **Frontend**: http://localhost:3000
- **Backend API**: http://localhost:4000
- **AI Service**: http://localhost:8000
- **PostgreSQL** (direct): localhost:5432
- **PgBouncer**: localhost:6432
- **Redis Master**: localhost:6379
- **PgAdmin** (dev): http://localhost:5050 (admin@example.com / admin)
- **Redis Commander** (dev): http://localhost:8082

Infrastructure Overview

Connection Pooling (PgBouncer)

What it does: Manages PostgreSQL connections efficiently **Why you need it**: Prevents connection exhaustion, improves performance **How to use**: Applications connect to port 6432 instead of 5432

```
# Check pool status
./docker-manage.sh pgbouncer pools
```

High Availability (Redis Sentinel)

What it does: Automatic failover for Redis **Why you need it**: Zero downtime for caching layer **How it works**: 3 sentinels monitor master, auto-promote replica on failure

```
# Check Redis status
./docker-manage.sh redis sentinel
```

Automated Backups

What it does: Scheduled backups with retention management **Why you need it**: Disaster recovery and data protection **Schedules**: - PostgreSQL Full: Daily at 2 AM - PostgreSQL Incremental: Every 6 hours - Redis: Every 4 hours

```
# Manual backup
./docker-manage.sh backup postgres-full

# List backups
ls -lh backups/postgres/
```

Common Operations

View Logs

```
# All services
docker-compose logs -f

# Specific service
docker-compose logs -f backend
./docker-manage.sh logs backend
```

Restart Services

```
# All services
./docker-manage.sh restart

# Single service
docker-compose restart backend
```

Run Backups

```
# PostgreSQL full backup
./docker-manage.sh backup postgres-full

# PostgreSQL incremental backup
./docker-manage.sh backup postgres-incremental

# Redis backup
./docker-manage.sh backup redis
```

Check Health

```
 # Comprehensive health check
./docker-manage.sh health

# Check specific service
docker-compose exec postgres pg_isready
docker-compose exec redis-master redis-cli ping
```

Access Database

```
 # Via PgBouncer (recommended)
docker-compose exec pgbouncer psql -h localhost -p 6432 -U postgre

# Direct PostgreSQL
docker-compose exec postgres psql -U postgres -d psscript

# Redis
docker-compose exec redis-master redis-cli
```

Monitoring

PgBouncer Statistics

```
# Connection pools
./docker-manage.sh pgbouncer pools

# Statistics
./docker-manage.sh pgbouncer stats

# Connected clients
./docker-manage.sh pgbouncer clients
```

Redis Cluster Status

```
# Master info
./docker-manage.sh redis info

# Sentinel status
./docker-manage.sh redis sentinel

# Replica status
./docker-manage.sh redis replicas
```

Backup Status

```
# View backup logs
tail -f backups/logs/backup.log

# View health logs
tail -f backups/logs/health.log

# List backups
ls -lh backups/postgres/
ls -lh backups/redis/
```

Troubleshooting

Services Won't Start

```
 # Check Docker status
docker info

# Check logs
docker-compose logs

# Rebuild services
./docker-manage.sh rebuild
./docker-manage.sh start
```

Connection Issues

```
 # Verify network
docker network ls
docker network inspect psscript_psscript-network

# Check service health
./docker-manage.sh health

# Test connectivity
docker-compose exec backend ping postgres
docker-compose exec backend ping pgbouncer
docker-compose exec backend ping redis-master
```

Backup Failures

```
# Check disk space
df -h

# Check backup logs
tail -100 backups/logs/backup.log

# Manual backup test
./docker-manage.sh backup postgres-full
```

PgBouncer Issues

```
# Check PgBouncer logs
docker-compose logs pgbouncer

# Verify configuration
docker-compose exec pgbouncer cat /etc/pgbouncer/pgbouncer.ini

# Test direct PostgreSQL connection
docker-compose exec postgres psql -U postgres -d psscript -c "SEL
```

Redis Failover Not Working

```
# Check sentinel logs
docker-compose logs redis-sentinel-1

# Verify sentinel configuration
docker-compose exec redis-sentinel-1 redis-cli -p 26379 SENTINEL
```

```
# Check replica status
docker-compose exec redis-master redis-cli INFO replication
```

Testing Disaster Recovery

Test PostgreSQL Restore

```
 # Create test backup
./docker-manage.sh backup postgres-full

# List backups
./docker-manage.sh restore postgres

# Restore (this will ask for confirmation)
./docker-manage.sh restore postgres /backups/postgres/psscript_fu
```

Test Redis Failover

```
 # Stop master
docker-compose stop redis-master

# Watch sentinel promote replica
docker-compose logs -f redis-sentinel-1

# Verify new master
./docker-manage.sh redis sentinel

# Restart original master (becomes replica)
docker-compose start redis-master
```

Stopping Services

Graceful Shutdown

```
 # Stop all services (keeps data)
./docker-manage.sh stop

# OR
docker-compose down
```

Complete Cleanup

```
 # Remove containers and volumes (DELETES ALL DATA)
./docker-manage.sh clean

# OR
docker-compose down -v
```

Production Deployment

Before Going to Production

1. **Update passwords in .env**:
2. DB_PASSWORD
3. JWT_SECRET

4. REFRESH_TOKEN_SECRET

5. **Configure S3 backups** (add to .env): `bash BACKUP_S3_BUCKET=your-production-bucket AWS_ACCESS_KEY_ID=your-key AWS_SECRET_ACCESS_KEY=your-secret`

6. **Enable Redis password** (edit redis configs):

7. Uncomment `requirepass` in `/docker/redis/redis-master.conf`
8. Uncomment `masterauth` in `/docker/redis/redis-replica.conf`

9. Update backend REDIS_URL to include password

10. **Configure SSL/TLS**:

11. Set up PostgreSQL SSL certificates
12. Configure Redis TLS

13. Use HTTPS for frontend/backend

14. **Adjust resource limits**:

15. Edit `docker-compose.yml` to add resource limits
16. Increase pool sizes for high traffic

17. Adjust memory limits for Redis/PostgreSQL

18. **Set up monitoring**:

19. Configure external monitoring
20. Set up alerting for backup failures
21. Monitor disk space and performance

Production docker-compose.yml Additions

```yaml
services:
  backend:
    deploy:
      resources:
        limits:
          cpus: '2'
          memory: 2G
        reservations:
          cpus: '1'
          memory: 1G
    restart: always
```

Next Steps

1. **Read full documentation**: `docs/DOCKER-INFRASTRUCTURE.md`
2. **Review backup documentation**: `docker/backup/README.md`
3. **Test backup and restore procedures**
4. **Set up monitoring and alerting**
5. **Configure S3 for cloud backups**

Management Script Reference

```
# Service management
./docker-manage.sh start        # Start all services
./docker-manage.sh stop         # Stop all services
./docker-manage.sh restart      # Restart all services
./docker-manage.sh status       # Show status

# Monitoring
./docker-manage.sh health       # Health check
./docker-manage.sh logs [svc]   # View logs

# Backups
./docker-manage.sh backup postgres-full
./docker-manage.sh backup redis
./docker-manage.sh restore postgres [file]

# PgBouncer
./docker-manage.sh pgbouncer pools
./docker-manage.sh pgbouncer stats

# Redis
./docker-manage.sh redis info
./docker-manage.sh redis sentinel

# Utilities
./docker-manage.sh shell backend
./docker-manage.sh rebuild
./docker-manage.sh clean

# Help
./docker-manage.sh help
```

Support

- Full Documentation: `docs/DOCKER-INFRASTRUCTURE.md`
- Backup Guide: `docker/backup/README.md`
- Project Setup: `docs/DOCKER-SETUP.md`
- Getting Started: `docs/GETTING-STARTED.md`

Useful Commands Cheat Sheet

```
# Start everything
./docker-manage.sh start

# Check if healthy
./docker-manage.sh health

# View backend logs
./docker-manage.sh logs backend

# Run backup
./docker-manage.sh backup postgres-full

# Check PgBouncer
./docker-manage.sh pgbouncer pools

# Check Redis
./docker-manage.sh redis sentinel

# Access PostgreSQL
docker-compose exec postgres psql -U postgres -d psscript

# Access Redis
docker-compose exec redis-master redis-cli

# Stop everything
./docker-manage.sh stop
```

Generated 2026-01-16 21:23 UTC