***Кафедра комп’ютерної та програмної інженерії***

**ЛАБОРАТОРНА РОБОТА №3**

**з курсу**

**«Універсальна кроссплатформна розробка на платформі .Net Core»**

**Студент: Перловська А. О.**

**Група: ІПЗ 42**

**Факультет: Математики, інформатики**

**та фізики**

**Перевірив: Січкаренко В. О.**

**Лабораторна робота №3**

**Тема:** Основні операції в системі контролю версій Git.

**Мета:** ознайомитись та навчитись використовувати основні команди системи контролю версій Git.

**Завдання:**

1. Створити відкритий (public) репозиторій на хостингу GitHub;
2. Клонувати на локальний комп’ютер (використовуючи команду git clone);
3. Створити в локальному репозиторії гілку (команда git checkout -b);
4. Додати в репозиторій файли з кодом або створити тестове рішення (можна використовувати будь-яку IDE);
5. Індексувати файли або рішення в локальному репозиторії (команда git add);
6. Зафіксувати зміни (команда git commit);
7. Надіслати запит на зміни у віддалений репозиторій (команда git push);
8. Поєднати код з запиту на зміни з головною гілкою (merge pull request);

**Хід роботи**

Оскільки я вже маю акаунт на GitHub та система контролю версій вже встановлено, то я одразу переходжу до роботи.

Обираю папку, в яку хочу клонувати репозиторій в мене це – С/Users/ANNA (рис. 1).

Далі створюю новий репозиторій на сайті, задаю йому певні параметри та вказую ім’я Nightshe2. За допомогою готового посилання клоную репозиторій, а також роблю перехід до нової гілки (рис. 2).

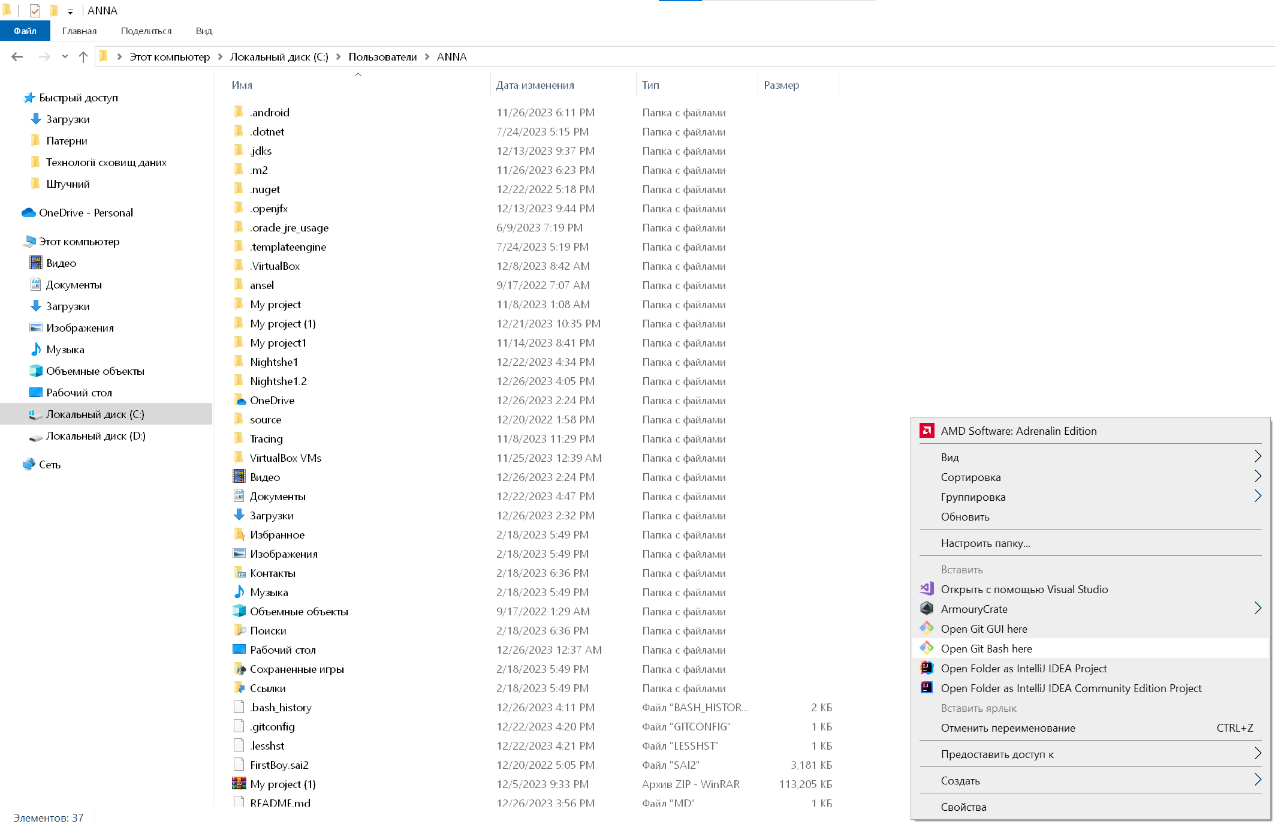


Рисунок 1 – Початок роботи з Git Bash

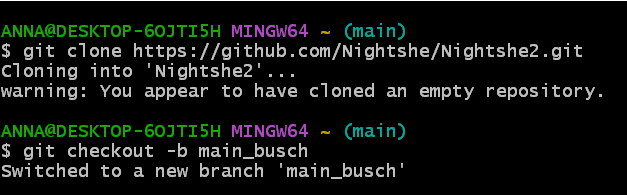


Рисунок 2 – Клонування + нова гілка

Далі я додаю вже існуючі файли до репозиторію на комп’ютері (рис. 3).

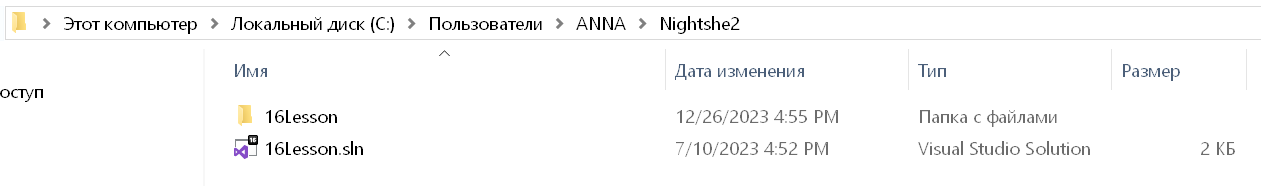


Рисунок 3 – Додаткові файли

Наступним чином я додаю усі файли, які знаходяться в папці, в індекс, а потім фіксую зміни (рис. 4). Для цього мені потрібно застосувати подану команду:

**git commit -m "message\_commit"**

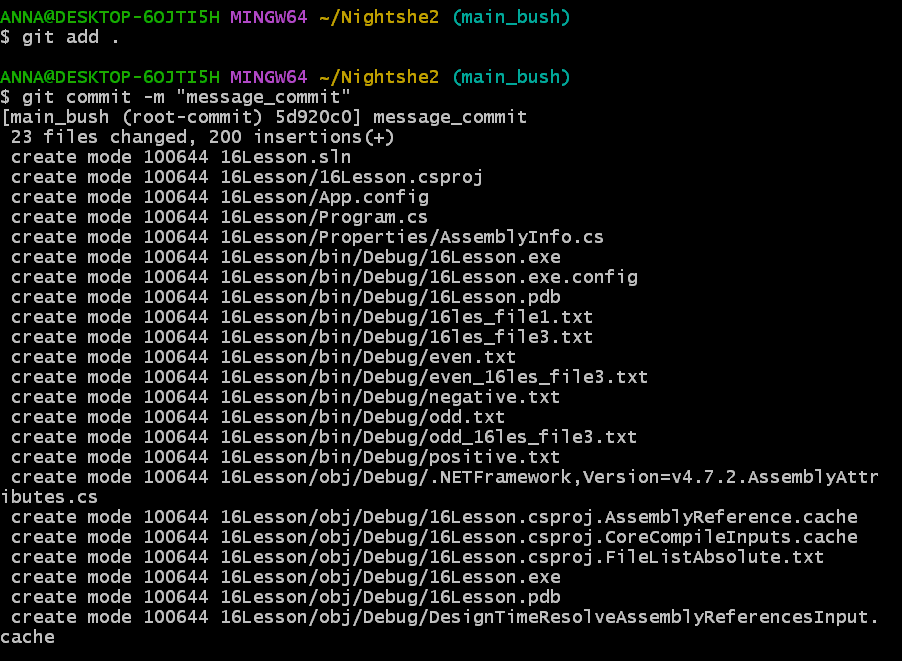


Рисунок 4 – Додавання файлів

Далі я відправляю зміни у віддалений репозиторій (рис. 5). Для цього я використовую назву гілки, яку я вводила раніше.

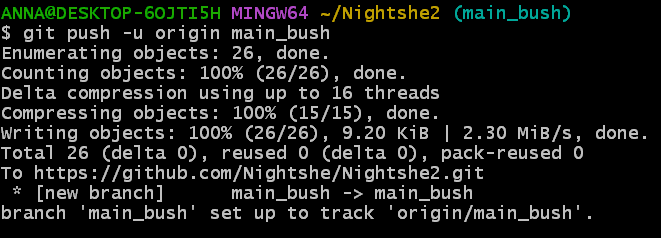


Рисунок 5 – Зміни у віддаленому репозиторію

Потім я працюю з злиттям коду з запитом на зміни. Для цього я перемикаюсь на іншу гілку, адже їх потрібно буде порівнювати, і вони повинні бути різні. (рис. 6)

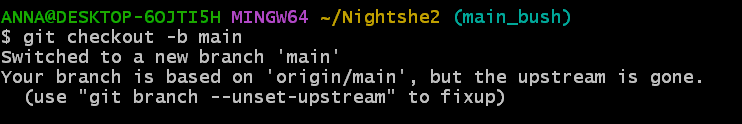


Рисунок 6 – Переключання на іншу гілку

Я додаю ще один файл, який буде показувати різницю між гілками та оброблюю ці зміни (рис. 7).

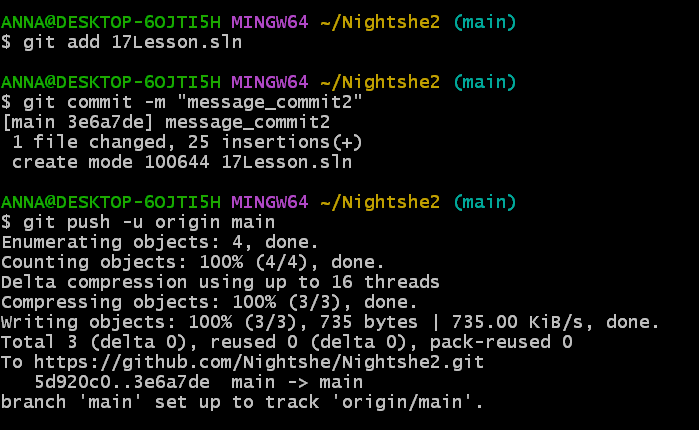


Рисунок 7 – Обробка інших файлів

Я переходжу на вкладку «Pull request», обираю дві різні гілки, які буду порівнювати (в моєму випадку це «main\_bush» та «main»). Після цього я натискаю «Create pull request», якщо файли однакові – їх неможливо буде порівняти.

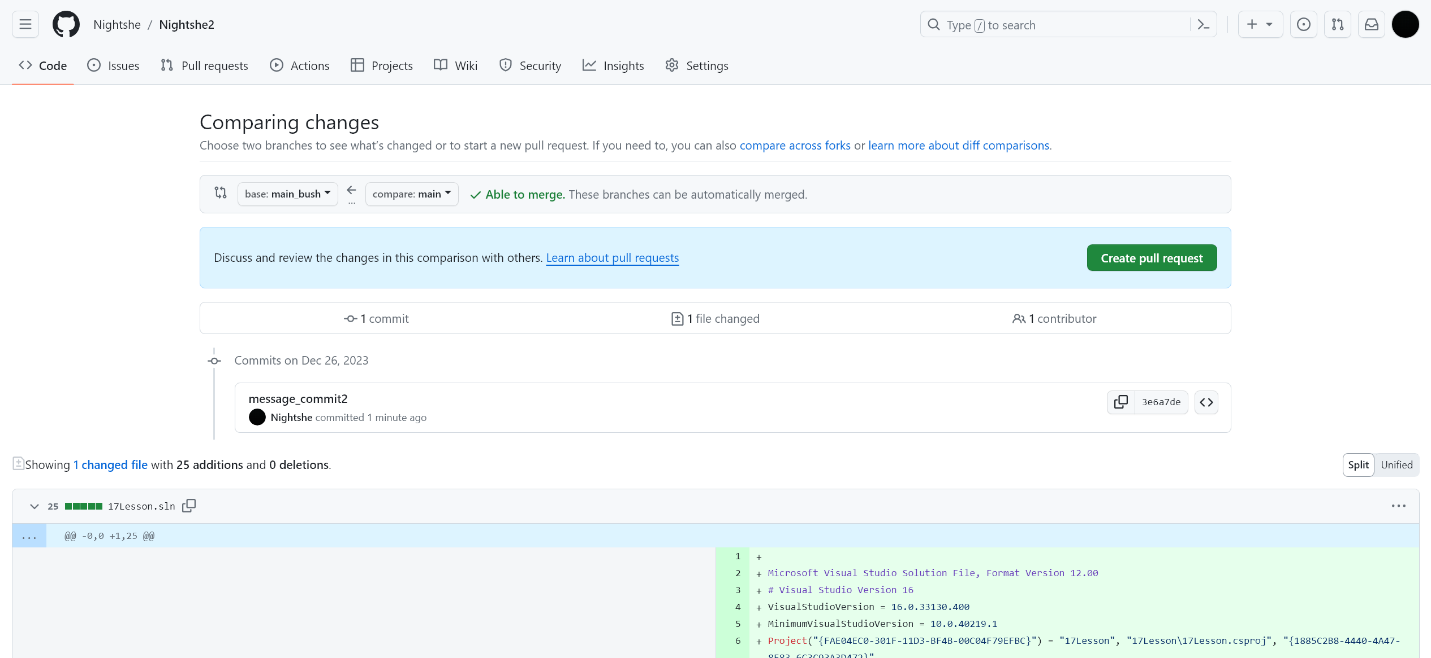


Рисунок 8 – Створення Pull request (витягуючий запит)

Я можу обрати цьому запиту ім’я, а також додати опис, залишаю за умовчуванням (рис. 9).

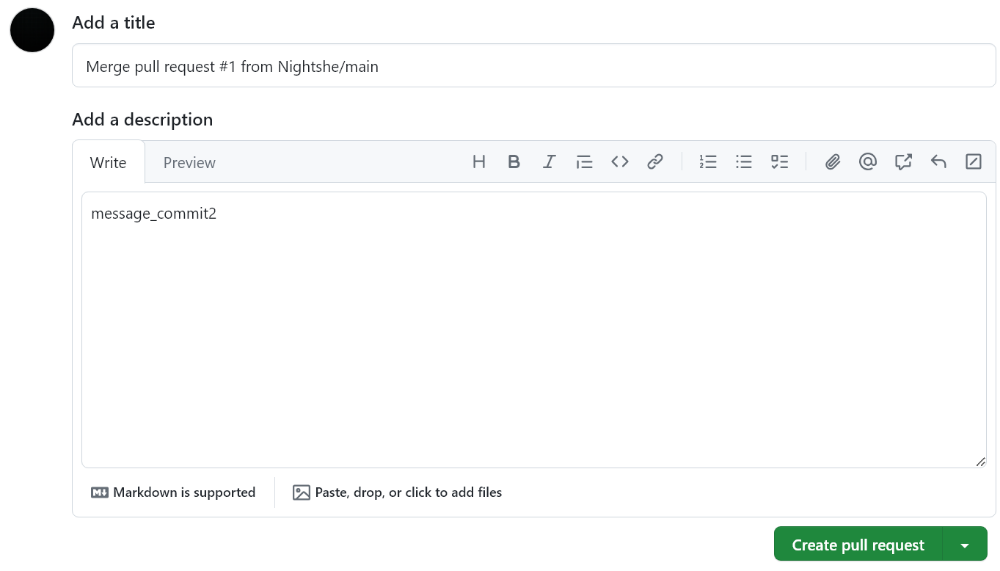


Рисунок 9 – Параметри запиту

Якщо все в порядку, Pull Request може бути злитий (merged) з головною гілкою, що видно на рисунках 10 та 11.

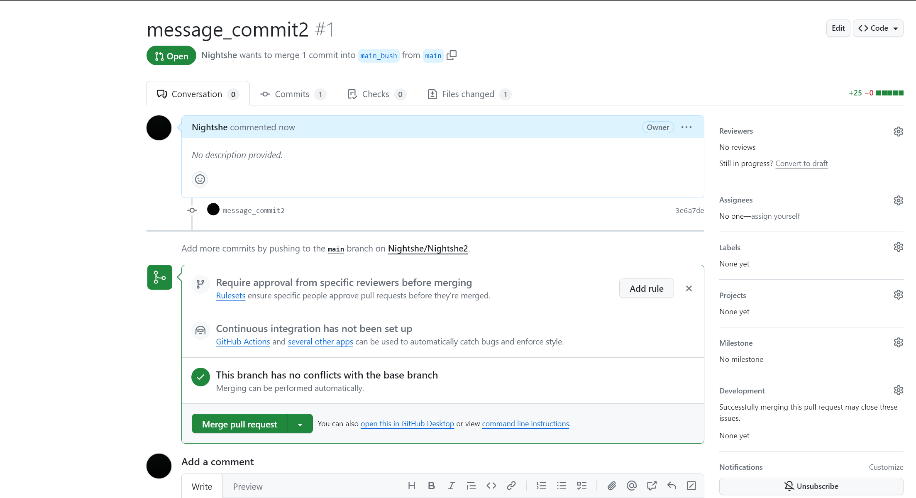


Рисунок 10 – Процес злиття

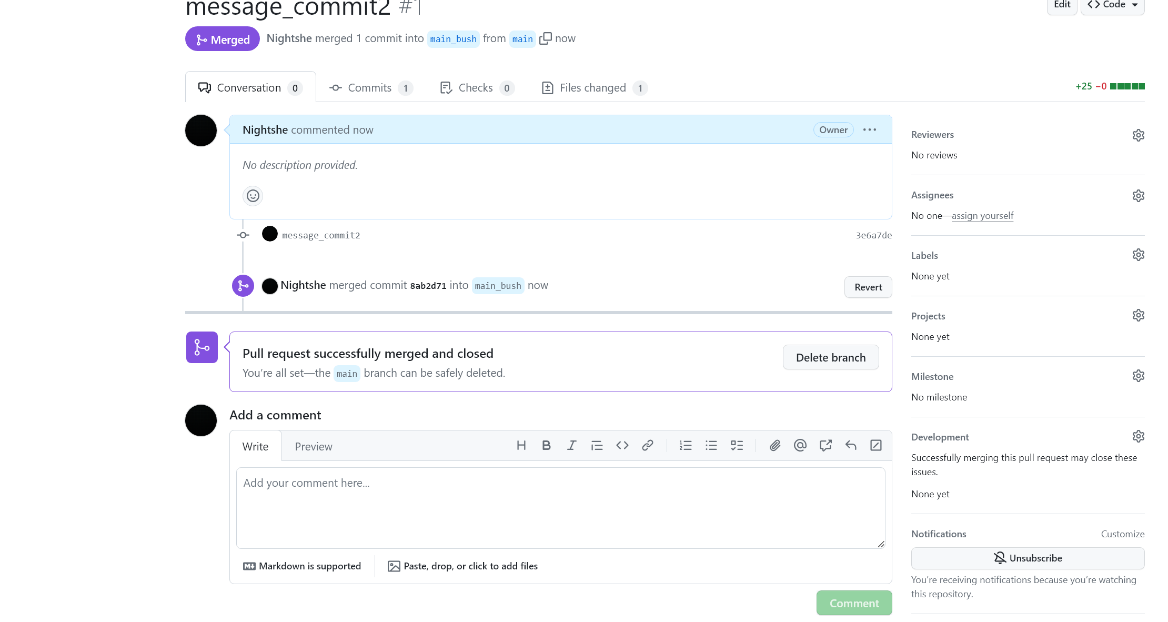


Рисунок 11 – Успішний процес злиття.

Це можна переглянути за посиланням: <https://github.com/Nightshe/Nightshe2.git>

**Висновок:** під час виконання цієї лабораторної роботи я ознайомилась та навчилась використовувати основні команди системи контролю версій Git такі, як процес додавання файлів, створення гілок, клонування репозиторію, а також познайомилась з процесом злиття.

**Контрольні запитання**

1. **Що таке системи контролю версій та їх призначення?**

Системи контролю версій (VCS) – це інструменти, що дозволяють відстежувати та управляти змінами в програмному коді та інших файлах. Вони фіксують історію змін, дозволяючи розробникам спільно працювати над проектом, визначаючи, хто та коли вніс зміни, та відновлюючи попередні стани проекту.

1. **Призначення хостингів проектів?**

Хостинги проектів (наприклад, GitHub, GitLab, Bitbucket) надають інфраструктуру для зберігання, спільного доступу та управління проектами, що використовують системи контролю версій. Вони дозволяють розробникам спільно працювати, відстежувати зміни та забезпечувати взаємодію у реальному часі.

1. **Що таке версія?**

Версія відображає конкретний стан проекту або програми в певний момент часу. Використовуючи системи контролю версій, можна створювати та відстежувати версії коду, щоб легко повертатися до попередніх станів проекту або визначати, коли та які зміни внесені.

1. **Що таке репозиторій та як поділяються за місцем розташування?**

Репозиторій – це сховище для коду та інших файлів, що використовується системою контролю версій. Вони можуть бути розташовані локально на комп'ютері розробника або віддалено на хостингу, такому як GitHub чи GitLab. Локальні репозиторії зазвичай використовуються для роботи розробників, а віддалені - для спільної роботи та зберігання проекту.

1. **Різниця між Git, GitHub, GitLab?**

* Git: система контролю версій.
* GitHub та GitLab: хостинги проектів, що використовують Git. Надають інфраструктуру для зберігання, спільного доступу та управління проектами.

1. **Яким чином можна створити репозиторій?**

* Локально: використовуйте команду git init в папці вашого проекту.
* Віддалено: використовуйте функцію створення репозиторію на хостингу, такому як GitHub чи GitLab.

1. **Для чого призначені гілки?**

Гілки дозволяють розробникам працювати паралельно над різними функціональностями чи виправленнями помилок безпосередньо в головній гілці. Вони полегшують виправлення конфліктів та підтримують організацію роботи.

1. **Стратегії створення та ведення гілок?**

* Git Flow: використовує гілку develop для розробки та гілку master для стабільних версій.
* GitHub Flow: всі зміни вносяться через гілку main, з кожним запитом на зміни (Pull Request).
* GitLab Flow: аналогічний до GitHub Flow, але дозволяє визначати власні гілки.

1. **Основні команди управління гілками?**

* **git branch:** відображення списку гілок.
* **git branch [назва\_гілки]:** створення нової гілки.
* **git checkout [назва\_гілки]:** перехід до іншої гілки.
* **git merge [назва\_гілки]:** злиття гілки з поточною.
* **git branch -d [назва\_гілки]:** видалення гілки.

1. **Виявлення файлів для індексації?**

**git add [назва\_файлу]** або **git add .** (для всіх файлів).

1. **Фіксація змін?**

**git commit -m "коміт-повідомлення".**

1. **Запит на зміни у віддалений репозиторій?**

Створіть Pull Request на GitHub або GitLab та слідуйте інструкціям на інтерфейсі хостингу для подання змін для обговорення та злиття.