# Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського» Факультет інформатики та обчислювальної техніки Кафедра інформаційних систем та технологій

# Лабораторна робота № 1

з дисципліни «Технології розроблення програмного забезпечення»

Тема: «Git»

Виконав: студент групи - IA-32 Непотачев Іван Дмитрович

Перевірив: Мягкий Михайло Юрійович

#### Зміст

Теоретичні відомості	2
Основні поняття	
Створення комітів	
Робота з гілками	
Додавання файлів у різні гілки	3
Хід роботи	
Висновки	

### Теоретичні відомості

Git – це розподілена система контролю версій, яка дозволяє зберігати історію змін у проєкті, працювати з різними версіями файлів і організовувати спільну роботу декількох розробників.

Чому Git незамінний у роботі розробника:

Контроль версій – будь-які зміни в коді зберігаються, і завжди можна повернутися до попередньої версії.

Командна робота – кілька розробників можуть одночасно працювати над одним проєктом без ризику втратити важливі дані.

Можливість тестування — можна створювати окремі гілки для нових функцій, тестувати їх і лише потім додавати в основний код.

#### Основні поняття

- Репозиторій сховище проєкту з усіма файлами та історією змін. Створюється командою: git init
- Коміт (commit) знімок (версія) стану проєкту на певний момент часу. Коміти утворюють історію змін. git commit -m "Опис змін"
- Гілка (branch) незалежна лінія розвитку проєкту. Використовується для паралельної роботи над різними завданнями.

Перегляд гілок: git branch

Перемикання: git switch branch\_name або git checkout branch\_name

• Основна гілка – зазвичай називається main або master. В ній зберігається стабільна версія проєкту.

#### Створення комітів

Git відслідковує зміни у два етапи:

- 1. Індекс (staging area) проміжна зона, куди додають змінені файли: git add file.txt
- 3. Фіксація змін створення коміту з усім, що додано в індекс: git commit -m "Додано файл file.txt"

Також можливий порожній коміт (без файлів):

git commit --allow-empty -m "Initial commit"

#### Робота з гілками

Є кілька способів створити нову гілку:

- 1. git branch b1 створює гілку b1, але не переходить у неї.
- 2. git checkout -b b2 створю $\epsilon$  гілку b2 і одразу переходить.
- 3. git switch -c b3 сучасна команда для створення і переходу.

Перевірити поточні гілки можна так: git branch

# Додавання файлів у різні гілки

Завдяки гілкам можна зберігати у кожній версії проєкту свої файли. Наприклад:

- у гілці b1 файл файл 1 а.txt;
- у гілці b2 файл файл 2 a.txt;
- y гілці b3 файл файл 3 с.txt.

Це дає можливість паралельно розробляти різні частини програми.

# Хід роботи

1. Створення нового каталогу та ініціалізація Git репозиторію

mkdir 1

git init

# 2. Створення першого порожнього коміту

git commit --allow-empty -m 'init'

Перевірка гілок:

git branch

## 3. Створення трьох гілок різними способами

```
git checkout -b b1 # спосіб 1: створити і перейти
```

git checkout main

git branch b2 # спосіб 2: створити гілку без переходу

git switch -c b3 # спосіб 3: створити і перейти

git checkout main

### 4. Створення трьох файлів

touch f1.txt f2.txt f3.txt

# 5. Редагування файлів

nano fl.txt

nano f2.txt

nano f3.txt

## 6. Розміщення файлів у відповідних гілках

Гілка b1:

git checkout b1

git add f1.txt

git commit -m 'f1'

В коміті гілки b1 знаходиться тільки f1.txt.

```
Гілка b2:
git checkout b2
git add f2.txt
git commit -m 'f2'

• В коміті гілки b2 знаходиться тільки f2.txt.
Гілка b3:
git checkout b3
git add f3.txt
git commit -m 'f3'

• В коміті гілки b3 знаходиться тільки f3.txt.
```

7. Додавання файлу fl.txt у гілку b3

```
echo '123' > f1.txt
git add f1.txt
git commit -m 'f1'
```

8. **Злиття гілки** b1 y b3

```
git merge b1
```

Виник конфлікт у файлі f1.txt (обидві гілки додали файл).

Конфлікт вирішено вручну

Після вирішення:

git add fl.txt

git commit -m 'resolved merge conflicts'

# 9. Перевірка історії комітів

git log --graph

Граф показує всі коміти з гілками і злиттям, включно з вирішенням конфліктів.

#### Висновки

Висновки: під час виконання лабораторної роботи, я навчився працювати с системою контролю версій git.