Отчёт по лабораторной работе №2

Первоначальная настройка Git

Киньябаева Аиша Иделевна

Содержание

1 Цель работы **5**

2 Задание **6**

3 Выполнение лабораторной работы **7**

4 Выводы **14**

Список иллюстраций

3.1 регистрация 7
3.2 Заполнение данных 7
3.3 Базовая настройка 8
3.4 Генерация ключа 8
3.5 Копирование ключа 8
3.6 Генерация ключа 9
3.7 Настройка подписей 9
3.8 Копирование шаблона 9
3.9 Настройка каталога 10
3.10 Отправка файлов 10

Список таблиц

1 Цель работы

Целью данной работы является изучение контроля версий и создание репозитория на основе шаблона курса

2 Задание

Создать репозиторий на Git

3 Выполнение лабораторной работы

Регистрация на Github и заполнение основных данных репозитория (рис. 3.1), (рис. 3.2)

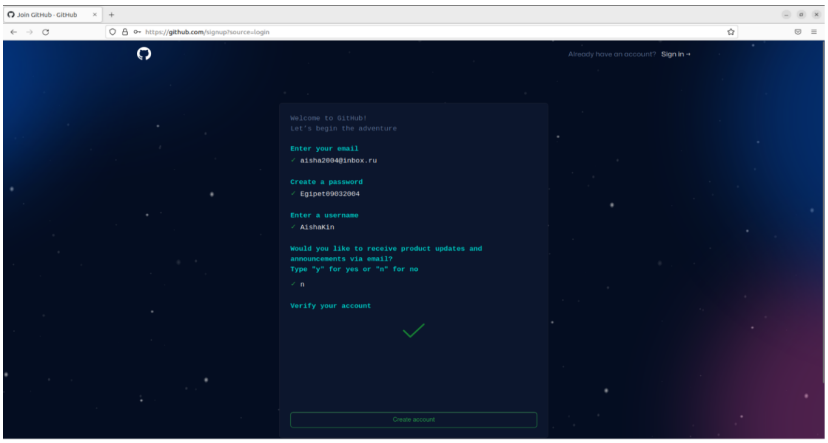

Рис. 3.1: регистрация

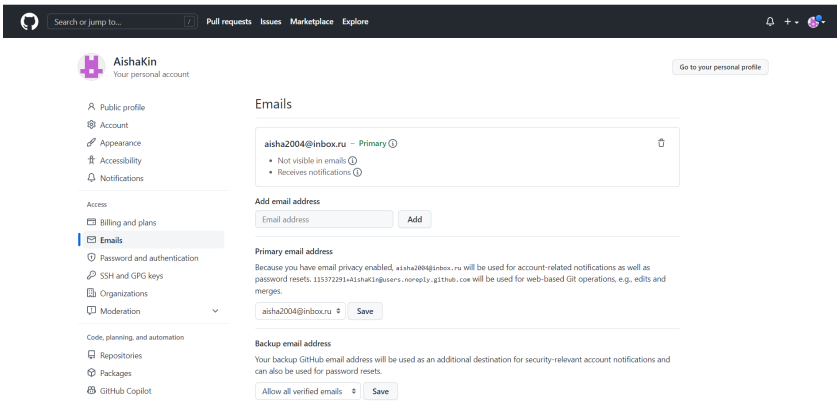

Рис. 3.2: Заполнение данных

Базовая настройка Git с использованием имени пользователя и электронной почты (рис. 3.3)

Рис. 3.3: Базовая настройка

Создание SSH ключа и копирование его в репозиторий (рис. 3.4), (рис. 3.5)

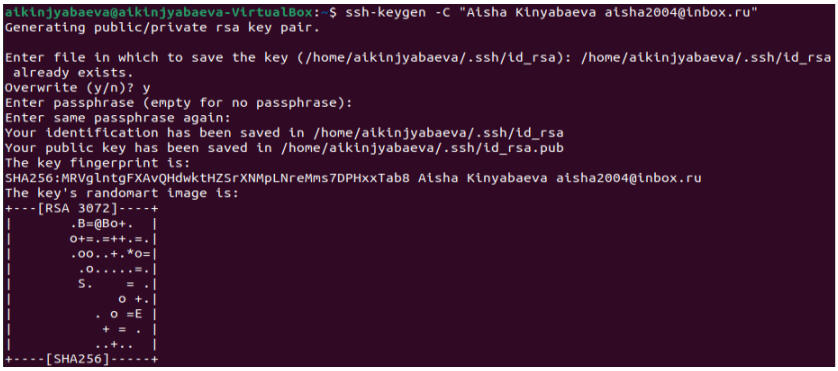

Рис. 3.4: Генерация ключа

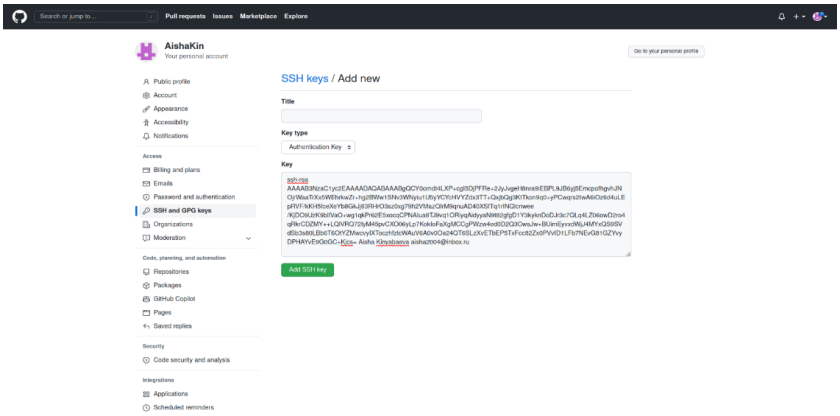

Рис. 3.5: Копирование ключа

Далее настраиваю параметры autocrlf и safecrlf. А также генерирую ключ gpg и копирую его на git (рис. 3.6)

```
fatal: не найден git репозиторий (или один из родительских каталогов): .git
aikinjyabaeva@aikinjyabaeva-VirtualBox:~$ git remote add origin
fatal: не найден git репозиторий (или один из родительских каталогов): .git
aikinjyabaeva@aikinjyabaeva-VirtualBox:~$ git config --global core.autocrlf input
aikinjyabaeva@aikinjyabaeva-VirtualBox:~$ git config --global core.safercrlf warn
aikinjyabaeva@aikinjyabaeva-VirtualBox:~$ gpg --full-generate-key
gpg (GnuPG) 2.2.27; Copyright (C) 2021 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Выберите тип ключа:
   (1) RSA и RSA (по умолчанию)
   (2) DSA и Elgamal
   (3) DSA (только для подписи)
   (4) RSA (только для подписи)
  (14) Имеющийся на карте ключ
Ваш выбор? 1
длина ключей RSA может быть от 1024 до 4096.
Какой размер ключа Вам необходим? (3072) 4096
Запрошенный размер ключа - 4096 бит
Выберите срок действия ключа.
         0 = не ограничен
      <n>  = срок действия ключа - n дней
      <n>w = срок действия ключа - n недель
      <n>m = срок действия ключа - n месяцев
      <n>y = срок действия ключа - n лет
```

Рис. 3.6: Генерация ключа

Настройка автоматических подписей коммитов git (рис. 3.7)

```
aikinjyabaeva@aikinjyabaeva-VirtualBox:~$ git config --global user.signingkey CDB5A54C069
18BC9
aikinjyabaeva@aikinjyabaeva-VirtualBox:~$ git config --global commit.gpgsign true
aikinjyabaeva@aikinjyabaeva-VirtualBox:~$ git config --global gpg.program $(which gpg2)
aikinjyabaeva@aikinjyabaeva-VirtualBox:~$
```

Рис. 3.7: Настройка подписей

Копирование шаблона курса (рис. 3.8)

```
aikinjyabaeva@aikinjyabaeva-VirtualBox:~$ cd ~/work/study/2022-2023/Операционные\ системы
/
aikinjyabaeva@aikinjyabaeva-VirtualBox:~/work/study/2022-2023/Операционные системы$ gh re
po create study_2022-2023_os-intro --template=yamadharma/course-directory-student-templat
e --public
GraphQL: Could not clone: Name already exists on this account (cloneTemplateRepository)
aikinjyabaeva@aikinjyabaeva-VirtualBox:~/work/study/2022-2023/Операционные системы$ git c
lone --recursive git@github.com:AishaKin/study_2022-2023_os-intro.git os-intro
Клонирование в «os-intro»…
remote: Enumerating objects: 27, done.
remote: Counting objects: 100% (27/27), done.
remote: Compressing objects: 100% (26/26), done.
remote: Total 27 (delta 1), reused 11 (delta 0), pack-reused 0
Получение объектов: 100% (27/27), 16.93 КиБ | 240.00 КиБ/с, готово.
Определение изменений: 100% (1/1), готово.
Подмодуль «template/presentation» (https://github.com/yamadharma/academic-presentation-ma
rkdown-template.git) зарегистрирован по пути «template/presentation»
Подмодуль «template/report» (https://github.com/yamadharma/academic-laboratory-report-tem
plate.git) зарегистрирован по пути «template/report»
Клонирование в «/home/aikinjyabaeva/work/study/2022-2023/Операционные системы/os-intro/te
mplate/presentation»…
remote: Enumerating objects: 82, done.
remote: Counting objects: 100% (82/82), done.
remote: Compressing objects: 100% (57/57), done.
```

Рис. 3.8: Копирование шаблона

Настройка каталога курса и отправка файлов на репозиторий (рис. 3.9), (рис. 3.10)

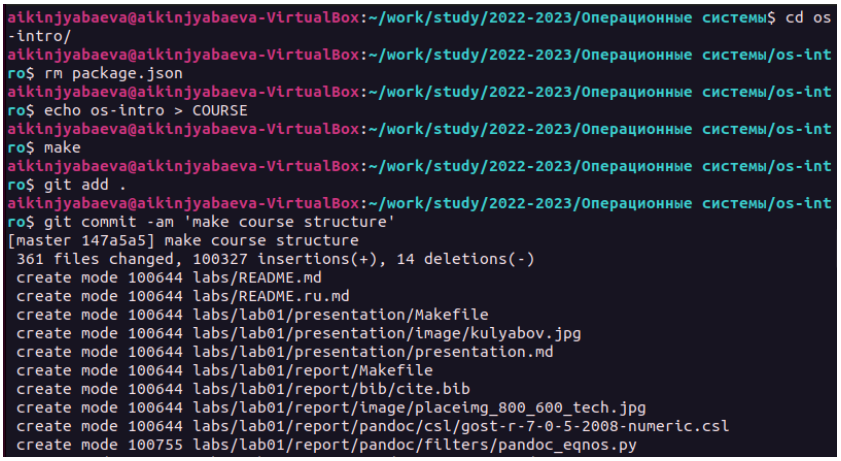

Рис. 3.9: Настройка каталога

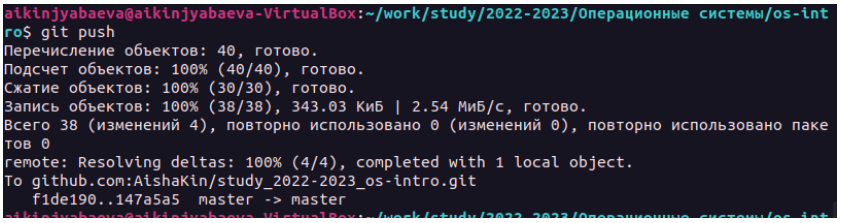

Рис. 3.10: Отправка файлов

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются? Системы контроля версий необходимы при работе нескольких человек над одним проектом. Прдназначены для решения таких задач как:

    - внесение изменений в проект разными участниками
    - совмещение изменений
    - возврат к более ранним версиям проекта
    - сохранение изменений

2. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия.

    - Репозиторий - место хранения файловой ситсемы каждого отдельного проекта. Существуют локальные и удаленные репозитории, где локальные для работы над проектом на компьютере, а удаленные в качестве хранилища
    - Commit - Операция, позволяющая сохранить текущее состояние проекта и добавить его в репозиторий
    - История - способность системы контроля версий сохранять и работать с несколькими версиями одного файла и его ветвлений
    - Рабочая копия - является снимком одной версии проекта. Эти файлы извлекаются из сжатой базы данных в каталоге Git и помещаются на диск, для того чтобы их можно было использовать или редактировать.

3. Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида.

    - Централизованные системы контроля версий предполагают сохранение версий проектов на общий сервер, с которого потом получают нужные версии клиенты. Примеры: CVS, Subversion, Perforce
    - В децентрализованных системах контроля версий при каждом копировании удалённого репозитория (расположенного на сервере) происходит полное копирование данных в локальный репозиторий (установленный на рабочем компьютере). Каждая копия содержит все данные, хранящиеся в удалённом репозитории. В случае, возникновения технической неисправности на стороне сервера, удаленный репозиторий можно перезаписать с любой сохраненной копии. Пример: Mercurial, Bazaar

4. Опишите действия с VCS при единоличной работе с хранилищем.

    - Создадим локальный репозиторий.

- Сделаем предварительную конфигурацию, указав имя и email владельца репозитория
- Инициализация локального репозитория, расположенного, например, в каталоге ~/tutorial
- После это в каталоге tutorial появится каталог .git, в котором будет храниться история изменений.
- Воспользуемся командой status для просмотра изменений в рабочем каталоге, сделанных с момента последней ревизии:git status
- Во время работы над проектом так или иначе могут создаваться файлы, которые не требуется добавлять в последствии в репозиторий. Например, временные файлы, создаваемые редакторами, или объектные файлы, создаваемые компиляторами. Можно прописать шаблоны игнорируемых при добавлении в репозиторий типов файлов в файл .gitignore с помощью сервисов. Для этого сначала нужно получить список имеющихся шаблонов

5. Каковы основные задачи, решаемые инструментальным средством git? Git не только позволяет сохранять контрольные точки проекта, но и помогает устранять конфликты. Часто бывает так, что программисты одновременно работают над одной функцией и заливают изменения в репозиторий. В этом случае система обнаруживает конфликт и пытается исправить его автоматически.
6. Назовите и дайте краткую характеристику командам 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 в текущей директории: git status

- Добавить все изменённые и/или созданные файлы и/или каталоги: git add .
- Удалить файл и/или каталог из индекса репозитория (при этом файл и/или каталог остаётся в локальной директории): git rm имена_файлов
- Сохранить все добавленные изменения и все изменённые файлы: git commit -am ‘Описание коммита’
- Сохранить добавленные изменения с внесением комментария через встроенный редактор: git commit
- Отправка изменений конкретной ветки в центральный репозиторий: git push origin имя_ветки

7. Что такое и зачем могут быть нужны ветви (branches)? Ветки нужны для того, чтобы программисты могли вести совместную работу над проектом и не мешать друг другу при этом. При создании проекта, Git создает базовую ветку. Она называется master веткой. Она считается центральной веткой, т.е. в ней содержится основной код приложения.
8. Как и зачем можно игнорировать некоторые файлы при commit? Игнорируемые файлы — это, как правило, артефакты сборки и файлы, генерируемые машиной из исходных файлов в вашем репозитории, либо файлы, которые по какой-либо иной причине не должны попадать в коммиты ЫИгнорируемые файлы отслеживаются в специальном файле .gitignore, который регистрируется в корневом каталоге репозитория. В Git нет специальной команды для указания игнорируемых файлов: вместо этого необходимо вручную отредактировать файл .gitignore, чтобы указать в нем новые файлы, которые должны быть проигнорированы. Файлы .gitignore содержат шаблоны, которые сопоставляются с именами файлов в репозитории для определения необходимости игнорировать эти файлы

4 Выводы

В ходе данной лабораторной работы были изучены основные моменты создания репозитория на git, основные команды гит и непосредственно копирование файлов на репозиторий.