Branch patterns

# Branch patterns#

The relationship between n8n instances and Git branches is flexible. You can create different setups depending on your needs.

Recommendation: don't push and pull to the same n8n instance

You can push work from an instance to a branch, and pull to the same instance. n8n doesn't recommend this. To reduce the risk of merge conflicts and overwriting work, try to create a process where work goes in one direction: either to Git, or from Git, but not both.

## Multiple instances, multiple branches#

This pattern involves having multiple n8n instances, each one linked to its own branch.

You can use this pattern for environments. For example, create two n8n instances, development and production. Link them to their own branches. Push work from your development instance to its branch, do a pull request to move work to the production branch, then pull to the production instance.

The advantages of this pattern are:

• An added safety layer to prevent changes getting into your production environment by mistake. You have to do a pull request in GitHub to copy work between environments.

• It supports more than two instances.

The disadvantage is more manual steps to copy work between environments.

## Multiple instances, one branch#

Use this pattern if you want the same workflows, tags, and variables everywhere, but want to use them in different n8n instances.

You can use this pattern for environments. For example, create two n8n instances, development and production. Link them both to the same branch. Push work from development, and pull it into production.

This pattern is also useful when testing a new version of n8n: you can create a new n8n instance with the new version, connect it to the Git branch and test it, while your production instance remains on the older version until you're confident it's safe to upgrade.

The advantage of this pattern is that work is instantly available to other environments when you push from one instance.

The disadvantages are:

• If you push by mistake, there is a risk the work will make it into your production instance. If you use a GitHub Action to automate pulls to production, you must either use the multi-instance, multi-branch pattern, or be careful to never push work that you don't want in production.

• Pushing and pulling to the same instance can cause data loss as changes are overridden when performing these actions. You should set up processes to ensure content flows in one direction.

## One instance, multiple branches#

The instance owner can change which Git branch connects to the instance. The full setup in this case is likely to be a Multiple instances, multiple branches pattern, but with one instance switching between branches.

This is useful to review work. For example, different users could work on their own instance and push to their own branch. The reviewer could work in a review instance, and switch between branches to load work from different users.

No cleanup

n8n doesn't clean up the existing contents of an instance when changing branches. Switching branches in this pattern results in all the workflows from each branch being in your instance.

## One instance, one branch#

This is the simplest pattern.