Branching & Commits

Last updated by | Daniels, Steve | Sep 8, 2025 at 2:08 PM GMT+5:30

Contents

Git strategy

We follow releaseflow ~~> strategy

Commit style

Commits must be in the following format:

```
#<ADO Work Item number> brief-description
```

ie.

```
#1234 Brief description of the app
```

Branching strategy

All branches must be from `main` and merge to `main` (excluding `release` branches see: Parallel delivery with independent release)

Feature branches

Development is done using feature branches typically following a naming convention of one of the following formats

```
feature/#<ADO Work Item number>-brief-description-separated-by-dashes
```

i.e `feature/#1234-some-description`

OR

```
feature/#<ADO Work Item number>
```

i.e `feature/#1234`

Bugfix branches

Bugfix branches must be in on of the following formats

```
bugfix/#<ADO Work Item number>-brief-description-separated-by-dashes
```

i.e.

```
bugfix/#1234-some-description
```

OR

```
bugfix/#<ADO Work Item number>
```

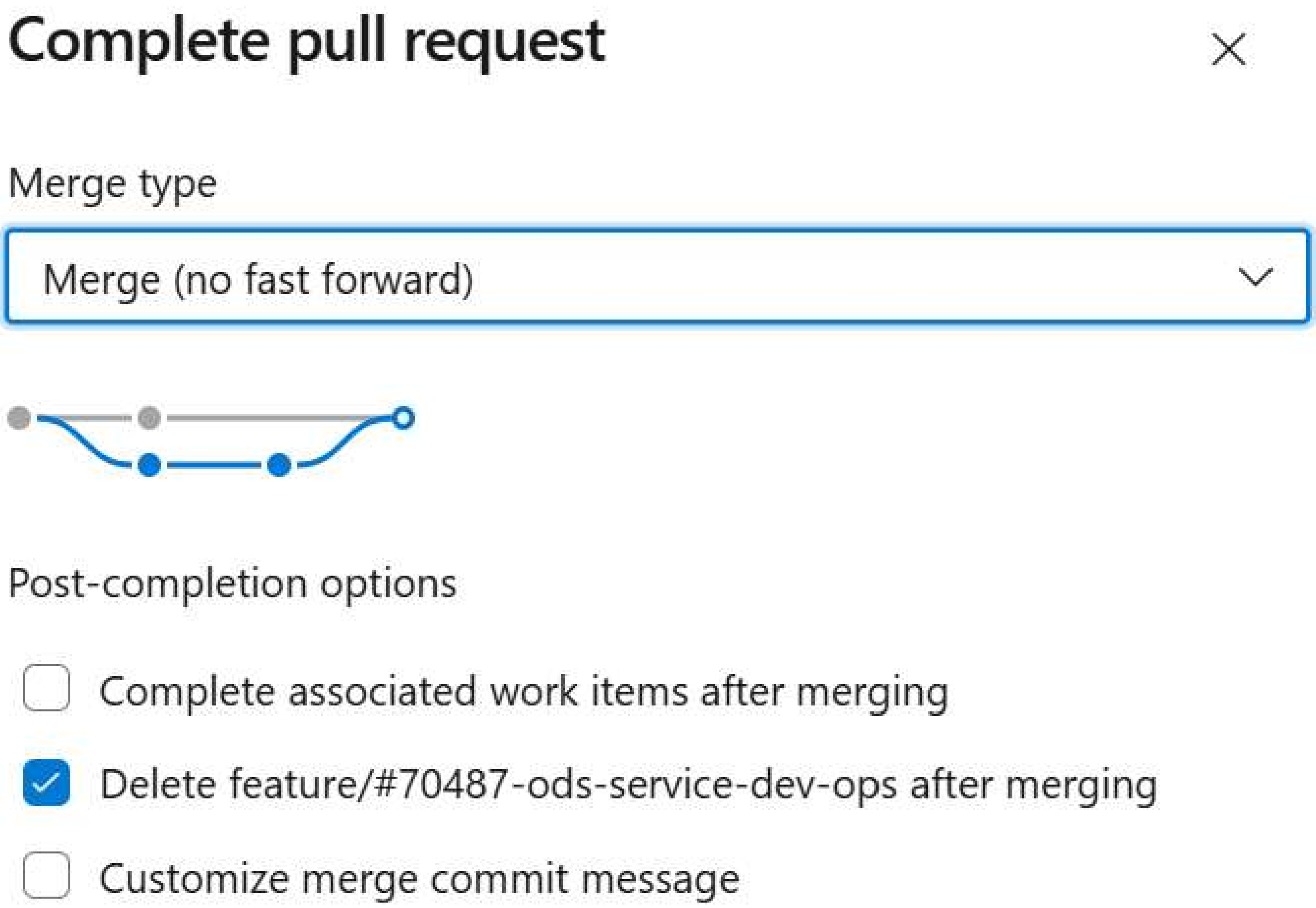
i.e `bugfix/#1234`

Parallel delivery with independent release

If we need to support parallel where we need to hold back work from a release:

1. Create a `release/<project-name>` branch from `main`
2. Create PRs from `release/<project-name>` (see Feature branches & Bugfix branches)
3. Merge into `release/<project-name>`
4. When ready to release create a merge from `release/<project-name>` into `main` and release

Merging Strategy



Complete pull request

Merge type

Merge (no fast forward)

Post-completion options

□ Complete associated work items after merging

☑ Delete feature/#70487-ods-service-dev-ops after merging

□ Customize merge commit message