Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра мережних технологій факультету інформатики

РОЗРОБКА ВЕБ-САЙТУ ІНТЕРНЕТ-ОГОЛОШЕНЬ

Текстова частина до курсової роботи
за спеціальністю “Інженерія програмного забезпечення” 121

Керівник курсової роботи
к.ф-м.н., с.в. Гречко А. В.
_____
(підпис)
“____” _____ 2020 р.
Виконала студентка студентка 3-го курсу Побережець Б. А.
“____” _____ 2020 р.

Київ 2020

Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра інформатики факультету інформатики

ЗАТВЕРДЖУЮ

Зав. кафедри мережних технологій,

доцент, к. тех. н.

_____О. В. Франчук

“____”_____2020 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на курсову роботу

студентці _____Побережець Богдані_____

__3-го__ курсу факультету інформатики

ТЕМА: Розробка веб-сайту інтернет-оголошень

Вихідні дані:

- Проста реалізація веб-додатку інтернет-оголошень

Зміст ТЧ до курсової роботи:

Індивідуальне завдання

Вступ

1. Аналіз предметної області. Постановка завдання курсової роботи

2.Теоретинчі відомості.

3. Опис реалізації веб-сайту інтернет-оголошень.

Висновки

Список джерел

Додатки (за необхідністю)

Дата видачі “___” _____ 2020 р.

Керівник _____ Завдання отримав _____

Календарний план виконання курсової роботи:

Тема: Розробка веб-сайту інтернет-оголошень

Календарний план виконання роботи:

№ п/п	Назва етапу курсової роботи	Термін виконання етапу	Примітка
1.	Отримання завдання на курсову роботу.	12.12.2019	
2.	Огляд технічної літератури за темою роботи.	31.12.2019	
3.	Проектування схеми бази даних.	15.01.2020	
4.	Розробка бази даних.	31.01.2020	
5.	Розробка серверної частини веб-сайту.	25.03.2020	
6.	Розробка клієнтської частини веб-сайту.	25.04.2020	
7.	Написання текстової частини.	30.04.2020	
8.	Створення слайдів для доповіді та написання доповіді.	12.05.2020	
9.	Обговорення отриманих результатів з керівником.	19.05.2020	
11.	Остаточне оформлення текстової частини та слайдів.	25.05.2020	
12.	Захист курсової роботи.	02.06.2020	

Студент Побережець Б. А. _____

Керівник Гречко А. В. _____

“_____”_____ 2020 р.

Зміст

Анотація

Дану роботу присвячено розробці MVP застосування веб-сайту інтернет оголошень.

В першому розділі проводиться дослідження та аналіз предметної області. А також порівняння різних існуючих систем. Розділ два присвячено дослідженню питання вибору технічних засобів для реалізація веб-застосування. В третьому розділі показано частина розробки та результат створеного веб-застосування, а також результати проведеного тестування.

C#, ASP.NET, MVC, DATABASE, MYSQL, ENTITY FRAMEWORK

Вступ

Здійснення покупок через Інтернет-ресурси набирає популярності з кожним роком все більше і більше. До основних переваг онлайн-шопінгу відноситься економія часу, адже у мережі надається можливість придбати необхідну річ у зручний час та відшукати інформацію про товар та його характеристики на будь-якому пристрої (ноутбуці, комп'ютері або ж телефоні).

Сайти з інтернет-оголошеннями допомагають спростити пошук необхідного товару, або ж навпаки вигідно знайти покупців. А ще, різні платформи, створені спеціально для таких цілей, створюють сприятливі умови для розвитку малого та середнього бізнесу.

Метою даної роботи є створення веб-порталу для людей, які хочуть розмістити оголошення в електронному вигляді, прорекламувати свої послуги або купити необхідну річ.

Об'єктом дослідження є інші платформи та організації, які надають змогу створювати свої оголошення.

До методів дослідження відносяться:

- спостереження – пошук сервісів, які надають можливості для продажу або купівлі товарів, для реклами послуг або обміном інформації;
- порівняння – відслідковування зручності кожного знайденого сайту, порівняння статистики популярності, проведення опитування серед потенційних користувачів;
- аналіз – поділ зібраної інформації на елементи досліджуваної системи з метою створення зв'язків.

Джерелами порівняння стали такі платформи:

- топ 1 платформа в Україні "OnLine eXchange";
- дошки оголошень "OBYAVA.UA", "OGOLOSHA.UA" та "RIA.COM" .

Методом проектування стало веб-програмне забезпечення, яке створене для реалізації та редагування діаграм – lucidchart.com. Lucidchart - робочий простір, де поєднуються побудова схем, візуалізація даних і спільна робота. [1]

До складових продукту відноситься:

- серверна частина для обробки запитів:
- клієнтська (візуальна частина) для безпосередньої роботи з сайтом
- використання сховища даних (базу даних) для зберігання інформації.

У даній роботі планується використовувати об’єктно-орієнтовану парадигму програмування, яка буде інтегрована з СУБД для збереження даних.

1. Аналіз предметної області. Постановка завдання курсової роботи

1.1. Аналіз сучасного стану питання

Ще декілька десятків років назад для того, щоб здійснити купівлю-продаж товару варто було відправляти лист до редакції газети. Після цього доводилося очікувати моменту появи оголошення в газеті. Крім того, що цей процес займав досить багато часу, можна було опинитися у ситуації, коли на момент публікації оголошення газеті, воно втрачало свою актуальність.

З розвитком мережі Інтернет для таких цілей почали створювати віртуальні дошки оголошень. **Дошка оголошень** – це сайт рекламної спрямованості, призначенням якого є реклама товарів та послуг. Перевага сайту дошки оголошень полягає в тому, що будь-який бажаючий може розмістити своє оголошення або підібрати підходящу пропозицію для себе. [2]

Взагалі кажучи, не зважаючи на широкий вибір дошок оголошень, у розробників досі існує можливість покращувати веб-застосування такого плану, щоб не лише йти в ногу з сучасністю, але й розширювати функціонал відповідно до потреб користувачів.

1.2. Огляд існуючих аналогів розробки

В якості аналогів створюваної системи було обрано дошку безкоштовних оголошень Оголоша та сервіс онлайн-оголошень OLX.

Оголоша – це спеціально створений майданчик для користувачів, які бажають:

- продати, купити або обмінятись товарами;
- запропонувати послуги або скористатись послугами;
- знайти роботу та співробітників [3].

Перейшовши на сайт можна помітити, що на ньому передбачено 14 розділів, де можна розмістити своє оголошення. На рис. 1 наведено інтерфейс головної сторінки сайту Ogolosha.ua.

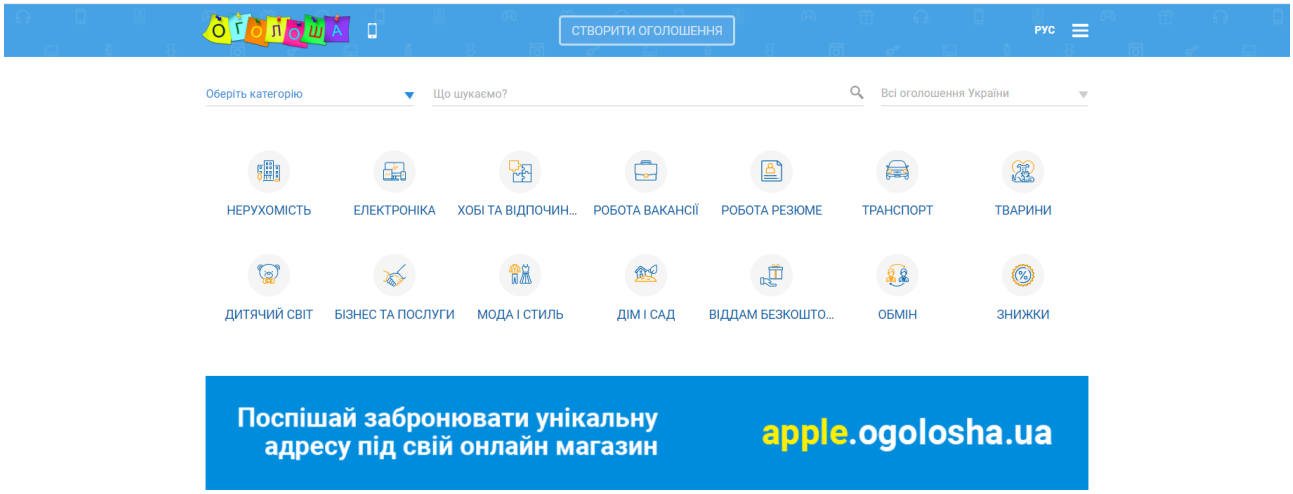


Рисунок 1 – Головна сторінка сайту Оголоша

Платформа OLX – порівняно молода компанія, яку було засновано у 2006 році. Сайт надає платформу для онлайн купівлі/продажу товарів і послуг як приватним особам, так і представникам бізнесу [4].

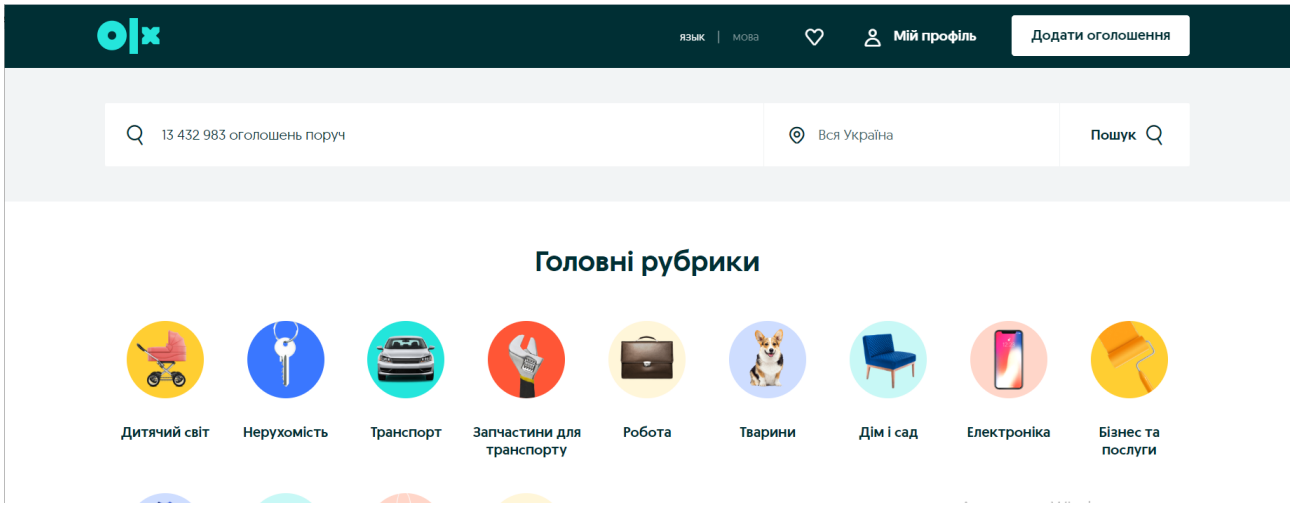


Рисунок 2 – Головна сторінка сайту OLX

Виділимо основні критерії, за якими буде проводитись порівняння існуючих систем аналогів. До базових вимог належать:

1) Наявність основного меню, що доступне на кожній сторінці сайту;
2) Зручний та не нагромаджений інтерфейс;
3) Доступність вибору мови;
4) Простота використання веб-порталу;
5) Ефективний пошук та фільтрація даних;

Для зручності порівняння, дані про наявність або відсутність вимог відображено у вигляді табл. 1. Варто зазначити, що результати оцінювання є виключно суб'єктивними та можуть різнитись в залежності від суб'єкту оцінювання.

Таблиця 1 – Результат порівняння систем-аналогів

Критерій	Оголоша	OLX
Наявність основного меню	+	+
Зручний та не нагромаджений інтерфейс	+	+/-
Доступність вибору мови	+/-	+
Простота використання веб-порталу	+	+
Ефективний пошук та фільтрація даних	+	-

1.3. Постановка завдання

Розглянемо основні функціональні вимоги веб-застосунку:

- Портал має давати змогу додавання, редагування та видалення одиниці товару;
- портал має забезпечувати пошук та сортування результатів пошуку об’єктів;
- порталу потрібно створити зручний інтерфейс;
- портал має забезпечувати розмежування прав доступу до своєї функціональності.

На діаграмі на рис. 3 наведено варіанти використання веб-застосунку для користувача, що не зареєстрований в системі.

До можливих дій відноситься:

- перегляд списку всіх можливих товарів;
- пошук оголошення, перегляд одиниці товару;
- доступ до контактних даних власника товару;
- вхід в портал під зареєстрованими даними;
- реєстрація нового користувача.

Рисунок 3 – Можливості не зареєстрованого користувача

Діаграма на рис. 4 показує можливості користувача, що зареєструвався на порталі. Додатково надаються наступні можливості:

- увійти до меню налаштувань, редагування персональних даних;
- перегляд особистих оголошень;
- видалення\редагування оголошення, створення нового оголошення;

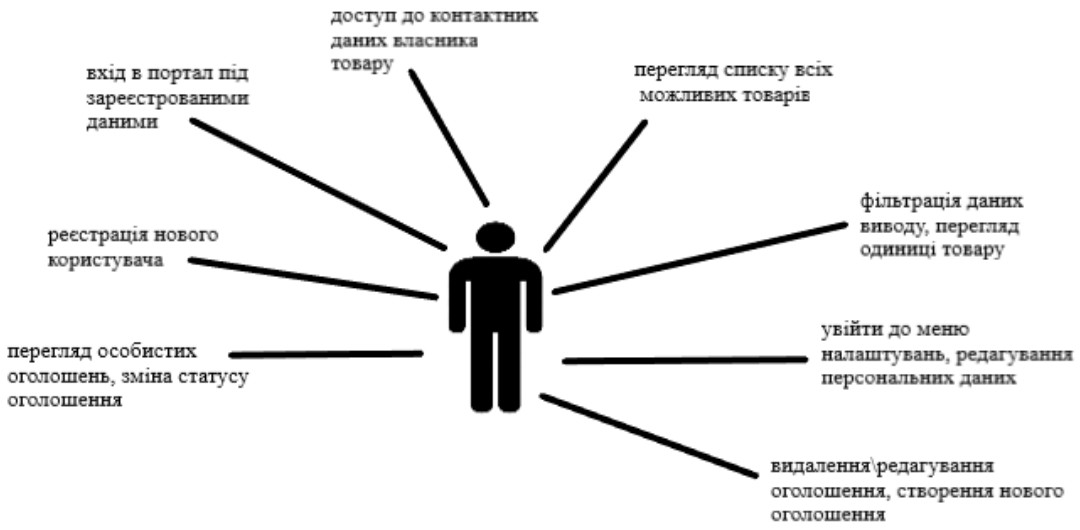


Рисунок 4 - Можливості зареєстрованого користувача

2. Теоретичні відомості

2.1. Вибір технічних засобів побудови системи

Під час аналізу предметної області було вирішено, що систему буде створено як веб-додаток. Саме тому під час вибору технологій розробки було зосереджено увагу на мови, що надають таку можливість. Серед об’єктів аналізу були такі мови як – Java, Python, C#, PHP.

Перший критерій, за яким оцінювався вибір мови був можливість розробки веб-додатку на різних операційних системах. На щастя, більшість мов програмування надають можливість кросплатформеності, наприклад Java, Python, PHP. Тим не менш існують в такі, що підтримуються лише однією ОС – C#.

Java – об’єктно- орієнтована мова програмування компанії Oracle. В офіційній реалізації Java - програми компілюються у байт-код, який при виконанні інтерпретується віртуальною машиною для конкретної платформи[5]. Основними перевагами Java є надійність, вбудований збирач сміття, велика кількість вбудованих бібліотек, що можуть покрити виконання різних задач.

PHP – лідер серед мов програмування для серверних сценаріїв. До переваг PHP відноситься легкий синтаксис (легкість вивчення), підтримка баз даних, висока швидкодія. Проте існують критично важливі недоліки, такі як умовна типізація та застаріла парадигма ООП.

Python - інтерпретована об'єктно-орієнтована мова програмування високого рівня зі строгою динамічною типізацією [6]. Python характеризуються простотою та гнучкістю мови. Наявність великої кількості модулів забезпечує різні розширені можливості. Недоліками Python є невисока швидкість виконання програми, відсутність статичної типізації та відсутня можливість модифікації вбудованих класів.

Для операційної системи Windows однією з кращих технологій є ASP.NET та мова програмування C#. Мова відмінно підходить для швидкого написання настільних додатків з зручним інтерфейсом. Крім того, він відноситься до однієї з мов технології ASP.NET для розробки веб-додатків.[7]. C# забезпечує строгу статичну типізацію, перевантаження операторів та підтримку поліморфізму.

Прийнявши до уваги деталі кожної з мов, було обрано C#, як універсальну об’єктно-орієнтовану мову програмування для загального призначення.

2.2. Архітектура системи

Як основу взято технологію ASP.NET Core. Платформа ASP.NET Core представляє технологію від компанії Microsoft, призначену для створення різного роду веб-додатків: від невеликих веб-сайтів до великих веб-порталів і веб-сервісів.[8]

Не зважаючи на те, що основним розширенням ASP.NET Core є використання шаблону MVC (Model-View-Controller), основною технологією було обрано Razor Pages – альтернативу системі MVC. Razor Pages дають змогу створювати сторінки з кодом Razor, що можуть оброблювати запити.

ASP.NET Core Razor Pages – це орієнтований на сторінки фреймворк для створення динамічних веб-сайтів з чітким розділенням проблем. Razor Pages досить легкий і надзвичайно гнучкий у плані використання, адже розробнику надається повний контроль над даними HTML. Фреймворк побудовано поверх стандарту ASP.NET Core MVC, який рекомендується використовувати на стороні сервера

Кожна сторінка Razor – це файл з розширенням .chtml, тобто суміш html та C#. Взагалі кажучи, це таке ж представлення як MVC, але на відміну від MVC, до кожної сторінки Razor прив’язано деякий файл логіки C# з розширенням .chtml.cs. У файлі логіки буде відбуватись взаємодія з базою даних, обробка запитів та вирішуватись подальша поведінка застосунку на певні дії користувача. На рис. 5 зображено архітектуру системи, що буде розроблюватись.

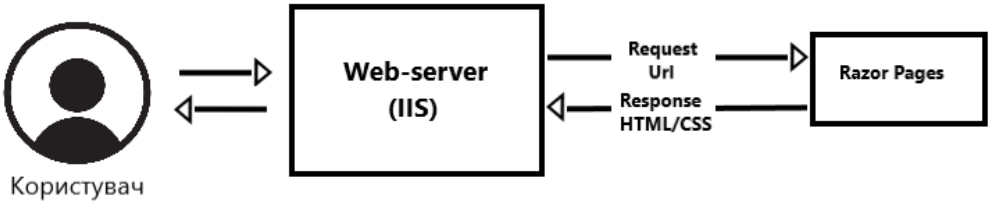


Рисунок 5 Архітектура системи

2.3. Вибір СУБД

База даних (БД) — це організована структура, яка призначена для зберігання, зміни та обробки взаємозалежної інформації, переважно великих обсягів[9]. В найпоширеніших випадках, бази даних мають схеми, таблиці подання, збережені процедури та деякі об’єкти. Сучасні бази даних містять засоби та опис для обробки даних. Мета бази даних полягає у задоволені інформаційних потреб користувачів.

Система керування базами даних (СКБД) – це програмне забезпечення, яке надає можливість контрольованого доступу до даних, створювати, підтримувати та визначати базу даних.

Основні вимоги та функції для СКБД:

- Можливість створення бази даних;
- додавання\редагування\видалення даних;
- підтримка системи забезпечення захисту та відновлення при доступі до бази даних;
- забезпечення мінімальної надлишковості та несуперечливості даних;
- можливість колективного використання даних;
- підвищений рівень безпеки;
- незалежність прикладних програм від даних.[10]

Об’єктами аналізу СКБД стали MySQL, PostgreSQL, та MSSQL. Перелічені програмні забезпечення працюють відповідно до архітектури “клієнт-сервер”. Тобто оператори, що використовують дані надсилають запити серверу. Після цього, відбувається опрацювання отриманих запитів та відправка відповіді клієнту.

Обрана СКБД має бути побудована на основі реляційної моделі даних. Тобто, база даних має містити реляції (таблиці), дані з яких будуть легкодоступними, а сама база даних спроектована на принципі нормалізованих відношень різного ступеня.

Модель роботи полягатиме у певних типах зв’язку між сутностями (таблицями). Кожна реляція матиме набір власних атрибутів, один з яких буде однозначно ідентифікувати рядок кожної таблиці. Кожен атрибут отримає обов’язковість (NOT NULL) або не обов’язковість (NULL) значення комірки.

Між таблицями існуватимуть зв’язки, тобто набір асоціацій між сутностями різних типів, що дасть змогу отримувати повну унікальну інформацію за певним запитом.

Аналіз переваг та недоліків об’єктів СКБД можна переглянути у табл. 2.

Таблиця 2 – Результат порівняння об’єктів СКБД

Критерій	MySQL	PostgreSQL	MSSQL
Легкість встановлення	-	+	+
Підвищений рівень безпеки	+	+	-
Використання файлової структури	-	+	+
Простота в роботі	+	+	+
Масштабованість	+	+	+
Швидкість	+	-	+

MySQL – повноцінна, вільно доступна СКБД, яка забезпечує необхідним функціоналом, швидко працює при використанні первинного ключа та має надвисоку продуктивність, тому її і було обрано для реалізації завдання.

Отже в результаті дослідження інформації про різні засоби розробки та СКБД було прийнято рішення реалізувати веб-застосування на об’єктно-орієнтованій мові програмування C# з використанням технології Razor Pages, а системою керування базами даних MySQL.

3. Опис реалізації програмного продукту

3.1. Результат реалізації бази даних

3.1.1. Схема ER – моделі

ER-модель - це семантична модель даних, яка призначена для спрощення процесу проектування бази даних [11]. Було поставлене завдання створити таку модель, яку можна було б легко розширити у майбутньому та яка б задовольняла вимоги основного завдання веб-застосування.

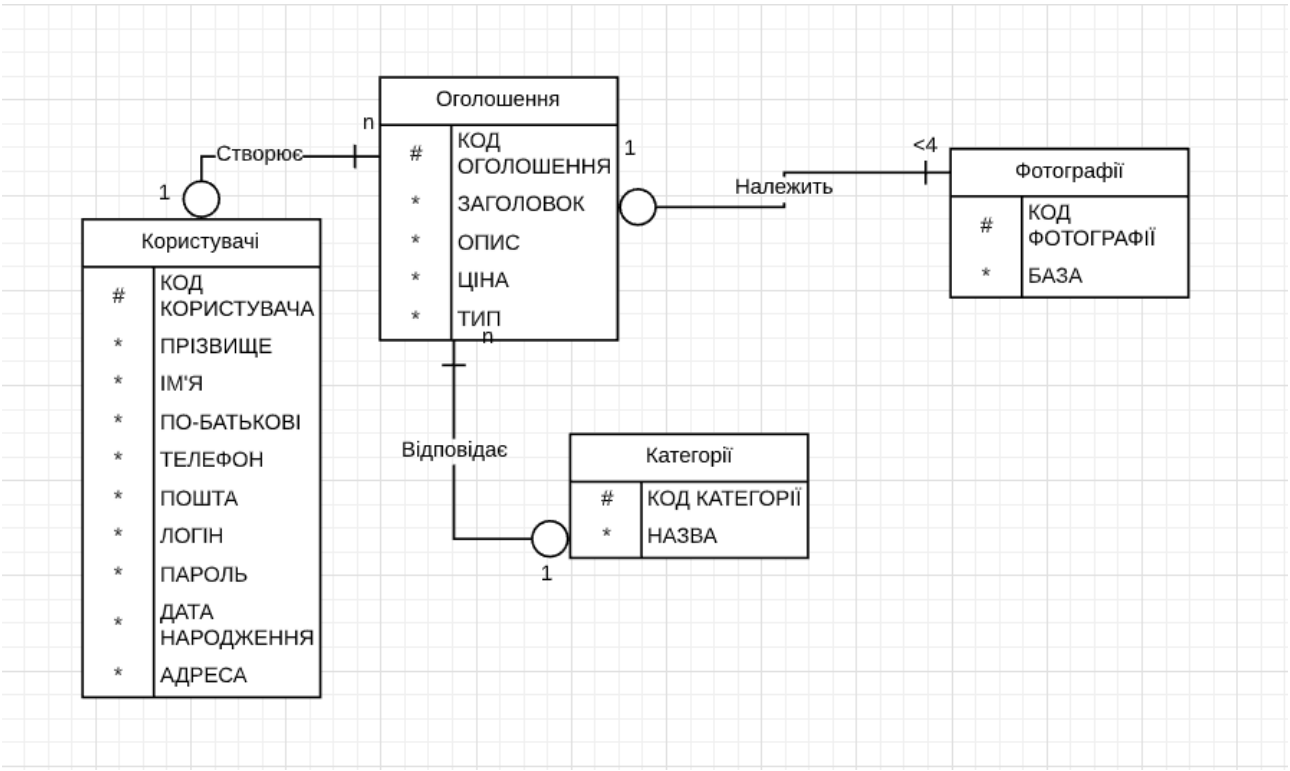


Рисунок 6 - ER-модель бази даних

3.1.2. Опис сутностей

Сутність – це клас однотипних об'єктів, інформація про яких повинна бути врахована в моделі. [12]. Значення сутностей ER-моделі показано у табл. 3.

Таблиця 3 – Результат порівняння об’єктів СКБД

Сутність	Значення
Користувачі	Людина, яка моє особистий профіль у веб-застосування

Оголошення	Оголошення, яке буде доступне для перегляду
Категорія	Категорія, до якої може належати оголошення
Фотографії	Фотографії, що можуть бути в оголошені

Первинний ключ – це одне або декілька полів (стовпців), комбінація значень яких однозначно визначає кожен запис в таблиці [13]. Значення первинного ключа для сутностей можна переглянути у табл. 4.

Таблиця 4. Таблиця ключів типів сутностей

Сутність	Первинний ключ
Користувачі	Код користувача
Оголошення	Код оголошення
Категорія	Код категорії
Фотографії	Код фотографії

3.1.3. Опис зв’язків

Зв’язок – це деяке відношення між двома типами сутностей[15]. Нижче описано зв’язки між створеними сутностями ER-моделі.

Опис зв’язку “Користувачі”- “Оголошення”:

- Мнемонічне подання зі стислим визначенням:
  | Користувачі |-o-1------n-|-| Оголошення|;
- Потужність:
  1 до багатьох, бо 1 користувач може мати декілька оголошень, але певне оголошення може належати лише одному користувачу.

Опис зв’язку “Категорії”- “Оголошення”:

- Мнемонічне подання зі стислим визначенням:

| Категорії |-o-1------n-|-| Оголошення|;

- Потужність:

  1 до багатьох, бо 1 оголошення може належати лише до однієї категорії, але в категорії може бути безліч оголошень.

Опис зв’язку “Оголошення”- “Фотографії”:

- Мнемонічне подання зі стислим визначенням:

  | Оголошення |-o-1------4-|-| Фотографії|;

- Потужність:

  1 до багатьох (4), бо 1 оголошення може мати до 4 фото, але певне фото належить лише одній фотографії.

3.1.4. Перелік корпоративних обмежень цілісності

- При введені дати народження, ця дата не повинна бути у майбутньому часі.
- Вартість зазначена в оголошені є цілочисельною, тобто копійок немає.

3.1.5. Реляційна модель даних

Реляційна база даних – це тіло зв'язаної інформації, що зберігається в двомірних таблицях [14]. У табл. 5 – 8 показано перехід від ER-моделі до реляційної. Саме схеми з цих таблиць взято за основу під час реалізації веб-застосування.

Таблиця 5. Users відповідно до типу сутності Користувачі з ER.

Users				
Ключ	Ім’я атрибута	Тип	NULL	Пояснення
PK	UserId	Char(36)	NO	Код користувача
	FirstName	Longtext	NO	Прізвище
	LastName	Longtext	NO	Ім’я
	Patronymic	Longtext	NO	По-батькові
	Email	Longtext	NO	Емейл
	Username	Longtext	NO	Логін

	Password	Longtext	NO	Пароль
	Phone	Longtext	NO	Телефон
	DateOfBirth	Longtext	NO	Дата народження
	Address	Longtext	NO	Адреса

Таблиця 6. Categories відповідно до типу сутності Категорії з ER.

Categories				
Ключ	Ім’я атрибута	Тип	NULL	Пояснення
PK	CategoryId	Char (36)	NO	Код категорії
	CategoryName	Longtext	NO	Назва категорії

Таблиця 7. Items відповідно до типу сутності Оголошення з ER.

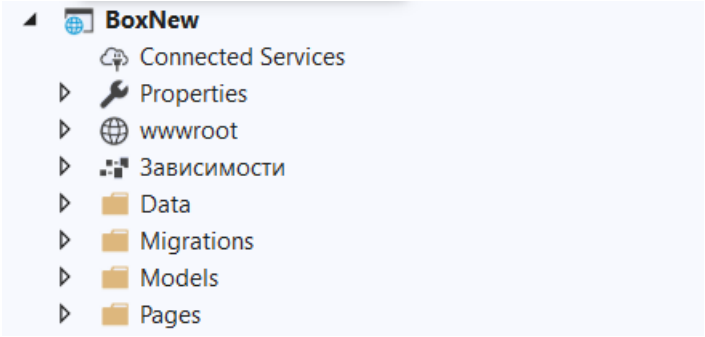
Items				
Ключ	Ім’я атрибута	Тип	NULL	Пояснення
PK	ItemId	Char (36)	NO	Код оголошення
	Headline	Longtext	NO	Заголовок оголошення
	Description	Longtext	NO	Опис
	MyType	Tinyint (1)	NO	Тип оголошення (1 або 0)
	Price	Integer	NO	Ціна оголошення
FK	CategoryId	Char (36)	NO	Код категорії

FK	UserId	Char (36)	NO	Код користувача

Таблиця 8. Photos відповідно до типу сутності Категорії з ER.

Photos				
Ключ	Ім’я атрибута	Тип	NULL	Пояснення
PK	PhotoId	Char (36)	NO	Код фотографії
FK	ItemId	Longtext	NO	Код оголошення
	Base	Longtext	NO	Назва фотографії

3.2. Опис розробки програми



В Data знаходяться класи, які відповідають за зв’язок між базою даних та Razor Pages. А також, тут спроектовано логіку пагінації сторінок.

В Models зібрано всі моделі. Саме в цій папці знаходиться класи-сутності спроектовані для співпраці з базою даних.

В Pages реалізація відображення інтерфейсу веб-застосування та опис логіки сайту на дії користувача.

3.3. Опис файлів даних та інтерфейсу програми

При відкриті веб-застосування надається можливість переглянути список всіх оголошень, зареєстрованих в системі, інтерфейс якого показано на рис. 7.

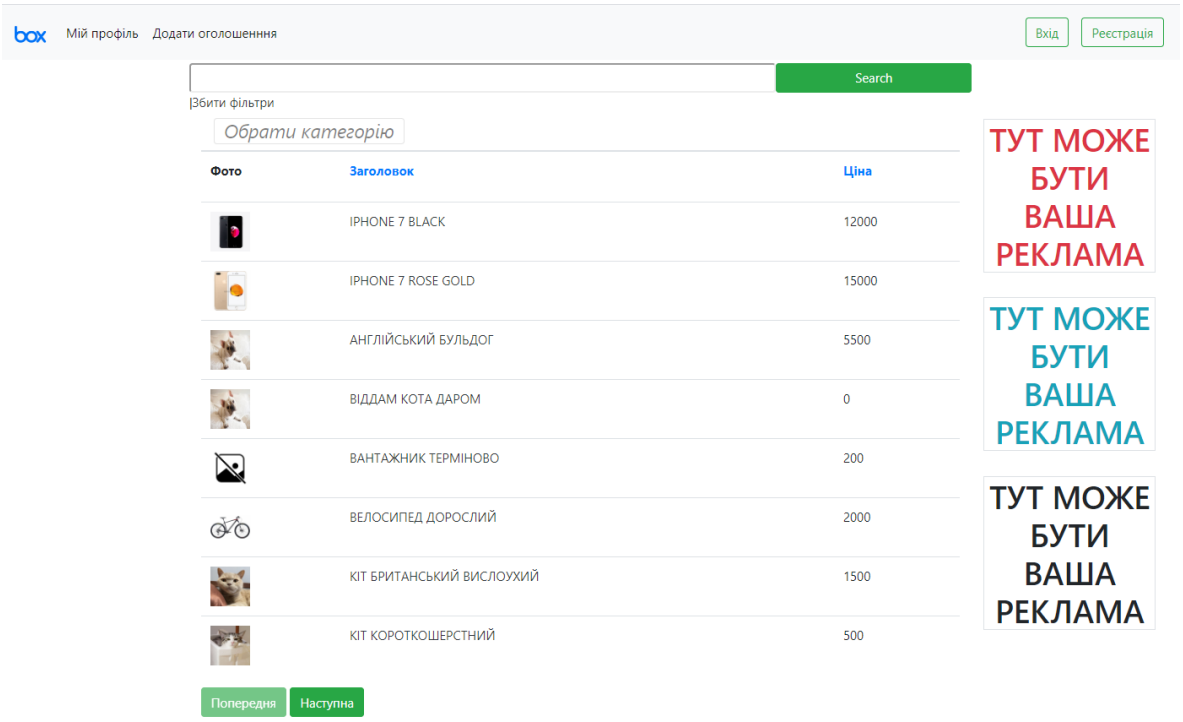


Рисунок 7 - основне вікно веб-застосування

На цьому етапі можна відсортувати оголошення за ціною або заголовком. А також, обрати категорію, по якій будуть виводитись оголошення.

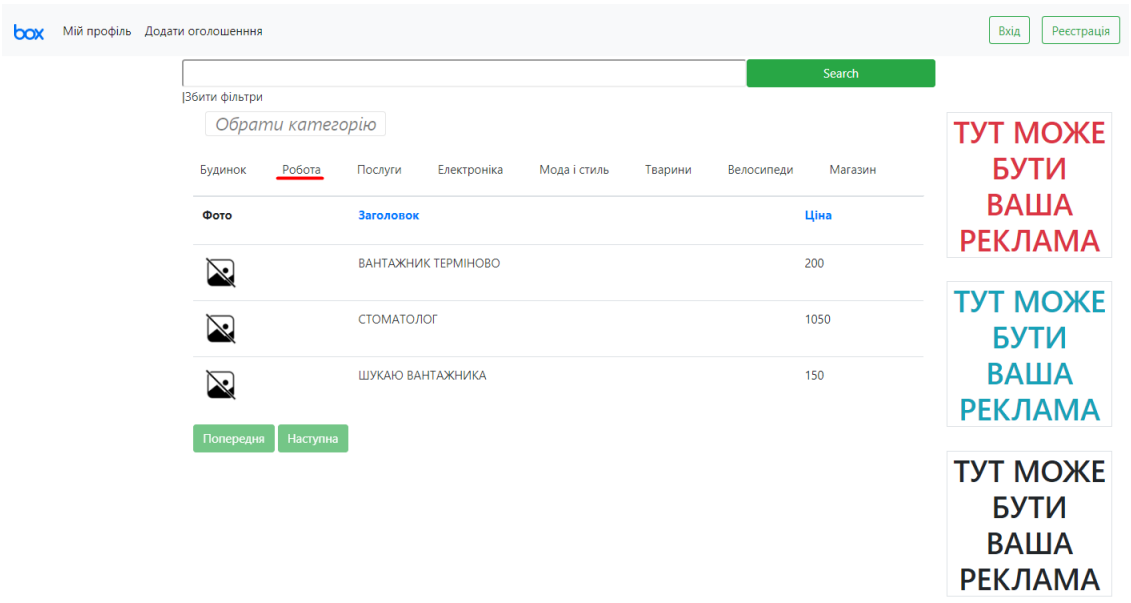


Рисунок 8 - оголошення категорії “Робота” відсортовані за заголовком

Деталі по оголошенню можна переглянути клікнувши на нього, як показано на рис. 9.

Рисунок 9 - перегляд інформації по оголошенню

Для того, щоб мати можливість додати оголошення, користувач має бути зареєстрованим у системі. Саме тому, передбачено 2 варіанти авторизації у систему.

Якщо користувач має обліковий запис, варто натиснути на кнопку “Вхід”. Тут, при коректному введені облікових даних, система дозволить увійти, якщо ні – буде відповідне повідомлення про не коректні дані.

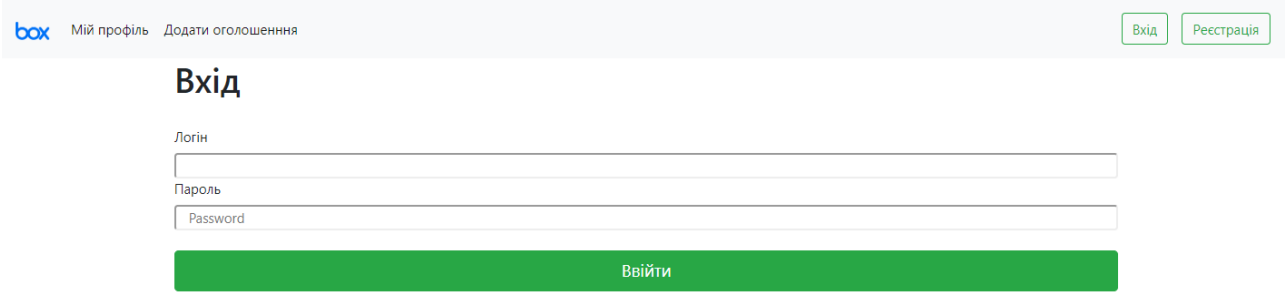


Рисунок 10 - вікно входу

У випадку, коли у користувача не створено облікового запису, надається можливість зареєструватись в системі.

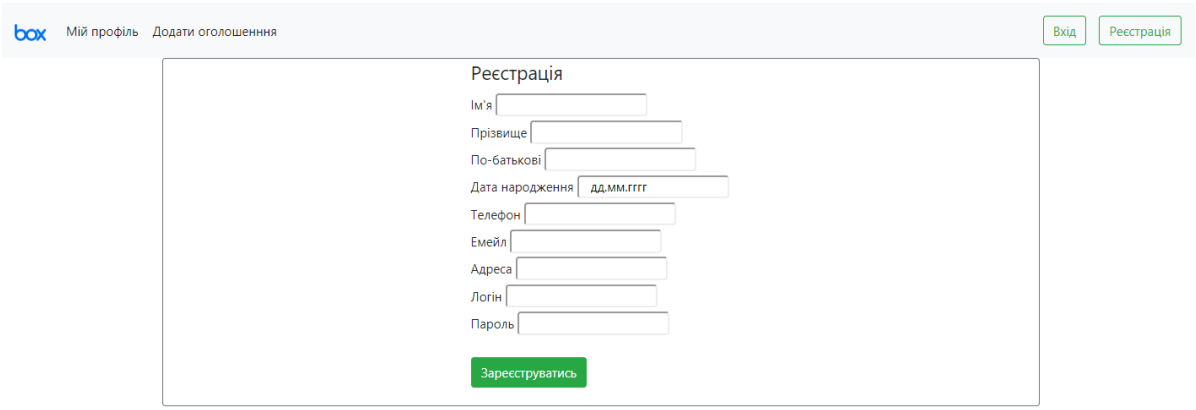


Рисунок 11 - вікно реєстрації

Після успішної авторизації надаються додаткові можливості. Якщо натиснути на вкладку “Мій профіль” - користувач потрапить до налаштувань особистого профілю. Тут можна переглянути та редагувати власні оголошення або змінити облікові дані. На рис. 12-13 показано інтерфейс вікон, що видно лише тому користувачу, що пройшов авторизацію.

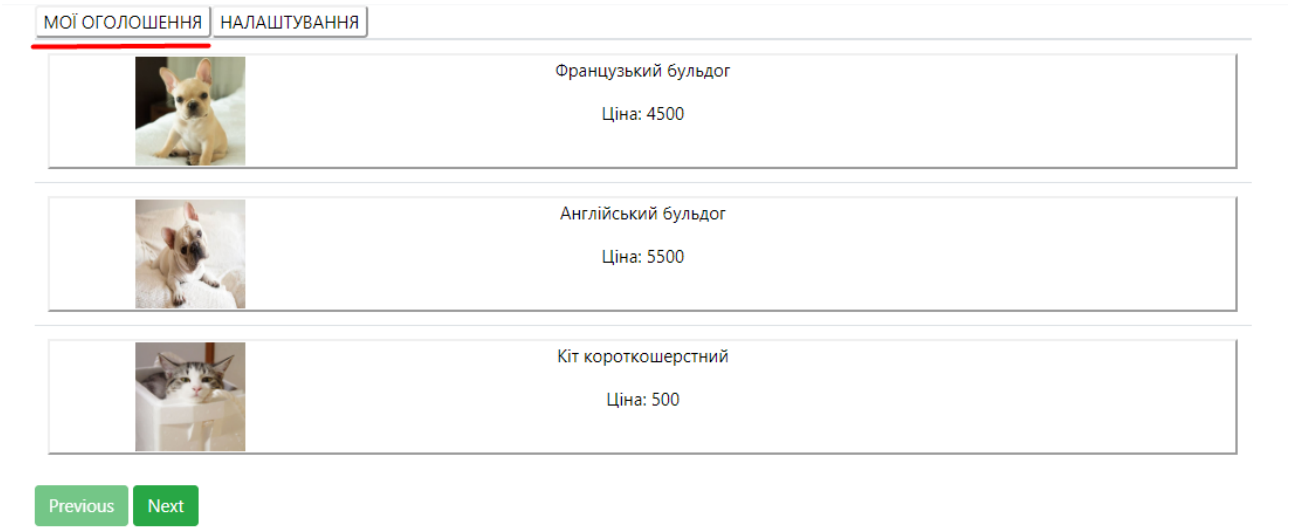


Рисунок 12 - вікно особистих оголошень

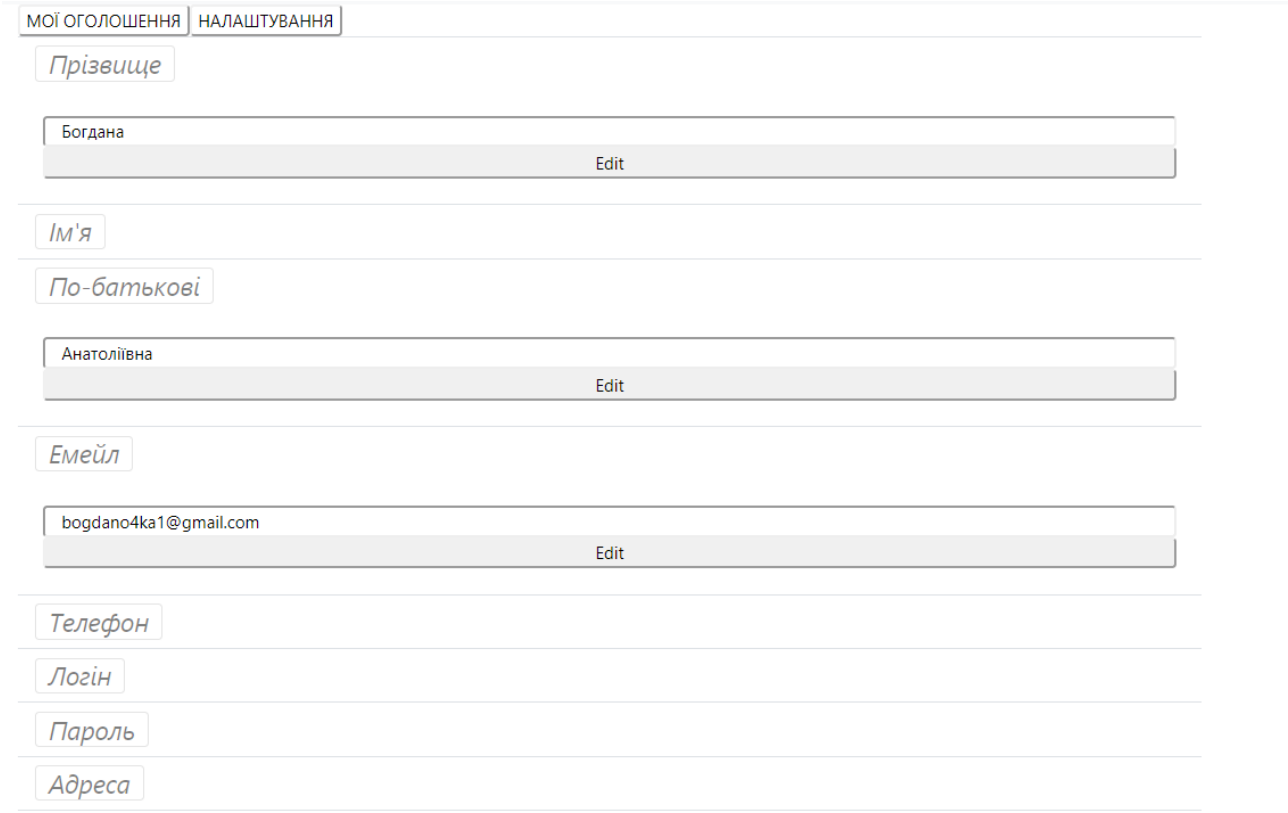


Рисунок 13 - вікно налаштувань

На рис. 14 показано інтерфейс вікна, якщо користувач вибрав оголошення, яке необхідно змінити або видалити.

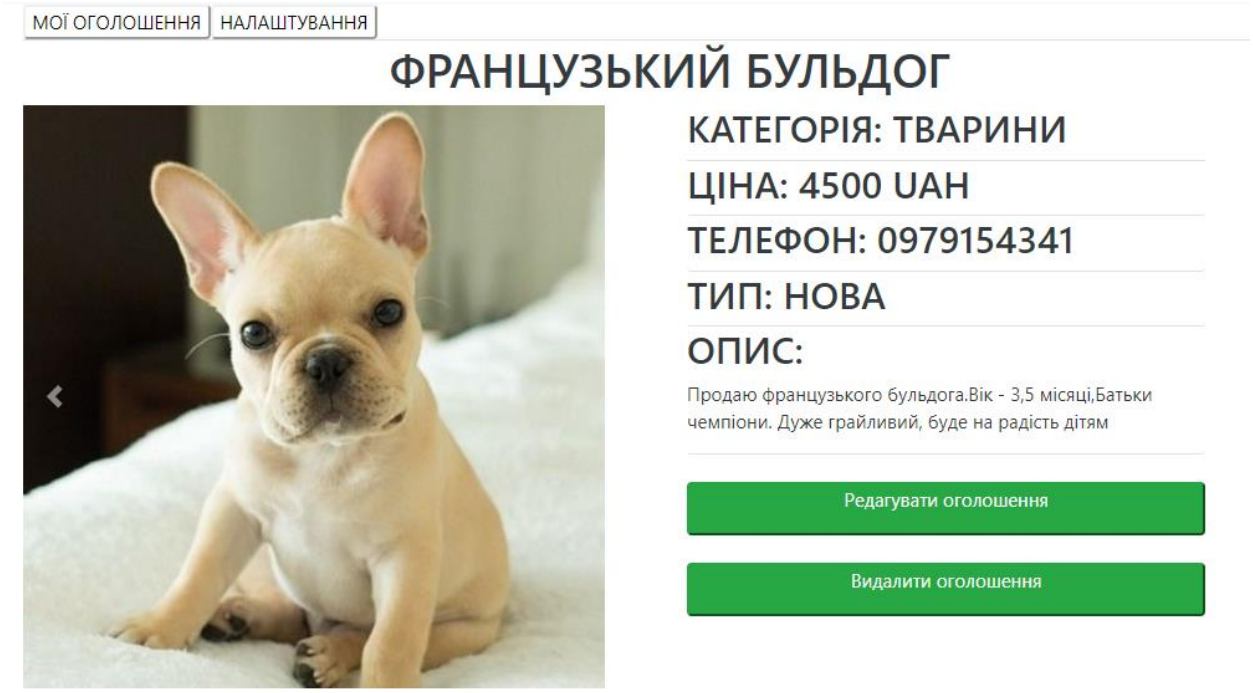


Рисунок 14 - редагування інформації про оголошення

Інтерфейс гілки “Налаштування” показано на рис. 15.

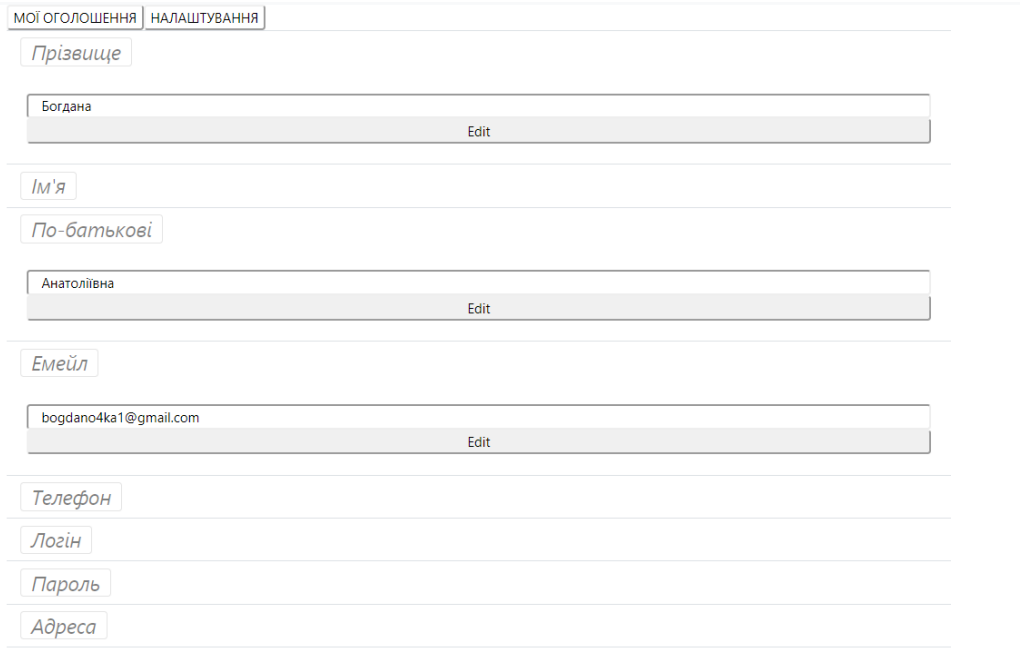


Рисунок 15 - вікно редагування особистих даних

3.4. Тестування програми і результати її виконання

Для того, щоб протестувати веб-застосування пропонується створити нового користувача та додати йому оголошення. На рис. 16 показано, що користувач не створить обліковий запис, поки на введе усі поля.

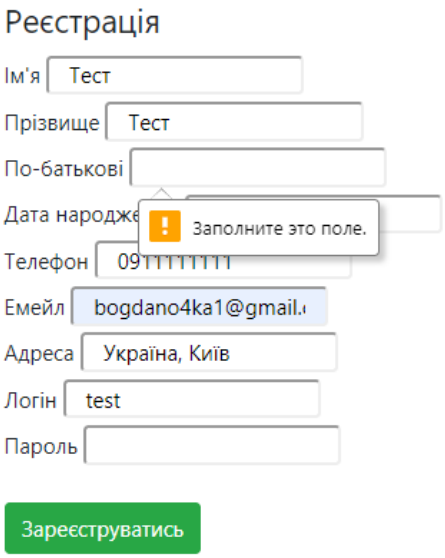


Рисунок 16 - деталі етапу реєстрації

Як ми бачимо, у вікні “Мої оголошення” є відповідне повідомлення про те, що у нього поки немає оголошень. Це можна побачити на рис. 17.

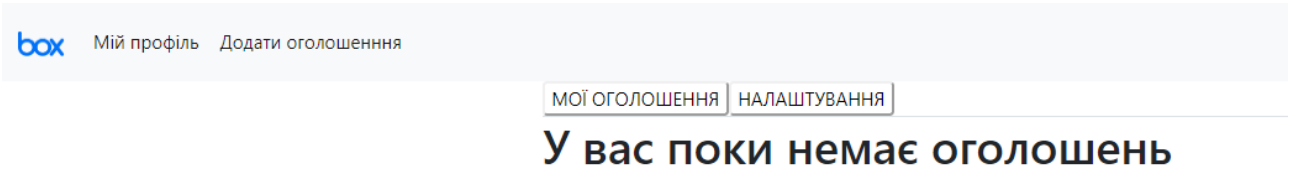


Рисунок 17 - інформація про відсутність оголошень

Далі, натиснувши на опцію “Додати оголошення”, ми потрапимо у вікно, яке дасть змогу створити нове оголошення.

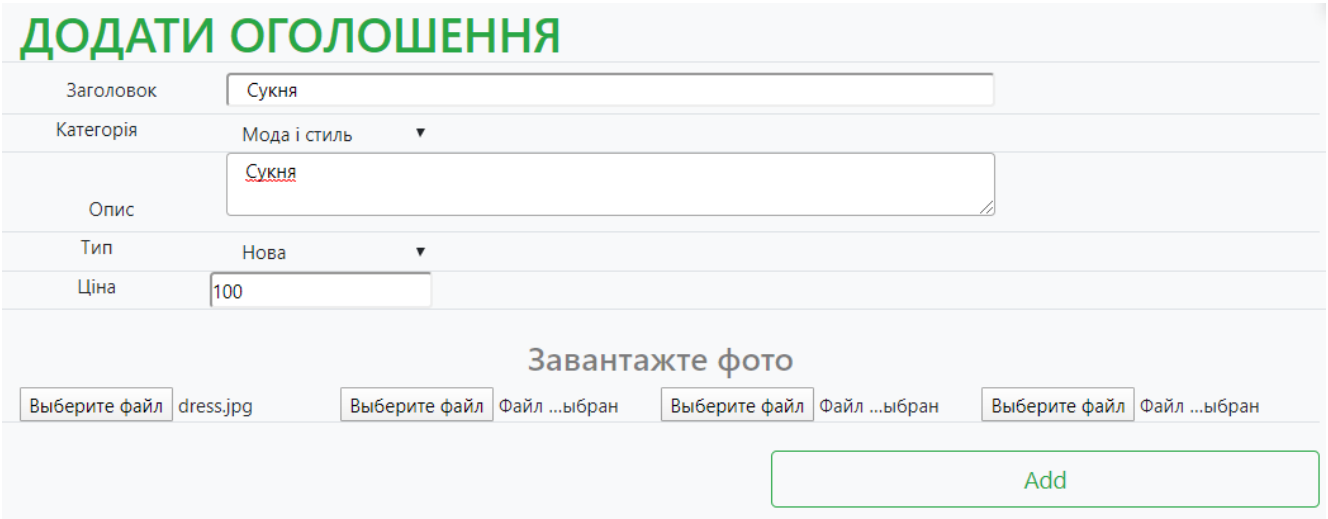


Рисунок 18 - етап створення оголошення

Заповнивши всю необхідну інформацію, у вікні “Мої оголошення”, буде відображена відповідна інформація про створене оголошення.

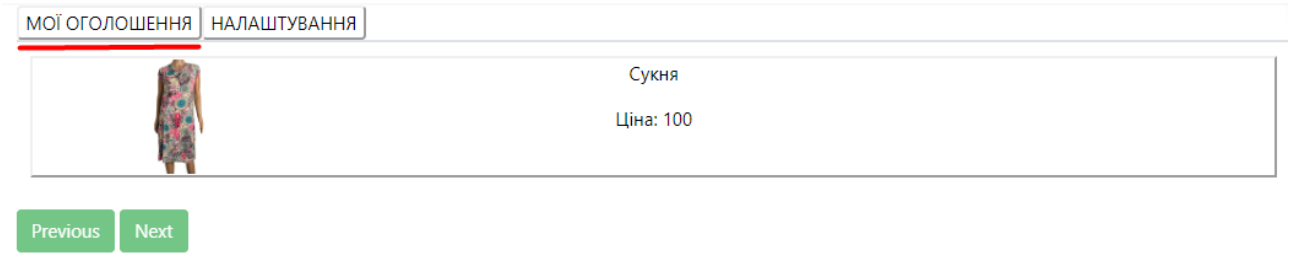


Рисунок 19 - інформація про нове оголошення

Давайте спробуємо відредагувати інформацію про дане оголошення. У вікні редагування оголошення ми змінили опис з “Сукня” на те, що показано на рис 20.

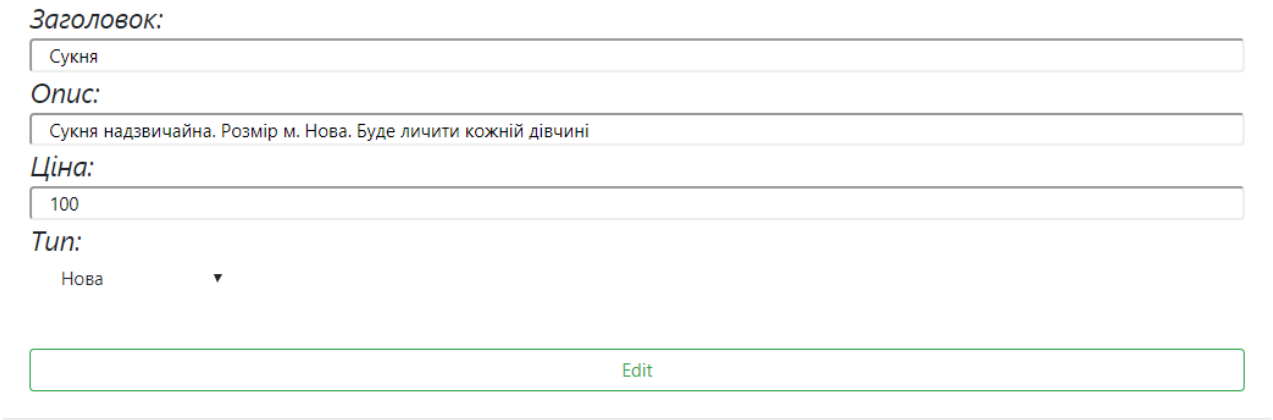


Рисунок 20 - редагування оголошення

Так, після збереження змін, на рис. 21 можна побачити, що оголошення було відредаговано.

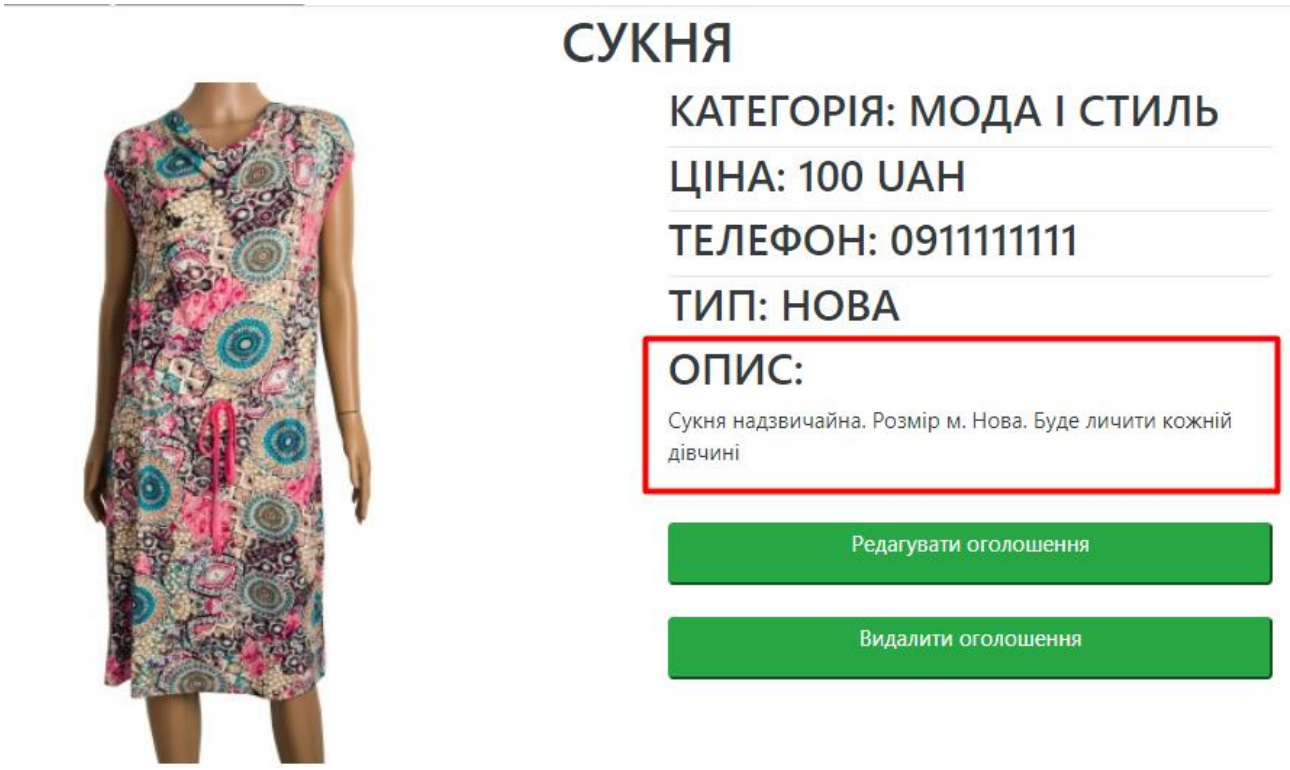


Рисунок 21 - відредаговане оголошення

Якщо ми натиснемо на кнопку видалити оголошення, воно зникне зі списку оголошень користувача і у вікні “Мої оголошення” не буде більше запису про дане оголошення.

Висновки

В результаті дослідження та розробки веб-сайту інтернет оголошень можна сформувати такі висновки:

1) Веб-застосування найкраще використовувати як прототип у вигляді мінімально життєздатного продукту.
2) У веб-застосуванні присутнє розмежування за правами доступу в залежності від того, чи авторизований користувач.
3) Веб-сайт можна покращити наступним чином:

- Для користувача додати можливість зберігання історії про колись створені оголошення.
- Розробка додаткових підсистем: чат для спілкування, інформація про інші оголошення користувача.
- Розширений пошук та фільтрація оголошень.

4) Застосування було створено мовою C#, використовуючи середовище Visual Studio 2019. Використано технології .NET Framework, Entity Framework та Bootstrap. А також MySQL Server в якості СУБД.

Список джерел

1. Lucidchart: https://www.lucidchart.com/
2. Створення сайту-дошки об’яв: http://ua.kulikoff.com.ua/desc_ob/
3. Дошка безкоштовних оголошень Оголоша: https://ogolosha.ua/uk/
4. OLX – вікіпедія: https://uk.wikipedia.org/wiki/OLX
5. Java - вікіпедія: https://uk.wikipedia.org/wiki/Java
6. Python – вікіпедія: https://uk.wikipedia.org/wiki/Python
7. C# для початківців: http://programer.in.ua/index.php/prohramuvannia/prohramuvannia-na-c/16-c-dlia-pochatkivtsiv-vstup
8. Введення в ASP.NET Core : https://metanit.com/sharp/aspnet5/1.1.php
9. Що таке база даних: http://apeps.kpi.ua/shco-take-basa-danykh
10. Матеріали з лекцій курсу “Реляційні бази даних” – основні теоретичні відомості: https://distedu.ukma.edu.ua/course/view.php?id=50
11. Поняття ER - моделі: https://www.bestprog.net/uk/2019/01/24/the-concept-of-er-model-the-concept-of-essence-and-communication-attributes-attribute-types-ua/
12. Модель Сутність – Зв’язок : http://easy-code.com.ua/2012/09/elementi-modeli-sutnist-zvyazok-integraciya-dodatkiv-i-danix-bazi-danix-statti/
13. Види логічного зв’язку: https://studopedia.info/1-46532.html
14. Що таке реляційна база даних: http://easy-code.com.ua/2010/10/shho-take-relyacijna-baza-danix/
15. Поняття зв’язку: https://www.bestprog.net/uk/2019/01/27/er-model-the-concept-of-relationship-the-relationship-capacity-types-of-relationships-examples-ua/#q01

Додаток А. Список прийнятих скорочень

- MVP - Minimum viable product;
- ООП - Об'єктно-орієнтоване програмування;
- СКБД - Система управління базами даних.

Додаток Б. Код програми серверної частини

Models:

```
public class Category
    {
        [Key]
        public Guid CategoryId { get; set; }
        [field: Required]
        public string CategoryName { get; set; }
        [field: Required]
        public List<Item> Items { get; set; }
    }


public class Item
    {
        [field: Key]
        public Guid ItemId { get; set; }
        [field: Required]
        public string Headline { get; set; }
        [field: Required]
        public string Description { get; set; }
        [field: Required]
        public Guid UserId { get; set; }
        [field: Required]
        public Guid CategoryId { get; set; }
        [field: Required]
        public Boolean MyType { get; set; }
        [field: Required]
        public List<Photo> Photos { get; set; }
        [field: Required]
        public double Price { get; set; }
    }

public class Photo
    {
        [field: Key]
        public Guid PhotoId { get; set; }
        [field: Required]
        public Guid ItemId { get; set; }
        [field: Required]
        public string Base { get; set; }
    }

public class User
    {
        [Key]
        public Guid UserId { get; set; }
        [field: Required]
        public string FirstName { get; set; }
        [field: Required]
        public string LastName { get; set; }
        [field: Required]
        public string Patronymic { get; set; }
        [field: Required]
        public string Email { get; set; }
        [field: Required]
        public string Username { get; set; }
        [field: Required]
        public string Password { get; set; }
        [field: Required]
        public string Phone { get; set; }
```

```
        [field: Required]
        [DataType(DataType.Date)]
        public DateTime DateOfBirth { get; set; }
        [field: Required]
        [DataType(DataType.Date)]
        public DateTime LastOnline { get; set; }
        [field: Required]
        public string Address { get; set; }

        public User()
        {
        }
        public void SetPassword(string password)
        {
            Password = password;
        }

        //method that checks if the password is correct
        public bool CheckPassword(string userCandidate)
        {
            try
            {
                return Password.Equals(userCandidate);
            }
            catch (Exception)
            {
                return false;
            }
        }
    }
```

Data:

```
public class ApplicationContext : DbContext
    {
        public DbSet<User> Users { get; set; }
        public DbSet<Category> Categories { get; set; }
        public DbSet<Item> Items { get; set; }
        public DbSet<Photo> Photos { get; set; }

        public ApplicationContext(DbContextOptions<ApplicationContext> options)
            : base(options)
        {
            Database.EnsureCreated();
        }
        protected override void OnModelCreating(ModelBuilder modelBuilder)
        {

             Guid u1 = Guid.NewGuid();

             modelBuilder.Entity<User>().HasData(

                new User
                {
                    UserId = u1,
                    FirstName = "Богдана",
                    LastName = "Поебережець",
                    Patronymic = "Анатоліївна",
                    Email = "bogdano4ka1@gmail.com",
                    Username = "bogdana",
                    Phone = "0979154341",
                    Password = "bogdana",
```

```
                DateOfBirth = DateTime.Now,
                Address = "Україна, Київ",
                LastOnline = DateTime.Now
            }
        );

 Guid c1 = Guid.NewGuid();
        Guid c2 = Guid.NewGuid();
        Guid c3 = Guid.NewGuid();
        Guid c4 = Guid.NewGuid();
        Guid c5 = Guid.NewGuid();
        Guid c6 = Guid.NewGuid();
        Guid c7 = Guid.NewGuid();
        Guid c8 = Guid.NewGuid();
        modelBuilder.Entity<Category>().HasData(
            new Category
            {
                CategoryId = c1,
                CategoryName = "Тварини"
            },
            new Category
            {
                CategoryId = c2,
                CategoryName = "Робота"
            }, new Category
            {
                CategoryId = c3,
                CategoryName = "Велосипеди"
            },
            new Category
            {
                CategoryId = c4,
                CategoryName = "Будинок"
            }, new Category
            {
                CategoryId = c5,
                CategoryName = "Послуги"
            }, new Category
            {
                CategoryId = c6,
                CategoryName = "Магазин"
            }, new Category
            {
                CategoryId = c7,
                CategoryName = "Електроніка"
            }
            , new Category
            {
                CategoryId = c8,
                CategoryName = "Мода і стиль"
            }
        );
Guid i1 = Guid.NewGuid();
 Guid i2 = Guid.NewGuid();
         modelBuilder.Entity<Item>().HasData(
            new Item
            {
                ItemId = i1,
                Headline = "кіт британський вислоухий",
                Description = "Продаю породистого кота." +
                              "Вік - 1,5 місяці," +
                              "З прививками. Дуже грайливий." +
```

```
                                "Всі питання по телефону",
                    UserId = u1,
                    CategoryId = c1,
                    MyType = true,
                    Price = 1500
                },
                new Item
                {
                    ItemId = i2,
                    Headline = "Кіт короткошерстний",
                    Description = "Продаю породистого кота." +
                                  "Вік - 2,5 місяці," +
                                  "З прививками дуже грайливий" +
                                  "В подарунок місячний запс корму",
                    UserId = u1,
                    CategoryId = c1,
                    MyType = true,
                    Price = 500
                });
        }

    }


public class PaginatedList<T> : List<T>
    {
        public int PageIndex { get; private set; }
        public int TotalPages { get; private set; }

        public PaginatedList(List<T> items, int count, int pageIndex, int pageSize)
        {
            PageIndex = pageIndex;
            TotalPages = (int)Math.Ceiling(count / (double)pageSize);

            this.AddRange(items);
        }

        public bool HasPreviousPage => (PageIndex > 1);

        public bool HasNextPage => (PageIndex < TotalPages);

        public static async Task<PaginatedList<T>> CreateAsync(
            IQueryable<T> source, int pageIndex, int pageSize)
        {
            var count = await source.CountAsync();
            var items = await source.Skip(
                    (pageIndex - 1) * pageSize)
                .Take(pageSize).ToListAsync();
            return new PaginatedList<T>(items, count, pageIndex, pageSize);
        }
    }
```

Pages:

```
public class AddItemModel : PageModel
    {
        private readonly IConfiguration _configuration;
        private readonly ApplicationContext _context;
        public object FileUpload { get; set; }

        public AddItemModel(ApplicationContext context, IConfiguration conf)
```

```
        {
            _configuration = conf;
            _context = context;
        }

        public IEnumerable<SelectListItem> GetCategories()
        {
            return _context.Categories.ToArray().Select(x =>
                new SelectListItem()
                {
                    Text = x.CategoryName,
                    Value = x.CategoryId.ToString()
                });
        }
        public  IEnumerable<SelectListItem> GetTypes()
        {
            return new SelectListItem[]
            {
                new SelectListItem() { Text = "Нова", Value = "1" },
                new SelectListItem() { Text = "Б/у", Value = "0" }

            };
        }

        public IActionResult OnGet()
        {
            if (!Startup.IsSignedIn)
                return new RedirectResult("SignIn");
            return Page();
        }


        public IActionResult OnPostAddItem(IFormFileCollection uploads)
        {
            var headline = Request.Form["headline"];
            var category = Request.Form["Category"];
            var description = Request.Form["description"];
            var price = Request.Form["price"];
            var phone = Request.Form["phone"];
            var myType = Request.Form["MyType"];
            var email = Request.Form["email"];
            Guid guid = Guid.NewGuid();
            string sql = "INSERT INTO Items VALUES" +
                         " ('"+guid+"'," +
                         "'"+headline+"', '" + description+"', '" +
IndexModel.CurrentUser.UserId+"', " +
                         "'"+category+"', '"+myType+"', " +
                         "'"+price+"');";

            MySqlConnection cnn = new
MySqlConnection(_configuration.GetConnectionString("SqlConnectionString"));
            try
            {
                cnn.Open();
                MySqlCommand cmd = new MySqlCommand(sql) { Connection = cnn,
CommandText = sql };
                cmd.ExecuteNonQuery();
                cnn.Close();
                AddFiles(uploads, guid);
                return Redirect("/main");
            }
            catch
```

```
            {
                cnn?.Close();
                return Page();
            }
        }

        public  void AddFiles(IFormFileCollection uploads, Guid item)
        {
            foreach (var uploadedFile in uploads)
            {
                string path = "./wwwroot/img/" + uploadedFile.FileName;
                using (var fileStream = new FileStream(path, FileMode.Create))
                {
                     uploadedFile.CopyTo(fileStream);
                    fileStream.Close();
                }
                string sql = "INSERT INTO Photos VALUES
('"+Guid.NewGuid()+"','"+item+"','"+ uploadedFile.FileName + "');";

                MySqlConnection cnn = new
MySqlConnection(_configuration.GetConnectionString("SqlConnectionString"));

                try
                {
                    cnn.Open();
                    MySqlCommand cmd = new MySqlCommand(sql) { Connection = cnn,
CommandText = sql };
                    cmd.ExecuteNonQuery();
                    cnn.Close();

                }
                catch
                {
                    cnn?.Close();
                }
            }



        }
    }


public class FilteringModel : PageModel
    {
        private readonly ApplicationContext _context;
        public string NameSort { get; set; }
        public bool ShowItemDetail { get; set; }
        public bool ShowItems { get; set; }
        public string DateSort { get; set; }
        public string CategorySort { get; set; }
        public string CurrentFilter { get; set; }
        public string CurrentSort { get; set; }
        public PaginatedList<Item> Items { get; set; }

        public FilteringModel(ApplicationContext context)
        {
            _context = context;
            ShowItems = false;
            ShowItems = true;
            ShowItemDetail = false;
        }
```

```
public string ReturnImage(Guid itemId)
{
    Photo res = _context.Photos.FirstOrDefault(o => o.ItemId.Equals(itemId));
    if (res==null)
        return "no_image.png";
    return res.Base;
}

public IEnumerable<SelectListItem> GetCategories()
{
    return _context.Categories.ToArray().Select(x =>
        new SelectListItem()
        {
            Text = x.CategoryName,
            Value = x.CategoryId.ToString()
        });
}

public async Task OnGetAsync(string sortOrder,
    string currentFilter, string searchString, int? pageIndex)
{

    CurrentSort = sortOrder;
    NameSort = String.IsNullOrEmpty(sortOrder) ? "name_desc" : "";
    DateSort = sortOrder == "Date" ? "date_desc" : "Date";
    if (sortOrder != null)
    {
        CategorySort = sortOrder;
    }

    if (searchString != null)
    {
        pageIndex = 1;
    }
    else
    {
        searchString = currentFilter;
    }

    CurrentFilter = searchString;

    IQueryable<Item> items = from s in _context.Items
        select s;
    if (!String.IsNullOrEmpty(searchString))
    {
        items = items.Where(s => s.Headline.Contains(searchString));
    }
    switch (sortOrder)
    {
        case "name_desc":
            items = items.OrderByDescending(s => s.Headline);
            break;
        case "Date":
            items = items.OrderBy(s => s.Price);
            break;
        case "date_desc":
            items = items.OrderByDescending(s => s.Price);
            break;
        default:
            foreach (var category in GetCategories())
            {
```

```
                        if(category.Value.Equals(CategorySort))
                        {
                            items = items.Where(s => s.CategoryId.Equals(new
Guid(category.Value)));
                            break;
                        }
                    }
                    items = items.OrderBy(s => s.Headline);
                    break;
            }
            int pageSize = 8;
            Items = await PaginatedList<Item>.CreateAsync(
                items.AsNoTracking(), pageIndex ?? 1, pageSize);
        }

    }
public class MainModel : PageModel
    {
        public bool SettingsSelected { get; set; }
        private readonly ApplicationContext _context;
        private readonly IConfiguration _configuration;

        public bool AddsSelected { get; set; }
        public bool ShowItemDetail { get; set; }
        public bool ShowItems { get; set; }

        public bool EditItemSelected { get; set; }
        public PaginatedList<Item> DisplayedItems { get; set; }
        public static Item DisplayedItem { get; set; }
        public static Category ItemCategory { get; set; }
        public List<Photo> Photos;

        public MainModel(ApplicationContext context, IConfiguration conf)
        {
            _context = context;
            _configuration = conf;
        }

        public async Task OnGetAsync(int? pageIndex)
        {
            if (Startup.IsSignedIn)
            {
                AddsSelected = true;
                ShowItems = true;
                ShowItemDetail = false;
                AddsSelected = true;
                ShowItems = true;
                ShowItemDetail = false;
                SettingsSelected = false;
                IQueryable<Item> items = from s in _context.Items
                    where s.UserId.Equals(IndexModel.CurrentUser.UserId)
                    select s;
                int pageSize = 3;
                DisplayedItems = await PaginatedList<Item>.CreateAsync(
                    items.AsNoTracking(), pageIndex ?? 1, pageSize);
            }


        }
        public async Task OnPostAdds(int? pageIndex)
        {
```

```
        AddsSelected = true;
        ShowItems = true;
        ShowItemDetail = false;
        SettingsSelected = false;

        IQueryable<Item> items = from s in _context.Items
            where s.UserId.Equals(IndexModel.CurrentUser.UserId)
            select s;
        int pageSize = 3;
        DisplayedItems = await PaginatedList<Item>.CreateAsync(
            items.AsNoTracking(), pageIndex ?? 1, pageSize);

    }

    public void OnPostSettings()
    {
        AddsSelected = false;
        SettingsSelected = true;
        ShowItems = false;
        ShowItemDetail = false;
    }


    public IActionResult OnPostEditFirstname(string editFirstname)
    {
        IndexModel.CurrentUser.FirstName = editFirstname;
        SettingsSelected = true;
        _context.Users.Update(IndexModel.CurrentUser);
        _context.SaveChanges();
        return Page();
    }

    public void OnPostEditLastname(string editLastname)
    {
        IndexModel.CurrentUser.LastName = editLastname;
        SettingsSelected = true;
        _context.Users.Update(IndexModel.CurrentUser);
        _context.SaveChanges();
    }

    public void OnPostEditPatronymic(string editPatronymic)
    {
        IndexModel.CurrentUser.Patronymic = editPatronymic;
        SettingsSelected = true;
        _context.Users.Update(IndexModel.CurrentUser);
        _context.SaveChanges();
    }

    public void OnPostEditEmail(string editEmail)
    {
        IndexModel.CurrentUser.Email = editEmail;
        SettingsSelected = true;
        _context.Users.Update(IndexModel.CurrentUser);
        _context.SaveChanges();
    }

    public void OnPostEditPhone(string editPhone)
    {
        IndexModel.CurrentUser.Phone = editPhone;
        SettingsSelected = true;
        _context.Users.Update(IndexModel.CurrentUser);
        _context.SaveChanges();
```

```
        }

        public void OnPostEditUsername(string editUsername)
        {
            IndexModel.CurrentUser.Username = editUsername;
            SettingsSelected = true;
            _context.Users.Update(IndexModel.CurrentUser);
            _context.SaveChanges();
        }

        public void OnPostEditPassword(string previousPassword, string editPassword,
string repeatPassword)
        {
            if (!IndexModel.CurrentUser.CheckPassword(previousPassword))
            {
                return;
            }
            else if (!editPassword.Equals(repeatPassword))
            {
                return;
            }

            IndexModel.CurrentUser.SetPassword(editPassword);
            SettingsSelected = true;
            _context.Users.Update(IndexModel.CurrentUser);
            _context.SaveChanges();
        }

        public void OnPostEditAddress(string address)
        {
            IndexModel.CurrentUser.Address = address;
            SettingsSelected = true;
            _context.Users.Update(IndexModel.CurrentUser);
            _context.SaveChanges();
        }
        public void OnPostShowDetail()
        {
            var showDetail = Request.Form["showDetail"];
            if (!string.IsNullOrEmpty(showDetail))
            {
                DisplayedItem = _context.Items.FirstOrDefault(o => o.ItemId.Equals(new
Guid(showDetail)));
                Photos = _context.Photos.FromSqlRaw("SELECT * FROM Photos WHERE itemId
='" + showDetail + "';").ToList();
                ItemCategory = _context.Categories.FirstOrDefault(c =>
c.CategoryId.Equals(DisplayedItem.CategoryId));
            }

            AddsSelected = true;
            ShowItems = false;
            ShowItemDetail = true;
            SettingsSelected = false;

        }
        public void OnPostEditItem()
        {
            AddsSelected = false;
            ShowItemDetail = false;
            ShowItems = false;
            EditItemSelected = true;

        }
```

```
        public void OnPostEditItemDetails()
        {
            var headline = Request.Form["editHeadline"];
            var description = Request.Form["editDescription"];
            var price = Request.Form["editPrice"];
            var myType = Request.Form["MyType"];
            DisplayedItem.Headline = headline;
            DisplayedItem.Description = description;
            DisplayedItem.Price = Double.Parse(price);
            if (int.Parse(myType) == 0)
            {
                DisplayedItem.MyType = false;
            }
            else
            {
                DisplayedItem.MyType = true;
            }
            _context.Items.Update(DisplayedItem);
            _context.SaveChanges();

        }


        private readonly bool _trueSelected = DisplayedItem?.MyType ?? true ;
        public IEnumerable<SelectListItem> GetTypes()
        {

            return new SelectListItem[]
            {
                new SelectListItem() { Text = "Нова", Value = "1", Selected =
_trueSelected },
                new SelectListItem() { Text = "Б/у", Value = "0", Selected =
!_trueSelected }

            };
        }
        public string ReturnImage(Guid itemId)
        {
            Photo res = _context.Photos.FirstOrDefault(o => o.ItemId.Equals(itemId));
            if (res == null)
                return "no_image.png";
            return res.Base;
        }
        public IActionResult OnPostRemoveItem()
        {
            var toRemove = Request.Form["RemoveItem"];

            DisplayedItem = _context.Items.FirstOrDefault(o => o.ItemId.Equals(new
Guid(toRemove)));
            if(DisplayedItem!=null)
            {
                _context.Items.Remove(DisplayedItem);
                _context.SaveChanges();
                return Redirect("/main");
            }

            return Page();
        }

    }
```

```
public class SignInModel : PageModel
    {
        public string Message { get; set; }
        private readonly ApplicationContext _context;
        public SignInModel(ApplicationContext context)
        {
            _context = context;
        }

        public void OnGet()
        {
        }
        public IActionResult OnPost()
        {
            var username = Request.Form["username"];
            var password = Request.Form["password"];
            User user = _context.Users.FirstOrDefault(o => o.Username.Equals(username)
&& o.Password.Equals(password));

            if (user == null)
            {
                Message = "User not found";
            }
            else
            {
                Startup.IsSignedIn = true;
                IndexModel.CurrentUser = user;
                return Redirect("/main");
            }

            return Page();

        }
    }


public class SignOutModel : PageModel
    {
        public IActionResult OnGet()
        {
            Startup.IsSignedIn = false;
            IndexModel.CurrentUser = null;
            return Redirect("/filtering");
        }
    }
public class SignUpModel : PageModel
    {
        private readonly IConfiguration _configuration;
        private readonly ApplicationContext _context;
        public string Max;
        public SignUpModel(ApplicationContext contex, IConfiguration conf)
        {
            _configuration = conf;
            _context = contex;
            Max = DateTime.Now.ToString("yyyy-MM-dd");

        }

        public void OnGet()
        {
```

```
        }

        public IActionResult OnPost()
        {
            var firstname = Request.Form["firstname"];
            var lastname = Request.Form["lastname"];
            var patronymic = Request.Form["patronymic"];
            var email = Request.Form["email"];
            var username = Request.Form["username"];
            var password = Request.Form["password"];
            var dateofBirth = Request.Form["dateofBirth"];
            var phone = Request.Form["phone"];
            var address = Request.Form["address"];
            Guid guid = Guid.NewGuid();

            string sql = "insert into users VALUES " +
                         "('"+guid+"', '"+firstname+"', '"+lastname+"', " +
                         "'"+patronymic+"', '"+email+"', '"+username+"',
'"+password+"', '"+phone+"', '"+dateofBirth+"', " +
                         "'"+dateofBirth+"', '"+address+"'); ";

            MySqlConnection cnn = new
MySqlConnection(_configuration.GetConnectionString("SqlConnectionString"));

            try
            {
                cnn.Open();
                MySqlCommand cmd = new MySqlCommand(sql) { Connection = cnn,
CommandText = sql };
                cmd.ExecuteNonQuery();

                cnn.Close();
                IndexModel.CurrentUser = _context.Users.FirstOrDefault(u =>
u.UserId.Equals(guid));
                Startup.IsSignedIn = true;
                return Redirect("/main");
            }
            catch
            {
                cnn?.Close();
                return Page();
            }

        }
    }
```

Додаток В. Код програми клієнтської частини

```
@page
@model BoxNew.AddItemModel
@{
    ViewData["Title"] = "AddItem";
}

<div class="container bg-light  border-dark rounded">
    <form asp-page-handler="AddFiles" method="post" enctype="multipart/form-data">
        <h1 class="col-auto ml-auto border-bottom  text-success text-uppercase"> Додати
оголошення</h1>
        <div class="border-bottom ">
            <label class="col-lg-2 text-center " for="headline"> Заголовок </label>
            <input class="rounded col-lg-7 col-sm-12" type="text" placeholder="input
headline" required name="headline" id="headline" />
        </div>
        <div class="border-bottom row">
            <label class="col-lg-2 text-center" for="category"> Категорія </label>
            @Html.DropDownList("Category", @Model.GetCategories(), new { @class =
"dropdown-item col-lg-2" })

        </div>
        <div class="border-bottom">
            <label class="col-lg-2 text-center" for="description"> Опис </label>
            <textarea class="rounded col-lg-7 " name="description" id="description"
placeholder="type descrition" required></textarea>
        </div>
        <div class="border-bottom row ">
            <label class="col-lg-2 text-center" for="type"> Тип </label>
            @Html.DropDownList("MyType", @Model.GetTypes(), new { @class = "dropdown-
item col-lg-2" })
        </div>
        <div class="border-bottom row ">
            <label class="col-lg-2 text-center" for="price">Ціна</label>
            <input class="rounded" type="number" name="price" id="price" />

        </div>
        <br />
        <div class="align-content-center border-bottom">
            <h3 class="text-center text-black-50 text-primary">Завантажте фото</h3>
            <div class="row container">
                <input class="col-lg-3 col-sm-12 rounded" type="file" name="uploads"
/><br>
                <input class="col-lg-3 col-sm-12" type="file" name="uploads" /><br>
                <input class="col-lg-3 col-sm-12" type="file" name="uploads" /><br>
                <input class="col-lg-3 col-sm-12" type="file" name="uploads" /><br>
            </div>
        </div>
        <br />
        <div class="ml-auto">
            <input class="btn btn-lg btn-outline-success btn-block col-lg-5 ml-auto"
type="submit" name="AddItem" value="Add" asp-page-handler="AddItem" />
            <br />
        </div>

    </form>

</div>

<div class="row col-lg-12 p-1">

    <div class="col-lg-2 p-5">
```

```
    </div>
    <div class="col-lg-8">
        <form asp-page="./Filtering" method="get">
            <div>
                <div class="row">
                    <br />
                    <input type="text" class="col-lg-9  rounded" name="SearchString"
value="@Model.CurrentFilter" />
                    <input type="submit" value="Search" class="btn btn-success col-3"
/> |
                    <a class="text-dark text-xl-right" asp-page="./Filtering">Збити
фільтри</a>
                    <br />
                </div>
                <div class="border-bottom">
                    <nav class="navbar navbar-light bg-transparent">
                        <button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#navbarTogglePatronymic" aria-
controls="navbarTogglePatronymic" aria-expanded="false" aria-label="Toggle navigation">
                            <i style='font-size: 24px'>Обрати категорію</i>
                        </button>
                    </nav>
                    <div class="collapse" id="navbarTogglePatronymic">

                        <div class="row">
                            @{
                                foreach (var category in @Model.GetCategories())
                                {
                                    <div class="bg-transparent p-4">
                                        <a class="text-dark text-xl-right" asp-
page="./Filtering" asp-route-sortOrder="@category.Value"
                                                                asp-route-
currentFilter="@Model.CurrentFilter"> @category.Text </a>
                                    </div>
                                }
                            }
                        </div>
                    </div>
                </div>
            </div>
        </form>

        <table class="table">
            <thead>
                <tr>
                    <th>
                        <p>Фото</p>
                    </th>
                    <th>
                        <a class="text-primary" asp-page="./Filtering" asp-route-
sortOrder="@Model.NameSort"
                           asp-route-currentFilter="@Model.CurrentFilter">
                            <p>Заголовок</p>
                        </a>
                    </th>
                    <th>
                        <a class="text-primary" asp-page="./Filtering" asp-route-
sortOrder="@Model.DateSort"
                           asp-route-currentFilter="@Model.CurrentFilter">
                            <p>Ціна</p>
                        </a>
```

```
                    </th>
                </tr>
            </thead>
            <tbody>
            @foreach (var item in Model.Items)
            {

                <tr>
                    <td>
                        @{
                            <a href="./Item/@item.ItemId">

                                <img src="~/img/@Model.ReturnImage(item.ItemId)"
width="50" height="50" alt="Alternate Text"/>
                            </a>
                        }

                    </td>
                    <td>
                        <a href="./Item/@item.ItemId">
                            <p class="text-uppercase text-dark">
                                @Html.DisplayFor(modelItem => item.Headline)
                            </p>
                        </a>
                    </td>
                    <td>
                        @Html.DisplayFor(modelItem => item.Price)
                    </td>
                </tr>
            }
            </tbody>
        </table>


        @{
            var prevDisabled = !Model.Items.HasPreviousPage ? "disabled" : "";
            var nextDisabled = !Model.Items.HasNextPage ? "disabled" : "";
        }

        <a asp-page="./Filtering"
           asp-route-sortOrder="@Model.CurrentSort"
           asp-route-pageIndex="@(Model.Items.PageIndex - 1)"
           asp-route-currentFilter="@Model.CurrentFilter"
           class="btn btn-success @prevDisabled">
            Попередня
        </a>
        <a asp-page="./Filtering"
           asp-route-sortOrder="@Model.CurrentSort"
           asp-route-pageIndex="@(Model.Items.PageIndex + 1)"
           asp-route-currentFilter="@Model.CurrentFilter"
           class="btn btn-success @nextDisabled">
            Наступна
        </a>
    </div>
    <div class=" col-2">
        <br />
        <br />
        <br />
        <h1 class="border text-danger text-center"> ТУТ МОЖЕ БУТИ ВАША РЕКЛАМА</h1>
        <br />
        <h1 class="border text-center text-info"> ТУТ МОЖЕ БУТИ ВАША РЕКЛАМА</h1>
        <br />
```

```
        <h1 class="border text-center"> ТУТ МОЖЕ БУТИ ВАША РЕКЛАМА</h1>
    </div>

</div>


public class ItemModel : PageModel
    {
        private readonly ApplicationContext _context;
        public List<Photo> Photos;
        public static Category ItemCategory { get; set; }
        public static User Owner { get; set; }

        public Item Item;
        public ItemModel(ApplicationContext context)
        {
            _context = context;
        }

        public void OnGet(Guid userId)
        {

            Item = _context.Items.FirstOrDefault(o => o.ItemId.Equals(userId));
            Photos = _context.Photos.FromSqlRaw("SELECT * FROM Photos WHERE itemId ='"
+ userId + "';").ToList();
            ItemCategory = _context.Categories.FirstOrDefault(c =>
c.CategoryId.Equals(Item.CategoryId));
            Owner = _context.Users.FirstOrDefault(o => o.UserId.Equals(Item.UserId));
        }
    }

@page "{UserId}"

@model BoxNew.ItemModel
    <body>

        <div class="container">

            <h1 class="text-center text-dark text-uppercase"> @Model.Item.Headline</h1>
            <div class="row">
                <div id="carouselExampleControls" class="carousel col-lg-5 col-sm-12"
data-ride="carousel">
                    <div class="carousel-inner">
                        @{
                            if (@Model.Photos.Count == 0)
                            {
                                <div class="carousel-item)">
                                    <img class="d-block" height="500"
src="~/img/no_image.png" alt="kek">
                                </div>
                            }
                            else
                            {

                                var first = true;

                                @foreach (var item in @Model.Photos)
                                {
                                    <div class="carousel-item @(first ?
Html.Raw("active") : Html.Raw(""))">
                                        <img class="d-block" height="500"
src="~/img/@item.Base" alt="kek">
```

```
                                </div>
                                first = false;
                            }
                        }
                    }

                </div>
                <a class="carousel-control-prev" href="#carouselExampleControls"
role="button"
                   data-slide="prev">
                    <span class="carousel-control-prev-icon" aria-
hidden="true"></span>
                    <span class="sr-only">Попередня</span>
                </a>
                <a class="carousel-control-next" href="#carouselExampleControls"
role="button"
                   data-slide="next">
                    <span class="carousel-control-next-icon" aria-
hidden="true"></span>
                    <span class="sr-only">Наступна</span>
                </a>
            </div>
            <div class="col-lg-5 col-sm-12 align-content-md-around">
                <div class="border rounded">
                    <h2 class="text-left text-dark text-uppercase  ">Категорія:
@ItemModel.ItemCategory.CategoryName </h2>
                </div>
                <div class="border rounded ">
                    <h1 class="text-left text-dark text-uppercase  ">Ціна:
@Model.Item.Price UAH </h1>
                </div>
                <div class="border rounded ">
                    @{
                        if (@Model.Item.MyType)
                        {
                            <h1 class="text-left text-dark text-uppercase  "> Тип:
Нова </h1>
                        }
                        else
                        {
                            <h1 class="text-left text-dark text-uppercase  "> Тип:
Б/у </h1>
                        }
                    }

                </div>
                <div>
                </div>
                <div class="border rounded ">
                    <h1 class="text-left text-dark text-uppercase  ">Телефон:
@ItemModel.Owner.Phone </h1>
                </div>
                <div class="border rounded ">
                    <h1 class="text-left text-dark text-uppercase  ">Адреса:
@ItemModel.Owner.Address </h1>
                </div>

            </div>
            <div class="border-top ">
                <h1 class="text-left text-dark text-uppercase  ">Опис: </h1>
                <p class="text-left text-dark   "> @Model.Item.Description</p>
            </div>
```

```
            </div>
        </div>
    </body>


@page
@using BoxNew.Models
@model BoxNew.MainModel
@{
    ViewData["Title"] = "Main";
}
<body>

<div class="container">

            @{
                if (!Startup.IsSignedIn)
                {
                    Response.Redirect("/signin");
                }
                else if (Startup.IsSignedIn)
                {
                    <form method="post">
                        <div>
                            <ul class="nav nav-tabs">
                                <li>
                                    <input class=" bg-transparent rounded text-
uppercase" name="AddItem" value="Мої оголошення" asp-page-handler="Adds" type="submit"
/>
                                </li>
                                <li>
                                    <input class=" bg-transparent rounded text-
uppercase" name="Settings" value="Налаштування" asp-page-handler="Settings"
type="submit" />
                                </li>
                            </ul>
                        </div>
                        @{ if (@Model.AddsSelected)
                            {
                                if (@Model.ShowItems)
                                {
                                    if (@Model.DisplayedItems.Count == 0)
                                    {
                                        <h1> У вас поки немає оголошень</h1>
                                    }
                                    else
                                    {
                                        <table class="table">
                                            @foreach (Item item in
@Model.DisplayedItems)
                                            {
                                                <tr>
                                                    <td>

                                                        <button class="bg-transparent
w-100" type="submit" name="showDetail" value="@item.ItemId" asp-page-
handler="ShowDetail">
                                                            <div class="row">
                                                                <div class="col-lg-3
col-sm-12">
```

```
                                                                    <img
src="~/img/@Model.ReturnImage(item.ItemId)" width="100" height="100" alt="Alternate
Text" />
                                                                </div>
                                                                <div class="col-lg-6
col-sm-12">
                                                                    <p>
@item.Headline</p>
                                                                    <p>Ціна:
@item.Price</p>
                                                                </div>

                                                            </div>
                                                        </button>
                                                    </td>
                                                </tr>
                                            }
                                        </table>

                                        var prevDisabled =
!Model.DisplayedItems.HasPreviousPage ? "disabled" : "";
                                        var nextDisabled =
!Model.DisplayedItems.HasNextPage ? "disabled" : "";

                                        <a asp-page="./Main"
                                           asp-route-
pageIndex="@(Model.DisplayedItems.PageIndex - 1)"
                                           class="btn btn-success @prevDisabled">
                                            Previous
                                        </a>
                                        <a asp-page="./Main"
                                           asp-route-
pageIndex="@(Model.DisplayedItems.PageIndex + 1)"
                                           class="btn btn-success @nextDisabled">
                                            Next
                                        </a>
                                    }
                                }

                                else if (@Model.ShowItemDetail)
                                {
                                    <div>
                                        <h1 class="text-center text-dark text-
uppercase"> @MainModel.DisplayedItem.Headline</h1>
                                        <div class="row">
                                            <div id="carouselExampleControls"
class="carousel col-6" data-ride="carousel">
                                                <div class="carousel-inner">
                                                    @{
                                                        if (@Model.Photos.Count == 0)
                                                        {
                                                            <div class="carousel-
item)">
                                                                <img class="d-block"
height="500" src="~/img/no_image.png" alt="kek">
                                                            </div>
                                                        }
                                                        else
                                                        {

                                                            var first = true;
```

```
                                                    @foreach (var item in
@Model.Photos)
                                                    {
                                                        <div class="carousel-
item @(first ? Html.Raw("active") : Html.Raw(""))">
                                                            <img class="d-
block" height="500" src="~/img/@item.Base" alt="kek">
                                                        </div>
                                                        first = false;
                                                    }
                                                }
                                            }

                                        </div>
                                        <a class="carousel-control-prev"
href="#carouselExampleControls" role="button"
                                           data-slide="prev">
                                            <span class="carousel-control-prev-
icon" aria-hidden="true"></span>
                                            <span class="sr-
only">Попередня</span>
                                        </a>
                                        <a class="carousel-control-next"
href="#carouselExampleControls" role="button"
                                           data-slide="next">
                                            <span class="carousel-control-next-
icon" aria-hidden="true"></span>
                                            <span class="sr-
only">Наступна</span>
                                        </a>
                                    </div>
                                    <div class="col-5">
                                        <div class=" align-content-md-around">
                                            <div class="border-bottom">
                                                <h2 class="text-left text-dark
text-uppercase  ">Категорія: @MainModel.ItemCategory.CategoryName </h2>
                                            </div>
                                            <div class="border-bottom rounded
">
                                                <h2 class="text-left text-dark
text-uppercase  ">Ціна: @MainModel.DisplayedItem.Price UAH </h2>
                                            </div>

                                            <div class="border-bottom rounded
">
                                                <h2 class="text-left text-dark
text-uppercase  ">Телефон: @IndexModel.CurrentUser.Phone </h2>
                                            </div>
                                            <div class="border-bottom rounded
">
                                                @{
                                                    if
(MainModel.DisplayedItem.MyType)
                                                    {
                                                        <h2 class="text-left
text-dark text-uppercase  "> Тип: Нова </h2>
                                                    }
                                                    else
                                                    {
                                                        <h2 class="text-left
text-dark text-uppercase  "> Тип: Б/у </h2>
```

```
                                                    }
                                                }

                                            </div>
                                            <div class="border-bottom rounded
">
                                                <h2 class="text-left text-dark
text-uppercase"> Опис:</h2>
                                                <p class="text-left text-dark">
@MainModel.DisplayedItem.Description </p>
                                            </div>
                                        </div>
                                        <div>
                                            <div>
                                                <br />
                                                <button asp-page-
handler="EditItem" type="submit" id="EditItem" name="EditItem" class=" rounded btn-
success col-sm-12" value="@MainModel.DisplayedItem.ItemId">
                                                    <p>Редагувати
оголошення</p>
                                                </button>
                                            </div>
                                            <div>
                                                <br />
                                                <button asp-page-
handler="RemoveItem" type="submit" id="RemoveItem" name="RemoveItem" class=" rounded
btn-success col-sm-12" value="@MainModel.DisplayedItem.ItemId">
                                                    <p>Видалити оголошення</p>
                                                </button>
                                            </div>
                                        </div>
                                    </div>

                                </div>
                            </div>
                        }

                    }
  else if (@Model.SettingsSelected)
                    {

                        <div class="border-bottom">
                            <nav class="navbar navbar-light bg-transparent">
                                <button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#navbarToggleFirstname" aria-
controls="navbarToggleFirstname" aria-expanded="false" aria-label="Toggle navigation">
                                    <i style='font-size: 24px'>Прізвище</i>
                                </button>
                            </nav>
                            <div class="collapse" id="navbarToggleFirstname">
                                <div class="bg-transparent p-4">
                                    <input type="text" id="editFirstname"
name="editFirstname" class=" rounded col-sm-12"
value="@IndexModel.CurrentUser.FirstName">
                                    <br/>
                                    <input asp-page-handler="EditFirstname"
type="submit" id="editFirstname" name="editFirstname" placeholder="editFirstname"
class=" rounded col-sm-12" value="Edit">

                                </div>
                            </div>
```

```
                    </div>
                    <div class="border-bottom">
                        <nav class="navbar navbar-light bg-transparent">
                            <button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#navbarToggleLastName" aria-
controls="navbarToggleLastName" aria-expanded="false" aria-label="Toggle navigation">
                                <i style='font-size: 24px'>Ім'я</i>
                            </button>
                        </nav>
                        <div class="collapse" id="navbarToggleLastName">
                            <div class="bg-transparent p-4">
                                <input type="text" id="editLastname"
name="editLastname" class=" rounded col-sm-12"
value="@IndexModel.CurrentUser.LastName">
                                <br/>
                                <input asp-page-handler="EditLastname"
type="submit" id="editLastname" name="editLastname" placeholder="editLastname" class="
rounded col-sm-12" value="Edit">

                            </div>
                        </div>
                    </div>
                    <div class="border-bottom">
                        <nav class="navbar navbar-light bg-transparent">
                            <button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#navbarTogglePatronymic" aria-
controls="navbarTogglePatronymic" aria-expanded="false" aria-label="Toggle navigation">
                                <i style='font-size: 24px'>По-батькові</i>
                            </button>
                        </nav>
                        <div class="collapse" id="navbarTogglePatronymic">
                            <div class="bg-transparent p-4">
                                <input type="text" id="editPatronymic"
name="editPatronymic" class=" rounded col-sm-12"
value="@IndexModel.CurrentUser.Patronymic">
                                <br/>
                                <input asp-page-handler="EditPatronymic"
type="submit" id="editPatronymic" name="editPatronymic" placeholder="editPatronymic"
class=" rounded col-sm-12" value="Edit">

                            </div>
                        </div>
                    </div> <div class="border-bottom">
                        <nav class="navbar navbar-light bg-transparent">
                            <button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#navbarToggleEmail" aria-controls="navbarToggleEmail"
aria-expanded="false" aria-label="Toggle navigation">
                                <i style='font-size: 24px'>Емейл</i>
                            </button>
                        </nav>
                        <div class="collapse" id="navbarToggleEmail">
                            <div class="bg-transparent p-4">
                                <input type="email" id="editEmail" name="editEmail"
class=" rounded col-sm-12" value="@IndexModel.CurrentUser.Email">
                                <br/>
                                <input asp-page-handler="EditEmail" type="submit"
id="editEmail" name="editEmail" placeholder="editEmail" class=" rounded col-sm-12"
value="Edit">

                            </div>
                        </div>
                    </div>
```

```
                        <div class="border-bottom">
                            <nav class="navbar navbar-light bg-transparent">
                                <button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#navbarTogglePhone" aria-controls="navbarTogglePhone"
aria-expanded="false" aria-label="Toggle navigation">
                                    <i style='font-size: 24px'>Телефон</i>
                                </button>
                            </nav>
                            <div class="collapse" id="navbarTogglePhone">
                                <div class="bg-transparent p-4">
                                    <input type="tel" id="editPhone" name="editPhone"
class=" rounded col-sm-12" value="@IndexModel.CurrentUser.Phone">
                                    <br/>
                                    <input asp-page-handler="EditPhone" type="submit"
id="editPhone" name="editPhone" placeholder="editPhone" class=" rounded col-sm-12"
value="Edit">

                                </div>
                            </div>
                        </div>

                        <div class="border-bottom">
                            <nav class="navbar navbar-light bg-transparent">
                                <button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#navbarToggleEditUsername" aria-
controls="navbarToggleEditUsername" aria-expanded="false" aria-label="Toggle
navigation">
                                    <i style='font-size: 24px'>Логін</i>
                                </button>
                            </nav>
                            <div class="collapse" id="navbarToggleEditUsername">
                                <div class="bg-transparent p-4">
                                    <input type="text" id="editUsername"
name="editUsername" class=" rounded col-sm-12"
value="@IndexModel.CurrentUser.Username">
                                    <br/>
                                    <input asp-page-handler="EditUsername"
type="submit" id="editUsername" name="editUsername" placeholder="editUsername" class="
rounded col-sm-12" value="Edit">

                                </div>
                            </div>
                        </div>
                        <div class="border-bottom">
                            <nav class="navbar navbar-light bg-transparent">
                                <button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#navbarToggleeditPasssword" aria-
controls="navbarToggleeditPasssword" aria-expanded="false" aria-label="Toggle
navigation">
                                    <i style='font-size: 24px'>Пароль</i>
                                </button>
                            </nav>
                            <div class="collapse" id="navbarToggleeditPasssword">
                                <div class="bg-transparent p-4">
                                    <label>Вкажіть поточний пароль</label>
                                    <input type="password" id="previousPassword"
name="previousPassword" class=" rounded col-sm-12">
                                    <label>Вкажіть новий пароль</label>
                                    <input type="password" id="editPassword"
name="editPassword" class=" rounded col-sm-12">
                                    <label>Повторіть новий пароль</label>
```

```
                                    <input type="password" id="repeatPassword"
name="repeatPassword" class=" rounded col-sm-12">
                                    <br/>
                                    <input asp-page-handler="EditPassword"
type="submit" id="editPassword" name="editPassword" placeholder="editPassword" class="
rounded col-sm-12" value="Edit">

                                </div>
                            </div>
                        </div>
                        <div class="border-bottom">
                            <nav class="navbar navbar-light bg-transparent">
                                <button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#navbarToggleEditCountry" aria-
controls="navbarToggleEditCountry" aria-expanded="false" aria-label="Toggle
navigation">
                                    <i style='font-size: 24px'>Адреса</i>
                                </button>
                            </nav>
                            <div class="collapse" id="navbarToggleEditCountry">
                                <div class="bg-transparent p-4">
                                    <input type="text" id="address" name="address"
class=" rounded col-sm-12" value="@IndexModel.CurrentUser.Address">
                                    <br/>
                                    <input asp-page-handler="EditAddress" type="submit"
id="address" name="address" placeholder="address" class=" rounded col-sm-12"
value="Edit">

                                </div>
                            </div>
                        </div>

                    }
else if (@Model.EditItemSelected)
                    {
                        <div>
                            <div class="border-bottom">
                                <div class="bg-transparent p-4">
                                    <i style='font-size: 24px'>Заголовок: </i>
                                    <input type="text" name="editHeadline" class="
rounded col-sm-12" value="@MainModel.DisplayedItem.Headline" required="">
                                    <br/>
                                    <i style='font-size: 24px'>Опис: </i>
                                    <input type="text" name="editDescription" class="
rounded col-sm-12" value="@MainModel.DisplayedItem.Description" required="">
                                    <br/>
                                    <i style='font-size: 24px'>Ціна: </i>
                                    <input type="number" name="editPrice" class="
rounded col-sm-12" value="@MainModel.DisplayedItem.Price" required="">
                                    <br/>
                                    <i style='font-size: 24px'>Тип: </i>
                                    @Html.DropDownList("MyType", @Model.GetTypes(), new
{@class = "dropdown-item col-lg-2"})
                                    <br/>
                                    <br/>
                                    <button asp-page-handler="EditItemDetails"
name="buttonEdit" class="btn btn-outline-success col-sm-12" type="submit"
value="@MainModel.DisplayedItem.ItemId">Edit</button>
                                </div>
                            </div>
```

```
                    </div>

                }
            }
            </form>
        }

    }
</div>
</body>


@page
@model BoxNew.SignInModel
@{
    ViewData["Title"] = "SignIn";
}

    <body>

        <div class="container">
            <h1>Вхід</h1>
            <h2>@Model.Message</h2>
            <form name="Signin" method="post">
                <br />
                <div>
                    <label for="username">Логін</label>
                    <input type="text" id="username" name="username"
laplaceholder="username" class="rounded col-lg-12 " required="">
                    <br />
                </div>
                <div>
                    <label for="username">Пароль</label>

                    <input type="password" id="Password" name="Password"
placeholder="Password" class="rounded col-lg-12 " required="">
                    <br />
                </div>
                <br />
                <button class="btn btn-lg btn-success btn-block"
type="submit">Ввійти</button>
            </form>
        </div>
    </body>

@page
@model BoxNew.SignUpModel
@{
    ViewData["Title"] = "SignUp";
}
    <body>
        <div class="container">

            <form class="d-flex justify-content-center border border-secondary rounded"
action='' method="post">
                <fieldset>
                    <legend class="">Реєстрація</legend>
                    <div>
                        <label for="Firstname">Ім'я</label>
                        <input type="text" id="Firstname" name="Firstname"
class="rounded col-lg-7 col-sm-12" required="">
```

```
                        <br />
                    </div>
                    <div>
                        <label for="Lastname">Прізвище</label>
                        <input type="text" id="Lastname" name="Lastname" class="rounded
col-lg-7 col-sm-12" required="">
                        <br />
                    </div>

                    <div>
                        <label for="Patronymic">По-батькові</label>
                        <input type="text" id="Patronymic" name="Patronymic"
class="rounded col-lg-7 col-sm-12" required="">
                        <br />
                    </div>
                    <div>
                        <label for="DateOfBirth">Дата народження</label>
                        <input type="date" id="DateOfBirth" name="DateOfBirth"
class="rounded col-lg-7 col-sm-12" required="" max='@Model.Today'>
                        <br />
                    </div>
                    <div>
                        <label for="email">Телефон</label>
                        <input type="tel" id="phone" name="phone" class="rounded col-
lg-7 col-sm-12" required="">
                        <br />
                    </div>
                    <div>
                        <label for="email">Емейл</label>
                        <input type="email" id="email" name="email" class="rounded col-
lg-7 col-sm-12" required="">
                        <br />
                    </div>
                    <div>
                        <label for="Country">Адреса</label>

                        <input type="text" id="address" name="address" class="rounded
col-lg-7 col-sm-12" required="">
                        <br />
                    </div>
                    <div>
                        <label for="Username">Логін</label>
                        <input type="text" id="Username" name="Username" class="
rounded col-lg-7 col-sm-12" required="">
                        <br />
                    </div>
                    <div>
                        <label for="Password">Пароль</label>
                        <input type="password" id="Password" name="Password"
placeholder="" class=" rounded col-lg-7 col-sm-12" required="">
                        <br />
                    </div>

                    <div>
                        <br />
                        <button class="btn btn-success ">Зареєструватись</button>
                        <br />
                        <br />
                    </div>
                </fieldset>
            </form>
        </div>
```

```
    </body>
@page
@model BoxNew.SignOutModel
@{
    ViewData["Title"] = "SignOut";
}

<h1>SignOut</h1>
```