JOJI Oi Deployment Guide

Overview

This guide covers deployment strategies, configuration options, and best practices for deploying the JOJI Oi memory system in various environments.

Deployment Options

1. Local Development

Quick Start

```
# Clone repository
git clone https://github.com/gtsurkav-sudo/JOJIAI.git
cd JOJIAI

# Create virtual environment
python -m venv venv
source venv/bin/activate  # On Windows: venv\Scripts\activate

# Install dependencies
pip install -e ".[dev,test]"

# Run tests
pytest tests/

# Start development server
python -m jojiai.core
```

Development Configuration

```
# Set development environment
export JOJIAI_ENV=development
export JOJIAI_LOG_LEVEL=DEBUG
export JOJIAI_MEMORY_PATH=./dev_memory
export JOJIAI_BACKUP_PATH=./dev_backups
```

2. Production Deployment

System Requirements

- **OS**: Linux (Ubuntu 20.04+ recommended)
- **Python**: 3.9 or higher
- **Memory**: 2GB RAM minimum, 4GB recommended
- **Storage**: 10GB minimum, SSD recommended
- **Network**: HTTP/HTTPS access for monitoring

Installation Steps

Step 1: System Preparation

```
# Update system
sudo apt update && sudo apt upgrade -y

# Install Python and dependencies
sudo apt install -y python3 python3-pip python3-venv build-essential

# Create application user
sudo useradd -r -s /bin/false -d /opt/jojiai jojiai
sudo mkdir -p /opt/jojiai
sudo chown jojiai:jojiai /opt/jojiai
```

Step 2: Application Installation

```
# Switch to application user
sudo -u jojiai -s

# Create virtual environment
cd /opt/jojiai
python3 -m venv venv
source venv/bin/activate

# Install JOJI Oi
pip install jojiai

# Create directory structure
mkdir -p {data,logs,backups,config}
```

Step 3: Configuration

```
# Create configuration file
cat > /opt/jojiai/config/jojiai.json << EOF
{
  "memory_path": "/opt/jojiai/data",
  "backup_path": "/opt/jojiai/backups",
  "wal_path": "/opt/jojiai/data/memory.wal",
  "max_memory_size": 104857600,
  "backup_interval": 3600,
  "wal_flush_interval": 60,
  "lock_timeout": 30,
  "log_level": "INFO",
  "metrics_port": 8000
}
EOF
```

Step 4: Systemd Service

```
# Create systemd service file
sudo tee /etc/systemd/system/jojiai.service << EOF
[Unit]
Description=JOJI Oi Memory System
After=network.target
Wants=network.target

[Service]
Type=simple
User=jojiai
Group=jojiai
WorkingDirectory=/opt/jojiai
Environment=PATH=/opt/jojiai/venv/bin
Environment=JOJIAI_CONFIG=/opt/jojiai/config/jojiai.json
ExecStart=/opt/jojiai/venv/bin/python -m jojiai.core
ExecReload=/bin/kill -HUP \$MAINPID
Restart=always
RestartSec=10
StandardOutput=journal
StandardError=journal
SyslogIdentifier=jojiai

# Security settings
NoNewPrivileges=yes
PrivateTmp=yes
ProtectSystem=strict
ProtectHome=yes
ReadWritePaths=/opt/jojiai/data /opt/jojiai/logs /opt/jojiai/backups

[Install]
WantedBy=multi-user.target
EOF

# Enable and start service
sudo systemctl daemon-reload
sudo systemctl enable jojiai
sudo systemctl start jojiai

# Check status
sudo systemctl status jojiai
```

3. Docker Deployment

Single Container

```
# Build image
docker build -t jojiai:latest .

# Run container
docker run -d \
  --name jojiai \
  --restart unless-stopped \
  -p 8000:8000 \
  -p 8001:8001 \
  -v jojiai_data:/app/data \
  -v jojiai_logs:/app/logs \
  -v jojiai_backups:/app/backups \
  -e JOJIAI_LOG_LEVEL=INFO \
  jojiai:latest
```

Docker Compose

```yaml
# docker-compose.yml
version: '3.8'

services:
  jojiai:
    build: .
    container_name: jojiai
    restart: unless-stopped
    ports:
      - "8000:8000"
      - "8001:8001"
    volumes:
      - jojiai_data:/app/data
      - jojiai_logs:/app/logs
      - jojiai_backups:/app/backups
    environment:
      - JOJIAI_LOG_LEVEL=INFO
      - JOJIAI_METRICS_PORT=8000
      - JOJIAI_MAX_MEMORY_SIZE=104857600
    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost:8000/health"]
      interval: 30s
      timeout: 10s
      retries: 3
      start_period: 40s

  prometheus:
    image: prom/prometheus:latest
    container_name: jojiai_prometheus
    ports:
      - "9090:9090"
    volumes:
      - ./monitoring/prometheus.yml:/etc/prometheus/prometheus.yml
      - prometheus_data:/prometheus
    command:
      - '--config.file=/etc/prometheus/prometheus.yml'
      - '--storage.tsdb.path=/prometheus'
      - '--web.console.libraries=/etc/prometheus/console_libraries'
      - '--web.console.templates=/etc/prometheus/consoles'

  grafana:
    image: grafana/grafana:latest
    container_name: jojiai_grafana
    ports:
      - "3000:3000"
    volumes:
      - grafana_data:/var/lib/grafana
      - ./monitoring/grafana:/etc/grafana/provisioning
    environment:
      - GF_SECURITY_ADMIN_PASSWORD=admin123

volumes:
  jojiai_data:
  jojiai_logs:
  jojiai_backups:
  prometheus_data:
  grafana_data:
```

4. Kubernetes Deployment

Namespace and ConfigMap

```yaml
# k8s/namespace.yaml
apiVersion: v1
kind: Namespace
metadata:
  name: jojiai

---
# k8s/configmap.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: jojiai-config
  namespace: jojiai
data:
  jojiai.json: |
    {
      "memory_path": "/app/data",
      "backup_path": "/app/backups",
      "wal_path": "/app/data/memory.wal",
      "max_memory_size": 104857600,
      "backup_interval": 3600,
      "wal_flush_interval": 60,
      "lock_timeout": 30,
      "log_level": "INFO",
      "metrics_port": 8000
    }
```

Deployment and Service

```yaml
# k8s/deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: jojiai
  namespace: jojiai
  labels:
    app: jojiai
spec:
  replicas: 1
  selector:
    matchLabels:
      app: jojiai
  template:
    metadata:
      labels:
        app: jojiai
    spec:
      containers:
      - name: jojiai
        image: jojiai:latest
        ports:
        - containerPort: 8000
          name: metrics
        - containerPort: 8001
          name: health
        env:
        - name: JOJIAI_CONFIG
          value: /app/config/jojiai.json
        volumeMounts:
        - name: config
          mountPath: /app/config
        - name: data
          mountPath: /app/data
        - name: logs
          mountPath: /app/logs
        - name: backups
          mountPath: /app/backups
        livenessProbe:
          httpGet:
            path: /health
            port: 8001
          initialDelaySeconds: 30
          periodSeconds: 10
        readinessProbe:
          httpGet:
            path: /health
            port: 8001
          initialDelaySeconds: 5
          periodSeconds: 5
        resources:
          requests:
            memory: "512Mi"
            cpu: "250m"
          limits:
            memory: "2Gi"
            cpu: "1000m"
      volumes:
      - name: config
        configMap:
          name: jojiai-config
      - name: data
```

```
        persistentVolumeClaim:
          claimName: jojiai-data
      - name: logs
        persistentVolumeClaim:
          claimName: jojiai-logs
      - name: backups
        persistentVolumeClaim:
          claimName: jojiai-backups

---
# k8s/service.yaml
apiVersion: v1
kind: Service
metadata:
  name: jojiai-service
  namespace: jojiai
  labels:
    app: jojiai
spec:
  selector:
    app: jojiai
  ports:
  - name: metrics
    port: 8000
    targetPort: 8000
  - name: health
    port: 8001
    targetPort: 8001
  type: ClusterIP
```

Persistent Volumes

```yaml
# k8s/pvc.yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: jojiai-data
  namespace: jojiai
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 10Gi
  storageClassName: fast-ssd

---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: jojiai-logs
  namespace: jojiai
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 5Gi
  storageClassName: standard

---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: jojiai-backups
  namespace: jojiai
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 20Gi
  storageClassName: standard
```

Configuration Management

Environment Variables

Variable	Default	Description
`JOJIAI_ENV`	`production`	Environment (development/staging/production)
`JOJIAI_CONFIG`	`./config.json`	Configuration file path
`JOJIAI_MEMORY_PATH`	`./memory`	Memory storage path
`JOJIAI_BACKUP_PATH`	`./backups`	Backup storage path
`JOJIAI_LOG_LEVEL`	`INFO`	Logging level
`JOJIAI_METRICS_PORT`	`8000`	Metrics server port
`JOJIAI_MAX_MEMORY_SIZE`	`104857600`	Maximum memory size (bytes)
`JOJIAI_BACKUP_INTERVAL`	`3600`	Backup interval (seconds)
`JOJIAI_WAL_FLUSH_INTERVAL`	`60`	WAL flush interval (seconds)
`JOJIAI_LOCK_TIMEOUT`	`30`	File lock timeout (seconds)

Configuration File Format

```
{
  "memory_path": "/app/data",
  "backup_path": "/app/backups",
  "wal_path": "/app/data/memory.wal",
  "max_memory_size": 104857600,
  "max_dialogues": 1000,
  "max_decisions": 500,
  "backup_interval": 3600,
  "wal_flush_interval": 60,
  "lock_timeout": 30,
  "log_level": "INFO",
  "metrics_port": 8000,
  "health_port": 8001,
  "enable_metrics": true,
  "enable_health_checks": true,
  "security": {
    "sanitize_input": true,
    "validate_paths": true,
    "max_content_size": 10240
  },
  "monitoring": {
    "structured_logging": true,
    "log_file": "/app/logs/jojiai.log",
    "metrics_endpoint": "/metrics",
    "health_endpoint": "/health"
  }
}
```

Monitoring Setup

Prometheus Configuration

```
# monitoring/prometheus.yml
global:
  scrape_interval: 15s
  evaluation_interval: 15s

scrape_configs:
  - job_name: 'jojiai'
    static_configs:
      - targets: ['jojiai:8000']
    scrape_interval: 10s
    metrics_path: /metrics

rule_files:
  - "jojiai_rules.yml"

alerting:
  alertmanagers:
    - static_configs:
        - targets:
          - alertmanager:9093
```

Grafana Dashboard

```
{
  "dashboard": {
    "title": "JOJI Oi System Metrics",
    "panels": [
      {
        "title": "Operation Rate",
        "type": "graph",
        "targets": [
          {
            "expr": "rate(jojiai_operations_total[5m])",
            "legendFormat": "{{operation_type}} - {{status}}"
          }
        ]
      },
      {
        "title": "Memory Usage",
        "type": "graph",
        "targets": [
          {
            "expr": "jojiai_memory_usage_bytes",
            "legendFormat": "{{memory_type}}"
          }
        ]
      },
      {
        "title": "Error Rate",
        "type": "singlestat",
        "targets": [
          {
            "expr": "rate(jojiai_errors_total[5m])"
          }
        ]
      }
    ]
  }
}
```

Load Balancing

Nginx Configuration

```
# /etc/nginx/sites-available/jojiai
upstream jojiai_backend {
    server 127.0.0.1:8000;
    # Add more servers for load balancing
    # server 127.0.0.1:8001;
    # server 127.0.0.1:8002;
}

server {
    listen 80;
    server_name jojiai.example.com;

    location / {
        proxy_pass http://jojiai_backend;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;

        # Health check
        proxy_connect_timeout 5s;
        proxy_send_timeout 10s;
        proxy_read_timeout 10s;
    }

    location /metrics {
        proxy_pass http://jojiai_backend/metrics;
        # Restrict access to metrics
        allow 10.0.0.0/8;
        deny all;
    }

    location /health {
        proxy_pass http://jojiai_backend/health;
        access_log off;
    }
}
```

Backup and Recovery

Automated Backup Script

```
#!/bin/bash
# /opt/jojiai/scripts/backup.sh

set -e

BACKUP_DIR="/opt/jojiai/backups"
RETENTION_DAYS=30
DATE=$(date +%Y%m%d_%H%M%S)

# Create backup
echo "Creating backup: backup_$DATE"
/opt/jojiai/venv/bin/memoryctl backup --name "backup_$DATE"

# Cleanup old backups
echo "Cleaning up backups older than $RETENTION_DAYS days"
find "$BACKUP_DIR" -name "backup_*" -type d -mtime +$RETENTION_DAYS -exec rm -rf {} \;

# Verify backup integrity
echo "Verifying backup integrity"
/opt/jojiai/venv/bin/memoryctl validate

echo "Backup completed successfully"
```

Cron Job Setup

```
# Add to crontab
crontab -e

# Backup every hour
0 * * * * /opt/jojiai/scripts/backup.sh >> /opt/jojiai/logs/backup.log 2>&1

# Cleanup logs daily
0 2 * * * find /opt/jojiai/logs -name "*.log" -mtime +7 -delete
```

Security Hardening

Firewall Configuration

```
# UFW rules
sudo ufw default deny incoming
sudo ufw default allow outgoing
sudo ufw allow ssh
sudo ufw allow from 10.0.0.0/8 to any port 8000  # Metrics
sudo ufw allow from 10.0.0.0/8 to any port 8001  # Health
sudo ufw enable
```

SSL/TLS Setup

```
# Install Certbot
sudo apt install certbot python3-certbot-nginx

# Obtain certificate
sudo certbot --nginx -d jojiai.example.com

# Auto-renewal
sudo crontab -e
0 12 * * * /usr/bin/certbot renew --quiet
```

Troubleshooting

Common Issues

Service Won't Start

```
# Check logs
sudo journalctl -u jojiai -f

# Check configuration
/opt/jojiai/venv/bin/memoryctl validate

# Check permissions
ls -la /opt/jojiai/data/
```

High Memory Usage

```
# Monitor memory
watch -n 1 'free -h && ps aux | grep jojiai'

# Check system metrics
curl http://localhost:8000/metrics | grep memory

# Cleanup old data
/opt/jojiai/venv/bin/memoryctl cleanup --keep 5
```

Performance Issues

```
# Check I/O performance
iostat -x 1

# Monitor operations
curl http://localhost:8000/metrics | grep duration

# Check concurrent operations
curl http://localhost:8000/metrics | grep concurrent
```

Scaling Considerations

Horizontal Scaling

- Use shared storage for data persistence
- Implement proper load balancing

- Consider database backend for large deployments
- Monitor resource usage across instances

Vertical Scaling

- Increase memory allocation
- Use faster storage (NVMe SSD)
- Optimize configuration parameters
- Monitor bottlenecks

Performance Optimization

```
# Tune kernel parameters
echo 'vm.swappiness=10' >> /etc/sysctl.conf
echo 'fs.file-max=65536' >> /etc/sysctl.conf

# Optimize file system
mount -o noatime,nodiratime /dev/sdb1 /opt/jojiai/data

# Increase file descriptor limits
echo 'jojiai soft nofile 65536' >> /etc/security/limits.conf
echo 'jojiai hard nofile 65536' >> /etc/security/limits.conf
```

Maintenance

Regular Tasks

```
# Weekly maintenance script
#!/bin/bash
# /opt/jojiai/scripts/maintenance.sh

# Update system packages
sudo apt update && sudo apt upgrade -y

# Cleanup old logs
find /opt/jojiai/logs -name "*.log" -mtime +30 -delete

# Optimize data files
/opt/jojiai/venv/bin/memoryctl cleanup --keep 10

# Check system health
/opt/jojiai/venv/bin/memoryctl status

# Restart service if needed
if ! curl -f http://localhost:8000/health; then
    sudo systemctl restart jojiai
fi
```

Monitoring Checklist

- [ ] Service health status
- [ ] Memory usage trends
- [ ] Error rates and patterns
- [ ] Backup completion
- [ ] Disk space utilization

- [ ] Network connectivity
- [ ] Security alerts

For additional support, see OPERATIONS.md (OPERATIONS.md) and SECURITY.md (SECURITY.md).