Національний університет «Одеська політехніка» Інститут комп'ютерних систем

Кафедра інформаційних систем

**КУРСОВА РОБОТА**

з дисципліни «Об'єктно-орієнтоване програмування»

Тема: «Розробка веб-додатка для відстеження погоди у містах»

Студентка 2 курсу АІ-224 групи Спеціальності 122 – «Комп’ютерні

науки»

Дришлюк В. А.

(прізвище та ініціали)

Керівник ст.викл, к.т.н. Годовиченко

М.

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала Кількість балів: Оцінка: ECTS

Члени комісії

(підпис) (прізвище та ініціали)

(підпис) (прізвище та ініціали)

(підпис) (прізвище та ініціали)

Одеса 2024

ЗМІСТ

ЗАВДАННЯ НА КУРСОВУ РОБОТУ 3

ВСТУП 6

1. ТЕОРЕТИЧНІ ВІДОМОСТІ ПРО КЛАСИ ТА ОБ'ЄКТНО 7

ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ В JAVASCRIPT 7

* 1. Теоретичні відомості 7

1. ПРОГРАМНА РЕАЛІЗАЦІЯ ООП У ПРОЕКТІ 8
   1. Програмна реалізація ООП 8
2. ІНСТРУКЦІЯ КОРИСТУВАЧА 11
   1. Інструкція користувача 11

ВИСНОВКИ 14

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 15

ЗАВДАННЯ НА КУРСОВУ РОБОТУ

1. Відображення інформації про обліковий запис користувача с GitHub.
2. Відображення проєктів користувача і можливість на них перейти.
3. Повідомлення про помилки при отриманні даних про користувача. Тобто, код містить обробку помилок, виводячи повідомлення у разі невдалого запиту до сервера GitHub.
4. Динамічне оновлення інформації про користувача. При кожному пошуку нового користувача відображається актуальна інформація про акаунт.
5. Показується встановлений аватар користувача, якщо він є.
6. Проєкти відображається в якості гіперпосилань, що дає змогу перейти на кожен з них.
7. Показується інформація про користувача, для якого отримано його ім’я, опис профілю, встановлений аватар, кількість проєктів, підписників та слідкувань, і гіперпосилання на проєкти.

АНОТАЦІЯ

Загальна ідея проекту полягає в створенні простого веб-додатку для пошуку користувачів GitHub. Користувач може ввести ім'я користувача у відповідне поле, після чого застосунок використовує отримані дані з відкритого API GitHub для відображення інформації про цього користувача.

Проект включає клас “ githubSearch”, який містить методи для отримання та відображення даних користувачів. Ці дані відображаються на сторінці веб-додатку, де користувач може бачити ім'я, аватар, кількість репозиторіїв, кількість підписників та іншу інформацію про обраного користувача GitHub.

У коді також врахована обробка помилок під час отримання даних про користувача, що дозволяє виводити користувачу повідомлення про будь-які проблеми з підключенням до сервера GitHub.

ABSTRACT

The general idea of ​​the project is to create a simple web application for searching GitHub users. The user can enter a username in the corresponding field, after which the application uses data obtained from GitHub's open API to display information about that user.

The project includes the “githubSearch” class, which contains methods for retrieving and displaying user data. This data is displayed on the web application page, where the user can see the name, avatar, number of repositories, number of subscribers, and other information about the selected GitHub user.

The code also includes error handling when retrieving user data, allowing the user to be notified of any problems connecting to the GitHub server.

ВСТУП

У цьому проекті ми використовували об'єктно-орієнтований підхід до програмування на JavaScript, створюючи клас “githubSearch”, який містить методи для взаємодії з зовнішнім API, отримання даних про користувача та їх подальшого відображення на сторінці.

Проект використовує асинхронний підхід для взаємодії з сервером GitHub, щоб забезпечити оновлення інформації без перезавантаження сторінки. Ми також використовували проміси та метод “axios()” для виконання запитів до API та подальшої обробки отриманих даних.

Проект демонструє вміння у розробці клієнтських веб-додатків з використанням сучасних інструментів та технологій. Використання об'єктно- орієнтованого підходу допомагає ефективно організувати функціональність додатку і спрощує його розвиток і підтримку в майбутньому.

* 1. ТЕОРЕТИЧНІ ВІДОМОСТІ ПРО КЛАСИ ТА ОБ'ЄКТНО- ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ В JAVASCRIPT
     1. Теоретичні відомості

Об'єктно-орієнтоване програмування (ООП) є основою для багатьох сучасних мов програмування, таких як JavaScript, який широко використовується для розробки веб-додатків. ООП дозволяє створювати більш структурований, зрозумілий і підтримуваний код.

У JavaScript концепція класів та об'єктів застосовується для створення шаблонів та їх екземплярів. Класи визначають структуру та поведінку об'єктів, а об'єкти є конкретними реалізаціями цих класів. Наприклад, клас автомобіля може мати властивості, такі як марка, модель та рік випуску, а також метод для отримання інформації про автомобіль. Потім можна створити об'єкт цього класу, який представлятиме конкретний автомобіль.

Основні переваги ООП включають модульність, що дозволяє легко керувати та розуміти код, повторне використання коду, яке спрощує розробку нового функціоналу та розширення програми, а також можливість створення більш структурованих та масштабованих програм.

Використання класів та об'єктів у JavaScript допомагає розробникам створювати більш організовані та підтримувані програми, що є ключовим фактором для успішного створення складних веб-додатків.

* 1. ПРОГРАМНА РЕАЛІЗАЦІЯ ООП У ПРОЕКТІ
     1. Програмна реалізація ООП

У нашому проекті ми використовували об'єктно-орієнтований підхід для створення класу githubSearch, який відповідає за взаємодію з API GitHub та відображення отриманих даних на веб-сторінці. Розглянемо детальніше:

1. Клас githubSearch – цей клас є центральним елементом проекту і містить усю логіку для отримання та відображення даних про користувача. Використання класу дозволило нам створити шаблон для роботи з даними користувача, який можна було б легко використовувати та розширювати, наприклад додати можливість відображати локацію користувача, профіль в інших соціальних мережах або коли останній раз оновлювався профіль.
2. Методи класу – клас githubSearch містить методи для отримання даних користувача за допомогою API, відображення поточної інформації про користувача, а також обробки помилок при виконанні запитів до API.
3. Інкапсуляція та модульність – використання класу дозволило нам зберегти код, пов'язаний з роботою з API GitHub, у відокремленому компоненті. Це сприяє модульності та зрозумілості коду, оскільки функціональність пов'язана з отриманням даних зосереджена в одному місці.
4. Створення екземпляру класу – під час запуску додатку ми створюємо екземпляр класу githubSearch, передаючи йому API URL, який буде виступати основною часткою URL під час формування запиту до API. Решта підставляється у методі getUser. Цей екземпляр використовується для взаємодії з іншими частинами програми та відображення результатів на сторінці.

Розглянемо приклад роботи ООР у Javascript за допомогою класа вчителя (рис. 2.1).

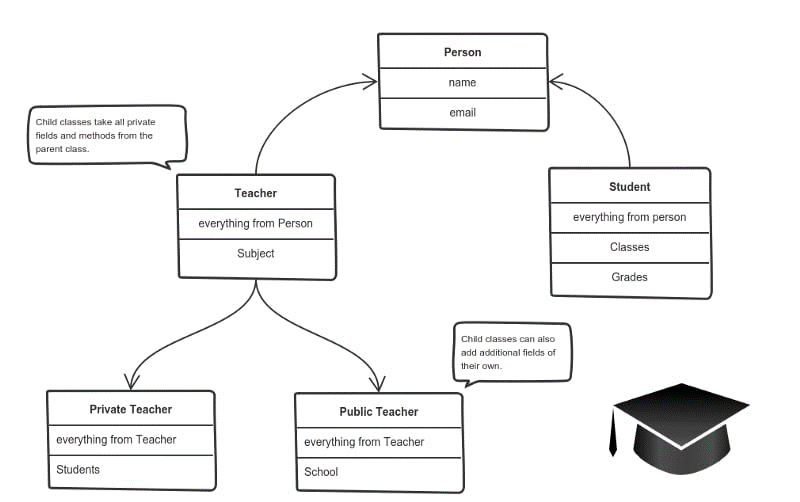


Рисунок 2.1 – Four Principles of Object-Oriented Programming

* + 1. Опис структури проекту

Наша структура проекту має такі ключові складові як:

1. Конструктор “constructor(apiUrl)” – конструктор класу, який приймає API URL як параметр і зберігає його у властивості “apiUrl”.
2. Метод “getColorName()” відповідає за повернення об’єкту кольору, який був обраний випадковим чином з масиву об’єктів, в якому були збережені кольори, які будуть використовуватись.
3. Метод “getUser(username)” відповідає за запит до API за допомогою axios() для отримання даних про користувача і викликає два методи: createUserCard(user) та getRepos(username). У разі невдачі показує помилку.
4. Метод “getRepos(username)” відповідає за запит до API за допомогою axios() для отримання даних о проєктах користувача. Викликає метод addReposToCard(repos). У разі невдачі показує помилку.
5. Метод “createUserCard(user)” відповідає за відображення блоку інформації о користувачі за допомогою інтерполяції та підстановки у неї даних з констант, які були оголошені спочатку методу.
6. Метод “createErrorCard(msg)” відповідає за відображення блоку з помилкою.
7. Метод “addReposToCard(repos)” відповідає за додавання гіперпосилань з проєктами до картки користувач
   1. ІНСТРУКЦІЯ КОРИСТУВАЧА
      1. Інструкція користувача

Після переходу на сайт у користувача з’являється користувач бачить перед собою поле для введення ім’я користувача (рис. 3.1).

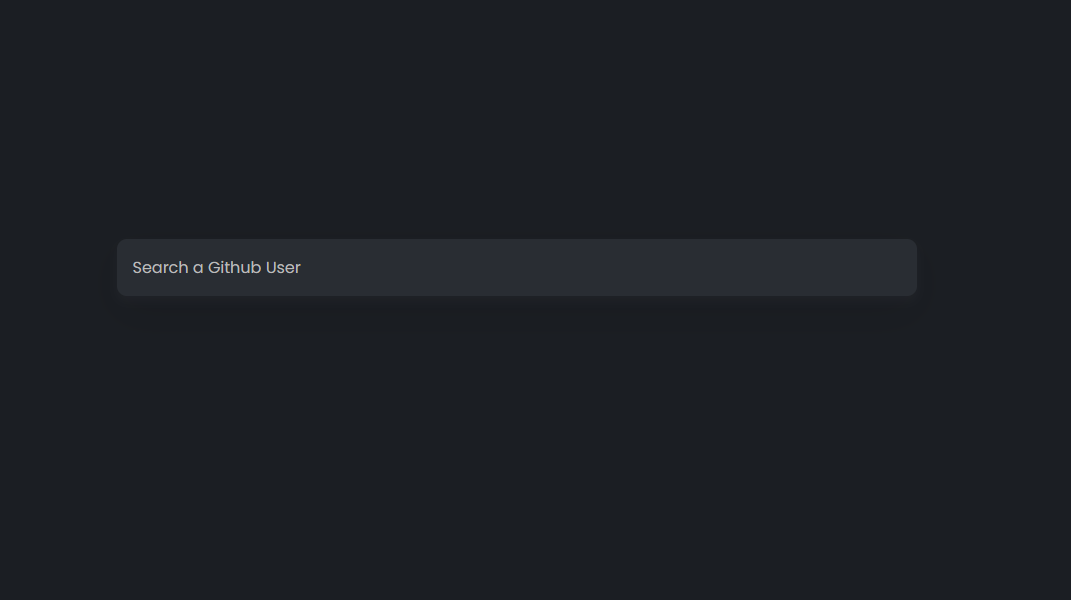


Рисунок 3.1 – Стартовий екран Користувач вводить ім’я користувача (рис 3.2).

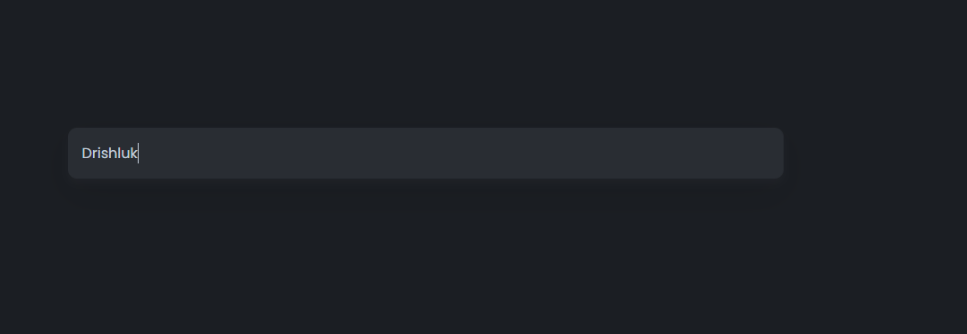


Рисунок 3.2 – Введення ім’я користувача

Після кліку на кнопку «Enter» виводиться вся інформація про користувача (рис 3.3).

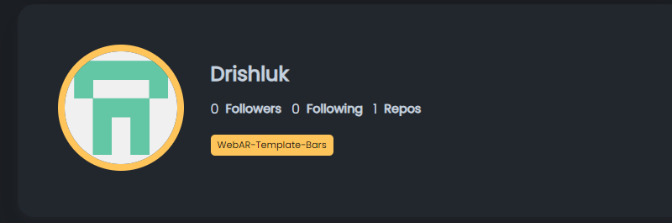


Рисунок 3.3 – Інформаційний блок користувача

Користувач може перейти через гіперпосилання на проєкт (рис 3.4).

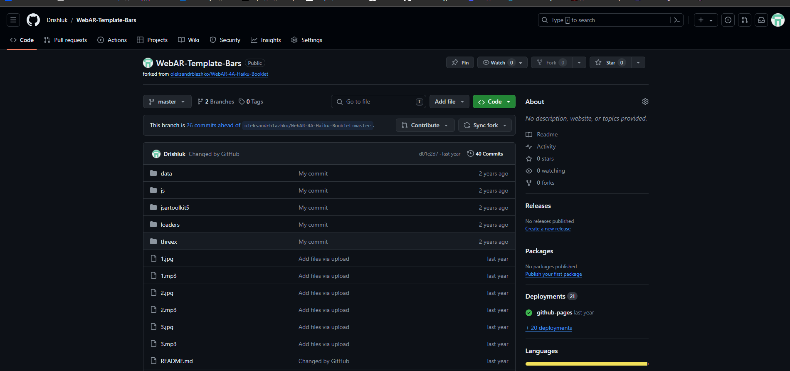


Рисунок 3.4 – Перехід на проєкт

Далі користувач може продивитись інший профіль (рис 3.5).

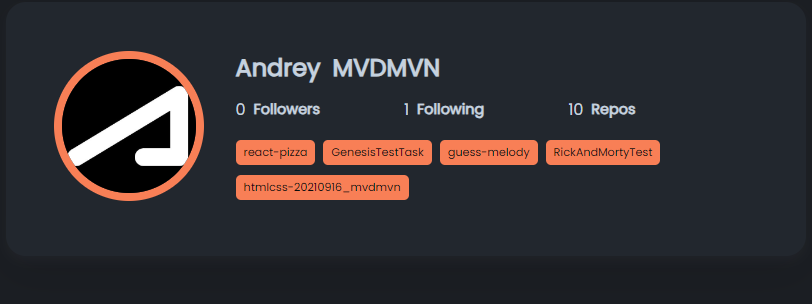


Рисунок 3.5 – Пошук іншого користувача

Теж після натискання на гіперпосилання, людина переходить на сторінку проєкту. (рис 3.6).

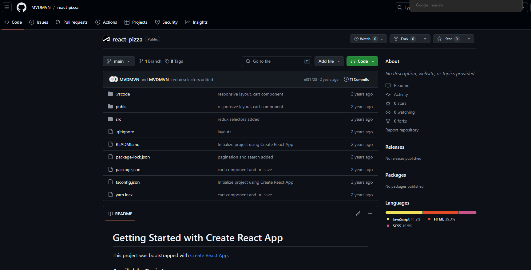


Рисунок 3.6 – Проект іншого користувача

ВИСНОВКИ

У цьому проекті я вивчила та застосувала об'єктно-орієнтований підхід до програмування на JavaScript. Створення класу “githubSearch” допомогло організувати код більш структуровано та ефективно. Використання методів цього класу для взаємодії з API GitHub та оновлення відповідних елементів HTML на сторінці дозволило створити функціональний веб-додаток для відображення даних користувача.

Проект надав мені можливість поглибити знання з JavaScript, зокрема вивчити асинхронне програмування та використання функції “axios()” для взаємодії з API. Також я отримала практичний досвід роботи з динамічним оновленням сторінки за допомогою JavaScript.

Крім того, я побачила важливість модульного підходу до розробки програмного забезпечення та можливість розширення функціональності проекту без значних змін в основному коді. В цілому, цей проект дав мені важливий досвід роботи з клієнтською частиною веб-додатків та збагатив мої знання у сфері веб-розробки.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Javascript Four Principles of OOP, URL: https://www.freecodecamp.org/news/object-oriented-programming-concepts-21bb035f7260/
2. Сучасний підручник з JavaScript, URL: https://uk.javascript.info/
3. JavaScript Підручник. Основи вебпрограмування, URL: <https://w3schoolsua.github.io/js/index.html#gsc.tab=0>
4. GitHub REST API documentation, URL: https://docs.github.com/en/rest?apiVersion=2022-11-28