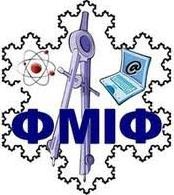
УКРАЇНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІМЕНІ МИХАЙЛА ДРАГОМАНОВА

ФАКУЛЬТЕТ МАТЕМАТИКИ, ІНФОРМАТИКИ ТА ФІЗИКИ

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



ЗВІТ ПРО ВИКОНАНУ РОБОТУ ДО ЛАБОРАТОРНОГО ЗАНЯТТЯ №3

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

***Виконала:*** студентка IV курсу, групи 42ІПЗ, спеціальності 121 Інженерія програмного забезпечення

(шифр і назва спеціальності)

Щеголь Анастасія

(прізвище та ім’я)

***Викладач:***

Січкаренко В.О. .

(прізвище та ініціали)

***Київ – 2025***

# Лабораторна робота

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

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

**Загальні завдання:**

1. Ознайомитися з теоретичною частиною.
2. Створити обліковий запис в GitHub за посиланням вказаному в джерелі [8];

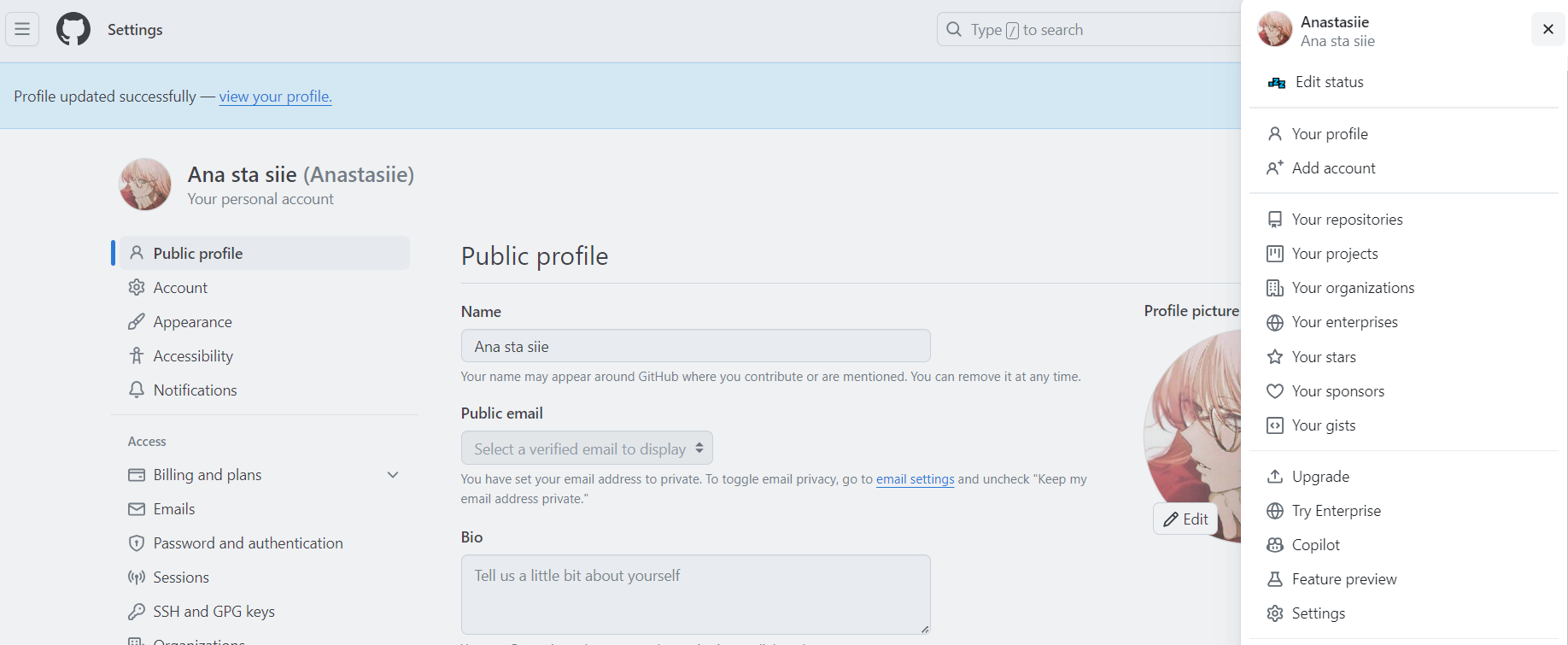


Рис.1 Створено обліковий запис в GitHub

1. Встановити систему контролю версій Git (бажано Git CMD або Git Bash);



Рис.2 Встановлено систему контролю версій Git

1. Створити відкритий (public) репозиторій на хостингу GitHub;

Для цього перейдемо до вкладки «Репозиторії» та натиснемо на клавішу «Новий».

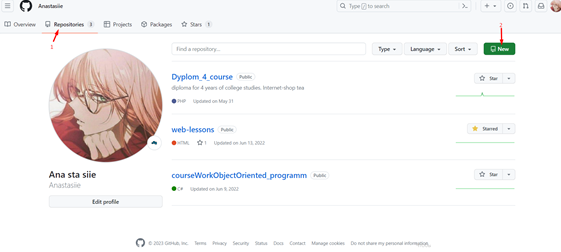


Рис.3. Сторінка репозиторіїв на GitHub

Записуємо його назву та обираємо рівень доступу «Публічний».

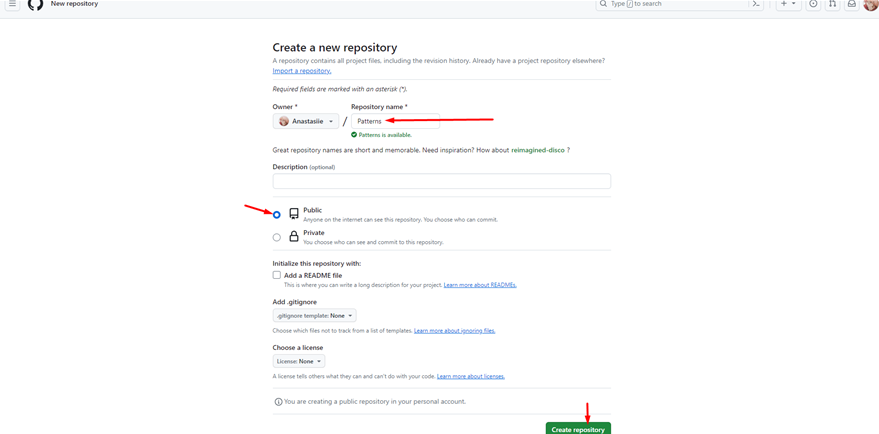


Рис.4. Створення нового репозиторію

1. Клонувати на локальний комп’ютер (використовуючи команду git clone);

Для клонування необхідно відкрити термінал Git Bash, що можна зробити – натиснувши на потрібній папці правою клавішею миші.

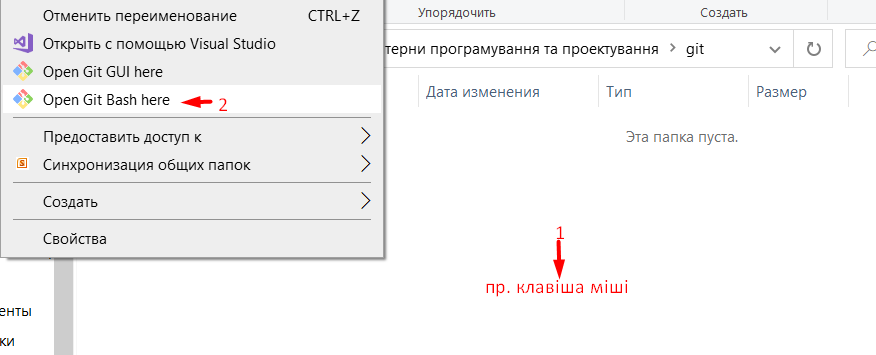


Рис.5. Git Bash



Рис.6 Термінал Git Bash

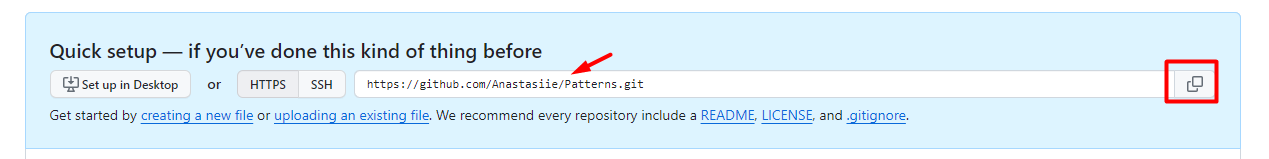


Рис.7 Копіюємо посилання на створений репозиторій

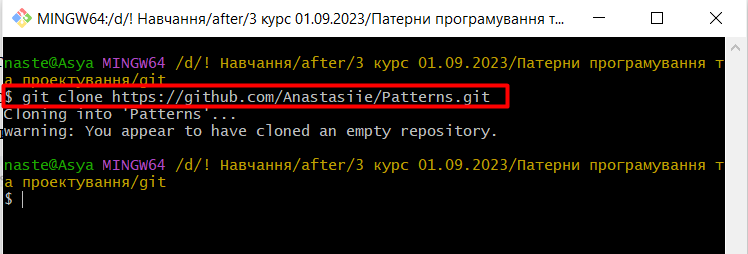


Рис.8 Виконуємо команду git clone для створеного репозиторію

1. Створити в локальному репозиторії гілку (команда git checkout -b);

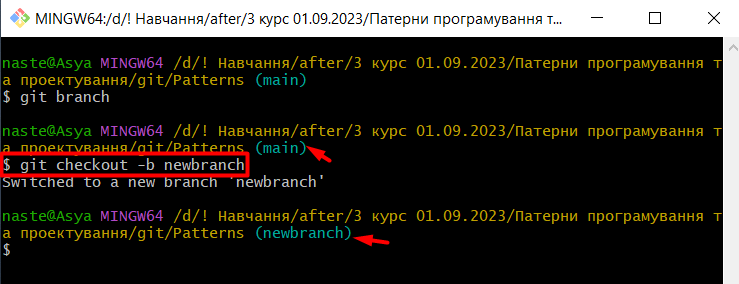


Рис.9 Створюємо гілку “newbranch” для створеного репозиторію

1. Додати в репозиторій файли з кодом або створити тестове рішення (можна використовувати будь-яку IDE);

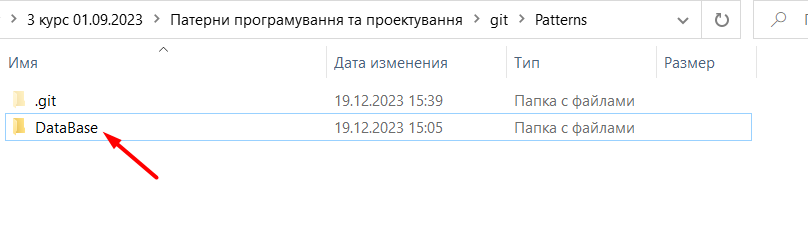


Рис.10 Додаємо в локальний репозиторій файли з кодом

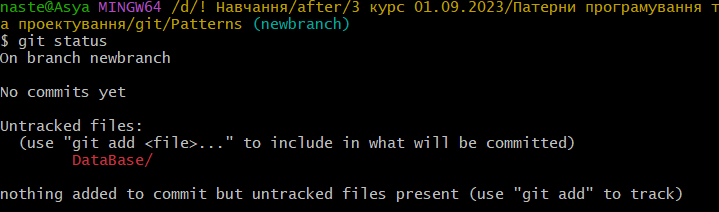


Рис.11 Перевіряємо Git (знайдено новий файл)

1. Індексувати файли або рішення в локальному репозиторії (команда git add);

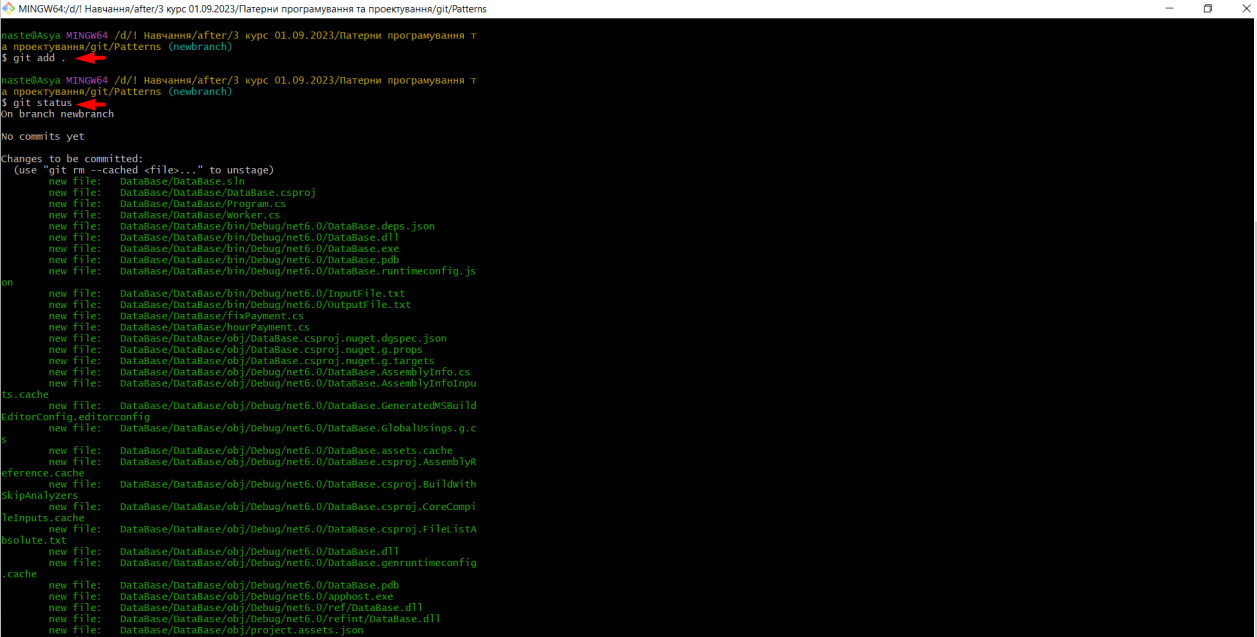


Рис.12 Додаємо файли в локальний репозиторій (команда git add);

1. Зафіксувати зміни (команда git commit);

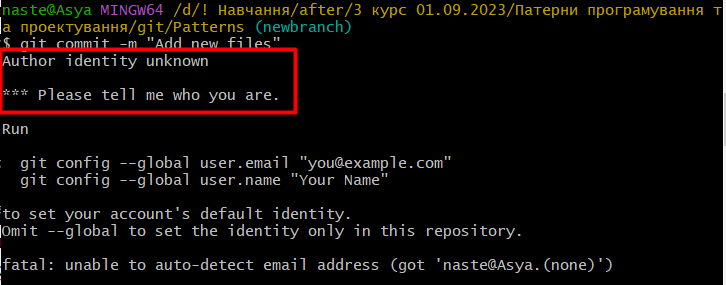


Рис. 13 Помилка через відсутність даних

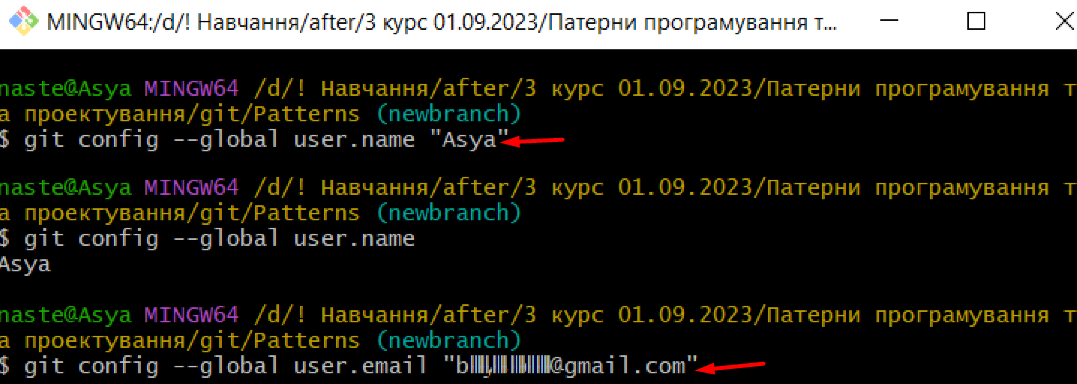


Рис.14 Встановлюємо власні дані (ім’я та пошта)

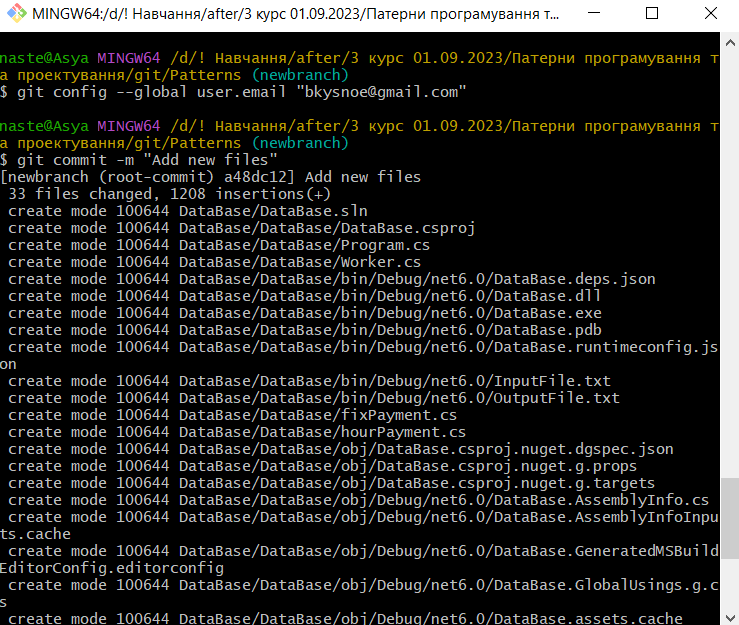


Рис.15 git commit –message “Add new files”

1. Надіслати запит на зміни у віддалений репозиторій (команда git push);



Рис.16 Відправляємо запит на надсилання даних в віддалений репозиторій GitHub

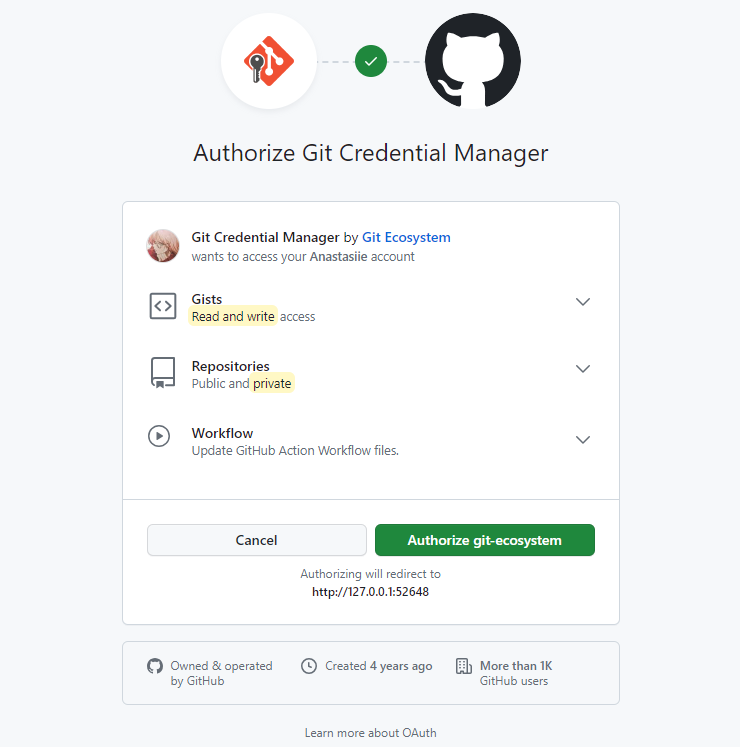


Рис.17 Авторизація даних користувача Git з даними GitHub

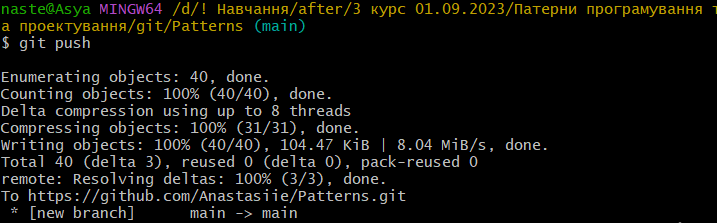


Рис.18 Успішна реалізація запиту на надсилання даних в віддалений репозиторій GitHub

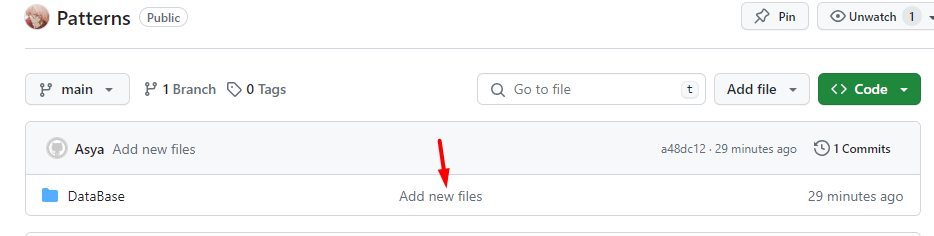


Рис.19 Результат в репозиторії GitHub

1. Надіслати викладачу посилання на створений репозиторій в GitHub

GitHub

https://github.com/Anastasiie/Patterns.git

***Відповіді на контрольні запитання:***

1. Що таке системи контролю версій та їх призначення

Системи контролю версій (СКВ) - це програмні засоби для відстеження змін у файлах та каталогах, а також управління цими змінами. Вони дозволяють фіксувати, відновлювати та порівнювати різні версії файлів.

1. Призначення хостингів проектів

Хостинги проектів надають місце для зберігання та обробки коду проекту. Вони часто підтримують системи контролю версій та забезпечують інструменти для спільної роботи розробників над проектом в Інтернеті.

1. Що таке версія?

Версія - це конкретний стан проекту або файлу в певний момент часу. В системах контролю версій кожна зміна зберігається як нова версія, що дозволяє відслідковувати та управляти історією змін.

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

Репозиторій - це сховище, де зберігається код та історія його змін. Репозиторії можуть бути локальними (на комп'ютері розробника) або віддаленими (зберігаються на сервері або хостингу).

1. Різниця між Git, GitHub, GitLab?

* Git: Система контролю версій, що дозволяє відстежувати зміни в програмному коді.
* GitHub: Веб - платформа для спільної роботи з Git - репозиторіями, надає інструменти для спільної розробки та управління проектами.
* GitLab: Альтернативний сервіс, подібний до GitHub, який також надає інструменти для роботи з Git та управління проектами.

1. Яким чином можна створити репозиторій?

Для створення репозиторію можна використовувати команду git init для локального репозиторію або використовувати функціонал веб-сервісу, такого як GitHub або GitLab.

1. Для чого призначені гілки?

Гілки в Git дозволяють розробникам відокремлювати робочий процес та експериментувати з новими функціями, не впливаючи на основний код. Гілки полегшують спільну роботу та управління різними напрямками розробки.

1. Стратегії створення та ведення гілок

* Feature Branch Workflow: Кожна нова функція розробляється у власній гілці.
* Gitflow Workflow: Визначені гілки для розробки, релізів та гарячих виправлень.
* GitHub Flow: Орієнтована на постійну поставку, де кожен новий пуш в гітхабі визначає новий деплой.

1. Основні команди управління гілками

* git branch: Перегляд гілок
* git branch <ім'я\_гілки>: Створення нової гілки
* git checkout <ім'я\_гілки>: Перехід до іншої гілки
* git merge <ім'я\_гілки>: Об'єднання гілок

1. Виявлення файлів для індексації

Використовуйте git add <ім'я\_файлу> або git add . для додавання файлів до індексу Git.

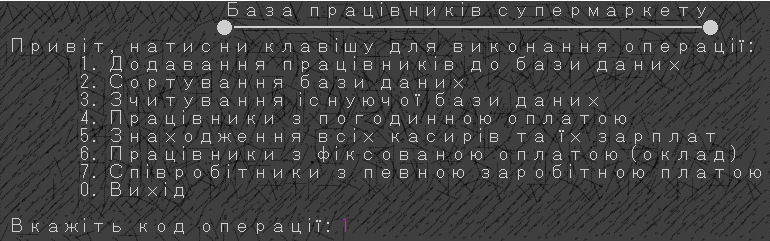
1. Фіксація змін

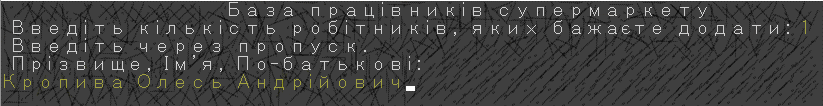
Використовуйте git commit -m "Повідомлення про коміт" для фіксації змін та створення нової версії.

1. Запит на зміни у віддалений репозиторій

Використовуйте git push origin <ім'я\_гілки> для відправлення своїх змін на віддалений репозиторій (наприклад, на GitHub чи GitLab).

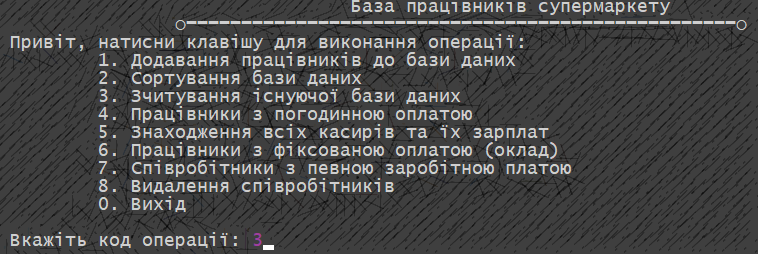
**source\_code\_front\_end.xaml**

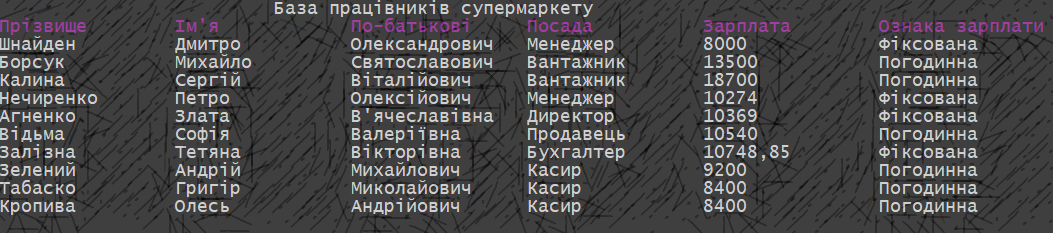


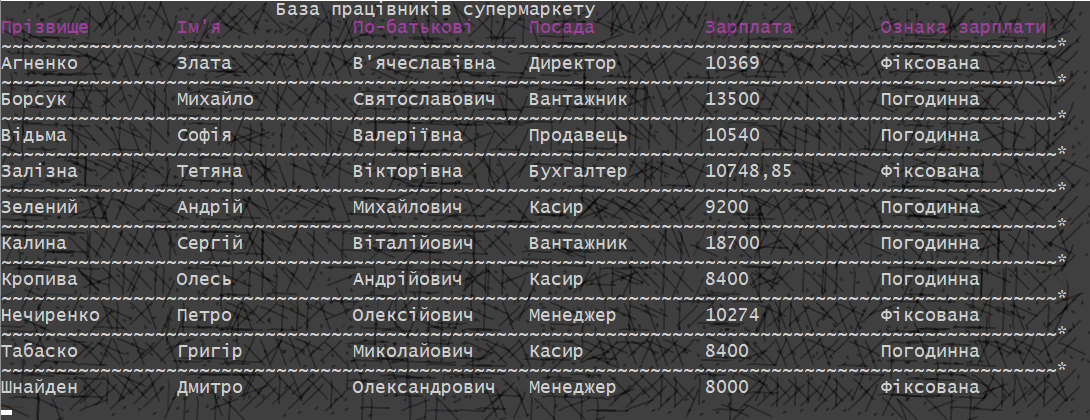












# source\_code\_back\_end.cs

Program.cs

using System;

using System.Collections.Generic; //list

using System.Threading.Tasks;

using System.IO; //input output string

using System.Linq;

using System.Text; //кодировка текста

namespace DataBase

{

public class MainClass

{

public static void Main(string[] args)

{

Encoding.RegisterProvider(CodePagesEncodingProvider.Instance);

var enc1251 = Encoding.GetEncoding(1251);

System.Console.OutputEncoding = System.Text.Encoding.UTF8;

System.Console.InputEncoding = enc1251;

string nameproject = "\t\t\t\tБаза працівників супермаркету";

int codeOperation;

do

{

Console.Clear();

//Console.BackgroundColor = ConsoleColor.White; //цвет фона консоли(маркер)

//Console.ForegroundColor = ConsoleColor.Green; //цвет текста консоли

//Console.ResetColor();

Console.WriteLine(nameproject);

Console.WriteLine("\t\t" +

Console.WriteLine(" Привіт, натисни клавішу для виконання операції: " +

"\n\t" + " 1. Додавання працівників до бази даних " +

"\n\t" + " 2. Сортування бази даних" +

"\n\t" + " 3. Зчитування існуючої бази даних" +

"\n\t" + " 4. Працівники з погодинною оплатою" +

"\n\t" + " 5. Знаходження всіх касирів та їх зарплат" +

"\n\t" + " 6. Працівники з фіксованою оплатою (оклад)" +

"\n\t" + " 7. Співробітники з певною заробітною платою" +

"\n\t" + " 8. Видалення співробітників" +

"\n\t" + " 9. Редагування даних співробітників" +

"\n\t" + " 0. Вихід");

Console.Write("\n Вкажіть код операції: ");

Console.ForegroundColor = ConsoleColor.DarkMagenta;

string operation = Console.ReadLine();

Console.ResetColor();

Console.Clear();

if (int.TryParse(operation, out codeOperation))

{

codeOperation = Convert.ToInt32(operation);

switch (codeOperation)

{

case 1:

Console.WriteLine(nameproject);

AddWorker();

break;

case 2:

Console.WriteLine(nameproject);

SortDescending();

break;

case 3:

Console.WriteLine(nameproject);

ReadFile();

break;

case 4:

Console.WriteLine(nameproject);

houremployer();

break;

case 5:

Console.WriteLine(nameproject);

cashier();

break;

case 6:

Console.WriteLine(nameproject);

fixemployer();

break; …