Основи тестування програмного забезпечення

Лабораторна робота №1

**Робота з GIT**

**Мета роботи**: Використовуючи теоретичне підґрунтя про роботу за git виконати дії що будуть вказано в завданні до лабораторної роботи.

**Теоретичні відомості**

Розроблені вказівки ставлять за мету надати первинне уявлення про можливості використання системи контролю версій **git**.

Вказівки розбити на три основні секції: встановлення програмного забезпечення, що дозволяє використовувати систему контроля версій **git**; створення та використання репозиторію на сервері **github**; практичні приклади використання під час розробки **власного** програмного забезпечення.

**Встановлення програмного забезпечення (git клієнта)**

1. Скачайте з офіційного сайту <https://git-scm.com/> інсталятор.

A computer monitor with a blue screen

Description automatically generated

Рисунок 1 – Приклад інтерфейсу

1. Запустіть інсталятор на виконання.
2. Використовуйте всі опції за замовченням окрім останнього пункту. В останньому пункті виберіть None.

A screenshot of a computer

Description automatically generated

Рисунок 2 – Приклад інтерфейсу

1. Програмне забезпечення буде встановлено.
2. Натисніть правою кнопкою миші в будь якій частині екрану та вибуріть наступний пункт.

A screenshot of a computer

Description automatically generated

Рисунок 3 – Приклад інтерфейсу

1. Відкриється термінал для роботи. Далі в документі, коли буде зустрічатись фраза «відкрийте гіт термінал» означитиме виконання пункту 5 у вказаній директорії.

A screenshot of a computer

Description automatically generated

Рисунок 4 – Приклад інтерфейсу

1. Створіть пару SSH ключів виконавши команду

|  |
| --- |
| ssh-keygen -t ed25519 -C [your\_email@example.com](mailto:your_email@example.com) |

A screen shot of a computer

Description automatically generated

Рисунок 5 – Приклад інтерфейсу

1. **Зверніть увагу** на те куди були збережені ключі, вони нам знадобляться для конфігурації **github** акаунта.

**Сервер контролю версій github**

1. Створіть аккаунт на <https://github.com/> та залогіньтесь.
2. Перейдіть в конфігурацію користувача:

A screenshot of a computer

Description automatically generated

Рисунок 6 – Приклад інтерфейсу

1. Перейдіть в розділ SSH and GPG keys

A close up of a logo

Description automatically generated

Рисунок 7 – Приклад інтерфейсу

1. Натисніть кнопку Додати новий ключ

A green rectangular sign with white text

Description automatically generated

Рисунок 8 – Приклад інтерфейсу

1. Вкажіть назву ключа та вставте вміст згенерованого файлу id\_ed25519.pub, що був згенерований в пункті 7 попереднього розділу.
2. Перейдіть на головну сторінку **github**.
3. Натисніть кнопку створення нового репозиторію.

A screenshot of a computer

Description automatically generated

Рисунок 9 – Приклад інтерфейсу

1. Створіть новий репозиторій. Вкажіть ім’я. Виберіть Public як специфікатор доступу. Відмітьте пункт додавання README файлу. Натисніть кнопку

A green rectangle with white text

Description automatically generated

Рисунок 10 – Приклад інтерфейсу

1. Після створення репозиторія відбудеться перенаправлення на цей репозиторій

A screenshot of a computer

Description automatically generated

Рисунок 11 - Приклад інтерфейсу

**Перші кроки з використання git**

1. На локальному комп’ютері створіть директорії та перейдіть в неї

A screenshot of a computer

Description automatically generated

Рисунок 12 – Приклад інтерфейсу

1. Запустіть git термінал через контекстне меню в цій директорії. В пункті **6** частини про встановлення програмного забезпечення детально вказано як запустити git термінал.

A screenshot of a computer

Description automatically generated

Рисунок 13 – Приклад інтерфейсу

1. На веб інтерфесі github відкрийте створений репозиторій та натисніть кнопку

A green rectangle with white text

Description automatically generated

Рисунок 14 - Приклад інтерфейсу

1. У вікні, що відкриється виберіть SSH та натисніть кнопку копіювання посилання.

A screenshot of a computer

Description automatically generated

Рисунок 15 – Приклад інтерфейсу

1. В буфері обміну буде збережено наступну інформацію.

|  |
| --- |
| git@github.com:idyuba/testRepo.git |

1. Поверніться в git термінал та виконайте команду **клонування репозиторію**. Необхідна адреса на даний час збережена в буфері обміну.

|  |
| --- |
| git clone git@github.com:idyuba/testRepo.git |

A screenshot of a computer

Description automatically generated

Рисунок 16 - Приклад інтерфейсу

1. Після успішного клонування з сервера github з’явиться директорія в якій буде розміщена локальна копія репозиторія, що був створений на github.
2. Перейдіть в склоновону директорію як через провідник так і через гіт термінал

A screenshot of a computer

Description automatically generated

Рисунок 17 – Приклад інтерфейсу

1. **Увага!** Всі git команди про які буде йти мова в наступному розділі мають виконувати тільки в директорії, що була створена під час клонування.

**Початок роботи**

1. Створіть новий файл index.html.

A screenshot of a computer

Description automatically generated

Рисунок 18 – Приклад інтерфейсу

1. Для першої спроби роботи з github цього достатньо.
2. У терміналі виконайте команду git status

|  |
| --- |
| git status |

A screenshot of a computer screen

Description automatically generated

Рисунок 19 – Приклад інтерфейсу

1. Як видно з малюнку вище створений файл index.html відмічений таким чином, що файл присутній проте система не знає для чого він потрібен. Також пропонується команда git add для того, щоб відмітити файл.
2. Виконайте команду

|  |
| --- |
| git add index.html |

Та знову

|  |
| --- |
| git status |

A screenshot of a computer

Description automatically generated

Рисунок 20 – Приклад інтерфейсу

1. Тепер git знає, що файл буде використовуватись. Зверніть увагу що після виконання команди git add файл index.html став відмічений як новий. Це цілком зрозуміло, бо файл був створений в попередніх пунктах на локальному комп’ютері, та сервер про цей файл нічого поки не знає.
2. Після того, як файл був відмічений, необхідно сформувати комміт команду зі зрозумілим повідомленням.

|  |
| --- |
| git commit -m "add index.html" |

A computer screen shot of a computer screen

Description automatically generated

Рисунок 21 - Приклад інтерфейсу

1. Створений комміт необхідно відправити на github сервер.

|  |
| --- |
| git push |

A screenshot of a computer screen

Description automatically generated

Рисунок 22 – Приклад інтерфейсу

1. Перейдіть на веб інтерфейс github та переконайтесь, що файл index.html присутній на сервері.

A screenshot of a computer

Description automatically generated

Рисунок 23 – Приклад інтерфейсу

1. Створіть ще два файли main.cpp та backend.py та виконайте повторно всі дії для розміщення цих файлів на github.

A screenshot of a computer

Description automatically generated

Рисунок 23 – Приклад інтерфейсу

|  |
| --- |
| git add main.cpp backend.py  git status  git commit -m "add main.cpp and backend.py"  git push |

A screenshot of a chat

Description automatically generated

Рисунок 24 – Приклад інтерфейсу

1. Перегляньте історію коммітів на веб інтерфейсі

A screenshot of a computer

Description automatically generated

Рисунок 25 – Приклад інтерфейсу

A screenshot of a chat

Description automatically generated

Рисунок 26 – Приклад інтерфейсу

1. Кожен комміт зберігає всю інформацію про зміни, що були внесені.

**Збереження змін під час роботи**

1. Відкрийте файл backend.py та додайте декілька рядків коду. Збережіть файл.

A screenshot of a computer screen

Description automatically generated

Рисунок 27 – Приклад інтерфейсу

1. Перейдіть в git термінал та виконайте команду

|  |
| --- |
| git status |

A screenshot of a computer screen

Description automatically generated

Рисунок 28 – Приклад інтерфейсу

1. Файл backend.py цього разу відмічений як змінений, а не як невідомий файл. Git пропонує дві опції: зберегти зміни використовуючи git add чи відмінити зміни використовуючи git restore.
2. Виконайте наступні команди для того, щоб зберігти зміни, та переконатись, що зміни збережені.

|  |
| --- |
| git add backend.py  git status |

A computer screen shot of a computer program

Description automatically generated

Рисунок 30 – Приклад інтерфейсу

1. Сформуйте комміт з відповідним повідомленням та відправте зміни на git сервер.

|  |
| --- |
| git commit -m "update backed.py"  git push |

A computer screen shot of a computer program

Description automatically generated

Рисунок 31 – Приклад інтерфейсу

1. Перейдіть на веб інтерфейс та відкрийте файл backed.py. Зміни, що були виконані на локальному комп’ютері відображаються на сервері.

A screenshot of a computer

Description automatically generated

Рисунок 32 -

1. Перейдіть в список коммітів на веб інтерфейсі та відкрийте комміт в якому були внесені зміни до файлу backend.py. Переконайтесь що і повідомлення і зміни відповідають тим, що вносились на локальному комп’ютері.

A screenshot of a computer

Description automatically generated

Рисунок 33 – Приклад інтерфейсу

**Завдання до лабораторної роботи**

1. Ознайомитись з інструкцією по базовому використанню git.
2. Виконати всі підготовчі дії для використання git на локальному комп’ютері.
3. Створити акаунт на github у випадку відсутності.
4. Створити репозиторій що має назву яка задовольняє наступному шаблону **QA-KB-21{1 or 2}-Name-Surname** (наприклад QA-KB213-Ihor-Diuba)
5. Склонувати репозиторій на локальний компьютер.
6. Виконати певну кількість дій з файлами: створення нового файлу, зміна фала, видалення файлу у склоновоному репозиторії. Кожну дію з файлами оформити в окремий коміт. Всю історію комітів завантажити на github.
7. Сформувати звіт, який описує всі виконані дії з прикріпленими скрінами.
8. Надати посилання на створений репозиторій в якому відображається список комітів для перевірки.