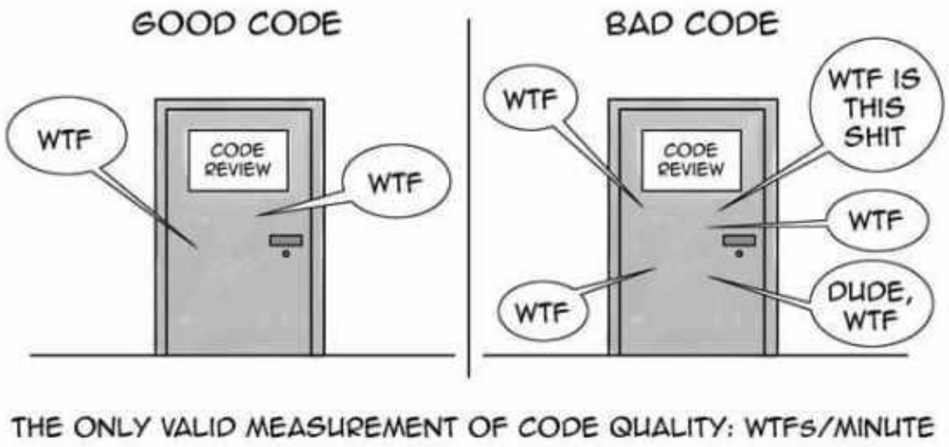
Code Review

Prof. Sven Apel

Universität des Saarlandes

Code Review

Code review is a software development practice, where code is reviewed by other people to detect problems. Usually the review is done before the introduction of the code into a codebase.



THE ONLY VALID MEASUREMENT OF CODE QUALITY: WTFs/MINUTE

Types of Code Review

Greenfield reviews

- Completely *new code* gets added to the codebase

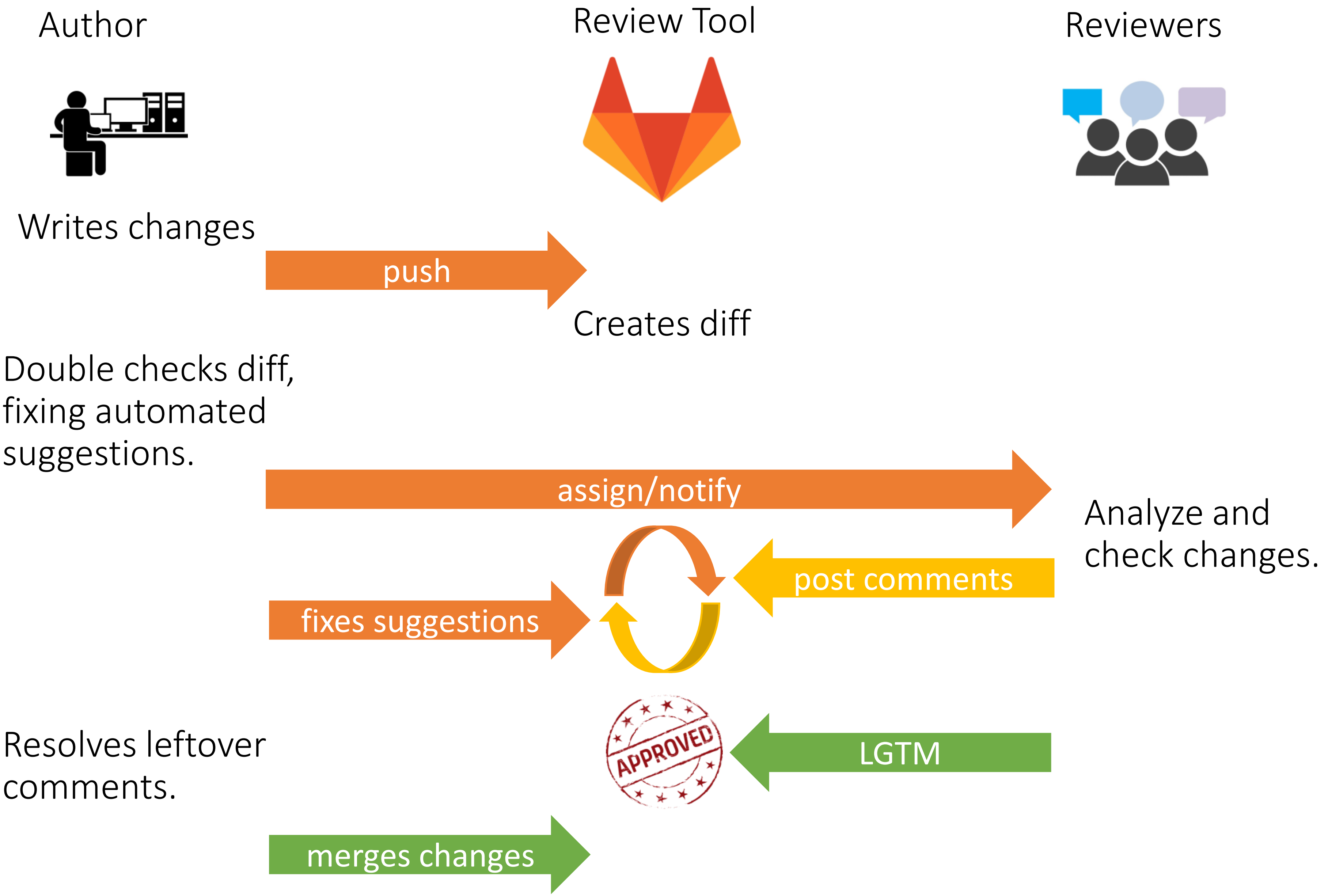
Changes, improvements and optimizations

- Modifies *existing* APIs and code

Bug fixes

- *Isolated* changes to fix specific problems

Code Review Process



Author

Review Tool

Reviewers

Writes changes

push

Creates diff

Double checks diff,
fixing automated
suggestions.

assign/notify

Analyze and
check changes.

post comments

fixes suggestions

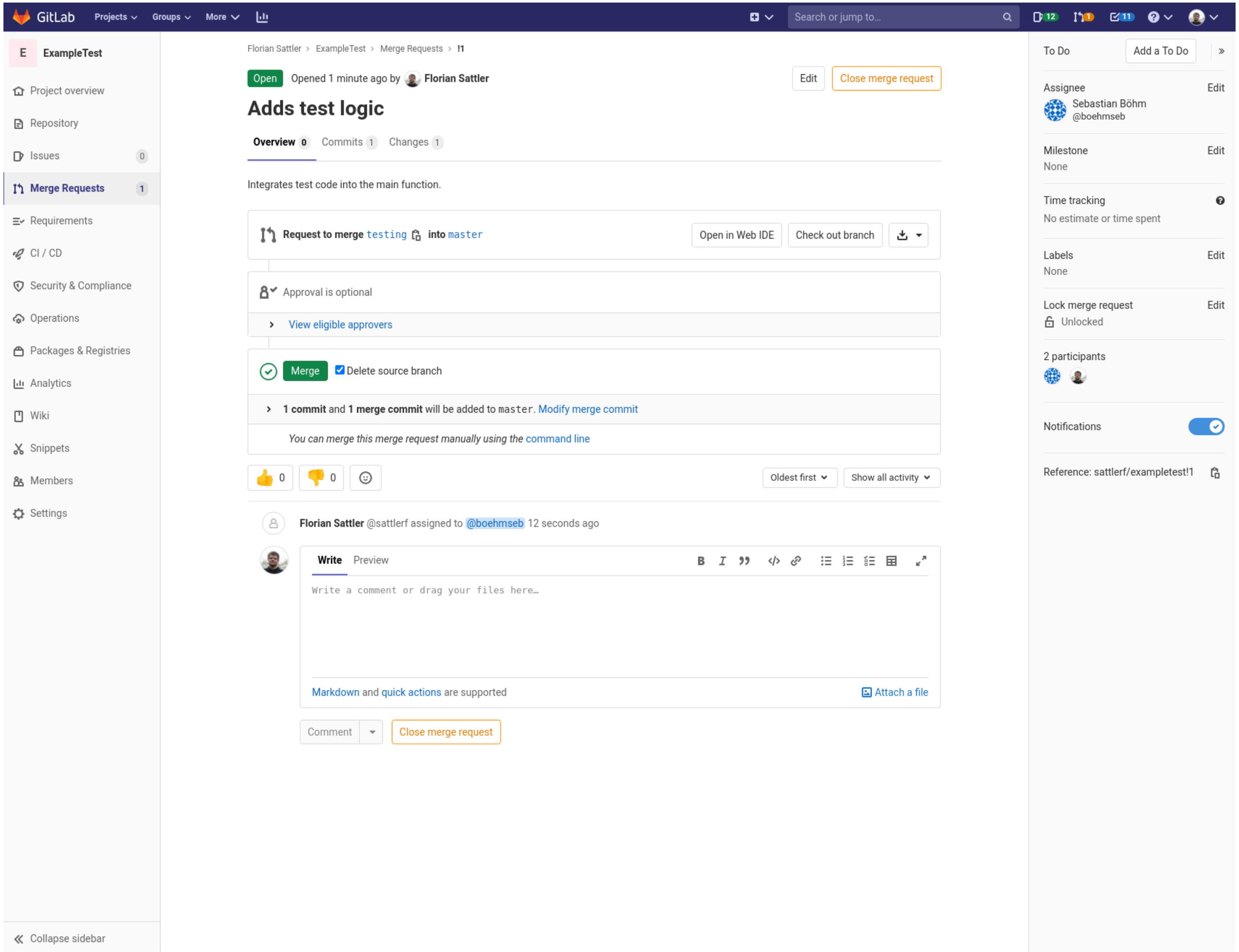
APPROVED

LGTM

Resolves leftover
comments.

merges changes

Code Review Process



Florian Sattler > ExampleTest > Merge Requests > !1

Open Opened 1 minute ago by Florian Sattler

Adds test logic

Integrates test code into the main function.

Request to merge `testing` into `master`

Approval is optional

☑ Delete source branch

1 commit and 1 merge commit will be added to `master`. Modify merge commit

You can merge this merge request manually using the command line

Florian Sattler @sattlerf assigned to @boehmseb 12 seconds ago

Markdown and quick actions are supported

Assignee
Sebastian Böhm
@boehmseb

Milestone
None

Time tracking
No estimate or time spent

Labels
None

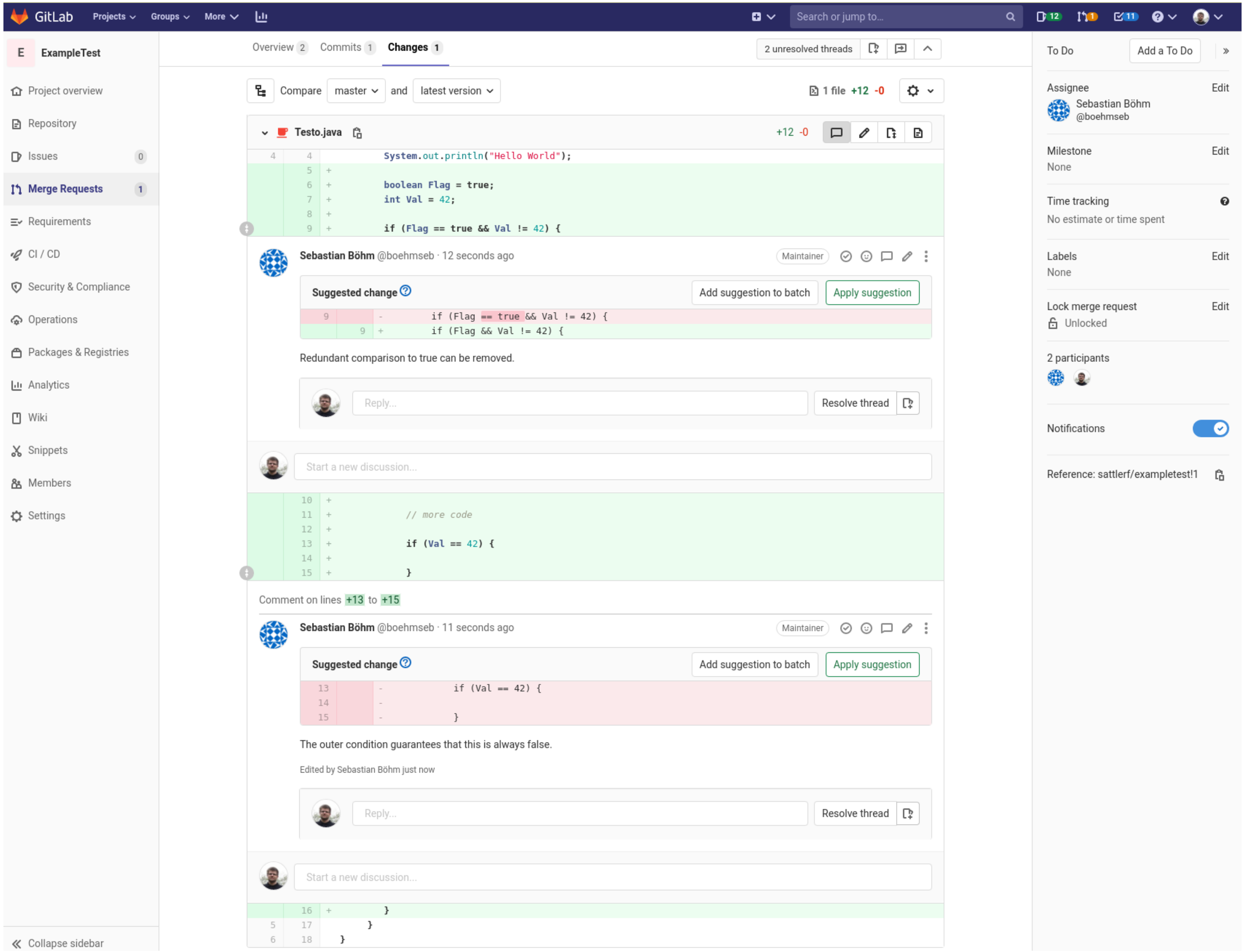
Lock merge request
Unlocked

2 participants

Notifications

Reference: sattlerf/exampletest!1

Code Review Process



```
4    4          System.out.println("Hello World");
     5   +
     6   +      boolean Flag = true;
     7   +      int Val = 42;
     8   +
     9   +      if (Flag == true && Val != 42) {
```

Sebastian Böhm @boehmseb · 12 seconds ago

Suggested change

```
9       -         if (Flag == true && Val != 42) {
     9  +         if (Flag && Val != 42) {
```

Redundant comparison to true can be removed.

```
     10  +
     11  +          // more code
     12  +
     13  +          if (Val == 42) {
     14  +
     15  +          }
```

Comment on lines +13 to +15

Sebastian Böhm @boehmseb · 11 seconds ago

Suggested change

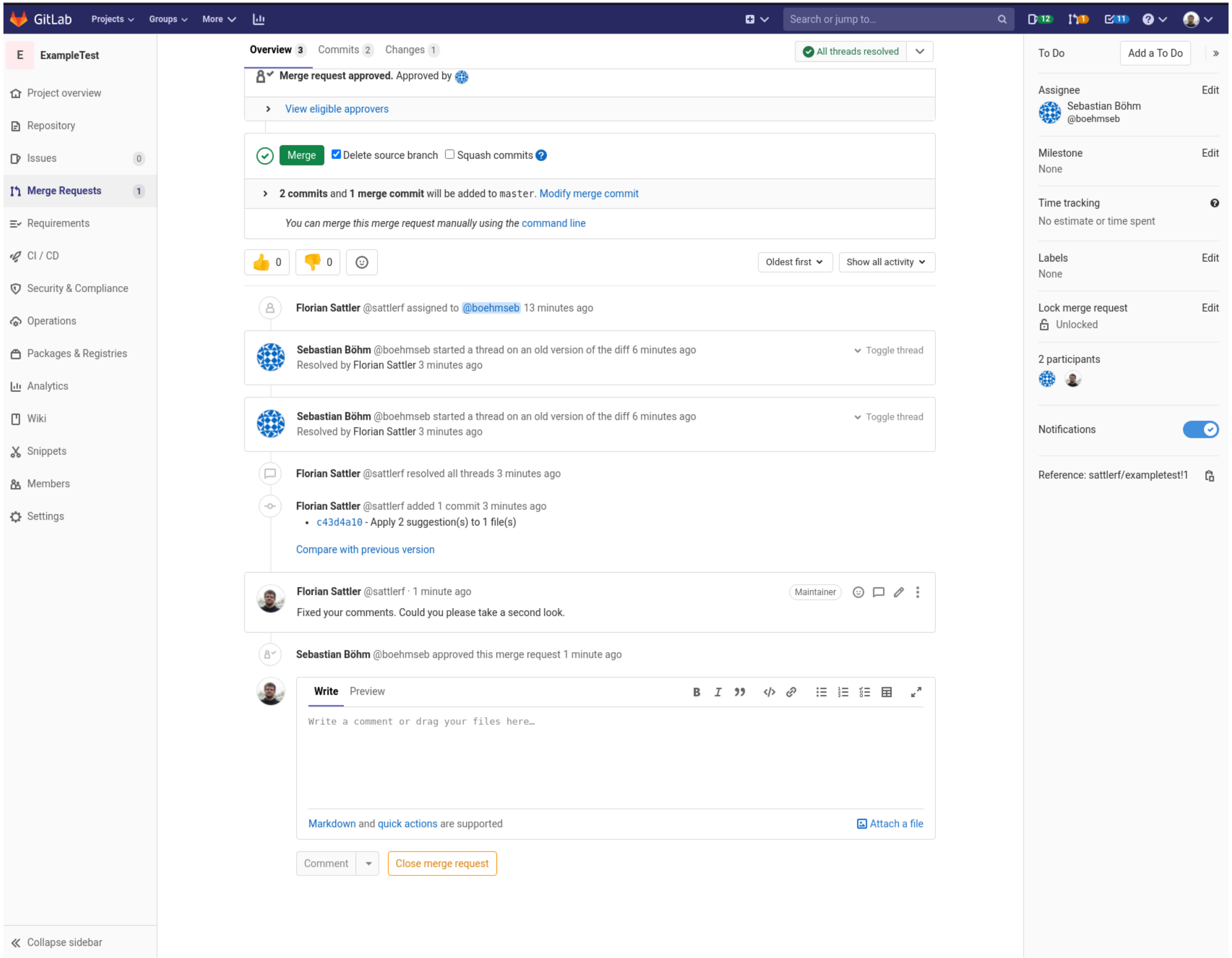
```
13      -             if (Val == 42) {
14      -
15      -             }
```

The outer condition guarantees that this is always false.

Edited by Sebastian Böhm just now

```
     16  +        }
5    17        }
6    18   }
```

Code Review Process



Merge request approved. Approved by

☑ Delete source branch ☐ Squash commits

2 commits and **1 merge commit** will be added to `master`. Modify merge commit

You can merge this merge request manually using the command line

👍 0 👎 0

Florian Sattler @sattlerf assigned to @boehmseb 13 minutes ago

Sebastian Böhm @boehmseb started a thread on an old version of the diff 6 minutes ago
Resolved by **Florian Sattler** 3 minutes ago

Sebastian Böhm @boehmseb started a thread on an old version of the diff 6 minutes ago
Resolved by **Florian Sattler** 3 minutes ago

Florian Sattler @sattlerf resolved all threads 3 minutes ago

Florian Sattler @sattlerf added 1 commit 3 minutes ago

- `c43d4a10` - Apply 2 suggestion(s) to 1 file(s)

Compare with previous version

Florian Sattler @sattlerf · 1 minute ago
Fixed your comments. Could you please take a second look.

Sebastian Böhm @boehmseb approved this merge request 1 minute ago

Write a comment or drag your files here…

Markdown and quick actions are supported

Assignee
Sebastian Böhm
@boehmseb

Milestone
None

Time tracking
No estimate or time spent

Labels
None

Lock merge request
Unlocked

2 participants

Reference: sattlerf/exampletest!1

Benefits of Code Review

Why should we use code review in our project?

- *Code correctness*: catching code/design bugs as early as possible, reduces development costs.
- Ensures that code *is comprehensible* by other developers.
- Increasing the *number of developers* that know about a certain part of the codebase (Bus Factor).
- Promotes *team ownership* of the code.
- *Learning* from others (code + design).
  - The contributor learns from the reviewer and vice versa
- Improves *consistency* across the codebase.

Who should review the code?

Reviewers based on *project* knowledge.

- Code Owners, i.e., developers that are responsible and care about a part of the codebase.
- Team leads that see the projects bigger picture.

Reviewers based on *language* knowledge.

- Developers that have a good understanding about the programming languages used.

How to 'Receive' Code Reviews

Write small changes.

Try to be proactive and thoroughly check your code before requesting feedback from reviewers.

Take on constructive feedback and learn from review comments.

Reviewers are not out to get you; they care about the health of their codebase.

How to 'Write' Code Reviews

Be polite and professional.

Try to note all problems in the first round of feedback.

Do not unnecessarily criticize the author's approach, only when an alternative is beneficial.

Try to provide feedback in a relative short period of time, don't keep your teammates waiting.

Automating parts of code review

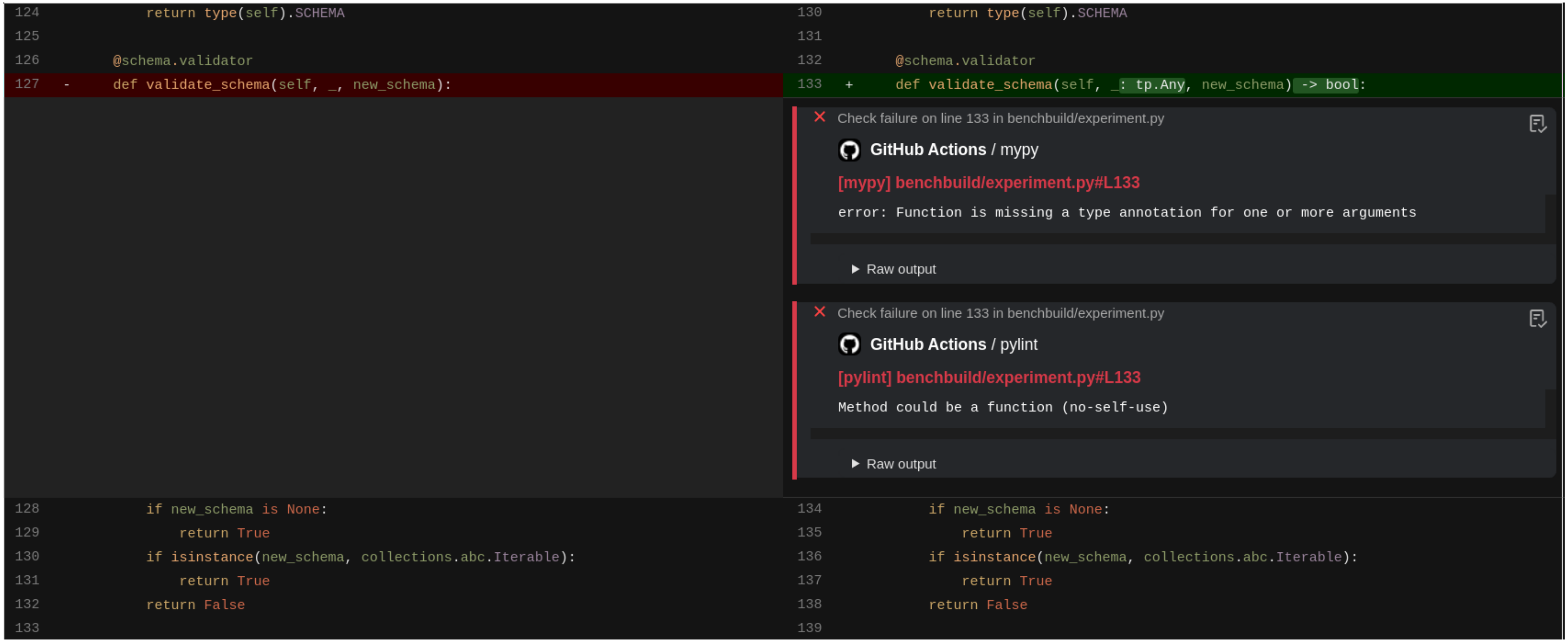
Code review can slow down development time while increasing code change quality.

To reduce slow down, parts of the code review process can and should be automated.

- Run automatic *code formatters*.
- Run *style checkers* to enforce style requirements.
- Run *static analysis* tools to detect bugs early.

Every error fixed before review, reduces work for the reviewer and cuts down overall code-review time.

Automating parts of code review



```
124         return type(self).SCHEMA
125
126     @schema.validator
127 -   def validate_schema(self, _, new_schema):
128         if new_schema is None:
129             return True
130         if isinstance(new_schema, collections.abc.Iterable):
131             return True
132         return False
133
```

```
130         return type(self).SCHEMA
131
132     @schema.validator
133 +   def validate_schema(self, _: tp.Any, new_schema) -> bool:
```

✕ Check failure on line 133 in benchbuild/experiment.py

GitHub Actions / mypy

[mypy] benchbuild/experiment.py#L133

```
error: Function is missing a type annotation for one or more arguments
```

▶ Raw output

✕ Check failure on line 133 in benchbuild/experiment.py

GitHub Actions / pylint

[pylint] benchbuild/experiment.py#L133

```
Method could be a function (no-self-use)
```

▶ Raw output

```
134         if new_schema is None:
135             return True
136         if isinstance(new_schema, collections.abc.Iterable):
137             return True
138         return False
139
```