

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування інституту, факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавра

(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ БРОНЮВАННЯ ПОСЛУГ З МИЙКИ
АВТОМОБІЛІВ
CAR WASH BOOKING SOFTWARE

Виконав: студент(ка) 4 курсу, групи КНТ-219сп
Спеціальності 122 Комп'ютерні науки

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Комп'ютерні науки

Кучеренко О.О.

(прізвище та ініціали)

Керівник Скрупський С.Ю.

(прізвище та ініціали)

Рецензент Голуб Т.В.

(прізвище та ініціали)

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

Інститут, факультет ФКНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 122 Комп'ютерні науки
(код і найменування)
Освітня програма (спеціалізація) Комп'ютерні науки
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
“ ” 2022 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Кучеренка Олексія Олександровича
(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення бронювання послуг з мийки автомобілів
Car Wash Booking Software

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

затверджені наказом закладу вищої освіти від «27» квітня 2022 року № 102

2. Строк подання студентом проєкту (роботи) червень 2022 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Розробка архітектури програми. 3. Розробка програми. 4. Експлуатація та тестування програми.

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

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	Скрупський С.Ю., доцент		
Нормоконтролер	Камінська Ж.К., асистент		

7. Дата видачі завдання « 4 » квітня 2022 року.

КАЛЕНДАРНИЙ ПЛАН

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

Студент(ка)

(підпис)

Кучеренко О.О.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис)

Скрупський С.Ю.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 84 с., 23 рис., 1 табл., 3 дод., 10 джерел.

АВТОМОБІЛЬ, МИТТЯ АВТОМОБІЛІВ, ДЕТЕЙЛІНГ, БРОНЮВАННЯ, ВЕБСИСТЕМА.

Об'єкт роботи – процес миття автомобілів. Предмет роботи – програмне забезпечення бронювання послуг з мийки автомобілів. Мета роботи – розробити програмне забезпечення бронювання послуг з мийки автомобілів для забезпечення автоматизації даного процесу.

Матеріали, методи та технічні засоби: мова Python, вебфреймворк Django, система керування базами даних MySQL.

Результати. Функціональні вимоги до програми було визначені, після чого їх використано при створенні сценаріїв використання програми клієнтом і працівником автомийки. Проектування бази даних забезпечено з визначенням повної структури бази даних. За допомогою фреймворку Django визначено структуру вебсистеми. Виконано реалізацію вебсистеми.

Висновки. Розроблена вебсистема дозволяє відслідковувати і організовувати процес бронювання послуг з мийки автомобілів менеджерами автомийок, створювати єдині правила на різних автомийках мережі, аналізувати завантаженість автомийки протягом дня або місяця, а також дозволяє віддалено бронювати мийку і додаткові процедури детейлінгу клієнтами. Програма може використовувати для підтримки роботи невеликої мережі мийок автомобілів або окремої мийки автомобілів, що складається з декількох боксів.

Галузь використання. Автоматизовані мийки автомобілів, підтримка бронювання обслуговування.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 84 pages, 23 figures, 1 table, 3 appendixes, 10 sources.

CAR, CAR WASH, DETAILING, BOOKING, WEB SYSTEM.

The object of the work is a process of washing cars. Subject of the work is a software for booking car wash services. The purpose of the work is to develop software for booking car wash services to ensure the automation of this process.

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

Results. The functional requirements were determined for the software and then used to create scenarios for the use of the software by a customer and a car wash employee. Database design is provided with the definition of the complete structure of the database. The structure of the web system was determined using the Django framework. The web system has been implemented.

Conclusions. The developed web system provides users with the opportunities to monitor and organize the process of booking car wash services, defining it for car wash managers, create uniform rules for different car washes in a network, analyze car wash occupancy during a day or a month, and to remotely book car wash and additional detailing procedures. The software can be used to support a small network of car washes or a separate car wash consisting of several boxes.

Field of use. Automated car washes to service booking support.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	7
Вступ.....	8
1 Аналіз предметної області.....	9
1.1 Організація мийки автомобілів.....	9
1.2 Огляд програмних аналогів.....	11
1.3 Основні сутності предметної області.....	13
1.4 Висновки за розділом 1 та завдання роботи.....	15
2 Розробка архітектури програми.....	16
2.1 Визначення функціональних вимог до програми.....	16
2.2 Вибір мови програмування	18
2.3 Проектування бази даних	19
2.4 Структура вебсистеми	28
2.5 Висновки за розділом 2	29
3 Розробка програми	30
3.1 Алгоритм функціонування програми.....	30
3.2 Опис реалізованих класів програми.....	31
3.3 Висновки за розділом 3	39
4 Експлуатація та тестування програми	40
4.1 Призначення програми	40
4.2 Умови виконання програми	40
4.3 Експлуатація програми.....	40
4.4 Тестування програми	45
4.5 Висновки за розділом 4	45
Висновки	46
Перелік джерел посилання	47
Додаток А Технічне завдання	49
Додаток Б Текст програми	53
Додаток В Слайди презентації.....	78

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

SUV – sport utility vehicle;

БД – база даних.

ВСТУП

Споживачі віддають перевагу послугам професійної мийки автомобілів перед домашнім миттям, ключовою тенденцією серед клієнтів із напруженим способом життя та напруженим графіком є перехід від «зроби сам» до «зроби це для мене»: за даними Міжнародної асоціації автомийки кількість водіїв, які регулярно миють свої автомобілі в професійних сервісних центрах, зростає з приблизно 48,0 % у 1994 році до понад 77 % у 2019 році, крім того, близько 66 відсотків американців миють свої автомобілі не менше ніж 1-2 рази на місяць, що призводить до в середньому 13 мийок на рік, а концепція «все нове, весь час», схоже, зумовила популярність мийок, оскільки автоматичне миття автомобілів стало нормою для власників транспортних засобів протягом останніх кількох десятиліть, в індустрії автотехніки нові технології та обладнання покращили процес, завоювали частку ринку та підвищили очікування клієнтів [1].

До того ж потрібно додати, що безпосередньо з мийкою автомобілів пов'язана і взагалі зараз з'єднана з нею сфера детейлінгу – у такому випадку після миття автомобіля обслуговуючий персонал повністю очищає та полірує автомобіль, зазвичай наносячи віск і використовуючи інструмент, щоб видалити віск та полірувати автомобіль, при цьому часто видаляється тьмяна фарба та дрібні подряпини, відпарюються килими та сидіння, освітлюється хром, видаляється смола та виконується ряд інших послуг [2]. Автоматична мийка і детейлінг автомобіля є тенденцією розвитку автомобільної галузі загалом.

Тому зважаючи на актуальність сфери мийки автомобілів, метою роботи було визначено розробити програмне забезпечення бронювання послуг з мийки автомобілів для забезпечення автоматизації даного процесу.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Організація мийки автомобілів

Згідно зі звітом компанії Grand View Research попит на послуги мийки автомобілів зростає протягом останнього десятиліття, при цьому п'ять ключових факторів керують цими тенденціями зростання в Північній Америці та за її межами, зокрема:

- збільшення транспортних засобів на домогосподарство: сучасне домогосподарство нерідко має три-чотири транспортні засоби, окрім того з розвитком нових варіантів житла з'являються нові можливості для нових автомийок, навіть у тих районах, де автомийки вже існують;

- клієнт вимагає більшої зручності: напружений графік і обмеженість у часі спонукали все більше споживачів використовувати професійні автомийки, а не чистити свої транспортні засоби вдома;

- екологічні норми: все більше громад прийняли правила, які забороняють миття автомобілів у житлових приміщеннях, це допомагає розширити клієнтську базу для професійних автомийних підприємств;

- можливості для прибуткового бізнесу: підприємства автомийки підходять для широкого кола власників, від пенсіонера, який хоче інвестувати пенсію, до групи прямих інвестицій з кількома автомийками в портфелі, надаючи в результаті значну віддачу від інвестицій з довгостроковим потоком доходу, до того ж цей достатньо простий по суті бізнес має високу вартість при перепродажі;

- нові технології: у минулі роки нерідко траплялися серйозні піки та спади в потоці доходів автомийок, особливо взимку, сучасні технології можуть забезпечити більшу стабільність потоку доходів, наприклад, модель підписки дозволяє клієнтам вибрати план мийки, сплатити щомісячну плату та отримати необмежену кількість мийок, в результаті ця модель стала популярнішою за останні 10 років [3].

Миття автомобілів і детейлінг автомобілів переживають зараз і будуть переживати за прогнозами бум у найближчі роки, при чому в цих тенденціях можна виділити наступне:

- значний ріст у галузі мийки автомобілів: з 2012 року багато автовласників активно використовують професійну автомийку, міжнародна асоціація автомийки вказує, що менше 28,4 % споживачів вважають за краще мити свої автомобілі вдома в порівнянні з 48 % у 1996 році;

- промисловість мийки автомобілів коштує понад 33 мільярди доларів: у 2018 році загальний капітал індустрії мийки автомобілів оцінювався в 33 мільярди доларів, очікується, що кількість зросте на 3,2 % у період з 2019 по 2025 рік, зростання в основному пов'язане зі збільшенням кількості власників транспортних засобів, щільним графіком роботи та збільшенням споживчої спроможності;

- найбільший ринок мийки автомобілів у США, де понад 60 тисяч автомийок, які щорічно приносять на ринок понад 5,8 мільярдів доларів;

- найбільш переважним є рішення для автоматичної мийки автомобіля: за статистикою 51 % автокористувачів віддають перевагу автоматичній мийці в салоні, оскільки вона передбачає швидке миття та сушіння, на другому місці стоїть самообслуговування, якому віддають перевагу приблизно 32 % автовласників;

- багато американців миють свій автомобіль приблизно 13 разів на рік;

- до 2025 року сфера мийки автомобілів досягне 3,18 мільярдів доларів, що очікується через збільшення купівельної спроможності людей;

- індустрія мийок автомобілів є однією з небагатьох, де основні гравці не домінують: дослідження, проведене IBIS World, показують, що в галузі автомийки немає великих гравців з часткою ринку, що перевищує 5 %;

- підприємство автомийок пропонує переважно шість видів мийок: кілька поширених типів автомобільних мийок включають самообслуговування, послуги детейлінгу, автоматичне миття в приміщенні,

ручне миття, очищення лише зовні та повне обслуговування (конвеєрна автомийка);

– лідером за темпами зростання на даний момент є Азіатсько-Тихоокеанська зона: публікація Market Watch визначила Китай, Таїланд, Індію та інші регіони Тихоокеанського регіону як найбільш процвітаючі з точки зору діяльності мийок автомобілів [4]-[5].

1.2 Огляд програмних аналогів

Аналоги програми бронювання послуг з мийки автомобілів належать величезному існуючому масиву. Це пов'язано зокрема з тим, що, як було описано в підрозділі 1.1, на даному ринку немає великих гравців з домінуючою роллю. Тобто кількість гравців насправді величезна і багато з них мають власні вебсайти. Слід відзначити, що за результатами аналізу програм автомийок в Україні, вони зазвичай несуть інформаційну функцію і не дозволяють виконувати бронювання, як наприклад, вебсайт автомийки «Престиж» (рис. 1.1) [6].

Автомийка "Престиж"				
Головна Послуги Про нас Контакти Вхід				
Послуга	Легкові авто	Кроссовер, мінівен	Автомобілі бізнес класу, рамні позашляховики 4x4	Буси, мікроавтобуси
Змивка	120	130	140	150
Експрес-мийка (поверхнева)	150	170	190	200
Мийка кузова контактна/безконтактна	230/260	250/280	280/320	300/340
Повна мийка контактна/безконтактна	330/360	350/380	380/420	420/470
Екстра мийка контактна	660	700	760	840
Чистка салону порошотягом	від 90	від 100	від 120	від 140
Екстра чистка салону порошотягом	від 180	від 200	від 240	вартість узгоджується на місці
Чистка скла в салоні Авто	від 50	від 50	від 60	50-100
Протирання пластмасових виробів в салоні	від 70	від 70	від 80	70-120
СУХЕ прибирання в салоні (ціна від)	150	170	200	150-250
Натирання ущільнювачів силіконом	100	100	100	100
Мийка двигуна За проблеми запуску двигуна після його миття відповідальності не несемо	330	330	330	330

Рисунок 1.1 – Автомийка «Престиж»

У той же час за кордоном такі сервіси мають достатньо розширені функції. Для прикладу було обрано вебсайт мережі автомийок Star Car Wash (рис. 1.2) в Австралії [7].

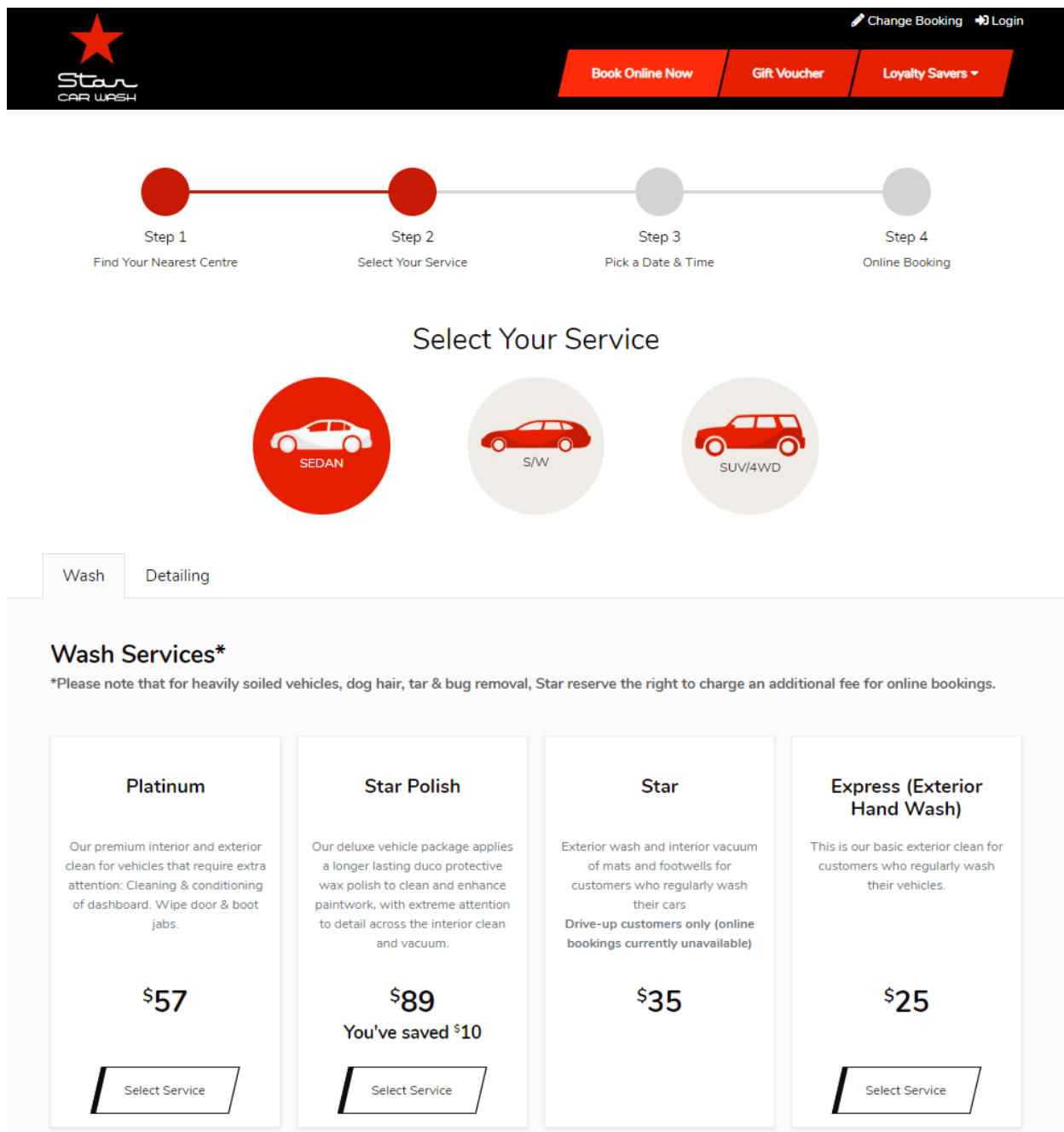


Рисунок 1.2 – Star Car Wash

Вебсайт Star Car Wash надає можливість виконання бронювання мийки автомобіля, під час даного процесу визначаючи конкретну автомийку з мережі, задаючи клас автомобіля, обираючи дату і час мийки з наданого

набору слотів, визначаючи дані клієнта. Окрім того на вебсайті можна отримати інформацію про програму лояльності в мережі, про мережу загалом, автомийки з мережі, вартість пакетів обслуговування, наявні пропозиції для клієнта.

1.3 Основні сутності предметної області

Під час вивчення предметної області було виділено основні її сутності, які в подальшому повинні стати основою побудови бази даних (БД), а також визначення побудови програми.

Мийка автомобілів – це заклад, де безпосередньо реалізуються послуги з мийки та детейлінгу автомобілів. Передбачається, що таких мийок може бути одразу декілька на всю систему. Тобто фактично може організовуватися певна мережа мийок, які розташовуються в одному місті або певній агломерації міста.

Бокс автомобільної мийки – це місце на мийці автомобілів, де безпосередньо один конкретний автомобіль може обслуговуватися, тобто фактично це складова мийки автомобілів.

Бронювання мийки автомобіля – це резервування надання послуг з мийки автомобіля та забезпечення процедур детейлінгу, що може реалізовуватися на конкретній мийці автомобілів у конкретний момент. Бронювати вирішено мийку не під конкретний автомобіль, а під клас базового розміру автомобіля, тобто замість одного легкого автомобіля можна вимити інший легковий автомобіль, якщо це необхідно, а бронювання вже було виконано.

Клас базового розміру автомобіля відповідно є загальною характеристикою автомобіля, за якою він позиціонується в системі бронювання. Іншої інформації про автомобіль фактично не надається при бронюванні, але вже під час мийки, коли фактичний автомобіль приїхав на мийку, його номер може бути зафіксований.

Пакет послуг з мийки автомобіля визначає, які саме процедури мийки автомобіля будуть здійснюватися. Загальна процедура може реалізовуватися на мийці і увесь час за допомогою, наприклад, трьох фаз, але різні пакети можуть визначати, наприклад, додатково мийку двигуна. Тобто це набір процедур, які стосуються тільки мийки, і які об'єднані в певний комплекс.

Вартість пакету послуг з мийки автомобіля є окремою сутністю, адже пакет послуг сам по собі може не визначати вартість, оскільки вартість цих процедур може бути різною для різних класів автомобілів. Наприклад, мийка легкого автомобіля може коштувати дешевше ніж мийка вантажівки або автобуса.

Процедура детейлінгу визначає конкретні дії, що стосуються підтримки автомобіля у найкращому косметичному стані.

Особливості процедури детейлінгу виділено окремо, оскільки необхідно визначити, яка тривалість і вартість конкретної процедури детейлінгу для різного класу автомобілів.

Стандартний розклад роботи мийки визначає робочий час і час перерви на конкретній мийці автомобілів протягом робочого дня (дня тижня).

Оскільки не завжди розклад роботи може бути стандартним, то ще однією сутністю є виключення з розкладу роботи мийки автомобілів, що стосується годин роботи в святкові, передсвяткові або загалом дні, наприклад, скорочені дні.

Фахівець мийки – це конкретна особа, яка виконує процедури мийки або детейлінгу автомобілів на мийці.

Тип чистячого засобу використовується для групування всіх чистячих засобів, які використовуються під час обслуговування автомобілів.

Чистячий засіб – це загалом будь-який нетехнічний засіб, який використовується під час обслуговування автомобіля на мийці.

Тип обладнання для мийки автомобілів.

Обладнання для мийки автомобілів

Виробник – компанія, яка виробляє обладнання або чистячі засоби, які використовуються в роботі мийки.

Клієнт – це користувач, який виконує бронювання мийки автомобіля в системі і в подальшому використовує це бронювання для мийки автомобіля.

Знижка необхідна для надання дисконтної накопичувальної системи клієнтам, щоб вони були зацікавлені в тривалому використанні послуг мийки.

Фах спеціаліста – це фах, на якому спеціалізується працівник мийки автомобілів і за яким визначається, в яких саме роботах на мийці він може брати участь.

1.4 Висновки за розділом 1 та завдання роботи

Завдання роботи:

- дослідження проблеми бронювання послуг з мийки автомобілів;
- визначення засобів розробки;
- визначення вимог до програми;
- проєктування програми бронювання послуг з мийки автомобілів;
- програмна реалізація.

За результатами дослідження проблеми бронювання послуг з мийки автомобілів було виявлено актуальність проблеми, тенденції її розвитку, проаналізовано аналоги, що стосуються як України, так і світу, виділено їх базові можливості.

2 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМИ

2.1 Визначення функціональних вимог до програми

Для реалізації бронювання послуг з мийки автомобілів користувачі загалом повинні отримувати можливість роботи з наступними функціями в програмі:

- перегляд власних бронювань;
- бронювання мийки автомобіля;
- вибір пакету послуг з мийки автомобіля та необхідних процедур детейлінгу;
- змінювання бронювання;
- перегляд бронювань за клієнтом;
- перегляд бронювань за мийкою на дату;
- перегляд годин використання автомийки за день або за місяць;
- визначення дисконтної системи для клієнтів;
- внесення і змінювання даних автомийки;
- внесення і змінювання даних пакетів послуг з мийки автомобіля;
- внесення і змінювання даних процедур детейлінгу;
- внесення і змінювання стандартного розкладу роботи автомийки;
- внесення і змінювання розкладу роботи в святкові дні;
- фіксація роботи працівника на автомийці на день;
- внесення і змінювання даних чистячих засобів і обладнання;
- перегляд автомийок за містом;
- перегляд пакетів послуг;
- перегляд асоційованих автомийок, тобто автомийок, де працює працівник;
- авторизація клієнтів і працівників та реєстрація клієнтів.

Частина з цих функцій має надаватися клієнту, а частина – працівникам автомийки.

Сценаріями для клієнта є (рис. 2.1):

- перегляд автомийок;
- перегляд власних бронювань;
- бронювання мийки автомобіля;
- перегляд власного профілю;
- перегляд пакетів послуг.



Рисунок 2.1 – Діаграма варіантів використання

Альтернативний сценарій для перегляду власного профілю – змінювання власного профілю, що необхідно у випадку необхідності коригування власних даних профілю.

Сценаріями для працівника автомийки є:

- бронювання мийки автомобіля;
- перегляд бронювань;
- перегляд годин використання автомийки;
- робота з даними автомийок;
- робота з даними пакетів послуг з мийки автомобіля;
- робота з даними процедур детейлінгу;
- робота з даними розкладу роботи автомийки;
- робота з даними чистячих засобів;
- робота з даними обладнання;
- фіксація роботи працівника на автомийці на день.

Альтернативний сценарій для перегляду бронювань – змінювання бронювання, що необхідно у випадку зміни стану бронювання, реалізації мийки автомобіля або уточнення даних бронювання.

2.2 Вибір мови програмування

У світі існує достатньо багато мов програмування і поступово їх сфери використання і загальне число збільшується. Мовами, які дійсно активно використовуються для веброзробки, є мови C# і Python [8]-[9]. Обидві є достатньо потужними для вирішення цієї задачі, тому було виконано докладне порівняння їх між собою (табл. 2.1). Порівняння продемонструвало, що процес розгортання готового застосунку, створеного за допомогою технологій, пов'язаних з мовою Python, є простішим, окрім того кількість наявних бібліотек тут також вища, тому процес повторного використання коду дозволяє значно зменшити кількість коду, яку потрібно написати і відлагодити. До того ж код у цьому випадку трохи простіше створювати, за рахунок зокрема динамічної типізації. Отож розробка виконувалась мовою Python.

Таблиця 2.1 – Вибір мови програмування для розробки програми

Показник	C#	Python
Відкрите і вільне програмне забезпечення	Так	Так
Порівнянні витрати розгортання	Більші	Менші
Високорівнева мова	Так	Так
Наявність потужного вебфреймворку	Так	Так
Охоплення бібліотеками	Добре	Потужне
Простота створення коду	Висока	Вища, за рахунок, наприклад, динамічної типізації

2.3 Проектування бази даних

БД створювалась на основі виділених сутностей предметної області в першому розділі з врахуванням функціональних вимог до програми та на основі системи MySQL для організації сервера БД.

Клас базового розміру автомобіля визначається таблицею Carsize такими полями:

- id – ідентифікатор типу базового розміру автомобіля;
- carclass – назва класу базового розміру автомобіля;
- size – розміри автомобіля;
- pic – зображення типу базового розміру автомобіля.

Бокс автомобільної мийки визначається таблицею Box такими полями:

- id – ідентифікатор боксу автомобільної мийки;
- number – номер або позначення боксу;
- carsize – найбільший клас базового розміру автомобіля, що може обслуговуватися в цьому боксі;
- wash – автомобільна мийка, де знаходиться цей бокс;
- worked – працює зараз бокс чи ні.

Таблиця Wash має зв'язок, визначений як один до багатьох, з таблицею Box: одна мийка може включати багато боксів, проте один бокс може знаходитись тільки на одній мийці.

Таблиця Carsize має зв'язок, визначений як один до багатьох, з таблицею Box: один клас базового розміру автомобіля може підходити до багатьох боксів, проте один бокс може визначати тільки один клас базового розміру автомобіля як максимальний для нього.

Бронювання мийки автомобіля визначається таблицею Booking такими полями:

- id – ідентифікатор бронювання мийки автомобіля;
- box – бокс, у якому буде відбуватися мийка;
- client – клієнт, який виконав бронювання;
- starttime – час початку мийки;
- endtime – час закінчення мийки;
- price – сумарна вартість мийки;
- carsize – клас базового розміру автомобіля, мийку якого бажає або виконує клієнт;
- washservice – тип пакету послуг з мийки автомобіля;
- booked – час здійснення бронювання.

Таблиця Box має зв'язок, визначений як один до багатьох, з таблицею Booking: один бокс може обслуговувати багато бронювань у різний час, проте одне бронювання може здійснюватися тільки в одному конкретному боксі.

Таблиця Client має зв'язок, визначений як один до багатьох, з таблицею Booking: один клієнт може створювати різні бронювання, проте одне бронювання має бути чітко створено одним клієнтом.

Таблиця Carsize має зв'язок, визначений як один до багатьох, з таблицею Booking: один клас базового розміру автомобіля може брати участь в різних бронюваннях, проте одне бронювання може стосуватися тільки одного такого класу автомобіля, що обслуговується, тобто визначити бронювання для двох різних автомобілів не можна.

Таблиця Washservice має зв'язок, визначений як один до багатьох, з таблицею Booking: один пакет послуг з мийки автомобілів може використовуватися в різних бронюваннях, проте одне бронювання може передбачати тільки одного пакету мийки.

Пакет послуг з мийки автомобіля визначається таблицею Washservice такими полями:

- id – ідентифікатор пакету послуг з мийки автомобіля;
- name – назва пакету послуг з мийки автомобіля;
- fullnote – повний опис пакету послуг з мийки автомобіля.

Вартість пакету послуг з мийки автомобіля визначається таблицею WashservicePrice такими полями:

- id – ідентифікатор вартості пакету послуг з мийки автомобіля;
- washservice – пакет послуг з мийки автомобіля, вартість якого визначається;
- price – вартість пакету послуг з мийки автомобіля;
- carsize – клас базового розміру автомобіля, для якого встановлена вартість;
- timeduration – тривалість мийки.

Таблиця Washservice має зв'язок, визначений як один до багатьох, з таблицею WashservicePrice: один пакет послуг з мийки автомобілів може мати багато різних тарифікацій, проте одна тарифікація може стосуватися тільки одного пакету послуг з мийки автомобілів.

Таблиця Carsize має зв'язок, визначений як один до багатьох, з таблицею WashservicePrice: один клас базового розміру автомобіля може бути задіяний в різних тарифікаціях пакетів послуг з мийки автомобілів, кожна тарифікація повинна бути визнана чітко для конкретного класу базового розміру автомобіля.

Процедура детейлінгу визначається таблицею Detailing такими полями:

- id – ідентифікатор процедури детейлінгу автомобіля;
- procedurename – назва процедури детейлінгу автомобіля;
- fullnote – повний опис процедури.

Особливості процедури детейлінгу визначаються таблицею DetailingDetail такими полями:

- id – ідентифікатор особливості;
- detailing – процедура детейлінгу;
- price – вартість процедури детейлінгу;
- timeduration – тривалість процедури детейлінгу;
- carsize – клас базового розміру автомобіля, для якого встановлена особливість процедури детейлінгу.

Таблиця Detailing має зв'язок, визначений як один до багатьох, з таблицею DetailingDetail: одна процедура детейлінгу може мати багато різни особливостей для різних класів базового розміру автомобілів, проте кожна така особливість повинна бути визначена тільки для однієї процедури детейлінгу.

Стандартний розклад роботи мийки визначається таблицею Timetable такими полями:

- id – ідентифікатор розкладу;
- wash – мийка, розклад якої визначається;
- weekday – день тижня;
- workstart – час початку роботи;
- workend – час закінчення роботи;

- breakstart – час початку перерви в роботі;
- breakend – час закінчення перерви в роботі.

Таблиця Wash має зв'язок, визначений як один до багатьох, з таблицею Timetable: одна мийка може мати багато записів про розклад роботи, проте кожен такий запис є розкладом лише однієї мийки.

Фахівець мийки визначається таблицею Worker такими полями:

- id – ідентифікатор працівника мийки;
- speciality – спеціальність працівника мийки.

Таблиця Speciality має зв'язок, визначений як один до багатьох, з таблицею Worker: один фах може мати багато фахівців, проте один фахівець повинен мати тільки один фах.

Тип чистячого засобу визначається таблицею Cleanertype такими полями:

- id – ідентифікатор працівника мийки;
- cltypename – назва типу чистячого засобу.

Чистячий засіб визначається таблицею Cleaner такими полями:

- id – ідентифікатор чистячого засобу;
- brand – виробник чистячого засобу.
- cleanertype – тип чистячого засобу;
- cleaname – назва чистячого засобу;
- reused – повторно використовуваний засіб або ні;
- volume – необхідний обсяг чистячого засобу.

Таблиця Brand має зв'язок, визначений як один до багатьох, з таблицею Cleaner: один виробник може мати багато різних чистячих засобів, проте один чистячий засіб може бути вироблений тільки одним виробником.

Таблиця Cleanertype має зв'язок, визначений як один до багатьох, з таблицею Cleaner: один тип чистячого засобу може мати багато різних чистячих засобів, проте один чистячий засіб може відноситися тільки до одного типу чистячого засобу.

Тип обладнання для мийки автомобілів визначається таблицею Tooltype такими полями:

- id – ідентифікатор типу обладнання для мийки автомобілів;
- tooltypename – назва типу обладнання для мийки автомобілів.

Виробник визначається таблицею Brand такими полями:

- id – ідентифікатор виробника;
- brandtitle – назва виробника.

Обладнання для мийки автомобілів визначається таблицею Tool такими полями:

- id – ідентифікатор обладнання для мийки автомобілів;
- toolname – назва обладнання для мийки автомобілів;
- tooltype – тип обладнання для мийки автомобілів;
- brand – виробник обладнання для мийки автомобілів;
- fullnote – детальний опис обладнання для мийки автомобілів;
- quantity – наявна кількість такого обладнання.

Таблиця Tooltype має зв'язок, визначений як один до багатьох, з таблицею Tool: один тип обладнання може мати багато різних одиниць, а одна одиниця обладнання може мати тільки один конкретний тип.

Таблиця Brand має зв'язок, визначений як один до багатьох, з таблицею Tool: один виробник може мати багато різних одиниць обладнання, проте одна одиниця повинна бути вироблена тільки одним виробником.

Клієнт визначається таблицею Client такими полями:

- id – ідентифікатор клієнта;
- payed – сума, яку загалом сплатив за послуги мийки клієнт.

Мийка автомобілів визначається таблицею Wash такими полями:

- id – ідентифікатор мийки автомобілів;
- washtitle – назва мийки автомобілів;
- city – місто, де знаходиться мийка;
- address – адреса мийки;
- plan – схема боксів мийки;

– call – номер телефону мийки.

Знижка визначається таблицею Discount такими полями:

- id – ідентифікатор знижки;
- points – сумарні витрати клієнта, з яких починає діяти знижка;
- drop – знижка, яка задіюється від суми мийки;
- droptype – тип знижки (абсолютна сума або відносна, визначена відсотком).

Виключення з розкладу роботи мийки автомобілів визначається таблицею Exclude такими полями:

- id – ідентифікатор виключення;
- wash – мийка, для якої створено виключення;
- exdate – дата, для якої діє виключення;
- exctype – тип виключення (вихідний або скорочений день);
- workstart – час початку роботи;
- workend – час закінчення роботи;
- breakstart – час початку перерви в роботі;
- breakend – час закінчення перерви в роботі.

Таблиця Wash має зв'язок, визначений як один до багатьох, з таблицею Exclude: одна мийка може мати багато різних виключень з розкладу її роботи, проте одне виключення може стосуватися тільки однієї мийки.

Фах спеціаліста визначається таблицею Speciality такими полями:

- id – ідентифікатор фаху;
- spresname – назва фаху.

Використання чистячого засобу для процедури детейлінгу визначається таблицею DetailingCleaner такими полями:

- id – ідентифікатор використання чистячого засобу для процедури детейлінгу;
- detailing – процедура детейлінгу;
- cleaner – чистячий засіб;

– volume – обсяг чистячого засобу, який використовується.

Таблиця DetailingCleaner фактично реалізовує зв'язок, визначений як багато до багатьох, між таблицею Detailing і таблицею Cleaner: одна процедура детейлінгу може потребувати багато чистячих засобів, а один чистячий засіб може використовуватися в різних процедурах детейлінгу.

Використання обладнання для процедури детейлінгу визначається таблицею DetailingTool такими полями:

- id – ідентифікатор використання обладнання для процедури детейлінгу;
- detailing – процедура детейлінгу;
- tool – обладнання, задіяне в процедурі детейлінгу.

Таблиця DetailingTool фактично реалізовує зв'язок, визначений як багато до багатьох, між таблицею Detailing і таблицею Tool: одна процедура детейлінгу може потребувати багато одиниць обладнання, а одна одиниця обладнання може використовуватися в різних процедурах детейлінгу.

Робочий день фахівця визначається таблицею WorkerDay такими полями:

- id – ідентифікатор робочого дня фахівця;
- worker – фахівець;
- workday – дата робочого дня;
- wash – мийка, на якій працює фахівець у цей день.

Таблиця WorkerDay фактично реалізовує зв'язок, визначений як багато до багатьох, між таблицею Worker і таблицею Wash: один фахівець може працювати на різних мийках, а одна мийка може потребувати роботи різних фахівців.

Особливості детейлінгу, використані або заброньовані до використання під час мийки автомобіля, визначаються таблицею BookingDetailingDetail такими полями:

- id – ідентифікатор особливості детейлінгу під час мийки автомобіля;
- booking – бронювання мийки;

– id – ідентифікатор необхідності у фахівці для процедури детейлінгу;

- detailing – процедура детейлінгу;
- speciality – фах, необхідний для процедури;
- quantity – кількість фахівців даного фаху, задіяних у процедурі.

Таблиця DetailingSpeciality фактично реалізовує зв'язок, визначений як багато до багатьох, між таблицею Detailing і таблицею Speciality: одна процедура детейлінгу може потребувати фахівців різного фаху, а кожен фах може застосовуватися в різних процедурах детейлінгу.

2.4 Структура вебсистеми

При проєктуванні вебсистеми було використано вебфреймворк Django [10], що визначило основну структуру проєкту (рис. 2.3).

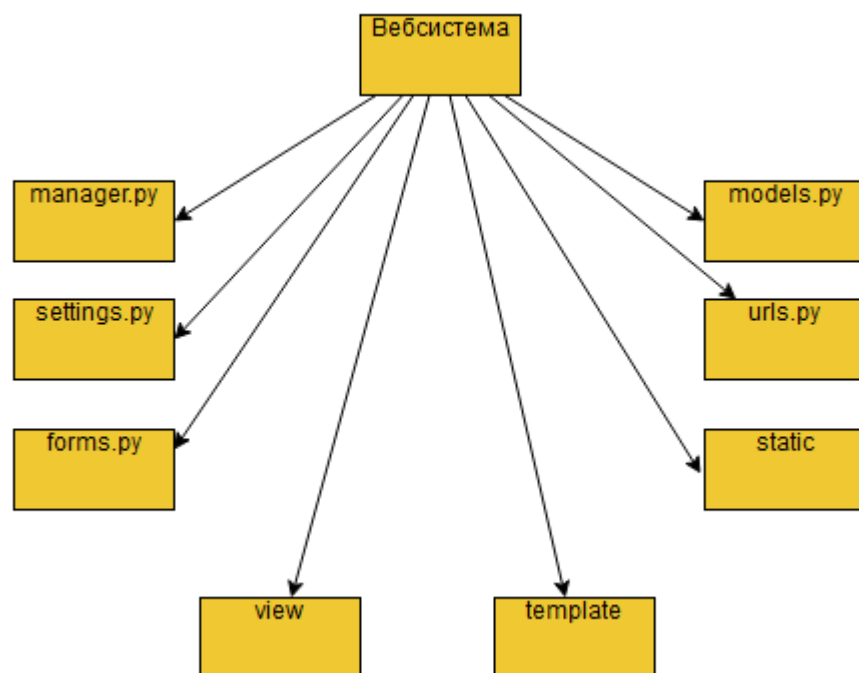


Рисунок 2.3 – Схема структури вебсистеми

До складу проєкту входять представлення і шаблони, які визначають підготовку і відображення даних користувачу, моделі, які визначають спосіб збереження даних, форми, необхідні для взаємодії з користувачем, статичні файли.

2.5 Висновки за розділом 2

Функціональні вимоги до програми було визначені, після чого їх використано при створенні сценаріїв використання програми клієнтом і працівником автомийки. Python визначений для розробки програми. Проєктування БД забезпечено у продовження виділених сутностей предметної області. За допомогою фреймворку Django визначено структуру вебсистеми.

3 РОЗРОБКА ПРОГРАМИ

3.1 Алгоритм функціонування програми

Програма фактично має різні послідовності роботи для клієнта (рис. 3.1) і для працівника автомийки.



Рисунок 3.1 – Схема функціонування вебсистеми для клієнта

Для клієнта спочатку відображається набір автомийок, а далі він може обрати відтворити послуги, власні бронювання, виконати бронювання, керувати власним профілем.

Працівник автомийки повинен одразу виконати авторизацію (рис. 3.2), а після цього йому надається можливість керування бронюваннями, після чого він може виконати роботу з даними, відтворити дані використання автомийок, керувати профілем, керувати бронюваннями.

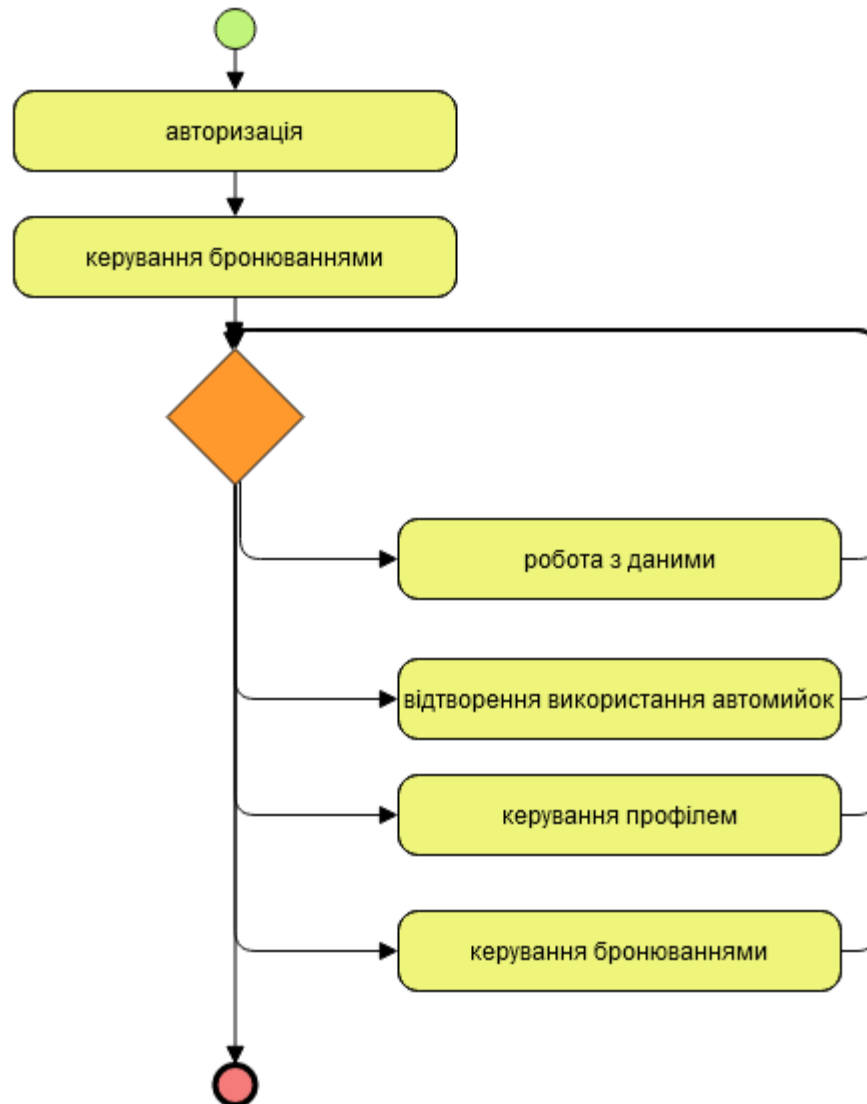


Рисунок 3.2 – Схема функціонування вебсистеми для працівника автомийки

3.2 Опис реалізованих класів програми

Клас моделей програми побудовані на основі структури таблиць, описаної в розділі 2. Відповідно на основі створення класів моделей в

подальшому була створена БД, тобто використовувався підхід зі створенням спочатку коду.

Клас центральної сутності предметної області, моделі Booking, створений як множина полів:

- box є полем, що визначає зовнішній ключ, з моделі Box, тобто боксом автомийки, де буде відбуватися або відбулось миття автомобіля;
- client є полем, що визначає зовнішній ключ, з моделі Client, тобто клієнтом, який виконує бронювання;
- discount є полем, що визначає зовнішній ключ, з моделі Discount, тобто дисконтом, який був задіяний під час виконання бронювання;
- starttime є полем дати і часу DateTimeField, що визначає час початку миття;
- endtime є полем дати і часу DateTimeField, що визначає час закінчення миття;
- price є сумарною вартістю бронювання за полем числа DecimalField з точністю до 1 знаку після крапки;
- received є сплаченою сумою за бронювання за полем числа DecimalField з точністю до 1 знаку після крапки;
- carsize є полем, що визначає зовнішній ключ, з моделі Carsize, тобто класом розміру автомобіля;
- carnumber є номером автомобіля, миття якого виконується, за символьним полем CharField з максимальною довжиною у 10 символів;
- washserviceprice є полем, що визначає зовнішній ключ, з моделі WashservicePrice, тобто вартістю пакету миття автомобіля;
- booked є полем дати і часу DateTimeField, що визначає час бронювання;
- detailingdetails є полем, яке визначає особливості детейлінгу автомобіля в бронюванні і представляє список відповідних особливостей, сформоване як поле за відношенням багато до багатьох, створюючи таблицю БД BookingDetailingDetail;

– status – поле з обмеженим набором значень, що представляє стан бронювання.

Для всіх моделей було створено наступні списки обмежених значень.

Список CARCLASSES визначає класи базового розміру автомобіля, включаючи значення:

- седан, якому відповідає символічне значення поля C;
- універсал, якому відповідає символічне значення поля U;
- автомобіль sport utility vehicle (SUV) , якому відповідає символічне значення поля SUV;
- автобус, якому відповідає символічне значення поля B;
- вантажівка, якому відповідає символічне значення поля T.

Список BOOKS визначає класи станів бронювань, включаючи значення:

- новий, якому відповідає символічне значення поля N;
- скасований, якому відповідає символічне значення поля C;
- виконаний, якому відповідає символічне значення поля P.

Список DAYS визначає дні тижня, включаючи значення:

- понеділок, якому відповідає символічне значення поля M;
- вівторок, якому відповідає символічне значення поля T;
- середа, якому відповідає символічне значення поля W;
- четвер, якому відповідає символічне значення поля H;
- п'ятниця, якому відповідає символічне значення поля F;
- субота, якому відповідає символічне значення поля S;
- неділя, якому відповідає символічне значення поля N.

Клас WashFreeListView формує представлення роботи зі списком вільних місць на певну дату на автомийці. Сама дата передається через параметр запиту до вебсистеми bookdate, номер автомийки – через параметр wash.

Для пошуку вільних місць на автомийці спочатку реалізується визначення бронювань на цій автомийці, які на цей день були виконані, з упорядкуванням їх за часом початку мийки:

```
washbookings = Booking.objects.filter(box__wash__id = wash)
bookdate = datetime.strptime(bookdate, '%d.%m.%y')
washbookings = washbookings.filter(starttime__gte bookdate)
bookdate2 = bookdate + datetime.timedelta(days=1)
washbookings = washbookings.filter(starttime__lt bookdate2)
washbookings = washbookings.order_by('starttime')
```

Після цього виділяється список всіх боксів на цій автомийці, а далі для кожного боксу виконується аналіз вільних місць в цьому боксі.

Під час аналізу доступних часових слотів спочатку з'ясовується графік роботи автомийки в цей день. Це може бути особливий графік у святковий день, зокрема це може бути вихідний день або скорочений день. У випадку скороченого дня визначаються часові рамки роботи разом з часом перерви:

```
start = excl.workstart
end = excl.workend
breakstart = excl.breakstart
breakend = excl.breakend
```

Якщо ця дата не є якимось святом або певним особливим днем з обмеженим часом роботи, то використовується стандартний графік роботи, для чого спочатку визначається, яким днем тижня є ця дата, а тоді визначається початок, завершення робочого дня та час перерви:

```
timet = Timetable.objects.filter(wash__id = wash, weekday =
weekda)
start = timet.workstart
end = timet.workend
breakstart = timet.breakstart
breakend = timet.breakend
```

Після цього для кожного боксу обробляється кожне бронювання за цей день. Визначивши попереднє і поточне бронювання, визначається наявність часового проміжку між цими бронюваннями, а також цей проміжок співставляється з часом початку, завершення робочого дня, визначеними на попередньому етапі, а також з часом перерви, який може включати, перетинатись або входити в часовий проміжок між бронюваннями:

```

        if start >= breakstart and start <= breakend and endtime >=
breakend and endtime >= breakstart:
            if start != washbook.starttime:
                intervals.append([breakend, start])
            elif start <= breakstart and start <= breakend and endtime
<= breakend and endtime >= breakstart:
                if start != breakend:
                    intervals.append([start, breakstart])
            elif start <= breakstart and start <= breakend and endtime
>= breakend and endtime >= breakstart:
                if start != excl.breakend:
                    intervals.append([start, breakstart])
                    intervals.append([breakend, endtime])

```

Далі ці інтервали узгоджуються для виділення вільних часових лотів не за боксами, а за всією автомийкою разом. Тобто якщо є вільне місце хоч в одному боксі автомийки в певний момент часу, то цей момент часу включається в перелік вільних, хоча вільним може бути тільки один бокс.

Клас `BookingView` формує представлення перегляду деталей бронювання, що ґрунтується на основі класу `DetailView`. При цьому одразу обробляється час початку і звершення мийки, перетворюючи їх на відформатовані рядки.

Клас `WashBookingsListView` формує представлення перегляду списку бронювань, що ґрунтується на основі класу `ListView`. При збиранні цих даних про бронювання використовуються значення параметрів, які передаються

через запит і стосуються дати бронювання (параметр `bookdate`), автомийки (параметр `wash`), боксу (параметр `box`). Якщо конкретний бокс не заданий, то список формується одразу для всіх боксів, які є на цій автомийці.

Клас `WashFreeBoxesListView` формує представлення перегляду вільних часових інтервалів на кожному боксі автомийки, що ґрунтується на основі класу `ListView`. Результати групуються за номером боксу, тобто результуючий словник має за кожним ключем, яким є номер боксу, переліки, які визначають доступні часові лоти. При збиранні цих даних про бронювання використовуються значення параметрів, які передаються через запит і стосуються дати бронювання (параметр `bookdate`), автомийки (параметр `wash`), боксу (параметр `box`). Якщо конкретний бокс не заданий, то список формується одразу для всіх боксів, які є на цій автомийці.

Клас `ClientBookingsListView` формує представлення перегляду бронювань за клієнтом, які були ним виконані, що ґрунтується на основі класу `ListView`. При збиранні цих даних про бронювання використовується значення параметра `client`, що визначає клієнта, який виконував ці бронювання.

Клас `PersonalBookingsListView` формує представлення перегляду бронювань, які були виконані авторизованим в даний момент користувачем, що ґрунтується на основі класу `ListView`.

Клас `NewBookingView` формує представлення створення бронювань, що ґрунтується на основі класу `CreateView`. При створенні бронювання визначаються поля боксу, де буде виконуватися мийка (автоматично вибирається вільний), часу початку мийки, часу завершення мийки та класу розміру автомобіля. Програмно вноситься час створення замовлення, клієнт, який створив замовлення. Також передається через параметри запиту пакет послуг мийки автомобілів `washservice`, а безпосередньо в представленні визначається, скільки коштує цей пакет і ця інформація вноситься в поле моделі. Також через параметр запиту `detailings` передається список ідентифікаторів процедур детейлінгу, які розділяються комами. Вони

аналізуються, виділяються окремі ідентифікатори, а за кожним ідентифікатор визначається відповідна процедура детейлінгу, встановлюється вартість такої процедури для даного класу розміру автомобіля і відповідний список таких вартостей вноситься у відповідне поле моделі. Також в результаті обчислюється вартість бронювання за вартістю пакету, вартістю процедур детейлінгу та дисконтною знижкою:

```
price = self.washserviceprice.price
if pers.discount.droptype:
    price = price * (1 - pers.discount.drop)
    for dt in self.detailingdetails:
        price = price + dt.price * (1 - pers.discount.drop)
else:
    for dt in self.detailingdetails:
        price = dt.price
    price = price - pers.discount.drop
self.price = price
```

Клас `UpdateBookingView` формує представлення оновлення бронювань, що ґрунтується на основі класу `UpdateView`. При оновленні бронювання визначаються поля боксу, часу початку мийки, часу завершення мийки, класу розміру автомобіля, номеру автомобіля, сплаченої суми та стану бронювання. Передається через параметри запиту пакет послуг мийки автомобілів `washservice`, а безпосередньо в представленні визначається, скільки коштує цей пакет і ця інформація вноситься в поле моделі. Також через параметр запиту `detailings` передається список ідентифікаторів процедур детейлінгу, які розділяються комами. Вони аналізуються, виділяються окремі ідентифікатори, а за кожним ідентифікатор визначається відповідна процедура детейлінгу, встановлюється вартість такої процедури для даного класу розміру автомобіля і відповідний список таких вартостей вноситься у відповідне поле моделі. Також в результаті обчислюється

вартість бронювання за вартістю пакету, вартістю процедур детейлінгу та дисконотною знижкою.

Клас `WashView` формує представлення перегляду деталей автомийки, що ґрунтується на основі класу `DetailView`.

Клас `CityWashListView` формує представлення перегляду автомийок за заданим містом, що ґрунтується на основі класу `ListView`. При збиранні цих даних про бронювання використовується значення параметра `city`, що визначає місто, де має знаходитись мийка автомобілів.

Клас `PersonalWashListView` формує представлення перегляду автомийок, у яких працює користувач, який є робітником автомийки, що ґрунтується на основі класу `ListView`.

Клас `NewWashView` формує представлення створення нової автомийки, що ґрунтується на основі класу `CreateView`. При створенні автомийки визначаються поля її назви, міста, де знаходиться автомийка, її адреса, номер телефону, файл зі схемою автомийки.

Клас `UpdateWashView` формує представлення оновлення даних автомийки, що ґрунтується на основі класу `UpdateView`. При створенні автомийки визначаються поля її назви, міста, де знаходиться автомийка, її адреса, номер телефону, файл зі схемою автомийки.

Клас `WashUsageDayListView` формує представлення перегляду даних про використання автомийки за задану дату, що ґрунтується на основі класу `ListView`. Надається інформація про загальну кількість доступного робочого часу автомийки, кількість використаного робочого часу, а також відсоток використання робочого часу. При збиранні цих даних використовується значення параметра `bookdate`, що визначає дату, для якої підсумовуються дані про використання автомийки, параметра `wash`, що визначає ідентифікатор самої автомийки.

Клас `WashUsageMonthListView` формує представлення перегляду даних про використання автомийки за заданий місяць, що ґрунтується на основі класу `ListView`. Надається інформація про загальну кількість

доступного робочого часу автомийки, кількість використаного робочого часу, а також відсоток використання робочого часу. При збиранні цих даних використовується значення параметра `bookdate`, що визначає місяць, для якого підсумовуються дані про використання автомийки, параметра `wash`, що визначає ідентифікатор самої автомийки.

Оскільки це представлення, які стосуються основних сутностей предметної області, то детально було описано саме їх, адже логіка формування інших представлень вебсистеми подібна до описаної.

3.3 Висновки за розділом 3

Наведено основні принципи створення моделей, представлень вебсистеми, що стосуються основних сутностей предметної області. Описані принципи їх роботи. Виконано реалізацію вебсистеми з використанням вебфреймворку Django.

4 ЕКСПЛУАТАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМИ

4.1 Призначення програми

Призначенням вебсистеми є створення системи бронювання послуг з мийки автомобілів для відслідковування і організації даного процесу менеджерами автомийок, створення єдиних правил на різних автомийках мережі, для віддаленого бронювання мийки і додаткових процедур детейлінгу клієнтами.

4.2 Умови виконання програми

Мінімальний рівень технічних засобів для забезпечення роботи програмної системи включає процесор не менш ніж 4 ядра і частотою 2,9 ГГц, оперативну пам'ять у 8 Гб, жорсткий диск у 1 Тб.

4.3 Експлуатація програми

Експлуатація вебсистеми виконується клієнтами з власних комп'ютерів і інших пристроїв та працівниками автомийки з робочих засобів.

Кожна сторінка для клієнта містить зверху карусель з зображеннями автомийки та кнопкою «Забронювати» для бронювання послуг з мийки автомобілів (рис. 4.1). Оскільки це є центральною функцією, то перехід до неї відбувається швидко з будь-якої сторінки, що надається клієнту.

Фактично клієнту надається можливість ознайомитися з доступними автомийками, пакетами послуг та роботи з власними бронюваннями. Усі ці розділи меню доступні будь-якому клієнту, але для того, щоб скористатися деякими функціями потрібно авторизуватися. Тобто якщо клієнт, будучи не авторизованим, відкриє сторінку власних бронювань через пункт меню «Бронювання», то йому буде запропоновано спершу авторизуватися.

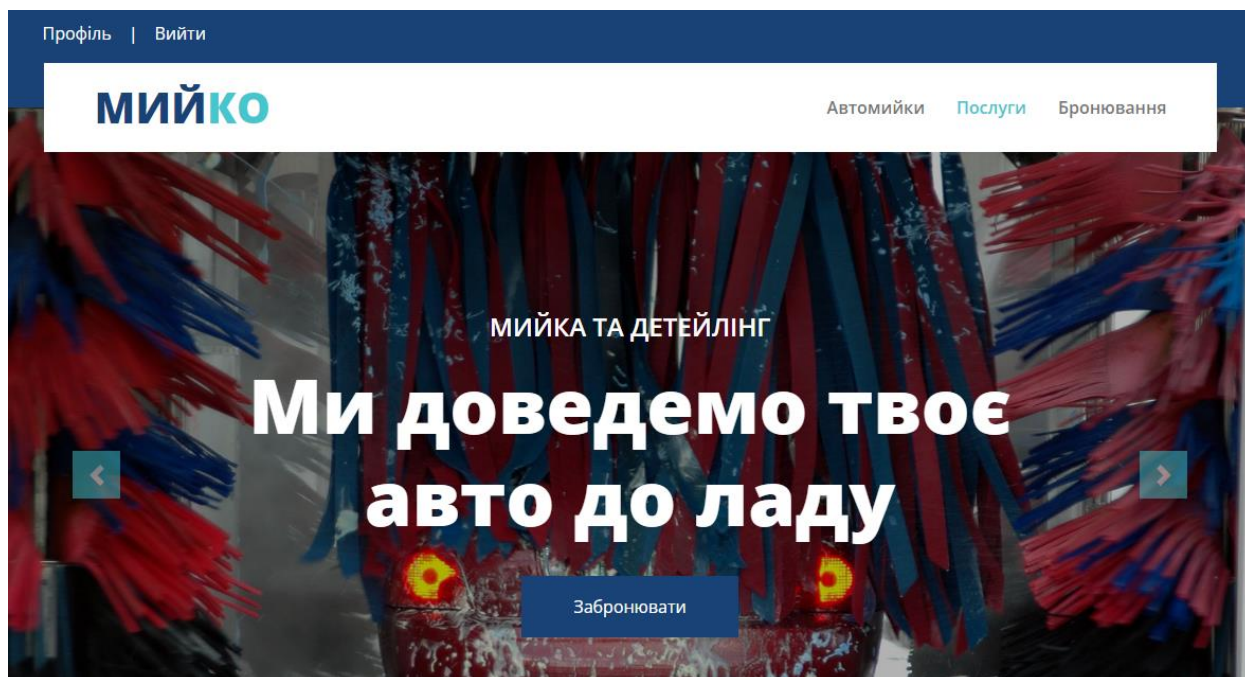


Рисунок 4.1 – Базовий інтерфейс вебсистеми

Якщо ж клієнт вже авторизувався, то йому надається можливість перегляду власних бронювань, створення нового бронювання. Процес створення нового бронювання надається за допомогою окремих етапів (рис. 4.2).

До етапів бронювання послуг з мийки автомобіля належать:

- вибір автомийки, де потрібно обрати одну з автомийок, які є в мережі, в якості автомийки, де буде реалізовуватися миття автомобіля: не можна забронювати миття автомобіля на всіх автомийках одразу;
- вибір пакету миття автомобіля: надаються варіанти миття автомобіля, при цьому спочатку потрібно обрати клас автомобіля, миття якого буде виконуватися, щоб визначити відповідно вартість кожного пакету, а тоді потрібно обрати конкретний пакет, що є основним результатом даного етапу;
- вибір необхідних процедур детейлінгу: потрібно обрати конкретну процедуру зі списку, натиснути на кнопку додати, а додана процедура додається до загального списку;

– вибір дати і часу мийки, до того ж тут відображається вартість таких послуг сумарна та необхідна тривалість виконання всіх цих процедур разом.

Нове бронювання послуг

1

2

3

4

Автомийка

Пакет миття

Детейлінг

Час

Клас автомобіля

Седан

Обрати

Вибір пакету миття

Базовий
€80

Експрес-мийка

Обрати

Повний
€150

Трьохфазна мийка

Миття вікон

Миття колес

Обрати

Преміум
€600

Трьохфазна мийка

Миття вікон

Миття колес

Миття двигуна

Чистка салону

Обрати

Рисунок 4.2 – Бронювання мийки автомобіля

У лівому верхньому куті надається можливість переходу до власного профілю, де можна зокрема побачити, скільки загалом на автомийках мережі витратив клієнт.

Працівник автомийки отримує окремий інтерфейс роботи з вебсистемою, доступний через додавання manager до посилання вебсайту, за яким надається можливість працювати з бронюваннями за автомийкою або за клієнтом через пункт «Бронювання». Також через пункт «Використання» можна отримати інформацію про те, на скільки активно використовується час роботи автомийки (рис. 4.3). Працівник одразу має авторизуватися, тільки тоді зможе працювати з цими функціями програми.

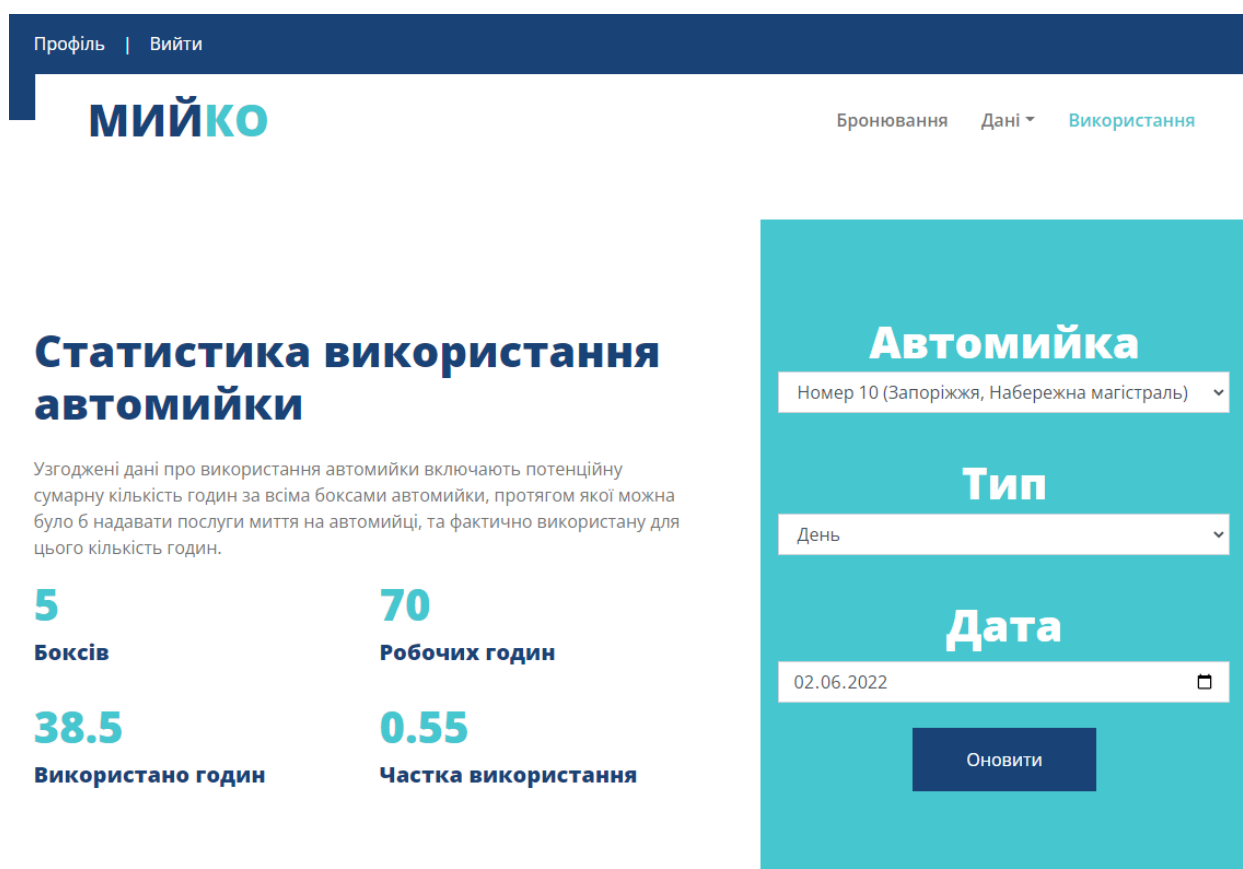


Рисунок 4.3 – Використання автомийки

Під час отримання інформації про використання потрібно обрати автомийку зі списку, тип (день або місяць відображення даних), а також конкретну дату з календаря (або конкретний місяць). Після цього необхідно натиснути на кнопку «Оновити», тоді зліва будуть відображені результати щодо кількості боксів на цій автомийці, кількості доступних робочих годин за цією датою, а також фактичну кількість використаних годин і відповідно

частку використання (співвідношення між кількістю використаних і доступних годин).

Доступний працівнику окремий розділ «Дані», який має окремі підрозділи, що дозволяють змінювати дані у відповідності з діаграмою прецедентів.

За кожним підрозділом відображається список відповідних даних, звідки можна відкрити конкретний об'єкт, наприклад, автомийку (рис. 4.4).

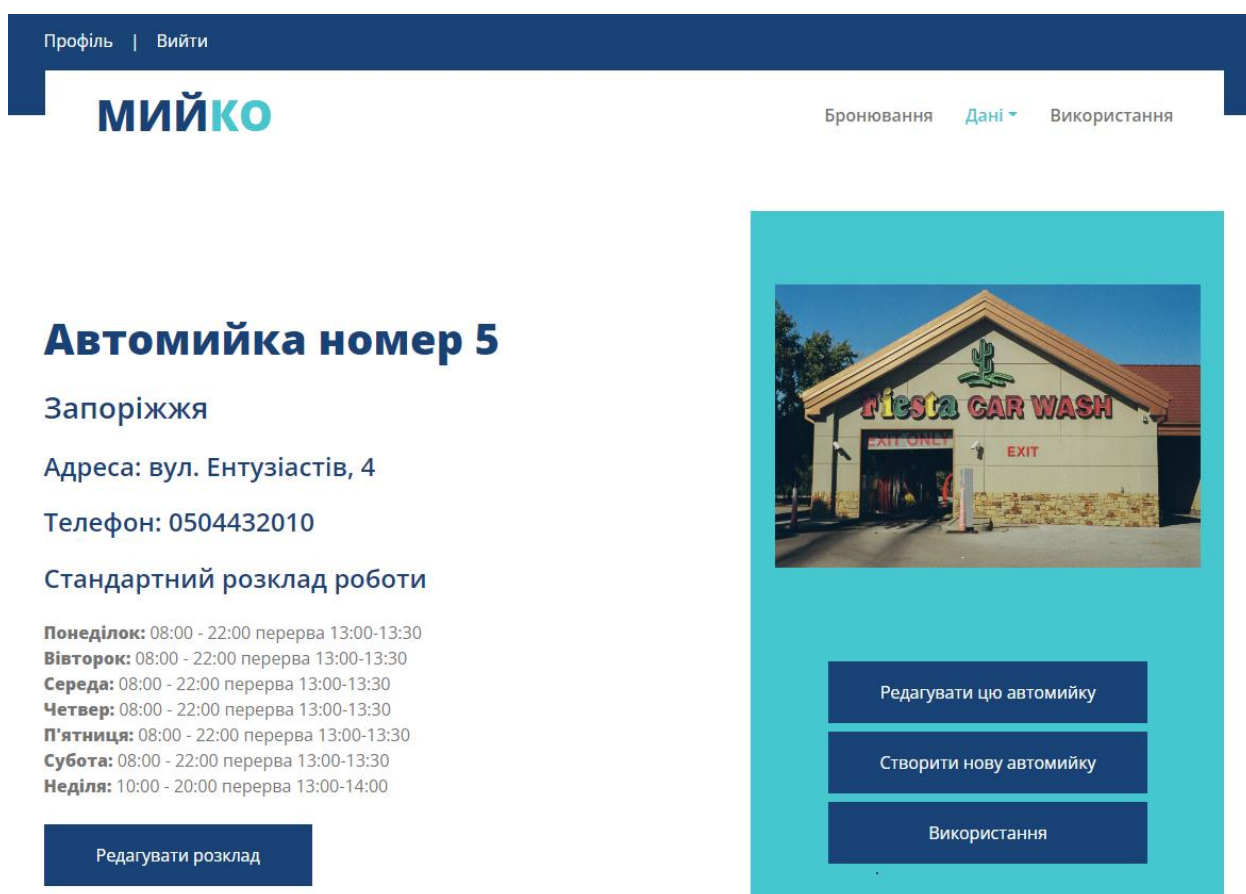


Рисунок 4.4 – Робота з даними автомийок

При відображенні даних автомийки доступна можливість редагування її даних за допомогою відповідної кнопки під схемою мийки, а також створення нової автомийки і переходу до даних про використання автомийки. За автомийкою також доступна можливість переходу до редагування розкладу автомийки. Також нижче на сторінці надається можливість внесення святкових дат з особливим розкладом роботи.

4.4 Тестування програми

Реалізовано тестування програми. Під час тестування перевірено роботу користувачів з функціями програми, які були визначені технічним завданням шляхом взаємодії користувача з програмою і введення відповідних даних, натискання на відповідні кнопки. Також було перевірено роботу користувача через стаціонарний та мобільний пристрій.

4.5 Висновки за розділом 4

За результатом виконання розділу представлено процес експлуатації вебсистеми зі сторони клієнта та працівника автомийки. Реалізовано тестування вебсистеми.

ВИСНОВКИ

Для виконання роботи реалізовано дослідження проблеми бронювання послуг з мийки автомобілів. За результатами дослідження підтверджено актуальність проблеми, тенденції її розвитку, проаналізовано аналоги, що стосуються як України, так і світу, виділено їх базові можливості і відмінності. Також визначено основні сутності предметної області.

Функціональні вимоги до програми було визначені, після чого їх використано при створенні сценаріїв використання програми клієнтом і працівником автомийки. Python визначений для розробки програми. Проєктування БД забезпечено з визначенням повної структури БД. За допомогою фреймворку Django визначено структуру вебсистеми.

Реалізуючи вебсистему, наведено основні принципи створення моделей, код яких забезпечив автоматичну побудову БД для вебсистеми, представлень вебсистеми, що стосуються основних сутностей предметної області. Описані принципи їх створення і роботи. Виконано реалізацію вебсистеми з використанням вебфреймворку Django.

Розроблена вебсистема дозволяє відслідковувати і організовувати процес бронювання послуг з мийки автомобілів менеджерами автомийок, створювати єдині правила на різних автомийках мережі, зокрема пакети миття автомобілів, окремі послуги, їх вартість, вимоги до цих процедур, аналізувати завантаженість автомийки протягом дня або місяця, а також дозволяє віддалено бронювати мийку і додаткові процедури детейлінгу клієнтами.

Програма може використовувати для підтримки роботи невеликої мережі мийок автомобілів або окремої мийки автомобілів, що складається з декількох боксів, надаючи можливість керувати обслуговуванням одночасно великої кількості клієнтів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Global Car Wash Services Market Report 2021: On-demand Car Wash Trend Continues as Consumer Preference for Professional Vehicle Wash Services Grows [Electronic resource]. – Access mode : <https://www.businesswire.com/news/home/20211202005482/en/Global-Car-Wash-Services-Market-Report-2021-On-demand-Car-Wash-Trend-Continues-as-Consumer-Preference-for-Professional-Vehicle-Wash-Services-Grows---ResearchAndMarkets.com>.
2. Tyson, J. How Car Washes Work [Electronic resource] / J. Tyson. – Access mode : <https://auto.howstuffworks.com/car-wash.htm>.
3. Norton, L. 7 Reasons The Carwash Business Is Growing [Electronic resource] / L. Norton. – Access mode : <https://www.carwash.com/7-reasons-carwash-business-growing/>.
4. 20 Car Wash Statistics in 2021 That Will Surprise You [Electronic resource]. – Access mode : <https://www.detailpropos.com/blog/car-wash-statistics/>.
5. 12 Shocking Statistics About The Car Wash Industry That You Need To Know [Electronic resource]. – Access mode : <https://www.washos.com/blog/statistics-car-wash-industry/>.
6. Автомийка «Престиж» [Електрон. ресурс]. – Режим доступу : <https://acw.com.ua/>.
7. Hand Car Wash And Detailing. Star Car Wash [Electronic resource]. – Access mode : <https://www.starcarwash.com.au/>.
8. Percival, H. Test-Driven Development with Python [Text] : Obey the Testing Goat: Using Django, Selenium, and JavaScript / H. Percival. – Massachusetts : O'Reilly Media, 2017. – 624 p.
9. Matthes, E. Python Crash Course [Text] : A Hands-On, Project-Based Introduction to Programming / E. Matthes. – San Francisco : No Starch Press, 2019. – 544 p.

10. Vincent, W. S. Django for Professionals [Text] : Production websites with Python & Django (Welcome to Django) / W. S. Vincent. – Bristol : WelcomeToCode, 2020. – 314 p.

ДОДАТОК А
Технічне завдання

Вступ

Миття автомобіля є однією з небагатьох сфер, де входження нових бізнесів є відносно простим і не потребує величезних капіталів великих мереж, адже великих гравців на цьому ринку з великою часткою ринку на даний момент у світі загалом немає, так само справи йдуть і в Україні. Тому такі окремі бізнеси потребують відповідного програмного забезпечення власної роботи з наданням зручного віддаленого бронювання послуг без необхідності обов'язкової участі менеджера.

А.1 Підстави для розробки

Розробка в межах дипломної кваліфікаційної роботи реалізується на підставі наказу № 102 від 27 квітня 2022 р. за Національним університетом «Запорізька політехніка» за темою «Програмне забезпечення бронювання послуг з мийки автомобілів».

А.2 Призначення розробки

Призначенням розробки є створення системи бронювання послуг з мийки автомобілів для відслідковування і організації даного процесу менеджерами автомийок, створення єдиних правил на різних автомийках мережі, для віддаленого бронювання мийки і додаткових процедур детейлінгу клієнтами.

А.3 Основні вимоги до програми

А.3.1 Вимоги до функціональних характеристик

Функціональні характеристики, які повинна мати програма, охоплюють:

- перегляд власних бронювань;
- бронювання мийки автомобіля;
- вибір пакету послуг з мийки автомобіля та необхідних процедур детейлінгу;
- змінювання бронювання;
- перегляд бронювань за клієнтом;
- перегляд бронювань за мийкою на дату;
- перегляд годин використання автомийки за день або за місяць;
- визначення дисконтної системи для клієнтів;
- внесення і змінювання даних автомийки;
- внесення і змінювання даних пакетів послуг з мийки автомобіля;
- внесення і змінювання даних процедур детейлінгу;
- внесення і змінювання стандартного розкладу роботи автомийки;
- внесення і змінювання розкладу роботи в святкові дні;
- фіксація роботи працівника на автомийці на день;
- внесення і змінювання даних чистячих засобів і обладнання;
- перегляд автомийок за містом;
- перегляд пакетів послуг;
- авторизація клієнтів і працівників та реєстрація клієнтів.

A.3.2 Умови експлуатації

Експлуатація програми має забезпечуватися з одного боку звичайними користувачами, які працюють з даними, наповнюють їх, а з іншого боку – адміністраторами системи та БД.

A.3.3 Вимоги до складу та параметрів технічних засобів

Мінімальний рівень технічних засобів для забезпечення роботи програмної системи включає процесор не менш ніж 4 ядра і частотою

2,9 ГГц, оперативну пам'ять у 8 Гб, жорсткий диск у 1 Тб.

А.4 Порядок контролю та приймання

Для контролю виконання роботи використовується календарний план, а приймання роботи виконується під час проведення процедури захисту роботи перед екзаменаційною комісією.

ДОДАТОК Б
Текст програми

Б.1 Текст файла bookingview.py

```

from django.contrib.auth.mixins import LoginRequiredMixin
from django.urls import reverse_lazy
from django.views.generic.edit import CreateView
from django.views.generic import ListView
from django.views.generic import UpdateView
from django.views.generic.detail import DetailView
from datetime import datetime
from datetime import timedelta
from .models import Booking
from .models import Exclude
from .models import Timetable
from .models import Box
from .models import Discount
from .models import Client

class WashFreeListView(ListView):
    model = Booking
    template_name = 'freewash.html'

    def get_queryset(self):
        bookdate = self.request.GET.get('bookdate')
        wash = self.request.GET.get('wash')
        washbookings = Booking.objects.filter(box__wash__id =
wash)

        bookdate = datetime.strptime(bookdate, '%d.%m.%y')
        washbookings = washbookings.filter(starttime__gte
bookdate)

        bookdate2 = bookdate + datetime.timedelta(days=1)
        washbookings = washbookings.filter(starttime__lt
bookdate2)

        washbookings = washbookings.order_by('starttime')
        allintervals = set()
        boxes = Box.objects.filter(wash__id = wash)
        for box in boxes:
            intervals = []
            book = None
            excl = Exclude.objects.get(wash__id = wash,
exdate = bookdate)

            start = date.today()
            end = date.today()
            breakstart = date.today()

```

```

breakend = date.today()
if excl and excl.extype:
    start = excl.workstart
    end = excl.workend
    breakstart = excl.breakstart
    breakend = excl.breakend
elif excl and not excl.extype:
    return []
else:
    weekd = bookdate.weekday()
    weekda = 'N'
    if weekd == 0:
        weekda = 'M'
    elif weekd == 1:
        weekda = 'T'
    elif weekd == 2:
        weekda = 'W'
    elif weekd == 3:
        weekda = 'H'
    elif weekd == 4:
        weekda = 'F'
    elif weekd == 5:
        weekda = 'S'
    timet = Timetable.objects.filter(wash__id =
wash, weekday = weekda)

    start = timet.workstart
    end = timet.workend
    breakstart = timet.breakstart
    breakend = timet.breakend
    lastwb = None
    for washbook in boxbookings:
        if (book and book.endtime !=
washbook.starttime) or not book:
            endtime = washbook.starttime
            if not book:
                start = book.endtime
            if start >= breakstart and start <=
breakend and endtime >= breakend and endtime >= breakstart:
                if start != washbook.starttime:
                    intervals.append([breakend,
start])

            elif start <= breakstart and start <=
breakend and endtime <= breakend and endtime >= breakstart:
                if start != breakend:

```

```

                                intervals.append([start,
breakstart])
                                elif start <= breakstart and start <=
breakend and endtime >= breakend and endtime >= breakstart:
                                    if start != excl.breakend:
                                        intervals.append([start,
breakstart])
                                intervals.append([breakend,
endtime])

                                book = washbook
                                lastwb = washbook
                                start = lastwb.endtime
                                if start != end:
                                    if start >= breakstart and start <= breakend
and end >= breakend and end >= breakstart:
                                        intervals.append([breakend, end])
                                    elif start <= breakstart and start <=
breakend and end >= breakend and end >= breakstart:
                                        intervals.append([start, breakstart])
                                        intervals.append([breakend, end])
                                    elif start >= breakstart and start <=
breakend and end >= breakend and end >= breakstart:
                                        intervals.append([breakend, end])
                                intervals.sort()
                                for interv in intervals:
                                    start = interv[0]
                                    end = interv[1]
                                    while start < end:
                                        start1 = start
                                        start = start + timedelta(minutes = 5)
                                        allintervals.add((start1, start))
                                return allintervals

    def get_context_data(self, **kwargs):
        bookparams = super(WashFreeListView,
self).get_context_data(**kwargs)
        bookdate = self.request.GET.get('bookdate')
        bookparams ['bookdate'] = bookdate
        wash = self.request.GET.get('wash')
        bookparams ['wash'] = wash
        return bookparams

class BookingView (DetailView):
    template_name = 'washbooking.html'
    model = Booking

```

```

def get_context_data(self, **kwargs):
    booking_view_data = super().get_context_data(**kwargs)
    thebooking = get_object_or_404(Booking, id =
self.object.pk)
    booking_view_data ['starttime'] =
thebooking.starttime.strftime("%d.%m.%y %H:%M")
    booking_view_data ['endtime'] =
thebooking.endtime.strftime("%d.%m.%y %H:%M")
    return booking_view_data

class WashBookingsListView(ListView):
    model = Booking
    template_name = 'washbookings.html'

def get_queryset(self):
    bookdate = self.request.GET.get('bookdate')
    wash = self.request.GET.get('wash')
    box = self.request.GET.get('box')
    if box:
        box = Booking.objects.get(id = box)
        boxes = [box]
    else:
        boxes = Box.objects.filter(wash__id = wash)
        boxes = boxes.filter(worked = True)
    washbookings = Booking.objects.filter(box__wash__id =
wash)

    bookdate = datetime.strptime(bookdate, '%d.%m.%y')
    washbookings = washbookings.filter(starttime__gte
bookdate)

    bookdate = bookdate + timedelta(days=1)
    washbookings = washbookings.filter(starttime__lt
bookdate)

    allwashbookings = []
    for box in boxes:
        boxbookings = washbookings.filter(box__id = box)
        boxbookings = boxbookings.order_by('starttime')
        allwashbookings.append(boxbookings)
    return allwashbookings

def get_context_data(self, **kwargs):
    bookparams = super(WashBookingsListView,
self).get_context_data(**kwargs)
    bookdate = self.request.GET.get('bookdate')
    bookparams ['bookdate'] = bookdate
    wash = self.request.GET.get('wash')

```

```

        bookparams ['wash'] = wash
        box = self.request.GET.get('box')
        bookparams ['box'] = box
        return bookparams

class WashFreeBoxesListView(ListView):
    model = Booking
    template_name = 'freewashbookings.html'

    def get_queryset(self):
        bookdate = self.request.GET.get('bookdate')
        wash = self.request.GET.get('wash')
        box = self.request.GET.get('box')
        if box:
            box = Booking.objects.get(id = box)
            boxes = [box]
        else:
            boxes = Box.objects.filter(wash__id = wash)
            boxes = boxes.filter(worked = True)
            washbookings = Booking.objects.filter(box__wash__id =
wash)

            bookdate = datetime.strptime(bookdate, '%d.%m.%y')
            washbookings = washbookings.filter(starttime__gte
bookdate)

            bookdate2 = bookdate + timedelta(days=1)
            washbookings = washbookings.filter(starttime__lt
bookdate2)

            washbookings = washbookings.order_by('starttime')
            allintervals = {}
            for box in boxes:
                boxbookings = washbookings.filter(box__id = box)
                boxbookings = boxbookings.order_by('starttime')
                intervals = []
                book = None
                excl = Exclude.objects.get(wash__id = wash,
exdate = bookdate)

                start = date.today()
                end = date.today()
                breakstart = date.today()
                breakend = date.today()
                if excl and excl.extype:
                    start = excl.workstart
                    end = excl.workend
                    breakstart = excl.breakstart
                    breakend = excl.breakend

```

```

elif excl and not excl.extype:
    return []
else:
    weekd = bookdate.weekday()
    weekda = 'N'
    if weekd == 0:
        weekda = 'M'
    elif weekd == 1:
        weekda = 'T'
    elif weekd == 2:
        weekda = 'W'
    elif weekd == 3:
        weekda = 'H'
    elif weekd == 4:
        weekda = 'F'
    elif weekd == 5:
        weekda = 'S'
    timet = Timetable.objects.filter(wash__id =
wash, weekday = weekda)

    start = timet.workstart
    end = timet.workend
    breakstart = timet.breakstart
    breakend = timet.breakend
    lastwb = None
    for washbook in boxbookings:
        if (book and book.endtime !=
washbook.starttime) or not book:
            endtime = washbook.starttime
            if not book:
                start = book.endtime
            if start >= breakstart and start <=
breakend and endtime >= breakend and endtime >= breakstart:
                if start != washbook.starttime:
                    intervals.append([breakend,
start])

                elif start <= breakstart and start <=
breakend and endtime <= breakend and endtime >= breakstart:
                    if start != breakend:
                        intervals.append([start,
breakstart])

                elif start <= breakstart and start <=
breakend and endtime >= breakend and endtime >= breakstart:
                    if start != excl.breakend:
                        intervals.append([start,
breakstart])

```

```

                                intervals.append([breakend,
endtime]))
                                book = washbook
                                lastwb = washbook
                                start = lastwb.endtime
                                if start != end:
                                    if start >= breakstart and start <= breakend
and end >= breakend and end >= breakstart:
                                        intervals.append([breakend, end])
                                    elif start <= breakstart and start <=
breakend and end >= breakend and end >= breakstart:
                                        intervals.append([start, breakstart])
                                        intervals.append([breakend, end])
                                    elif start >= breakstart and start <=
breakend and end >= breakend and end >= breakstart:
                                        intervals.append([breakend, end])
                                allintervals[box.number] = intervals
                                return allintervals

    def get_context_data(self, **kwargs):
        bookparams = super(WashFreeBoxesListView,
self).get_context_data(**kwargs)
        bookdate = self.request.GET.get('bookdate')
        bookparams ['bookdate'] = bookdate
        wash = self.request.GET.get('wash')
        bookparams ['wash'] = wash
        box = self.request.GET.get('box')
        bookparams ['box'] = box
        return bookparams

class ClientBookingsListView(ListView):
    model = Booking
    template_name = 'clientbookings.html'

    def get_queryset(self):
        bookclient = self.request.GET.get('client')
        clbookings = Booking.objects.filter(client__id =
bookclient)

        clbookings = clbookings.order_by('starttime')
        return clbookings

    def get_context_data(self, **kwargs):
        bookparams = super(ClientBookingsListView,
self).get_context_data(**kwargs)
        bookclient = self.request.GET.get('client')

```

```

        bookparams ['client'] = bookclient
        return bookparams

class PersonalBookingsListView(ListView):
    model = Booking
    template_name = 'persbookings.html'

    def get_queryset(self):
        pers = self.request.user
        clbookings = Booking.objects.filter(client__id = pers)
        clbookings = clbookings.order_by('starttime')
        return clbookings

class NewBookingView(LoginRequiredMixin, CreateView):
    model = Booking
    fields = ['box', 'starttime', 'endtime', 'carsize']
    template_name = 'newbooking.html'

    def get_success_url(self):
        return reverse_lazy('booking-new', kwargs={'pk':
self.object.pk})

    def get_context_data(self, **kwargs):
        washservice = get_object_or_404(Washservice,
id=self.kwargs['washservice'])
        kwargs['washservice'] = washservice
        return super().get_context_data(**kwargs)

    def form_valid(self, form):
        self.booked = datetime.now()
        pers = get_object_or_404(Client, id=self.request.user)
        self.client = pers
        self.discount = pers.discount
        washservice = get_object_or_404(Washservice,
id=self.kwargs['washservice'])
        self.washserviceprice =
WashservicePrice.objects.get(carsize = self.carsize, washservice
= washservice)
        detailings = str(self.request.POST.get('detailings'))
        detailings = detailings.split(',')
        dets = []
        for d in detailings:

            dets.append(DetailingDetail.objects.filter(detailing_id = d,
carsize = self.carsize))

```

```

        self.detailingdetails = dets
        price = self.washserviceprice.price
        if pers.discount.droptype:
            price = price * (1 - pers.discount.drop)
            for dt in self.detailingdetails:
                price = price + dt.price * (1 -
pers.discount.drop)
        else:
            for dt in self.detailingdetails:
                price = dt.price
            price = price - pers.discount.drop
        self.price = price
        duration = self.washserviceprice.timeduration
        for dt in self.detailingdetails:
            duration = duration + dt.timeduration
        self.time = duration
        return super().form_valid(form)

class UpdateBookingView(LoginRequiredMixin, UpdateView):
    model = Booking
    fields = ['box', 'starttime', 'endtime', 'carsize',
'carnumber', 'received', 'status']
    template_name = 'updatebooking.html'

    def get_success_url(self):
        return reverse_lazy('booking-update', kwargs={'pk':
self.object.pk})

    def get_context_data(self, **kwargs):
        washservice = get_object_or_404(Washservice,
id=self.kwargs['washservice'])
        kwargs['washservice'] = washservice
        return super().get_context_data(**kwargs)

    def form_valid(self, form):
        washservice = get_object_or_404(Washservice,
id=self.kwargs['washservice'])
        self.washserviceprice =
WashservicePrice.objects.get(carsize = self.carsize, washservice
= washservice)
        detailings = str(self.request.POST.get('detailings'))
        detailings = detailings.split(',')
        dets = []
        for d in detailings:

```

```

        dets.append(DetailingDetail.objects.filter(detailing_id = d,
carsize = self.carsize))
        self.detailingdetails = dets
        price = self.washserviceprice.price
        if pers.discount.droptype:
            price = price * (1 - pers.discount.drop)
            for dt in self.detailingdetails:
                price = price + dt.price * (1 -
pers.discount.drop)
        else:
            for dt in self.detailingdetails:
                price = dt.price
            price = price - pers.discount.drop
        self.price = price
        duration = self.washserviceprice.timeduration
        for dt in self.detailingdetails:
            duration = duration + dt.timeduration
        self.time = duration
        return super().form_valid(form)

```

Б.2 Текст файла models.py

```

from django.db import models
from django.utils import timezone
from django.urls import reverse
from django.contrib.auth.models import AbstractUser

```

```

CARCLASSES = [
    ('C', 'седан'),
    ('U', 'універсал'),
    ('S', 'SUV'),
    ('B', 'автобус'),
    ('T', 'вантажівка'),
]

```

```

BOOKS = [
    ('N', 'нове'),
    ('C', 'скасоване'),
    ('P', 'виконане'),
]

```

```

DAYS = [
    ('M', "понеділок"),
    ('T', "вівторок"),
    ('W', "середа"),
    ('H', "четвер"),
    ('F', "п'ятниця"),
    ('S', "субота"),
    ('N', "неділя"),
]

class Carsize (models.Model):
    carclass = models.CharField(max_length=1,
choices=CARCLASSES, default='C')
    size = models.CharField(max_length=50)
    pic = models.FileField(upload_to='images/cars/')

    def __str__(self):
        if self.carclass == 'C':
            return 'седан'
        elif self.carclass == 'U':
            return 'універсал'
        elif self.carclass == 'S':
            return 'SUV'
        elif self.carclass == 'B':
            return 'автобус'
        elif self.carclass == 'T':
            return 'вантажівка'

class Box (models.Model):
    number = models.CharField(max_length=5)
    carsize = models.ForeignKey(Carsize,
on_delete=models.CASCADE)
    wash = models.ForeignKey(Wash, on_delete=models.CASCADE)
    worked = models.BooleanField()

    def __str__(self):
        return self.number

class Booking (models.Model):
    box = models.ForeignKey(Box, related_name='boxbookings',
on_delete=models.CASCADE)
    client = models.ForeignKey(Client,
related_name='clientbookings', on_delete=models.CASCADE)
    discount = models.ForeignKey(Discount,
related_name='discountbookings', on_delete=models.CASCADE)

```

```

        starttime = models.DateTimeField()
        endtime = models.DateTimeField()
        price = models.DecimalField(max_digits = 10, decimal_places
= 1, default = 0)
        received = models.DecimalField(max_digits = 10,
decimal_places = 1, default = 0)
        carsize = models.ForeignKey(Carsize,
on_delete=models.CASCADE)
        carnumber = models.CharField(max_length=10)
        washserviceprice = models.ForeignKey(WashservicePrice,
on_delete=models.CASCADE)
        booked = models.DateTimeField()
        detailingdetails = models.ManyToManyField(DetailingDetail,
related_name = 'detailingdetailbookings', db_table =
'BookingDetailingDetail')
        status = models.CharField(max_length=1, choices=BOOKS,
default='N')

    def __str__(self):
        return self.box + ' ' +
self.starttime.strftime("%H:%M") + ' - ' +
self.endtime.strftime("%H:%M")

    def get_absolute_url(self):
        return reverse('bookingview', kwargs={'pk': self.pk})

    def save(self, *args, **kwargs):
        self.booked = datetime.now
        uclient = self.client
        books = Booking.objects.filter(client = uclient)
        total = 0
        for book in books:
            total = total + book.received
        uclient.payed = total
        books = Discount.objects.filter(points__lte = total)
        books = books.order_by('-points')
        uclient.discount = books.first()
        super(Booking, self).save(*args, **kwargs)

class Washservice (models.Model):
    name = models.CharField(max_length=100)
    fullnote= models.TextField()

    def __str__(self):
        return self.name

```

```

    def get_absolute_url(self):
        return reverse('washservview', kwargs={'pk': self.pk})

class WashservicePrice (models.Model):
    washservice = models.ForeignKey(Washservice,
related_name='washservprices', on_delete=models.CASCADE)
    price = models.DecimalField(max_digits = 10, decimal_places
= 1, default = 0)
    carsize = models.ForeignKey(Carsize,
on_delete=models.CASCADE)
    timeduration = models.TimeField()

class Detailing (models.Model):
    procedurename = models.CharField(max_length=100)
    fullnote= models.TextField()
    tools = models.ManyToManyField(Tool, related_name =
'tooldetailings', db_table = 'DetailingTool')

    def __str__(self):
        return self.procedurename

    def get_absolute_url(self):
        return reverse('detailingview', kwargs={'pk':
self.pk})

class DetailingDetail (models.Model):
    detailing = models.ForeignKey(Detailing,
related_name='detailingdetails', on_delete=models.CASCADE)
    price = models.DecimalField(max_digits = 10, decimal_places
= 1, default = 0)
    carsize = models.ForeignKey(Carsize,
on_delete=models.CASCADE)
    timeduration = models.TimeField()

class Timetable (models.Model):
    wash = models.ForeignKey(Wash,
related_name='washtimetables', on_delete=models.CASCADE)
    weekday = models.CharField(max_length=1, choices=DAYS,
default='M')
    workstart = models.TimeField()
    workend = models.TimeField()
    breakstart = models.TimeField()
    breakend = models.TimeField()

    def __str__(self):

```

```

        return self.workstart + ' - ' + self.workend

class Worker (AbstractUser):
    speciality = models.ForeignKey(Speciality,
related_name='specialityworkers', on_delete=models.CASCADE)

    def __str__(self):
        return self.first_name + ' ' + self.last_name

    def get_absolute_url(self):
        return reverse('workerview', kwargs={'pk': self.pk})

class Cleanertype (models.Model):
    cltypename = models.CharField(max_length=100)

    def __str__(self):
        return self.cltypename

    def get_absolute_url(self):
        return reverse('cleanertypeview', kwargs={'pk':
self.pk}))

class Tooltype (models.Model):
    tooltypename = models.CharField(max_length=100)

    def __str__(self):
        return self.tooltypename

    def get_absolute_url(self):
        return reverse('tooltypeview', kwargs={'pk': self.pk}))

class Cleaner (models.Model):
    brand = models.ForeignKey(Brand,
related_name='brandcleaners', on_delete=models.CASCADE)
    cleanertype = models.ForeignKey(Brand,
related_name='cleanertypecleaners', on_delete=models.CASCADE)
    cleaname = models.CharField(max_length=100)
    reused = models.BooleanField()
    volume = models.DecimalField(max_digits = 5, decimal_places
= 2, default = 0)

    def __str__(self):
        return self.cleaname

    def get_absolute_url(self):

```

```

        return reverse('cleanerview', kwargs={'pk': self.pk})

class Brand (models.Model):
    brandtitle = models.CharField(max_length=30)

    def __str__(self):
        return self.brandtitle

class Tool (models.Model):
    brand = models.ForeignKey(Brand,
related_name='brandcleaners', on_delete=models.CASCADE)
    tooltype = models.ForeignKey(Tooltype,
related_name='tooltypetools', on_delete=models.CASCADE)
    toolname = models.CharField(max_length=100)
    fullnote = models.TextField()
    quantity = models.IntegerField()

    def __str__(self):
        return self.toolname

    def get_absolute_url(self):
        return reverse('toolview', kwargs={'pk': self.pk})

class Client (AbstractUser):
    payed = models.DecimalField(max_digits = 10 decimal_places =
1, default = 0)
    discount = models.ForeignKey(Discount,
on_delete=models.CASCADE)

    def __str__(self):
        return self.first_name + ' ' + self.last_name

    def get_absolute_url(self):
        return reverse('clientview', kwargs={'pk': self.pk})

class Wash (models.Model):
    washtitle = models.CharField(max_length=50)
    city = models.CharField(max_length=30)
    address = models.CharField (max_length=200)
    plan = models.FileField(upload_to='images/cars/')
    call = models.CharField(max_length=13)

    def __str__(self):
        return self.washtitle

```

```

    def get_absolute_url(self):
        return reverse('washview', kwargs={'pk': self.pk})

class Discount (models.Model):
    points = models.DecimalField(max_digits = 10 decimal_places
= 1, default = 0)
    drop = models.DecimalField(max_digits = 10 decimal_places =
2, default = 0)
    droptype = models.BooleanField()

class Exclude (models.Model):
    wash = models.ForeignKey(Wash, related_name='brandcleaners',
on_delete=models.CASCADE)
    exdate = models.DateTimeField()
    extype = models.BooleanField()
    workstart = models.TimeField()
    workend = models.TimeField()
    breakstart = models.TimeField()
    breakend = models.TimeField()

    def __str__(self):
        return self.workstart + ' - ' + self.workend

class Speciality (models.Model):
    specname = models.CharField(max_length=30)

    def __str__(self):
        return self.specname

class DetailingCleaner (models.Model):
    detailing = models.ForeignKey(Detailing,
related_name='ddetailingcleaners', on_delete=models.CASCADE)
    cleaner = models.ForeignKey(Cleaner,
related_name='cleanerdetailingcleaners', on_delete=models.CASCADE)
    volume = models.DecimalField(max_digits = 5, decimal_places
= 2, default = 0)

class DetailingSpeciality (models.Model):
    detailing = models.ForeignKey(Detailing,
related_name='ddetailingcleaners', on_delete=models.CASCADE)
    speciality = models.ForeignKey(Speciality,
related_name='specialitydetailingspecialities',
on_delete=models.CASCADE)
    quantity = models.IntegerField()

```

```

class WorkerDay (models.Model):
    worker = models.ForeignKey(Worker,
related_name='wworkerdays', on_delete=models.CASCADE)
    wash = models.ForeignKey(Wash,
related_name='washworkerdays', on_delete=models.CASCADE)
    workday = models.DateTimeField()

```

Б.3 Текст файла washview.py

```

from django.contrib.auth.mixins import LoginRequiredMixin
from django.urls import reverse_lazy
from django.views.generic.edit import CreateView
from django.views.generic import ListView
from django.views.generic import UpdateView
from django.views.generic.detail import DetailView
from .models import Wash
from .models import WorkerDay
from .models import Worker

class WashView (DetailView):
    template_name = 'wash.html'
    model = Wash

class CityWashListView(ListView):
    model = Wash
    template_name = 'citywashes.html'

    def get_queryset(self):
        cityname = self.request.GET.get('city')
        washes = Wash.objects.filter(city = cityname)
        return washes

    def get_context_data(self, **kwargs):
        washparams = super(CityWashListView,
self).get_context_data(**kwargs)
        cityname = self.request.GET.get('city')
        washparams ['cityname'] = cityname
        return washparams

class PersonalWashListView(ListView):
    model = Wash
    template_name = 'perswashes.html'

```

```

def get_queryset(self):
    pers = self.request.user
    work = Worker.objects.get(id = pers)
    wdays = WorkerDay.objects.filter(worker = work)
    washes = []
    for wd in wdays:
        if wd.wash not in washes:
            washes.append(wd.wash)
    return washes

class NewWashView(LoginRequiredMixin, CreateView):
    model = Wash
    fields = ['washtitle', 'city', 'address', 'call', 'plan']
    template_name = 'newwash.html'

    def get_success_url(self):
        return reverse_lazy('wash-new', kwargs={'pk':
self.object.pk})

class UpdateWashView(LoginRequiredMixin, UpdateView):
    model = Wash
    fields = ['washtitle', 'city', 'address', 'call', 'plan']
    template_name = 'updatewash.html'

    def get_success_url(self):
        return reverse_lazy('wash-update', kwargs={'pk':
self.object.pk})

class WashUsageDayListView(ListView):
    model = Wash
    template_name = 'washusage.html'

    def get_queryset(self):
        bookdate = self.request.GET.get('bookdate')
        wash = self.request.GET.get('wash')
        boxes = Box.objects.filter(wash__id = wash)
        washbookings = Booking.objects.filter(box__wash__id =
wash)

        bookdate = datetime.strptime(bookdate, '%d.%m.%y')
        washbookings = washbookings.filter(starttime__gte
bookdate)

        bookdate2 = bookdate + timedelta(days=1)
        washbookings = washbookings.filter(starttime__lt
bookdate2)

        washbookings = washbookings.order_by('starttime')

```

```

alltime = 0
usedtime = 0
for box in boxes:
    boxbookings = washbookings.filter(box__id = box)
    boxbookings = boxbookings.order_by('starttime')
    intervals = []
    book = None
    excl = Exclude.objects.get(wash__id = wash,
exdate = bookdate)
    start = date.today()
    end = date.today()
    breakstart = date.today()
    breakend = date.today()
    if excl and excl.extype:
        start = excl.workstart
        end = excl.workend
        breakstart = excl.breakstart
        breakend = excl.breakend
    elif excl and not excl.extype:
        return []
    else:
        weekd = bookdate.weekday()
        weekda = 'N'
        if weekd == 0:
            weekda = 'M'
        elif weekd == 1:
            weekda = 'T'
        elif weekd == 2:
            weekda = 'W'
        elif weekd == 3:
            weekda = 'H'
        elif weekd == 4:
            weekda = 'F'
        elif weekd == 5:
            weekda = 'S'
        timet = Timetable.objects.filter(wash__id =
wash, weekday = weekda)
        start = timet.workstart
        end = timet.workend
        breakstart = timet.breakstart
        breakend = timet.breakend
    res = end - start
    alltime = alltime + res.total_seconds()
    lastwb = None
    for washbook in boxbookings:

```

```

        if (book and book.endtime !=
washbook.starttime) or not book:
            endtime = washbook.starttime
            if not book:
                start = book.endtime
            if start >= breakstart and start <=
breakend and endtime >= breakend and endtime >= breakstart:
                if start != washbook.starttime:
                    ut = start - breakend
                    usedtime = usedtime +
ut.total_seconds()
                elif start <= breakstart and start <=
breakend and endtime <= breakend and endtime >= breakstart:
                    if start != breakend:
                        ut = breakstart - start
                        usedtime = usedtime +
ut.total_seconds()
                elif start <= breakstart and start <=
breakend and endtime >= breakend and endtime >= breakstart:
                    if start != excl.breakend:
                        ut = breakstart - start
                        usedtime = usedtime +
ut.total_seconds()
                    ut = endtime - breakend
                    usedtime = usedtime +
ut.total_seconds()
            book = washbook
            lastwb = washbook
            start = lastwb.endtime
            if start != end:
                if start >= breakstart and start <= breakend
and end >= breakend and end >= breakstart:
                    ut = end - breakend
                    usedtime = usedtime +
ut.total_seconds()
                elif start <= breakstart and start <=
breakend and end >= breakend and end >= breakstart:
                    ut = breakstart - start
                    usedtime = usedtime +
ut.total_seconds()
                    ut = end - breakend
                    usedtime = usedtime +
ut.total_seconds()
                elif start >= breakstart and start <=
breakend and end >= breakend and end >= breakstart:

```

```

        ut = end - breakend
        usedtime = usedtime +

ut.total_seconds()
    allres = {}
    allres['used'] = usedtime
    allres['all'] = alltime
    allres['part'] = usedtime / alltime
    allres['boxes'] = boxes.count()
    return allres

    def get_context_data(self, **kwargs):
        bookparams = super(WashUsageDayListView,
self).get_context_data(**kwargs)
        bookdate = self.request.GET.get('bookdate')
        bookparams ['bookdate'] = bookdate
        wash = self.request.GET.get('wash')
        bookparams ['wash'] = wash
        return bookparams

class WashUsageMonthListView(ListView):
    model = Wash
    template_name = 'washmonthusage.html'

    def get_queryset(self):
        bookdate = self.request.GET.get('bookdate')
        wash = self.request.GET.get('wash')
        boxes = Box.objects.filter(wash__id = wash)
        washbookings = Booking.objects.filter(box__wash__id =
wash)

        startbookdate = datetime.strptime(bookdate,
'%d.%m.%y')

        endbookdate = startbookdate
        if startbookdate.month == 12:
            endbookdate =
datetime.datetime(startbookdate.year + 1, 1, startbookdate.day) -
timedelta(days = 1)
        else:
            endbookdate =
datetime.datetime(startbookdate.year, startbookdate.month + 1,
startbookdate.day) - timedelta(days = 1)
        while bookdate < endbookdate:
            washbookings = washbookings.filter(starttime__gte
bookdate)

            bookdate2 = bookdate + timedelta(days=1)

```

```

washbookings = washbookings.filter(starttime__lte
bookdate2)

washbookings = washbookings.order_by('starttime')
alltime = 0
usedtime = 0
for box in boxes:
    boxbookings = washbookings.filter(box__id =
box)

    boxbookings =
boxbookings.order_by('starttime')
    intervals = []
    book = None
    excl = Exclude.objects.get(wash__id = wash,
exdate = bookdate)

    start = date.today()
    end = date.today()
    breakstart = date.today()
    breakend = date.today()
    if excl and excl.extype:
        start = excl.workstart
        end = excl.workend
        breakstart = excl.breakstart
        breakend = excl.breakend
    elif excl and not excl.extype:
        return []
    else:
        weekd = bookdate.weekday()
        weekda = 'N'
        if weekd == 0:
            weekda = 'M'
        elif weekd == 1:
            weekda = 'T'
        elif weekd == 2:
            weekda = 'W'
        elif weekd == 3:
            weekda = 'H'
        elif weekd == 4:
            weekda = 'F'
        elif weekd == 5:
            weekda = 'S'
        timet =
T timetable.objects.filter(wash__id = wash, weekday = weekda)
        start = timet.workstart
        end = timet.workend
        breakstart = timet.breakstart

```

```

        breakend = timet.breakend
        res = end - start
        alltime = alltime + res.total_seconds()
        lastwb = None
        for washbook in boxbookings:
            if (book and book.endtime !=
washbook.starttime) or not book:
                endtime = washbook.starttime
                if not book:
                    start = book.endtime
                    if start >= breakstart and start
<= breakend and endtime >= breakend and endtime >= breakstart:
                        if start !=
washbook.starttime:
                            ut = start - breakend
                            usedtime = usedtime +
ut.total_seconds()
                        elif start <= breakstart and
start <= breakend and endtime <= breakend and endtime >= breakstart:
                            if start != breakend:
                                ut = breakstart -
start
                                usedtime = usedtime +
ut.total_seconds()
                        elif start <= breakstart and
start <= breakend and endtime >= breakend and endtime >= breakstart:
                            if start != excl.breakend:
                                ut = breakstart -
start
                                usedtime = usedtime +
ut.total_seconds()
                                ut = endtime -
breakend
                                usedtime = usedtime +
ut.total_seconds()
                    book = washbook
                    lastwb = washbook
                    start = lastwb.endtime
                    if start != end:
                        if start >= breakstart and start <=
breakend and end >= breakend and end >= breakstart:
                            ut = end - breakend
                            usedtime = usedtime +
ut.total_seconds()

```

```

                                elif start <= breakstart and start <=
breakend and end >= breakend and end >= breakstart:
                                    ut = breakstart - start
                                    usedtime = usedtime +

ut.total_seconds()

                                    ut = end - breakend
                                    usedtime = usedtime +

ut.total_seconds()

                                elif start >= breakstart and start <=
breakend and end >= breakend and end >= breakstart:
                                    ut = end - breakend
                                    usedtime = usedtime +

ut.total_seconds()
        allres = {}
        allres['used'] = usedtime
        allres['all'] = alltime
        allres['part'] = usedtime / alltime
        allres['boxes'] = boxes.count()
        return allres

    def get_context_data(self, **kwargs):
        bookparams = super(WashUsageMonthListView,
self).get_context_data(**kwargs)
        bookdate = self.request.GET.get('bookdate')
        bookparams ['bookdate'] = bookdate
        wash = self.request.GET.get('wash')
        bookparams ['wash'] = wash
        return bookparams

```

ДОДАТОК В
Слайди презентації

Національний університет «Запорізька політехніка»
Кафедра програмних засобів

Дипломна кваліфікаційна робота

**ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ БРОНЮВАННЯ ПОСЛУГ З
МИЙКИ АВТОМОБІЛІВ**

Виконав
студент групи КНТ-219сп

О.О. Кучеренко

Керівник
к.т.н., доцент

С.Ю. Скрупський

2022

Рисунок В.1 – Слайд № 1

Об'єкт, предмет, мета роботи

Об'єкт роботи – процес миття автомобілів.

Предмет роботи – програмне забезпечення бронювання послуг з мийки автомобілів.

Мета роботи – розробити програмне забезпечення бронювання послуг з мийки автомобілів для забезпечення автоматизації даного процесу.

Рисунок В.2 – Слайд № 2

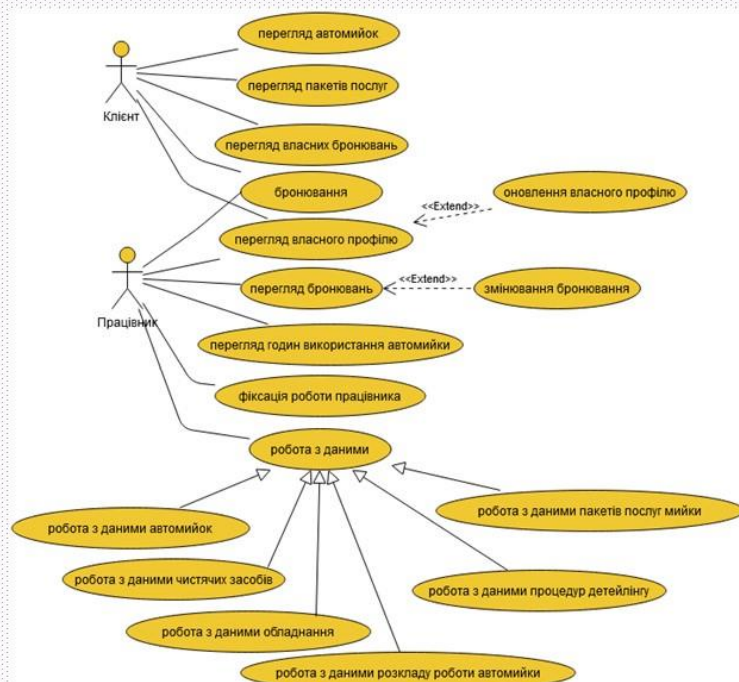
Завдання роботи

- ✓ дослідження проблеми бронювання послуг з мийки автомобілів;
- ✓ визначення засобів розробки;
- ✓ визначення вимог до програми;
- ✓ проєктування програми бронювання послуг з мийки автомобілів;
- ✓ програмна реалізація.

3

Рисунок В.3 – Слайд № 3

Діаграма варіантів використання



4

Рисунок В.4 – Слайд № 4

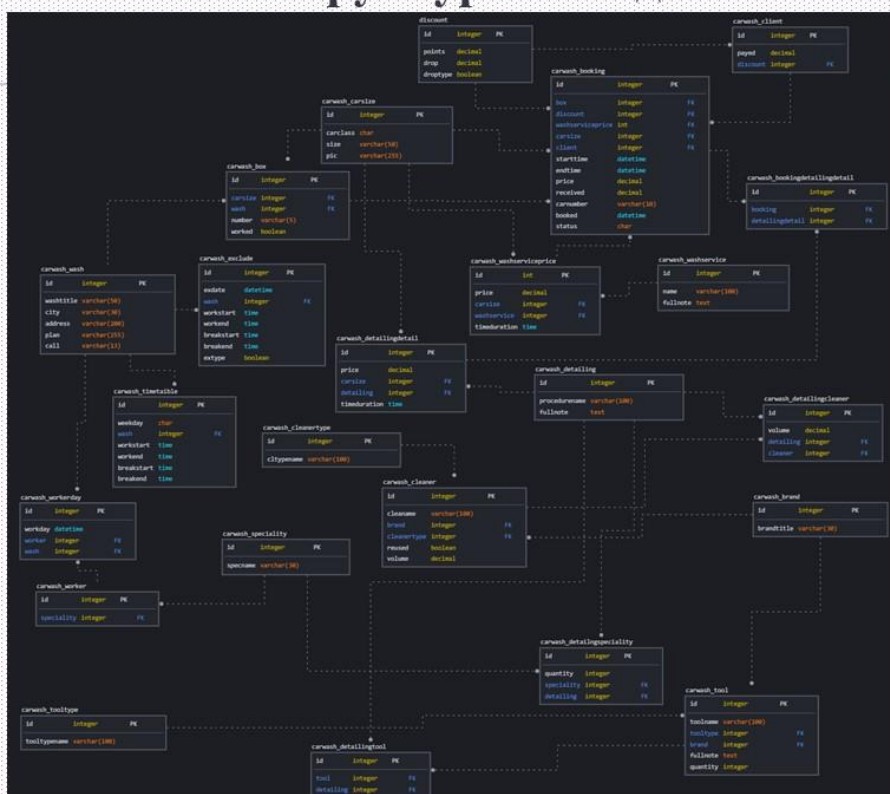
Вибір мови програмування для розробки програми

Показник	C#	Python
Відкрите і вільне програмне забезпечення	Так	Так
Порівнянні витрати розгортання	Більші	Менші
Високорівнева мова	Так	Так
Наявність потужного вебфреймворку	Так	Так
Охоплення бібліотеками	Добре	Потужне
Простота створення коду	Висока	Вища, за рахунок, наприклад, динамічної типізації

5

Рисунок В.5 – Слайд № 5

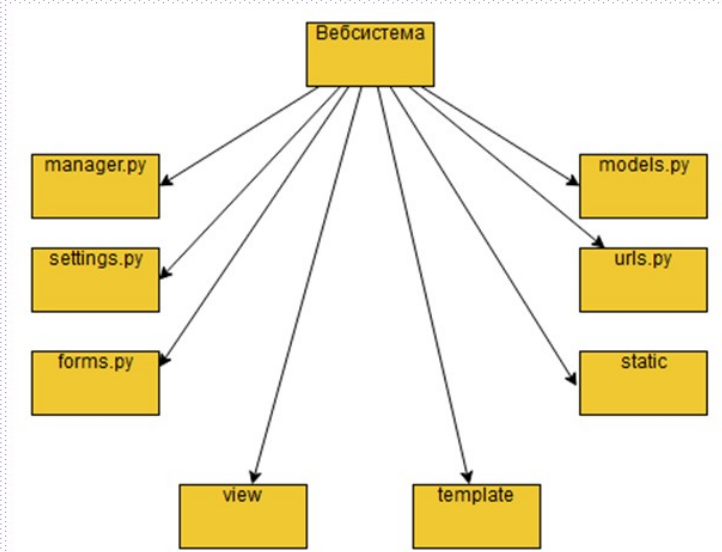
Схема структури бази даних



6

Рисунок В.6 – Слайд № 6

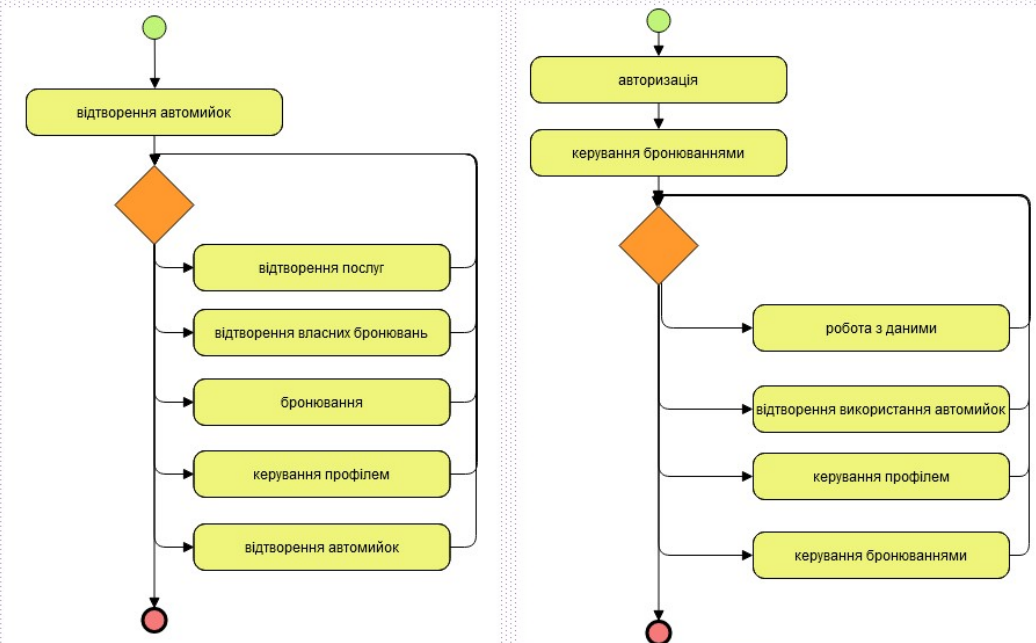
Схема структури вебсистеми



7

Рисунок В.7 – Слайд № 7

Схеми функціонування програми



8

Рисунок В.8 – Слайд № 8

Бронювання миття автомобіля в програмі

Нове бронювання послуг

1

2

3

4

Автомийка

Пакет миття

Детейлінг

Час

Клас автомобіля

Седан

Обрати

Вибір пакету миття

Базовий

80

Експрес-мийка

Обрати

Повний

150

Трьохфазна мийка

Миття вікон

Миття колес

Обрати

Преміум

600

Трьохфазна мийка

Миття вікон

Миття колес

Миття двигуна

Чистка салону

Обрати

9

Рисунок В.9 – Слайд № 9

Використання автомийки в програмі

Профіль | Вийти

МИЙКО

Бронювання Дані Використання

Статистика використання автомийки

Узгоджені дані про використання автомийки включають потенційну сумарну кількість годин за всіма боксами автомийки, протягом якої можна було б надавати послуги миття на автомийці, та фактично використану для цього кількість годин.

5

Боксів

38.5

Використано годин

70

Робочих годин

0.55

Частка використання

Автомийка

Номер 10 (Запоріжжя, Набережна магістраль)

Тип

День

Дата

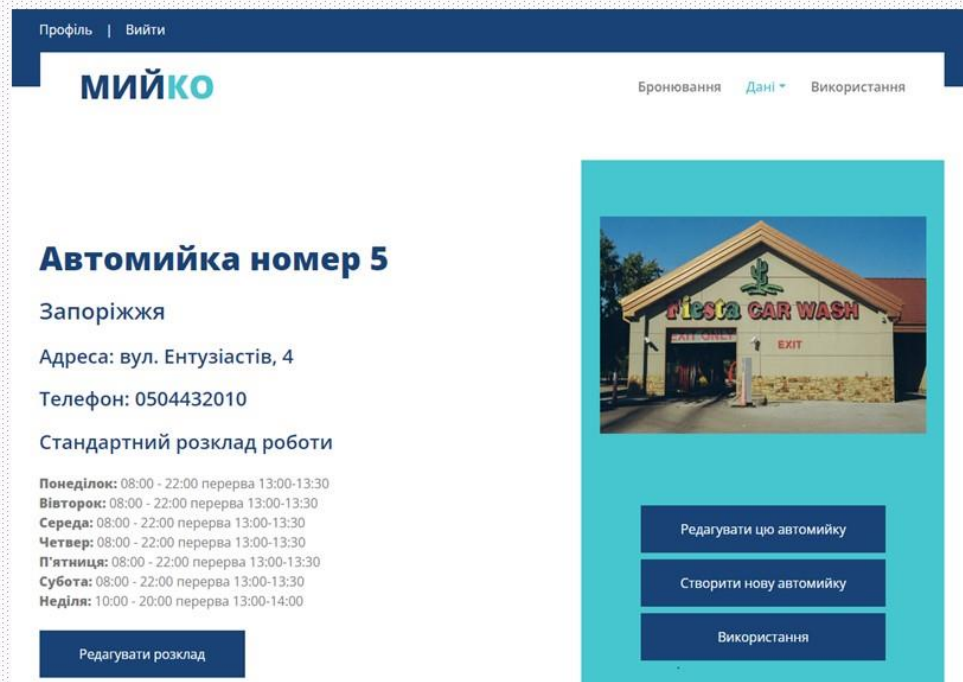
02.06.2022

Оновити

10

Рисунок В.10 – Слайд № 10

Робота з даними автомийок в програмі



11

Рисунок В.11 – Слайд № 11

Висновки

Розроблена вебсистема дозволяє відслідковувати і організовувати процес бронювання послуг з мийки автомобілів менеджерами автомийок, створювати єдині правила на різних автомийках мережі, зокрема пакети миття автомобілів, окремі послуги, їх вартість, вимоги до цих процедур, аналізувати завантаженість автомийки протягом дня або місяця, а також дозволяє віддалено бронювати мийку і додаткові процедури детейлінгу клієнтами.

Програма може використовувати для підтримки роботи невеликої мережі мийок автомобілів або окремої мийки автомобілів, що складається з декількох боксів, надаючи можливість керувати обслуговуванням одночасно великої кількості клієнтів.

12

Рисунок В.12 – Слайд № 12