|  |
| --- |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ |
| ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ |
| **«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»** |
| **(МОСКОВСКИЙ ПОЛИТЕХ)** |
| Факультет информационных технологий |

Кафедра «Прикладная информатика»

Форма обучения: очная

**Лабораторная работа №3**

**Git и GitHub**

**по дисциплине**

**«Хранилища данных»**

Студент (А. Костромин)

(личная подпись) (И.О. Фамилия)

**Москва 2021**

# ОГЛАВЛЕНИЕ

[ВВЕДЕНИЕ 3](#_Toc87115300)

[ХОД РАБОТЫ 4](#_Toc87115301)

[ЗАКЛЮЧЕНИЕ 8](#_Toc87115302)

[СПИСОК ИСТОЧНИКОВ 9](#_Toc87115303)

# ВВЕДЕНИЕ

Система контроля версий — это система, записывающая изменения в файл или набор файлов в течение времени и позволяющая вернуться позже к определённой версии. Git – это очень популярная система контроля версий и совместной разработки проектов с открытым исходным кодом. С помощью Git можно отслеживать изменения в исходном коде создаваемых проектов, возвращать предыдущие версии в случае критических ошибок [[1](https://tokmakov.msk.ru/blog/item/171#:~:text=Git%20—%20это%20популярная%20система,и%20принимать%20от%20них%20исправления)].

Основное отличие Git от любой другой СКВ (включая Subversion и её собратьев) – это подход к работе со своими данными. Концептуально, большинство других систем хранят информацию в виде списка изменений в файлах. Эти системы (CVS, Subversion, Perforce, Bazaar и т. д.) представляют хранимую информацию в виде набора файлов и изменений, сделанных в каждом файле, по времени (обычно это называют контролем версий, основанным на различиях). Git не хранит и не обрабатывает данные таким способом. Подход Git к хранению данных больше похож на набор снимков миниатюрной файловой системы. Каждый раз, когда происодит коммит, то есть сохраняется состояние проекта в Git, система запоминает, как выглядит каждый файл в этот момент, и сохраняет ссылку на этот снимок. Репозиторием называют хранилище кода и историю его изменений. Git работает локально и все репозитории хранятся в определенных папках на жестком диске. Помимо этого, репозитории можно хранить и в интернете. Git позволяет отправлять данные на удаленный сервер. Отправляются не только готовая версия, но и все коммиты, таким образом, любой человек из команды может посмотреть историю изменений. В качестве такого сервиса в данной лабораторной работе будет использоваться GitHub [2].

Целью данной лабораторной работы является ознакомление и тестирование системы контроля версий git. В ходе выполнения лабораторной работы будет создан локальный проект в git, а после он будет размещен в удаленном репозитории на GitHub.

# ХОД РАБОТЫ

Для установки Git в Windows имеется несколько способов. Официальная сборка доступна для скачивания на официальном сайте Git (https://git-scm.com/download/win), загрузка запустится автоматически. Стоит отметить, что это отдельный проект, называемый Git для Windows. Для автоматической установки можно использовать пакет Git Chocolatey.

Интерфейсом для работы с Git-ом является консоль/терминал. В процессе установки была выбрана возможность работать с Git в стандартной командой строке Windows cmd.

После установки git запускаем cmd и начинаем пошагово выполнять задание лабораторной работы. Сначала запускаем git и создаем папку будущего проекта при помощи команды:

mkdir testlab3

Далее переходим в созданный каталог при помощи команды:

cd testlab3

Вводим следующую команду для инициирования локального репозитория:

git init

Создаем файл с кодом для будущей загрузки в удаленный репозиторий. Для демонстрационного файла был выбран файл Mysql\_create.sql с кодом из предыдущей лабораторной работы. Перед этим создаем удаленный репозиторий на сайте github.com. Процесс создания репозитория представлен на Рисунке 1.

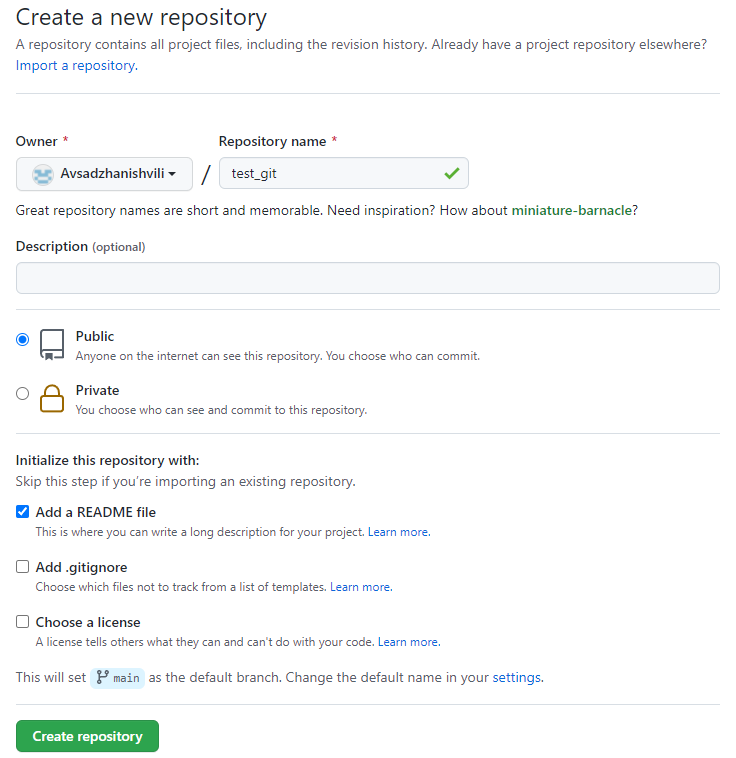


Рисунок – Процесс создания репозитория на GitHub

Созданный репозиторий представлен на Рисунке 2.

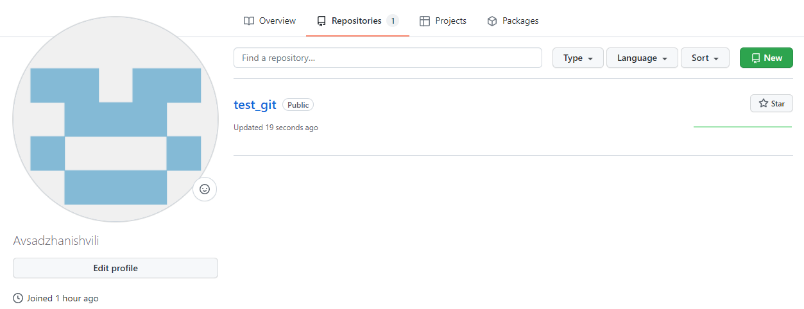


Рисунок – Удаленный репозиторий

Теперь подготавливаем ранее созданный файл для фиксации при помощи следующей команды:

git add .

Создаем коммит с комментарием:

git commit -m "first commit, laboratory work"

Создаем подключение с удаленным репозиторием:

git remote add origin https://github.com/xhgcs/test\_project

Перемещаем изменения на локальном компьютере в удаленный репозиторий:

git push -u origin master

После выполнения всех программ, проверим наличие файла в удаленном репозитории (Рисунок 3). Полный код выполнения лабораторной работы представлен на Рисунке 4. Ссылка на удаленный репозиторий: <https://github.com/Avsadzhanishvili/test_git>

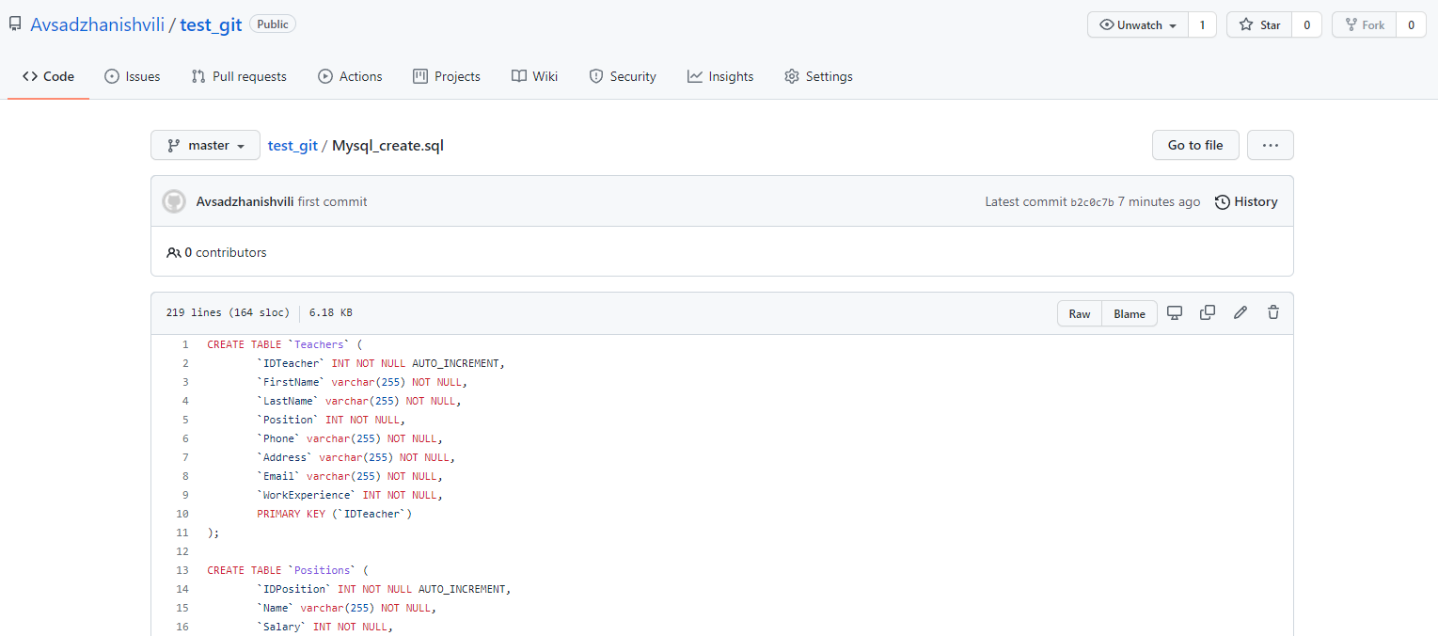


Рисунок 3 – Файл в удаленном репозитории test\_git

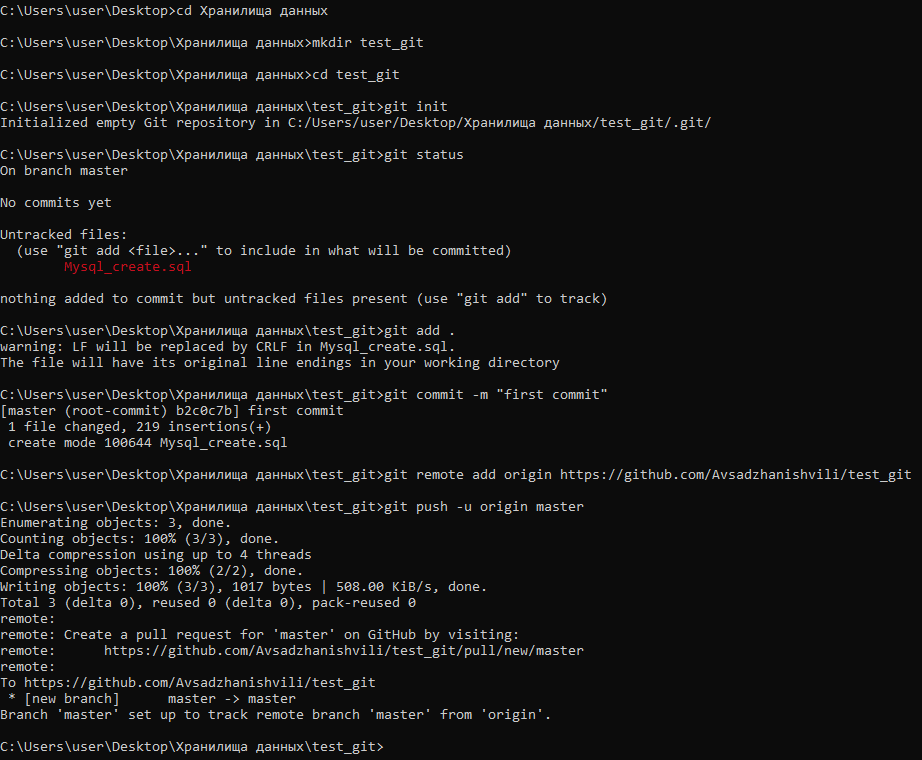


Рисунок – Полный код программы

# ЗАКЛЮЧЕНИЕ

Выполнение данной лабораторной работы позволило ознакомится с системой контроля версий Git и с платформой, хранящей различные Git-репозитории GitHub.

После выполнения лабораторной работы можно выделить следующие достоинства Git:

* гибкая система ветвления проектов и слияния веток между собой;
* высокая производительность и скорость работы;
* наличие локального репозитория, содержащего полную информацию обо всех изменениях, позволяет вести полноценный локальный контроль версий и заливать в главный репозиторий только полностью прошедшие проверку изменения;
* возможность делать контрольные точки, что позволяет уменьшить скорость восстановления данных, так как за основу берется ближайшая контрольная точка, и восстановление идет от нее.

Помимо этого, можно выделить и недостатки:

* при начальном (первом) создании репозитория и синхронизации его с другими разработчиками, потребуется достаточно длительное время для скачивания данных, особенно, если проект большой, так как требуется скопировать на локальный компьютер весь репозиторий.
* Unix – ориентированность, Git на Microsoft Windows может быть громоздким.

# СПИСОК ИСТОЧНИКОВ

1. Начало работы с Git – Узелки на память – [Электронный ресурс]. Режим доступа: https://tokmakov.msk.ru/blog/item/171
2. Введение - Что такое Git? – Git Book – [Электронный ресурс]. Режим доступа: https://clck.ru/WF9fc