РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2

дисциплина: Архитектура компьютера

Студент: Шилов Александр

Группа: НКАбд-05-24

МОСКВА 2024 г.

Оглавление

1. Цель работы

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

2. Задание

Изучить идеологию и применение средств контроля версий. Приобрести практические навыки по работе с системой 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. 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 зависимости 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. Системы контроля версий 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. Кроме того, 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. В отличие от классических, в распределённых системах контроля версий центральный репозиторий не является обязательным. Среди классических VCS наиболее известны CVS, Subversion, а среди распределённых — Git, Bazaar, Mercurial. Принципы их работы схожи, отличаются они в основном синтаксисом используемых в работе команд.

Команда	Описание
git init	создание основного дерева репозитория
git pull	получение обновлений (изменений) текущего дерева из центрального репозитория
git push	отправка всех произведённых изменений локального дерева в центральный репозиторий
git status	просмотр списка изменённых файлов в текущей директории

Команда	Описание
git diff	просмотр текущих изменений
git add	добавить все изменённые и/или созданные файлы и/или каталоги
git add	добавить конкретные изменённые и/или созданные файлы и/или каталоги
git rm	удалить файл и/или каталог из индекса репозитория (при этом файл и/или каталог остаётся в локальной директории)

Таблица 3.1 Основные команды взаимодействия пользователя с файловой системой

4. Выполнение лабораторной работы

4.1 Настройка Github

Существует несколько доступных серверов репозиториев с возможностью бесплатного размещения данных. Например, http://bitbucket.org/, https://github.com/ и https://gitflic.ru. Для выполнения лабораторных работ я использовал Github. Я создал учётную запись на сайте https://github.com/ и заполнил основные данные.

4.2 Базовая настройка git

Сначала сделаем предварительную конфигурацию git. Откройте терминал и введите следующие команды, указав имя и email владельца репозитория, настроил utf-8 в выводе сообщений git, задал имя начальной ветки, параметр autocrlf, параметр safecrlf: (рис. 4.2.1)

```
oem123@oem123-Medio:~$ git config --global user.name "1132246746"
oem123@oem123-Medio:~$ git config --global user.email "sashandr000chanel000999000@gmail.com"
oem123@oem123-Medio:~$ git config --global core.quotepath false
oem123@oem123-Medio:~$ git config --global init.defaultBranch master
oem123@oem123-Medio:~$ git config --global core.autocrlf input
oem123@oem123-Medio:~$ git config --global core.safecrlf warn
oem123@oem123-Medio:~$ ssh-keygen -C "1132246746 sashandr000chanel000999000@gmail.com"
Generating public/private ed25519 key pair.
```

Рис. 1: Рис. 4.2.1 Базовые настройки Git

Рис. 4.2.1 Базовые настройки Git

4.3 Создание SSH ключа

Для последующей идентификации пользователя на сервере репозиториев необходимо сгенерировать пару ключей (приватный и открытый):

Рис. 2: Рис. 4.3.1 Генерация ключа

Далее я загрузил сгенерённый открытый ключ. Для этого я зашёл на сайт http://github.org/ под своей учётной записью и перейти в меню Setting . После этого выбрал в боковом меню SSH and GPG keys и нажал кнопку New SSH key . Скопировав из локальной консоли ключ в буфер обмена вставил ключ в появившееся на сайте поле и указываем для ключа имя (Title).

4.4 Сознание рабочего пространства и репозитория курса на основе шаблона

При выполнении лабораторных работ следует придерживаться структуры рабочего пространства. Рабочее пространство по предмету располагается в следующей иерархии:

~/work/study/ └── / └── / └── /

Название проекта на хостинге git имеет вид: study__ Я открыл терминал и создал каталог для предмета «Архитектура компьютера»:

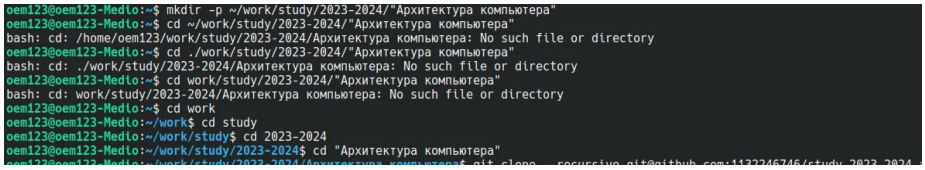

Рис. 3: Рис 4.4.1 Создание необходимых папок и каталогов

4.3 Сознание репозитория курса на основе шаблона

Репозиторий на основе шаблона можно создать через web-интерфейс github. Я перешёл на станицу репозитория с шаблоном курса https://github.com/yamadharma/course-directory-student-template.

В открывшемся окне я задал имя репозитория (Repository name) study_2024-2025_arhpc и создал репозиторий. Я открыл терминал и перешёл в каталог курса

и клонировал созданный репозиторий:

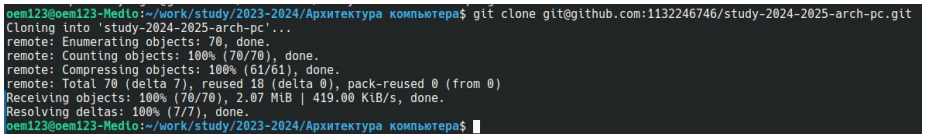

Рис. 4: Рис 4.3.1 Клонирование репозитория

4.4 Настройка каталога курса

Я перешёл в каталог курса, удалил лишние файлы, создал необходимые каталоги и отправил файлы на сетьел.

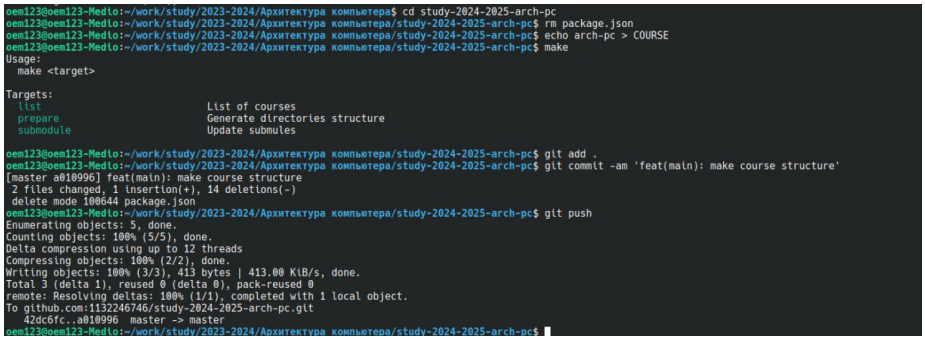

Рис. 5: изображение

5 Выводы

Я изучил идеологию и применение средств контроля версий. Я приобрёл практические навыки по работе с системой git.

Список литературы

1. Ссылка на ресурс
2. Ссылка на ресурс
3. Ссылка на ресурс
4. Ссылка на ресурс