Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии
Департамент цифровых, робототехнических систем и электроники

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №1.1
дисциплины «Искусственный интеллект в профессиональной сфере»

#
#
#
#
#

Выполнил:
Оганесов Артур Витальевич
3 курс, группа ЭНЭ-б-о-22-1,
11.03.04 «Электроника и наноэлектроника», направленность (профиль) «Промышленная электроника», очная форма обучения

_____
(подпись)

Проверил:
Воронкин Роман Александрович,
доцент

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2024 г.

Тема работы: исследование основных возможностей Git и GitHub.

Цель работы: исследовать базовые возможности системы контроля версий Git и веб-сервиса для хостинга IT-проектов GitHub.

Аппаратура и материалы: ПК, операционная система Windows 10, Git, браузер для доступа к web-сервису GitHub.

Ход работы:

1. Изучить теоретический материал работы. На основе полученных знаний создал общедоступный репозиторий на GitHub, в котором использована лицензия MIT и выбранный мной язык программирования (python).

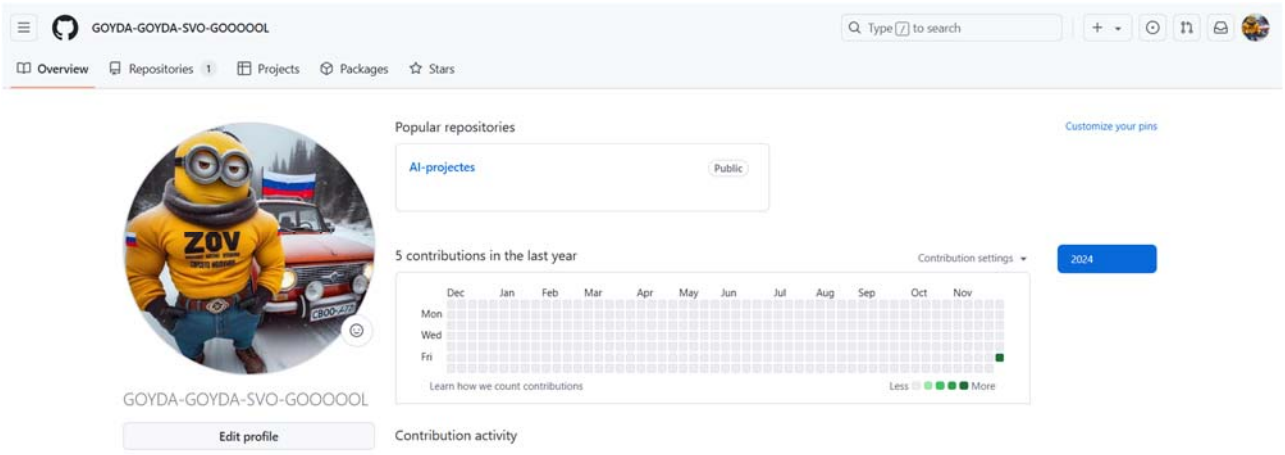

Рисунок 1 – Профиль в GitHub после регистрации

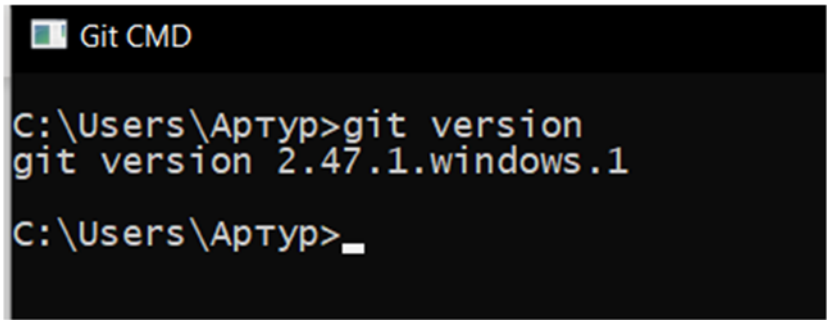

Рисунок 2 – Установленный Git

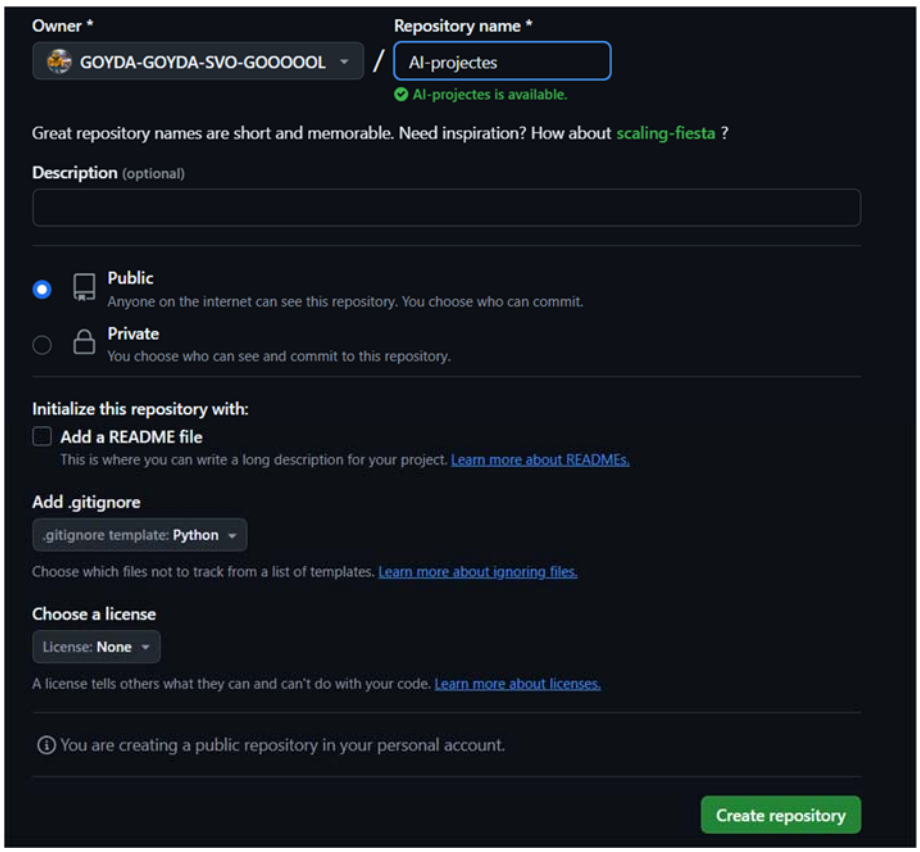

Рисунок 3 – Параметры нового репозитория

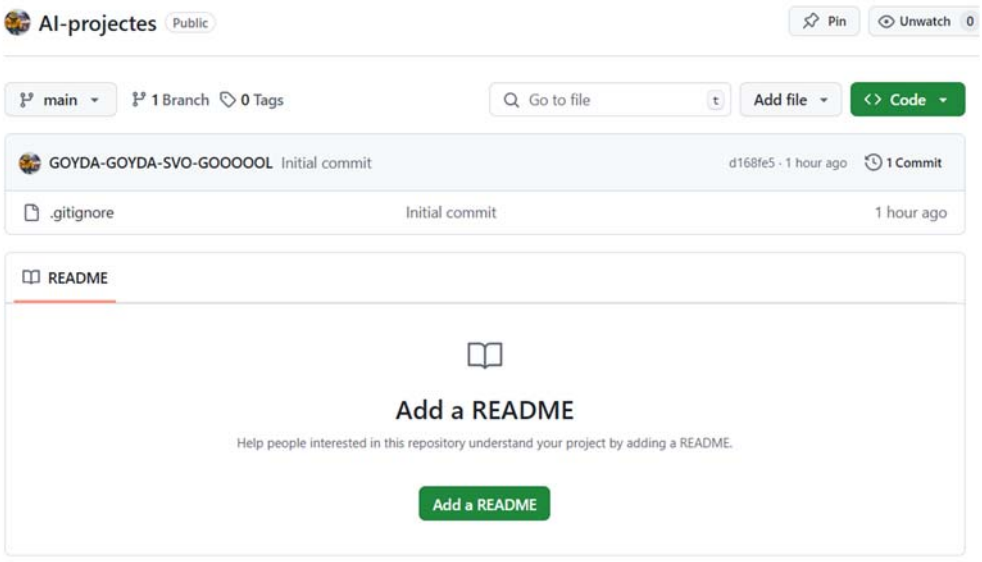

Рисунок 4 – Новый репозиторий

2. Клонировал созданный репозиторий на рабочий компьютер.

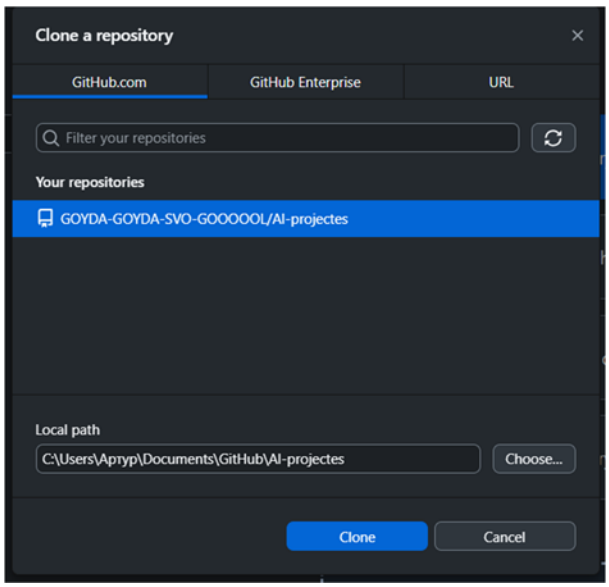

Рисунок 5 – Клонирование

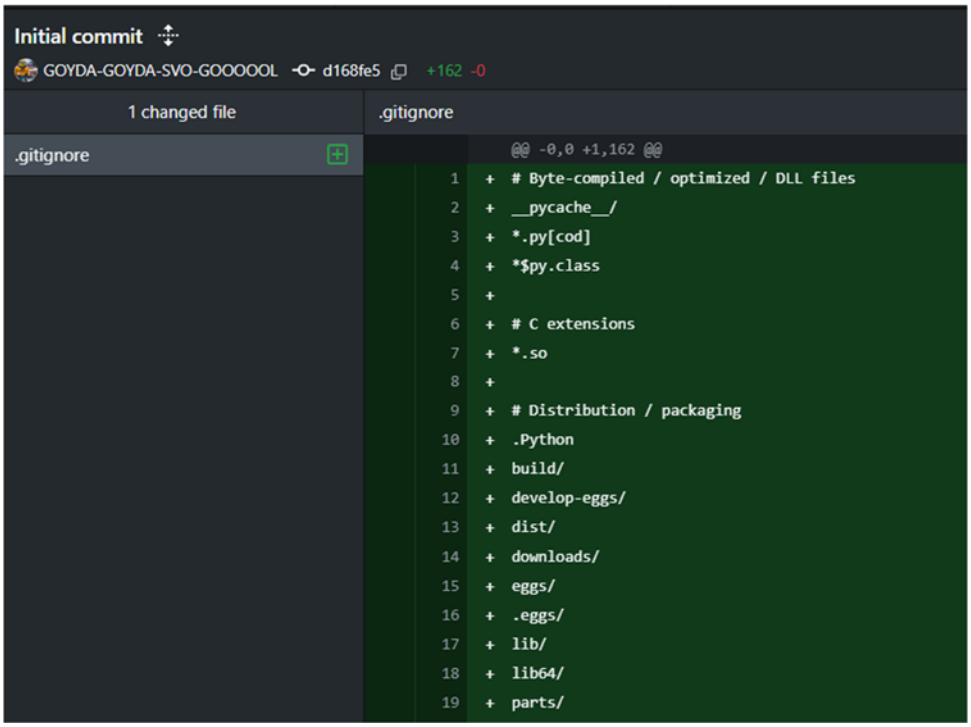

Рисунок 5 – выбран Python

3. Дополнил файл .gitignore необходимыми правилами для выбранного языка программирования и интегрированной среды разработки.

4. Добавил в файл README.md информацию о группе и ФИО и зафиксировал изменения.

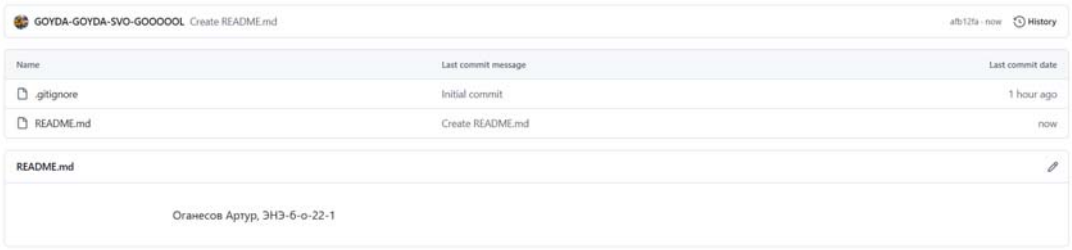

Рисунок 6 – README

5. Написал небольшую программу на выбранном языке программирования Python. Зафиксировал изменения при написании программы в локальном репозитории. Отправил преподавателю ссылку на Git.

```
... age = input("Введите ваш возраст: ")
... print(f"Привет, {name}! Тебе {age} лет.")
... years_until_100 = 100 - int(age)
... print(f"Через {years_until_100} лет тебе будет 100 лет!")
Введите ваше имя: >? Артурио
Введите ваш возраст: >? 20
Привет, Артурио! Тебе 20 лет.
Через 80 лет тебе будет 100 лет!
```

Рисунок 7 – Простая первая программа

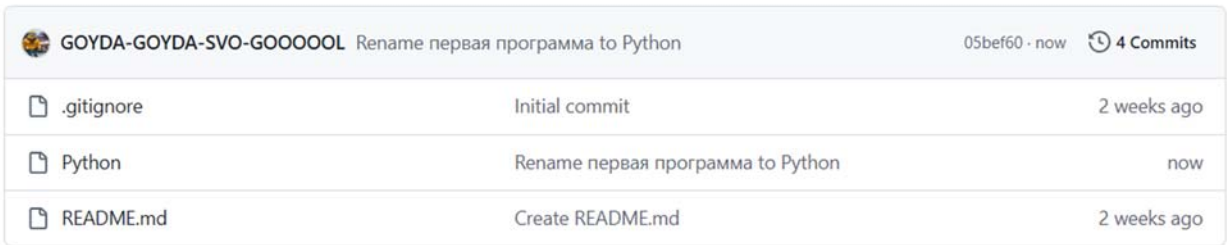

Рисунок 8 – Добавленный файл

Ответы на вопросы:

1. Что такое СКВ и каково ее назначение?

Это инструмент, используемый разработчиками программного обеспечения для управления изменениями в исходном коде и других файловых ресурсах.

2. В чем недостатки локальных и централизованных СКВ?

Локальные системы контроля версий хранят исходный код и все его версии в одном месте — на компьютере разработчика. Главный недостаток таких систем — в случае сбоя может быть потеряна не только какая-то из версий, но и весь проект целиком. Кроме того, локальные СКВ неудобны для организации командной разработки ПО.

Централизованные системы контроля версий предполагают хранение проекта на едином сервере. Недостаток таких систем — уязвимость сервера. В случае сбоя на сервере или проблем с подключением к нему данные могут быть потеряны или недоступны сразу для всех разработчиков.

3. К какой СКВ относится Git?

Git относится к распределённой системе контроля версий (РСКВ).

4. В чем концептуальное отличие Git от других СКВ?

Такие системы отличаются тем, что каждый пользователь держит у себя полную копию репозитория вместе с историей. Если любой из серверов обмена данными исчезнет, клиентский репозиторий может быть скопирован на другой сервер для продолжения работы.

5. Как обеспечивается целостность хранимых данных в Git?

Git сохраняет все объекты в свою базу данных не по имени, а по хеш-сумме содержимого объекта.

6. В каких состояниях могут находится файлы в Git? Как связаны эти состояния?

Файлы в Git могут находиться в трёх основных состояниях:

Зафиксированное (committed) — данные надёжно сохранены в локальной базе.

Модифицированное (modified) — изменения уже внесены в файл, но пока не зафиксированы в базе данных.

Индексированное (staged) — текущая версия модифицированного файла помечена как предназначенная для следующей фиксации.

7. Что такое профиль пользователя в GitHub?

Профиль пользователя в GitHub — это страница, на которой можно разместить контакты, список навыков, код проектов и ссылки на важные ресурсы.

8. Какие бывают репозитории в GitHub?

Репозитории в GitHub бывают:

Частные (private) — скрытые от посторонних глаз.

Публичные (public) — открытые для всех желающих.

9. Укажите основные этапы модели работы с GitHub.

- Регистрация на GitHub: создание аккаунта.
- Установка Git.
- Создание первого репозитория.
- Работа с ветками.
- Работа в команде: конфликты версий и git pull.

10. Как осуществляется первоначальная настройка Git после установки?

Для первоначальной настройки Git после установки можно выполнить следующие действия:

Указать имя пользователя и адрес электронной почты. Это важно, потому что каждый коммит в Git содержит эту информацию. Выбрать текстовый редактор. По умолчанию Git использует стандартный редактор вашей системы, которым обычно является Vim. Настроить ветку по умолчанию. Когда вы инициализируете репозиторий командой git init, Git создаёт ветку с именем master по умолчанию. Начиная с версии 2.28, вы можете задать другое имя для создания ветки по умолчанию.

11. Опишите этапы создания репозитория в GitHub.

Нажмите на кнопку «Start a project». Введите название и описание репозитория. Поставьте или нет галочку на «Initialize this repository with a

README». Выберите нужный тип лицензии и нажмите на кнопку «Create project».

12. Какие типы лицензий поддерживаются GitHub при создании репозитория?

- Public. Это наиболее распространённый тип лицензии. Она позволяет любому человеку использовать ваш код в любых целях, если это не коммерческая деятельность.
- Commercial. Эта лицензия позволяет продавать ваше программное обеспечение или услуги с использованием вашего кода.
- Academic. Этот тип лицензии доступен только для университетских исследовательских целей. Перед использованием в других целях он должен быть одобрен комитетом по авторскому праву учреждения.

13. Как осуществляется клонирование репозитория GitHub? Зачем нужно клонировать репозиторий?

Клонирование репозитория GitHub осуществляется с помощью команды git clone.

Эта команда позволяет получить клоны всех версий файлов репозитория. Это может быть полезно при доработке готового проекта или при внедрении его компонентов в свой.

14. Как проверить состояние локального репозитория Git?

Для проверки состояния локального репозитория Git можно использовать команду git status.

Эта команда покажет, какие файлы изменили, удалили или добавили в проект. При этом статус «Закоммичен» не отобразится.

Выводы: в ходе лабораторной работы были исследованы базовые возможности системы контроля версий Git и веб-сервиса для хостинга IT-проектов GitHub.