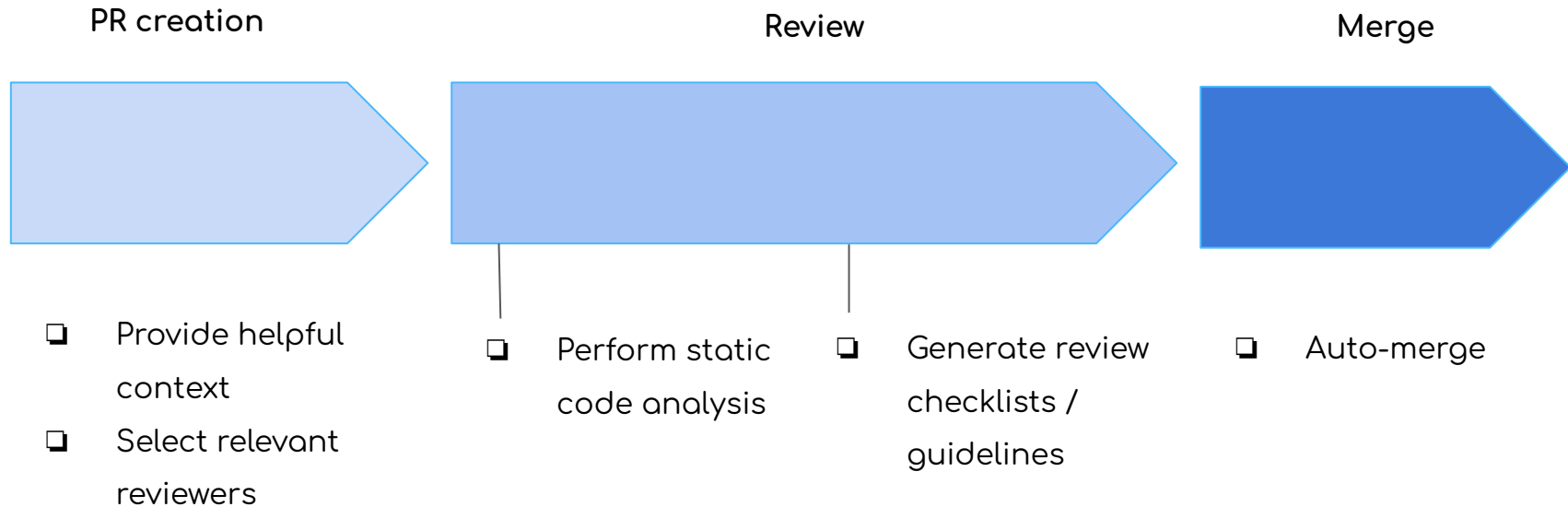


AUTOMATE TO ELEVATE

OPTIMIZING YOUR PULL REQUEST WORKFLOW

CODE BETTER, MERGE FASTER

The different stages of a Pull Request workflow with possibly automated steps



PR creation - Provide helpful context




Problems

- ❑ Delays to understand clearly the code changes purpose and implementation (when PR titles and descriptions are not explicit)
- ❑ Inconsistency in format and content details

How automation helps

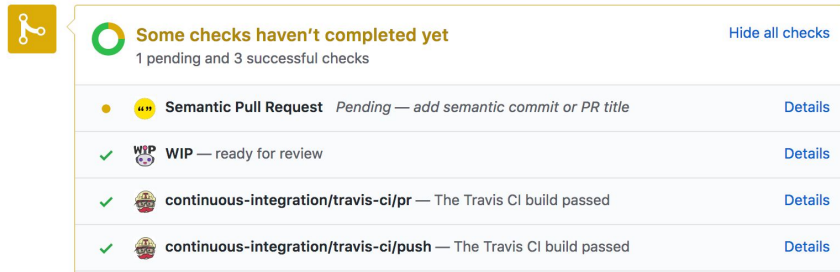
- ❑ PR title validation
Add checks to ensure PR titles follow a defined format
- ❑ PR title generation
Get AI-powered recommendations based on commit messages, code diff, issues
- ❑ PR description template
Create template to provide structure and guidelines for descriptions
- ❑ PR description generation
Get AI-powered recommendations based on commit messages, code diff, issues

PR creation - Provide helpful context

	Agnostic solution			
PR title validation	add job in CI/CD system to enforce formatting rules (GitLab ex)	Semantic Pull Request for semantic titles PR Title Checker	n.a.	n.a.
PR title generation	n.a.	pr-agent (plugin)	n.a.	code review assistant (plugin)
PR description template	add job in CI/CD system to enforce formatting rules	pull request template (native feature)	merge request template (native feature)	pull request description (native feature)
PR description generation	n.a.	GitHub copilot (native feature) pr-agent , coderabbit AI (plugin)	GitLab duo (native feature) coderabbit AI (plugin)	Code Review Assistant (plugin)

PR creation - Provide helpful context - example (Semantics PR + Copilot x GitHub)

title validation



A screenshot of a GitHub pull request interface. At the top, a yellow box with a GitHub logo icon says "Some checks haven't completed yet" with a link to "Hide all checks" and a note "1 pending and 3 successful checks". Below this is a table of checks:

Check Status	Check Name	Details
Pending	Semantic Pull Request — add semantic commit or PR title	Details
Passed	WIP — ready for review	Details
Passed	continuous-integration/travis-ci/pr — The Travis CI build passed	Details
Passed	continuous-integration/travis-ci/push — The Travis CI build passed	Details

The PR title or at least one commit message needs to follow the Conventional Commits spec.

`<type>[optional scope]: <description>`

`[optional body]`

`[optional footer(s)]`

Examples:

feat: allow provided config object to extend other configs

fix: prevent endless loop

docs: correct spelling of README

Add a title

My pull request

Auto-generate

description

Add a description

Write

Preview



H

B

I

≡

<>

🔗

≡

≡

≡

Add your description here...

Summary

Generate a summary of the changes in this pull request

PR creation - Auto-assign reviewers

Problems

- ❑ Delay incurred from reviewers either not familiar with app design, code changes purpose and objective or without the skills needed
- ❑ Unfocusing team members that are notified for reviews they are not relevant to perform

How automation helps

- ❑ Assignment of relevant reviewer for specific code changes using the **Code Owners** concept
- => Relying on the Code Owners concept, defining users as owners for specific folders, files or file types that will be automatically added as reviewers when the conditions are met by a PR

PR creation - Auto-assign reviewers



Auto-assign reviewers

Assign reviewers based on defined ownership

[Code Owners](#) (native feature)

[Code Owners](#) (native feature)

[Code Owners](#) (native feature)

[DevSensei](#) (plugin)

Random assignment

Assign random subset of owners

[Code Owners](#) (native feature)

n.a.

[DevSensei](#) (plugin)

Fallback reviewers

Allow review by fallback users if owners are unavailable

n.a.

n.a.

[DevSensei](#) (plugin)

Minimum number of approval

flexible merge checks, requiring only a specific amount of approvals among the code owners

n.a.

[Code Owners](#) (native feature)

[DevSensei](#) (plugin)

PR creation - Auto-assign reviewers - example (GitHub)

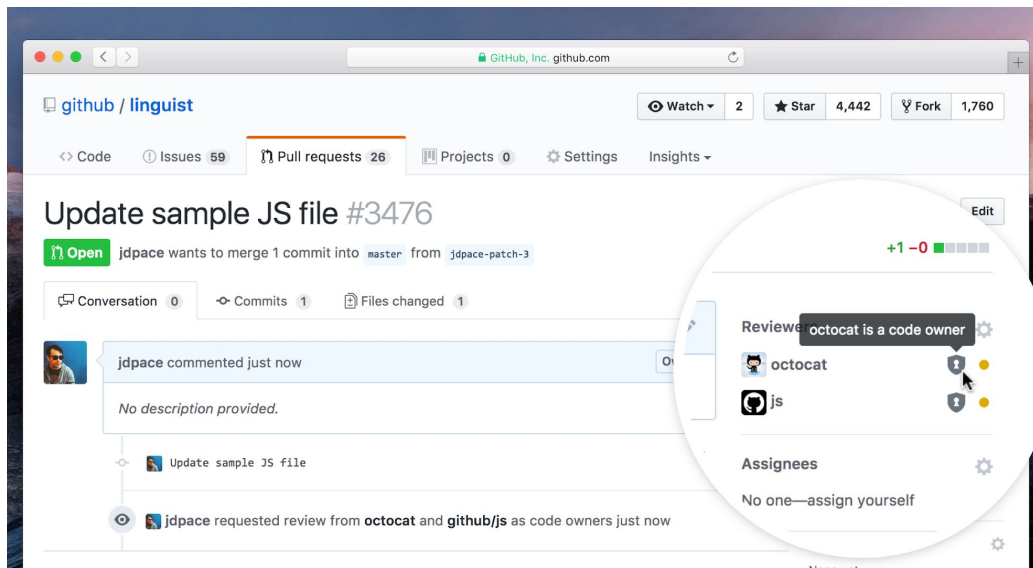
CODEOWNERS file

```
# Lines starting with '#' are comments.
# Each line is a file pattern followed by one
# or more # owners.

# These owners will be the default owners for
# everything in the repo.
* @defunkt

# Order is important. The last matching
# pattern has
# the most precedence.
# So if a pull request only touches javascript
# files,
# only these owners will be requested to
# review.
*.js @octocat @github/js

# You can also use email addresses if you
# prefer.
docs/* docs@example.com
```



PR review - Static Code Analysis




Problems

- ❑ Code quality and style consistency
- ❑ Vulnerability from third party dependencies

How automation helps

- ❑ Trigger automated code analyses and generate actionable reports
- ❑ Enforce code quality through merge checks based on code analyses results


PR review - Static Code Analysis

	Agnostic solution			
code safety and quality checks	Codacy , Sonar , Snyk (to name a few SaaS)	Codacy , Sonar , Snyk (integrations)	Code Quality (native feature) Codacy , Sonar , Snyk	Codacy (integration)
Code style	ESLint (Javascript), Checkstyle , PMD	ESLint annotate from Report , Annotate Pull Request from Checkstyle (actions)	Code Quality (native feature)	Code Review Assistant (plugin)
Dependencies vulnerability	Sonatype OSS (enterprise) OWASP dependency-check (open source)	dependency check (action)	Code Quality , Dependency Scanning (native feature)	Code Review Assistant (plugin)
Secret detection	GitGuardian , Spectrol (SaaS) gitLeaks (open source)	Secret scanning , AI-powered secret detection (native feature)	Secret Detection (native feature)	Secret scanning (native feature)

PR review - Static Code Analysis - example (Codacy x GitHub)

version-select.js

```
13 + window.versionPages[version] = [];  
14 +  
15 + var xhrSitemap = new XMLHttpRequest;  
16 + var sitemapURL = window.location.origin + versionPath + '/sitemap.xml';
```



 **codacy-producti...** (bot) 2 minutes ago



Codacy has a fix for the issue: [Strings must use doublequote.](#)


Suggested change ⓘ

```
16 - var sitemapURL = window.location.origin + versionPath + '/sitemap.xml';  
16 + var sitemapURL = window.location.origin + versionPath + "/sitemap.xml";
```


[Commit suggestion](#) [Add suggestion to batch](#)

  **All checks have passed** [Hide all checks](#)
1 successful check

  **Codacy Static Code Analysis** Successful in 2m — Codacy Static Code Analysis [Details](#)



 **This branch has no conflicts with the base branch**
Merging can be performed automatically.

[Merge pull request](#) or view [command line instructions](#).

 **codacy-producti...** (bot) commented 5 minutes ago

Coverage summary from Codacy

Merging #12 ([106ba10](#)) into main ([63c3f4a](#)) - See PR on Codacy

Coverage variation	Diff coverage
 -26.67% (target: +0.00%)	 66.67% (target: 62.00%)

▼ Coverage variation details

	Coverable lines	Covered lines	Coverage
Common ancestor commit (63c3f4a)	12	8	66.67%
Head commit (106ba10)	15 (+3)	6 (-2)	40.00% (-26.67%)

Coverage variation is the difference between the coverage for the head and common ancestor commits of the pull request branch: $\text{coverage of head commit} - \text{coverage of common ancestor commit}$

▼ Diff coverage details

	Coverable lines	Covered lines	Diff coverage
Pull request (#12)	3	2	66.67%

Diff coverage is the percentage of lines that are covered by tests out of the coverable lines that the pull request added or modified: $\frac{\text{covered lines added or modified}}{\text{coverable lines added or modified}} * 100\%$

[See your quality gate settings](#) [Change summary preferences](#)

PR review - Guidelines and Checklists

Problems

- ❑ Reviewers may overlook certain topics during their review
- ❑ Hard to ensure that a review has properly been made

How automation helps

- ❑ Add description templates with general guidelines
- ❑ Add checklists to guide reviewers and audit reviews

PR review - Guidelines and Checklists

Agnostic solution



Guidelines /
checklists

Add conditional
comments through a
CI/CD job ([GitHub](#) ex)

[Mergify](#), [Pull Request
Checklist Buddy](#), [Pull
Checklist](#) (plugin)

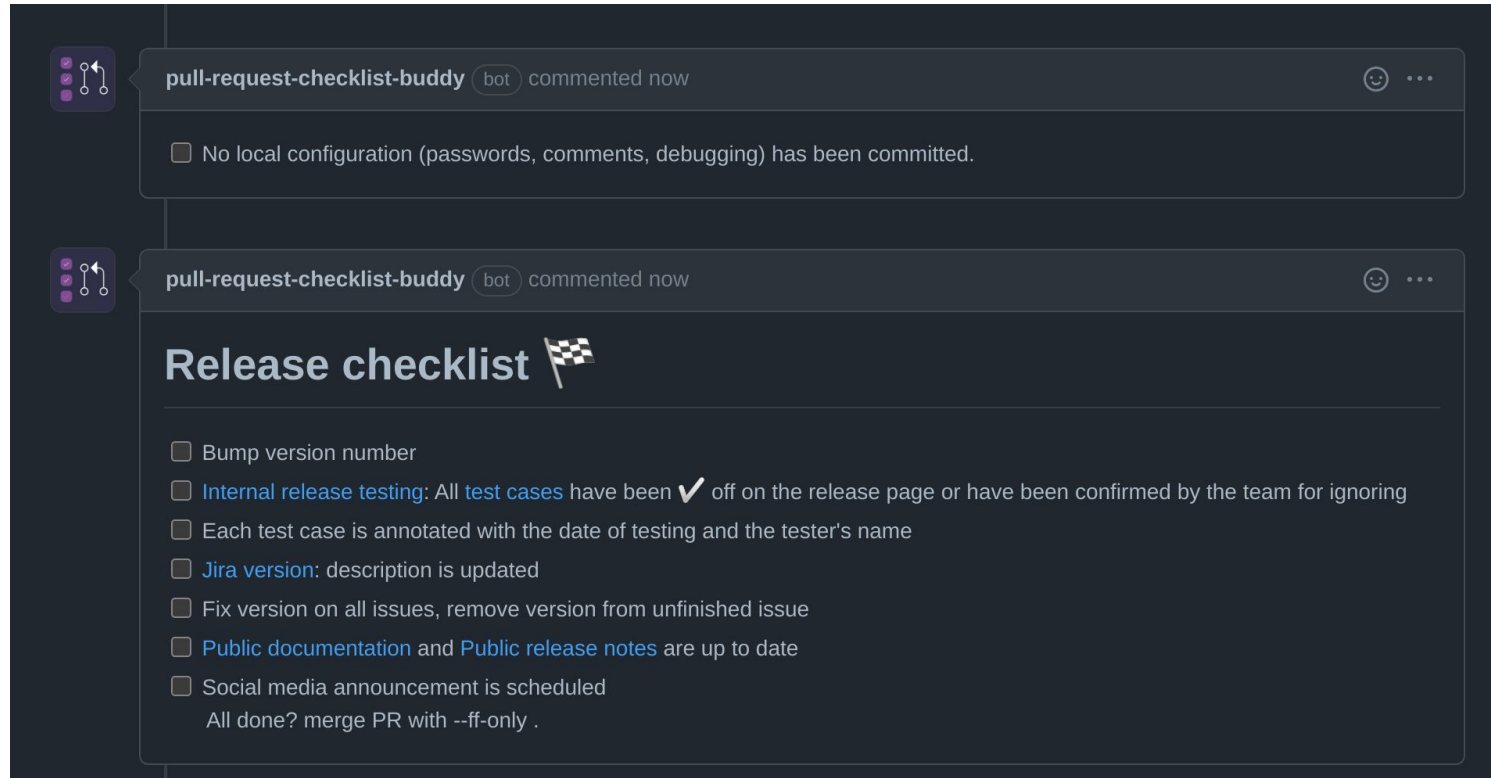


n.a.



[Pull Request Checklist
Buddy](#) (plugin)

PR review - Guidelines and Checklists - example (Checklist Buddy x GitHub)



The screenshot shows two comments from the 'pull-request-checklist-buddy' bot on a GitHub pull request. The first comment contains a single checklist item. The second comment is titled 'Release checklist' and contains a list of items, some of which are marked as complete.

pull-request-checklist-buddy (bot) commented now

- ☐ No local configuration (passwords, comments, debugging) has been committed.

pull-request-checklist-buddy (bot) commented now

Release checklist 🚩

- ☐ Bump version number
- ☐ **Internal release testing:** All **test cases** have been ☒ off on the release page or have been confirmed by the team for ignoring
- ☐ Each test case is annotated with the date of testing and the tester's name
- ☐ **Jira version:** description is updated
- ☐ Fix version on all issues, remove version from unfinished issue
- ☐ **Public documentation** and **Public release notes** are up to date
- ☐ Social media announcement is scheduled

All done? merge PR with --ff-only .

PR merge - Auto-merge




Problems

- ❑ Manual work to merge Pull Requests meaning they can stay unnecessarily stale
- ❑ Superfluous review time wasted on Pull Request that could already have been merge
- ❑ Some Pull Requests may not need to be manually reviewed at all and just need to go through the CI/CD checks (ex: dependency upgrade)

How automation helps

- ❑ Increase development velocity by enabling auto-merge for a pull request so that the pull request will merge automatically when all merge requirements are met
- ❑ Implement Ship/Show/Ask workflow

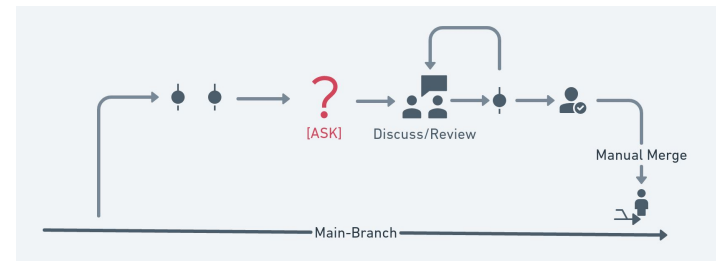
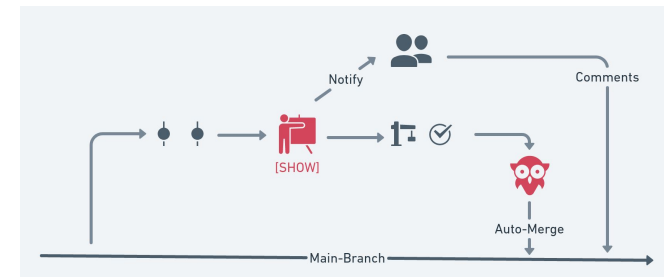
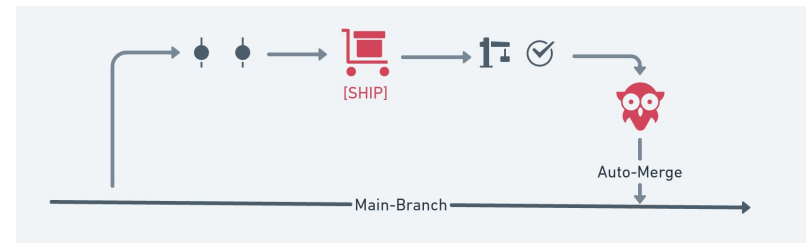
PR merge - Auto-merge

	Agnostic solution			
Auto-merge	create custom CI/CD job	automerger (native feature) Mergify (plugin)	automerger (native feature)	automerger (native feature) DevSensei (plugin)
auto rebase PRs	create custom CI/CD job	Automatic Rebase (action) Mergify (plugin)	n.a.	n.a.

PR merge - Ship / Show / Ask principle (helpful for auto-merging strategies)

Not every PR is equal:

- **Ship:** for small changes that don't require people to review, nor even to be aware of (e.g. fixing typos)
- **Show:** you want to show what has been done, but otherwise you would like it to be merged automatically. You still want people to be notified.
- **Ask:** when you want feedback and discussions on the changes you made (classical PR).



CASE STUDIES IN ACTION



GitHub example:

<https://www.youtube.com/watch?v=0l-VI3Tr5mA>

Features shown:

- auto-assign reviewers
- pull request description template
- reviewer checklist
- static code analysis
- auto-merge

Bitbucket example:

https://www.youtube.com/watch?v=u6_QOa_yyS4

Features shown:

- auto-assign reviewers
- AI-suggested pull request title and description
- reviewer checklist
- static code analysis
- auto-merge

Links to the tools used for each use case are provided in the video descriptions

Try it yourself!

A nice playground can be to clone the open source repository for mermaid

<https://github.com/mermaid-js/mermaid>

It contains already description templates for Pull Request and CI/CD actions like

- ❑ static code quality analysis
 - ❑ linter
 - ❑ dependency check
- ❑ Pull Request labelling for release

You can for example, to replicate the example given in the video:

- ❑ add a [CODEOWNERS](#) file for auto-assigning reviewers
- ❑ add [Semantic PR](#) for validation of PR titles and commits
- ❑ customize the [Pull Request description template](#) file
- ❑ add your own checklists with [Pull Request Checklist Buddy](#)
- ❑ add [Codacy](#) to replace the existing code analysis actions
- ❑ add [Mergify](#) for testing auto-merge workflows