ЛАБОРАТОРНА РОБОТА

**«Використання Git та GitHub у JavaScript-проєктах»**

## 🎯 Мета роботи

Навчитися:

- використовувати Git для контролю версій JavaScript-проєктів;
- працювати з локальним та віддаленим репозиторіями;
- створювати та використовувати гілки;
- публікувати проєкт на GitHub.

## 🧰 Обладнання та ПЗ

- ОС: Windows
- Git
- GitHub (акаунт)
- VS Code
- Node.js (необов’язково, для простих JS-проєктів)
- Браузер Google Chrome

## 📂 Завдання

1. Створити JavaScript-проєкт
2. Ініціалізувати Git-репозиторій
3. Створити репозиторій на GitHub
4. Створити окрему гілку для нової функції
5. Додати JavaScript-файл у гілці
6. Злити гілку з main
7. Опублікувати проєкт на GitHub

## 📝 Хід роботи

### 🔷 Крок 1. Створення JavaScript-проєкту

Створити папку проєкту:

mkdir js-git-lab

cd js-git-lab

Створити файл index.html:

```
<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="UTF-8">
  <title>Git + JS</title>
</head>
<body>
  <h1>Hello GitHub</h1>
  <script src="script.js"></script>
</body>
</html>
```

Створити файл script.js:
*console.log("Hello Git and GitHub!");*
📸 **Скріншот 1:** структура проєкту у VS Code

### 🔷 Крок 2. Ініціалізація Git
У папці проєкту виконати:
git init
git add .
git commit -m "Initial commit"
📸 **Скріншот 2:** результат git status та git commit

### 🔷 Крок 3. Файл .gitignore
Створити файл .gitignore:
node_modules
.env
*.log

### 🔷 Крок 4. Створення репозиторію на GitHub
1. Увійти на GitHub
2. Натиснути **New repository**
3. Назва: js-git-lab
4. Public
5. Create repository

Підключення:
git branch -M main
git remote add origin https://github.com/USERNAME/js-git-lab.git
git push -u origin main
📸 **Скріншот 3:** репозиторій на GitHub

### 🔷 Крок 5. Створення гілки
Створити гілку для нової функції:
git checkout -b feature-alert
📸 **Скріншот 4:** команда git branch

### 🔷 Крок 6. Робота в гілці
Відредагувати script.js:
alert("This is feature branch!");
Зберегти зміни:
git add script.js
git commit -m "Add alert feature"
📸 **Скріншот 5:** commit у гілці feature-alert

### 🔷 Крок 7. Злиття гілки

git switch main
git merge feature-alert
📸 **Скріншот 6:** результат merge

### 🔷 Крок 8. Оновлення GitHub
git push
📸 **Скріншот 7:** оновлений код на GitHub

### ☑ Результат роботи
- створено JavaScript-проєкт;
- ініціалізовано Git-репозиторій;
- створено гілку для нової функції;
- виконано злиття гілок;
- проєкт опубліковано на GitHub.

### ❓ Контрольні питання
1. Що таке Git?
2. Що таке GitHub?
3. Для чого використовують гілки?
4. Чим відрізняється git commit від git push?
5. Які файли не варто додавати в Git?