Министерство образования Республики Беларусь

Учреждение образования

Белорусский Государственный университет информатики и радиоэлектроники

Инженерно-экономический факультет

Кафедра экономической информатики

Отчет

по лабораторной работе № 1

на тему «Системы контроля версий(Git)»

|  |  |
| --- | --- |
| Выполнил: | Летко А.С. |
|  | студент группы 377901 |
| Проверил: | Юдашкин В.О. |
|  | ассистент кафедры ЭИ |

Минск 2025

**Цель работы**

Приобрести навыки работы с системой контроля версий Git, ознакомиться с особенностями платформы GitHub.

**Задачи**

1 Создать репозиторий на платформе GitHub.

2 Установить приложение Git Bash.

3 Воспользоваться основными Git-командами и некоторыми возможностями платформы GitHub.

1. **Теоретическая часть**
   1. **Введение в Git и GitHub**

**– Git** – это распределённая система контроля версий, которая позволяет отслеживать изменения в коде, управлять версиями и работать над проектами в команде. Устанавливается локально на компьютер.

**– GitHub** – это веб-сервис для хостинга IT-проектов и их совместной разработки, использующий Git. Предоставляет удобный интерфейс для управления репозиториями.

**– Взаимосвязь:** Git – это инструмент, а GitHub – платформа для работы с этим инструментом в команде.

* 1. **Основные понятия и команды Git (Краткий конспект)**

**– Репозиторий (Repository)** – хранилище проекта и его истории изменений.

– git init – инициализация нового локального репозитория.

**– Рабочая область, Индекс (Staging Area), Репозиторий** – три основных состояния файлов.

– git status – просмотр статуса файлов.

– git add <файл> – добавление файла в индекс для подготовки к коммиту.

**– Коммит (Commit)** – снимок состояния файлов на определённый момент времени.

– git commit -m "сообщение" – создание коммита с комментарием.

– git log – просмотр истории коммитов.

**– Ветка (Branch)** – изолированная линия разработки.

– git branch <имя\_ветки> – создание новой ветки.

– git checkout <имя\_ветки> – переключение между ветками.

– git merge <имя\_ветки> – слияние указанной ветки с текущей.

**– Удалённый репозиторий (Remote Repository)** – репозиторий, размещённый на сервере (например, на GitHub).

– git remote add origin <url> – связывание локального репозитория с удалённым.

– git push -u origin main – отправка локальных коммитов на удалённый репозиторий.

– git clone <url> – клонирование удалённого репозитория на локальный компьютер.

– git pull – получение изменений из удалённого репозитория и их слияние с локальной версией.

1. **Практическая часть. Ход работы**

**Задание №1: Базовые операции с локальным репозиторием**

**Цель задания:** Освоить создание репозитория, отслеживание файлов и фиксацию изменений.

**Ход выполнения:**

**1 Создание директории и репозитория:**

– Создал новую папку lab1\_project на рабочем столе.

– Открыл эту папку и запустил в ней Git Bash.

– Выполнил команду git init для инициализации репозитория.

1. **Создание и коммит файлов:**

– Создал в папке два текстовых файла: file1.txt и file2.txt.

– Проверил статус командой git status. Увидел, что файлы помечены как Untracked files.

– Добавил файлы в индекс командой git add ..

– Проверил статус git status повторно. Убедился, что файлы теперь в индексе (Changes to be committed).

– Создал первый коммит командой git commit -m "Добавлены file1 и file2".

**3 Внесение изменений и второй коммит:**

– Открыл файл file1.txt, добавил в него текст и сохранил.

– Проверил статус git status. Увидел, что файл file1.txt помечен как modified.

– Проиндексировал изменения: git add file1.txt.

– Создал второй коммит: git commit -m "Обновлено содержимое file1.txt".

**4 Просмотр истории:**

– Выполнил команду git log и убедился, что в истории присутствуют два созданных коммита с указанными сообщениями.

**Вывод по заданию №1:** Были успешно освоены базовые команды Git для работы с локальным репозиторием: инициализация, добавление файлов в индекс, создание коммитов и просмотр истории.

**Задание №2: Работа с ветками**

**Цель задания:** Освоить создание и переключение между ветками, а также слияние изменений.

**Ход выполнения:**

**1 Создание и переключение на ветку Dev:**

– Находясь в папке проекта, создал новую ветку командой git branch dev.

– Переключился на неё командой git checkout dev.

**2 Внесение изменений в ветке Dev:**

– Внёс изменения в файл file2.txt в ветке dev.

– Проиндексировал изменения и создал коммит: git add file2.txt, git commit -m "Изменение file2 в ветке dev".

**3 Слияние изменений (Merge):**

– Переключился обратно на ветку main: git checkout main.

– Выполнил слияние ветки dev с main: git merge dev.

– Убедился, что изменения из dev появились в main.

**4 Восстановление состояния (опционально, например, через revert):**

– Для демонстрации отката использовал команду git log чтобы найти хеш последнего коммита.

– Выполнил команду git revert <хеш\_коммита>, чтобы создать новый коммит, отменяющий изменения.

**Вывод по заданию №2:** Были освоены методы работы с ветками: их создание, переключение и слияние с помощью merge. Также была опробована функция отката изменений через revert.

**Задание №3: Работа с удалённым репозиторием (GitHub)**

**Цель задания:** Научиться работать с GitHub: создавать репозитории, отправлять изменения, взаимодействовать с другими разработчиками через Fork и Pull Request.

**Ход выполнения:**

**1 Создание удалённого репозитория:**

– Создал новый публичный репозиторий на GitHub с именем web-labs.

– Следовал инструкциям GitHub, чтобы связать существующий локальный репозиторий с удалённым:

bash

git remote add origin https://github.com/<мой\_аккаунт>/web-labs.git

git branch -M main

git push -u origin main

**2 Совместная работа (имитирована с помощью второго аккаунта или напарника):**

*–* Напарник перешёл по моей ссылке на репозиторий и нажал кнопку **Fork**.

– Он склонировал свой форк к себе на компьютер: git clone <url\_его\_форка>.

– Внёс изменения в один из файлов (например, добавил свою фамилию в README.md).

– Отправил изменения в свой форк: git add ., git commit -m "Добавил фамилию", git push.

– На странице своего форка на GitHub создал **Pull Request** в мой исходный репозиторий.

1. **Проверка и слияние Pull Request:**

– На странице своего репозитория на GitHub во вкладке **Pull requests** увидел запрос от напарника.

– Просмотрел внесённые изменения.

– Нажал кнопку **Merge pull request** для слияния изменений.

1. **Получение изменений в локальный репозиторий:**

– Вернулся в свою локальную папку проекта.

– Выполнил команду git pull --rebase, чтобы получить актуальную версию из GitHub и объединить историю с помощью rebase.

– Убедился, что файлы на локальном компьютере обновились и содержат изменения, сделанные напарником.

**Вывод по заданию №3:** Были успешно освоены основы работы с GitHub: создание репозитория, отправка кода, а также механизм совместной работы через Fork и Pull Request. Локальный репозиторий был синхронизирован с удалённым.

**3. Заключение**

В ходе выполнения лабораторной работы были успешно достигнуты все поставленные цели. Приобретены практические навыки работы с системой контроля версий Git, начиная от базовых операций (инициализация, коммиты) и заканчивая продвинутыми механизмами (ветвление, слияние). Ознакомились с платформой GitHub, научились создавать удалённые репозитории, отправлять и загружать изменения, а также изучили модель совместной работы через Fork и Pull Request. Полученные знания являются фундаментальными для дальнейшей работы в области веб-программирования и совместной разработки программного обеспечения.