

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»  
Кафедра автоматизації проектування енергетичних процесів і  
систем

## КУРСОВА РОБОТА

з дисципліни: «Основи web-програмування - 1»  
(назва дисципліни)  
на тему: Розробка сайту-форуму

Студента(-ки) групи ТІ-81  
напряму підготовки **Програмна інженерія**

Андрєєв Д М  
(прізвище та ініціали)

Керівник Титенко С.В.  
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна оцінка \_\_\_\_\_  
Кількість балів: \_\_\_\_\_ Оцінка: ECTS \_\_\_\_\_

Члени комісії

\_\_\_\_\_  
(підпис) (вчене звання, науковий ступінь, прізвище та ініціали)

\_\_\_\_\_  
(підпис) (вчене звання, науковий ступінь, прізвище та ініціали)

\_\_\_\_\_  
(підпис) (вчене звання, науковий ступінь, прізвище та ініціали)

Київ- 2020 рік

**Національний технічний університет України**  
**“Київський політехнічний інститут імені Ігоря Сікорського”**

Факультет

ТЕПЛОЕНЕРГЕТИЧНИЙ

( повна назва )

Кафедра автоматизації проектування енергетичних процесів та систем

( повна назва )

Напрямок підготовки Програмна інженерія

(шифр і назва)

**З А В Д А Н Н Я**  
**НА КУРСОВУ РОБОТУ СТУДЕНТУ**

Андрєєву Данилу Максимовичу

(прізвище, ім'я, по батькові)

1. Темароботи

Керівник курсової роботи – ст.вик. Титенко С.В.

( прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

2. Строк подання студентом роботи

3. Вихідні дані до проекту (роботи): середовище розробки –

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) –

**КАЛЕНДАРНИЙ ПЛАН**

№	Назва етапів виконання дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Написання ТЗ	31.03.2020	
2	Розробка дизайну сайту	22.04.2020	
3	Розробка бета-версії сайту	06.05.2020	
4	Оформлення пояснювальної записки	26.05.2020	
5	Презентація курсової роботи	29.05.2020	
6			
7			

Студент

Андрєєв Д М  
( підпис ) (прізвище та ініціали)

Керівник курсової роботи

( підпис ) (прізвище та ініціали)

## АНОТАЦІЯ

При виконанні курсової роботи на тему: «Розробка сайту-форуму», командою було прийнято рішення розподілити роботу на 3 частини (дизайн, фронт-енд, бек-енд), при чому кожен з членів команди відповідав за свою частину, але усі головні рішення повинні обговорюватися та прийматися усім складом команди.

Склад команди:

Андрєєв Д.М. – фронт-енд

Нестеров М.А. – дизайн

Соломаха О.О. – бек-енд

У пояснювальній записці описується поетапне створення сайту для курсової роботи.

## ANOTATION

During the course work on the topic: "Development of the site-forum", the team had decided to divide the work into 3 parts (design, front-end, back-end), with each member of the team responsible for their part, but all the main decisions should be discussed and accepted by the entire team.

Team members:

Andreev D.M. – front-end

Nesterov M.A. – design

Solomaha O.O. – back-end

The explanatory note describes the step-by-step creation of a site for course work.

# Зміст

Анотація .....	3
Зміст.....	4
Вступ.....	5
Основна частина .....	6
1.Теоретичні відомості.....	6
1.1 Основні етапи розробки сайту.....	6
1.2 Інструменти використані для розробки.....	8
2. Тема проекту.....	11
2.1 Постановка завдання.....	11
2.2 Інтернет форум.....	13
2.3 Функціонал сайту.....	14
2.4 Створення Фронт-енду.....	14
2.5 Опис засобів виконання індивідуальної роботи.....	21
2.6 Опис результатів загальної веб системи.....	24
Висновок.....	26
Список використаних джерел.....	2
Додаток 1 Код Крограми.....	28

## **Вступ**

З часу створення першого сайту минуло чимало часу, і навколишній світ достатньо змінився. Сталося це завдяки розвитку технологій, появі нових мов програмування та фреймворків, які давали змогу робити сайти кращими ніж вони були. По мірі розвитку технологій стали збільшуватися і бажання людей. Їм було недостатньо веб-сторінки, яка виглядає лише як документ. Стали з'являтися дизайни, інтернет подолав великий шлях з чорно – білого до того, яким ми його знаємо зараз. З появою смартфонів, люди й самі того не помітили як вони перейшли до того, що зараз більшість інтернет трафіку йде з мобільних пристроїв. Так з'явилася ще більша потреба у адаптивній верстці і розробці мобільних версій сторінок, які нічим не уступають десктопним.

## **Основна частина**

### ***1. ТЕОРЕТИЧНІ ВІДОМОСТІ***

#### ***1.1 Основні етапи розробки сайту***

##### **Створення технічного завдання**

Складанням технічного завдання для фахівців займається менеджер проекту. Робота з замовником починається з заповнення брифу, в якому замовник викладає свої побажання щодо візуального представлення і структури сайту, вказує на помилки в старій версії сайту, наводить приклади сайтів конкурентів. Виходячи з брифа, менеджер складає технічне завдання, враховуючи можливості програмних і дизайнерських засобів. Етап закінчується після затвердження технічного завдання замовником.

##### **Дизайн основний і типових сторінок сайту**

Починається робота зі створення дизайну, зазвичай в графічному редакторі. Дизайнер створює один або кілька варіантів дизайну, відповідно до технічного завдання. При цьому окремо створюється дизайн головної сторінки, і дизайни типових сторінок (наприклад: статті, новини, каталог продукції). Власне «дизайн сторінки» представляє собою графічний файл, листовий малюнок, що складається з найбільш дрібних картинок-шарів елементів загального малюнка.

При цьому дизайнер повинен враховувати обмеження стандартів HTML (не створювати дизайн, який потім не зможе бути реалізований стандартними засобами HTML). Виняток становить Flash-дизайн.

Кількість ескізів і порядок їх надання обмовляється з проект-менеджером. Також менеджер проекту здійснює контроль термінів. У великих веб-студіях в процесі бере участь арт-директор, який контролює якість графіки. Етап також закінчується затвердженням ескізу замовником.

## **HTML - верстка**

Затверджений дизайн передається HTML-верстальщику, який «нарізає» графічну картинку на окремі малюнки, з яких згодом складає HTML-сторінку. В результаті створюється код, який можна переглядати за допомогою браузера. А типові сторінки згодом будуть використовуватися як шаблони програмування.

Далі готові HTML-файли передають програмісту.

## **Завершальним етапом розробки сайту є тестування**

Процес тестування може включати в себе найрізноманітніші перевірки: вид сторінки зі збільшеними шрифтами, при різних розмірах вікна браузера, при відсутності флеш-плеєра і багато інших. Також – юзабіліті-тестування. Виявлені помилки відправляються на виправлення до тих пір, поки не будуть усунуті. Терміни контролює менеджер проекту. Також, на цьому етапі залучають до роботи дизайнера, щоб він провів авторський нагляд.

## **Розміщення сайту в Інтернеті**

Файли сайту розміщують на сервері провайдера (хостингу) і виробляють потрібні налаштування. На цьому етапі сайт поки закритий для відвідувачів. Сайт наповнюють контентом – текстами, зображеннями, файлами для скачування і т. Д. Іноді тексти складаються фахівцем студії, іноді контентом займається відповідальна особа з боку замовника. Це вирішується на етапі складання технічного завдання. У разі якщо контент складається представником студії, то це відбувається і затверджується паралельно з іншими етапами проекту.

## *1.2 Інструменти використані для розробки*

### **React JS**

React - JavaScript-бібліотека з відкритим вихідним кодом для розробки призначених для користувача інтерфейсів. React розробляється і підтримується Facebook, Instagram і співтовариством окремих розробників і корпорацій. React може використовуватися для розробки односторінкових і мобільних додатків.

React JS був створений у 2015 році компанією facebook, і з тих пір він набув значної популярності

### **Superagent**

Бібліотека Superagent підходить для сучасних браузерів. Вона надає розробнику просте і зрозуміле API, з яким зручно працювати. Дана бібліотека має дуже велику кількість сильних сторін і майже позбавлена недоліків. До плюсів використання superagent можна додати:

- Піддається конфігурації.
- Має приємний інтерфейс для виконання HTTP-запитів.
- Підтримує об'єднання в ланцюжок декількох викликів для виконання запитів.
- Підтримує індикацію прогресу для вивантаження і завантаження даних.
- Підтримує механізм chunked-transfer encoding.
- Підтримує коллбеки.
- Для цієї бібліотеки розроблено безліч плагінів.

### **RНР**



PHP - це широко використовувана мова сценаріїв загального призначення з відкритим вихідним кодом. Говорячи простіше, PHP це мова програмування, спеціально розроблений для написання web-додатків (сценаріїв), що виконуються на Web-сервері. Аббревіатура PHP означає "Hypertext Preprocessor (Препроцесор Гіпертексту)".

PHP був створений у 1994 році Расмусом Лердорфом коли той намагався дізнатися хто з роботодавців все ж таки читає його резюме. Почалося с того, що Лердорф розсилав потенційним роботодавцям своє міні-резюме з URL посиланням на його повну версію. Для того щоб зібрати інформацію о переходах на сторінку він створив CGI скрипт на мові програмування Perl і вставив його як тег до HTML коду своєї сторінки з резюме. З метою справлення враження на потенційних працедавців, Лердорф дозволив будь-якому відвідувачу переглядати зібрану статистику відвідувань. Цей код для збору статистики він назвав «PHP-Tools for Personal Home Page», оскільки сам використовував його на своїй персональній домашній сторінці. Декілька компаній були зацікавлені у отриманні такого інструменту, і творець скрипту погодився надати його іншим особам. Так бере свій початок одна з найпопулярніших та найефективніших мов програмування для web-додатків – PHP.

До сильних сторін цієї мови можна віднести:

- Вивчення PHP не вимагає багато часу
- Кроссплатформенність. PHP може бути запущений в будь-якій операційній системі, включаючи юніксоїди
- Підтримка веб-серверів. Складно знайти той, який би не працював з PHP
- Безкоштовне розповсюдження. Можливо, PHP не був такий популярний для створення web-додатків, якби не був безкоштовним, як і більшість інструментів для роботи з ним. Аналоги, які, в основному, виконують ту ж роботу, як правило коштують дорожче.

- Наявність великої кількості навчальних матеріалів.
- Неперервний розвиток

## **Laravel**

Laravel - безкоштовний веб-фреймворк з відкритим кодом, призначений для розробки з використанням архітектурної моделі MVC. Laravel випущений під ліцензією MIT. Вихідний код проекту розміщується на GitHub. Також Laravel є найпопулярнішим серед php фреймворків, в свою чергу php є найпопулярнішою мовою веб-програмування, якщо не найпопулярнішою серед всіх мов програмування а не тільки веб. Це робить Laravel найпопулярнішим фреймворком на сьогодні. Сьогодні найбільше проектів, що розробляються за допомогою фреймворків, створюються саме з використанням Laravel. При розміщенні вакансій PHP-розробників, веб-студії все частіше включають знання фреймворка Laravel як обов'язкову умову. Такої популярності Laravel досяг через:

- високий рівень безпеки
- Підвищена продуктивність. Кешування
- Шаблонизатор blade
- Міграції баз даних
- MVC-архітектура
- Інтеграція з mail-сервісом
- Помилки і виключення

І це навіть не половина з того, за що програмісти обирають саме Laravel.

## **Figma**

Figma – це онлайн-сервіс для розробки інтерфейсів і прототипування з можливістю організації спільної роботи в режимі реального часу. Сервіс має широкі можливості для інтеграції з корпоративним месенджером Slack і інструментом для високорівневого прототипування Framer. Дизайнеру важливі швидкість роботи над проектом і комунікація в процесі. Ці та багато інших завдань дозволяє вирішити онлайн-сервіс Figma. У Figma низький поріг входження і потужний потенціал при уявній простоті.

## **2. ТЕМА ПРОЕКТУ**

### *2.1. Постановка завдання*

Для виконання курсової роботи пропонувалось обрати собі команду з 2-3 людей та обрати тему, яка була би цікава для всіх членів команди. Таким чином була обрана тема розробки – «інтернет форум», і зібраний такий склад команди розробників:

- Андрєєв Д.М.
- Нестеров М.А.
- Соломаха О.О.

Після обговорення та обмірковування теми, було створено технічне завдання, яке би відображало усі основні задачі та вимоги до майбутнього сайту (веб-додатку). Технічне завдання було презентовано та узгоджено з керівником курсової роботи. До основних вимог сайту входило: розробка концептуальної моделі сайту, створення дизайну за розробленою моделлю, верстка сторінок за дизайном, розміщення кінцевої версії сайту на сервері(хостингу). До технічних вимог були віднесені такі аспекти, як адаптивність програмного продукту під різні масштаби та співвідношення екрану, а також його підтримка більшою кількістю сучасних веб браузерів (Google Chrome, Opera, Mozilla firefox).

Окрім цього, були описані функціональні можливості користувачем на різних сторінках веб-додатку. Так для розділу «усі питання», який за сумісністю також є і головною сторінкою, було визначено таке: має містити перелік усіх питань, які були створені на сайті, та можливість відсортувати їх за кількістю переглядів, або датою створення; має містити навігаційне меню по сайту, строку пошуку запитань за їх назвою, кнопку для створення питання. Для розділу із авторизацією користувачів було описане таке: можливість реєстрації (створення аккаунту) або логіну до вже існуючого аккаунту за допомогою електронної пошти та паролю, авторизація користувачів необхідна задля можливості створення своїх питань, або коментування питань інших; звичайний користувач, що ввійшов як гість має

можливість тільки переглядати контент без можливості створення або редагування. На сторінці профілю передбачено дії: зміни паролю, видалення існуючого аккаунту, зміни аватару, написання інформації про себе, зміна пошти та нікнейму. За головну мету поставлено досягнення найбільшої зручності у використанні продукту кінцевим користувачем.

## *2.2. ІНТЕРНЕТ ФОРУМ*

Суть роботи форуму полягає в створенні користувачами (відвідувачами форуму) своїх тем з їх подальшим обговоренням, шляхом розміщення повідомлень всередині цих тем. Користувачі можуть коментувати заявлену тему, задавати питання по ній і отримувати відповіді, а також самі відповідати на питання інших користувачів форуму і давати їм поради. Питання і відповіді зберігаються в базі даних форуму, і в подальшому можуть бути корисні як учасникам форуму, так і будь-яким користувачам мережі Інтернет, які можуть зайти на форум, знаючи адресу сайту, або отримавши його від пошукових систем при пошуку інформації. Тематика форумів може бути найрізноманітнішою, охоплюючи всі сфери життя, і визначається або власниками форуму або його адміністрацією. У випадку із, створюваним нашою командою, форумом була обрана тематика програмування. Тобто сайт орієнтований на програмістів, при чому не на всіх, а тільки на Android, Java та web розробників.

### 2.3. Функціонал сайту

Приймаючи до уваги те, що тематика розроблюваного сайту – інтернет форум, наша команда вирішила перед розробкою визначити головний функціонал який повинен бути реалізован:

- Можливість створити питання
- Можливість залишати відповіді
- Створення аккаунту
- Авторизація
- Зміна паролю
- Видалення аккаунту
- Сортування питань за кількістю переглядів, датою створення, популярністю
- Пошук питань
- Можливість редагування вже написаних відповіді та питань
- Можливість надання привілеїв адміну для редагування або видалення відповідей та питань, інформації о користувачах та їх аккаунтах.
- Додання аватарів до профілю

Усі ці можливості було успішно додано до сайту, а також протестовано. Таким чином сайт задовольняє усім необхідним потребам користувачів. Реалізовано це було за допомогою бази даних:

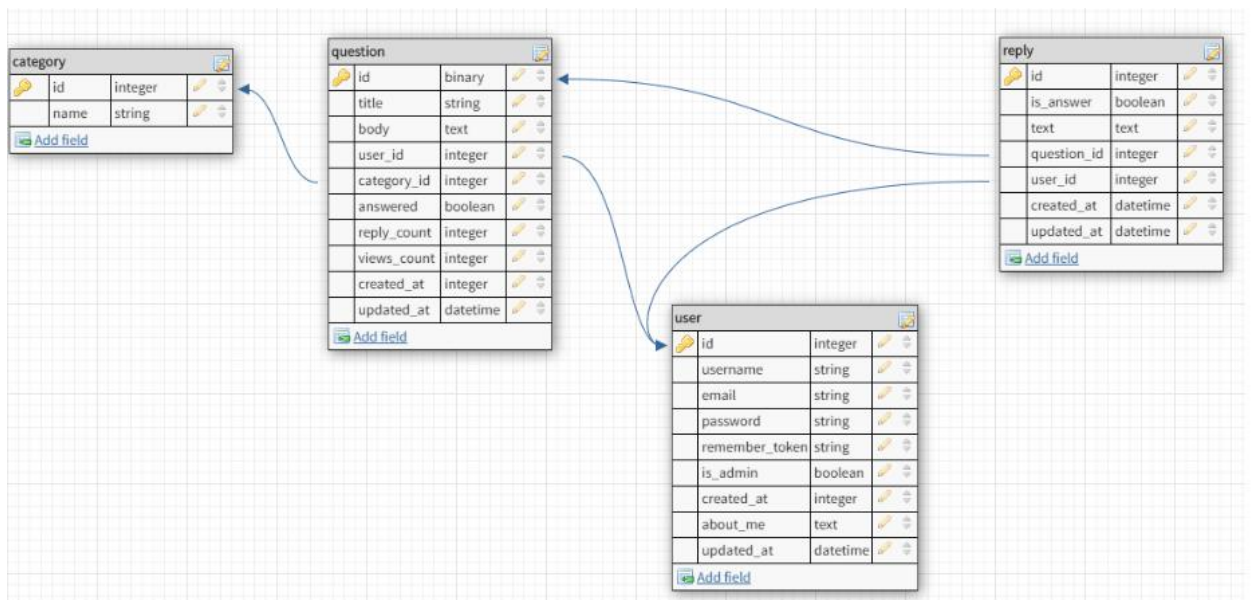


Рисунок 2.3.1 База даних сайту



## 2.4 Створення Фронт-енду

Одним з перших етапів розробки, після написання тз, було створення дизайну сайту, а саме його головної сторінки, сторінки питання та сторінки профілю у обох форматах (як мобільний так і десктопний). Дизайн було узгоджено усією командою. Створено дизайн було у графічному редактору Figma. Після цього було проведена робота по верстці макету за допомогою css, html і компонентів Material UI.

Через те що основною функцією сайту є створення та обговорення питань а також пошук інформації, було вирішено зробити мінімалістичний дизайн без яскравих кольорів. Для цього було обрано 3 основні кольори (білий, сіро-чорний, сіро-синій).

Розробники	Розділи	Використані технології	Розробка
Андрєєв Данило	Питання	React	Github репозиторій
Соломаха Олександр	Користувачі	Material UI	Github репозиторій front-end
Нестеров Микита	Створити питання	Laravel	Github репозиторій back-end
Copyright (C) Andriev Solomaha Nesterov 2020 All rights reserved			

Рисунок 2.4.1 Футер сайту для демонстрації кольорової палітри

Для зручної навігації по сайту зроблено меню зліва.

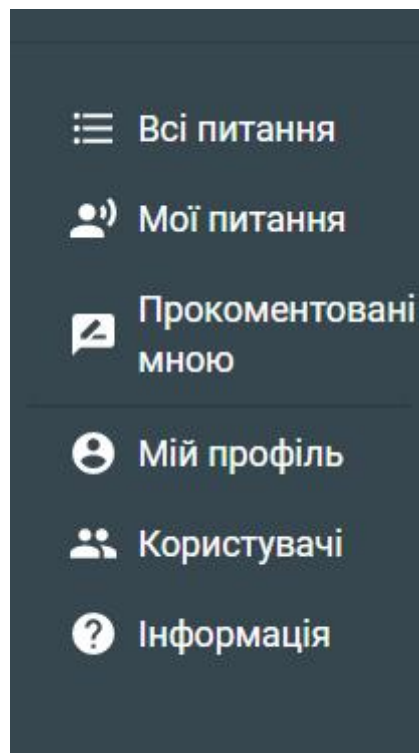
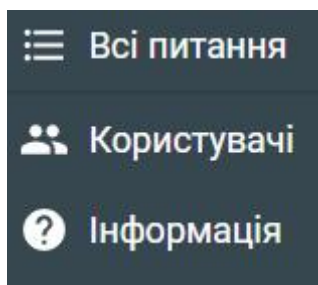


Рисунок 2.4.2 Меню сайту

Як видно з рисунку меню складається з шести пунктів:

- Всі питання – веде на головну сторінку, де можна побачити перелік питань, які були задані
- Мої питання – сторінка, що відображає питання створені користувачем
- Прокоментовані мною – сторінка, де знаходяться питання на які користувач залишив відповіді
- Мій профіль – сторінка, що містить інформацію о користувачеві
- Користувачі – перелік всіх зареєстрованих юзерів
- Інформація – коротко про проект

Слід зазначити, що не всі функції доступні неавторизованим користувачам, наприклад вони не можуть створювати та коментувати питання. Тому навігаційне меню відрізняється для авторизованих та неавторизованих юзерів. На рисунку віще було зображено повне меню (Для перегляду обмеженого меню дивитись рисунок 2.3.3).



*Рисунок 2.4.3 Меню для не авторизованих користувачів*

Для сайту був зроблений логотип за допомогою графічного редактору Adobe Photoshop і обрано назву «Readit» співзвучну з назвою вже популярного форуму «Reddit». Даний вибір зумовлен тим, що така назва повинна підвищити популярність сайту.



*Рисунок 2.4.4. Логотип сайту*

Логотип знаходиться як у хедері сайту так і на інформаційній сторінці. Також у хедері можна побачити кнопку «logout» ім'я користувача та аватар (якщо авторизований), або кнопки «Sign up» та «Login» (якщо користувач не увійшов до свого профілю)



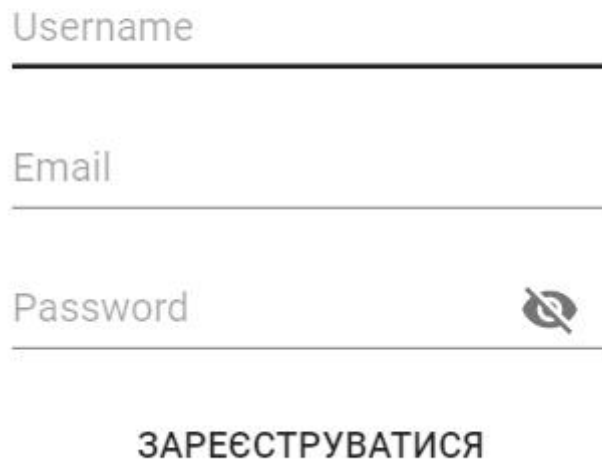
*Рисунок 2.4.5. Хедер для авторизованого користувача*



*Рисунок 2.4.6. Хедер для невідомого користувача*

При натисканні на кнопку «Sign up» або «Login» з'явиться вікно з полями для введення даних.


## Реєстрація



Registration form UI. It consists of three input fields: 'Username', 'Email', and 'Password'. The 'Password' field has a toggle icon (an eye with a diagonal line) to its right. Below the input fields is a button labeled 'ЗАРЕЄСТРУВАТИСЯ'.

Username

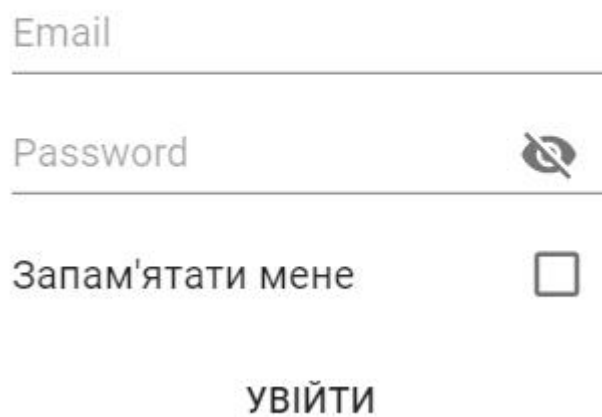
Email

Password 

ЗАРЕЄСТРУВАТИСЯ


*Рисунок 2.4.7 Вікно реєстрації*

## Аутентифікація



Authentication form UI. It consists of two input fields: 'Email' and 'Password'. The 'Password' field has a toggle icon (an eye with a diagonal line) to its right. Below the input fields is a checkbox labeled 'Запам'ятати мене'. At the bottom is a button labeled 'УВІЙТИ'.

Email

Password 

Запам'ятати мене ☐

УВІЙТИ

*Рисунок 2.4.8 Вікно авторизації*

Верстка була зроблена за допомогою правил css, таких як padding, margin, position, color, background, transition, border, flex тощо.

Ієрархія файлів

Тека web - це тека з фронт ендом

В ній можна знайти теки build, public, sources, src

В теці build зібраний проект, який відправляється на сервер

В теці public знаходяться стартові файли, іконка favicon, початковий index.html

В теці sources зібрані картинки високої якості

В теці src знаходяться програмні компоненти, наприклад index.js

Також в цій теці знаходяться теки Pages з сторінками, Themes з налаштуваннями теми, Utilities з деякими компонентами, які використовуються в багатьох місцях, наприклад провайдери, confirmDialog, тощо

В сторінках знаходяться теки з назвами сторінок, в яких початковими файлами є index.js. В файлах Style.js знаходяться JSS правила для оформлення.

## *2.5. Опис засобів виконання індивідуальної роботи*

Для виконання роботи було використано такі засоби:

1. ReactJS
2. Lodash
3. Clsx
4. React-router-dom
5. React-helmet
6. Superagent
7. Qs
8. MaterialUI

React дозволяє створювати додатки за парадигмою single page application, що означає, що весь сайт буде на одній веб сторінці, і в зв'язку з діями користувача буде перемальовуватись. Це забезпечує швидкодію при користуванні, але при першій загрузці сторінки процес буде довший ніж у класичному MVC. Для організації зв'язку з ядром додатку був використаний принцип restful api, що означає зв'язок з сервером під час дії сторінки за допомогою методів http put, get, post, delete.

Для створення інтерфейсу було використано бібліотеку MaterialUI, яка дозволяє дуже зручно динамічно змінювати стилі, за допомогою JSS, а також впроваджує зручну систему темінгу і компоненти.

За допомогою бібліотеки react-router-dom було впроваджено управління за допомогою url. Це дає такі переваги:

1. Користувач може поділитися, наприклад, якоюсь з сторінок додатку, наприклад, питанням, і отримувачу посилання не прийдеться шукати це питання серед купи інших, а лише перейти по посиланню на потрібний.
2. Також це надає можливість дуже зручно користуватись змахуванням вліво на на безграничному екрані абоо кнопкою назад для відміни операції чи переходу на попередню сторінку.

Весь макет було розбито на составні частини і об'єднано в компоненти. Наприклад, комплнент App містить компонент Layout, в якому зібрано загальне оформлення сторінки. Компонент Layout містить компоненти сторінок, наприклад, ThreadDetails, в якому, в свою чергу, є комопненти Question і AnswerListItem, і так далі.

Для зміни вмісту head блоку біло використано бібліотеку react-helmet, яка дозволила динамічно змінювати title сторінки в залежності від контенту, який продивляється користувач.

Було написано деякі компоненти, які використовувались в декількох місцях додатку, наприклад confirmDialog. Він реалізован за допомогою провайдеру, в якому є асинхронна функція, при викликанні якої встановлюється флаг, при якому відкривається дівлогове вікно з заданим текстом.

Також було написано кастомний компонент для загрузки аватару в профіль.

Для зручної розробки було створено декілька оточень, development і production, в яких було описані посилання на API і шлях для деплоїнгу коду.

Для менеджменту програмних пакетів було використано npm (node package manager), яких дозволяє зручно встановлювати пакети і управляти їми.

Для створення початкової конфігурації додатку було використано утиліту create-react-app.

Також була проведена робота по інтеграції додатку в різні браузери і адаптації сторінки під різні пристрої. Так, наприклад, в мобільній версії меню перенесено у окрему виїжджаючу панель для компактності і зручності, також туди було перенесено розділ тоа 10 питань.



## *2.6. Опис результатів загальної веб системи*

Веб система являє собою клієнт-серверний додаток, в якому клієнтську частину реалізує браузер, який здійснює діалог з користувачем і відображення інформації, а серверну частину - веб-сервер і сервер додатків, які реалізують основну логіку системи.

У розробленій веб системі, реалізовано усі функції які вже були описані вище у документі (див. розділ 2.3 Функціонал сайту). У системі є свого роду ієрархія користувачів, яка полягає у тому, що кожен користувач залежно від його статусу на форумі може або навпаки не має можливості виконувати ті чи інші дії. Таким чином є три групи користувачів: гості, авторизовані користувачі (надалі юзери) та адміністратори. Якщо говорити про цю ієрархію як про наслідування класів в об'єктно орієнтованому програмуванні, то тоді можна уявити гостя (неопізаного користувача) як батьківській клас, тоді юзер буде похідним класом від гостя, а адміністратор в свою чергу – похідним від юзера.

У гостя на нашому сайті є можливості переглядати публічний контент (питання, створені юзерами, та відповіді на них; перелік усіх юзерів сайту; інформація у профілі; інформація про сайт) та зареєструватися або авторизуватися (увійти у вже створений акаунт). У юзера додаються до цих можливостей ще такі, як створення питань, залишання коментарів (відповідей) на питання інших або власні, редагування свого профілю. Що стосується адміну, то він може змінювати будь-яку інформацію створену юзерами або видаляти її, також має можливість видалити акаунт іншого користувача.

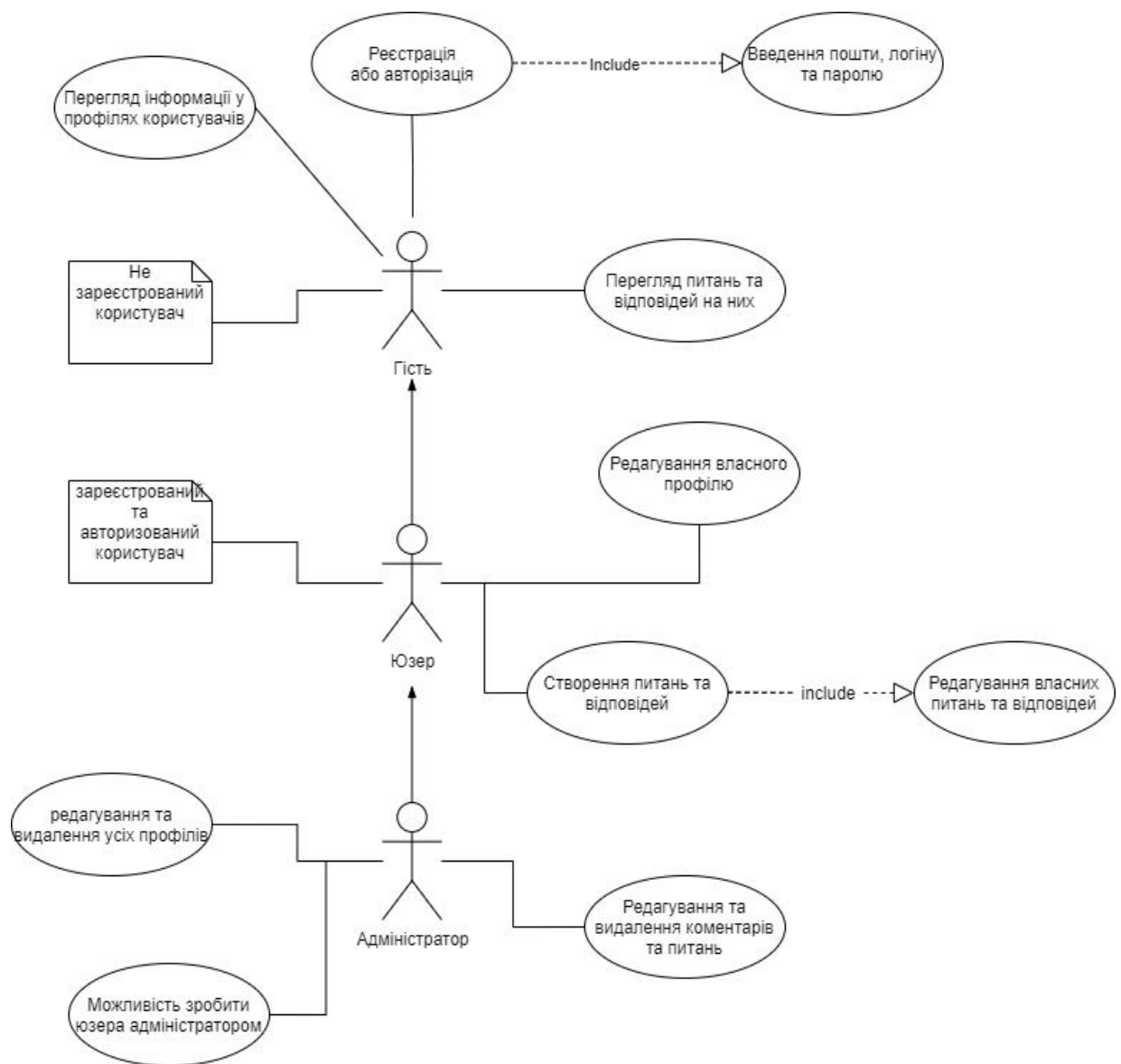


Рисунок 2.5.1 Ієрархія користувачів

## **ВИСНОВОК**

В результаті проведеної роботи було закріплено матеріал, вивчений в курсі дисципліни «Основи web-програмування».

Мета курсової роботи створення повноцінного веб-сайту, в моєму випадку сайту-форуму. Розроблений сайт задовольняє всім вимогам, поставленим на етапі створення технічного завдання. Як і планувалося на етапі постановки завдання, сайт містить всі необхідні структурні і навігаційні елементи: форму пошуку, навігаційне меню і так далі.

Було проведено тестування отриманого продукту та результатів його роботи. Форум є повністю готовим програмним продуктом і може бути використаний не тільки для навчальної діяльності а й для комерційної.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. «PHP, MySQL и DreamweaverMX 2004. Разработка интерактивных Web-сайтов.» Дронов В. А. — СПб.: БХВ-Петербург, 2005. — 448 с :ил.
2. «PHP and MySQL Web Development (4th Edition)», Luke Welling, Laura Thomson 848 стр., сил.; ISBN 978-5-8459-1574-0, 978-0-672-32916-6.
3. Мардан Азат. React быстро. Веб-приложения на React, JSX, Redux и GraphQL. — СПб.: «Питер», 2019. — С. 560. — ISBN 978-5-4461-0952-4
4. Shawn McCool. Laravel Starter. — Packt Publishing, 2012. — 64 p. — ISBN 978-1-78216-091-5.
5. Зандстра М. PHP. Объекты, шаблоны и методики программирования. — 5-е изд.. — СПб.: «Диалектика», 2019. — С. 736. — ISBN 978-5-907144-54-5.
6. Кузнецов М., Симдянов И. Самоучитель PHP 7. — 2-е изд.. — СПб., 2018. — С. 448. — ISBN 978-5-9775-3817-6.

## **ДОДАТОК 1. КОД ПРОГРАМИ**

## ThreadsViewer/index.js

```
import React from 'react';
import useStyles from './style';
import {useHistory, useLocation, useParams} from 'react-
router-dom'
import {coreRequest} from "../../Utilities/Rest";
import qs from 'qs';
import {useAuth} from "../../Utilities/Auth";
import {Helmet} from "react-helmet";

//MUI components
import Box from "@material-ui/core/Box";
import Grid from "@material-ui/core/Grid";
import List from "@material-ui/core/List";
import ListItem from '@material-ui/core/ListItem';
import ListItemText from '@material-ui/core/ListItemText';
import ListItemAvatar from '@material-
ui/core/ListItemAvatar';
import Avatar from '@material-ui/core/Avatar';
import Typography from "@material-ui/core/Typography";
import Divider from "@material-ui/core/Divider";
import Badge from "@material-ui/core/Badge";
import ListItemSecondaryAction from "@material-
ui/core/ListItemSecondaryAction";
import Pagination from '@material-ui/lab/Pagination';
import ToggleButton from '@material-ui/lab/ToggleButton';
import ToggleButtonGroup from '@material-
ui/lab/ToggleButtonGroup';
import Breadcrumbs from "@material-ui/core/Breadcrumbs";
import Link from "@material-ui/core/Link";

//MUI icons
import VisibilityIcon from '@material-ui/icons/Visibility';
import QuestionAnswerIcon from '@material-
ui/icons/QuestionAnswer';

function ThreadListItem({thread, ...props}) {
  const classes = useStyles();
  const history = useHistory();

  function changeRoute(route) {
    history.push(route);
  }

  const primary = (
    <Typography>
      {thread.title}
    </Typography>
  );
  const secondary = (
    <ListItemSecondaryAction>
```

```

        <Badge badgeContent={thread.reply_count}
showZero color="primary">
            <QuestionAnswerIcon
className={classes.ratingBadge}/>
        </Badge>
        <Badge badgeContent={thread.views_count}
showZero color="primary">
            <VisibilityIcon
className={classes.ratingBadge}/>
        </Badge>
    </ListItemSecondaryAction>
  );

  return (
    <>
      <ListItem button onClick={event =>
changeRoute(`/thread/${thread.id}`)}>
        <ListItemAvatar>
          <Avatar
src={`${process.env.REACT_APP_CORE_AVATARS}/${thread.user.id}.jp
g`}
          >
        </Avatar>
      </ListItemAvatar>
      <ListItemText
        primary={primary}
        secondary={`${thread.user.name} | ${new
Date(thread.created_at).toLocaleString()}`}
        className={classes.threadsList}
      >
        {secondary}
      </ListItemText>
      <Divider/>
    </>
  );
}

export default function ThreadsViewer({articles,
setArticles, ...props}) {
  const classes = useStyles();
  const history = useHistory();
  const [sortBy, setSortBy] = React.useState('created_at');
  const [pages, setPages] = React.useState(1);
  const [page, setPage] = React.useState(1);
  const location = useLocation();
  const {mode} = useParams();
  const {search} = qs.parse(location.search,
{ignoreQueryPrefix: true});
  const {user} = useAuth();
  const updater = React.useRef();

```

```

function getArticles(page) {
  if (!search) {
    coreRequest().get(`questions${(mode && user) ?
`/${mode}` : ''}`)
      .query({page: page, sort: sortBy})
      .then(response => {
        setArticles(response.body.data);
        setPages(response.body.meta.last_page);
      })
      .catch(err => {
        console.error(err);
      });
  } else {
    coreRequest().get('questions')
      .query({search, page, sort: sortBy})
      .then(response => {
        setArticles(response.body.data);
      }).catch(error => {
        console.error(error);
      });
  }
}

React.useEffect(() => {
  getArticles(1);
  clearInterval(updater.current);
  updater.current = setInterval(() => {
    if (!search) {
      console.log(`Sync [threads]: synchronizing
(${new Date().toLocaleString()})`);
      getArticles();
    } else {
      console.log(`Sync [threads]: skipping
synchronization (${new Date().toLocaleString()})`);
    }
  }, 30000);

  return () => {
    clearInterval(updater.current);
  }
}, [mode, sortBy]);

React.useEffect(() => {
  getArticles(1);
}, [search, sortBy]);

function changeRoute(route) {
  history.push(route);
}

function handleToggleSort(event, newValue) {
  setSortBy(newValue);
}

```



```

function handleChangePage(event, page) {
    setPage(page);
    getArticles(page);
}

return (
    <Grid item xs={12}>
        <Helmet>
            <title>Readit | Threads</title>
        </Helmet>
        <Box p={1}>
            <List>
                <ListItem>
                    <Breadcrumbs aria-
label="breadcrumb">
                        {!search && <Typography
color="textPrimary">Питання</Typography>
                        {search && <Link onClick={event
=> changeRoute('/threads')} color="textPrimary">Питання</Link>
                        {search && <Typography
color="textPrimary">Результати для: {search}</Typography>}
                    </Breadcrumbs>
                </ListItem>
                <ListItem>
                    <ToggleButtonGroup
                        value={sortBy}
                        size="small"
                        exclusive
                        onChange={handleToggleSort}
                    >
                        <ToggleButton
value="created_at">
                            Найновіші
                        </ToggleButton>
                        <ToggleButton
value="reply_count">
                            По відповідям
                        </ToggleButton>
                        <ToggleButton
value="views_count">
                            Популярні
                        </ToggleButton>
                    </ToggleButtonGroup>
                </ListItem>
                {articles.map((item, index) => {
                    return (
                        <ThreadListItem thread={item}
key={`articles_${index}`}>
                            </ThreadListItem>
                        </ThreadListItem>
                    );
                })}
            </List>

```

```

        <div
className={classes.paginationContainer}>
            <Pagination count={pages} page={page}
onChange={handleChangePage} className={classes.pagination}/>
        </div>
    </Box>
</Grid>

    );
}

```

### **UserPolicy.php**

```

namespace App\Policies;
use App\User;
use Illuminate\Auth\Access\HandlesAuthorization;

class UserPolicy
{
    use HandlesAuthorization;

    public function before($user, $ability)
    {
        if ($user->isAdmin()) {
            return true;
        }

        if ($user->is_deleted) {
            return false;
        }
    }
}

```

```

public function viewAny(?User $user)
{
    return true;
}
public function view(?User $user, ?User $model)
{
    return !$model->is_deleted;
}
public function create(User $user)
{
    return false;
}
public function update(User $user, User $model)
{
    return $user->id === $model->id;
}
public function delete(User $user, User $model)
{
    return $user->id === $model->id;
}
}

```

## **QuestionController.php**

```

<?php
namespace App\Http\Controllers;
use App\Http\Resources\Question as QuestionResource;
use App\Http\Resources\ViewQuestion;
use App\Question;
use Illuminate\Http\Request;
use Illuminate\Database\Eloquent\Builder;

class QuestionController extends Controller

```

```

{
    public function __construct()
    {
        $this->middleware('auth')->except('index', 'show',
'top10');
        $this->authorizeResource(Question::class, 'question');
    }
    public function index(Request $request)
    {
        $search = $request->get('search');
        $sort = $request->get('sort');
        return QuestionResource::collection(
            Question::when($search, function ($query, $search) {
                return $query->where('title', 'like', '%' .
$search . '%');
            })
            ->when($sort, function ($query, $sort) {
                return $query->orderBy($sort, 'desc');
            }, function ($query) {
                return $query->orderBy('created_at', 'desc');
            })
            ->paginate()
        );
    }
    public function my(Request $request)
    {
        $search = $request->get('search');
        $sort = $request->get('sort');

        return QuestionResource::collection(
            Question::when($search, function ($query, $search) {
                return $query->where('title', 'like', '%' .
$search . '%');
            })
            ->when($sort, function ($query, $sort) {
                return $query->orderBy($sort, 'desc');
            })
        );
    }
}

```

```

        }, function ($query) {
            return $query->orderBy('created_at', 'desc');
        })
        ->where('user_id', $request->user()->id)
        ->paginate()

    );
}

public function commented(Request $request)
{
    $search = $request->get('search');
    $sort = $request->get('sort');

    return QuestionResource::collection(
        Question::when($search, function ($query, $search) {
            return $query->where('title', 'like', '%' .
$search . '%');
        })
        ->when($sort, function ($query, $sort) {
            return $query->orderBy($sort, 'desc');
        }, function ($query) {
            return $query->orderBy('created_at', 'desc');
        })
        ->whereHas('replies', function (Builder $query)
use ($request) {
            $query->where('user_id', $request->user()-
>id);

        })
        ->paginate()

    );
}

```

[https://github.com/DanilAndreev/starscream\\_project](https://github.com/DanilAndreev/starscream_project)