**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«РОССИЙСКАЯ АКАДЕМИЯ НАРОДНОГО ХОЗЯЙСТВА   
И ГОСУДАРСТВЕННОЙ СЛУЖБЫ**

**ПРИ ПРЕЗИДЕНТЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»**

**НИЖЕГОРОДСКИЙ ИНСТИТУТ УПРАВЛЕНИЯ – филиал РАНХиГС**

Факультет: Управления

Кафедра информатики и информационных технологий

Направление подготовки / специальность: 09.03.03 / Прикладная информатика

Направленность (профиль) / специализация: Корпоративные информационные системы управления

**Лабораторная работа № 1.**

|  |  |
| --- | --- |
| По дисциплине: | Программная инженерия |

|  |  |
| --- | --- |
|  | **АВТОР** |
|  | Обучающийся 2 курса группы ИБ-321 |
|  | очной формы обучения |
|  |  |
|  | Шнягина Полина Романовна |
|  |  |
|  | **ПРЕПОДАВАТЕЛЬ** |
|  | Окулич Виктор Иванович |

Нижний Новгород

2026 г.

Оглавление

[Цель работы 3](#_Toc222002898)

[Оборудование и ПО 3](#_Toc222002899)

[Теоретическая часть 3](#_Toc222002900)

[Практическая часть 6](#_Toc222002901)

[Контрольные вопросы 14](#_Toc222002902)

[Вывод 14](#_Toc222002903)

# Цель работы

Освоить базовые операции работы с системой контроля версий Git и облачным сервисом GitHub: регистрацию, создание репозитория, настройку SSH‑ключей, базовые команды Git, редактирование файлов и управление изменениями.

# Оборудование и ПО

* Персональный компьютер с ОС Windows 10
* Установленный Git
* Браузер
* Терминал/командная строка

# Теоретическая часть

**Git** — распределённая система контроля версий (VCS), предназначенная для:

* отслеживания изменений в файлах (прежде всего — в исходном коде);
* совместной работы над проектами;
* хранения истории правок и возможности отката к любым предыдущим состояниям.

Ключевые особенности:

* каждый участник имеет полную копию репозитория (включая всю историю);
* работа возможна офлайн (коммиты создаются локально, синхронизация — позже);
* высокая скорость операций за счёт локального хранения данных;
* устойчивость к сбоям (нет единого «центрального» сервера).

Типы систем контроля версий:

1. Локальные системы контроля версий (ЛСКВ)

* Принцип: все данные хранятся на одном компьютере разработчика.
* Как работает: файлы копируются в отдельную директорию; изменения сохраняются в виде патчей (разниц между версиями) с отметками времени.

1. Централизованные системы контроля версий (ЦСКВ)

* Принцип: один центральный сервер хранит все версии файлов; клиенты получают и отправляют изменения на этот сервер.
* Как работает: разработчики «проверяют» (checkout) файлы из сервера, вносят правки и «возвращают» (commit) их обратно.

1. Распределённые системы контроля версий (РСКВ)

* Принцип: каждый участник имеет полную копию репозитория (включая всю историю изменений).
* Как работает: клиент при клонировании получает весь репозиторий (все коммиты, ветки, теги); изменения фиксируются локально, затем синхронизируются с удалёнными репозиториями; можно работать офлайн; несколько удалённых репозиториев.

Команды Git

1. Инициализация и настройка

* git init — создаёт новый локальный репозиторий в текущей директории.
* git clone <URL> — копирует удалённый репозиторий на ваш компьютер.
* git config --global user.name "Имя" и git config --global user.email "email@example.com" — задают имя и почту автора коммитов (глобально).
* git remote add origin <URL> — связывает локальный репозиторий с удалённым (например, на GitHub).

2. Работа с изменениями

* git status — показывает состояние рабочей директории (какие файлы изменены, добавлены или не отслеживаются).
* git add <файл> — добавляет файл в индекс (подготавливает к коммиту).
* git add . — добавляет все изменённые и новые файлы из текущей директории.
* git commit -m "Сообщение" — фиксирует изменения в локальном репозитории с комментарием.
* git commit --amend -m "Новое сообщение" — изменяет сообщение последнего коммита (или добавляет в него новые изменения).

3. Просмотр истории и различий

* git log — выводит список коммитов с хешами, авторами и сообщениями.
* git log --oneline — краткий формат истории (хеш + сообщение).
* git diff — показывает различия между рабочей директорией и индексом.
* git diff --staged — показывает различия между индексом и последним коммитом.

4. Ветки

* git branch — выводит список локальных веток.
* git branch <имя> — создаёт новую ветку.
* git checkout <имя> или git switch <имя> — переключается на указанную ветку.
* git checkout -b <имя> или git switch -c <имя> — создаёт новую ветку и переключается на неё.
* git merge <ветка> — сливает указанную ветку в текущую.
* git branch -m <старое> <новое> — переименовывает ветку.
* git branch -d <имя> — удаляет локальную ветку (если она слита).
* git push --delete origin <имя> — удаляет ветку на удалённом репозитории.

5. Синхронизация с удалённым репозиторием

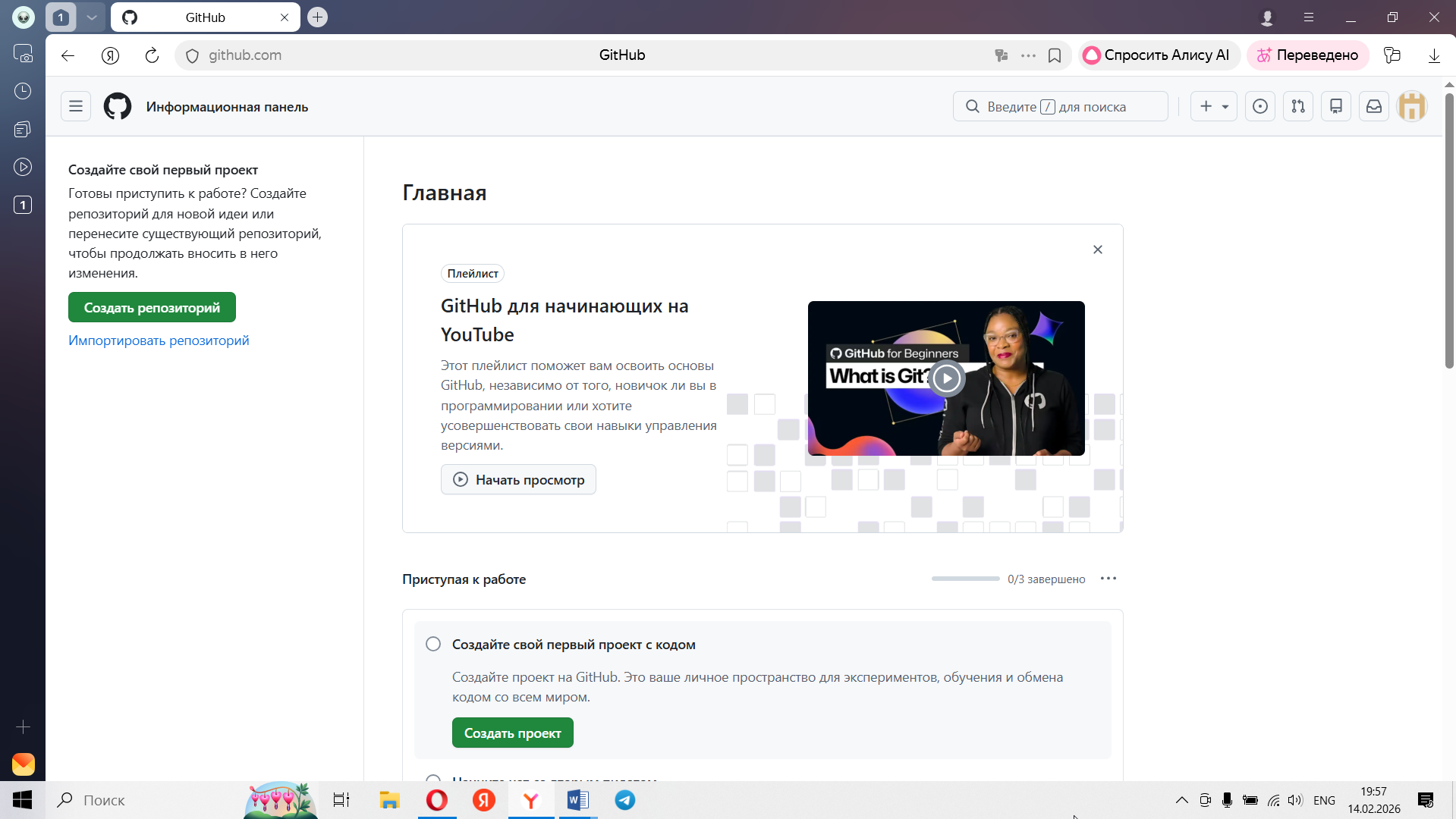
* git fetch — загружает изменения из удалённого репозитория, но не сливает их.
* git pull — загружает изменения и автоматически сливает их с текущей веткой (комбинация fetch + merge).
* git push — отправляет локальные коммиты в удалённый репозиторий.
* git push -u origin <ветка> — отправляет новую ветку и связывает её с удалённой (флаг -u устанавливает отслеживание).

6. Отмена и откат

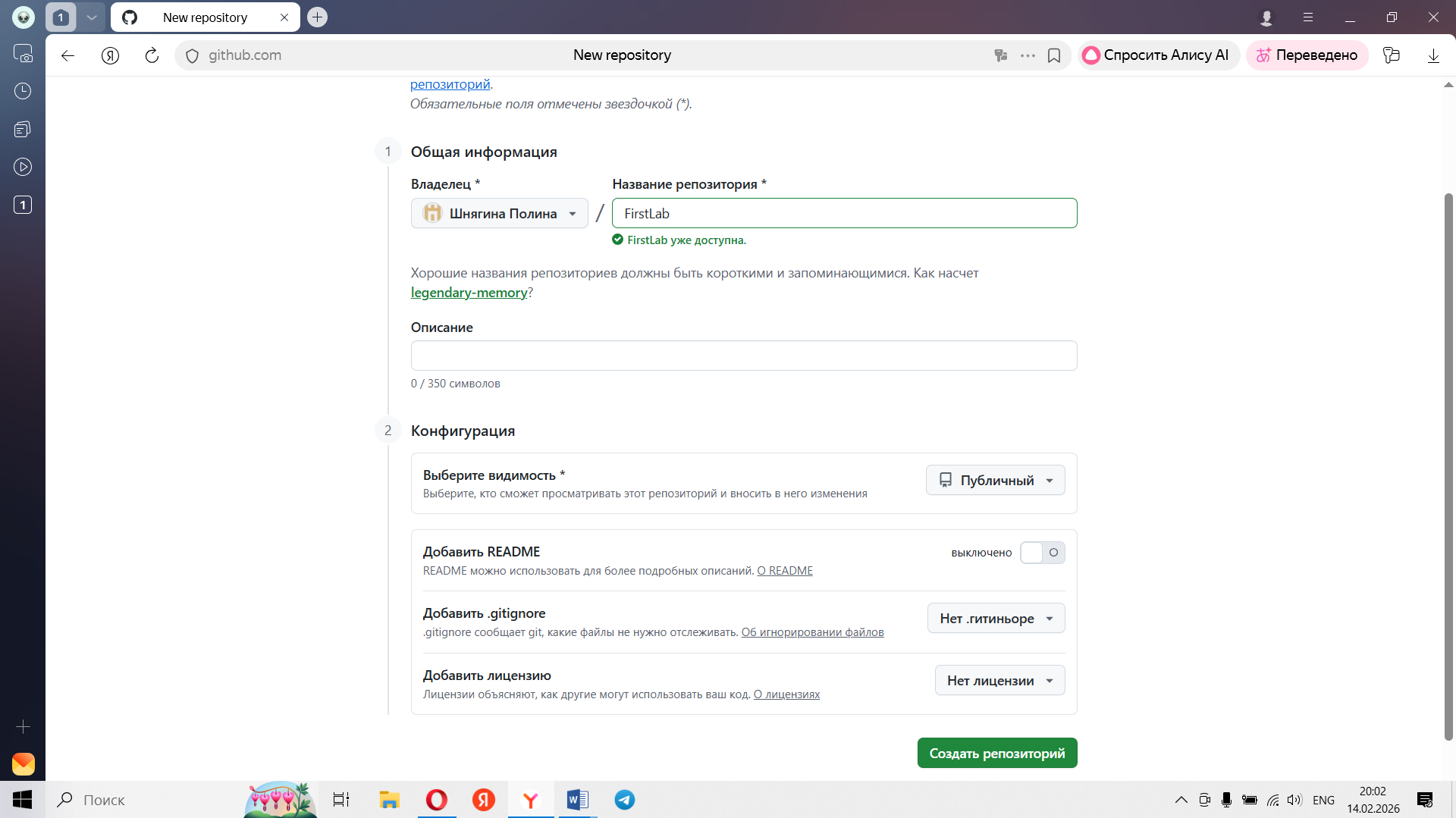
* git restore <файл> — отменяет изменения в файле (возвращает к последнему коммиту).
* git reset HEAD <файл> — убирает файл из индекса, но оставляет изменения в рабочей директории.
* git reset --hard — полностью откатывает все изменения (индекс и рабочую директорию) до указанного коммита.
* git revert <коммит> — создаёт новый коммит, отменяющий изменения указанного коммита (без удаления истории).
* git clean -f — удаляет неотслеживаемые файлы из рабочей директории.

# Практическая часть

1. **Регистрация на GitHub**



1. **Создание репозитория**

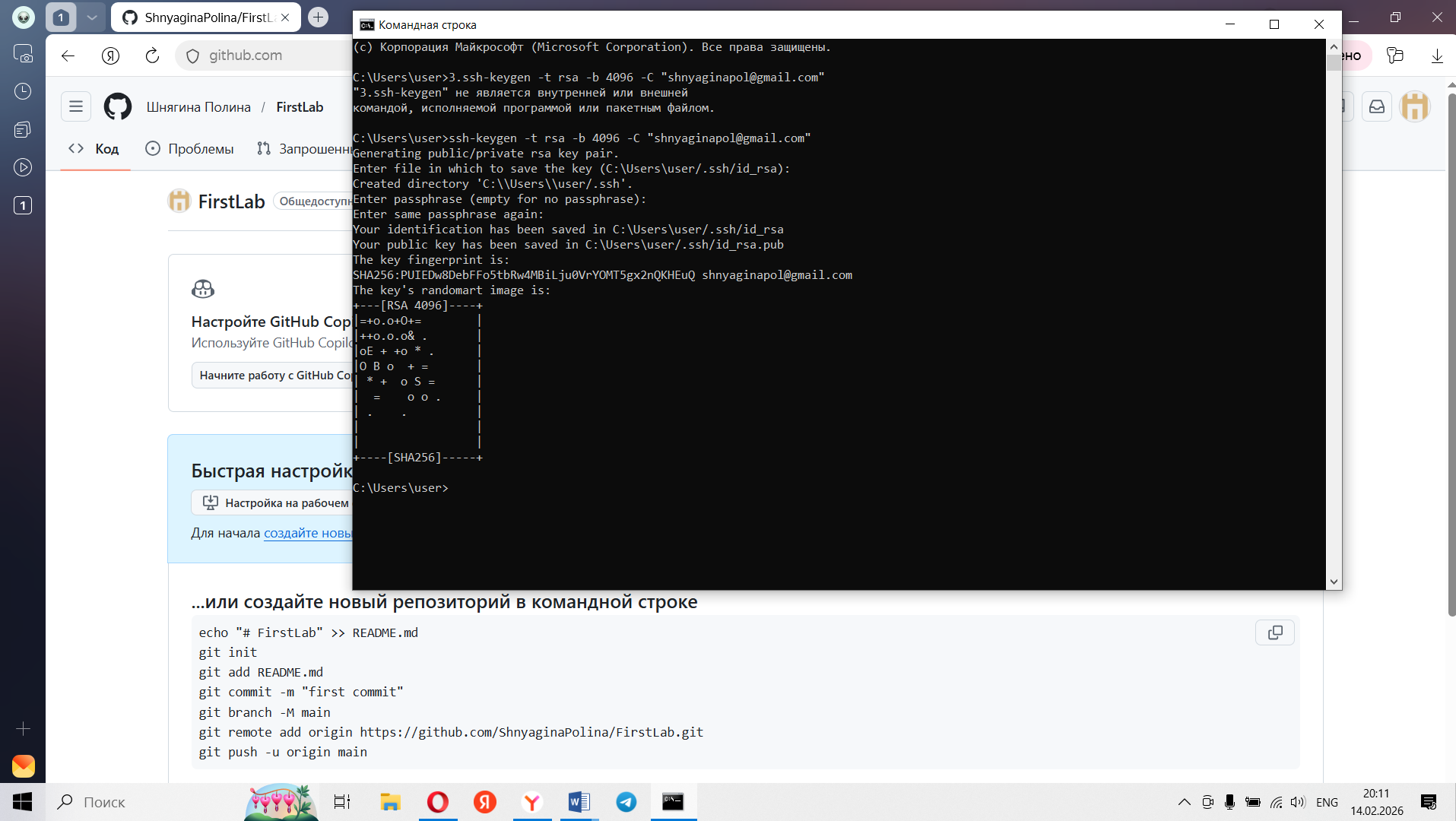




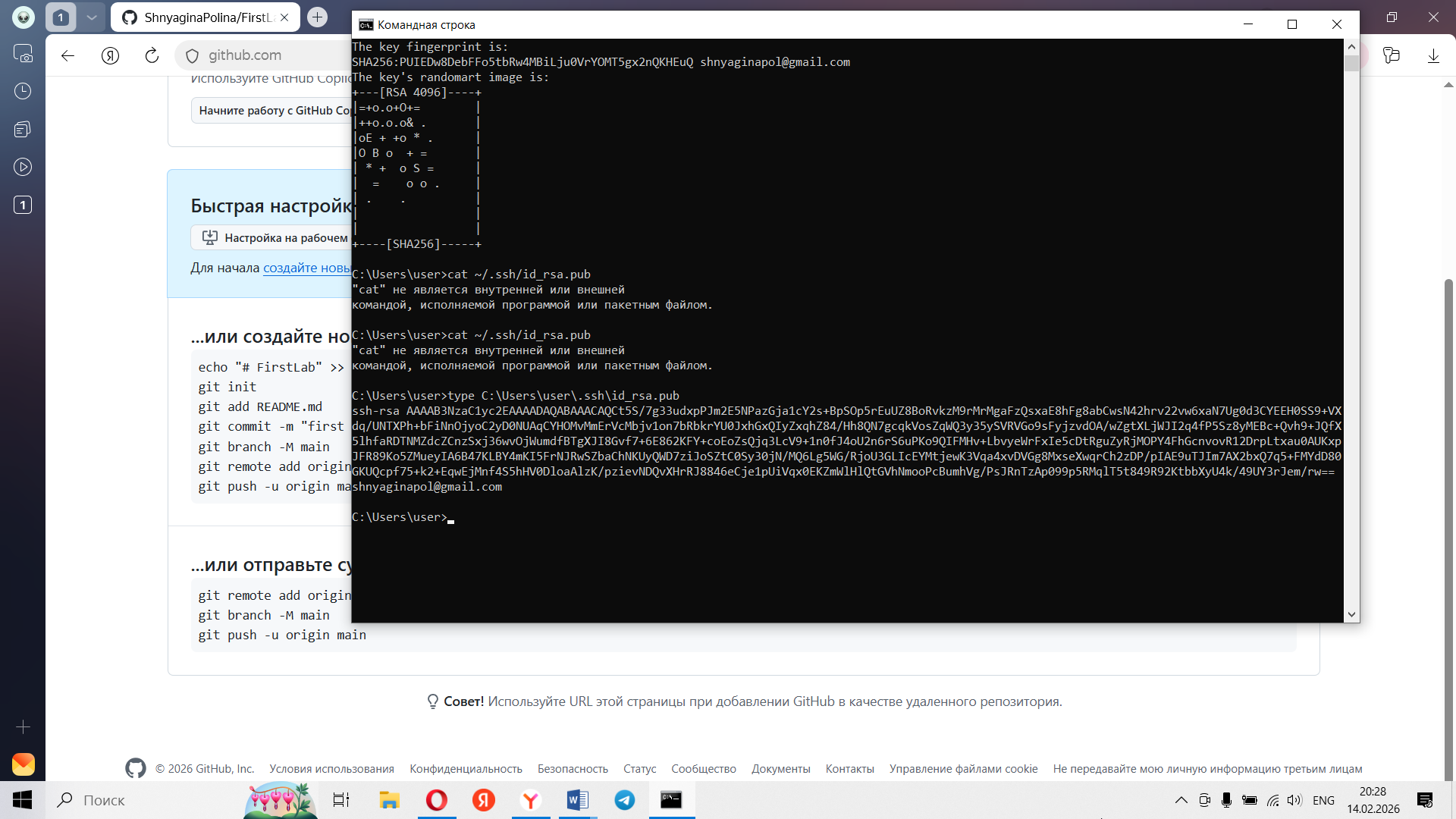
1. **Настройка SSH‑ключа (для безопасной аутентификации)**

**SSH‑ключ** (Secure Shell Key) — это пара криптографических ключей для безопасной аутентификации при подключении к удалённым системам по протоколу SSH.

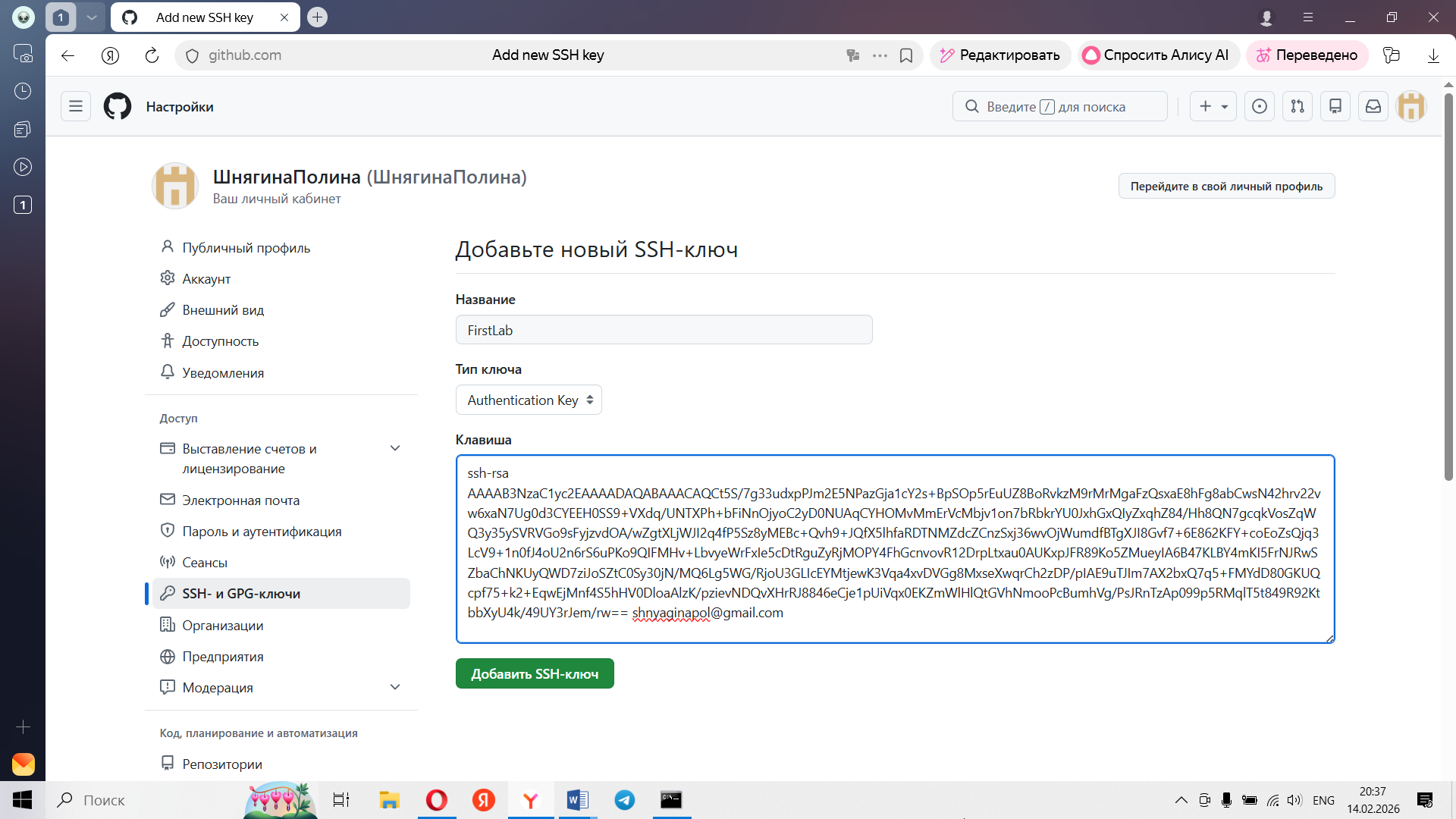
Сгенерировала SSH‑ключ с помощью команды ssh-keygen -t rsa -b 4096 -C shnyaginapol@gmail.com.

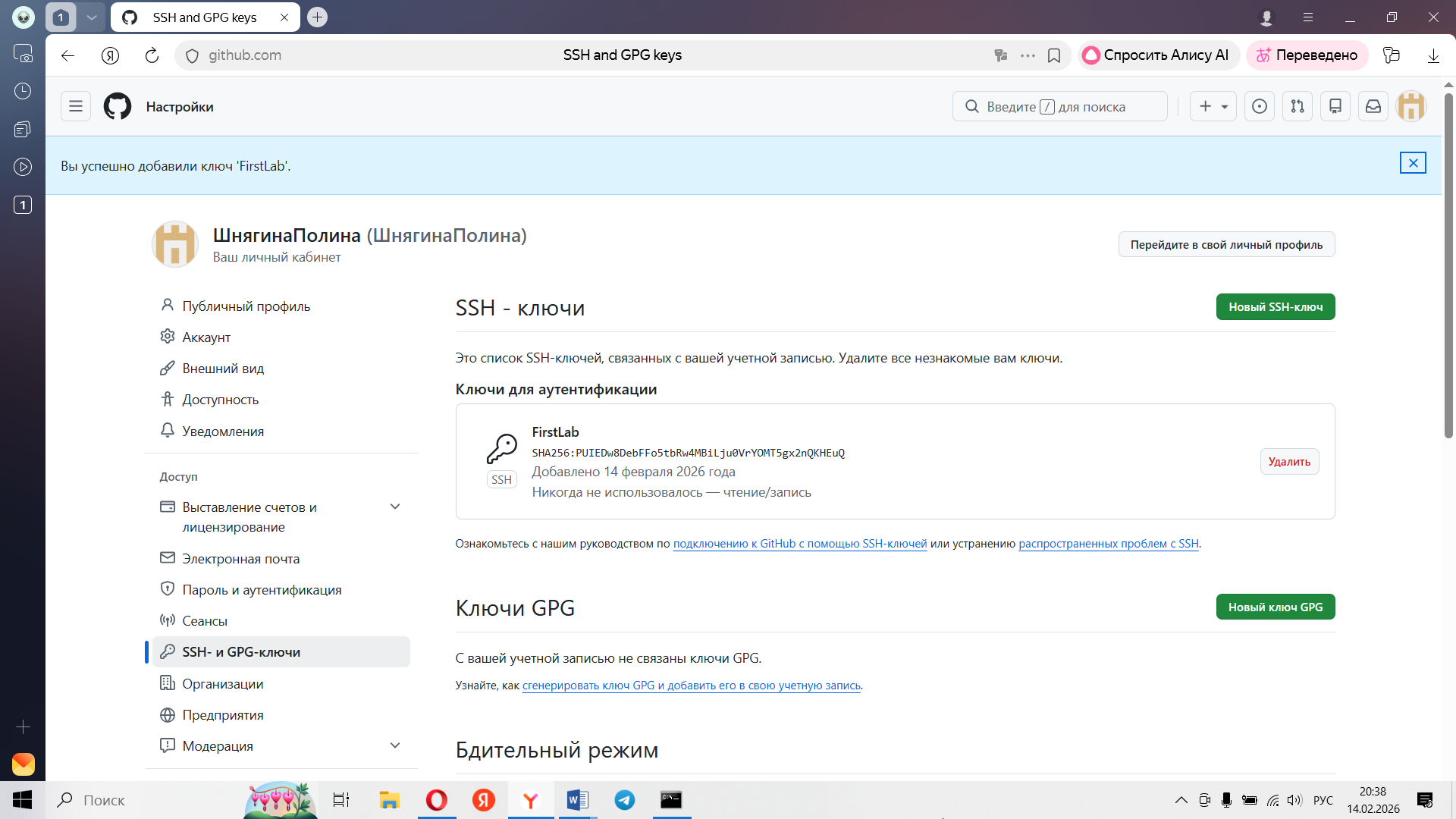


Просмотрела публичный ключ

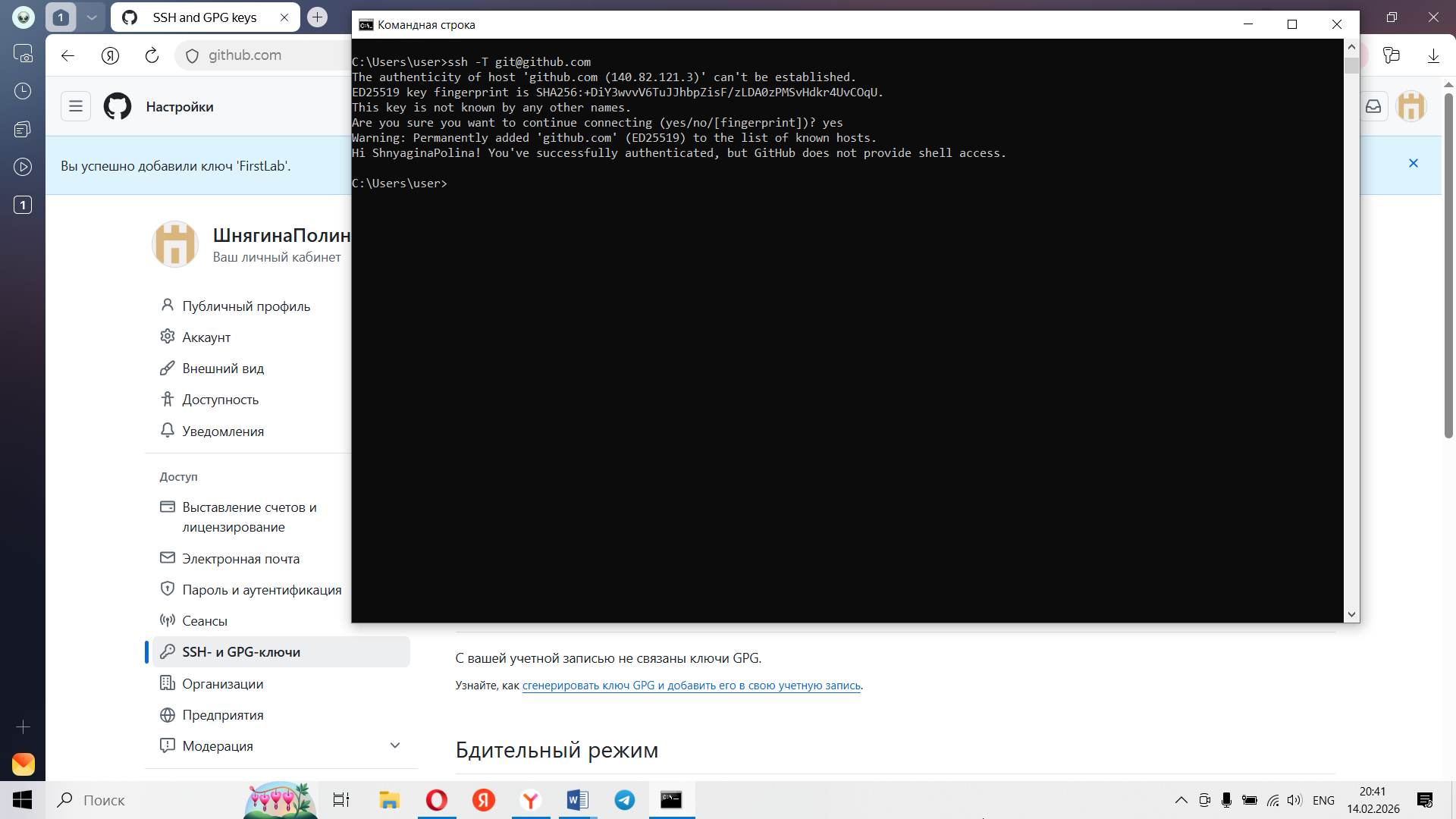


Добавила ключ в настройках



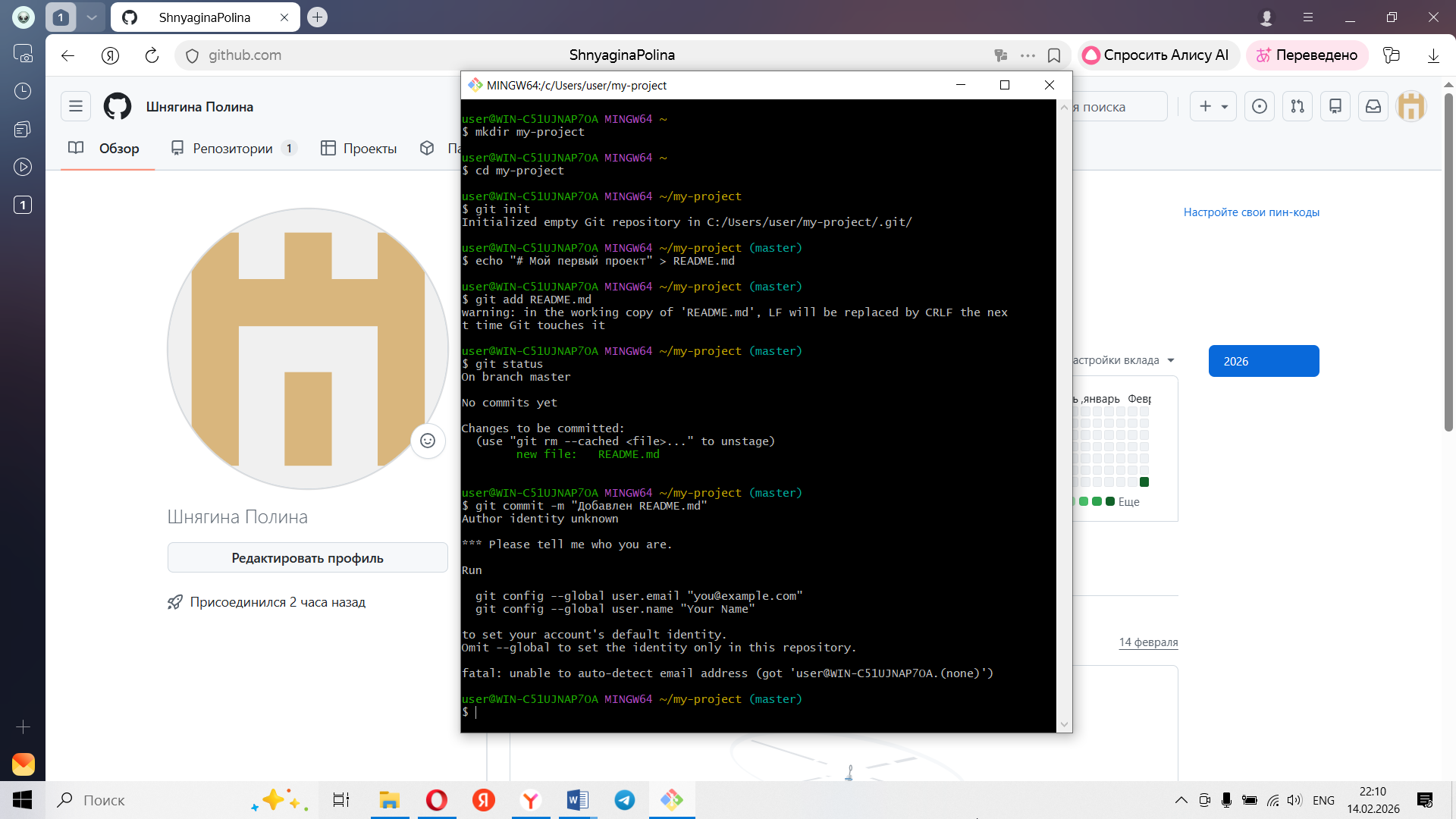


Проверила подключение с помощью команды ssh -T git@github.com



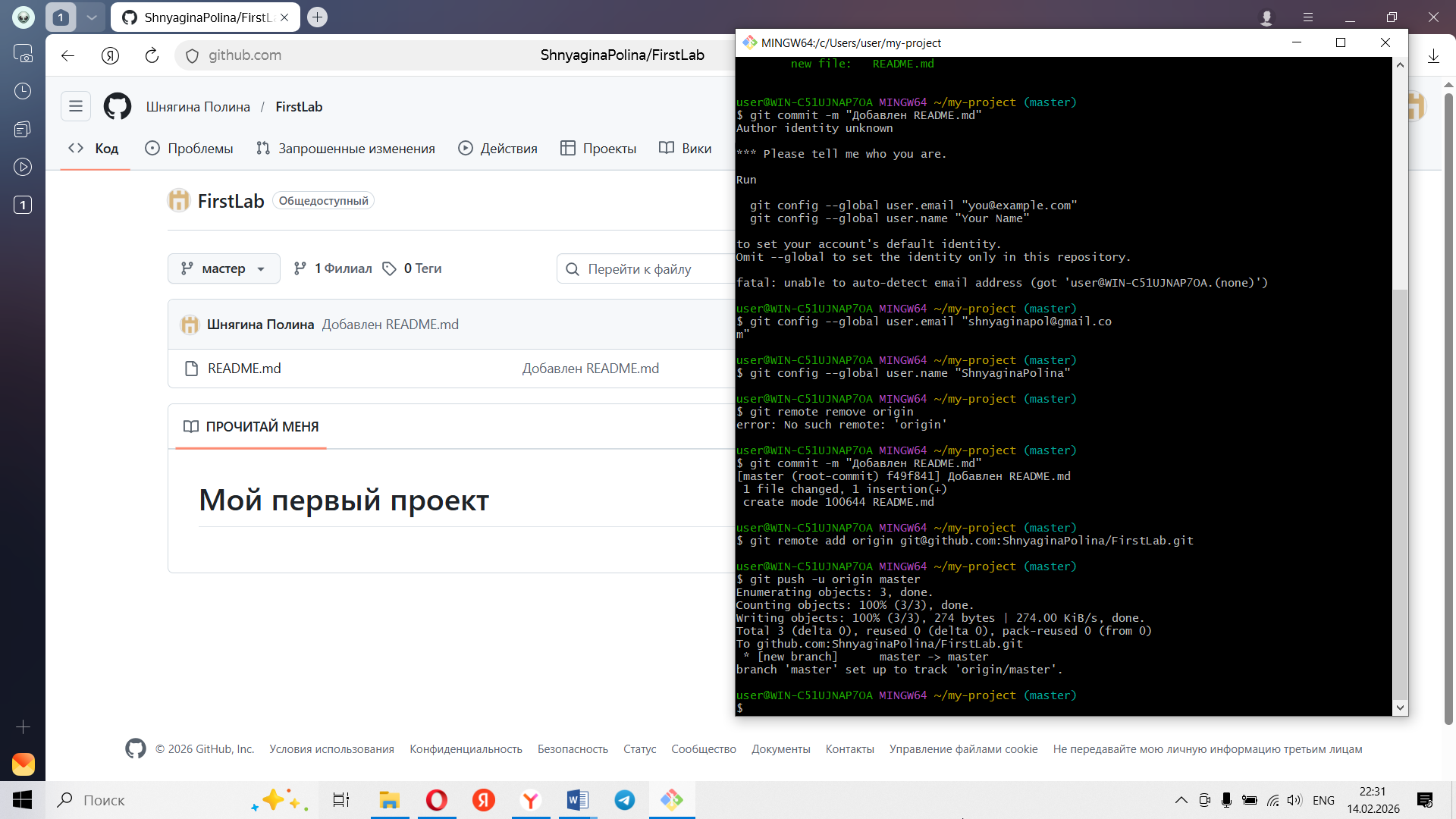
1. **Инициализация локального репозитория**

Создала папку для проекта, инициализировала Git, создала файл README.md, добавила файл в индекс, проверила статус, зафиксировала изменения.



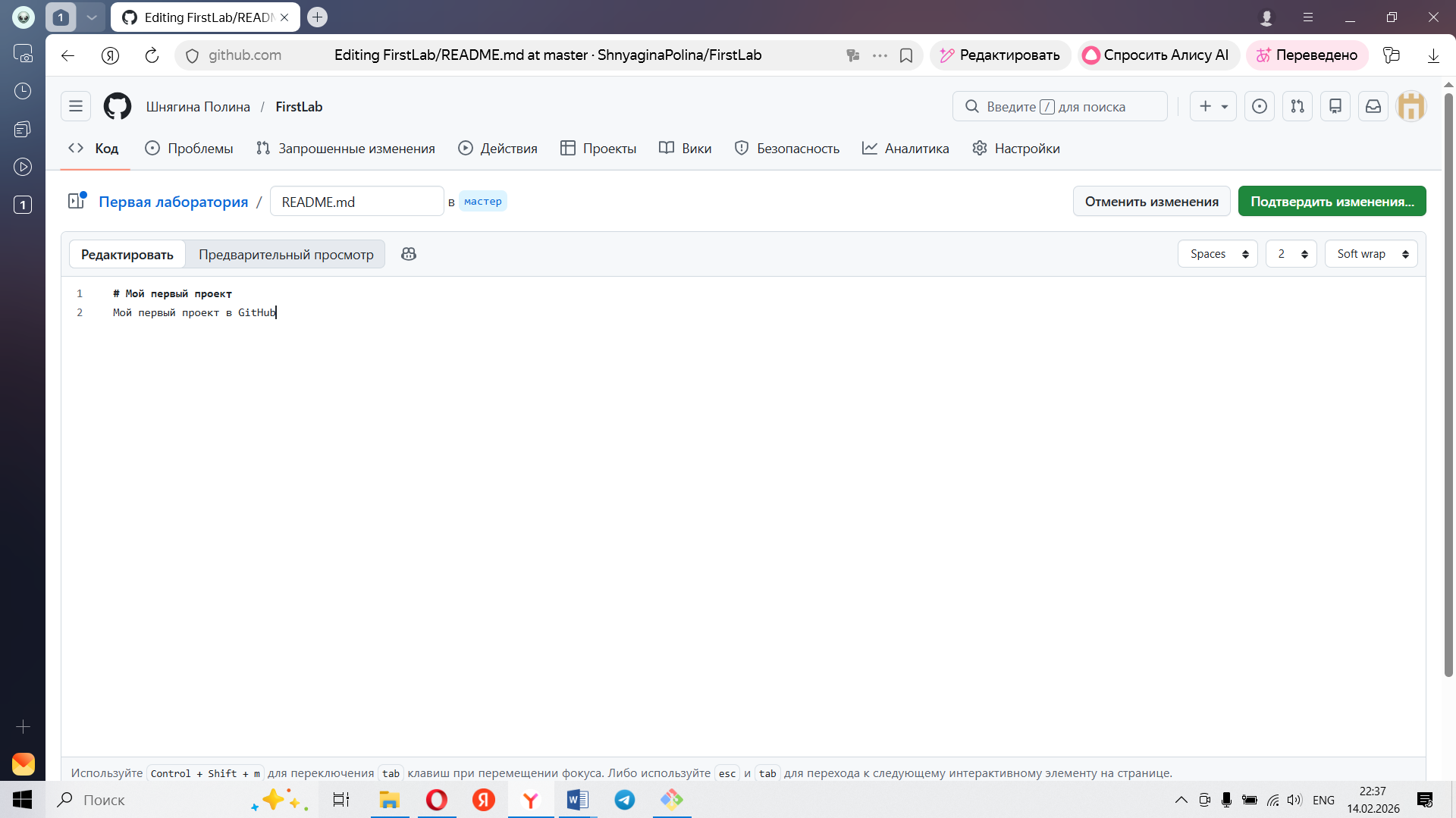
1. **Подключение удалённого репозитория и отправка изменений**

Подключила удалённый репозиторий через команду git remote add origin git@github.com:ShnyaginaPolina/FirstLab.git и отправила изменения на GitHub.

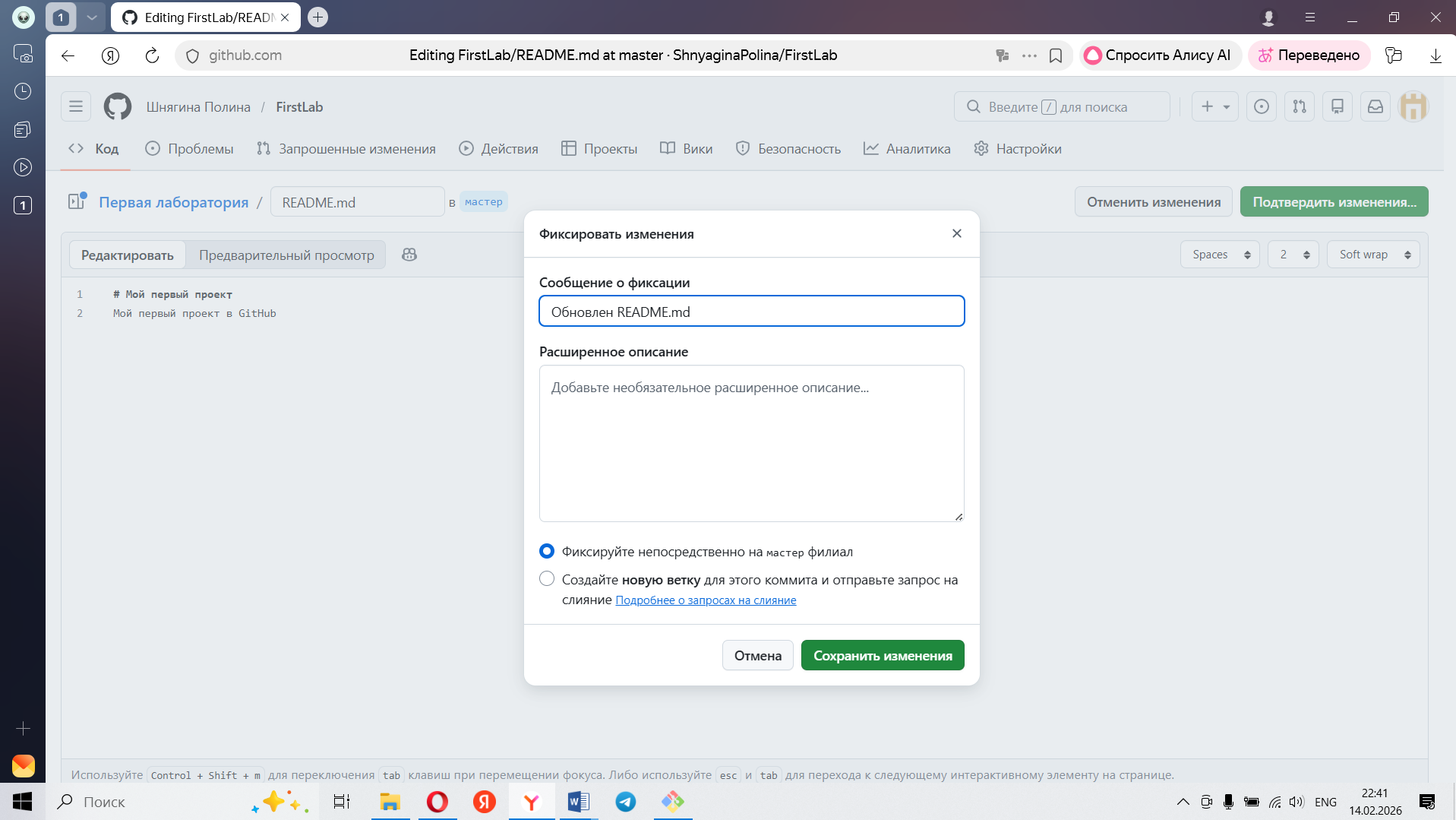


1. **Работа с файлами на GitHub**

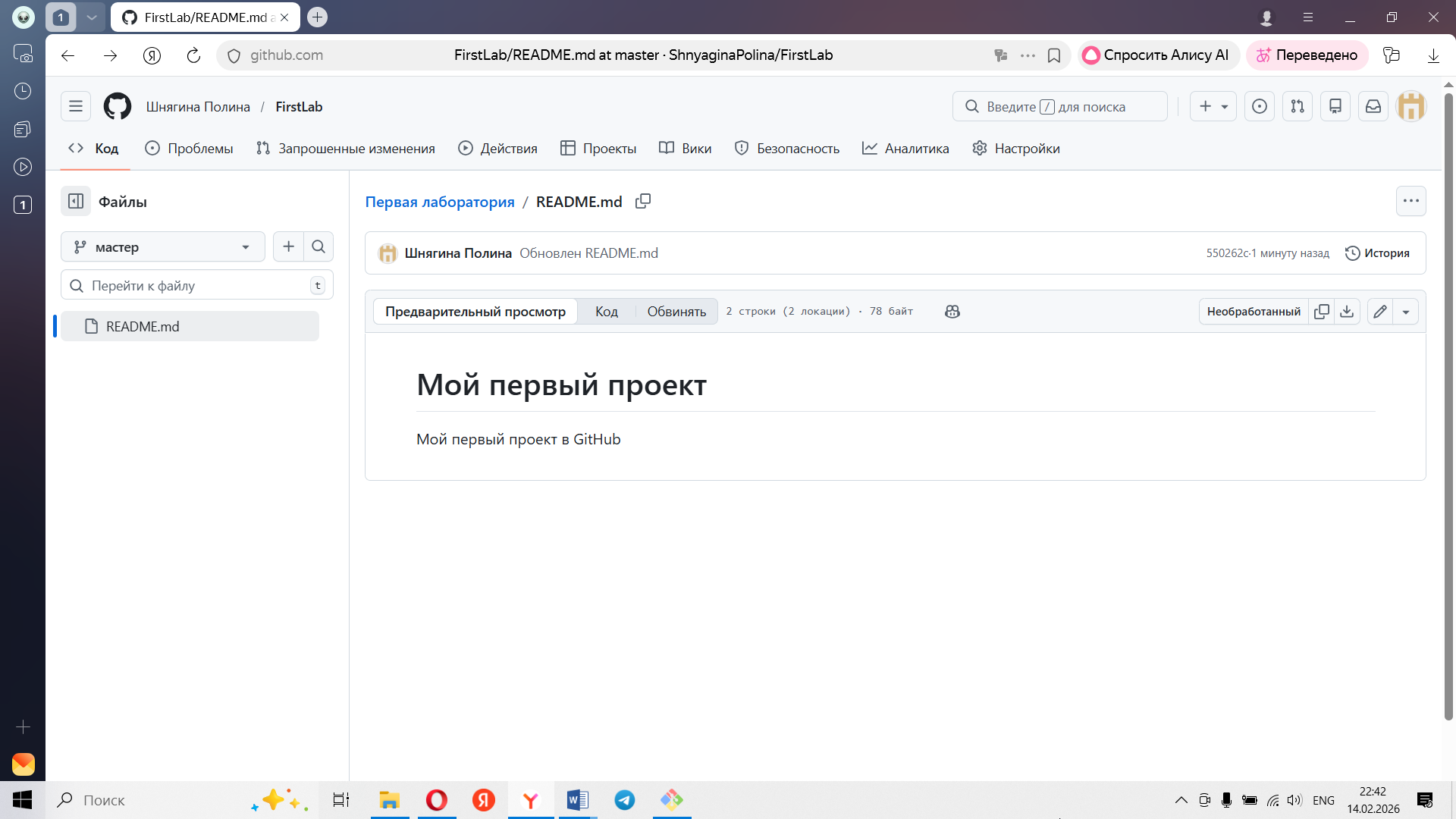
Добавила текст с описанием проекта



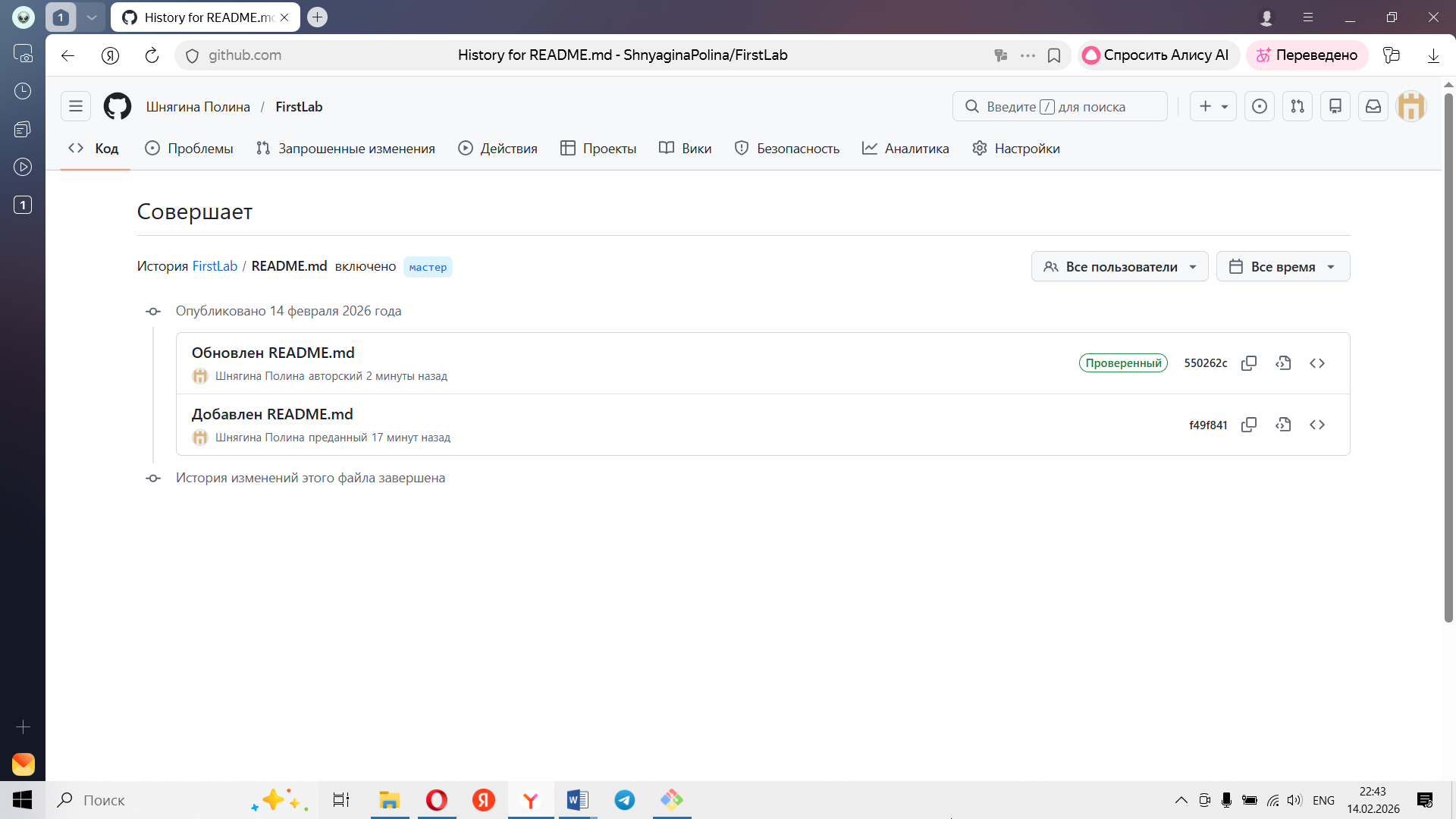
В сообщении о фиксации написала «Обновлен README».



Сохранила изменения

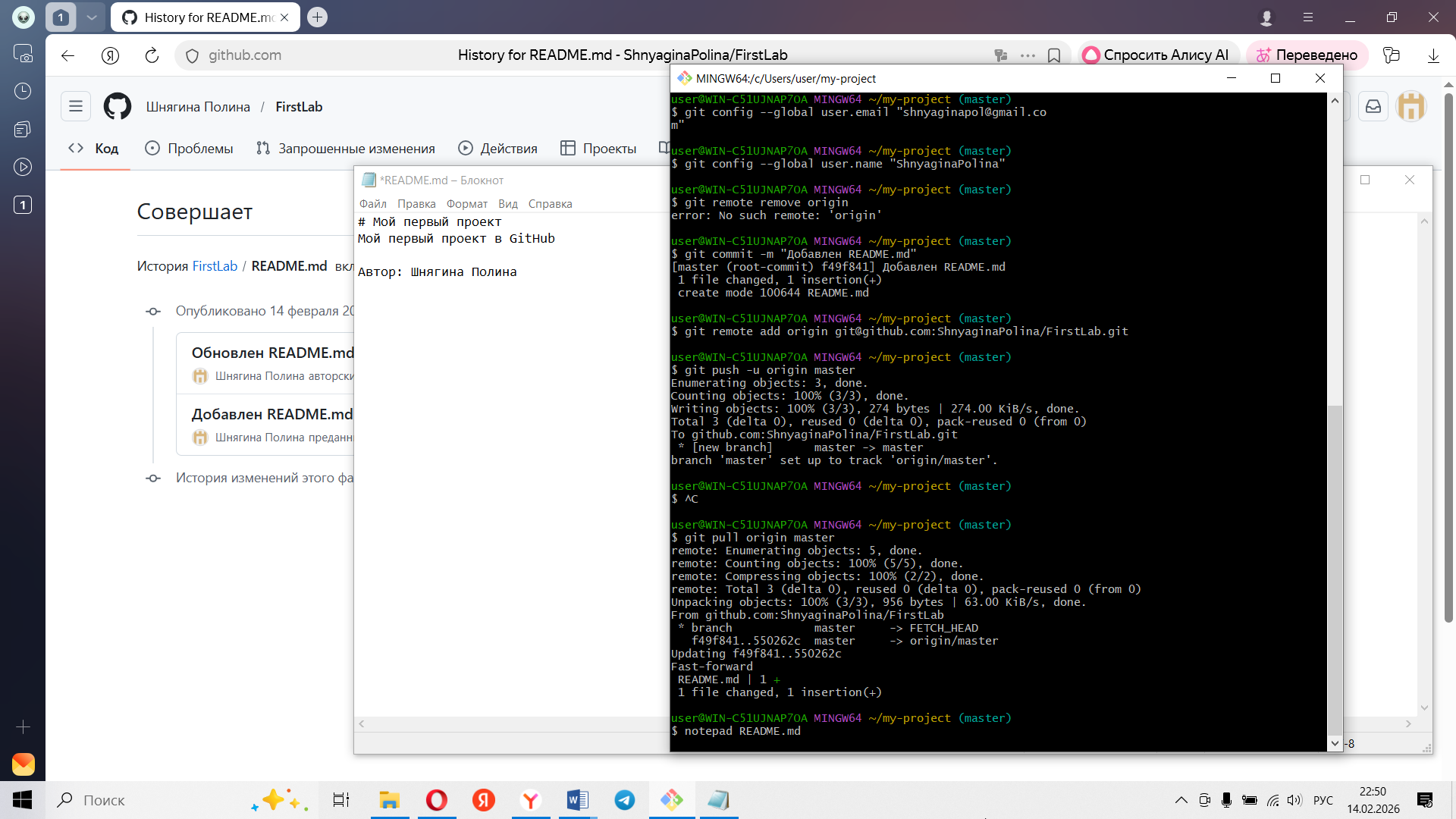


Проверила историю изменений

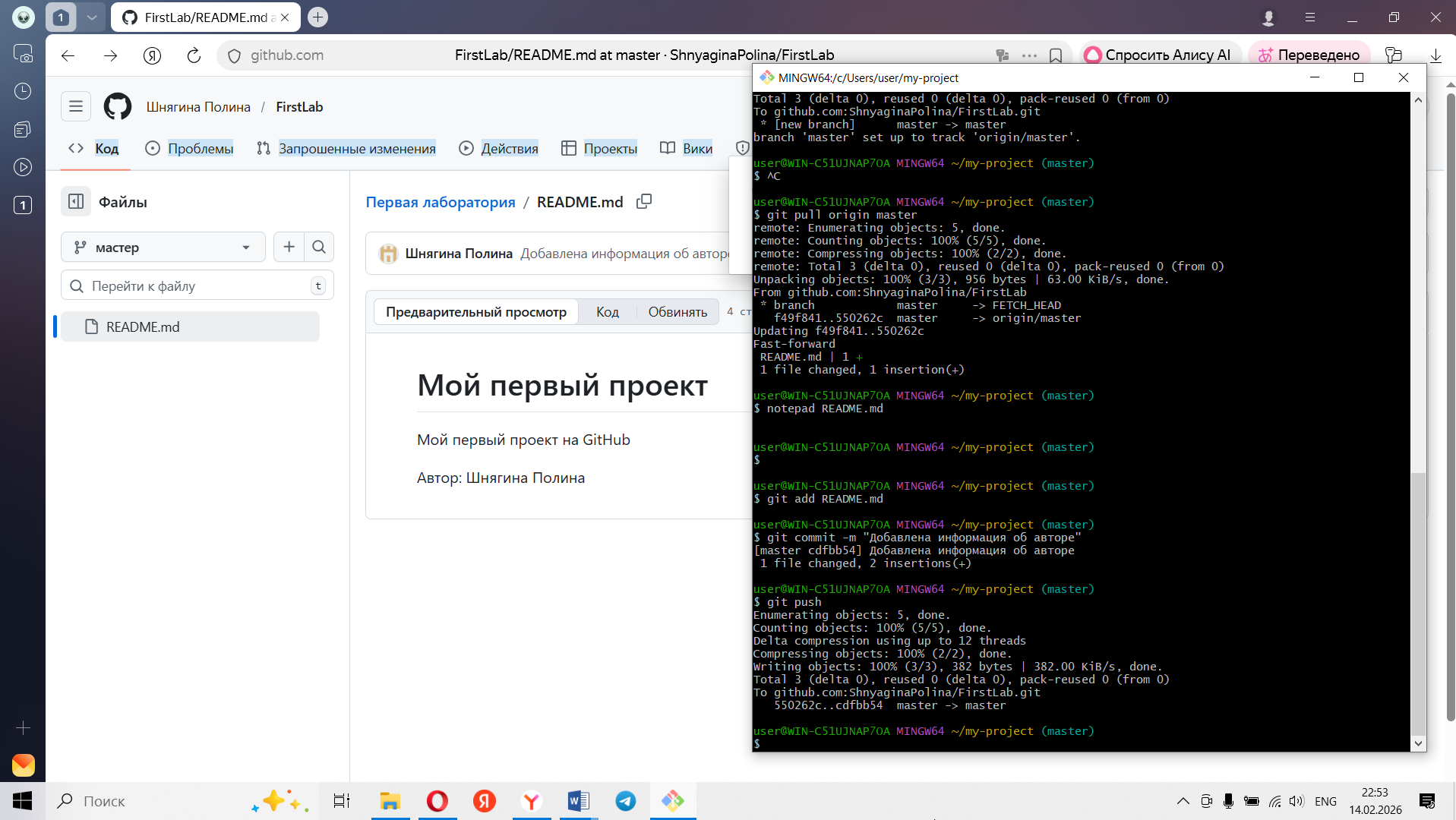


1. **Синхронизация локального и удалённого репозиториев**

Внесла локальные изменения в README.md: в блокноте добавила строку, содержащую информацию об авторе.

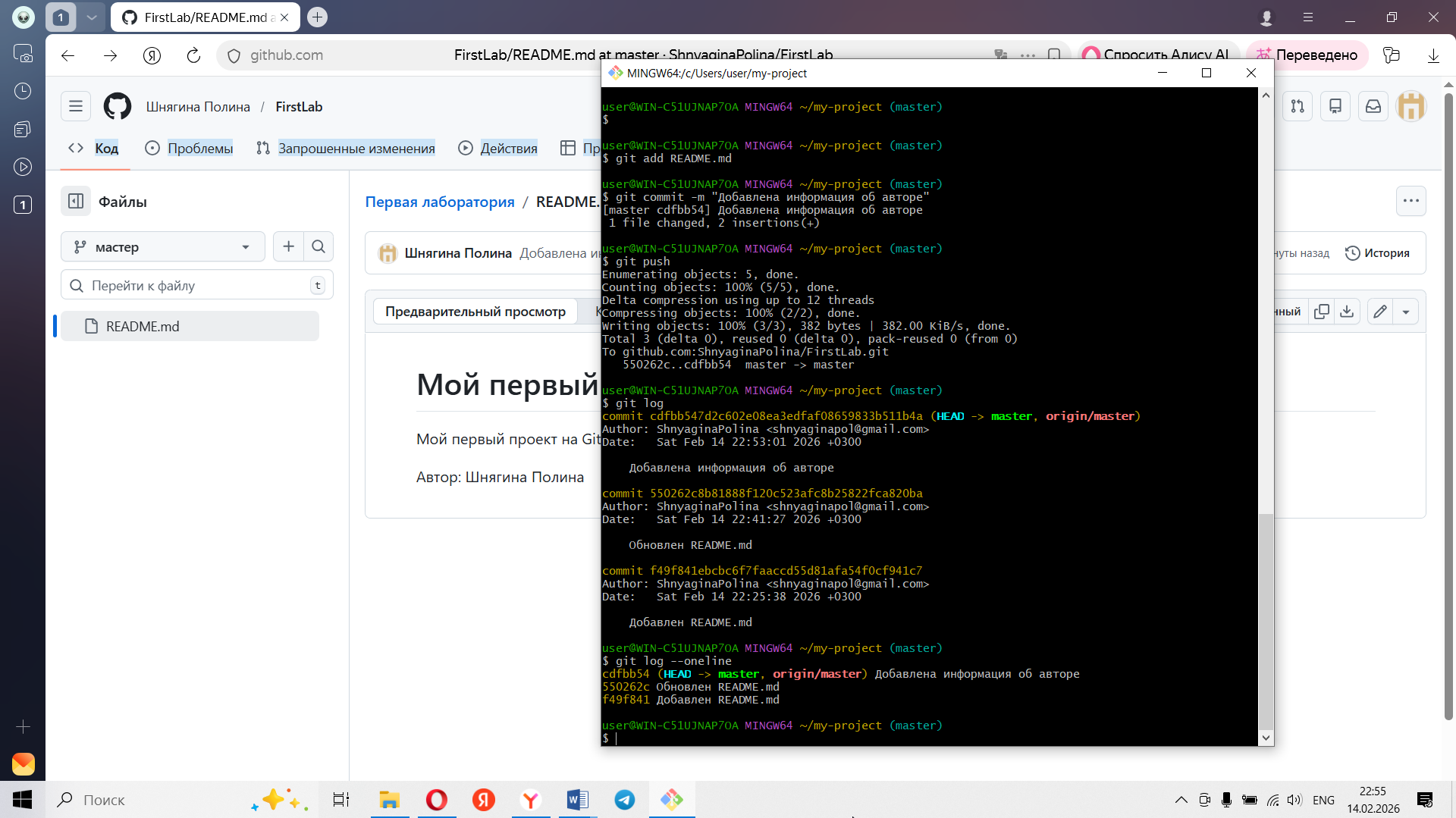


Зафиксировала и отправила, используя команды: git add README.md, git commit -m "Добавлена информация об авторе", git push.



1. **Изучение истории изменений**

Для просмотра списка коммитов я использовала команду git log. Для краткого вывода – команду git log --oneline.



# Контрольные вопросы

1. Репозиторий — хранилище файлов проекта с историей изменений; бывает локальным (на ПК) и удалённым (на сервере, например, GitHub).
2. SSH-ключ для GitHub — пара ключей (приватный + публичный) для безопасной аутентификации без пароля (клонирование, push, автоматизация).
3. Разница git commit и git push:
   * commit — фиксирует изменения локально;
   * push — отправляет локальные коммиты на удалённый сервер.
4. Просмотр изменённых файлов перед коммитом: git status, git diff, git diff --staged, git log --name-status.
5. Git pull — обновляет локальный репозиторий, загружая изменения с удалённого сервера (git fetch + git merge); пример: git pull origin main.

Вывод**:** в результате работы я освоила базовые операции работы с системой контроля версий Git и облачным сервисом GitHub: регистрацию, создание репозитория, настройку SSH‑ключей, базовые команды Git, редактирование файлов и управление изменениями.