

Хмельницький національний університет
Факультет інформаційних технологій
Кафедра комп'ютерних наук

Звіт
до лабораторної роботи №3
з дисципліни «Основи програмної інженерії та тестування програмного
забезпечення»
на тему: «Використання GitHub та GitHub Desktop для керування версіями та
спільної роботи»

Варіант – № 14

Виконав:
ст. гр. КН-23-1
Надопта Кирило
Перевірила:
Викладач Залуцька О. О.

Хмельницький 2025

Завдання 1. Зареєструватися у веб-сервісі спільної роботи GitHub. Відкоригувати дані профілю, завантаживши фотографію.

Завдання 2. Створити власний приватний репозиторій на GitHub.

Завдання 3. Надати доступ до репозиторію викладачу, знайшовши користувача за логіном.

Завдання 4. Скачати та встановити настільний клієнт GitHub Desktop на робочий комп'ютер. Авторизуватися у настільному клієнті за допомогою створеного раніше профілю GitHub.

Завдання 5. Клонувати створений репозиторій на робочий комп'ютер.

Завдання 6. Створити Windows Forms застосунок на мові програмування C# у папці локального репозиторію GitHub згідно варіанту. Варіант виконання завдання призначається викладачем індивідуально. Передбачити введення відомих даних користувачем з клавіатури, забезпечити пояснення задачі, яку вирішує застосунок.

Завдання 7. Завантажити створений програмний застосунок на віддалений сервер GitHub. За потреби внести зміни в код, створивши для цього окрему гілку, після чого виконати злиття.

Завдання 8. Оформити звіт з виконання лабораторної роботи, завантажити звіт на віддалений репозиторій GitHub.

Варіант завдання: 14. Знайти площу трикутника, якщо відомі його основа та висота.

Переходимо на GitHub та натискаємо «Sign Up». Заповнюємо форму реєстрації та входимо в обліковий запис. Далі переходимо в налаштування профілю після чого натискаємо «Edit» та завантажуюмо фото.

Name

Kyrylo Nadopta

Your name may appear around GitHub where you contribute or are mentioned. You can remove it at any time.

Public email

Select a verified email to display

You can manage verified email addresses in your [email settings](#)

Bio

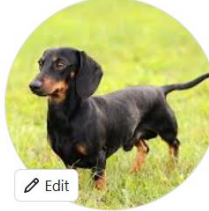
Tell us a little bit about yourself

You can @mention other users and organizations to link to them.

Pronouns

Don't specify

Profile picture



Edit

Рисунок 1 – Налаштування профілю


У верхньому правому куті натискаємо на випадаюче меню та вибираємо «New repository». Вводимо назву та опис, вибираємо видимість та натискаємо «Create repository».

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Required fields are marked with an asterisk (*).

Owner *

 KirillNadopta

Repository name *

my-first-repo

my-first-repo is available.

Great repository names are short and memorable. Need inspiration? How about [verbose-waffle](#) ?

Description (optional)

Мій перший репозиторій на GitHub

☒ Public

Anyone on the internet can see this repository. You choose who can commit.

☐ Private

You choose who can see and commit to this repository.

Initialize this repository with:

☐ Add a README file

This is where you can write a long description for your project. [Learn more about READMEs](#).

Add .gitignore

.gitignore template: None

Choose which files not to track from a list of templates. [Learn more about ignoring files](#).

Choose a license

License: None

A license tells others what they can and can't do with your code. [Learn more about licenses](#).

ⓘ You are creating a public repository in your personal account.

Create repository

Рисунок 2 – Створення репозиторію

Завантажуємо GitHub Desktop та встановлюємо. Відкриваємо програму, в меню натискаємо «Sign in», в браузері вводимо дані GitHub. Після реєстрації, натискаємо кнопку «Clone», обираємо локальну папку та клонуємо репозиторій.

Далі запускаємо Microsoft Visual Studio. Натискаємо «Create new project», вибираємо тип проєкту «Windows Forms .NET», вказуємо шлях до клонованого репозиторію та натискаємо «Create».

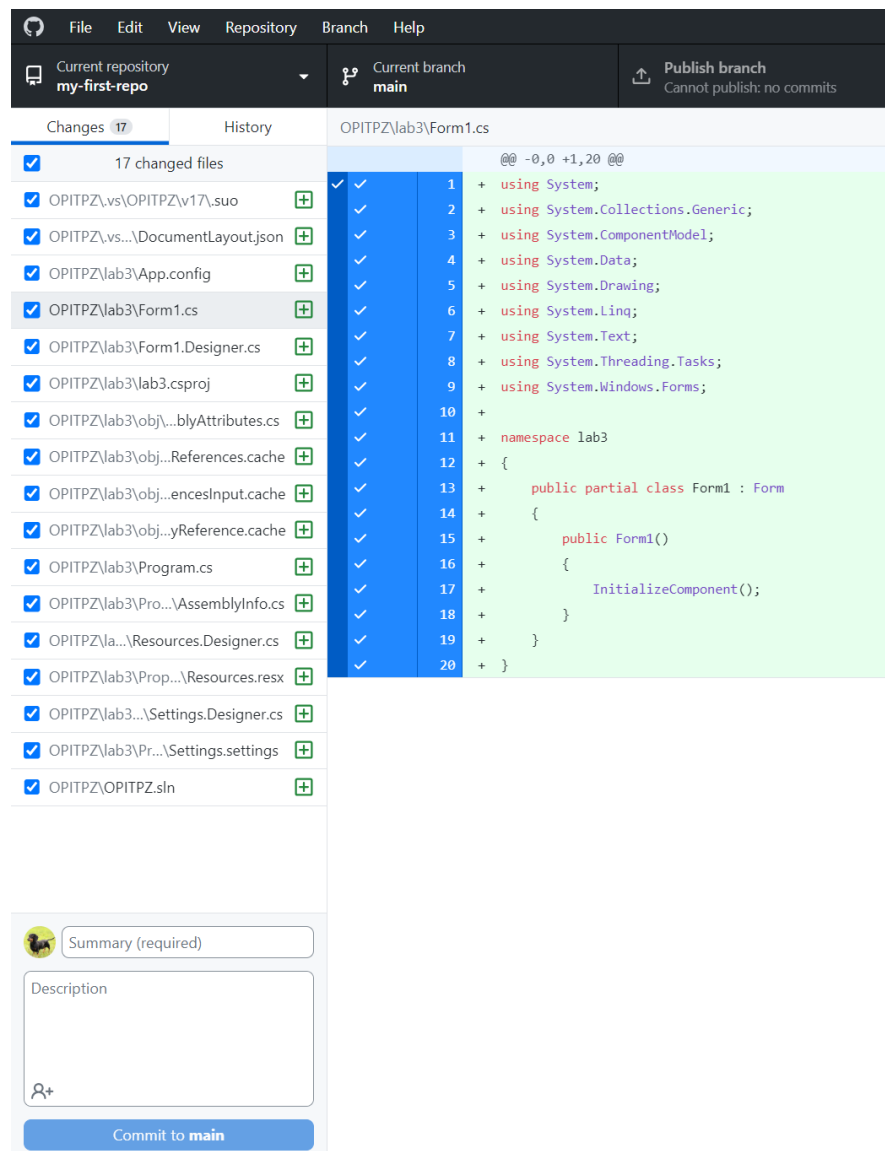


Рисунок 3 – Відображення змін у локальному репозиторії

Тепер виконуємо завдання «14. Знайти площу трикутника, якщо відомі його основа та висота. ». Результат зображено на рисунку 5.

```

namespace lab3
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            if (double.TryParse(textBoxBase.Text, out double baseLength) &&
                double.TryParse(textBoxHeight.Text, out double height))
            {
                double area = 0.5 * baseLength * height;
                labelResult.Text = $"S = {area:F2}";
            }
            else
            {
                MessageBox.Show("Будь ласка, введіть коректні числові значення", "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
    }
}

```

Рисунок 4 – Лістинг коду

Рисунок 5 – Вигляд форми з введеними даними та результатом обчислень

Зберігаємо проєкт у Visual Studio, після чого переходимо у GitHub Desktop та вводимо опис змін і натискаємо «Commit to main». Після цих дій, програмний застосунок завантажився на віддалений сервер GitHub.

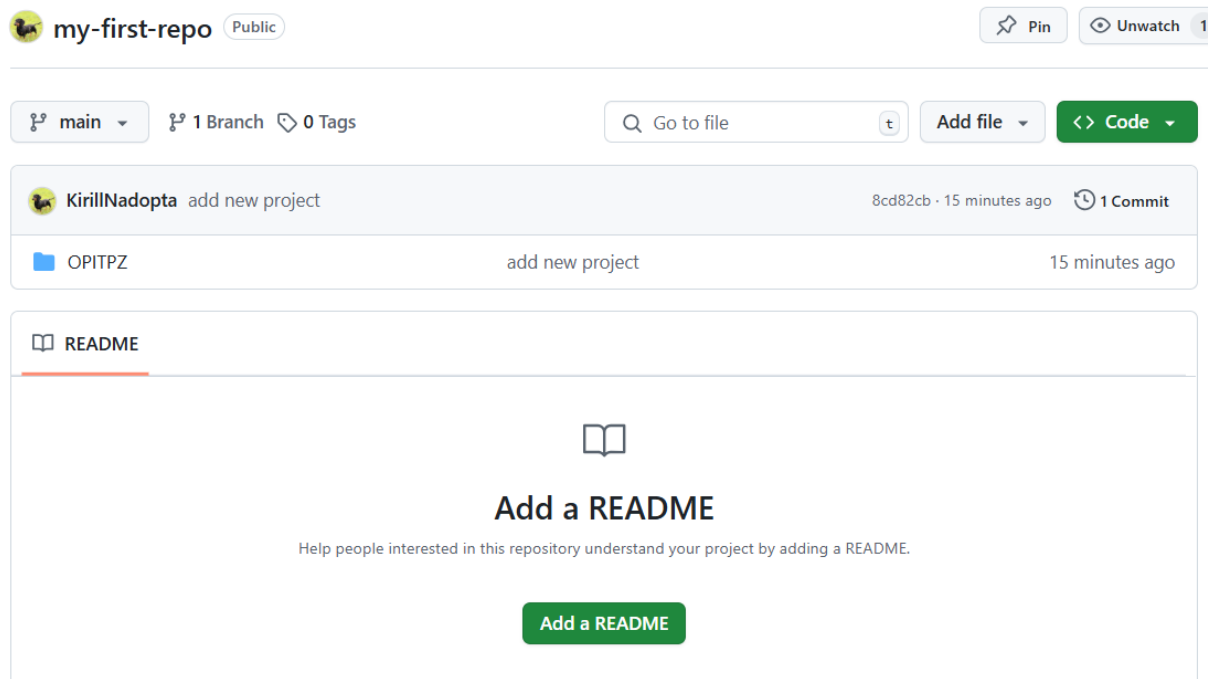


Рисунок 6 – Програмний застосунок на віддаленому сервері GitHub

Контрольні запитання

Що таке система керування версіями?

Це програмне забезпечення, яке відстежує зміни в коді, дозволяє зберігати різні версії файлів, працювати спільно та повертатися до попередніх станів у разі необхідності.

Які типи систем керування версіями є?

- Локальні (Local VCS) – зміни зберігаються тільки на одному комп'ютері.
- Централізовані (CVCS) – всі версії зберігаються на одному сервері (наприклад, SVN, Perforce).
- Розподілені (DVCS) – кожен користувач має повну копію репозиторію (наприклад, Git, Mercurial).

Які СКВ входять в сімейство Git?

- Git
- GitHub
- GitLab
- Bitbucket

Які компоненти та команди Git ви знаєте?

Компоненти:

- Робочий каталог (Working Directory)
- Проіндексована область (Staging Area)
- Локальний репозиторій (Local Repository)
- Віддалений репозиторій (Remote Repository)

Команди:

- `git init` – ініціалізація нового репозиторію
- `git clone` – клонування віддаленого репозиторію
- `git add` – додавання змін у staging
- `git commit -m "повідомлення"` – збереження змін у локальному репозиторії
- `git push` – відправка змін на віддалений сервер
- `git pull` – отримання змін із віддаленого репозиторію
- `git branch` – перегляд та створення гілок
- `git checkout` або `git switch` – перемикання між гілками
- `git merge` – об'єднання гілок

З якою ціллю створюються гілки?

Для паралельної розробки нових функцій, виправлення помилок або тестування без впливу на основний код проєкту.

За допомогою якої команди можна отримати на локальний комп'ютер останні зміни з віддаленого серверу?

`git pull`

Якою командою можна внести зміни на віддалений сервер?

`git push`

Що таке конфлікт злиття? Як його вирішити?

Конфлікт злиття (merge conflict) виникає, коли дві гілки містять зміни в одних і тих же рядках файлу. Вирішення: Відкрити конфліктний файл Вручну вибрати, які зміни залишити Виконати `git add [файл]`, щоб підтвердити вирішення конфлікту Виконати `git commit`

Які переваги використання GitHub для спільної роботи над проєктами? Зручний хостинг репозиторіїв Інструменти для командної роботи (Pull Requests, Issues, Wiki) Контроль версій та історія змін Інтеграція з CI/CD Безкоштовні публічні репозиторії

Висновок

У ході виконання лабораторної роботи було освоєно основи використання платформи GitHub. Було створено обліковий запис на GitHub, налаштовано профіль, створено приватний репозиторій та надано доступ викладачеві. Завантажено та налаштовано GitHub Desktop, виконано клонування репозиторію на локальний комп'ютер. Також було створено Windows Forms застосунок у середовищі Visual Studio для обчислення площі трикутника за заданою основою та висотою. Застосунок збережено у локальному репозиторії, а зміни передано на віддалений сервер GitHub.