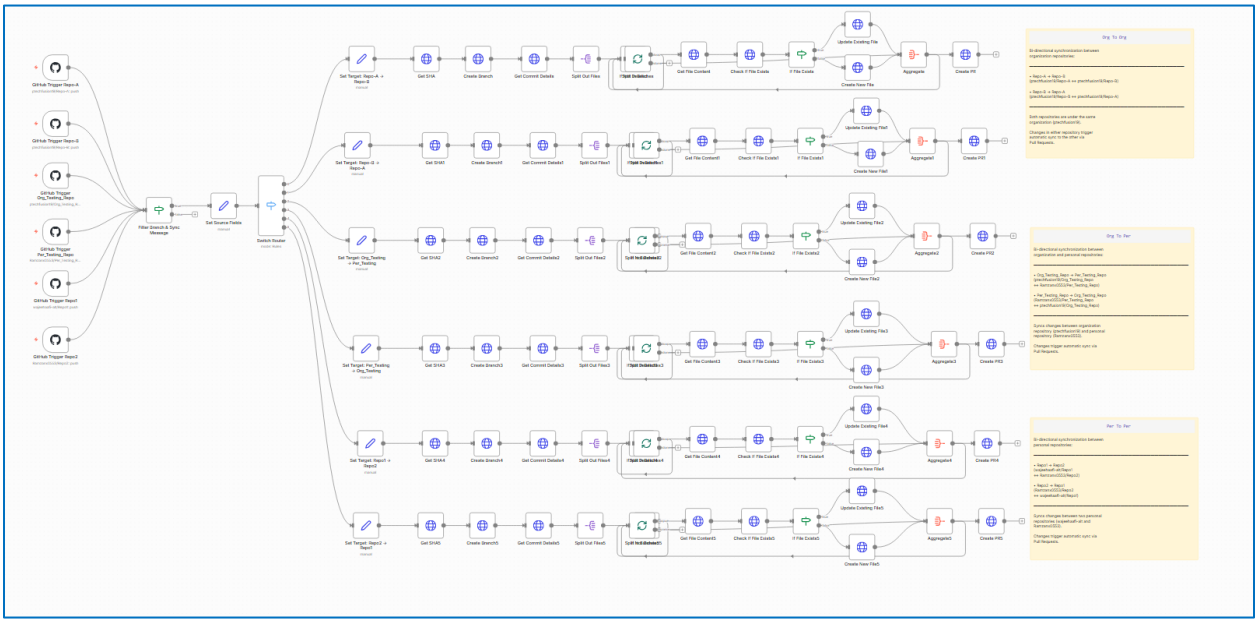
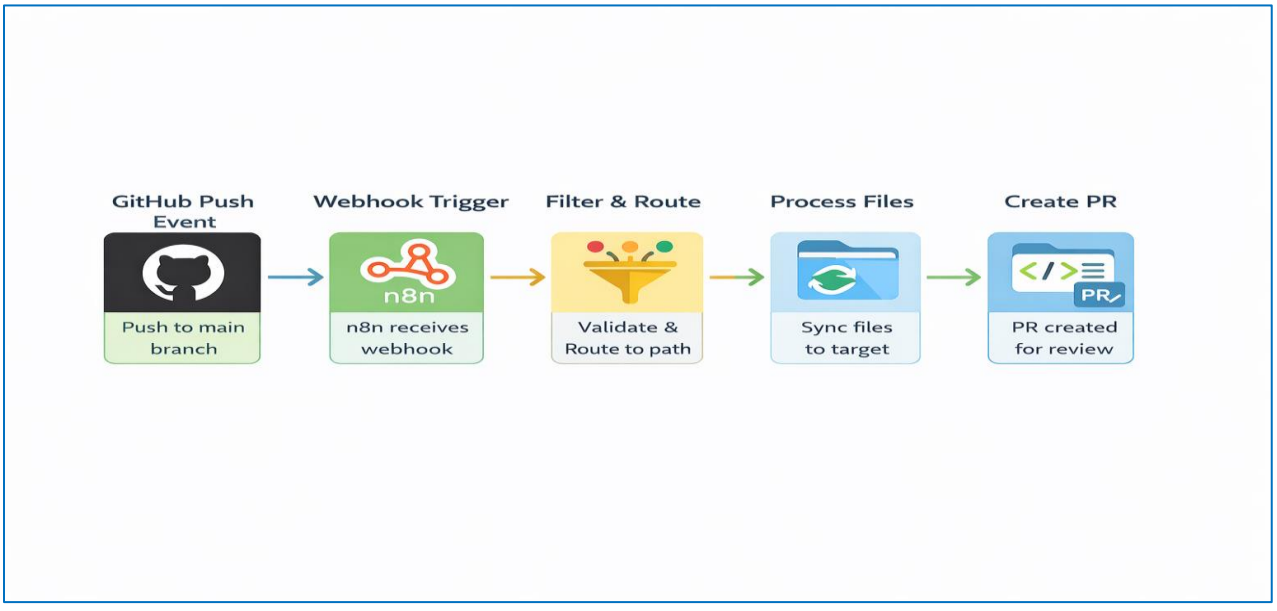
User Manual-Bi-Directional GitHub Repository Sync

Workflow



Workflow Architecture



1. Prerequisites

- Before setting up the workflow, ensure you have:
- An n8n instance (cloud or self-hosted)
- GitHub accounts with access to the repositories you want to sync
- Admin or write access to all source and target repositories
- Basic understanding of GitHub repositories and Pull Requests

2. Initial Setup

Import the Workflow:

- Open your n8n instance.
- Click "Workflows" → "Import from File".
- Upload the workflow JSON file.
- The workflow will be imported with all nodes and connections.

Activate the Workflow:

- Open the imported workflow.
- Click the "Active" toggle in the top right.
- The workflow is now listening for events (after credentials are configured).

3. Required Changes for Repositories

Change 1: Update GitHub Trigger Nodes
For each of the 6 GitHub Trigger nodes, update:

- **Owner:** Change to your GitHub username or organization name
- **Repository:** Change to your repository names

Locations to Update:

- GitHub Trigger Repo-A → Your first repository
- GitHub Trigger Repo-B → Your second repository
- GitHub Trigger Org_Testing_Repo → Your organization repository (if applicable)
- GitHub Trigger Per_Testing_Repo → Your personal repository
- GitHub Trigger Repo1 → Your third repository
- GitHub Trigger Repo2 → Your fourth repository

How to Update:

- Click on each GitHub Trigger node
- In the "Repository" field, select or type your repository name
- In the "Owner" field, select or type your username/organization
- Save the node

Change 2: Update Switch Router Conditions

Update the routing logic in the "Switch Router" node to match your repositories:

Current Conditions (Example):

- Condition 1: sourceOwner == "ptechfusion19" AND sourceRepo == "Repo-A"
- Condition 2: sourceOwner == "ptechfusion19" AND sourceRepo == "Repo-B"

What to Change:

- Click on "Switch Router" node
- For each of the 6 rules, update:
  - → sourceOwner value → Your GitHub username/organization
  - → sourceRepo value → Your repository names
  - → Update all 6 conditions to match your repository pairs

Example:

- If your repos are myorg/backend and myorg/frontend:
- Rule 1: sourceOwner == "myorg" AND sourceRepo == "backend"
- Rule 2: sourceOwner == "myorg" AND sourceRepo == "frontend"

Change 3: Update Target Repository Settings

Update each "Set Target" node with your target repositories:

Locations to Update:

- Set Target: Repo-A → Repo-B → Your first sync pair
- Set Target: Repo-B → Repo-A → Reverse of first pair
- Set Target: Org_Testing → Per_Testing → Your org-to-personal pair
- Set Target: Per_Testing → Org_Testing → Reverse of org-to-personal pair
- Set Target: Repo1 → Repo2 → Your personal-to-personal pair
- Set Target: Repo2 → Repo1 → Reverse of personal-to-personal pair

How to Update:

- Click on each "Set Target" node
- Update targetOwner → Your target repository owner
- Update targetRepo → Your target repository name
- Save the node

Change 4: Update HTTP Request URLs

Update all HTTP Request nodes to use your repository names:

Nodes to Update:

- All "Get SHA" nodes
- All "Create Branch" nodes
- All "Get Commit Details" nodes
- All "Get File Content" nodes
- All "Check If File Exists" nodes
- All "Update Existing File" nodes
- All "Create New File" nodes
- All "Create PR" nodes

What to Change:

- Click on each HTTP Request node
- In the URL field, replace repository names with yours
- Example: Change ptechfusion19/Repo-A to yourusername/yourrepo
- The URLs use expressions like {{ $('Set Target').item.json.targetOwner }} - these will automatically use your updated target settings

Note: Most URLs use dynamic expressions, so updating the "Set Target" nodes should be sufficient. Verify URLs that have hardcoded repository names.

Change 5: Add Your GitHub Credentials

Replace existing credentials with your own:

Credential 1: GitHub API Credential (for Triggers)

- Go to Settings → Credentials
- Create new "GitHub API" credential
- Name it (e.g., "My GitHub Account")
- Add your Personal Access Token
- Update all GitHub Trigger nodes to use this credential

Credential 2: HTTP Header Auth (for API Requests)

- Create new "HTTP Header Auth" credential
- Name: "My GitHub Header Auth"
- Header Name: "Authorization"
- Header Value: "Bearer YOUR_TOKEN" or "token YOUR_TOKEN"
- Update all HTTP Request nodes to use this credential

4. Testing Setup

Test Individual Components:

Test 1: Verify Credentials

1. Click on a GitHub Trigger node
2. Click "Test" button
3. Should show successful connection

Test 2: Test HTTP Requests

1. Click on a "Get SHA" node
2. Manually execute the node
3. Should return branch SHA without errors

Test 3: Test Routing

1. Create a test push event
2. Verify workflow routes to correct path
3. Check "Switch Router" output

Test Full Workflow:

Test Scenario:

1. Make a small change in source repository (add a test file)
2. Commit and push to main branch
3. Wait 1-2 minutes
4. Check target repository for:
5. New sync branch created
6. Files synced correctly
7. Pull Request created
8. Expected Result:
9. PR created in target repository
10. PR contains synced files
11. PR title includes source repository name
12. PR body contains commit information

5. Requirements

- n8n instance (cloud or self-hosted).
- GitHub accounts with admin/write access to repositories.
- GitHub Personal Access Tokens with repo and admin:repo_hook permissions.
- Minimum 2 repositories needing bidirectional sync.
- All repositories must have 'main' as default branch.
- Basic knowledge of GitHub, n8n, and webhooks.
- Stable internet connection for webhook delivery.

6. Deliverables

- Fully configured n8n workflow (JSON file).
- 6 bidirectional sync paths (12 total sync directions).
- Automated Pull Request creation in target repositories.
- Loop prevention mechanism to avoid infinite syncs.
- User manual with setup instructions.
- Technical documentation with architecture details.
- Workflow diagram showing complete node structure.

==========