# **Pull request и code review**

В прошлом уроке мы рассказали о подходе Feature Branch Workflow, в котором вся новая функциональность разрабатывается в feature-ветках. Как только «фича» готова, ветка вливается в main.

Но перед тем, как влиться в main, ветка проходит разные проверки, в том числе изменения просматривают другие участники команды. О таких проверках и пойдёт речь в этой теме.

### **Pull (или merge) request**

В большинстве команд новые функциональности и исправления попадают в main через **запрос на слияние** (англ. *pull request*, или *merge request*). Его так и называют — **пул-реквест** или **мёрж-реквест**. В переводе с языка Git это значит: «Вот моя ветка, хочу «вмёржить» её в main».

**Так как правильнее — пул-реквест или мёрж-реквест?**

Названия «пул-реквест» и «мёрж-реквест» означают одно и то же. GitHub и BitBucket используют термин «пул-реквест», а GitLab — «мёрж-реквест».

В наших примерах мы показываем работу с GitHub, поэтому будем называть запрос на слияние пул-реквестом, или PR.

«Запросы на слияние» — это та часть процесса (workflow), которая может сильно отличаться от команды к команде. Например, в некоторых проектах строго следят за качеством изменений и тщательно проверяют их до того, как влить в main. А некоторые проще относятся к тому, что попадает в основную ветку.

На этапе пул-реквеста можно сделать разные проверки. Например:

* запустить автоматические тесты, которые покажут, не «сломают» ли новые изменения уже существующую в проекте логику;
* просмотреть изменения «глазами» — это называется **code review** (англ. «осмотр», «рецензия кода») или просто **review**. Мы будем использовать вариант написания «ревью».

### **Ревью**

Многие команды уделяют особое внимание процессу ревью, потому что оно выполняет сразу две важные функции:

* проверяется качество и необходимость предлагаемых изменений;
* те, кто выполняет ревью, узнают подробности о новых изменениях в проекте.

Для оценки качества изменений выделяют такие критерии:

* стиль изменений (стиль кода) совпадает с принятым в проекте;
* нет очевидных проблем — ошибок, или **багов** (от англ. *bug*, «жук»), а также уязвимостей — так называемых «дырок», через которые можно взломать систему;
* нет ничего лишнего — иногда опытный участник знает, как сделать то же самое проще, чем предлагается в пул-реквесте.

В процессе ревью можно и нужно обсуждать предлагаемые изменения. Например, можно задать вопрос: «Почему это сделано именно так?». Ответ может быть полезен как опытному участнику проекта — он поймёт замысел автора фичи, так и неопытному — он сможет научиться чему-то новому. В итоге ревью ценно не только для того, кто «принёс» изменения, но и для того, кто их просматривает.

Если ревьюеров всё устраивает, они нажимают кнопку **Approve** (англ. «согласовать», «одобрить»). Иногда вместо кнопки используется комментарий-аббревиатура **LGTM** (англ. ***L****ooks* ***G****ood* ***T****o* ***M****e* — «на мой взгляд, всё хорошо»).

После одобрения пул-реквеста его автор (или один из участников проекта) может нажать кнопку **Merge**. Тогда GitHub (или GitLab/BitBucket) вольёт feature-ветку в main. В зависимости от настроек система также предложит удалить саму feature-ветку, потому что она уже не нужна: все изменения есть в main после слияния.