Whats New?

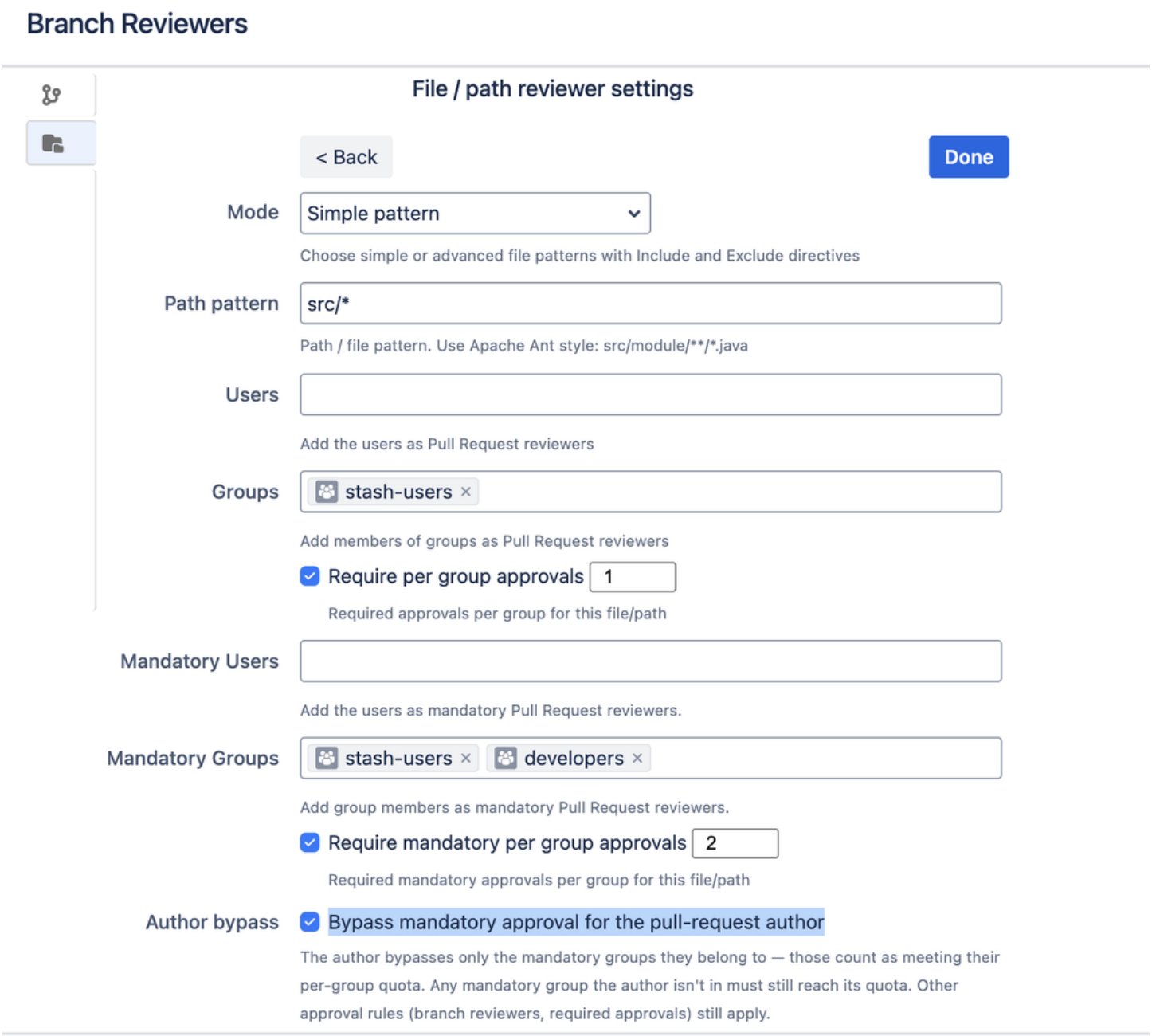
More Resources Workzone · https://izymes.com/docs/whats-new/

2026-09-16 - Mandatory reviewer approval bypass for pull request author

When a mandatory reviewer authors a pull request that changes a file they are responsible for, they cannot approve their own work. For a rule with only a few eligible reviewers, that can leave the pull request blocked until someone else is available.

Workzone now offers an optional **Author bypass** setting for individual file/path reviewer rules. Enable it under **Reviewers → Files/Paths**. If the author is a listed mandatory reviewer, their rule’s mandatory-user approval requirement is treated as met. For per-group quotas, the bypass applies only to mandatory groups the author belongs to; other groups must still meet their quotas.

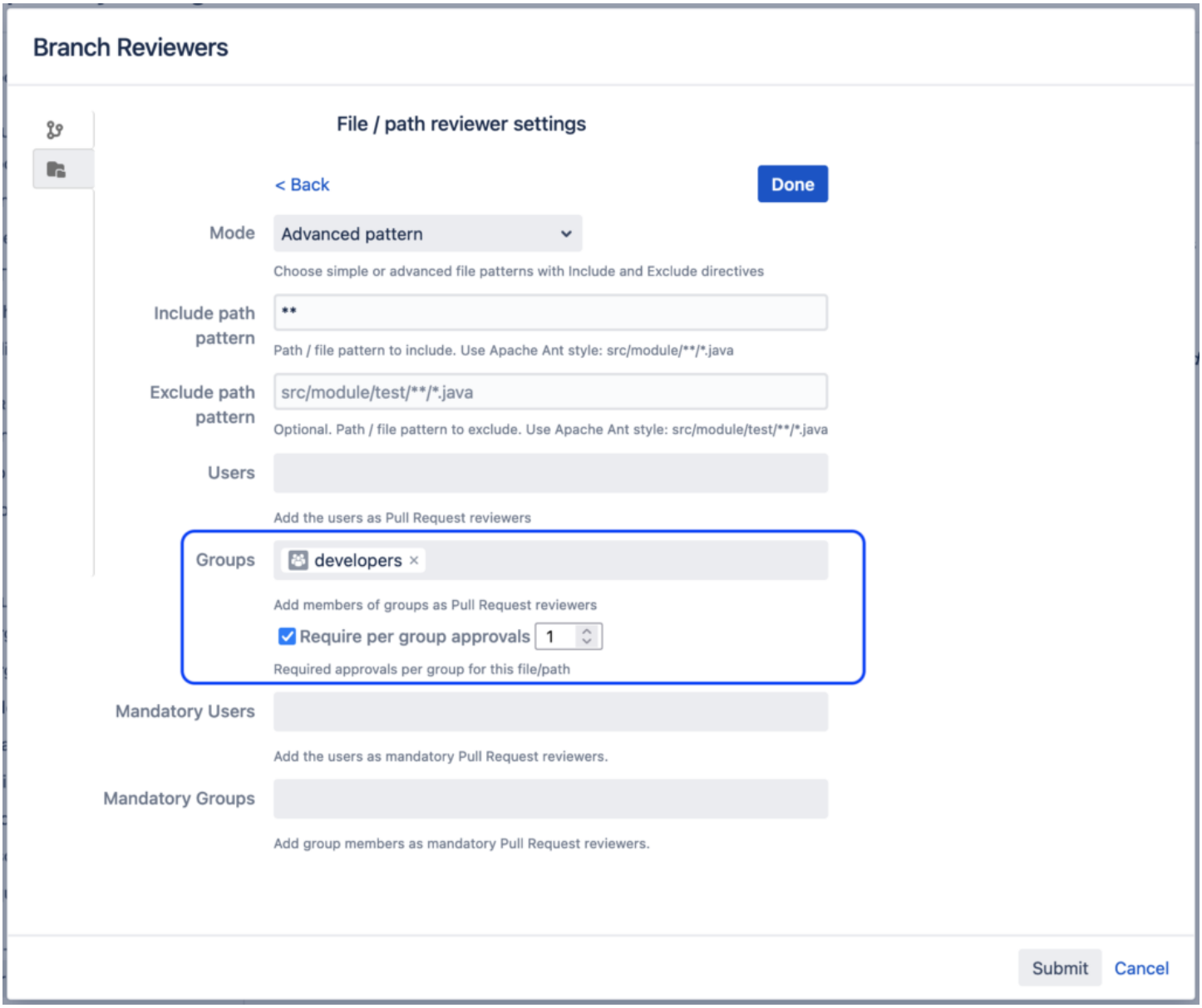
The setting is off by default and affects only the file/path rule where it is enabled. Other file/path rules and separately configured approval requirements still apply, so teams can resolve this reviewer deadlock without relaxing review requirements across the branch.



2023-05-19 - Additional merge condition token filePathMergeChecksPass

The Workzone advanced merge condition expression can now incorporate a new condition called **filePathMergeChecksPass == true**. Use **filePathMergeChecks == true** in the Workzone Boolean Expression to evaluate file/path reviewer quota checks explicitly. This condition comes into play if the Workzone reviewer file/path configuration has also required approvals per file/path defined.

For example the Workzone reviewer configuration may contain file/path reviewers with required approval quotas:



These required reviewer approvals are *implied* when Workzone evaluates the pull request merge condition. A Workzone merge condition like `approvalQuota >= 1% & requiredBuildsCount >= 1` would effectively read as **`( approvalQuota >= 1% & requiredBuildsCount >= 1) & filePathMergeChecksPass == true`** Workzone enforces that all necessary file/path reviewer approvals must be satisfied on top of the given merge condition.

In order to include the file/path merge condition check in the merge condition's boolean expression include **`filePathMergeChecksPass == true`** into the advanced merge condition. For example to bypass required file/path reviewer approvals with a digital signature approval use `(filePathMergeChecksPass == true & requiredBuildsCount >= 1) | requiredSignaturesCount >= 1`



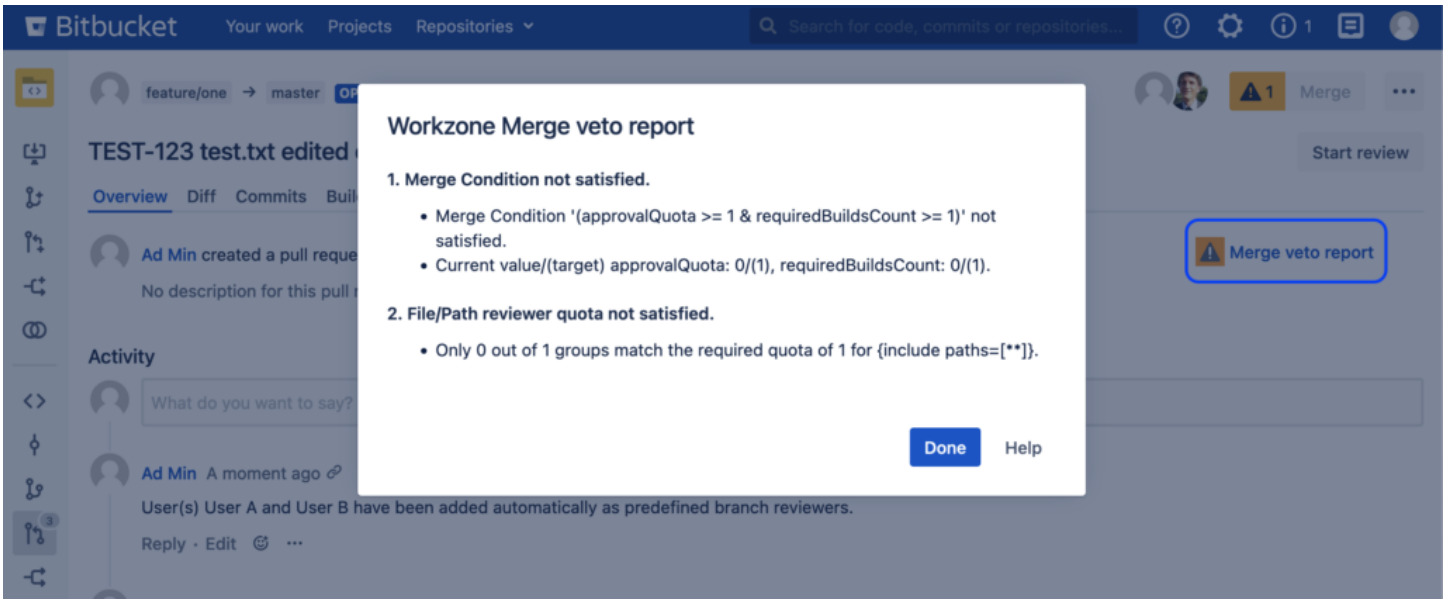
2023-05-19 - Additional permission checks for digital signature approvals

The additional security filter makes sure that a reviewer that must approve with a digitally signature can not approve the pull request by quickly clicking on the pull request approve button.

Embedded media: https://izymes.com/wp-content/uploads/2023/05/user-must-sign-to-approve-720.mov

2023-05-19 - Detailed Workzone merge veto report

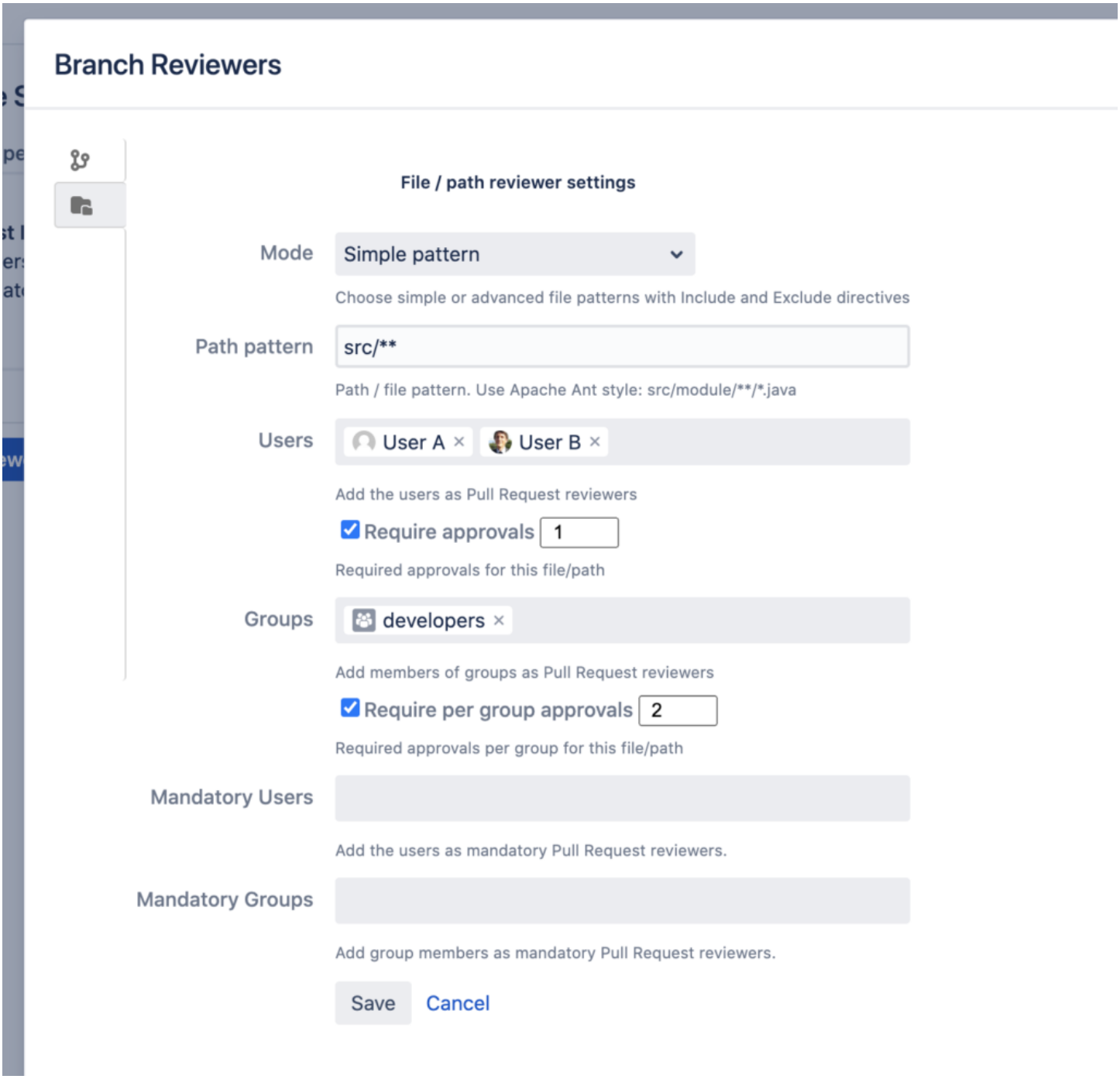
If Workzone vetoes a pull request merge a **detailed report** is accessible via the pull request page actions side bar. The merge veto report contains all details as to why the current pull request cannot be merged, including missing file/path reviewer approvals:



2023-02-16 - Configure required File/Path reviewer approval count

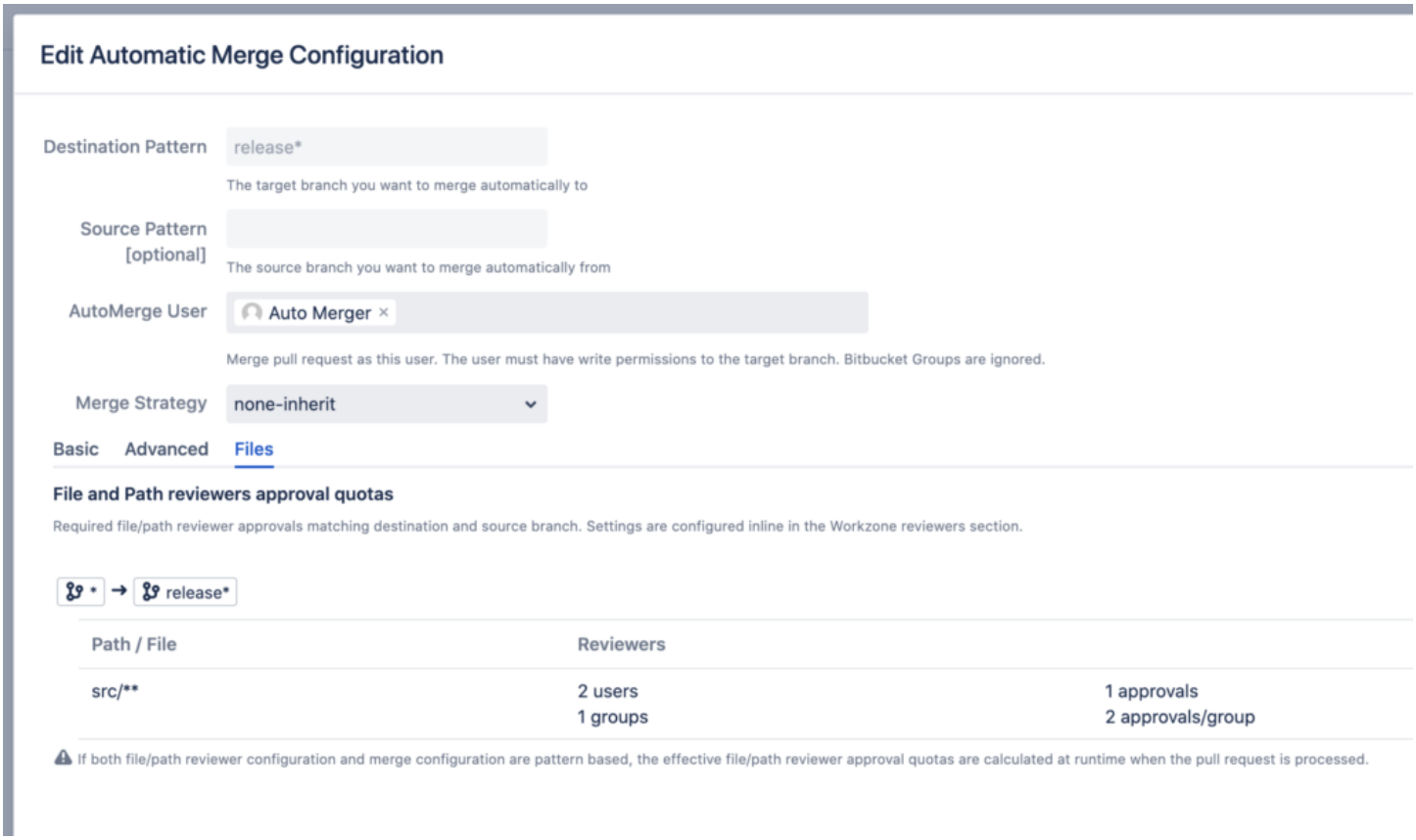
Required approval counts for file/path reviewers and groups can now be configured in Workzone Reviewers > Files/Paths.

In file/path reviewers configuration enter a user or group and check 'Require approvals' to enable an approval quota for this file/path only.



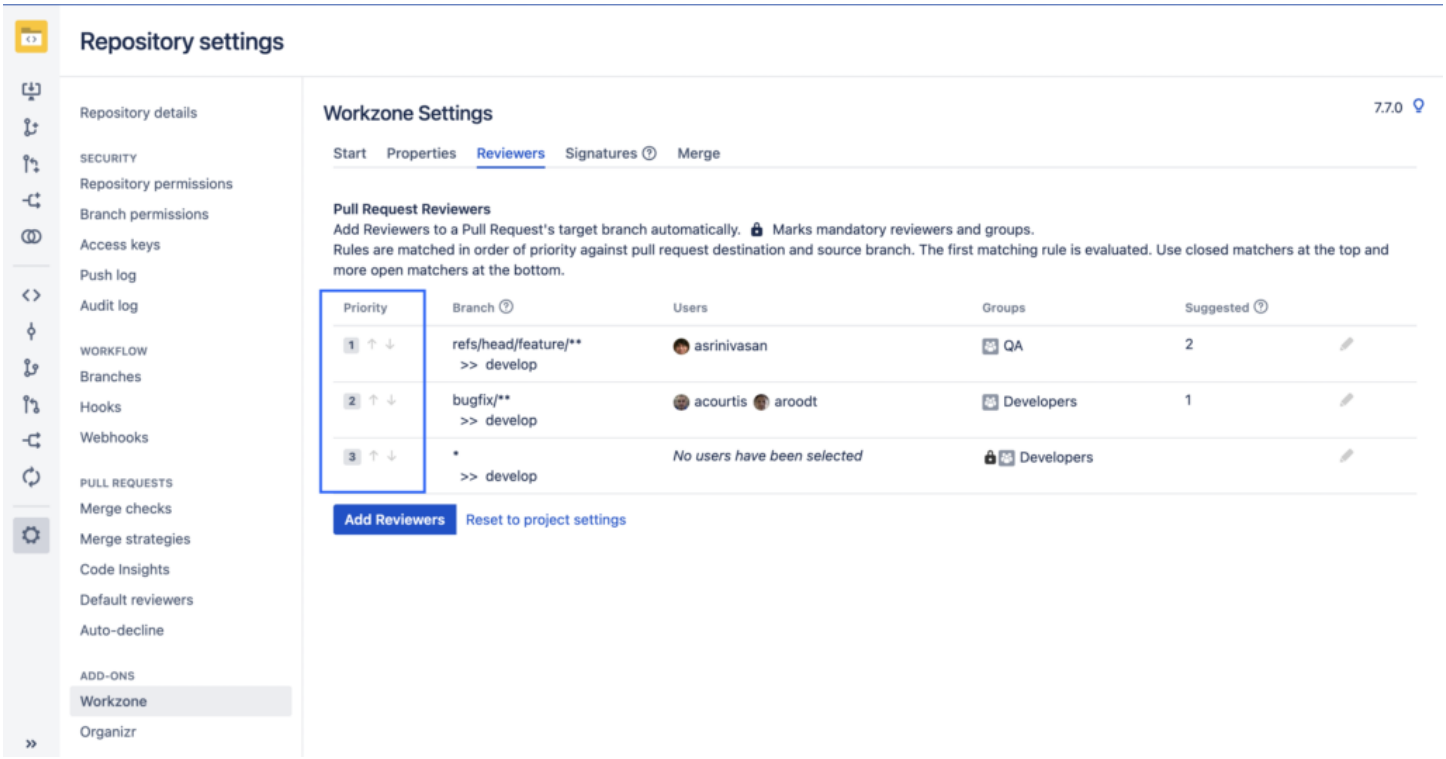
Set required file/path approvals in reviewer configuration

File/Path quotas will be evaluated as merge-checks along with all other Workzone merge conditions for matching pull requests.



Required file/path approvals as additional merge checks for matching pull requests.

2022-11-03 - Setting priority for reviewers, signatures & merge



Within the Project Settings and Repository Settings, you can change the priority order of reviewers, signatures, and merge conditions. To do this, simply click the up or down arrow. The changes are saved automatically.

The first rule that matches a pull request’s source and destination branch will be executed. Place rules with close or direct branch matching expressions at the top of the list and the catch-all rule at the bottom. An example can be seen in the screenshot above.

Another example may include:

1. (release/current > main) >> reviewers: team1, qa-group

2. (release/* > main) >> reviewers: qa-group

3. (> main) >> reviewers: bot-reviewer

In this case, if someone creates a pull request from any branch other than release/* a bot user gets assigned. Any pull request from release/current gets reviewers groups team1 and qa-group, but if the pull request source branch matches anything else then release/current only qa-group is added as reviewers. All reviewer, signature approval, and (auto)merge rules follow the same match by priority approach.

Try Workzone!