**МІНІСТЕРСТВО**

**ОСВІТИ ТА НАУКИ УКРАЇНИ**

**Національний технічний університет України**

**«Київський політехнічний інститут імені Ігоря Сікорського»**

**Кафедра інформаційних систем та технологій**

**Звіт**

**Лабораторна робота 1**

**Тема: Розробка веб-сайт прогнозу погоди**

# «Програмування – 2. Основи програмування»

|  |  |  |
| --- | --- | --- |
| **Перевірив: Драган М.С.** |  | **Виконали: Ферчук Кіра, Постоюк Роман** |
|  |  | **Студенти гр. ІС-43 ФІОТ** |
|  |  | **1 курс,**  **залікова книжка № IC-4310** |

**Київ 2025**

1. Назва застосунку;

*Градусфак / Gradusfuck*

2. Мета (які задачі вирішує);

Мета даного проекту полягає у наданні користувачам актуальну інформацію про погодні умови. Основні задачі:

* **Отримання та відображення прогнозу**

- показувати поточну температуру, вологість, швидкість вітру тощо.

* **Простий інтерфейс**

- зробити зручний, мінімалістичний дизайн для швидкого доступу до інформації

* **Актуальні джерела**

- використовувати АРІ для отримання даних

3. Користувачі та їх можливості;

**адміністратор**

* Керування джерелами даних – вибір API для отримання погоди, можливість змінювати постачальника даних.
* Редагування контенту – зміна текстів, описів, порад користувачам, банерів тощо.
* Моніторинг роботи сайту – перегляд логів, статистики відвідувань, завантаження сторінок.
* Налаштування дизайну – зміна теми, кольорів, розташування блоків

**звичайний користувач**

* Перегляд погоди – отримання актуальної інформації про температуру, опади, вітер тощо.
* Пошук погоди – введення назви міста або використання геолокації.
* Прогноз на кілька днів – перегляд погоди на тиждень або більше.
* Вибір мови інтерфейсу (якщо підтримується).

4. Основні сутності (основні об'єкти, якими оперує застосунок, та їх поля);

**Фактична погода**

* Дата та час вимірювання
* Температура
* Вологість
* Швидкість вітру
* Іконка стану погоди

**Користувач**

* Власний ID
* Геолокація (Координати)
* Країна та місто

**Коментар (відгук про погоду)**

* Ім’я користувача
* Дата коментаря
* Текст коментаря
* Оцінка погоди ( від 1 до 5)

5. Можливості застосунку;

1. **Перегляд актуальної погоди**

* Відображення поточних погодних умов (температура, вологість, вітер тощо).
* Графічне представлення стану погоди (сонячно, дощ, сніг тощо).
* Підтримка різних одиниць вимірювання (°C/°F, м/с або км/год).

2. **Прогноз погоди**

* Детальний прогноз на кілька днів (наприклад, на 7 або 14 днів).
* Погодинний прогноз (наприклад, на 24 години).
* Відображення ймовірності опадів, рівня хмарності та УФ-індексу.

3**. Історія погоди**

* Доступ до архіву погоди за минулі дні, місяці та роки.
* Аналіз змін температури, опадів та інших параметрів у певному регіоні.

6. Вибір технології UI.

Blazor – сучасний фреймворк від Microsoft для створення інтерактивного веб-інтерфейсу на C#.

***Теоретичні питання та практичні навички***

1. Що таке CRUD і як він використовується у C#?;

CRUD – це чотири операції для роботи з даними (додавання записів у базу даних, Зчитування їх, Зміна даних та видалення)

1. Як правильно визначати сутності та їх взаємозв’язки?;

Визначення сутності базується на її структурі даних. Потрібно проаналізувати функціональні вимоги застосунку та віднайти зв’язки між сутностями

1. Яку технологію UI ви обрали та які вона має переваги?;

Ми обрали сучасний фреймворк від Microsoft Blazor WebAssembly, бо він дозволяє створювати інтерактивні клієнтські застосунки мовою С#. його основною перевагою являється повна інтеграція з .Net, що дозволяє використовувати численні бібліотеки та інструменти

1. Що таке Git та які основні функції він виконує?;

**Git** – це розподілена система контролю версій, яка дозволяє відстежувати історію змін у проекті та координувати роботу кількох розробників.

Основні функції:

* Об’єднання змін з різних гілок для формування єдиного проекту
* Збереження історії змін коду, що дозволяє повернутися до попередніх версій за потреби
* Робота з повною копією проекту, що підвищує автономність і забезпечує додаткове резервне копіювання.

1. Що таке commit, push, pull, merge та branch? Який повсякденний синтаксис використання зазначених команд?

**Commit** – фіксує та зберігає усі змін зроблені в проекті у локальному репозиторію

**Push** – відправляє коміти до віддаленого репозиторію для синхронізації зміни між різними користувачами та забезпечує резервної копії коду

**Pull** - завантажує останні зміни з віддаленого репозиторію та об'єднує їх з вашим локальним репозиторієм для отримування оновлення, зроблені іншими користувачами

**Merge –** об’єднує зміни з однієї гілки в іншу для інегрування змін в основну гілку

**Branch** – Створює нові гілки у репозиторії

Висновок: У цьому проекті ми освоїли основи роботи з GitHub, навчившись ефективно керувати версіями коду через коміти, гілки, push, pull, merge та branch, а також обрали підходящий фреймворк Blazor для створення застосунку мовою С#