МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
Факультет електроніки та інформаційних технологій

Кафедра комп‘ютерних наук

Кваліфікаційна робота магістра

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ МАШИННОГО НАВЧАННЯ ДЛЯ ДЕТЕКТУВАННЯ ТА РОЗПІЗНАВАННЯ ТЕКСТУ В ВЕБ-СИСТЕМІ ДЛЯ АВТОСЕРВІСУ

Здобувач освіти гр. ІН.м – 12ан Марина ФІЛАТОВА

Науковий керівник,
доцент, к.т.н. Наталія БАРЧЕНКО

В. о. завідувача кафедри
доцент, к.т.н. Ігор ШЕЛЕХОВ

Суми 2022

Сумський державний університет
(назва вузу)

Факультет *ЕлІТ* Кафедра *Комп'ютерних наук*
Спеціальність *«122 - Комп'ютерні науки»*

Затверджую:
В.о. зав.кафедри_____
_____
"_____" _____ 20___р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ МАГІСТРА

Філатовій Марині Володимирівні
(прізвище, ім'я, по батькові)

1. Тема роботи *Інформаційна технологія машинного навчання для детектування та розпізнавання тексту в веб-системі для автосервісу*

затверджую наказом по інституту від "____" _____20___ р. № _____

2. Термін здачі здобувачем вищої закінченого роботи_____

3. Вхідні дані до роботи_____
_____
_____
_____

4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)

1) Аналіз проблеми. Огляд існуючих рішень. Огляд алгоритмів 2) Постановка задачі й формування завдань дослідження. 3) Вибір методів рішення задачі. 4) Моделювання та проєктування системи. 5) Практична реалізація

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) _____
_____

6. Консультанти до проекту (роботи), із значенням розділів проекту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання _____

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів дипломного проекту (роботи)	Термін виконання проекту (роботи)	Примітка
1.	Аналіз проблеми. Огляд існуючих аналогів. Огляд алгоритмів		
2.	Постановка завдання та вибір методів реалізації		
3.	Моделювання та проєктування вебдодатку.		
4.	Практична реалізація.		
5.	Оформлення пояснювальної записки до дипломної роботи		

Студент – дипломник _____
(підпис)

Керівник проекту _____
(підпис)

РЕФЕРАТ

Записка: 60 стор., 32 рис., 3 додатків, 25 джерел.

Об'єкт дослідження — бізнес-процес з надання сервісних послуг та ремонту автомобілів.

Предмет дослідження — засоби автоматизації бізнес-процесу та засоби детектування зони інтересу на зображенні з використанням машинного навчання.

Мета роботи — розробити веб-систему для автоматизації бізнес-процесів сервісного обслуговування та ремонту автомобілів для замовників та сервісів з використанням алгоритмів машинного навчання для детектування й розпізнавання номерних знаків автомобілів з подальшим пошуком в базі даних автомобілів ДАІ. Дана система повинна бути швидкою, простою в дизайні і зрозумілою для користувача. Сайт повинен бути адаптивним до всіх пристроїв, також повинен працювати у всіх сучасних веббраузерах.

Результати — виконано вибір методів вирішення поставленої задачі; на основі фреймворку Laravel, мови Python з використанням моделей машинного навчання та алгоритмів OCR, розроблено веб-систему, що автоматизувала основні процеси автосервісу та може розпізнавати номерний знак на зображенні й діставати всю інформацію про знайдене авто через API актуальної на даний момент бази автомобілів ДАІ України.

ІНФОРМАЦІЙНА СИСТЕМА, АВТОСЕРВІС, LARAVEL, HTML, CSS, JAVASCRIPT, AJAX, JSON, API, PYTHON, OCR

ЗМІСТ

ВСТУП

Сьогодні не буде перебільшенням сказати, що кожен із нас стикається з машинним навчанням кілька разів на день — мобільний банкінг, спливаючі вікна з головними новинами, рекомендований вміст у соціальних мережах, чат-боти — список можна продовжувати. Компанії майже в усіх галузях розгортають рішення на основі машинного навчання для підвищення продуктивності, прийняття рішень, інноваційних продуктів і послуг тощо.

Будучи розширенням, а не заміною людських здібностей, машинне навчання дозволяє компаніям автоматизувати складні процеси, покращувати якість, ефективність і креативність рішень співробітників за допомогою багатих можливостей аналітики та прогнозування шаблонів, а також виявляти прогалини та можливості на ринку для впровадження нових продуктів і послуг, гіперперсоналізація клієнтського досвіду та багато іншого.

Сервісний центр (СЦ) — це організація, що займається наданням послуг з сервісної підтримки та обслуговування техніки, обладнання та іншої продукції [1].

На поточний момент облік робіт з ремонту та обслуговування техніки проводиться програмним забезпеченням на базі застарілих технологій та обмеженого функціоналу або взагалі не фіксується. Внаслідок цього суттєво збільшується час на обслуговування клієнтів та неможливість оперативного отримання даних про хід робіт.

Тому для поліпшення ситуації вирішено автоматизувати процеси сервісного центру, запровадивши інформаційну систему (ІС). У випадку СЦ, вона надасть співробітникам оперативну інформацію про хід виконання ремонтних робіт, зайнятість майстрів та грошовий оборот. Така інформація дозволить правильно приймати рішеня у процесі роботи підприємства. ІС дозволить автоматизувати та стандартизувати управління відносин з клієнтами.

Метою даної роботи є розробка веб-системи для керування автосервісами, яка пропонує автосервісам можливість автоматизувати щоденні завдання та робочі процеси, а клієнтам — бути впевненими в якості обслуговування, прозорості та чесності підприємства.

Майстри автосервісів, які будуть використовувати автоматизовану систему, значно полегшать собі роботу. Це допоможе їм зосередитися на більшій кількості завдань, оскільки системи беруть під контроль різноманітні ручні завдання, які займають багато часу, але не забезпечують високої якості.

Доступ до такої інформації, як ідентифікаційні номери транспортних засобів і всі деталі автомобілів, також робить процес швидшим і прозорішим.

Замість того, щоб вручну зв’язуватися з водіями та майстернями з ремонту авто, менеджери автосервісів отримують постійну інформацію про процес ремонту та попереджені, якщо щось потребує їхньої уваги. Усі ці функції в сукупності скорочують час, який менеджери витрачають на спілкування та координацію, дозволяючи швидше та економічно ефективніше ремонтувати за рахунок мінімізації діяльності, яка не має додаткової вартості.

Настав час відмовитися від застарілих методів, і досвідчені механіки та автовласники тепер зможуть отримати доступ до будь-якої інформації про заявки на ремонт чи сервісне обслуговування, просто на своєму смартфоні.

ІНФОРМАЦІЙНИЙ ОГЛЯД

1.1 Дослідження предметної області

Клієнти завжди вимагали прозорості в ремонті автомобілів. Однак зараз, як ніколи, клієнти наполягають на тому, щоб точно знати, що відбувається з їхніми автомобілями. Те, як автомобільні технічні спеціалісти впораються з цією потребою, може вплинути на успіх ремонтної майстерні.

Клієнти хочуть прозорості не лише в ціноутворенні послуг з ремонту, але й у фактичних послугах, які виконуються [2]. Програмне забезпечення для керування взаємовідносинами з клієнтами автосервісу та автоматизації його процесів може подолати цей розрив і показати роботу, виконану на кожному автомобілі.

За допомогою програмного забезпечення для керування автосервісом ремонт автомобілів стане більш прозорим і завоює довіру клієнтів странції технічного обслуговування (СТО). Програмне забезпечення для управління сервісним центром використовуватиме потужність сучасного спілкування за допомогою фотографій, відео, текстових повідомлень і чату. Як наслідок, програмне забезпечення для ремонту автомобілів вирішує проблеми, збільшує обсяг продажів і генерує більшу ефективність і прибуток.

Отже, впровадження програмного забезпечення для СТО надасть:

– повну автоматизацію ремонту автомобілів — з моменту отримання заявки на сайті або по телефону до видачі авто;

– спрощене узгодження дій майстрів із клієнтом;

– мінімізацію рутинних завдань — автоматична підготовка замовлення нарядів, оформлення замовлень на запчастини, укладання договорів, виписка рахунків.

– спілкування з клієнтами, постачальниками та представниками страхових із закладених скриптів для прискорення відповідей.

– формування карт ремонту, постановка завдань менеджерам та

майстрам, зміна завдань для виконавців.

– прозорий розрахунок заробітної плати працівників [3].

CRM-системи — це ПЗ (програмне забезпечення) для управління взаємовідносинами з клієнтами.

ПЗ CRM є особливою системою з основною метою — покращення відносин з клієнтами шляхом надання бізнесу інтелектуальних інструментів для ефективної розробки успішної стратегії. Іншими словами, ПЗ CRM може пришвидшити важливі бізнес-операції, такі як: маркетинг, обслуговування та продажі — щоб підвищити якість обслуговування і лояльність клієнтів від першої взаємодії до наступних.

1.2 Огляд веб-систем сервісних центрів

Розробка нового продукту вимагає попередньої постановки цілей, завдань, а також аналізу ринкової ніші. Одним із завдань дипломного проєкту є розробка веб-орієнтованої системи для СТО. Тому, перед початком розробки потрібно провести дослідження існуючих веб-систем сервісних центрів.

Для досягнення мети було проведено дослідження наступних систем подібної тематики:

- autoleap.com
- mitchell1.com
- garageplug.com

Один з рессурсів подібного призначення — autoleap.com. AutoLeap надає потужне комплексне програмне забезпечення для автосервісів, яке дозволяє автосервісам керувати малим бізнесом, техніками з турбонаддувом [4]. Це хмарне рішення з такими динамічними функціями, як розумне планування, виставлення рахунків одним клацанням миші, інтелектуальна робоча дошка, інтеграція QuickBooks, керування Google Reviews, розширені звіти (рис. 1.1).

До переваг цієї системи можна віднести простоту у використанні та службу підтримки клієнтів.

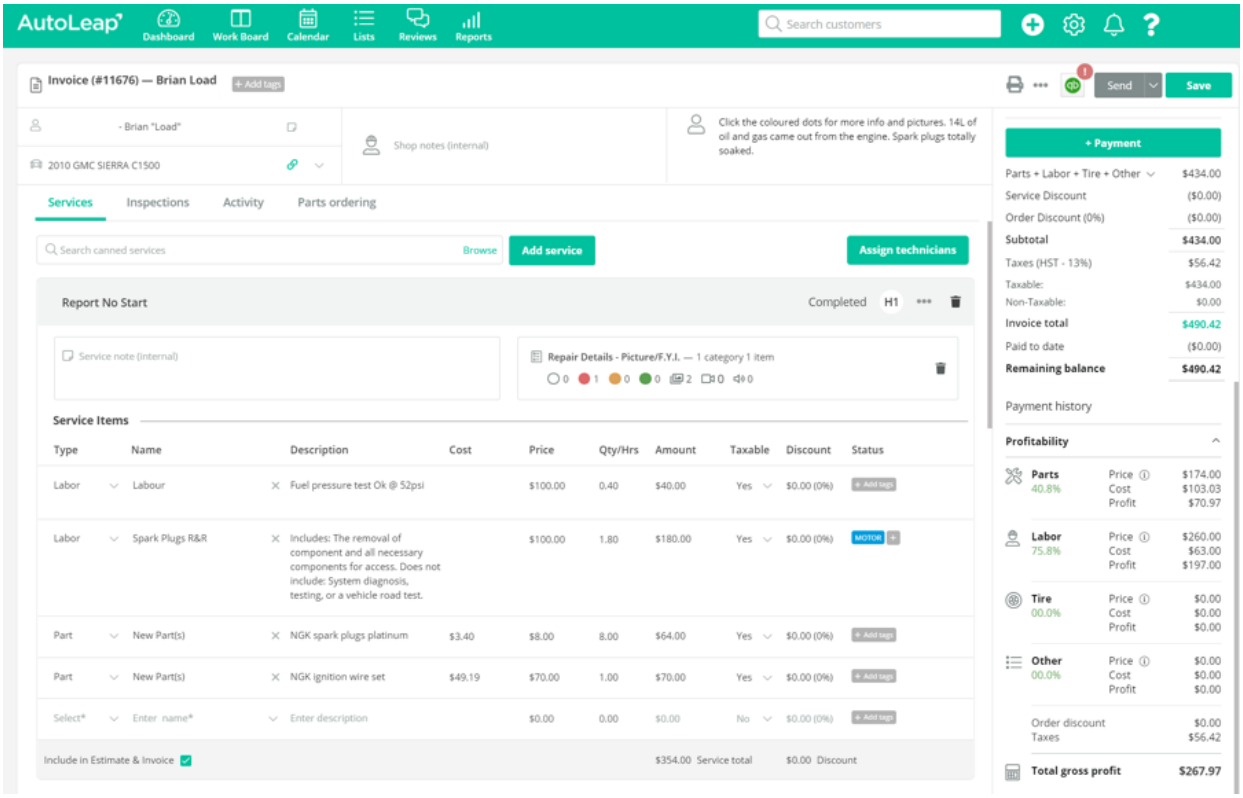

Рисунок 1.1 — Головна сторінка autoleap.com

Але представлена система має деякі недоліки. До них можна віднести неможливість здійснити платіж для клієнта, система автоматично не надає цю інформацію в квитанції. Іноді в системі виникають невеликі збої, коли деякі функції не працюють одразу, чи потрібен деякий час, щоб завантажити сторінки.

Наступним для аналізу ресурсом було обрано Mitchell 1 Automotive Repair — mitchell1.com [5]. Це програмне забезпечення для ремонту автомобілів і керування гаражем, що допомагає в аналізі, управлінні і маркетингу (рис. 1.2).

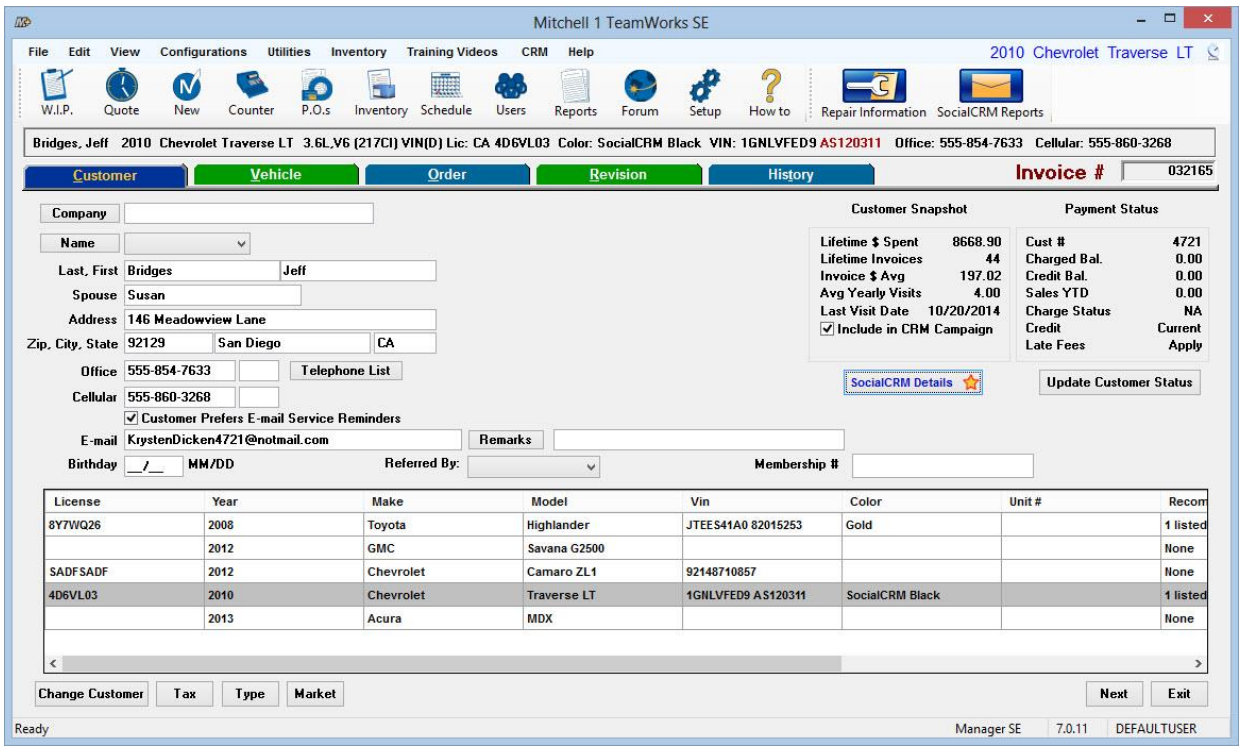

Рисунок 1.2 — Головна сторінка mitchell1.com

Представлена система має декілька переваг, а саме прості макети розкладу для календарів і звітів, одночасний перегляд всіх робочих замовлень та можливість впорядковувати їх за багатьма різними факторами. Також автоматичне виставлення рахунків.

Проте дана система містить значну кількість недоліків. До них можна віднести швидкість роботи, застарілість дизайну, складність навігації у вікнах довідки.

Наступну систему для аналізу було обрано GaragePlug — garageplug.com (рис. 1.3). Це платформа для автосервісів різних брендів для управління та відстеження бізнесу [6]. У ній є такі модулі, як контроль запасів, сканування штрих-кодів, зворотний зв’язок із послугами, нагадування, зустрічі, облік, перевірка, бізнес-звіти.

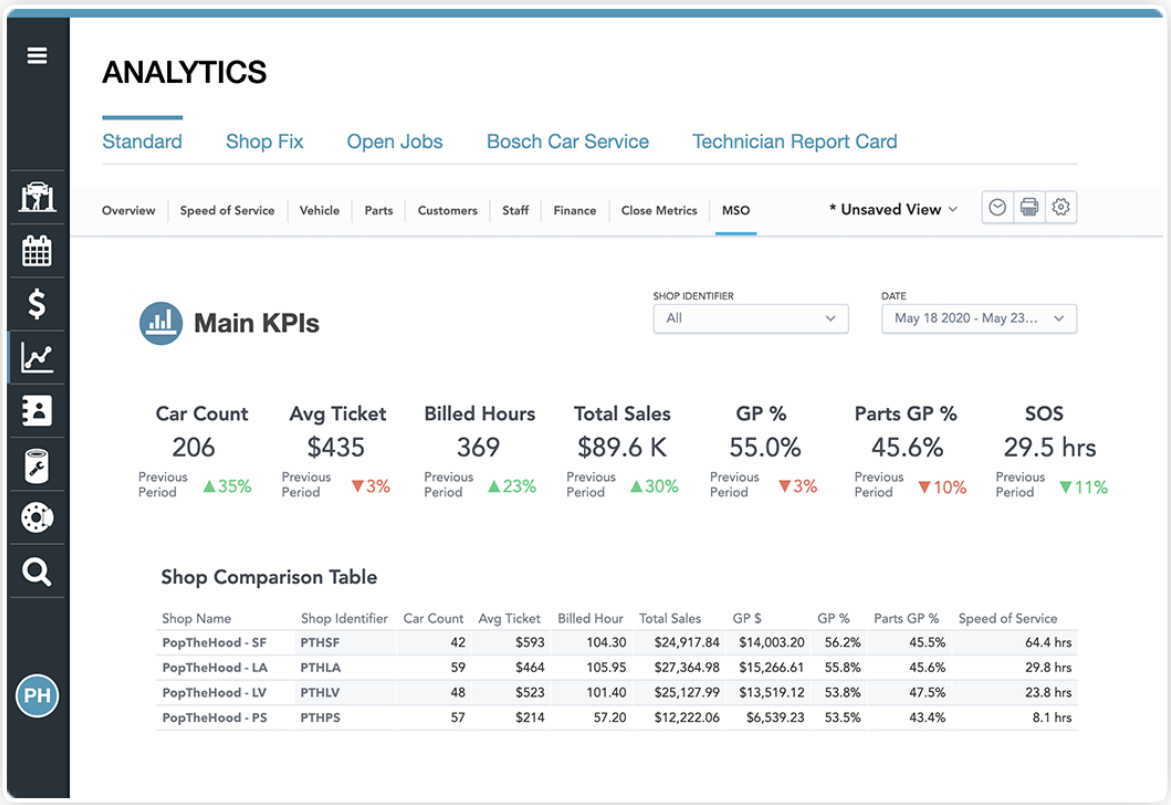

Рисунок 1.3 — Головна сторінка garageplug.com

Система має такі переваги, як відстеження в режимі реального часу, можна використовувати на будь-якому пристрої, звіти, створені за потреби. Але є і суттєві недоліки, такі, як складність у використанні, відсутність зручної панелі навігації.

Все ж, незважаючи на численні CRM, індивідуальне рішення завжди виграє, і ось чому:

– індивідуальні інструменти дозволяють вирішувати складні завдання;

– відсутність великої кількості опцій і функцій, з яких випливає велика кількість полів та іконок;

– високий рівень безпеки;

– зручний та унікальний інтерфейс для співробітників та автовласників;

– окремі маркетингові інструменти для просування бізнесу.

1.3 Огляд алгоритмів

Найбільш переважними підходами до виявлення об’єктів є машинне або глибоке навчання. Плюсом використання алгоритму машинного навчання для виявлення об’єктів є те, що він покладається на введені вручну дані для класифікації, а не на дані автоматичного навчання. Це робить загальний алгоритм менш схильним до помилок і більш стабільним.

Виявлення об’єктів — це проблема контрольованого машинного навчання, що означає, що потрібно використовувати попередньо навчені моделі для запуску детекторів об’єктів. Список класів у навчальному наборі даних алгоритму машинного навчання має належати певному зображенню або списку зображень.

Розпізнавання номерних знаків — використання технології виявлення об’єктів і оптичного розпізнавання символів (OCR) для розпізнавання буквено-цифрових символів на автомобілі [7]. Можна використовувати функцію виявлення об’єктів для захоплення зображень і виявлення транспортних засобів на певному зображенні. Коли модель розпізнає номерний знак, технологія оптичного розпізнавання символів перетворює двовимірні дані в машинно-кодований текст.

Оскільки одним із завдань даної роботи є розпізнавання номерного знаку авто, необхідно провести дослідження алгоритмів, за допомогою яких можна вирішиту цю задачу.

По-перше необхідно знайти зону інтересу (номерний знак на зображенні). Для вирішення цієї задачі є декілька підходів:

– алгоритми машинного зору;

– за допомогою нейронних мереж для обробки графічних зображень (задачі детектування об'єктів або сегментації зони інтересу).

По-друге необхідно розпізнати номер. Дану задачу можна реалізувати через оптичне розпізнавання символів (OCR).

Найбільш популярні алгоритми, які використовуються для виявлення об’єктів, включають згорточні нейронні мережі (R-CNN, згорткові нейронні мережі на основі регіону), Faster R-CNN і YOLO (You Only Look Once). R-CNN належать до сімейства R-CNN, а YOLO — до сімейства одноразових детекторів [8].

You only look one (YOLO) — одна з найпопулярніших модельних архітектур і алгоритмів виявлення об’єктів. Це передбачає використання єдиної нейронної мережі, навченої наскрізно, для отримання фотографії як вхідних даних і прогнозування обмежувальних рамок і міток класів безпосередньо для кожної обмежувальної рамки [9].

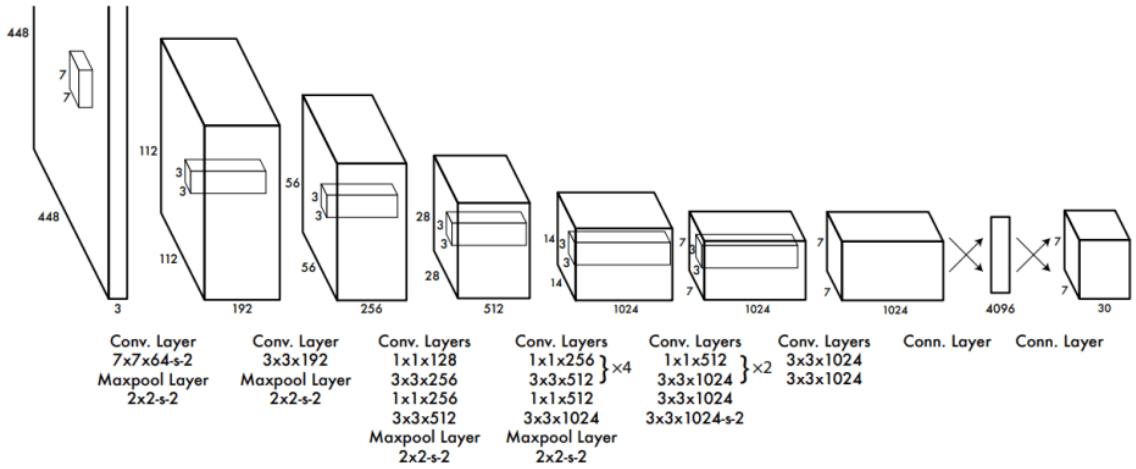

Рисунок 1.4 — Процес роботи YOLO

До переваг YOLO можна віднести швидкість обчислень і обробки особливо в режимі реального часу (в порівнянні з більшістю інших методів навчання та алгоритмів виявлення об’єктів), високу точність із зменшенням фонових помилок, які спостерігаються в інших методах. Архітектура YOLO дозволяє моделі вивчати та розвивати розуміння численних об’єктів більш ефективно.

Проте даний алгоритм має такі обмеження, як нездатність виявити менші об’єкти на зображенні чи відео через нижчу швидкість запам’ятовування, неможливість виявити два об’єкти, які знаходяться дуже близько один до одного через обмеження обмежувальних рамок.

Згорточні нейронні мережі на основі регіонів або регіони з функціями CNN (R-CNN) є новаторськими підходами, які застосовують глибокі моделі для виявлення об’єктів. Моделі R-CNN спочатку вибирають кілька запропонованих регіонів із зображення (наприклад, опорні рамки є одним із типів методу вибору), а потім позначають їхні категорії та обмежувальні рамки (наприклад, зміщення). Ці мітки створюються на основі попередньо визначених класів, наданих програмі. Потім вони використовують згортову нейронну мережу для виконання прямих обчислень, щоб витягти характеристики з кожної запропонованої області. У R-CNN введене зображення спочатку ділиться на майже дві тисячі областей, а потім для кожної області застосовується згортка нейронної мережі відповідно. Обчислюється розмір областей, і правильна область вставляється в нейронну мережу. Ще одним серйозним недоліком моделі R-CNN є не тільки низька швидкість навчання, але й великий час прогнозування. Рішення вимагає використання великих обчислювальних ресурсів, що підвищує загальну здійсненність процесу. Отже, загальну архітектуру можна вважати досить дорогою.

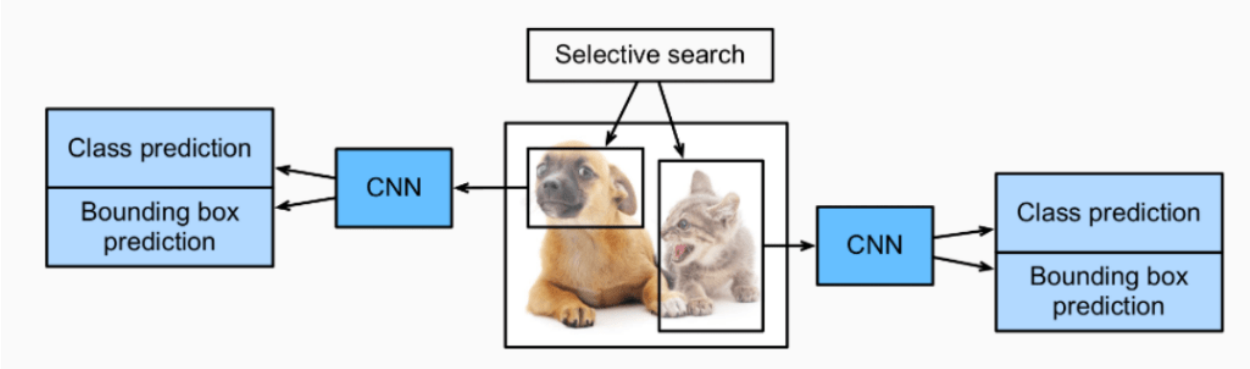

Рисунок 1.5 — Процес роботи R-CNN

Можна зробити висновок, що такий детальний метод може спричинити обмеження часу. Час навчання значно довший порівняно з YOLO, оскільки він класифікує та створює обмежувальні рамки окремо, а нейронна мережа застосовується до однієї області за раз.

Незважаючи на те, що модель R-CNN змогла виконати обчислення

виявлення об’єктів і досягти бажаних результатів, були деякі основні недоліки, особливо швидкість моделі. Отже, щоб подолати проблеми, які існували в R-CNN, потрібно було запровадити швидші методи вирішення деяких із цих проблем. По-перше, Fast R-CNN було запроваджено для боротьби з деякими існуючими проблемами R-CNN [10].

У швидкому методі R-CNN все зображення пропускається через попередньо підготовлену згортову нейронну мережу замість того, щоб розглядати всі підсегменти. Об’єднання регіону інтересу (RoI) — це спеціальний метод, який використовує два входи попередньо навченої моделі та алгоритм вибіркового пошуку, щоб забезпечити повністю зв’язаний рівень із виходом.

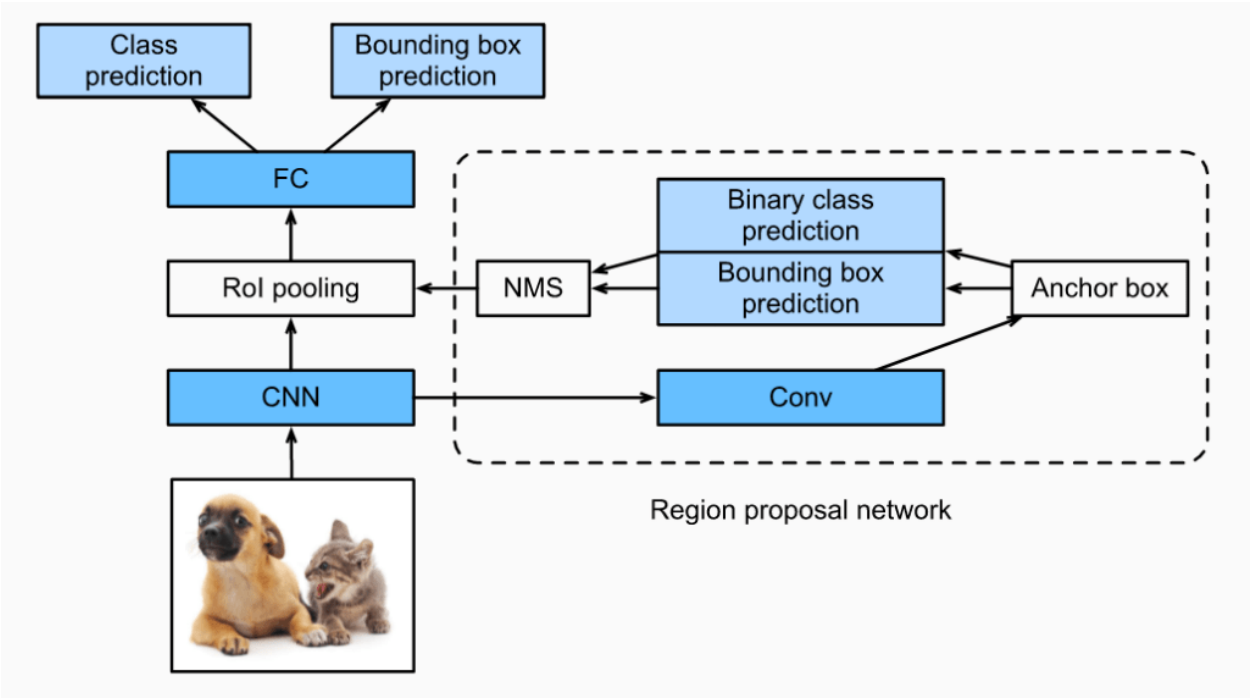

Рисунок 1.6 — Процес роботи Faster R-CNN

Одноразовий детектор (SSD) для передбачень у кількох вікнах є одним із найшвидших способів досягти обчислення завдань виявлення об’єктів у реальному часі. Хоча методології Faster R-CNN можуть досягти високої точності прогнозування, загальний процес займає досить багато часу, і вимагає виконання завдання в реальному часі зі швидкістю приблизно 7 кадрів на секунду, що далеко не бажано.

SSD вирішує цю проблему, покращуючи кількість кадрів за секунду майже в п’ять разів більше, ніж у моделі Faster R-CNN [11]. Він усуває використання мережі регіональних пропозицій і натомість використовує багатомасштабні функції та вікна за замовчуванням.

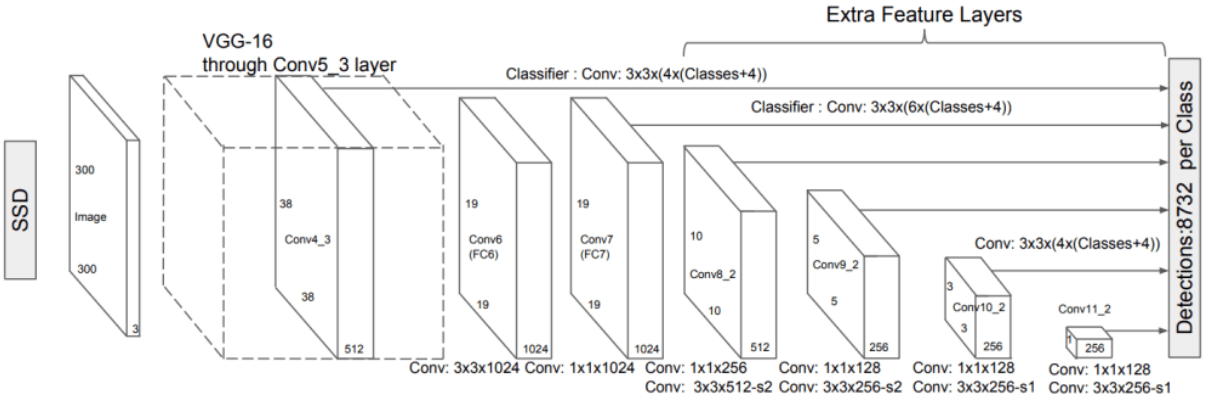

Рисунок 1.7 — Процес роботи SSD

Для невеликих об’єктів архітектура SSD зазвичай працює гірше, ніж Faster R-CNN. Одноразовий детектор часто є кращим методом. Основною причиною використання одноразового детектора є те, що ми переважно віддаємо перевагу швидшим передбаченням на зображенні для виявлення більших об’єктів, де точність не є надзвичайно важливою проблемою. Однак для більш точних прогнозів для менших і точних об’єктів необхідно розглянути інші методи.

Optical Character Recognition (OCR) — це процес розпізнавання символів на зображеннях з використанням комп'ютерного зору та методів машинного навчання [12].

Як раніше було зазначено, першим етапом є виявлення (локалізація) тексту: на цьому етапі використовуються такі моделі, як Mask-RCNN, YoloV5, SSD тощо, які визначають місцезнаходження тексту на зображеннях. Ці моделі зазвичай створюють обмежувальні рамки (квадратні або прямокутні рамки) над кожним текстом, визначеним на зображенні чи документі.

Наступним етапом є розпізнавання тексту: після визначення розташування тексту кожен обмежувальний квадрат надсилається до моделі розпізнавання тексту, яка зазвичай є комбінацією мереж RNN, CNN. Остаточним результатом цих моделей є текст, витягнутий з документів. Деякі моделі розпізнавання тексту з відкритим кодом, такі як Tesseract, MMOCR, TrOCR, EasyOCR або Google Cloud Vision API тощо, можуть допомогти досягти високої точності. Можна навчити таку модель за допомогою власних даних (можна наслідувати їхній приклад набору даних, щоб відформатувати свій власний набір даних) або використовувати існуючі моделі для обслуговування власної програми [13].

1.4 Постановка задачі

Інформаційна система має забезпечувати автоматизацію роботи автосервісу з використанням алгоритмів машинного навчання для детектування й розпізнавання номерних знаків автомобілів з подальшим пошуком в базі даних автомобілів ДАІ. Проєкт повинен бути реалізований у вигляді веб-додатку, доступного в мережі Інтернет. Сайт повинен складатися із взаємозалежних розділів із чітко розділеними функціями.

Інформаційна система повинна мати наступні можливості:

- перегляд повної інформації про сервіс (послуги, що надаються та їх вартість);
- розпізнавати номерні знаки та зберігати в базі даних системи всю інформацію про знайдені авто;
- формувати заявки на ремонт;
- автоматично формувати ціну за ремонт та сервісне обслуговування;
- додавати фото-/відеозвіт нової проблеми в процесі ремонту для автовласника;
- можливість реєстрації та авторизації автовласників;
- 3 типи користувачів (адміністратор, майстер та автовласник);

- отримувати звіти по кожній заявці по наданим послугам з описом використаних деталей;
- здійснювати пошук в системі за різними параметрами;
- залишати відгуки;
- записуватись на ремонт.

2 ВИБІР МЕТОДІВ ВИРІШЕННЯ ЗАДАЧІ

2.1 Вибір методів розроблення

CRM — це, насамперед, підвищення ефективності та зниження ризиків для організації. Коли взаємодія з клієнтами інтегрується в стратегію компанії, це індивідуальне рішення CRM, яке зв’язує всіх зацікавлених сторін з точками даних, одночасно дозволяючи оптимізувати процеси.

Розробники CRM часто комплектують свої готові рішення великою кількістю функцій. Багато з них спрямовані на охоплення більшої аудиторії. Це робить інтерфейс менш інтуїтивно зрозумілим і створює проблеми з навчанням персоналу. CRM-рішення від постачальників повинні містити лише ті функції, які потрібні бізнесу. В іншому випадку можна витратити багато грошей на налаштування та навчання персоналу.

Створення CRM з нуля може принести більше користі бізнесу. На даний момент 90% розробників програмують CRM мовою PHP, решта 10% використовують якісь інші мови або готові програмні продукти, наприклад, від Microsoft SharePoint, які абсолютно не бюджетні [14]. Мова PHP дуже поширена серед програмістів і тут є вибір. Хтось пише на чистому PHP, а хтось пише, використовуючи фреймворки.

Якщо писати на чистому PHP, то рівень розробника має бути дуже і дуже високим, інакше замовник може отримати поганий код (з поганою архітектурою: він не продуманий та не задокументований, тому підтримувати його буде вкрай складно). Отже, розробка веб-системи для СТО відбуватиметься із застосуванням фреймворку PHP.

PHP — це серверна сценарна мова з відкритим кодом, яку багато розробників використовують для веброзробки. Це також мова загального призначення, яку можна використовувати для створення багатьох проєктів, у тому числі графічних інтерфейсів користувача (GUI).

Фреймворк PHP — платформа для створення вебдодатків на PHP. Він містить бібліотеки з елементами для реалізації шаблонів розробки програмного забезпечення [15].

До переваг використання фреймворків можна віднести: розробку сайту будь-якої складності і функціоналу, високу продуктивність, фреймворки можна використовувати паралельно з сайтом, розробленому на CMS (система управління контентом), наявність якісної і повної документації з прикладами, можливість тестування.

2.2 Вибір засобів програмування

Для створення інформаційних систем використовуються комбінації HTML для базової структури сторінки та вмісту, CSS, для візуального редагування та JavaScript для надання інтерактивності та певного функціоналу вебсторінкам.

З мови програмування на стороні клієнта JavaScript перетворився на повноцінну мову веб-розробки. Він стабільно утримує позицію однієї з найпопулярніших сучасних мов програмування. А JavaScript-фреймворки підвищують зручність його використання. Це зручний інструмент для швидкої та легкої розробки. Вони зазвичай містять набір компонентів, які можна використовувати для створення будь-якої веб-програми.

React — це фреймворк з відкритим кодом, що належить Facebook (Meta) [16]. Це вільний і декларативний фреймворк, що широко використовується для розробки інтуїтивних інтерфейсів для веб-додатків. Крім того, React має невеликі пакети, тому легко освоїти новачкам. React підходить для розробки великомасштабних веб-застосунків, що використовують різні дані. Інші переваги React полягають у:

– доступності повторного використання його компонентів у різних частинах web-додатку;

– наявності одностороннього потоку інформації;

– достатньо великому співтоваристві, яке забезпечує стабільність

бібліотеки;

– універсальності та можливості застосування з іншими фреймворками.

Що стосується недоліків, то вони полягають у:

– застосуванні складного розширення — JSX;

– ризику впливу його динаміки на SEO-оптимізацію;

– підтримці лише результатів зовнішнього користувальницького інтерфейсу.

Фреймворк Angular також має відкритий вихідний код і оптимальний для розробки односторінкових вебдодатків, вебсайтів, настільних та мобільних додатків. Створений розробниками Google ще в 2009-му році, він побудований мовою TypeScript, завдяки якому його функціонування відрізняється плавністю і високою ефективністю [17].

Будучи одним із найкращих фреймворків 2021-го року, він порадує цілою низкою переваг:

– доступність простого перенесення та взаємодії між компонентами системи за рахунок інтегрування залежностей;

– здатність самостійної зміни HTML-коду завдяки автоматичному зчитуванню змін на рівні моделі;

– наявність інтегрованої системи REST;

– можливість вбудовування модульних тестів та проведення тестування;

– простота та зручність роботи з цілим рядом зовнішніх бібліотек.

Що стосується недоліків, то вони полягають у:

– необхідності попереднього досвіду роботи у програміста з TypeScript;

– необхідність застосування зовнішніх інструментів для повної індексації вебсайту;

– уповільнення роботи програмістів та зниження її ефективності через

одночасне завантаження скриптів під час відкриття програми.

Frontend-фреймворк Vue — ще один веб-фреймворк з відкритим вихідним кодом, що входить до топ 2021 року. За період свого існування пережив кілька оновлень, відрізняється можливістю легкої інтеграції до проектів із залученням інших бібліотек JS [18]. Його переваги незаперечні:

– універсальність та здатність функціонувати як JS-framework та поєднувати в собі такі інструменти, як Angular, React;

– наявність серверного рендерингу на основі "React" та "Angular";

– легкість у порівнянні з цілим рядом попередників;

– робота, побудована на концепції програмування віртуальної DOM та використання двосторонньої прив'язки;

– забезпечення високої продуктивності та максимальної ефективності не тільки одно-, а й багатосторінкових сайтів;

– інтуїтивна зрозумілість функцій фреймворку та простота його синтаксису, що полегшує роботу з ним тих, хто робить перші кроки у програмуванні;

– підтримка «пластичності» програмування завдяки зрозумілому та читальному коду;

– підтримка використання машинописного тексту.

Недоліки даного фреймворку:

– недостатньо велика спільнота;

– надмірно велика кількість компонентів;

– відсутність перекладу цілої низки нових плагінів.

Проаналізувавши переваги та недоліки вищеописаних веб-фреймворків, для розробки веб-системи було обрано останній веб-фреймворк.

Оскільки методом розробки веб-системи було обрано фреймворк мови PHP, далі буде розглянутий такий фреймворк, як Laravel. За даними багатьох

інформаційних порталів рейтинг найкращих PHP-фреймворків очолює саме Laravel (див.рис. 2.1).

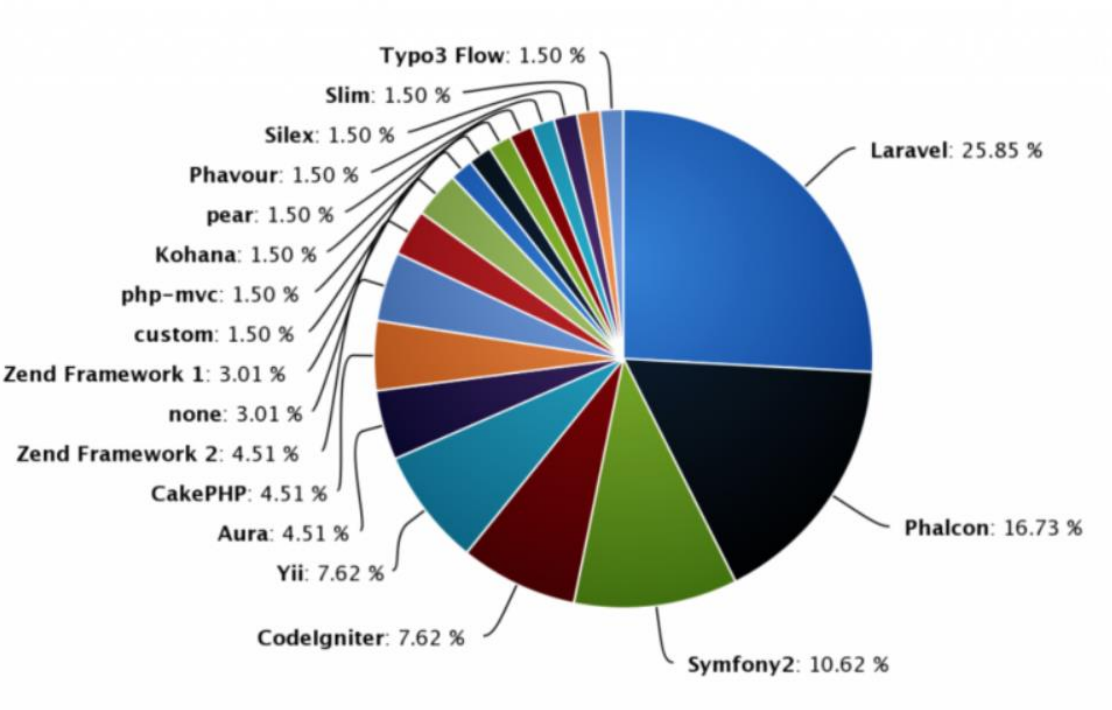

Рисунок 2.1 — Рейтинг популярності PHP-фреймворків

Laravel — це безкоштовний фреймворк PHP з відкритим вихідним кодом, створений Тейлором Отвеллом для розробки вебдодатків за архітектурним шаблоном MVC [19]. Laravel є найсильнішим суперником в екосистемі PHP просто тому, що він включає функції, необхідні для створення сучасних, підтримуваних, розподілених вебдодатків в реальному часі.

Переваги використання Laravel:

– Ком'юніті — Велика спільнота підтримує фреймворк.
– MVC — Розділення коду покращує його підтримуваність.
– ORM — Загальні підходи та доступність даних у програмі.
– Шаблонізатор — Уніфікація відображення програми.
– Вбудовані механізми аутентифікація та інтеграція з сервісами.
– PHPUnit-тести — Автотести для впевненості в якості коду.
– Модульність — Велика кількість готових рішень економить час на

типових завданнях.

– Продуктивність — Висока швидкість завантаження. Завдяки кешуванню, оптимізації фронтенду та правильному поділу коду на компоненти вдається забезпечувати справді швидкий доступ до даних.

– Мовні версії.

– Помилки та винятки — під контролем. Обробка помилок та виключень доступна «з коробки» для будь-якого нового проєкту на Laravel. Крім того, Laravel інтегрований з бібліотекою журналів Monolog, яка забезпечує підтримку багатьох потужних обробників журналів.

– Система міграцій БД — Дозволяє оновити базу даних автоматично. Зручна та безпечна робота з базами даних. Захист від SQL-ін'єкцій, захист від підробки міжсайтових запитів (CSRF), захист від XSS атак та ін.

Для реалізації ідей, пов'язаних із застосуванням машинного навчання, знадобиться надійна, гнучка мова програмування з багатим інструментарієм. Python — саме така мова, і тому сьогодні нею розробляється безліч проєктів.

Python допомагає розробникам працювати продуктивно та впевнено, причому на всіх стадіях проєкту. Ця мова має певні характеристики, які роблять її найкращим вибором: вона проста і логічна, гнучка, має відмінні бібліотеки та фреймворки для машинного навчання, а ще за нею стоїть численна спільнота розробників. Завдяки цьому Python є однією з найпопулярніших мов програмування у світі.

Численні фреймворки та бібліотеки Python допомагають суттєво зменшити кількість часу, необхідного для розробки програм [20]. Python має широкий набір бібліотек для штучного інтелекту та машинного навчання:

– Keras, TensorFlow і Scikit-learn — для машинного навчання;
– NumPy — для високопродуктивних наукових обчислень та аналізу;
– SciPy — для розвинених обчислень;
– Pandas — для загального аналізу даних;
– Seaborn — для візуалізації даних.

ПРОЄКТУВАННЯ ТА МОДЕЛЮВАННЯ

3.1 Структурно-функціональне моделювання діяльності СТО

Метод моделювання функцій (IDEF0) — це метод, призначений для моделювання рішень, дій і діяльності організації чи системи. Це описові моделі, які показують активність процесу високого рівня. Модель вказує на основні види діяльності та вхід, контроль, вихід та механізми, пов’язані з кожним основним видом діяльності. Моделі IDEF0 дозволяють розробнику моделей відобразити уявлення про процес, входи, елементи керування над процесом, виходи та механізми, що діють на процес.

Далі буде розглянуто головний напрямок діяльності СТО — обслуговування та ремонт авто. Контекстну діаграму даного процесу представлено на рис. 3.1.

Рисунок 3.1 — IDEF0 інформаційної системи

Метод інформаційного моделювання — IDEF1 використовується для створення інформаційної моделі, яка представляє структуру інформації, необхідної для підтримки функцій виробничої системи або середовища.

Дана методологія дозволяє на основі простих графічних зображень моделювати інформаційні взаємозв'язки та відмінності між:

– реальними об'єктами;

– фізичними та абстрактними залежностями, що існують серед

реальних об'єктів;

– інформацією, що стосується реальних об'єктів;

– структурою даних, що використовується для придбання, накопичення, застосування та керування інформацією.

Однією з основних переваг методології IDEF1 є забезпечення послідовного та строго структурованого процесу аналізу інформаційних потоків у рамках діяльності підприємства. Іншою відмінною властивістю IDEF1 є широко розвинена модульність, що дозволяє ефективно виявляти та коригувати неповноту та неточності існуючої структури інформації, протягом всьому етапу моделювання. Головна батьківська діаграма на першому рівні деталізації моделі декомпозується на блоки (див рис. 3.2).

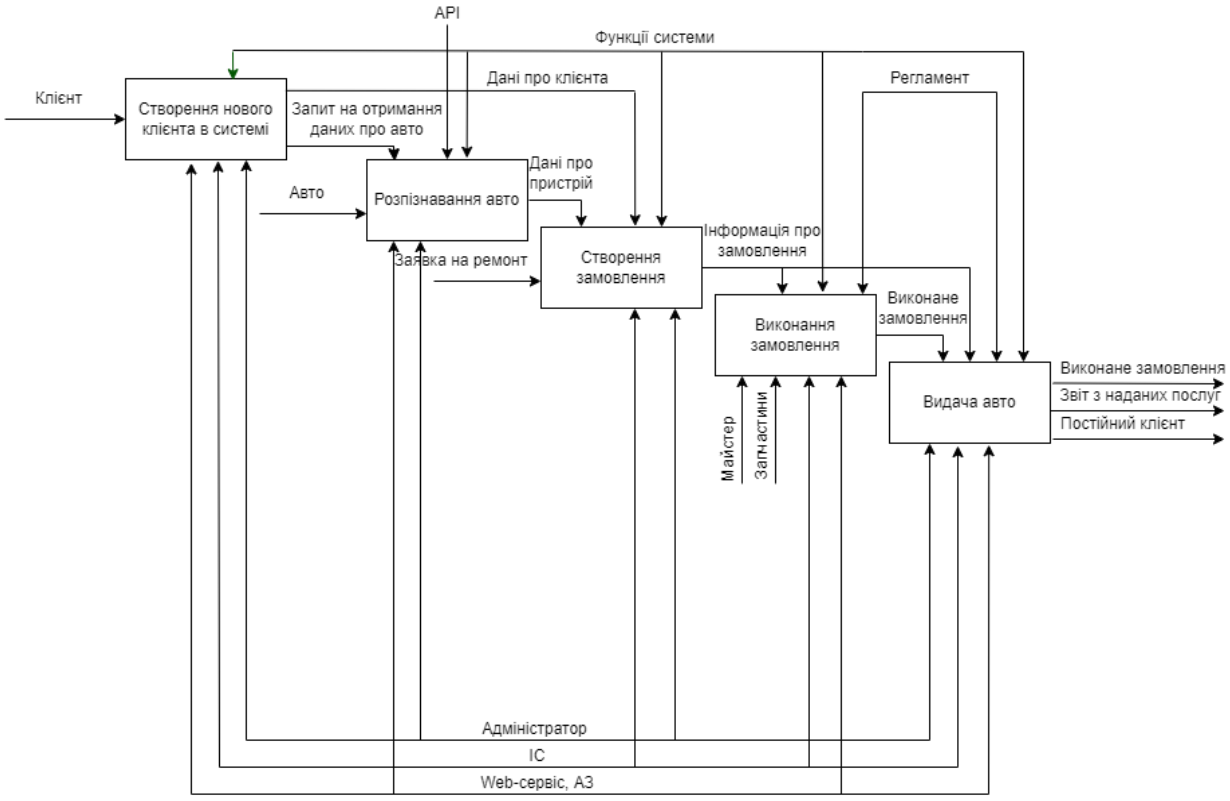

Рисунок 3.2 — Діаграма декомпозиції IDEF1

Методологія IDEF1 поділяє елементи структури інформаційної галузі, їх властивості та взаємозв'язки на класи. Центральним поняттям методології IDEF1 є поняття сутності.

3.2 Моделювання варіантів використання інформаційної системи

Діаграма варіантів використання (англ. use-case diagram) — діаграма, що описує, який функціонал програмної системи, що розробляється, доступний кожній групі користувачів. У модель варіантів використання входить опис акторів (табл. 3.1), варіантів використання (табл. 3.2) програмного продукту та діаграми варіантів використання (рис. 3.3).

Таблиця 3.1 — Опис акторів

№	Назва	Опис
1	Адміністратор	Співробітник, який створює заявки в ІС при зверненні клієнта до СТО, керує ходом виконання замовлень. Має найбільші можливості в системі.
2	Майстер	Співробітник, котрий надає послуги з ремонту та обслуговування авто, обслуговує заявки в ІС.
3	Клієнт	Власник авто, котрий має особистий кабінет з всією інформацією про замовлення.

Таблиця 3.2 — Опис варіантів використання

№	Назва	Опис
1	Авторизація	Можливість входу в систему з певним рівнем доступу.
2	Управління користувачами системи	Адміністратор має можливість створювати користувачів (майстрів та клієнтів) в системі або видаляти.
3	Редагування профілю	Клієнт та майстер мають можливість редагувати власний профіль.
4	Створення замовлень	Адміністратор має можливість створювати замовлення.

Продовження таблиці 3.2 — Опис варіантів використання

5	Управління замовленнями	Адміністратор та майстер можуть управляти замовленнями залежно від наданих прав.
6	Перегляд історії замовлень та інформації клієнта	Майстер, адміністратор та клієнт можуть переглядати історії замовлень та інформації клієнта.
7	Перегляд фінансового звіту та архіву виконаних замовлень	Майстер та адміністратор мають можливість переглядати фінансовий звіт та архів виконаних замовлень.
8	Управління контентом	Адміністратор має можливість управляти інформаційним наповненням системи.

Рисунок 3.3 — Діаграма варіантів використання

ПРАКТИЧНА РЕАЛІЗАЦІЯ

4.1 Реалізації основних компонентів

Для реалізації вебдодатку було використано фреймворк php – Laravel. Він дотримується архітектурного шаблону на основі MVC (моделі, виду та контролера) (рис. 4.1), а також має виразний красивий синтаксис, який робить його об’єктно-орієнтованим. Саме тому обраний спосіб найкраще підходить для вирішення поставленої задачі.

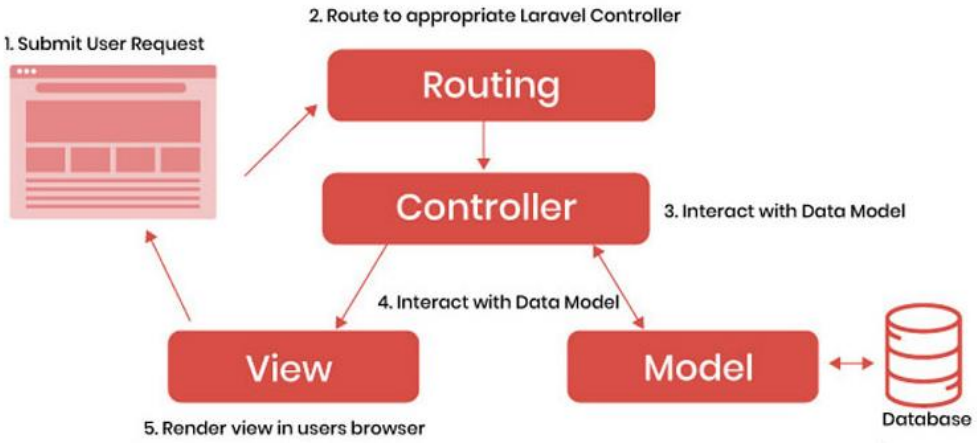

Рисунок 4.1 — Архітектура MVC

Модель в архітектурі MVC обробляє всю логіку даних. Вид відображає інформацію з моделі для користувача, а контролер контролює потік даних в об’єкт моделі та оновлює подання щоразу, коли дані змінюються.

Веб-система містить (БД) базу даних, розроблену в PhpMyAdmin. На рисунку 4.2 представлена створена БД системи.

В даній базі даних міститься наступні 6 таблиць:

– accounts — містить інформацію про зареєстрованих користувачів;
– account_car — зберігає дані автомобілів клієнтів автосервісу;
– orders — зберігає інформацію заявок на ремонт;
– reviews — призначена для зберігання відгуків та листів клієнтів;
– services — містить напрямки послуг, які надає автосервіс;
– sub_services — містить інформацію про послуги автосервісу.

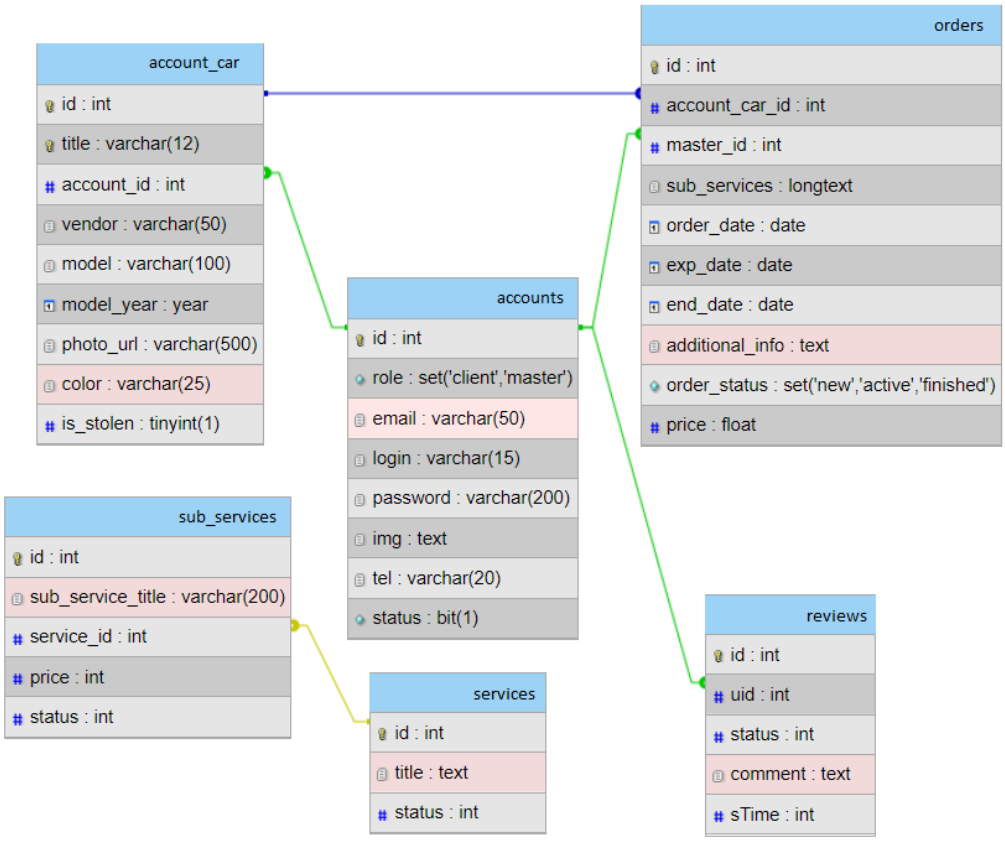

Рисунок 4.2 — База даних в PhpMyAdmin

Клієнтська частина веб-системи була розроблена за допомогою таких інструментів як HTML, CSS, Bootstrap та JavaScript. Використовуючи Bootstrap інтернет-ресурс коректно відображатиметься в сучасних браузерах і на екранах пристроїв різних розмірів, незалежно від діагоналі. Для цього не потрібно вносити зміни до розмітки. Імовірність непередбачених помилок у функціоналі та верстці зведена до мінімуму.

Щоб дістати інформацію про авто за номером було використано API бази даних автомобілів ДАІ України (https://baza-gai.com.ua/api):

```
$opts = [
  "http" => [
    "method" => "GET",
    "header"      =>      "Accept:      application/json\r\n"."X-Api-Key:
fd987d8129f5947b9a9cddd69fdf7b3f\r\n"
  ]
];
$context = stream_context_create($opts);
$data   =   json_decode(file_get_contents("https://baza-gai.com.ua/nomer/
".$model."", false, $context));
```

Змінна `$model` передає або розпізнаний номер авто, або введений вручну у випадку некоректного розпізнавання.

Параметри, за якими здійснювався пошук та передача інформації про знайдене авто у базу даних веб-системи:

– `digits` — номер авто;

– `vendor` — марка авто;

– `model` — модель авто;

– `model_year` — рік випуску автомобіля;

– `photo_url` — фото моделі авто за url адресою;

– `color->slug` — колір автомобіля;

– `is_stolen` — параметр чи знаходиться авто у розшуку.

4.2 Розробка алгоритму детектування

В якості інструменту для навчання моделей детектування та розпізнавання тексту було обрано бібліотеку PaddleOCR. Вона включає в себе провідні та практичні інструменти для OCR, які допомагають швидко та зручно тренувати моделі та застосовувати їх на практиці.

Ця бібліотека містить багато сучасних алгоритмів для детектування тексту, таких як: DDRG [21], FCENet [22], DB[23], EAST, SAST та інші.

Для вирішення поставленої задачі було обрано алгоритм DB. Ключовим етапом розпізнавання тексту є детектування самого тексту, яке спрямоване на локалізацію обмежувальної рамки або області кожного екземпляру тексту. Ця задача все ще є складним завданням, оскільки ділянки з текстом часто мають різні масштаби та форми. Детектування області тексту на основі сегментації останнім часом все більше використовується, оскільки воно може описувати текст різних форм та дає переваги при прогнозуванні на піксельному рівні.

Однак, більшість методів, заснованих на сегментації, вимагають комплексу постобробки даних для групування результатів прогнозування на

рівні пікселів у виявленні ділянки тексту, що призводить до значних втрат часу на процедуру прогнозування. Більшість існуючих методів детектування використовують аналогічний конвейєр постобробки, як показано на рисунку 4.3 синіми стрілками.

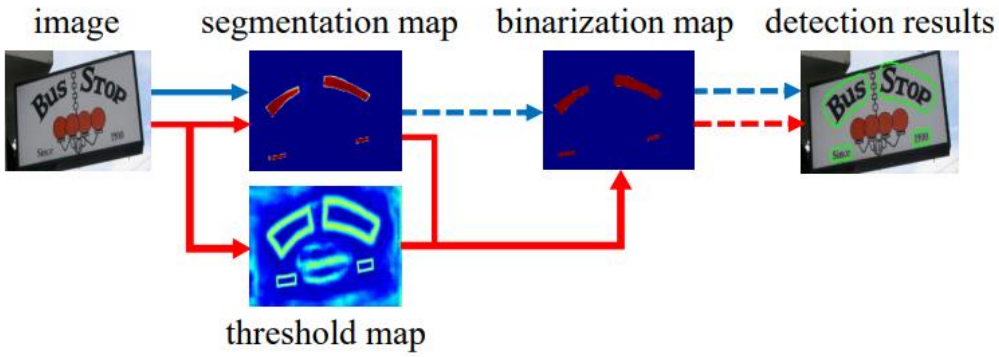

Рисунок 4.3 — Традиційний етап постобробки (сині лінії) та використаний в алгоритмі (червоні лінії)

По-перше, традиційні методи постобробки встановлюють фіксований поріг для перетворення карти ймовірностей, створеної мережею сегментації, у двійкове зображення. Потім для групування пікселів у текстові екземпляри використовуються деякі евристичні методи, такі як кластеризація пікселів. Обраний шлях з червоними лініями має на меті додати операцію бінаризації в мережу сегментації для спільної оптимізації вагових коефіцієнтів.

Таким чином, можна адаптивно передбачити порогове значення в кожній частині зображення, що може повністю відрізнити пікселі від переднього плану та фону. Однак, стандартна функція бінаризації не є диференційованою. Тому в алгоритмі використана наближена функція бінаризації, яка називається диференційованою бінаризацією (DB), яка є повністю диференційованою коли навчається разом із мережею сегментації.

Для тонкого налаштування моделі було використано дані з датасету ICDAR 2015 та власні дані (300 зразків). Приклад розмічення даних наведено на рисунку 4.4.

Рисунок 4.4 — Приклад розмічення даних

Для донавчання моделі детектування використовувався файл конфігурації, наведений в Додатку А. Програмний код для запуску донавчання наведений у Додатку Б.

PaddleOCR містить багато сучасних алгоритмів для розпізнавання тексту, такі як: ABINet[24], CRNN, SRN, SVTR[25] та інші. Було використано алгоритм SVTR. Файл конфігурації для розпізнавання наведено у Додатку В.

Код для запуску детектування та розпізнавання тексту на зображенні, наступний:

```
python3 tools/infer/predict_system.py --
image_dir="./doc/imgs/00018069.jpg" --det_model_dir="./ch_PP-
OCRv3_det_infer/" --rec_model_dir="./ch_PP-OCRv3_rec_infer/" --
use_angle_cls=false
```

При навчанні використовувалась така функція втрат, як Dice Loss.

$$Dice\ Loss = 1 - 2x\frac{Y \cap Y_{pred}}{Y + Y_{pred}} \qquad (4.1)$$

Чисельник містить площу перетину позиції реального боксу і спрогнозованого, а знаменник — суму площ спрогнозованих і реальних ділянок тексту.

Було отримано значення 0,0923, що означає невелике відхилення і вцілому задовільняє потреби нашої задачі детектування ділянок з текстом.

При навчанні моделі розпізнавання тексту використовувалась комбінована функція втрат, яка складається з комбінації наступних функцій втрат та їх параметрів:

```
Loss:
  name: CombinedLoss
  loss_config_list:
  - DistillationDMLLoss:
      weight: 1.0
      act: "softmax"
      use_log: true
      model_name_pairs:
      - ["Student", "Teacher"]
      key: head_out
      multi_head: True
      dis_head: ctc
      name: dml_ctc
  - DistillationDMLLoss:
      weight: 0.5
      act: "softmax"
      use_log: true
      model_name_pairs:
      - ["Student", "Teacher"]
      key: head_out
      multi_head: True
      dis_head: sar
      name: dml_sar
  - DistillationDistanceLoss:
      weight: 1.0
      mode: "l2"
      model_name_pairs:
      - ["Student", "Teacher"]
      key: backbone_out
  - DistillationCTCLoss:
      weight: 1.0
      model_name_list: ["Student", "Teacher"]
      key: head_out
      multi_head: True
  - DistillationSARLoss:
      weight: 1.0
      model_name_list: ["Student", "Teacher"]
      key: head_out
      multi_head: True
```

На комбінованому датасеті отримали значення похибки 0,1205. Такі значення метрик задовільняють потребам роботи системи.

4.3 Використання програмного додатку

4.3.1 Користувач — адміністратор

Пройшовши авторизацію в системі (рис. 4.5) адміністратор потрапляє на головну сторінку (сторінку «Управління замовленнями») (рис. 4.6), де

відображається меню зі всіма доступними сторінками для адміністратора. Тут можливо переглядати історію замовлень, здійснювати пошук за номером авто або логіном клієнта, та редагувати нові замовлення або активні.

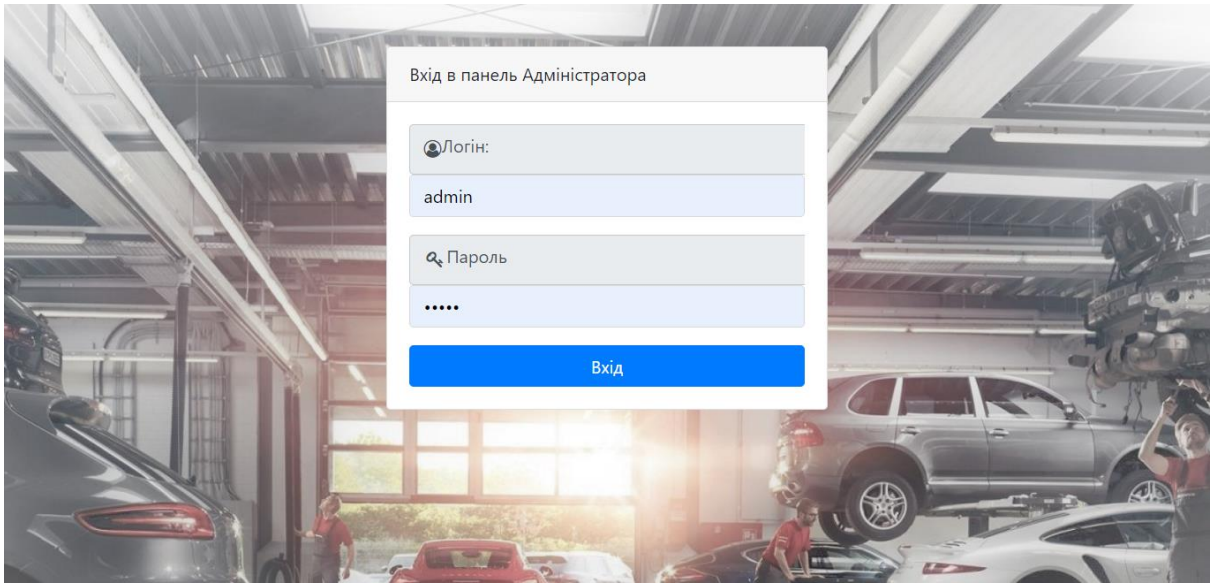

Рисунок 4.5 — Сторінка авторизації адміністратора

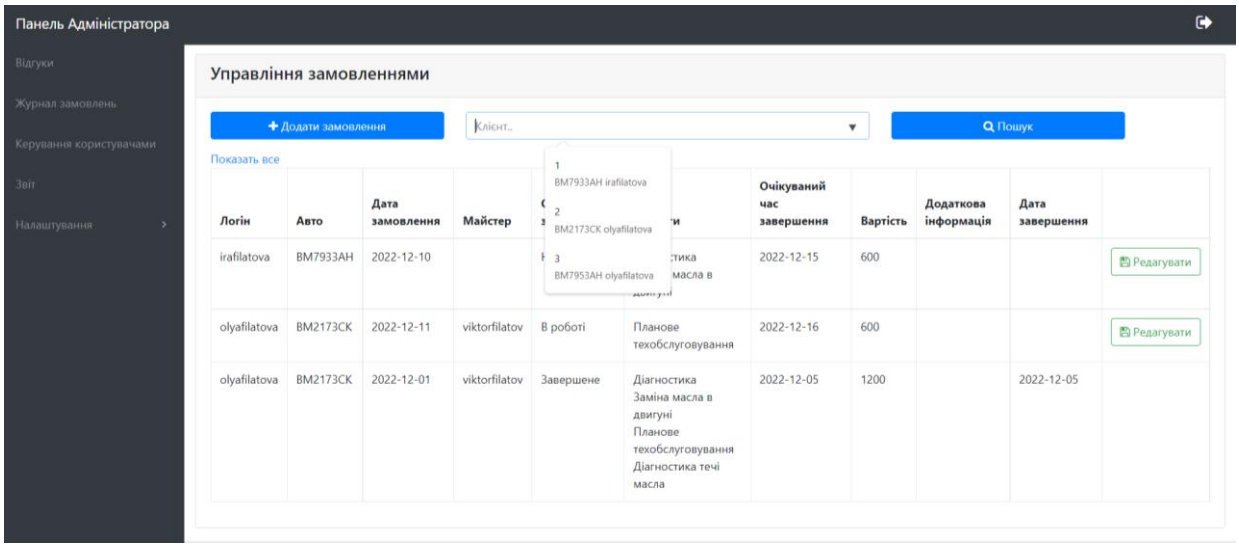

Рисунок 4.6 — Сторінка «Управління замовленнями»

Для створення нового замовлення адміністратору необхідно натиснути кнопку «Додати замовлення». В результаті відкриється модальне вікно з формою (рис. 4.7).

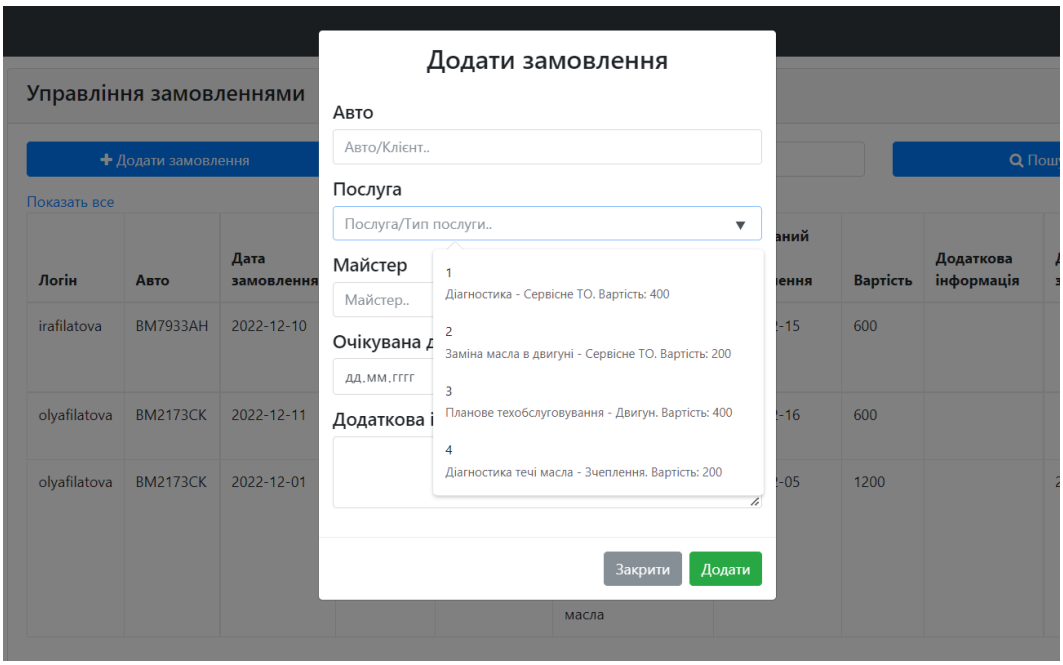

Рисунок 4.7 — Модальне вікно для створення нового замовлення

На сторінці «Управління користувачами» (рис. 4.8) адміністратор може здійснювати пошук користувачів за логіном або роллю (клієнт або майстер), переглядати інформацію про авто клієнтів натиснувши «Детальніше», додавати нове авто, видаляти користувачів з системи та строрювати нових (рис. 4.9-4.10) натиснувши на кнопку «Новий користувач».

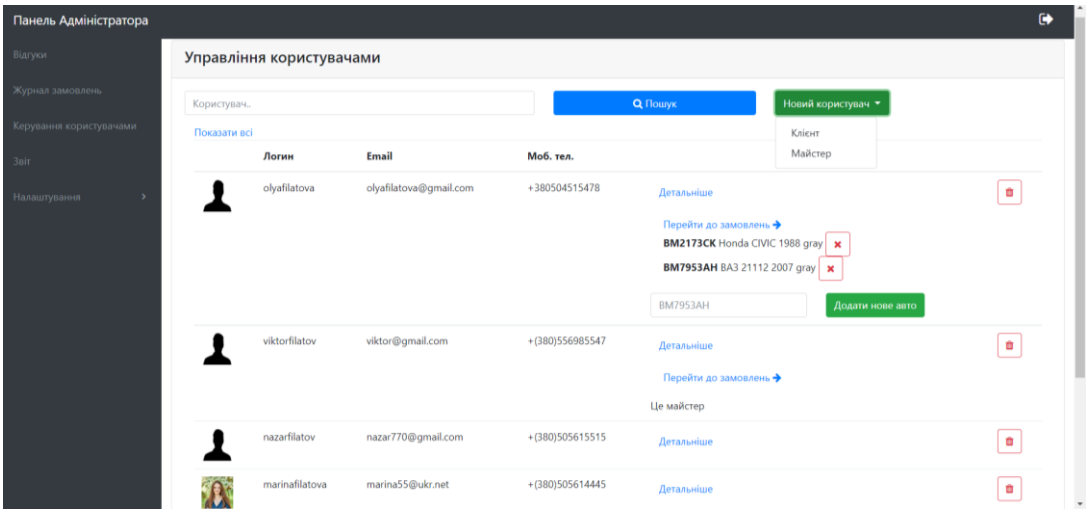

Рисунок 4.8 — Сторінка «Управління користувачами»

Форма для створення нового клієнта містить наступні поля:

– унікальний логін користувача;
– e-mail;

– пароль;
– моб. телефон;
– кнопка для завантаження фото з номером авто;
– поле для номеру авто.

Якщо номер авто розпізнано некоректно або модуль розпізнавання не працює — адміністратор може змінити його вручну. Після реєстрації, новий користувач та авто будуть відображені у списку користувачів сторінки «Управління користувачами».

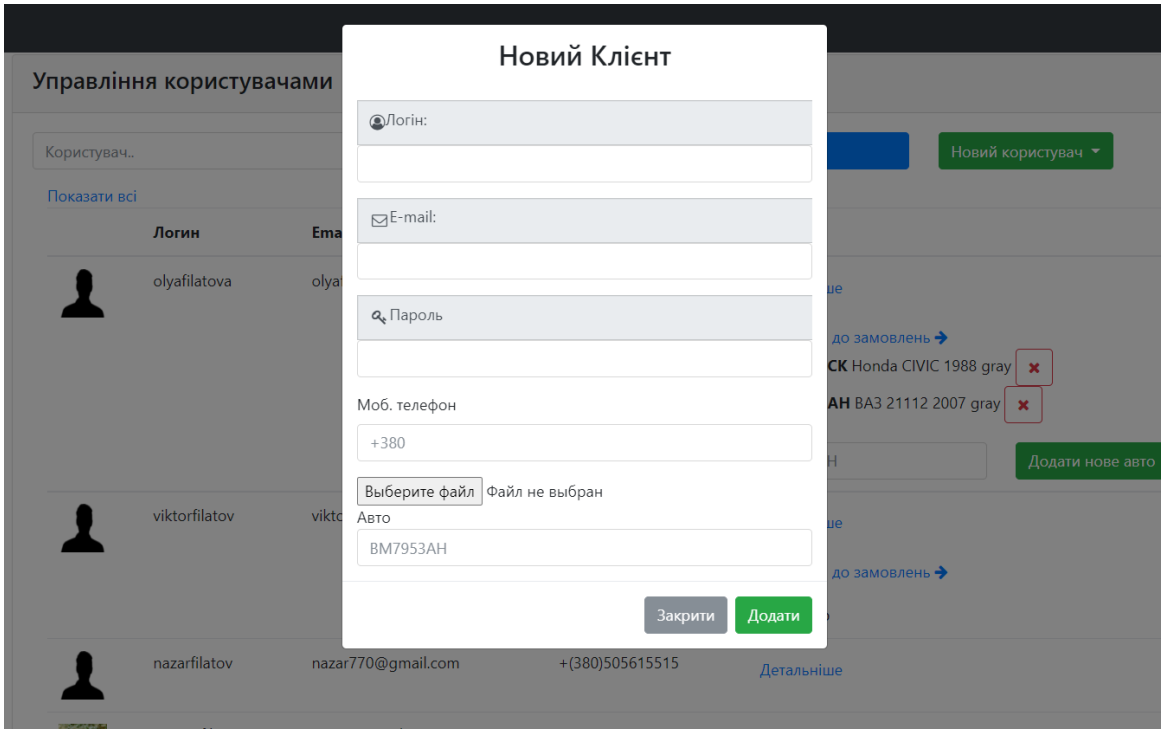

Рисунок 4.9 — Модальне вікно для створення нового клієнта в системі

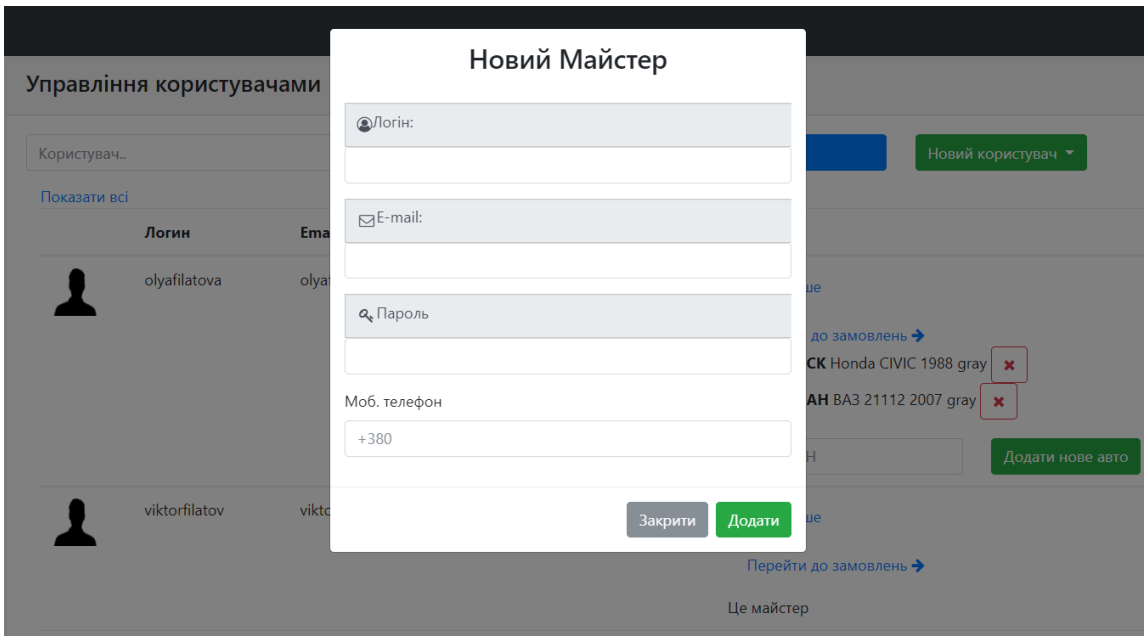

Рисунок 4.10 — Модальне вікно для створення нового майстра в системі

У розділі «Відгуки» (рис. 4.11) адміністратор може видаляти відгуки користувачів.

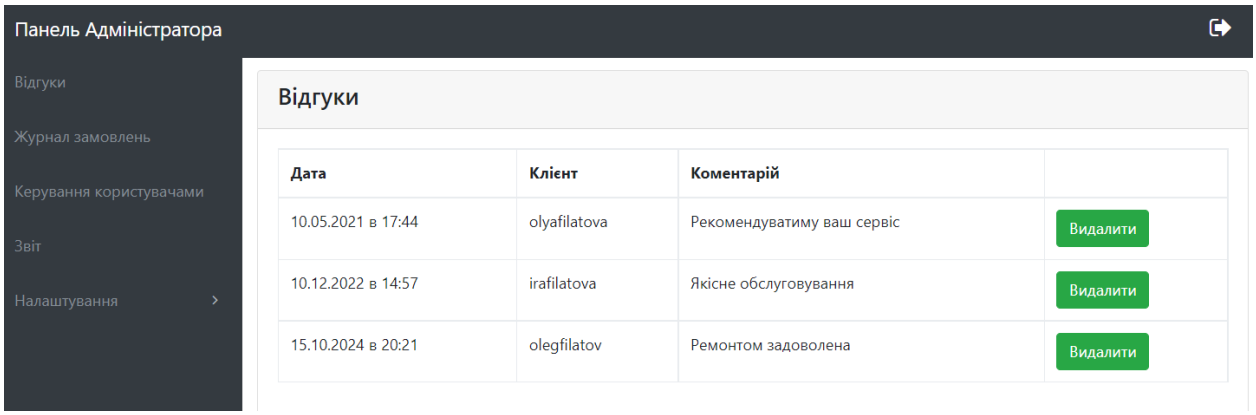

Рисунок 4.11 — Сторінка управління відгуками

Адміністратор може управляти контентом (створювати, редагувати та видаляти послуги та типи послуг) з розділу «Управління послугами» (рис. 4.12-13).

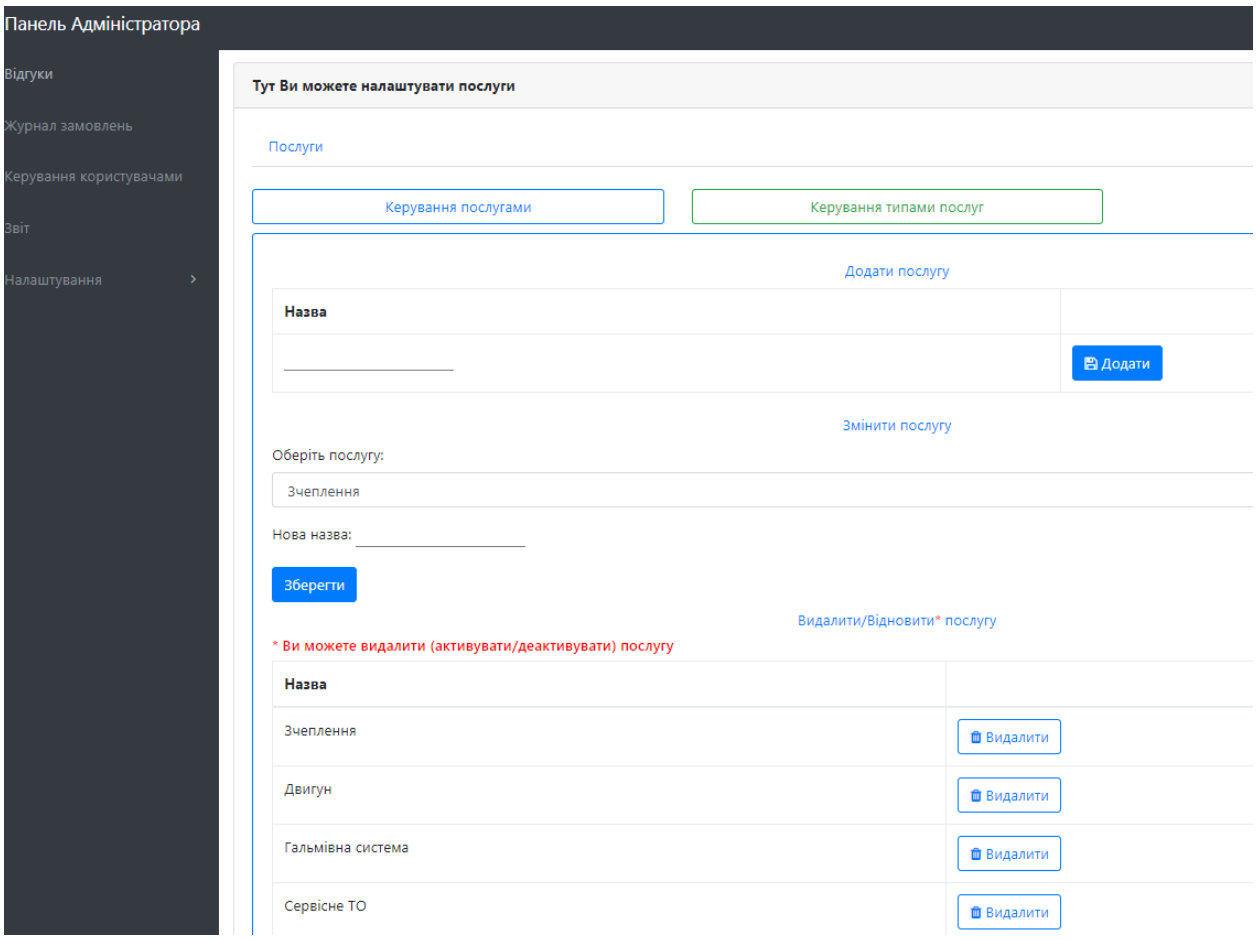

Рисунок 4.12 — Вкладка для додавання та редагування послуг

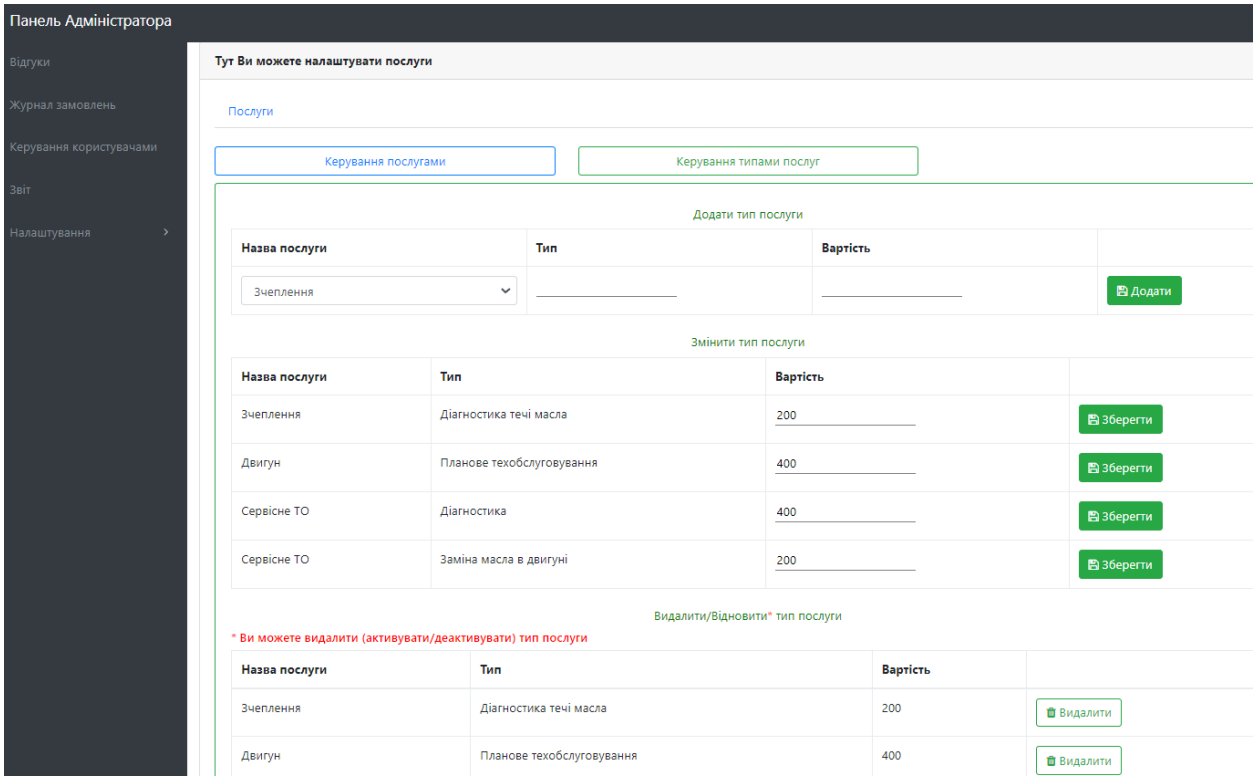

Рисунок 4.13 — Вкладка для додавання та редагування типів послуг

4.3.2 Користувач — механік

Після успішної авторизації (рис. 4.14) майстер автосервісу потрапляє на головну сторінку, де має доступ до наступних розділів: «Журнал замовлень», «Керування клієнтами», «Профіль» та «Звіт».

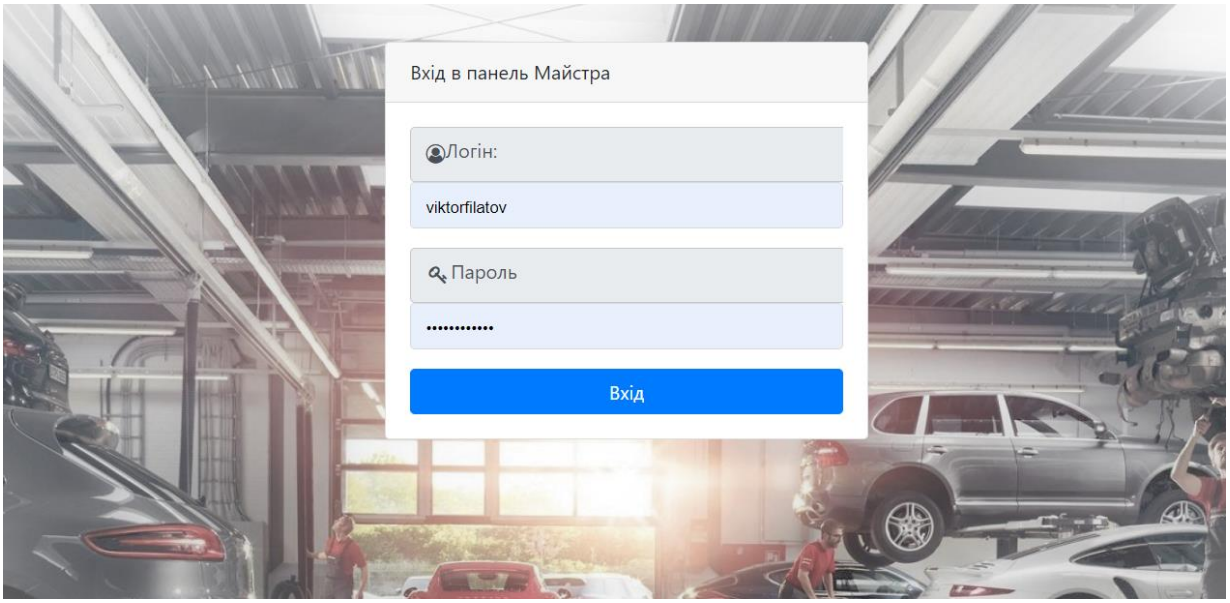

Рисунок 4.14 — Сторінка авторизації користувача типу «майстер»

На сторінці «Журнал замовлень» (рис. 4.15) майстер може переглядати інформацію про замовлення клієнтів та здійснювати пошук клієнтів по логіну та номеру авто, яких він обслуговує; брати нові замовлення; додавати послуги в нові або активні замовлення та редагувати замовлення зі статусом «в роботі».

Пошук та перегляд інформації по власним клієнтам майстер може здійснити зі сторінки «Управління клієнтами» (рис. 4.16).

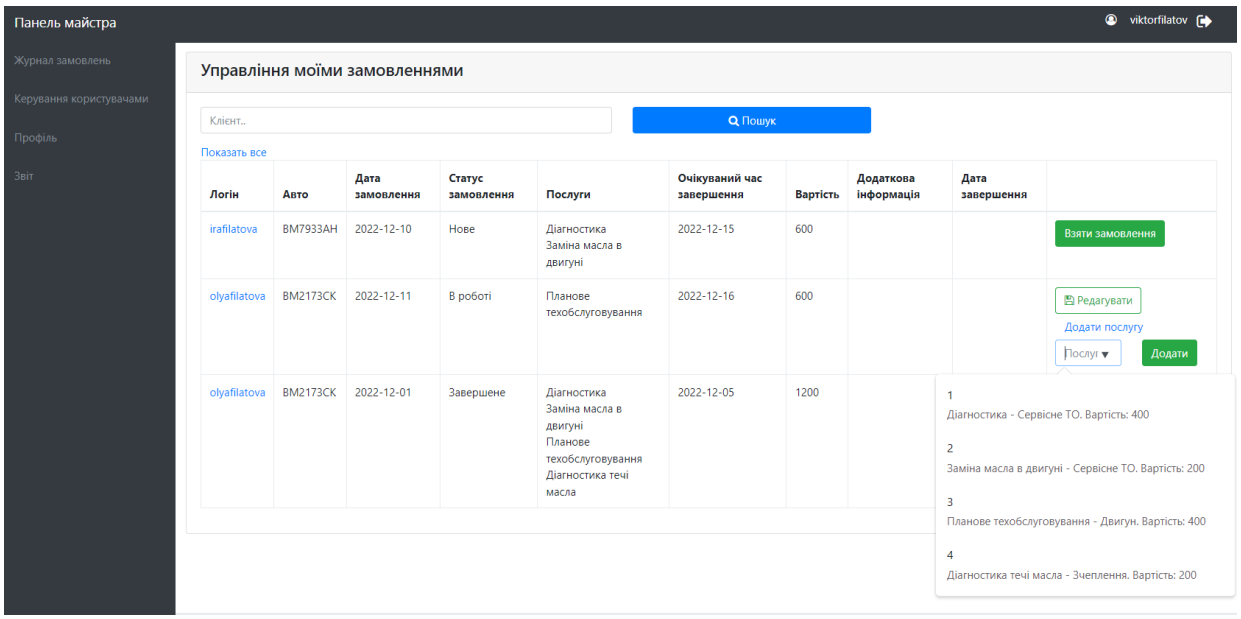

Рисунок 4.15 — Сторінка «Журнал замовлень» користувача типу «майстер»

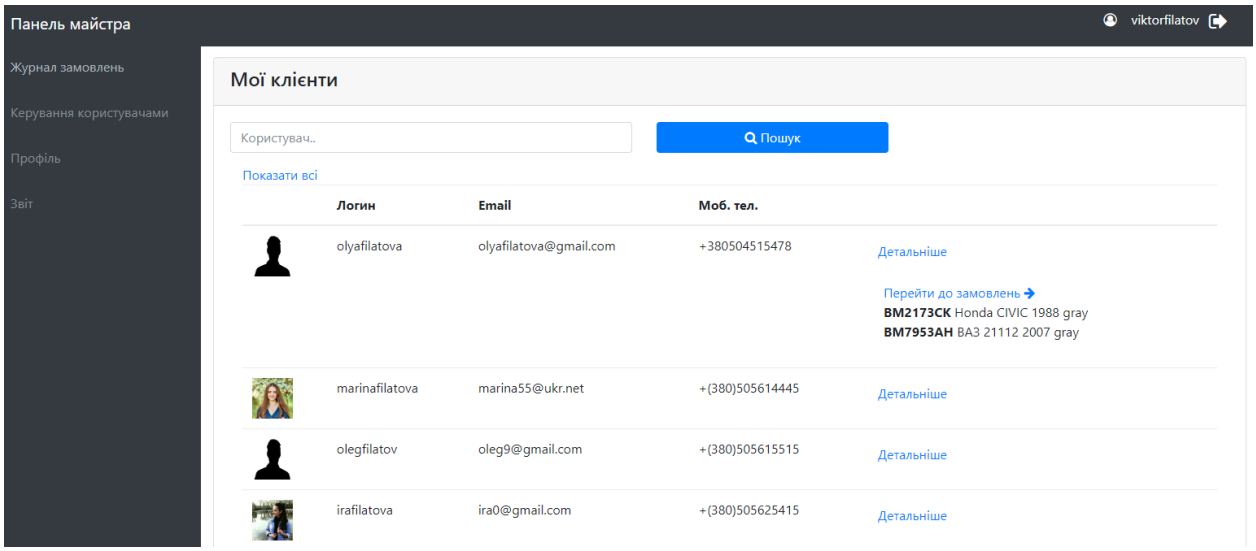

Рисунок 4.16 — Сторінка «Керування клієнтами» користувача типу «майстер»

4.3.3 Користувач — клієнт СТО

Перейшовши за посиланням на сайт, в якості клієнта СТО, потрапляємо на сторінку авторизації (рис. 4.17). Користувач може відновити пароль, натиснувши кнопку «Відновити пароль» та потрапить на сторінку «Відновлення пароля» (рис. 4.18). Новий пароль отримає на пошту, на яку реєструвався.

Після авторизації, користувач потрапляє на сторінку власного кабінету, а саме «Історія замовлень», зміст якої можна бачити на рис. 4.19.

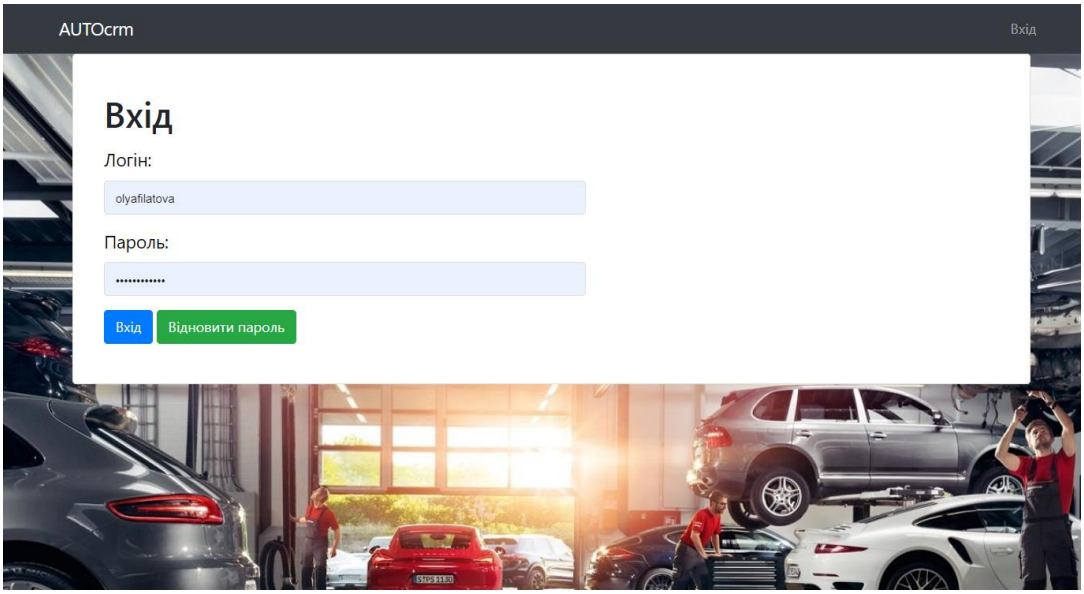

Рисунок 4.17 — Сторінка авторизації користувача типу «клієнт»

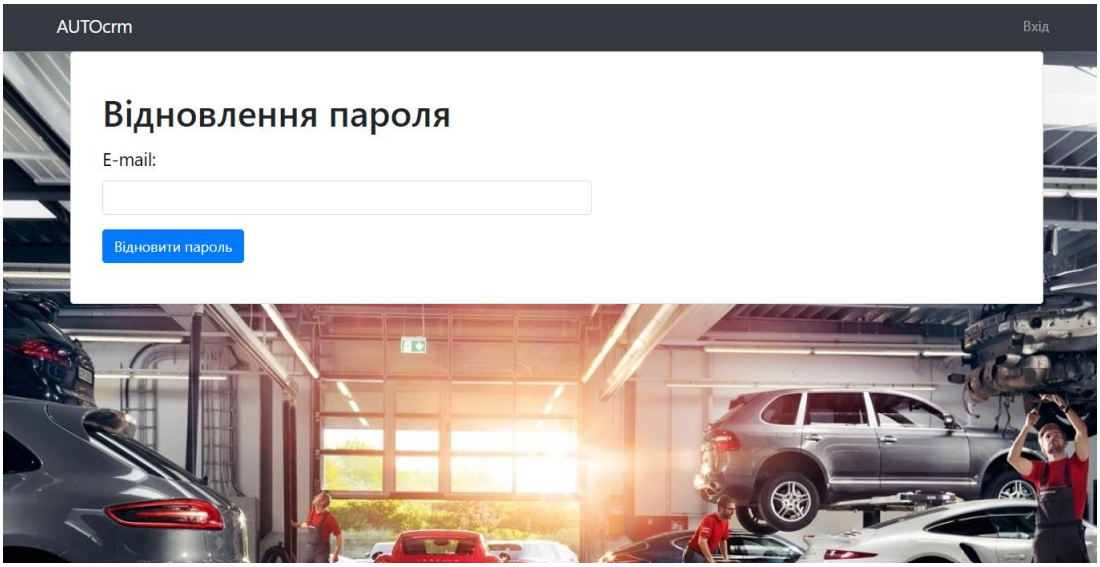

Рисунок 4.18 — Сторінка відновлення пароля

Відображення можливостей користувача типу «клієнт», продемонстровано на рисунках 4.20-21.

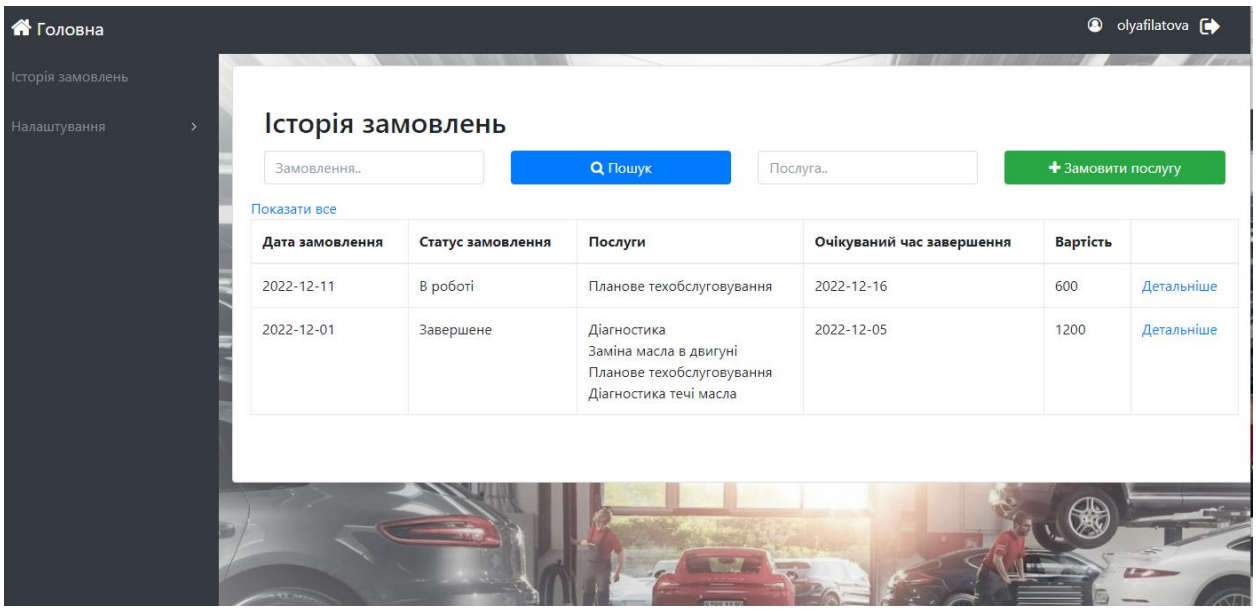

Рисунок 4.19 — Сторінка «Історія замовлень» користувача типу «клієнт»

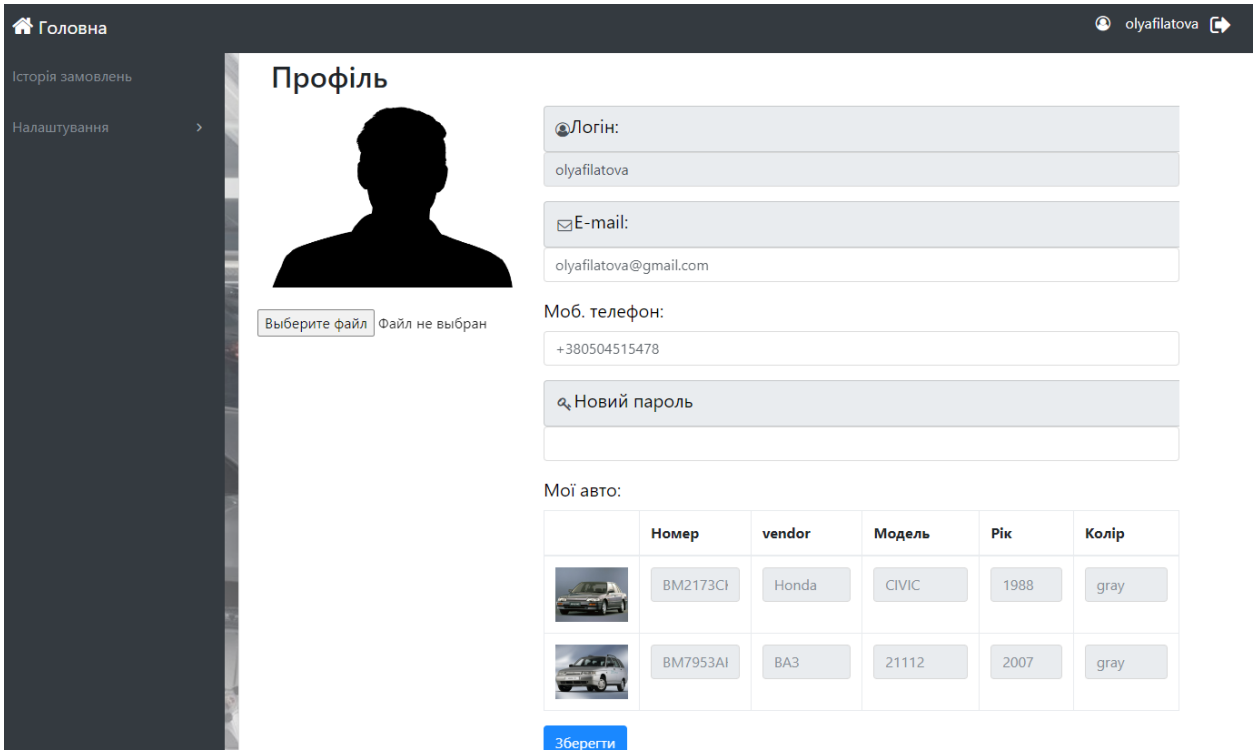

Рисунок 4.20 — Сторінка «Профіль» користувача типу «клієнт»

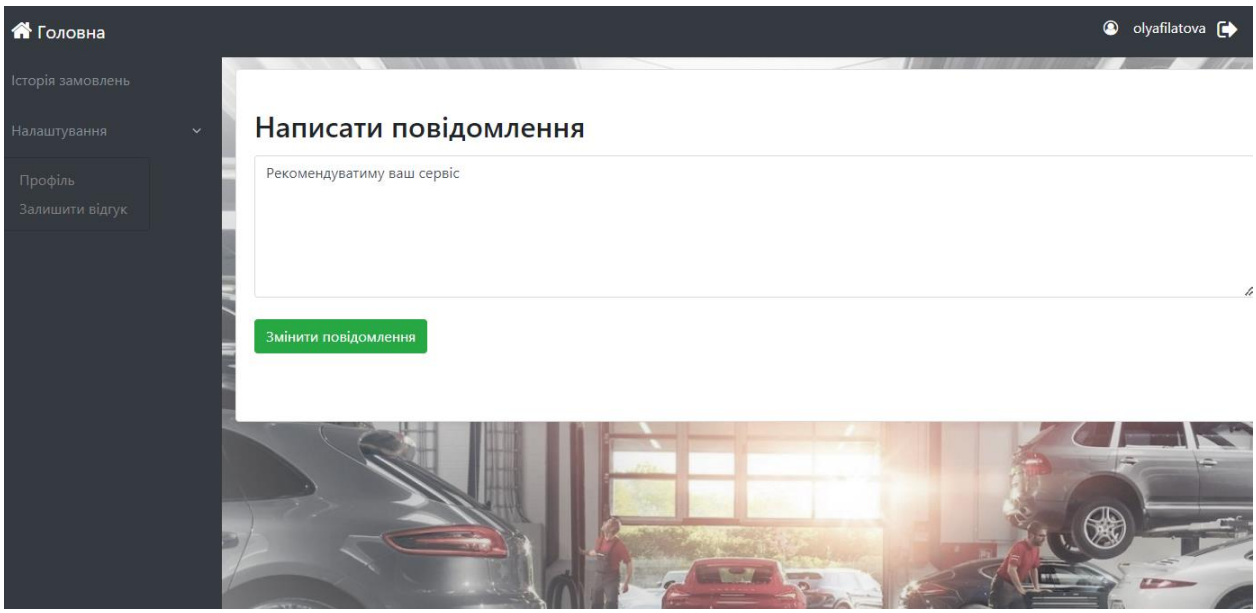

Рисунок 4.21 — Сторінка «Відгук» користувача типу «клієнт»

ВИСНОВКИ

Метою даної роботи була розробка веб-системи для автоматизації бізнес-процесів сервісного обслуговування та ремонту автомобілів для замовників та сервісів з використанням алгоритмів машинного навчання для детектування й розпізнавання номерних знаків автомобілів.

Під час виконання роботи були вирішені наступні задачі:

- визначено об’єкт та предмет дослідження;
- виявлена проблема та актуальність розробки;
- виконано аналіз продуктів-аналогів, у результаті чого були виявлені недоліки та переваги;
- визначені учасники інформаційного обміну у системі та функції, які повинна виконувати;
- створено схему взаємодії користувачів (use-case);
- проведено вибір програмних засобів для реалізації;
- виконано вибір методів вирішення поставленої задачі;

Програмна реалізація системи була виконана за допомогою мов програмування PHP, JavaScript та Python, фреймворку Laravel з використанням моделей машинного навчання та алгоритмів OCR.

Інформаційна система надасть співробітникам оперативну інформацію про хід виконання ремонтних робіт, зайнятість майстрів та грошовий оборот. Така інформація дозволить правильно приймати рішеня у процесі роботи підприємства. ІС дозволить автоматизувати та стандартизувати управління відносин з клієнтами.

В результаті роботи було створено веб-систему, що дозволить не тільки автоматизувати щоденні завдання та робочі процеси, а й дозволить клієнтам бути впевненими в якості обслуговування, прозорості та чесності підприємства.

СПИСОК ЛІТЕРАТУРИ

1. Сервісний центр [Електронний ресурс]. — Режим доступу: https://servicecenters.org.ua/blog/shho-take-avtoryzovanyj-servisnyj-tsentr/ — (дата звернення 01.11.2022) — Назва з титулу екрана.

2. Close the Trust Gap with Transparency [Електронний ресурс]. — Режим доступу: https://dealerportal.truecar.com/dealer-true-insights/closing-the-trust-gap-through-transparency — (дата звернення 01.11.2022) — Назва з титулу екрана.

3. ТИПОВІ БІЗНЕС-ПРОЦЕСИ АВТОСЕРВІСУ [Електронний ресурс]. — Режим доступу: https://netix.com/tipovyie-biznes-protsessyi-avtoservisa-v-netix-triceps.html — (дата звернення 01.11.2022) — Назва з титулу екрана.

4. Modern Auto Repair Software for Your Business [Електронний ресурс]. — Режим доступу: https://autoleap.com/ — (дата звернення 01.11.2022) — Назва з титулу екрана.

5. Auto Repair Software For All Of Your Needs [Електронний ресурс]. — Режим доступу: https://mitchell1.com/ — (дата звернення 01.11.2022) — Назва з титулу екрана.

6. Transforming Customer Journey for Automotive Service Businesses [Електронний ресурс]. — Режим доступу: https://www.garageplug.com/ — (дата звернення 01.11.2022) — Назва з титулу екрана.

7. Introduction to Deep Learning and Neural Networks with PythonTM - 1st Edition [Електронний ресурс]. — Режим доступу: https://www.elsevier.com/books/introduction-to-deep-learning-and-neural-networks-with-python/gad/978-0-323-90933-4 (дата звернення 01.11.2022).

8. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems / Géron Aurélien. — Boston: O'Reilly Media, 2019. — 510 с.

9. Python Machine Learning By Example: The easiest way to get into machine learning / Liu, Yuxi — Boston: Packt Publishing, 2018. — 462 с.

10. Machine Learning for Hackers: Case Studies and Algorithms to Get You Started Paperback – Illustrated / John Myles — Tokyo: O′Reilly, 2017. — 322 с.

11. Machine Learning: A Probabilistic Perspective (Adaptive Computation and Machine Learning series) Hardcover / Kevin Murphy — Boston: The MIT Press, 2018. — 642 с.

12. Python Crash Course A Hands-On Project - Based Introduction to Programming / Eric Matthes— San Francisco: No Starch Press, 2017. — 955 с.

13. The Python Book: The ultimate guide to coding with Python / Alex Hoskins — London: Richmond House, 2016. — 472 с.

14. Nam D., Lee J., Lee H. Business analytics use in CRM: A nomological net from IT competence to CRM performance // Int J Inf Manage. Pergamon, 2019. Vol. 45. P. 233–245.

15. Pro PHP 8 MVC: Model View Controller Architecture-Driven Application Development / Christopher Pitt — Бландфорд: APRESS LP, 2019. — 388 с.

16. ROAD TO REACT: Your journey to master plain yet pragmatic React.js / Robin Wieruch — Morocco: INDEPENDENTLY PUBLISHED, 2019. — 290 с.

17. Learning AngularJS: A Guide to AngularJS Development / Ken Williamson — Boston: O'Reilly Vlg. GmbH & Co., 2016. — 212 с.

18. Fullstack Vue: The Complete Guide to Vue.js / Hassan Djirdeh — Boston: CreateSpace Independent Publishing Platform, 2018. — 442 с.

19. Laravel: Up & Running: A Framework for Building Modern PHP Apps / Matt Stauffer — Boston: O'Reilly Media, 2019. — 554 с.

20. Best Python libraries for Machine Learning – GeeksforGeeks [Електронний ресурс]. — Режим доступу: https://www.geeksforgeeks.org/best-python-libraries-for-machine-learning/ – (дата звернення 01.11.2021) — Назва з титулу екрана.

21. Deep Relational Reasoning Graph Network for Arbitrary Shape Text Detection [Електронний ресурс]. — Режим доступу: https://arxiv.org/abs/2003.07493 – (дата звернення 13.11.2021) — Назва з титулу екрана.

22. Fourier Contour Embedding for Arbitrary-Shaped Text Detection [Електронний ресурс]. — Режим доступу: https://arxiv.org/abs/2104.10442 – (дата звернення 13.11.2021) — Назва з титулу екрана.

23. Real-time Scene Text Detection with Differentiable Binarization [Електронний ресурс]. — Режим доступу: https://arxiv.org/abs/1911.08947 – (дата звернення 13.11.2021) — Назва з титулу екрана.

24. Read Like Humans: Autonomous, Bidirectional and Iterative Language Modeling for Scene Text Recognition [Електронний ресурс]. — Режим доступу: https://openaccess.thecvf.com/content/CVPR2021/papers/Fang_Read_Like_Humans_Autonomous_Bidirectional_and_Iterative_Language_Modeling_for_CVPR_2021_paper.pdf – (дата звернення 13.11.2021) — Назва з титулу екрана.

25. SVTR: Scene Text Recognition with a Single Visual Model [Електронний ресурс]. — Режим доступу: https://arxiv.org/pdf/2205.00159.pdf – (дата звернення 13.11.2021) — Назва з титулу екрана.

ДОДАТКИ

Додаток А

Файл конфігурації для донавчання моделі детектування.

```
Global:
  debug: false
  use_gpu: true
  epoch_num: 500
  log_smooth_window: 20
  print_batch_step: 10
  save_model_dir: ./output/ch_PP-OCR_V3_det/
  save_epoch_step: 100
  eval_batch_step:
  - 0
  - 400
  cal_metric_during_train: false
  pretrained_model: null
  checkpoints: null
  save_inference_dir: null
  use_visualdl: false
  infer_img: doc/imgs_en/img_10.jpg
  save_res_path: ./checkpoints/det_db/predicts_db.txt
  distributed: true

Architecture:
  model_type: det
  algorithm: DB
  Transform:
  Backbone:
    name: MobileNetV3
    scale: 0.5
    model_name: large
    disable_se: True
  Neck:
    name: RSEFPN
    out_channels: 96
    shortcut: True
  Head:
    name: DBHead
    k: 50
Loss:
  name: DBLoss
  balance_loss: true
  main_loss_type: DiceLoss
  alpha: 5
  beta: 10
  ohem_ratio: 3
Optimizer:
  name: Adam
  beta1: 0.9
  beta2: 0.999
  lr:
    name: Cosine
    learning_rate: 0.001
    warmup_epoch: 2
  regularizer:
    name: L2
    factor: 5.0e-05
```

```
      PostProcess:
        name: DBPostProcess
        thresh: 0.3
        box_thresh: 0.6
        max_candidates: 1000
        unclip_ratio: 1.5
      Metric:
        name: DetMetric
        main_indicator: hmean
      Train:
        dataset:
          name: CarsDataSet
          data_dir: ./cars/train_data/icdar2015/text_localization/
          label_file_list:
            -
./cars/train_data/icdar2015/text_localization/train_icdar2015_label.txt
          ratio_list: [1.0]
          transforms:
          - DecodeImage:
              img_mode: BGR
              channel_first: false
          - DetLabelEncode: null
          - IaaAugment:
              augmenter_args:
              - type: Fliplr
                args:
                  p: 0.5
              - type: Affine
                args:
                  rotate:
                  - -10
                  - 10
              - type: Resize
                args:
                  size:
                  - 0.5
                  - 3
- EastRandomCropData:
              size:
              - 960
              - 960
              max_tries: 50
              keep_ratio: true
          - MakeBorderMap:
              shrink_ratio: 0.4
              thresh_min: 0.3
              thresh_max: 0.7
          - MakeShrinkMap:
              shrink_ratio: 0.4
              min_text_size: 8
          - NormalizeImage:
              scale: 1./255.
              mean:
              - 0.485
              - 0.456
              - 0.406
              std:
              - 0.229
              - 0.224
              - 0.225
              order: hwc
```

```
- ToCHWImage: null
          - KeepKeys:
              keep_keys:
              - image
              - threshold_map
              - threshold_mask
              - shrink_map
              - shrink_mask
        loader:
          shuffle: true
          drop_last: false
          batch_size_per_card: 8
          num_workers: 4
      Eval:
        dataset:
          name: SimpleDataSet
          data_dir: ./cars/train_data/icdar2015/text_localization/
          label_file_list:
            -
./cars/train_data/icdar2015/text_localization/test_icdar2015_label.txt
          transforms:
          - DecodeImage:
              img_mode: BGR
              channel_first: false
          - DetLabelEncode: null
          - DetResizeForTest: null
          - NormalizeImage:
              scale: 1./255.
              mean:
              - 0.485
              - 0.456
              - 0.406
              std:
              - 0.229
              - 0.224
              - 0.225
              order: hwc
- ToCHWImage: null
          - KeepKeys:
              keep_keys:
              - image
              - shape
              - polys
              - ignore_tags
        loader:
          shuffle: false
          drop_last: false
          batch_size_per_card: 1
          num_workers: 2
```

Додаток Б

У даному додатку приведено код, який запускає процес донавчання моделі детектування.

```
from __future__ import absolute_import
from __future__ import division
from __future__ import print_function

import os
import sys

__dir__ = os.path.dirname(os.path.abspath(__file__))
sys.path.append(__dir__)
sys.path.insert(0, os.path.abspath(os.path.join(__dir__, '..')))

import yaml
import paddle
import paddle.distributed as dist

from ppocr.data import build_dataloader
from ppocr.modeling.architectures import build_model
from ppocr.losses import build_loss
from ppocr.optimizer import build_optimizer
from ppocr.postprocess import build_post_process
from ppocr.metrics import build_metric
from ppocr.utils.save_load import load_model
from ppocr.utils.utility import set_seed
from ppocr.modeling.architectures import apply_to_static
import tools.program as program

dist.get_world_size()


def main(config, device, logger, vdl_writer):
    # init dist environment
    if config['Global']['distributed']:
        dist.init_parallel_env()
    global_config = config['Global']

    # build dataloader
    train_dataloader = build_dataloader(config, 'Train', device, logger)
    if len(train_dataloader) == 0:
        logger.error(
            "No Images in train dataset, please ensure\n" +
            "\t1. The images num in the train label_file_list should be
larger than or equal with batch size.\n"
            +
            "\t2. The annotation file and path in the configuration file
are provided normally."
        )
        return

    if config['Eval']:
        valid_dataloader = build_dataloader(config, 'Eval', device,
logger)
    else:
        valid_dataloader = None
```

```
        # build post process
           post_process_class = build_post_process(config['PostProcess'],
                                                   global_config)
           # build model
           # for rec algorithm
           if hasattr(post_process_class, 'character'):
               char_num = len(getattr(post_process_class, 'character'))
               if config['Architecture']["algorithm"] in ["Distillation",
                                                          ]:  # distillation
model
                   for key in config['Architecture']["Models"]:
                       if config['Architecture']['Models'][key]['Head'][
                               'name'] == 'MultiHead':  # for multi head
                           if config['PostProcess'][
                                   'name'] == 'DistillationSARLabelDecode':
                               char_num = char_num - 2
                           # update SARLoss params
                           assert list(config['Loss']['loss_config_list'][-
1].keys())[
                               0] == 'DistillationSARLoss'
                           config['Loss']['loss_config_list'][-1][
                               'DistillationSARLoss']['ignore_index'] =
char_num + 1
                           out_channels_list = {}
                           out_channels_list['CTCLabelDecode'] = char_num
                           out_channels_list['SARLabelDecode'] = char_num + 2
                           config['Architecture']['Models'][key]['Head'][
                               'out_channels_list'] = out_channels_list
                       else:
                           config['Architecture']["Models"][key]["Head"][
                               'out_channels'] = char_num
               elif config['Architecture']['Head'][
                       'name'] == 'MultiHead':  # for multi head
                   if config['PostProcess']['name'] == 'SARLabelDecode':
                       char_num = char_num - 2
                   # update SARLoss params
                   assert list(config['Loss']['loss_config_list'][1].keys())[
                       0] == 'SARLoss'
                   if config['Loss']['loss_config_list'][1]['SARLoss'] is None:
                       config['Loss']['loss_config_list'][1]['SARLoss'] = {
                           'ignore_index': char_num + 1
                       }
else:
                      config['Loss']['loss_config_list'][1]['SARLoss'][
                          'ignore_index'] = char_num + 1
                   out_channels_list = {}
                   out_channels_list['CTCLabelDecode'] = char_num
                   out_channels_list['SARLabelDecode'] = char_num + 2
                   config['Architecture']['Head'][
                       'out_channels_list'] = out_channels_list
               else:  # base rec model
                   config['Architecture']["Head"]['out_channels'] = char_num

               if config['PostProcess']['name'] == 'SARLabelDecode':  # for SAR
model
                   config['Loss']['ignore_index'] = char_num - 1

           model = build_model(config['Architecture'])

           use_sync_bn = config["Global"].get("use_sync_bn", False)
           if use_sync_bn:
               model = paddle.nn.SyncBatchNorm.convert_sync_batchnorm(model)
               logger.info('convert_sync_batchnorm')
```

```
            model = apply_to_static(model, config, logger)
            # build loss
            loss_class = build_loss(config['Loss'])

            # build optim
            optimizer, lr_scheduler = build_optimizer(
                config['Optimizer'],
                epochs=config['Global']['epoch_num'],
                step_each_epoch=len(train_dataloader),
                model=model)

            # build metric
            eval_class = build_metric(config['Metric'])

            logger.info('train dataloader has {}
iters'.format(len(train_dataloader)))
            if valid_dataloader is not None:
                logger.info('valid dataloader has {} iters'.format(
                    len(valid_dataloader)))

            use_amp = config["Global"].get("use_amp", False)
            amp_level = config["Global"].get("amp_level", 'O2')
            amp_custom_black_list =
config['Global'].get('amp_custom_black_list', [])
            if use_amp:
                AMP_RELATED_FLAGS_SETTING = {'FLAGS_max_inplace_grad_add': 8, }
                if paddle.is_compiled_with_cuda():
                    AMP_RELATED_FLAGS_SETTING.update({
                        'FLAGS_cudnn_batchnorm_spatial_persistent': 1
                    })
                paddle.fluid.set_flags(AMP_RELATED_FLAGS_SETTING)
                scale_loss = config["Global"].get("scale_loss", 1.0)
                use_dynamic_loss_scaling = config["Global"].get(
                    "use_dynamic_loss_scaling", False)
                scaler = paddle.amp.GradScaler(
                    init_loss_scaling=scale_loss,
                    use_dynamic_loss_scaling=use_dynamic_loss_scaling)
                if amp_level == "O2":
                    model, optimizer = paddle.amp.decorate(
                        models=model,
                        optimizers=optimizer,
                        level=amp_level,
                        master_weight=True)
            else:
                scaler = None

            # load pretrain model
            pre_best_model_dict = load_model(config, model, optimizer,

config['Architecture']["model_type"])

            if config['Global']['distributed']:
                model = paddle.DataParallel(model)
            # start train
            program.train(config, train_dataloader, valid_dataloader, device,
model,
                          loss_class, optimizer, lr_scheduler,
post_process_class,
                          eval_class, pre_best_model_dict, logger, vdl_writer,
scaler,
                          amp_level, amp_custom_black_list)
```

```
    def test_reader(config, device, logger):
        loader = build_dataloader(config, 'Train', device, logger)
        import time
        starttime = time.time()
        count = 0
        try:
            for data in loader():
                count += 1
                if count % 1 == 0:
                    batch_time = time.time() - starttime
                    starttime = time.time()
                    logger.info("reader: {}, {}, {}".format(
                        count, len(data[0]), batch_time))
        except Exception as e:
            logger.info(e)
        logger.info("finish reader: {}, Success!".format(count))


    if __name__ == '__main__':
        config, device, logger, vdl_writer =
program.preprocess(is_train=True)
        seed = config['Global']['seed'] if 'seed' in config['Global'] else
1024
        set_seed(seed)
        main(config, device, logger, vdl_writer)
        # test_reader(config, device, logger)
```

Додаток В

Конфігураційний файл для розпізнавання.

```
Global:
  debug: false
  use_gpu: true
  epoch_num: 800
  log_smooth_window: 20
  print_batch_step: 10
  save_model_dir: ./output/rec_ppocr_v3_distillation
  save_epoch_step: 3
  eval_batch_step: [0, 2000]
  cal_metric_during_train: true
  pretrained_model:
  checkpoints:
  save_inference_dir:
  use_visualdl: false
  infer_img: doc/imgs_words/ch/word_1.jpg
  character_dict_path: ppocr/utils/ppocr_keys_v1.txt
  max_text_length: &max_text_length 25
  infer_mode: false
  use_space_char: true
  distributed: true
  save_res_path: ./output/rec/predicts_ppocrv3_distillation.txt


Optimizer:
  name: Adam
  beta1: 0.9
  beta2: 0.999
  lr:
    name: Piecewise
    decay_epochs : [700, 800]
    values : [0.0005, 0.00005]
    warmup_epoch: 5
  regularizer:
    name: L2
    factor: 3.0e-05


Architecture:
  model_type: &model_type "rec"
  name: DistillationModel
  algorithm: Distillation
  Models:
    Teacher:
      pretrained:
      freeze_params: false
      return_all_feats: true
      model_type: *model_type
      algorithm: SVTR
      Transform:
      Backbone:
        name: MobileNetV1Enhance
        scale: 0.5
        last_conv_stride: [1, 2]
        last_pool_type: avg
      Head:
        name: MultiHead
        head_list:
```

```
          CTCHead:
             Neck:
               name: svtr
               dims: 64
               depth: 2
               hidden_dims: 120
               use_guide: True
             Head:
               fc_decay: 0.00001
           SARHead:
             enc_dim: 512
             max_text_length: *max_text_length
    Student:
      pretrained:
      freeze_params: false
      return_all_feats: true
      model_type: *model_type
      algorithm: SVTR
      Transform:
      Backbone:
        name: MobileNetV1Enhance
        scale: 0.5
        last_conv_stride: [1, 2]
        last_pool_type: avg
      Head:
        name: MultiHead
        head_list:
          - CTCHead:
              Neck:
                name: svtr
                dims: 64
                depth: 2
                hidden_dims: 120
                use_guide: True
              Head:
                fc_decay: 0.00001
          - SARHead:
              enc_dim: 512
              max_text_length: *max_text_length
Loss:
  name: CombinedLoss
  loss_config_list:
  - DistillationDMLLoss:
      weight: 1.0
      act: "softmax"
      use_log: true
      model_name_pairs:
      - ["Student", "Teacher"]
      key: head_out
      multi_head: True
      dis_head: ctc
      name: dml_ctc
  - DistillationDMLLoss:
      weight: 0.5
      act: "softmax"
      use_log: true
      model_name_pairs:
      - ["Student", "Teacher"]
      key: head_out
      multi_head: True
      dis_head: sar
      name: dml_sar
```

```
DistillationDistanceLoss:
      weight: 1.0
      mode: "l2"
      model_name_pairs:
      - ["Student", "Teacher"]
      key: backbone_out
  - DistillationCTCLoss:
      weight: 1.0
      model_name_list: ["Student", "Teacher"]
      key: head_out
      multi_head: True
  - DistillationSARLoss:
      weight: 1.0
      model_name_list: ["Student", "Teacher"]
      key: head_out
      multi_head: True

PostProcess:
  name: DistillationCTCLabelDecode
  model_name: ["Student", "Teacher"]
  key: head_out
  multi_head: True

Metric:
  name: DistillationMetric
  base_metric_name: RecMetric
  main_indicator: acc
  key: "Student"
  ignore_space: False
Train:
  dataset:
    name: CarsDataSet
    data_dir: ./cars/train_data/
    ext_op_transform_idx: 1
    label_file_list:
    - ./train_data/train_list.txt
    transforms:
    - DecodeImage:
        img_mode: BGR
        channel_first: false
    - RecConAug:
        prob: 0.5
        ext_data_num: 2
        image_shape: [48, 320, 3]
        max_text_length: *max_text_length
    - RecAug:
    - MultiLabelEncode:
    - RecResizeImg:
        image_shape: [3, 48, 320]
    - KeepKeys:
        keep_keys:
        - image
        - label_ctc
        - label_sar
        - length
        - valid_ratio
  loader:
    shuffle: true
    batch_size_per_card: 128
    drop_last: true
    num_workers: 4
Eval:
```

```
dataset:
    name: SimpleDataSet
    data_dir: ./train_data
    label_file_list:
    - ./train_data/val_list.txt
    transforms:
    - DecodeImage:
        img_mode: BGR
        channel_first: false
    - MultiLabelEncode:
    - RecResizeImg:
        image_shape: [3, 48, 320]
    - KeepKeys:
        keep_keys:
        - image
        - label_ctc
        - label_sar
        - length
        - valid_ratio
  loader:
    shuffle: false
    drop_last: false
    batch_size_per_card: 128
    num_workers: 4
```