

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

магістра

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ

(назва теми)

ПІДТРИМКИ ПРОДАЖУ ВЖИВАНИХ МОТОЦИКЛІВ

RESEARCH AND SOFTWARE IMPLEMENTATION OF METHODS

FOR SUPPORTING USED MOTORCYCLES SALE

Виконав(ла): студент(ка) 2 курсу, групи КНТ-113м

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

НУЖНИЙ О.Ю.

(ПРІЗВИЩЕ та ініціали)

Керівник

СКРУПСЬКИЙ С.Ю.

(ПРІЗВИЩЕ та ініціали)

Рецензент

ПОЛЯКОВ М.О.

(ПРІЗВИЩЕ та ініціали)

2024_____

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет _____ КНТ _____
Кафедра _____ програмних засобів _____
Ступінь вищої освіти _____ магістр _____
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.
Сергій СУББОТІН
« _____ » _____ 2024 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

НУЖНОГО Олександр Юрійович

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація методів підтримки продажу вживаних мотоциклів. Research and Software Implementation of Methods for Supporting Used Motorcycles Sale

керівник проєкту (роботи) к.т.н., доцент СКРУПСЬКИЙ Степан Юрійович _____,
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від « 7 » жовтня 2024 року № 398

2. Строк подання студентом проєкту (роботи) 01 грудня 2024 року
3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз проблеми та постановка завдань дослідження. 2. Матеріали і методи. 3. Проєктування програмного забезпечення. 4. Основні рішення щодо реалізації програмного забезпечення. 5. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількості слайдів, плакатів)
Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	СКРУПСЬКИЙ С.Ю., доцент		
Нормоконтроль	КАМІНСЬКА Ж.К., асистент		

7. Дата видачі завдання « 06 » вересня 2024 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 тиждень	Розділ 1
3	Дослідження проблеми	3-5 тижні	Розділ 2
4	Проектування та розробка архітектури програми	6 тиждень	Розділ 3
5	Розробка програми	7-8 тижні	Розділ 4
6	Тестування та експериментальне дослідження	9-10 тижні	Розділ 5
7	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	11 тиждень	Додатки
8	Захист роботи	12 тиждень	

Студент(ка)

(підпис) Олександр НУЖНИЙ
(Ім'я ПРІЗВИЩЕ)

Керівник проєкту (роботи)

(підпис) Степан СКРУПСЬКИЙ
(Ім'я ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:
99 с., 29 рис., 3 табл., 3 дод., 10 джерел.

ВЖИВАНИЙ МОТОЦИКЛ, ПРОДАЖ, ОГОЛОШЕННЯ,
ВЕБЗАСТОСУНОК, ПРОГНОЗУВАННЯ.

Об'єкт роботи – процес продажу вживаних мотоциклів.

Предмет роботи – методи підтримки продажу вживаних мотоциклів.

Метою роботи є розробка програмного забезпечення підтримки продажу вживаних мотоциклів для забезпечення зручності потенційних продавців під час публікації оголошень про продаж вживаних мотоциклів на основі засобів прогнозування вартості.

Матеріали, методи та технічні засоби: дерева рішень, мова програмування Python, фреймворк Django, система управління базами даних MySQL.

Результати. Програма призначена для забезпечення підтримки продажу вживаних мотоциклів шляхом надання відповідних засобів для виставлення власних вживаних мотоциклів на продаж, пошуку вживаних мотоциклів для придбання. Мотоцикли представлені виробниками, типами, моделями, версіями цих моделей, що визначаються представниками сервісу продажу вживаних мотоциклів, а вже на основі цих даних визначаються оголошення про продаж вживаних мотоциклів самими власниками.

Висновки. Наукова новизна роботи полягає в методі прогнозування вартості вживаних мотоциклів, який базується на створенні та навчанні дерева рішень, визначає процес створення і відбору вхідних ознак, що дозволяє встановити вартість вживаного мотоцикла на етапі формування оголошення.

Галузь використання. Платформа для підтримки продажу вживаних мотоциклів від власників або продавців, які мають одну або декілька одиниць.

ABSTRACT

Explanatory note to the diploma qualifying work of the master: 99 pages, 29 figures, 3 tables, 3 appendixes, 10 sources.

USED MOTORCYCLE, SALE, ADVERTISEMENT, WEBAPPLICATION, FORECASTING.

The object of the work is the process of selling used motorcycles.

The subject of the work is methods of supporting the sale of used motorcycles.

The purpose of the work is to develop a software to support the sale of used motorcycles to ensure the convenience of potential sellers when publishing ads for the sale of used motorcycles based on value forecasting tools.

Materials, methods and technical means: decision trees, Python programming language, Django framework, MySQL database management system.

The results. The program is designed to support the sale of used motorcycles by providing the appropriate means to list own used motorcycles for sale, search for used motorcycles to purchase. Motorcycles are represented by manufacturers, types, models, versions of these models, which are determined by the representatives of the service for the sale of used motorcycles, and based on this data, the ads for the sale of used motorcycles are determined by the owners themselves.

Conclusions. The scientific novelty of the work consists in the method of forecasting the price of used motorcycles, which is based on the creation and training of a decision tree, determines the process of creating and selecting input features, which allows to determine the price of a used motorcycle at the stage of creating an ad.

Field of use. A platform to support the sale of used motorcycles from owners or sellers who have one or more units.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	8
Вступ.....	9
1 Аналіз проблеми та постановка завдань дослідження	10
1.1 Функціональні можливості та концепція існуючих програмних застосунків підтримки продажу вживаних мотоциклів	10
1.2 Визначення рівня створюваного вебзастосунку підтримки продажу вживаних мотоциклів	14
1.3 Висновки за розділом 1	15
2 Матеріали і методи.....	17
2.1 Задача прогнозування вартості вживаних мотоциклів	17
2.2 Метод прогнозування вартості вживаних мотоциклів.....	18
2.3 Висновки за розділом 2	21
3 Проєктування програмного забезпечення	22
3.1 Визначення сценаріїв роботи користувачів з вебзастосунком підтримки продажу вживаних мотоциклів	22
3.2 Вибір засобів розробки вебзастосунку підтримки продажу вживаних мотоциклів.....	28
3.3 Проєктування структури бази даних.....	29
3.4 Висновки за розділом 3	38
4 Основні рішення щодо реалізації програмного забезпечення	40
4.1 Загальний порядок роботи з замовленнями перегляду або придбання вживаних мотоциклів	40
4.2 Реалізовані програмні одиниці у складі програмного забезпечення підтримки продажу вживаних мотоциклів	41
4.3 Висновки за розділом 4	55
5 Експлуатація, тестування та експериментальне дослідження програми	56
5.1 Основне призначення програми	56
5.2 Виконання програми.....	56

5.3 Експериментальне дослідження прогнозування вартості вживаних мотоциклів	62
5.4 Висновки за розділом 5	63
Висновки	64
Перелік джерел посилання	65
Додаток А Технічне завдання	66
Додаток Б Текст програми	71
Додаток В Слайди презентації	92

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

- САП – середня абсолютна похибка;
СКП – середня квадратична похибка.

ВСТУП

За наявною на даний момент статистикою весь загальний світовий ринок вживаних мотоциклів на 2021 рік оцінювався сумарно в 38,7 мільярда американських доларів, при цьому за наявними прогнозами вважається, що до 2031 року світовий ринок вживаних автомобілів має досягти 66,2 мільярда американських доларів, демонструючи зростання одразу на 5,6 % у середньому до 2031 року [1]. Звісно, що такі показники можна вважати достатньо потужними, а такий ринок вживаних мотоциклів заслуговує на увагу, підтримку та використання ресурсів для підтримки його розширення.

У звіті [1] вказується на те, що існує цілий ряд джерел придбання вживаних мотоциклів, до яких належать франчайзингові та незалежні дилери, компанії з прокату мотоциклів, аукціони, приватні розпродажі і лізингові агентства. При цьому нерідко сучасне покоління молодих людей, а також і сучасні люди загалом надають перевагу саме вживаним мотоциклам, адже це дозволяє зменшити витрати, що рухає цим напрямком. Відповідно створення програмного забезпечення підтримки продажу вживаних мотоциклів, яке б дозволяло індивідуальним власникам вживаних мотоциклів виставляти їх на продаж, при цьому підтримуючи таким чином розширення ринку на основі надання їм відповідного майданчика, а також забезпечення таких користувачів засобами прогнозування для підтримки зокрема на етапі публікації оголошення є важливим.

Метою роботи є розробка програмного забезпечення підтримки продажу вживаних мотоциклів для забезпечення зручності потенційних продавців під час публікації оголошень про продаж вживаних мотоциклів на основі засобів прогнозування вартості.

1 АНАЛІЗ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ

1.1 Функціональні можливості та концепція існуючих програмних застосунків підтримки продажу вживаних мотоциклів

Продаж вживаних мотоциклів є доволі популярною сферою, а тому існує багато вебзастосунків, які підтримують роботу в цій сфері.

UsedMotorcycleStore.com (рис. 1.1) реалізує продаж вживаних мотоциклів у Сполучених Штатах Америки, де це один з найпопулярніших сервісів [2]. При цьому відображається форма пошуку вживаних мотоциклів, яка містить ряд фільтрів, що включають проміжок років виробництва вживаного мотоцикла, виробника (можна обрати декілька зі списку), проміжок вартості вживаного мотоциклу, тип мотоциклу, категорію, проміжок відстані, подоланої вживаним мотоциклом.

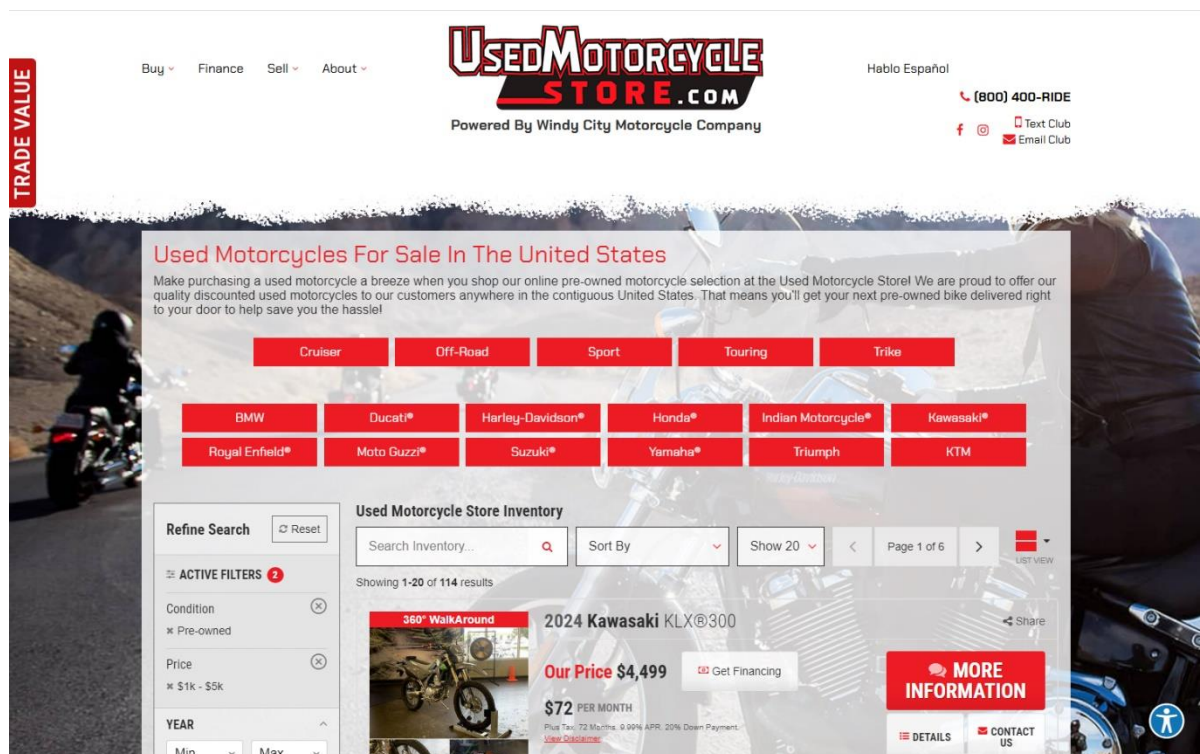


Рисунок 1.1 – Пошук вживаних мотоциклів у програмі
UsedMotorcycleStore.com

За кожним мотоциклом можна переглянути деталізовану інформацію,

що включає фотографії (рис. 1.2), деталізований опис (рис. 1.3).

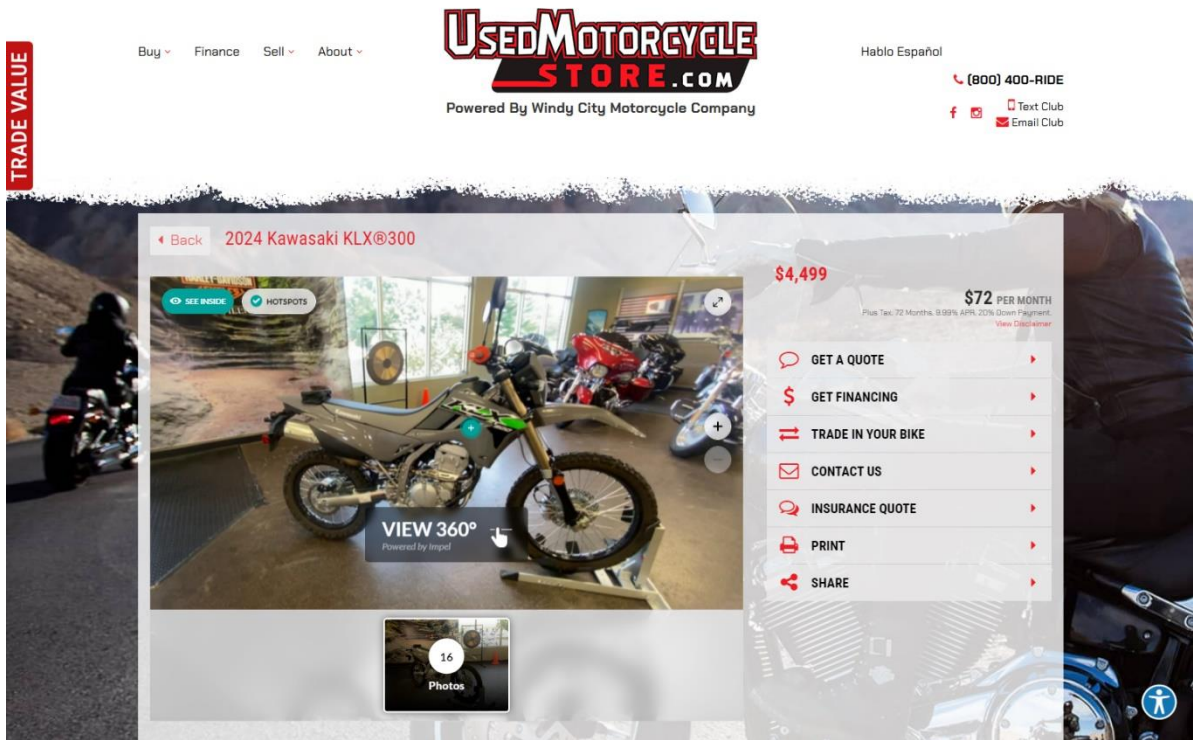


Рисунок 1.2 – Верх сторінки вживаного мотоциклу в програмі
UsedMotorcycleStore.com

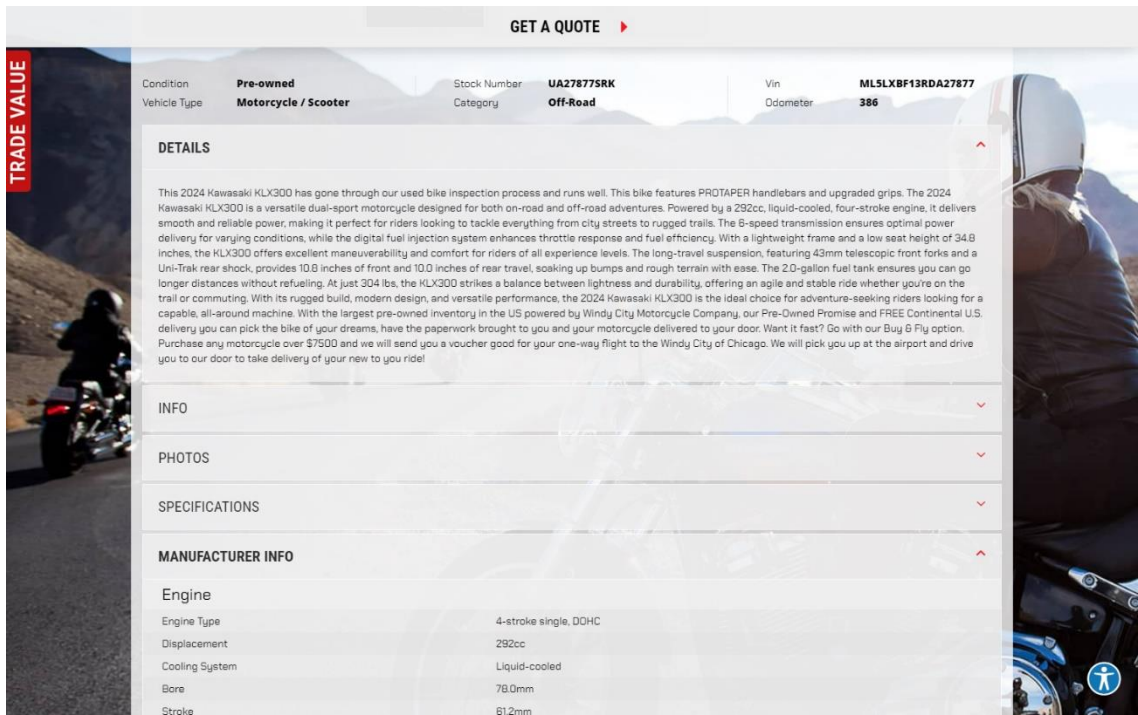


Рисунок 1.3 – Нижня частина сторінки вживаного мотоциклу в програмі
UsedMotorcycleStore.com

Рисунок 1.4 – Сторінка вживаного мотоциклу в програмі Bikesales

Під час пошуку в Bikesales можна задати за мотоциклами їх виробника, тип, місце розташування, вартість, подолану відстань, особливості та ключові слова, особливості двигуна мотоцикла, висоту сидіння, вагу мотоцикла, колір мотоцикла. За кожним мотоциклом надається розширена інформація про нього.

AutoScout24 (рис. 1.5) представляє і нові, і вживані мотоцикли [4]. За мотоциклами можна виконати пошук, задаючи виробника, модель мотоцикла, максимальну вартість мотоцикла, його тип, місце розташування, а також додаткові параметри, що зокрема можуть включати кількість місць, подолану відстань мотоциклом, потужність двигуна, коробку передач, стан мотоцикла, колір.

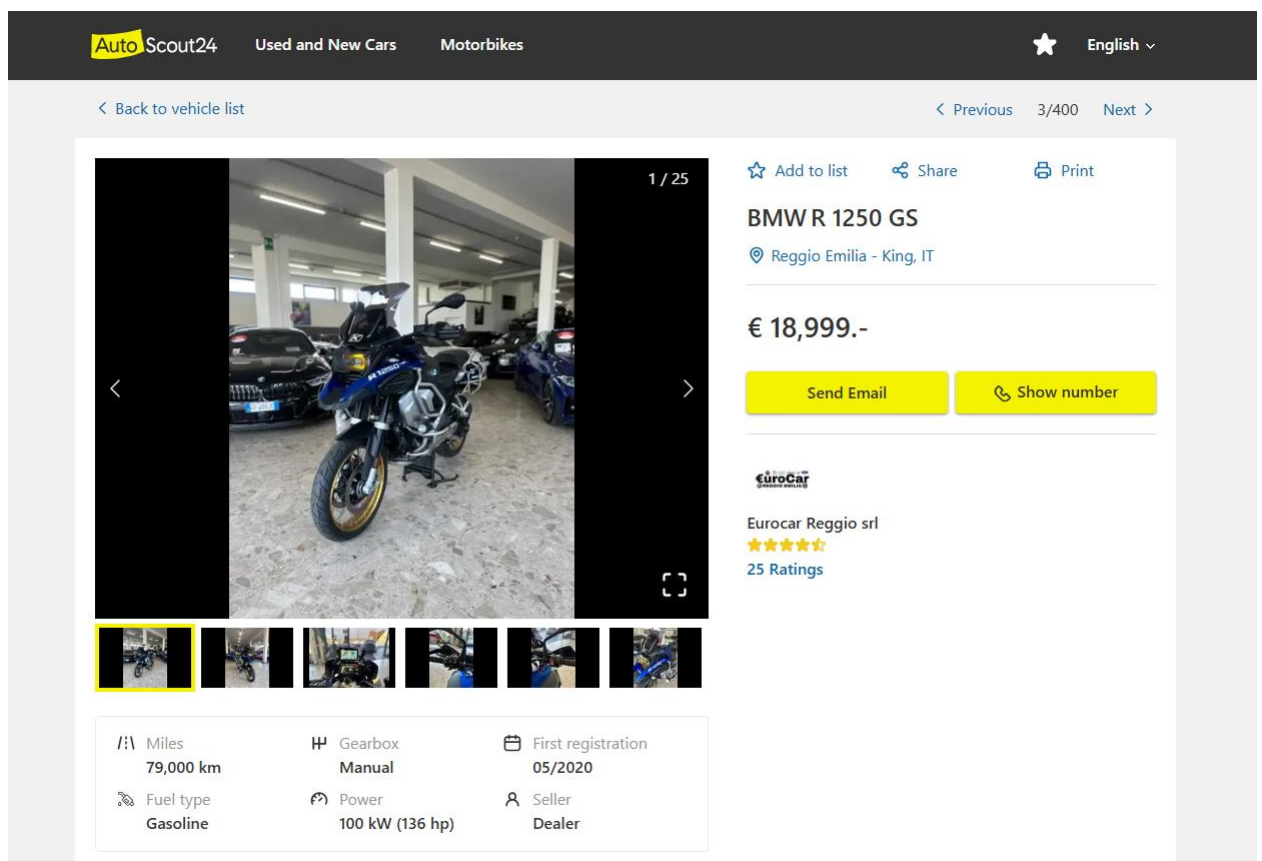


Рисунок 1.5 – Сторінка вживаного мотоциклу в програмі AutoScout24

Motolux (рис. 1.6) представляє український ресурс, де можна придбати, а відповідно і продати вживаний мотоцикл [5]. На вебсайті представлені не

тільки мотоцикли, але і інша техніка. Під час пошуку можна задати проміжок вартості мотоцикла, виробника мотоцикла, тип мотоцикла, проміжок значень об'єму двигуна мотоцикла, колір мотоцикла, потужність двигуна, систему охолодження, паливну систему, країну виробника мотоцикла.

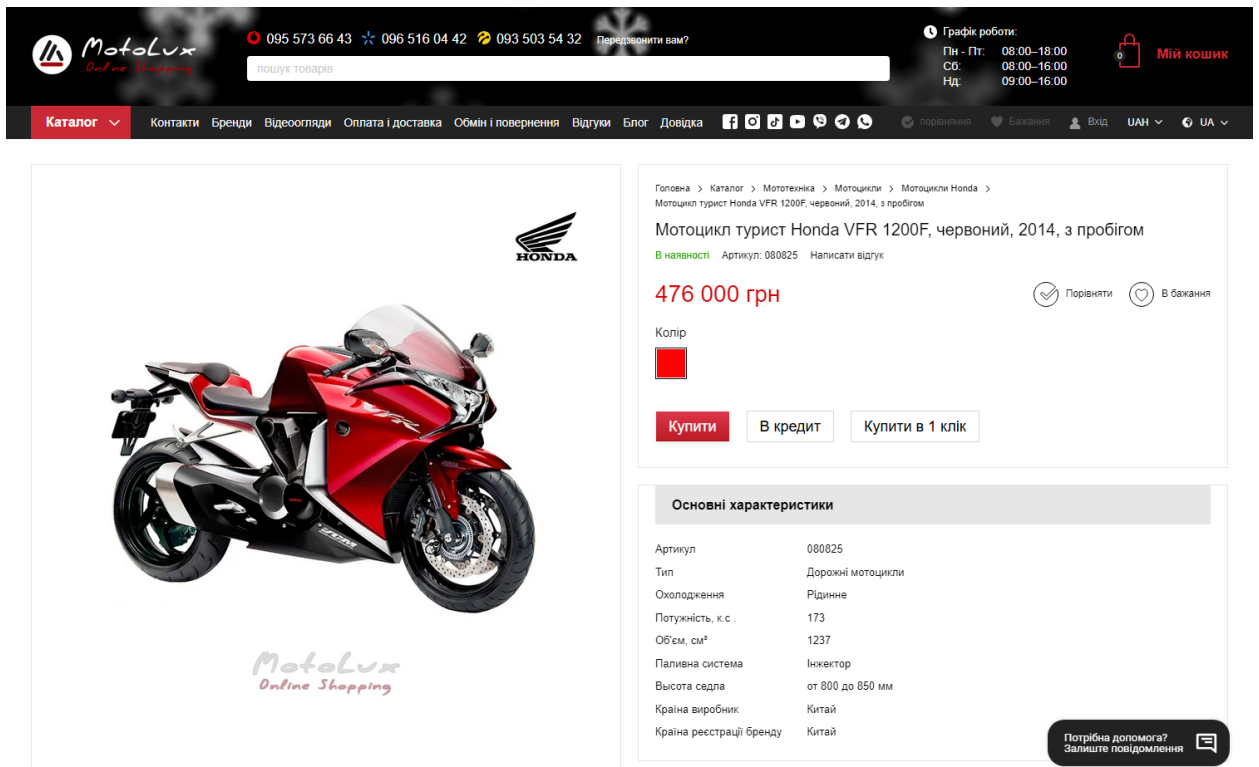


Рисунок 1.6 – Сторінка вживаного мотоциклу в програмі Motolux

1.2 Визначення рівня створюваного вебзастосунку підтримки продажу вживаних мотоциклів

Порівняння аналогів розробки вебзастосунку підтримки продажу вживаних мотоциклів включило в себе Motolux, Bikesales, UsedMotorcycleStore.com (табл. 1.1). Враховуючи те, що у розробці виконується продаж тільки вживаних мотоциклів, то створення програмного інструменту для автоматизованого визначення вартості продажу вживаного мотоцикла є дуже важливим.

Таблиця 1.1 – Порівняння можливостей аналогів розробки вебзастосунку підтримки продажу вживаних мотоциклів

Показник	Motolux	Bikesales	UsedMotorcycle Store.com	Розробка
Продаж вживаних мотоциклів	Так	Так	Так	Так
Пошук мотоциклів	Розши- рений	Розши- рений	Основні параметри	Розши- рений
Написання повідомлення власнику	Так	Так	Ні	Так
Формування заявок на перегляд мотоцикла	Ні	Ні	Ні	Так
Оцінювання переглянутого мотоцикла	Так, але фактично використовується для запитань	Ні	Ні	Так
Визначення вартості продажу вживаного мотоцикла	Ні	Ні	Ні	Так

1.3 Висновки за розділом 1

Розробка вебзастосунку підтримки продажу вживаних мотоциклів передбачає виконання завдань:

- виконати аналіз програмних застосунків підтримки продажу вживаних мотоциклів;
- розробити метод прогнозування вартості продажу вживаного мотоциклу;
- виконати проєктування вебзастосунку підтримки продажу вживаних мотоциклів;
- реалізувати вебзастосунок підтримки продажу вживаних мотоциклів;
- виконати експериментальне дослідження прогнозування вартості продажу вживаного мотоциклу.

У першому розділі виконано завдання аналізу програмних застосунків підтримки продажу вживаних мотоциклів, де було розглянуто зокрема як аналоги Motolux, Bikesales, UsedMotorcycleStore.com, а в підсумку визначені особливості програми, що розробляється.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Задача прогнозування вартості вживаних мотоциклів

Задача прогнозування вартості вживаних мотоциклів полягає в тому, щоб на основі значень показників, які характеризують вживаний мотоцикл, виконати прогнозування значення вартості продажу цього вживаного мотоцикла.

При цьому для виконання цього прогнозування необхідно створити модель, яка виконує перетворення значень показників, які характеризують вживаний мотоцикл, на його вартість, тобто вартість, за яку готовий продати мотоцикл його власник (або продавець). Для створення такої моделі наявна навчальна вибірка, для якої задано набір екземплярів, кожен з яких представляє окремий вживаний мотоцикл з визначеними для нього значеннями показників (вхідних ознак моделі) та значенням вартості продажу вживаного мотоцикла (вихідна ознака моделі).

Відповідно для розв'язання задачі прогнозування вартості вживаних мотоциклів необхідно на основі навчальної вибірки даних побудувати таку модель, а тоді використати її для прогнозування вартості продажу вживаних мотоциклів, які не зустрічались у навчальній вибірці.

Кожне оголошення про продаж вживаного мотоцикла представлено наступними ознаками:

- вартість продажу вживаного мотоцикла;
- відстань, яку подолав мотоцикл за час свого використання поточним або всіма своїми власниками;
- потужність двигуна вживаного мотоцикла;
- виробник та модель вживаного мотоцикла, що може бути представлена окремо виробником і окремо моделлю, бути спільним рядком, що містить обидва значення;

– дата виготовлення вживаного мотоцикла, що може бути представлена повною датою, тільки місяцем і роком або тільки роком (останнього варіанта також достатньо);

- тип палива вживаного мотоцикла;
- коробка передач вживаного мотоцикла;
- тип пропозиції щодо вживаного мотоцикла.

Відповідно необхідно визначити, яким чином перетворити ці ознаки вживаних мотоциклів на вхідні ознаки моделі прогнозування вартості продажу вживаних мотоциклів, якщо це необхідно зробити. А далі вже на основі вхідних ознак моделі визначити її вихідну ознаку – вартість продажу вживаного мотоцикла, яку рекомендовано використати у випадку зазначеного оголошення.

2.2 Метод прогнозування вартості вживаних мотоциклів

Метод прогнозування вартості вживаних мотоциклів дозволяє розв'язати задачу прогнозування вартості вживаних мотоциклів через послідовність наступних етапів.

На першому етапі має бути зібрано дані з оголошень про продаж вживаних мотоциклів для формування навчальної вибірки в подальшому. Ці дані мають бути зібрані як набір значень, кожне з яких характеризує конкретний вживаний мотоцикл:

- вартість продажу цього вживаного мотоцикла;
- відстань, яку подолав цей вживаний мотоцикл за час свого використання;
- потужність двигуна цього вживаного мотоцикла;
- виробник та модель цього вживаного мотоцикла;
- дата виготовлення цього вживаного мотоцикла;
- тип палива, що використовується цим вживаним мотоциклом;
- коробка передач цього вживаного мотоцикла;

– тип пропозиції щодо цього вживаного мотоцикла.

На другому етапі має бути сформовано безпосередньо навчальну вибірку даних про вживані мотоцикли, для чого необхідно спочатку вилучити всі наявні дублікати з даних, вилучити ті екземпляри даних вживаних мотоциклів, які мають пропущені значення ознак (якщо тільки пропуск не відповідає певному значенню, яке чітко наперед визначено, а якщо це все ж так, то замість виключення таких екземплярів необхідно підставити відповідне значення за замовчуванням).

Далі необхідно перетворити характеристики вживаних мотоциклів на вхідні ознаки майбутньої моделі наступним чином:

– виконати нормування вартості продажу вживаних мотоциклів, зводячи ці значення до інтервалу від 0 до 1;

– виділення виробника вживаного мотоцикла з запису про модель вживаного мотоциклу: якщо виробник зазначений окремо, то саме це значення виключно має бути використано як значення цієї вхідної ознаки, проте якщо це значення разом з моделлю представляє єдиний рядок, то спочатку має бути сформовано перелік виробників мотоциклів, а тоді для кожної моделі має бути перевірено входження у модель в якості першої (лівої) частини назви виробника мотоцикла, а відповідно виходячи з того, який саме виробник складає ліву частину моделі, таке значення виробника і має бути встановлено в результаті для вживаного мотоцикла, а далі ці значення мають бути перетворені на категорії (кожне унікальне значення виробника має бути перетворено на унікальне ціле число);

– дата виготовлення вживаного мотоцикла має бути перетворена на ціле число, для чого має бути визначено дату виготовлення вживаного мотоцикла, яка є найбільш ранньою у вибірці даних (або це може бути встановлено дату, коли було виготовлено перший мотоцикл в принципі): для кожної дати виготовлення вживаного мотоцикла необхідно обчислити повну кількість років, яка пройшла з цієї найбільш ранньої дати виготовлення мотоцикла до дати виготовлення цього мотоцикла;

– тип палива, що використовується вживаним мотоциклом, має бути перетворений на категоріальне значення, тобто кожне унікальне значення типу палива вживаного мотоцикла має бути перетворено на унікальне ціле число;

– коробка передач вживаного мотоцикла має бути перетворена на категоріальне значення, тобто кожне унікальне значення коробки передач вживаного мотоцикла має бути перетворено на унікальне ціле число;

– серед всіх типів пропозиції щодо вживаного мотоцикла має бути залишено тільки ті, які стосуються вживаних мотоциклів, а всі інші, якщо вони є, мають бути вилучені з навчальної вибірки.

Після того як всі ці перетворення виконано, буде сформовано навчальну вибірку щодо продажу вживаних мотоциклів.

Наступний етап передбачає створення моделі на основі дерева рішень [6] та навчання цієї моделі за допомогою сформованої навчальної вибірки даних. Під час навчання моделі має бути реалізовано відбір входних ознак цієї моделі. Для цього на основі дерева рішень має бути оцінено інформативність кожної ознаки. Після чого має бути обрано підмножину входних ознак, які мають найбільшу інформативність серед всіх доступних та які є достатньо інформативними, тобто мають відносно інших ознак зокрема достатньо великі значення, для подальшого використання в моделі.

Останній етап базується на постійному повторному виконанні, коли з'являється оголошення про продаж вживаного мотоцикла. Для такого оголошення виділяються значення характеристик, виділених на першому етапі, та реалізується перетворення цих значень на входні ознаки моделі за допомогою дій, передбачених на другому етапі. Після цього таким чином отримані значення входних ознак вживаного мотоцикла подаються на вхід навченої моделі, а вже вона реалізує прогнозування вартості продажу цього вживаного мотоцикла.

2.3 Висновки за розділом 2

Виконано загальне представлення постановки задачі прогнозування вартості вживаних мотоциклів, запропоновано метод, який дозволяє ці задачу розв'язати.

Наукова новизна роботи полягає в методі прогнозування вартості вживаних мотоциклів, який базується на створенні та навчанні дерева рішень, визначає процес створення і відбору вхідних ознак, що дозволяє встановити вартість вживаного мотоцикла на етапі формування оголошення.

3 ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Визначення сценаріїв роботи користувачів з вебзастосунком підтримки продажу вживаних мотоциклів

Основні сценарії роботи користувача вебзастосунку підтримки продажу вживаних мотоциклів, який є потенційним покупцем (користувач-покупець):

- перегляд даних версії мотоциклу;
- перегляд існуючих версій моделі мотоцикла;
- пошук вживаних мотоциклів;
- перегляд даних виробника мотоциклів;
- перегляд оголошення про продаж вживаного мотоциклу;
- перегляд оголошень про продаж вживаних мотоциклів у користувача;
- авторизація;
- перегляд даних профілю користувача як власника вживаних мотоциклів та покупця мотоциклів;
- формування замовлення для перегляду або придбання вживаного мотоциклу;
- створення відгуку про вживаний мотоцикл;
- перегляд відгуків про вживаний мотоцикл;
- перегляд відгуків про вживані мотоцикли, створених користувачем;
- збереження оголошення про продаж вживаного мотоциклу;
- перегляд збережених оголошень про продаж вживаного мотоциклу;
- відправлення повідомлення з запитанням за оголошенням про продаж вживаного мотоцикла;
- перегляд повідомлень, які написані користувачем за оголошенням про продаж вживаного мотоцикла;
- перегляд оголошень про продаж вживаних мотоциклів, за якими написав повідомлення користувач;

– перегляд власних замовлень для перегляду або придбання вживаного мотоциклу (рис. 3.1).



Рисунок 3.1 – Діаграма прецедентів користувача-покупця

Альтернативні сценарії роботи користувача вебзастосунку підтримки продажу вживаних мотоциклів, який є потенційним покупцем:

– оновлення даних власного профілю користувача як покупця мотоциклів, коли дані в профілі застаріли, то це є альтернативою перегляду даних профілю користувача;

– скасування замовлення для перегляду або придбання вживаного мотоциклу, коли замовлення для перегляду або придбання вживаного мотоциклу тільки створене як запит, то це є альтернативою формуванню замовлення для перегляду або придбання вживаного мотоциклу;

– скасування збереження оголошення про продаж вживаного мотоциклу, коли оголошення про продаж вживаного мотоциклу тим самим користувачем-покупцем вже було збережено, то це є альтернативою збереженню оголошення про продаж вживаного мотоциклу;

– реєстрація користувача, коли у такого користувача-покупця відсутній користувацький запис.

Основні сценарії роботи користувача вебзастосунку підтримки продажу вживаних мотоциклів, який є власником або продавцем вживаних мотоциклів (користувач-власник):

- перегляд даних версії мотоциклу;
- перегляд існуючих версій моделі мотоцикла;
- пошук вживаних мотоциклів;
- перегляд даних виробника мотоциклів;
- публікація оголошення про продаж вживаного мотоциклу;
- перегляд оголошення про продаж вживаного мотоциклу;
- перегляд оголошень про продаж вживаних мотоциклів у користувача;
- авторизація;
- перегляд даних профілю користувача як власника вживаних мотоциклів та покупця мотоциклів;
- керування замовленням для перегляду або придбання вживаного мотоциклу;
- перегляд відгуків про вживаний мотоцикл;
- перегляд відгуків про вживані мотоцикли, створених користувачем;

- відправлення повідомлення у відповідь за оголошенням про продаж вживаного мотоцикла;
- перегляд користувачів, які написали за оголошенням про продаж вживаного мотоцикла;
- перегляд повідомлень, які написані користувачем за оголошенням про продаж вживаного мотоцикла;
- перегляд замовлень за оголошенням про продаж вживаного мотоцикла (рис. 3.2).



Рисунок 3.2 – Діаграма прецедентів користувача-власника

Альтернативні сценарії роботи користувача вебзастосунку підтримки продажу вживаних мотоциклів, який є власником вживаних мотоциклів або продавцем:

- оновлення оголошення про продаж вживаного мотоциклу, коли дані в оголошенні певним чином застаріли, то це є альтернативою для публікації оголошення про продаж вживаного мотоциклу;
- оновлення даних власного профілю користувача, коли дані в профілі застаріли, то це є альтернативою перегляду даних профілю користувача;
- реєстрація користувача як власника вживаних мотоциклів, то це є альтернативою для авторизації, коли у такого користувача-власника відсутній користувацький запис.

Основні сценарії роботи користувача вебзастосунку підтримки продажу вживаних мотоциклів, який є менеджером сервісу:

- додавання версій мотоциклів;
- перегляд даних версії мотоциклу;
- перегляд існуючих версій моделі мотоцикла;
- пошук вживаних мотоциклів;
- перегляд даних виробника мотоциклів;
- перегляд даних профілю користувача;
- перегляд оголошення про продаж вживаного мотоциклу;
- перегляд оголошень про продаж вживаних мотоциклів у користувача;
- авторизація;
- керування замовленням для перегляду або придбання вживаного мотоциклу;
- перегляд відгуків про вживаний мотоцикл;
- перегляд відгуків про вживані мотоцикли, створених користувачем;
- перегляд користувачів, які написали за оголошенням про продаж вживаного мотоцикла;
- перегляд повідомлень, які написані користувачем за оголошенням про продаж вживаного мотоцикла;

- перегляд оголошень про продаж вживаних мотоциклів, за якими написав повідомлення користувач;
- перегляд замовлень для перегляду або придбання вживаного мотоциклу користувача;
- перегляд замовлень за оголошенням про продаж вживаного мотоцикла (рис. 3.3).



Рисунок 3.3 – Діаграма прецедентів менеджера сервісу

Альтернативним сценарієм є оновлення даних версії мотоцикла для додавання версії мотоцикла, коли дані були внесені неточно або потребують певних змін.

3.2 Вибір засобів розробки вебзастосунку підтримки продажу вживаних мотоциклів

Засоби вебзастосунку підтримки продажу вживаних мотоциклів зведено на основі початкового вибору до фреймворків Django [7] і Pyramid [8] (табл. 3.1), які є фреймворками веброзробки для мови Python.

Таблиця 3.1 – Порівняння можливих рішень для реалізації вебзастосунку підтримки продажу вживаних мотоциклів

	Django	Pyramid
Робота з базою даних	Засоби взаємодії з базою даних та вбудоване об'єктно-реляційне відображення	Необхідно використовувати засоби зовнішніх розробників для реалізації об'єктно-реляційного відображення, інсталювати
Наповнення даними вебзастосунку (зокрема стандартними)	Вбудована адміністративна панель повністю вирішує цю функціональність	Необхідно використовувати засоби зовнішніх розробників, інсталювати
Структуризація коду	Окремі складові на основі шаблону Model-View-Template з можливостями розширення	Весь вебзастосунок представлений одним файлом
Робота з засобами прогнозування і аналітики	Доступні всі актуальні Python-засоби	

Наявність мови Python дозволяє повноцінно задіяти всі необхідні можливості роботи з засобами прогнозування і аналітики. Django обрано, зважаючи, що він одразу надає засоби взаємодії з базою даних та вбудоване об'єктно-реляційне відображення, має вбудовану адміністративну панель, що повністю вирішує функціональність наповнення даними вебзастосунку, окрім того вебзастосунок має структурований код, що дозволяє його необхідним чином далі розширювати зручно.

3.3 Проєктування структури бази даних

Реалізація бази даних як і її проєктування виконані на основі системи MySQL, вибір на користь якої чітко визначений тим, що це вільне програмне забезпечення, а також підтверджений актуальним практичним застосуванням, наявністю потужних засобів, достатніх для розробки програмного забезпечення підтримки продажу вживаних мотоциклів.

У базі даних таблиця `Motorcycletype` необхідна для збереження даних типів мотоциклів (дорожні, позашляховики тощо) на основі стовпців:

- `id` – ідентифікаційний номер типу мотоцикла;
- `motortypename` – назва типу мотоцикла.

У базі даних таблиця `Motorcycleproducer` необхідна для збереження даних виробників мотоциклів на основі стовпців:

- `id` – ідентифікаційний номер виробника мотоциклів;
- `motorprodname` – назва виробника мотоциклів;
- `motorpdorsite` – вебсайт виробника мотоциклів;
- `registrationcountrypk` – країна, до якої належить цей виробник мотоциклів, представлена зовнішнім зв'язком з таблицею `Locationcountry`.

У базі даних таблиця `Motorcyclemodel` необхідна для збереження даних моделей мотоциклів на основі стовпців:

- `id` – ідентифікаційний номер моделі мотоцикла;

- motorcycleproducerpk – виробник цієї моделі мотоцикла, представлений зовнішнім зв'язком з таблицею Motorcycleproducer;
- motormodelname – назва моделі мотоцикла;
- motorcycletypepk – тип мотоцикла, до якого належить даний, представлений зовнішнім зв'язком з таблицею Motorcycletype;
- startproduced – рік, з якого почалось виготовлення мотоциклів цієї моделі виробником;
- endproduced – рік, коли закінчилось виготовлення мотоциклів цієї моделі виробником.

У базі даних таблиця Motorcycleversion необхідна для збереження даних різних версій моделей мотоциклів на основі стовпців:

- id – ідентифікаційний номер версії моделі мотоцикла;
- motorcyclemodelpk – модель мотоцикла, версія якої представлена, представлена зовнішнім зв'язком з таблицею Motorcyclemodel;
- motoversionname – назва версії моделі мотоцикла;
- cooling – система охолодження, яка використовується в цій версії моделі мотоцикла;
- enginetype – тип двигуна, який використовується в цій версії моделі мотоцикла;
- enginepower – потужність двигуна, який використовується в цій версії моделі мотоцикла;
- enginevolume – обсяг двигуна, який використовується в цій версії моделі мотоцикла;
- enginebore – діаметр двигуна, який використовується в цій версії моделі мотоцикла;
- enginestroke – хід двигуна, який використовується в цій версії моделі мотоцикла;
- enginecompressionration – рівень компресії двигуна, який використовується в цій версії моделі мотоцикла;

- fuelsystem – система палива, яка використовується в цій версії моделі мотоцикла;
- engineignition – тип запалення, який використовується в цій версії моделі мотоцикла;
- chassisframetype – тип рами шасі, який використовується в цій версії моделі мотоцикла;
- frontbrakingsystem – передня гальмівна система, яка використовується в цій версії моделі мотоцикла;
- rearbrakingsystem – задня гальмівна система, яка використовується в цій версії моделі мотоцикла;
- fronttire – гума переднього колеса, яка використовується в цій версії моделі мотоцикла;
- frontwheeltravel – хід переднього колеса, який використовується в цій версії моделі мотоцикла;
- reartire – гума заднього колеса, яка використовується в цій версії моделі мотоцикла;
- rearwheeltravel – хід заднього колеса, який використовується в цій версії моделі мотоцикла;
- minheigh – мінімальна висота посадки, яка доступна в цій версії моделі мотоцикла;
- maxheight – максимальна висота посадки, яка доступна в цій версії моделі мотоцикла;
- producedcountrypk – країна, в якій виготовлялась ця версія моделі мотоцикла, представлена зовнішнім зв'язком з таблицею Locationcountry;
- gearbox – тип коробки передач, який використовувався в цій версії моделі мотоцикла;
- seats – кількість місць у цій версії моделі мотоцикла;
- cylindernumber – кількість циліндрів двигуна, який використовується в цій версії моделі мотоцикла;
- emptyweight – вага цієї версії моделі мотоцикла в порожньому стані;

- fullweight – повна вага цієї версії моделі мотоцикла;
- lengthsize – максимальна довжина цієї версії моделі мотоцикла в метрах;
- widthsize – максимальна ширина цієї версії моделі мотоцикла в метрах;
- heightsize – максимальна висота цієї версії моделі мотоцикла в метрах;
- wheelbase – розмір колісної бази в метрах;
- maxspeed – максимальна швидкість, яку може розвивати ця версія мотоцикла в кілометрах на годину.

У базі даних таблиця Motorcyclead необхідна для збереження даних оголошень про продаж вживаних мотоциклів на основі стовпців:

- id – ідентифікаційний номер оголошення про продаж вживаних мотоциклів;
- motorcycleversionpk – версія моделі вживаного мотоцикла, представленого в оголошенні, представлена зовнішнім зв'язком з таблицею Motorcycleversion;
- motorcycleownerpk – власник вживаного мотоцикла, який представлений в оголошенні, або його продавець, представлений зовнішнім зв'язком з таблицею Motorcycleowner;
- description – опис вживаного мотоцикла з оголошення;
- firstregistration – рік або дата першої реєстрації вживаного мотоцикла з оголошення;
- price – вартість вживаного мотоцикла з оголошення, за яку готовий продати його власник або продавець;
- colour – колір вживаного мотоцикла;
- mileage – відстань в кілометрах, які подолав цей вживаний мотоцикл на момент продажу;
- usestate – стан вживаного мотоцикла;
- adstate – стан оголошення про продаж вживаного мотоцикла;

– viewrate – середня оцінка, що була виставлена вживаному мотоциклу користувачами;

– locationplacepk – місто, де знаходиться вживаний мотоцикл, представлена зовнішнім зв'язком з таблицею Locationplace.

У базі даних таблиця Motorcycleadphoto необхідна для збереження даних фотографій вживаних мотоциклів з оголошень на основі стовпців:

– id – ідентифікаційний номер фотографії вживаного мотоцикла з оголошення;

– motorcycleadpk – оголошення про продаж вживаного мотоцикла, до якого відноситься ця фотографія, представлена зовнішнім зв'язком з таблицею Motorcyclead;

– photo – фотографія вживаного мотоцикла з оголошення.

У базі даних таблиця Motorcycleowner необхідна для збереження даних користувача, який є власником або продавцем вживаного мотоцикла або мотоциклів, на основі стовпців:

– id – ідентифікаційний номер користувача, який є власником або продавцем вживаного мотоцикла або мотоциклів;

– user – користувацький запис в Django, що використовується цим власником або продавцем вживаного мотоцикла або мотоциклів, представлене зовнішнім зв'язком з таблицею User, що є стандартною для Django;

– fullname – повне ім'я власника або продавця вживаного мотоцикла або мотоциклів;

– presentation – опис власника або продавця вживаного мотоцикла або мотоциклів;

– photo – фотографія власника або продавця вживаного мотоцикла або мотоциклів;

– locationplacepk – місце (місто), де знаходиться продавець або власник вживаного мотоцикла або мотоциклів і де відповідно знаходяться його мотоцикли для продажу, представлене зовнішнім зв'язком з таблицею Locationplace;

– `additionaladdress` – повна адреса в зазначеному місті, за якою знаходяться мотоцикли цього власника або продавця.

У базі даних таблиця `Motorcyclesearcher` необхідна для збереження даних користувача, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, на основі стовпців:

– `id` – ідентифікаційний номер користувача, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем;

– `user` – користувацький запис в Django, що використовується цим користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, представлений зовнішнім зв'язком з таблицею `User`, що є стандартною для Django;

– `fullname` – повне ім'я користувача, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем;

– `presentation` – опис користувача, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем;

– `photo` – фотографія користувача, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем;

– `phone` – номер телефону користувача, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем;

– `locationplacerk` – місце (місто), де знаходиться користувач, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, представлене зовнішнім зв'язком з таблицею `Locationplace`;

– `additionaladdress` – повна адреса в зазначеному місті, за якою знаходиться користувач, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем.

У базі даних таблиця `Motorcycleorder` необхідна для збереження даних замовлення вживаного мотоцикла для придбання на основі стовпців:

– `id` – ідентифікаційний номер замовлення вживаного мотоцикла для придбання;

– motorcycleadpk – оголошення про продаж вживаного мотоцикла, якого стосується замовлення, представлене зовнішнім зв'язком з таблицею Motorcyclead;

– orderstate – стан замовлення вживаного мотоцикла для придбання;

– ordereddate – час замовлення вживаного мотоцикла для придбання;

– motorcyclesearcherpk – користувач, який створив замовлення на придбання або перегляд вживаного мотоцикла.

У базі даних таблиця Motorcycleview необхідна для збереження даних відгуків про вживані мотоцикли на основі стовпців:

– id – ідентифікаційний номер відгуку про вживаний мотоцикл користувача, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем;

– motorcycleadpk – оголошення про продаж вживаного мотоцикла, за яким залишив свій відгук користувач, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, представлене зовнішнім зв'язком з таблицею Motorcyclead;

– motorcyclesearcherpk – користувач, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, що залишив відгук про вживаний мотоцикл, представлене зовнішнім зв'язком з таблицею Motorcyclesearcher;

– viewratemark – виставлена оцінка (значення) у відгуку про вживаний мотоцикл користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем;

– viewratetext – текст відгуку про вживаний мотоцикл користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем;

– viewdate – час створення відгуку про вживаний мотоцикл користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем.

У базі даних таблиця `Motorcyclesave` необхідна для збереження даних збереження оголошення про продаж вживаного мотоцикла на основі стовпців:

- `id` – ідентифікаційний номер збереження оголошення про продаж вживаного мотоцикла;

- `motorcycleadpk` – оголошення про продаж вживаного мотоцикла, яке було збережено користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, представлене зовнішнім зв'язком з таблицею `Motorcyclead`;

- `motorcyclesearcherpk` – користувач, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, що зберіг оголошення про продаж вживаного мотоцикла, представлений зовнішнім зв'язком з таблицею `Motorcyclesearcher`;

- `savedate` – час збереження оголошення про продаж вживаного мотоцикла.

У базі даних таблиця `Locationcountry` необхідна для збереження даних країн, в яких можуть знаходитись вживані мотоцикли та їхні власники, на основі стовпців:

- `id` – ідентифікаційний номер країни, в якій може знаходитись вживаний мотоцикл та його власник;

- `countryname` – назва країни, в якій може знаходитись вживаний мотоцикл та його власник.

У базі даних таблиця `Locationplace` необхідна для збереження даних міст (місць), в яких може знаходитись вживаний мотоцикл та його власник, на основі стовпців:

- `id` – ідентифікаційний номер міста (місця), в якому може знаходитись вживаний мотоцикл та його власник;

- `locationcountrypk` – країна, в якій знаходиться місто (місце), в якому може знаходитись вживаний мотоцикл та його власник, представлений зовнішнім зв'язком з таблицею `Locationcountry`;

– placename – назва міста (місця), в якому може знаходитись вживаний мотоцикл та його власник.

У базі даних таблиця Motorcycletexting необхідна для збереження даних повідомлень, надісланих між користувачами, які виконують пошук вживаного мотоцикла для придбання і є потенційними покупцями, і користувачами, які є власниками або продавцями вживаних мотоциклів, на основі стовпців:

– id – ідентифікаційний номер повідомлення, надісланого між користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, і користувачем, який є власником або продавцем вживаних мотоциклів;

– motorcycleadpk – оголошення про продаж вживаного мотоцикла, за яким було відправлене повідомлення, надіслане між користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, і користувачем, який є власником або продавцем вживаних мотоциклів, представлене зовнішнім зв'язком з таблицею Motorcyclead;

– motorcyclesearcherpk – користувач, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, який створив або якому направлено повідомлення, надіслане між користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, і користувачем, який є власником або продавцем вживаних мотоциклів, представлений зовнішнім зв'язком з таблицею Motorcyclesearcher;

– textowner – власництво відправленого повідомлення, тобто це повідомлення, створене користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, або повідомлення, створене користувачем, який є власником або продавцем вживаних мотоциклів;

– texting – текст повідомлення, надісланого між користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, і користувачем, який є власником або продавцем вживаних мотоциклів;

– textdate – час відправлення повідомлення, надісланого між користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, і користувачем, який є власником або продавцем вживаних мотоциклів;

– textread – позначка про прочитання повідомлення, надісланого між користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, і користувачем, який є власником або продавцем вживаних мотоциклів, стороною, якій повідомлення адресоване (рис. 3.4).

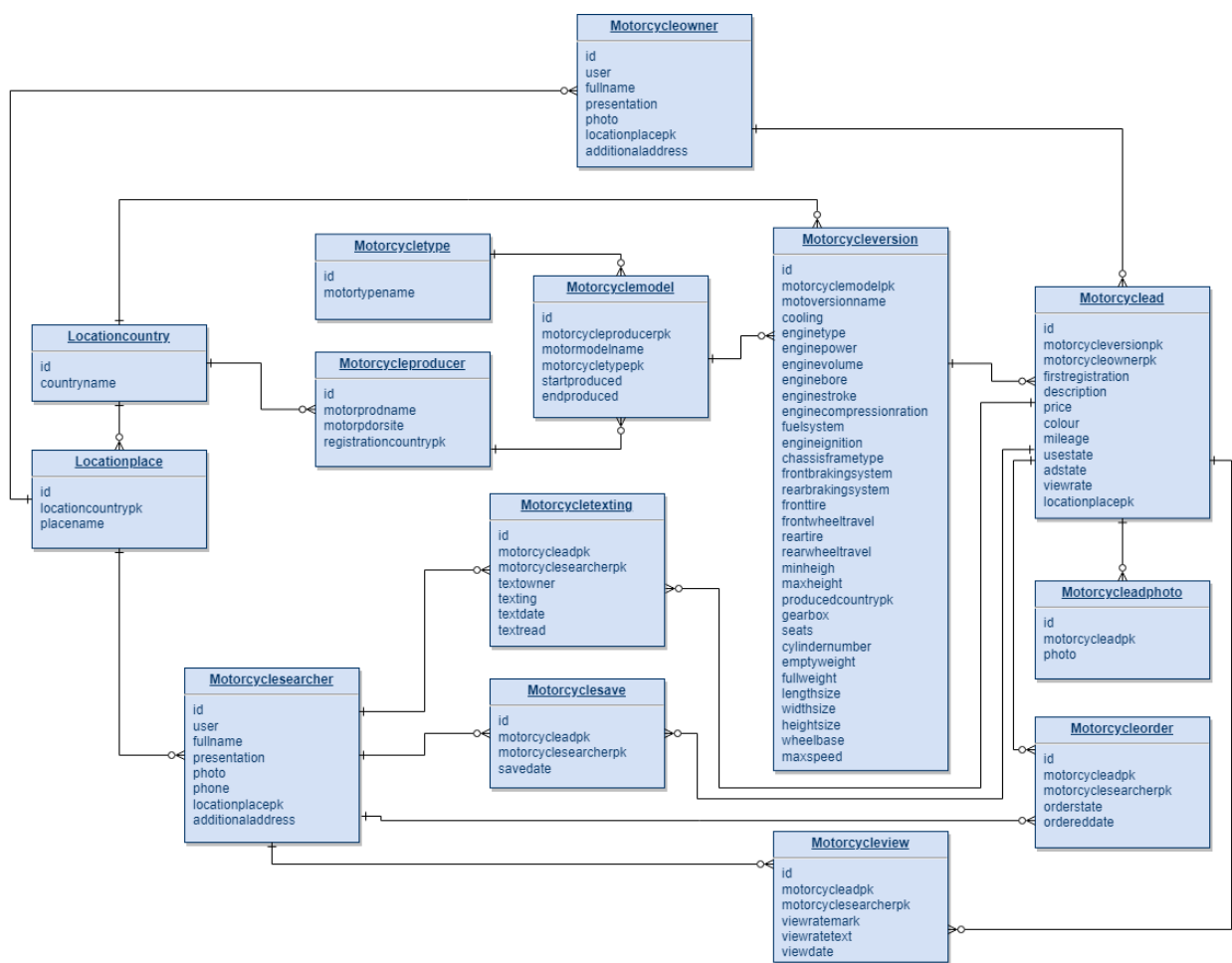


Рисунок 3.4 – Схема бази даних програмного забезпечення підтримки продажу вживаних мотоциклів

3.4 Висновки за розділом 3

У розділі 3 для забезпечення проєктування програмного забезпечення

підтримки продажу вживаних мотоциклів виконано моделювання сценаріїв користувача, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, користувача, який є власником або продавцем вживаних мотоциклів, та менеджера сервісу продажу вживаних мотоциклів, визначено мову і фреймворк розробки як Python та Django, визначено структуру бази даних програмного забезпечення підтримки продажу вживаних мотоциклів.

4 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Загальний порядок роботи з замовленнями перегляду або придбання вживаних мотоциклів

Замовлення перегляду або придбання вживаних мотоциклів було впорядковано за логікою роботи з основними засобами керування за допомогою створення відповідної діаграми станів (рис. 4.1).

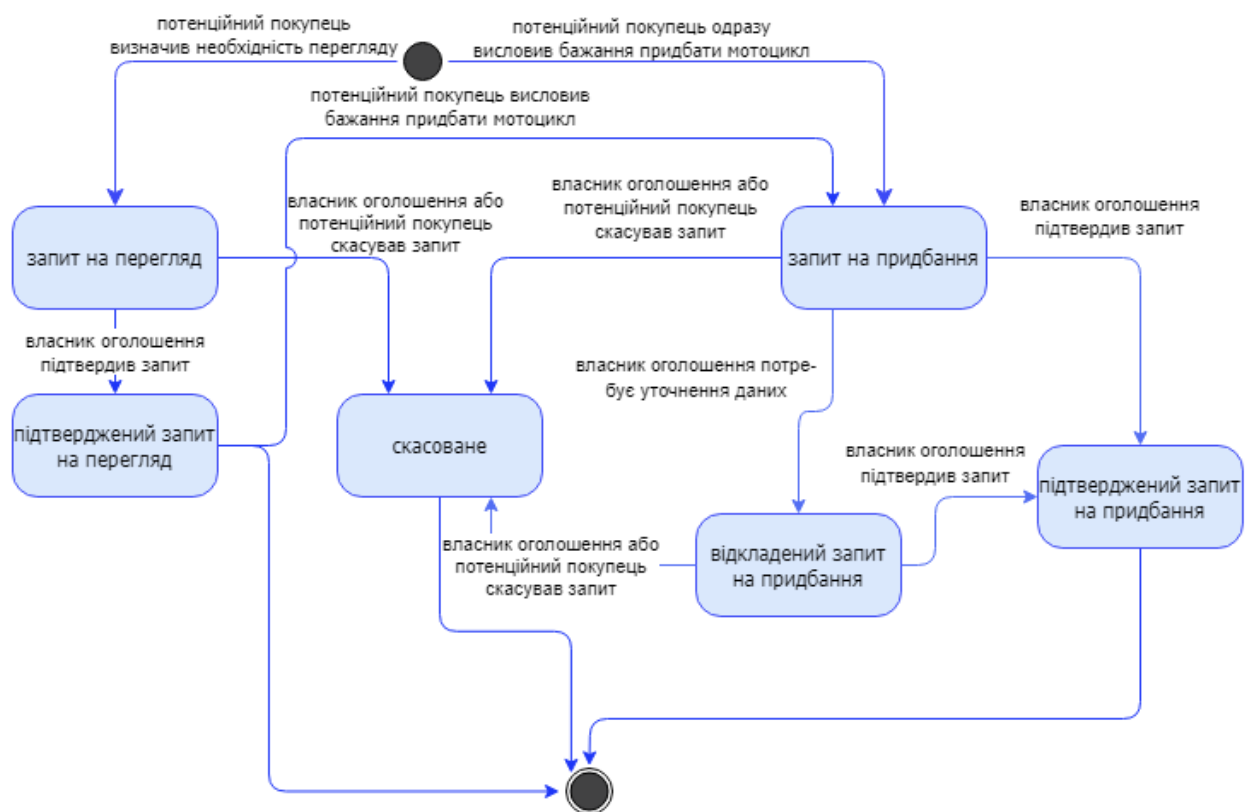


Рисунок 4.1 – Діаграма станів замовлення перегляду або придбання вживаних мотоциклів

Загальна логіка роботи передбачає або одразу запит на придбання, або спочатку запит на перегляд, а тоді за відповідної необхідності ще перехід до придбання.

4.2 Реалізовані програмні одиниці у складі програмного забезпечення підтримки продажу вживаних мотоциклів

Основну частину реалізованих одиниць програмного забезпечення підтримки продажу вживаних мотоциклів описано нижче.

Для реалізації самої роботи з деревами рішень у запропонованому методі прогнозування вартості вживаних мотоциклів було використано клас `DecisionTreeRegressor`, що знаходиться в підпакеті `tree` пакету `scikit-learn`. З цього класу застосовано навчання моделі через виклик та передачу вибірки даних у метод `fit`, а відповідно коли модель вже навчена, то використовується метод `predict`.

Перегляд існуючих версій моделі мотоцикла реалізовано в першу чергу на основі функції представлення `motorcyclemodel_view` з ідентифікатором моделі мотоцикла `motorcycle_model_id`, що є параметром цієї функції. Під час роботи цієї функції використовується шаблон вебсторінки `motorcyclemodel.html`, який було створено на основі шаблону [9], який є безкоштовним для такого застосування. Окрім даних самих версій моделі мотоцикла передаються також у сторінку дані самої моделі мотоцикла та фотографія мотоцикла.

Перегляд даних версії мотоциклу реалізовано на основі функції представлення `motorcycleversion_view`, в якій параметром `motorcycle_version_id` задається ідентифікатор самої версії мотоциклу. Відповідний шаблон вебсторінки має ім'я `motorcycleversion.html`. Передаються дані про саму версію моделі мотоцикла, 5 найкращих активних оголошень про продаж вживаних мотоциклів цієї версії, фотографія.

Перегляд даних виробника мотоциклів реалізовано на основі функції представлення `motorcycleproducer_view`. Параметр функції `motorcycle_producer_id` задає ідентифікатор виробника мотоциклів. Відповідний шаблон вебсторінки має ім'я `motorcycleproducer.html`. Передаються дані про самого виробника мотоциклів, моделі його мотоциклів,

5 найкращих активних оголошень про продаж вживаних мотоциклів цього виробника.

Пошук вживаних мотоциклів реалізовано на основі функції представлення `motorcycles_search_view` на основі принципу застосування різних фільтрів. Фільтри встановлюються до результатів пошуку на основі GET-запиту, включаючи такі:

- мінімальна вартість вживаного мотоцикла `pricemin`;
- максимальна вартість вживаного мотоцикла `pricemax`;
- колір вживаного мотоцикла `colour`;
- мінімальна відстань, яку подолав вживаний мотоцикл, – `mileagemin`;
- максимальна відстань, яку подолав вживаний мотоцикл, – `mileagemax`;
- мінімальна середня оцінка, яку мав отримати вживаний мотоцикл, – `viewratemin`;
- максимальна середня оцінка, яку мав отримати вживаний мотоцикл, – `viewratemax`;
- мінімальний рік першої реєстрації вживаного мотоцикла `regmin`;
- максимальний рік першої реєстрації вживаного мотоцикла `regmax`;
- версія моделі вживаного мотоцикла `version`;
- система охолодження, що використовується у вживаному мотоциклі, – `cooling`;
- тип двигуна, який використовується у вживаному мотоциклі, `enginetype`;
- мінімальна потужність двигуна вживаного мотоцикла `enginepowermin`;
- максимальна потужність двигуна вживаного мотоцикла `enginepowermax`;
- мінімальний обсяг двигуна вживаного мотоцикла `enginevolumemin`;
- максимальний обсяг двигуна вживаного мотоцикла `enginevolumemax`;

- мінімальний діаметр двигуна вживаного мотоцикла engineboremin;
- максимальний діаметр двигуна вживаного мотоцикла engineboremax;
- мінімальний хід двигуна вживаного мотоцикла enginestrokemin;
- максимальний хід двигуна вживаного мотоцикла enginestrokemax;
- мінімальний рівень компресії двигуна вживаного мотоцикла compressmin;
- максимальний рівень компресії двигуна вживаного мотоцикла compressmax;
- система палива, яка використовується у вживаному мотоциклі, – fuelsystem;
- тип запалення, який використовується у вживаному мотоциклі, – engineignition;
- тип рами шасі, який використовується у вживаному мотоциклі, – chassisframetype;
- передня гальмівна система, яка використовується у вживаному мотоциклі, – frontbrake;
- задня гальмівна система, яка використовується у вживаному мотоциклі, – rearbrake;
- гума переднього колеса, яка використовується у вживаному мотоциклі, – fronttire;
- гума заднього колеса, яка використовується у вживаному мотоциклі, – reartire;
- мінімальний хід переднього колеса вживаного мотоцикла frontwheelmin;
- максимальний хід переднього колеса вживаного мотоцикла frontwheelmax;
- мінімальний хід заднього колеса вживаного мотоцикла rearwheelmin;
- максимальний хід заднього колеса вживаного мотоцикла rearwheelmax;
- мінімальна висота посадки вживаного мотоцикла heightmin;

- максимальна висота посадки вживаного мотоцикла heightmax;
- назва країни, в якій виготовлено вживаний мотоцикл, – producedcountry;
- мінімальна кількість місць вживаного мотоцикла seatsmin;
- максимальна кількість місць вживаного мотоцикла seatsmax;
- тип коробки передач вживаного мотоцикла gearbox;
- мінімальна кількість циліндрів двигуна вживаного мотоцикла cylindermin;
- максимальна кількість циліндрів двигуна вживаного мотоцикла cylindermax;
- мінімальна вага вживаного мотоцикла в порожньому стані emptyweightmin;
- максимальна вага вживаного мотоцикла в порожньому стані emptyweightmax;
- мінімальна повна вага вживаного мотоцикла fullweightmin;
- максимальна повна вага вживаного мотоцикла fullweightmax;
- мінімальна довжина вживаного мотоцикла lengthsizein;
- максимальна довжина вживаного мотоцикла lengthsizemax;
- мінімальна ширина вживаного мотоцикла widthsizein;
- максимальна ширина вживаного мотоцикла widthsizemax;
- мінімальна висота вживаного мотоцикла heightsizein;
- максимальна висота вживаного мотоцикла heightsizemax;
- мінімальний розмір колісної бази в метрах wheelbasemin;
- максимальний розмір колісної бази в метрах wheelbasemax;
- потрібний інтервал швидкості вживаного мотоцикла від maxspeedmin до maxspeedmax;
- тип вживаного мотоцикла type;
- ідентифікатор виробника вживаного мотоцикла producer;
- напрямок сортування відібраних вживаних мотоциклів orderway.

Перегляд оголошення про продаж вживаного мотоциклу реалізовано на основі функції представлення `motorcycle_ad_view` з параметром `motorcycle_ad_id`, який задає номер оголошення про продаж вживаного мотоциклу.

Перегляд оголошень про продаж вживаних мотоциклів у користувача реалізовано на основі функції представлення `motorcycle_owner_ads_view` з параметром `owner_username`, який задає логін такого користувача. Відповідно перегляд власних оголошень про продаж вживаних мотоциклів у користувача реалізовано на основі функції представлення `own_motorcycle_owner_ads_view`.

Перегляд відгуків про вживаний мотоцикл реалізовано на основі функції представлення `motorcycle_views_view`, а параметром `motorcycle_ad_id` задається ідентифікатор оголошення про продаж вживаного мотоцикла, якого стосується набір відгуків.

Перегляд відгуків про вживані мотоцикли, створених користувачем, реалізовано на основі функції представлення `motor_searcher_views_view`, а параметром `searcher_username` задається логін такого користувача. Відповідно перегляд власних відгуків про вживані мотоцикли, створених користувачем реалізовано на основі функції представлення `own_motor_searcher_views_view`.

Збереження оголошення про продаж вживаного мотоциклу та скасування збереження оголошення про продаж вживаного мотоциклу в залежності від наявності або відсутності в базі даних такого збереження реалізовано функцією представлення `motorcycle_save_view`, а параметр `motorcycle_ad_id` задає номер оголошення про продаж вживаного мотоциклу.

Перегляд замовлень за оголошенням про продаж вживаного мотоцикла реалізовано на основі функції представлення `motorcyclelead_orders_view`, а параметр `motorcycle_ad_id` задає номер оголошення про продаж вживаного мотоциклу. Дані надаються тільки власнику оголошення або представнику сервісу з продажу вживаних мотоциклів.

Клас `Motorcycletype` забезпечує реалізацію моделі типів мотоциклів. Поле `motortypename` цього класу задає назву типу мотоцикла на основі класу `CharField`.

Клас `Motorcycleproducer` забезпечує реалізацію моделі виробників мотоциклів.

Поле `motorprodname` цього класу задає назву виробника мотоциклів на основі класу `CharField`.

Поле `motorpdorsite` цього класу задає вебсайт виробника мотоциклів на основі класу `URLField`.

Поле `registrationcountrypk` цього класу задає країну, до якої належить цей виробник мотоциклів, на основі класу `ForeignKey` за допомогою моделі `Locationcountry`.

Клас `Motorcyclemodel` забезпечує реалізацію моделі моделей мотоциклів.

Поле `motorcycleproducerpk` цього класу задає виробника цієї моделі мотоцикла, на основі класу `ForeignKey` за допомогою моделі `Motorcycleproducer`.

Поле `motormodelname` цього класу задає назву моделі мотоцикла на основі класу `CharField`.

Поле `motorcycletypepk` цього класу задає тип мотоцикла, до якого належить даний, на основі класу `ForeignKey` за допомогою моделі `Motorcycletype`.

Поле `startproduced` цього класу задає рік, з якого почалось виготовлення мотоциклів цієї моделі виробником, на основі класу `PositiveSmallIntegerField`.

Поле `endproduced` цього класу задає рік, коли закінчилось виготовлення мотоциклів цієї моделі виробником, на основі класу `PositiveSmallIntegerField`.

Клас `Motorcycleversion` забезпечує реалізацію моделі різних версій моделей мотоциклів.

Поле `motorcyclemodelpk` цього класу задає модель мотоцикла, версія якої представлена, на основі класу `ForeignKey` за допомогою моделі `Motorcyclemodel`.

Поле `motoversionname` цього класу задає назву версії моделі мотоцикла на основі класу `CharField`.

Поле `cooling` цього класу задає систему охолодження, яка використовується в цій версії моделі мотоцикла, на основі класу `CharField`.

Поле `enginetype` цього класу задає тип двигуна, який використовується в цій версії моделі мотоцикла, на основі класу `CharField`.

Поле `enginepower` цього класу задає потужність двигуна, який використовується в цій версії моделі мотоцикла, на основі класу `DecimalField`.

Поле `enginevolume` цього класу задає обсяг двигуна, який використовується в цій версії моделі мотоцикла, на основі класу `DecimalField`.

Поле `enginebore` цього класу задає діаметр двигуна, який використовується в цій версії моделі мотоцикла, на основі класу `DecimalField`.

Поле `enginestroke` цього класу задає хід двигуна, який використовується в цій версії моделі мотоцикла, на основі класу `DecimalField`.

Поле `enginecompressionration` цього класу задає рівень компресії двигуна, який використовується в цій версії моделі мотоцикла, на основі класу `DecimalField`.

Поле `fuelsystem` цього класу задає систему палива, яка використовується в цій версії моделі мотоцикла, на основі класу `CharField`.

Поле `engineignition` цього класу задає тип запалення, який використовується в цій версії моделі мотоцикла, на основі класу `CharField`.

Поле `chassisframetype` цього класу задає тип рами шасі, який використовується в цій версії моделі мотоцикла, на основі класу `CharField`.

Поле `frontbrakingsystem` цього класу задає передню гальмівну систему, яка використовується в цій версії моделі мотоцикла, на основі класу `CharField`.

Поле `rearbrakingsystem` цього класу задає задню гальмівну систему, яка використовується в цій версії моделі мотоцикла, на основі класу `CharField`.

Поле `fronttire` цього класу задає гуму переднього колеса, яка використовується в цій версії моделі мотоцикла, на основі класу `CharField`.

Поле `frontwheeltravel` цього класу задає хід переднього колеса, який використовується в цій версії моделі мотоцикла, на основі класу `DecimalField`.

Поле `reartire` цього класу задає гуму заднього колеса, яка використовується в цій версії моделі мотоцикла, на основі класу `CharField`.

Поле `rearwheeltravel` цього класу задає хід заднього колеса, який використовується в цій версії моделі мотоцикла, на основі класу `DecimalField`.

Поле `minheigh` цього класу задає мінімальну висоту посадки, яка доступна в цій версії моделі мотоцикла, на основі класу `DecimalField`.

Поле `maxheight` цього класу задає максимальну висоту посадки, яка доступна в цій версії моделі мотоцикла, на основі класу `DecimalField`.

Поле `producedcountrypk` цього класу задає країну, в якій виготовлялась ця версія моделі мотоцикла, на основі класу `ForeignKey` за допомогою моделі `Locationcountry`.

Поле `gearbox` цього класу задає тип коробки передач, який використовувався в цій версії моделі мотоцикла, на основі класу `CharField`.

Поле `seats` цього класу задає кількість місць у цій версії моделі мотоцикла на основі класу `PositiveSmallIntegerField`.

Поле `cylindernumber` цього класу задає кількість циліндрів двигуна, який використовується в цій версії моделі мотоцикла, на основі класу `PositiveSmallIntegerField`.

Поле `emptyweight` цього класу задає вагу цієї версії моделі мотоцикла в порожньому стані на основі класу `DecimalField`.

Поле `fullweight` цього класу задає повну вагу цієї версії моделі мотоцикла на основі класу `DecimalField`.

Поле `lengthsize` цього класу задає максимальну довжину цієї версії моделі мотоцикла в метрах на основі класу `DecimalField`.

Поле `widthsize` цього класу задає максимальну ширину цієї версії моделі мотоцикла в метрах на основі класу `DecimalField`.

Поле `heightsize` цього класу задає максимальну висоту цієї версії моделі мотоцикла в метрах на основі класу `DecimalField`.

Поле `wheelbase` цього класу задає розмір колісної бази в метрах на основі класу `DecimalField`.

Поле `maxspeed` цього класу задає максимальну швидкість, яку може розвивати ця версія мотоцикла в кілометрах на годину, на основі класу `DecimalField`.

Клас `Motorcyclead` забезпечує реалізацію моделі оголошень про продаж вживаних мотоциклів.

Поле `motorcycleversionpk` цього класу задає версію моделі вживаного мотоцикла, представленого в оголошенні, на основі класу `ForeignKey` за допомогою моделі `Motorcycleversion`.

Поле `motorcycleownerpk` цього класу задає власника вживаного мотоцикла, який представлений в оголошенні, або його продавця, на основі класу `ForeignKey` за допомогою моделі `Motorcycleowner`.

Поле `firstregistration` цього класу задає дату першої реєстрації вживаного мотоцикла з оголошення на основі класу `DateField`.

Поле `price` цього класу задає вартість вживаного мотоцикла з оголошення, за яку готовий продати його власник або продавець, на основі класу `DecimalField`.

Поле `colour` цього класу задає колір вживаного мотоцикла на основі класу `CharField`.

Поле `mileage` цього класу задає відстань в кілометрах, які подолав цей вживаний мотоцикл на момент продажу, на основі класу `DecimalField`.

Поле `usestate` цього класу задає стан вживаного мотоцикла на основі класу `CharField`.

Поле `adstate` цього класу задає стан оголошення про продаж вживаного мотоцикла на основі класу `CharField`.

Поле `viewrate` цього класу задає середню оцінку, що була виставлена вживаному мотоциклу користувачами, на основі класу `DecimalField`.

Поле `locationplacepk` цього класу задає місто, де знаходиться вживаний мотоцикл, на основі класу `ForeignKey` за допомогою моделі `Locationplace`.

Поле `description` цього класу задає опис вживаного мотоцикла з оголошення на основі класу `TextField`.

Клас `Motorcycleadphoto` забезпечує реалізацію моделі фотографій вживаних мотоциклів з оголошень.

Поле `motorcycleadpk` цього класу задає оголошення про продаж вживаного мотоцикла, до якого відноситься ця фотографія, на основі класу `ForeignKey` за допомогою моделі `Motorcyclead`.

Поле `photo` цього класу задає фотографію вживаного мотоцикла з оголошення на основі класу `ImageField`.

Клас `Motorcycleowner` забезпечує реалізацію моделі користувача, який є власником або продавцем вживаного мотоцикла або мотоциклів.

Поле `user` цього класу задає користувацький запис в Django, що використовується цим власником або продавцем вживаного мотоцикла або мотоциклів, на основі класу `ForeignKey` за допомогою моделі `User`.

Поле `fullname` цього класу задає повне ім'я власника або продавця вживаного мотоцикла або мотоциклів на основі класу `CharField`.

Поле `presentation` цього класу задає опис власника або продавця вживаного мотоцикла або мотоциклів на основі класу `TextField`.

Поле `photo` цього класу задає фотографію власника або продавця вживаного мотоцикла або мотоциклів на основі класу `ImageField`.

Поле `locationplacepk` цього класу задає місце (місто), де знаходиться продавець або власник вживаного мотоцикла або мотоциклів і де відповідно знаходяться його мотоцикли для продажу, на основі класу `ForeignKey` за допомогою моделі `Locationplace`.

Поле `additionaladdress` цього класу задає повну адресу в зазначеному місті, за якою знаходяться мотоцикли цього власника або продавця, на основі класу `CharField`.

Клас `Motorcyclesearcher` забезпечує реалізацію моделі користувача, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем.

Поле `user` цього класу задає користувацький запис в Django, що використовується цим користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, на основі класу `ForeignKey` за допомогою моделі `User`.

Поле `fullname` цього класу задає повне ім'я користувача, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, на основі класу `CharField`.

Поле `presentation` цього класу задає опис користувача, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, на основі класу `TextField`.

Поле `photo` цього класу задає фотографію користувача, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, на основі класу `ImageField`.

Поле `phone` цього класу задає номер телефону користувача, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, на основі класу `CharField`.

Поле `locationplacepk` цього класу задає місце (місто), де знаходиться користувач, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, на основі класу `ForeignKey` за допомогою моделі `Locationplace`.

Поле `additionaladdress` цього класу задає повну адресу в зазначеному місті, за якою знаходиться користувач, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, на основі класу `CharField`.

Клас `Motorcycleorder` забезпечує реалізацію моделі замовлення вживаного мотоцикла для придбання.

Поле `motorcycleadpk` цього класу задає оголошення про продаж вживаного мотоцикла, якого стосується замовлення, на основі класу `ForeignKey` за допомогою моделі `Motorcyclead`.

Поле `motorcyclesearcherpk` цього класу задає користувача, який створив замовлення на придбання або перегляд вживаного мотоцикла, на основі класу `ForeignKey` за допомогою моделі `Motorcyclesearcher`.

Поле `orderstate` цього класу задає стан замовлення вживаного мотоцикла для придбання на основі класу `CharField`.

Поле `orderdate` цього класу задає час замовлення вживаного мотоцикла для придбання на основі класу `DateTimeField`.

Клас `Motorcycleview` забезпечує реалізацію моделі відгуків про вживані мотоцикли.

Поле `motorcycleadpk` цього класу задає оголошення про продаж вживаного мотоцикла, за яким залишив свій відгук користувач, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, на основі класу `ForeignKey` за допомогою моделі `Motorcyclead`.

Поле `motorcyclesearcherpk` цього класу задає користувача, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, що залишив відгук про вживаний мотоцикл, на основі класу `ForeignKey` за допомогою моделі `Motorcyclesearcher`.

Поле `viewratemark` цього класу задає виставлену оцінку (значення) у відгуку про вживаний мотоцикл користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, на основі класу `PositiveSmallIntegerField`.

Поле `viewratetext` цього класу задає текст відгуку про вживаний мотоцикл користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, на основі класу `CharField`.

Поле `viewdate` цього класу задає час створення відгуку про вживаний мотоцикл користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, на основі класу `DateTimeField`.

Клас `Motorcyclesave` забезпечує реалізацію моделі збереження оголошення про продаж вживаного мотоцикла.

Поле `motorcycleadpk` цього класу задає оголошення про продаж вживаного мотоцикла, яке було збережено користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, на основі класу `ForeignKey` за допомогою моделі `Motorcyclead`.

Поле `motorcyclesearcherpk` цього класу задає користувача, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, що зберіг оголошення про продаж вживаного мотоцикла, на основі класу `ForeignKey` за допомогою моделі `Motorcyclesearcher`.

Поле `savedate` цього класу задає час збереження оголошення про продаж вживаного мотоцикла на основі класу `DateTimeField`.

Клас `Locationcountry` забезпечує реалізацію моделі країн, в яких можуть знаходитись вживані мотоцикли та їхні власники.

Поле `countryname` цього класу задає назву країни, в якій може знаходитись вживаний мотоцикл та його власник, на основі класу `CharField`.

Клас `Locationplace` забезпечує реалізацію моделі міст (місць), в яких може знаходитись вживаний мотоцикл та його власник.

Поле `locationcountrypk` цього класу задає країну, в якій знаходиться місто (місце), в якому може знаходитись вживаний мотоцикл та його власник, на основі класу `ForeignKey` за допомогою моделі `Locationcountry`.

Поле `placename` цього класу задає назву міста (місця), в якому може знаходитись вживаний мотоцикл та його власник, на основі класу `CharField`.

Клас `Motorcycletexting` забезпечує реалізацію моделі повідомлень, надісланих між користувачами, які виконують пошук вживаного мотоцикла для придбання і є потенційними покупцями, і користувачами, які є власниками або продавцями вживаних мотоциклів.

Поле `motorcycleadpk` цього класу задає оголошення про продаж вживаного мотоцикла, за яким було відправлене повідомлення, надіслане між користувачем, який виконує пошук вживаного мотоцикла для придбання і є

потенційним покупцем, і користувачем, який є власником або продавцем вживаних мотоциклів, на основі класу `ForeignKey` за допомогою моделі `Motorcyclead`.

Поле `motorcyclesearcherpk` цього класу задає користувача, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, який створив або якому направлено повідомлення, надіслане між користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, і користувачем, який є власником або продавцем вживаних мотоциклів, на основі класу `ForeignKey` за допомогою моделі `Motorcyclesearcher`.

Поле `textowner` цього класу задає власництво відправленого повідомлення, тобто це повідомлення, створене користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, або повідомлення, створене користувачем, який є власником або продавцем вживаних мотоциклів, на основі класу `BooleanField`.

Поле `texting` цього класу задає текст повідомлення, надісланого між користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, і користувачем, який є власником або продавцем вживаних мотоциклів, на основі класу `CharField`.

Поле `textdate` цього класу задає час відправлення повідомлення, надісланого між користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, і користувачем, який є власником або продавцем вживаних мотоциклів, на основі класу `DateTimeField`.

Поле `textread` цього класу задає позначку про прочитання повідомлення, надісланого між користувачем, який виконує пошук вживаного мотоцикла для придбання і є потенційним покупцем, і користувачем, який є власником або продавцем вживаних мотоциклів, стороною, якій повідомлення адресоване, на основі класу `BooleanField`.

4.3 Висновки за розділом 4

У розділі 4 для забезпечення реалізації програмного забезпечення підтримки продажу вживаних мотоциклів виконано представлення можливих станів замовлення покупки вживаного мотоцикла, опис реалізованих одиниць програмного застосунку підтримки продажу вживаних мотоциклів, включаючи зокрема моделі, представлення.

5 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ
ДОСЛІДЖЕННЯ ПРОГРАМИ

5.1 Основне призначення програми

Програма призначена для забезпечення підтримки продажу вживаних мотоциклів шляхом надання відповідних засобів для виставлення власних вживаних мотоциклів на продаж, пошуку вживаних мотоциклів для придбання.

5.2 Виконання програми

При зверненні користувача-покупця до програми йому надається на сторінках вебзастосунку меню, що включає зокрема пункт для пошуку вживаних мотоциклів у розділі «Шукати» (рис. 5.1).

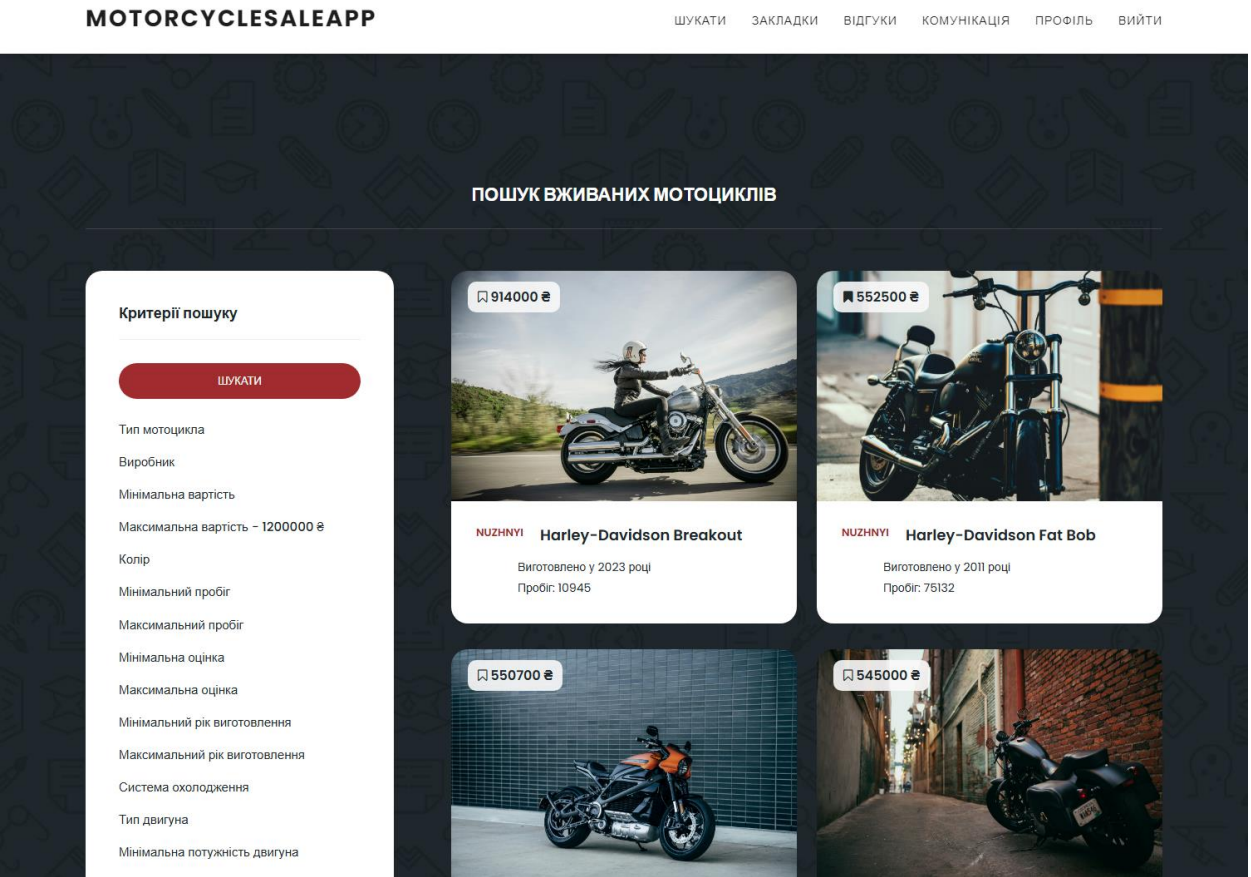


Рисунок 5.1 – Пошук вживаних мотоциклів

Повний перелік параметрів, які використовуються під час реалізації пошуку вживаних мотоциклів:

- мінімальна вартість вживаного мотоцикла;
- максимальна вартість вживаного мотоцикла;
- колір вживаного мотоцикла;
- мінімальна відстань, яку подолав вживаний мотоцикл;
- максимальна відстань, яку подолав вживаний мотоцикл;
- мінімальна середня оцінка, яку мав отримати вживаний мотоцикл;
- максимальна середня оцінка, яку мав отримати вживаний мотоцикл;
- мінімальний рік першої реєстрації вживаного мотоцикла;
- максимальний рік першої реєстрації вживаного мотоцикла;
- версія моделі вживаного мотоцикла;
- система охолодження, що використовується у вживаному мотоциклі;
- тип двигуна, який використовується у вживаному мотоциклі;
- мінімальна потужність двигуна вживаного мотоцикла;
- максимальна потужність двигуна вживаного мотоцикла;
- мінімальний обсяг двигуна вживаного мотоцикла;
- максимальний обсяг двигуна вживаного мотоцикла;
- мінімальний діаметр двигуна вживаного мотоцикла;
- максимальний діаметр двигуна вживаного мотоцикла;
- мінімальний хід двигуна вживаного мотоцикла;
- максимальний хід двигуна вживаного мотоцикла;
- мінімальний рівень компресії двигуна вживаного мотоцикла;
- максимальний рівень компресії двигуна вживаного мотоцикла;
- система палива, яка використовується у вживаному мотоциклі;
- тип запалення, який використовується у вживаному мотоциклі;
- тип рами шасі, який використовується у вживаному мотоциклі;
- передня гальмівна система, яка використовується у вживаному мотоциклі;

- задня гальмівна система, яка використовується у вживаному мотоциклі;
- гума переднього колеса, яка використовується у вживаному мотоциклі;
- гума заднього колеса, яка використовується у вживаному мотоциклі;
- мінімальний хід переднього колеса вживаного мотоцикла;
- максимальний хід переднього колеса вживаного мотоцикла;
- мінімальний хід заднього колеса вживаного мотоцикла;
- максимальний хід заднього колеса вживаного мотоцикла;
- мінімальна висота посадки вживаного мотоцикла;
- максимальна висота посадки вживаного мотоцикла;
- назва країни, в якій виготовлено вживаний мотоцикл;
- мінімальна кількість місць вживаного мотоцикла;
- максимальна кількість місць вживаного мотоцикла;
- тип коробки передач вживаного мотоцикла;
- мінімальна кількість циліндрів двигуна вживаного мотоцикла;
- максимальна кількість циліндрів двигуна вживаного мотоцикла;
- мінімальна вага вживаного мотоцикла в порожньому стані;
- максимальна вага вживаного мотоцикла в порожньому стані;
- мінімальна повна вага вживаного мотоцикла;
- максимальна повна вага вживаного мотоцикла;
- мінімальна довжина вживаного мотоцикла;
- максимальна довжина вживаного мотоцикла;
- мінімальна ширина вживаного мотоцикла;
- максимальна ширина вживаного мотоцикла;
- мінімальна висота вживаного мотоцикла;
- максимальна висота вживаного мотоцикла;
- мінімальний розмір колісної бази в метрах;
- максимальний розмір колісної бази в метрах;
- потрібний інтервал швидкості вживаного мотоцикла;

- тип вживаного мотоцикла;
- ідентифікатор виробника вживаного мотоцикла;
- напрямок сортування відібраних вживаних мотоциклів (рис. 5.1).

Під час пошуку кожен знайдений вживаний мотоцикл містить основну фотографію, вартість, назву виробника та моделі, рік виготовлення мотоцикла, пробіг мотоцикла та можливість збереження вживаного мотоцикла користувачем. Можна відкрити сторінку самого оголошення (рис. 5.2).

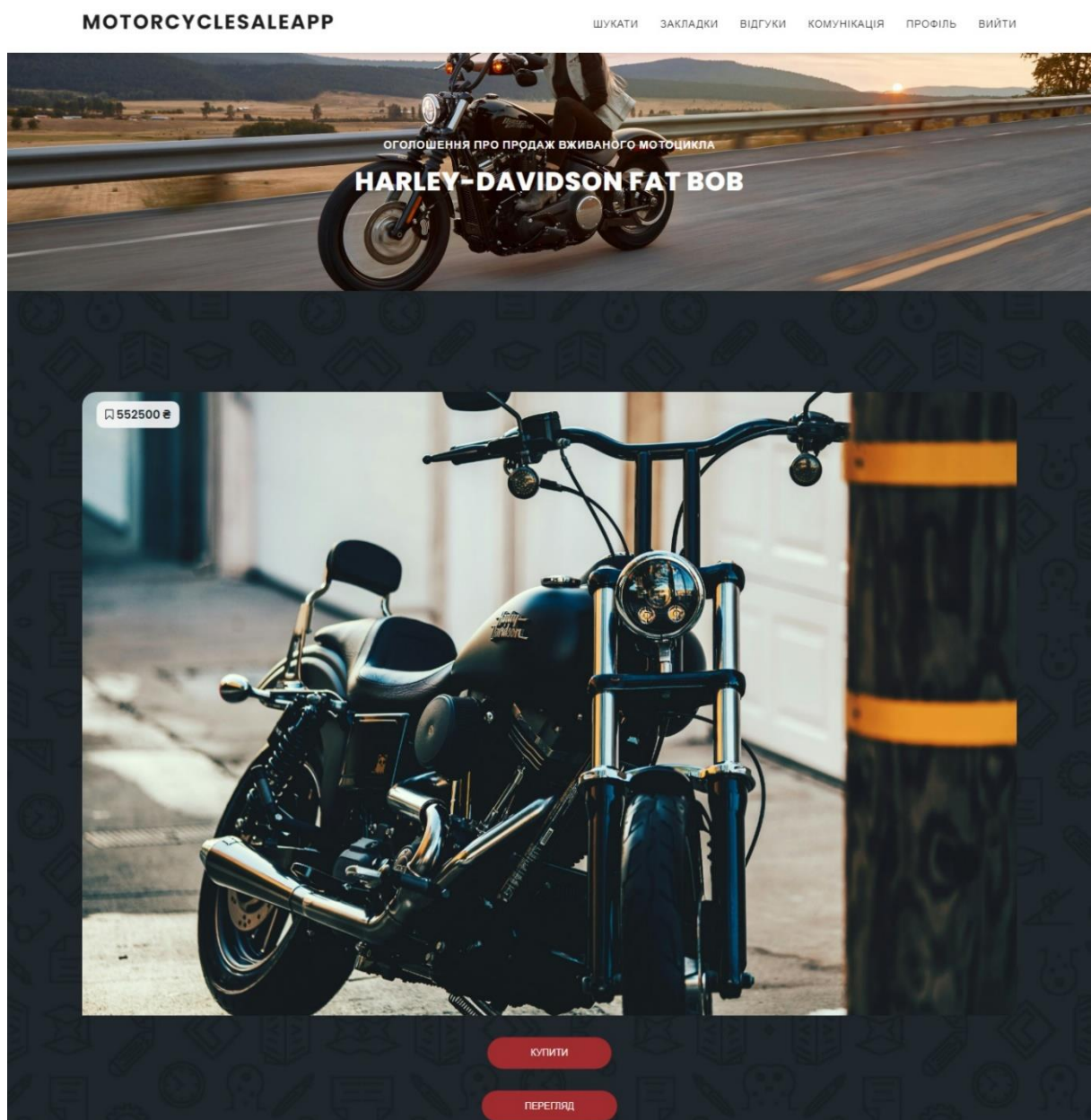


Рисунок 5.2 – Перегляд оголошення про продаж вживаного мотоцикла користувачем-покупцем

При зверненні користувача-покупця до програми йому надається на сторінках вебзастосунку меню, що включає пункти для пошуку вживаних мотоциклів (розділ «Шукати»), перегляду збережених вживаних мотоциклів (розділ «Закладки»), перегляду створених самостійно відгуків про вживані мотоцикли (розділ «Відгуки»), перегляду написаних повідомлень у комунікаціях користувача (розділ «Комунікація»), перегляду та оновлення профілю користувача (розділ «Профіль»).

Сторінка оголошення про продаж вживаного мотоцикла для користувача-покупця містить перелік характеристик оголошення, відгуки, якщо вони були створені, а для користувача-продавця, якщо це оголошення, яке створене цим користувачем, така сторінка містить замість кнопок для покупки та організації перегляду кнопки для переходу до повідомлень від користувачів-покупців та замовлень (рис. 5.3).

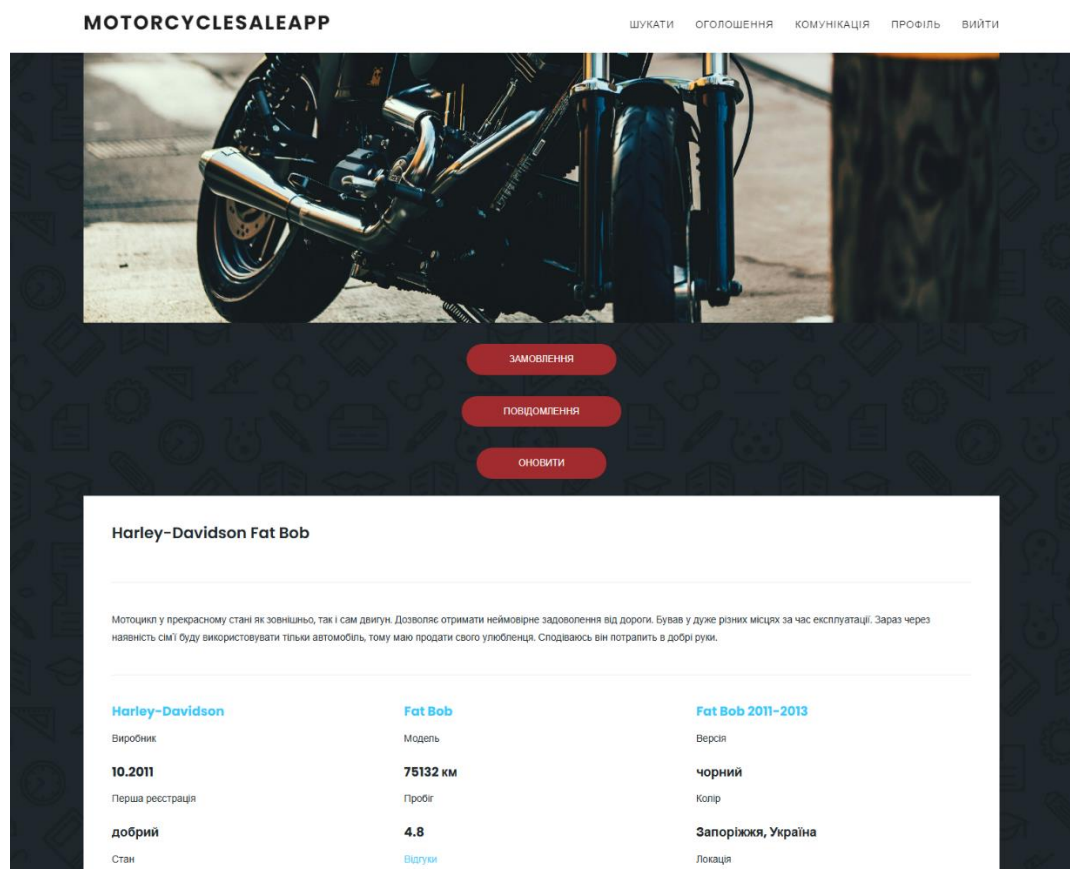


Рисунок 5.3 – Перегляд оголошення про продаж вживаного мотоцикла користувачем-власником

При зверненні користувача-продавця до програми йому надається на сторінках вебзастосунку меню, що включає пункти для пошуку вживаних мотоциклів (розділ «Шукати»), перегляду власних оголошень про продаж вживаних мотоциклів (розділ «Оголошення»), перегляду написаних повідомлень у комунікаціях користувача (розділ «Комунікація»), перегляду та оновлення профілю користувача (розділ «Профіль»).

З розділу оголошень можна не тільки переглянути власні оголошення про продаж вживаного мотоциклу, але і натиснути кнопку для публікації оголошення про продаж вживаного мотоцикла (рис. 5.4). На цій сторінці доступна кнопка «Запропонувати вартість», яка дозволяє автоматично внести в поле з вартістю обчислене значення на основі запропонованого методу. Далі користувач може або підтвердити внесені автоматично значення, або змінити його потрібним чином, після чого натиснути кнопку для публікації оголошення.

The screenshot displays the 'MOTORCYCLESaleApp' interface. At the top, a navigation bar includes links for 'ШУКАТИ', 'ОГОЛОШЕННЯ', 'КОМУНІКАЦІЯ', 'ПРОФІЛЬ', and 'ВИЙТИ'. The main content area is titled 'ВАШЕ НОВЕ ОГОЛОШЕННЯ ПРО ПРОДАЖ ВЖИВАНОГО МОТОЦИКЛА'. It contains a form with the following fields and controls:

- Виробник:** Honda NC 750X 2022 (dropdown)
- Модель:** NC 750X (dropdown)
- Версія:** NC 750X 2022 (dropdown)
- Колір:** чорний (dropdown)
- Стан:** відмінний (dropdown)
- Країна:** Україна (dropdown)
- Місто:** Запоріжжя (dropdown)
- Перша реєстрація:** 01.04.2022 (text input)
- Пробіг, км:** 11000 (text input)
- Вартість, грн:** 420000 (text input)
- Опис:** Ваш опис... (text area)

Buttons and actions include:

- ДОДАТИ ФОТОГРАФІЮ** (red button)
- ЗАПРОПОНУВАТИ ВАРТІСТЬ** (red button)
- ОПУБЛІКУВАТИ** (red button)

Рисунок 5.4 – Публікація оголошення про продаж вживаного мотоцикла

5.3 Експериментальне дослідження прогнозування вартості вживаних мотоциклів

Експериментальне дослідження прогнозування вартості вживаних мотоциклів було реалізовано на основі вибірки даних [10], де дані зібрані з платформи Autoscout, яка є однією з найвідоміших, що відповідно сприяє проведенню дослідження в реальних практичних умовах.

Для проведення експериментального дослідження було використано запропонований метод прогнозування вартості вживаних мотоциклів з використанням декількох варіацій, відмінність в яких полягає в тому, скільки саме вхідних ознак використовувалось при побудові дерева рішень в основі прогнозування. Ознаки відбирались на основі визначення інформативності цих ознак за допомогою окремого дерева рішень. Відповідно у випадку, коли обрано було 3 ознаки, зі всього набору вхідних ознак було відібрано тільки 3 ознаки з найбільшою інформативністю та використано як вхідні дані моделі. Альтернативно до дерев рішень було також розглянуто регресійну модель на основі методу k-найближчих сусідів та метод опорних векторів.

Порівняння описаних моделей виконувалось на основі двох показників: середньої абсолютної похибки (САП) та середньої квадратичної похибки (СКП).

Отримані результати прогнозування вартості вживаних мотоциклів вказують, що значне покращення відбулось за використання дерев рішень за запропонованим методом на основі показника САП, за якого результати методу опорних векторів, виявились в 269 разів гіршими, а результати регресійної моделі на основі методу k-найближчих сусідів були покращені на 11,69 % (табл. 5.1). Також певне покращення відбулось і за показником СКП у підсумку. Все це разом зі значенням похибок демонструє, що запропонований метод прогнозування вартості вживаних мотоциклів можна використовувати на практиці.

Таблиця 5.1 – Результати прогнозування вартості вживаних мотоциклів

Модель прогнозування	САП	СКП
Дерево рішень зі всіма ознаками	0,0002566078692600107	0,00024789303370922357
Дерево рішень з 4 ознаками	0,0002581239572394642	0,00024789895139848823
Запропонований метод (дерево рішень з 3 ознаками)	0,0002563973463753253	0,0002478683712292277
Регресійна модель на основі методу k-найближчих сусідів	0,00029035257843259643	0,00024834775563353
Метод опорних векторів	0,06897435748210325	0,004941252562452996

5.4 Висновки за розділом 5

Реалізовані та описані в попередньому розділі функціональні можливості програмного забезпечення підтримки продажу вживаних мотоциклів представлені у вигляді порядку роботи з програмою для користувачів. Виконано експериментальне дослідження прогнозування вартості вживаних мотоциклів, що підтвердило доцільність практичного застосування методу, який був запропонований.

ВИСНОВКИ

У результаті виконання роботи реалізована розробка програмного забезпечення підтримки продажу вживаних мотоциклів для забезпечення зручності потенційних продавців під час публікації оголошень про продаж вживаних мотоциклів на основі засобів прогнозування вартості. Це дозволило створити платформу для підтримки продажу вживаних мотоциклів від власників або продавців, які мають одну або декілька одиниць.

Реалізацію програмного забезпечення підтримки продажу вживаних мотоциклів виконано мовою програмування Python з застосуванням фреймворку Django та на основі підтримки системи управління базами даних MySQL.

Створена програма призначена для забезпечення підтримки продажу вживаних мотоциклів шляхом надання відповідних засобів для виставлення власних вживаних мотоциклів на продаж, пошуку вживаних мотоциклів для придбання. Мотоцикли представлені виробниками, типами, моделями, версіями цих моделей, що визначаються представниками сервісу продажу вживаних мотоциклів, а вже на основі цих даних визначаються оголошення про продаж вживаних мотоциклів самими власниками.

Наукова новизна роботи полягає в методі прогнозування вартості вживаних мотоциклів, який базується на створенні та навчанні дерева рішень, визначає процес створення і відбору вхідних ознак, що дозволяє встановити вартість вживаного мотоцикла на етапі формування оголошення. Проведено експериментальне дослідження прогнозування вартості вживаних мотоциклів на основі відкритої вибірки даних з порівнянням запропонованого методу і відповідно дерев рішень з методом опорних векторів та регресійною моделлю на основі методу k-найближчих сусідів, що в результаті продемонструвало меншу похибку прогнозування у дерев рішень та запропонованого методу зокрема.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Used Bike Market Sales, Trends, Size, Analysis, Share, Demand [Electronic resource]. – Access mode : <https://www.alliedmarketresearch.com/used-bike-market-A09641>.
2. Used Motorcycle Store. We Ship Nationwide [Electronic resource]. – Access mode : <https://www.usedmotorcyclestore.com/>.
3. Used Motorcycle For Sale – Buy and Sell Motorcycles Australia [Electronic resource]. – Access mode : <https://www.bikesales.com.au/>.
4. New and Used cars marketplace AutoScout24 [Electronic resource]. – Access mode : <https://www.autoscout24.com/>.
5. Інтернет магазин MotoLux. Купити в онлайн магазині Мотолукс з доставкою по Україні. Кращий інтернет магазин в Закарпатській області [Електрон. ресурс]. – Режим доступу : <https://motolux.ua/>.
6. Erden, K. Decision Trees in Machine Learning [Electronic resource] : CART and Advanced Trees / K. Erden. – Access mode : <https://medium.com/@kaanerdenn/decision-trees-in-machine-learning-cart-and-advanced-trees-8b3fe375e9f9>.
7. Django introduction – Learn web development. MDN [Electronic resource]. – Access mode : https://developer.mozilla.org/en-US/docs/Learn_web_development/Extensions/Server-side/Django/Introduction.
8. The Pyramid Web Framework – The Pyramid Web Framework v2.0.2 [Electronic resource]. – Access mode : <https://docs.pylonsproject.org/projects/pyramid/en/2.0-branch/>.
9. Free Template 569 Edu Meeting [Electronic resource]. – Access mode : <https://templatemo.com/tm-569-edu-meeting>.
10. Motorbike Marketplace [Electronic resource]. – Access mode : <https://www.kaggle.com/datasets/mexwell/motorbike-marketplace>.

ДОДАТОК А
Технічне завдання

Вступ

Нерідко сучасне покоління молодих людей, а також і сучасні люди загалом надають перевагу вживаним мотоциклам замість нових, адже це дозволяє зменшити витрати, що рухає цим напрямком. Відповідно створення програмного забезпечення підтримки продажу вживаних мотоциклів, яке б дозволяло індивідуальним власникам вживаних мотоциклів виставляти їх на продаж, при цьому підтримуючи таким чином розширення ринку на основі надання їм відповідного майданчика, а також забезпечення таких користувачів засобами прогнозування для підтримки зокрема на етапі публікації оголошення є важливим.

A.1 Підстави для розробки

Наказом № 398 від 7 жовтня 2024 року за Національним університетом «Запорізька політехніка» визначена тема дипломної кваліфікаційної роботи магістра «Дослідження та програмна реалізація методів підтримки продажу вживаних мотоциклів», що обумовило отримання завдання на роботу.

A.2 Призначення розробки

Розробка програми призначена для забезпечення підтримки продажу вживаних мотоциклів шляхом надання відповідних засобів для виставлення власних вживаних мотоциклів на продаж, пошуку вживаних мотоциклів для придбання.

A.3 Основні вимоги до програми

A.3.1 Вимоги до функціональних характеристик

У роботі шляхом реалізації програмного забезпечення необхідно

надати наступні функціональні можливості:

- додавання версій мотоциклів менеджером сервісу;
- перегляд даних версії мотоциклу;
- перегляд існуючих версій моделі мотоцикла;
- пошук вживаних мотоциклів;
- перегляд даних виробника мотоциклів;
- оновлення даних версії мотоцикла;
- публікація оголошення про продаж вживаного мотоциклу;
- оновлення оголошення про продаж вживаного мотоциклу;
- перегляд оголошення про продаж вживаного мотоциклу;
- перегляд оголошень про продаж вживаних мотоциклів у користувача;
- реєстрація користувача як власника вживаних мотоциклів та авторизація;
- реєстрація користувача як покупця мотоциклів та авторизація;
- перегляд даних профілю користувача як власника вживаних мотоциклів та покупця мотоциклів;
- оновлення даних профілю користувача як власника вживаних мотоциклів та покупця мотоциклів;
- формування замовлення для перегляду або придбання вживаного мотоциклу;
- керування замовленням для перегляду або придбання вживаного мотоциклу;
- створення відгуку про вживаний мотоцикл;
- перегляд відгуків про вживаний мотоцикл;
- перегляд відгуків про вживані мотоцикли, створених користувачем;
- збереження оголошення про продаж вживаного мотоциклу;
- скасування збереження оголошення про продаж вживаного мотоциклу;
- перегляд збережених оголошень про продаж вживаного мотоциклу;

- відправлення повідомлення з запитанням за оголошенням про продаж вживаного мотоцикла;
- перегляд користувачів, які написали за оголошенням про продаж вживаного мотоцикла;
- перегляд повідомлень, які написані користувачем за оголошенням про продаж вживаного мотоцикла;
- перегляд оголошень про продаж вживаних мотоциклів, за якими написав повідомлення користувач;
- перегляд замовлень для перегляду або придбання вживаного мотоциклу користувача;
- перегляд замовлень за оголошенням про продаж вживаного мотоцикла.

А.3.2 Умови експлуатації

Забезпечення експлуатації програмного забезпечення підтримки продажу вживаних мотоциклів вимагає наявності адміністратора системи або делегування відповідних функцій іншому працівнику.

Адміністратор має налагодити роботу сервера та розгорнути на ньому вебзастосунок, підтримувати роботу з налаштуваннями у подальшому, а також вносити за необхідності базові дані каталогу в базу даних через адміністративну панель.

А.3.3 Вимоги до складу та параметрів технічних засобів

Враховуючи, що програмне забезпечення підтримки продажу вживаних мотоциклів має бути створено у вигляді вебзастосунку, то вимоги до сервера є основними вимогами до технічних засобів, а відповідно мінімально включають двоядерний процесор, 4 Гб оперативної пам'яті, 100 Гб вільного обсягу жорсткого диску, щоби зберігати дані, які всі окрім файлів

кодів вебзастосунку мають бути представлені базою даних, де велику частину обсягу потенційно мають займати фотографії.

A.4 Порядок контролю та приймання

Завдання на роботу визначає календарний план виконання роботи, включаючи періоди виконання, очікувані результати та консультантів, які забезпечують контроль виконання відповідних етапів.

У підсумку відбувається захист дипломної кваліфікаційної роботи у відведений для цього випусковою кафедрою програмних засобів період, де представляються остаточні результати з підтвердженням проходження всіх попередніх етапів.

ДОДАТОК Б
Текст програми

```

from .models import Motorcycletype
from .models import Motorcycleproducer
from .models import Motorcyclemodel
from .models import Motorcycleversion
from .models import Motorcyclelead
from .models import Motorcycleleadphoto
from .models import Motorcycleowner
from .models import Motorcyclesearcher
from .models import Motorcycleorder
from .models import Motorcycleview
from .models import Motorcyclesave
from .models import Locationcountry
from .models import Locationplace
from .models import Motorcycletexting
from django.shortcuts import render
from django.core.paginator import Paginator

#перегляд існуючих версій моделі мотоцикла
def motorcyclemodel_view(request, motorcycle_model_id):
    try:
        motorcycle_model =
Motorcyclemodel.objects.prefetch_related("motorcycleproducerpk",
"motorcycletypepk",
"motorcycleproducerpk__registrationcountrypk").get(id =
motorcycle_model_id)
    except Motorcyclemodel.DoesNotExist:
        motorcycle_model = None
        motorcycle_versions = []
        motorcycle_photo = None
        if motorcycle_model:
            motorcycle_versions =
Motorcycleversion.objects.filter(motorcyclemodelpk = motorcycle_model)
            motorcycle_versions =
motorcycle_versions.order_by("motoversionname")
            motorcycle_photo =
Motorcycleleadphoto.objects.filter(motorcycleleadpk__motorcycleversionpk__
motorcyclemodelpk = motorcycle_model)[0]
        return render(request,
"motorcyclesaleapp/motorcyclemodel.html",
{"model": motorcycle_model,
"versions": motorcycle_versions,
"photo": motorcycle_photo })

#перегляд даних версії мотоциклу
def motorcycleversion_view(request, motorcycle_version_id):

```

```

        try:
            motorcycle_version =
Motorcycleversion.objects.select_related("motorcyclemodelpk",
"producedcountrypk", "motorcyclemodelpk__motorcycleproducerpk",
"motorcyclemodelpk__motorcycleproducerpk__registrationcountrypk",
"motorcyclemodelpk__motorcycletypepk").get(id = motorcycle_version_id)
            motorcycle_photo =
Motorcycleleadphoto.objects.filter(motorcycleleadpk__motorcycleversionpk =
motorcycle_version)[0]
            motorcycle_ads =
Motorcyclelead.objects.filter(motorcycleversionpk = motorcycle_version,
adstate = 0)

            motorcycle_ads = motorcycle_ads.order_by("-viewrate")
            motorcycle_ads = motorcycle_ads[:5]
        except Motorcycleversion.DoesNotExist:
            motorcycle_version = None
            if not motorcycle_version:
                return render(request,
"motorcyclesaleapp/motorcycleversion.html",
                    {"version": None,
                     "ads": [],
                     "photo": None})
            else:
                if request.user.is_authenticated:
                    user_auth_data = request.user
                    motorcycle_searcher =
Motorcyclesearcher.objects.filter(user = user_auth_data)
                    if motorcycle_searcher:
                        motorcycle_searcher = motorcycle_searcher [0]
                        for motorcycle_ad in motorcycle_ads:
                            motorcycle_save =
Motorcyclesave.objects.filter(motorcycleleadpk = motorcycle_ad,
motorcyclesearcherpk = motorcycle_searcher)
                            return render(request,
"motorcyclesaleapp/motorcycleversion.html",
                                {"version": motorcycle_version,
                                 "ads": motorcycle_ads,
                                 "photo": motorcycle_photo})

#перегляд даних виробника мотоциклів
def motorcycleproducer_view(request, motorcycle_producer_id):
    try:
        motorcycle_producer =
Motorcycleproducer.objects.select_related("registrationcountrypk").get
(id = motorcycle_producer_id)
    except Motorcycleproducer.DoesNotExist:

```

```

        motorcycle_producer = None
    if not motorcycle_producer:
        return render(request,
            "motorcyclesaleapp/motorcycleproducer.html",
            {"producer": None,
             "models": [],
             "ads": []})
    else:
        motorcycle_models =
Motorcyclemodel.objects.select_related("motorcycletypepk").filter(moto
rcycleproducerpk = motorcycle_producer)
        motorcycle_models =
motorcycle_models.order_by("motormodelname")
        motorcycle_models_paginator =
Paginator(motorcycle_models, 10)
        motorcycle_models_page = request.GET.get('page')
        motorcycle_models =
motorcycle_models_paginator.get_page(motorcycle_models_page)
        motorcycle_ads =
Motorcyclead.objects.filter(motorcycleversionpk__motorcyclemodelpk__mo
torcycleproducerpk = motorcycle_producer, adstate = 0)
        motorcycle_ads = motorcycle_ads.order_by("-viewrate")
        motorcycle_ads = motorcycle_ads[:5]
        if request.user.is_authenticated:
            user_auth_data = request.user
            motorcycle_searcher =
Motorcyclesearcher.objects.filter(user = user_auth_data)
            if motorcycle_searcher:
                motorcycle_searcher = motorcycle_searcher [0]
                for motorcycle_ad in motorcycle_ads:
                    motorcycle_save =
Motorcyclesave.objects.filter(motorcycleadpk = motorcycle_ad,
motorcyclesearcherpk = motorcycle_searcher)
                    if motorcycle_save:
                        motorcycle_ad.save = True
                    else:
                        motorcycle_ad.save = False
        return render(request,
            "motorcyclesaleapp/motorcycleproducer.html",
            {"producer": motorcycle_producer,
             "models": motorcycle_models,
             "ads": motorcycle_ads })

#пошук вживаних мотоциклів
def motorcycles_search_view(request):

```

```

        motorcycles =
Motorcyclead.objects.prefetch_related("motorcycleversionpk",
        "motorcycleownerpk",
        "motorcycleversionpk__motorcyclemodelpk",

"motorcycleversionpk__motorcyclemodelpk__motorcycleproducerpk",

"motorcycleversionpk__motorcyclemodelpk__motorcycletypepk").all()
        motorcycles = motorcycles.filter(adstate = 0)
        motorcycle_pricemin = request.GET.get("pricemin")
        motorcycle_pricemax = request.GET.get("pricemax")
        if motorcycle_pricemin:
            motorcycle_pricemin = float(motorcycle_pricemin)
            motorcycles = motorcycles.filter(price__gte =
motorcycle_pricemin)
        if motorcycle_pricemax:
            motorcycle_pricemax = float(motorcycle_pricemax)
            motorcycles = motorcycles.filter(price__lte =
motorcycle_pricemax)
        motorcycle_colour = request.GET.get("colour")
        if motorcycle_colour:
            motorcycles = motorcycles.filter(colour =
motorcycle_colour)
        motorcycle_mileage_min = request.GET.get("mileagemin")
        motorcycle_mileage_max = request.GET.get("mileagemax")
        if motorcycle_mileage_min:
            motorcycle_mileage_min = float(motorcycle_mileage_min)
            motorcycles = motorcycles.filter(mileage__gte =
motorcycle_mileage_min)
        if motorcycle_mileage_max:
            motorcycle_mileage_max = float(motorcycle_mileage_max)
            motorcycles = motorcycles.filter(mileage__lte =
motorcycle_mileage_max)
        motorcycle_usestate = request.GET.get("mileagemin")
        if motorcycle_usestate:
            motorcycles = motorcycles.filter(usestate =
motorcycle_usestate)
        motorcycle_viewrate_min = request.GET.get("viewratemin")
        if motorcycle_viewrate_min:
            motorcycle_viewrate_min = float(motorcycle_viewrate_min)
            motorcycles = motorcycles.filter(viewrate__gte =
motorcycle_viewrate_min)
        motorcycle_viewrate_max = request.GET.get("viewratemax")
        if motorcycle_viewrate_max:
            motorcycle_viewrate_max = float(motorcycle_viewrate_max)
            motorcycles = motorcycles.filter(viewrate__lte =
motorcycle_viewrate_max)

```

```

        motorcycle_firstregistration_min = request.GET.get("regmin")
        motorcycle_firstregistration_max = request.GET.get("regmax")
        if motorcycle_firstregistration_min:
            motorcycle_firstregistration_min =
int(motorcycle_firstregistration_min)
            motorcycles = motorcycles.filter(firstregistration__gte
= motorcycle_firstregistration_min)
            if motorcycle_firstregistration_max:
                motorcycle_firstregistration_max =
int(motorcycle_firstregistration_max)
                motorcycles = motorcycles.filter(firstregistration__lte
= motorcycle_firstregistration_max)
            motorcycle_version = request.GET.get("version")
            if motorcycle_version:
                motorcycles =
motorcycles.filter(motorcycleversionpk__motoversionname =
motorcycle_version)
            motorcycle_cooling = request.GET.get("cooling")
            if motorcycle_cooling:
                motorcycles =
motorcycles.filter(motorcycleversionpk__cooling = motorcycle_cooling)
            motorcycle_enginetype = request.GET.get("enginetype")
            if motorcycle_enginetype:
                motorcycles =
motorcycles.filter(motorcycleversionpk__enginetype =
motorcycle_enginetype)
            motorcycle_enginepower_min =
request.GET.get("enginepowermin")
            if motorcycle_enginepower_min:
                motorcycle_enginepower_min =
float(motorcycle_enginepower_min)
                motorcycles =
motorcycles.filter(motorcycleversionpk__enginepower__gte =
motorcycle_enginepower_min)
            motorcycle_enginepower_max =
request.GET.get("enginepowermax")
            if motorcycle_enginepower_max:
                motorcycle_enginepower_max =
float(motorcycle_enginepower_max)
                motorcycles =
motorcycles.filter(motorcycleversionpk__enginepower__lte =
motorcycle_enginepower_max)
            motorcycle_enginevolume_min =
request.GET.get("enginevolumemin")
            if motorcycle_enginevolume_min:
                motorcycle_enginevolume_min =
float(motorcycle_enginevolume_min)

```

```

        motorcycles =
motorcycles.filter(motorcycleversionpk__enginevolume__gte =
motorcycle_enginevolume_min)
        motorcycle_enginevolume_max =
request.GET.get("enginevolumemax")
        if motorcycle_enginevolume_max:
            motorcycle_enginevolume_max =
float(motorcycle_enginevolume_max)
        motorcycles =
motorcycles.filter(motorcycleversionpk__enginevolume__lte =
motorcycle_enginevolume_max)
        motorcycle_enginebore_min = request.GET.get("engineboremin")
        if motorcycle_enginebore_min:
            motorcycle_enginebore_min =
float(motorcycle_enginebore_min)
        motorcycles =
motorcycles.filter(motorcycleversionpk__enginebore__gte =
motorcycle_enginebore_min)
        motorcycle_enginebore_max = request.GET.get("engineboremax")
        if motorcycle_enginebore_max:
            motorcycle_enginebore_max =
float(motorcycle_enginebore_max)
        motorcycles =
motorcycles.filter(motorcycleversionpk__enginebore__lte =
motorcycle_enginebore_max)
        motorcycle_enginestroke_min =
request.GET.get("enginestrokemin")
        if motorcycle_enginestroke_min:
            motorcycle_enginestroke_min =
float(motorcycle_enginestroke_min)
        motorcycles =
motorcycles.filter(motorcycleversionpk__enginestroke__gte =
motorcycle_enginestroke_min)
        motorcycle_enginestroke_max =
request.GET.get("enginestrokemax")
        if motorcycle_enginestroke_max:
            motorcycle_enginestroke_max =
float(motorcycle_enginestroke_max)
        motorcycles =
motorcycles.filter(motorcycleversionpk__enginestroke__lte =
motorcycle_enginestroke_max)
        motorcycle_enginecompressionration_min =
request.GET.get("compressmin")
        if motorcycle_enginecompressionration_min:
            motorcycle_enginecompressionration_min =
float(motorcycle_enginecompressionration_min)

```



```

        motorcycles =
motorcycles.filter(motorcycleversionpk__reartire =
motorcycle_reartire)
        motorcycle_frontwheeltravel_min =
request.GET.get("frontwheelmin")
        if motorcycle_frontwheeltravel_min:
            motorcycle_frontwheeltravel_min =
float(motorcycle_frontwheeltravel_min)
            motorcycles =
motorcycles.filter(motorcycleversionpk__frontwheeltravel__gte =
motorcycle_frontwheeltravel_min)
            motorcycle_frontwheeltravel_max =
request.GET.get("frontwheelmax")
            if motorcycle_frontwheeltravel_max:
                motorcycle_frontwheeltravel_max =
float(motorcycle_frontwheeltravel_max)
                motorcycles =
motorcycles.filter(motorcycleversionpk__frontwheeltravel__lte =
motorcycle_frontwheeltravel_max)
                motorcycle_rearwheeltravel_min =
request.GET.get("rearwheelmin")
                if motorcycle_rearwheeltravel_min:
                    motorcycle_rearwheeltravel_min =
float(motorcycle_rearwheeltravel_min)
                    motorcycles =
motorcycles.filter(motorcycleversionpk__rearwheeltravel__gte =
motorcycle_rearwheeltravel_min)
                    motorcycle_rearwheeltravel_max =
request.GET.get("rearwheelmax")
                    if motorcycle_rearwheeltravel_max:
                        motorcycle_rearwheeltravel_max =
float(motorcycle_rearwheeltravel_max)
                        motorcycles =
motorcycles.filter(motorcycleversionpk__rearwheeltravel__lte =
motorcycle_rearwheeltravel_max)
                        motorcycle_heigh_min = request.GET.get("heightmin")
                        if motorcycle_heigh_min:
                            motorcycle_heigh_min = float(motorcycle_heigh_min)
                            motorcycles =
motorcycles.filter(motorcycleversionpk__minheigh__gte =
motorcycle_heigh_min)
                            motorcycle_heigh_max = request.GET.get("heightmax")
                            if motorcycle_heigh_max:
                                motorcycle_heigh_max = float(motorcycle_heigh_max)
                                motorcycles =
motorcycles.filter(motorcycleversionpk__maxheight__lte =
motorcycle_heigh_max)

```

```

        motorcycle_produced_country =
request.GET.get("producedcountry")
        if motorcycle_produced_country:
            location_country =
Locationcountry.objects.get(countryname = motorcycle_produced_country)
            motorcycles =
motorcycles.filter(motorcycleversionpk__producedcountrypk =
location_country)
            motorcycle_gearbox = request.GET.get("gearbox")
            if motorcycle_gearbox:
                motorcycles =
motorcycles.filter(motorcycleversionpk__gearbox = motorcycle_gearbox)
            motorcycle_seats_min = request.GET.get("seatsmin")
            if motorcycle_seats_min:
                motorcycle_seats_min = int(motorcycle_seats_min)
                motorcycles =
motorcycles.filter(motorcycleversionpk__seats__gte =
motorcycle_seats_min)
            motorcycle_seats_max = request.GET.get("seatsmax")
            if motorcycle_seats_max:
                motorcycle_seats_max = int(motorcycle_seats_max)
                motorcycles =
motorcycles.filter(motorcycleversionpk__seats__lte =
motorcycle_seats_max)
            motorcycle_cylindernumber_min =
request.GET.get("cylindermin")
            if motorcycle_cylindernumber_min:
                motorcycle_cylindernumber_min =
int(motorcycle_cylindernumber_min)
                motorcycles =
motorcycles.filter(motorcycleversionpk__cylindernumber__gte =
motorcycle_cylindernumber_min)
            motorcycle_cylindernumber_max =
request.GET.get("cylindermax")
            if motorcycle_cylindernumber_max:
                motorcycle_cylindernumber_max =
int(motorcycle_cylindernumber_max)
                motorcycles =
motorcycles.filter(motorcycleversionpk__cylindernumber__lte =
motorcycle_cylindernumber_max)
            motorcycle_emptyweight_min =
request.GET.get("emptyweightmin")
            if motorcycle_emptyweight_min:
                motorcycle_emptyweight_min =
float(motorcycle_emptyweight_min)

```

```

        motorcycles =
motorcycles.filter(motorcycleversionpk__emptyweight__gte =
motorcycle_emptyweight_min)
        motorcycle_emptyweight_max =
request.GET.get("emptyweightmax")
        if motorcycle_emptyweight_max:
            motorcycle_emptyweight_max =
float(motorcycle_emptyweight_max)
            motorcycles =
motorcycles.filter(motorcycleversionpk__emptyweight__lte =
motorcycle_emptyweight_max)
            motorcycle_fullweight_min = request.GET.get("fullweightmin")
            if motorcycle_fullweight_min:
                motorcycle_fullweight_min =
float(motorcycle_fullweight_min)
                motorcycles =
motorcycles.filter(motorcycleversionpk__fullweight__gte =
motorcycle_fullweight_min)
                motorcycle_fullweight_max = request.GET.get("fullweightmax")
                if motorcycle_fullweight_max:
                    motorcycle_fullweight_max =
float(motorcycle_fullweight_max)
                    motorcycles =
motorcycles.filter(motorcycleversionpk__fullweight__lte =
motorcycle_fullweight_max)
                    motorcycle_lengthsize_min = request.GET.get("lengthsizemin")
                    if motorcycle_lengthsize_min:
                        motorcycle_lengthsize_min =
float(motorcycle_lengthsize_min)
                        motorcycles =
motorcycles.filter(motorcycleversionpk__lengthsize__gte =
motorcycle_lengthsize_min)
                        motorcycle_lengthsize_max = request.GET.get("lengthsizemax")
                        if motorcycle_lengthsize_max:
                            motorcycle_lengthsize_max =
float(motorcycle_lengthsize_max)
                            motorcycles =
motorcycles.filter(motorcycleversionpk__lengthsize__lte =
motorcycle_lengthsize_max)
                            motorcycle_widthsize_min = request.GET.get("widthsizemin")
                            if motorcycle_widthsize_min:
                                motorcycle_widthsize_min =
float(motorcycle_widthsize_min)
                                motorcycles =
motorcycles.filter(motorcycleversionpk__widthsize__gte =
motorcycle_widthsize_min)
                                motorcycle_widthsize_max = request.GET.get("widthsizemax")

```

```

        if motorcycle_widthsize_max:
            motorcycle_widthsize_max =
float(motorcycle_widthsize_max)
            motorcycles =
motorcycles.filter(motorcycleversionpk__widthsize__lte =
motorcycle_widthsize_max)
            motorcycle_heightsize_min = request.GET.get("heightsizemin")
            if motorcycle_heightsize_min:
                motorcycle_heightsize_min =
float(motorcycle_heightsize_min)
                motorcycles =
motorcycles.filter(motorcycleversionpk__heightsize__gte =
motorcycle_heightsize_min)
                motorcycle_heightsize_max = request.GET.get("heightsizemax")
                if motorcycle_heightsize_max:
                    motorcycle_heightsize_max =
float(motorcycle_heightsize_max)
                    motorcycles =
motorcycles.filter(motorcycleversionpk__heightsize__lte =
motorcycle_heightsize_max)
                    motorcycle_wheelbase_min = request.GET.get("wheelbasemin")
                    if motorcycle_wheelbase_min:
                        motorcycle_wheelbase_min =
float(motorcycle_wheelbase_min)
                        motorcycles =
motorcycles.filter(motorcycleversionpk__wheelbase__gte =
motorcycle_wheelbase_min)
                        motorcycle_wheelbase_max = request.GET.get("wheelbasemax")
                        if motorcycle_wheelbase_max:
                            motorcycle_wheelbase_max =
float(motorcycle_wheelbase_max)
                            motorcycles =
motorcycles.filter(motorcycleversionpk__wheelbase__lte =
motorcycle_wheelbase_max)
                            motorcycle_maxspeed_min = request.GET.get("maxspeedmin")
                            if motorcycle_maxspeed_min:
                                motorcycle_maxspeed_min = float(motorcycle_maxspeed_min)
                                motorcycles =
motorcycles.filter(motorcycleversionpk__maxspeed__gte =
motorcycle_maxspeed_min)
                                motorcycle_maxspeed_max = request.GET.get("maxspeedmax")
                                if motorcycle_maxspeed_max:
                                    motorcycle_maxspeed_max = float(motorcycle_maxspeed_max)
                                    motorcycles =
motorcycles.filter(motorcycleversionpk__maxspeed__lte =
motorcycle_maxspeed_max)
                                    motorcycle_type = request.GET.get("type")

```

```

        if motorcycle_type:
            motorcycle_type = int(motorcycle_type)
            motorcycle_type = Motorcycletype.objects.get(id =
motorcycle_type)
            motorcycles =
motorcycles.filter(motorcycleversionpk__motorcyclemodelpk__motorcyclet
ypepk = motorcycle_type)
            motorcycle_producer = request.GET.get("producer")
            if motorcycle_producer:
                motorcycle_producer = int(motorcycle_producer)
                motorcycle_producer = Motorcycloproducer.objects.get(id
= motorcycle_producer)
                motorcycles =
motorcycles.filter(motorcycleversionpk__motorcyclemodelpk__motorcyclep
roducerpk = motorcycle_producer)
                for motorcycle in motorcycles:
                    motorcycle.photo =
Motorcycleadphoto.objects.filter(motorcycleadpk = motorcycle)[0]
                    if request.user.is_authenticated:
                        user_auth_data = request.user
                        motorcycle_searcher =
Motorcyclesearcher.objects.filter(user = user_auth_data)
                        if motorcycle_searcher:
                            motorcycle_searcher = motorcycle_searcher[0]
                            for motorcycle_ad in motorcycles:
                                motorcycle_save =
Motorcyclesave.objects.filter(motorcycleadpk = motorcycle_ad,
motorcyclesearcherpk = motorcycle_searcher)
                                if motorcycle_save:
                                    motorcycle_ad.save = True
                                else:
                                    motorcycle_ad.save = False
            motorcycle_orderway = request.GET.get("orderway")
            if not motorcycle_orderway:
                motorcycle_orderway = int(motorcycle_orderway)
            else:
                motorcycle_orderway = 0
            if motorcycle_orderway == 0:
                motorcycles = motorcycles.order_by("-price")
            elif motorcycle_orderway == 1:
                motorcycles = motorcycles.order_by("price")
            elif motorcycle_orderway == 2:
                motorcycles = motorcycles.order_by("-mileage")
            elif motorcycle_orderway == 3:
                motorcycles = motorcycles.order_by("mileage")
            elif motorcycle_orderway == 4:
                motorcycles = motorcycles.order_by("-viewrate")

```

```

elif motorcycle_orderway == 5:
    motorcycles = motorcycles.order_by("viewrate")
elif motorcycle_orderway == 6:
    motorcycles = motorcycles.order_by("-firstregistration")
elif motorcycle_orderway == 7:
    motorcycles = motorcycles.order_by("firstregistration")
elif motorcycle_orderway == 8:
    motorcycles = motorcycles.order_by("-
motorcycleversionpk__motoversionname")
elif motorcycle_orderway == 9:
    motorcycles =
motorcycles.order_by("motorcycleversionpk__motoversionname")
    motorcycles_paginator = Paginator(motorcycles, 10)
    motorcycles_page = request.GET.get("page")
    motorcycles =
motorcycles_paginator.get_page(motorcycles_page)
    motorcycle_types = Motorcycletype.objects.all()
    motorcycle_producers = Motorcycleproducer.objects.all()
    motorcycle_models =
Motorcyclemodel.objects.select_related("motorcycleproducerpk",
"motorcycletypepk").all()
    return render(request,
"motorcyclesaleapp/motorcyclesearch.html",
        {"types": motorcycle_types,
         "producers": motorcycle_producers,
         "ads": motorcycles,
         "models": motorcycle_models })

#перегляд оголошення про продаж вживаного мотоциклу
def motorcycle_ad_view(request, motorcycle_ad_id):
    try:
        motorcycle_ad =
Motorcyclead.objects.select_related("motorcycleversionpk",
"motorcycleownerpk", "motorcycleownerpk__user",
"motorcycleversionpk__motorcyclemodelpk",
"motorcycleversionpk__motorcyclemodelpk__motorcycletypepk",
"motorcycleversionpk__motorcyclemodelpk__motorcycleproducerpk",
"motorcycleversionpk__motorcyclemodelpk__motorcycleproducerpk__registr
ationcountrypk", "motorcycleversionpk__producedcountrypk").get(id =
motorcycle_producer_id)
    except Motorcyclead.DoesNotExist:
        motorcycle_ad = None
    if not motorcycle_ad:
        return render(request,
"motorcyclesaleapp/motorcyclelead.html",
            {"ad": None,
             "views": []},

```

```

        "photos": []})
    else:
        motorcycle_ad_views =
Motorcycleview.objects.select_related("motorcyclesearcherpk",
"motorcyclesearcherpk__user").filter(motorcycleadpk = motorcycle_ad)
        motorcycle_ad_views = motorcycle_ad_views.order_by("-
viewdate")[:10]
        photos = Motorcycleadphoto.objects.filter(motorcycleadpk
= motorcycle_ad)
        if request.user.is_authenticated:
            user_auth_data = request.user
            motorcycle_searcher =
Motorcyclesearcher.objects.filter(user = user_auth_data)
            if motorcycle_searcher:
                motorcycle_searcher = motorcycle_searcher[0]
                motorcycle_save =
Motorcyclesave.objects.filter(motorcycleadpk = motorcycle_ad,
motorcyclesearcherpk = motorcycle_searcher)
                if motorcycle_save:
                    motorcycle_ad.save = True
                else:
                    motorcycle_ad.save = False
                motorcycle_view =
Motorcycleview.objects.filter(motorcycleadpk = motorcycle_ad,
motorcyclesearcherpk = motorcycle_searcher)
                motorcycle_view = motorcycle_view.order_by("-
viewdate")

                motorcycle_view = motorcycle_view[0]
                motorcycle_order =
Motorcycleorder.objects.filter(motorcycleadpk = motorcycle_ad,
motorcyclesearcherpk = motorcycle_searcher, ordereddate__gte =
motorcycle_view.viewdate)
                if motorcycle_order:
                    motorcycle_ad.toview = True
                else:
                    motorcycle_ad.toview = False
        return render(request,
"motorcyclesaleapp/motorcycleproducer.html",
        {"ad": motorcycle_ad,
"views": motorcycle_ad_views,
"photos": photos })

```

```

#перегляд оголошень про продаж вживаних мотоциклів у користувача
def motorcycle_owner_ads_view(request, owner_username):
    try:

```

```

        motorcycle_owner =
Motorcycleowner.objects.select_related("user").get(user__username =
owner_username)
        except Motorcycleowner.DoesNotExist:
            motorcycle_owner = None
        if not motorcycle_owner:
            return render(request,
"motorcyclesaleapp/ownerads.html",
{"ads": []})
        else:
            motorcycle_ads =
Motorcyclelead.objects.filter(motorcycleownerpk = motorcycle_owner,
adstate = 0)
            for motorcycle in motorcycle_ads:
                motorcycle.photo =
Motorcycleleadphoto.objects.filter(motorcycleleadpk = motorcycle)[0]
                if request.user.is_authenticated:
                    user_auth_data = request.user
                    motorcycle_searcher =
Motorcyclesearcher.objects.filter(user = user_auth_data)
                    if motorcycle_searcher:
                        motorcycle_searcher = motorcycle_searcher[0]
                        for motorcycle_ad in motorcycle_ads:
                            motorcycle_save =
Motorcyclesave.objects.filter(motorcycleleadpk = motorcycle_ad,
motorcyclesearcherpk = motorcycle_searcher)
                            if motorcycle_save:
                                motorcycle_ad.save = True
                            else:
                                motorcycle_ad.save = False
                        motorcycle_orderway = request.GET.get("orderway")
                        if not motorcycle_orderway:
                            motorcycle_orderway = int(motorcycle_orderway)
                        else:
                            motorcycle_orderway = 0
                        if motorcycle_orderway == 0:
                            motorcycle_ads = motorcycle_ads.order_by("-
price")
                        elif motorcycle_orderway == 1:
                            motorcycle_ads =
motorcycle_ads.order_by("price")
                        elif motorcycle_orderway == 2:
                            motorcycle_ads = motorcycle_ads.order_by("-
mileage")
                        elif motorcycle_orderway == 3:
                            motorcycle_ads =
motorcycle_ads.order_by("mileage")

```

```

        elif motorcycle_orderway == 4:
            motorcycle_ads = motorcycle_ads.order_by("-
viewrate")
        elif motorcycle_orderway == 5:
            motorcycle_ads =
motorcycle_ads.order_by("viewrate")
        elif motorcycle_orderway == 6:
            motorcycle_ads = motorcycle_ads.order_by("-
firstregistration")
        elif motorcycle_orderway == 7:
            motorcycle_ads =
motorcycle_ads.order_by("firstregistration")
        elif motorcycle_orderway == 8:
            motorcycle_ads = motorcycle_ads.order_by("-
motorcycleversionpk__motoversionname")
        elif motorcycle_orderway == 9:
            motorcycle_ads =
motorcycle_ads.order_by("motorcycleversionpk__motoversionname")
        motorcycles_paginator = Paginator(motorcycle_ads,
10)

        motorcycles_page = request.GET.get("page")
        motorcycle_ads =
motorcycles_paginator.get_page(motorcycles_page)
        return render(request,
"motorcyclesaleapp/ownerads.html",
                        {"ads": motorcycle_ads})

#перегляд власних оголошень про продаж вживаних мотоциклів у
користувача
def own_motorcycle_owner_ads_view(request):
    if request.user.is_authenticated:
        try:
            motorcycle_owner =
Motorcycleowner.objects.select_related("user").get(user =
request.user)
        except Motorcycleowner.DoesNotExist:
            motorcycle_owner = None
    else:
        return render(request, "motorcyclesaleapp/ownads.html",
                        {"ads": []})
    if not motorcycle_owner:
        return render(request, "motorcyclesaleapp/ownads.html",
                        {"ads": []})
    else:
        motorcycle_ads =
Motorcyclead.objects.filter(motorcycleownerpk = motorcycle_owner)

```

```

        for motorcycle in motorcycle_ads:
            motorcycle.photo =
Motorcycleadphoto.objects.filter(motorcycleadpk = motorcycle)[0]
            motorcycle_orderway = request.GET.get("orderway")
            if not motorcycle_orderway:
                motorcycle_orderway = int(motorcycle_orderway)
            else:
                motorcycle_orderway = 0
            if motorcycle_orderway == 0:
                motorcycle_ads = motorcycle_ads.order_by("-price")
            elif motorcycle_orderway == 1:
                motorcycle_ads = motorcycle_ads.order_by("price")
            elif motorcycle_orderway == 2:
                motorcycle_ads = motorcycle_ads.order_by("-mileage")
            elif motorcycle_orderway == 3:
                motorcycle_ads = motorcycle_ads.order_by("mileage")
            elif motorcycle_orderway == 4:
                motorcycle_ads = motorcycle_ads.order_by("-
viewrate")
            elif motorcycle_orderway == 5:
                motorcycle_ads = motorcycle_ads.order_by("viewrate")
            elif motorcycle_orderway == 6:
                motorcycle_ads = motorcycle_ads.order_by("-
firstregistration")
            elif motorcycle_orderway == 7:
                motorcycle_ads =
motorcycle_ads.order_by("firstregistration")
            elif motorcycle_orderway == 8:
                motorcycle_ads = motorcycle_ads.order_by("-
motorcycleversionpk__motoversionname")
            elif motorcycle_orderway == 9:
                motorcycle_ads =
motorcycle_ads.order_by("motorcycleversionpk__motoversionname")
            motorcycles_paginator = Paginator(motorcycle_ads, 10)
            motorcycles_page = request.GET.get("page")
            motorcycle_ads =
motorcycles_paginator.get_page(motorcycles_page)
            return render(request, "motorcyclesaleapp/ownads.html",
                           {"ads": motorcycle_ads})

```

```

#перегляд відгуків про вживаний мотоцикл
def motorcycle_views_view(request, motorcycle_ad_id):
    try:
        motorcycle_ad =
Motorcyclead.objects.select_related("motorcycleversionpk",
"motorcycleownerpk", "motorcycleownerpk__user",

```

```

"motorcycleversionpk__motorcyclemodelpk",
"motorcycleversionpk__motorcyclemodelpk__motorcycletypepk",
"motorcycleversionpk__motorcyclemodelpk__motorcycleproducerpk",
"motorcycleversionpk__motorcyclemodelpk__motorcycleproducerpk__registr
ationcountrypk", "motorcycleversionpk__producedcountrypk").get(id =
motorcycle_producer_id)
    except Motorcyclead.DoesNotExist:
        motorcycle_ad = None
    if not motorcycle_ad:
        return render(request,
"motorcyclesaleapp/motorcycleadvviews.html",
                        {"ad": None,
                         "views": [],
                         "photos": []})
    else:
        motorcycle_ad_views =
Motorcycleview.objects.select_related("motorcyclesearcherpk",
"motorcyclesearcherpk__user").filter(motorcycleadpk = motorcycle_ad)
        motorcycle_ad_views = motorcycle_ad_views.order_by("-
viewdate")
        motorcycle_ad_views_paginator =
Paginator(motorcycle_ad_views, 10)
        motorcycle_ad_views_page = request.GET.get('page')
        motorcycle_ad_views =
motorcycle_ad_views_paginator.get_page(motorcycle_ad_views_page)
        photos = Motorcycleadphoto.objects.filter(motorcycleadpk
= motorcycle_ad)
        if request.user.is_authenticated:
            user_auth_data = request.user
            motorcycle_searcher =
Motorcyclesearcher.objects.filter(user = user_auth_data)
            if motorcycle_searcher:
                motorcycle_searcher = motorcycle_searcher[0]
                motorcycle_save =
Motorcyclesave.objects.filter(motorcycleadpk = motorcycle_ad,
motorcyclesearcherpk = motorcycle_searcher)
                if motorcycle_save:
                    motorcycle_ad.save = True
                else:
                    motorcycle_ad.save = False
        return render(request,
"motorcyclesaleapp/motorcycleadvviews.html",
                        {"ad": motorcycle_ad,
                         "views": motorcycle_ad_views,
                         "photos": photos })

```

```

#перегляд відгуків про вживані мотоцикли, створених користувачем
def motor_searcher_views_view(request, searcher_username):
    try:
        motorcycle_searcher =
Motorcyclesearcher.objects.select_related("user").get(user__username =
searcher_username)
    except Motorcyclesearcher.DoesNotExist:
        motorcycle_searcher = None
    if not motorcycle_searcher:
        return render(request,
"motorcyclesaleapp/seracherviews.html",
{"views": []})
    else:
        motorcycle_ad_views =
Motorcycleview.objects.select_related("motorcyclesearcherpk",
"motorcyclesearcherpk__user",
"motorcycleadpk").filter(motorcyclesearcherpk = motorcycle_searcher)
        motorcycle_ad_views = motorcycle_ad_views.order_by("-
viewdate")
        motorcycle_ad_views_paginator =
Paginator(motorcycle_ad_views, 10)
        motorcycle_ad_views_page = request.GET.get('page')
        motorcycle_ad_views =
motorcycle_ad_views_paginator.get_page(motorcycle_ad_views_page)
        for motorcycle_ad_view in motorcycle_ad_views:
            motorcycle_ad_view.photo =
Motorcycleadphoto.objects.filter(motorcycleadpk =
motorcycle_ad_view.motorcycleadpk)[0]
            if request.user.is_authenticated:
                user_auth_data = request.user
                motorcycle_searcher =
Motorcyclesearcher.objects.filter(user = user_auth_data)
                if motorcycle_searcher:
                    motorcycle_searcher = motorcycle_searcher[0]
                    for motorcycle_ad_view in motorcycle_ad_views:
                        motorcycle_save =
Motorcyclesave.objects.filter(motorcycleadpk =
motorcycle_ad_view.motorcycleadpk, motorcyclesearcherpk =
motorcycle_searcher)

                        if motorcycle_save:
                            motorcycle_ad_view.save = True
                        else:
                            motorcycle_ad_view.save = False
        return render(request,
"motorcyclesaleapp/seracherviews.html",
{"views": motorcycle_ad_views })

```

```

#перегляд власних відгуків про вживані мотоцикли, створених
користувачем
def own_motor_searcher_views_view(request):
    if request.user.is_authenticated:
        try:
            motorcycle_searcher =
Motorcyclesearcher.objects.select_related("user").get(user =
request.user)
        except Motorcyclesearcher.DoesNotExist:
            motorcycle_searcher = None
    else:
        return render(request,
"motorcyclesaleapp/ownseracherviews.html",
{"views": []})
    if not motorcycle_searcher:
        return render(request,
"motorcyclesaleapp/ownseracherviews.html",
{"views": []})
    else:
        motorcycle_ad_views =
Motorcycleview.objects.select_related("motorcyclesearcherpk",
"motorcyclesearcherpk__user",
"motorcycleadpk").filter(motorcyclesearcherpk = motorcycle_searcher)
        motorcycle_ad_views = motorcycle_ad_views.order_by("-
viewdate")
        motorcycle_ad_views_paginator =
Paginator(motorcycle_ad_views, 10)
        motorcycle_ad_views_page = request.GET.get('page')
        motorcycle_ad_views =
motorcycle_ad_views_paginator.get_page(motorcycle_ad_views_page)
        for motorcycle_ad_view in motorcycle_ad_views:
            motorcycle_ad_view.photo =
Motorcycleadphoto.objects.filter(motorcycleadpk =
motorcycle_ad_view.motorcycleadpk)[0]
        return render(request,
"motorcyclesaleapp/ownseracherviews.html",
{"views": motorcycle_ad_views })

```

ДОДАТОК В
Слайди презентації

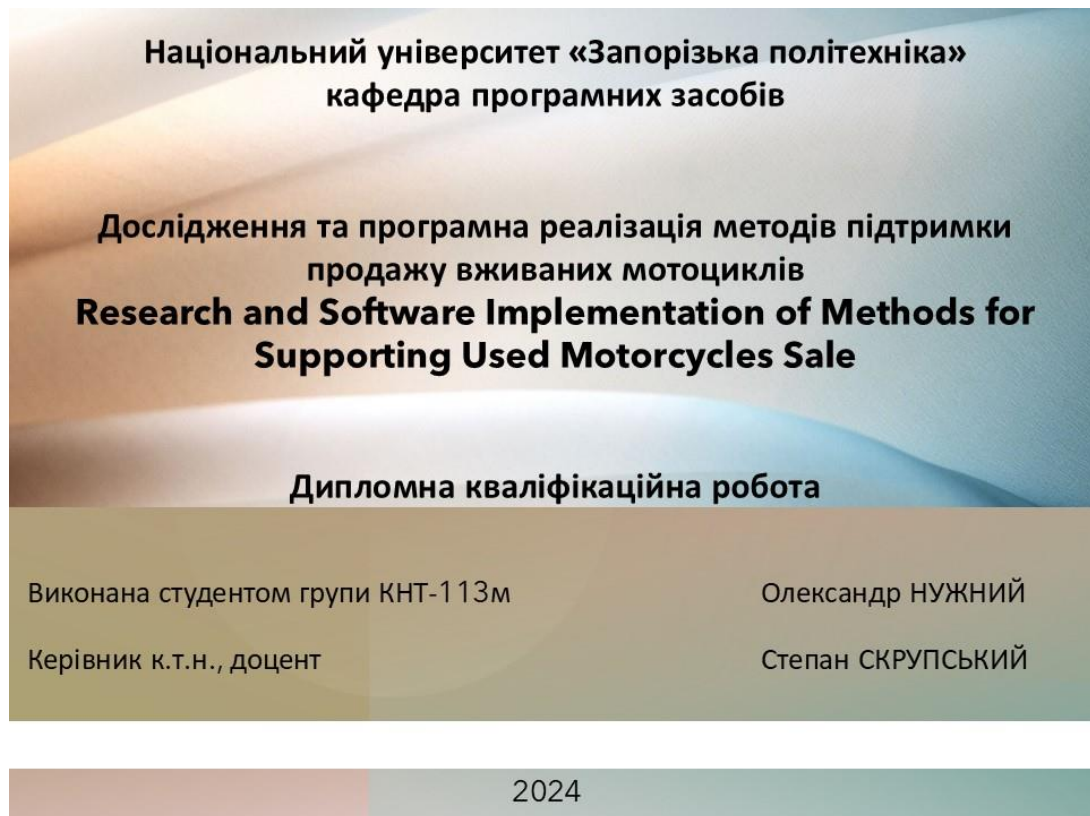


Рисунок В.1 – Початковий слайд

ОБ'ЄКТ, ПРЕДМЕТ ТА МЕТА РОБОТИ

- Об'єкт роботи – процес продажу вживаних мотоциклів.
- Предмет роботи – методи підтримки продажу вживаних мотоциклів.
- Метою роботи є розробка програмного забезпечення підтримки продажу вживаних мотоциклів для забезпечення зручності потенційних продавців під час публікації оголошень про продаж вживаних мотоциклів на основі засобів прогнозування вартості.



Рисунок В.2 – Об'єкт, предмет, мета роботи

ЗАВДАННЯ РОБОТИ

- виконати аналіз програмних застосунків підтримки продажу вживаних мотоциклів;
- розробити метод прогнозування вартості продажу вживаного мотоциклу;
- виконати проєктування вебзастосунку підтримки продажу вживаних мотоциклів;
- реалізувати вебзастосунок підтримки продажу вживаних мотоциклів;
- виконати експериментальне дослідження прогнозування вартості продажу вживаного мотоциклу.

3

Рисунок В.3 – Завдання роботи

ПОРІВНЯННЯ МОЖЛИВОСТЕЙ АНАЛОГІВ РОЗРОБКИ ВЕБЗАСТОСУНКУ ПІДТРИМКИ ПРОДАЖУ ВЖИВАНИХ МОТОЦИКЛІВ

Показник	Motolux	Bikesales	UsedMotorcycle Store.com	Розробка
Продаж вживаних мотоциклів	Так	Так	Так	Так
Пошук мотоциклів	Розширений	Розширений	Основні параметри	Розширений
Написання повідомлення власнику	Так	Так	Ні	Так
Формування заявок на перегляд мотоцикла	Ні	Ні	Ні	Так
Оцінювання переглянутого мотоцикла	Так, але фактично використовується для запитань	Ні	Ні	Так
Визначення вартості продажу вживаного мотоцикла	Ні	Ні	Ні	Так

4

Рисунок В.4 – Порівняння можливостей аналогів розробки вебзастосунку підтримки продажу вживаних мотоциклів

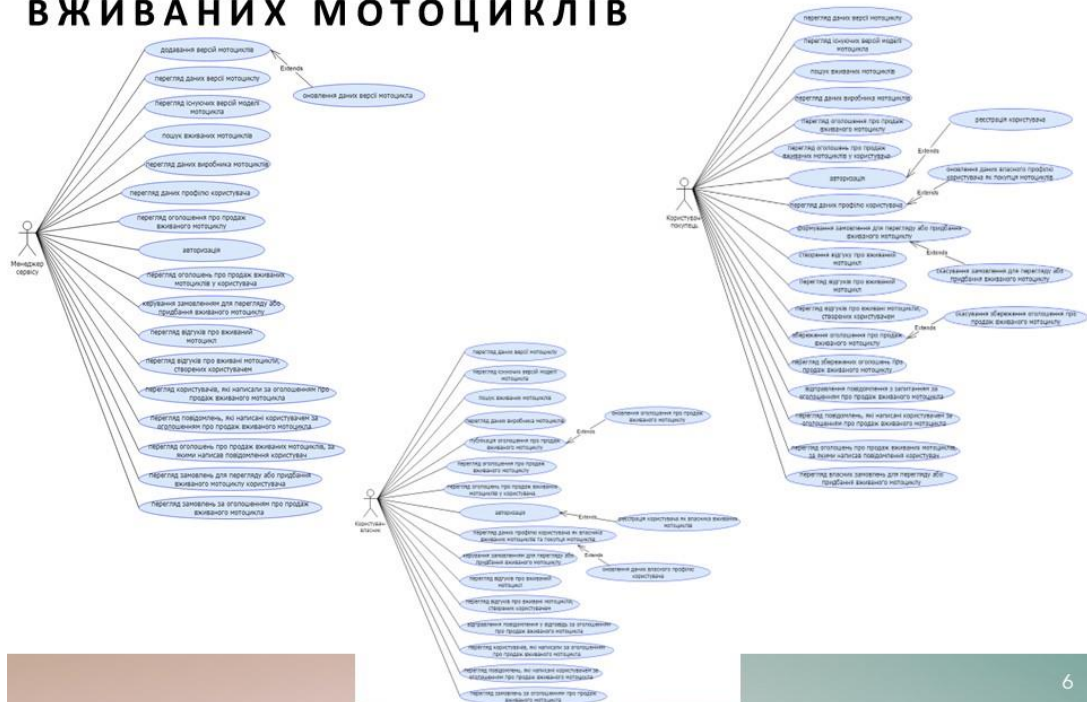
МЕТОД ПРОГНОЗУВАННЯ ВАРТОСТІ ВЖИВАНИХ МОТОЦИКЛІВ

- збирання даних з оголошень про продаж вживаних мотоциклів;
- формування навчальної вибірки даних про вживані мотоцикли через вилучення всіх наявних дублікатів з даних, вилучення екземплярів даних, які мають пропущені значення ознак, перетворення характеристик вживаних мотоциклів на вхідні ознаки;
- створення моделі на основі дерева рішень та навчання цієї моделі за допомогою сформованої навчальної вибірки даних з відбором вхідних ознак моделі;
- формування значень вхідних ознак вживаного мотоцикла, подання на вхід навченої моделі та прогнозування вартості продажу вживаного мотоцикла.

5

Рисунок В.5 – Метод прогнозування вартості вживаних мотоциклів

ДІАГРАМИ ПРЕЦЕДЕНТІВ ДЛЯ ВЕБЗАСТОСУНКУ ПІДТРИМКИ ПРОДАЖУ ВЖИВАНИХ МОТОЦИКЛІВ



6

Рисунок В.6 – Діаграми прецедентів для вебзастосунку підтримки продажу вживаних мотоциклів

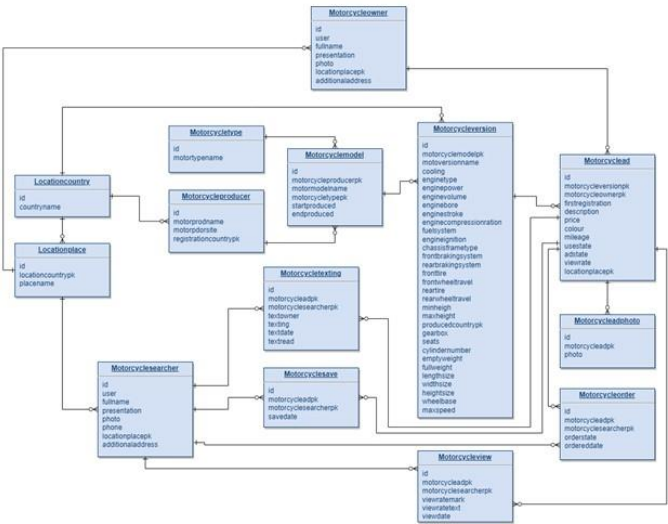
**ПОРІВНЯННЯ МОЖЛИВИХ РІШЕНЬ ДЛЯ
РЕАЛІЗАЦІЇ ВЕБЗАСТОСУНКУ ПІДТРИМКИ
ПРОДАЖУ ВЖИВАНИХ МОТОЦИКЛІВ**

	Django	Pyramid
Робота з базою даних	Засоби взаємодії з базою даних та вбудоване об'єктно-реляційне відображення	Необхідно використовувати засоби зовнішніх розробників для реалізації об'єктно-реляційного відображення, інсталиувати
Наповнення даними вебзастосунку (зокрема стандартними)	Вбудована адміністративна панель повністю вирішує цю функціональність	Необхідно використовувати засоби зовнішніх розробників, інсталиувати
Структуризація коду	Окремі складові на основі шаблону Model-View-Template з можливостями розширення	Весь вебзастосунок представлений одним файлом
Робота з засобами прогнозування і аналітики	Доступні всі актуальні Python-засоби	

7

Рисунок В.7 – Порівняння можливих рішень для реалізації вебзастосунку підтримки продажу вживаних мотоциклів

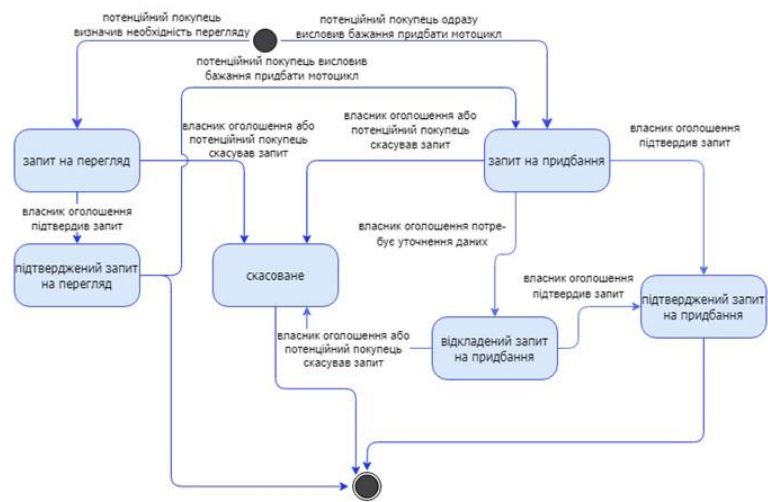
**СХЕМА БАЗИ ДАНИХ ПРОГРАМНОГО
ЗАБЕЗПЕЧЕННЯ ПІДТРИМКИ ПРОДАЖУ
ВЖИВАНИХ МОТОЦИКЛІВ**



8

Рисунок В.8 – Схема бази даних програмного забезпечення підтримки продажу вживаних мотоциклів

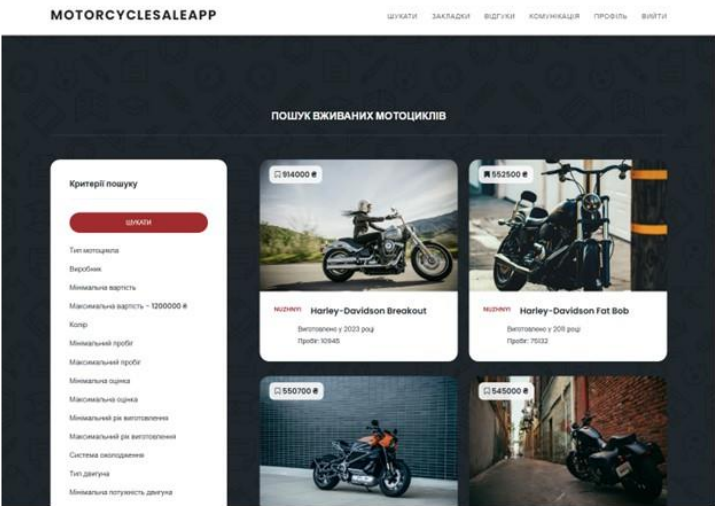
ДІАГРАМА СТАНІВ ЗАМОВЛЕННЯ ПЕРЕГЛЯДУ АБО ПРИДБАННЯ ВЖИВАНИХ МОТОЦИКЛІВ



9

Рисунок В.9 – Діаграма станів замовлення перегляду або придбання вживаних мотоциклів

ПОШУК ВЖИВАНИХ МОТОЦИКЛІВ



10

Рисунок В.10 – Пошук вживаних мотоциклів

ПЕРЕГЛЯД ОГЛОШЕННЯ ПРО ПРОДАЖ ВЖИВАНОГО МОТОЦИКЛА

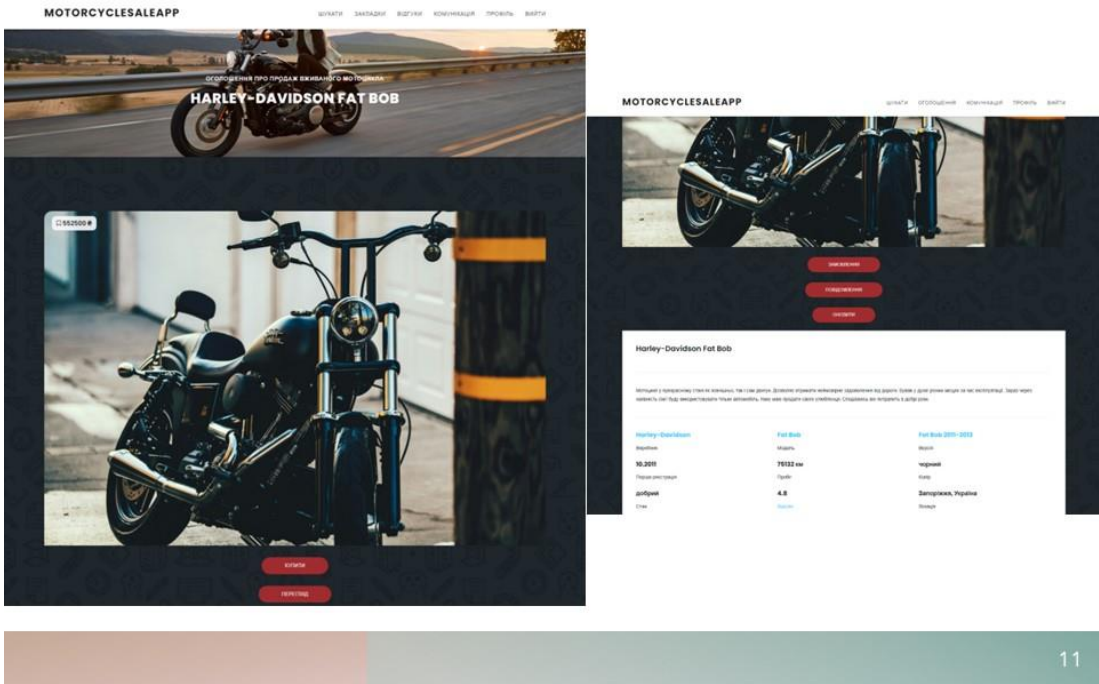


Рисунок В.11 – Перегляд оголошення про продаж вживаного мотоцикла

ПУБЛІКАЦІЯ ОГЛОШЕННЯ ПРО ПРОДАЖ ВЖИВАНОГО МОТОЦИКЛА

The screenshot shows the 'ВАШЕ НОВЕ ОГЛОШЕННЯ ПРО ПРОДАЖ ВЖИВАНОГО МОТОЦИКЛА' form. The form includes fields for 'Виробник' (Harley-Davidson), 'Модель' (Fat Bob), 'Версія' (NC 750X 2022), 'Колір' (чорний), 'Світла' (хромовані), 'Країна' (Україна), 'Місто' (Закарпаття), 'Початок реєстрації' (01.04.2022), 'Пробіг, км' (1000), 'Вартість, грн' (45000), and a text area for 'Опис'. There are red buttons for 'Додати в користування', 'Завершити вартість', and 'Опублікувати'.

Рисунок В.12 – Публікація оголошення про продаж вживаного мотоцикла

РЕЗУЛЬТАТИ ПРОГНОЗУВАННЯ ВАРТОСТІ ВЖИВАНИХ МОТОЦИКЛІВ

Модель прогнозування	Середня абсолютна похибка	Середня квадратична похибка
Дерево рішень зі всіма ознаками	0,0002566078692600107	0,00024789303370922357
Дерево рішень з 4 ознаками	0,0002581239572394642	0,00024789895139848823
Запропонований метод (дерево рішень з 3 ознаками)	0,0002563973463753253	0,0002478683712292277
Регресійна модель на основі методу k-найближчих сусідів	0,00029035257843259643	0,00024834775563353
Метод опорних векторів	0,06897435748210325	0,004941252562452996

13

Рисунок В.13 – Результати прогнозування вартості вживаних мотоциклів

ВИСНОВКИ

- Створена програма призначена для забезпечення підтримки продажу вживаних мотоциклів шляхом надання відповідних засобів для виставлення власних вживаних мотоциклів на продаж, пошуку вживаних мотоциклів для придбання. Мотоцикли представлені виробниками, типами, моделями, версіями цих моделей, що визначаються представниками сервісу продажу вживаних мотоциклів, а вже на основі цих даних визначаються оголошення про продаж вживаних мотоциклів самими власниками.
- Наукова новизна роботи полягає в методі прогнозування вартості вживаних мотоциклів, який базується на створенні та навчанні дерева рішень, визначає процес створення і відбору вхідних ознак, що дозволяє встановити вартість вживаного мотоцикла на етапі формування оголошення.

14

Рисунок В.14 – Висновки