Вариант 5. Встроенные средства IDE

Работа с командами Git используя терминал VS code

1. **Инициализация репозитория:**

Чтобы создать новый репощиторий ГИТ необходимо ввести команду «git init». При инициализации она создаст папку, которой содержатся все объекты и ссылки, которые Git использует и создаёт в истории работы над проектом.



1. **Статус репозитория**

Чтобы посмотреть статус нужного репозитория необходимо ввести команду «status». Она действует на подготовленные, неподготовленные и неотслеживаемые файлы.



Создание текстового файла с каким-нибудь текстом



Получаем файл



1. **Добавление файлов в область подготовленных файлов**

Команда «add» с указанием имени файла позволяет добавить отдельный файл в область подготовленных файлов.



После выполнения команды мы сразу проверяем статус репозитория и видим, что файл добавился.

1. **Внесение изменений однострочным сообщением или через редактор**

Чтобы добавить комментарий при создании коммита в репозитории можно использовать параметр «commit» с флагом «-m». Само сообщение вводится сразу после флага, в кавычках.



И сразу проверяем статус репозитория



Файл успешно зафиксирован

1. **Просмотр истории коммитов с изменениями:**

Чтобы просматривать изменения, внесённые в репозиторий, воспользуемся параметром «log». Он отображает список последних коммитов в порядке выполнения. А добавив флаг «-p», мы можем подробно изучить изменения, внесённые в каждый файл.



1. **Добавление удаленного репозитория:**

Чтобы добавить удаленный репозиторий, воспользуемся параметром «remote add», указав «имя» и «url» необходимого репозитория.



1. **Отправка изменений в удаленный репозиторий:**

Благодаря параметру «push» с указанием имени репозитория и ветки можно отправлять изменения в удалённый репозиторий . Если же у нас новая ветка, то обязателен флаг «-u».



Далее, перейдя в GitHub, видим, что наш файл сохранился



1. **Получение изменений из удалённого репозитория**

Параметр  «рull» используется для загрузки изменений из удалённого репозитория. Он скачивает копию текущей ветки с указанного удалённого репозитория и объединяет её с локальной копией.



1. **Просмотр изменений до коммита:**

используя параметр «diff», можно просматривать список изменений, внесённых в репозиторий.



Перед применением команды я удалила из файла строчку, а в репозитории этот же файл не содержит данную строчку, поэтому в терминале мы можем увидеть изменения в файлах с момента последнего коммита.

Для просмотра подготовленных изменений необходимо добавить флаг «–staged».

Например, сейчас у нас имеется измененный файл superjama.txt, в котором была удалена строчка. Добавим этот файл в область подготовленных файлов при помощи параметра «add» и сразу же проверим статус репозитория:



Далее используем параметр «diff» с флагом «–staged»



1. **Отмена подготовленных и неподготовленных изменений:**

Восстановить подготовленный файл рабочего дерева можно параметром «reset», и если после этого мы снова попробуем использовать параметр «diff» с флагом «–staged», то терминал вернет пустое значение:



Отменить неподготовленные изменения можно параметром «checkout» с добавлением «--», но для начала проверим статус репозитория:



Как видим, у нас есть измененный файл superjama.txt. Далее вводим команду «git checkout» и ещё раз проверяем статус репозитория:



Теперь в статусе не отображается наш файл readme.txt, а также в файле вернулась строчка «мосты и ты осталишь лишь воспоминания»:



1. **Создание новой ветки и переход в неё:**

Создать новую ветку можно с помощью параметра «branch», указав имя ветки. Но так как Git не переключится на неё автоматически, нужно добавить флаг «-b» и параметр «checkout».



Чтобы узнать в какой ветке мы сейчас находимся, достаточно использовать параметр «branch»:



Для переключения между ветками используется параметр «checkout», но уже без флага:



При создании новой ветки она наследует все файлы от ветки master.

1. **Удаление файлов:**

С помощью параметра «rm» можно удалять файлы из текущего рабочего дерева.

Можно также использовать маски файлов (например \*.txt – удалятся все файлы типа .txt) для удаления всех файлов, соответствующих критерию.





Мы успешно удалили файл superjama.txt.

1. **Слияние двух веток:**

Объединить две ветки можно параметром «merge» указав имени ветки. Данная команда объединит указанную ветку с основной.



1. **Удаление ветки:**

Ненужную ветку можно параметром «branch» с добавлением флага «-d» и указанием имени ветки.

Сначала проверим в какой ветке мы сейчас находимся, а после удалим ветку «student»





Изначально у нас было две ветки «master» и «student», но после удаления одной, отображается только ветка «master».