Отчёт по лабораторной работе №2

дисциплина: Операционные системы

Максим Александрович Мишонков

Содержание

Список иллюстраций

1 Цель работы

Целью данной лабораторной работы является изучение идеологии и применения средств системы контроля версий Git, а также приобретение практических навыков работы с ней.

2 Задание

Научиться работать с системой контроля версий Git.

3 Теоретическое введение

Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом. Обычно основное дерево проекта хранится в локальном или удалённом репозитории, к которому настроен доступ для участников проекта. При внесении изменений в содержание проекта система контроля версий позволяет их фиксировать, совмещать изменения, произведённые разными участниками проекта, производить откат к любой более ранней версии проекта, если это требуется.

Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов, которые могут возникнуть при работе нескольких человек над одним файлом.

4 Выполнение лабораторной работы

1. Создал каталог “Операционные системы”. (рис. [4.1], [4.2])

Рис. 4.1: Создание каталога

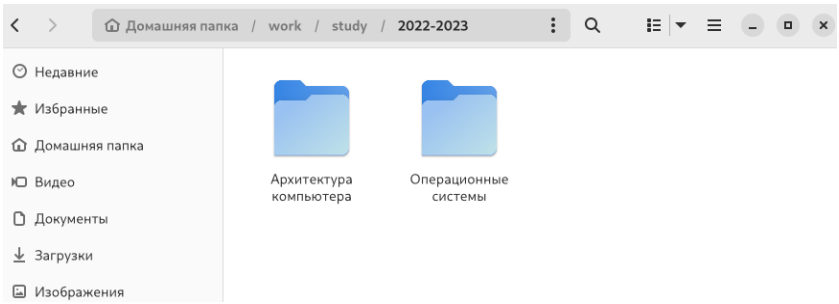

Рис. 4.2: Каталог “Операционные системы”

2. Перешёл в каталог “Операционные системы” и начал процесс создания репозитория на GitHub. (рис. [4.3], [4.4], [4.5], [4.6], [4.7], [4.8], [4.9])

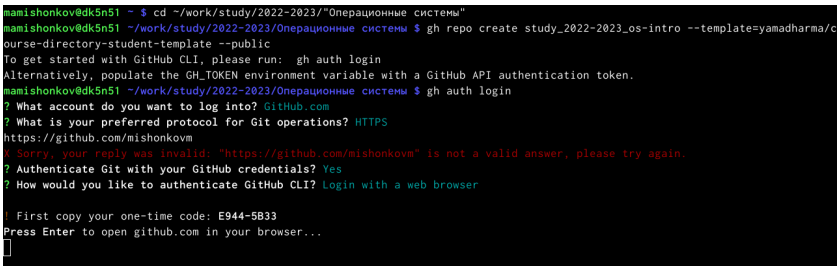

Рис. 4.3: Создание репозитория

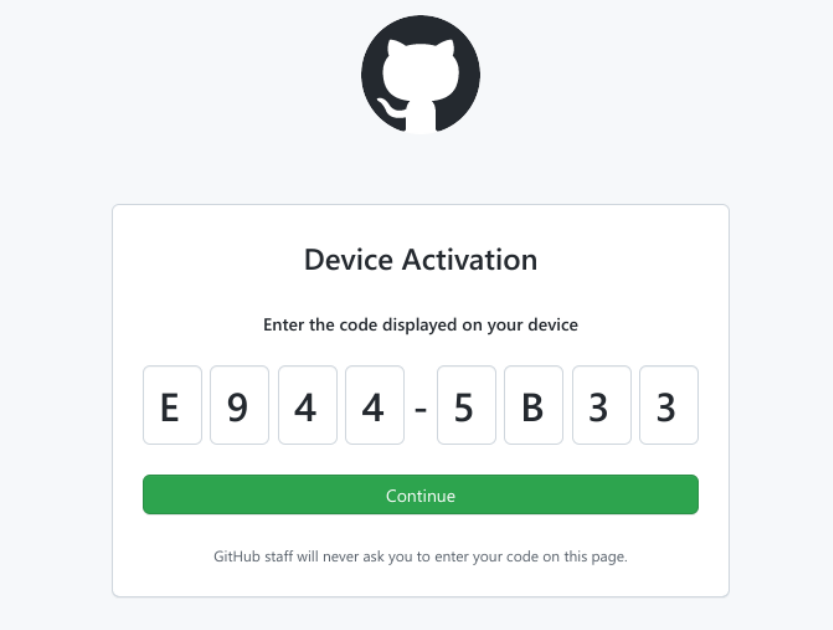

Рис. 4.4: Создание репозитория

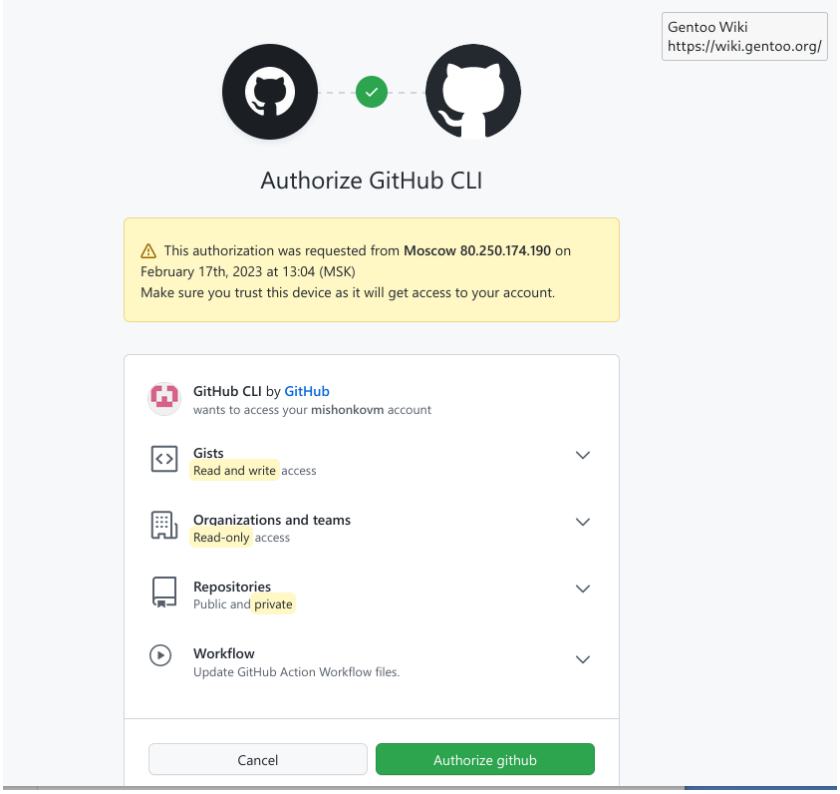

Рис. 4.5: Создание репозитория

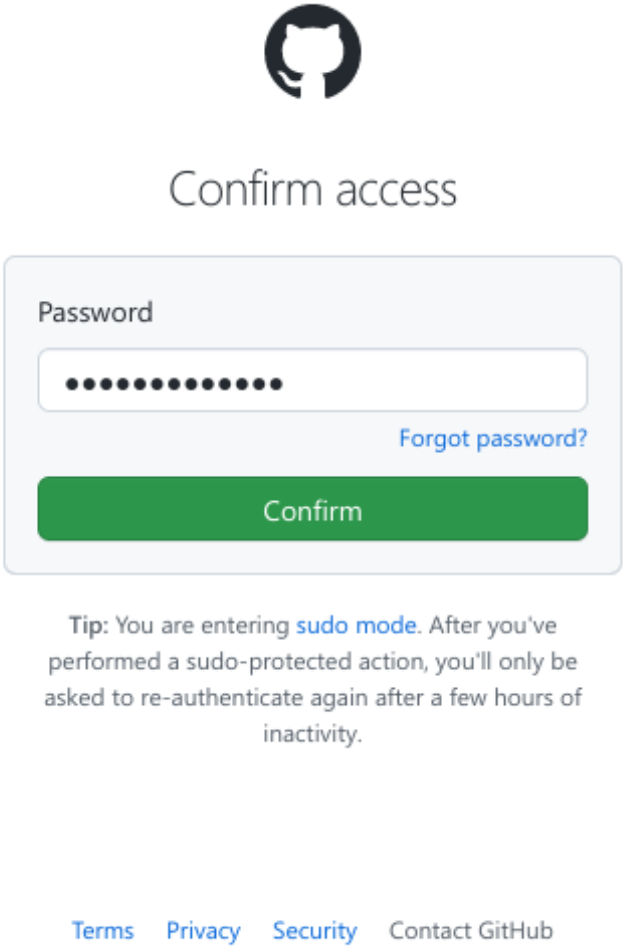

Рис. 4.6: Создание репозитория

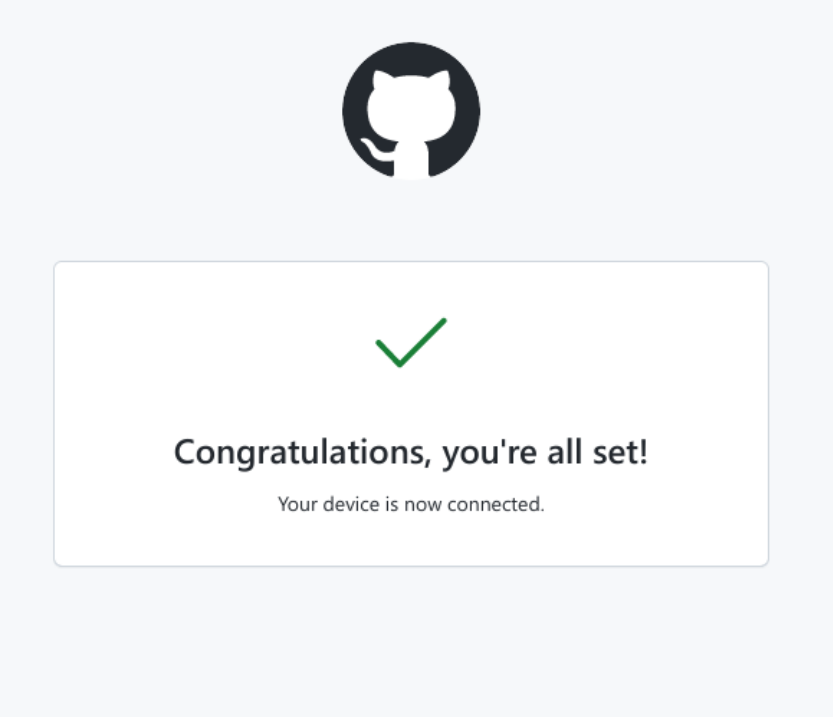

Рис. 4.7: Создание репозитория

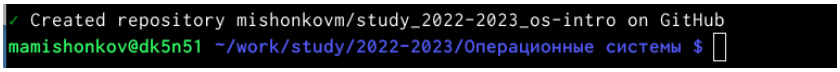

Рис. 4.8: Создание репозитория

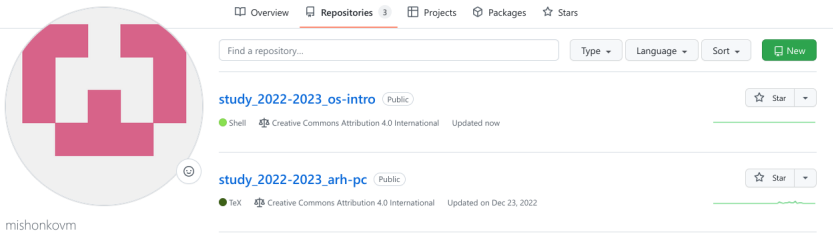

Рис. 4.9: Созданный репозиторий

3. Клонировал созданный репозиторий. (рис. [4.10])

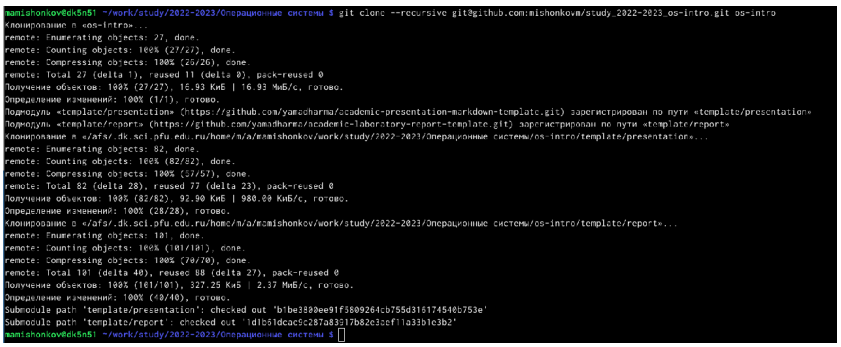

Рис. 4.10: Клонирование созданного репозитория

4. Удалил лишние файлы и создал необходимые каталоги. (рис. [4.11], [4.12])

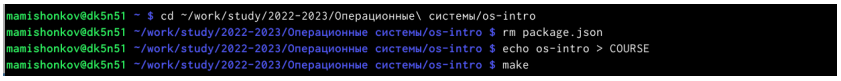

Рис. 4.11: Удаление лишних файлов и создание необходимых каталогов

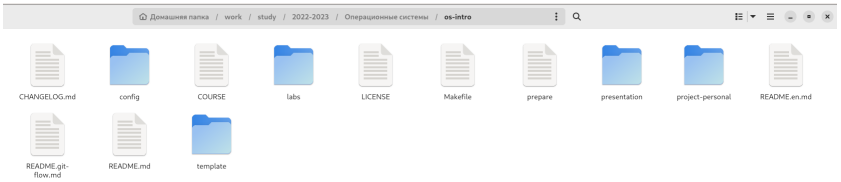

Рис. 4.12: Созданные каталоги

5. Отправил файлы на сервер. (рис. [4.13], [4.14], [4.15], [4.16], [4.17])

Рис. 4.13: Отправка файлов на сервер

Рис. 4.14: Отправка файлов на сервер

```
[master 87d1a16] feat(main): make course structure
 361 files changed, 100327 insertions(+), 14 deletions(-)
 create mode 100644 labs/README.md
 create mode 100644 labs/README.ru.md
 create mode 100644 labs/lab01/presentation/Makefile
 create mode 100644 labs/lab01/presentation/image/kulyabov.jpg
 create mode 100644 labs/lab01/presentation/presentation.md
 create mode 100644 labs/lab01/report/Makefile
 create mode 100644 labs/lab01/report/bib/cite.bib
 create mode 100644 labs/lab01/report/image/placeimg_800_600_tech.jpg
 create mode 100644 labs/lab01/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
 create mode 100755 labs/lab01/report/pandoc/filters/pandoc_eqnos.py
 create mode 100755 labs/lab01/report/pandoc/filters/pandoc_fignos.py
 create mode 100755 labs/lab01/report/pandoc/filters/pandoc_secnos.py
 create mode 100755 labs/lab01/report/pandoc/filters/pandoc_tablenos.py
 create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/__init__.py
 create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/core.py
 create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/main.py
 create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/pandocattributes.py
 create mode 100644 labs/lab01/report/report.md
```

Рис. 4.15: Отправка файлов на сервер

```
mamishonkov@dk5n51 ~/work/study/2022-2023/Операционные системы/os-intro $ git push
Перечисление объектов: 40, готово.
Подсчет объектов: 100% (40/40), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (30/30), готово.
Запись объектов: 100% (38/38), 342.39 КиБ | 3.32 МиБ/с, готово.
Всего 38 (изменений 4), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:mishonkovm/study_2022-2023_os-intro.git
   66e26d7..87d1a16  master -> master
mamishonkov@dk5n51 ~/work/study/2022-2023/Операционные системы/os-intro $ git commit -am 'feat(main): make course structure'
```

Рис. 4.16: Отправка файлов на сервер

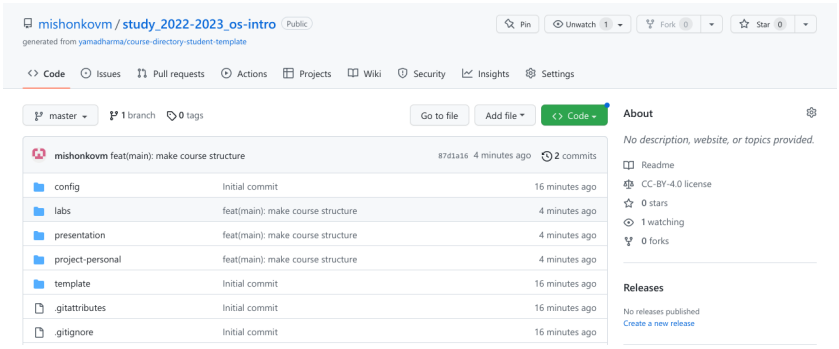

Рис. 4.17: Отправленные на сервер файлы

Ответы на контрольные вопросы

1. Система управления версиями (Version Control System, VCS) — программное обеспечение для облегчения работы с изменяющейся информацией. Система управления версиями позволяет хранить несколько версий одного

и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение, и многое другое.

2. В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером.Участник проекта (пользователь) перед началом работы посредством определённыхкоманд получает нужную ему версию файлов. После внесения изменений, пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляютсяиз центрального хранилища и к ним можно вернуться в любой момент. Сервер может сохранять неполную версию изменённых файлов, а производить так называемую дельта-компрессию—сохранять только изменения между последовательными версиями,чтопозволяет уменьшить объём хранимых данных. Системы контроля версий также могут обеспечивать дополнительные, более гибкие функциональные возможности. Например,они могут поддерживать работу с несколь-ми версиями одного файла,сохраняя общую историю изменений до точки ветвления версий и собственные истории изменений каждой ветви. Крометого, обычно доступна информация о том, кто из участников, когда и какие изменения вносил. Обычно такого рода информация хранится в журнале изменений, доступ к которому можно ограничить.

3. Централизованные системы — это системы, которые используют архитектуру клиент / сервер, где один или несколько клиентских узлов напрямую подключены к центральному серверу. Пример - Wikipedia. В децентрализованных системах каждый узел принимает свое собственное решение. Конечное поведение системы является совокупностью решений отдельных узлов. Пример — Bitcoin. В классических системах контроля версий

используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов.Выполнение большинства функций по управлению версиями осуществляется специальным сервером.

4. Создадим локальный репозиторий. Сначала сделаем предварительную конфигурацию, указав имя и email владельца репозитория: git config –global user.name”Имя Фамилия” git config –global user.email”work@mail” и настроив utf-8 в выводе сообщенийgit: git config –global quotepath false Для инициализации локального репозитория, расположенного, например, в каталоге ~/tutorial, необходимо ввести в командной строке: cd mkdir tutorial cd tutorial git init

5. Для последующей идентификации пользователя на сервере репозиториев необходимо сгенерировать пару ключей (приватный и открытый): ssh-keygen -C”Имя Фамилия work@mail“. Ключи сохраняться в каталоге~/.ssh/. Скопировав из локальной консоли ключ в буфер обмена cat ~/.ssh/id_rsa.pub | xclip -sel clip вставляем ключ в появившееся на сайте поле.

6. У Git две основных задачи: первая — хранить информацию о всех изменениях в вашем коде, начиная с самой первой строчки, а вторая — обеспечение удобства командной работы над кодом.

7. Основные команды git: Наиболее часто используемые команды git: – создание основного дерева репозитория:git init–получение обновлений (изменений)текущего дерева из центрального репозитория:git pull–отправка всех произведённых изменений локального дерева в центральный репозиторий:git push–просмотр списка изменённых файлов втекущей директории:git status–просмотртекущих изменения:git diff–сохранениетекущих изменений:–добавить все изменённые и/или созданные файлы и/или каталоги:git add .–добавить конкретные изменённые и/или созданные файлы и/или каталоги:git add имена_файлов – удалить файл и/или каталог из индекса репозитория (приэтомфайл и/илик аталог остаётся в локальной

директории): git rm имена_файлов – сохранение добавленных изменений: – сохранить все добавленные изменения и все изменённые файлы: git commit -am ‘Описание коммита’–сохранить добавленные изменения с внесением комментария через встроенный редактор:git commit–создание новой ветки, базирующейся натекущей: git checkout -b имя_ветки–переключение на некоторую ветку: git checkout имя_ветки (при переключении на ветку, которой ещё нет в локальном репозитории, она будет создана и связана с удалённой) – отправка изменений конкретной ветки в центральный репозиторий: git push origin имя_ветки–слияние ветки стекущим деревом:git merge –no-ff имя_ветки–удаление ветки: – удаление локальной уже слитой с основным деревом ветки:git branch -d имя_ветки–принудительное удаление локальной ветки:git branch -D имя_ветки–удаление ветки с центрального репозитория: git push origin :имя_ветки

8. Использования git при работе с локальными репозиториями (добавления текстового документа в локальный репозиторий): git add hello.txt git commit -am’Новый файл

9. Проблемы, которые решают ветки git:

- нужно постоянно создавать архивы с рабочим кодом
- сложно “переключаться” между архивами
- сложно перетаскивать изменения между архивами
- легко что-то напутать или потерять

10. Во время работы над проектомтак или иначе могутсоздаваться файлы,которые нетребуется добавлять в последствии в репозиторий. Например, временные файлы, со-здаваемые редакторами,или объектные файлы, создаваемые компиляторами.Можно прописать шаблоны игнорируемых при добавлении в репозиторийтипов файлов в файл.gitignore с помощьюс ервисов. Для этого сначала нужно получить списоки меющихся шаблонов: curl -L -s https://www.gitignore.io/api/list Затем скачать шаблон,например,

для C и C++ curl -L -s https://www.gitignore.io/api/c » .gitignore curl -L -s https://www.gitignore.io/api/c++ » .gitignore

5 Выводы

В ходе выполнения данной лабораторной работы я создал каталог “Операционные системы” и новый репозиторий на сервере GitHub на основе шаблона, внёс в него изменения. Также я создал рабочее пространство для выполнения следующих лабораторных работ.