Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут»

Інститут Прикладного системного аналізу
Кафедра Системного проектування

Лабораторна робота № 1

з дисципліни «Проектування інформаційних систем»

«Системи контролю версій»

Виконав:
студент групи ДА-71
Факультету «ІПСА»
Кузнєцов Олексій

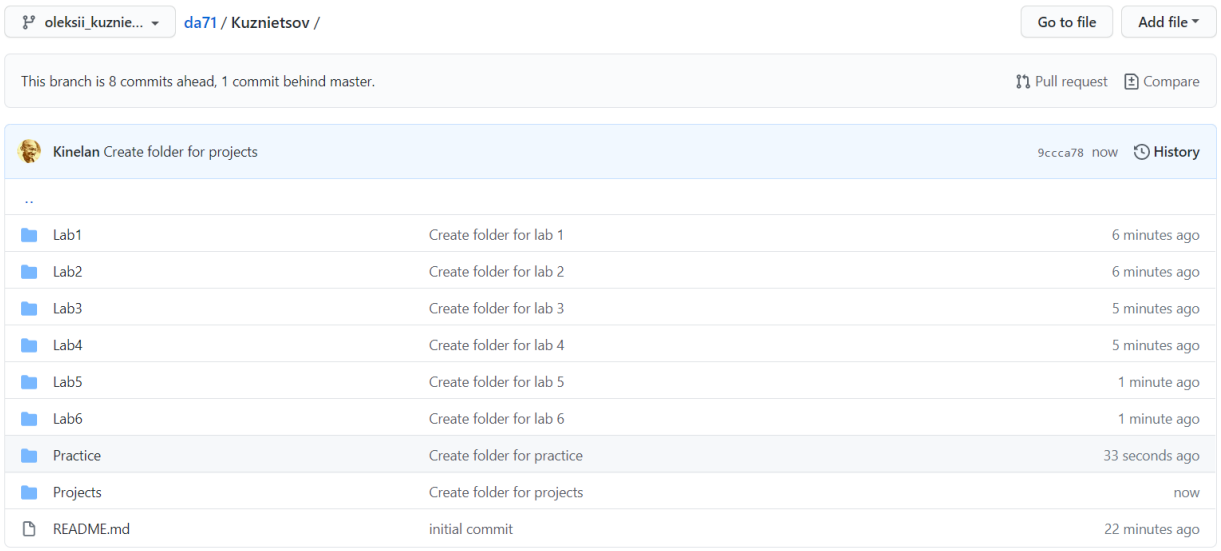
Київ – 2020

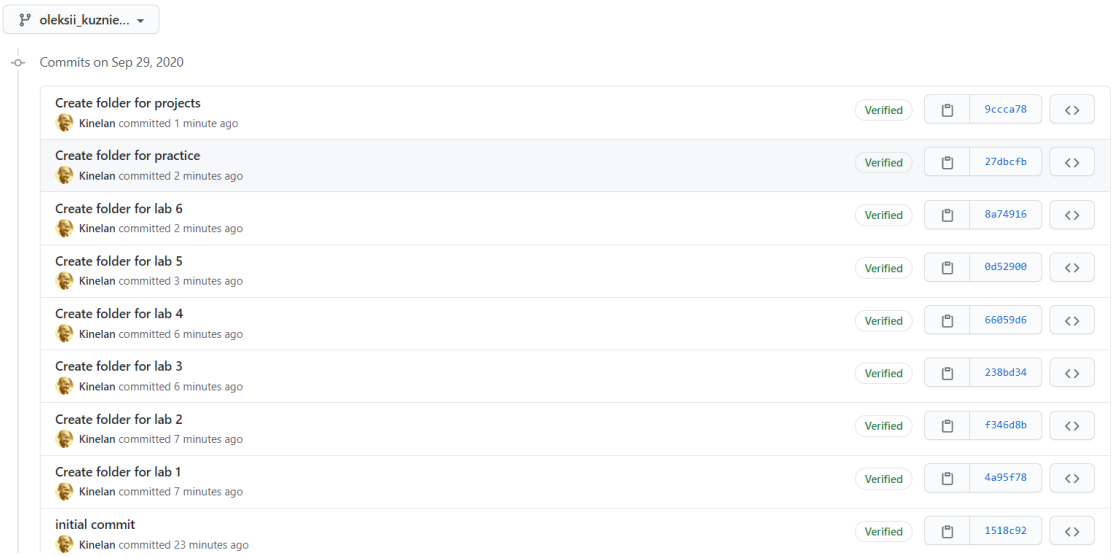
Мета роботи: за допомогою системи контролю версій завантажити коди програми у репозіторій. Відтворити типовий цикл розробки програмного забезпечення з використанням системи контролю версій.

1. **Обрати безкоштовну систему репозиторія для системи контролю версіями, наприклад projectlocker, або інш.**

Для виконання лабораторних робіт було вибрано систему контролю версій Github.

Був створенний репозиторій, у якому є дві гілки: dev and master. В гілці dev буде проводись вся робота.

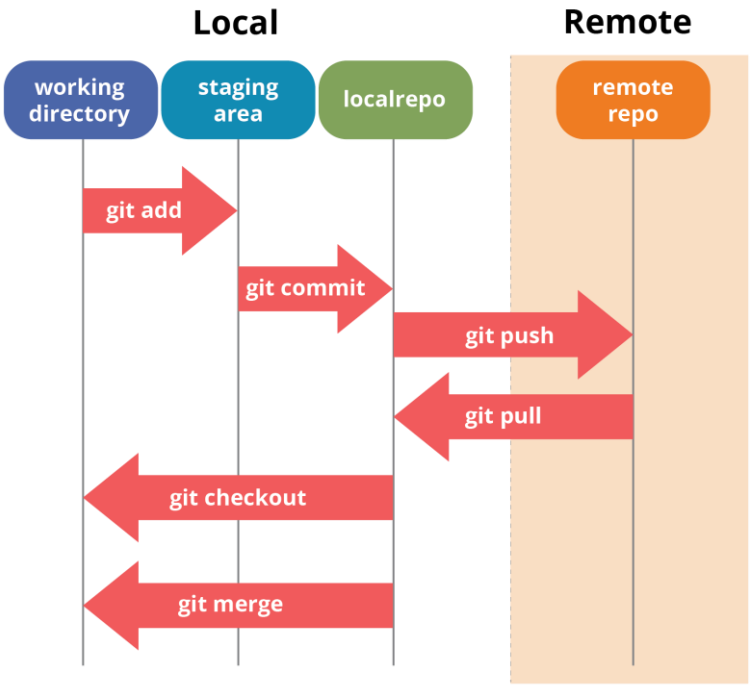




2. **Встановити клієнтське безкоштовне програмне забезпечення для роботи с системою контролю версій (GIT, SVN clients).**

Було встановлено системою контролю версій – GIT.

```
oleks@DESKTOP-MJ7GMD3 MINGW64 ~
$ git --version
git version 2.28.0.windows.1
```



3.1. Опис команд, які використовувалися протягом виконання роботи з системою контролю версіями.

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 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 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 log

Команда git log используется для просмотра истории коммитов, начиная с самого свежего и уходя к истокам проекта. По умолчанию, она показывает лишь историю текущей ветки, но может быть настроена на вывод истории других, даже нескольких сразу, веток. Также её можно использовать для просмотра различий между ветками на уровне коммитов.

git tag

Команда git tag используется для задания постоянной метки на какой-либо момент в истории проекта. Обычно она используется для релизов.

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.

4. Різниця між GIT та SVN

- Git - это распределенная VCS; SVN является нераспределенной VCS.
- Git имеет централизованный сервер и репозиторий; У SVN нет централизованного сервера или хранилища.
- Содержимое в Git хранится в виде метаданных; SVN хранит файлы контента.
- С Git-ветками легче работать, чем с SVN-ветками.

- У Git нет функции глобального номера ревизии, как у SVN.
- Git имеет лучшую защиту контента, чем SVN.
- Git был разработан для ядра Linux Линусом Торвальдсом; SVN был разработан CollabNet, Inc.
- Git распространяется под GNU, а его обслуживание контролируется Junio Hamano; Apache Subversion, или SVN, распространяется по лицензии с открытым исходным кодом