n8n Workflow Source Control Strategy

The Problem

n8n stores workflows in a database (SQLite/PostgreSQL), not as files. This means:

- ❌ Workflows are not automatically version controlled
- ❌ No git history of workflow changes
- ❌ No code review process for workflow changes
- ❌ Difficult to sync between environments (dev/staging/prod)
- ❌ Risk of data loss if database corrupted

The Solution: Workflows as JSON Files

Architecture

```
repo/
├── workflows/
│   ├── production/          # Active production workflows
│   │   ├── sms-hello-world.json
│   │   ├── sms-background-processor.json
│   │   └── customer-support-agent.json
│   ├── templates/           # Reusable templates
│   │   ├── twilio-webhook-starter.json
│   │   └── llm-routing-template.json
│   └── archive/             # Deprecated workflows
│       └── old-sms-handler.json
├── scripts/
│   ├── export-workflows.sh  # Export DB → JSON files
│   ├── import-workflows.sh  # Import JSON → DB
│   └── sync-workflows.sh    # Bi-directional sync
└── .github/workflows/
    └── workflow-sync.yml    # Auto-sync on push
```

Setup Instructions

Step 1: Create Workflows Directory Structure

```
bash
```

```
# SSH to VPS
ssh agent-vps
cd /opt/ai-agent-platform

# Create directory structure
mkdir -p workflows/production
mkdir -p workflows/templates
mkdir -p workflows/archive

# Create README
cat > workflows/README.md << 'EOF'
# n8n Workflows

## Directory Structure
- `production/` - Active workflows running in production
- `templates/` - Reusable workflow templates
- `archive/` - Deprecated/old workflows (kept for reference)

## Workflow Naming Convention
- Use kebab-case: `sms-hello-world.json`
- Include purpose in name: `customer-support-agent.json`
- Date archive files: `old-sms-handler-2025-10-01.json`

## Sync Process
1. Export workflows: `./scripts/export-workflows.sh`
2. Review changes: `git diff workflows/`
3. Commit: `git add workflows/ && git commit -m "Update workflows"`
4. Push: `git push origin main`

## Import to New Environment
```bash
./scripts/import-workflows.sh
```

EOF
```

Step 2: Create Export Script

Create `scripts/export-workflows.sh`:
```bash
#!/bin/bash

# Export n8n workflows to JSON files for version control
# This should be run periodically or after making workflow changes

set -e  # Exit on error

WORKFLOWS_DIR="/opt/ai-agent-platform/workflows/production"
TIMESTAMP=$(date +%Y%m%d_%H%M%S)

echo "📤 Exporting n8n workflows to git..."
echo "📅 $(date)"

# Ensure directory exists
mkdir -p "$WORKFLOWS_DIR"

# Get list of all workflow IDs and names
# This uses n8n CLI inside the container
WORKFLOWS=$(docker exec n8n n8n list:workflow --output=json)

# Check if we got workflows
if [ -z "$WORKFLOWS" ] || [ "$WORKFLOWS" = "[]" ]; then
    echo "⚠️  No workflows found to export"
    exit 0
fi

# Parse JSON and export each workflow
echo "$WORKFLOWS" | docker exec -i n8n node -e "
const workflows = JSON.parse(require('fs').readFileSync('/dev/stdin', 'utf8'));
workflows.forEach(wf => {
    // Sanitize name for filename
    const filename = wf.name.toLowerCase()
        .replace(/[^a-z0-9]+/g, '-')
        .replace(/^-|-$/g, '') + '.json';
    console.log(JSON.stringify({id: wf.id, name: wf.name, filename: filename}));
});
" | while IFS= read -r line; do
    # Extract workflow details
```

```
    WF_ID=$(echo "$line" | jq -r '.id')
    WF_NAME=$(echo "$line" | jq -r '.name')
    WF_FILE=$(echo "$line" | jq -r '.filename')

    echo "  📝 Exporting: $WF_NAME → $WF_FILE"

    # Export workflow to file
    docker exec n8n n8n export:workflow \
      --id="$WF_ID" \
      --output="/data/workflows/production/$WF_FILE" \
      --pretty
done

# Create backup of all workflows (timestamped)
BACKUP_FILE="/root/n8n-data/backups/workflows-backup-${TIMESTAMP}.json"
docker exec n8n n8n export:workflow --all --output="$BACKUP_FILE" --pretty

echo "✅ Export complete!"
echo "📊 Workflows exported to: $WORKFLOWS_DIR"
echo "💾 Backup created: $BACKUP_FILE"
echo ""
echo "Next steps:"
echo "  git add workflows/"
echo "  git commit -m 'Update workflows'"
echo "  git push origin main"
```

Make it executable:

```
bash

chmod +x scripts/export-workflows.sh
```

Step 3: Create Import Script

Create `scripts/import-workflows.sh`:

```
bash
```

```
#!/bin/bash

# Import workflows from JSON files to n8n database
# Used for: new environments, disaster recovery, reverting changes

set -e

WORKFLOWS_DIR="/opt/ai-agent-platform/workflows/production"

echo "📥 Importing workflows to n8n..."
echo "📅 $(date)"

# Check if directory exists and has files
if [ ! -d "$WORKFLOWS_DIR" ] || [ -z "$(ls -A $WORKFLOWS_DIR/*.json 2>/dev/null)" ]; then
    echo "❌ No workflow files found in $WORKFLOWS_DIR"
    exit 1
fi

# Import each workflow file
for workflow_file in "$WORKFLOWS_DIR"/*.json; do
    filename=$(basename "$workflow_file")
    echo "  📝 Importing: $filename"

    # Import workflow (will update if exists, create if new)
    docker exec -i n8n n8n import:workflow \
        --input="/data/workflows/production/$filename" \
        --separate
done

echo "✅ Import complete!"
echo ""
echo "🌐 Access n8n UI to verify: https://agents-platform.ngrok.io"
```

Make it executable:

```
bash

chmod +x scripts/import-workflows.sh
```

Step 4: Create Sync Script (Convenience)

Create `scripts/sync-workflows.sh`:

bash

```bash
#!/bin/bash

# Sync workflows: Export from n8n → Git commit → Push

set -e

echo "🔄 Syncing workflows to git..."

# Export workflows
./scripts/export-workflows.sh

# Check if there are changes
if [ -z "$(git status --porcelain workflows/)" ]; then
    echo "✅ No workflow changes detected"
    exit 0
fi

# Show changes
echo ""
echo "📊 Changes detected:"
git status workflows/

# Ask for confirmation
read -p "Commit and push these changes? (y/n) " -n 1 -r
echo
if [[ ! $REPLY =~ ^[Yy]$ ]]; then
    echo "❌ Sync cancelled"
    exit 0
fi

# Get commit message
echo "Enter commit message (or press Enter for default):"
read -r COMMIT_MSG

if [ -z "$COMMIT_MSG" ]; then
    COMMIT_MSG="Update workflows - $(date +%Y-%m-%d)"
fi

# Commit and push
git add workflows/
git commit -m "$COMMIT_MSG"
git push origin main
```

```
echo "✅ Workflows synced to git!"
```

Make it executable:

```bash
chmod +x scripts/sync-workflows.sh
```

Step 5: Update .gitignore

Your current `.gitignore` excludes ALL n8n-data. We need to be more specific:

```bash
# Edit .gitignore
nano .gitignore
```

Change from:

```gitignore
n8n-data/
```

Change to:

```gitignore
```

```
# n8n runtime data (excluded)
n8n-data/database.sqlite*
n8n-data/.n8n/
n8n-data/logs/
n8n-data/.cache/

# Keep workflows in git
!workflows/

# Other exclusions
.env
*.log
.DS_Store
node_modules/
*.bak
*.swp
.idea/
.vscode/
```

Step 6: Update docker-compose.yml Volume Mounts

Add workflow directory mount so n8n can access the JSON files:

yaml

```yaml
services:
  n8n:
    volumes:
      - /root/n8n-data:/home/node/.n8n
      - /root/n8n-data/transactions:/data/transactions
      - /root/n8n-data/backups:/data/backups
      - ./workflows:/data/workflows  # ADD THIS LINE
```

Restart n8n:

bash

```bash
docker compose down
docker compose up -d
```

Daily Workflow

When You Create/Modify Workflows

After making changes in n8n UI:

```bash
# SSH to VPS
ssh agent-vps
cd /opt/ai-agent-platform

# Export and sync to git
./scripts/sync-workflows.sh

# Follow prompts to commit and push
```

When Deploying to New Environment

```bash
# On new server
git clone https://github.com/YOUR_USERNAME/ai-agent-platform.git
cd ai-agent-platform

# Start services
docker compose up -d

# Wait for n8n to be ready
sleep 30

# Import workflows from git
./scripts/import-workflows.sh
```

Emergency Workflow Recovery

```bash
```

```
# If you accidentally deleted a workflow in UI
# 1. Find it in git history
git log --oneline workflows/

# 2. Restore the file
git checkout <commit-hash> workflows/production/workflow-name.json

# 3. Import back to n8n
./scripts/import-workflows.sh
```

Automated Sync (Advanced)

Option A: Cron Job (Automatic Export)

Add to crontab:

bash

```
# Export workflows to git every hour
0 * * * * cd /opt/ai-agent-platform && ./scripts/export-workflows.sh && git add workflows/ && git commit -m "Auto-sync v
```

Option B: GitHub Actions (Import on Push)

Create `.github/workflows/workflow-sync.yml`:

yaml

```yaml
name: Sync Workflows to VPS

on:
  push:
    paths:
      - 'workflows/**'
  workflow_dispatch:

jobs:
  sync:
    runs-on: ubuntu-latest
    steps:
      - name: Import workflows to VPS
        uses: appleboy/ssh-action@v0.1.5
        with:
          host: ${{ secrets.VPS_HOST }}
          username: root
          key: ${{ secrets.VPS_SSH_KEY }}
          script: |
            cd /opt/ai-agent-platform
            git pull origin main
            ./scripts/import-workflows.sh
```

Best Practices

Workflow Naming Convention

- ✅ `sms-hello-world.json`
- ✅ `customer-support-agent.json`
- ✅ `data-pipeline-processor.json`
- ❌ `Workflow 1.json`
- ❌ `test.json`
- ❌ `New Workflow Copy.json`

Commit Messages

```bash
```

```
# Good
git commit -m "Add Twilio SMS hello world workflow"
git commit -m "Fix error handling in background processor"
git commit -m "Update LLM routing logic for cost optimization"

# Bad
git commit -m "Update workflows"
git commit -m "Changes"
git commit -m "Fixed stuff"
```

Code Review Process

1. Export workflows: `./scripts/export-workflows.sh`
2. Review JSON diff: `git diff workflows/`
3. Create branch: `git checkout -b feature/new-workflow`
4. Commit: `git add workflows/ && git commit -m "Add feature"`
5. Push: `git push origin feature/new-workflow`
6. Create Pull Request
7. Review, merge
8. Import to production: `./scripts/import-workflows.sh`

Environment Strategy

```
workflows/
├── production/      # Production-ready workflows
├── staging/         # Testing workflows
└── development/     # Work-in-progress
```

Different environments:

bash

```
# Development
./scripts/import-workflows.sh workflows/development

# Staging
./scripts/import-workflows.sh workflows/staging

# Production
./scripts/import-workflows.sh workflows/production
```

Troubleshooting

"No workflows found"

```bash
# Check if n8n has workflows
docker exec n8n n8n list:workflow

# If empty, create a test workflow in UI first
```

"Permission denied"

```bash
# Make scripts executable
chmod +x scripts/*.sh
```

"Git won't track workflows/"

```bash
# Check .gitignore doesn't exclude it
cat .gitignore | grep workflows

# Should NOT have: workflows/
# Should have: n8n-data/database.sqlite
```

"Import fails"

```bash
```

```
# Check volume mount exists
docker exec n8n ls /data/workflows/production

# Check JSON is valid
cat workflows/production/workflow-name.json | jq .
```

Migration: Existing Workflows to Git

One-time migration for existing setup:

bash

```
# 1. Export current workflows
./scripts/export-workflows.sh

# 2. Verify they're in workflows/production/
ls -la workflows/production/

# 3. Add to git
git add workflows/
git commit -m "Initial import: Add existing workflows to source control"
git push origin main

# 4. Verify on GitHub
# Check: https://github.com/YOUR_USERNAME/ai-agent-platform/tree/main/workflows
```

Done! Now your workflows are in git forever.

Summary: Before vs After

Before (Current State)

❌ Workflows in database only
❌ No version history
❌ No code review
❌ Manual backup only
❌ Hard to sync environments

After (With This Setup)

- ✅ Workflows as JSON files in git
- ✅ Full version history
- ✅ Pull request workflow
- ✅ Automatic backups via git
- ✅ Easy environment sync
- ✅ Disaster recovery via git

Quick Reference

```bash
# Export workflows from n8n → git
./scripts/export-workflows.sh

# Import workflows from git → n8n
./scripts/import-workflows.sh

# Export + Commit + Push (one command)
./scripts/sync-workflows.sh

# List workflows in n8n
docker exec n8n n8n list:workflow

# Export single workflow
docker exec n8n n8n export:workflow --id=WORKFLOW_ID --output=/data/workflows/test.json
```

Next Steps

1. **Create directory structure** (workflows/production, etc.)
2. **Create export/import scripts** (make them executable)
3. **Update .gitignore** (allow workflows/)
4. **Update docker-compose.yml** (add volume mount)
5. **Export existing workflows** (run export script)
6. **Commit to git** (git add workflows/)
7. **Set up automated sync** (cron or GitHub Actions)

Now your workflows are code! 🎉