

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ

ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

# Кафедра системного програмування та спеціалізованих комп’ютерних систем

**Лабораторна робота №3**

з дисципліни **Веб-дизайн**

*на тему:* **“Розробка Web-додатка засобами Javascript/ReactJS”**

Виконав:

студент ІII курсу

групи КВ-11

Рибалка Д. В.

Перевірив:

Київ – 2024

*Мета:* знайомитись із засобами фреймворка VueJS та навчитись створювати асинхронні запити до Web-сервера.

*Загальне завдання:* адаптувати програмний код ЛР№2 до вимог фреймворка ReactJS та забезпечити завантаження необхідних даних з Web-сервера

*Посилання на Github-репозиторій*:

<https://github.com/denisrybalka-kpi/web-lab3>

*Контакт студента у месенджері Телеграм*:

<https://t.me/denisrybalka10>

**Назва додатку:**

IdeaHub

**Короткий опис його функціональності:**

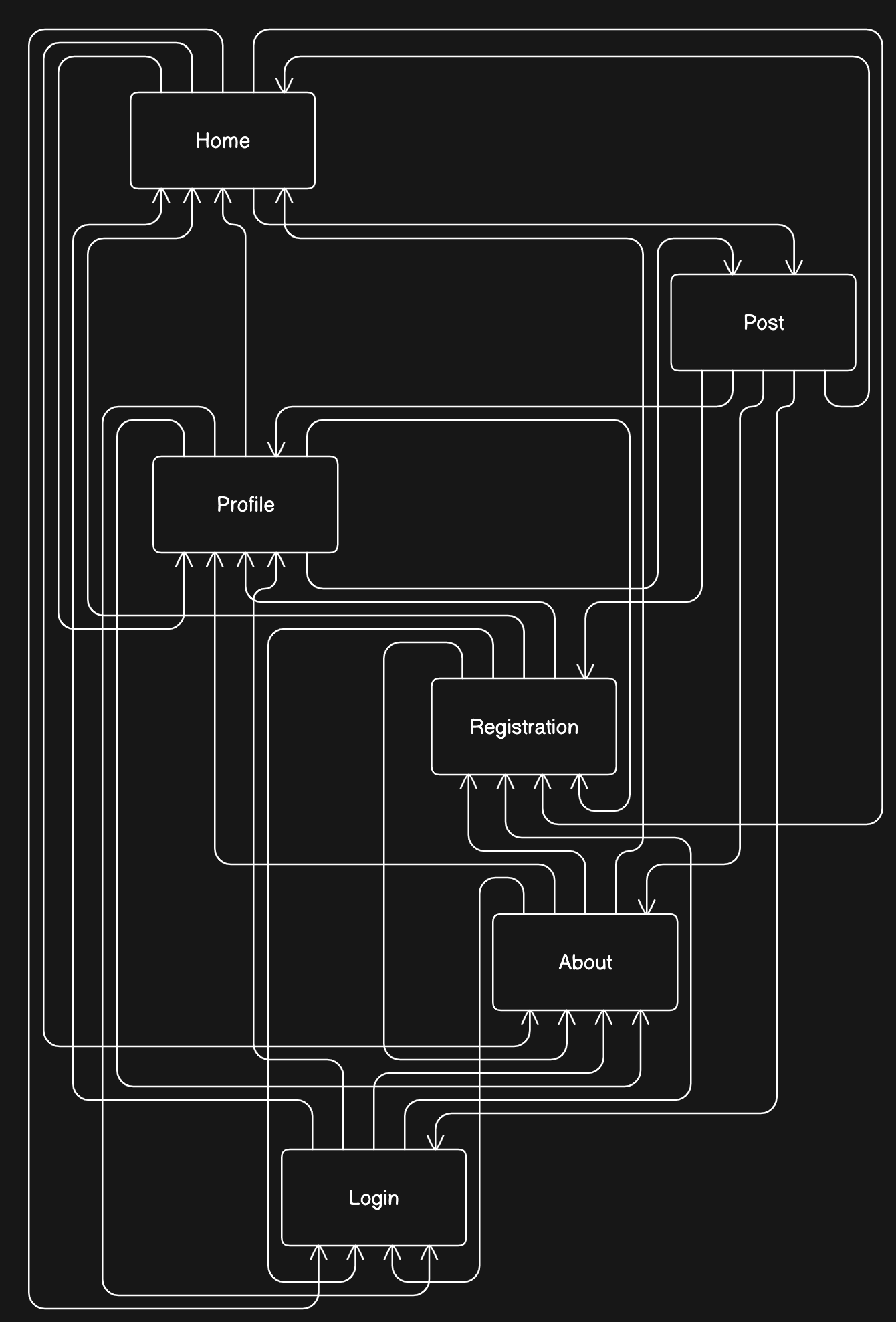
Цей веб-додаток представляє ідею соціальної платформи для обміну ідеями та думками. Користувачі можуть додавати свої дописи, лайкати дописи інших, використовувати пагінацію та пошук по назві дописів.

**Інструменти розробки:**

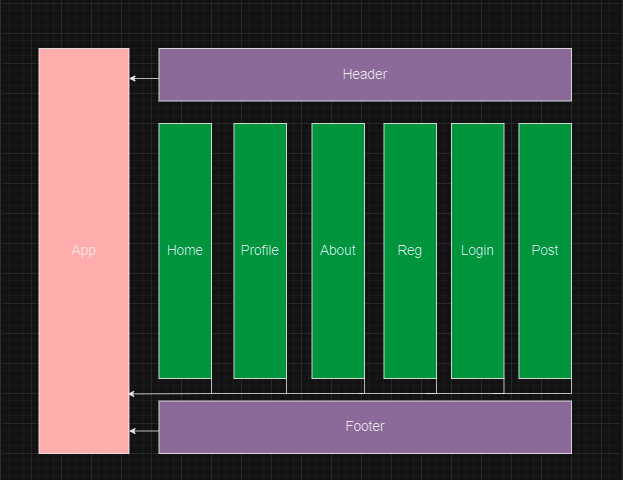
*Frontend: JS, React, React-Query, React-Router-Dom*

*Backend: Express.js, SQLite*

**Схема переходу:**



**Схема взаємодії React-компонентів:**



**Скріншоти усіх сторінок веб-додатку:**

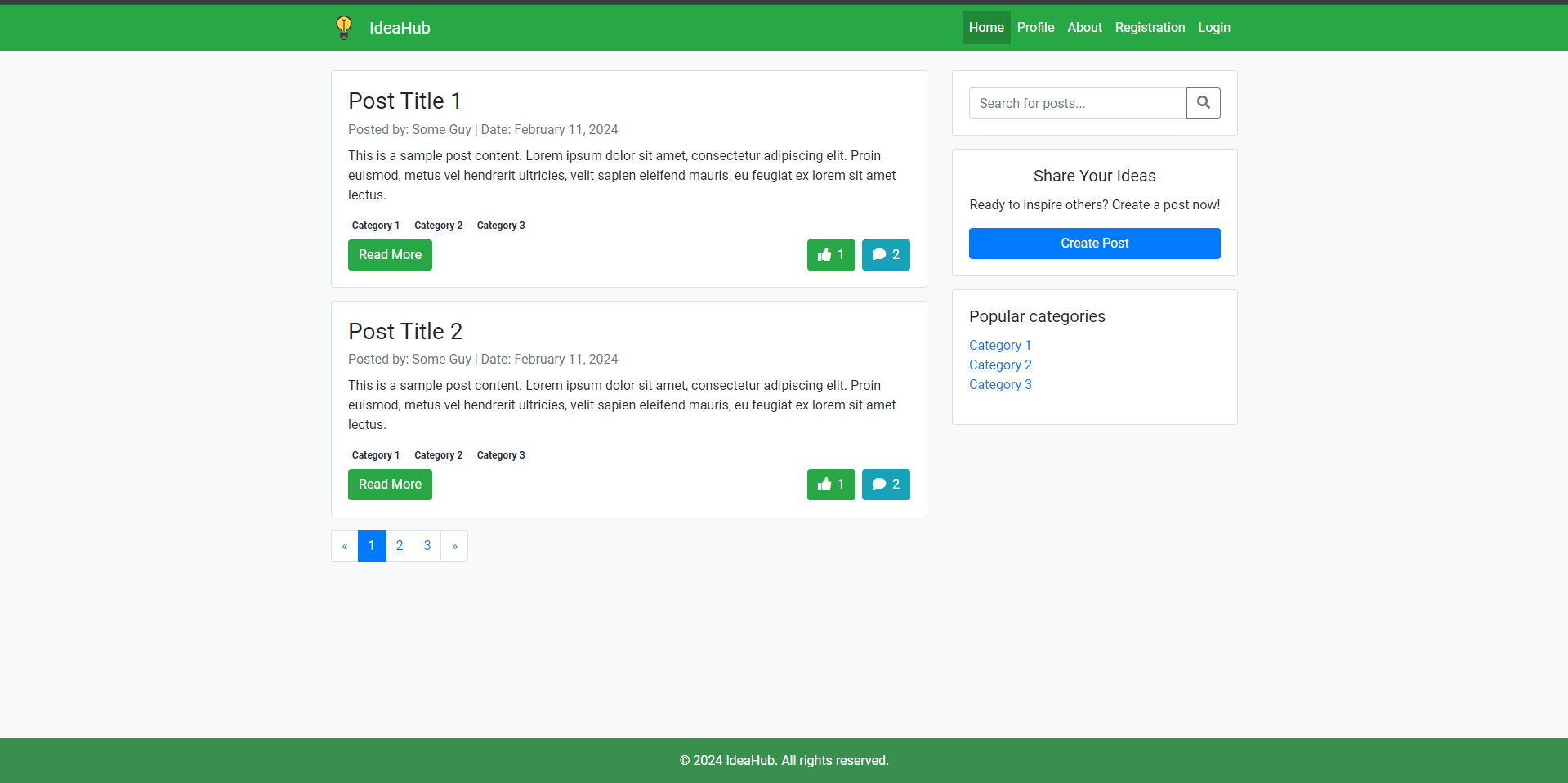


Рисунок 1 – Головна сторінка додатку

Рисунок 2 – Сторінка профілю

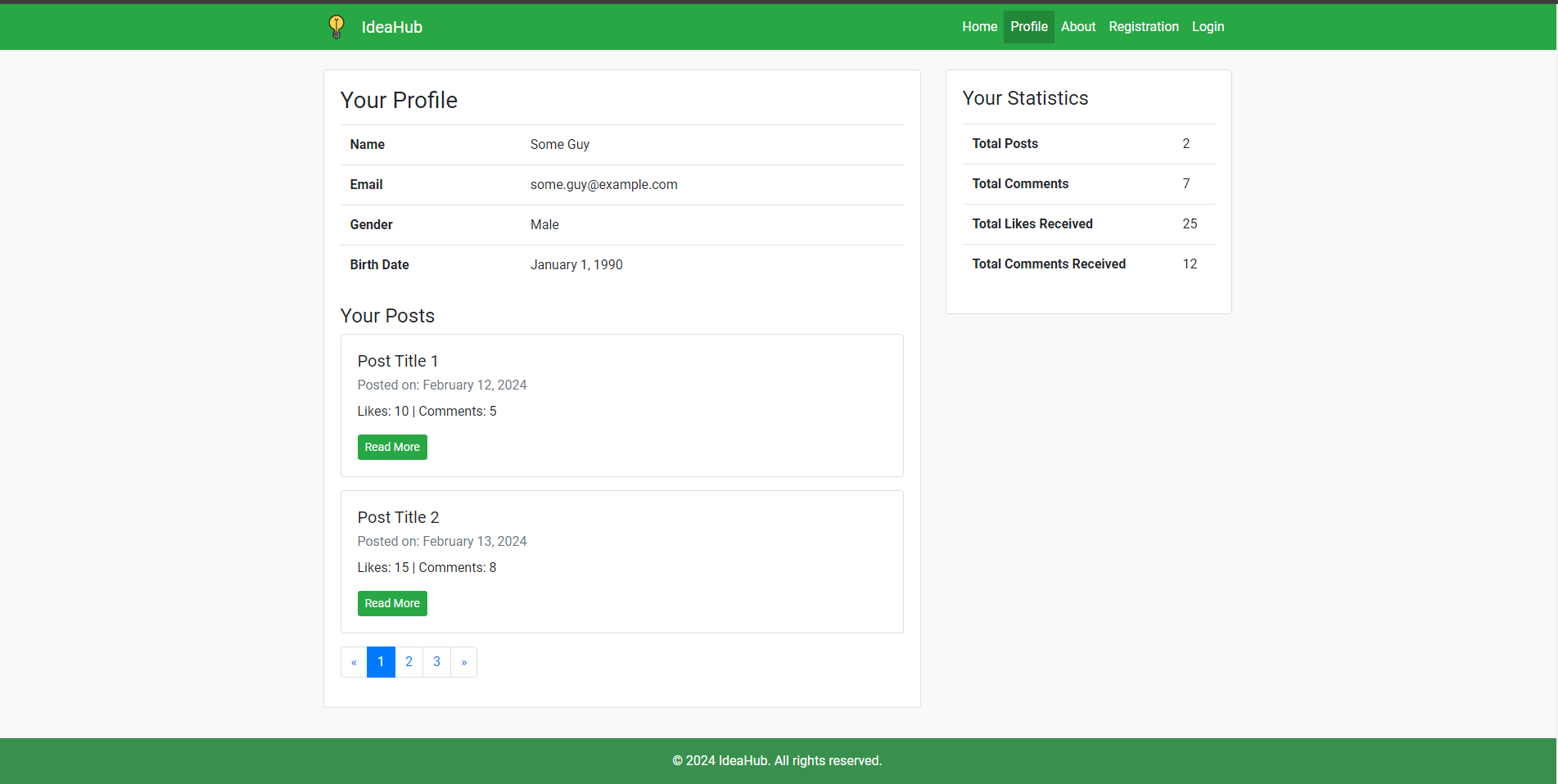


Рисунок 3 – Сторінка про додаток

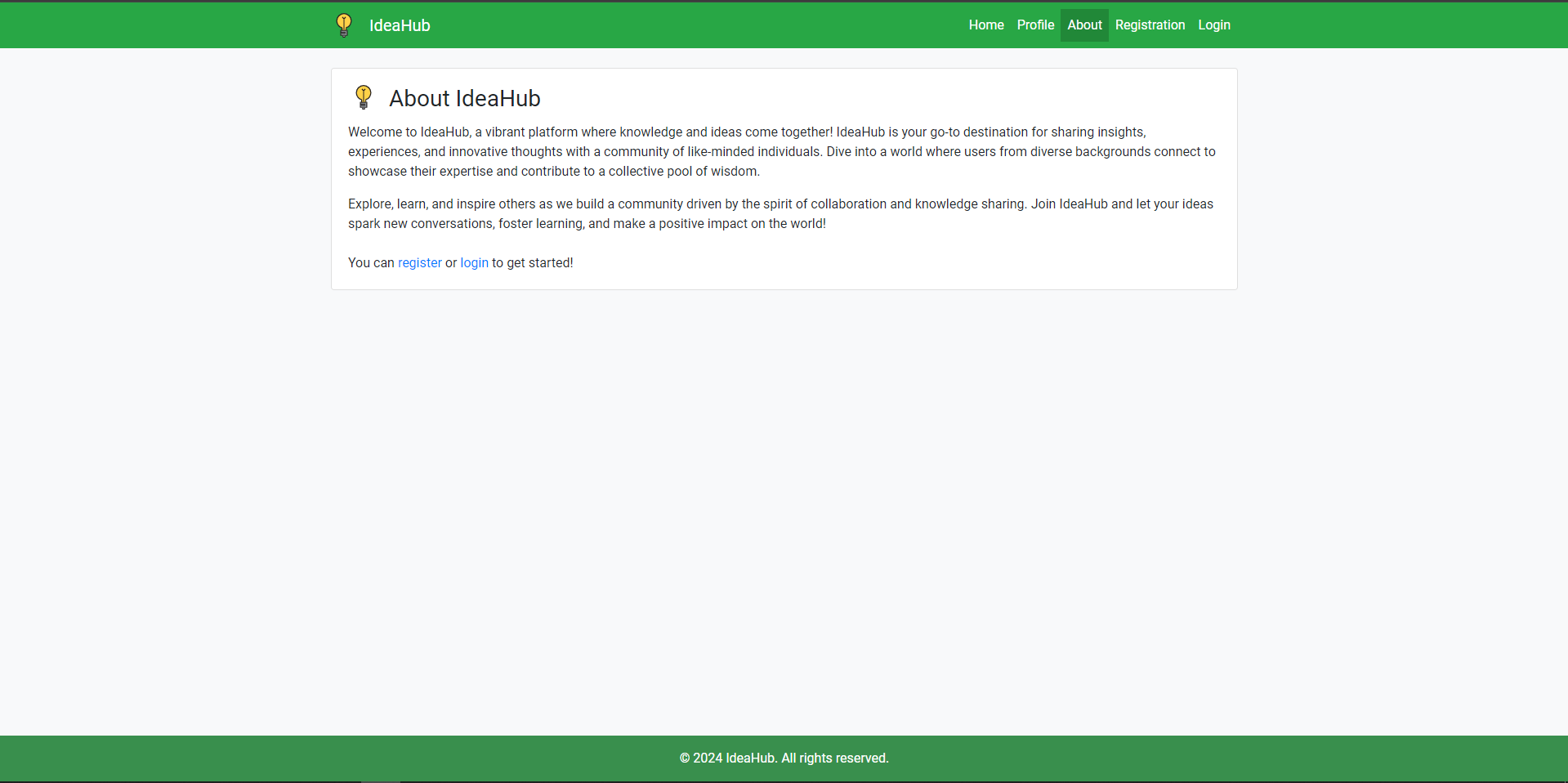


Рисунок 4 – Сторінка реєстрації

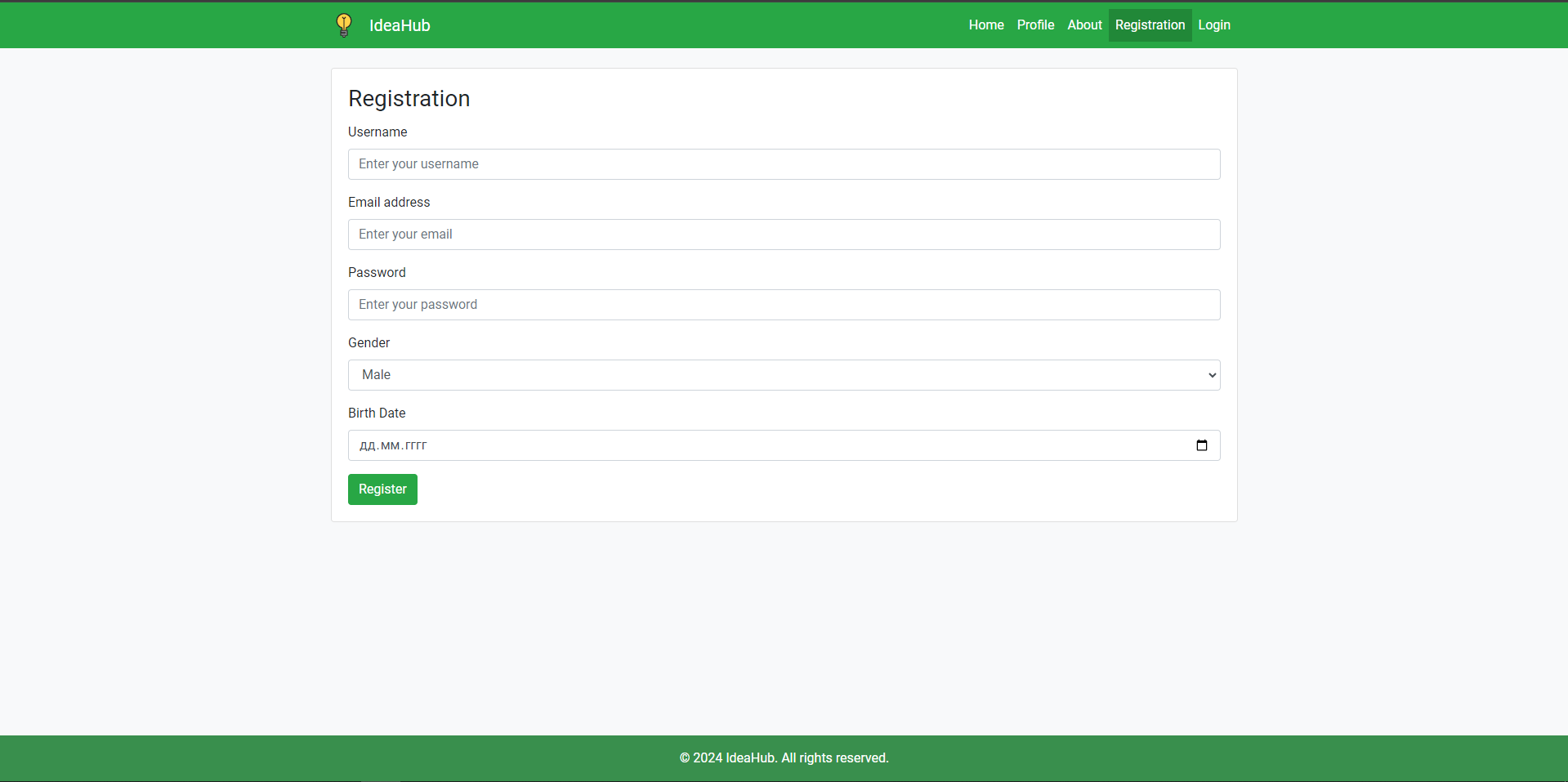


Рисунок 5 – Сторінка логіну

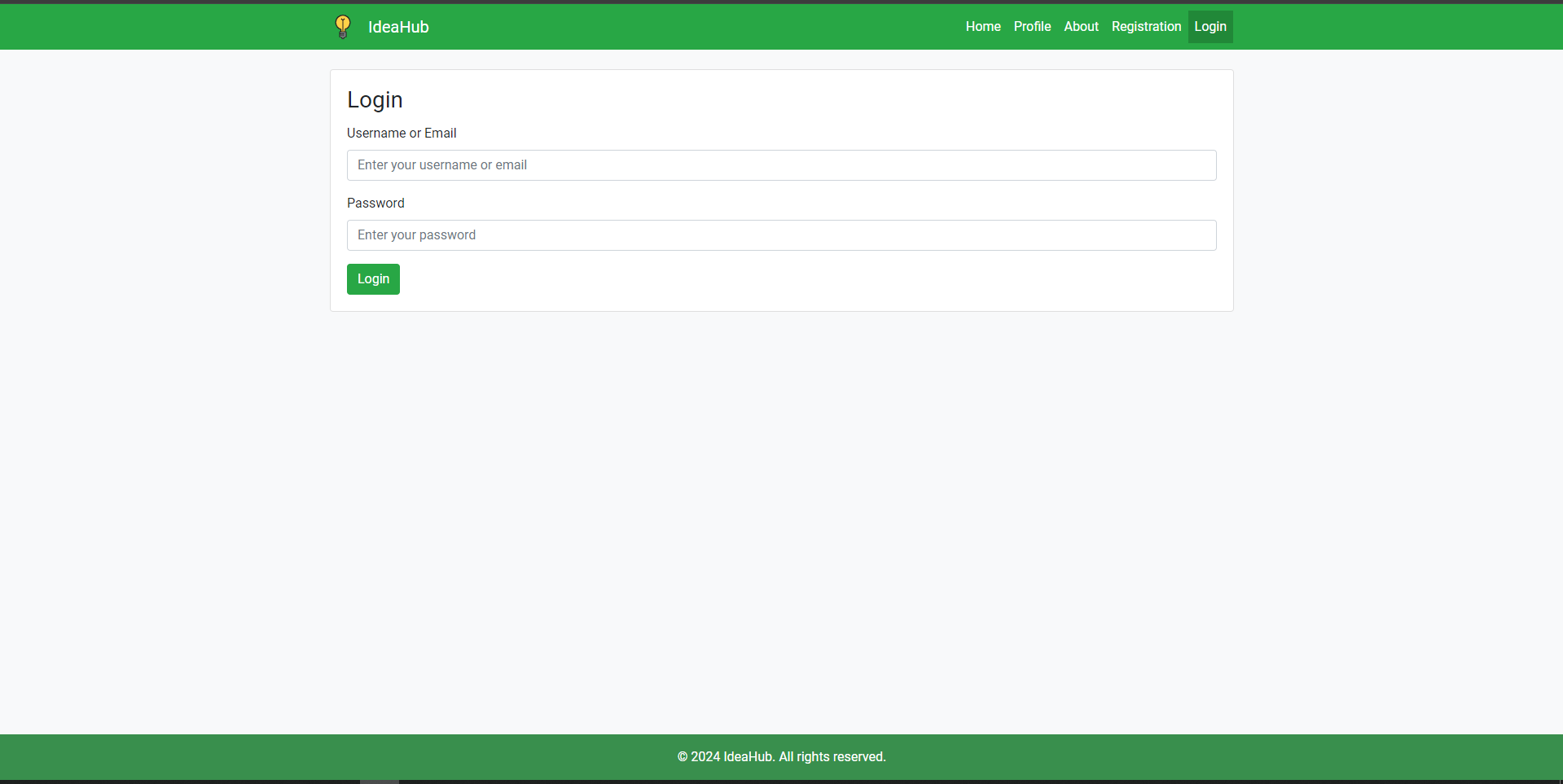


Рисунок 6 – Сторінка перегляду посту

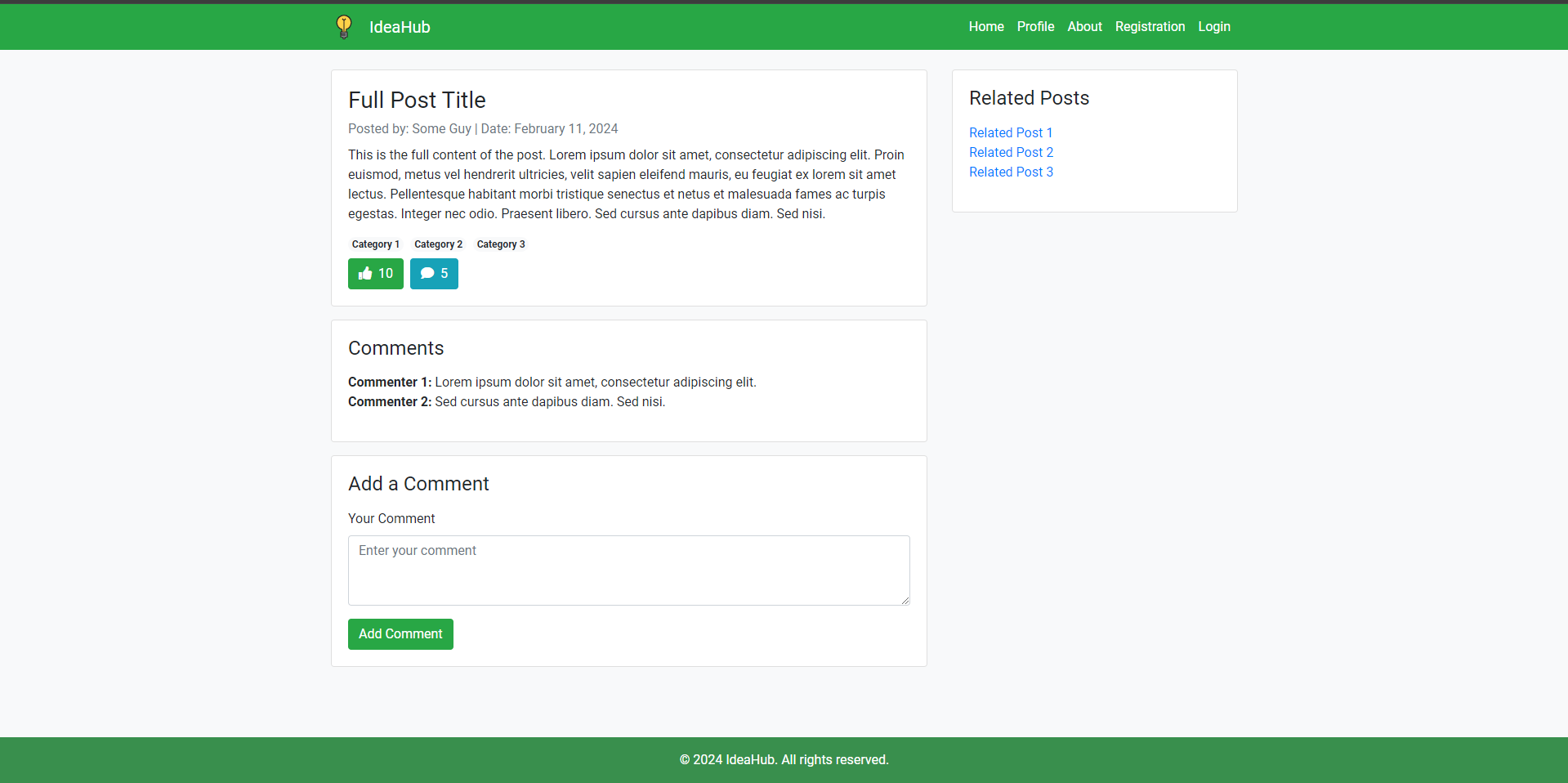


Рисунок 7 – Модальне вікно для створення посту

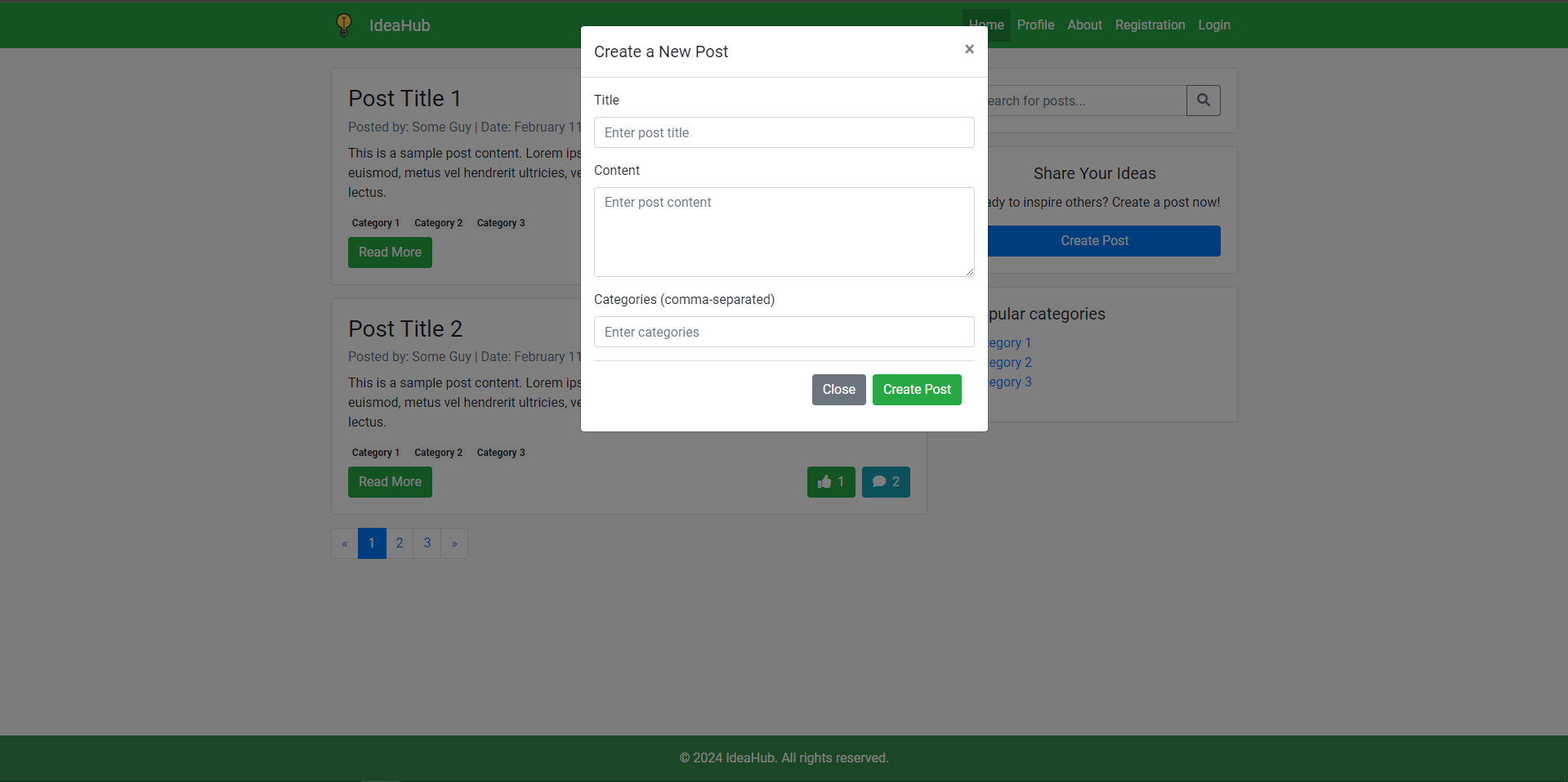
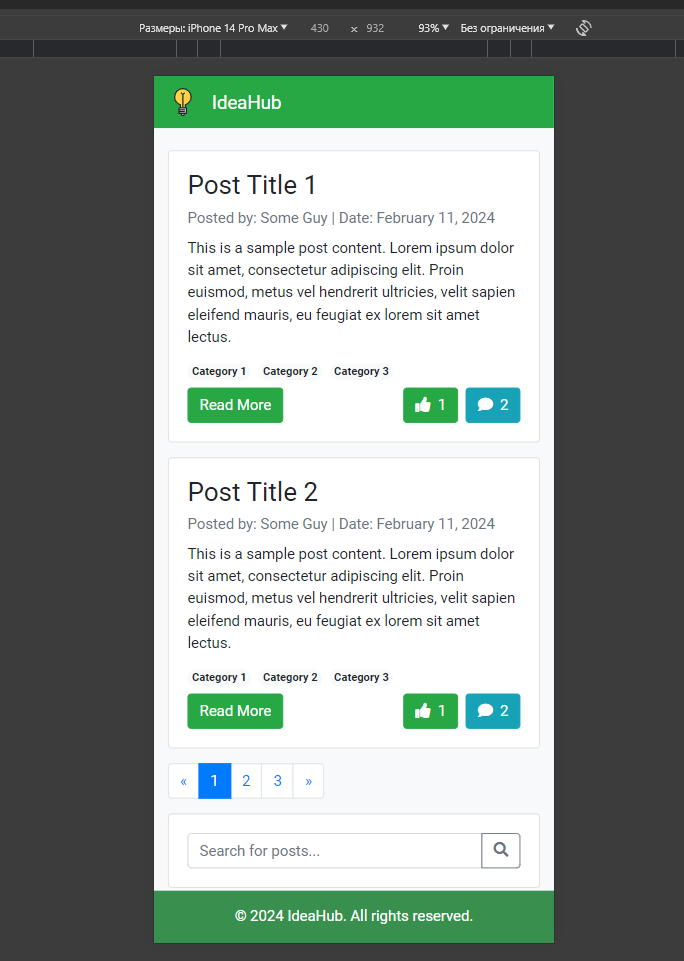


Рисунок 8 – Перегляд головної сторінки на мобільному



**Код backend частини (express.js + sqlite)**

*server.js:*

const express = require("express");

const bodyParser = require("body-parser");

const cors = require("cors");

const bcrypt = require("bcrypt");

const app = express();

app.use(cors());

app.use(bodyParser.json());

const PORT = process.env.PORT || 3001;

const sqlite3 = require("sqlite3").verbose();

const db = new sqlite3.Database("./myapp.db", (err) => {

if (err) {

console.error(err.message);

}

console.log("Connected to the myapp.db database.");

});

db.serialize(() => {

db.run(`DROP TABLE IF EXISTS users`); // This line is just for development purposes, remove it in production

db.run(`CREATE TABLE IF NOT EXISTS users (

id INTEGER PRIMARY KEY AUTOINCREMENT,

username TEXT NOT NULL,

email TEXT NOT NULL UNIQUE,

password TEXT NOT NULL,

gender TEXT NOT NULL,

birthdate DATE NOT NULL,

totalPosts INTEGER NOT NULL,

totalComments INTEGER NOT NULL,

totalLikesReceived INTEGER NOT NULL,

totalCommentsReceived INTEGER NOT NULL

)`);

db.run(`DROP TABLE IF EXISTS posts`); // This line is also just for development purposes, remove it in production

db.run(`CREATE TABLE IF NOT EXISTS posts (

id INTEGER PRIMARY KEY AUTOINCREMENT,

title TEXT NOT NULL,

content TEXT NOT NULL,

author\_id INTEGER,

FOREIGN KEY (author\_id) REFERENCES users (id)

)`);

});

app.post("/register", async (req, res) => {

const { username, email, password, gender, birthdate } = req.body;

if (!(email && password && username && gender && birthdate)) {

return res.status(400).send("All input is required");

}

const encryptedPassword = await bcrypt.hash(password, 10);

db.run(

`INSERT INTO users (username, email, password, gender, birthdate, totalPosts, totalComments, totalLikesReceived, totalCommentsReceived) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)`,

[username, email, encryptedPassword, gender, birthdate, 0, 0, 0, 0],

function (err) {

if (err) {

console.error(err.message);

return res.status(500).send("Failed to create user");

}

res.status(201).send("User created successfully");

}

);

});

app.post("/login", (req, res) => {

const { username, password } = req.body;

if (!(username && password)) {

return res.status(400).send("All input is required");

}

db.get(

`SELECT \* FROM users WHERE username = ?`,

[username],

async (err, user) => {

if (err) {

console.error(err);

return res.status(500).send("Internal Server Error");

}

if (user && (await bcrypt.compare(password, user.password))) {

res.status(200).json({ message: "Login successful", user });

} else {

res.status(400).send("Invalid Credentials");

}

}

);

});

app.post("/posts", (req, res) => {

const { title, content, author\_id } = req.body;

if (!title || !content || !author\_id) {

return res.status(400).send("Missing title, content, or author ID");

}

const sql = `INSERT INTO posts (title, content, author\_id) VALUES (?, ?, ?)`;

const params = [title, content, author\_id];

db.run(sql, params, function (err) {

if (err) {

console.error(err.message);

return res.status(500).send("Failed to create new post");

}

res.status(201).send({ id: this.lastID });

});

});

app.get("/posts", (req, res) => {

const sql =

"SELECT p.id, p.title, p.content, u.username AS author FROM posts p JOIN users u ON p.author\_id = u.id";

db.all(sql, [], (err, posts) => {

if (err) {

console.error(err.message);

return res.status(500).send("Failed to retrieve posts");

}

res.json(posts);

});

});

app.listen(PORT, () => {

console.log(`Server is running on port ${PORT}`);

});

*Код компонента Home, frontend частини:*

import React, { useEffect, useState } from "react";

import Post from "../../components/Post/Post";

import Navigation from "../../components/Navigation/Navigation";

import SearchBar from "../../components/SearchBar/SearchBar";

import ShareIdeas from "../../components/ShareIdeas/ShareIdeas";

import PopularCategories from "../../components/PopularCategories/PopularCategories";

import CreatePostModal from "../../modals/CreatePost";

import useModal from "../../hooks/useModal";

import { useMutation } from "react-query";

import { fetchPosts } from "../../utils/requests";

const Home = () => {

const [posts, setPosts] = useState([]);

const { openModal, closeModal, isModalOpen } = useModal();

const fetchMutation = useMutation(fetchPosts, {

onSuccess: (data) => {

setPosts(data);

},

});

useEffect(() => {

fetchMutation.mutate();

}, []);

return (

<>

<section className='container mt-4' style={{ marginBottom: "100px" }}>

<div className='row'>

<div className='col-lg-8'>

{posts.map((post) => (

<Post {...post} />

))}

<Navigation />

</div>

<div className='col-lg-4'>

<SearchBar />

<ShareIdeas openAddPostModal={() => openModal("addPostModal")} />

<PopularCategories />

</div>

</div>

</section>

<CreatePostModal

opened={isModalOpen("addPostModal")}

onClose={() => closeModal("addPostModal")}

/>

</>

);

};

export default Home;