**Создание базового репозитория в Git и организация работы с ветками.**

**Цель работы**

Освоить базовые принципы работы с системой контроля версий Git: создание репозитория, управление ветками, слияние изменений и взаимодействие с удаленным репозиторием.

**Выполнение работы**

**1. Установка и настройка Git**

**Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.**Git успешно установлен и настроен. Команда git --version подтвердила установку   
(git version 2.51.1).

**2. Создание локального репозитория**

Изображение выглядит как текст, снимок экрана, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.На рабочем столе создана новая папка “my-git-progect”. Осуществлен переход   
в папку, далее в ней инициализирован новый репозиторий (git init), создана базовая структура проекта и файл README.  
  
Вид в папке:

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Содержимое, созданное искусственным интеллектом, может быть неверным.

**3. Работа с коммитами**

**Изображение выглядит как текст, снимок экрана, программное обеспечение

Содержимое, созданное искусственным интеллектом, может быть неверным.**Файлы добавлены staging area (git add .) и создан первый коммит.

Изображение выглядит как текст, снимок экрана, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.История коммитов (git log) показывает автора, дату и сообщение коммита.

**4. Работа с ветками**

Создана и использована новая ветка feature/login.

git branch feature/login *# Создание ветки*

git checkout feature/login *# Переключение на ветку*

*# Внесены изменения в новой ветке (созданы файлы `src/login.py` и login.test.js)*

*Изображение выглядит как текст, снимок экрана, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.*

**

Вид в папке:

Изображение выглядит как текст, снимок экрана, Шрифт, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

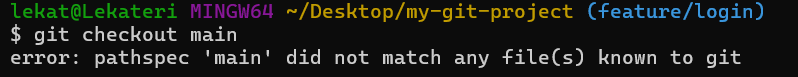
Изображение выглядит как текст, снимок экрана, Шрифт, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, снимок экрана, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.Добавлены и закомминчены изменения (важно после введения команды git add . нажать не enter, а shift+enter во избежание ошибок). Ветка feature/login создана, в нее успешно закоммичены изменения.

**5. Слияние веток**, разрешение возможных конфликтов (при их наличии).

Осуществляем возврат в основную ветку (main), при этом возникает ошибка, т.к. эта ветка не была создана.

  
Создаем ветку main.

Слияние изменений (git merge feature/login).

Изображение выглядит как текст, Шрифт, снимок экрана, Графика

Содержимое, созданное искусственным интеллектом, может быть неверным.Чтобы выполнить слияние (merge) ветки feature/login с текущей веткой, выполним команду git merge feature/login.

Получаем сообщение об ошибке. Ветка не слилась, потому что ветка feature/login не существует или уже удалена.

Изображение выглядит как текст, снимок экрана, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.Проверим, какие ветки существуют (git branch -a).

Видим, что их две: main и master. Ветки feature/login нет. Решаем конфликт (по одной вводим команды):

* Создаем ветку заново: git branch feature/login
* Переключаемся на нее: git checkout feature/login
* Сохраняем изменения: git add .
* Возвращаемся в main и сливаем: git checkout main (выполняем через Shift + Enter), git merge feature/login (выполняем через Shift + Enter).

Изображение выглядит как текст, снимок экрана, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

Сохраняем файлы и завершаем слияние:

* git add . - через Shift + Enter
* git commit -m "Разрешение конфликтов после слияния feature/login" - через Shift + Enter

Изображение выглядит как текст, снимок экрана, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.Проверяем изменения – git log --oneline --graph --all

Видим, что изменения из feature/login успешно перенесены в main. Конфликтов слияния не возникло. История коммитов теперь показывает слияние.

**6. Работа с удаленным репозиторием (**[**GitHub.com**](https://github.com/)**)**

* Изображение выглядит как текст, снимок экрана, Шрифт, число

  Содержимое, созданное искусственным интеллектом, может быть неверным.Создан репозиторий на GitHub**.**

Изображение выглядит как текст, снимок экрана, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.Локальный репозиторий my-git-progect связан с удаленным репозиторием на GitHub (git remote add origin <https://github.com/lekateri/my-git-project.git>).

Изменения отправлены на сервер (git push -u origin main).

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

**Результат:** Код и история коммитов доступны в удаленном репозитории по указанной ссылке.

**7. Дополнительные задания**

**Использование .gitignore**

* Создан файл .gitignore. Сохранены изменения и добавлены коммиты.

Изображение выглядит как текст, снимок экрана, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

В папке

Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Содержимое, созданное искусственным интеллектом, может быть неверным.

* Отмена изменений

Отмена незакоммиченных изменений в файле:

git checkout -- имя\_файла

Изображение выглядит как текст, Шрифт, снимок экрана

Содержимое, созданное искусственным интеллектом, может быть неверным.

Отмена индексации (убрать из staging area):

Изображение выглядит как текст, снимок экрана, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

Создание отменяющего коммита

Для этого нужна команда git revert хеш\_коммита

Чтобы получить хеш коммита водим в Git bash команду:

$ git log –oneline

Выбираем номер для отмены коммита:

Изображение выглядит как текст, снимок экрана, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

Вводим git revert ваш хеш\_коммита

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Содержимое, созданное искусственным интеллектом, может быть неверным.

### Проверка результатов на удаленном репозитории:

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.Изображение выглядит как текст, снимок экрана, Шрифт, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

Все файлы загружены, история коммитов проверена.

**Результаты работы**

* Создан локальный Git-репозиторий с основной веткой main и feature-веткой feature/login.
* Удаленный репозиторий размещен на платформе GitHub.
* История коммитов, демонстрирующая работу с ветками.
* Выполнены дополнительные практики для лучшего понимания Git.
* Создан данный текстовый документ Word и подготовлены скриншоты, демонстрирующие выполнение работы.

**Выводы**

В ходе работы были успешно освоены и применены на практике основные команды Git. Задача выполнена в полном объеме, все критерии оценки соблюдены.