МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

ИНСТИТУТ НЕПРЕРЫВНОГО И ДИСТАНЦИОННОГО ОБРАЗОВАНИЯ

|  |
| --- |
| КАФЕДРА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ ПРЕДПРИНИМАТЕЛЬСТВА |

ОЦЕНКА

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| кандидат технических наук |  |  |  | Е.Л.Турнецкая |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О КОНТРОЛЬНОЙ РАБОТЕ |
|  |
| по дисциплине: Программная инженерия |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | Z6411k |  |  |  | Макаров И.Ю. |
|  | номер группы |  | подпись, дата |  | инициалы, фамилия |
| Студенческий билет № | 2007\1912 | |  |  |  |

Санкт-Петербург 2021

**Цель выполнения работы:** познакомитьсяс технологией распределённой системы управления версиями Git. Научиться загружать данные на сайт GitHub.

**Порядок выполнения работы:**

1. Зарегистрироваться на сервере GitHub.
2. Установить приложение GitHub Desktop.
3. Процессы подготовки программной среды показать на скриншотах.
4. Выложить в хранилище любой файл: код программы, выполненный на любом языке программирования, или текстовый файл, например отчеты всем последующим работы по Программной инженерии .
5. Каждое изменение (новую ветвь, дополнение ррепозитория) фиксировать в системе контроле версий.
6. На скриншотах показать процесс работы с GitHub при внесении изменений в программный код архиватора.

**Описание**

Git — это набор консольных утилит, которые отслеживают и фиксируют изменения в файлах (чаще всего речь идет об исходном коде программ, но также можно использовать его для любых файлов). С его помощью можно откатиться на более старую версию проекта, сравнивать, анализировать, сливать изменения и многое другое. Этот процесс называется контролем версий. Существуют различные системы для контроля версий: SVN, Mercurial, Perforce, CVS, Bitkeeper и другие.

Git является распределенным, то есть не зависит от одного центрального сервера, на котором хранятся файлы. Вместо этого он работает полностью локально, сохраняя данные в папках на жестком диске, которые называются репозиторием. Тем не менее, вы можете хранить копию репозитория онлайн, это сильно облегчает работу над одним проектом для нескольких людей. Для этого используются сайты вроде github и bitbucket.

**Регистрация**

Для того чтобы начать пользоваться Git необходимо пройти регистрацию на сайте перейдя по ссылке <https://github.com>

Для регистрации понадобится имя пользователя, email, пароль

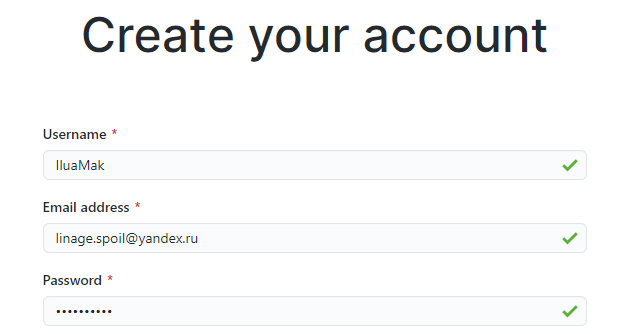


Рисунок 1 – Регистрация на сайте GitHub

После регистрации требуется подтвердить свою почту перейдя по ссылке из письма.

Далее создадим публичный репозиторий с названием «SoftwareEngineering»

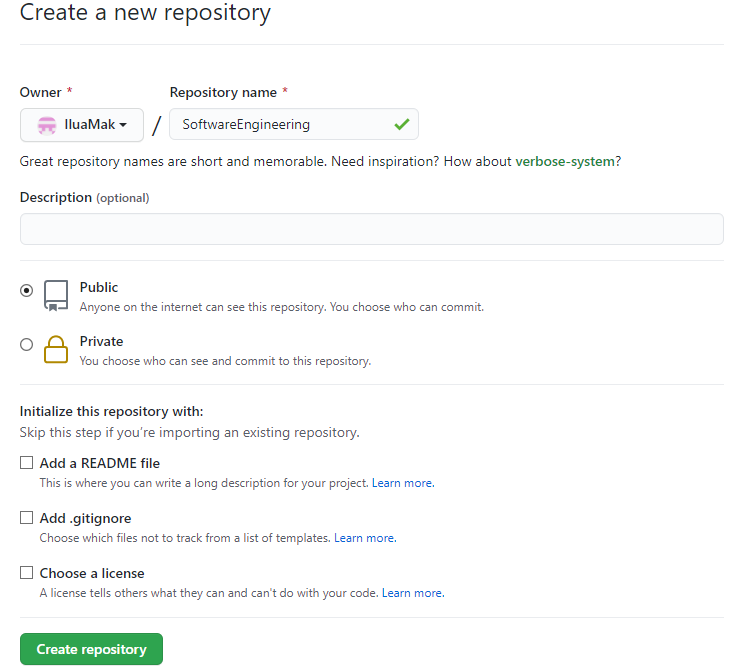


Рисунок 2 – Создание нового репозитория

**Установка приложения**

Переходим по ссылке <https://desktop.github.com> и нажимаем скачать приложение



Рисунок 3 – Сайт приложения GitHub Desktop

При запуске скачанного файла вылезает предложение привязать приложение к аккаунту на GitHub

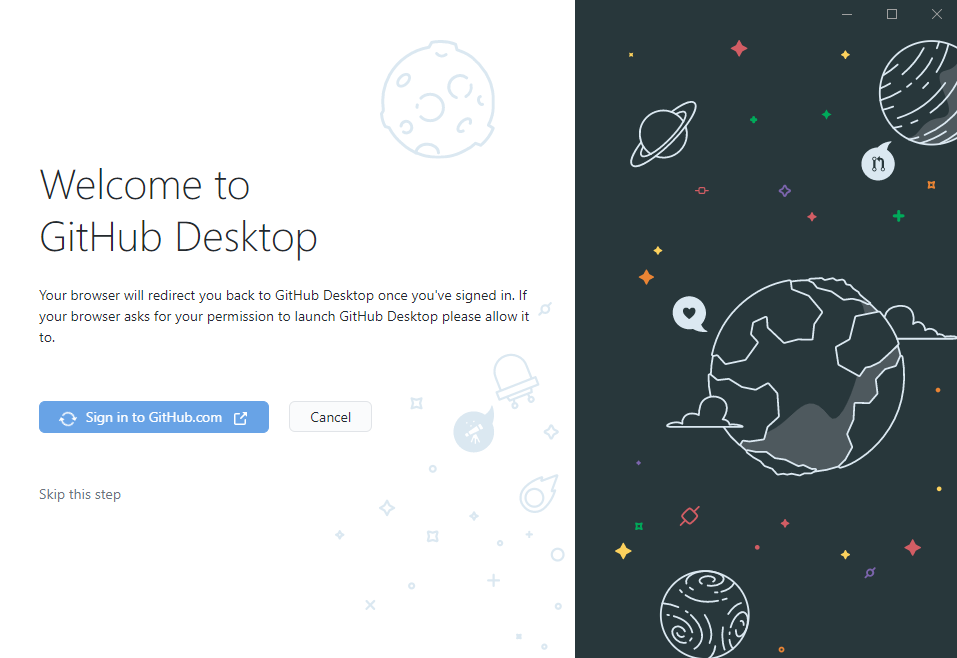


Рисунок 4 – Окно установки приложения

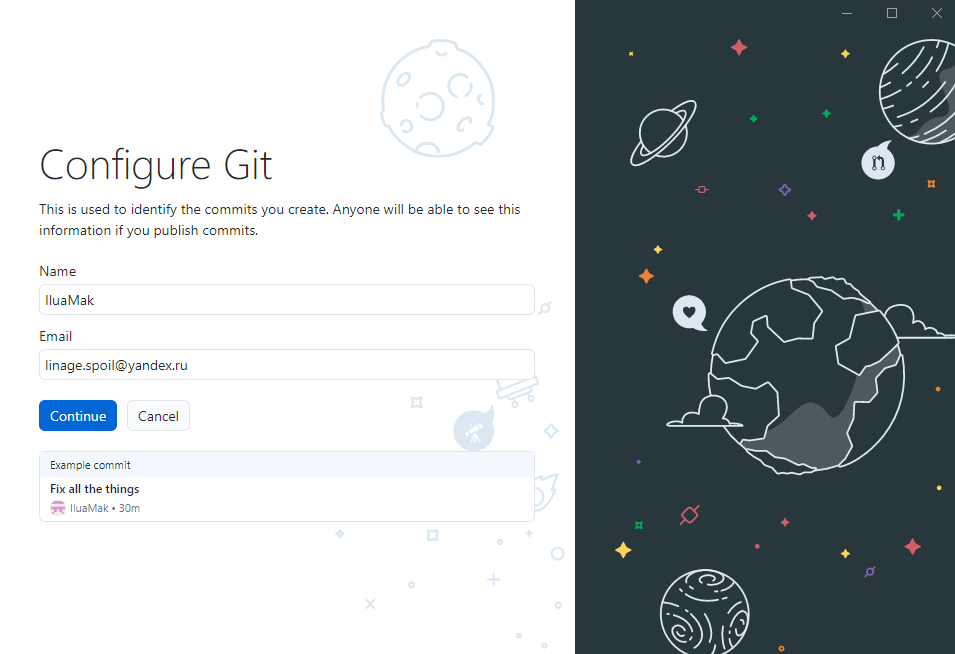


Рисунок 5 – Окно привязки аккаунта

Создадим папку на компьютере для использования Git и укажем ее для клонирования файлов проекта с GitHub

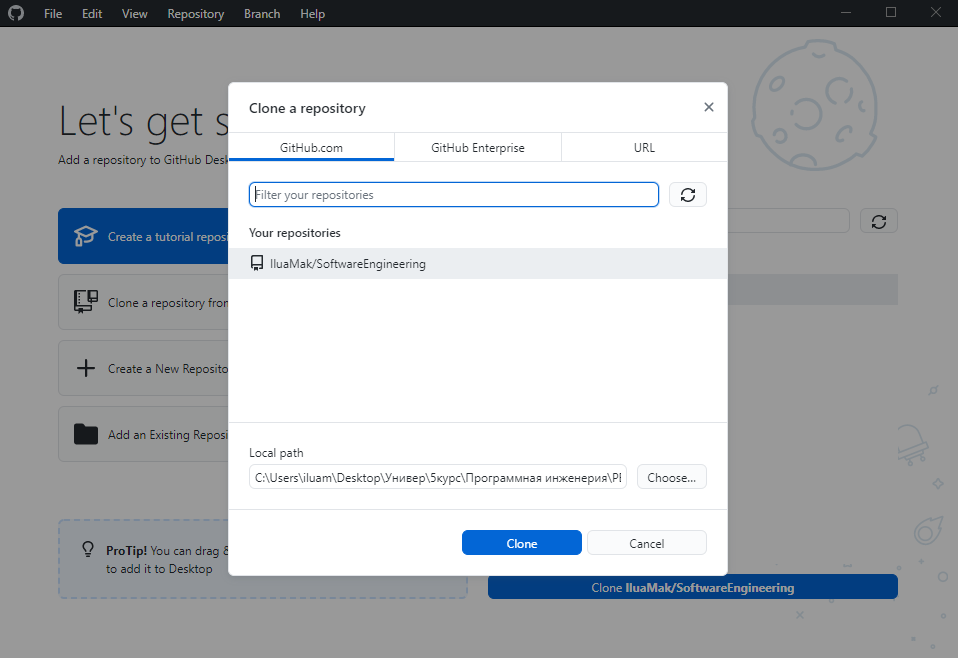


Рисунок 6 – Клонирование репозитория

После синхронизации папки с сайтом GitHub заметим, что в указанной папке создалась скрытая папка «.git» с системными файлами Git’a.

Перенесем незаконченный файл с этой контрольной работой в папку, созданную приложением «SoftwareEngineering», можно заметить, что приложение обнаружило этот файл и может добавить его в нашу ветку. Необходимо добавить описание к этому файлу чтобы избежать путаницы.

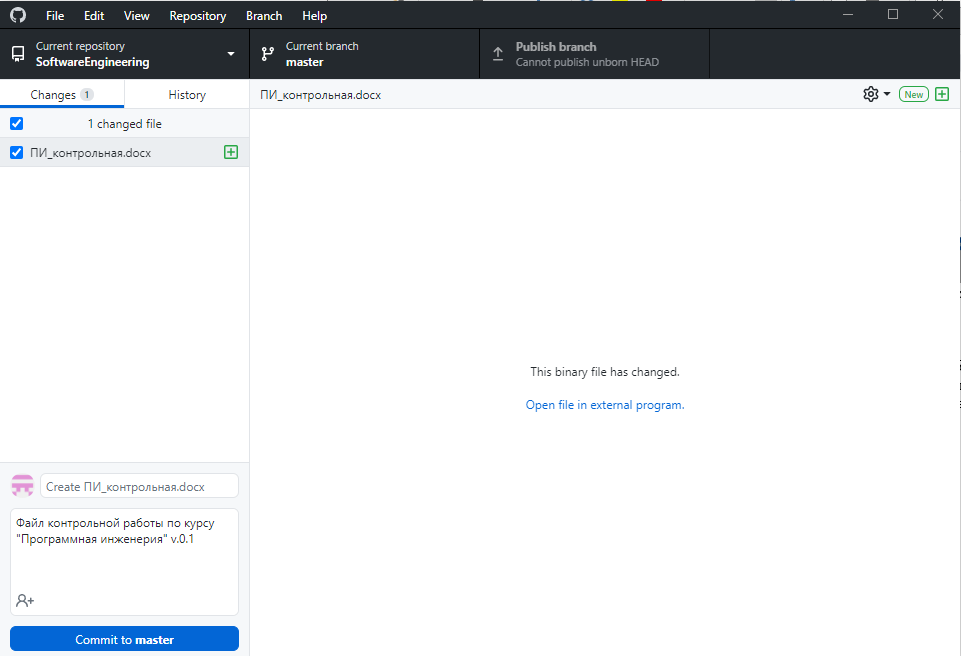


Рисунок 7 – Добавление нового файла

После добавления нового файла нажимаем «Commit to master»

Также добавим файл «README.md» с описанием ветки.

Перейдем на сайт GitHub и увидим что данный файл добавился в нашу ветку

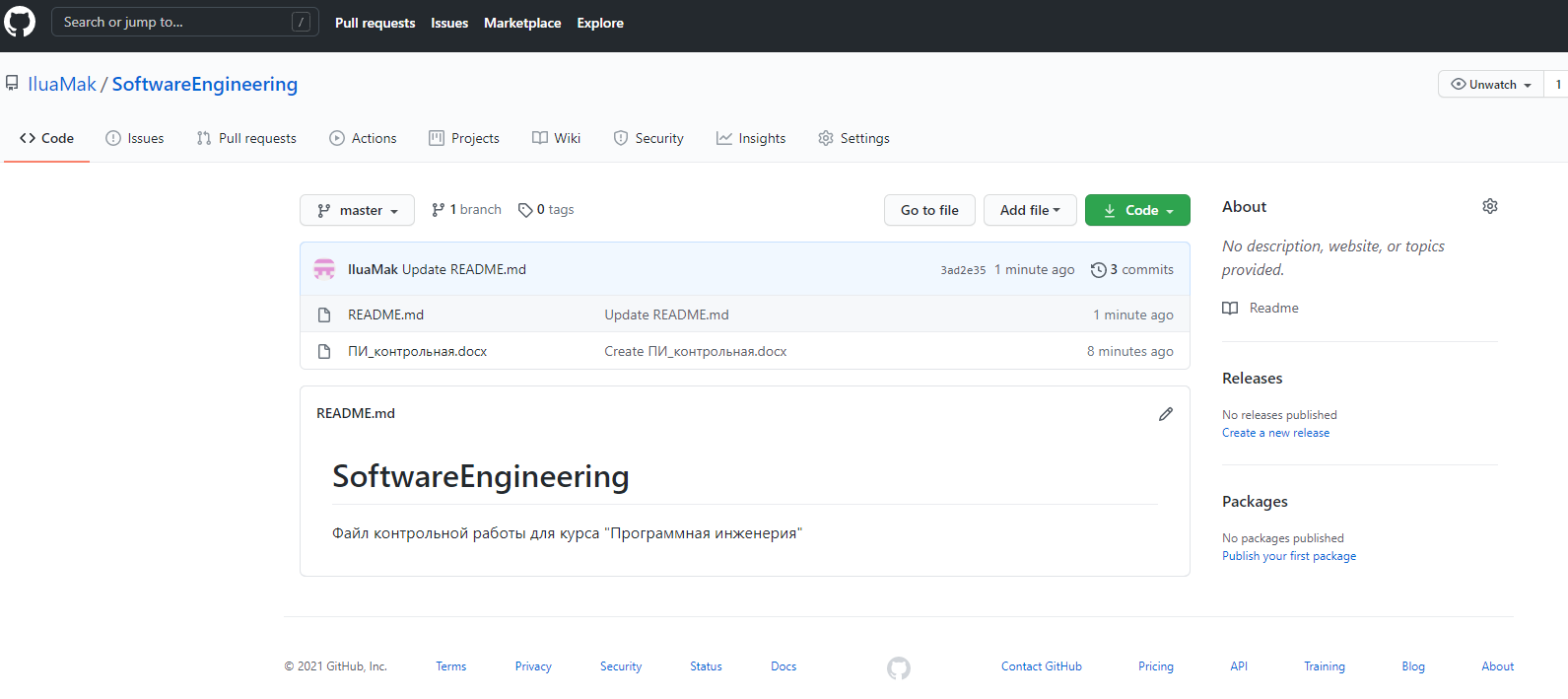


Рисунок 8 – Отображение файлов на сайте GitHub

Сохраним файл контрольной с добавлением последних скриншотов, после чего замети что приложение готово загрузить новую версию на сайт.

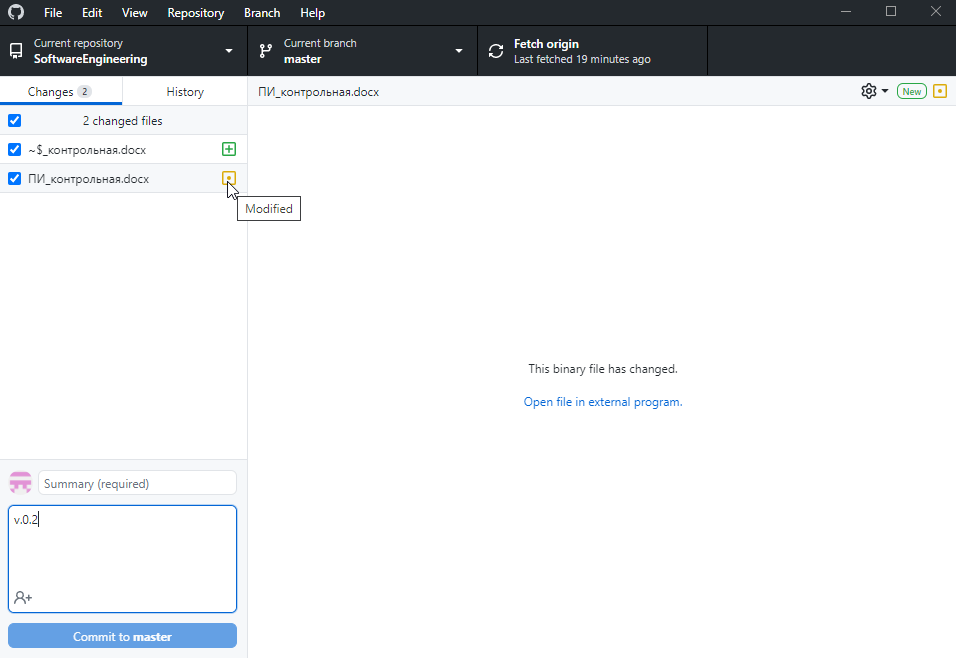


Рисунок 9 – Внесение изменений в файл

Проверим что изменения сохранились в ветке

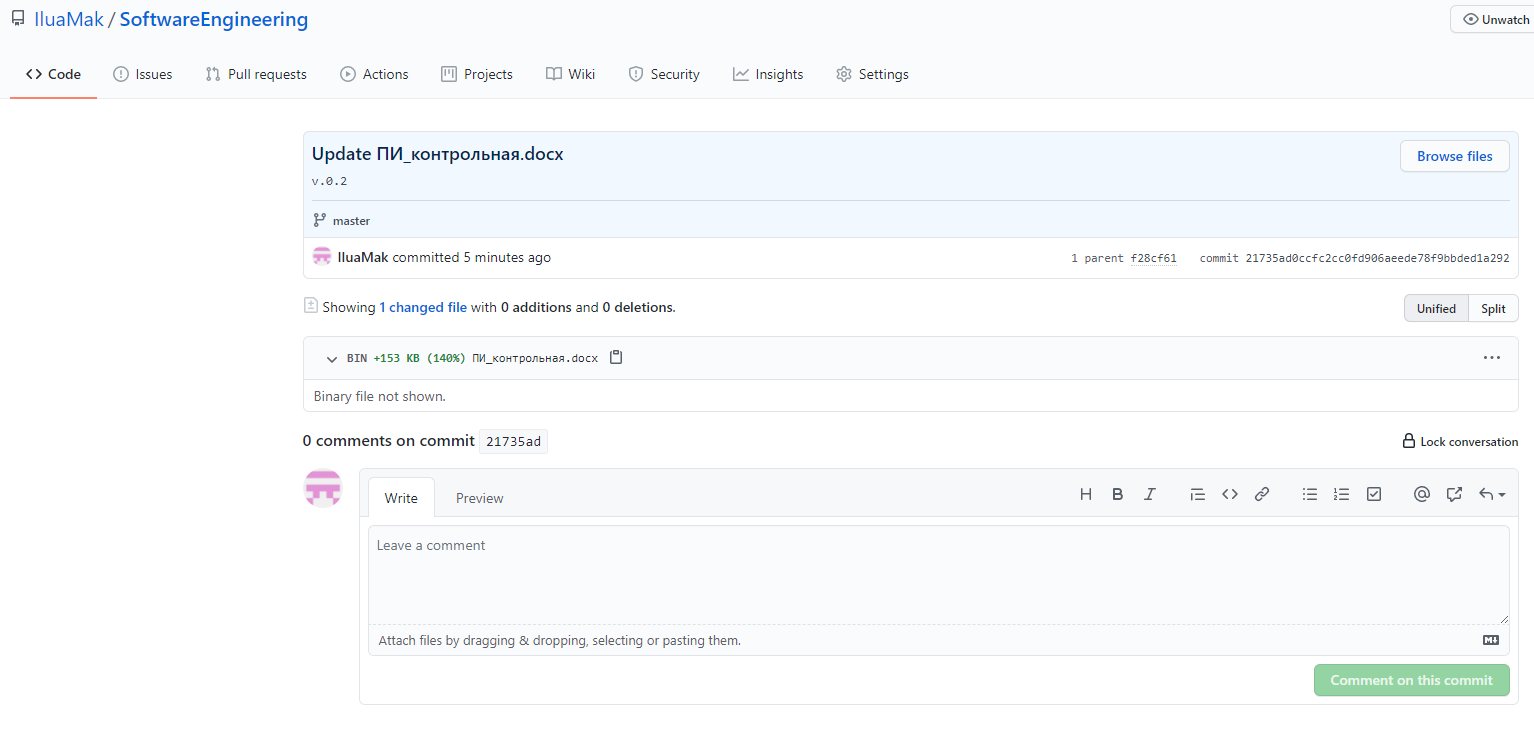


Рисунок 10 – Изменения на сайте

Как видно, на сайте появился наш новый файл, который прибавил в размере 153Кб и имеет новое описание изменения.

**Вывод о проделанной работе**

В данной работе мы познакомились с технологией распределённой системы управления версиями Git. Создали аккаунт на сайте GitHub. Загрузили новый файли добавили в него изменения.

Git — это распределенная система контроля версий нашего кода. Зачем она нам? Для распределенных команд нужна какая-то система управления работы. Нужна, чтобы отслеживать изменения, которые происходят со временем.

То есть шаг за шагом мы видим, какие файлы изменились и как. Особенно это важно, когда анализируешь, что было проделано в рамках одной задачи: это дает возможность возвращаться назад.

Представим себе ситуацию: был работающий код, всё в нем было хорошо, но мы решили что-то улучшить, там подправить, сям подправить. Все ничего, но такое улучшение поломало половину функционала, сделало невозможным работу. И что дальше? Без Гита нужно было бы часами сидеть и вспоминать, как же все было изначально. А так мы просто откатываемся на коммит назад — и все.

Или что делать, если есть два разработчика, которые делают одновременно свои изменения в коде? Без Гита это выглядит так: они скопировали код из оригинала, сделали что нужно. Наступает момент, и оба хотят добавить свои изменения в главную папку. И что делать в этой ситуации? Я даже не берусь оценить время, чтоб проделать эту работу.

Таких проблем не будет вовсе, если пользоваться Гитом.