Branching Strategy for Development, Staging, Production Releases, and Hotfixes

Branching Strategy

This document describes the branching workflow used for development, staging validation, production releases, and hotfixes.

Branch Names

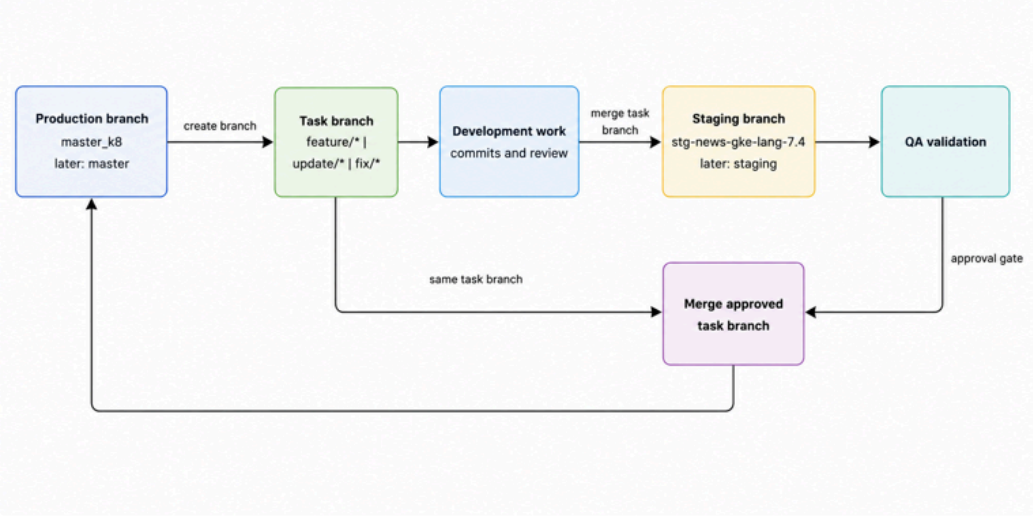
The branch names are being simplified. Until the rename is completed, use the current branch names shown below.

Purpose	Current branch name	Future branch name
Production	`master_k8`	`master`
Staging / QA	`stg-news-gke-lang-7.4`	`staging`

In this document:

- **Production branch** means `master_k8` now and `master` after the rename.
- **Staging branch** means `stg-news-gke-lang-7.4` now and `staging` after the rename.

Task Branch Flow



For each task:

1. Create a task branch from the production branch.

2. Complete development work and code review on the task branch.
3. Merge the task branch into the staging branch.
4. Validate the change in the staging environment.
5. After QA approval, merge the same task branch into the production branch.
6. Delete the task branch after the production release is stable.

Do not merge the staging branch into the production branch for an individual task release. The approved task branch is merged into production so unrelated staging changes are not released accidentally.

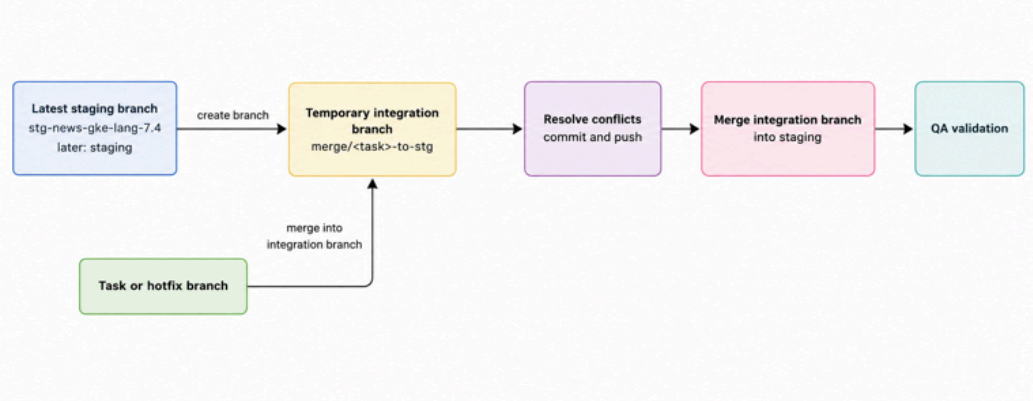
Task Branch Naming

Use a short-lived branch with a lowercase prefix and a ticket ID where one is available.

Branch pattern	Purpose	Example
`feature/<ticket>-<description>`	New feature	`feature/NEWS-57910-liveblog`
`hotfix/<ticket>-<description>`	Urgent production fix	`hotfix/NEWS-12345-publish-issue`
`merge/<task>-to-stg`	Resolve staging merge conflicts	`merge/hotfix-tag-deplyoment-to-stg`

Staging Conflict Resolution

If a task or hotfix branch conflicts with the staging branch, create a temporary integration branch from the latest staging branch. Merge the task branch into the integration branch and resolve conflicts there.



Example used for `hotfix/tag_deplyoment`:

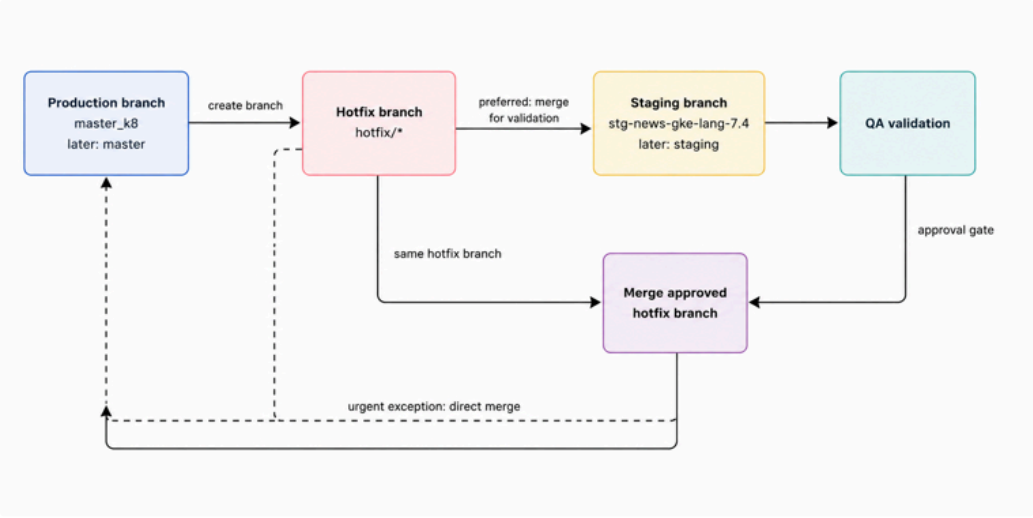
```
git fetch origin
git checkout stg-news-gke-lang-7.4
git pull origin stg-news-gke-lang-7.4
git checkout -b merge/hotfix-tag-deplyoment-to-stg
git merge origin hotfix/tag_deplyoment
```

After running the commands:

1. Resolve any conflicts on `merge/hotfix-tag-deplyoment-to-stg`.
2. Commit and push the integration branch.
3. Open a merge request from the integration branch into the staging branch.
4. Complete QA validation after the integration branch is merged into staging.
5. For the production release, merge the original approved task or hotfix branch into the production branch. Do not merge the staging integration branch into production.

Hotfix Flow

Hotfixes normally follow the same validation flow as task ranches:



For an urgent production issue, the hotfix branch may be merged directly into the production branch. When the direct path is used, merge the same hotfix branch into the staging branch afterward so the environments remain aligned.

Release Checklist

1. Confirm that the task branch was created from the latest production branch.
2. Merge the task branch into the staging branch.
3. Complete QA validation in staging.
4. Merge the same approved task branch into the production branch.
5. For a direct production hotfix, sync the hotfix branch back into staging.
6. Use a temporary `merge/<task>-to-stg` branch if the staging merge has conflicts.
7. Create a release tag when required by the deployment process.