**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Отчет**

по лабораторной работе №2 «Изучение системы управления версиями Git»

по дисциплине «**Инстрментальные средства разработки ПО**»

Авторы: Триголос А.П

Факультет: ИТиП

Группа: М3204 (M3213)

Преподаватель: Смирнов Станислав Михайлович



Санкт-Петербург 2020

**Цель работы:**

Нужно изучить и описать систему управления версиями Git в объёме не

более пяти страниц, не считая титульника. Формат сдачи, следующий: я

задаю вопросы по основным командам Git, вам нужно дать ответы, опираясь

на ваши отчеты.

За каждый неправильный ответ отнимается один балл от числа максимального

количества баллов за работу.

**Задачи решаемые при выполнении работы:**

1) Первый студент заводит репозиторий, второй делает в нее Pull request.

2) Каждому:

a. Создание веток по модели GitFlow - https://danielkummer.github.io/git-flow-

cheatsheet/index.ru\_RU.html - Обязательное наличие веток фич, релиз, девелоп, хот

фикс.

b. В репозитории обязательно оформлен ReadMe.

c. Наличие тегов.

d. Submodules

e. LFS

3) Также всем студентам обязательно подготовить справочник по основным

командам Git с примерами. Уметь кратко ответить на вопросы о предназначении

основных команд Git.

**Выполнение работы:**

git add

Команда git add добавляет содержимое рабочей директории в индекс (staging area) для последующего коммита. По умолчанию git commit использует лишь этот индекс, так что вы можете использовать git add для сборки слепка вашего следующего коммита.

git status

Команда git status показывает состояния файлов в рабочей директории и индексе: какие файлы изменены, но не добавлены в индекс; какие ожидают коммита в индексе. Вдобавок к этому выводятся подсказки о том, как изменить состояние файлов.

git diff

Команда git diff используется для вычисления разницы между любыми двумя Git деревьями. Это может быть разница между вашей рабочей директорией и индексом (собственно git diff), разница между индексом и последним коммитом (git diff --staged), или между любыми двумя коммитами (git diff master branchB).

git difftool

Команда git difftool просто запускает внешнюю утилиту сравнения для показа различий в двух деревьях, на случай если вы хотите использовать что-либо отличное от встроенного просмотрщика git diff.

git commit

Команда git commit берёт все данные, добавленные в индекс с помощью git add, и сохраняет их слепок во внутренней базе данных, а затем сдвигает указатель текущей ветки на этот слепок.

git reset

Команда git reset, как можно догадаться из названия, используется в основном для отмены изменений. Она изменяет указатель HEAD и, опционально, состояние индекса. Также эта команда может изменить файлы в рабочей директории при использовании параметра --hard, что может привести к потере наработок при неправильном использовании, так что убедитесь в серьёзности своих намерений прежде, чем использовать его.

git rm

Команда git rm используется в Git для удаления файлов из индекса и рабочей директории. Она похожа на git add с тем лишь исключением, что она удаляет, а не добавляет файлы для следующего коммита.

git mv

Команда git mv — это всего лишь удобный способ переместить файл, а затем выполнить git add для нового файла и git rm для старого.

git clean

Команда git clean используется для удаления мусора из рабочей директории. Это могут быть результаты сборки проекта или файлы конфликтов слияний.

git branch

Команда git branch — это своего рода “менеджер веток”. Она умеет перечислять ваши ветки, создавать новые, удалять и переименовывать их.

git checkout

Команда git checkout используется для переключения веток и выгрузки их содержимого в рабочую директорию.

git merge

Команда git merge используется для слияния одной или нескольких веток в текущую. Затем она устанавливает указатель текущей ветки на результирующий коммит.

git fetch

Команда git fetch связывается с удалённым репозиторием и забирает из него все изменения, которых у вас пока нет и сохраняет их локально.

git pull

Команда git pull работает как комбинация команд git fetch и git merge, т. е. Git вначале забирает изменения из указанного удалённого репозитория, а затем пытается слить их с текущей веткой.

git push

Команда git push используется для установления связи с удалённым репозиторием, вычисления локальных изменений отсутствующих в нём, и собственно их передачи в вышеупомянутый репозиторий. Этой команде нужно право на запись в репозиторий, поэтому она использует аутентификацию.

git remote

Команда git remote служит для управления списком удалённых репозиториев. Она позволяет сохранять длинные URL репозиториев в виде понятных коротких строк, например "origin", так что вам не придётся забивать голову всякой ерундой и набирать её каждый раз для связи с сервером. Вы можете использовать несколько удалённых репозиториев для работы и git remote поможет добавлять, изменять и удалять их.

git archive

Команда git archive используется для упаковки в архив указанных коммитов или всего репозитория.

git submodule

Команда git submodule используется для управления вложенными репозиториями. Например, это могут быть библиотеки или другие, используемые не только в этом проекте ресурсы. У команды submodule есть несколько подкоманд — add, update, sync и др. — для управления такими репозиториями.

**Вывод:** ближе познакомился с git’ом, разобрался в основных командах