**Лабораторная работа № 20**

Системы контроля версий (CVS):

1. Назначение систем CVS (Concurrent Versions System) - это управление версиями и изменениями в исходном коде и других текстовых файлов. CVS позволяет команде разработчиков совместно работать над проектами, отслеживать изменения, вносить правки и восстанавливать предыдущие версии файлов.

2. Виды CVS и их особенности:

- CVS (Concurrent Versions System) - стандартная система CVS.

- SVN (Subversion) - развитие и улучшение CVS с поддержкой более продвинутых функций, таких как атомарные коммиты и переименование файлов.

- Git - не CVS, но система контроля версий, которая существенно отличается от CVS и SVN, используя распределенную структуру хранения данных и обладая множеством преимуществ.

Система контроля версий Git:

3. Git - это распределенная система контроля версий, которая позволяет отслеживать и управлять версиями файлов. Особенности Git:

- Распределенная структура, что позволяет каждому участнику проекта иметь полную копию репозитория.

- Ветвление и слияние (branching and merging) для эффективной работы над различными фичами.

- Быстрая скорость и эффективное сжатие данных.

- Возможность работать оффлайн.

- Поддержка набора инструментов для совместной разработки (например, GitHub).

4. Процесс установки и настройки Git на ПК:

- Скачайте установочный пакет Git с официального сайта (https://git-scm.com/).

- Запустите установщик и следуйте инструкциям по установке, выбирая опции по вашему усмотрению.

- После установки, настройте имя пользователя и адрес электронной почты в Git, используя команды:

```

git config --global user.name "Ваше имя"

git config --global user.email "ваша@почта.com"

```

GitHub:

1. Назначение сервиса GitHub - это хостинг для хранения и управления репозиториями Git. GitHub обеспечивает совместную работу над проектами, управление задачами и многое другое.

2. GitHub Desktop - это графический клиент для Git и GitHub. Он предоставляет удобный интерфейс для работы с репозиториями, коммитами, ветками и совместной разработкой.

Термины Git и GitHub:

1. Репозиторий - это место, где хранятся все файлы, история изменений и данные проекта.

2. Рабочая область (Working Directory) и хранилище (Repository) - рабочая область - это каталог на вашем ПК, в котором вы редактируете файлы. Хранилище - это место, где Git хранит версии файлов и историю изменений.

3. Форк (Fork) - это копия чужого репозитория на GitHub, позволяющая внести изменения и предложить их для слияния обратно в исходный репозиторий.

4. Ветка (Branch) - это отдельная линия разработки, которая позволяет работать над определенной функциональностью или исправлениями.

5. Мастер (Master) - это основная ветка, на которой обычно находится стабильная версия проекта.

6. Коммит (Commit) - это сохранение изменений в репозитории с описанием, что было изменено.

7. Пул (Pull) - это запрос на включение изменений из одной ветки в другую.

8. Пуш (Push) - это отправка ваших локальных изменений в удаленный репозиторий.

9. Пулреквест (Pull Request) - это запрос на внесение изменений из вашей ветки в исходный репозиторий (часто используется в контексте GitHub).

10. Мердж (Merge) - объединение изменений из одной ветки в другую.

11. Кодревью (Code Review) - процесс, при котором другие разработчики анализируют и обсуждают ваш код, чтобы обнаружить ошибки и улучшить его качество.

Ссылка на репозиторий курсового проекта: <https://github.com/MikutsevichWadim/BAP.CourseWork-Python.Kivy>

Задание 1.

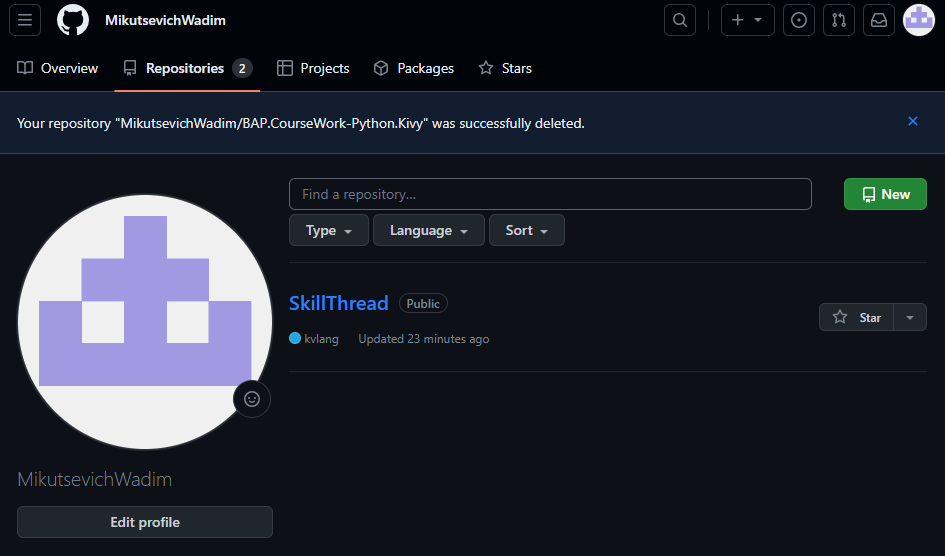
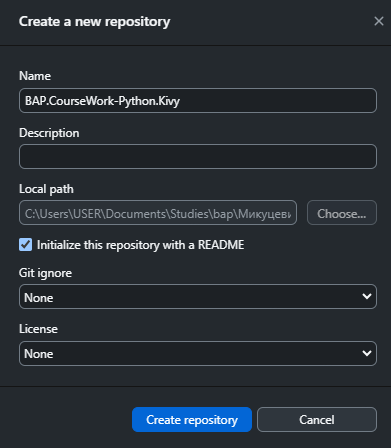
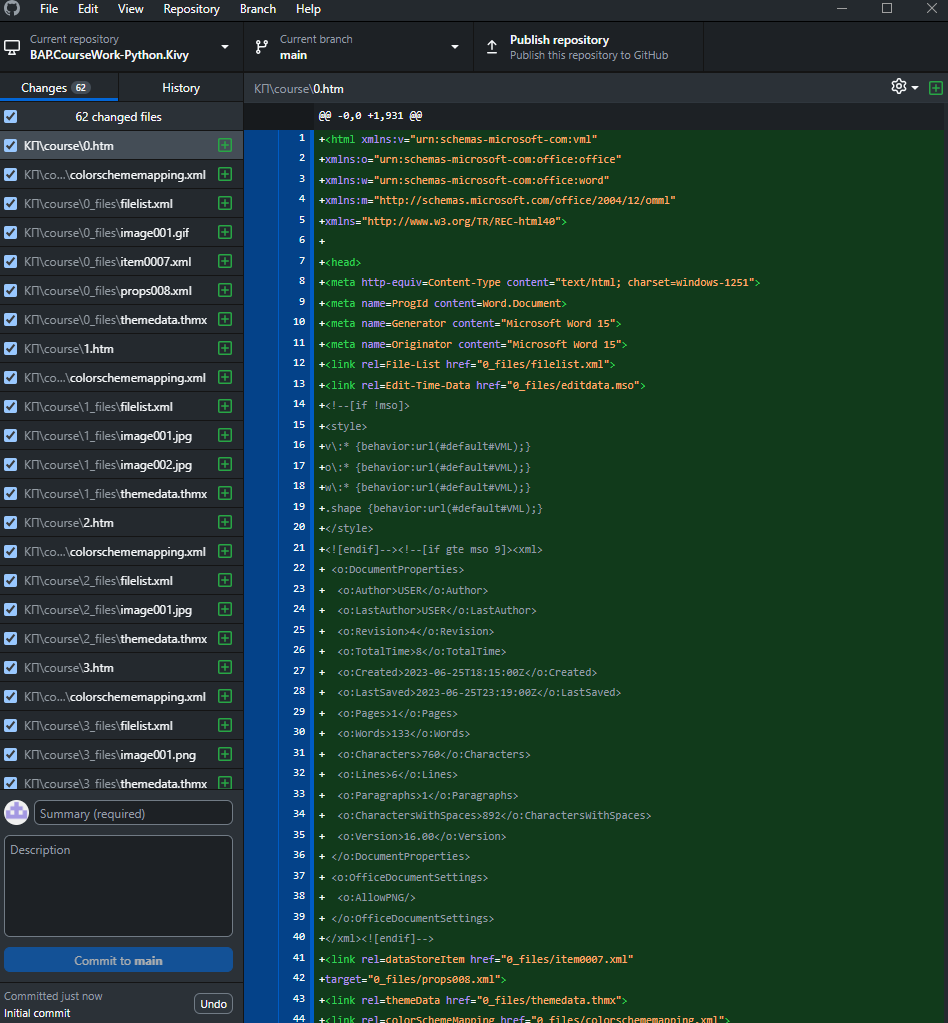
1. Создать свой профиль на github.

Рисунок 1.1 – Профиль GitHub

2. Установить на свой ПК GitHub Desktop и создать в нем новый репозиторий.

Рисунок 1.2 – Новый репозиторий

3. Разместить в нем свой КУРСОВОЙ ПРОЕКТ (программу-исходники, файл SETUP, ПЗ, ГЧ)

Рисунок 1.3 – Файлы в локальном репозитории

4. Сделать интеграцию вашего репозитория с GitHub.

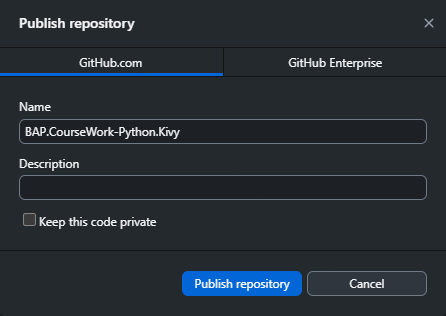


Рисунок 1.4 – Публикация репозитория

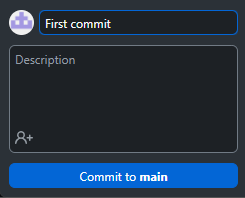


Рисунок 1.5- Первый коммит

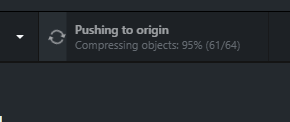


Рисунок 1.6 – Пуш репозитория

5. Ссылку на свой профиль в github на Курсовой проект сбросить в google class в ЛР работу.

Ссылка на репозиторий курсового проекта: <https://github.com/MikutsevichWadim/BAP.CourseWork-Python.Kivy>

Задание 2.

На примере любого вашего проекта по c# , python И др. Необходимо показать основные возможные Типовые действия с проектом:

Создание репозитория

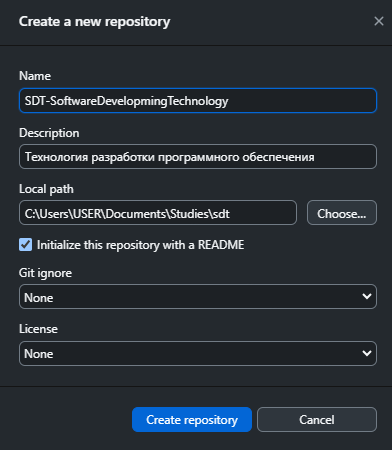


Рисунок 2.1 – Создание репозитория

Создание мастера, ветки, форка

2) Создание ветки

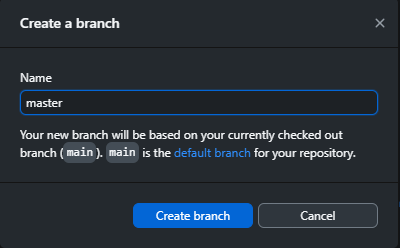


Рисунок 2.2 – Создание мастера

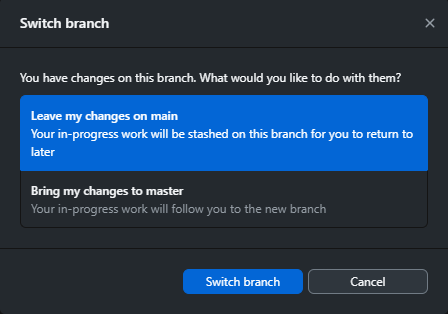


Рисунок 2.3 – Выбор ветки коммитов

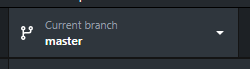


Рисунок 2.4 – Готовый мастер

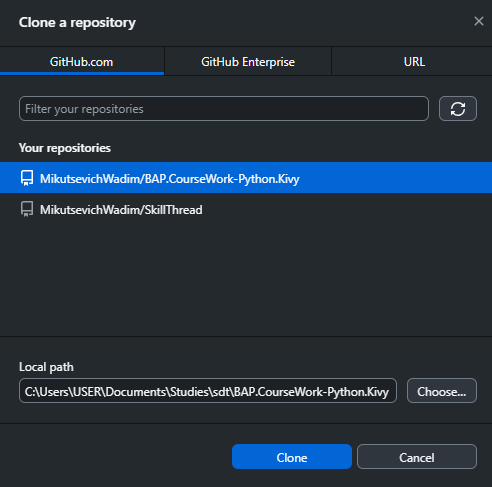
3) Создание форка

Рисунок 2.5 – Создание форка курсового проекта