

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ
ФЕДЕРАЦИИ

Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Основы кроссплатформенного программирования

Отчет по лабораторной работе №1

«Исследование основных возможностей Git и GitHub»

Выполнил студент группы ИТС-б-з-22-1

Потеев Антон Сергеевич

«___» _____ 2023г.

Подпись студента_____

Проверил: Доцент, к.т.н, доцент
кафедры инфокоммуникаций

Воронкин Роман Александрович

Работа защищена с оценкой:

(подпись)

Ставрополь, 2023 г.

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

Ход работы:

1. Изучил теоретический материал работы.
2. Создал общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и выбранный мною язык программирования:

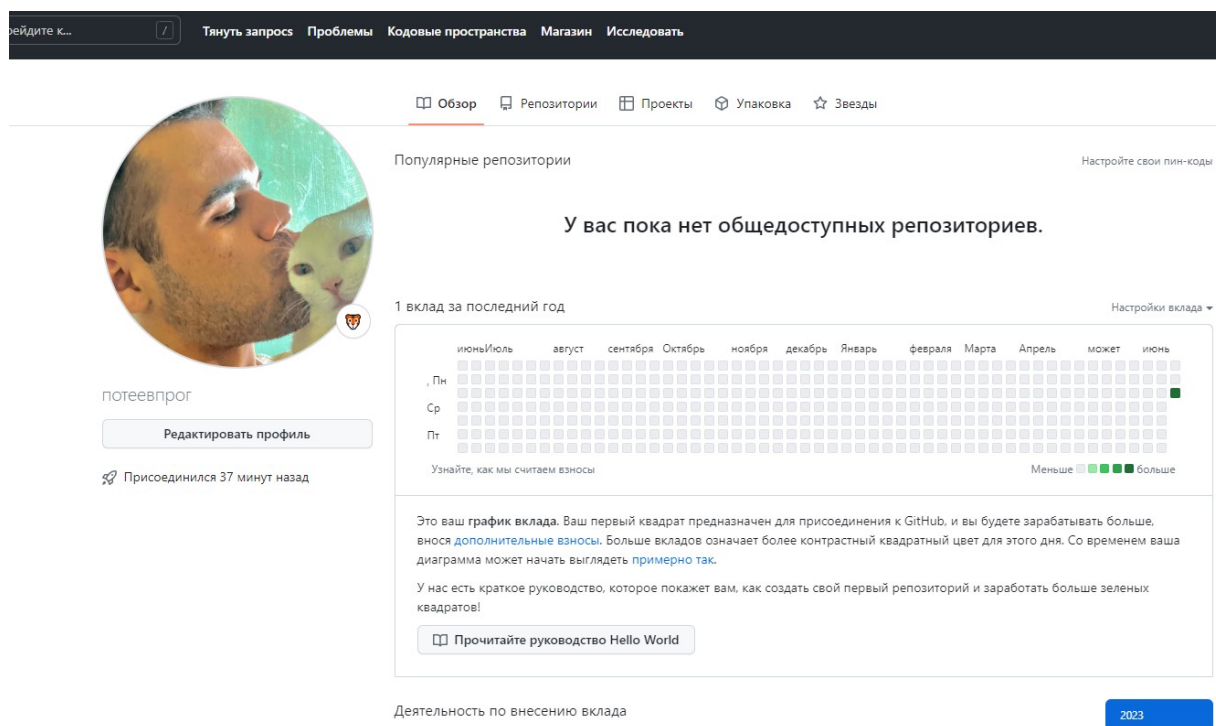


Рис 1. Зарегистрировался на GitHub

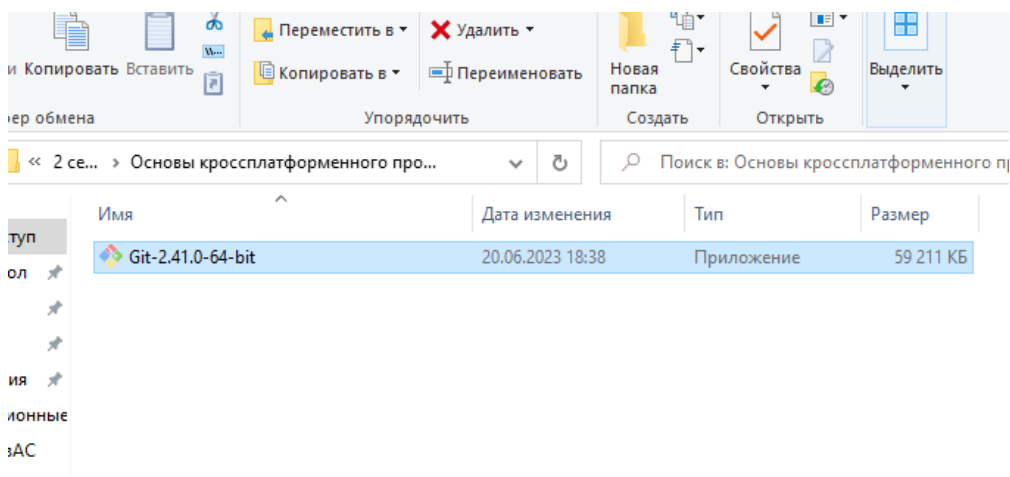


Рис 2. Git bush на моем ПК

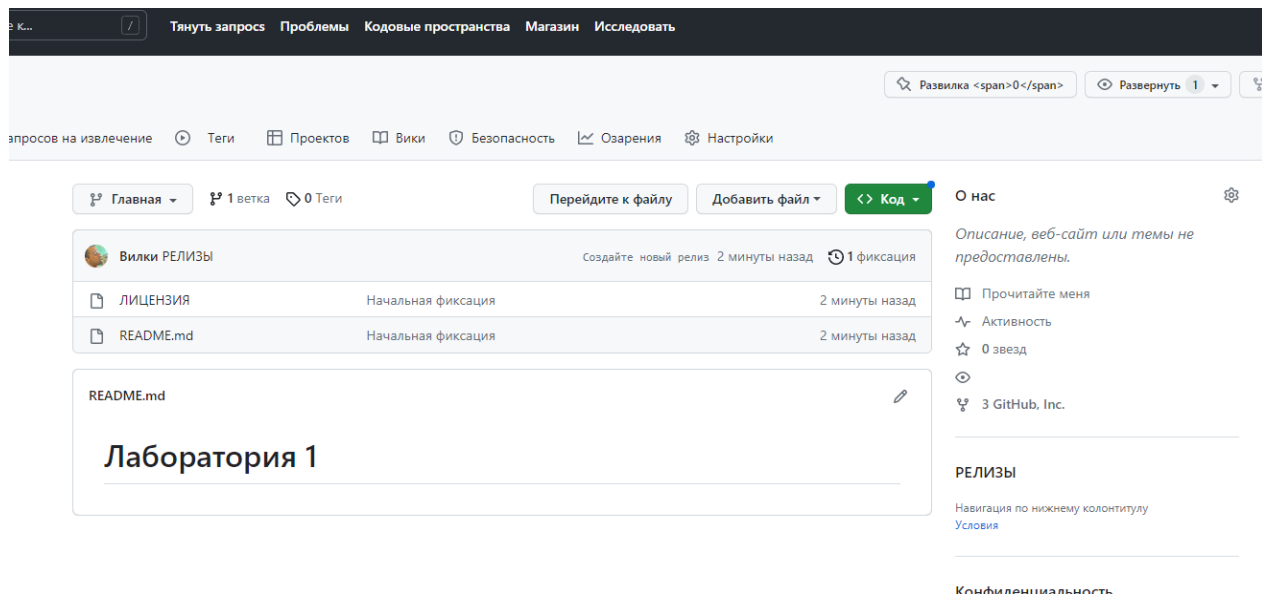


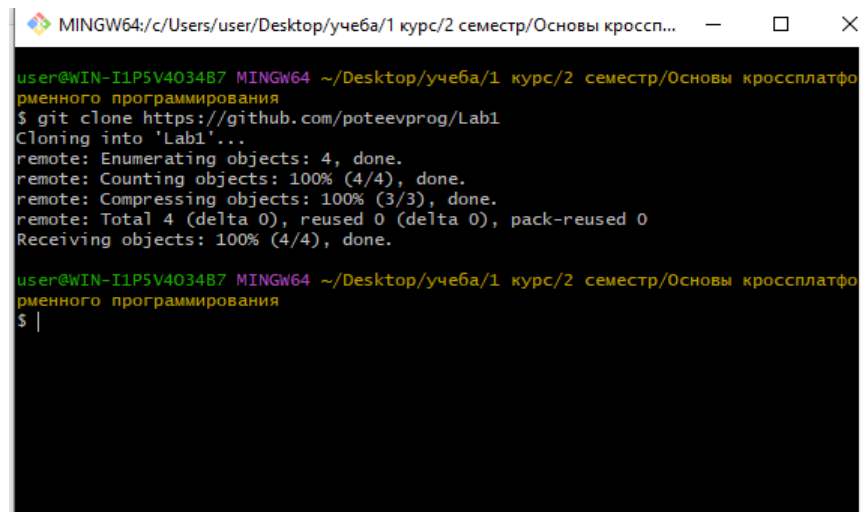
Рис 3. Создал репозиторий

3. Выполнил клонирование созданного репозитория на рабочий компьютер:

```
MINGW64~/Desktop/учеба/1 курс/2 семестр/Основы кроссплатформенного программирования
$ git version
git version 2.41.0.windows.1

MINGW64~/Desktop/учеба/1 курс/2 семестр/Основы кроссплатформенного программирования
$ git config --global user.name poteevprog

MINGW64~/Desktop/учеба/1 курс/2 семестр/Основы кроссплатформенного программирования
$ git config --global user.email poteev.1999@yandex.ru
```

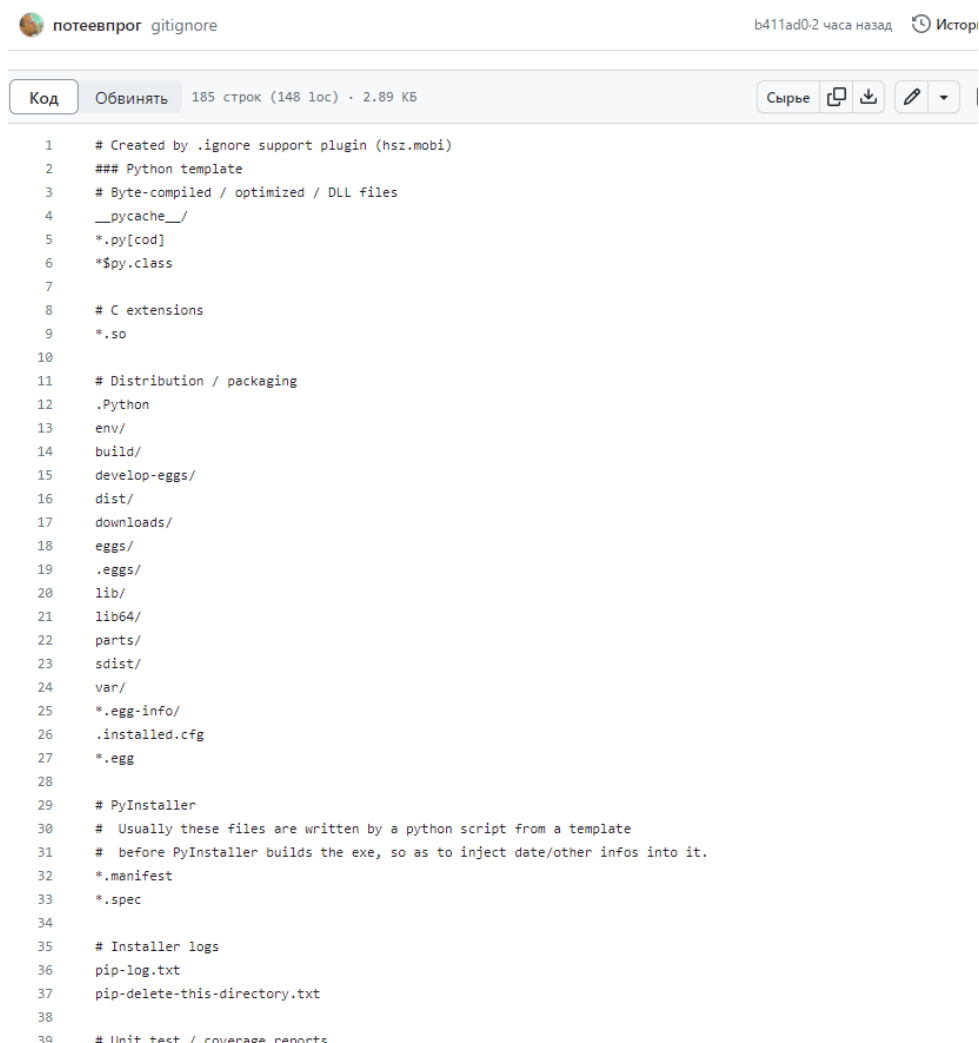


```
MINGW64~/c/Users/user/Desktop/учеба/1 курс/2 семестр/Основы кроссплатформенного программирования
$ git clone https://github.com/poteevprog/Lab1
Cloning into 'Lab1'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.

user@WIN-I1P5V4034B7 MINGW64 ~/Desktop/учеба/1 курс/2 семестр/Основы кроссплатформенного программирования
$ |
```

Рис 4. Клонировал и создал локальное хранилище проекта.

4. Дополнил файл .gitignore необходимыми правилами для выбранного языка программирования и интегрированной среды разработки:



```
пoteевпрог gitignore b411ad0-2 часа назад История

Код Обвинять 185 строк (148 loc) · 2.89 КБ Сырые

1 # Created by .ignore support plugin (hsz.mobi)
2 ### Python template
3 # Byte-compiled / optimized / DLL files
4 __pycache__/
5 *.py[cod]
6 *$py.class
7
8 # C extensions
9 *.so
10
11 # Distribution / packaging
12 .Python
13 env/
14 build/
15 develop-eggs/
16 dist/
17 downloads/
18 eggs/
19 .eggs/
20 lib/
21 lib64/
22 parts/
23 sdist/
24 var/
25 *.egg-info/
26 .installed.cfg
27 *.egg
28
29 # PyInstaller
30 # Usually these files are written by a python script from a template
31 # before PyInstaller builds the exe, so as to inject date/other infos into it.
32 *.manifest
33 *.spec
34
35 # Installer logs
36 pip-log.txt
37 pip-delete-this-directory.txt
38
39 # Unit test / coverage reports
```

Рис 5. Описание игнорируемых файлов и папок.

5. Добавил в файл README.md информацию о группе и ФИО студента, выполняющего лабораторную работу.

```
Lab1 Выполнил студент: Потеев А.С.  
Группа: ИТС-6-э-22-1
```

Рис 6. ФИО, группа.

6. Создал простой калькулятор и сделал 7 коммитов:

```
1 a = int(input("Введите значение a: "))  
2 b = int(input("Введите значение b: "))  
3 c = int(input("Введите значение c: "))  
4 s = a + b + c  
5 print("Сумма трёх чисел: ", s)
```

Рис 7. Калькулятор

Совершает

Главная

Совершается 20 июня 2023 года

прочитайте меня потеевпрог совершен 6 минут назад	3be9e88
прочитайте меня потеевпрог совершен 34 минуты назад	a052be0
прочитайте меня потеевпрог совершен 35 минут назад	fec71bf
Сумма трёх чисел потеевпрог совершен 43 минуты назад	c6d409d
прочитайте меня потеевпрог совершен 45 минут назад	58c4a78
gitignore потеевпрог совершен 2 часа назад	b411ad0
Первоначальная фиксация потеевпрог совершен 2 часа назад	Проверено e744952

Рис 8. 7 коммитов.

7. Добавил файл README и зафиксировал сделанные изменения.

```

user@WIN-E7Q6E2NIJTE MINGW64 /c/git/RUSSKIE-VPERED (main)
$ git add .

user@WIN-E7Q6E2NIJTE MINGW64 /c/git/RUSSKIE-VPERED (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

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

user@WIN-E7Q6E2NIJTE MINGW64 /c/git/RUSSKIE-VPERED (main)
$ git commit -m "commit7"
[main e612911] commit7
 1 file changed, 2 insertions(+), 1 deletion(-)

user@WIN-E7Q6E2NIJTE MINGW64 /c/git/RUSSKIE-VPERED (main)
$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 409 bytes | 409.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/lesnaya1shelupon/RUSSKIE-VPERED.git
   2982140..e612911  main -> main

user@WIN-E7Q6E2NIJTE MINGW64 /c/git/RUSSKIE-VPERED (main)
$

```

Рис 8. Изменение файла README

8. Добавил отчет по лабораторной работе в формате PDF в папку репозитория. Зафиксировал изменения.
9. Отправил изменения в локальном репозитории в удаленный репозиторий
10. Проконтролировал изменения, произошедшие в репозитории GitHub.

Ответы на контрольные вопросы:

- 1) 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. Программисты обычно помещают в систему контроля версий исходные коды программ, но на самом деле под версионный контроль можно поместить файлы практически любого типа.
- 2) Недостатки

1. Отслеживает изменения только отдельных файлов, что не позволяет использовать ее для управления версиями больших проектов.

2. Не позволяет одновременно вносить изменения в один и тот же файл несколькими пользователями.

3. Низкая функциональность, по сравнению с современными системами контроля версий.

3) Git – это разновидность VCS (Version Control System). А VCS – это программа для работы с постоянно изменяющейся информацией. Такое ПО может хранить множество версий одного и того же файла (документа) и возвращаться к более раннему состоянию (версии).

4) 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. Эти системы (CVS, Subversion, Perforce, Bazaar и т. д.) представляют хранимую информацию в виде набора файлов и изменений, сделанных в каждом файле, по времени (обычно это называют контролем версий, основанным на различиях

5) подход Git к хранению данных больше похож на набор снимков миниатюрной файловой системы. Каждый раз, когда вы делаете коммит, то есть сохраняете состояние своего проекта в Git, система запоминает, как выглядит каждый файл в этот момент, и сохраняет ссылку на этот снимок. Для увеличения эффективности, если файлы не были изменены, Git не запоминает эти файлы вновь, а только создаёт ссылку на предыдущую версию идентичного файла, который уже сохранён. Git представляет свои данные как, скажем, поток снимков

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

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

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

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

7) В профиле пользователя отображаем информацию о нем. Вы можете открыть свой профиль и внести изменения (например, вы можете добавить биографию или установить изображение). Создание аккаунта на GitHub ничем

не отличается от создания аккаунтов на других сервисах подобного рода.

8) GitHub содержит миллионы проектов, написанных на разных языках программирования. Каждый проект размещается в своем собственном контейнере, который называется репозиторием. В нем можно хранить код, конфигурации, наборы данных, изображения и другие файлы, включенные в ваш проект. Любые изменения файлов в репозитории будут отслеживаться с помощью контроля версий. Если вы хотите найти какой-то конкретный репозиторий проекта, введите его имя или часть имени в поле поиска. Вы увидите список подходящих репозиториях

9) Стандартный подход к работе с проектом состоит в том, чтобы иметь локальную копию репозитория и фиксировать ваши изменения в этой копии, а не в удаленном репозитории, размещенном на GitHub. Этот локальный репозиторий имеет полную историю версий проекта, которая может быть полезна при разработке без подключения к интернету. После того, как вы что-то изменили в локальном, вы можете отправить свои изменения в удаленный репозиторий, чтобы сделать их видимыми для других разработчиков.

10) После установки GitHub нужно убедиться, что Git был успешно установлен, введите команду ниже в терминале, чтобы отобразить текущую версию вашего Git: (git version)

Если она сработала, то нужно добавить в настройки Git ваше имя, фамилию и адрес электронной почты, связанный с вашей учетной записью GitHub: git config

```
--global user.name  
<YOUR_NAME> git config --global  
user.email <EMAIL>
```

11) В правом верхнем углу, рядом с аватаром есть кнопка с плюсиком, нажимая которую мы переходим к созданию нового репозитория. В результате будет выполнен переход на страницу создания репозитория. Наиболее важными на ней являются следующие поля:

Имя репозитория. Оно может быть любое, необязательно уникальное во всем github, потому что привязано к вашему аккаунту, но уникальное в рамках тех репозиториях, которые вы создавали.

Описание (Description). Можно оставить пустым.

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).

.gitignore и LICENSE можно сейчас не выбирать.

После заполнения этих полей нажимаем кнопку Create repository.

Отлично, ваш репозиторий готов!

12) Сейчас, **при создании** нового **репозитория** вы можете указать параметры файла .gitignore и инициировать **создание** файла README.

Кроме этого, **GitHub** предлагает вам выбрать **тип лицензии** вашего кода, такие как Apache, GPL, MIT и другие, не применяя **лицензию LGPL** автоматически, как это происходило раньше.

13) После создания репозитория его необходимо клонировать на ваш компьютер. 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 адрес:

```
git clone https://github.com/kushedow/flask-html
```

14) Перейдите в каталог хранилища и посмотрите на содержимое. Локальный репозиторий включает в себя все те же файлы, ветки и историю коммитов, как и удаленный репозиторий. Введите эту команду, чтобы проверить состояние вашего репозитория: `git status`

15) Следующие четыре команды предназначены для копирования файлов между рабочей директорией, сценой, также известной как «индекс», и историей, представленной в форме коммитов.

1. `git add` файлы копирует файлы в их текущем состоянии на сцену;
2. `git commit` сохраняет снимок сцены в виде коммита;
3. `git reset` — файлы восстанавливает файлы на сцене, а именно копирует файлы из последнего коммита на сцену. Используйте эту команду для отмены изменений, внесённых командой `git add` файлы. Вы также можете выполнить `git reset`, чтобы восстановить все файлы на сцене;
4. `git checkout` — файлы копирует файлы со сцены в рабочую директорию. Эту команду удобно использовать, чтобы сбросить нежелательные изменения в рабочей директории.

16) С одним репозиторием с разных компьютеров может работать несколько разработчиков или вы сами, если например работаете над одним и тем же проектом дома и на работе.

Для получения обновлений с удаленного репозитория воспользуйтесь 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
```

Вместо `github` подставьте название вашего удаленного репозитория, которое вы зарегистрировали командой `git push -u`.

Как мы уже знаем, для того чтобы изменения выложить на удаленный репозиторий используется команда: `git push` случае, если в удаленном репозитории лежат файлы с версией более новой, чем у вас в локальном, то команда `git push` выдаст ошибку. Если вы уверены, что хотите перезаписать файлы в удаленном репозитории несмотря на конфликт версий, то воспользуйтесь командой: `git push -f`

Иногда возникает необходимость отложить ваши текущие изменения и поработать над файлами, которые находятся в удаленном репозитории. Для этого отложите текущие изменения командой: `git stash`

После выполнения этой команды ваша локальная директория будет содержать файлы такие же, как и при последнем коммите. Вы можете загрузить новые файлы из удаленного репозитория командой `git pull` и после этого вернуть ваши изменения которые вы отложили командой: `git stash pop`

17) Bitbucket — платформа, разработанная компанией Atlassian, которой также принадлежат инструменты для командной работы Jira, Trello и Confluence. Сервис интегрирован с этими и другими проектами — например, инструментами развёртывания приложений AWS CodeDeploy и

Deploy to Azure, сервисом для поиска ошибок Instabug, средами разработки Visual Studio и Unity и прочими.

18) Tower это платный графический интерфейс Git для macOS и Windows. В настоящее время это один из ведущих профессиональных инструментов подобного типа. С его помощью вы сможете лучше

познакомиться с системой контроля версий. Вам будут доступны в визуальном представлении все действия, которые можно совершать в Git. Сюда входит и разрешение конфликтов слияния, и совместная работа над проектами. Есть бесплатный пробный период.

Вывод: я исследовал основные базовые возможности системы контролей версий Git и веб сервис-хостинга для IT- проектов GitHub.