Лабораторная работа №2

дисциплина: Архитектура компьютера

Гаврилейко Алина Александровна

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Целью работы является изучить идеологию и применение системы контроля версий. Приобрести практические навыки по работе с системой git

2 Задание

На основе методических указаний провести работу с базовыми командами системы контроля версий git, выучить применение команд для разных случаев использования, настроить GitHub.

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. 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 определённых 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 изменённых файлов, а производить так называемую дельта-компрессию — сохранять только изменения между последовательными версиями, что позволяет уменьшить объём хранимых данных. Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов, которые могут возникнуть при работе нескольких человек над одним файлом. Можно объединить (слить) изменения, сделанные разными участниками (автоматически или вручную), вручную выбрать нужную версию, отменить изменения вовсе или заблокировать файлы для изменения. В зави-

симости от настроек блокировка не позволяет другим пользователям получить рабочую копию или препятствует изменению рабочей копии файла средствами файловой системы ОС, обеспечивая таким образом, привилегированный доступ только одному пользователю, работающему с файлом.

4 Выполнение лабораторной работы

4.1 Настройка GitHub

Создаю учетную запись на сайте GitHub, заполняю данные учетной записи и создаю аккаунт (рис. 4.1).

Создание аккаунта на GitHub

Рис. 4.1: Создание аккаунта на GitHub

Открываю виртуальную машину, затем терминал и делаю предварительную конфигурацию Git. Вбиваю команду git config –global user.name ’’, указываю свое имя и фамилию (рис. 4.2).

Предварительная конфигурация Git

Рис. 4.2: Предварительная конфигурация Git

затем пишу команду git config –global user.email ’’, указываю почту владельца. (рис. 4.3).

Предварительная конфигурация Git

Рис. 4.3: Предварительная конфигурация Git

Настраиваю utf-8 в выводе сообщений git для корректного отображения символов (рис. 4.4).

Настройка кодировки

Рис. 4.4: Настройка кодировки

Задаю имя master для начальной ветки (рис. 4.5).

Создание имени для начальной ветки

Рис. 4.5: Создание имени для начальной ветки

Задаю параметр autocrlf со значением input (рис. 4.6).

Параметр autocrlf

Рис. 4.6: Параметр autocrlf

Задаю параметр safecrlf (рис. 4.7).

Параметр safecrlf

Рис. 4.7: Параметр safecrlf

4.2 Создание SSH-ключа

Для идентификации пользователя на сервере репозиториев нужно сгенерировать пару ключей – приватный и открытый. Для этого в терминал ввожу команду ssh-keygen –C ‘Имя Фамилия, work@email’ и ввожу туда имя, фамилию, почту. Ключ автоматически сохранится в каталоге ~/.ssh/. (рис. 4.8).

Создание SSH-ключа

Рис. 4.8: Создание SSH-ключа

Далее на дистрибутив fedora необходимо скачать команду xclip. Устанавливаю ее с помощью команды sudo dnf install xclip (рис. 4.9).

Установка команды xclip

Рис. 4.9: Установка команды xclip

Копирую открытый ключ из директории, в которой он был сохранен, с помощью команды xclip (рис. 4.10).

Копирование содержимого файла

Рис. 4.10: Копирование содержимого файла

Далее захожу на сайт GitGub, открываю свой профиль и выбираю страницу “SSH and GPG keys”. Нажимаю на кнопку “New SSH key”, вставляю в поле скопированный ключ, в поле ‘Title’ указываю имя ключа, затем нажимаю “Add SSH- key”. (рис. 4.11).

Вставка SSH-ключа

Рис. 4.11: Вставка SSH-ключа

4.3 Создание рабочего пространства и репозитория курса на основе шаблона

Открываю терминал, создаю директорию, рабочее пространство с помощью утилиты mkdir, с помощью ключа –p создаю рекурсивно все директории после домашней ~/work/study/2024-2025/”Архитектура компьютера”. (рис. 4.12).

Создание рабочего пространства

Рис. 4.12: Создание рабочего пространства

В браузере перехожу на страницу репозитория с шаблоном курса по адресу https://github.com/yamadharma/course-directory-student-template, выбираю use this template, чтобы использовать данный шаблон для своего репозитория. В открывшемся окне задаю название репозитория: study_2024-2025_arch-pc и создаю репозиторий, нажимая на кнопку “Create repository” (рис. 4.13).

Страница создания репозитория

Рис. 4.13: Страница создания репозитория

Через терминал перехожу в созданный каталог курса с помощью утилиты cd (рис. 4.14).

Переход в каталог курса

Рис. 4.14: Переход в каталог курса

Далее клонирую созданны репозиторий (рис. 4.15).

Клонирование репозитория

Рис. 4.15: Клонирование репозитория

Ссылку для клонирования копирую на странице созданного репозитория, перейдя сначала в окно ‘code’, выбрав затем вкладку ‘SSH’ (рис. 4.16).

Окно с ссылкой на копирование репозитория

Рис. 4.16: Окно с ссылкой на копирование репозитория

4.4 Настройка каталога курса

Перехожу в каталог arch-pc с помощью утилиты cd (рис. 4.17).

Перемещение между директориями

Рис. 4.17: Перемещение между директориями

Удаляю лишние файлы с помощью утилиты rm (рис. 4.18).

Удаление файлов

Рис. 4.18: Удаление файлов

Создаю необходимые каталоги (рис. 4.19).

Создание каталогов

Рис. 4.19: Создание каталогов

Отправляю созданные каталоги с локального репозитория на сервер: добавляю все созданные каталоги с помощью git add, комментирую и сохраняю изменения на сервере как добавление курса с помощью git commit (рис. 4.20).

Добавление и сохранение изменений на сервере

Рис. 4.20: Добавление и сохранение изменений на сервере

Затем все отправляю на сервер с помощью команды git push (рис. 4.21).

Выгрузка изменений на сервер

Рис. 4.21: Выгрузка изменений на сервер

4.5 Задания для самостоятельной работы

Перехожу в директорию labs/lab02/report, создаю в каталоге файл для отчета по второй лабораторной работе с помощью команды touch (рис. 4.22).

Создание файла

Рис. 4.22: Создание файла

Перехожу в подкаталог lab01/report, затем копирую первую лабораторную работу с помощью команды cp, а далее с помощью ls удостоверяюсь, что скопировано все верно (рис. 4.23).

Добавление файла

Рис. 4.23: Добавление файла

С помощью команды git add добавляю в коммит файл «Л01_Гаврилейко_отчет», перехожу в директорию, в которой находится отчет, добавляю нужный файл и сохраняю изменения на сервере с помощью команды git commit –m ’’ и далее с помощью git push –f origin master отправляю изменения в главный репозиторий (рис. 4.24).

Отправка изменений в центральный репозиторий

Рис. 4.24: Отправка изменений в центральный репозиторий

4.6 Вывод

При выполнении данной лабораторной работы я изучила практические навыки по работе с системой Git, изучила применение и идеологию средств контроля версий.

4.7 Список литературы

https://esystem.rudn.ru/pluginfile.php/2089082/mod_resource/content/0/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D0%B0%D1%8F%20%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0%20%E2%84%962.%20%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0%20%D0%BA%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D1%8F%20%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D0%B9%20Git.pdf

5 Выводы

Здесь кратко описываются итоги проделанной работы.

Список литературы