Отчет по лабораторной работе №2

Дисциплина: Архитектура компьютера

Максимова Дарья Валерьевна

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Целью данной работы является изумление идеологии и применение средств контроля версий, а также приобрести практические навыки работы с системой git.

2 Задание

1. Настройка GitHub.
2. Базовая настройка Git.
3. Создание SSH-ключа.
4. Создание рабочего пространства и репозитория курса на основе шаблона.
5. Создание репозитория курса на основе шаблона.
6. Настройка каталога курса.
7. Выполнение заданий для самостоятельной работы.

3 Теоретическое введение

Системы управления версией (Система управления версией, VCS) применяются при работе нескольких человек над одним проектом. Обычно основное дерево проекта хранится в локальном или удаленном репозитории, к которому настроен доступ для участников проекта. При внесении изменений в содержание проекта система контроля версий позволяет их фиксировать, совмещать изменения, произведенные разными участниками проекта, производить откат к любой более ранней версии проекта, если это требуется. В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций управления версиями осуществляется специальным сервером. Участник проекта (пользователь) перед началом работы с опреем опредедных команд получает нужную ему версию файлов. После внесения изменений пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются из центрального хранилища и к ним можно вернуться в любой момент. Сервер может сохранять не полную версию измененных файлов, а производить так называемое дельта-сжатие — сохранять только изменения между последовательными версиями, что позволяет уменьшить объем хранимых данных. Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов, которые могут возникнуть при работе нескольких человек над одним файлом. Можно объединить изменения, сделанные разными участниками, вручную выбрать нужную версию, отменить изменения или заблокировать файлы для изменения. В зависимости от настро-

ек блокировка не позволяет другим пользователям получить рабочую копию или препятствует изменению рабочей копии файла с средствами файловой системы ОС, обеспечивая таким образом привилегированный доступ только одному пользователю, работающему с файлом. Системы управления версиями также могут обеспечивать дополнительные, более гибкие функциональные возможности. Например, они могут поддерживать работу с несколькими версиями одного файла, сохраняя общую историю изменений до точки ветвления версий и собственные истории изменений каждой ветви. Обычно доступна информация о том, кто из участников, когда и какие изменения вносили. Обычно такая информация хранится в журнале изменений, доступ к которому можно ограничить. В отличие от классических, в распределённых системах управления версий центральный репозиторий не является обязательным. Среди классических VCS наиболее известны CVS, Subversion, а среди распределенных — Git, Bazaar, Mercurial. Принципы их работы похожи, они отличаются в основном синтаксисом используемых в работе команд. Система управления вериансиями Git представляет собой набор программ командной строки. Доступ к ним можно получить из терминала, введя git-колама с различными опциями. Благодаря тому, что Git является распределенной системой контроля версий, резервную копию локального хранилища можно сделать простым копированием или архивацией. Работа пользователя со своей веткой начинается с проверки и получения изменений из центрального репозитория (при этом в локальное дерево до начала этой процедуры не должно быть внесено изменений). Затем можно внести изменения в локальное дерево и/или ветку. После внесения каких-либоких-лименей в файлы и/или каталоги проекта необходимо разместить их в центральном репозитории.

4 Выполнение лабораторной работы

Создаю учетную запись на сайте https://github.com/ и заполняю основные данные. (рис. [-fig:001])

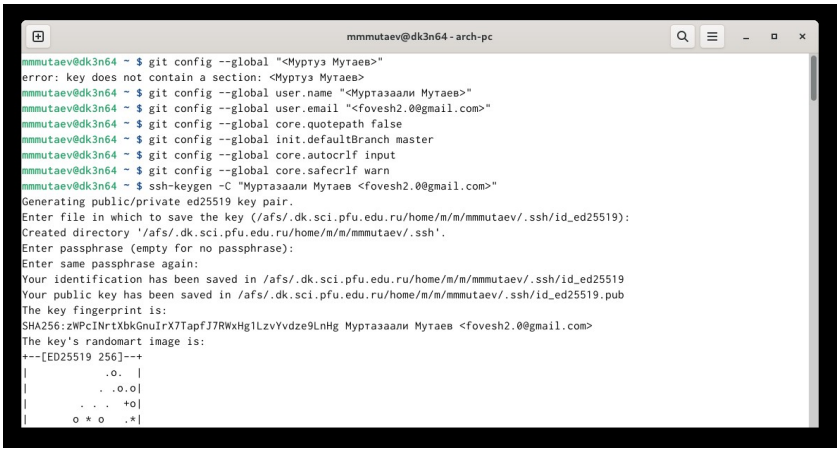

Рис. 4.1: рис. 1 Создание учетной записи на github

Делаю предварительную конфигурацию git, указав имя и email владельца репозитория (рис. [-fig:002])

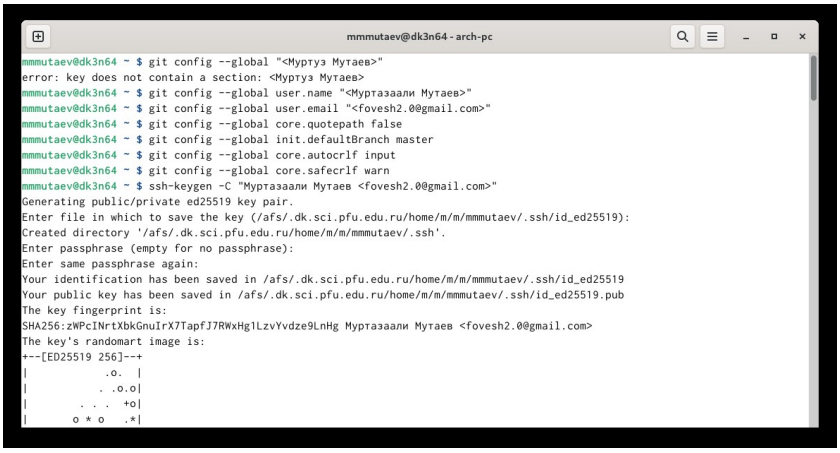

Рис. 4.2: рис. 2 Предварительная конфигурация

Настроим utf-8 в выводе сообщений git, зададим имя начальной ветке(будем называть её master), укажем значение параметров autocrlf и safecrlf (рис. [-fig:003])

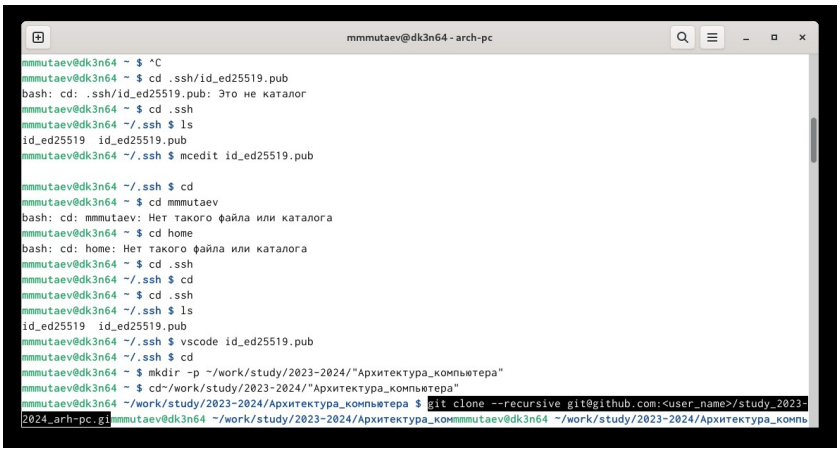

Рис. 4.3: рис. 3 Настройка utf-8

Для последующей идентификации пользователя на сервере репозиториев сгенерируем пару ключей (приватный и открытый) (рис. [-fig:004])

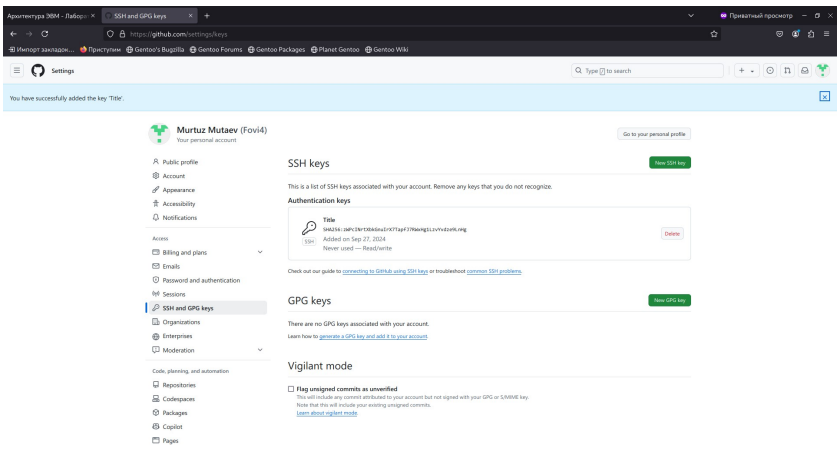

Рис. 4.4: рис. 4

(у меня уже был сгенерирован код) Имеющийся ключ я загрузила на github, загрузив его в буфер обмена. Вставляю скопированный ключ в поле «Ключ». В поле Название указываю имя для ключа. Нажимаю «Добавить SSH-ключ», чтобы завершить добавление ключа. (рис. [-fig:005])

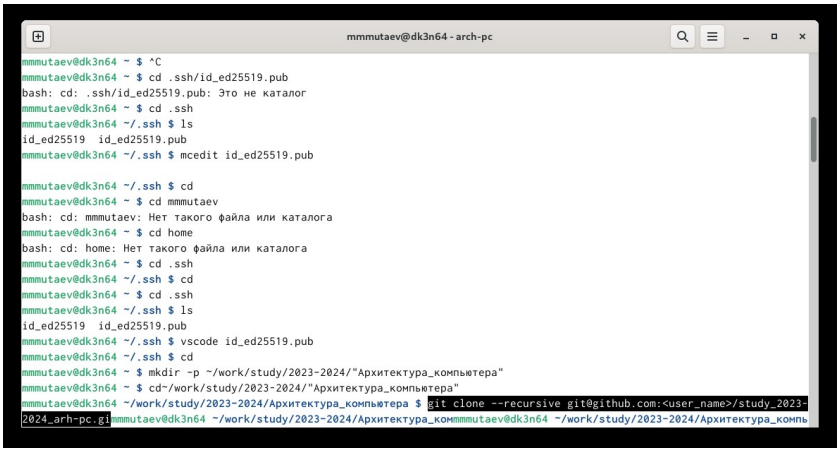

Рис. 4.5: рис. 5

Создадим каталог для предмета «Архитектура компьютера» для последующего создания рабочего пространства. (рис. [-fig:006])

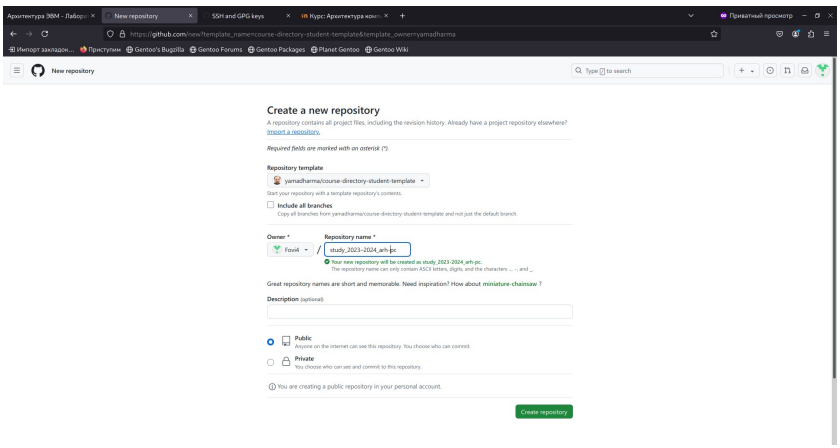

Рис. 4.6: рис. 6

Через web-интерфейс github создадим репозиторий на основе шаблона, указав имя study_2024–2025_arh-pc (рис. [-fig:007])

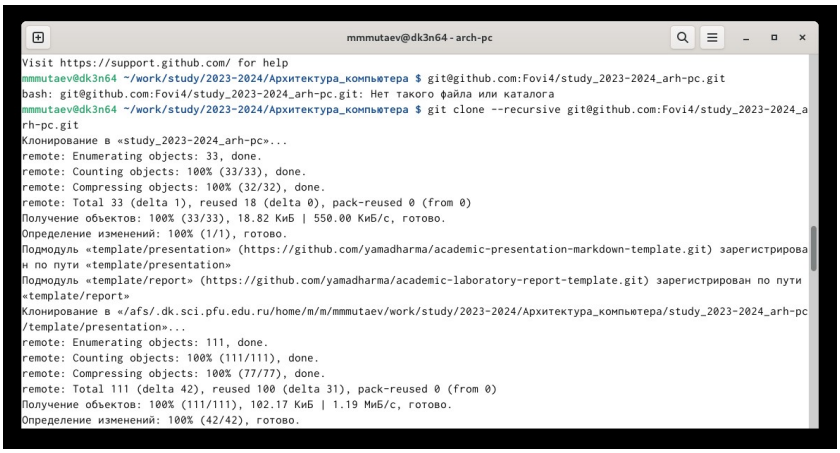

Рис. 4.7: рис. 7

Перейдем в каталог курса и скопируем в него созданный репозиторий с помощью ссылки для клонирования: (рис. [-fig:008])

Рис. 4.8: рис. 8

Перейдём в каталог курса, удалим лишние файлы, создадим нужные каталоги и загрузим файлы на сервер: (рис. [-fig:009])

Рис. 4.9: рис. 9

Создание нужных каталогов: (рис. [-fig:010])

рис. 10

Рис. 4.10: рис. 10

Проверим правильность введённых команд: (рис. [-fig:011])

рис. 11

Рис. 4.11: рис. 11

5 Выводы

В ходе выполнения этой я исследовала концепции и познакомилась с использованием систем контроля версий, а также приобрела практические навыки работы с git.