|  |
| --- |
|  |

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«МИРЭА – Российский технологический университет»**

**РТУ МИРЭА**

|  |
| --- |
|  |

Институт информационных технологий (ИТ)

Кафедра математического обеспечения и стандартизации информационных технологий (МОСИТ)

**ОТЧЕТ**

**ПО ПРАКТИЧЕСКОЙ РАБОТЕ №1**

**«Системы контроля версий»**

по дисциплине

**«Технология разработки программных приложений»**

|  |  |
| --- | --- |
| Выполнил студент группы ИНБО-03-22 | Алтухов А.В. |
|  |  |
| Принял ассистент кафедры МОСИТ | Исабекова О.А. |

Москва 2024

**СОДЕРЖАНИЕ**

[ЦЕЛЬ РАБОТЫ 3](#_Toc160143016)

[ПОСТАНОВКА ЗАДАЧИ 4](#_Toc160143017)

[ЧАСТЬ 1 6](#_Toc160143018)

[ЧАСТЬ 2 16](#_Toc160143019)

[ЧАСТЬ 3 28](#_Toc160143020)

[ОТВЕТЫ НА ВОПРОСЫ 38](#_Toc160143021)

[ВЫВОД 40](#_Toc160143022)

# **ЦЕЛЬ РАБОТЫ**

Получить навыки по работе с командной строкой и git’ом.

# **ПОСТАНОВКА ЗАДАЧИ**

**Часть 1. Основные команды Git**

1. Установите и настройте клиент git на своей рабочей станции.

2. Создайте локальный репозиторий и добавьте в него несколько файлов.

3. Внесите изменения в один из файлов.

4. Проиндексируйте изменения и проверьте состояние.

5. Сделайте коммит того, что было проиндексировано в репозиторий. Добавьте к коммиту комментарий.

6. Измените еще один файл. Добавьте это изменение в индекс git. Измените файл еще раз. Проверьте состояние и произведите коммит проиндексированного изменения. Теперь добавьте второе изменение в индекс, а затем проверьте состояние с помощью команды git status. Сделайте коммит второго изменения.

7. Просмотрите историю коммитов с помощью команды git log. Ознакомьтесь с параметрами команды и используйте некоторые из них для различного формата отображения истории коммитов.

8. Верните рабочий каталог к одному из предыдущих состояний.

9. Изучите, как создавать теги для коммитов для использования в будущем.

10. Отмените некоторые изменения в рабочем каталоге (до и после индексирования).

11. Отмените один из коммитов в локальном репозитории.

**Часть 2. Системы управления репозиториями**

1. Создайте аккаунт на GitHub (у кого нет).

2. Создайте репозиторий на GitHub и на локальной машине, согласно выбранной теме проекта.

3. Создайте несколько файлов на локальной машине при помощи консоли.

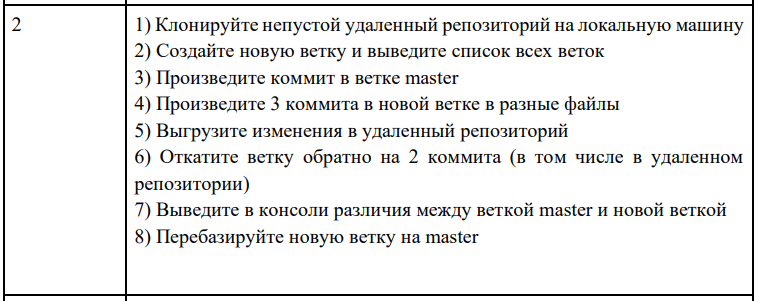
4. Создайте SSH-ключ для авторизации.

5. Свяжите репозиторий локальной машины с репозиторием на GitHub при помощи консоли.

6. Создайте новую ветку в репозитории с помощью команды, произведите в ней какие-нибудь изменения, а после слейте с веткой master.

7. Выполните цепочку действий в репозитории, согласно вариантам.

№ варианта: 2



**Часть 3. Работа с ветвлением и оформление кода**

1. Сделайте форк репозитория в соответствии с вашим вариантом

2. Склонируйте его на локальную машину

3. Создайте две ветки branch1 и branch2 от последнего коммита в master'е

4. Проведите по 3 коммита в каждую из веток, которые меняют один и тот же кусочек файла

5. Выполните слияние ветки branch1 в ветку branch2, разрешив конфликты при этом

6. Выгрузите все изменения во всех ветках в удаленный репозиторий

7. Проведите еще 3 коммита в ветку branch1

8. Склонируйте репозиторий еще раз в другую директорию

9. В новом клоне репозитории сделайте 3 коммита в ветку branch1

10. Выгрузите все изменения из нового репозитория в удаленный репозиторий 11. Вернитесь в старый клон с репозиторием, выгрузите изменения с опцией --force

12. Получите все изменения в новом репозитории.

№ варианта: 2

https://github.com/google/model\_search

# **ЧАСТЬ 1**

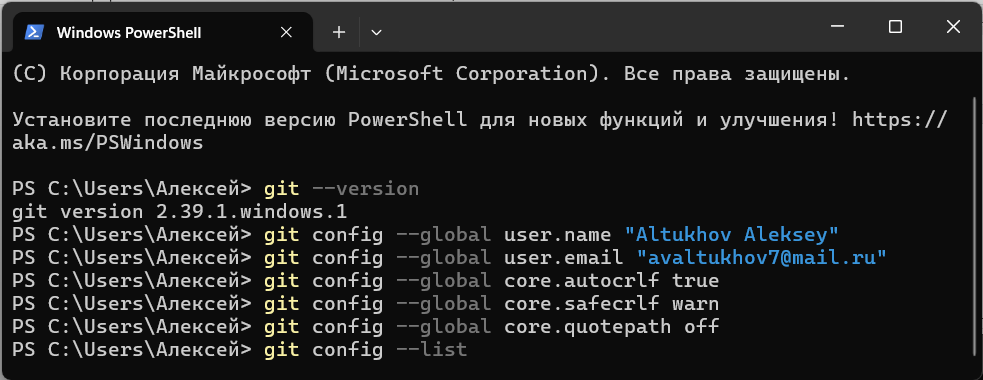
1. Установите и настройте клиент git на своей рабочей станции. 

Рисунок 1 – Настройка Git

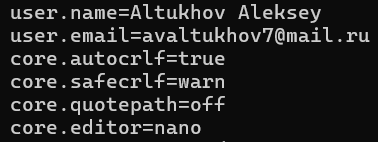


Рисунок 2 – Проверка правильности настройки Git

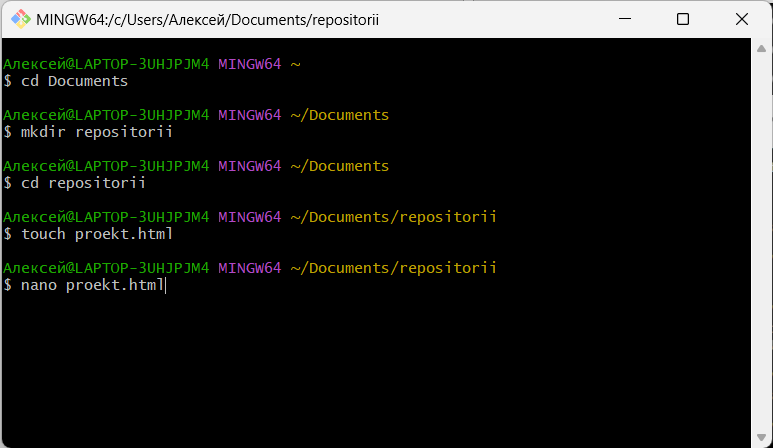


Рисунок 3 – Создание новой папки и файла

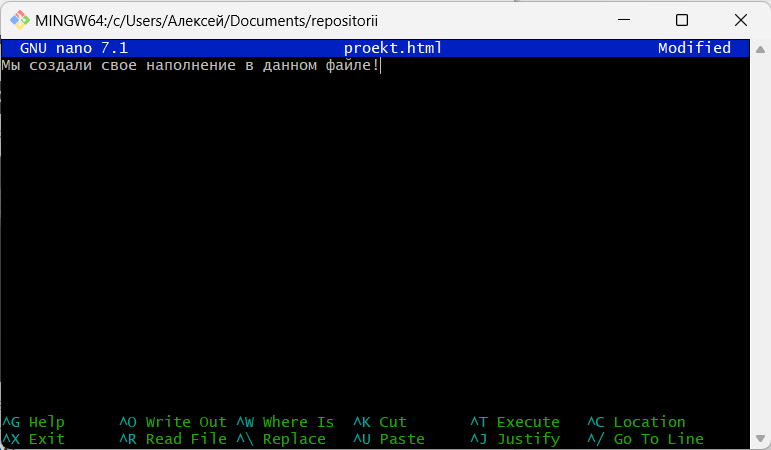


Рисунок 4 – Добавление содержания в созданный файл

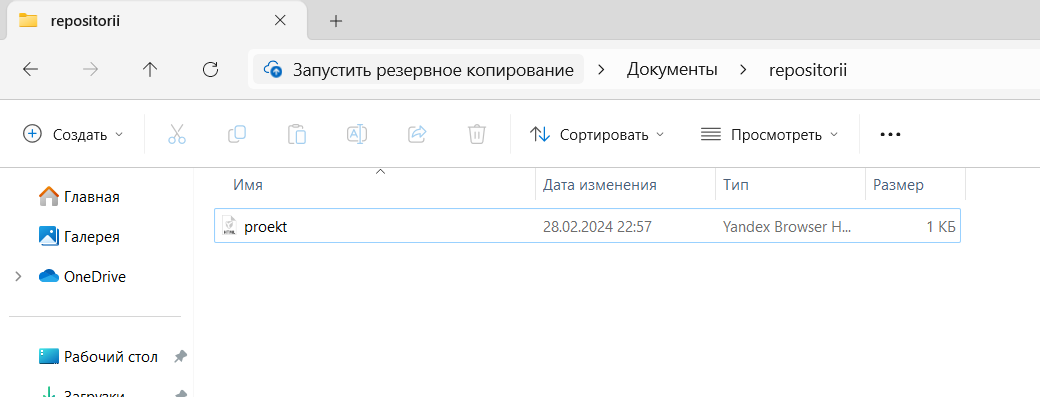


Рисунок 5 – Созданный файл

2. Создайте локальный репозиторий и добавьте в него несколько файлов.



Рисунок 6 – Создание репозитория

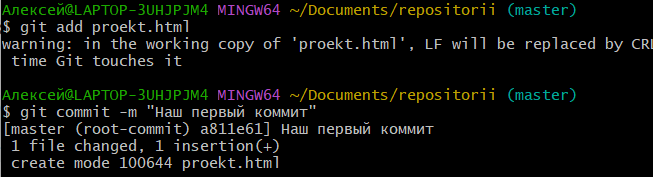


Рисунок 7 – Добавление файла в репозиторий

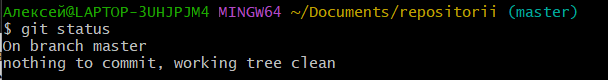


Рисунок 8 – Проверка состояния репозитория

3. Внесите изменения в один из файлов.

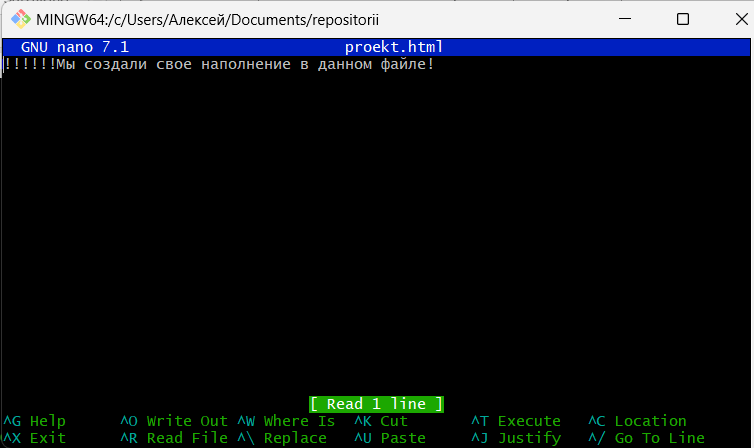


Рисунок 9 – Внесение изменений в файл

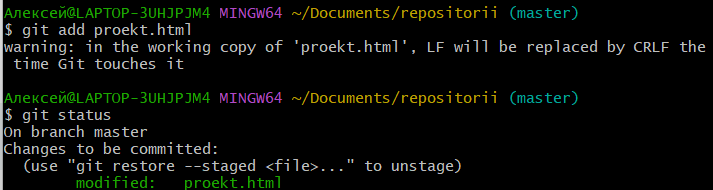
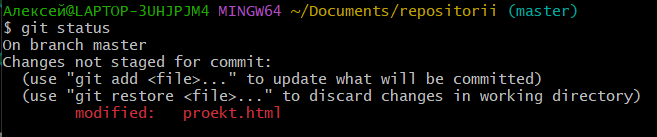


Рисунок 10 – Проверка состояния репозитория после внесения изменений

4. Проиндексируйте изменения и проверьте состояние

5. Сделайте коммит того, что было проиндексировано в репозиторий. Добавьте к коммиту комментарий.

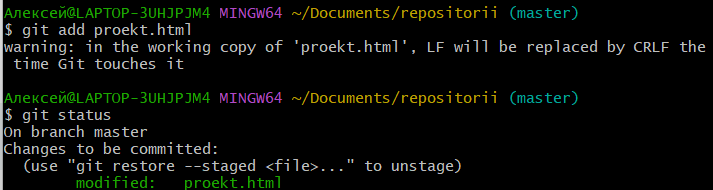


Рисунок 11 – Индексация изменений

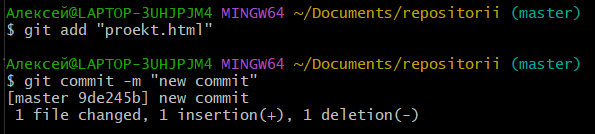


Рисунок 12 – Выполнение коммита

6. Измените еще один файл. Добавьте это изменение в индекс git. Измените файл еще раз. Проверьте состояние и произведите коммит проиндексированного изменения.

Теперь добавьте второе изменение в индекс, а затем проверьте состояние с помощью команды git status. Сделайте коммит второго изменения.

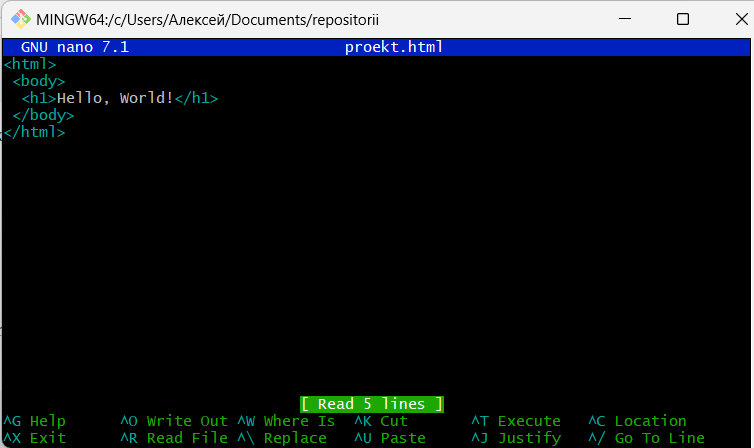


Рисунок 13 – Новое внесение изменений в файл

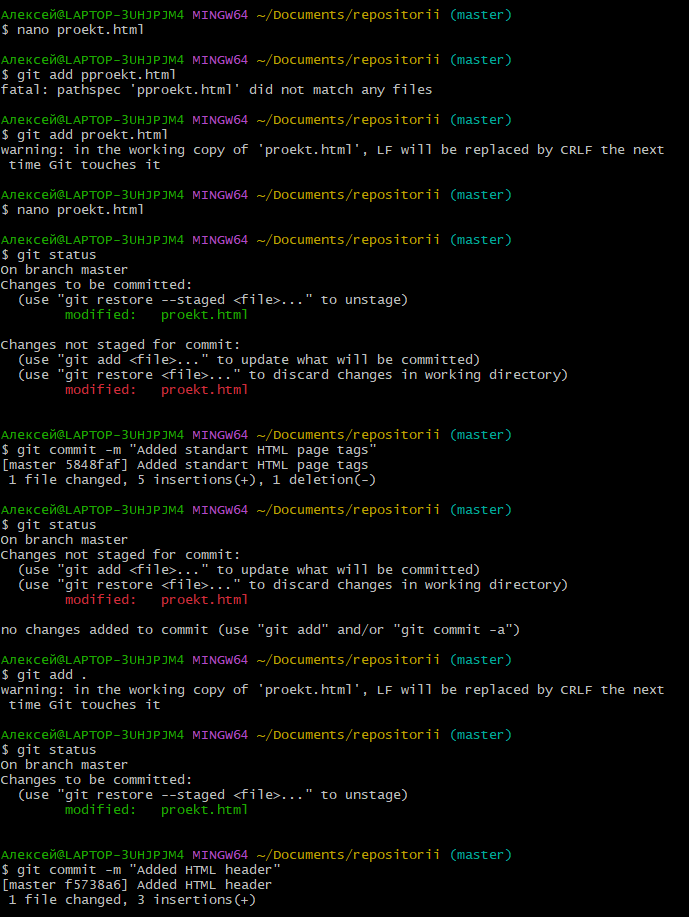


Рисунок 14 – Проверка статуса репозитория, добавление коммита

7. Просмотрите историю коммитов с помощью команды git log. Ознакомьтесь с параметрами команды и используйте некоторые из них для различного формата отображения истории коммитов.

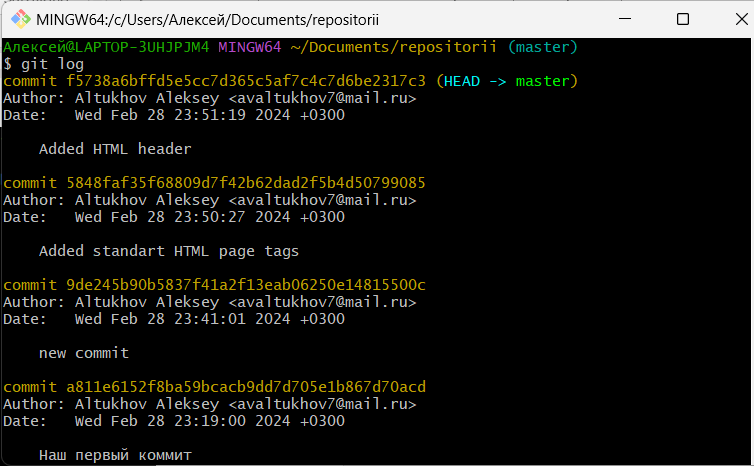


Рисунок 15 – Просмотр истории коммитов

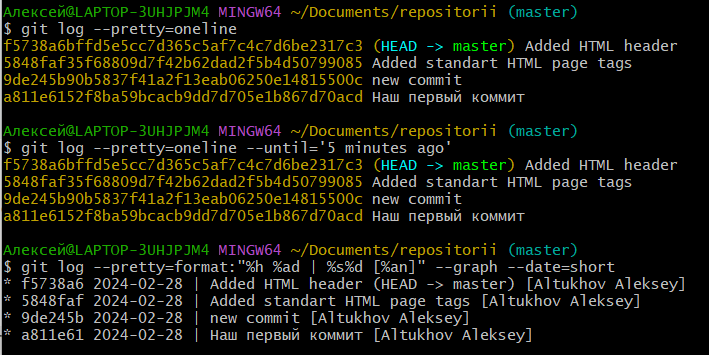


Рисунок 16 – Изменение формата вывода информации о коммитах

8. Верните рабочий каталог к одному из предыдущих состояний.

9. Изучите, как создавать теги для коммитов для использования в будущем.

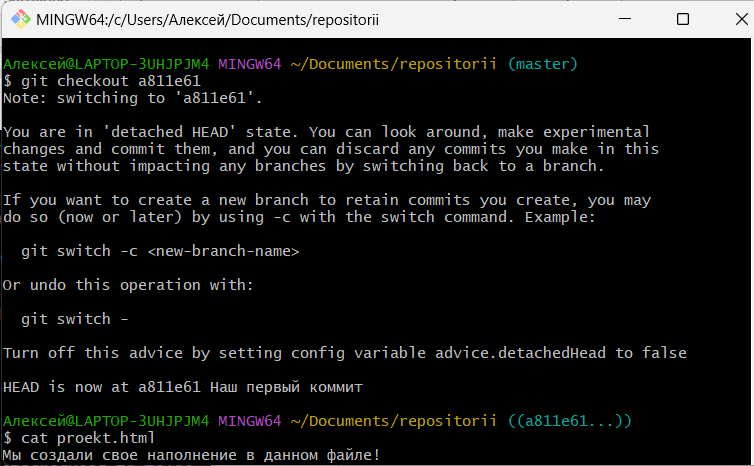


Рисунок 17 – Получение старой версии каталога

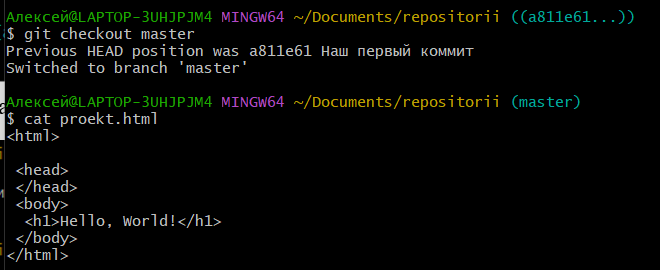


Рисунок 18 – Возврат в нынешнее состояние

10. Отмените некоторые изменения в рабочем каталоге (до и после индексирования).

11. Отмените один из коммитов в локальном репозитории.

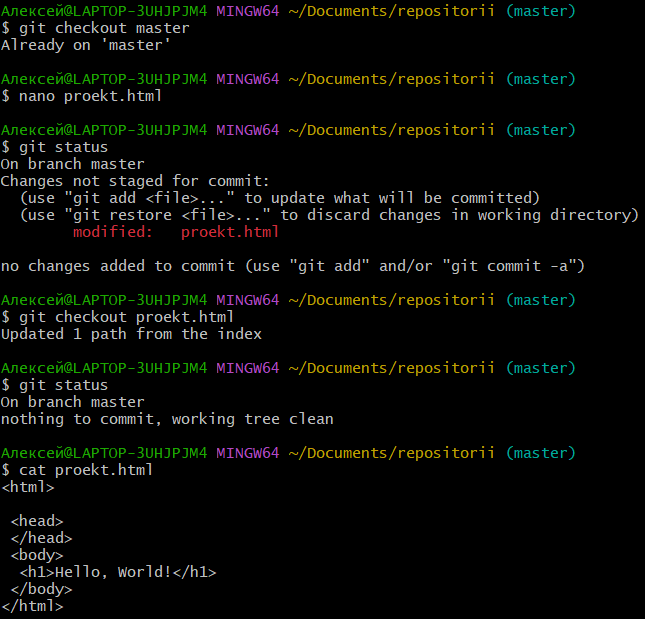


Рисунок 19 – Внесение и отмена локальных изменений

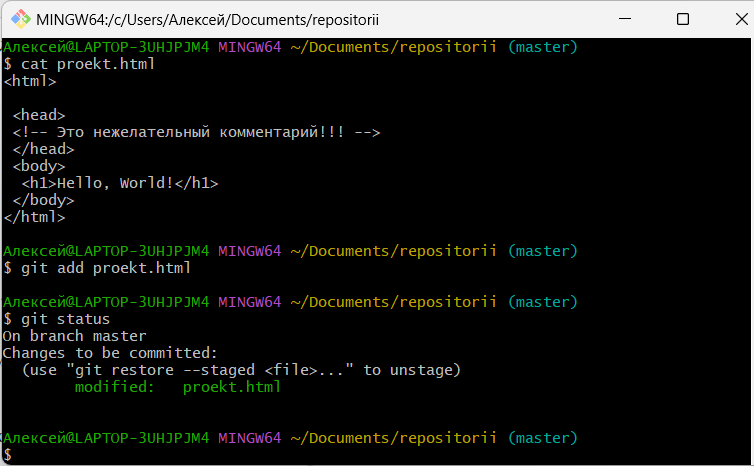


Рисунок 20 – Внесение и индексация изменений

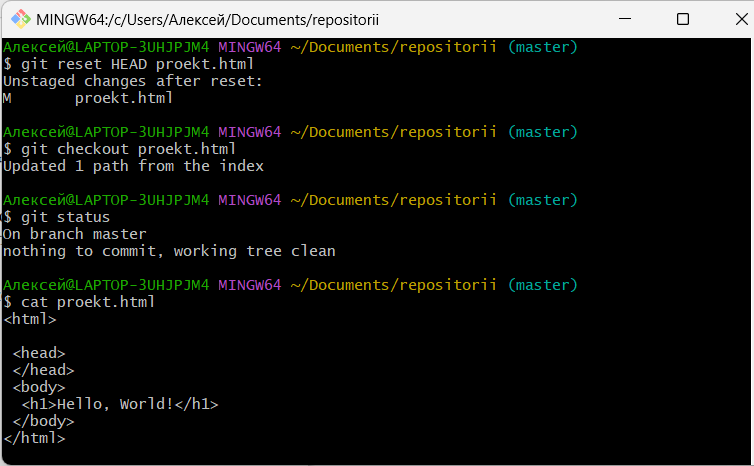


Рисунок 21 – Отмена индексации

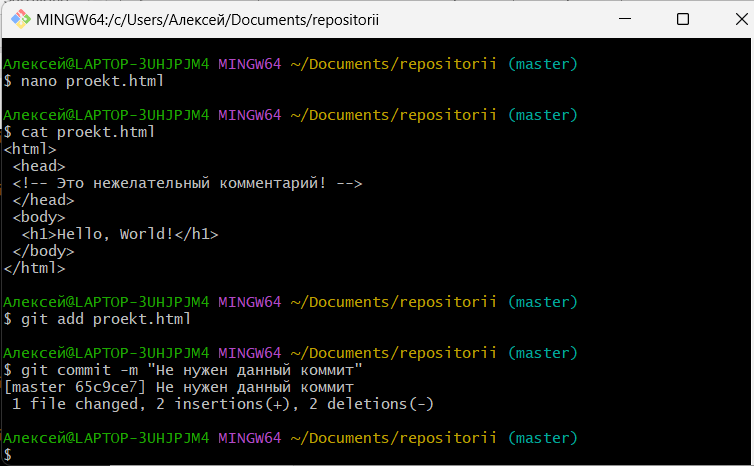


Рисунок 22 – Внесение изменений, индексация, произведение коммита

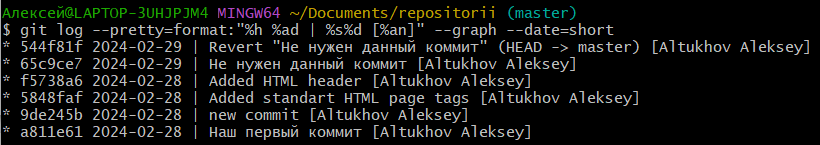


Рисунок 23 – Отмена коммита

# **ЧАСТЬ 2**

1. Создайте аккаунт на GitHub (у кого нет).

Так как в процессе предыдущей учебы мне уже приходилось работать с GitHub, этап создания профиля можно пропустить.

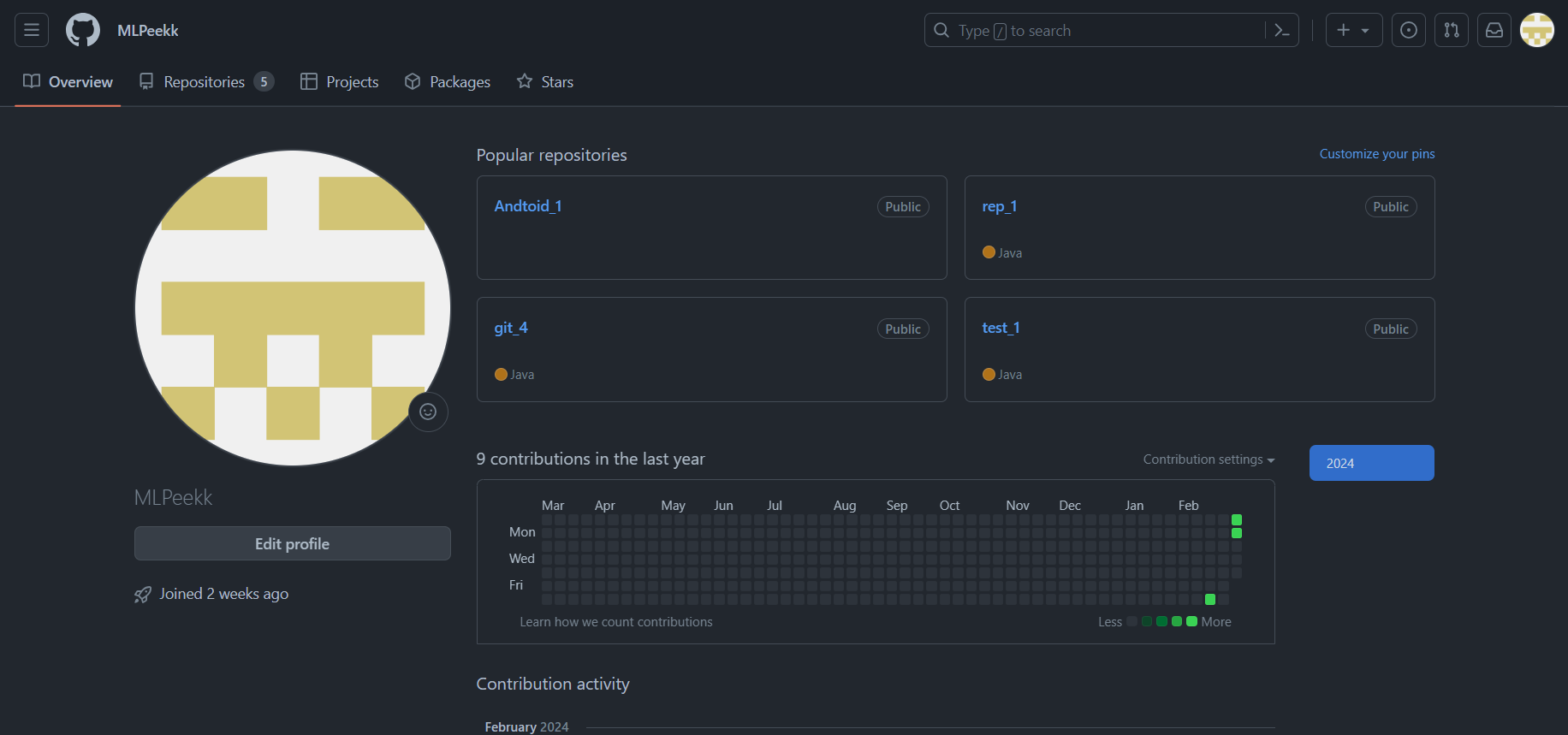


Рисунок 24 – Профиль GitHub

2. Создайте репозиторий на GitHub и на локальной машине, согласно выбранной теме проекта.

3. Создайте несколько файлов на локальной машине при помощи консоли.

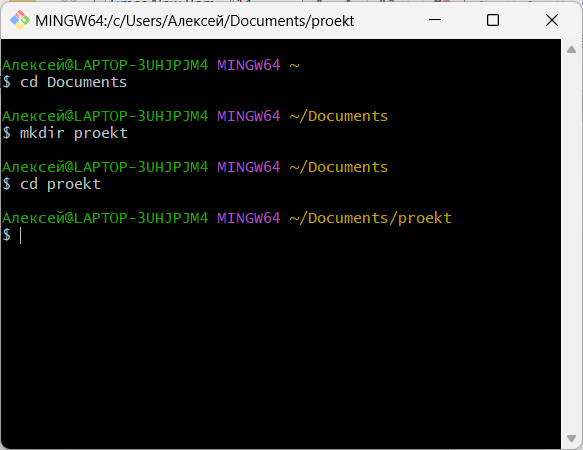


Рисунок 25 – Создание и переход в папку проекта

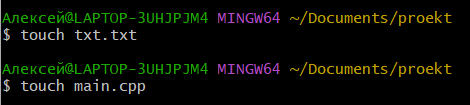


Рисунок 26 – Создание файлов

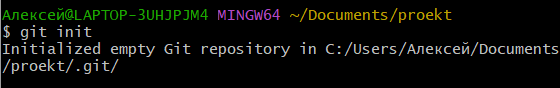


Рисунок 27 – Создание нового локального репозитория

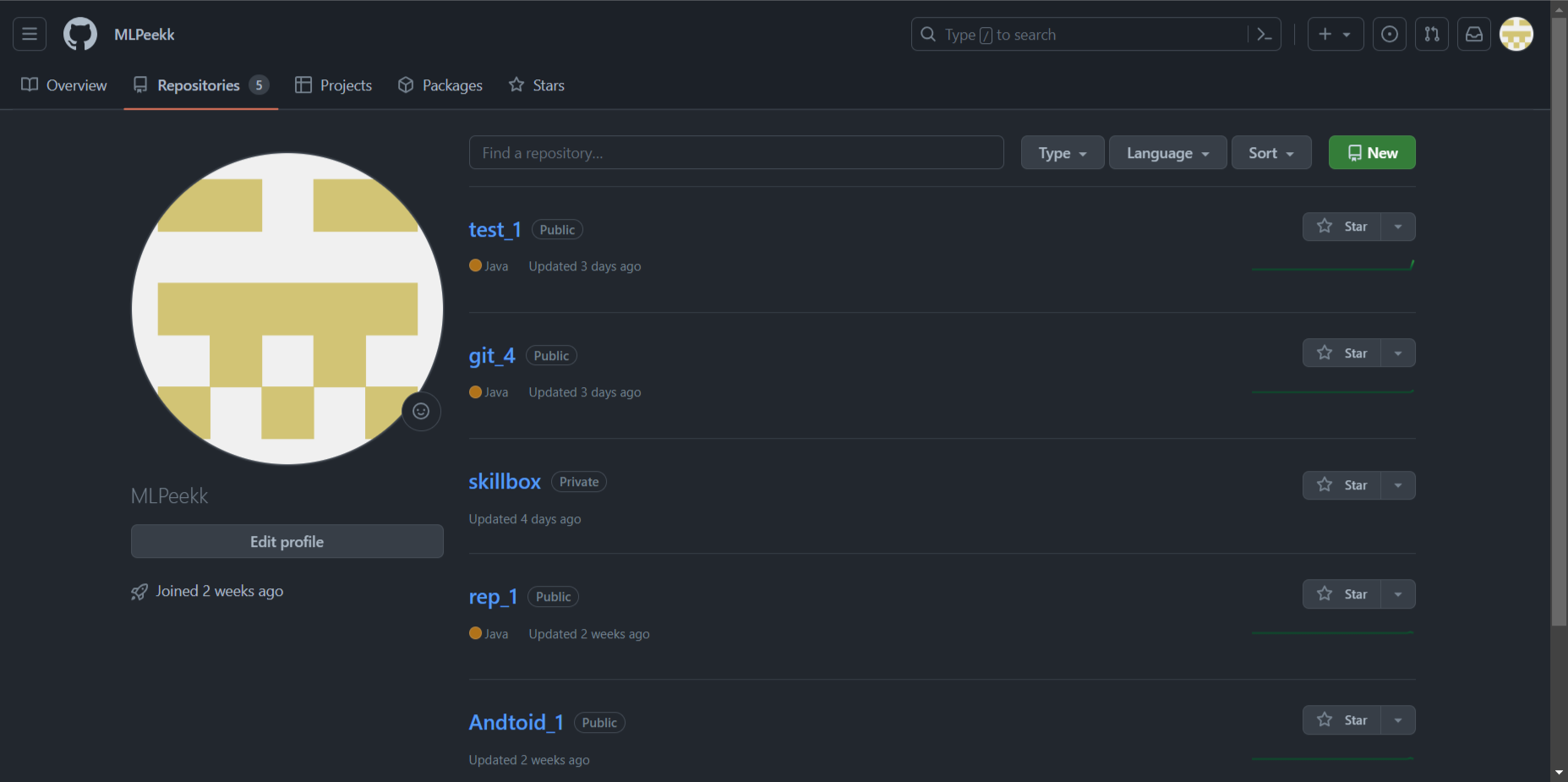


Рисунок 28 – Вкладка с репозиториями

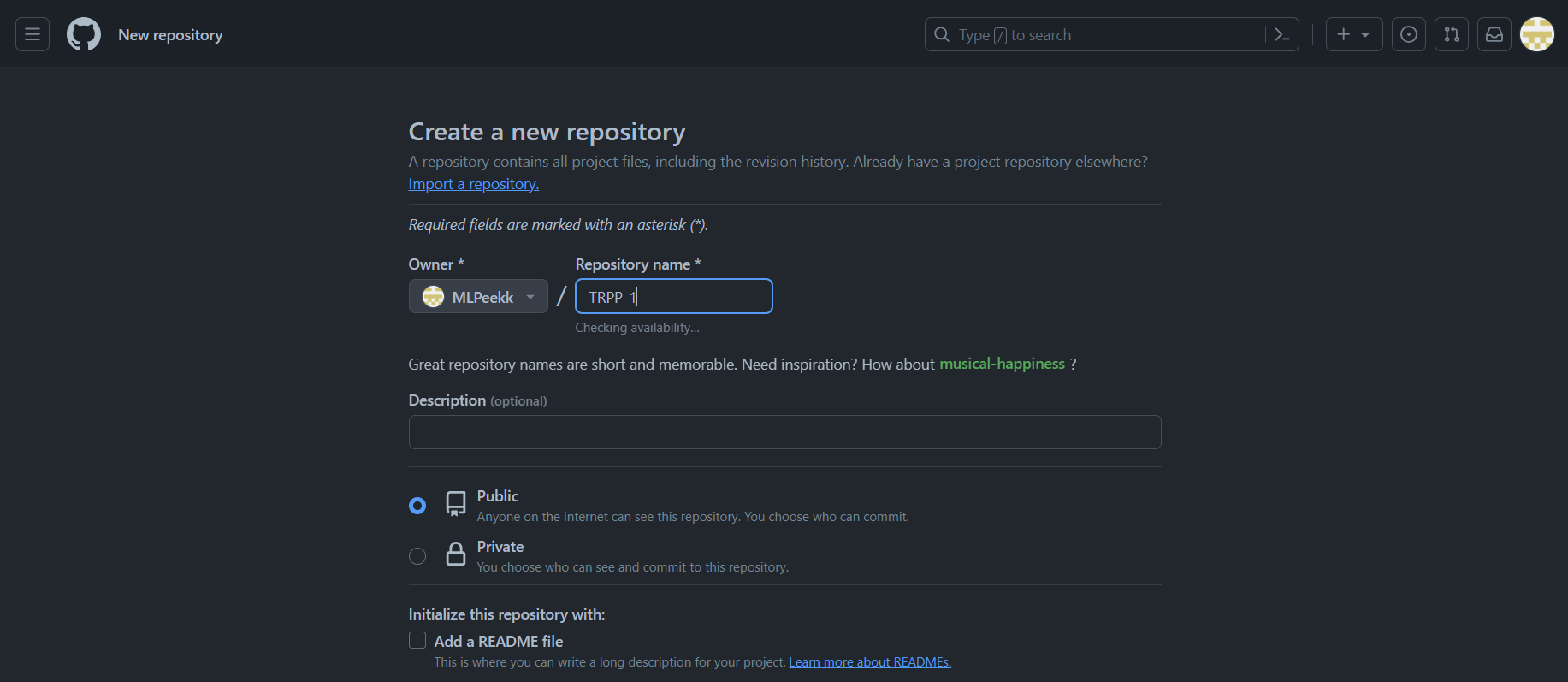


Рисунок 29 – Создание репозитория

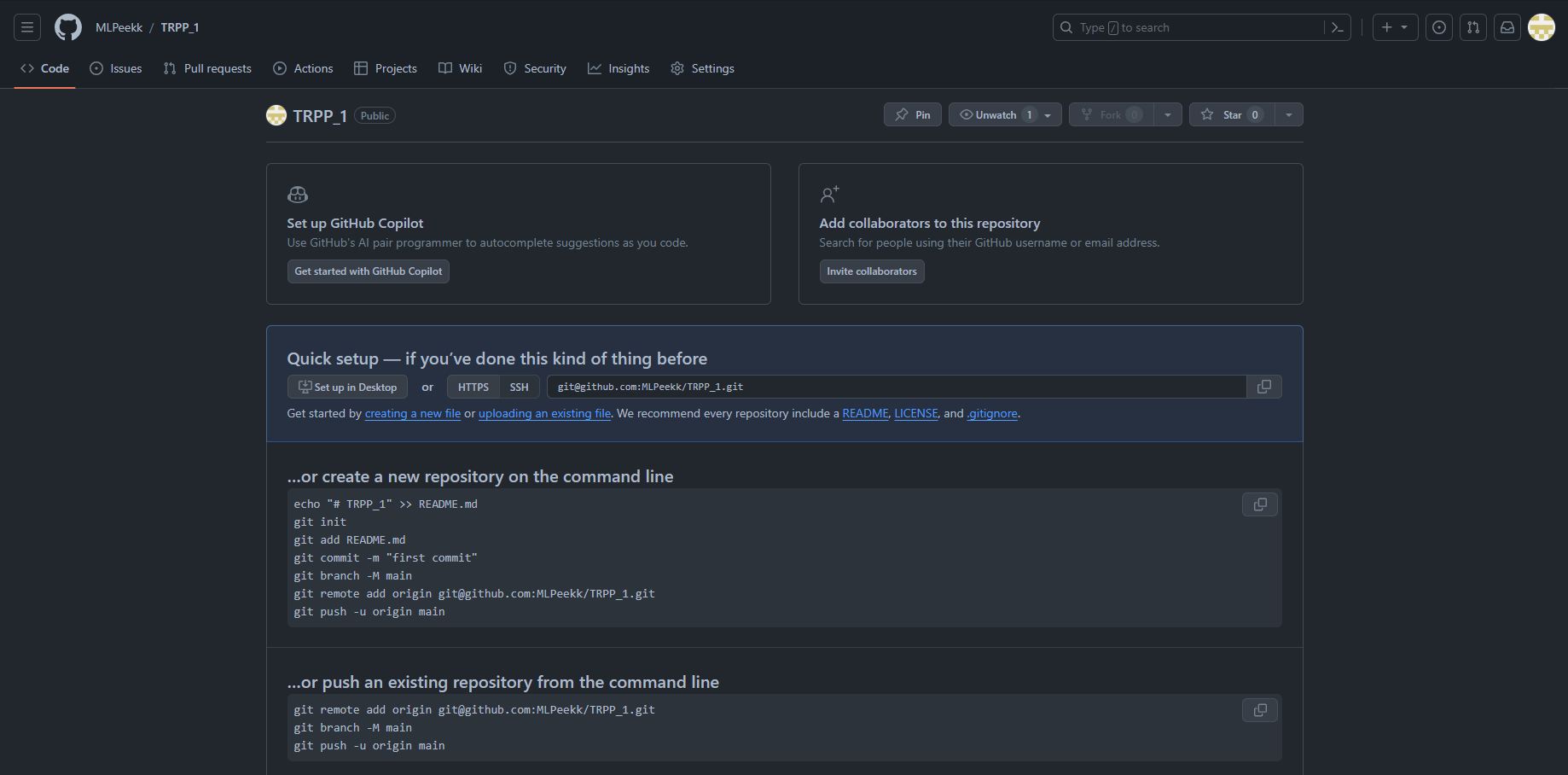


Рисунок 30 – Созданный репозиторий

4. Создайте SSH-ключ для авторизации.

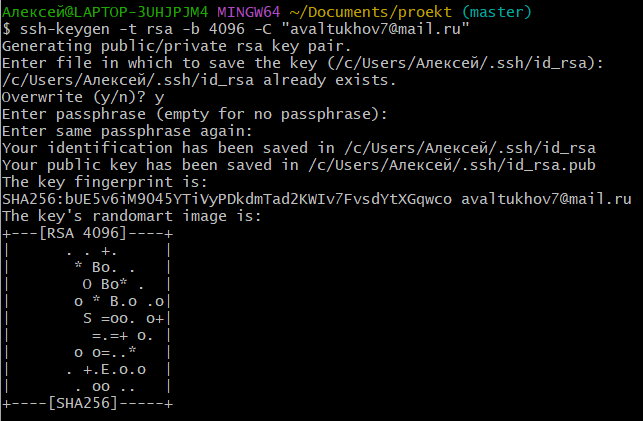


Рисунок 31 – Создание SSH-ключа

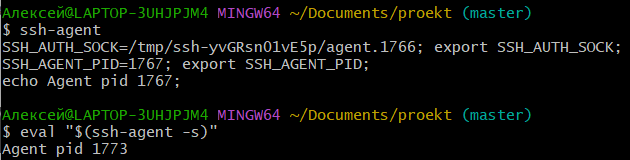


Рисунок 32 – Добавление ключа в SSH-агент и проверка доступности

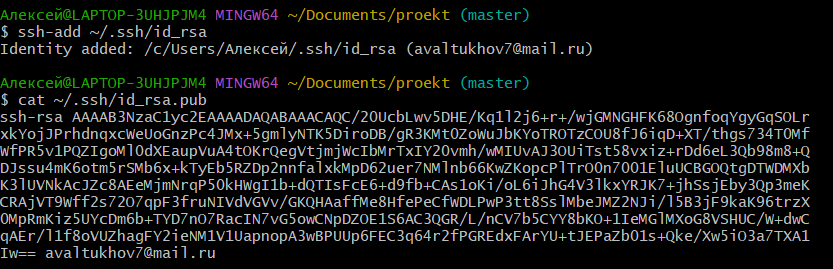


Рисунок 33 – Получение ключа

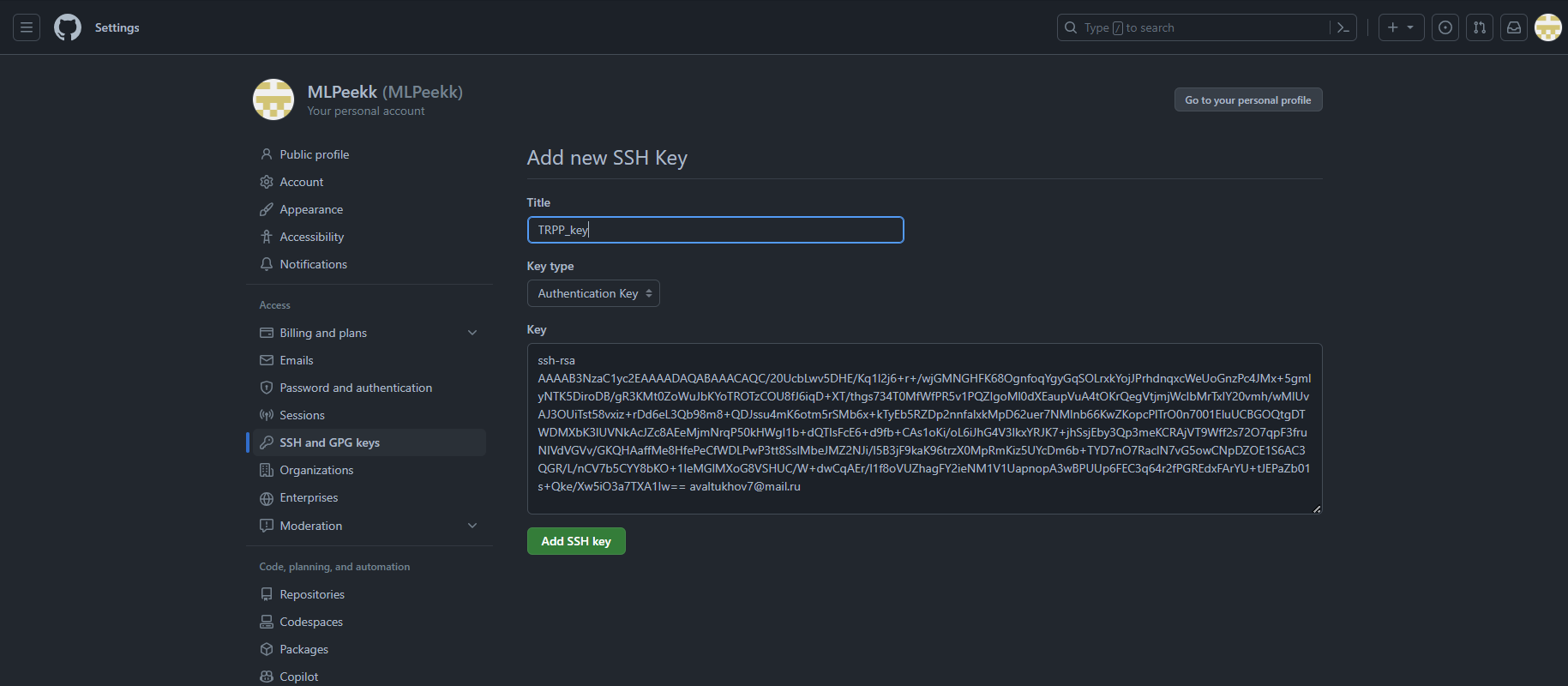


Рисунок 34 – Добавление ключа на сайте GitHub

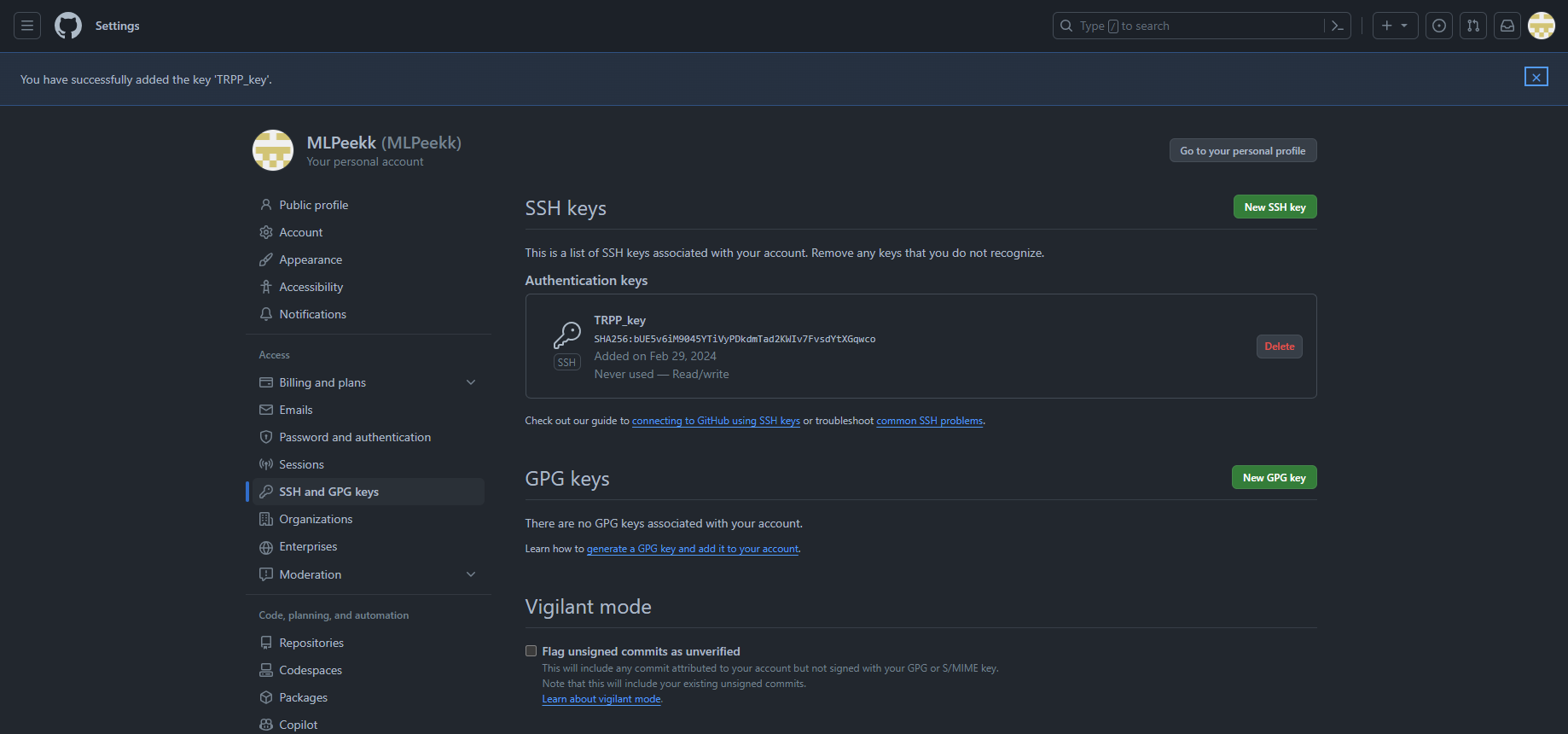


Рисунок 35 – Добавленный ключ

5. Свяжите репозиторий локальной машины с репозиторием на GitHub при помощи консоли.



Рисунок 36 – Связка удаленного и локального репозиториев

6. Создайте новую ветку в репозитории с помощью команды, произведите в ней какие-нибудь изменения, а после слейте с веткой master.

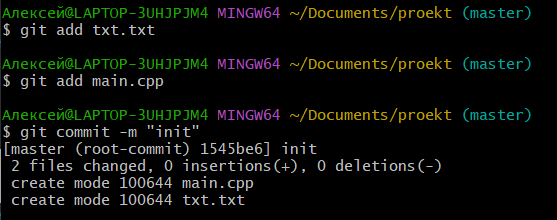


Рисунок 37 – Коммит для инициализации



Рисунок 38 – Создание новой ветки

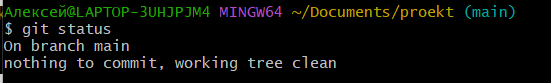


Рисунок 39 – Проверка состояния новой ветки

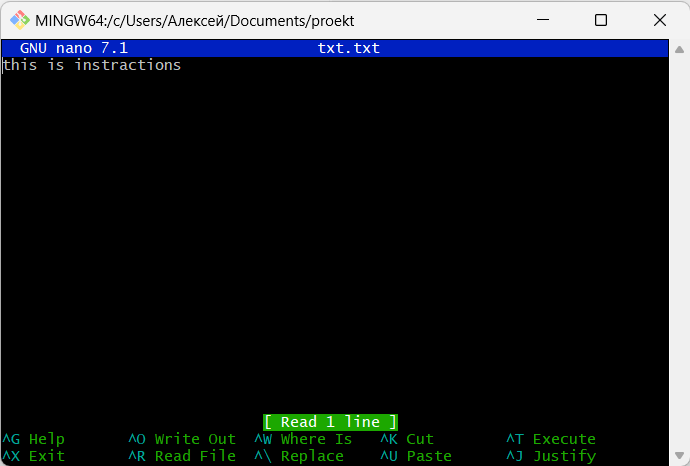


Рисунок 40 – Внесение изменений в файл txt.txt

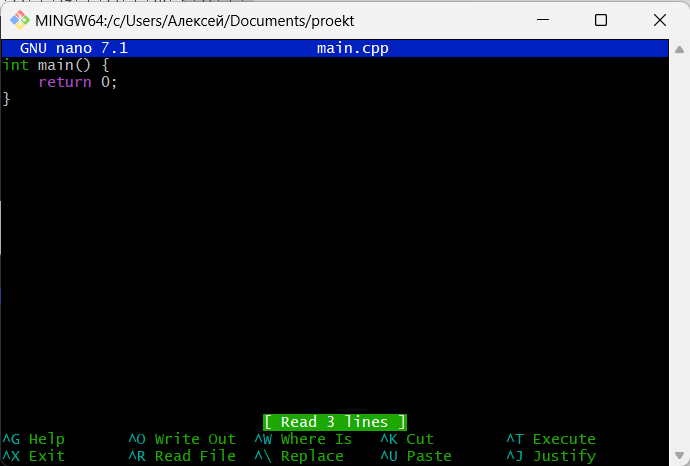


Рисунок 41 – Внесение изменений в файл main.cpp

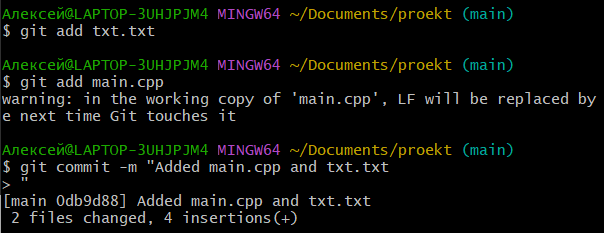


Рисунок 42 – Индексация и коммит изменений

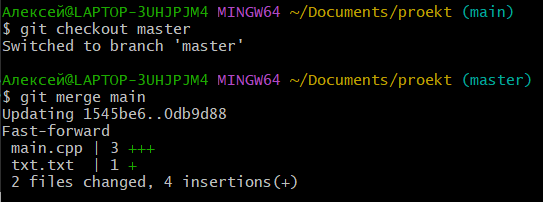


Рисунок 43 – Слияние веток

7. Выполните цепочку действий в репозитории, согласно вариантам.

1) Клонируйте непустой удаленный репозиторий на локальную машину

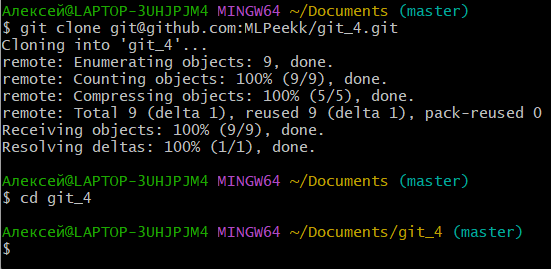


Рисунок 44 – Клонирование репозитория

2) Создайте новую ветку и выведите список всех веток

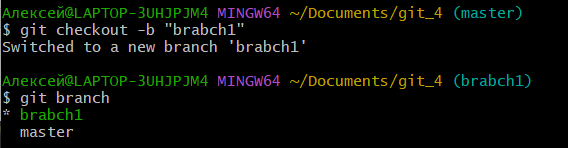


Рисунок 45 – Создание новых веток и вывод списка всех веток

3) Произведите коммит в ветке master

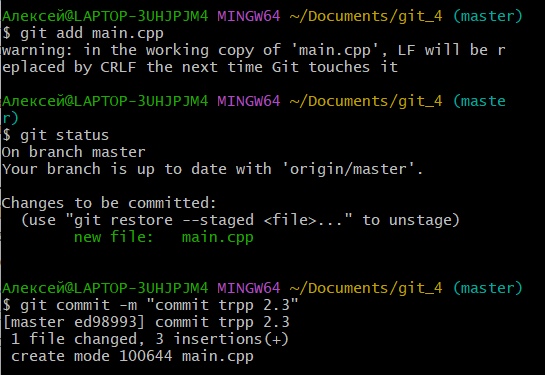


Рисунок 46 – Коммит в ветке master

4) Произведите 3 коммита в новой ветке в разные файлы

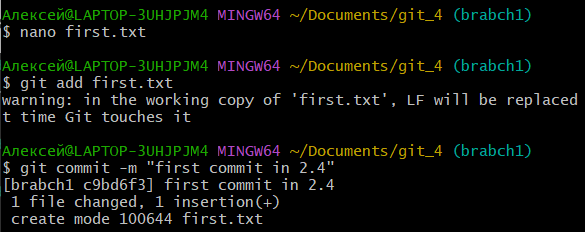


Рисунок 47 – Коммит файла first.txt

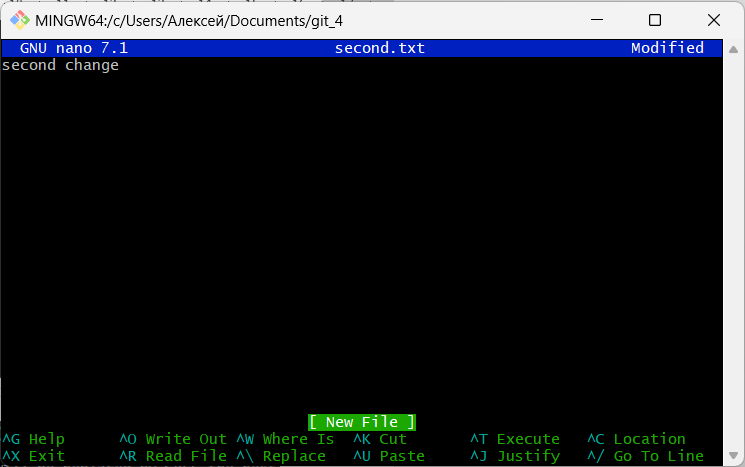


Рисунок 48 – Внесение изменений в файл second.txt

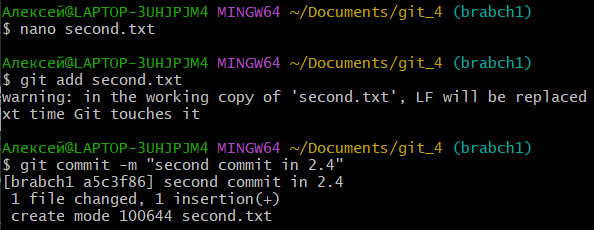


Рисунок 49 – Коммит файла second.txt

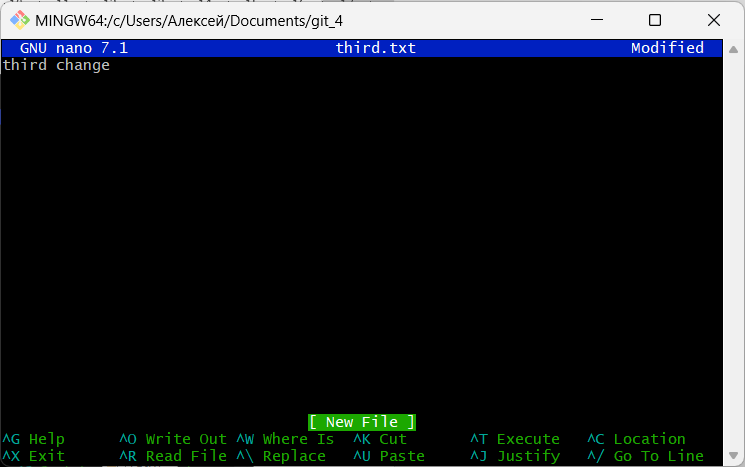


Рисунок 50 – Внесение изменений в файл third.txt

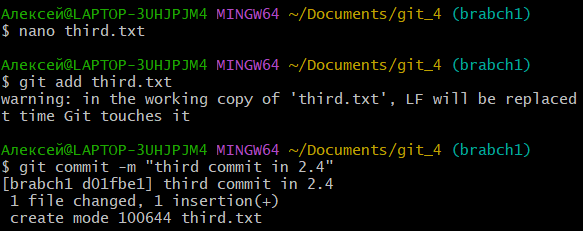


Рисунок 51 – Коммит файла third.txt

5) Выгрузите изменения в удаленный репозиторий

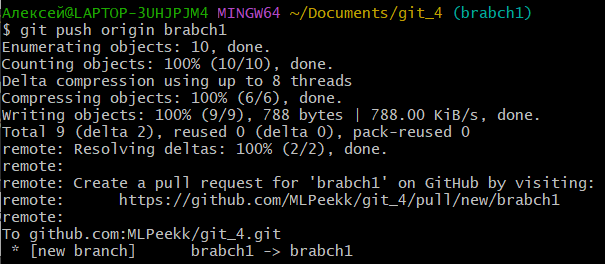


Рисунок 52 – Выгрузка изменений в удаленный репозиторий

6) Откатите ветку обратно на 2 коммита (в том числе в удаленном

репозитории)

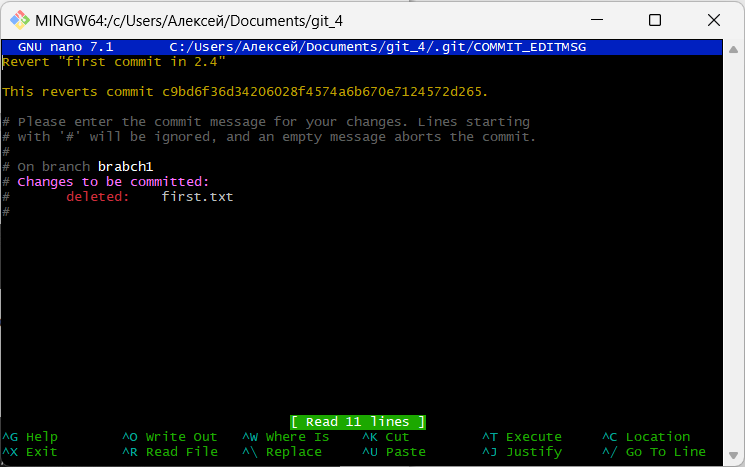


Рисунок 53 – Откат первого коммита

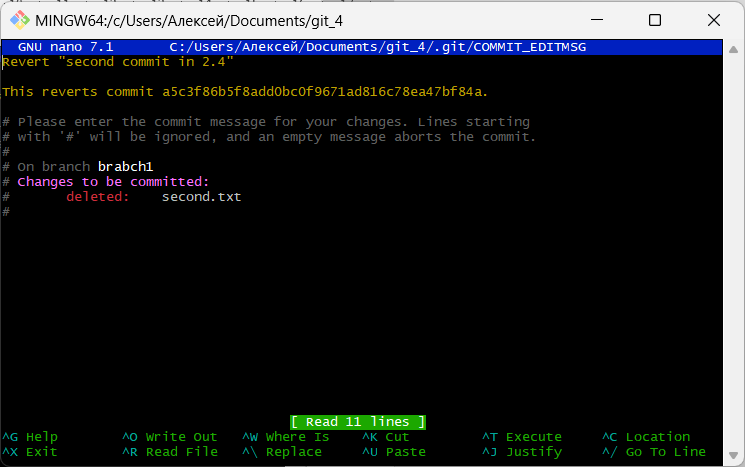


Рисунок 54 – Откат второго коммита

7) Выведите в консоли различия между веткой master и новой веткой

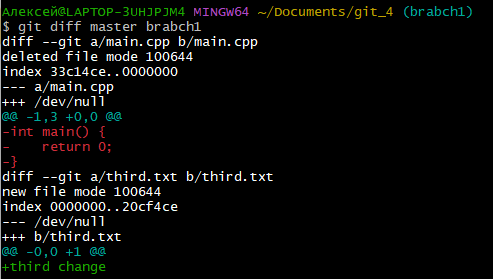


Рисунок 55 – Различия между ветками

8) Перебазируйте новую ветку на master

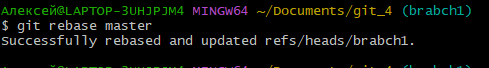


Рисунок 56 – Перебазирование новой ветки на master

# **ЧАСТЬ 3**

1. Сделайте форк репозитория в соответствии с вашим вариантом

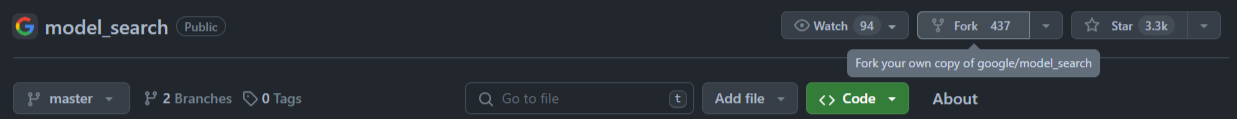


Рисунок 57 –Кнопка “Fork”

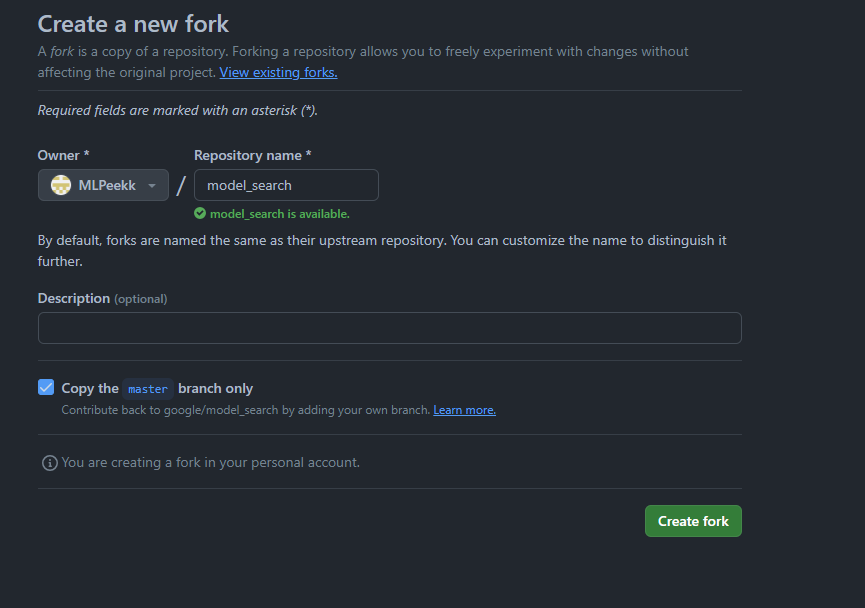


Рисунок 58 – Создание копии репозитория

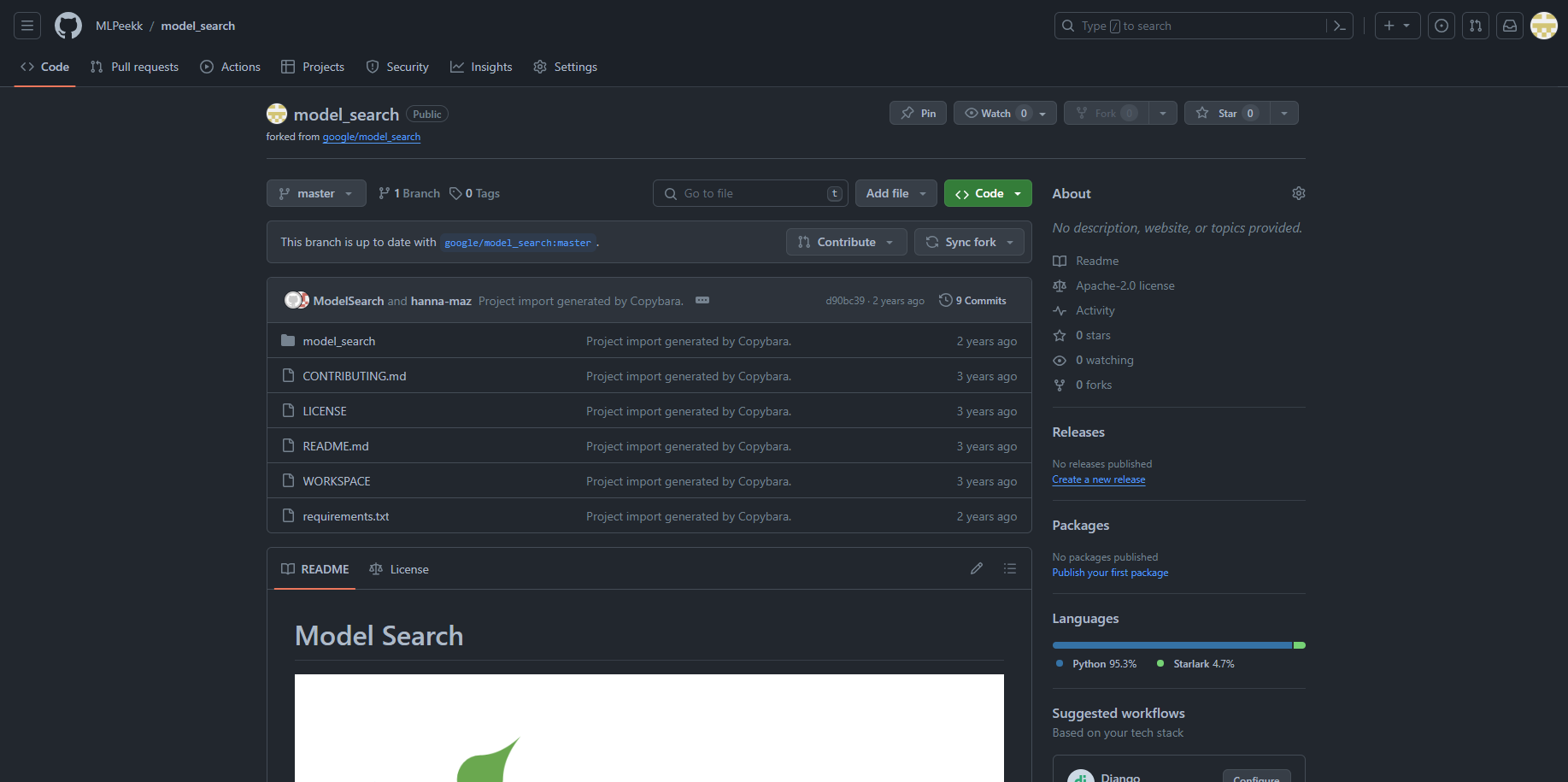


Рисунок 59 – Созданная копия репозитория

2. Склонируйте его на локальную машину

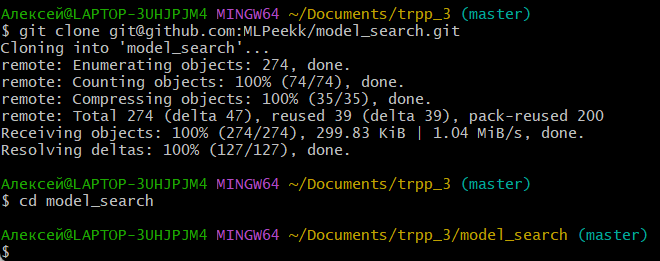


Рисунок 60 – Склонированный удаленный репозиторий

3. Создайте две ветки branch1 и branch2 от последнего коммита в master'е

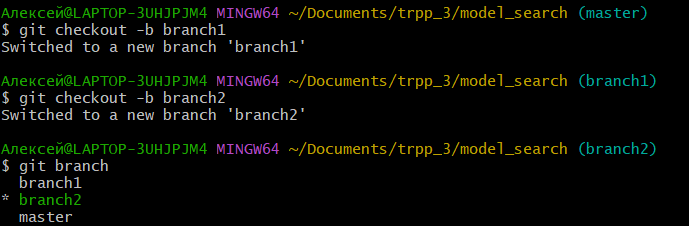


Рисунок 61 – Создание двух новых веток

4. Проведите по 3 коммита в каждую из веток, которые меняют один и тот же кусочек файла

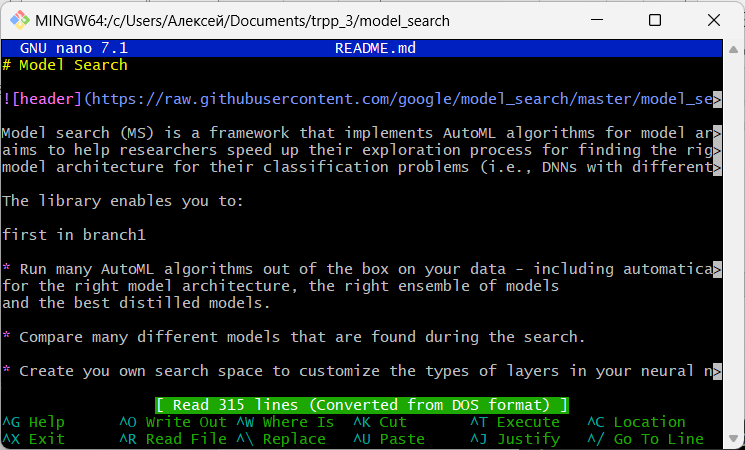


Рисунок 62 – Первое изменение в первой ветке

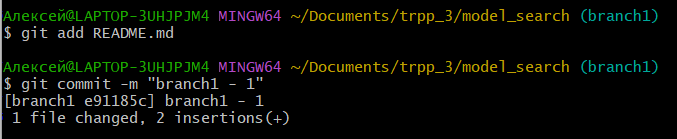


Рисунок 63 – Коммит первого изменения в первую ветку

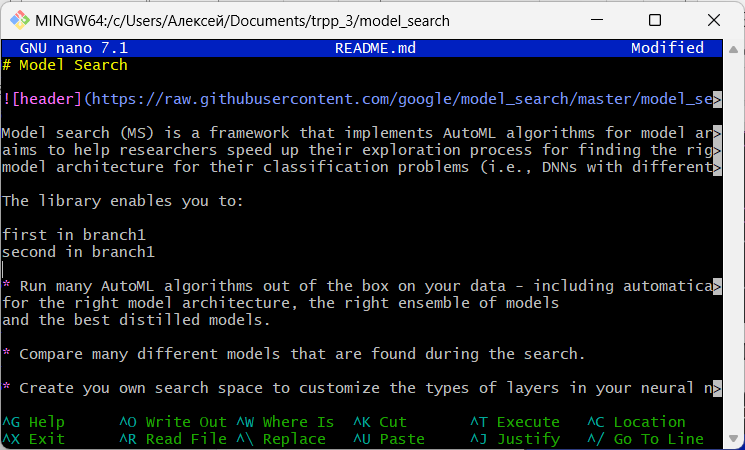


Рисунок 64 – Второе изменение в первой ветке

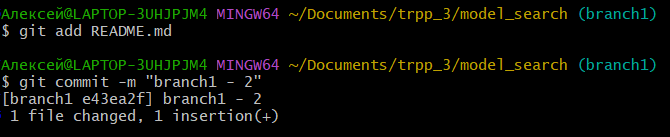


Рисунок 65 – Коммит второго изменения в первую ветку

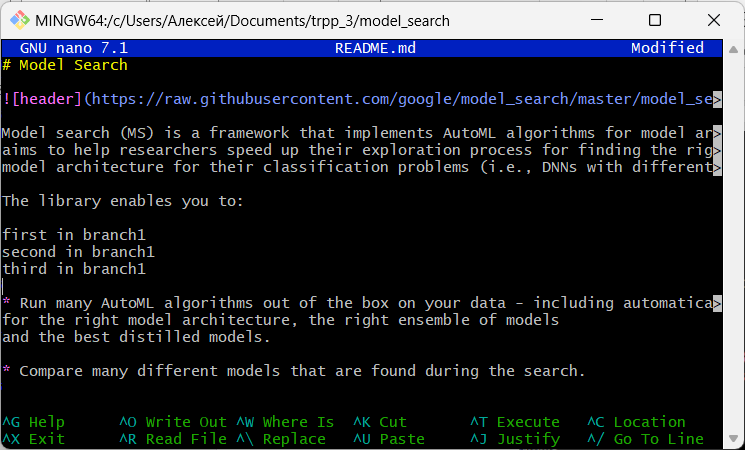


Рисунок 66 – Третье изменение в первой ветке

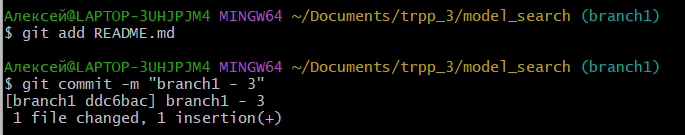


Рисунок 67 – Коммит третьего изменения в первую ветку

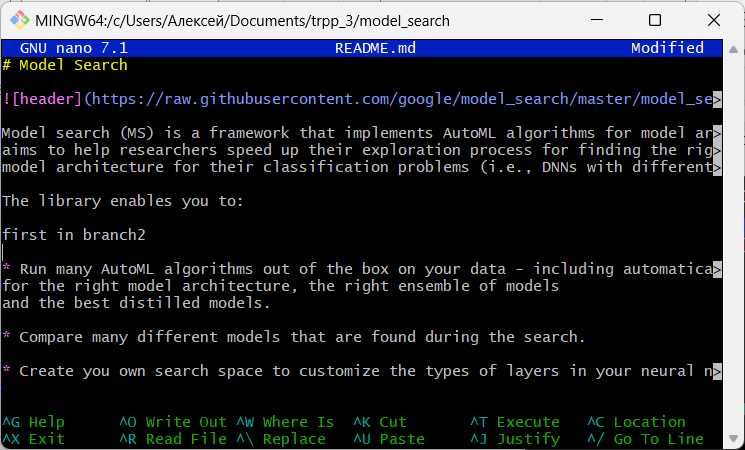


Рисунок 68 – Первое изменение во второй ветке

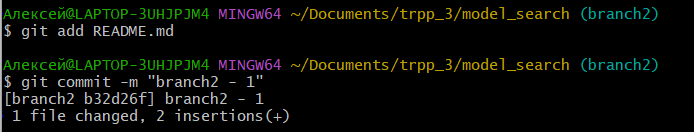


Рисунок 69 – Коммит первого изменения во вторую ветку

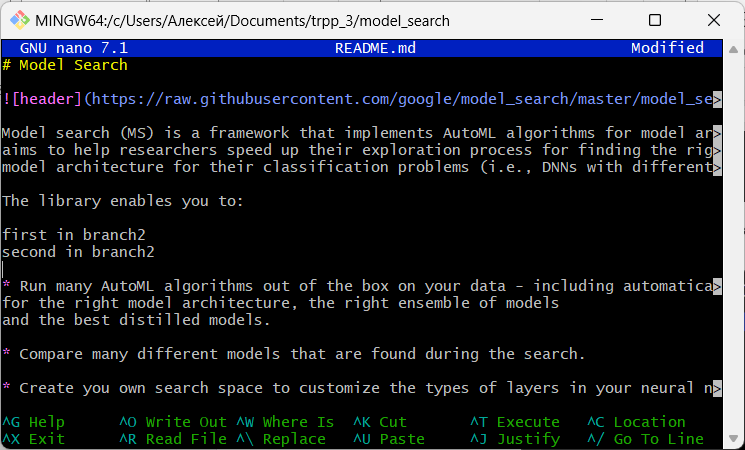


Рисунок 70 – Второе изменение во второй ветке

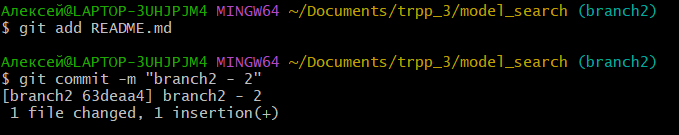


Рисунок 71 – Коммит второго изменения во вторую ветку

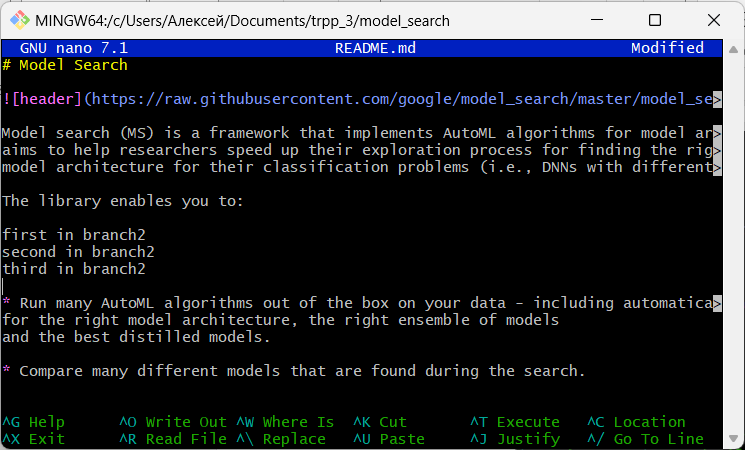


Рисунок 72 – Третье изменение во второй ветке

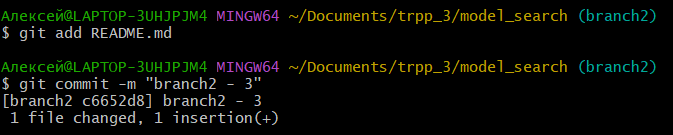


Рисунок 73 – Коммит третьего изменения во вторую ветку

5. Выполните слияние ветки branch2 в ветку branch1, разрешив конфликты при этом

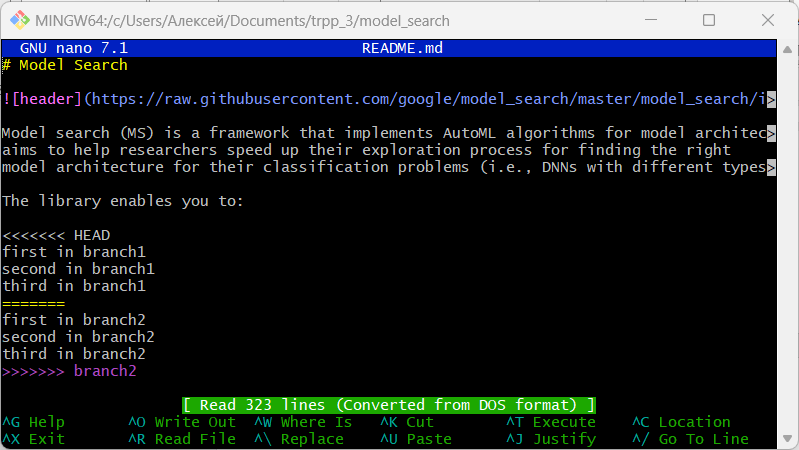


Рисунок 74 – Разрешение конфликта при слиянии

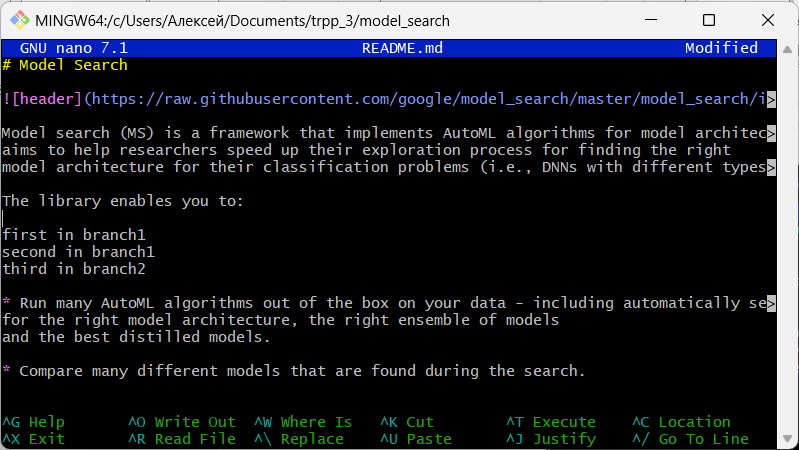


Рисунок 75 – Отредактированный вручную файл

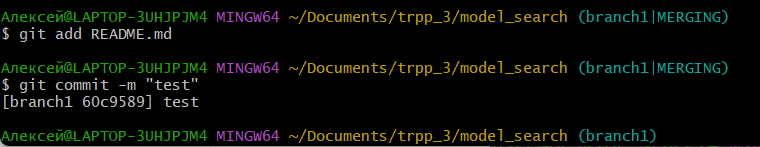


Рисунок 76 – Разрешенный конфликт

6. Выгрузите все изменения во всех ветках в удаленный репозиторий

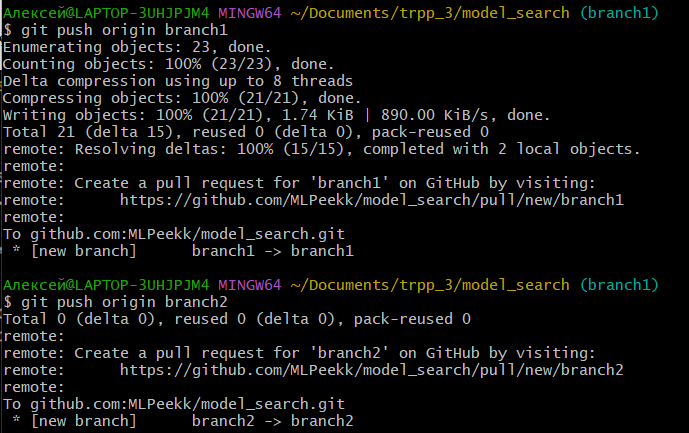


Рисунок 77 – Выгрузка изменений в удаленный репозиторий

7. Проведите еще 3 коммита в ветку branch1

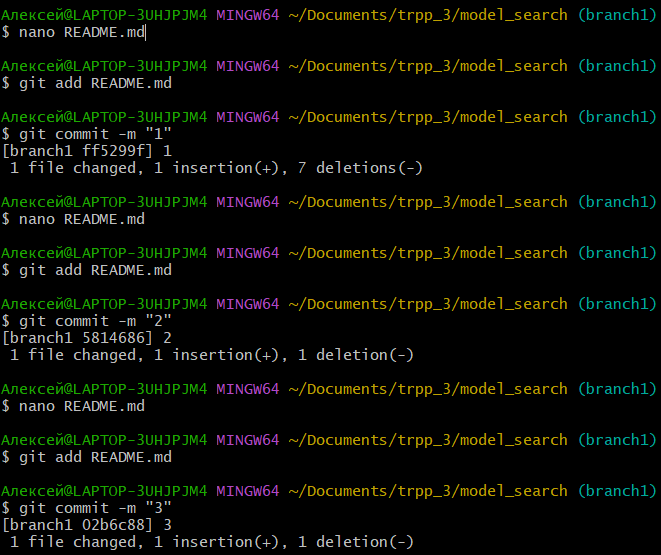


Рисунок 78 – Проведение трех коммитов в первую ветку

8. Склонируйте репозиторий еще раз в другую директорию

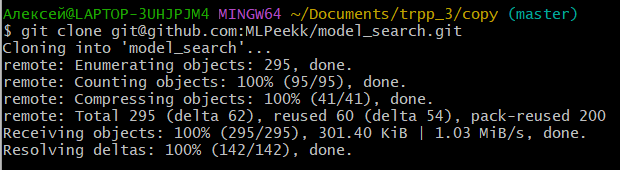


Рисунок 79 – Повторное клонирование репозитория

9. В новом клоне репозитории сделайте 3 коммита в ветку branch1

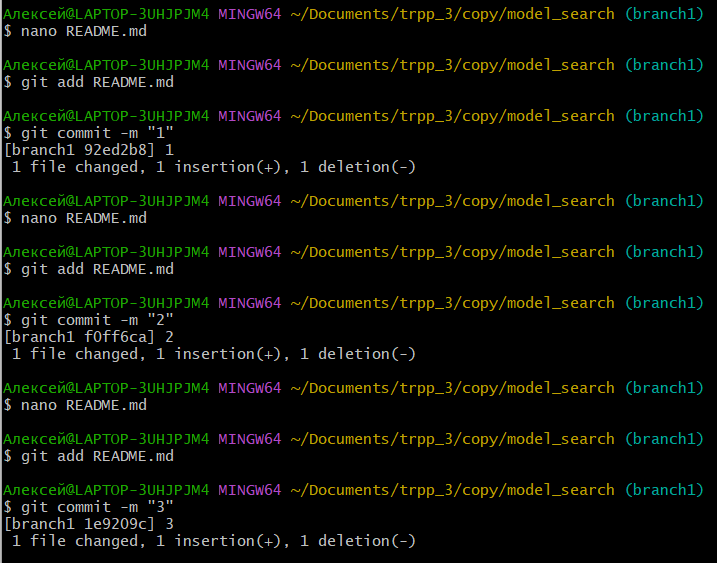


Рисунок 80 – Три коммита в первую ветку

10. Выгрузите все изменения из нового репозитория в удаленный репозиторий

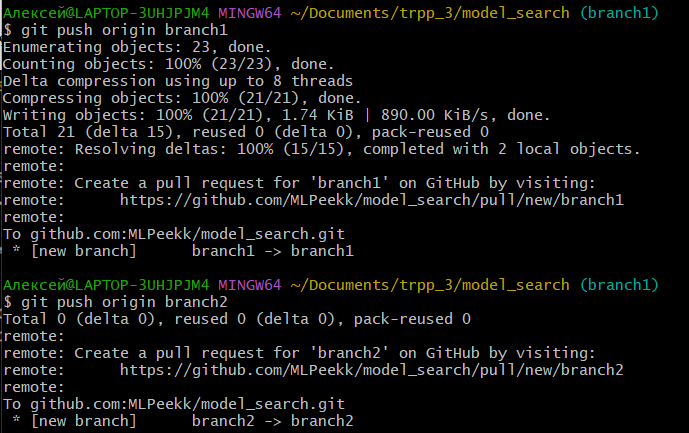


Рисунок 81 – Выгрузка изменений из нового репозитория

11. Вернитесь в старый клон с репозиторием, выгрузите изменения с опцией --force

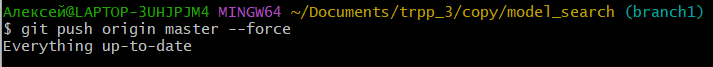


Рисунок 82 – Выгрузка изменений из старого репозитория с опцией force

12. Получите все изменения в новом репозитории.

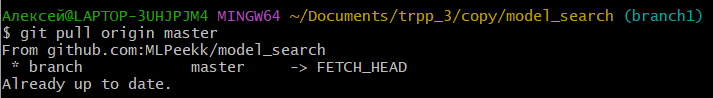


Рисунок 83 – Получение изменений из удаленного репозитория

# **ОТВЕТЫ НА ВОПРОСЫ**

1. Что делает команда git status?

Эта команда показывает текущее состояние рабочего каталога и индекса. Она сообщает, какие файлы изменены, но еще не добавлены в индекс, какие ожидают коммита, и так далее.

2. Что делает команда git add?

Эта команда добавляет изменения в индекс (или стaging area) для последующего коммита. Вы можете использовать `git add <файл>` для добавления конкретного файла или `git add .` для добавления всех измененных файлов.

3. Что делает команда git log?

Эта команда показывает историю коммитов. Она выводит список коммитов, начиная с самого нового, с информацией о хэше коммита, авторе, дате и сообщении коммита.

4. Что делает команда git diff?

Эта команда показывает различия между рабочим каталогом и индексом, или между индексом и последним коммитом. Это полезно для просмотра изменений перед коммитом.

5. Что делает команда git show?

Эта команда показывает информацию о конкретном коммите или объекте, включая изменения в файлах, комментарии и метаданные коммита.

6. Что делает команда git stash?

Эта команда временно сохраняет изменения в рабочем каталоге и индексе, чтобы вы могли переключиться на другую ветку или выполнить другие операции без коммита незавершенных изменений.

7. Что такое репозиторий Git?

Это хранилище для проекта, которое содержит все файлы, историю изменений, ветки и конфигурационные данные. Он обеспечивает систему управления версиями для отслеживания изменений в коде и совместной работы между разработчиками.

8. Что такое коммит?

Это фиксированный набор изменений, который добавляется в историю репозитория. Каждый коммит имеет уникальный хэш, автора, дату и сообщение, описывающее внесенные изменения. Коммиты служат для отслеживания истории изменений и восстановления предыдущих состояний проекта.

# **ВЫВОД**

Были получены навыки по работе с командной строкой и git’ом.