Министерство науки и высшего образования РФ
Федеральное государственное автономное образовательное
учреждение высшего образования
«Национальный исследовательский технологический университет
«МИСиС»

Институт Информационных технологий и компьютерных наук (ИТКН)
Кафедра Автоматизированных Систем Управления (АСУ)

ОТЧЁТ ПО УЧЕБНОЙ ПРАКТИКЕ

По теме: “Домашняя работа №2”

Выполнил:
студент группы БИВТ-20-1
Янушка А.В.
Проверил:
доцент кафедры АСУ
Громов С.В.

МОСКВА 2022

Задачи работы:

1. Введя не более трех команд в командной строке, обеспечить запуск тетрадки из репозитория в таком виде, который может показать обычный браузер без всяких серверов.
2. Объединиться в группы по три человека и убедиться, что тетрадь каждого члена команды автоматически запускается и выполняется в .html.

Используемые технологии:

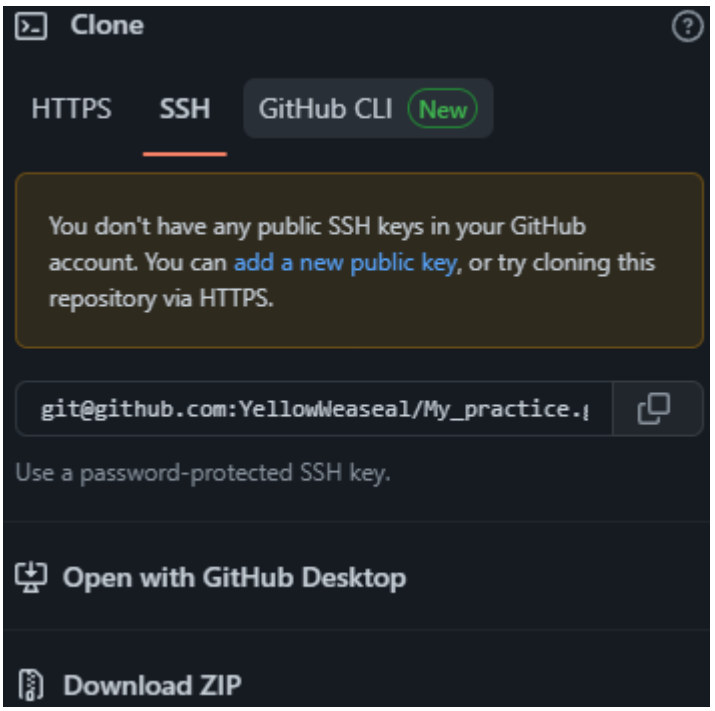
Git Bash - это приложение для сред Microsoft Windows, эмулирующее работу командной строки Git. Bash - это аббревиатура от Bourne Again Shell. Оболочка (Shell) представляет собой приложение терминала для взаимодействия с операционной системой с помощью письменных команд. Bash - популярная оболочка, используемая по умолчанию в Linux и macOS.

Описание решения задач и использованных подходов:

Для начала, разберемся как работать с Git Bush. Для этого рассмотрим все что мы делали раньше, но уже в консоли:

1) Заходим на сайт GitHub и во вкладке “Your repositories” создаем новый репозиторий. Называем его “my_practice”. При создании, можем добавить файл “README”

2) Смотрим ссылку на наш репозиторий, все еще находясь на странице сайта GitHub:



3) Открываем Git Bash и переходим (или создаем) в папку, где, в дальнейшем, будем хранить локальный репозиторий:

```
kilim@DESKTOP-ANH2VAM MINGW64 /c/Лабы/upd
$
```

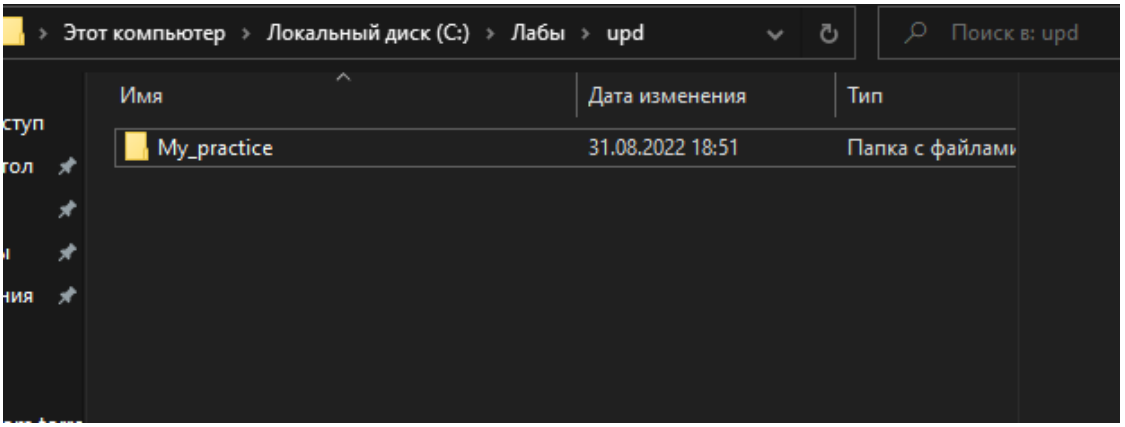
Примечание: Команда "*cd*" (change directory) предназначена для навигации между директориями.

4) Клонируем удаленный репоиторий в локальную папку при помощи точной ссылки на репозиторий:

```
kilim@DESKTOP-ANH2VAM MINGW64 /c/Лабы/upd
$ git clone https://github.com/YellowWeaseal/My_practice.git
Cloning into 'My_practice'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), 9.18 KiB | 9.18 MiB/s, done.
```

Примечание: Команда "*git clone*" используется для создания копии определенного репозитория или ветви в репозитории.

5) Проверяем что проект появился в локальной папке на компьютере:



6) При помощи команды “*cd*” переходим в наш локальный репозиторий для проекта:

```
kilim@DESKTOP-ANH2VAM MINGW64 /c/Лабы/upd
$ cd My_practice/
```

7) Делаем проверку, является ли это расположение git-репозиторием:

```
kilim@DESKTOP-ANH2VAM MINGW64 /c/Лабы/upd/My_practice (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
```

Примечание: Команда “*git status*” показывает состояние рабочего дерева. Так как появляется информация о “статусе”, значит это git-репозиторий.
Примечание: В новых версиях стандартная (первая) ветка master названа main, не будем это менять.
Из данных, видно что сейчас мы на ветке main, что нечего коммитить и рабочее дерево сейчас чисто.

8) Добавляем проект из “Домашки 1” в наш репозиторий и проверяем:

```
kilim@DESKTOP-ANH2VAM MINGW64 /c/Лабы/upd/My_practice (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
```

Видно, что найден новый файл.

9) При помощи команды “*git add*” добавляем изменение из рабочего каталога в раздел проиндексированных файлов и проверяем:

```
kilim@DESKTOP-ANH2VAM MINGW64 /c/Лабы/upd/My_practice (main)
$ git add summer_un_pr.ipynb
warning: LF will be replaced by CRLF in summer_un_pr.ipynb.
The file will have its original line endings in your working directory

kilim@DESKTOP-ANH2VAM MINGW64 /c/Лабы/upd/My_practice (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)
        new file:   summer_un_pr.ipynb
```

Примечание: Команда “*git add*” добавляет содержимое файла в индекс.

10) Коммитим изменения при помощи команды “*git commit*”

```
kilim@DESKTOP-ANH2VAM MINGW64 ~/vv/u_practice (main)
$ git commit -m "workOne"
[main bdff82c] workOne
 1 file changed, 76 insertions(+)
 create mode 100644 summer_un_pr.ipynb
```

Примечание: Команда “*git commit*” Записывает изменения в репозиторий. При помощи флага “*-m*” и последующего описания уточняем что это коммит с работой из 1 работы, а именно, пишем описание “workOne”. По информации видим, что коммит прошел успешно, один файл изменен, 76 строчек добавлено.

11) При помощи команды “*git push*” записываем все изменения локального репозитория в удаленный репозиторий.

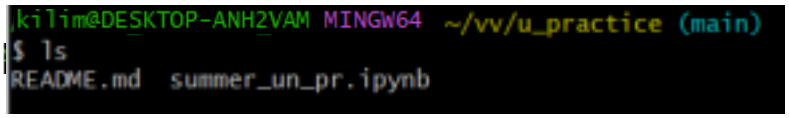
```
kilim@DESKTOP-ANH2VAM MINGW64 /c/Лабы/upd/My_practice (main)
$ git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 6.11 KiB | 6.11 MiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/TheFedor/u_practice.git
   e6f1443..bdff82c  main -> main
```

Примечание: Команда “*git push*” отвечает за обновление удаленных ссылок вместе со связанными объектами.
Теперь все изменения видны и на удаленном репозитории.
Примечание: Указывать *origin* и *main* в данном случае не обязательно, так как мы находимся в папке, куда склонировали репозиторий с GitHub, и в нашем проекте только одна ветка развития и сразу понятно куда отправлять изменения.

Файл добавился, описание к нему именно такое, какое мы и вводили, а значит теперь все изменения сохранены как на локальном, так и на удаленном репозиториях.

Домашняя работа 2:

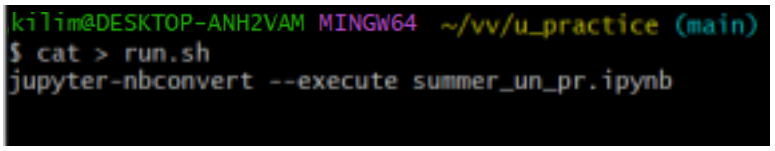
1) В данный момент мы уже находимся в локальном git-репозитории нашего проекта на главной ветке “main”:



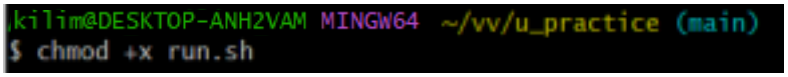
При помощи команды “*ls*” проверяем содержимое нашего локального репозитория: файл “README.md” и файл “sumpr.ipynb”. То есть все файлы, добавленные в “Домашке 1” тут.

Примечание: Команда “*ls*” - просмотр каталога и информации о файлах.

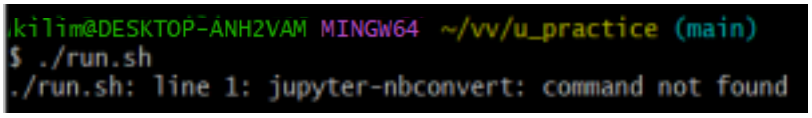
2) Добавляем в наш репозиторий файл “run.sh” при помощи команды “*cat*”,который выполняет .ipynb, делает его читаемым без запуска Jupyter:



3) Далее прописываем:



4) Не забываем добавить команду “source active py3”, чтобы не было такого, что “jupyter-nbconvert” не оказалось в текущем окружении:



5) После можем спокойно открыть наш jupyter проект через Git Bash при помощи команды “*start filename*”:

Заключение:

В ходе этой работы были поставлены задачи, которые были достигнуты. Я узнал что такое Git Bash, разобрал команды для работы с Git, получил практические навыки в написание на Git Bash.