**Практическая работа № 1**

**ОСНОВЫ GIT И GITHUB**

**Цель работы:** Выполнение практической работы направлено на изучение:

1. наиболее распространенных практик в области контроля версий программного обеспечения, его использования в командной разработке ПО и DevOps;

2. концепции Git, основанной на понятиях репозитория и ветвления версий ПО;

3. порядка использования GitHub и его базовых операций.

**Теоретические сведения:**

GitHub – один из самых популярных веб-сервисов для репозиториев Git.

- Git спроектирован как распределенная система контроля версий.

- Git в первую очередь сосредоточен на отслеживании исходного кода во время разработки.

- Git содержит элементы для координации действий программистов, отслеживания изменений и поддержки нелинейных рабочих процессов.

Git – это распределенная система контроля версий, которая используется для отслеживания изменений контента. Она служит центральной точкой для сотрудничества. В централизованной системе контроля версий каждый разработчик должен извлечь код из центральной системы и снова внести в него изменения.

GitHub – это онлайн-хостинг для репозиториев Git. GitHub – продукт дочерней компании Microsoft. GitHub предлагает бесплатные, профессиональные и корпоративные учетные записи. По состоянию на август 2019 года у GitHub было более 100 миллионов репозиториев.

Репозиторий – это структура данных для хранения документов, включая исходный код приложения.

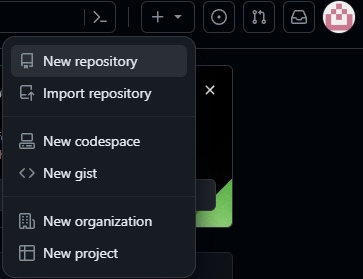
**Ход работы:**

1. Регистрация в GitHub. Нужно зайти на сайт GitHub, и произвести регистрацию на сайте. Сначала нужно определиться с именем, после этого пройти опрос и наконец подтвердить почту, которую мы привязали к учетной записи.

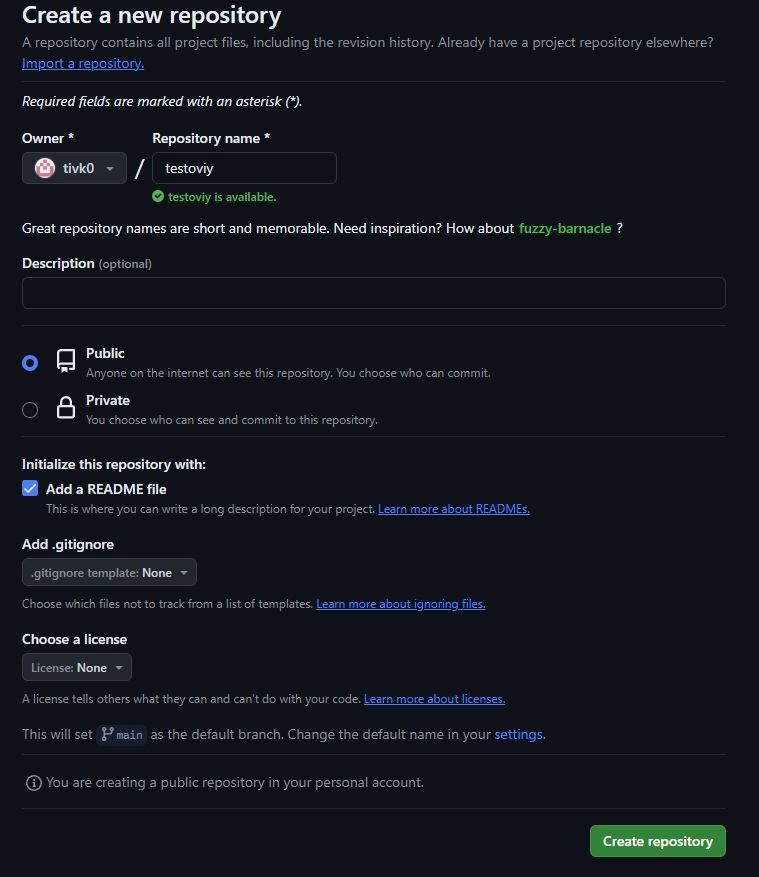
Рисунок 1. Страница регистрации

1. Создание репозитория вGitHub

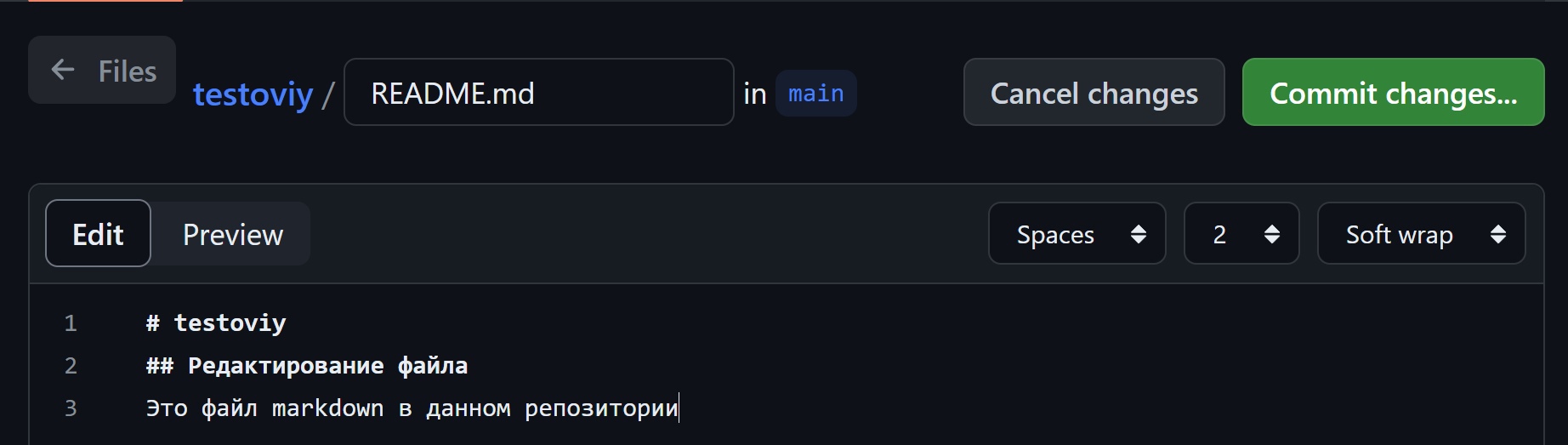
Нажать «+», затем нажать «New Repository».

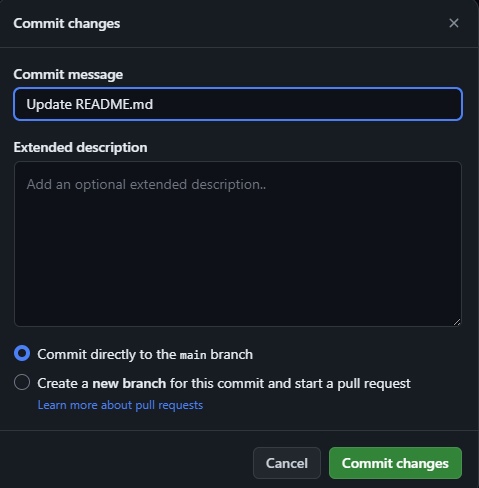
Рисунок 2. Создание репозитория

Чтобы создать новый репозиторий, необходимо заполнить следующие данные: имя, описание, выбрать видимость репозитория, выбрать опцию «Initialize this repository with a README».

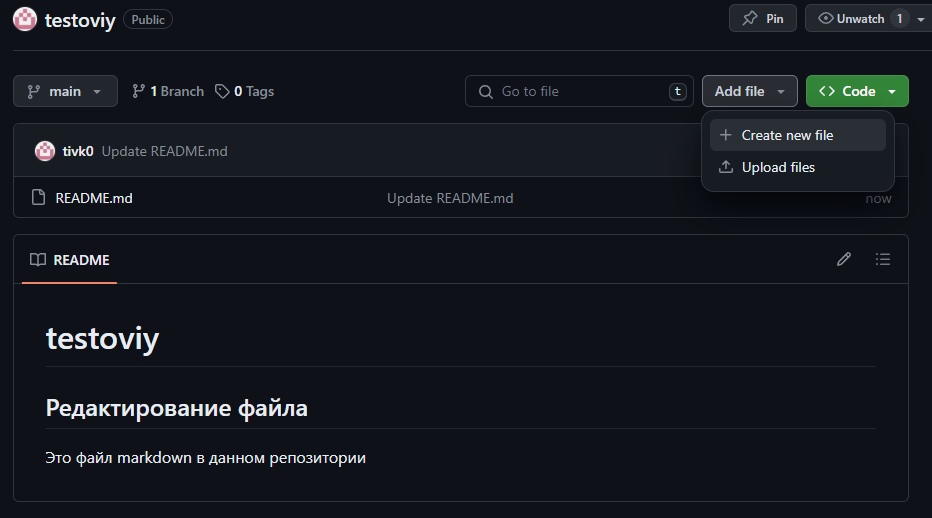
Рисунок 3. Создание репозитория

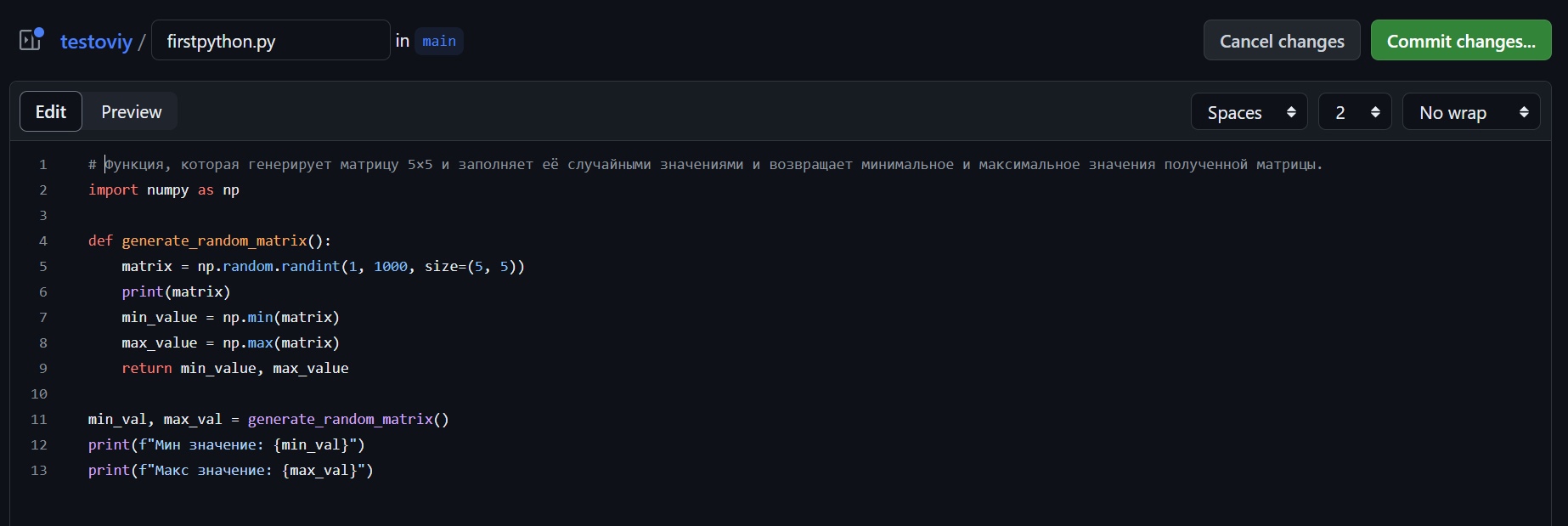
Следующим шагом мы должны отредактировать файл README. После внесения изменений прокрутить вниз до раздела «Commit changes». «Commit changes» используется для сохранения изменений в репозитории.

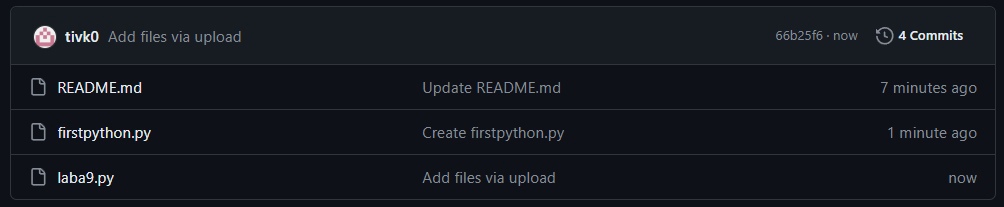
Рисунок 4. Редактирование файла README

Рисунок 5. Подтверждение изменений

Создание нового файла с помощью встроенного веб-редактора GitHub, который запускается в браузере. Нажимаем «Add File», затем нажмите «Create New File», чтобы создать новый файл. Создаем файл Python с именем firstpython.py. Сначала указываем имя файла. Затем добавляем комментарий, описывающий наш код, после этого добавляем код. Позже зафиксируем изменения в репозитории. Мы можете видеть, что наш файл теперь добавлен в репозиторий, а в списке репозитория показано, когда файл был добавлен или изменен.

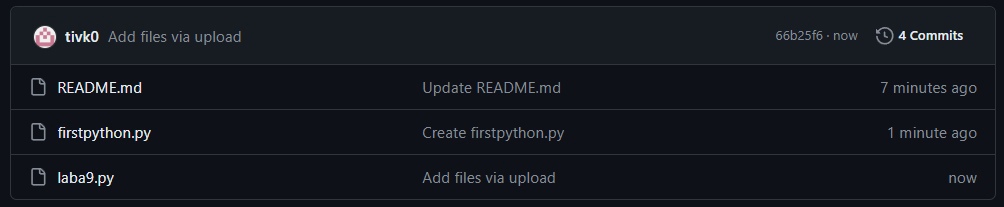
Рисунок 6. Создание файла

Рисунок 7. Создание файла

Рисунок 8. Добавление файла

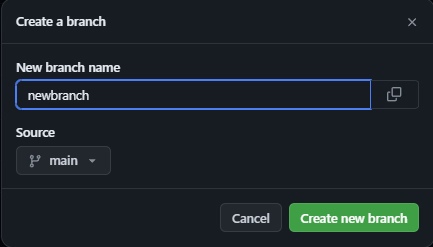
Загрузить файл из локальной системы в репозиторий.

Рисунок 9. Файл в локальной системе

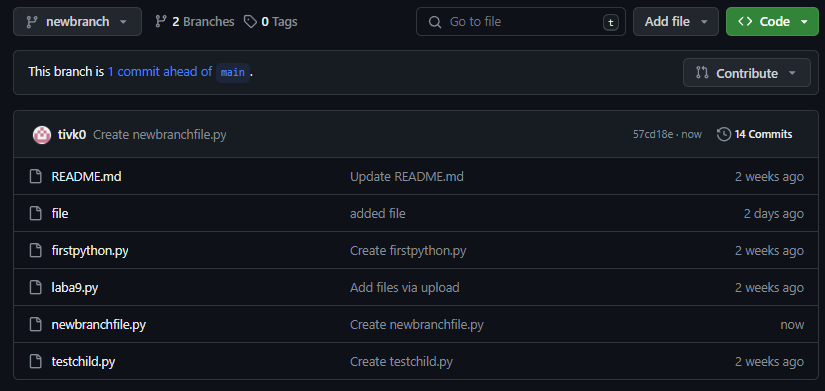
Рисунок 10. Добавление файла из локальной сети

1. Создание branch

В верхней части списка файлов найти раскрывающееся меню «branch». Щелкнуть раскрывающееся меню, ввести имя ветки и нажать Enter на клавиатуре.

Рисунок 11. Создание второй branch

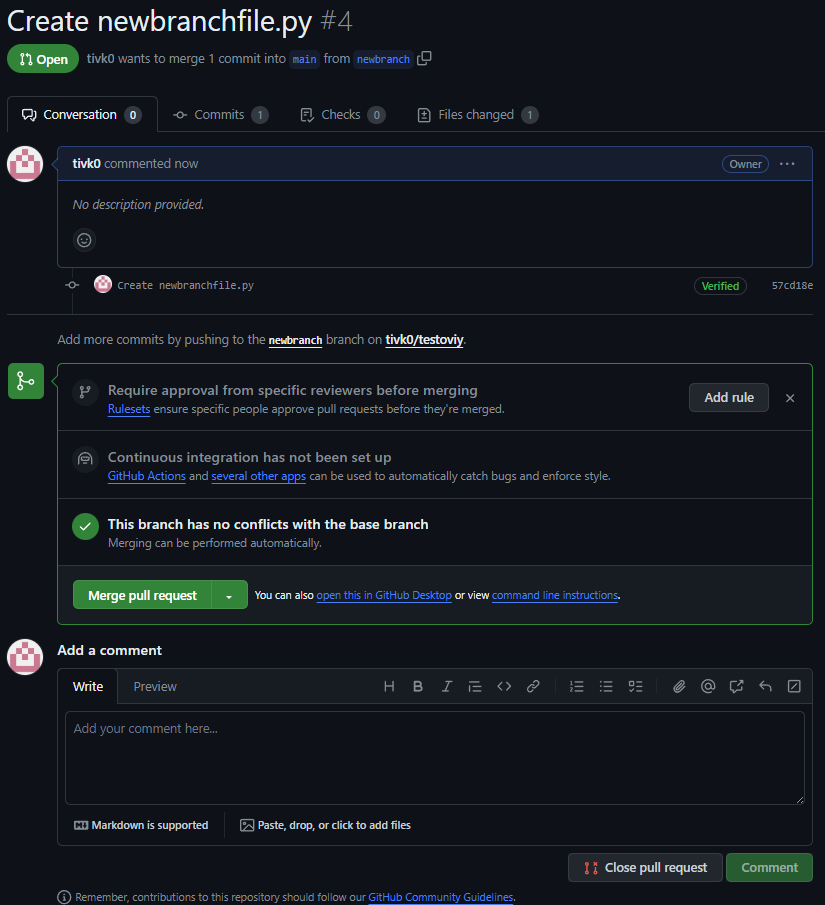
Добавить файл в новую ветку.

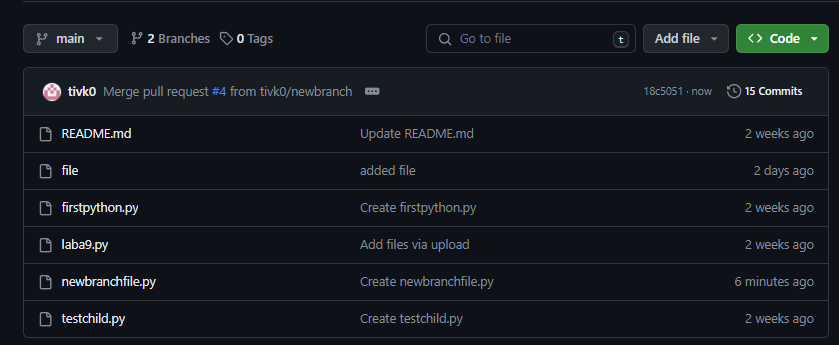
Рисунок 12. Добавление файла newbranchfile

Сравнить две ветки и открыть запрос на включение, который позволит вам скопировать изменения, внесенные вами в дочернюю ветку (в данном случае добавление нового файла), в основную ветку. В newbranch нажать кнопку «Compare & pull request».

Рисунок 13. Compare & pull request

4. Операция «Merge» на базе «pull request». Чтобы объединить ветки по запросу pull request в проекте, выполнить следующие действия: Открыть вкладку «Pull requests». Щелкнуть pull request, который вы хотите объединить с основным проектом. Если вас устраивают изменения, нажмите «Merge pull request», чтобы принять запрос на включение и объединить обновления.

Рис.14. Операция «*Merge*»

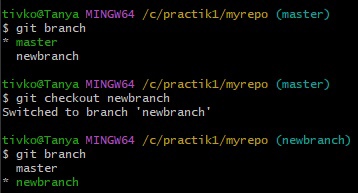
Рис 15. Результат слияния

**Задание №1:**

5. Работа с локальным репозиторием посредством командной строки. Данный вариант доступен либо в Windows PowerShell через специальное приложение Git. Создаем директорию для локального репозитория: Создайте каталог myrepo, скопировав и вставив в терминал приведенную ниже команду mkdir*: «*mkdirmyrepo». Перейдите в каталог myrepo, скопировав и вставив команду cd: «cd myrepo». В этом каталоге myrepoможно создать новый локальный репозиторий git с помощью команды git init. Скопируйте и вставьте в терминал команду: «git init» Теперь создан новый локальный репозиторий, который вы можете проверить, выполнив список каталогов, вставив следующую команду в окно терминала: «ls». Дальше создаем новую ветку newbranch.

Рисунок 16. Создание ветки

После этого новую ветку делаем активной.

Рисунок 17. Новая ветка — активна

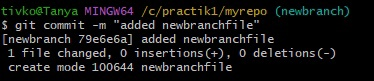
Создайте пустой файл с именем newbranchfile.

Рисунок 18. Создание файла

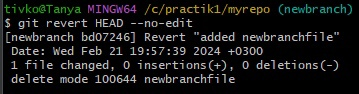
Добавьте вновь созданный файл в свою ветку

Рисунок 19. Создание файла

Зафиксируйте изменения в новой ветке.

Рисунок 20. Фиксация изменений

Отмените последние зафиксированные изменения.

Рисунок 21. Отмена зафиксированных изменений

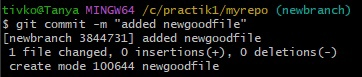
Создайте новый файл с именем newgoodfile.

Рисунок 22. Создание файла

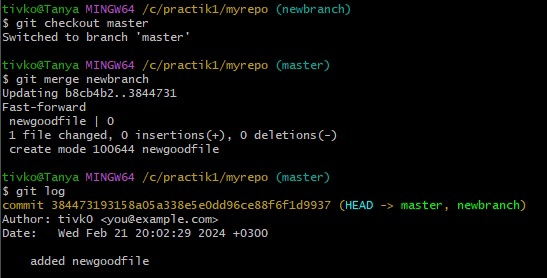
Добавьте последний файл в новую ветку.

Рисунок 23. Добавление файла

Зафиксируйте изменения.

Рисунок 24. Фиксация изменений

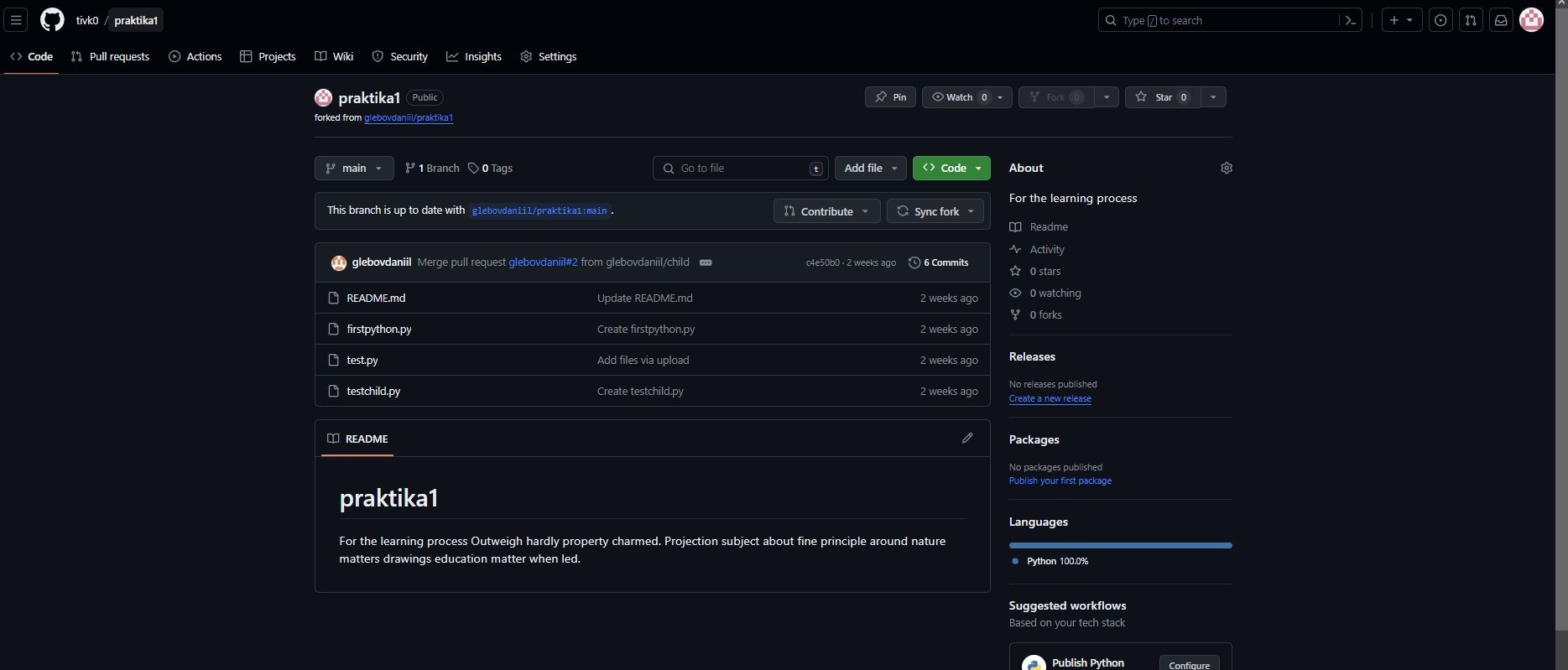
Объедините изменения в новой ветке с основной.

Рисунок 25. Объединение веток

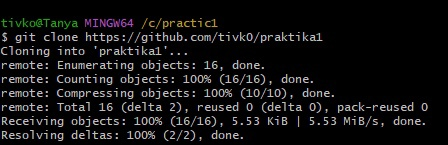
**Задание №2:**

1.Сделать fork проекта второго студента.

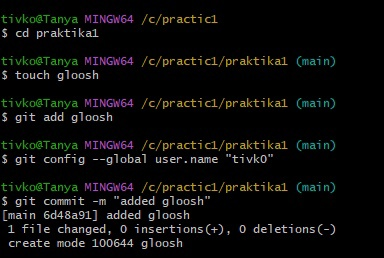
Зайти в чужой проект и сделать fork*.*

Рисунок 26. fork проекта второго студента

2. Сделать clone проекта в локальный репозиторий. Зайти в GitBashвести команду «git clone (ссылка на репозиторий)».

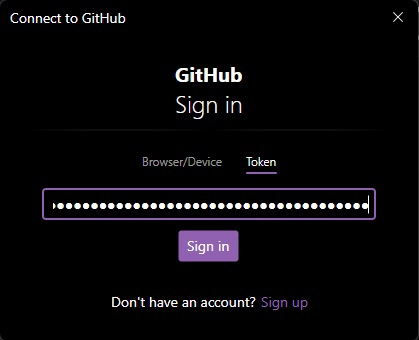
Рисунок 27. Локальный clone проекта

3. Добавление и фиксирование файла.

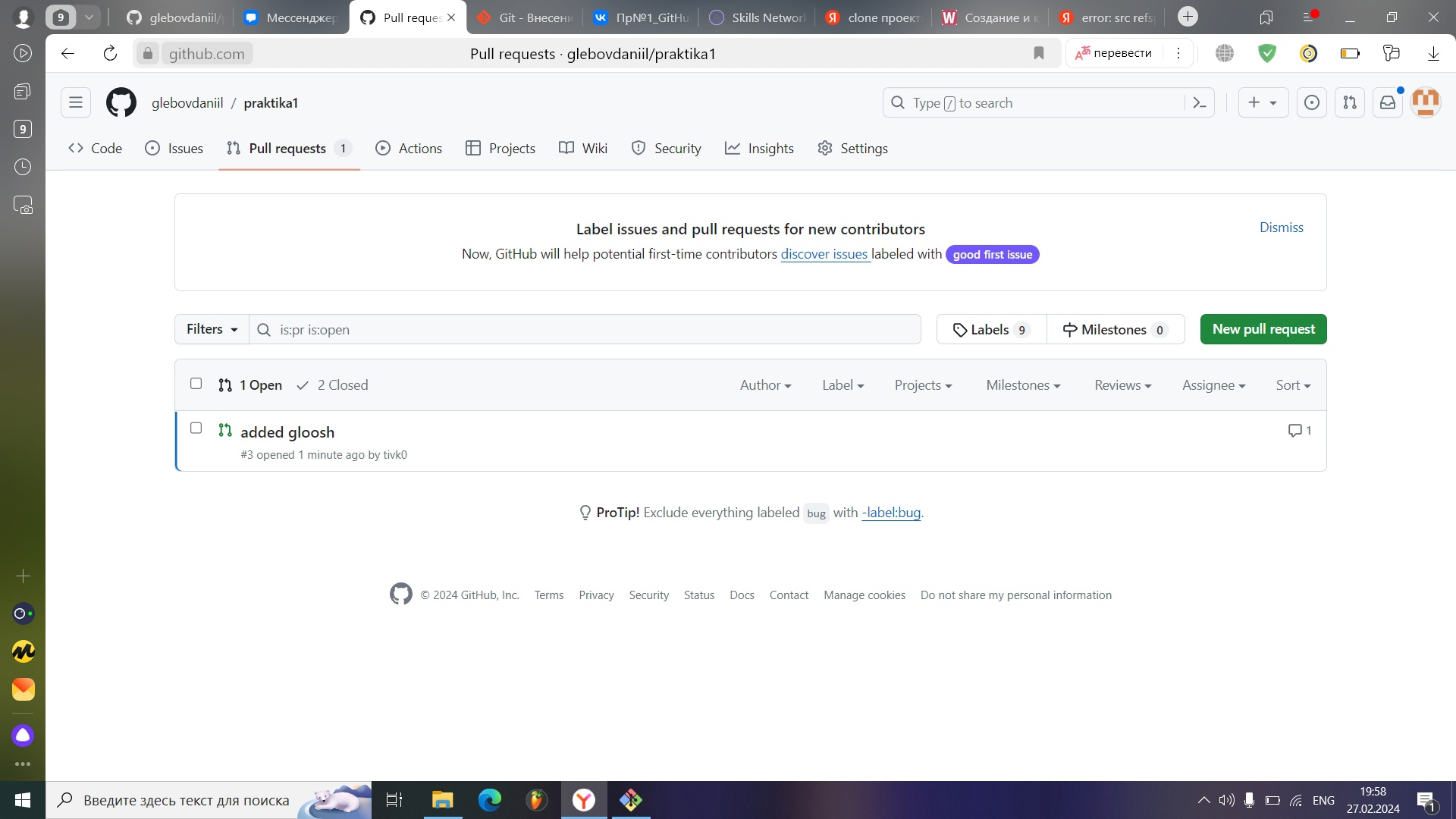
Рисунок 28. Добавление и фиксирование файла

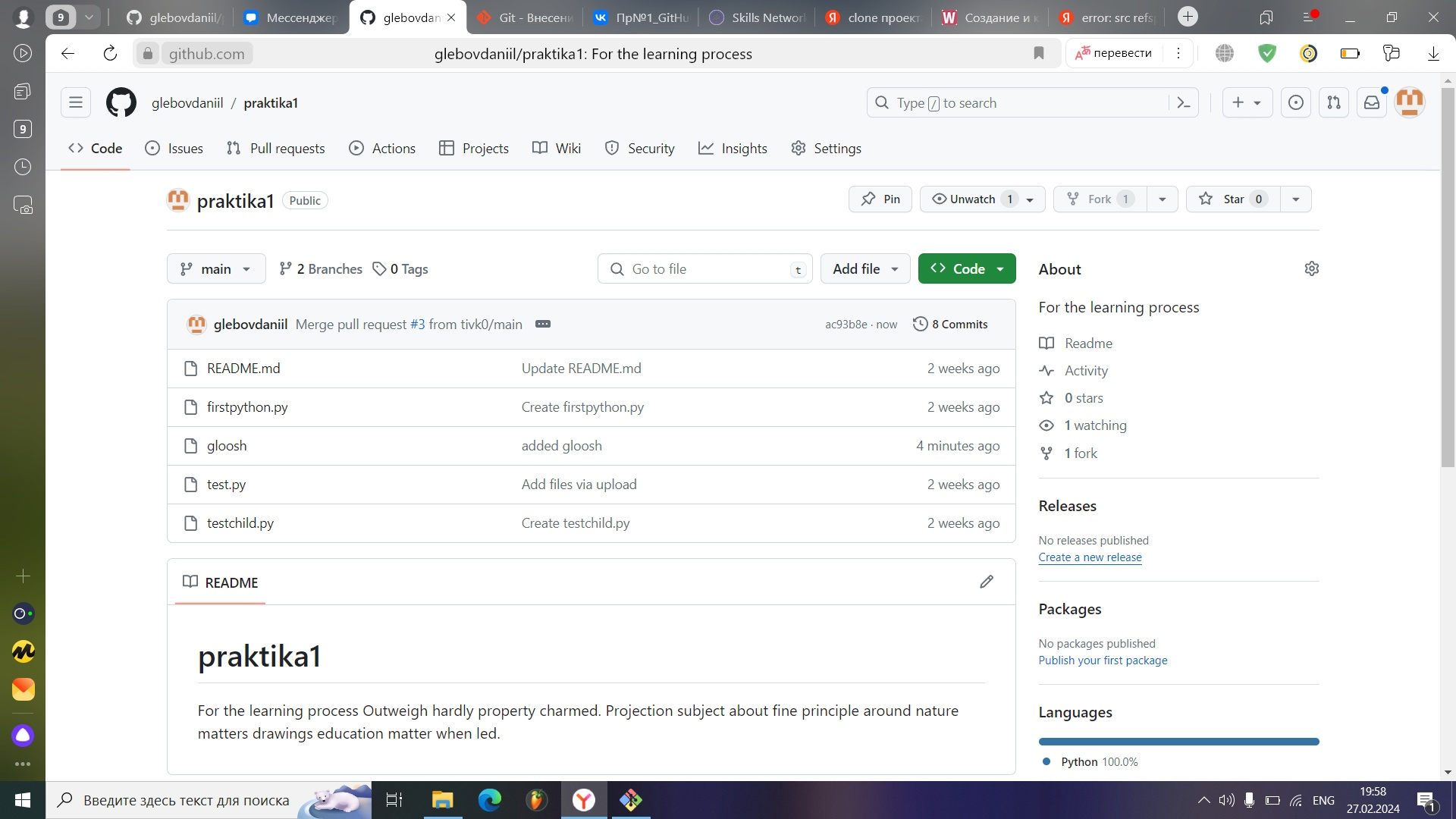
1. Синхронизация с fork-репозиторием. Для git push понадобится имя пользователя и не простой пароль пользователя, а token (classic).



Рисунок 29. Синхронизация

5. Формирование pull request к origin проекту на прием данного изменения.

Рисунок 30. Pull request

Рисунок 31. Итог принятия изменений у другого пользователя

**Вывод**: В результате практической работы, были получены начальные навыки работы с Gitи *G*ithub*.* Была произведена регистрация, создание репозитория и новой ветки, добавление созданного файла в новую ветку, фиксация изменения в ветке, объединение изменения в новой ветке с основной; *fork* проекта второго студента. Мы клонировали проект в локальный репозиторий, добавили и фиксировали, отменяли фиксацию файла, синхронизировались с *fork-*репозиторием, формировали pull requestдляприема изменений.