Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Российская академия народного хозяйства и государственной службы при Президенте Российской Федерации»

Нижегородский институт управления

Кафедра Информатики и информационных технологий

ОТЧЕТ

О ПРОДЕЛАННОЙ ЛАБОРАТОРНОЙ РАБОТЕ №1

по курсу «Программная инженерия»

Выполнил: студент группы:

ИБ-531

Законов А.С.

Нижний Новгород

2024 г.

Оглавление

[Освоение работы Git, GitHub, GitLab. 3](#_Toc159005996)

[Назначение, Возможности, Особенности. 3](#_Toc159005997)

[Ниже перечислены некоторые из особенностей Гитлаб: 3](#_Toc159005998)

[Runner от GitLab 4](#_Toc159005999)

[CI/CD 6](#_Toc159006000)

[Сравнение Git GitHub GitLab 8](#_Toc159006001)

[Gitflic 9](#_Toc159006002)

[Команды системы Git: названия и назначение 10](#_Toc159006003)

# Освоение работы Git, GitHub, GitLab.

## Назначение, Возможности, Особенности.

|  | **Git** | **GitHub** | **GitLab** |
| --- | --- | --- | --- |
| Кто создал | **Линус Торвальдс (Linus Torvalds)** | **Chris Wanstrath, Tom Preston-Werner, PJ Hyett** | **Дмитрий Запорожец,** **Валерий Сизов** |
| Возможности | Отслеживание изменений в исходном коде, создание и слияние веток, управление версиями | Хостинг репозиториев, отслеживание изменений, совместная работа над проектами, код-ревью, управление задачами и ошибками | Управление репозиториями Git, непрерывная интеграция и доставка, управление задачами, код-ревью, управление доступом, интеграция с Docker и Kubernetes |
| Особенности | Локальная система контроля версий, распределенная, работа в команде с использованием командной строки | Облачный сервис для хостинга Git-репозиториев, совместная разработка, социальная сеть для разработчиков | Платформа управления репозиториями Git с широким спектром инструментов для поддержки DevOps-практик, самостоятельное развертывание в корпоративной среде, поддержка CI/CD, управление задачами, код-ревью, управление доступом, интеграция с Docker и Kubernetes |

### Ниже перечислены некоторые из особенностей Гитлаб:

GitLab может похвастаться высокой совместимостью с многими популярными платформами и сервисами, такими как Docker, Kubernetes, [Jira](https://softlist.kz/ru/catalog/jira-software) и сервисы от Google, а также предлагает инструменты для интеграции с практически любым приложением. Это делает интеграцию GitLab в любую корпоративную среду простой задачей.

Одной из особенностей GitLab является система меток, которая значительно упрощает процесс разработки, позволяя классифицировать ошибки или запросы и отслеживать изменения для личных или внешних проектов. Система документации платформы также хорошо организована, и каждый проект документируется в своем собственном репозитории.

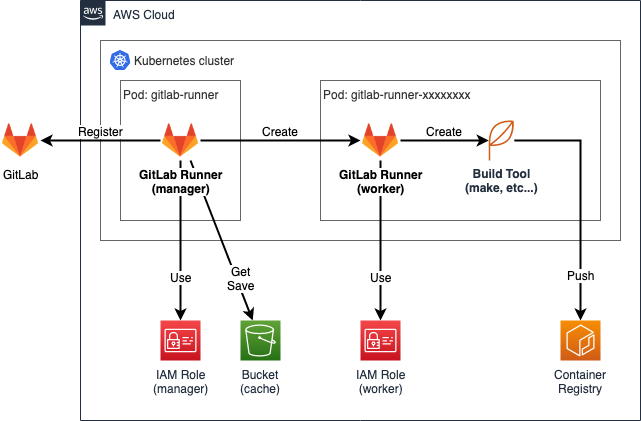
Настройки доступа к репозиториям GitLab высоко настраиваемы в зависимости от группы пользователя. Закрытые ветки могут быть созданы с использованием встроенного модуля, который позволяет настраивать права доступа для каждого отдельного пользователя. Собственная система защиты GitLab обеспечивает безопасную аутентификацию.

Для удобного импорта и экспорта данных GitLab позволяет легко импортировать большие объемы данных из различных источников, включая Jira. Пользователи также могут использовать инструменты для синхронизации кода.

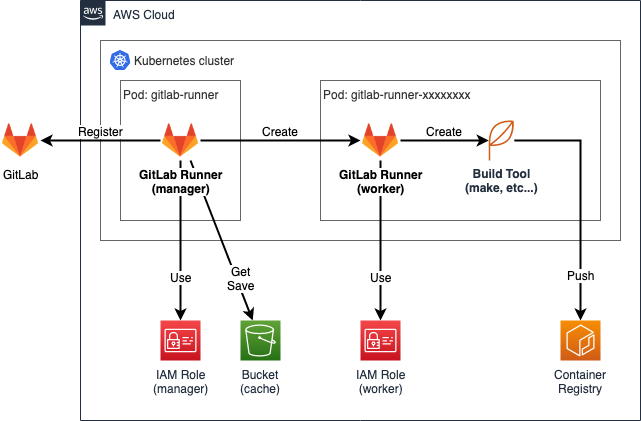
GitLab является отличным выбором для разработчиков и тестировщиков приложений благодаря использованию Kubernetes, который является самым популярным оркестратором в среде контейнеризации. Кроме того, GitLab предлагает пользователям бесплатное размещение своих проектов в облаке и возможность создания частных репозиториев для хранения открытого кода.  
Аналитические инструменты GitLab предоставляют информацию о времени, затраченном на каждую задачу, позволяя эффективно планировать и отслеживать деятельность каждого разработчика. Инструмент Burndown Chart особенно полезен для тех, кто использует метод спринтов в своей работе.

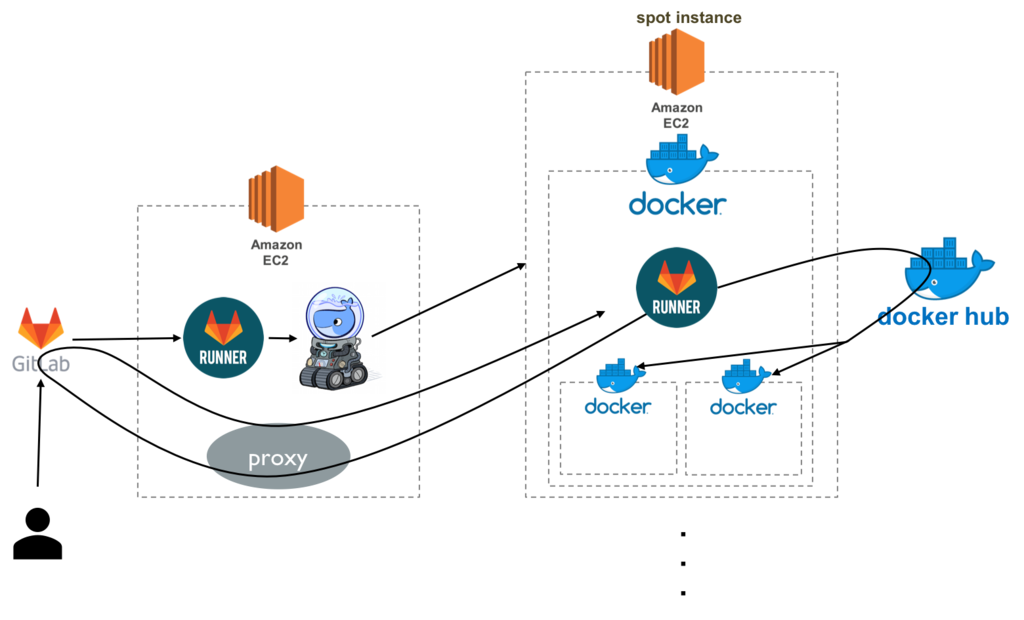
GitLab обновляется каждый месяц, с основным фокусом на удобстве использования и безопасности. С каждым обновлением, на платформу добавляются новые функции и улучшения.

### Runner от GitLab



Гитлаб ранер – это веб-tools, который позволяет выполнить инструкции. Этот инструмент является необходимые людям, кто планирует настраивать CI/CD своего проекта на GitLab. Он позволяет автоматизировать процесс сборки и тестирования кода, что помогает ускорить процесс разработки и улучшить качество конечного продукта.

Для использования GitLab Runner необходимо сначала установить Docker – платформу для контейнеризации. С помощью Docker создаются образы, которые могут быть использованы для развертывания контейнеров на любой системе. Это позволяет разработчикам создавать приложения и микросервисы, которые могут быть запущены в любой среде, не зависимо от того, на какой платформе они были созданы.

GitLab Runner поддерживает несколько различных режимов работы, в том числе Docker, Shell и Parallels. Он также позволяет настроить несколько экземпляров Runner'ов для одного репозитория, что помогает ускорить процесс сборки и тестирования кода. Кроме того, GitLab Runner поддерживает параллельное выполнение задач, что позволяет уменьшить время выполнения и ускорить процесс разработки.

GitLab Runner – это мощный и гибкий инструмент для автоматизации процесса разработки. Он позволяет ускорить процесс сборки и тестирования кода, а также улучшить качество конечного продукта. Если вы планируете юзать GitLab для dev своего проекта, то установка GitLab Runner является необходимым шагом для настройки CI/CD и оптимизации процесса разработки.

### CI/CD



CI/CD (Continuous Integration/Continuous Delivery) – это методология разработки программного обеспечения, которая позволяет автоматизировать процесс сборки, тестирования и доставки приложений. Она позволяет сократить время разработки, увеличить качество и стабильность приложений, а также уменьшить количество ошибок в процессе разработки.

GitLab CI/CD – это инструмент, который позволяет автоматизировать процесс сборки, тестирования и доставки приложений на платформе GitLab. Он позволяет настроить непрерывную интеграцию и доставку приложений, а также автоматически выполнять задачи, связанные с разработкой, тестированием и развертыванием приложений.

С помощью GitLab CI/CD можно управлять конфигурациями через YAML-файлы, запускать задачи в различных средах, собирать и выполнять код в различных операционных системах. Кроме того, он предоставляет инструменты для интеграции с Kubernetes и Docker, что позволяет упростить и ускорить процесс разработки.

В сравнении с аналогом на платформе GitHub – GitHub Actions, GitLab CI/CD имеет ряд преимуществ. Например, GitLab CI/CD предоставляет мощные инструменты для управления проектами и настройки непрерывной интеграции и доставки, в то время как GitHub Actions имеет более ограниченный функционал. Кроме того, GitLab CI/CD предлагает бесплатное хранение кода в GitLab, а GitHub Actions – это платный сервис.

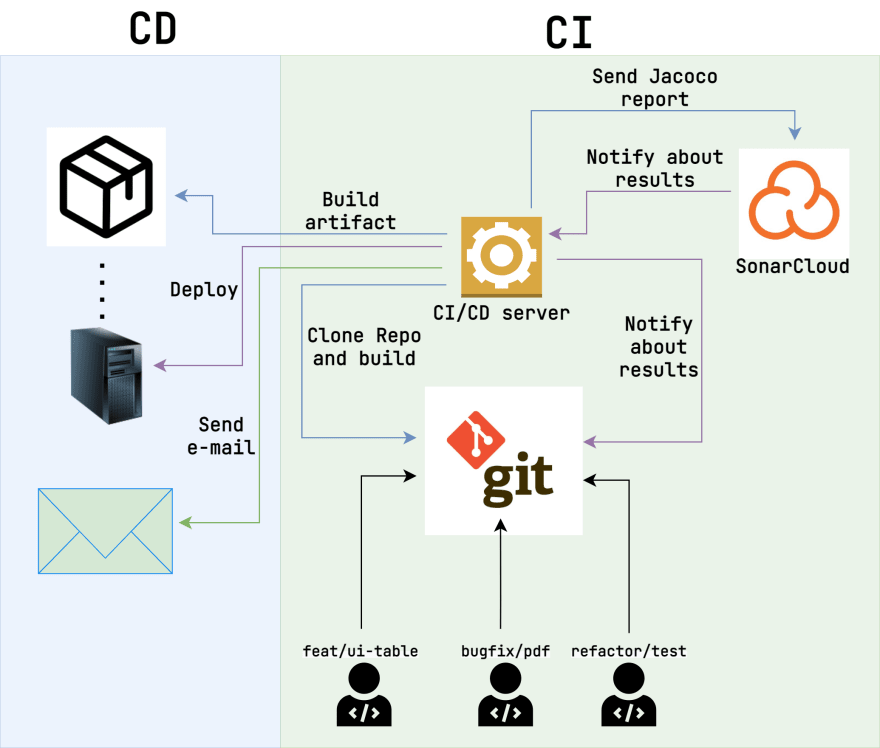
При сравнении GitLab CI/CD и GitHub Actions можно выделить несколько отличий, которые могут быть важны для выбора инструмента:

Синтаксис. Оба инструмента используют YAML-файлы для определения задач. Однако GitLab CI/CD использует ключ script для выполнения задач, в то время как в GitHub Actions для этого используется ключ run.

Запуск на различных платформах. Для выполнения задач на разных платформах в GitLab CI/CD используется ключ tags, а в GitHub Actions — runs-on.

Работа с контейнерами. Оба инструмента могут работать с задачами в образах Docker и указывать дополнительные контейнеры. Однако в GitLab CI/CD для этого используется ключ image, а в GitHub Actions — container.

Условия выполнения задач. Для выполнения задач с условиями в GitLab CI/CD используется ключ rules, а в GitHub Actions — if.

 Зависимости между задачами. Для выполнения задач с зависимостями в GitLab CI/CD используется ключ stages, а в GitHub Actions — needs.

Если вы хотите создать свой первый проект на GitLab, вам нужно выполнить несколько простых шагов:

Создайте учетную запись и рабочую группу. Для этого перейдите на официальный сайт GitLab и зарегистрируйтесь. Затем создайте рабочую группу, где вы сможете хранить все свои проекты.

Создайте репозиторий. Нажмите кнопку «New Project» и выберите, хотите ли вы, чтобы ваш проект был общедоступным или закрытым. Затем дайте проекту имя и описание, и выберите группу, в которой он будет создан.

Загрузите файлы. Чтобы загрузить файлы в репозиторий, вам необходимо создать новый коммит. Нажмите кнопку «Upload file» и выберите файлы, которые вы хотите загрузить.

Добавьте ключи аутентификации. Чтобы работать с репозиторием, вам необходимо добавить ключи аутентификации. Перейдите в раздел "Настройки" вашего проекта и выберите пункт «CI/CD». Затем добавьте необходимые ключи аутентификации для доступа к вашему репозиторию.

## Сравнение Git GitHub GitLab

Разработчики могут использовать GitLab для работы через командную строку, встроенный Web IDE или сторонние Git-клиенты. Не существует единственно правильного подхода, поскольку каждый разработчик выбирает наиболее удобный вариант, опираясь на требования проекта и доступные ресурсы. Независимо от того, какой способ работы вы выберете, GitLab поможет вам эффективно организовать процесс разработки и достичь лучших результатов.

GitLab имеет свои особенности и преимущества, которые могут быть важными для некоторых проектов и команд разработчиков. Например, GitLab предоставляет больше возможностей для управления правами доступа, имеет интеграцию с CI/CD и другими инструментами разработки.

GitLab и GitHub на самом деле очень похожи и отличаются несущественно, но есть несколько ключевых отличий:

GitLab отличается от GitHub тем, что это проект с открытым исходным кодом, что позволяет сообществу разработчиков улучшать и расширять его функциональность. В то же время, только внутренние разработчики могут улучшать и дополнять платформу GitHub.

Существует второе отличие GitLab от GitHub, заключающееся в том, что начиная с две тысячи восемнадцатого года компания Microsoft стала овнером GitHub. Это вызвало неоднозначную реакцию в сообществе разработчиков.

Тем не менее, несмотря на это, популярность GitHub остается выше, чем у GitLab. GitHub был создан в 2008 году и не имел конкурентов до появления GitLab в 2011 году, который на старте был не очень активный.

Гитлаб и Гитхаб обладают схожими возможностями для хранения и управления Git-репозиториями, а еще для роботе в команде. Однако, каждая платформа имеет свои уникальные особенности плюсы, и конечный выбор между ними должен опираться на конкретные потребности и задачи Dev Teams.

## Gitflic



Gitflic — это **российский облачный сервис хранения репозиториев исходного кода, основан на системе контроля версий Git**.

GitFlic ― новый отечественный аналог GitHub. Его запустили в декабре 2021 года.

GitFlic ― очень молодой и перспективный проект. Его возможности и даже интерфейс во многом повторяют GitHub, но пока, например, не хватает «Панели проектов». Для больших IT-проектов корпоративные потребители часто используют Jira. Для личных или образовательных — можно использовать, например, Trello. Но если аналог этих сервисов появится в GitFlic, это будет ему дополнительным плюсом.

Также хорошо бы реализовать в GitFlic аналог Wiki GitHub. Если проект пользователя сервиса перерос простой README файл, имеет смысл организовать хранение руководств пользователей или более подробной технической документации по проекту подобно страницам «Википедии». 

## Команды системы Git: названия и назначение

git init: Создает новый репозиторий Git.

git clone: Клонирует существующий репозиторий в новый каталог.

git add: Добавляет файлы в индекс (стaging area) для последующего коммита.

git commit: Фиксирует изменения в индексированных файлах в репозитории.

git status: Показывает состояние рабочего каталога и индекса.

git diff: Показывает изменения между коммитами, индексом и рабочим каталогом.

git log: Показывает историю коммитов.

git branch: Показывает список веток, создает новую ветку или удаляет существующую.

git checkout: Переключается между ветками, восстанавливает файлы из индекса и т. д.

git merge: Объединяет указанную ветку в текущую ветку.

git remote: Управляет наборами отслеживаемых репозиториев.

git fetch: Забирает изменения из удаленного репозитория, но не сливает их.

git pull: Забирает изменения из удаленного репозитория и сливает их с текущей веткой.

git push: Отправляет локальные изменения в удаленный репозиторий.

git revert: Создает новый коммит, отменяющий изменения одного или нескольких существующих коммитов.

git reset: Сбрасывает текущее состояние индекса и рабочего каталога к состоянию определенного коммита.

git tag: Создает, показывает или удаляет метки (теги) объектов в репозитории.