Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии
Департамент цифровых, робототехнических систем и электроники

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №1
дисциплины
«Основы кроссплатформенного программирования»

Выполнил:
Борцов Богдан Михайлович
2 курс, группа ИТС-б-о-23-1,
11.03.02 «Инфокоммуникационные технологии и системы связи»,
направленность (профиль) «Инфокоммуникационные системы и сети»,
очная форма обучения

_____
(подпись)

Проверил:
Воронкин Р.А.

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2024 г.

Тема: Исследование основных возможностей Git и GitHub

Цель: исследовать базовые возможности системы контроля версий Git и веб-сервиса для хостинга IT-проектов GitHub.

Порядок выполнения работы:

1. Изучил и законспектировал основные сведения.

Система контроля версий (СКВ) — это система, регистрирующая изменения в одном или нескольких файлах с тем, чтобы в дальнейшем была возможность вернуться к определённым старым версиям этих файлов.

Когда вы производите какие-либо действия в Git, практически все из них только добавляют новые данные в базу Git. Очень сложно заставить систему удалить данные либо сделать что-то, что нельзя впоследствии отменить. Как и в любой другой СКВ, вы можете потерять или испортить свои изменения, пока они не зафиксированы, но после того, как вы зафиксируете снимок в Git, будет очень сложно что-либо потерять, особенно, если вы регулярно синхронизируете свою базу с другим репозиторием. Всё это превращает использование Git в одно удовольствие, потому что мы знаем, что можем экспериментировать, не боясь серьёзных проблем

У Git есть три основных состояния, в которых могут находиться ваши файлы: зафиксированное (committed), изменённое (modified) и подготовленное (staged).

Зафиксированный значит, что файл уже сохранён в вашей локальной базе.

К изменённым относятся файлы, которые изменились, но ещё не были зафиксированы.

Подготовленные файлы — это изменённые файлы, отмеченные для включения в следующий коммит.

2. Ознакомился с методическими указаниями и выполнил их.

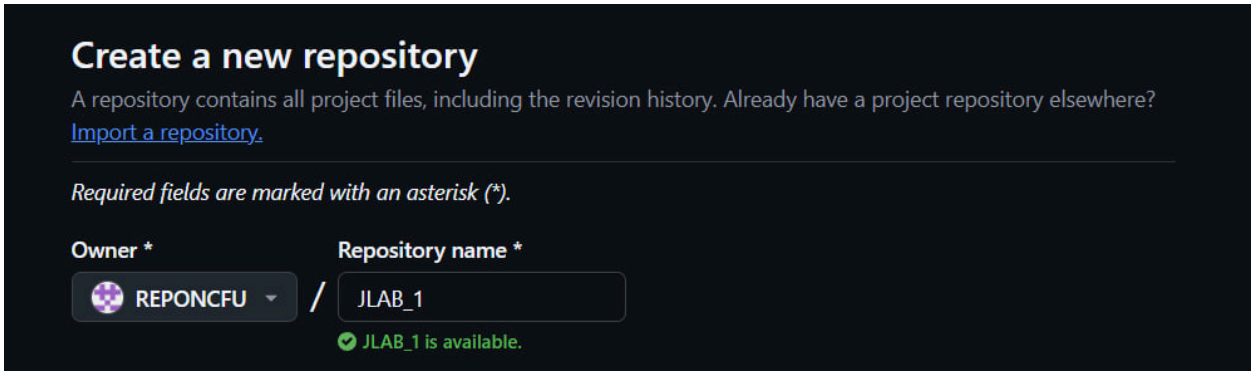

Рис. 1 – Создание репозитория, используя GitHub:

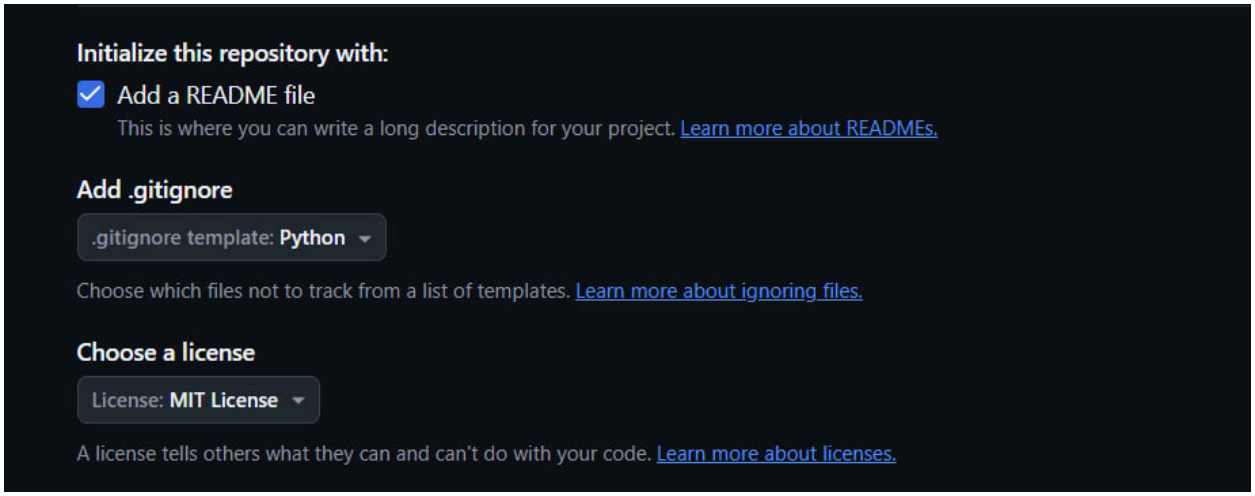

Рис.2. – Добавление Readme & .gitignore, а также MIT лицензии

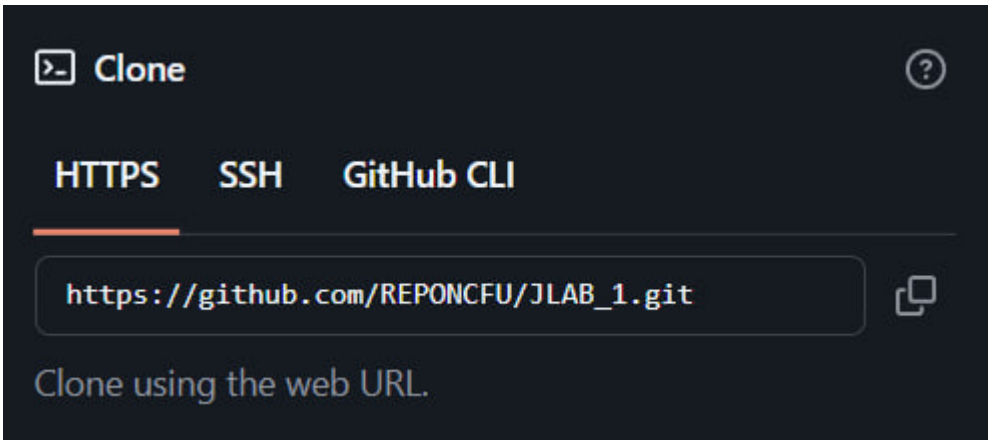

Рис.3 – Ссылка на репозиторий

```
C:\Users\Elony\Desktop\Portable\PortableGit>git clone https://github.com/REPONCFU/JLAB_1.git
Cloning into 'JLAB_1'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (5/5), done.

C:\Users\Elony\Desktop\Portable\PortableGit>cd JLAB_1

C:\Users\Elony\Desktop\Portable\PortableGit\JLAB_1>
```

Рис. 4 – Клонирование репы на локальную машину

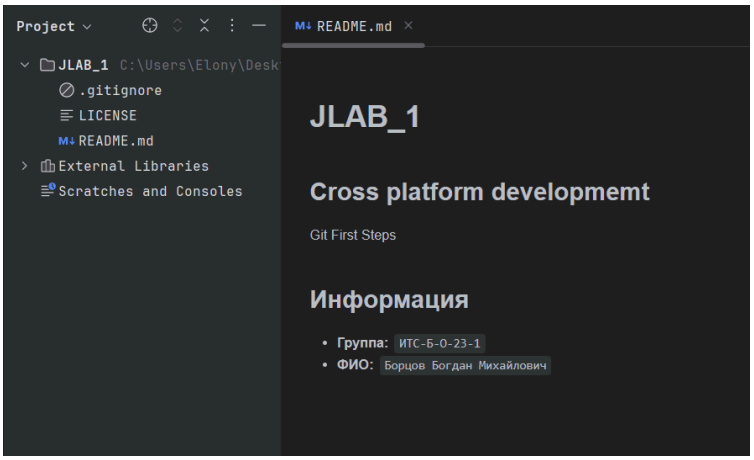

Рис. 5 – Отредактированный файл README.md

```
C:\Users\Elony\Desktop\Portable\PortableGit\JLAB_1>git add .

C:\Users\Elony\Desktop\Portable\PortableGit\JLAB_1>git commit -m "Добавлена библиотека requests и requirements.txt"
[main 2e0d977] Добавлена библиотека requests и requirements.txt
 3 files changed, 2 deletions(-)
 create mode 100644 main.py
 create mode 100644 requirements.txt

C:\Users\Elony\Desktop\Portable\PortableGit\JLAB_1>git add .

C:\Users\Elony\Desktop\Portable\PortableGit\JLAB_1>git commit -m "Реализована базовая функция для получения данных по IP"
[main 26c3609] Реализована базовая функция для получения данных по IP
 3 files changed, 14 insertions(+)
 create mode 100644 .idea/.name

C:\Users\Elony\Desktop\Portable\PortableGit\JLAB_1>git commit -m "Извлечены данные о широте, долготе и городе"

On branch main
Your branch is ahead of 'origin/main' by 2 commits.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   main.py

no changes added to commit (use "git add" and/or "git commit -a")

C:\Users\Elony\Desktop\Portable\PortableGit\JLAB_1>git add .

C:\Users\Elony\Desktop\Portable\PortableGit\JLAB_1>git commit -m "Извлечены данные о широте, долготе и городе"

[main 3551a38] Извлечены данные о широте, долготе и городе
 1 file changed, 10 insertions(+), 1 deletion(-)

C:\Users\Elony\Desktop\Portable\PortableGit\JLAB_1>git add .

C:\Users\Elony\Desktop\Portable\PortableGit\JLAB_1>git commit -m "Добавлена обработка ошибок и комментарии"
[main 3614f37] Добавлена обработка ошибок и комментарии
 1 file changed, 4 insertions(+)

C:\Users\Elony\Desktop\Portable\PortableGit\JLAB_1>git add .

C:\Users\Elony\Desktop\Portable\PortableGit\JLAB_1>git commit -m "Добавлен пример использования функции для проверки"
[main 64000ff] Добавлен пример использования функции для проверки
 1 file changed, 9 insertions(+), 1 deletion(-)

C:\Users\Elony\Desktop\Portable\PortableGit\JLAB_1>git add .

C:\Users\Elony\Desktop\Portable\PortableGit\JLAB_1>git commit -m "Финальная стилизация и улучшения кода"
[main 52dbc0b] Финальная стилизация и улучшения кода
 1 file changed, 10 insertions(+), 6 deletions(-)
```

Рис. 6 – Улучшение кода и постепенное создание коммитов

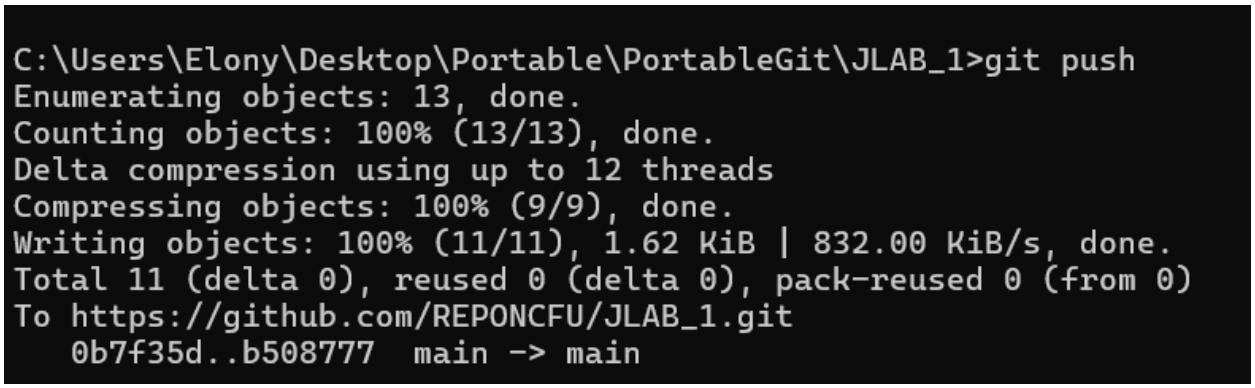

Рис. 7 – Push на GitHub

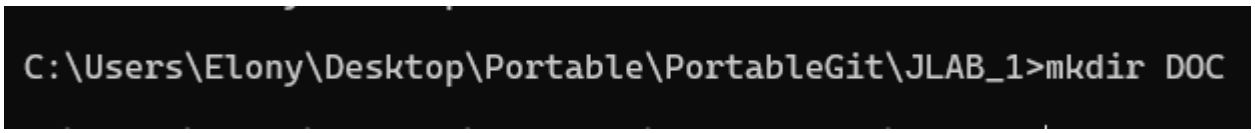

Рис. 7 – Создал папку DOC в локальном репозитории

3. Ответил на контрольные вопросы.
4. Написал вывод на основе выполненной работы.
5. Сохранил отчет в PDF формате
6. Добавил отчет в папку DOC
7. Отправил локальные изменения в репозиторий GitHub.

Ответы на контрольные вопросы:

1. Что такое СКВ и каково ее назначение?

Система контроля версий (СКВ) — это программное обеспечение, которое позволяет разработчикам отслеживать изменения в исходном коде, возвращаться к предыдущим версиям файлов, сравнивать их, сливать изменения, а также управлять командной работой над проектом.

2. В чем недостатки локальных и централизованных СКВ?

- Локальные СКВ: изменения отслеживаются только на одном компьютере. Это усложняет командную работу, восстановление данных в случае поломки компьютера.

- Централизованные СКВ: единственный сервер хранит все данные и историю изменений. Если сервер недоступен или поврежден, работа затрудняется.

3. К какой СКВ относится Git?

Git относится к распределенным системам контроля версий (РСКВ), где каждый разработчик имеет полную копию репозитория.

4. В чем концептуальное отличие Git от других СКВ?

Главное отличие Git заключается в его распределенной природе: каждый пользователь имеет локальную копию всего репозитория с полной историей изменений. Это делает систему более гибкой и устойчивой к сбоям, в отличие от централизованных систем.

5. Как обеспечивается целостность хранимых данных в Git?

Git использует хеширование (SHA-1) для всех объектов (файлы, каталоги, коммиты), что гарантирует целостность данных. Если данные изменены, хеш изменяется, и система сразу замечает проблему.

6. В каких состояниях могут находиться файлы в Git? Как связаны эти состояния?

Файлы могут находиться в трех состояниях:

- Измененные (modified): файл был изменен, но не добавлен в индекс.
- Проиндексированные (staged): файл добавлен в индекс (через `git add`), готов к коммиту.
- Зафиксированные (committed): изменения сохранены в истории репозитория.

Связаны эти состояния процессом работы: сначала файл изменяется, затем индексируется, и в конечном итоге фиксируется коммитом.

7. Что такое профиль пользователя в GitHub?

Профиль пользователя в GitHub — это учетная запись, содержащая информацию о пользователе (имя, контактные данные, проекты, активности), которая также позволяет управлять репозиториями и взаимодействовать с другими разработчиками.

8. Какие бывают репозитории в GitHub?

Репозитории могут быть:

- Публичные: доступны всем пользователям GitHub.
- Приватные: видны только владельцам и приглашенным пользователям.

9. Укажите основные этапы модели работы с GitHub.

1. Создание репозитория.
2. Клонирование репозитория на локальный компьютер.
3. Внесение изменений и фиксация (коммит).
4. Отправка изменений на сервер (push).
5. Получение изменений из удаленного репозитория (pull).
6. Управление ветками и слияниями (merge).

10. Как осуществляется первоначальная настройка Git после установки?

После установки необходимо настроить имя пользователя и email:

git config --global user.name "Ваше Имя"

git config --global user.email "ваш.email@example.com"

11. Опишите этапы создания репозитория в GitHub.

1. Войти в GitHub.
2. Нажать "New Repository".
3. Указать имя репозитория.
4. Выбрать тип репозитория (публичный или приватный).

5. Создать репозиторий.

12. Какие типы лицензий поддерживаются GitHub при создании репозитория?

GitHub поддерживает лицензии:

- MIT
- Apache 2.0
- GPL (GNU General Public License)
- BSD и др.

13. Как осуществляется клонирование репозитория GitHub? Зачем нужно клонировать репозиторий?

Клонирование осуществляется командой:

git clone https://github.com/username/repository.git

Клонирование необходимо для того, чтобы получить копию удаленного репозитория на локальный компьютер и начать с ним работу.

14. Как проверить состояние локального репозитория Git?

Команда:

git status

Показывает текущее состояние файлов: изменены ли они, проиндексированы ли и нужно ли их зафиксировать.

15. Как изменяется состояние локального репозитория Git после выполнения следующих операций?

- Добавление/изменение файла: файл переходит в состояние "modified".

- git add: файл перемещается в состояние "staged".

- git commit: файл переходит в состояние "committed".

- git push: изменения отправляются на удаленный сервер.

16. Как синхронизировать два локальных репозитория с репозиторием GitHub?

1. На первом компьютере:

git clone https://github.com/username/repository.git

2. На втором компьютере:

git clone https://github.com/username/repository.git

Для синхронизации изменений:

git pull

git add .

git commit -m "Сообщение"

git push

17. GitHub является не единственным сервисом, работающим с Git. Какие сервисы еще Вам известны?

Среди других сервисов:

- GitLab

- Bitbucket

18. Какие Вам известны программные средства с графическим интерфейсом пользователя для работы с Git?

- GitKraken

- SourceTree

- GitHub Desktop

Вывод:

В ходе лабораторной работы я изучил возможности Git и GitHub. Git, являясь распределенной системой контроля версий, дает большую гибкость в управлении изменениями и облегчает командную работу над проектами, позволяя каждому участнику эффективно вносить и отслеживать изменения. GitHub, в свою очередь, предлагает мощные инструменты для хостинга репозиториев и совместной работы, включая инструменты для код-ревью и автоматизации процессов. В ходе работы мы освоили основные команды Git, а также принципы работы с удалёнными репозиториями, что помогло понять, как синхронизировать изменения между локальной и удаленной версиями проекта. Использование Git и GitHub позволяет не только эффективно управлять версиями кода, но и значительно повысить надёжность хранения данных, а также упростить обмен и совместную работу в рамках команды.