BeeCR

Руководство пользователя

Команда BeeCR

Содержание

1. Руководство пользователя

1.0.1 Взаимодействие с BeeCR

Примечание: Данный раздел документации предполагает, что в вашем GitLab проекте уже была настроена поддержка BeeCR. Иначе сперва требуется выполнить настройку, используя один из следующих подходов:

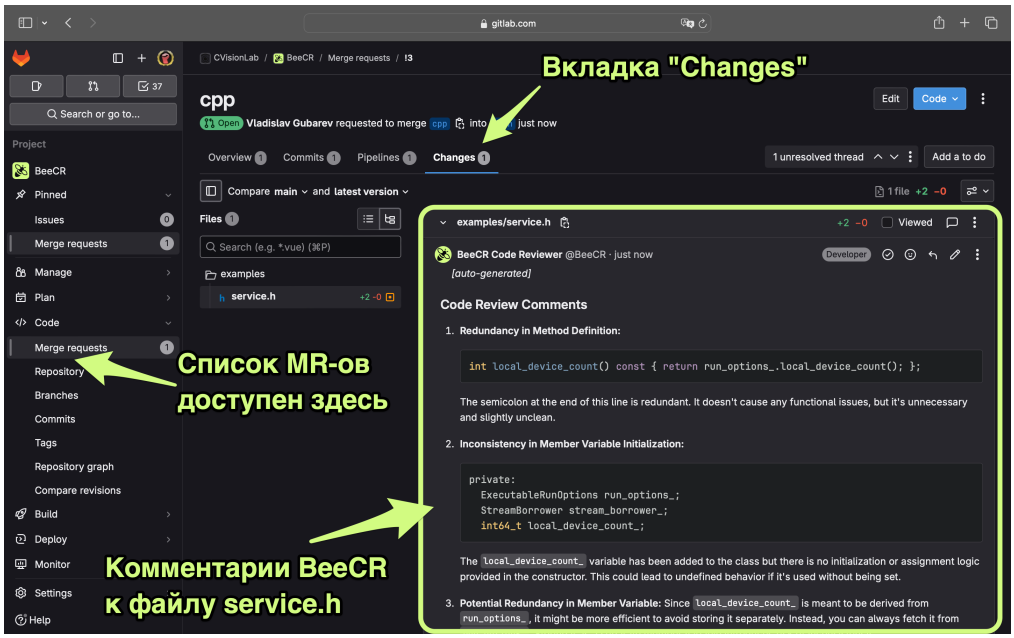
- Через webhook
- Через компонент CI/CD (поддерживается GitLab-ом начиная с версии 17)
  - Обзор компонента
  - Пошаговое руководство по интеграции через CI/CD компонент
- Через шаблон CI/CD (поддерживается и в более старых версиях GitLab)
  - Обзор шаблона
  - Пошаговое руководство по интеграции через CI/CD шаблон

1.1 Краткая инструкция

BeeCR анализирует изменения в коде в запросах на слияние (Merge Request) и добавляет замечания в виде комментариев к файлам. Как только создается новый запрос на слияние или код в существующем запросе на слияние обновляется, GitLab инициирует проверку патчей кода.

Пример:

1. Предположим, что в вашем GitLab проекте уже есть ветка (git branch) с названием `main`, и в ней содержатся некоторые файлы исходного кода.
2. Создайте новую ветку, например, `develop`.
3. В ветке `develop` измените файлы исходного кода (или добавьте новые файлы) и отправьте их в репозиторий средствами `git`.
4. На странице вашего проекта в web-интерфейсе GitLab откройте новый запрос на слияние (Merge Request) ветки `develop` в ветку `main`.
5. На странице вашего запроса на слияние в web-интерфейсе GitLab перейдите на вкладку с изменениями кода (Changes) и пролистайте список изменений до файла, который вас интересует. Комментарии к изменениями в файле будут отображаться непосредственно над блоком изменений к этому файлу.



1.2 Особенности

- BeeCR проверяет изменения в коде (патчи) только в запросах на слияние.
- BeeCR добавляет замечания в виде комментариев к файлам.
- Если файл уже содержит комментарии от BeeCR и не был изменен с момента прошлой проверки, BeeCR не проверяет патч кода для такого файла снова.
- Если же файл был изменен с момента последней проверки, то BeeCR либо удалит, либо скроет свой устаревший комментарий:
  - Устаревший комментарий будет просто удален, если другие участники проекта не оставляли ответных комментариев.
  - Устаревший комментарий будет свернут (с возможностью развернуть обратно), если другие участники проекта оставляли свои ответные комментарии.

1.3 Настройка через конфигурационный файл в репозитории

BeeCR имеет гибкую систему настроек, которыми можно управлять различными способами. Один из наиболее удобных способов — через конфигурационный файл `.beecr.yml` в корне репозитория.

С помощью конфигурационного файла `.beecr.yml` в репозитории вы можете настраивать следующие проектно-специфичные параметры:

- `model` — Модель ИИ (только для on-premises; недоступно в SaaS версии). Значение по умолчанию: `ollama/codestral:22b`.
- `ollama-host` — Адрес сервера Ollama (только для on-premises; недоступно в SaaS версии).
- `openai-host` — Адрес сервера OpenAI (только для on-premises; недоступно в SaaS версии). Значение по умолчанию: `https://api.openai.com`.
- `openai-api-key` — Ключ для доступа к OpenAI API (только для on-premises; недоступно в SaaS версии). Значение по умолчанию: пустая строка.
- `target` — Регулярное выражение для определения файлов, подлежащих проверке. Значение по умолчанию: `\.(py|c|h|cpp|hpp|cs|java|kt|swift|php|go|sh|(j|t)sx?)$`.
- `target-extra` — Дополнительное регулярное выражение для определения файлов, подлежащих проверке (полезно в тех случаях, когда вы не хотите полностью переопределять `target`, а лишь расширить его). Значение по умолчанию: пустая строка.
- `target-exclude` — Регулярное выражение для исключения файлов из проверки (полезно в тех случаях, когда вы не хотите полностью переопределять `target` или `target-extra`). Значение по умолчанию: пустая строка.
- `language` — Язык, на котором BeeCR будет писать комментарии. Значение по умолчанию: `English`.
- `instructions` — Дополнительные инструкции для передачи модели ИИ дополнительного контекста или корректировки процесса проверки кода. Значение по умолчанию: пустая строка.
- `note-trigger` — Выражение-триггер, отслеживаемое в пользовательских комментариях. Значение по умолчанию: `/beecr`.

Пример конфигурационного файла:

.beecr.yml

```yaml
model: 'gpt-4o'
ollama-host: 'http://my-ai-server:11434'
openai-host: 'https://api.openai.com'
openai-api-key: 'sk-my-openai-api-key'
target: '\.(py|c|h|cpp|hpp|cs|java|kt|swift|php|go|sh|(j|t)sx?)$'
target-extra: '\.hs$'
target-exclude: '\.sh$'
language: 'English'
instructions: 'Do not focus on code style, highlight bugs only.'
note-trigger: '/ai'
```

Более подробную информацию о всех параметрах и способах настройки вы можете получить в разделах документации Вебхук BeeCR для GitLab: Конфигурирование и Настройки API-сервера BeeCR.

1.4 Дополнительные возможности BeeCR при интеграции через вебхуки

1.4.1 Ручной запуск проверки кода

Если события "Comment" были разрешены в вашем проекте для вебхука BeeCR, то вы можете инициировать процесс проверки вручную, оставив комментарии в своем запросе на слияние (Merge Request). Комментарии, которые инициируют проверку, должны содержать специальное выражение `/beecr`.

Примечание: Ваш DevOps или системный администратор может задать другое выражение вместо `/beecr` в параметрах вебхука или в настройках сервера API (если вы используете BeeCR, развернутый на вашем сервере).

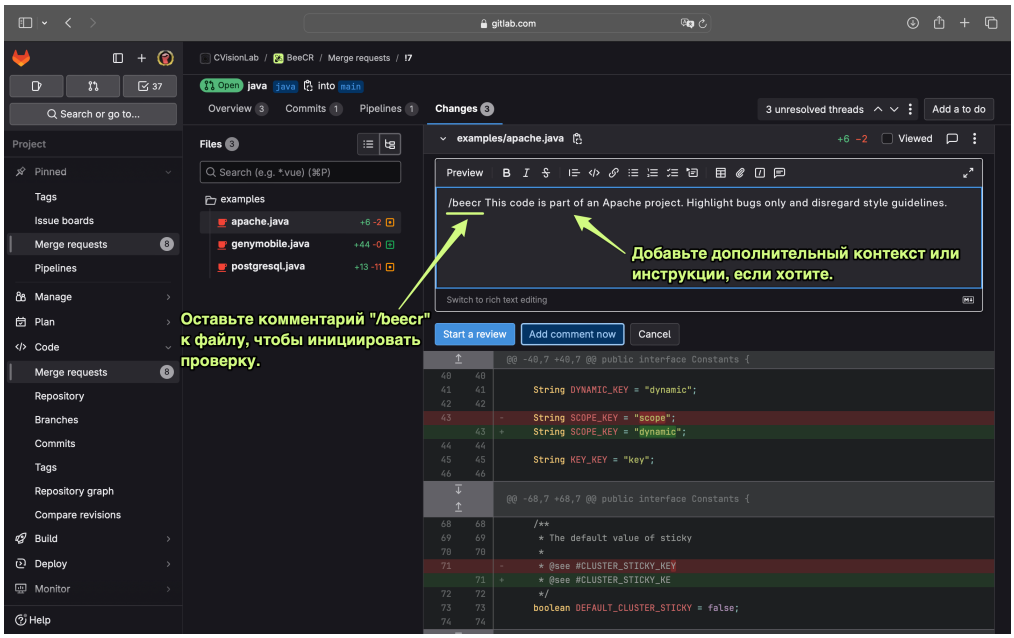
Дополнительные инструкции в комментариях

Вместе с ключевым словом вы можете предоставить дополнительную информацию, чтобы дать модели ИИ больше контекста или скорректировать процесс проверки. Например:

- `/beecr` Не обращай внимания на стиль кода; выделяй только ошибки.
- `/beecr` Проверяй только проблемы безопасности.
- `/beecr` Оставляй краткие комментарии, отмечая максимум 5 проблем.

Проверка одного файла

Оставьте комментарий к определенному файлу в своем запросе на слияние, чтобы начать проверку этого файла. В этом случае любые настройки, определяющие набор целевых файлов, будут проигнорированы.



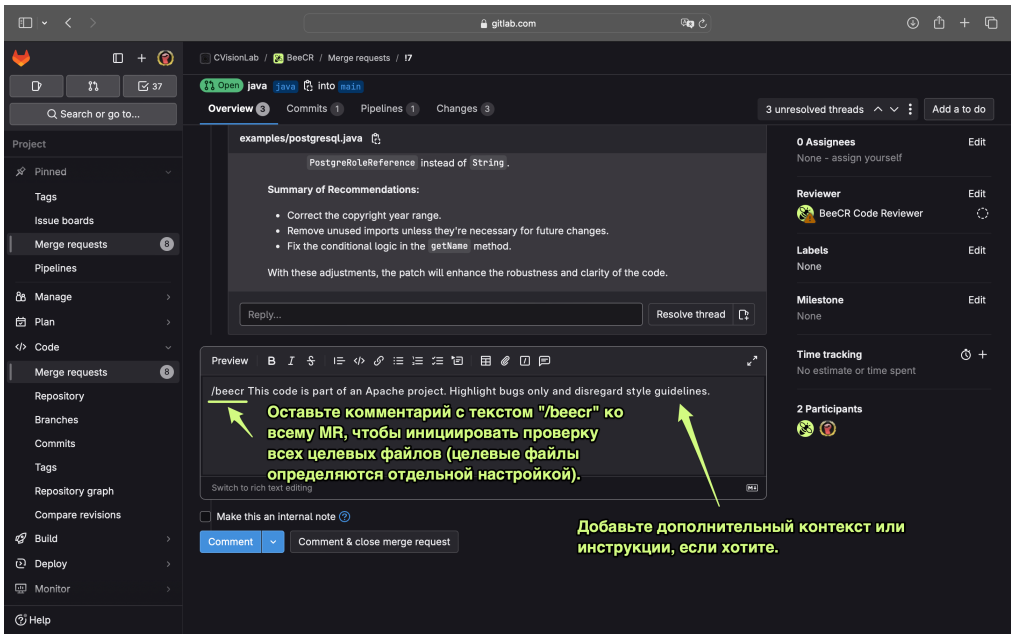
Проверка всех файлов

Оставьте комментарий ко всему запросу на слияние, чтобы проверить все целевые файлы.

Примечание: По умолчанию проверка включена для файлов на следующих языках:

- Python: `.py`
- C/C++: `.h`, `.hpp`, `.c`, `.cpp`
- C#: `.cs`
- Java: `.java`
- Kotlin: `.kr`
- Swift: `.swift`
- PHP: `.php`
- Go: `.go`
- Bash/Shell: `.sh`
- JavaScript/Typescript: `.js`, `.jsx`, `.mjs`, `.mjsx`, `.ts`, `.tsx`

Однако вы можете настроить ее так, как вам удобно, чтобы поддерживать больше языков или, наоборот, исключить некоторые из стандартных.



1.5 Дополнительные возможности BeeCR при интеграции через CI/CD

Примечание: Информация в этом разделе актуальна только в случае интеграции BeeCR через компонент или шаблон задачи CI/CD.

1.5.1 Пропустить выполнение CI/CD задачи `beecr`

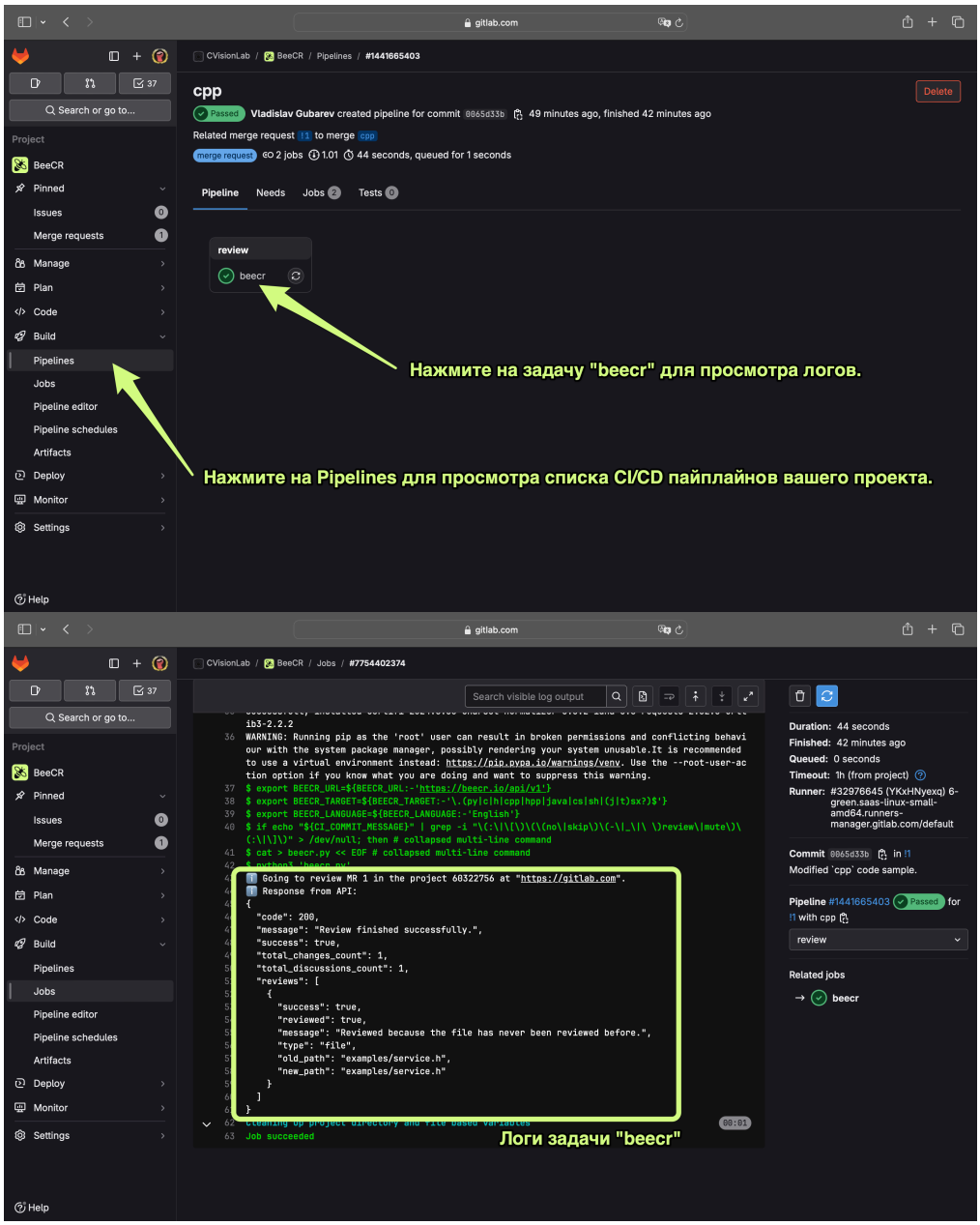
Задача `beecr` не будет создана, если сопроводительное сообщение в коммите (git commit) содержит любое из следующих выражений:

- `:mute:`, `[mute]`
- `:no-review:`, `:no_review:`, `:no review:`, `[no review]`, `[no-review]`, `[no_review]`
- `:skip-review:`, `:skip_review:`, `:skip review:`, `[skip-review]`, `[skip_review]`, `[skip review]`

1.5.2 Логи CI/CD задачи `beecr`

В случае необходимости посмотреть логи CI/CD задачи `beecr` выполните следующие действия:

1. Перейдите к списку пайплайнов CI/CD в вашем проекте (разверните меню "Build" в web-интерфейсе GitLab на боковой панели и нажмите "Pipelines").
2. Выберите пайаплайн, соответствующий вашему запросу на слияние (помечен дополнительным бейджем "merge request").
3. Перейдите к задаче BeeCR в деталях вашего конвейера.



В приведенном выше примере BeeCR сформировал следующий отчет о проверке кода:

```
{
  "code": 200,
  "message": "Review finished successfully.",
  "success": true,
  "total_changes_count": 1,
  "total_discussions_count": 1,
  "reviews": [
    {
      "success": true,
      "reviewed": true,
      "message": "Reviewed because the file has never been reviewed before.",
      "type": "file",
      "old_path": "examples/service.h",
      "new_path": "examples/service.h"
    }
  ]
}
```

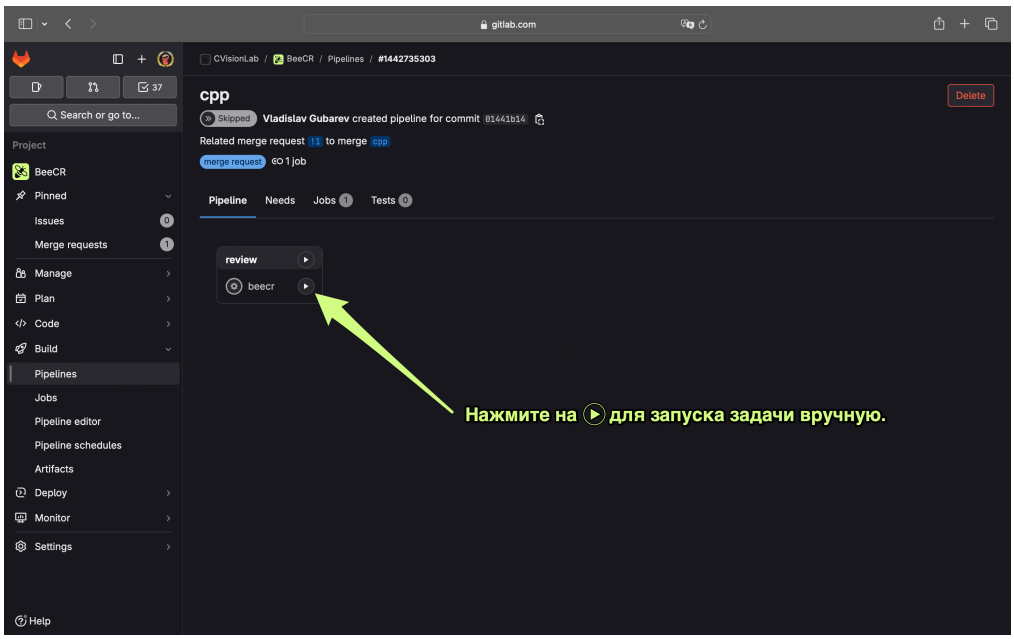
Из этого примера отчета мы можем получить следующую информацию:

- Процедура проверки кода завершена успешно.
- В запросе на слияние был проверен один файл: `examples/service.h`
- BeeCR оставил комментарии к файлу `examples/service.h`, потому что файл ранее проверен не был.

1.5.3 Ручной запуск CI/CD задачи `beecr`

По умолчанию CI/CD задача `beecr` выполняется автоматически. Однако можно настроить выполнение задачи вручную. Если на вашем проекте настроено ручное выполнение CI/CD задачи `beecr`, то для проведения проверки кода вам нужно запускать её самостоятельно:

1. Перейдите к пайплайну с задачей `beecr`.
2. Рядом с задачей `beecr` нажмите на кнопку запуска .



Обратитесь к этому разделу документации GitLab, чтобы узнать больше о ручном запуске задач.

Чтобы узнать, как настроить ручной запуск CI/CD задачи `beecr`, перейдите к соответствующему разделу в документации об интеграции BeeCR через CI/CD:

- Обзор компонента (поддерживается GitLab-ом начиная с версии 17)
- Обзор шаблона (поддерживается и в более старых версиях GitLab)