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

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО РАБОТЕ №1.1.
дисциплины «Основы кроссплатформенного программирования»

Выполнил:
Сластёнов Андрей Сергеевич
1 курс, группа ИТС-б-о-21-1,
11.03.02 «Инфокоммуникационные технологии и системы связи»,
направленность (профиль) Сети связи и системы коммутации,
очная форма обучения

_____
(подпись)

Руководитель практики от университета:
Воронкин Р.А., канд. тех. наук, доцент,
доцент кафедры инфокоммуникаций

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г

Тема: исследование основных возможностей Git и GitHub.

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

Порядок выполнения работы:

1. Создание нового репозитория в GitHub.

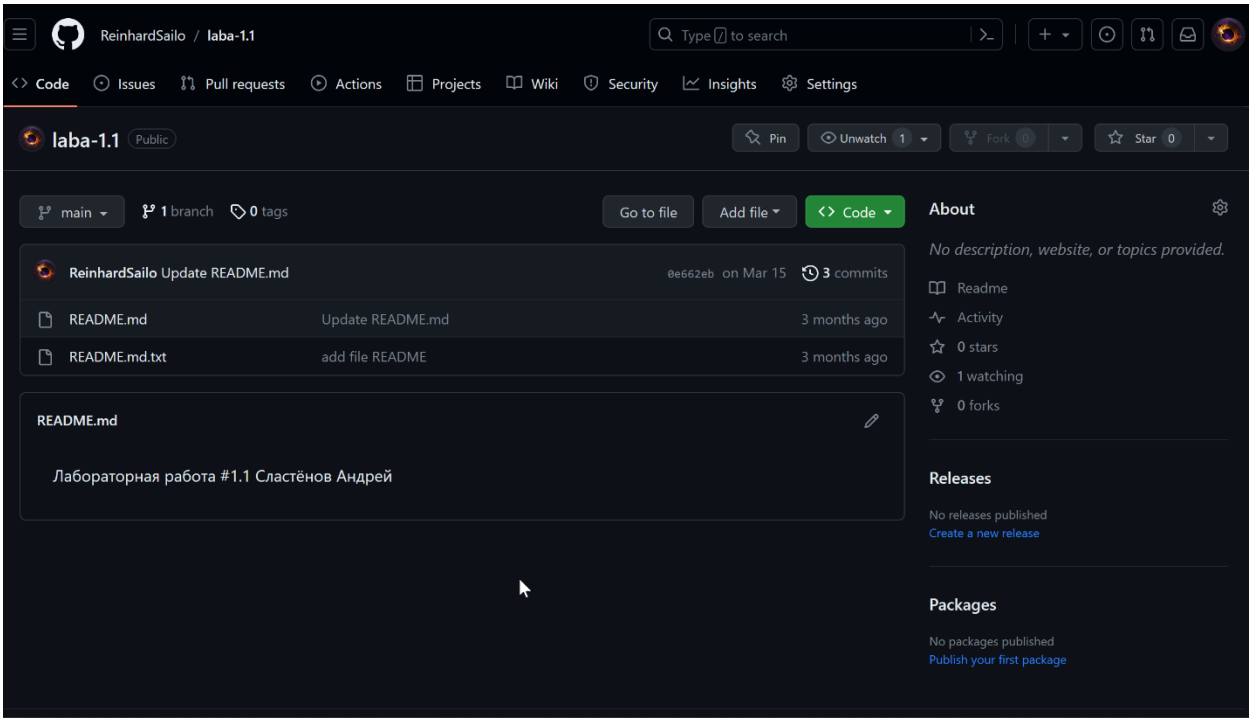

Рисунок 1. Новый репозиторий.

2. Ввел в командную строку git -v, таким образом проверил, что все работает.

```
C:\Users\Andre>git -v
git version 2.39.2.windows.1
```

Рисунок 2. git version.

3. Ввел свое имя и свой email.

```
C:\Users\Andre>git config --global user.name "Andrey"

C:\Users\Andre>git config --global usr.emai "mrsailoris@gmail.com"
```

Рисунок 3. Имя и почта

4. Дополнил файл .gitignore необходимым правилом игнорировать файлы .idea

```
# Prerequisites
*.d

# Compiled Object files
*.slo
*.lo
*.o
*.obj

# Precompiled Headers
*.gch
*.pch

# Compiled Dynamic libraries
*.so
*.dylib
*.dll

# Fortran module files
*.mod
*.smod

# Compiled Static libraries
*.lai
*.la
*.a
*.lib

# Executables
*.exe
*.out
```

Рисунок 6. Дополнение файла gitignore.

5. Внес изменения в файл README.md

```
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   README.md
```

Рисунок 6. Результат изменений в README.md

6. Написал небольшую программу на языке JavaScript, фиксировал изменения при написании в локальном репозитории, сделал не менее 7 коммитов. Все это в файле README, как указанно в методических указаниях

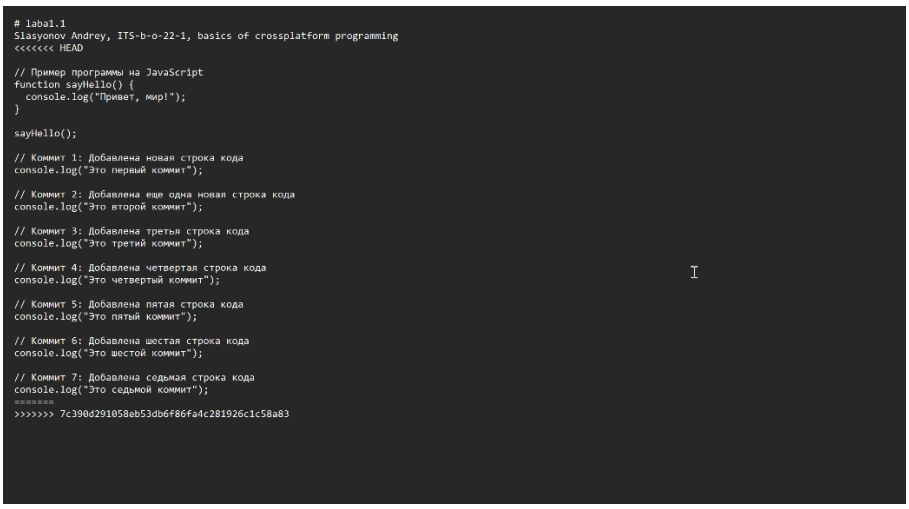

Рисунок 7. Написание программы и добавление ее в файл README

```
C:\Users\Andre\laba1.1>git pull
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the result.

C:\Users\Andre\laba1.1>git add README.md

C:\Users\Andre\laba1.1>git commit -m "Добавлен код программы с коммитами"
[main cb9828d] Добавлен код программы с коммитами

C:\Users\Andre\laba1.1>git push
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 8 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 1.05 KiB | 1.05 MiB/s, done.
Total 6 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/ReinhardSailo/laba1.1.git
   7c390d2..cb9828d  main -> main
```

Рисунок 8. Список коммитов

6. Отправил в удаленный репозиторий GitHub.

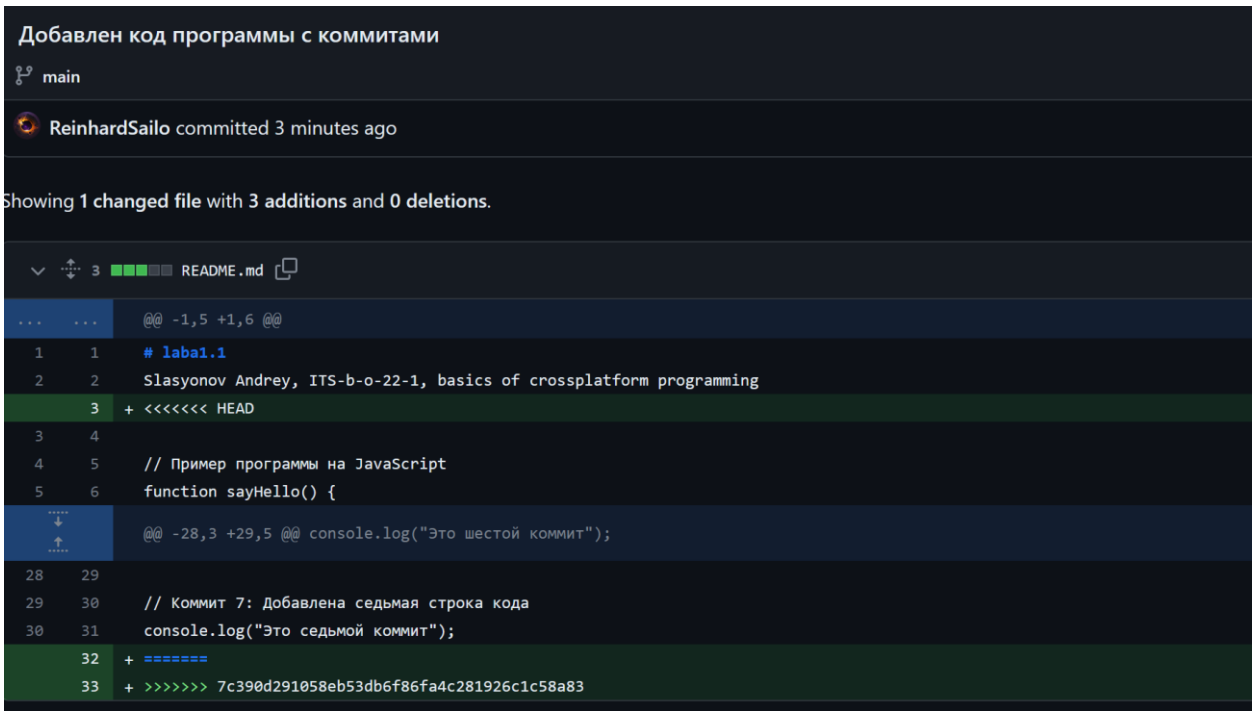

Рисунок 11. Проверка изменений в GitHub

Ссылка: https://github.com/ReinhardSailo/laba1.1

Ответы на контрольные вопросы:

1) Что такое СКВ и каково ее назначение? 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.

2) В чем недостатки локальных и централизованных СКВ? Основной недостаток локальных СКВ – можно легко забыть, в какой директории мы находимся, и случайно изменить не тот файл или скопировать не те файлы, которые мы хотели. Основной недостаток централизованных СКВ заключается в том, что это единая точка отказа, представленная централизованным сервером. Если этот сервер выйдет из строя на час, то в течение этого времени никто не сможет использовать контроль версий для сохранения изменений, над которыми работает, а также никто не сможет обмениваться этими изменениями с другими разработчиками.

3) К какой СКВ относится Git? Git относится к распределенным СКВ (РСКВ)

4) В чем концептуальное отличие Git от других СКВ? Основное отличие Git от любой другой СКВ (включая Subversion и её собратьев) — это подход к работе со своими данными. Концептуально, большинство других систем хранят информацию в виде списка изменений в файлах. Git не хранит и не обрабатывает данные таким способом. Вместо этого, подход Git к хранению данных больше похож на набор снимков миниатюрной файловой системы.

5) Как обеспечивается целостность хранимых данных в Git? В Git для всего вычисляется хеш-сумма, и только потом происходит сохранение. В дальнейшем обращение к сохранённым объектам происходит по этой хеш-сумме. Это значит, что невозможно изменить содержимое файла или директории так, чтобы Git не узнал об этом.

6) В каких состояниях могут находится файлы в Git? Как связаны эти состояния? 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 поменялись, но ещё не были зафиксированы. Подготовленные файлы — это изменённые файлы, отмеченные для включения в следующий коммит.

7) Что такое профиль пользователя в GitHub? Профиль - это наша публичная страница на GitHub, как и в социальных сетях.

8) Какие бывают репозитории в GitHub? Репозиторий бывает трех видов: локальный, централизованный, распределенный.

9) Укажите основные этапы модели работы с GitHub. GitHub содержит в себе два хранилища: А) upstream - это оригинальный репозиторий проекта, который мы скопировали. Б) origin - ваш fork (копия) на GitHub, к которому у вас есть полный доступ. Чтобы перенести изменения с вашей копии в исходному репозиторий проекта, нам нужно сделать запрос на извлечение.

10) Как осуществляется первоначальная настройка Git после установки? Чтобы убедиться в том, что мы установили Git правильно необходимо вписать команду git version, если она сработала необходимо написать свое имя и почту с помощью следующих команд: git config --global user.name “Name” git config --global user.email “Email”

11) Опишите этапы создания репозитория в GitHub. a) Имя репозитория. Оно может быть любое, необязательно уникальное во всем github, потому что привязано к вашему аккаунту, но уникальное в рамках тех репозиториев, которые вы создавали. b) Описание (Description). Можно оставить пустым. c) Public/private. Выбираем открытый (Public), НЕ ставим галочку “Initialize this repository with a README” (В README потом будет лежать какая-то основная информация, что же такое ваш проект и как с ним работать). d) .gitignore и LICENSE можно сейчас не выбирать.

12) Какие типы лицензий поддерживаются GitHub при создании репозитория? a) Лицензия Apache 2.0; b) MIT License; c) Публичная лицензия Eclipse 2.0; d) GNU Affero General Public License 2.0; И многие другие.

13) Как осуществляется клонирование репозитория GitHub? Зачем нужно клонировать репозиторий? Для этого на странице репозитория необходимо найти кнопку Clone или Code и щелкнуть по ней, чтобы отобразить адрес репозитория для клонирования. Откройте командную строку или терминал и перейдите в каталог, куда вы хотите скопировать хранилище. Затем напишите git clone и введите скопированный адрес.

14) Как проверить состояние локального репозитория Git? Проверить состояние локального репозитория можно с помощью команды git status.

15) Как изменяется состояние локального репозитория Git после выполнения следующих операций: добавления/изменения файла в локальный репозиторий Git; добавления нового/ измененного файла под версионный контроль с помощью команды git add ; фиксации (коммита) изменений с помощью команды git commit и отправки изменений на сервер с помощью команды git push ? При добавлении/изменении файла в локальных

репозиторий Git состояние локального реозитория измениться на modified – измененное. При добавлении нового/изменного файала под версионный контроль состояние локального репозитория измениться на staged – подготовленное. При фиксации и отправки изменений на сервер состояние перейдет в commited – зафиксированное.

16) У Вас имеется репозиторий на GitHub и два рабочих компьютера, с помощью которых Вы можете осуществлять работу над некоторым проектом с использованием этого репозитория. Опишите последовательность команд, с помощью которых оба локальных репозитория, связанных с репозиторием GitHub будут находиться в синхронизированном состоянии. Примечание: описание необходимо начать с команды git clone . Для получения обновлений с удаленного репозитория можно воспользоваться командой: git pull. Если вы изменили ваши локальные файлы, то команда git pull выдаст ошибку. Если вы уверены, что хотите перезаписать локальные файлы, файлами из удаленного репозитория то выполните команды: git fetch --all git reset --hard github/master

17) GitHub является не единственным сервисом, работающим с Git. Какие сервисы еще Вам известны? Приведите сравнительный анализ одного из таких сервисов с GitHub. Сервисы работающие с Git: a) Fork; b) Tower; c) Sourcetree; d) SmartGit; e) GitKraken. Сравню сервис Fork с GitHub. В фокусе этого инструмента скорость, дружественность к пользователю и эффективность. К особенностям Fork можно отнести красивый вид, кнопки быстрого доступа, встроенную систему разрешения конфликтов слияния, менеджер репозитория. Основная его черта – скорость и простота для пользователя.

18) Интерфейс командной строки является не единственным и далеко не самым удобным способом работы с Git. Какие Вам известны программные средства с графическим интерфейсом пользователя для работы с Git? Приведите как реализуются описанные в лабораторной работе операции Git с помощью одного из таких программных средств. Существует и другое программное средство с графическим интерфейсом, например, Git GUI –

предназначен для тех, кто не любит командную строку. Для создания локального репозитория: в нашем графическом интерфейсе Git нажмите “Создать новый репозиторий”. Выбрать местоположение, в котором вы хотите сохранить свой репозиторий. Чтобы клонировать репозиторий, нажмите на ссылку “Клонировать существующий репозиторий” в окне Git GUI. Существующий репозиторий - это тот, который уже инициализирован и / или имеет отправленные в него коммиты. Когда мы перемещаем файлы в каталог Git, вы увидите все файлы в окне “Неустановленные изменения”. Это в основном означает, что новые файлы были добавлены, удалены, обновлены и т.д. Когда мы нажимаем кнопку “Этап изменен”, он попытается добавить все новые файлы в индекс Git. Так осуществляются похожие действия в Git GUI, которые были описаны в лабораторной работе.

Выводы: исследовал базовые возможности системы контроля версий Git и веб-сервиса для хостинга IT-проектов GitHub