<https://github.com/Ilya-Olenchenko/test_SE.git>

**Лабораторна робота №1**

**Встановлення та налаштування сервера git**

**Мета роботи:** дослідження архітектури, синтаксису та принципів роботи з роз-поділеними системами керування версіями документів на прикладі git. Створення та налаштування серверу git-репозиторію.

**Короткі иеоретичні відомості:** системи контролю версій Системи контролю версій – програмне забезпечення, що допомагає керувати змінами в програмному проєкті. Система контролю версій відстежує будь-які модифікації проєкту та зберігає їх у спеціальній базі даних таким чином, щоб у будь-який момент можна було: - переглянути історію змін - дізнатись дату/час та автора, який вніс відповідні зміни в проект - переглянути старі версії проекту - відмінити будь-які зміни, які, наприклад, привели до помилок, повернувшись до більш старої версії проекту - зберігати декілька паралельних версій проекту - проводити злиття декількох паралельних версій проекту - інтегрувати системи контролю версій з іншим програмним забезпеченням для зменшення впливу людського фактору на процеси розгортання проекту і т.д. Системи контролю версій відіграють дуже важливу роль у командній розробці. Розробники зазвичай працюють в командах, постійно доповнюють, модифікують код проекту, який в свою чергу організовується у файловій системі у вигляді дерева папок з файлами. Члени команди розробки можуть працювати одночасно над різними функціями проекту, або одночасно як і над новими функціями, так і над виправленням помилок в поточній версії проекту. Внесення змін в цьому разі, очевидно, призводить до змін файлів проекту та змін до дерева каталогів проекту. Одночасна робота над проектом іноді призводить до конфліктів рішень, оскільки зміни, внесені декількома учасниками, можуть бути не сумісними одна з одною. Система контролю версій допомагає вирішувати подібні проблеми, відстежуючи кожну зміну кожного учасника, надаючи інструменти відстеження та виправлення конфліктів. Проблеми несумісності рішень необхідно виявляти та вирішувати впорядкова- Методичні вказівки до виконання лабораторних робіт 6 но, не блокуючи робіт інших команд. Системи контролю версій при правильному використанні забезпечують плавний і безперервний потік змін до коду від декількох учасників проекту, не блокуючи один одного. Якщо при розробці проекту не використовувати системи контролю версій, то це може призвести до наступних проблем: - втрата інформації: хто, коли і як вніс відповідні зміни в проект - внесення конфліктуючих змін, які призводять до а) блокування подальшої розробки; б) неправильної роботи відразу декількох функції проекту, зміни до яких конфліктують - складності відстеження причин, які призвели до помилок в програмному забезпечення - наявності декількох реалізацій функції програмного забезпечення у файлах проекту, коли робочою є лише одна із них, а всі інші тільки нагромаджують код проекту роблячи його менш зрозумілим і т.д

**Хід роботи**

1. **Створити пустий git-репозиторій (створити локальний репозиторій проєкту на власній машині або створити репозиторій проєкту на публічному сервісі** [**https://github.com/**](https://github.com/)**).**

На рисунку 1 зображено створений репозиторій.

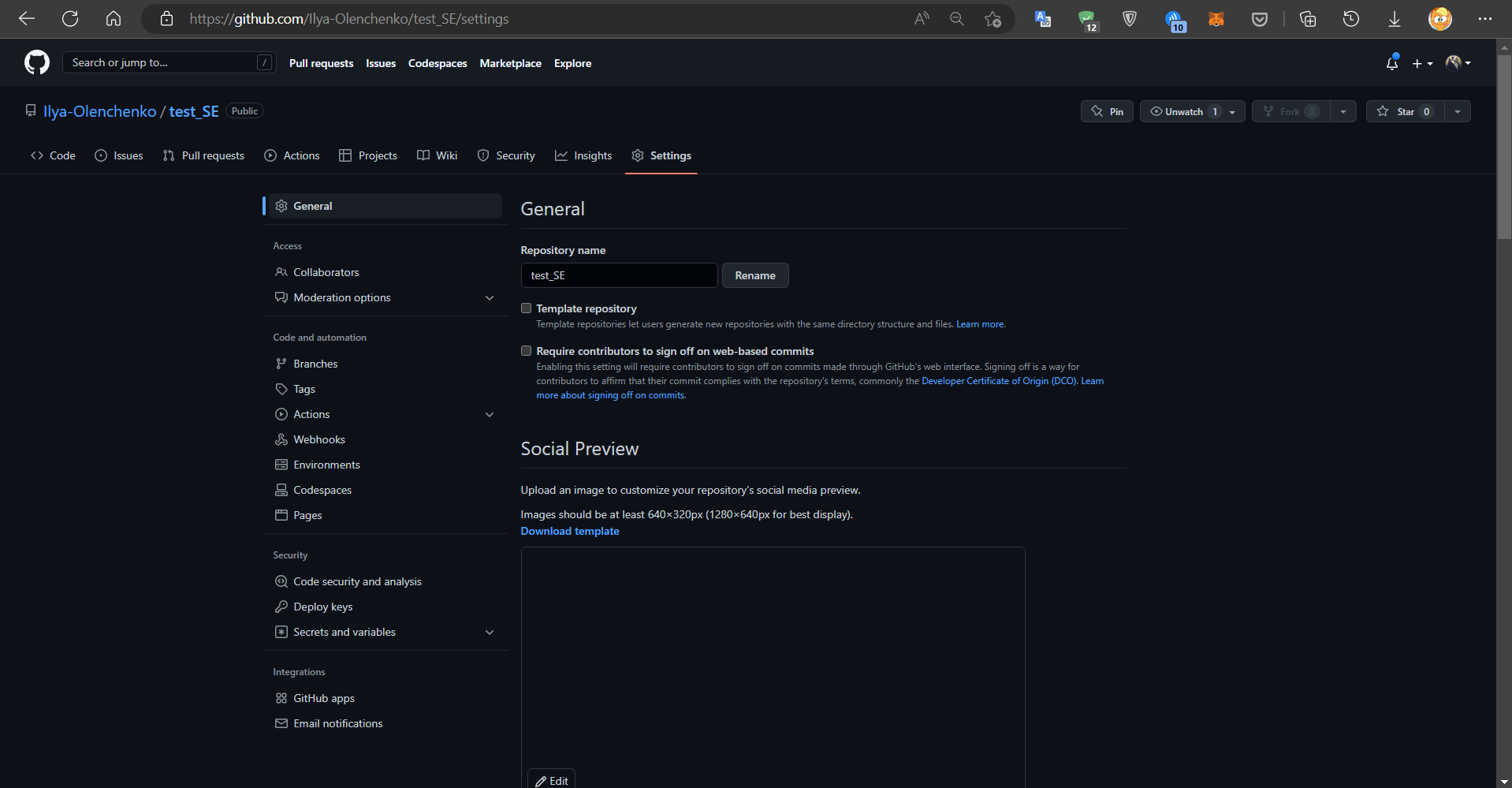
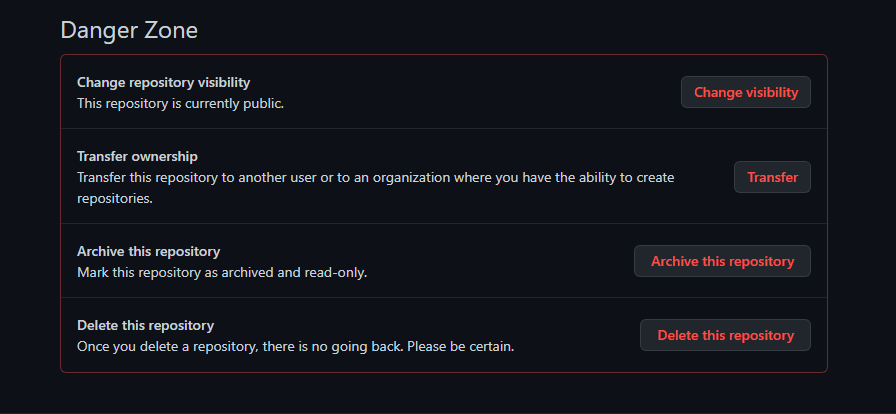


Рисунок 1 – Налаштування репозиторія

**2. Налаштувати права доступу до репозиторію.**

Рисунок 2 - Встановлені публічні права доступу

**3.Виконати клонування репозиторію на клієнтську машину.**

**4. Додати файли будь-якого проєкту (наприклад, вихідний код будь-якого курсового проєкту) до локального репозиторію проєкту.**

На рисунку 3 зображено клонування репозиторію та файл доданий до папки репозиторію.

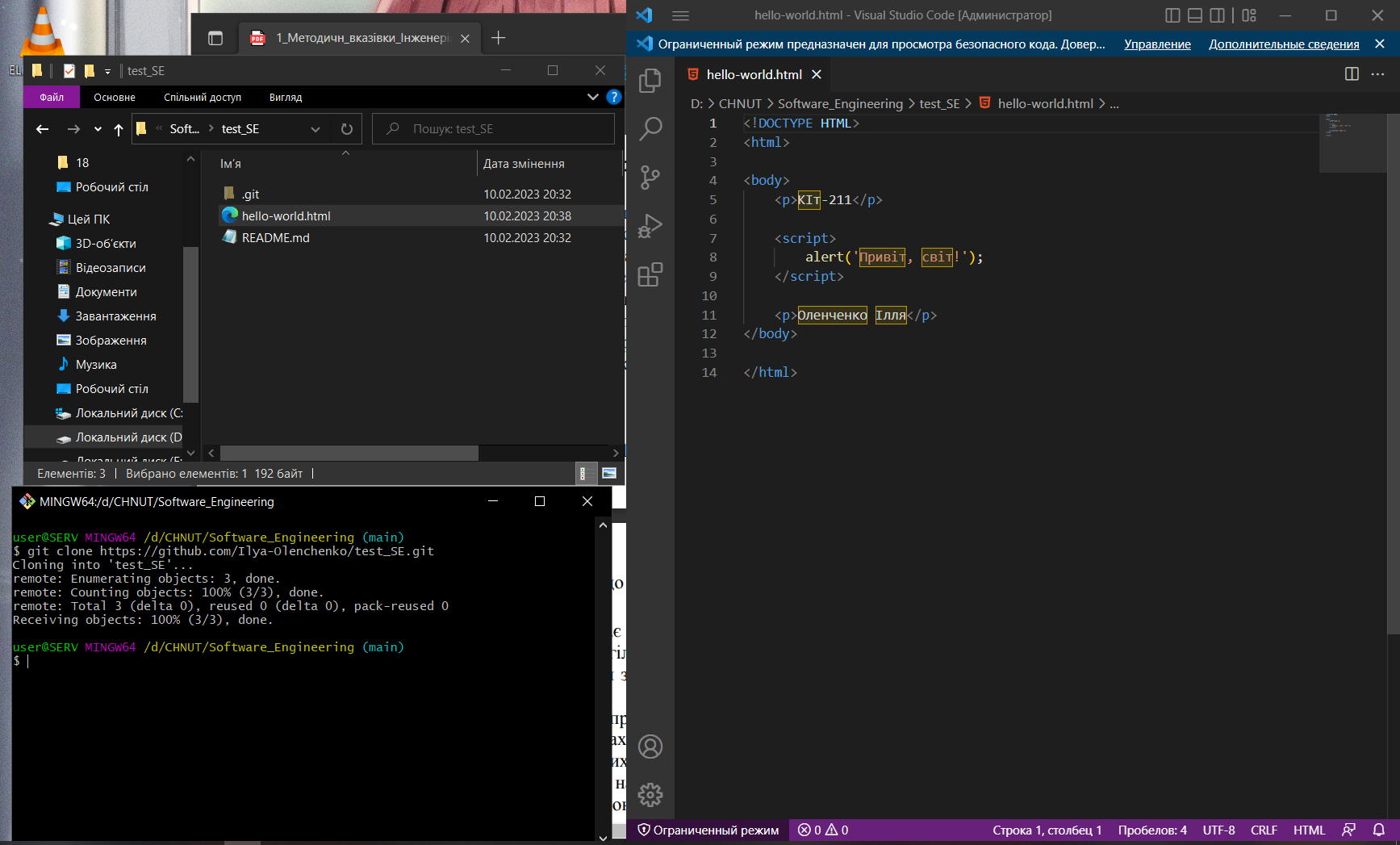


Рисунок 3 – Файли репозиторію

**5. Виконати коміт, який включає всі додані файли проекту. За замовчуванням буде створена гілка master.**

За допомогою команди **git add .** обрали всі файли в папці, і за допомогою команди **git commit –m "Додано файл hello world.html"** ми виконали коміт і дали йому ім’я "Додано файл hello world.html". Результат зображено на рисунку 4.

**6 Відправити коміт з внесеними змінами на віддалений репозиторій.**

Щоб відправити коміт використовується команда **git push** Результат зображено на рисунку 4.

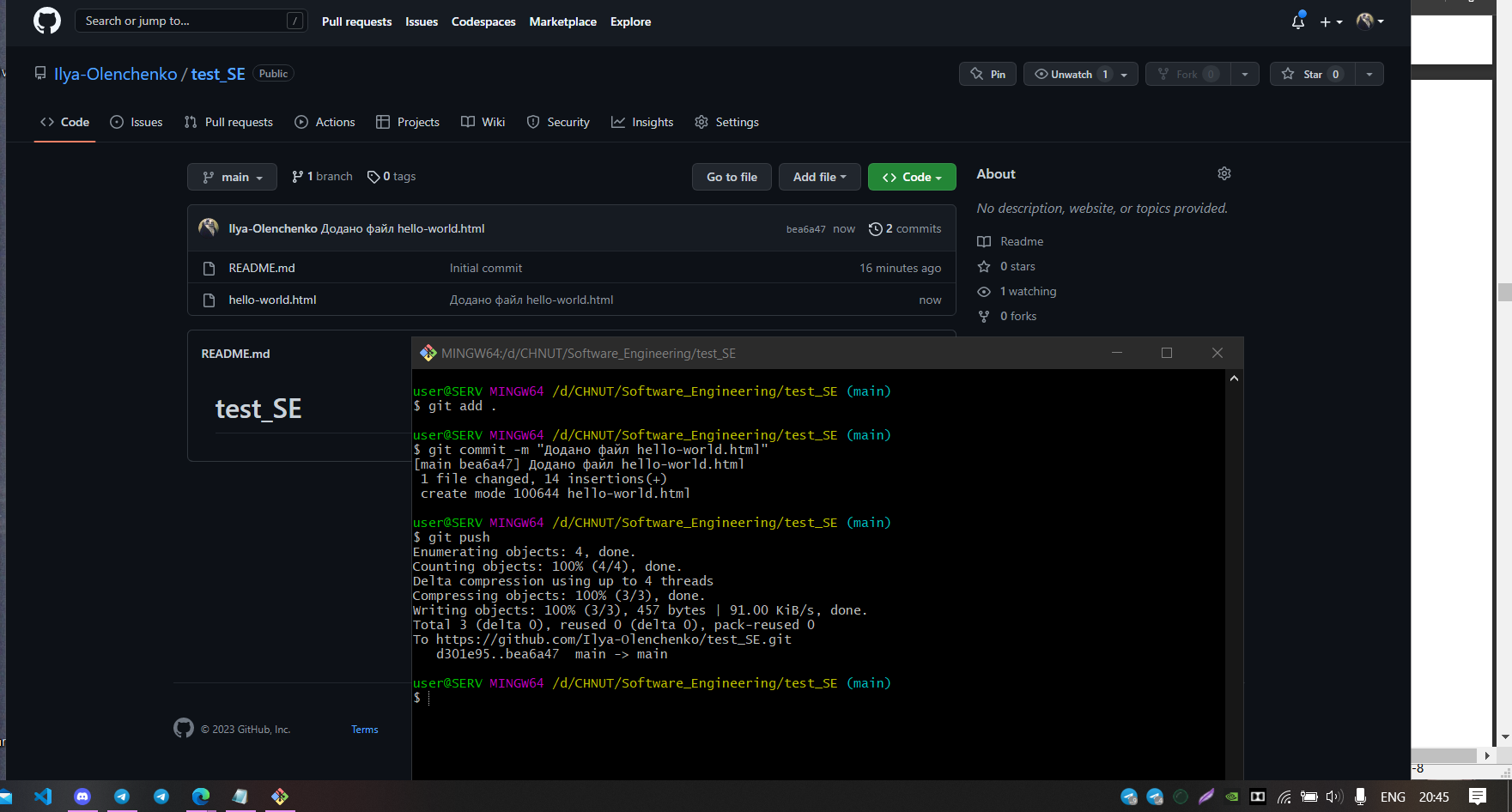


Рисунок 4 – Результат git push

**7. Створити дві локальні гілки проєкту та створити по одному додатковому коміту в цих гілках.**

Для створення нової гілки використовувалась команда **git branch <name>** в проєкте було створено дві гілки b1, b2. Для кожної гілки в файлі були додані зміни, щоб додати зміни в гілку ми повинні перйти в робочу область гілки b1 за допомогою команди **git checkout b1** і зробити в цій гілці коміт командою **git commit –m <name>** така процедура була зроблена і для гілки b2. На рисунках 5 - 6 зображено зміни в файлі які були додані в гілки.

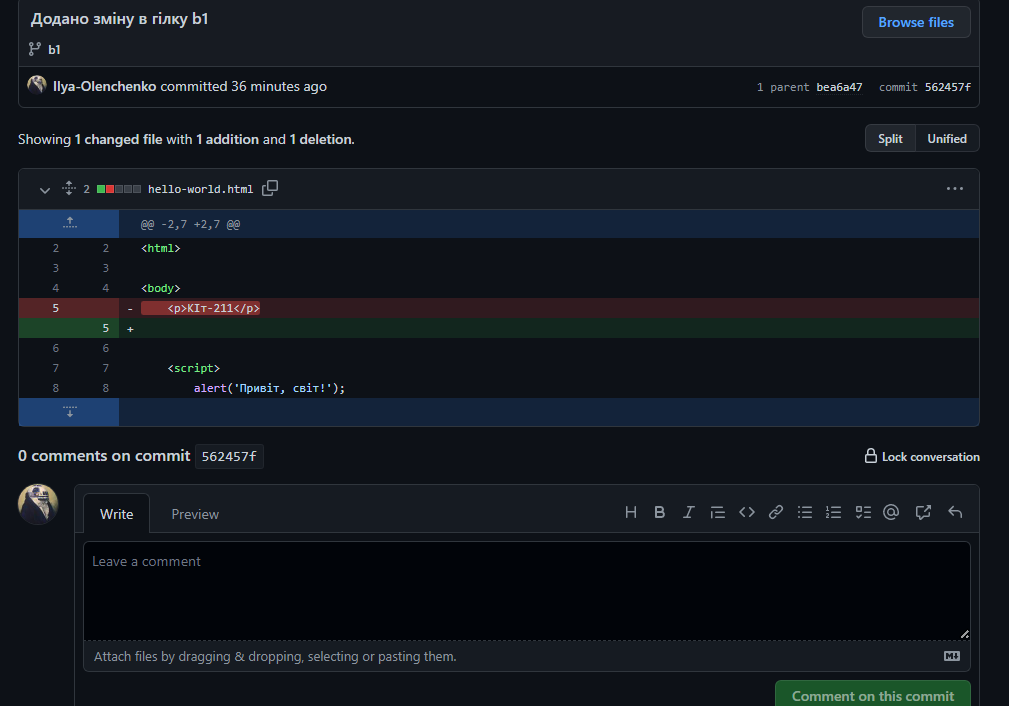


Рисунок 5 – Результат змін в гілці b1

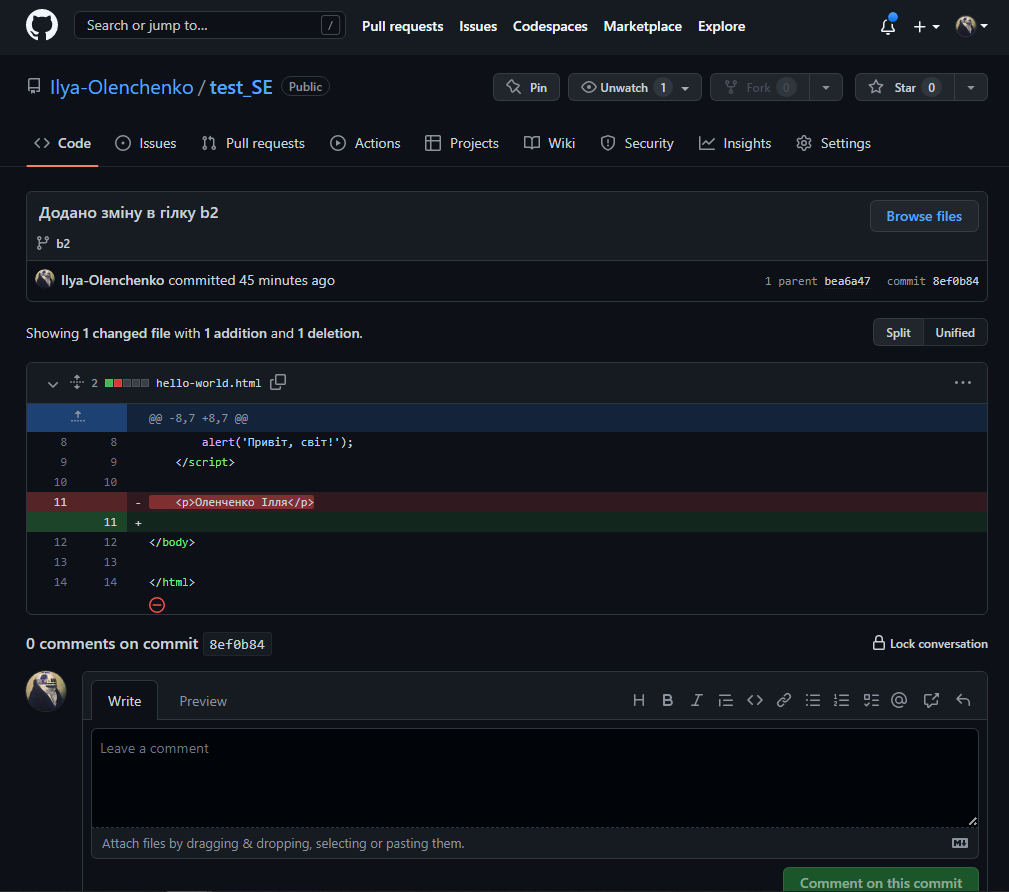


Рисунок 6 – Результат змін в гілці b2

**8. Виконати злиття змін, створених в гілках, в гілку master.**

Для злиття змін в гілку master потрібно зробити pull request для двох гілок в репозиторії. На рисунку 7-8 зображено злиття гілок в master.

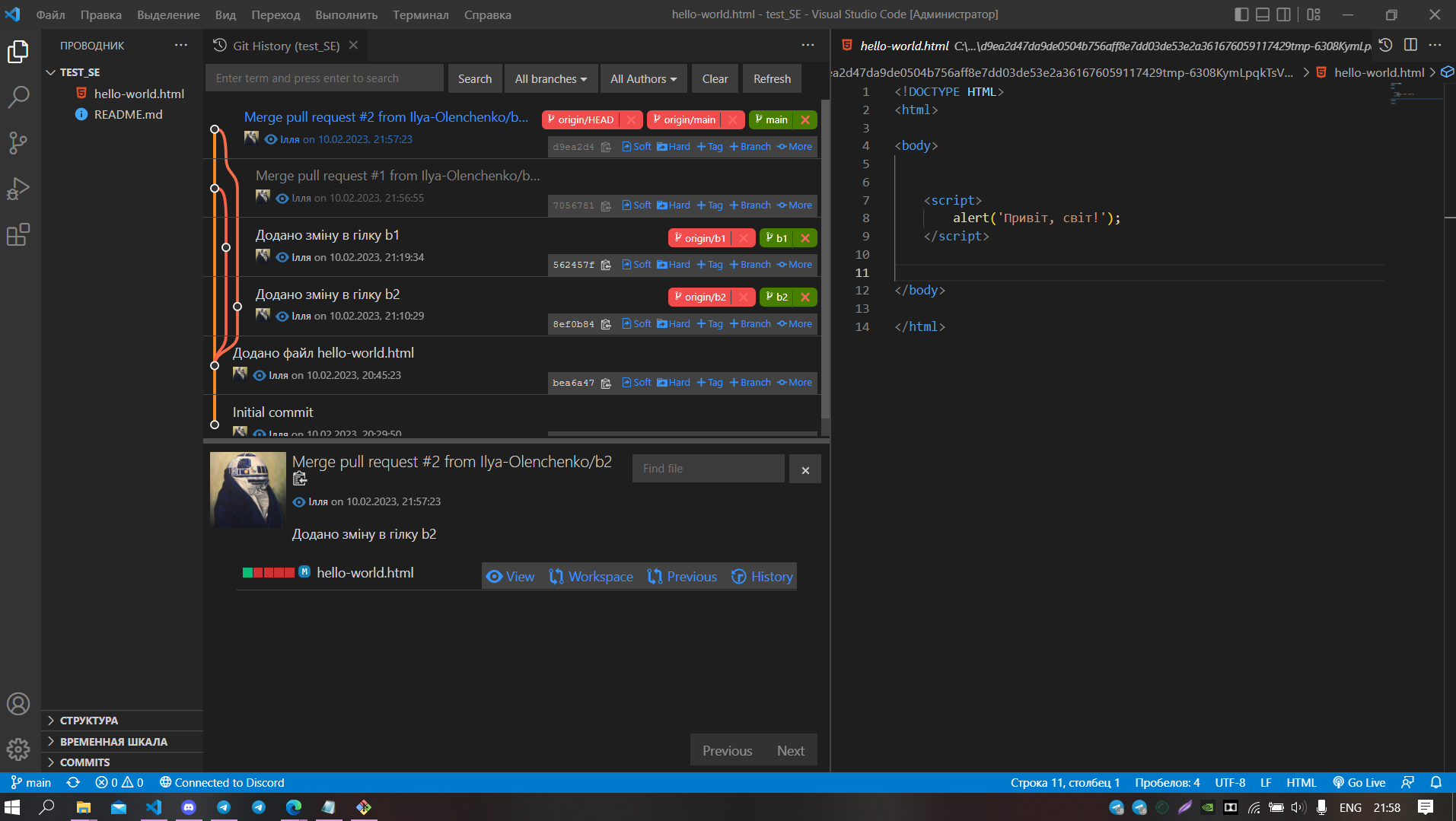


Рисунок 7 – Злиття гілок

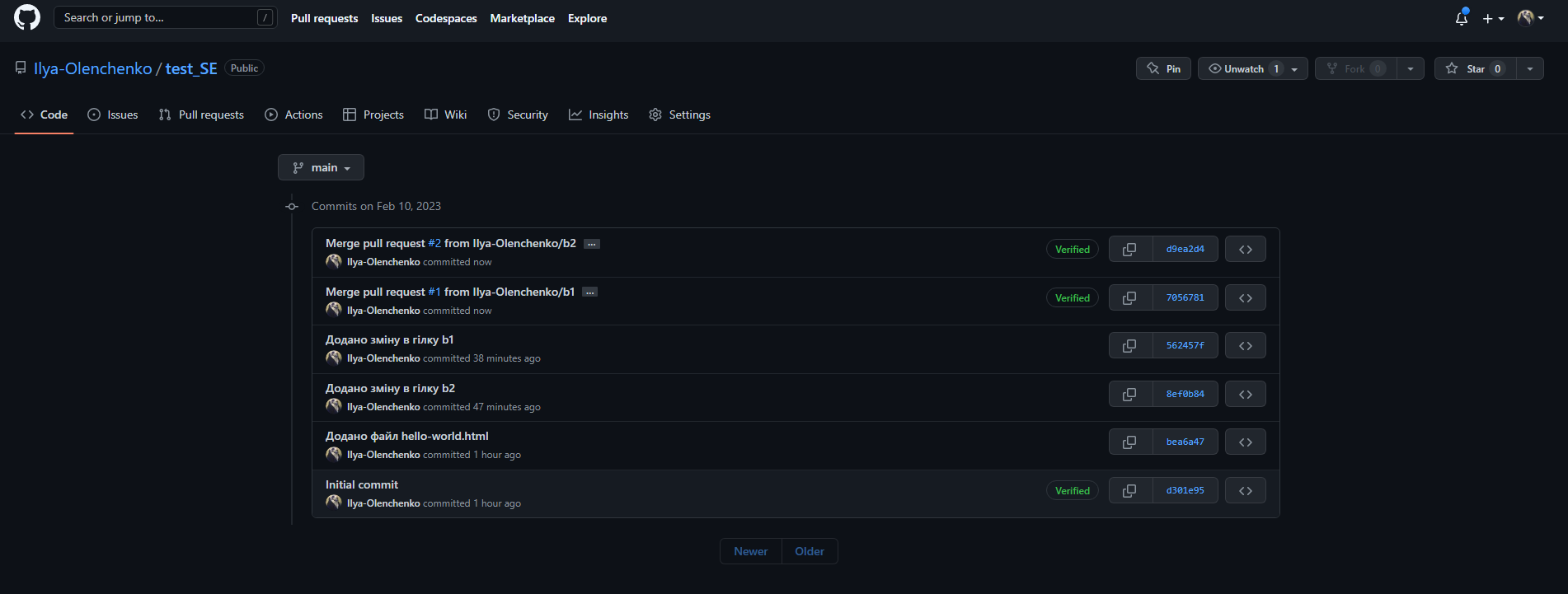


Рисунок 8 – Злиття гілок

**9. Внести на гілках проєкту конфліктуючі зміни (таких, щоб при злитті змін в гілку master утворився конфлікт).**

Для створення помилки в файлі з кодом було створено конфлікт в гілці b1 та відправлено на репозиторій для спроби злиття з головною гілкою main. На рисунку 9 зображено помилку.

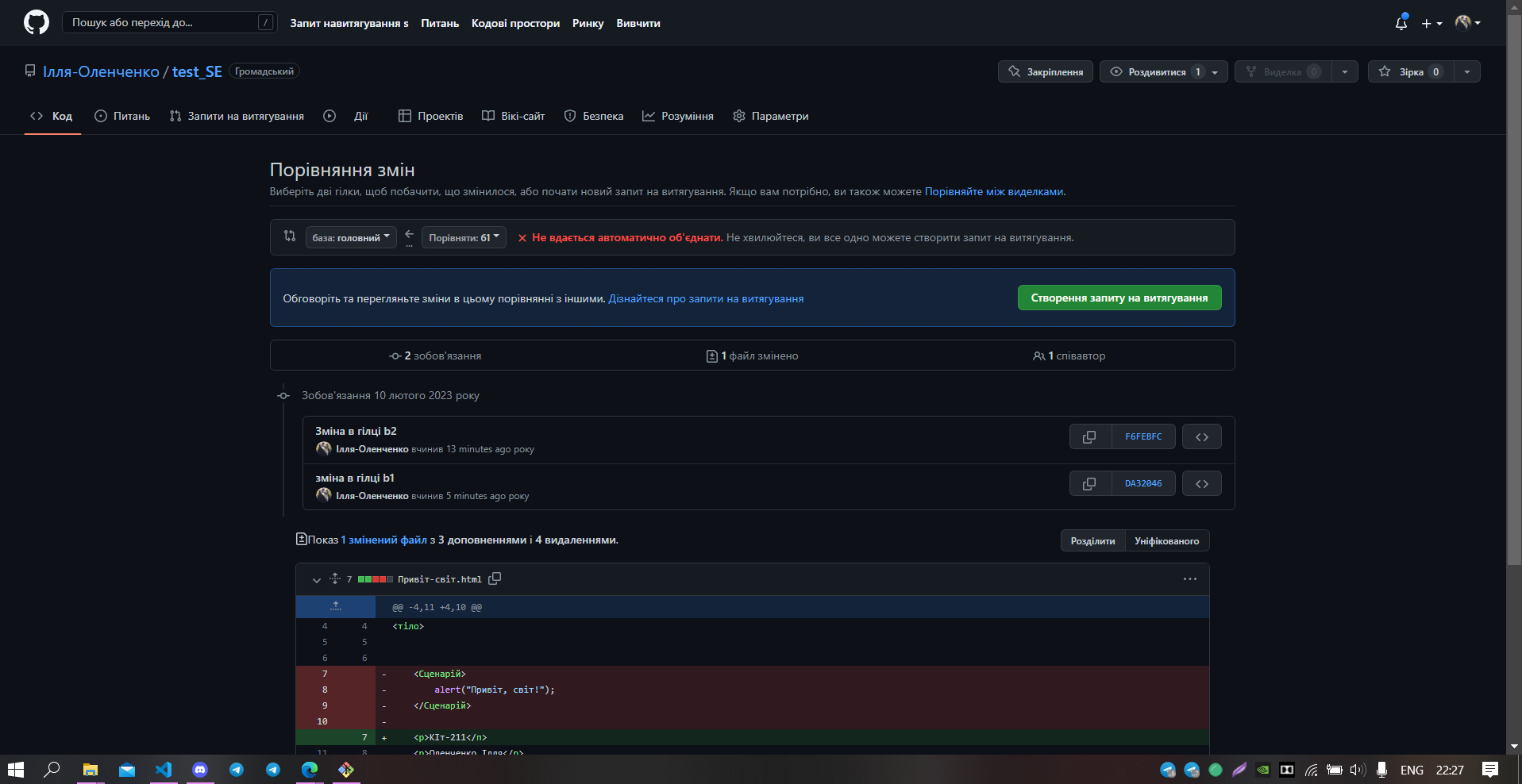


Рисунок 9 – Помилка злиття

**10. Виправити конфлікт. Виконати додатковий коміт після виправлення конфлікту.**

Для вирішення конфлікту код було змінено вручну.

На рисунку 9 зображено результат зміни коду.

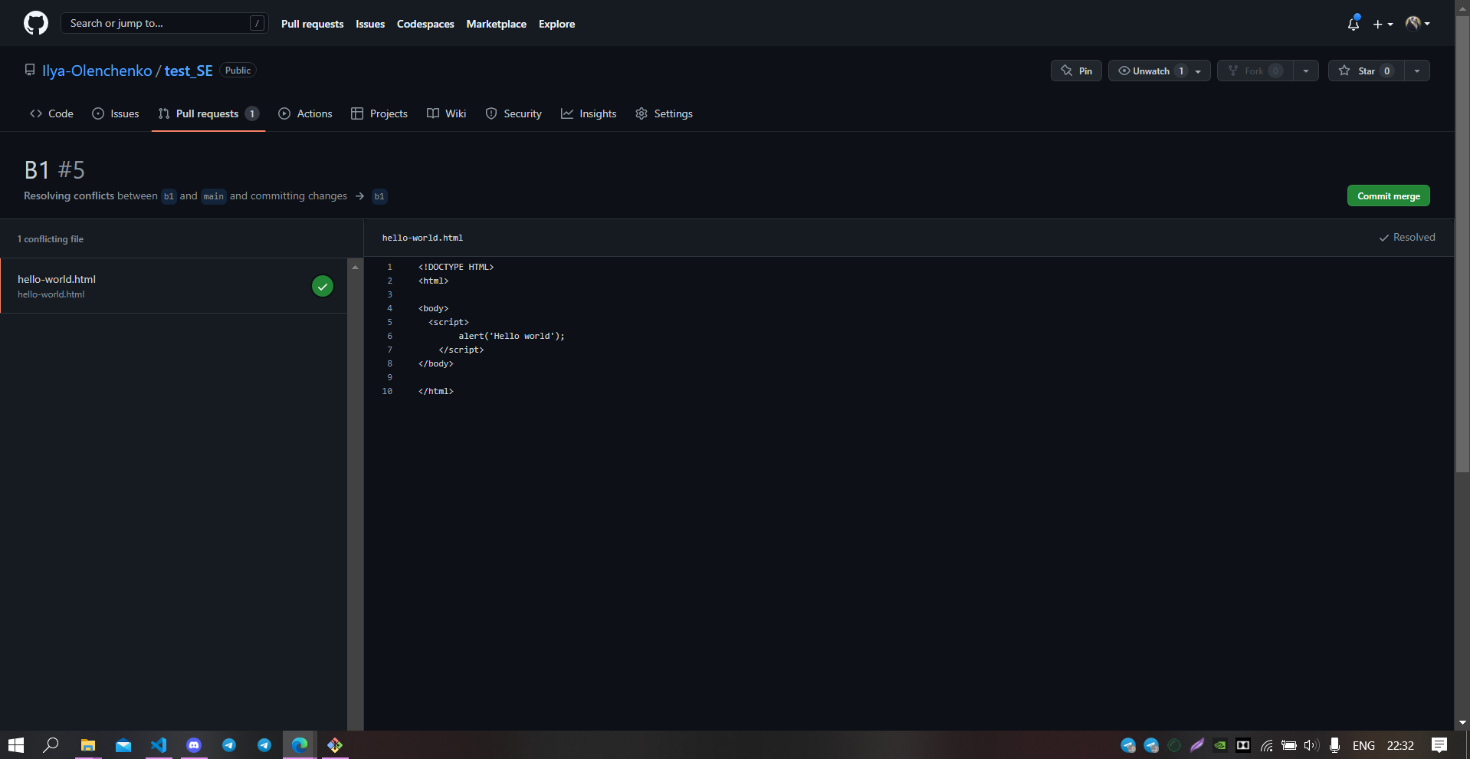


Рисунок 9 – Результат зміни коду.

На рисунку 10 зображено схему злиття гілок b1 і b2 з головною гілкою main після вирішення конфлікту.

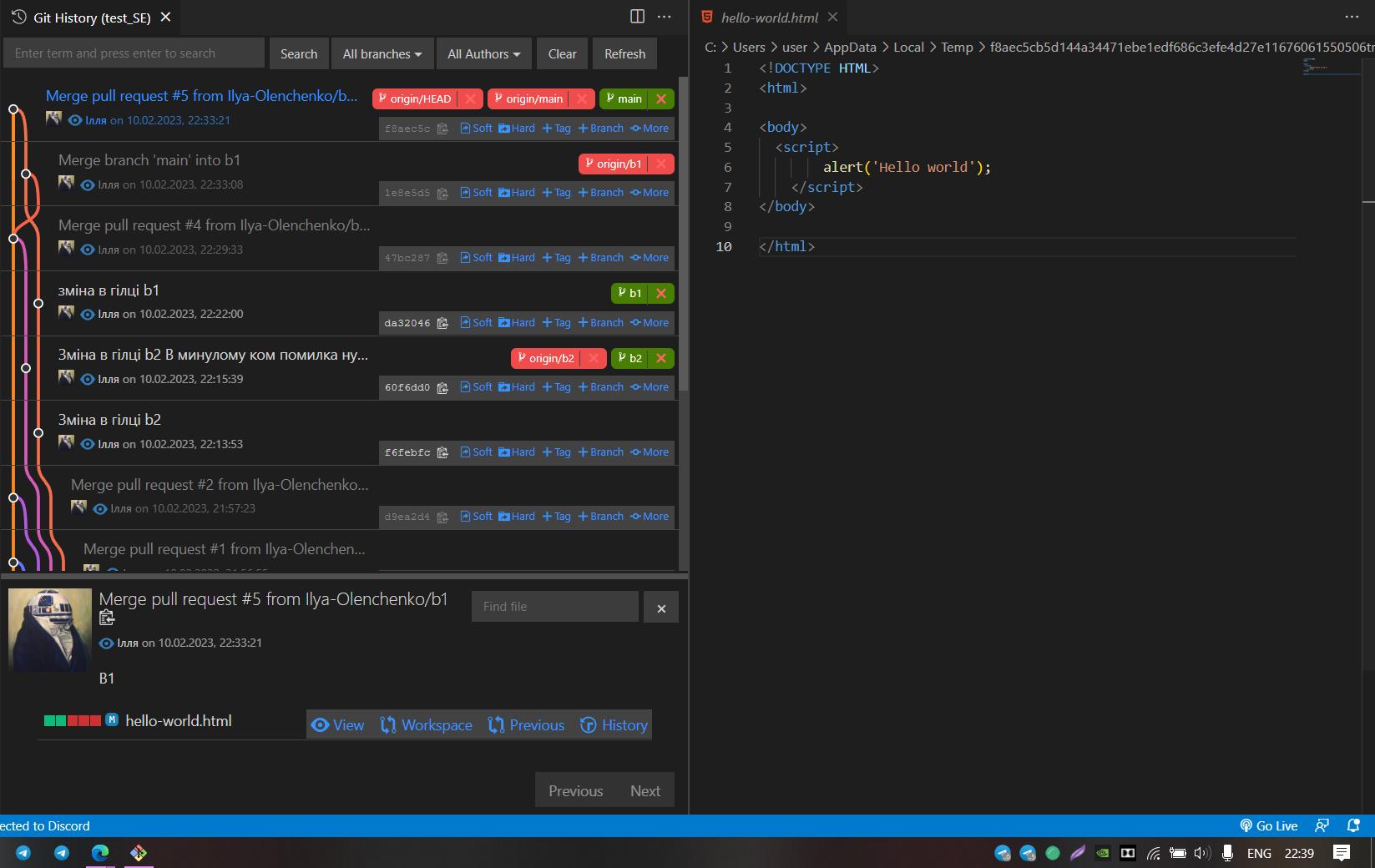


Рисунок 10 – Результат злиття гілок

**Висновок:** дослідив архітектуру, синтаксис та принцип роботи з роз-поділеними системами керування версіями документів на прикладі git. Створив та налаштував git-репозиторію.