

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування інституту, назва факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНА РЕАЛІЗАЦІЯ ВЕБСЕРВІСУ ДОШКИ ОГолоШЕНЬ
SOFTWARE IMPLEMENTATION OF THE BULLETIN BOARD WEB
SERVICE

Виконав: студент 4 курсу, групи КНТ-118
Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

Клоков А.О.

(прізвище та ініціали)

Керівник Гладкова О.М.

(прізвище та ініціали)

Рецензент Зеленьова І.Я.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
 (повне найменування закладу вищої освіти)

Інститут, факультет ФКНТ
 Кафедра програмних засобів
 Ступінь вищої освіти бакалавр
 Спеціальність 121 Інженерія програмного забезпечення
 (код і найменування)

Освітня програма (спеціалізація) Інженерія програмного забезпечення
 (назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
 “ ” 20 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Клокова Артема Олександровича
 (прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Програмна реалізація вебсервісу дошки оголошень.
Software implementation of the Bulletin Board Web Service.
- керівник проєкту (роботи) Гладкова Ольга Миколаївна, к.т.н.,
 (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
- затверджені наказом закладу вищої освіти від “27” квітня 2022 року № 102
2. Строк подання студентом проєкту (роботи) червень 2022 року
3. Вихідні дані до проєкту (роботи) рекомендована література, технічне
завдання
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Розробка архітектури програми.
3. Реалізація компонентів системи. 4. Керівництво користувача.
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) _____
Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	Гладкова О.М., доцент		
Нормоконтроль	Дейнега Л.Ю., ст.викладач		

7. Дата видачі завдання “04” квітня 2022 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області	2-3 тижні	Розділ 1
3	Розробка архітектури програми	4 тиждень	Розділ 2
4	Розробка програми	5-6 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми	7 тиждень	Розділ 4
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	8 тиждень	Додатки
7	Захист роботи	9 тиждень	

Студент(ка)

(підпис) Клоков А.О.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис) Гладкова О.М.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра:
75с., 32 рис., 1 табл., 2 дод., 22 джерел.

ДОШКА ОГОЛОШЕНЬ, ОГОЛОШЕННЯ, WEB, JAVASCRIPT, VUE.JS,
EXPRESS.JS.

Об'єкт дослідження – розміщення та розповсюдження інформації в мережі Інтернет.

Предмет дослідження – вебзастосунок дошки оголошень.

Мета роботи полягає в розробці вебзастосунку дошки оголошень, що буде мати необхідний набір інструментів та засобів комунікації з користувачем для зручного користування сервісом.

Матеріали, методи та технічні засоби: структурне, функціональне та об'єктно-орієнтоване програмування, мова програмування JavaScript, фреймворки Vue.js та Express.js, персональний комп'ютер з чіп-сетом Ryzen 5 4600p під управлінням операційної системи Microsoft Windows 11.

Результати. Розроблено програмне забезпечення у вигляді вебзастосунку дошки оголошень, для перегляду та публікації оголошень комерційного характеру, з реалізацією бази даних збереження даних створених оголошень.

Висновки. Розроблене програмне забезпечення забезпечує користувачів можливістю реєстрації, авторизації, перегляду списків оголошень та їх фільтрацією за містом і категорією, публікацією власних оголошень з подальшою можливістю обмеженого редагування та видалення.

Галузь використання – такий застосунок використовується приватними особами та компаніями для споживання або реалізації товарів та послуг, проведення рекламних компаній.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 75 pages, 32 figures, 1 tables, 2 appendixes, 22 sources.

BULLITEN BOARD, BULLITENS, WEB, JAVASCRIPT, VUE.JS, EXPRESS.JS.

The object of the study is a posting and disseminating information on the Internet.

The subject of research is the development of a bulletin board web application.

The purpose of the work is to develop a web application bulletin board that will have the necessary set of tools and means of communication with the user for easy use of the service.

Materials, methods and technical means: structural, functional and object-oriented programming, JavaScript programming language, Vue.js and Express.js frameworks, personal computer with Ryzen 5 4600h chipset running Microsoft Windows 11 operating system.

Results. Software has been developed in the form of a bulletin board web application for viewing and publishing commercial ads, with the implementation of a database to save the data of created ads.

Conclusions. The developed software provides users with the ability to register, authorize, view ad lists and filter them by city and category, publish their own ads with the possibility of limited editing and deletion.

Field of use - this application is used by individuals and companies for the consumption or sale of goods and services, advertising campaigns.

ЗМІСТ

	С.
ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК.....	8
ВСТУП.....	9
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ.....	11
1.1 Огляд існуючих вебзастосунків	11
1.1.1 OLX	11
1.1.2 Work.ua.....	15
1.1.3 AUTO.RIA.....	18
1.1.4 Booking.com	22
1.1.5 Trello.....	24
1.2 Висновки за розділом.....	25
2 РОЗРОБКА АРХІТЕКТУРИ ВЕБЗАСТОСУНКУ	27
2.1 Функціональні вимоги до вебзастосунку.....	27
2.2 Обґрунтування засобів розробки.....	28
2.3 Вибір фреймворку	29
2.4 Робота з даними.....	31
2.5 Висновки за розділом.....	32
3 РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ.....	33
3.1 Визначення функціональної поведінки вебзастосунку.....	33
3.2 Робота з даними.....	34
3.3 Структура застосунку	37
3.3.1 Структура клієнтської частини.....	37
3.3.2 Структура серверної частини.....	39
3.4 Розробка інтерфейсу користувача.....	40
3.5 Висновки за розділом.....	41
4 КЕРІВНИЦТВО КОРИСТУВАЧА	43
4.1 Призначення вебзастосунку	43

	7
4.2 Умови використання вебзастосунку	43
4.3 Опис процесу використання вебзастосунку	43
4.4 Висновки за розділом.....	50
ВИСНОВКИ	51
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ	52
ДОДАТОК А Текст програми	54
ДОДАТОК Б Слайди презентації.....	70

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

API	– Application Programming Interface;
BE	– Back-end, серверна частина застосунку;
FE	– Front-end, клієнтська частина застосунку;
ORM	– Object-Relational Mapping;
SPA	– Single Page Application;
БД	– база даних;
ПЗ	– програмне забезпечення;
СКБД	– система керування базами даних.

ВСТУП

Оголошення – змістовні повідомлення в стислому форматі, які містять у собі різноманітну інформацію. Зазвичай, оголошення напрямлені на комерційну діяльність (продаж або купівля товарів різного напрямлення, оренда тимчасового житла, афіші культурних та освітніх заходів, робочі пропозиції). У меншій кількості зустрічаються інформаційні оголошення, які висвітлюють суспільно значущі теми, або виступають у роль нагадувань чи застережень.

Ще не так давно, найпопулярнішим середовищем для поширення оголошень були газети. Різні друковані видавництва були зацікавлені в тому, щоб залишати сторінку, або цілий розділ для публікації повідомлень. Саме так поширювалася реклама, або оголошення купівлі-продажу. Але з плином часу, популярність друкованих ЗМІ має спадну динаміку, і втрачає аудиторію.

Ще одним архаїчним засобом, створеним для концентрування оголошень були дошки на зупинках автотранспорту та біля житлових будинків. Але таке рішення, спрямоване на розміщення оголошень має низку недоліків. Насамперед, це локальність таких засобів, та засмічення дошок повідомленнями, що втратили актуальність.

Сьогодні, інтеграція оголошень у життя сучасної людини можлива тільки шляхом використання технологій веброзробки, що дозволяє охопити найбільшу аудиторію та запровадити всі необхідні інструменти для виправданого й максимально раціонального функціонування принципів дошки оголошень.

Мета роботи полягає в розробці вебзастосунку дошки оголошень, що буде мати необхідний набір інструментів та засобів комунікації з користувачем для зручного користування сервісом.

Завдання роботи полягає в розв'язання таких завдань:

- аналіз існуючих рішень вебзастосунків;
- встановлення вимог до вебзастосунку;

- проектування архітектури власного застосунку;
- реалізація вебзастосунку дошки оголошень.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Онлайн дошки оголошень, зазвичай, користуються схожою логікою та допомагають вирішувати подібні завдання користувача. Спільним для таких сервісів є ролі діяльності користувачів, адже існує всього 2 сторони. Одна зі сторін створює пропозицію, інша – забезпечує попит. Також не відрізняється структура, яка демонструє користувачеві пропозиції у стислому форматі, але користувач може роздивитися кожну пропозицію більш детально і зв'язатися з суб'єктом, який створив певну пропозицію.

1.1 Огляд існуючих вебзастосунків

Майже кожен користувач в інтернеті зустрічався з вебзастосунками, що допомагають створювати або продивлятися оголошення різного характеру. Більшість таких застосунків мають спільне призначення, але працюють у відмінних галузях чи напрямленнях. Можна виділити декілька найпопулярніших серед основних напрямлень використання подібних сервісів.

1.1.1 OLX

OLX – платформа для розміщення оголошень. Засновали компанію французькі підприємці Фабріс Гранд і Олександр Оксенфорт ще у 2006 році. Діяльність компанія розпочала в Індії, достатньо перспективному ринку того часу. Протягом десяти років сервіс виходив на ринки інших країн по всьому світі. На території України, OLX отримав популярність після злиття зі вже розвинутою локальною онлайн дошкою оголошень Slando, що відбулося в 2015 році [1].

Поданий вебзастосунок є платформою для онлайн купівлі або продажу товарів та послуг приватними особами та компаніями.

Користувач може переглядати пропозиції на головній сторінці сайту навіть будучи не авторизованим рис. 1.1 [1].

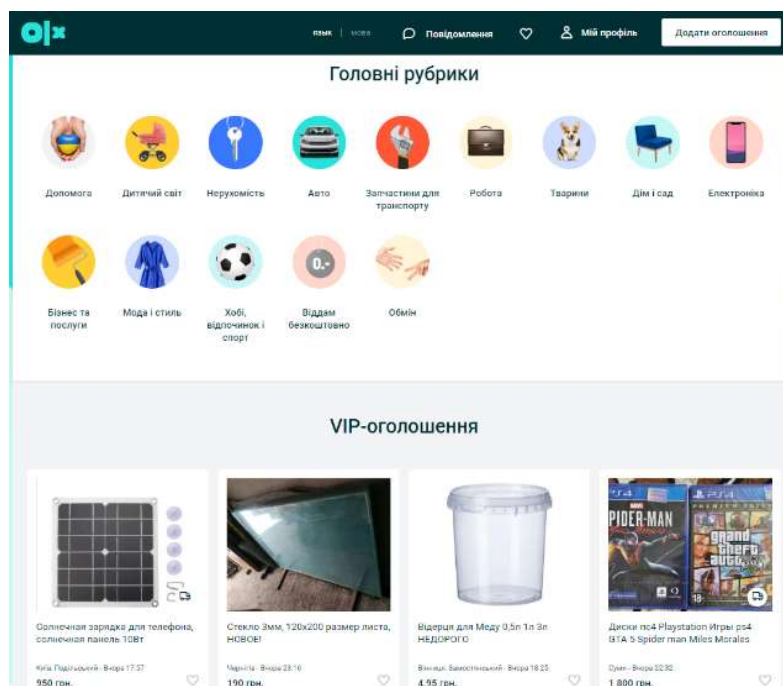


Рисунок 1.1 – Головна сторінка OLX

Також, не авторизованому користувачу доступні інструменти пошуку, фільтрації та сортування товарів на сторінці.

Щодо фільтрів, користувач може обрати оголошення за регіоном, за порядком цін, а також за певною рубрикою. рис. 1.2 [1].

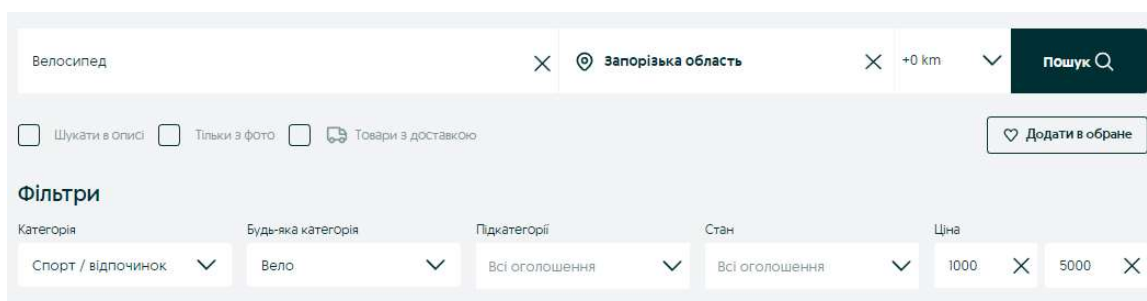


Рисунок 1.2 – Фільтри OLX

У списку, за замовчуванням, сайти сортуються від найновіших. Але, саме для продавців існують деякі платні можливості, для просування оголошень у топ.

Сайт надає можливість й створювати оголошення авторизованим користувачам або компаніям. У формі подачі оголошення рис. 1.3 [1] потрібно заповнити обов'язкові поля назви товару або послуги, опису та контактних даних продавця, обрати рубрику товару чи послуги. Серед необов'язкових даних, є можливість додати одну або декілька фотографій, обрати їх черговість.

Створити оголошення

Опишіть у подробицях

Вкажіть назву*

Наприклад, iPhone 8

Заголовок відіграє важливу роль, не забудьте додати його

Категорія*

Вибірть категорію

Фото

Перше фото буде на обкладинці оголошення. Перегляньте, щоб замінити зображення фото

Додати фото

Опис*

Подумайте, що ви хотіли би дізнатися з оголошення. І додайте це в опис

Напишіть ще 80 символів

0/9000

Ваші контактні дані

Місцезнаходження*

Запоріжжя, Шевченківський

E-mail-адреса

nsidomon.nsss@gmail.com

Номер телефону

0967253067

Рисунок 1.3 – Сторінка створення оголошень OLX

Користувач, що подав оголошення має можливість до перегляду, редагування або видалення рис. 1.4 [1]. Також, передбачена можливість відповідати на повідомлення чи коментарі, що залишають інші користувачі стосовно товару або послуги рис. 1.5 [1].

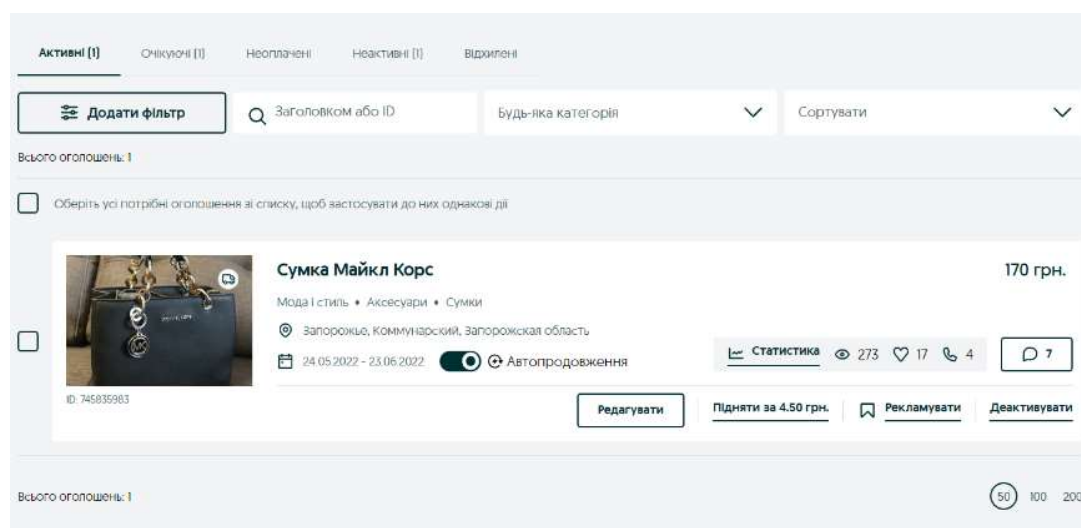


Рисунок 1.4 – Сторінка зі списком оголошень користувача OLX

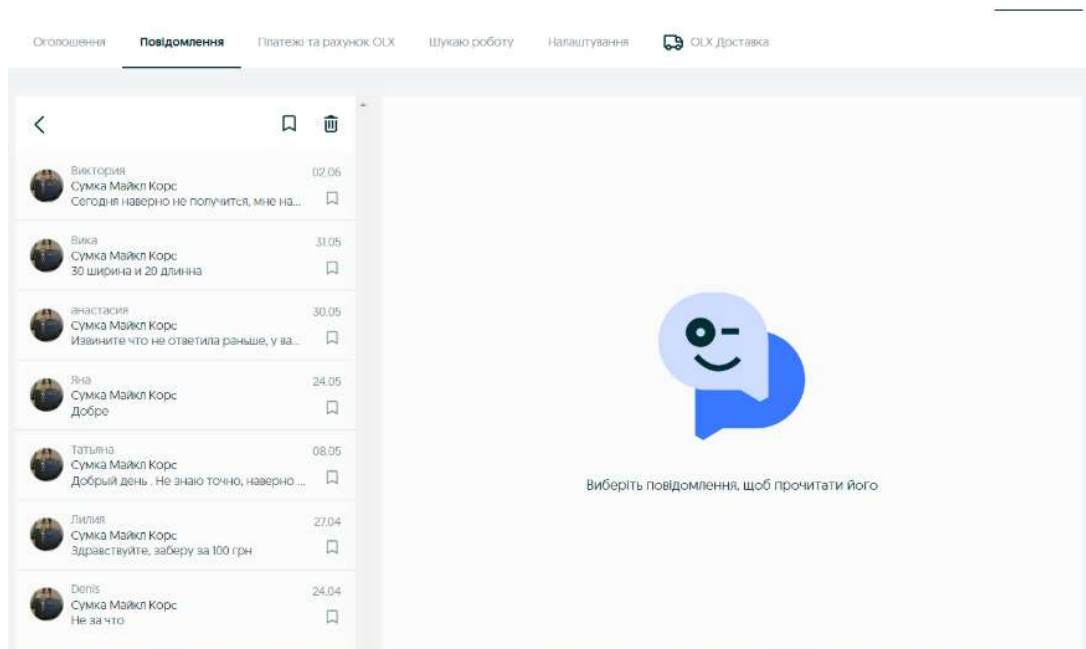


Рисунок 1.5 – Сторінка повідомлень та чатів OLX

1.1.2 Work.ua

Work.ua – це сервіс з пошуку роботи або працівників. Створено його в 2006 році дніпровською компанією «Реактор». Наразі платформа працює лише на території України, і є флагманом серед подібних [2].

Функціональні можливості сайту Work.ua поділяють користувачів на 2 ролі, що обирає власне користувач на головній сторінці, у правому верхньому куту рис. 1.6 [2].

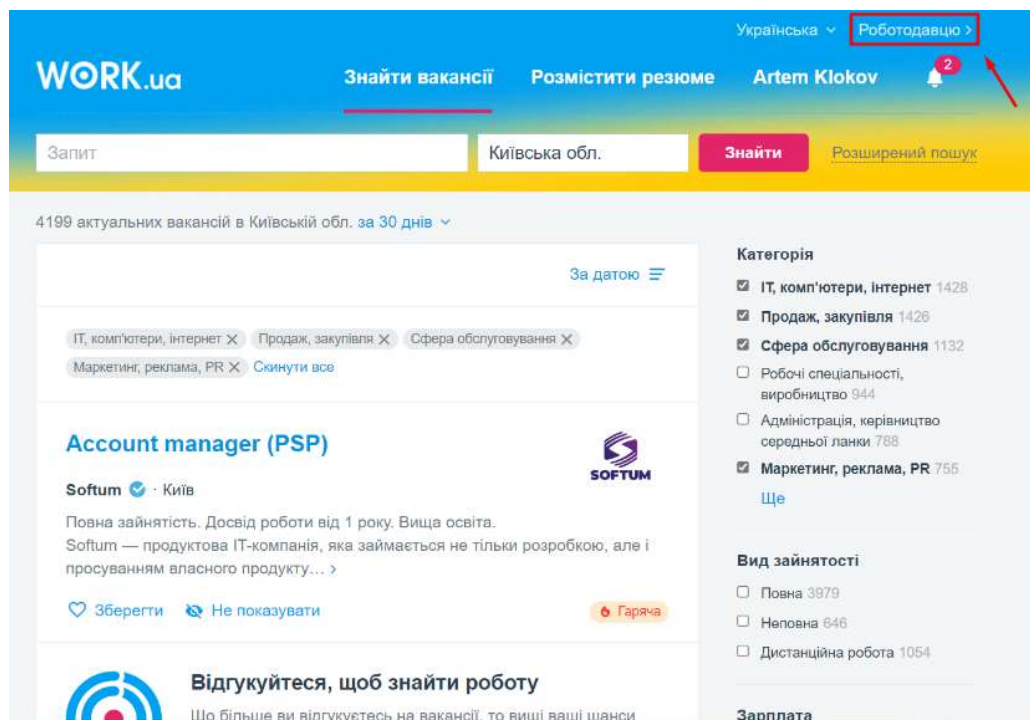
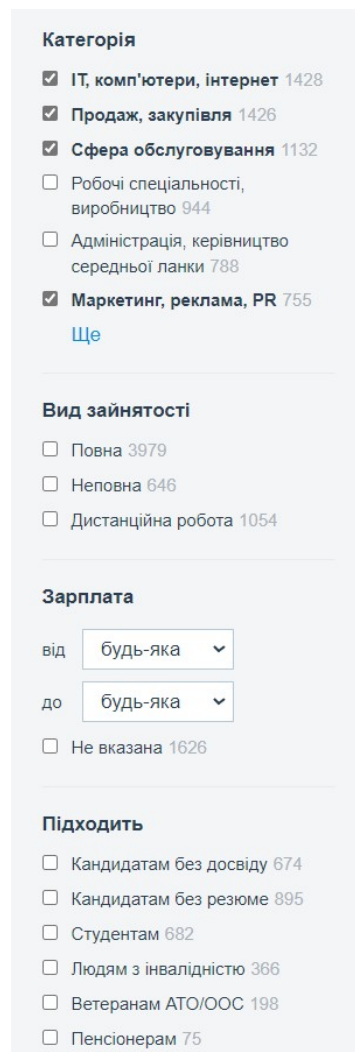


Рисунок 1.6 – Головна сторінка Work.ua

Користувач в ролі шукача має на меті знайти вакансії, що задовольняють умови. Це можливо здійснити двома способами:

- моніторинг відкритих вакансій (активний пошук);
- подача резюме на сайті (пасивний пошук).

Щодо моніторингу вакансій, для користувача доступні списки відкритих вакансій, фільтри та пошук. Пошук здійснюється як за назвою посади або галузі, так і за ключовими словами, що з нею зв'язані. Фільтрувати вакансії можливо за категоріями, посадами, містами, та окремими вимогами роботодавця рис. 1.7 [2].



Категорія

- ☒ ІТ, комп'ютери, інтернет 1428
- ☒ Продаж, закупівля 1426
- ☒ Сфера обслуговування 1132
- ☐ Робочі спеціальності, виробництво 944
- ☐ Адміністрація, керівництво середньої ланки 788
- ☒ Маркетинг, реклама, PR 755

[Ще](#)

Вид зайнятості

- ☐ Повна 3979
- ☐ Неповна 646
- ☐ Дистанційна робота 1054

Зарплата

від

до

- ☐ Не вказана 1626

Підходить

- ☐ Кандидатам без досвіду 674
- ☐ Кандидатам без резюме 895
- ☐ Студентам 682
- ☐ Людям з інвалідністю 366
- ☐ Ветеранам АТО/ООС 198
- ☐ Пенсіонерам 75

Рисунок 1.7 – Фільтри пошуку Work.ua

У списках вакансії за замовченням сортуються згідно дати її відкриття (від найновіших), але у користувача існує можливість відсортувати доступні вакансії й за рівнем зарплатні рис. 1.8 [2].

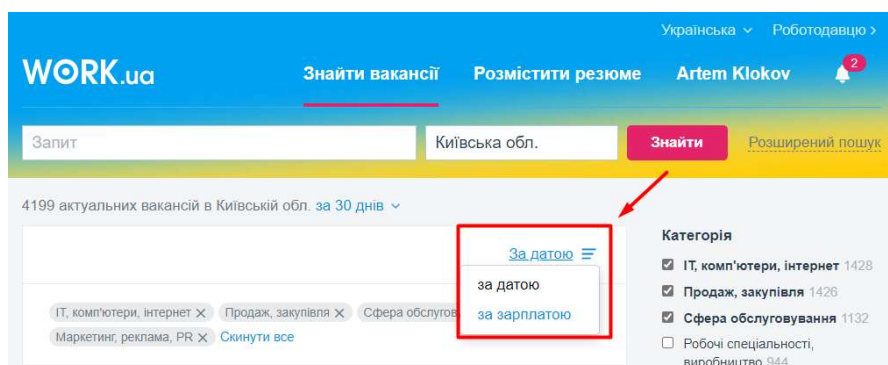


Рисунок 1.8 – Варіанти сортування Work.ua

Сценарій пасивного пошуку шляхом подачі резюме шукачем відбувається у декілька етапів. Перший з них – заповнити візитку рис. 1.9 [2] необхідним мінімумом інформації. Після чого, за бажанням, можна додати певну інформацію про себе, як працівника. Надалі, розміщуючи резюме на сайті, користувач дає згоду на поширення цієї інформації серед роботодавців, і можливість зв'язатися із собою ейчарам зацікавлених компаній.

 The image shows a 'Візитка' (Profile) creation form on the Work.ua website. At the top, there's a progress bar with four steps, the first of which is active. The form has the following fields: 'Ваше ім'я та прізвище*' (Artem Klovov), 'Посада, на якій ви хочете працювати*' (Javascript-розробник), 'Місто пошуку роботи*' (Запоріжжя), and 'Дата народження:' (10 травня 2001). At the bottom, there's a blue button labeled 'Далі >'.

Рисунок 1.9 – Сторінка створення резюме Work.ua

Роль роботодавця теж передбачає декілька сценаріїв. Серед них:

- розміщення запиту, для пошуку робітників на певну вакансію;
- моніторинг резюме, що залишають на сайті шукачі.

Обидва сценарії мають на меті знайти потенційного працівника, що буде відповідати вимогам.

Моніторинг резюме, здійснюється завдяки інструментам пошуку за галуззю або посадою рис. 1.10 [2].

Кандидати за категоріями	
• IT, комп'ютери, інтернет 190914 • Адміністрація, керівництво середньої ланки 447873 • Будівництво, архітектура 203913 • Бухгалтерія, аудит 235381 • Готельно-ресторанний бізнес, туризм 266145 • Дизайн, творчість 111912 • ЗМІ, видавництво, поліграфія 162066 • Краса, фітнес, спорт 90926 • Культура, музика, шоу-бізнес 69407 • Логістика, склад, ЗЕД 271907 • Маркетинг, реклама, PR 169838 • Медицина, фармацевтика 151216 • Нерухомість 9868 • Освіта, наука 191352 • Охорона, безпека 97059	• Продаж, закупівля 404423 • Робочі спеціальності, виробництво 379255 • Роздрібна торгівля 458244 • Секретаріат, діловодство, АГВ 420714 • Сільське господарство, агробізнес 28125 • Страхування 5277 • Сфера обслуговування 435702 • Телекомунікації та зв'язок 126212 • Топменеджмент, керівництво вищої ланки 71711 • Транспорт, автобізнес 290386 • Управління персоналом, HR 71561 • Фінанси, банк 158846 • Юриспруденція 74361 • Інші сфери діяльності 34456

Рисунок 1.10 – Список категорій пошуку резюме Work.ua

Сценарій розміщення запиту потребує окремої реєстрації своєї фірми на порталі. Після якої, роботодавець може виставити оголошення з необхідними вимогами до шукачів, та декларуванням умов їх потенційної праці.

1.1.3 AUTO.RIA

AUTO.RIA — український бренд автомобільної інтернет-торгівлі, один з проєктів RIA.com. З 2014 року сайт працює як майданчик для розміщення оголошень як від приватних осіб, так і від компаній, націлених на продаж нових

та вживаних автомобілів. Офіси в Україні знаходяться у Вінниці та Києві. Сайт функціонує як віртуальний торговельний майданчик для автомобілістів, де у відкритому доступі розміщуються оголошення про продаж автомобілів та автомобільних товарів [3].

Сервіс надає можливість користувачу переглядати транспорт на головній сторінці будучи не авторизованим. Для користувачів ,на головній сторінці, доступні фільтри для пошуку рис. 1.11 [3].

Рисунок 1.11 – Фільтри пошуку AUTO.RIA

Для більш точного пошуку доступна розширена база фільтрації, що включає у себе наступні критерії:

- стан транспорту (вживані, нові, під пригон, після ДТП та інше);
- вид транспорту (легкові, мото, вантажівки, спецтехніка тощо) ;
- тип кузова (універсал, седан, хетчбек тощо);
- країна виробник;
- марка авто (BMW, Audi та інші);
- модель авто;
- рік випуску;
- вартість;
- регіон, в якому автомобіль розташовано;

- технічні характеристики (паливо, КПП, тип приводу, колір тощо);
- наявність додаткових опцій.

Список оголошень на сайті сортується від найновіших, за виключенням оголошень, які просуваються за додаткову плату.

Після авторизації користувач має можливість створювати онлайн оголошення для купівлі транспорту та автомобільних товарів.

У формі подачі оголошення рис. 1.13 [3] існують обов’язкові поля, де необхідно вказати основні відомості про автомобіль, та додаткові поля, що дають можливість додати фото, VIN-номер та іншу інформацію. Для публікації оголошення необхідно заповнити також контактні дані продавця.

Для продавців сервіс є платним, тому після заповнення всіх необхідних полів треба сплатити за публікацію оголошення. Продавцям доступні й додаткові платні функції для просування оголошення рис. 1.12 [3].

Користувач, який опублікував оголошення може перейти до перегляду, а також редагувати та видаляти раніше створені оголошення. Є можливість спілкування з іншими користувачами через особисті повідомлення та коментарі під оголошеннями.

Простий продаж	Швидкий продаж	Турбо продаж
6 грн за день	8 грн за день	9 грн за день
1 місяць у пошуку AUTO.RIA - Перевірка по VIN-коду - Реклама на ТОП позиціях в пошуку	1 місяць у пошуку AUTO.RIA - Перевірка по VIN-коду - Реклама на ТОП позиціях 4 дні	1 місяць у пошуку AUTO.RIA - Перевірка по VIN-коду - Реклама на ТОП позиціях 4 дні
Без реклами	ТОП 1 - № 1 - 10 грн за день	ТОП 2 - № 1 - 20 грн за день
4 дні розміщення в ТОП	4 дні розміщення в ТОП	4 дні розміщення в ТОП
200 грн	240 грн	280 грн
Опублікувати	Опублікувати	Опублікувати
24 Pay	24 Pay	24 Pay

Рисунок 1.12 – Можливості просування оголошення AUTO.RIA

Додавання оголошення

Основна інформація
Вибір характеристик вашого авто

Публікація оголошення
Вибір способу публікації оголошення

1
Додайте 2-3 фото з відкритим держ. номером

Щоб автоматично заповнити технічні дані авто
На перемикачі, ви можете **завантажити держ. номер**

Додати фото

Додати відео з YouTube

2
Основна інформація

тип транспорту, марка, модель, регіон

* Поля обов'язкові для заповнення

Тип транспорту

Оберіть

Марка авто

Оберіть

Модель авто

Оберіть

Рік випуску

Оберіть

Модифікація

Модифікація

Тип кузова

Оберіть

Пробіг

тис. км

Регіон

Оберіть

Місто

Оберіть

Авто зі статусом «Перевірений VIN-код» продаються удвічі швидше

Введіть VIN-код і ми **безкоштовно** перевіримо дані на предмет розшуку й дані останньої операції (за розстрани МСЄ), а також закриємо інформацію з порталом відкритих даних і дисковими базами.

VIN-код

VIN-код

Навіщо вказувати VIN-код авто?

3
Опис автомобіля

детально опишіть особливості вашого авто

В даному полі забороняється:

- Прогнозувати послуги (прикрену під замовлення, с інші авто, допомогу вибрати)
- Залишати посилання або контактні дані

Опис українською

Опис українською

Переклад

Зробіть ваше оголошення доступним для тих, хто шукає на російській. Для автоматичного перекладу оголошення на російську мову натисніть кнопку «Перекласти оголошення». Перевірте коректність запропонованого перекладу.

Перекласти на російську

Доступно 2000 символів

4
Характеристики авто

Зацікавте більше покупців

Колір

Оберіть

Металік

Оберіть

Участь в ДТП

Оберіть

Технічний стан

Оберіть

Лак/фарбове покриття

Оберіть

Призначення з

Оберіть

Рисунок 1.13 – Сторінка створення оголошень AUTO.RIA

1.1.4 Booking.com

Booking.com — система інтернет-пошуку та бронювання помешкання, заснована в Амстердамі. Наразі штаб-квартира знаходиться в Амстердамі та Нідерландах [4].

З самого початку функціонал сайту поділяє користувачів на два типи. Тих, хто пропонують помешкання, та тих, хто його шукає.

Користувач, який шукає житло має можливість скористатись швидким пошуком, де потрібно вказати куди він прямує, який термін поїздки та кількість осіб рис. 1.14 [4].



Рисунок 1.14 – Фільтри пошуку Booking.com

Також, є функція пошуку за типом помешкання (готелі, апартаменти, шале, котеджі тощо) рис. 1.15 [4].



Рисунок 1.15 – Рубрикатор Booking.com

При розширеному пошуку, для користувача, який шукає житло стають доступне встановлення таких критеріїв:

- бюджет (за ніч);

- рівень ековідповідальності помешкання;
- кількість зірок;
- відстань до центра міста;
- політика скасування;
- наявність харчування;
- наявність визначних місць неподалік;
- тип ліжка;
- наявність додаткових послуг.

За посиланням, що знаходиться зверху праворуч, відкривається функціонал, що дозволить розмістити своє помешкання для подальшого пошуку користувачів, яким ця пропозиція буде цікава рис. 1.16 [4].

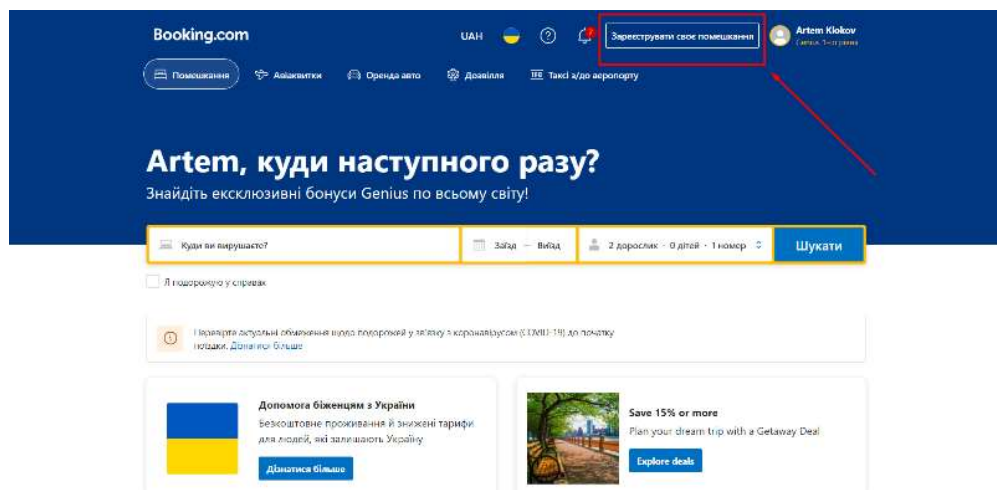


Рисунок 1.16 – Посилання для зміни ролевої моделі користувача Booking.com

Для того, щоб скористатися цим функціоналом, необхідно авторизуватись та обрати тип помешкання. Після цього треба пройти реєстрацію помешкання, де користувач має зазначити адресу, кількість гостей, тип ліжка, кількість ванних кімнат, доступний комфорт, інформацію про сніданки (тип сніданку), наявність паркінгу для гостей, мову персоналу та порядок проживання. Після реєстрації помешкання користувач може оформити профіль господаря, але це не

обов'язково. Також, необхідно додати фото помешкання та вказати ціну за помешкання (за одну ніч проживання). Реєстрація буде не можлива, якщо не зазначити, що ваше помешкання зареєстроване як платник податку на товари і послуги.

1.1.5 Trello

Trello – сервіс для керування проектами. Застосунок бува розроблений компанією Fog Creek Software. Діяльність компанії була націлена на спрощення командної взаємодії під час роботи над проектами різної напрямленості, складності та галузі. Але саме механіки Trello надають один з варіантів взаємодії користувачів з вебсервісами, на кшталт дошки оголошень рис. 1.17 [5].

У процесі використання, користувач Trello може створювати, або відслідковувати повідомлення, задачі та строки їх виконання. Як з іншими застосунками, користувачі діляться на дві ролі. Але й на відміну від вже проаналізованих сервісів, ці ролі не зв'язані з ролями покупця й продавця [5].

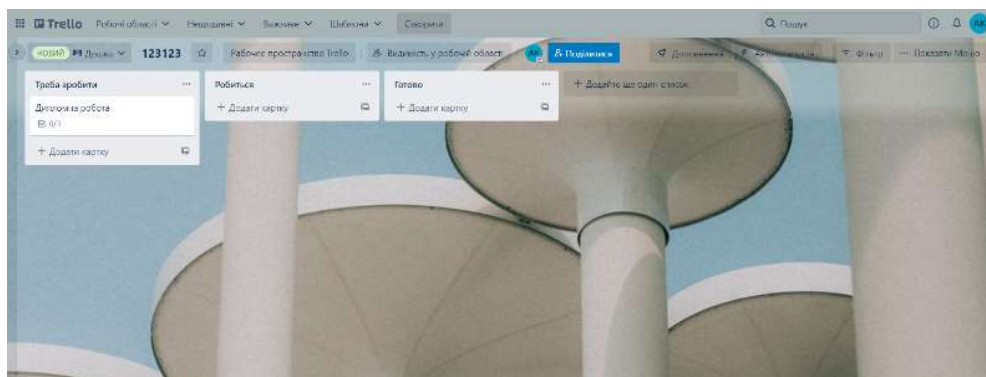


Рисунок 1.17 – Інтерфейс Trello

Перша з ролей – працівник, який зацікавлений в виконанні поставлених перед ним задач. Саме такі задачі відслідковує працівник на дошці, він може реагувати на завдання, змінюючи його статус виконання, тим саме, надаючи

лідеру проекту можливість в будь-який час тримати усі доступні завдання та їх статус на увазі.

Інша роль – лідер, який створює завдання, визначає строки та інші умови. Він може давати завдання конкретним користувачам, або сповіщати усю команду.

У контексті цієї роботи, що виконується, Trello нас цікавить своїми механіками створення та таргетування завдань або оголошень. Візуальна складова Trello нагадує дошку оголошень в її аутентичному вигляді.

1.2 Висновки за розділом

У цьому розділі було проаналізовано деякі існуючі рішення, визначена їх напрямленість та порівняна логіка роботи.

Кожен з розглянутих вебзастосунків має вузьку напрямленість. Так, Work.ua займається лише робочими пропозиціями [2], Booking.com – арендою, Trello – інформуванням та організацією робочого процесу [5], AUTO.RIA – комерційними пропозиціями щодо автомобілів [3]. Серед обраних прикладів можна відокремити OLX, як майданчик з найширшим колом можливостей, оскільки функціонал цього вебзастосунку дає можливість створювати або розглядати оголошення не тільки з продажу товарів, а й продажу послуг, рекрутіngu, оренди [1].

Щодо взаємодії користувача і системи, принцип роботи кожного з подібних сервісів мало відрізняється. Таким чином, користувач має можливість створювати оголошення, продивлятися їх списки, або кожне окреме детально, користуватися пошуком, фільтрами пошуку та рубрикатором. У цьому сенсі найнезвичайнішим, через свою специфікацію виявився вебзастосунок Trello. По-перше, оголошення на Trello має відбірну аудиторію та конкретизує адресантів

окремого повідомлення. По-друге, в кожному оголошенні (завданні) додається набір умов, таких як строки, поточний стан виконання, інше.

OLX, AUTO.RIA, Work.ua заради зручності використання вебзастосунку відповідають принципу single-page (рубрикатор, пошук та список оголошень знаходяться на одній сторінці, і використовуються без підвантаження інших). Перехід на іншу сторінку здійснюється тільки для створення власного оголошення, або перегляду деталей конкретного оголошення зі списку.

Слід зазначити, що створення оголошень на кожному з розглянутих вебзастосунків має жорстку структуру, диктуючи умови написання користувачем нових оголошень (виключенням є Trello, через відмінність у напрямленості та функціональному навантаженні). Також, щоб уникнути порушення законодавства, фейків та спаму, кожен сервіс включає у процес створення оголошень етапи машинної та ручної модерації оголошень. Такі заходи необхідні, щоб захистити користувача від шахраїв та іншої можливої небезпеки під час використання сервісу, а також попередження репутаційних втрат.

2 РОЗРОБКА АРХІТЕКТУРИ ВЕБЗАСТОСУНКУ

2.1 Функціональні вимоги до вебзастосунок

Вебзастосунок, що розробляється, повинен виконувати мінімально необхідний ряд функцій:

- реєстрація нового користувача;
- авторизація користувача;
- вивід на головний екран оголошення, що відсортовані за датою публікації;
- зміна міста у фільтрах пошуку оголошень;
- зміна типу оголошення (продаж, оренда, робота) в фільтрі пошуку;
- перегляд повної інформації та контактних даних в окремому оголошенні;
- створювання користувачем власних оголошень різних типів (продаж, оренда, робота, реклама);
- редагування показнику «вартість» користувачем у власному оголошенні;
- видалення користувачем власних оголошень.

Функціональні можливості вебзастосунок залежать від статусу користувача у системі. За статусом користувачі поділяються на авторизованих та неавторизованих рис. 2.1.

Можливості неавторизованого користувача обмежуються переглядом списку оголошень, використанням фільтрів пошуку, можливістю отримати додаткову інформацію за обраним оголошенням.

Авторизованому користувачу доступні усі можливості, що й неавторизованому, але їх спектр розширюється додатковими. Так, авторизований користувач може ще й створювати власні оголошення, переглядати їх, редагувати деяку інформацію та видаляти.

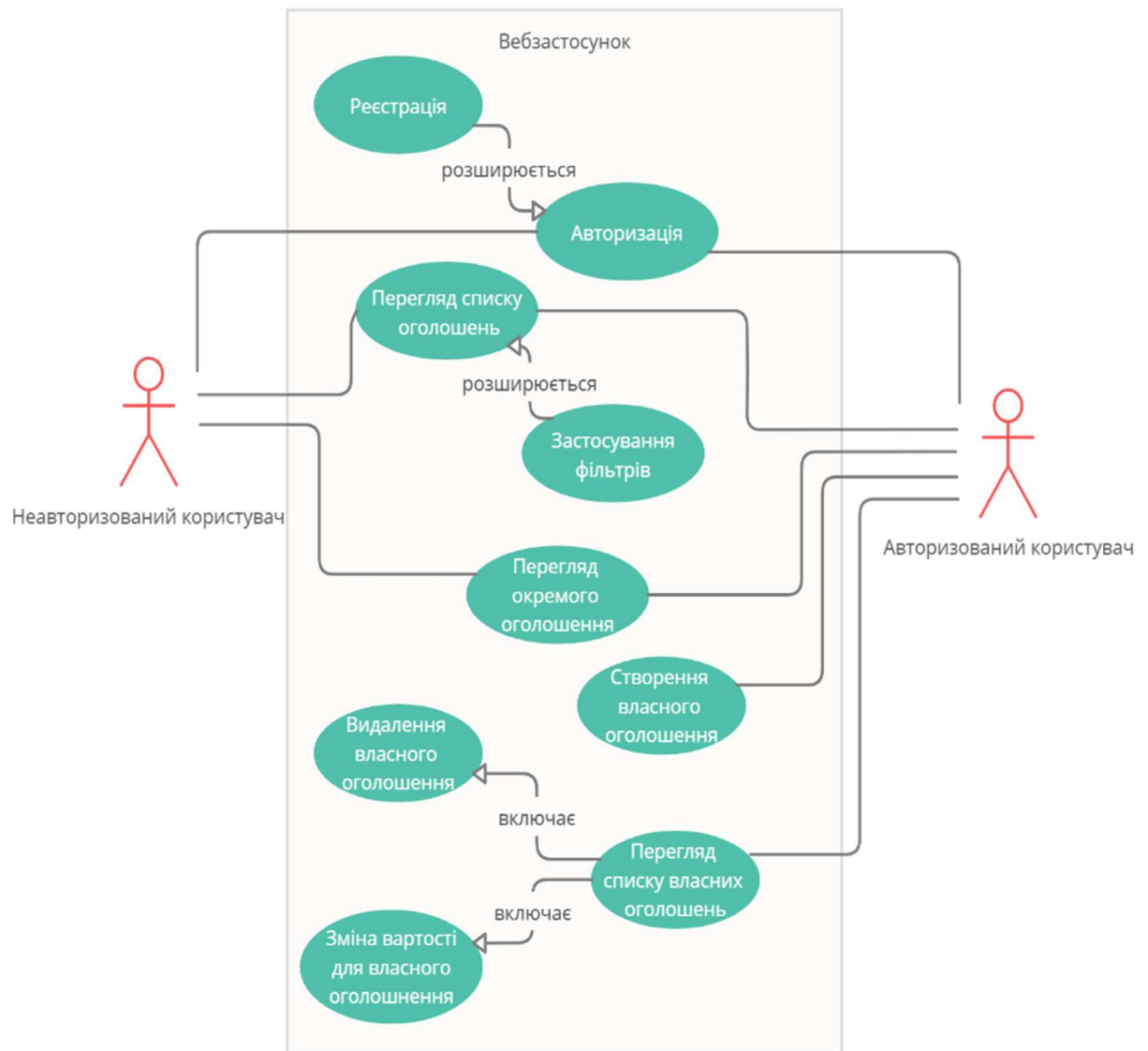


Рисунок 2.1 – Діаграма прецедентів авторизованого та неавторизованого користувача

2.2 Обґрунтування засобів розробки

Для розробки вебзастосунку дошки оголошень, а сама, клієнтської частини застосунку, було обрано наступний набір засобів:

- HTML;
- CSS;

– JavaScript.

Такий набір засобів вважається стандартним, адже CSS та Java Script можуть бути інтегровані в HTML по замовченню, що є безперечною перевагою для їх сумісного застосування.

HTML – мова розмітки гіпертексту. За її допомогою різні конструкції (текстові блоки, таблиці тощо), медіа (зображення, анімації тощо) та інші об'єкти відображаються на вебсторінці. Більшість браузерів мають саме вбудовані HTML інтерпретатори, що перетворюють HTML код, отриманий браузером з локального диску або за протоколами HTTP/HTTPS, в інтерфейс, що відображається користувачу [6].

CSS – каскадна таблиця стилів, вбудована в HTML. Застосовується для задання візуальних характеристик, таких як форма, шрифт, колір тощо, для блоків, тексту та інших елементів сторінки [7].

Вибір скриптової мови JavaScript зумовлений наступними чинниками:

- велика кількість фреймворків, завдяки яким, спрощується структура коду, та власне процес його написання;
- нативний зв'язок при роботі з HTML та CSS;
- динамічну типізацію, що зменшує ризик помилок при обробці даних різних типів.

2.3 Вибір фреймворку

Оскільки для розробки клієнтської частини була обрана мова JavaScript, одною з особливостей її є велика кількість фреймворків та бібліотек, що спрощують структуру коду та облегшують його написання.

Для того, щоб обрати найзручніше рішення, потрібно розуміти різницю між бібліотеками та фреймворками.

Фреймворк – програмна платформа, яка визначає структуру написання коду та розширює стандартний набір можливостей за допомогою вбудованих функцій для певної мови програмування [8].

Бібліотека – багаторазова частина коду, що має у собі певний перелік функцій, націлених на розширення можливостей або спрощення реалізації деяких функцій для розробника.

Нижче наведено таблицю порівняння бібліотеки та фреймворку, як інструменту для розробника (табл. 2.1).

Таблиця 2.1 – Порівняння бібліотек та фреймворків

Критерії порівняння	Бібліотека	Фреймворк
1. Розширення стандартних можливостей за допомогою вбудованих функцій	+	+
2. Повна свобода структури написання коду (відсутність каркасності)	+	-
3. Високий рівень абстракції	-	+
4. Нативна підтримка AJAX	-	+

В свою чергу, існує фреймворків та бібліотеки, що користуються попитом серед веб розробників:

- React;
- Angular;
- Vue.js;
- JQuery.

Було прийнято рішення обирати саме серед фреймворків, адже простота використання, каркасність та високий рівень абстрактності спрощує процес написання коду для вебзастосунку, що розроблюється.

При аналізі доступних рішень, вибір був зумовлений такими характеристиками, як реактивність, простота використання, динамічна типізація, що відрізняли Vue.js [9]. Особливістю використання Vue.js є реалізація SPA застосунку. SPA (Single Page Application) – вебзастосунок, що завантажує лише один HTML документ, в подальшому оновлюючи основний вміст цього документа за допомогою API JavaScript в момент запиту на оновлення інформації. Отже, для розробки клієнтської частини обрано саме фреймворк Vue.js.

2.4 Робота з даними

У парі з Vue.js для клієнтської частини, через сумісну реактивність (функції реагування клієнтської частини на зміни серверної, та навпаки), було використано один з найрозповсюдженіших фреймворків роботи серверної частини для Node.js – Express.js. Такий вибір обґрунтовується й зручністю використання, адже з поданим фреймворком є можливість зменшити часові витрати та об'єми написаного коду.

У якості СКБД було прийняте рішення використовувати MySQL. Ця система має відкритий вихідний код. MySQL використовує логічну модель з високим рівнем абстракції даних та мову структурованих запитів SQL [10]. Саме через реляційну модель в цій СКБД, MySQL досить проста в використанні та зрозуміла в налаштуванні, і разом з тим, містить достатню кількість функцій, сумісність з більшістю мов програмування, у тому числі й з обраними.

Авторизація користувача реалізована за допомогою розподіленого швидкого сховища Redis (Remote Dictionary Server) [11], що зберігає пари ключ-значення. Таке рішення є одним з найпопулярніших із відкритим вихідним кодом у багатьох сферах (ігри, фінансові сервіси, рекламні технології тощо).

2.5 Висновки за розділом

В цьому розділі було розглянуто можливі рішення для розробки вебзастосунку дошка оголошень, порівняно рішення в окремих переліках за призначенням, проаналізовано та обрано найзручніші у контексті цієї роботи.

Так, було обрано стандартний наступний набір засобів для WEB розробки клієнтської частини: HTML, CSS, JS. Для спрощення роботи з кодом, його формалізації та розширення вбудованого функціоналу було використано фреймворк Vue.js.

Сервісна частина реалізована за допомогою Express.js – фреймворку для Node.js, що має сумісність і реалізує реактивність з фреймворком клієнтської частини Vue.js. Роль СКБД у застосунку виконує MySQL [12], а в якості сховища, за допомогою якого реалізовано функцію авторизації, було обрано Redis.

Використання саме таких засобів створює необхідні умови для розробки проєкту, що виконується. Адже обраний набір є стандартним, має достатньо інформації в мережі та актуальну документацію.

3 РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ

3.1 Визначення функціональної поведінки вебзастосунку

Взаємодія користувача з застосунком обумовлює створення логічної послідовності виконання програми рис. 3.1. Така логічна послідовність має бути інтуїтивно зрозуміла й побудована на загальних принципах людино-машинної взаємодії.

Логіка поведінки вебзастосунку у рамках цього проекту дуже схожа на аналогічні рішення, розглянуті раніше.

Користувач, як і в прикладах, має 2 ролі. Перша з яких – потенційний споживач товарів та послуг (подалі покупець). Друга роль – користувач, що пропонує товари та послуги.

Спочатку роботи, неавторизований користувач потрапляє на головну сторінку, де відображається список актуальних оголошень. Такий користувач має можливість продивлятися список, переходити на сторінки обраних оголошень, для детального вивчення. Якщо користувача зацікавить окреме оголошення, він має можливість натиснути на контактні дані (номер телефону), для здійснення дзвінка і корекції подальшої взаємодії з продавцем.

Неавторизованому користувачу доступні фільтри в якості інструментів пошуку. Також користувач має змогу зареєструватися та авторизуватися.

Авторизований користувач може переглядати списки оголошень на головній сторінці, фільтрувати їх, переглядати список власних оголошень, видаляти їх та редагувати параметр «вартість».

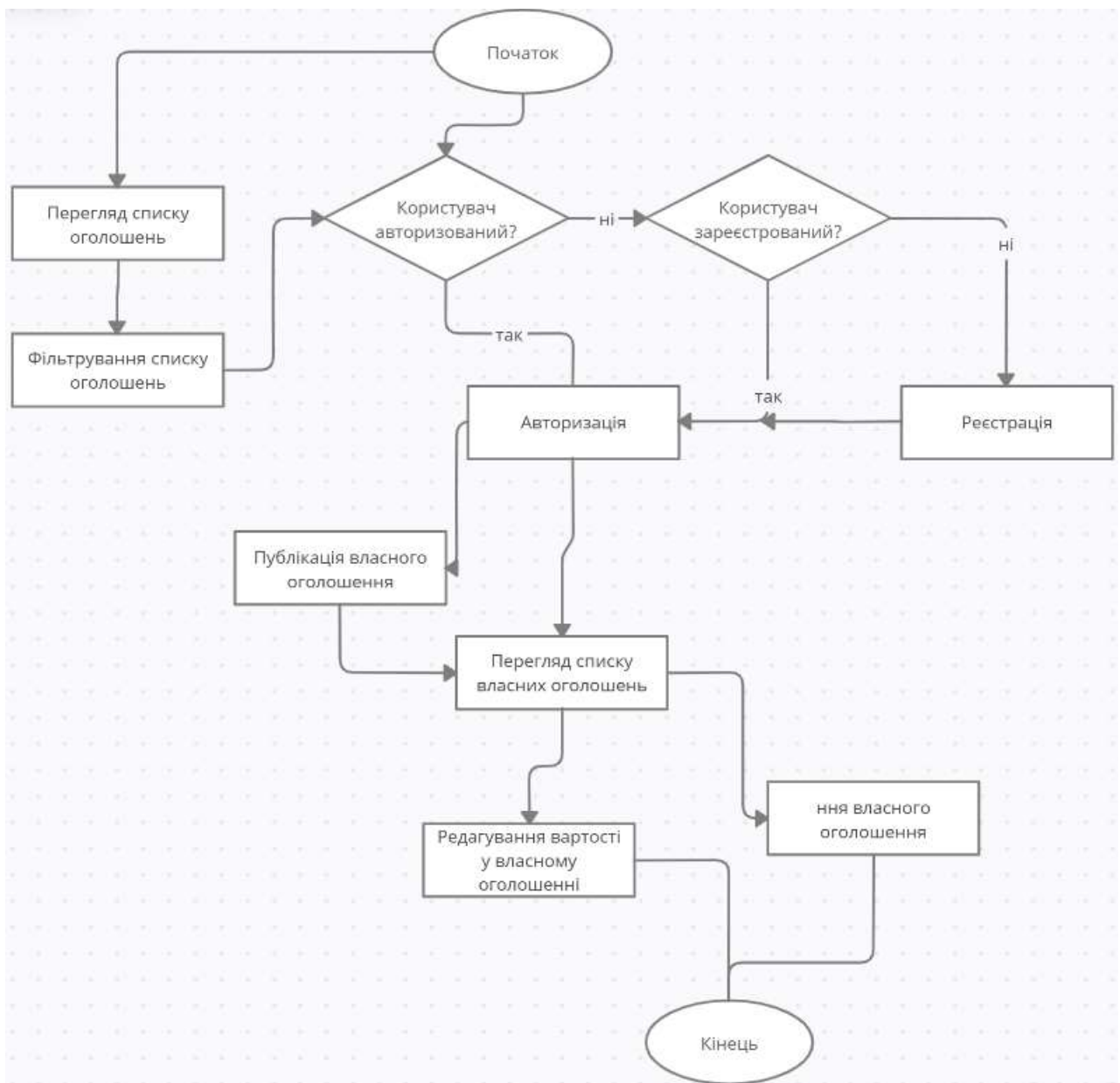


Рисунок 3.1 – Схема логіки роботи вебзастосунку

3.2 Робота з даними

В контексті поданого проєкту існують дві сутності, що потребують представлення у БД.

Такими сутностями є User та Lot.

User – раніше зареєстрований обліковий запис, що має певний набір даних.

Lot – створене користувачем оголошення, що містить у собі набір даних.

Для таблиці User існують такі поля, як:

- id (первинний ключ користувача);
- userName (ім'я користувача);
- password (пароль, що задає користувач для доступу до облікового запису);
- telephone (номер телефону користувача);
- email (адрес електронної пошти користувача).

Таблиця Lot має наступні поля:

- id (первинний ключ оголошення);
- title (назва оголошення);
- description (опис оголошення);
- type (тип оголошення);
- photoOfLot (фотографії, що додаються до оголошення);
- userName (ім'я користувача, що опублікував оголошення);
- telephone (номер телефону користувача, що опублікував оголошення);
- price (ціна, вказана для предмету оголошення);
- city (місто, в якому дійсне оголошення).

Для таблиць User та Lot, за допомогою Sequelize ORM, були створені відповідні моделі [13]. Таким чином, таблиці User відповідає наступний клас User рис. 3.2.

```

1  "use strict";
2  const { Model } = require("sequelize");
3
4  class User extends Model {
5    static init(sequelize, DataTypes) {
6      return super.init(
7        {
8          userName: DataTypes.STRING,
9          password: DataTypes.STRING,
10         email: {
11           type: DataTypes.STRING,
12           validate: { isEmail: true },
13         },
14         telephone: DataTypes.STRING,
15       },
16       {
17         sequelize,
18         modelName: "User",
19       }
20     );
21   }
22 }
23
24 module.exports = User;

```

Рисунок 3.2 – Файл, що містить клас User

Таблиці Lot відповідає клас Lot рис. 3.3.

```

auction > auction-be > models > JS lot.js > Lot > init
1  "use strict";
2
3  const { Model } = require("sequelize");
4
5
6  class Lot extends Model {
7    static init(sequelize, DataTypes) {
8      return super.init(
9        {
10         title: DataTypes.STRING,
11         description: DataTypes.STRING,
12         type: DataTypes.STRING,
13         photoOfLot: DataTypes.TEXT,
14         userName: DataTypes.STRING,
15         telephone: DataTypes.STRING,
16         price: DataTypes.INTEGER,
17         city: DataTypes.INTEGER
18       },
19       {
20         sequelize,
21         modelName: "Lot",
22       }
23     );
24   }
25 }
26
27 module.exports = Lot;

```

Рисунок 3.3 – Файл, що містить клас Lot

Нижче представлений код файлу `index.js`, представлений у серверній частині вебзастосунку, за допомогою якого дані імпортуються з БД для подальшого використання:

```
const Sequelize = require("sequelize");
const User = require("../models/user");
const Lot = require("../lot");

const sequelize = new Sequelize(
  process.env.MYSQL_DB,
  process.env.MYSQL_USER,
  process.env.MYSQL_PASSWORD,
  {
    host: process.env.MYSQL_HOST,
    port: process.env.MYSQL_PORT,
    dialect: "mysql",
  }
);

const models = {
  User: User.init(sequelize, Sequelize),
  Lot: Lot.init(sequelize, Sequelize)
};

Object.values(models)
  .filter((model) => typeof model.associate === "function")
  .forEach((model) => model.associate(models));

const db = { ...models, sequelize };

module.exports = db;
```

3.3 Структура застосунку

3.3.1 Структура клієнтської частини

Клієнтська частина або FE рис. 3.4 містить наступні складові:

- `views` (представлення) – динамічні вебсторінки з власною розміткою, стилізацією та певною логікою;
- `components` (компоненти) – елементи, що використовуються повторно;

- `store` (файли сховища) – частина глобального сховища, які забезпечують логіку залежності елементів БД, правила їх обробки та збереження під час використання вебзастосунку;

- `middleware` – файл, який забезпечує логіку застосунку для неавторизованого користувача.

Серед представлень існують:

- `HomeView` – домашня сторінка, що відображає список оголошень;
- `MyLotsView` – сторінка, що відображає список власних оголошень авторизованого користувача;

- `LoginView` – сторінка авторизації;
- `RegisterView` – сторінка реєстрації.

Застосунок використовує наступні компоненти:

- `PageContainer` – область для відображення списку оголошень;
- `AuctionItem` – область відображення оголошення у списку;
- `Sidebar` – область бокового меню;
- `SidebarItem` – елементи бокового меню, з якими взаємодіє користувач.

У якості файлу сховища у папці `store` знаходиться `index.js`, а логіка поведінки для неавторизованого користувача описана в файлі `auth.js`, що знаходиться в папці `middleware`.

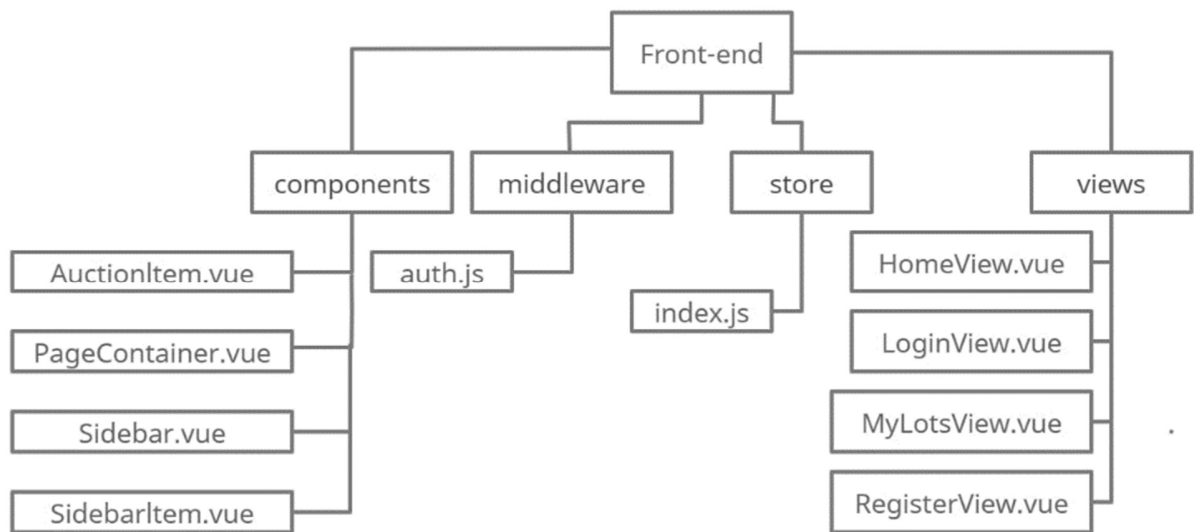


Рисунок 3.4 – Схема структури клієнтської частини

3.3.2 Структура серверної частини

Структура серверної частини або BE рис. 3.5 вебзастосунку має такі складові:

- models (моделі) – описані у коді класи, відповідні до таблиць БД;
- controllers (контролери) – функціональна частина вебзастосунку, що забезпечує виконання логіки, при взаємодії сервісів;
- services (сервіси) – частини вебзастосунку, які взаємодіють з СКБД, отримуючи та направляючи дані;
- middleware – файл, який забезпечує логіку застосунку для неавторизованого користувача.

В якості моделей для вебзастосунку створені наступні файли:

- user.js – містить опис моделі User;
- lot.js – містить опис моделі Lot;
- index.js – керуючий файл, завдяки якому відбувається імпорт моделей до функціональної частини програми.

Серед контролерів використовуються:

- `user.controller.js` – відповідає за логіку, пов'язану з обліковими записами;

- `lot.controller.js` – відповідає за логіку, пов'язану з оголошеннями.

Сервісами застосунку є:

- `user.service.js` – відповідає за взаємодію з моделлю User;

- `lot.service.js` – відповідає за взаємодію з моделлю Lot.

Логіка поведінки для неавторизованого користувача описана в файлі `auth.js`, що знаходиться в папці `middleware`.

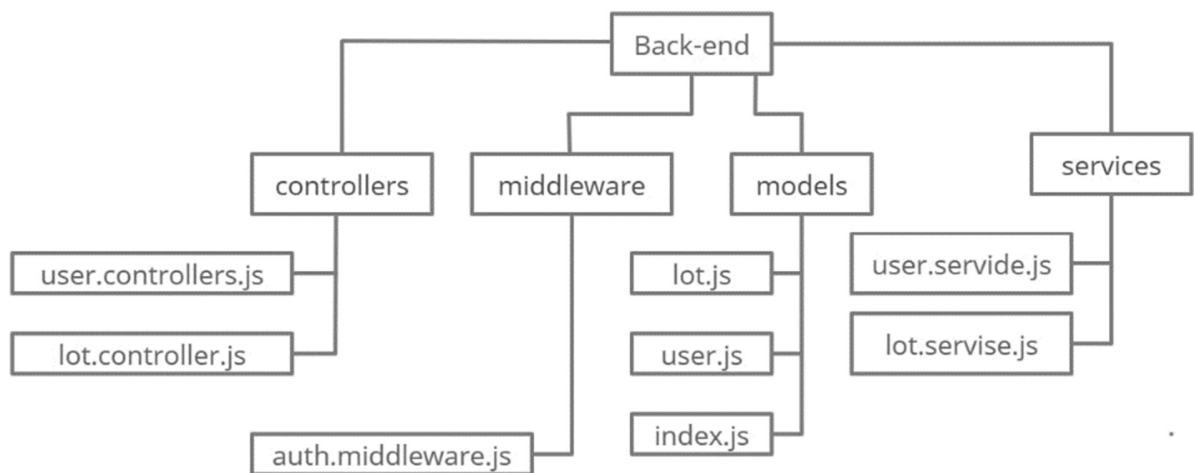


Рисунок 3.5 – Схема структури серверної частини

3.4 Розробка інтерфейсу користувача

Задачею інтерфейсу користувача є забезпечення взаємодії користувача з логікою застосунку, надання йому можливості інтуїтивно просто знаходити елементи керування, необхідні для використання тієї чи іншої функції застосунку.

В контексті поданого проекту інтерфейс користувача має бути однорідний (різні сторінки не повинні відрізнятися за дизайном та навігацією), інтуїтивно зрозумілий (елементи керування, необхідні для використання функціональних можливостей застосунку, мають бути помітними, однозначно підписаними, за можливістю супроводжуватися іконками), не мати важкої ієрархії (функції, за можливістю, мають бути доступні з головного меню) [14].

Таким чином, зважаючи на всі фактори, була створена схема розмітки та дизайну головної сторінки, яка й диктує зовнішній вигляд іншим сторінкам рис. 3.6.

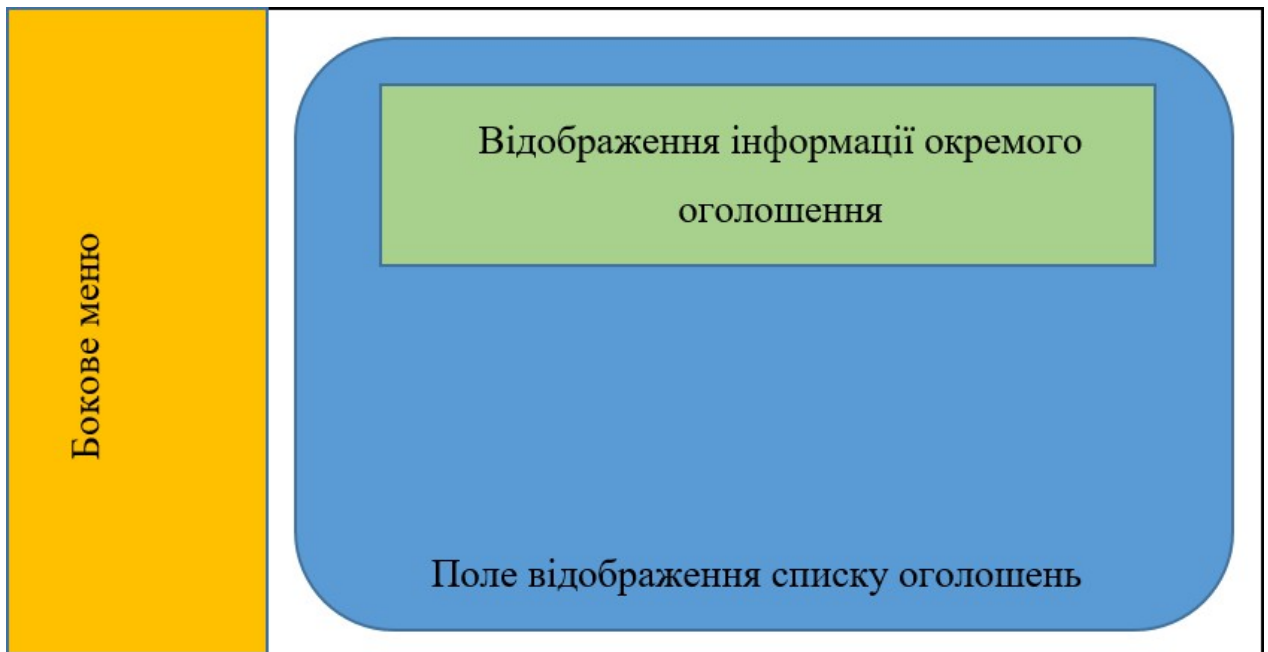


Рисунок 3.6 – Схема побудови інтерфейсу користувача

3.5 Висновки за розділом

В поданому розділі було описано функціональну поведінку застосунку, структуру клієнтської та серверної частини проєкту, здійснено опис роботи з БД,

надані відповідні схеми. Здійснено опис файлів застосунку, класів моделей та визначення зв'язку між клієнтською, серверною частинами та таблицями БД.

4 КЕРІВНИЦТВО КОРИСТУВАЧА

4.1 Призначення вебзастосунку

Поданий вебзастосунок створений у цілях організації комерційної діяльності щодо купівлі-продажу, оренди, найму робітників підприємцями або приватними особами. Реалізація проекту надає можливість користувачам до реєстрації, авторизації, перегляду існуючих оголошень купівлі, створенню власних оголошень з подальшою можливістю їх редагування та видалення. Оголошення поділяються на 4 типи, надаючи можливість продажу, найму, здачі в оренду та проведення рекламної компанії. Такий принцип одночасно закриває найпопулярніші сценарії взаємодії користувача з подібними застосунками.

4.2 Умови використання вебзастосунку

Мінімальні технічні вимоги до серверу:

- жорсткий диск з вільним простором – 8 Гб;
- оперативна пам'ять – 4 Гб;
- процесор: 4 ядра, 3,5 ГГц частота;
- доступ до мережі інтернет;

Взаємодія користувача та вебзастосунку виконується через браузер, що потребує наявності сучасного браузера, на кшталт Microsoft Edge, Mozilla Firefox, Google Chrome. Версія браузера має відповідати актуальній.

4.3 Опис процесу використання вебзастосунку

Для використання поданого застосунку в контексті цієї роботи, користувачу необхідно обрати веб-браузер, та ввести адресу сайту, що був

розроблений. Наразі сайт є локальним, і для його запуску необхідно ввести в адресну строку «localhost:3000»

Взаємодія користувача з вебзастосунком починається відвідуванням головної сторінки. На головній сторінці відображується глобальний список оголошень, створених іншими користувачами рис. 4.1. У цьому списку представлена інформація по кожному окремому оголошенню, а також можливість зв'язатися з продавцем, роботодавцем, орендодавцем чи рекламодавцем, в залежності від типу оголошення, натиснувши на контактний номер телефону, який вказано в оголошенні.

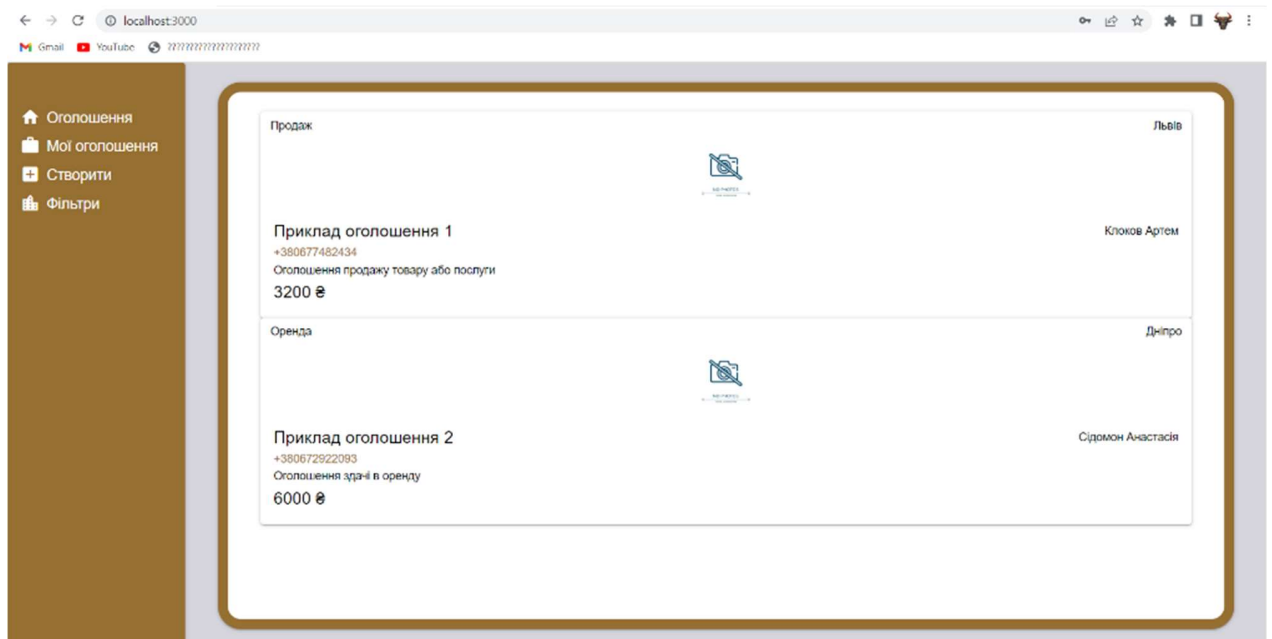


Рисунок 4.1 – Головна сторінка вебзастосунку

У лівій частині екрану, під час використання вебзастосунку, користувач має перманентний доступ до бокового меню, яке в свою чергу є і головним меню усього застосунку.

Ще не авторизований користувач має доступ до фільтрів оголошень, при натисканні на кнопку з написом «Фільтри» у боковому меню. Доступ до фільтрів організовано за допомогою модального вікна рис. 4.2.

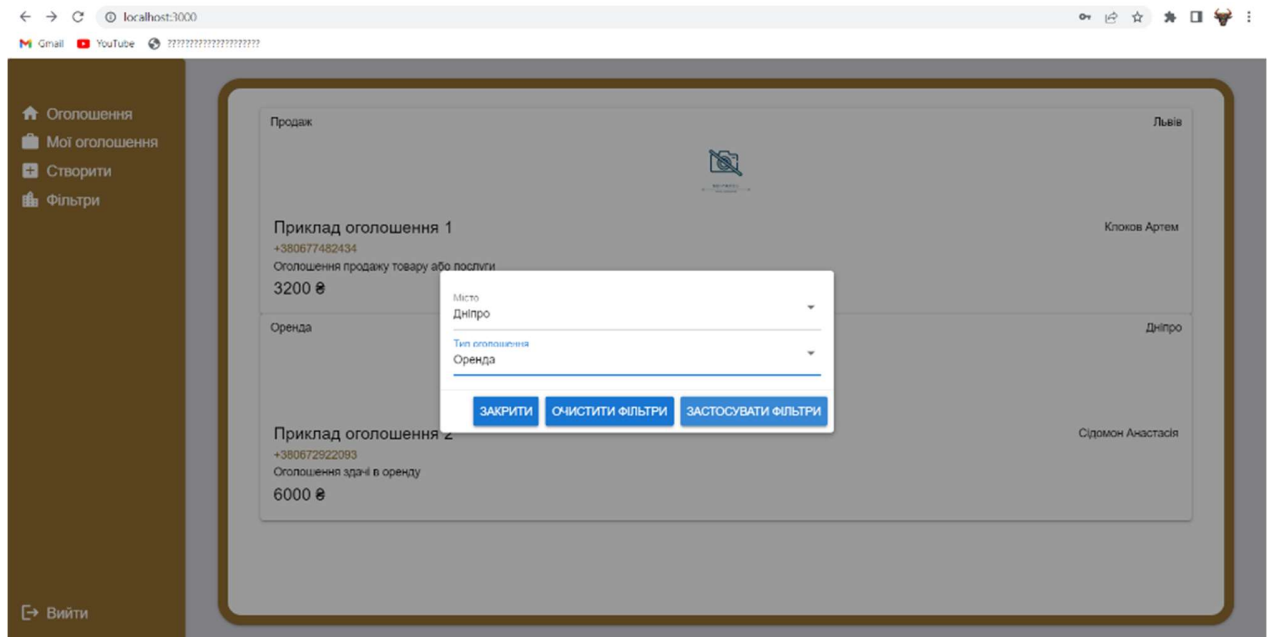


Рисунок 4.2 – Вікно фільтрів для оголошень

Після застосування фільтрів змінюється й список, відображаючи тільки ті оголошення, що відповідають вказаним фільтрам.

При натисканні на кнопки бокового меню з написами «Мої оголошення» або «Створити», користувач бачить на екрані модальне вікно авторизації (рис. 4.3), поля якого необхідно заповнити валідними даними існуючого облікового запису. Якщо користувач новий, і не має облікового запису, то при натисканні на «Зареєструватися» у формі авторизації, користувач переходить до форми реєстрації (рис. 4.4), де, також, необхідно заповнити усі поля валідними даними. Після реєстрації користувач має пройти авторизацію свого облікового запису.

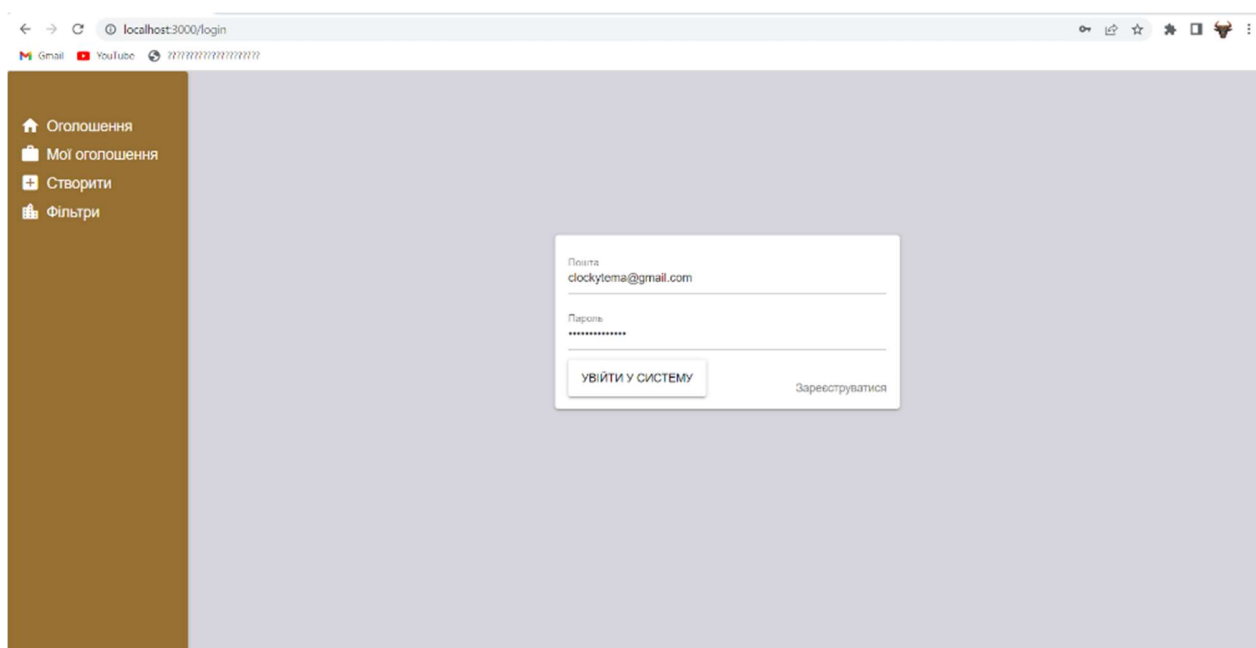


Рисунок 4.3 – Модальне вікно авторизації, з полями, що заповнені валідними даними існуючого облікового запису

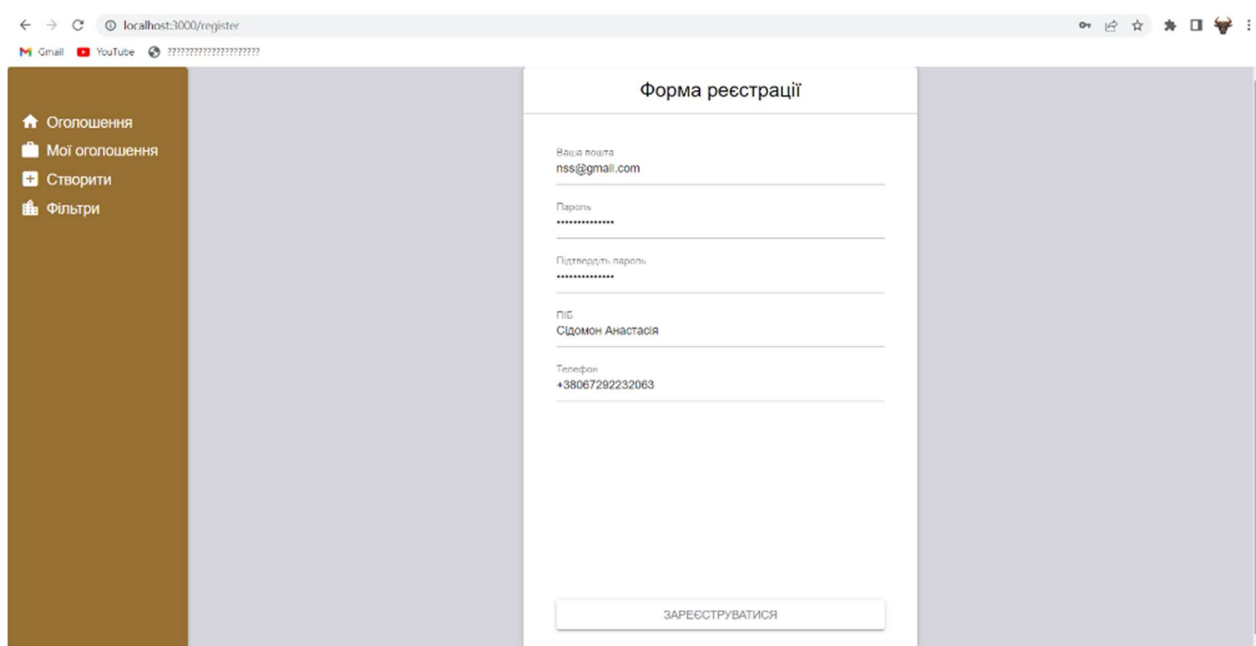


Рисунок 4.4 – Модальне вікно реєстрації нового користувача, з полями, що заповнені валідними даними

Авторизований користувач має повний доступ до функціоналу веб застосунку, а отже, при натисканні на кнопку «Створити», що знаходиться у боковому меню, він отримує доступ до модального вікна створення оголошення рис. 4.5, де користувачеві необхідно заповнити поля та обрати фото. Оголошення буде опубліковано після натискання на кнопку «Опублікувати». Якщо користувач за будь-якою причиною не хоче створювати оголошення, йому достатньо натиснути кнопку «Закрити». При цьому, протягом сесії, зберігається умовна чернетка, тобто уся додана, але не опублікована інформація. Після чого, користувач в будь-який момент може повернутися до створення ще неопублікованого оголошення з моменту, на якому він закінчив.

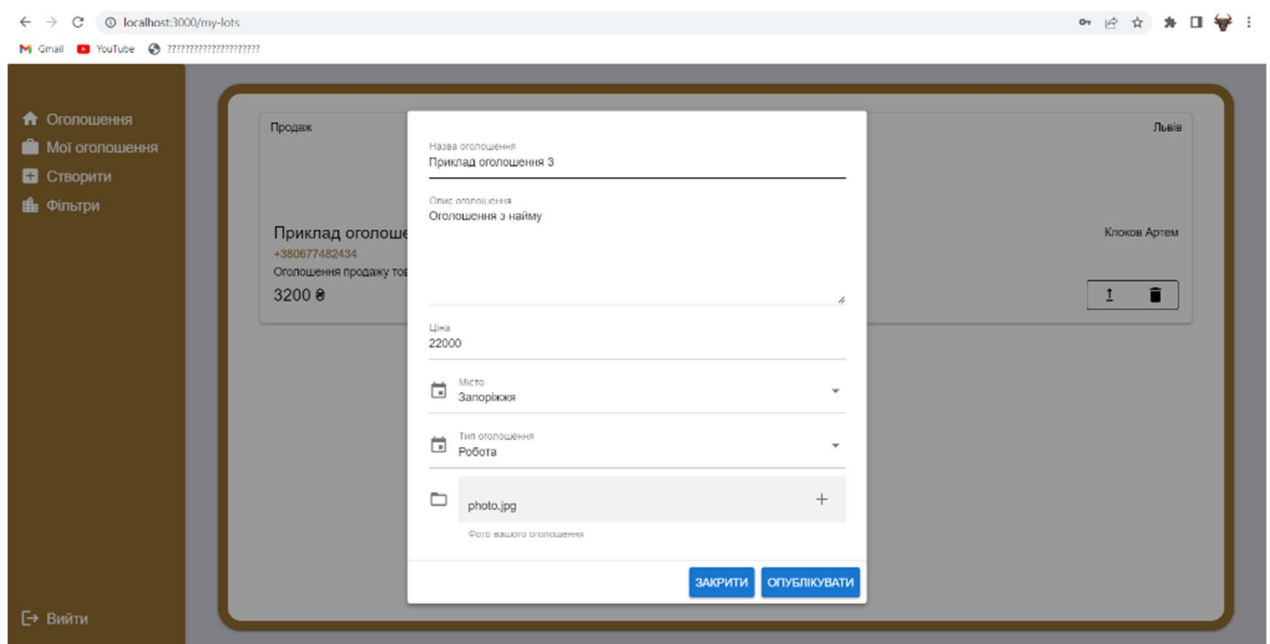


Рисунок 4.5 – Модальне вікно створення оголошення авторизованим користувачем

Після публікації оголошення користувачем, воно відображається у глобальному списку оголошень при відповідно встановлених фільтрах. Створене оголошення відображається й у списку власних оголошень користувача рис. 4.6.

До цього списку можна перейти, натиснувши на кнопку «Мої оголошення» у боковому меню.

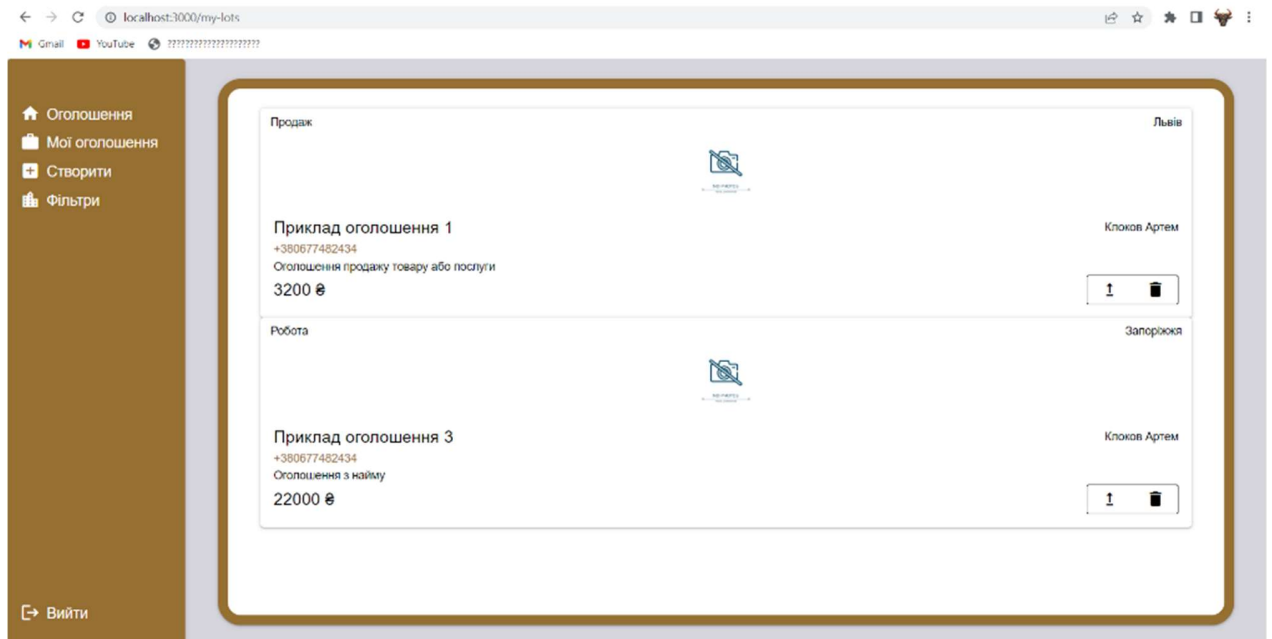


Рисунок 4.6 – Сторінка «Мої оголошення» зі списком оголошень, створених авторизованим користувачем

На сторінці «Мої оголошення» рис. 4.6, користувач має можливість не тільки переглядати власні оголошення, а й редагувати вартість рис. 4.7, вказану для лоту в оголошенні, або видалити оголошення рис. 4.8, натискаючи на відповідні кнопки, що знаходяться знизу праворуч від кожного окремого оголошення. Зміну вартості або видалення оголошення необхідно підтвердити в діалоговому вікні, що відкривається відразу після натискання на описані вище кнопки.

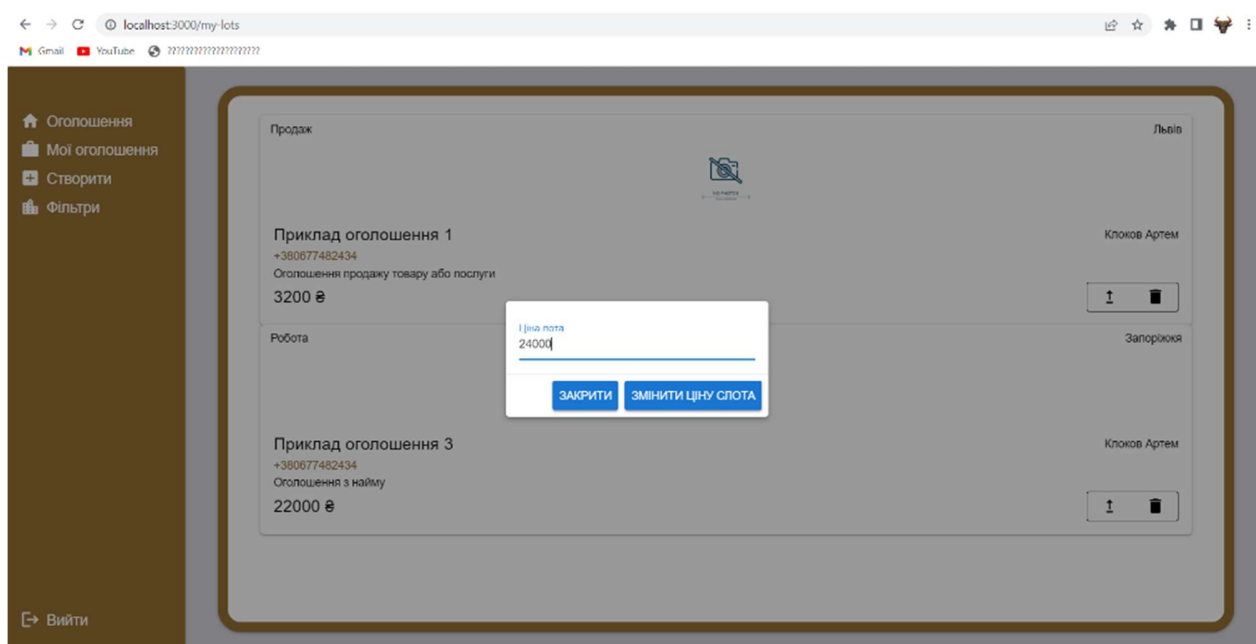


Рисунок 4.7 – Модальне вікно зміни вартості лоту для оголошення авторизованого користувача

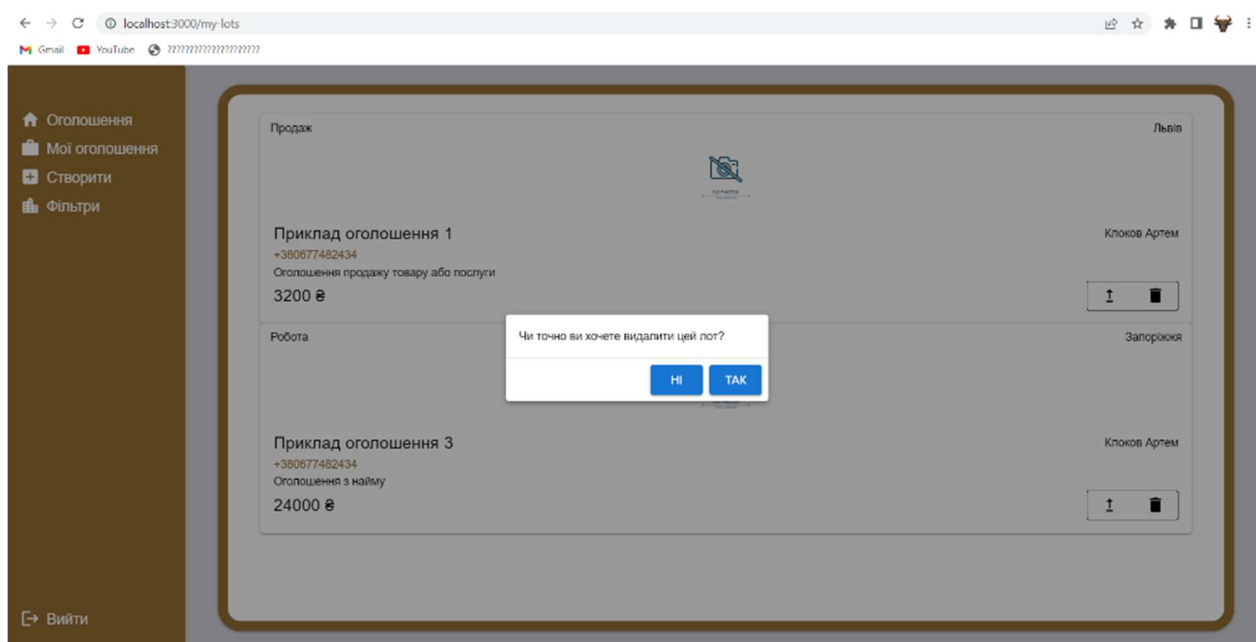


Рисунок 4.8 – Модальне вікно видалення оголошення авторизованого користувача

4.4 Висновки за розділом

У цьому розділі було описано загальне призначення вебзастосунку, що розробляється. Також складено список мінімальних технічних вимог, які дозволять користуватися вебзастосунком без перебоїв. Вебзастосунок має просту логіку використання, поділяючи користувачів за статусом на авторизованих та не авторизованих. У розділі детально описано й процес використання сайту, до кожного прецеденту наведено приклад роботи власне розробленого вебзастосунку.

ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи було розв’язано такі задачі:

- аналіз предметної області, шляхом дослідження специфіки роботи вебзастосунків типу дошка оголошень з боку користувача. При цьому були розглянуті відмінні за напрямленням застосунки, виділено ключові відмінності кожного з них;

- проектування архітектури вебзастосунку, що розроблюється, шляхом формулювання функціональних вимог, створення моделі роботи програми у вигляді діаграм прецедентів, обґрунтуванням засобів розробки, таких як мова програмування, фреймворки, система керування БД, з подальшим визначенням структури БД;

- кінцева реалізація програмного забезпечення з повним описом алгоритмів роботи програми у вигляді блок-схеми та текстової складової, структури взаємодії клієнтської та серверної частини вебзастосунку, таблиць і моделей СКБД;

- створення керівництва користувача з визначенням призначення застосунку, технічних вимог, описом та прикладами роботи розробленого рішення.

Розроблений вебзастосунок дошки оголошень надає користувачеві можливість здійснювати певні комерційні операції з купівлі/продажу, оренди, найму та ведення рекламної компанії на базі сайту. Користувач має можливість продивлятися пропозиції певних категорій, які його цікавлять, у вигляді списку оголошень, фільтрувати оголошення за певними критеріями, а також власне створювати такі пропозиції у вигляді оголошень, з подальшою можливістю редагування або видалення. Також, для користувача доступні авторизація та реєстрація на сайті застосунку дошка оголошень.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. OLX [Електронний ресурс]. – Режим доступу: <https://www.olx.ua/>.
2. Work.ua [Електронний ресурс]. – Режим доступу: <https://www.work.ua/?setlp=ua>.
3. AUTO.RIA [Електронний ресурс]. – Режим доступу: <https://auto.ria.com/uk/>.
4. Booking [Електронний ресурс]. – Режим доступу: <https://www.booking.com/index.uk.html>.
5. Trelo [Електронний ресурс]. – Режим доступу: <https://trello.com/>.
6. HTML [Electronic resource]. – Access mode: <https://html.spec.whatwg.org/multipage/>.
7. CSS [Електронний ресурс] – Режим доступу: <https://css.in.ua/css/properties>.
8. JS Фреймворки [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Порівняння_фреймворків_JavaScript.
9. Vue.js [Електронний ресурс]. – Режим доступу: <http://xn--80adth0aefm3i.xn--j1amh/Vue.js>.
10. MySQL [Electronic resource]. – Access mode: <https://www.mysql.com/>.
11. Redis [Electronic resource]. – Access mode: <https://redis.io/>.
12. MySQL/Startup Guide [Electronic resource]. – Access mode: https://wiki.gentoo.org/wiki/MySQL/Startup_Guide.
13. Siquelize ORM [Electronic resource]. – Access mode: <https://sequelize.org/>.
14. User Interfaces – Information, People, and Technology [Electronic resource]. – Access mode: <https://unizin.org>.
15. JavaScrit, Учебный материал [Электронный ресурс] – Режим доступа: <https://developer.mozilla.org/ru/docs/Web/JavaScript>.

16. Довідник по HTML тегам [Електронний ресурс]. – Режим доступу: <https://css.in.ua/html/tags>.

17. JavaScript [Electronic resource]. – Access mode: https://developer.mozilla.org/uk/docs/Learn/Getting_started_with_the_web/JavaScript_basics.

18. Веб приложение. Разница между сайтом и веб-приложением, SPA И PWA [Электронный ресурс]. – Режим доступа: <https://webcase.com.ua/blog/cho-takoe-web-prilozhenie-vse-vidy/>.

19. Microsoft Visual Studio Code [Electronic resource]. – Access mode: <https://code.visualstudio.com/>.

20. jQuery [Электронный ресурс]. – Режим доступа: <http://www.wisdomweb.ru/JQ/index.php>.

21. Angular [Электронный ресурс]. – Режим доступа: https://developer.mozilla.org/ru/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks/Angular_getting_started.

22. React [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/React/>

ДОДАТОК А
Текст програми

A.1 Код файла lot.js

```
"use strict";
const { Model } = require("sequelize");
class Lot extends Model {
  static init(sequelize, DataTypes) {
    return super.init(
      {
        title: DataTypes.STRING,
        description: DataTypes.STRING,
        type: DataTypes.STRING,
        photoOfLot: DataTypes.TEXT,
        userName: DataTypes.STRING,
        telephone: DataTypes.STRING,
        price: DataTypes.INTEGER,
        city: DataTypes.INTEGER
      },
      {
        sequelize,
        modelName: "Lot",
      }
    );
  }
}
module.exports = Lot;
```

A.2 Код файла user.js

```
"use strict";
const { Model } = require("sequelize");
class User extends Model {
  static init(sequelize, DataTypes) {
    return super.init(
      {
        userName: DataTypes.STRING,
        password: DataTypes.STRING,
        email: {
          type: DataTypes.STRING,
          validate: { isEmail: true },
        },
      },
      {
        sequelize,
        modelName: "User",
      }
    );
  }
}
module.exports = User;
```

```

    },
    telephone: DataTypes.STRING,
  },
  {
    sequelize,
    modelName: "User",
  }
);
}
}
module.exports = User;

```

A.3 Код файла index.js

```

const Sequelize = require("sequelize");
const User = require("../models/user");
const Lot = require("../lot");

const sequelize = new Sequelize(
  process.env.MYSQL_DB,
  process.env.MYSQL_USER,
  process.env.MYSQL_PASSWORD,
  {
    host: process.env.MYSQL_HOST,
    port: process.env.MYSQL_PORT,
    dialect: "mysql",
  }
);

const models = {
  User: User.init(sequelize, Sequelize),
  Lot: Lot.init(sequelize, Sequelize)
};

Object.values(models)
  .filter((model) => typeof model.associate === "function")
  .forEach((model) => model.associate(models));

const db = { ...models, sequelize };

module.exports = db;

```

A.4 Код файлу lot.controller.js

```

const asyncHandler = require("express-async-handler");
const router = require("express").Router();
const auth = require("../middleware/auth.middleware");
const LotService = require("../services/lot.service");
async function getLots(req, res) {
  const data = await LotService.findAll(req.query.city, req.query.type);
  return res.status(200).json(data);
}
async function getMyLots(req, res) {
  const data = await LotService.getMy(req.user.userName);
  return res.status(200).json(data);
}
async function createNewLot(req, res) {
  const lot = {
    ...req.body,
    userName: req.user.userName,
    telephone: req.user.telephone,
  };
  const data = await LotService.create(lot);
  return res.status(200).json( {message: "Оголошення створено!"} );
}
async function updateLot(req, res) {
  const data = await LotService.update(req.body, req.params.id);
  return res.status(200).json( {message: "Оголошення оновлено!"} );
}
async function deleteLot(req, res) {
  const data = await LotService.destroy(req.params.id);
  return res.status(200).json( {message: "Оголошення видалено!"} );
}
module.exports = (app) => {
  router.post("/", auth, asyncHandler(createNewLot));
  router.get("/", asyncHandler(getLots));
  router.get("/getMy", auth, asyncHandler(getMyLots));
  router.patch("/:id", auth, asyncHandler(updateLot));
  router.delete("/:id", auth, asyncHandler(deleteLot));
  app.use("/api/lot", router);
};

```

A.5 Код файла lot.service.js

```

const db = require("../models");
const Lot = db.Lot;
const { Op } = require("sequelize");
exports.create = async function (body) {
  return await Lot.create(body);
};
exports.update = async function (body, id) {
  return await Lot.update(body, {
    where: { id: id },
  });
};
exports.findAll = async function (type, city) {
  return await Lot.findAll({
    where: {
      type: { [Op.substring]: type },
      city: { [Op.substring]: city },
    },
  });
};
exports.destroy = async function (id) {
  return await Lot.destroy({
    where: { id: id },
  });
};
exports.getMy = async function (userName) {
  return await Lot.findAll({
    where: { userName: { [Op.substring]: userName } },
  });
};

```

A.6 Код файла auth.middleware.js

```

const jwt = require("jsonwebtoken");

module.exports = (req, res, next) => {
  if (req.method === "OPTIONS") {
    return next();
  }

```

```

}
try {
  const token = req.headers.token;
  if (!token) {
    return res.status(401).json({ message: "Користувач не авторизований" });
  }
  const decoded = jwt.verify(token, "lot_secret");
  req.user = decoded;
  next();
} catch (e) {
  res.status(401).json({ message: "Користувач не авторизований" });
}
};

```

A.7 Код файлу HomeView.vue

```

<template>
  <PageContainer>
    <template #page>
      <div class="items-wrapper" v-for="(lot, index) in lotList" :key="index">
        <auction-item
          :title="lot.title"
          :description="lot.description"
          :telephone="lot.telephone"
          :author="lot.userName"
          :photoOfLot="lot.photoOfLot"
          :type="lot.type"
          :city="lot.city"
          :price="lot.price"
        />
      </div>
    </template>
  </PageContainer>
</template>

<script>
import PageContainer from "@components/PageContainer.vue";
import { mapActions, mapGetters, mapState } from "vuex";
import AuctionItem from "../components/AuctionItem.vue";
export default {

```

```

name: "HomePage",
components: {
  PageContainer,
  AuctionItem,
},
computed: {
  ...mapState(["lotList"]),
},
methods: {
  ...mapActions(["getList"]),
},
mounted() {
  this.getList();
},
};
</script>

```

```

<style lang="less" scoped>
.title {
  padding-bottom: 10px;
  border-bottom: 1px solid #d6d5db;
}
.items-wrapper {
  display: flex;
  flex-direction: column;
  align-items: center;
}
</style>

```

A.8 Код файла LoginView.vue

```

<template>
  <q-card class="login-form">
    <q-card-section>
      <q-form @submit="onSubmit" @reset="onReset">
        <q-input v-model="email" label="Почта" />

        <q-input
          v-model="password"
          class="mt-12"

```

```

    label="Пароль"
    type="password"
  />

  <div class="mt-12">
    <q-btn label="Увійти у систему" type="submit" class="200px" />
    <div class="register" @click="toRegister">Зареєструватися</div>
  </div>
</q-form>

</q-card-section>
</q-card>
</template>
<script>
import { mapActions, mapMutations, mapState } from "vuex";
import { required, email } from "vuelidate/lib/validators";
import useVuelidate from "@vuelidate/core";
import _ from "lodash";
import { SET_LOGIN_FORM } from "@constants/mutation-types";
export default {
  name: "Login",
  data() {
    return {
      v$: useVuelidate(),
    };
  },
  methods: {
    ...mapActions(["login"]),
    ...mapMutations([SET_LOGIN_FORM]),
    onSubmit() {
      this.v$.touch();
      if (this.v$.error) {
        return;
      } else {
        this.login().then(() => {
          this.$router.push({ name: "my-lots" });
        });
      }
    },
    toRegister() {
      this.$router.push({ path: "register" });
    },
  },

```

```

    },
    computed: {
      ...mapState(["loginForm"]),
      password: {
        get() {
          return _.get(this.loginForm, "password");
        },
        set(value) {
          this[SET_LOGIN_FORM]({ password: value });
        },
      },
    },
    email: {
      get() {
        return _.get(this.loginForm, "email");
      },
      set(value) {
        this[SET_LOGIN_FORM]({ email: value });
      },
    },
  },
  validations: {
    password: {
      required,
    },
    email: {
      required,
      email,
    },
  },
};
</script>

```

```

<style lang="less" scoped>
.mt-12 {
  margin-top: 12px;
  display: flex;
  justify-content: space-between;
}
.login-form {
  margin: 200px auto;
  width: 420px;
  max-height: 212px;
}

```

```

.full-width {
  width: 100%;
}
.register {
  margin-top: 24px;
  color: #858585;
  cursor: pointer;
}
}
</style>

```

A.9 Код файлу MyLotsView.vue

```

<template>
  <PageContainer>
    <template #page>
      <div class="items-wrapper" v-for="(lot, index) in myLots" :key="index">
        <auction-item
          :id="lot.id"
          :title="lot.title"
          :description="lot.description"
          :telephone="lot.telephone"
          :author="lot.userName"
          :photoOfLot="lot.photoOfLot"
          :type="lot.type"
          :city="lot.city"
          :price="lot.price"
          update-edit
          @changePrice="openPriceModal"
          @removeLot="openRemoveLotModal"
        />
      </div>
      <q-dialog v-model="priceModalOpened">
        <q-card :style="{ width: '320px' }">
          <q-card-section>
            <q-form>
              <div class="price-lot">
                <q-input v-model="price" label="Ціна лота" class="field" />
              </div>
            </q-form>
          </q-card-section>
        </q-card>
      </q-dialog>
    </template>
  </PageContainer>
</template>

```

```

    </q-card-section>

    <q-separator />

    <q-card-actions align="right">
      <q-btn label="Закрити" color="primary" v-close-popup />
      <q-btn
        label="ЗМІНИТИ ЦІНУ СЛОТА"
        color="primary"
        @click.prevent="onChange"
      />
    </q-card-actions>
  </q-card>
</q-dialog>
<q-dialog v-model="removeLotModalOpened">
  <q-card :style="{ width: '320px' }">
    <q-card-section>
      <q-form> Чи точно ви хочете видалити цей лот? </q-form>
    </q-card-section>

    <q-separator />

    <q-card-actions align="right">
      <q-btn label="Hi" color="primary" v-close-popup />
      <q-btn label="Так" color="primary" @click.prevent="onRemove" />
    </q-card-actions>
  </q-card>
</q-dialog>
</template>
</PageContainer>
</template>

<script>
import PageContainer from "@components/PageContainer.vue";
import { mapActions, mapGetters, mapState } from "vuex";
import AuctionItem from "../components/AuctionItem.vue";
export default {
  name: "MyLots",
  components: {
    PageContainer,
    AuctionItem,
  },

```

```

data() {
  return {
    price: null,
    priceModalOpened: false,
    removeLotModalOpened: false,
    id: null,
  };
},
computed: {
  ...mapState(["myLots"]),
},
methods: {
  ...mapActions(["getMyLots", "deleteLot", "changePrice"]),
  openPriceModal(id) {
    this.id = id;
    this.priceModalOpened = true;
  },
  openRemoveLotModal(id) {
    this.id = id;
    this.removeLotModalOpened = true;
  },
  onRemove() {
    this.deleteLot(this.id).then(() => {
      this.removeLotModalOpened = false;
      this.id = null;
    });
  },
  onChange() {
    this.changePrice({ id: this.id, price: this.price }).then(() => {
      this.priceModalOpened = false;
      this.id = null;
      this.price = null;
    });
  },
},
mounted() {
  this.getMyLots();
},
};
</script>

```

```
<style lang="less" scoped>
```

```
.title {
  padding-bottom: 10px;
  border-bottom: 1px solid #d6d5db;
}
.items-wrapper {
  display: flex;
  flex-direction: column;
  align-items: center;
}
</style>
```

A.10 Код файлу RegisterView.vue

```
<template>
  <q-card class="register-form-container">
    <div class="form-header text-h5">Форма реєстрації</div>
    <q-card-section class="register-form-section">
      <q-form @submit="onSubmit" class="register-form">
        <div class="form-data">
          <q-input v-model="email" label="Ваша пошта" />
          <q-input v-model="password" type="password" label="Пароль" />
          <q-input
            v-model="passwordVerify"
            type="password"
            label="Підтвердіть пароль"
          />
          <q-input v-model="userName" label="ПІБ" />
          <q-input v-model="telephone" label="Телефон" />
        </div>
        <q-btn
          label="Зареєструватися"
          type="submit"
          class="full-width"
          @click.prevent="onSubmit"
        />
      </q-form>
    </q-card-section>
  </q-card>
</template>
```

```

<script>
import { required, minLength, sameAs, email } from "vuelidate/lib/validators";
import useVuelidate from "@vuelidate/core";

import { mapActions } from "vuex";
export default {
  name: "Register",
  methods: {
    ...mapActions(["register"]),
    onSubmit() {
      this.v$.touch();
      if (this.v$.error) {
        return;
      } else {
        this.register({
          userName: this.userName,
          password: this.password,
          email: this.email,
          telephone: this.telephone,
        }).then(() => {
          this.$router.push({ name: "login" });
        });
      }
    },
  },
  validations: {
    userName: {
      required,
    },
    password: {
      required,
      minLength: minLength(6),
    },
    passwordVerify: {
      required,
      sameAsPassword: sameAs("password"),
    },
    email: {
      required,
      email,
    },
    telephone: {

```

```

    required,
  },
},
data() {
  return {
    password: null,
    passwordVerify: null,
    email: null,
    userName: null,
    telephone: null,
  };
},
setup: () => ({ v$: useVuelidate() }),
};
</script>

```

```

<style lang="less" scoped>
@color-gray: #858585;
@color-light-gray: #d6d5db;
.register-form-container {
  margin: auto;
  width: 480px;
  height: 720px;
  .form-header {
    text-align: center;
    padding: 12px 0;
    border-bottom: 1px solid @color-light-gray;
  }
  .register-form-section {
    padding: 20px 40px;
    height: 90%;
    .register-form {
      display: flex;
      flex-direction: column;
      justify-content: space-between;
      height: 100%;
      .q-field {
        margin: 10px 0;
      }
      .full-width {
        color: @color-gray;
      }
    }
  }
}

```

```
}  
}  
}  
</style>
```

ДОДАТОК Б
Слайди презентації



Рисунок Б.1 – Слайд №1

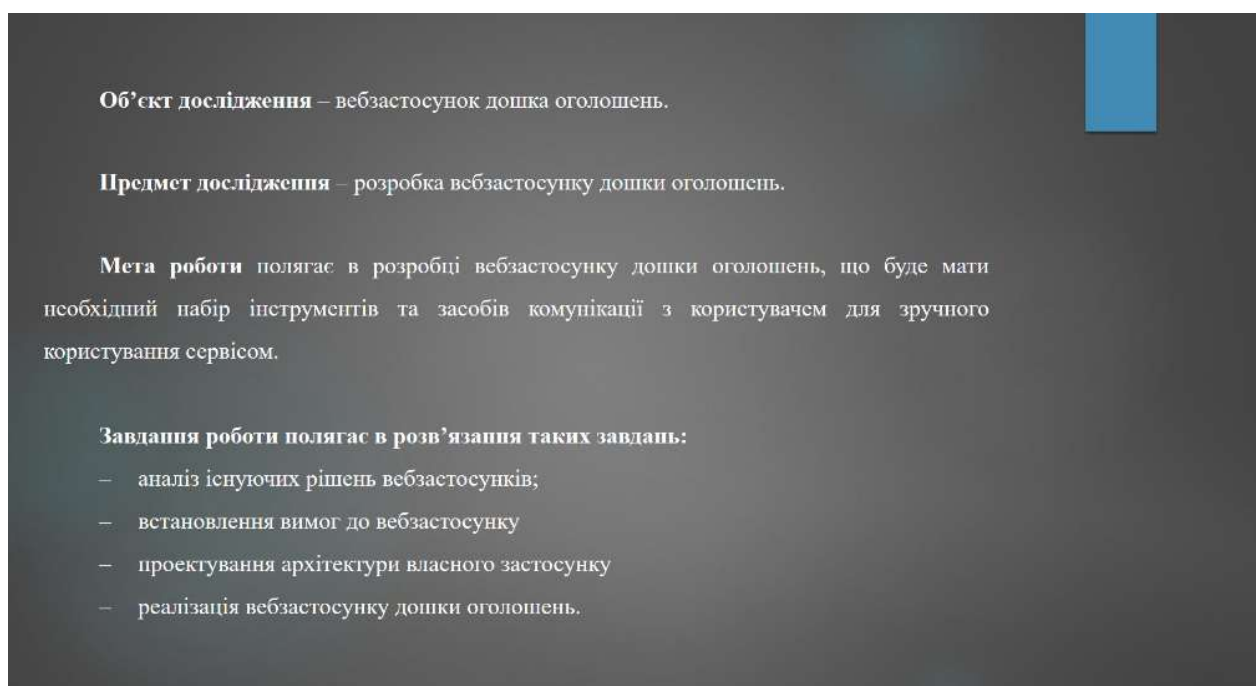


Рисунок Б.2 – Слайд №2



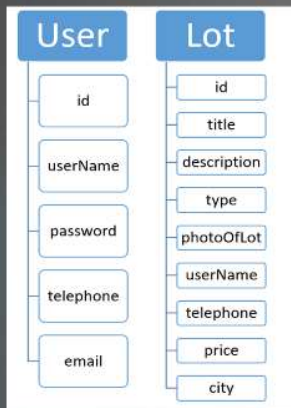
Рисунок Б.3 – Слайд №3

Обрані засоби розробки

Мови розмітки та стилів	HTML та CSS
Мова програмування	JavaScript
Фреймворк для клієнтської частини	Vue.js
Фреймворк для серверної частини	Express.js
СКБД	MySQL

Рисунок Б.4 – Слайд №4

Робота з даними



Діаграма незв'язних таблиць БД

```

1 "use strict";
2 const { Model } = require("sequelize");
3
4 class User extends Model {
5   static init(sequelize, DataTypes) {
6     return super.init({
7       {
8         username: DataTypes.STRING,
9         password: DataTypes.STRING,
10        email: {
11          type: DataTypes.STRING,
12          validate: { isEmail: true },
13        },
14        telephone: DataTypes.STRING,
15      },
16      {
17        sequelize,
18        modelName: "User",
19      },
20    });
21  }
22 }
23
24 module.exports = User;
  
```

Модель класу User

```

1 "use strict";
2 const { Model } = require("sequelize");
3
4
5
6 class Lot extends Model {
7   static init(sequelize, DataTypes) {
8     return super.init({
9       {
10        title: DataTypes.STRING,
11        description: DataTypes.STRING,
12        type: DataTypes.STRING,
13        photoOfLot: DataTypes.TEXT,
14        username: DataTypes.STRING,
15        telephone: DataTypes.STRING,
16        price: DataTypes.INTEGER,
17        city: DataTypes.INTEGER,
18      },
19      {
20        sequelize,
21        modelName: "Lot",
22      },
23    });
24  }
25 }
26
27 module.exports = Lot;
  
```

Модель класу Lot

Рисунок Б.5 – Слайд №5

Логіка роботи вебзастосунку

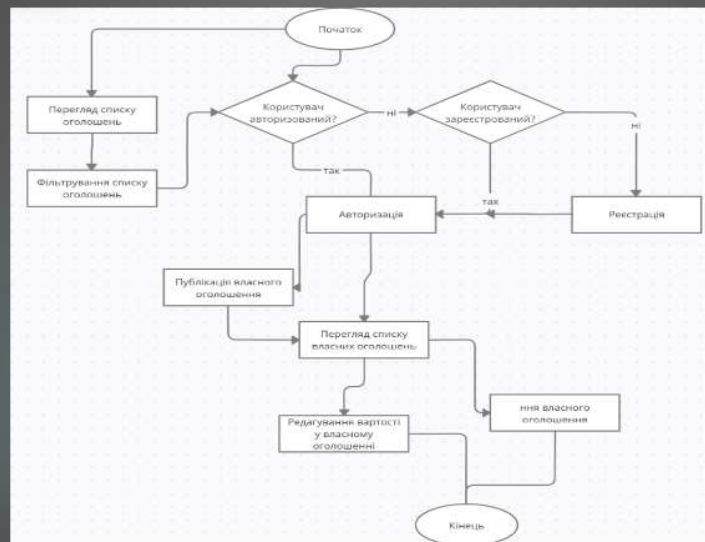


Рисунок Б.6 – Слайд №6

Структура застосунку

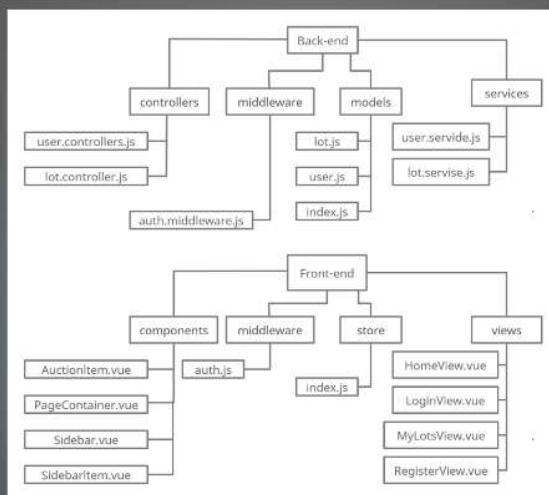


Рисунок Б.7 – Слайд №7

Інтерфейси розробленого вебзастосунку

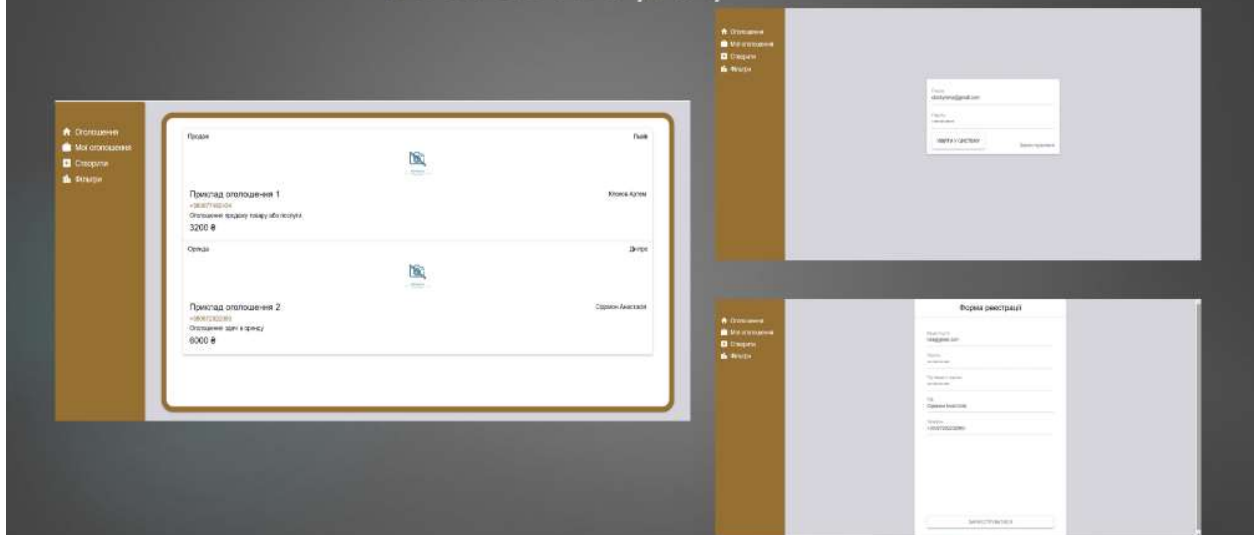


Рисунок Б.8 – Слайд №8

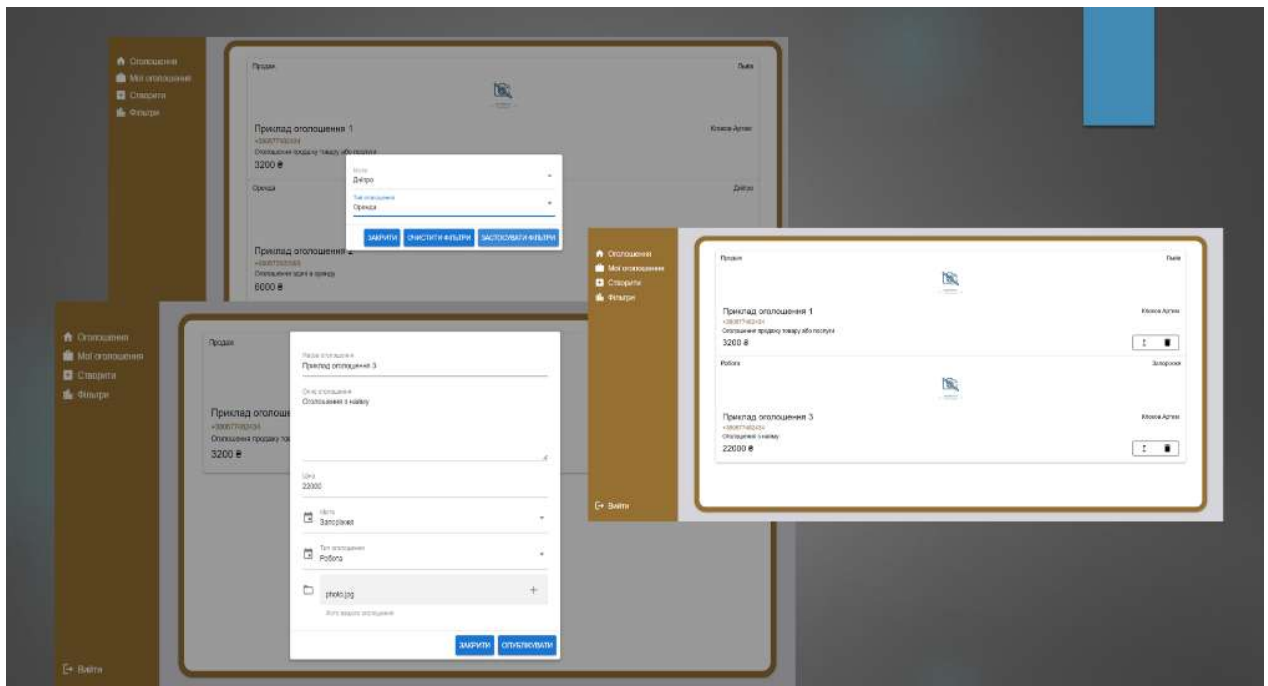


Рисунок Б.9 – Слайд №9

Висновки

В ході виконання дипломної кваліфікаційної роботи було розв'язано такі задачі:

- аналіз предметної області, шляхом дослідження специфіки роботи вебзастосунків типу дошка оголошень;
- проектування архітектури вебзастосунку, формулювання функціональних вимог, створення моделі роботи програми, обґрунтуванням засобів розробки та визначенням структури БД;
- кінцева реалізація програмного забезпечення з описом алгоритмів роботи програми;
- створення керівництва користувача з визначенням призначення застосунку, технічних вимог, описом та прикладами роботи розробленого рішення.

Розроблений вебзастосунок дошки оголошень надає користувачеві можливість здійснювати операції з купівлі/продажу, оренди, найму та ведення рекламної компанії на базі сайту. Користувач має можливість продивлятися оголошення, фільтрувати їх, створювати власні оголошення, редагувати та видаляти їх.

Рисунок Б.10 – Слайд №10