Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

Факультет Информационных технологий и программирования

# *Лабораторная работа по Git №3*

Выполнил: Нечаев Александр Сергеевич,

группа M3111

Проверил: Повышев Владислав Вячеславович

Санкт-Петербург

2022

# 1. GitHub против GitLab против Bitbucket

## Что такое Git?

Git - это в основном инструмент контроля исходного кода, который отслеживает изменения, внесенные вами в определенные файлы и папки с течением времени. В частности, Git отслеживает все изменения, которые вы вносили в исходный код и связанные с ним зависимости в процессе разработки.

С точки зрения непрофессионала, Git - это инструмент контроля исходного кода, который позволяет вам управлять и отслеживать историю вашего исходного кода.

## GitHub

GitHub, Inc. является дочерней компанией Microsoft, которая предоставляет хостинг для разработки программного обеспечения и контроля версий с помощью Git. Он предлагает функциональность Git по распределённому контролю версий и управлению исходными кодами, а также собственные возможности. Он предоставляет контроль доступа и несколько функций для совместной работы, таких как отслеживание ошибок, запросы функций, управление задачами, непрерывная интеграция и вики для каждого проекта.

Основная цель GitHub - облегчить контроль версий и аспекты отслеживания проблем при разработке программного обеспечения. Ярлыки, контрольные точки, распределение ответственности и поисковая система доступны для отслеживания проблем. Для управления версиями Git позволяет pull запросами предлагать изменения в исходный код. Пользователи, у которых есть возможность просматривать предлагаемые изменения, могут видеть разницу в запрошенных изменениях и одобрять их. В терминологии Git это действие называется «committing», и один из его экземпляров - «commit». История всех коммитов сохраняется и может быть просмотрена позже.

Кроме того, GitHub поддерживает следующие форматы и функции:

- Документация, включая автоматически визуализированные файлы README в различных форматах файлов, подобных Markdown.

- Вики

- GitHub Actions, который позволяет создавать конвейеры непрерывной интеграции и непрерывного развертывания для тестирования, выпуска и развертывания программного обеспечения без использования сторонних веб-сайтов/платформ.

- Диаграммы: участники, коммиты, частота кода, перфокарта, сеть, участники

- Справочник интеграций

- Уведомления по электронной почте

- Обсуждения

- Возможность подписать кого-то на уведомления, упомянув их.

- Вложенные списки задач в файлах

- Визуализация геопространственных данных

- Файлы SD-рендеринга, которые можно предварительно просмотреть с помощью новой интегрированной программы просмотра файлов STL, которая отображает файлы на «SD-холсте». Программа просмотра работает на WebGL и Three.js

Условия использования GitHub не требуют, чтобы проекты общедоступного программного обеспечения, размещенные на GitHub, соответствовали определению открытого исходного кода.

**GitLab**

GitLab - это веб-инструмент жизненного цикла DevOps, предоставляющий Git-репозиторий, функции вики, отслеживания проблем и непрерывной интеграции и развёртывания, используя лицензию с открытым исходным кодом, разработанную GitLab Inc.

Первоначально код был написан на языке Ruby, затем некоторые его части были переписаны на языке Go, первоначально как решение для управления исходным кодом для совместной работы в команде по разработке программного обеспечения. Позднее он превратился в комплексное решение, охватывающее жизненный цикл разработки программного обеспечения, а затем - весь жизненный цикл DevOps. Текущий стек технологий включает в себя Go, Ruby on Rails и Vue.js.

Он следует модели разработки с открытым ядром, где основной функционал выпускается по лицензии с открытым исходным кодом MIT, а дополнительный функционал - по проприетарной лицензии

**BitBucket**

Bitbucket - это веб-сервис хостинга репозиториев контроля версий, принадлежащий Atlassian, для исходного кода и проектов по разработке, использующих системы контроля версий Mercurial или Git.

Bitbucket в основном используется для кода и его проверки. Он поддерживает следующие функции:

- Pull запрос с обзором кода и комментариями

- «Bitbucket Pipelines», служба непрерывной доставки

- Двухэтапная проверка и обязательная двухэтапная проверка

- Проверки слияния

- Поиск кода

- Git Large File Storage

- Документация, включая автоматически визуализированные файлы README в различных форматах файлов, подобных Markdown.

- Отслеживание проблем

- Вики

- Статические сайты, размещенные в Bitbucket Cloud: статические сайты, имеют домен bitbucket.io в своем URL.

- Дополнения и интеграции

- REST API для создания сторонних приложений, которые могут использовать любой язык разработки

- Сниппеты, которые позволяют разработчикам делиться фрагментами кода или файлами

- Умное зеркалирование

**Отличия**

Ключевым отличием всех этих трех платформ - Bitbucket, GitHub и GitLab - является поддержка репозиториев с открытым исходным кодом в платформе. Из всех трёх платформ GitLab - единственная, которая поддерживает репозитории с открытым исходным кодом, так же она предоставляет пользователю возможность увидеть полный код на официальном сайте. На платформе GitHub, несмотря на то, что существует большая категория свободных проектов с открытым исходным кодом, которые не относятся к репозиториям с открытым исходным кодом, так же существует большое количество свободных проектов с открытым исходным кодом, которые помогают связать людей, с похожими интересами. Bitbucket можно использовать автономно, получив возможность приватного размещения своего проекта, но возможность создания репозитория с открытым исходным кодом на платформе Bitbucket отсутствует.

Следующим различием между всеми этими платформами является импорт репозиториев. Для платформы управления репозиториями она должна иметь возможность импортировать репозитории из других платформ. В случае с Bitbucket пользователь может импортировать репозитории с платформы Mercurial. В GitHub также поддерживаются эти функции, и пользователь также может импортировать репозитории с других платформ. Но в случае с GitLab пользователь может импортировать репозитории только с платформы git. Если необходимо импортировать проект с какой-либо другой платформы, то GitHub или Bitbucket - идеальный вариант для использования.

Еще одно отличие между платформами Bitbucket, GitHub и GitLab - это дистрибуция проектов. Платформа Bitbucket используется для распространения проекта среди членов команды и является выгодной для них. GitHub использует организационный уровень для распространения проекта и широко используется различными компаниями и организациями. В GitLab проекты распределены по группам, а члены групп могут получить доступ к проекту и поделиться кодом. Управление на уровне группы осуществляется путем добавления пользователей в группу, а управление группой осуществляется с помощью функции оповещений.

Bitbucket поддерживает функцию pull запроса, которая помогает загрузить проект из платформы. GitHub также поддерживает функцию pull запроса и помогает пользователю получить проект с платформы. В GitLab такая функция pull запроса отсутствует, и вместо нее в платформе GitLab поддерживается merge запрос.

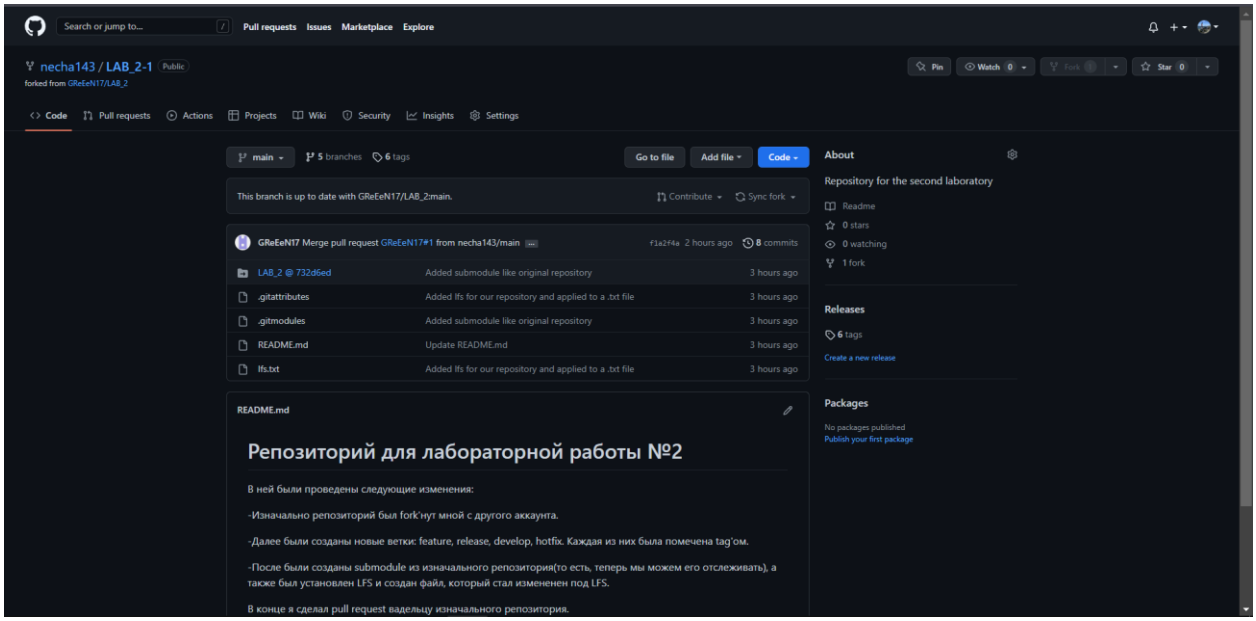
Сниппет кода может быть использован совместно в платформах Bitbucket и GitLab, в то время как в GitHub сущность кода разделяется между пользователями. Фрагмент кода может быть общедоступным, приватным или внутренним. Сниппет исходного кода может помочь пользователю получить основную идею и может быть использован в проекте.

**Заключение**

Как итог, GitHub имеет лучшую поддержку сообщества из трех. Так что, если у вас все в порядке с общедоступным репозиторием и вы планируете выпустить что-то с открытым исходным кодом и прислать кучу людей, GitHub определенно является номером один. BitBucket и GitLab служат альтернативными вариантами, если вы не хотите использовать GitHub по какой-либо причине.

Однако, если ваша работа немного более дискретна, и вам нужен частный репозиторий, и вы не хотите на него ничего тратить, BitBucket и GitLab - отличные варианты.

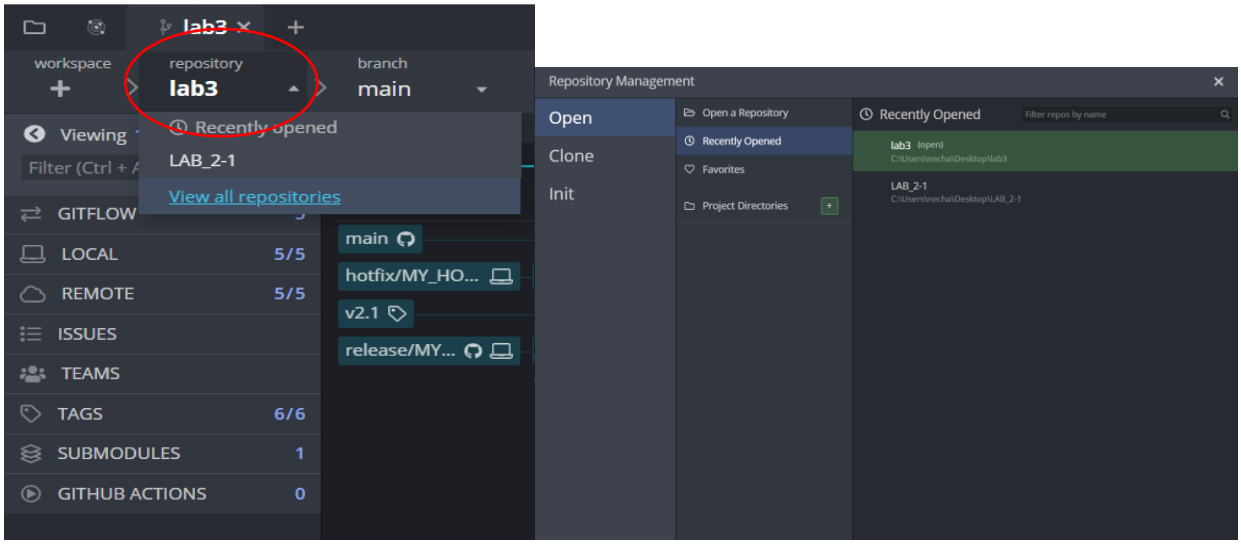
**2**. Оформление лабораторной работы №2 в GitHub выглядит так:



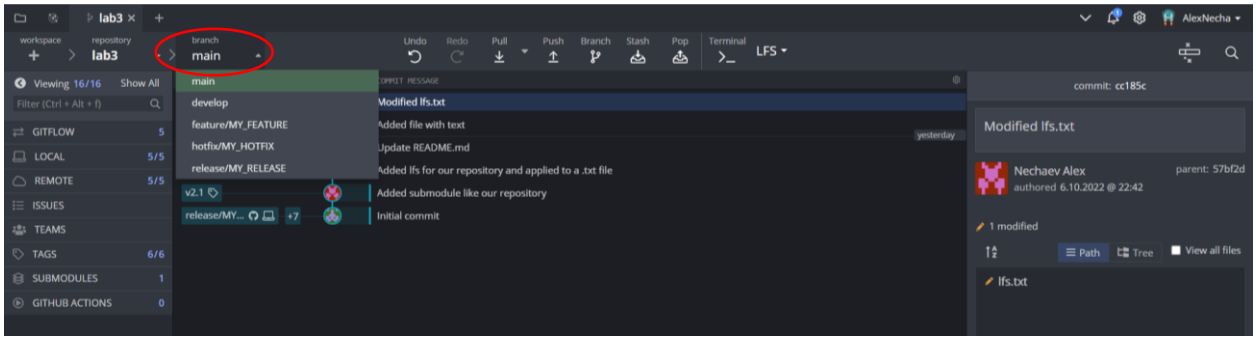

**3.** В GitGraken у нас все те же команды, но при этом у нас появляется более удобный ввод и вывод, а также мы можем видеть наши дерево веток сразу, а не в отдельном приложении.

Для примера покажем 10+ команд в графическом интерфейсе GitKraken:

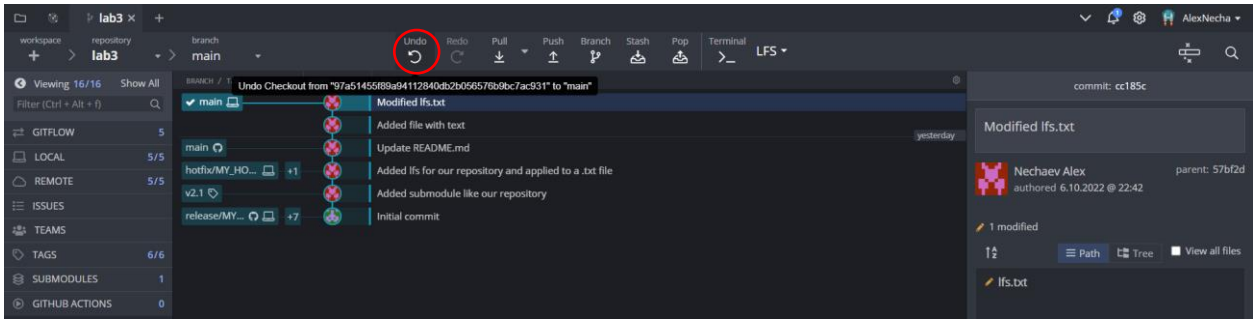
1) Переключение между репозиториями прямо в приложении:



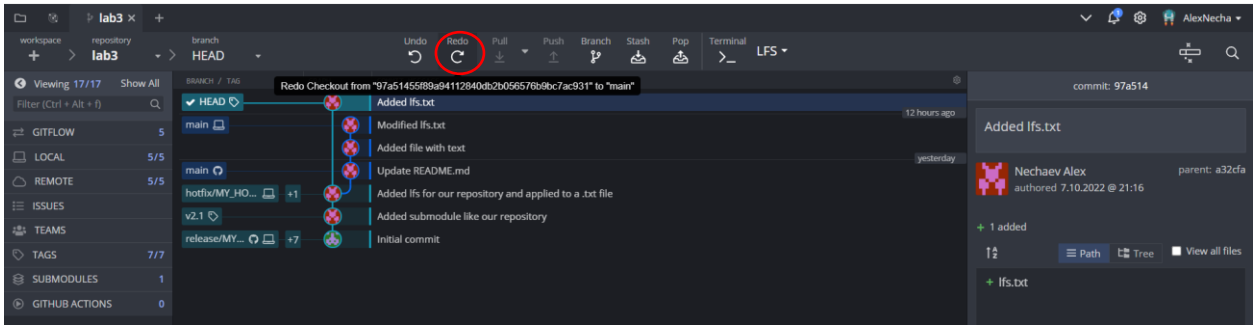

2) Переключение между ветками, не вводя команду для Git:



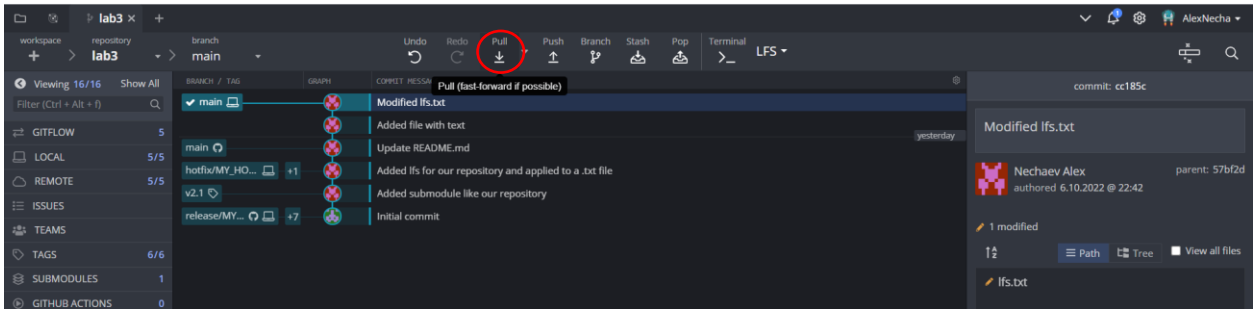
3) Можно как, например, в Word вернуться на шаг назад:



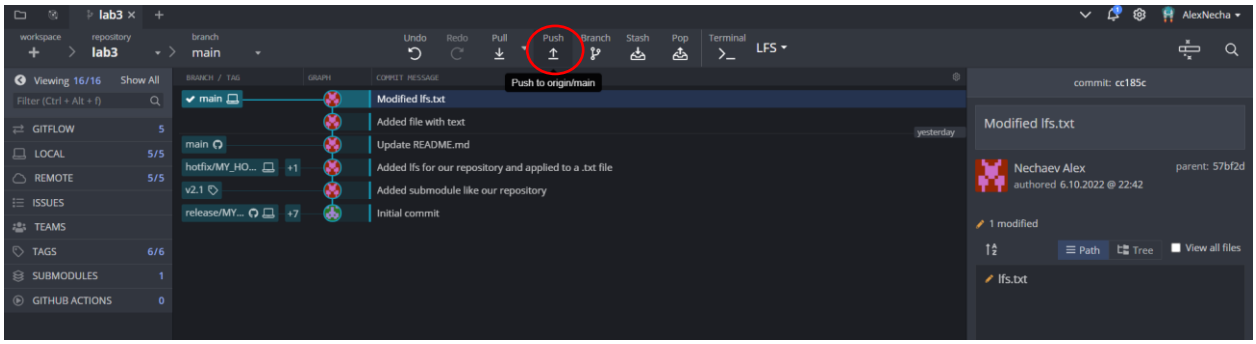

4) И также перейти на шаг вперед:



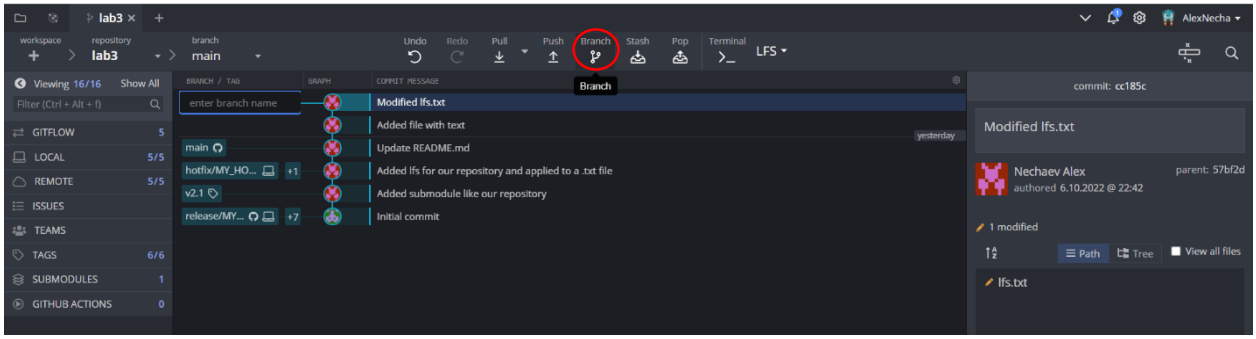

5) Автоматический git pull с удаленного репозитория:



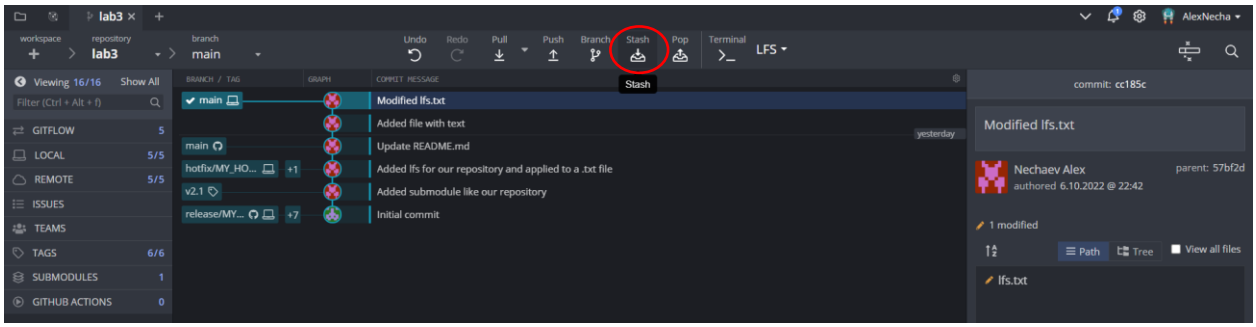

6) Автоматический git push в удаленный репозиторий:



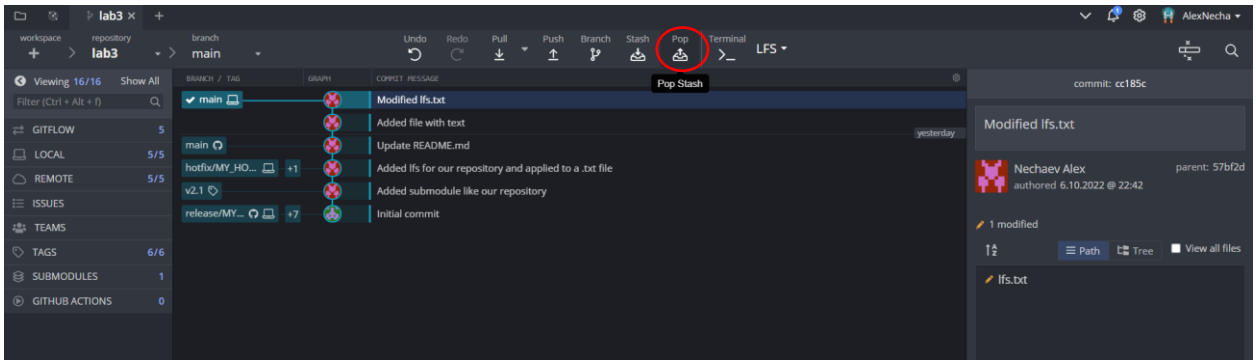
7) Создание ветки (также в самом дереве можно ее назвать):



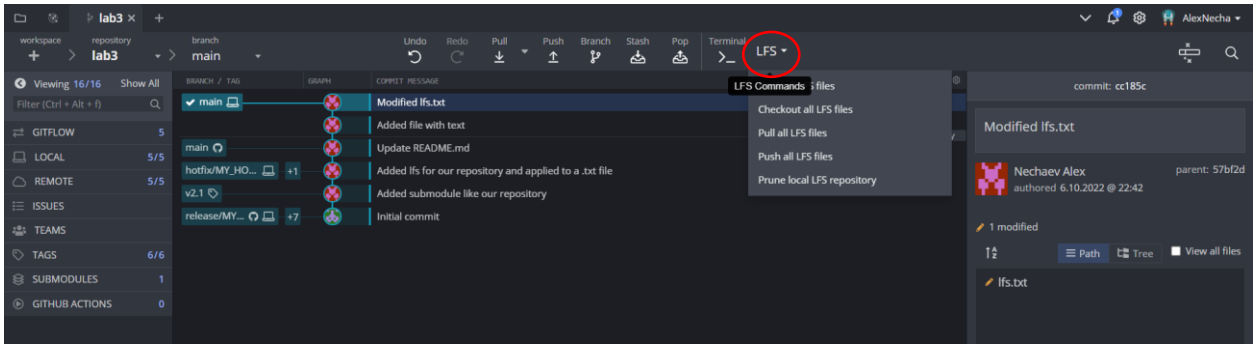

8) Команда временного скрытия текущих изменений в репозитории Git:



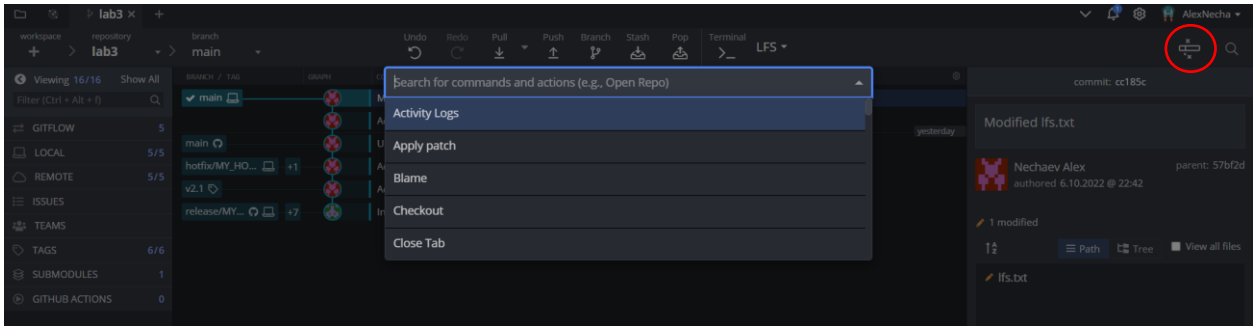

9) А когда мы готовы вернуться назад к незаконченной работе, git stash pop применяет назад ваши сохраненные изменения:



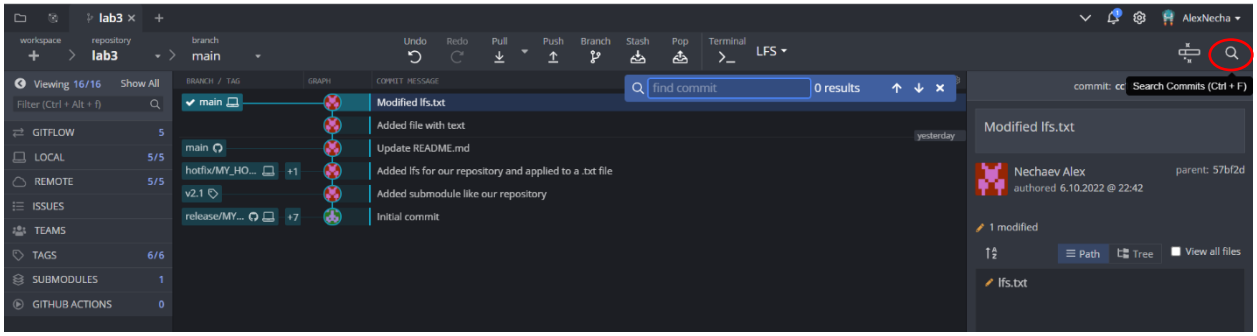

10) Можно применить автоматическую команду к определённой функции в Git:



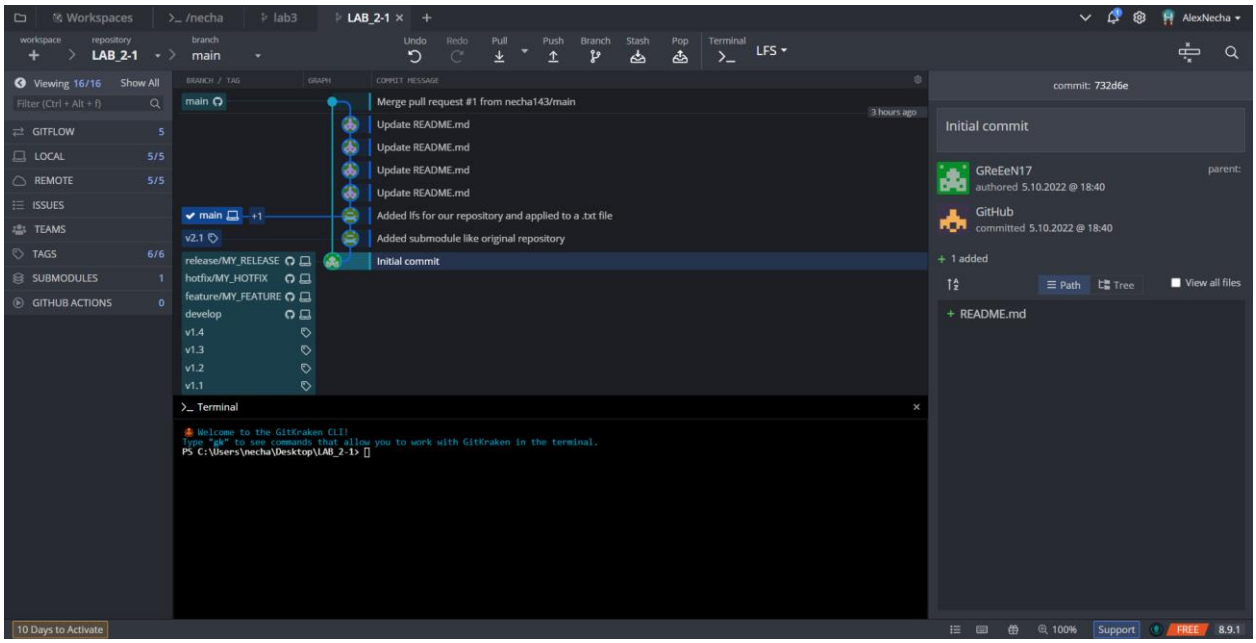
11) Не вводя команды для Git, репозиторием также можно управлять, через специальное окно с предложенными командами:



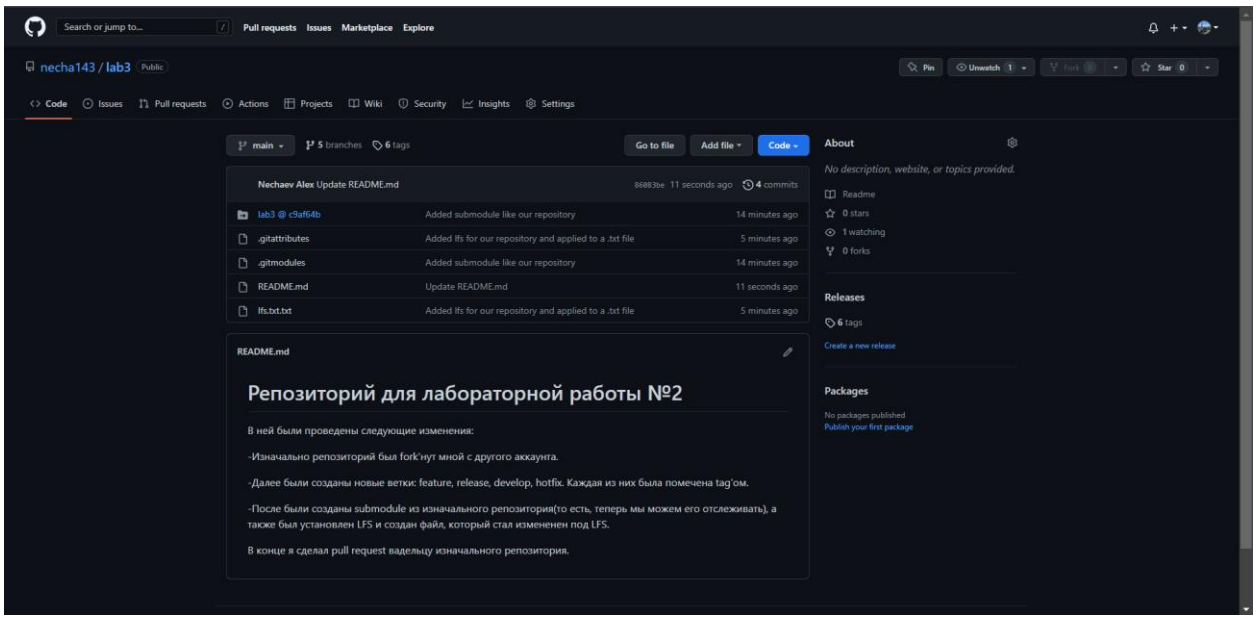

12) Нахождение коммита через дополнительное окно поиска:



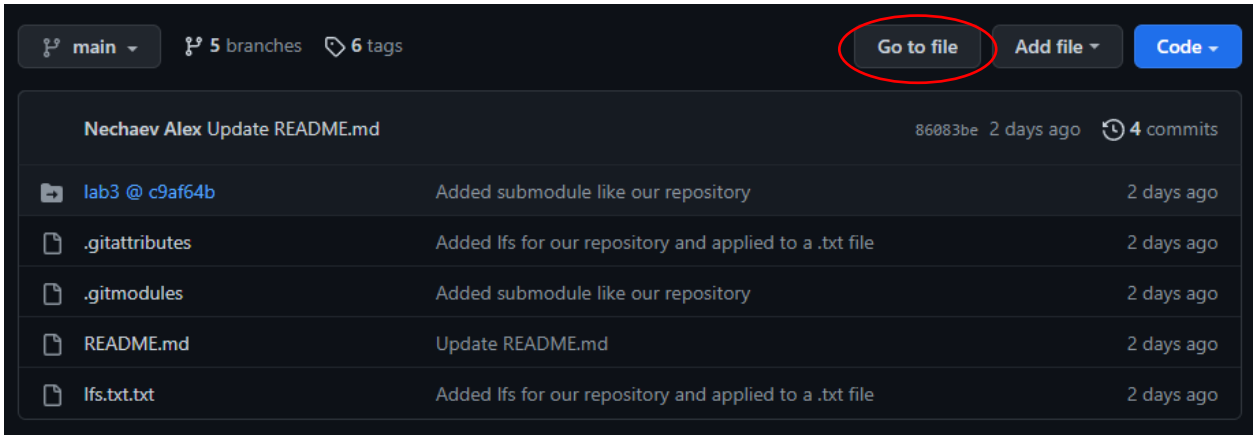

**4.** Если выполнять лабораторную работу №2 в GitKraken, то можно сразу следить за всеми изменениями в ветках через верхнее окно с деревом, это гораздо удобнее, нежели в обычном GitBash. А сам процесс работы никак не отличается, те же самые команды и функции Git.

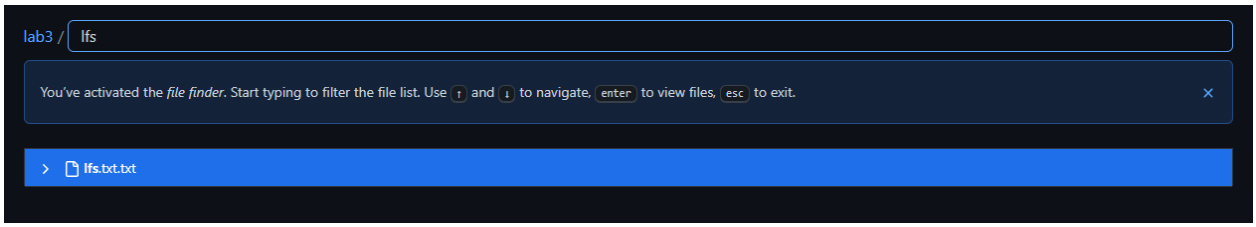



Выполнив все те же самые действия, как итог, видим то же самое в нашем репозитории на GitHub, что и в лабораторной работе №2:

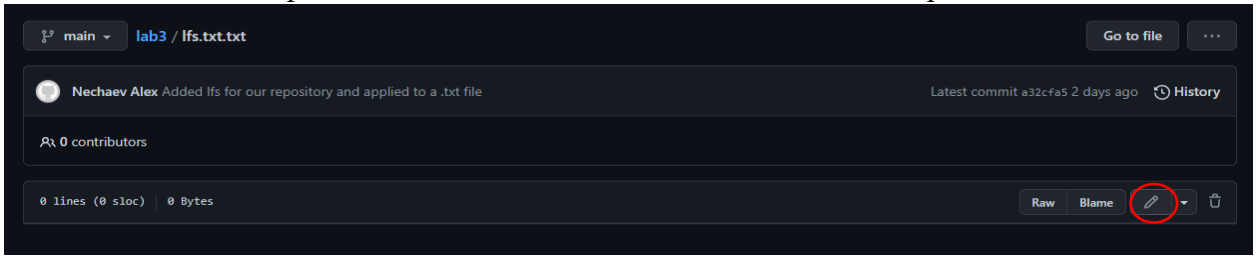


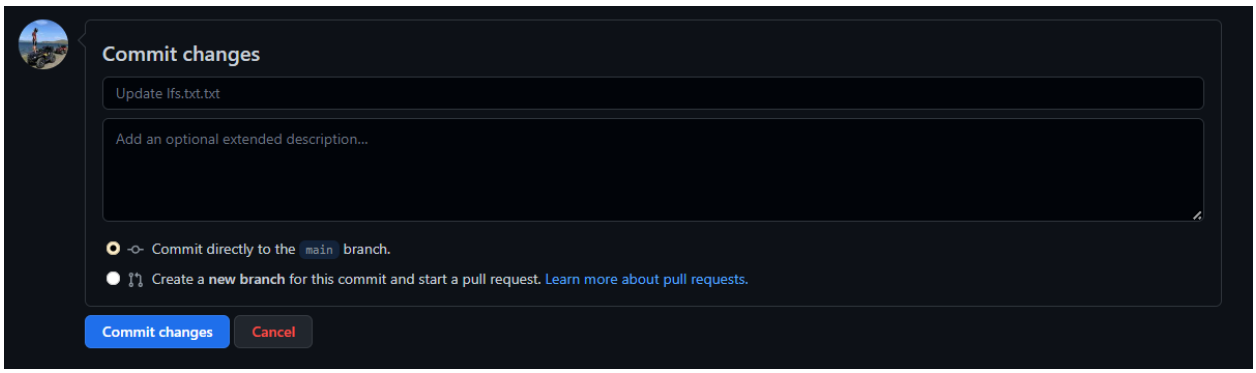

**5.** Поиск файла. В GitHub для поиска необходимо нажать на «Go to file», и в появившемся окне искать необходимый нам файл.



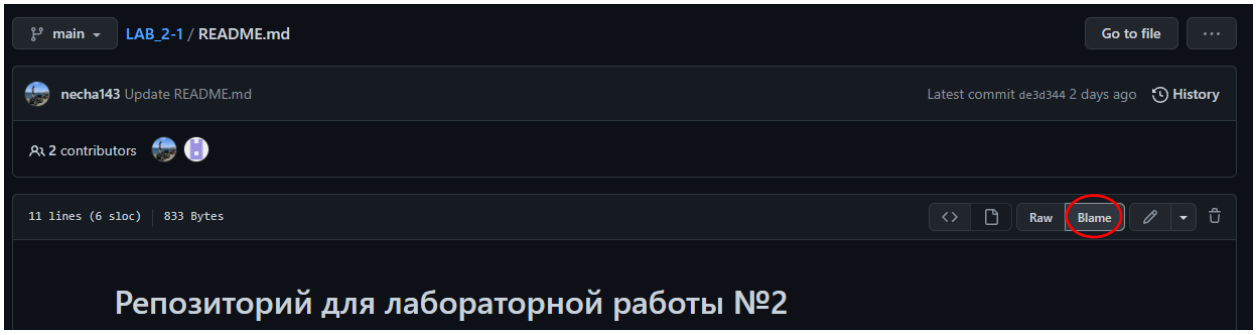



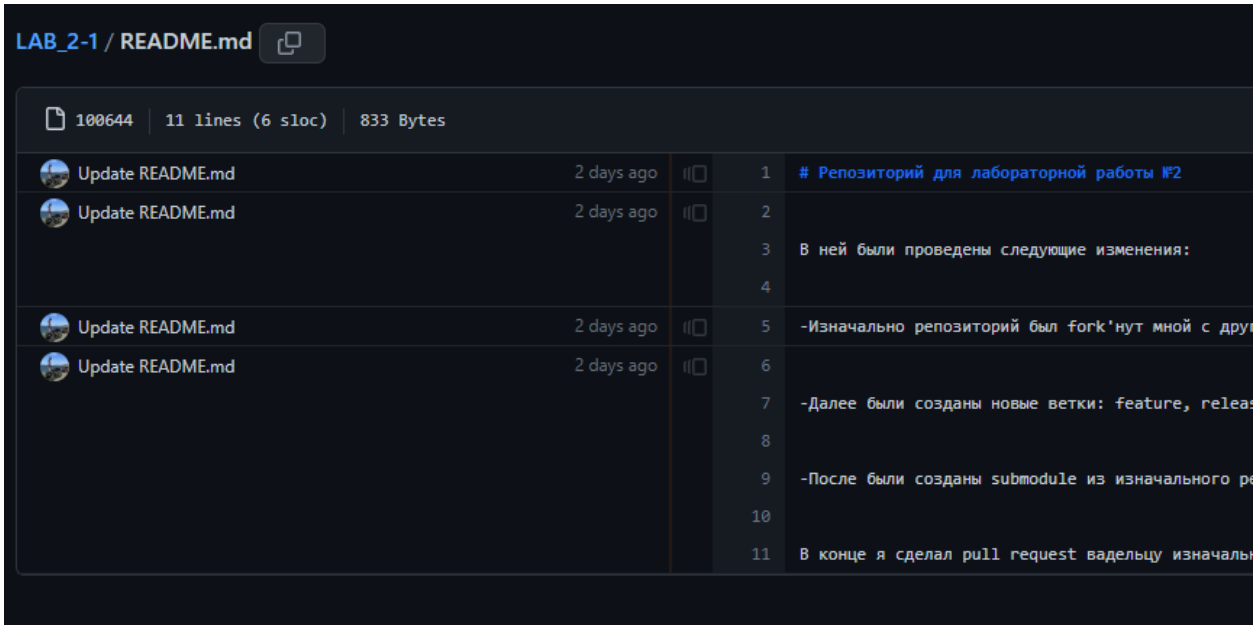

Внесение изменений в файл. В GitHub для этого необходимо нажать на «карандаш» и внести изменения в файл, затем обязательно это закоммитить и сохранить.



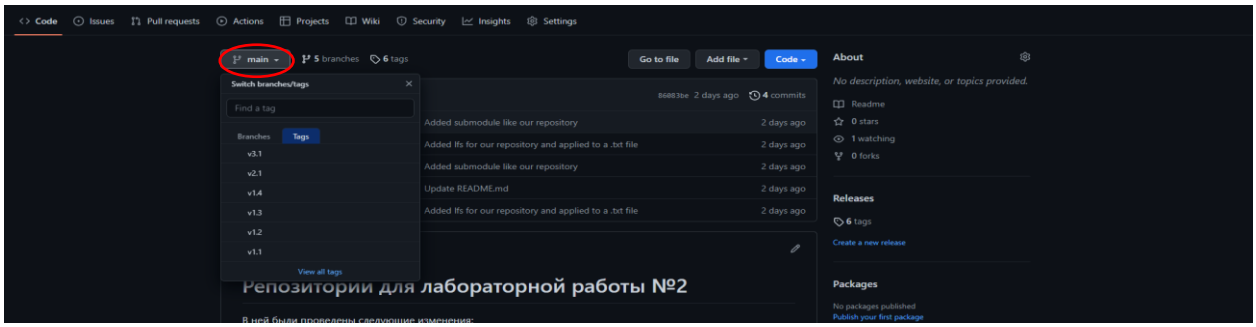


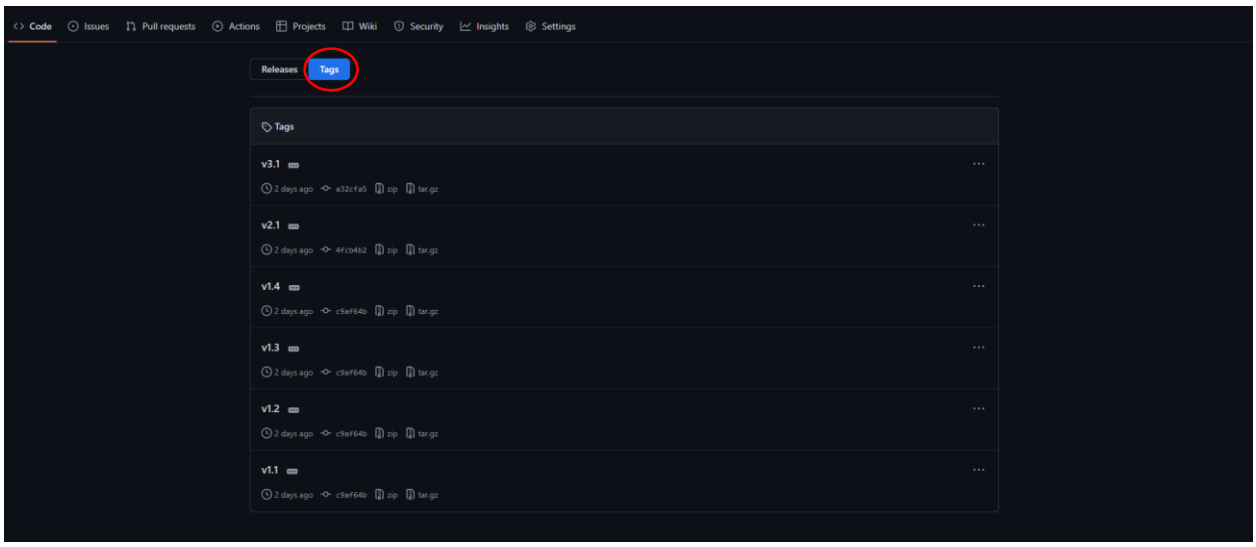

Отслеживание изменений файла. Нажмем на «blame» и увидим коммиты нашего файла, переключаясь по ним, мы сможем увидеть, какие изменения были внесены:




Отслеживание тэгов. Для этого в GitHub можно либо переключаться по ним, с помощью всплывающего окна, либо перейти в раздел с самими тэгами, и смотреть, какой текст подписан на самих тэгах:






**Вывод**

В ходе работы я познакомился с отличиями разных систем управлений репозиториями, установил и изучил графический интерфейс для Git под названием GitKraken, а также рассмотрел несколько функций GitHub.