ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Розробка програмного забезпечення для створення персоналізованого звукового оточення на основі бібліотеки звуків»

на здобуття освітнього ступеня бакалавра
зі спеціальності 121 Інженерія програмного забезпечення
освітньо-професійної програми «Інженерія програмного забезпечення»

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Владислав Чернодуб
(підпис)

Виконав: здобувач вищої освіти групи ПД-42
Владислав Чернодуб

Керівник: Владислав Яскевич
к.т.н., доц. каф. ІПЗ, доц.

Рецензент: _____

Київ 2025

ДЕРЖАВНИЙ УНІВЕРСИТЕТ

ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти Бакалавр

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ

Завідувач кафедри
Інженерії програмного забезпечення
_____Ірина ЗАМРІЙ
« ____» _____2025 р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

Чернодубу Владиславу Олеговичу

1. Тема кваліфікаційної роботи: «Веб-застосунок з управління автосервісом»

керівник кваліфікаційної роботи к.т.н., доц.каф. ІПЗ., доцент Владислав Яскевич, затверджені наказом Державного університету інформаційно-комунікаційних технологій від «24» лютого 2025 р. № 56.

2. Строк подання кваліфікаційної роботи «02» червня 2025 р.

3. Вихідні дані до кваліфікаційної роботи: теоретичні відомості про веб-застосунки для автоматизації бізнес-процесів у сфері технічного обслуговування транспорту, опис методів організації клієнтських записів, контролю завантаженості працівників і обліку замовлень, технічна документація PHP, MySQL, HTML та CSS для реалізації застосунку.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Проаналізувати існуючі рішення для автоматизації автосервісів і визначити їх переваги та недоліки.
2. Визначити функціональні та нефункціональні вимоги до застосунку.

3. Реалізувати основний функціонал веб-застосунку, включаючи запис клієнтів, облік робіт і управління співробітниками.

4. Провести загальне тестування системи з метою виявлення помилок і визначення відповідності додатку функціональних вимог.

5. Перелік графічного матеріалу: *презентація*

1. Аналіз аналогів.
2. Вимоги до програмного забезпечення.
3. Програмні засоби реалізації.
4. Архітектура застосунку.
5. Алгоритм роботи застосунку.
6. Екранні форми.

6. Дата видачі завдання «25» лютого 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Підбір та аналіз науково-технічної літератури	25.02.-04.03.2025	
2	Аналіз та дослідження існуючих аналогів	05.03-13.03.2025	
3	Огляд сучасних підходів до розробки веб-застосунків для бізнесу	14.03-20.03.2025	
4	Розробка інтерфейсу авторизації та основних сторінок (HTML/CSS)	21.03-27.03.2025	
5	Реалізація основної серверної логіки застосунку (PHP + MySQL)	28.03-17.04.2025	
6	Тестування веб-застосунку	18.04-28.05.2025	
7	Оформлення роботи: вступ, висновки, реферат	29.04-11.05.2025	
8	Розробка демонстраційних матеріалів	12.05-18.05.2025	
9	Попередній захист роботи	19.05-30.05.2025	

Здобувач вищої освіти _____ Владислав Чернодуб
(підпис)

Керівник
кваліфікаційної роботи _____ Владислав Яскевич
(підпис)

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 72 стор(від вступу до кінця висновка без презентації і коду).,15 рис., 29 джерел.

Мета роботи – підвищення ефективності роботи автосервісів за рахунок автоматизації бізнес-процесів у сферу обслуговування транспортних засобів.

Об'єкт дослідження – процеси автоматизації роботи автосервісу за допомогою веб-застосунку. Особлива увага приділяється розробці функціоналу для управління записом клієнтів, обліком виконаних робіт і взаємодії з працівниками сервісу.

Предмет дослідження – веб-застосунок призначений для управління процесами автосервісу, включаючи запис клієнтів, ведення історії обслуговування, контроль за виконанням ремонтних робіт та управління персоналом.

Короткий зміст роботи: У роботі розглянуто проблему автоматизації бізнес-процесів автосервісу з метою підвищення ефективності обслуговування клієнтів та оптимізації роботи персоналу. Проведено аналіз існуючих веб-рішень, визначено їхні сильні сторони та обмеження, а також сформульовано вимоги до функціоналу нового застосунку. Запропоновано архітектуру веб-застосунку, реалізовано прототип системи з інтерфейсом для керування записами, замовленнями, клієнтами та завантаженням працівників. Прототип створено з використанням PHP, MySQL, HTML і CSS. Результати тестування підтвердили стабільність роботи системи, її прикладну цінність і потенціал до впровадження в реальні умови діяльності автосервісу.

Сферою використання застосунку є автоматизація процесів обслуговування клієнтів в автосервісі, включаючи облік замовлень, запис на ремонт, контроль завантаженості майстрів, ведення історії обслуговування автомобілів та формування звітності для адміністрації.

КЛЮЧОВІ СЛОВА: АВТОСЕРВІС, ВЕБ-ЗАСТОСУНОК, PHP, MYSQL, HTML, CSS АВТОМАТИЗАЦІЯ, ОБЛІК КЛІЄНТІВ, ЗАПИС НА ОБСЛУГОВУВАННЯ, КЕРУВАННЯ ЗАМОВЛЕННЯМИ, СИСТЕМА УПРАВЛІННЯ.

ЗМІСТ

ВСТУП

Розвиток сучасних інформаційних технологій суттєво впливає на автоматизацію бізнес-процесів різних галузей, зокрема автосервісної сфери. Автосервіси сьогодні являють собою складні підприємства, які потребують ефективної системи управління всіма аспектами діяльності – від взаємодії з клієнтами до контролю виконання технічних робіт. За сучасних умов висококонкурентного ринку ефективне управління бізнес-процесами стає не просто перевагою, а необхідною умовою успішного функціонування автосервісу.

У цьому контексті розробка спеціалізованого веб-застосунку для управління автосервісом набуває особливого значення. Такий застосунок дозволяє автоматизувати ключові процеси діяльності – управління клієнтською базою, облік автомобілів, планування та контроль ремонтних робіт, розподіл завдань між персоналом, ведення складського обліку запчастин, формування різноманітної звітності. Використання веб-технологій забезпечує доступ до системи з будь-якого пристрою, підключеного до Інтернету, що особливо важливо в умовах сучасної мобільності бізнесу.

Автоматизація бізнес-процесів автосервісу дозволяє вирішити ряд важливих завдань. По-перше, це підвищення якості обслуговування клієнтів шляхом скорочення часу оформлення замовлень, уникнення помилок при розрахунках, забезпечення прозорості процесу ремонту. По-друге, це оптимізація використання ресурсів підприємства – часу персоналу, обладнання, запасів запчастин. По-третє, це накопичення структурованої інформації про діяльність автосервісу, що дозволяє проводити різноманітні аналітичні дослідження та приймати обґрунтовані управлінські рішення.

Об'єктом дослідження даної дипломної роботи є процес управління автосервісом, включаючи всі його складові: роботу з клієнтами, управління замовленнями на ремонт, планування робіт та розподіл ресурсів, контроль виконання робіт, облік запчастин, формування документації та звітності. Особлива увага приділяється аналізу специфіки діяльності автосервісних підприємств в Україні, їх потреб у автоматизації бізнес-процесів та вимог до інформаційних систем.

Предметом дослідження є методи, технології та інструменти розробки веб-застосунків для автоматизації бізнес-процесів автосервісу, зокрема технології PHP, MySQL, HTML, CSS, JavaScript, які використовуються для створення ефективних, масштабованих та безпечних веб-застосунків.

Метою роботи є розробка та впровадження веб-застосунку для управління автосервісом, який забезпечить автоматизацію основних бізнес-процесів, підвищить ефективність діяльності підприємства, покращить якість обслуговування клієнтів та надасть потужні інструменти для аналізу та контролю діяльності автосервісу.

Для досягнення поставленої мети в роботі вирішуються наступні завдання:

- Аналіз сучасного стану автосервісної галузі та виявлення потреб у автоматизації бізнес-процесів.
- Огляд існуючих рішень для управління автосервісом та визначення їх переваг і недоліків.
- Формування вимог до функціональності веб-застосунку для управління автосервісом.
- Проектування архітектури веб-застосунку, бази даних та користувацького інтерфейсу.
- Розробка серверної та клієнтської частини веб-застосунку, реалізація всіх необхідних функціональних модулів.
- Тестування розробленого застосунку, виявлення та усунення помилок, оптимізація роботи системи.
- Розробка рекомендацій щодо впровадження та використання веб-застосунку в реальних умовах роботи автосервісу.

Практична значимість роботи полягає в розробці готового до впровадження веб-застосунку, який відповідає потребам сучасних автосервісів в Україні та може бути адаптований до конкретних бізнес-процесів окремого підприємства.

Розроблений застосунок дозволить підвищити ефективність діяльності автосервісу, покращити якість обслуговування клієнтів, оптимізувати використання ресурсів та забезпечити повний контроль над усіма аспектами діяльності підприємства.

Робота складається з вступу, чотирьох розділів, висновків та списку використаних джерел.

У першому розділі розглядаються теоретичні основи розробки веб-застосунків для автосервісу.

Другий розділ присвячений проектуванню веб-застосунку, включаючи аналіз бізнес-процесів, формування функціональних вимог, проектування бази даних та інтерфейсу користувача.

Третій розділ описує процес реалізації веб-застосунку, включаючи вибір технологій, розробку бази даних та програмну реалізацію основних функціональних модулів.

Четвертий розділ присвячений тестуванню, впровадженню та використанню розробленого веб-застосунку.

РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ РОЗРОБКИ ВЕБ-ЗАСТОСУНКІВ ДЛЯ АВТОСЕРВІСУ

1.1. Аналіз сучасного стану автосервісної галузі

Сучасна автосервісна галузь демонструє стабільне зростання на ринку послуг з технічного обслуговування та ремонту автомобілів. Зокрема, в Україні спостерігається щорічне збільшення обсягів ринку автосервісних послуг на 5-7%, що пояснюється зростанням кількості автомобілів та збільшенням середнього віку автопарку. У таких умовах автосервіси прагнуть підвищити власну конкурентоспроможність шляхом вдосконалення процесів обслуговування клієнтів, впровадження сучасних технологій діагностики та ремонту, а також оптимізації внутрішніх бізнес-процесів.

Технологічний розвиток автомобільної галузі призводить до появи більш складних транспортних засобів, насичених електронними системами та інноваційними компонентами. Це створює додаткові вимоги до технічного оснащення автосервісів та кваліфікації персоналу. Сучасний автосервіс повинен мати спеціалізоване діагностичне обладнання для різних марок автомобілів, програмне забезпечення для роботи з бортовими комп'ютерами та кваліфікованих спеціалістів, здатних вирішувати складні технічні завдання [1]. Цифровізація галузі стає не просто трендом, а необхідною умовою виживання на ринку.

Поведінка споживачів автосервісних послуг також зазнає суттєвих змін. Понад 70% автовласників використовують інтернет для пошуку та вибору автосервісу, порівнюючи відгуки, ціни та спектр послуг. Близько 60% клієнтів хотіли б мати можливість онлайн-запису на обслуговування, відстеження статусу ремонту через мобільний додаток та отримання електронних документів. Такі очікування стимулюють автосервіси розвивати цифрові канали комунікації з клієнтами та впроваджувати інформаційні системи управління.

Економічні фактори, зокрема зростання цін на запчастини, енергоносії та оренду приміщень, спонукають автосервіси оптимізувати операційні процеси для збереження рентабельності. Ефективне управління запасами, контроль використання матеріалів, планування завантаженості персоналу стають ключовими напрямками

підвищення економічної ефективності [2]. В таких умовах автоматизація бізнес-процесів за допомогою спеціалізованого програмного забезпечення перетворюється з конкурентної переваги на необхідну умову успішної діяльності автосервісного підприємства.

1.2. Огляд існуючих рішень для управління автосервісом

На сучасному ринку програмного забезпечення представлено широкий спектр рішень для автоматизації діяльності автосервісів, які можна класифікувати за кількома категоріями. Комплексні ERP-системи, такі як "1С:Підприємство" з галузевими конфігураціями, SAP Business One та Microsoft Dynamics, пропонують функціональність для управління всіма аспектами бізнесу, включаючи облік запчастин, управління клієнтською базою, ведення фінансового обліку. Ці системи забезпечують високий рівень інтеграції між різними модулями та можливість отримання консолідованої звітності, однак вимагають значних інвестицій у придбання ліцензій, впровадження та навчання персоналу.

Спеціалізовані галузеві рішення, орієнтовані безпосередньо на автосервіси, включають такі продукти як "АвтоДилер", "АвтоПредприятие", "Мой Автосервис", "CarManager". Ці системи розроблені з урахуванням специфіки роботи автосервісів і пропонують функціонал для ведення клієнтської бази, управління замовленнями на ремонт, планування завантаження постів та обліку запчастин. Їх перевагою є оптимальне співвідношення ціни та функціональності, орієнтованої на потреби конкретно автосервісів, простота впровадження та використання, однак недоліком може бути обмежена інтеграція з іншими бізнес-системами [3].

Хмарні рішення за моделлю SaaS (Software as a Service), такі як "АвтоСервіс Онлайн", "GarageCloud", "AutoRepair Cloud", набувають все більшої популярності завдяки відсутності необхідності встановлення програмного забезпечення на локальні сервери та оплаті за фактичне використання. Користувач отримує доступ до системи через веб-браузер, а всі дані зберігаються на серверах провайдера послуг. Переваги таких рішень включають швидке впровадження, доступ з будь-якого пристрою, підключеного до інтернету, та автоматичні оновлення системи. Однак,

залежність від якості інтернет-з'єднання та питання безпеки даних можуть стати суттєвими недоліками.

Індивідуальні розробки під замовлення залишаються актуальним вибором для автосервісів зі специфічними вимогами до функціональності, які не можуть забезпечити готові рішення. Такі системи розробляються відповідно до конкретних потреб бізнесу, що дозволяє врахувати всі особливості робочих процесів та інтегрувати систему з існуючою IT-інфраструктурою підприємства. Перевагою індивідуальної розробки є максимальна відповідність потребам конкретного бізнесу, а недоліками – висока вартість розробки, тривалий період впровадження та залежність від розробника для подальшої підтримки та розвитку системи.

Аналіз функціональності існуючих рішень показує, що більшість систем включають модулі для управління клієнтською базою, облік замовлень на ремонт, управління складом запчастин та планування роботи персоналу. Більш просунуті системи також пропонують функціональність для інтеграції з діагностичним обладнанням, онлайн-запис клієнтів на обслуговування, мобільні додатки для клієнтів та співробітників, аналітичні інструменти для бізнес-аналізу [4]. Однак, часто спостерігається відсутність гнучкості у налаштуванні під конкретні потреби автосервісу, обмежена можливість інтеграції з іншими системами та недостатня адаптивність до змін у бізнес-процесах.

1.3. Вимоги до функціональності веб-застосунків для автосервісу

Функціональні вимоги до веб-застосунків для управління автосервісом повинні охоплювати всі ключові бізнес-процеси підприємства, починаючи від взаємодії з клієнтами і закінчуючи формуванням фінансової звітності. Ядром системи має бути модуль управління клієнтською базою, який забезпечує зберігання та управління інформацією про клієнтів та їх автомобілі, історії звернень, контактними даними. Функціональність цього модуля повинна включати додавання нових клієнтів, редагування інформації, пошук за різними параметрами (ім'я, телефон, номер автомобіля), сегментацію клієнтської бази за різними критеріями та ведення комунікацій з клієнтами через різні канали.

Управління замовленнями на ремонт є центральним процесом автосервісу, тому відповідний модуль має забезпечувати повний цикл обробки замовлення – від створення до закриття. Функціональність має включати: реєстрацію нового замовлення з вибором клієнта та автомобіля з бази, фіксацію початкового стану автомобіля, опис проблеми, планування діагностичних та ремонтних робіт, додавання результатів діагностики (з можливістю прикріплення фото/відео), формування переліку необхідних робіт та запчастин, відстеження статусу виконання, формування всіх необхідних документів (акти, наряди, рахунки), а також розрахунок вартості та прийом оплати [5].

Модуль планування робіт та розподілу ресурсів має забезпечувати оптимальне використання наявних ресурсів автосервісу – слюсарів, постів, обладнання. Функціональність повинна включати візуалізацію завантаженості у вигляді календаря чи діаграми Ганта, призначення виконавців з урахуванням їх спеціалізації та кваліфікації, планування часу виконання з урахуванням трудомісткості, автоматичне формування оптимального розкладу, можливість оперативного перепланування при виникненні непередбачених ситуацій, контроль фактичного часу виконання та аналіз ефективності використання ресурсів.

Управління запасами запчастин та матеріалів є важливим аспектом діяльності автосервісу, тому відповідний модуль має забезпечувати повний контроль над складськими процесами. Функціональність має включати ведення каталогу запчастин з детальним описом, облік наявності на складі, резервування під конкретні замовлення, контроль мінімальних залишків з автоматичним формуванням заявок на поповнення, облік руху запчастин, інтеграцію з каталогами постачальників для автоматичного оновлення асортименту та цін, формування заявок постачальникам та аналіз ефективності управління запасами.

Фінансовий облік та звітність є необхідними компонентами для ефективного управління бізнесом, тому веб-застосунок має включати відповідну функціональність. Вона повинна забезпечувати формування первинних фінансових документів, облік доходів та витрат, розрахунок собівартості послуг, аналіз рентабельності різних видів робіт, формування фінансової звітності за різні періоди, інтеграцію з бухгалтерськими системами, розрахунок заробітної плати та контроль виконання фінансових планів [6]. Система також має надавати інструменти для

аналітики, включаючи формування стандартних звітів, конструктор для створення нестандартних звітів, візуалізацію даних у вигляді графіків та діаграм.

1.4. Особливості розробки веб-застосунків для бізнес-процесів

Розробка веб-застосунків для автоматизації бізнес-процесів, зокрема для управління автосервісом, має ряд специфічних особливостей, які відрізняють її від створення інформаційних або розважальних веб-сайтів. Перш за все, така розробка вимагає глибокого розуміння предметної області та бізнес-процесів підприємства. Розробник повинен не лише володіти технічними навичками, але й розуміти специфіку роботи автосервісу, логіку взаємодії різних підрозділів, вимоги законодавства та галузеві стандарти. Це зумовлює необхідність тісної співпраці між розробниками та фахівцями галузі на всіх етапах створення веб-застосунку, починаючи від формування вимог і закінчуючи тестуванням та впровадженням.

Архітектура веб-застосунку для бізнес-процесів повинна забезпечувати високу надійність, продуктивність та безпеку даних. Типовим рішенням є багаторівнева архітектура, що включає рівень представлення (користувацький інтерфейс), рівень бізнес-логіки та рівень доступу до даних. Такий підхід дозволяє розділити відповідальність між різними компонентами системи, спростити тестування та подальший розвиток застосунку. Веб-застосунок для автосервісу повинен обробляти складні бізнес-правила, працювати з великими обсягами даних та забезпечувати одночасний доступ багатьох користувачів, що вимагає ретельного проектування архітектури системи з урахуванням масштабованості та продуктивності.

Вибір технологічного стеку для розробки веб-застосунку є важливим рішенням, яке впливає на весь процес розробки та подальшу експлуатацію системи. Для серверної частини (backend) найчастіше використовуються такі технології як PHP (з фреймворками Laravel, Symfony), Python (Django, Flask), JavaScript (Node.js, Express), Java (Spring), .NET. Вибір конкретної технології повинен враховувати її зрілість, наявність документації та спільноти розробників, можливості інтеграції з іншими системами. Для клієнтської частини (frontend) стандартом є використання HTML5, CSS3 та JavaScript, часто з застосуванням фреймворків та бібліотек (React,

Angular, Vue.js), що дозволяють створювати інтерактивні та відзивчиві інтерфейси, адаптовані для різних пристроїв [7].

Безпека даних є критично важливим аспектом розробки веб-застосунків для бізнес-процесів, особливо коли мова йде про системи, що обробляють персональні дані клієнтів та фінансову інформацію. Веб-застосунок повинен забезпечувати надійну аутентифікацію та авторизацію користувачів, шифрування даних при передачі та зберіганні, захист від поширених вразливостей веб-застосунків (SQL-ін'єкції, XSS, CSRF). Розробники повинні слідувати принципу "безпека за замовчуванням", впроваджуючи механізми захисту на всіх рівнях застосунку та проводячи регулярне тестування безпеки. Також важливо забезпечити відповідність системи вимогам законодавства щодо захисту персональних даних, зокрема GDPR в європейському контексті.

Інтерфейс користувача є ключовим елементом веб-застосунку, який безпосередньо впливає на ефективність роботи персоналу. При розробці інтерфейсу для автосервісу необхідно враховувати особливості роботи різних категорій персоналу (адміністратори, менеджери, слюсарі) та умови використання системи (офіс, ремонтна зона). Інтерфейс повинен бути інтуїтивно зрозумілим, забезпечувати швидкий доступ до найбільш часто використовуваних функцій, мінімізувати кількість дій для виконання типових операцій. Адаптивний дизайн повинен забезпечувати комфортну роботу як на стаціонарних комп'ютерах, так і на мобільних пристроях, що особливо актуально для технічного персоналу, який працює в ремонтній зоні.

Впровадження та супровід веб-застосунку є не менш важливими етапами, ніж розробка. Для успішного впровадження необхідно забезпечити міграцію даних з існуючих систем, навчання персоналу, налаштування інтеграцій з іншими системами (бухгалтерія, CRM, IP-телефонія). Важливо розробити детальну документацію, яка включає інструкції для користувачів різних ролей, опис архітектури системи, API для інтеграцій, інструкції з налаштування та оновлення [8]. Супровід передбачає не лише виправлення помилок, але й розвиток системи відповідно до змін у бізнес-процесах, законодавстві та технологіях, що вимагає планування ресурсів на постійну роботу з системою після впровадження.

РОЗДІЛ 2. ПРОЕКТУВАННЯ ВЕБ-ЗАСТОСУНКУ З УПРАВЛІННЯ АВТОСЕРВІСОМ

2.1. Аналіз бізнес-процесів автосервісу

Бізнес-процеси сучасного автосервісу представляють собою складну систему взаємопов'язаних операцій, спрямованих на надання послуг з технічного обслуговування та ремонту автомобілів. Ефективне управління цими процесами є ключовим фактором успішної діяльності підприємства в умовах високої конкуренції на ринку. Аналіз бізнес-процесів дозволяє визначити основні операційні потоки, визначити учасників та відповідальних осіб, встановити послідовність дій та взаємозв'язки між ними, що є необхідною умовою для створення ефективної системи автоматизації.

Процес обслуговування клієнта в автосервісі починається з первинного звернення, яке може відбуватися різними каналами: телефонним дзвінком, особистим візитом, через соціальні мережі або сайт компанії. На цьому етапі відбувається первинна комунікація з клієнтом, збір інформації про автомобіль та проблему, що виникла, консультування щодо можливих причин несправності та орієнтовної вартості ремонту [9]. Ключовими аспектами цього процесу є швидкість реагування на звернення, професіоналізм консультації та зручність запису на обслуговування. Розроблювана система повинна надавати інструменти для фіксації всіх звернень, організації черги на обслуговування та автоматичного інформування клієнтів про статус їх запиту.

Прийом автомобіля на діагностику або ремонт є наступним важливим бізнес-процесом. При цьому оформляється заявка на обслуговування, фіксується стан автомобіля при прийомі (наявні пошкодження, рівень палива, пробіг), уточнюються скарги клієнта та обсяг робіт. На практиці цей процес часто супроводжується значним обсягом паперової документації та ручного введення даних, що збільшує ймовірність помилок та знижує ефективність роботи персоналу. Веб-застосунок повинен автоматизувати цей процес, забезпечуючи швидке створення замовлення на ремонт, фіксацію всіх необхідних даних, формування акту

прийому-передачі автомобіля та інших документів, а також інтеграцію з клієнтською базою для швидкого доступу до історії обслуговування конкретного автомобіля.

Діагностика та виконання ремонтних робіт є ключовими технічними процесами, що визначають подальший хід ремонту. В залежності від типу автомобіля та характеру проблеми, може проводитися комп'ютерна діагностика з використанням спеціалізованого обладнання, механічна перевірка вузлів та агрегатів, тестові заїзди [10]. Результати діагностики визначають необхідні ремонтні роботи, запчастини та матеріали, що будуть використані. Система повинна забезпечувати можливість детального документування результатів діагностики, прикріплення фото та відеоматеріалів, формування технічних звітів, а також зв'язок з каталогами запчастин для швидкого підбору необхідних компонентів.

Планування та розподіл робіт є складним логістичним процесом, що вимагає врахування багатьох факторів: кваліфікації персоналу, завантаженості ремонтних постів, наявності необхідних запчастин та інструментів, пріоритетності замовлень. Неефективне планування призводить до простоїв обладнання та персоналу, затримок у виконанні робіт, зниження рівня задоволеності клієнтів [11]. Веб-застосунок повинен надавати засоби для візуалізації завантаженості ресурсів, автоматизованого планування робіт з урахуванням усіх обмежень, оперативного перепланування у разі виникнення непередбачених ситуацій, а також контролю виконання плану.

2.2. Функціональні вимоги до застосунку

Функціональні вимоги до веб-застосунку з управління автосервісом формуються на основі аналізу бізнес-процесів та потреб різних категорій користувачів системи. Ключовою вимогою є забезпечення комплексної автоматизації всього циклу обслуговування клієнтів — від первинного звернення до післяпродажного супроводу. Система повинна надавати зручні інструменти для управління клієнтською базою, замовленнями на ремонт, ресурсами автосервісу (персонал, обладнання, запчастини), а також забезпечувати формування аналітичної інформації для прийняття управлінських рішень. При цьому важливо забезпечити інтуїтивно зрозумілий інтерфейс, що відповідає рівню комп'ютерної грамотності персоналу автосервісу.

Модуль управління клієнтською базою повинен забезпечувати зберігання та управління інформацією про клієнтів та їх автомобілі. Функціональні вимоги до цього модуля включають: реєстрацію нових клієнтів з введенням контактної інформації (ПІБ, телефон, email); додавання та редагування інформації про автомобілі клієнта (марка, модель, рік випуску, VIN-код, державний номер, технічні характеристики); пошук клієнтів за різними параметрами; перегляд історії обслуговування кожного автомобіля; сегментацію клієнтської бази за різними критеріями; формування нагадувань про планове технічне обслуговування; експорт та імпорт даних у стандартних форматах. Особлива увага має бути приділена захисту персональних даних клієнтів відповідно до вимог законодавства [12].

Модуль управління замовленнями на ремонт повинен забезпечувати повний цикл обробки звернень клієнтів — від створення заявки до закриття замовлення. Функціональні вимоги до цього модуля включають: створення нового замовлення з вибором клієнта та автомобіля з бази або додаванням нового; фіксацію початкового стану автомобіля; опис скарг клієнта та симптомів несправності; планування діагностичних та ремонтних робіт; додавання результатів діагностики, включаючи можливість прикріплення фото та відеоматеріалів; формування переліку необхідних робіт та запчастин з розрахунком вартості; відстеження статусу виконання замовлення; формування та друк всіх необхідних документів; фіксацію фактично використаних запчастин та витратних матеріалів; розрахунок вартості та прийом оплати.

Модуль планування та розподілу робіт повинен забезпечувати оптимальне використання ресурсів автосервісу та контроль виконання робіт. Функціональні вимоги до цього модуля включають: візуалізацію завантаженості ремонтних постів та персоналу у вигляді календаря або діаграми Ганта; призначення конкретних виконавців на кожне замовлення з урахуванням їх спеціалізації та кваліфікації; планування часу виконання робіт з урахуванням їх трудомісткості та доступності ресурсів; автоматичне формування оптимального розкладу робіт на день/тиждень; перепланування у разі виникнення непередбачених ситуацій; контроль фактичного часу виконання робіт у порівнянні з нормативним; формування звітів про завантаженість персоналу та ефективність використання робочого часу.

Модуль управління запчастинами та матеріалами повинен забезпечувати контроль запасів та оптимізацію закупівель. Функціональні вимоги до цього модуля включають: ведення каталогу запчастин та матеріалів з детальним описом; облік наявності запчастин на складі; резервування запчастин під конкретні замовлення; контроль мінімальних залишків та автоматичне формування заявок на поповнення; облік руху запчастин; інтеграцію з каталогами постачальників для автоматичного оновлення асортименту та цін; формування заявок постачальникам та контроль їх виконання; формування звітів з аналізу оборотності запасів для оптимізації закупівельної політики [13].

2.3. Проектування бази даних

2.3.1. Концептуальна модель даних

Концептуальна модель даних є першим етапом проектування бази даних веб-застосунку з управління автосервісом і відображає основні інформаційні сутності системи та зв'язки між ними на рівні предметної області, без урахування особливостей конкретної СУБД. При розробці концептуальної моделі для автосервісу було враховано специфіку бізнес-процесів галузі, вимоги до функціональності системи та перспективи її розвитку. Модель створювалась з метою забезпечення повноти відображення всіх аспектів діяльності автосервісу, підтримки всіх необхідних бізнес-операцій, а також забезпечення цілісності та несуперечливості даних.

Ключовими сутностями концептуальної моделі даних для веб-застосунку з управління автосервісом є: «Клієнти», «Автомобілі», «Замовлення на ремонт», «Роботи», «Запчастини», «Слюсарі», «Діагностика». Сутність «Клієнти» містить інформацію про всіх клієнтів автосервісу і є однією з центральних у моделі, оскільки саме клієнти є замовниками послуг. Сутність «Автомобілі» зберігає дані про транспортні засоби, що обслуговуються в автосервісі, і пов'язана з сутністю «Клієнти» зв'язком типу «один-до-багатьох», оскільки один клієнт може мати декілька автомобілів, але кожен автомобіль належить лише одному клієнту.

Сутність «Замовлення на ремонт» є центральною в моделі і відображає факт звернення клієнта до автосервісу для проведення певних робіт з його автомобілем. Ця сутність пов'язана з сутностями «Клієнти» та «Автомобілі» зв'язками типу «багато-до-одного», оскільки один клієнт і один автомобіль можуть мати багато замовлень на ремонт протягом часу. Кожне замовлення має унікальний ідентифікатор, дату створення, опис проблеми, статус виконання та інші атрибути, необхідні для контролю процесу обслуговування. У контексті предметної області, замовлення на ремонт є документом, що ініціює процес обслуговування клієнта та фіксує всі деталі взаємодії з ним.

Сутність «Роботи» представляє конкретні технічні операції, що виконуються в рамках замовлення на ремонт. Ця сутність пов'язана із сутністю «Замовлення на ремонт» зв'язком типу «багато-до-одного», оскільки одне замовлення може включати декілька різних робіт. Кожна робота характеризується типом (діагностика, ремонт, технічне обслуговування), описом, нормативним часом виконання, вартістю, фактичним часом виконання. Для забезпечення можливості аналізу та формування статистики, роботи можуть бути класифіковані за різними категоріями, наприклад, «двигун», «трансмісія», «ходова частина», «електрообладнання», що відображається відповідним атрибутом [14].

Сутність «Запчастини» містить інформацію про всі деталі та матеріали, що використовуються при обслуговуванні автомобілів. Ця сутність пов'язана з сутністю «Роботи» через проміжну сутність «Використані запчастини», що реалізує зв'язок типу «багато-до-багатьох», оскільки одна запчастина може використовуватися в багатьох роботах, а одна робота може вимагати декількох запчастин. Кожна запчастина характеризується найменуванням, артикулом, виробником, ціною, кількістю на складі, мінімальним залишком для автоматичного замовлення.

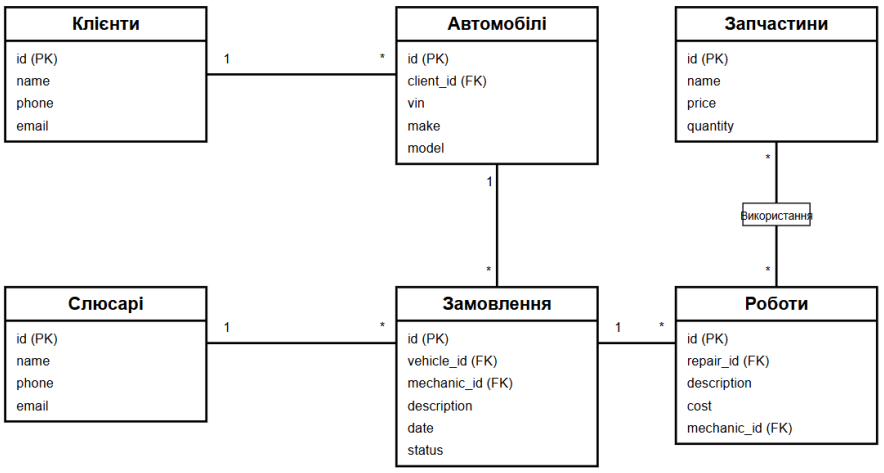


Рис. 2.1 - Концептуальна модель даних автосервісу

Також для запчастин може бути вказана сумісність з певними марками та моделями автомобілів, що дозволяє системі автоматично підбирати підходящі деталі для конкретного ремонту.

2.3.2. Логічна модель даних

Логічна модель даних для веб-застосунку з управління автосервісом є наступним кроком після концептуальної моделі і представляє структуру даних з урахуванням особливостей конкретної системи управління базами даних, але без деталей фізичної реалізації. На цьому етапі визначаються таблиці, їх атрибути, типи даних, обмеження цілісності та зв'язки між таблицями. Логічна модель розроблена з урахуванням особливостей реляційних баз даних, зокрема MySQL/MariaDB, яка використовується у веб-застосунку, і спрямована на забезпечення ефективного зберігання та обробки даних при збереженні їх логічної структури, визначеної на концептуальному рівні.

Центральною таблицею логічної моделі є «clients» (клієнти), яка містить наступні атрибути: id (первинний ключ, унікальний ідентифікатор клієнта), name (ім'я клієнта), phone (номер телефону), email (електронна пошта). Таблиця «clients» пов'язана з таблицею «vehicles» (автомобілі) зв'язком «один-до-багатьох», який реалізується через зовнішній ключ client_id в таблиці «vehicles». Таблиця «vehicles» містить атрибути: id (первинний ключ), client_id (зовнішній ключ, що вказує на клієнта-власника), vin (VIN-код автомобіля), make (марка), model (модель), year (рік

випуску). Таке розділення даних про клієнтів та їх автомобілі відповідає принципам нормалізації бази даних і дозволяє ефективно зберігати інформацію про клієнтів, які мають декілька автомобілів.

Таблиця «repairs» (ремонти) є ключовою для відображення бізнес-процесів автосервісу і містить наступні атрибути: id (первинний ключ), vehicle_id (зовнішній ключ, що вказує на автомобіль), description (опис проблеми або необхідних робіт), parts_needed (необхідні запчастини), date (дата створення або планована дата ремонту), mechanic_id (зовнішній ключ, що вказує на призначеного слюсаря), status (статус ремонту, який може приймати значення «Очікує», «В процесі», «Завершено»), client_id (зовнішній ключ, що дублює зв'язок з клієнтом для оптимізації запитів). Зв'язок таблиці «repairs» з таблицею «vehicles» типу «багато-до-одного» реалізується через зовнішній ключ vehicle_id, що дозволяє зберігати історію всіх ремонтів для кожного автомобіля.

Таблиця «mechanics» (слюсарі) містить інформацію про працівників автосервісу і має атрибути: id (первинний ключ), name (ім'я слюсаря), phone (телефон), email (електронна пошта). Ця таблиця пов'язана з таблицею «repairs» зв'язком «один-до-багатьох», який реалізується через зовнішній ключ mechanic_id в таблиці «repairs». Такий зв'язок дозволяє призначати конкретного слюсаря для виконання кожного ремонту і аналізувати завантаженість та ефективність роботи персоналу. У більш складній моделі може бути передбачена додаткова таблиця «specializations» (спеціалізації) і таблиця зв'язку «mechanic_specializations» для реалізації відношення «багато-до-багатьох» між слюсарями та їх спеціалізаціями [15].

Для детального обліку виконаних робіт логічна модель розширена таблицею «repair_works» (ремонтні роботи), яка містить атрибути: id (первинний ключ), repair_id (зовнішній ключ, що вказує на замовлення ремонту), work_type (тип роботи), description (опис), normative_time (нормативний час виконання), actual_time (фактичний час виконання), cost (вартість), mechanic_id (зовнішній ключ, що вказує на виконавця роботи), completion_date (дата завершення).

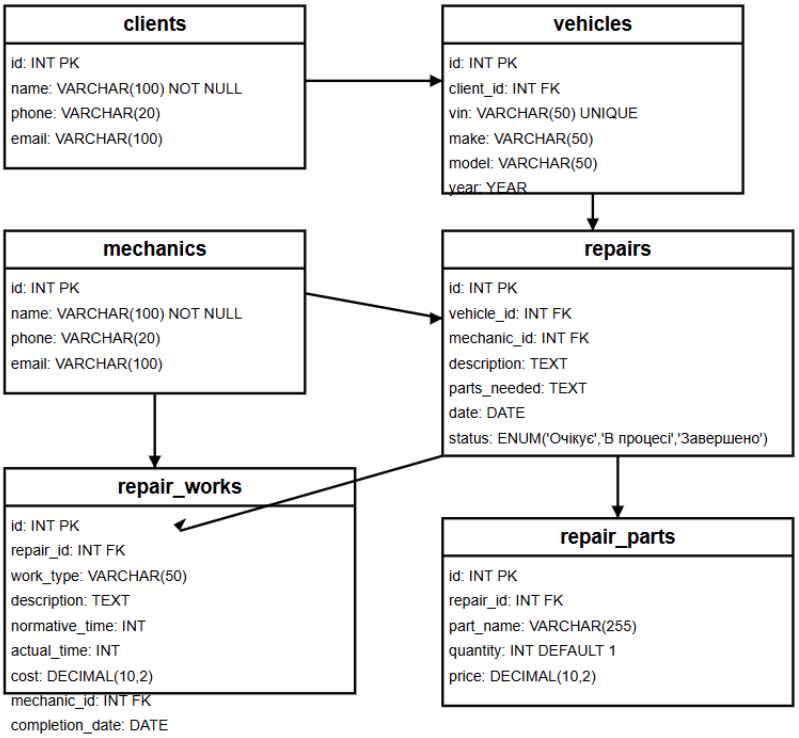


Рис. 2.2 - Логічна модель даних автосервісу

Ця таблиця дозволяє вести детальний облік кожної технічної операції в рамках загального замовлення на ремонт, що важливо для розрахунку вартості послуг, аналізу ефективності роботи персоналу та формування звітності.

2.3.3. Фізична модель даних

Фізична модель даних для веб-застосунку з управління автосервісом є деталізацією логічної моделі з урахуванням особливостей конкретної системи управління базами даних MySQL/MariaDB, і включає точні специфікації таблиць, полів, типів даних, індексів, обмежень та інших фізичних аспектів зберігання даних. Фізична модель розроблена з метою забезпечення оптимальної продуктивності, цілісності, безпеки та масштабованості бази даних. При проектуванні фізичної моделі особлива увага приділялась оптимізації структури для забезпечення швидкого виконання типових запитів, характерних для автосервісного бізнесу, таких як пошук клієнтів та їх автомобілів, формування звітів про виконані роботи, аналіз завантаженості персоналу.

Для таблиці "clients" (клієнти) у фізичній моделі визначено наступні характеристики: первинний ключ id є цілим числом з автоінкрементом, що

забезпечує автоматичне генерування унікальних ідентифікаторів для нових клієнтів. Поля name, phone та email мають тип VARCHAR з обмеженнями на максимальну довжину (100, 20 та 100 символів відповідно), що оптимізує використання дискового простору. Створено індекси для полів phone та email, що прискорює пошук клієнтів за цими атрибутами. Використовується кодування utf8mb4, що забезпечує підтримку Unicode символів, включаючи емодзі, що важливо для коректного відображення різних мов та спеціальних символів. Таблиця використовує движок InnoDB, який підтримує транзакції та зовнішні ключі, що важливо для забезпечення цілісності даних.

Для таблиці "vehicles" (автомобілі) у фізичній моделі визначено наступні характеристики: первинний ключ id є цілим числом з автоінкрементом. Зовнішній ключ client_id є цілим числом, що посилається на поле id таблиці "clients", з обмеженням ON DELETE CASCADE, яке забезпечує автоматичне видалення всіх автомобілів клієнта при видаленні самого клієнта. Поле vin має тип VARCHAR(50) з обмеженням UNIQUE, що гарантує унікальність VIN-кодів автомобілів у базі даних. Поля make та model мають тип VARCHAR(50), що достатньо для зберігання найменувань марок та моделей автомобілів. Поле year має спеціальний тип YEAR, оптимізований для зберігання років. Створено складений індекс по полях make і model для прискорення пошуку автомобілів за маркою та моделлю.

Для таблиці "mechanics" (слюсарі) у фізичній моделі визначено наступні характеристики: первинний ключ id є цілим числом з автоінкрементом. Поля name, phone та email мають тип VARCHAR з обмеженнями на максимальну довжину, аналогічно таблиці "clients". Створено індекс по полю name для прискорення пошуку слюсарів за іменем. Індекси по полях phone та email не створюються, оскільки пошук слюсарів за цими атрибутами не є типовою операцією. Такий підхід дозволяє оптимізувати використання ресурсів бази даних, створюючи індекси лише для полів, що активно використовуються в запитах.

Для таблиці "repairs" (ремонти) у фізичній моделі визначено наступні характеристики: первинний ключ id є цілим числом з автоінкрементом. Зовнішні ключі vehicle_id, mechanic_id та client_id є цілими числами, що посилаються на відповідні таблиці. Для поля vehicle_id використовується обмеження ON DELETE CASCADE, яке забезпечує автоматичне видалення всіх ремонтів при видаленні

автомобіля. Для поля mechanic_id використовується обмеження ON DELETE SET NULL, яке встановлює значення NULL для поля mechanic_id при видаленні слюсаря, що дозволяє зберегти історію ремонтів навіть після звільнення працівника. Поля description та parts_needed мають тип TEXT, що дозволяє зберігати великі обсяги тексту. Поле status використовує тип ENUM, що обмежує можливі значення статусу ремонту трьома варіантами: "Очікує", "В процесі", "Завершено".

Для зберігання деталей про використані запчастини у фізичній моделі визначена таблиця "repair_parts" (запчастини для ремонту) з наступними характеристиками: первинний ключ id є цілим числом з автоінкрементом. Зовнішній ключ repair_id є цілим числом, що посилається на поле id таблиці "repairs", з обмеженням ON DELETE CASCADE, яке забезпечує автоматичне видалення всіх записів про використані запчастини при видаленні відповідного ремонту. Поле part_name має тип VARCHAR(255), що достатньо для зберігання найменувань запчастин. Поле quantity має тип INT з значенням за замовчуванням 1, що відображає кількість використаних запчастин. Поле price має тип DECIMAL(10,2), що забезпечує точне зберігання грошових значень з двома знаками після коми [16].

Для зберігання інформації про користувачів системи у фізичній моделі визначена таблиця "users" (користувачі) з наступними характеристиками: первинний ключ id є цілим числом з автоінкрементом. Поле username має тип VARCHAR(50) з обмеженням UNIQUE, що гарантує унікальність логінів користувачів у системі. Поле password має тип VARCHAR(255), що достатньо для зберігання хешів паролів, створених сучасними алгоритмами хешування, такими як bcrypt або Argon2. Поле role використовує тип ENUM, що обмежує можливі ролі користувачів чотирма варіантами: "admin", "manager", "mechanic", "client". Поле last_login має тип DATETIME, що дозволяє зберігати дату та час останнього входу користувача в систему. Поле is_active має тип TINYINT(1), еквівалентний булевому типу в MySQL.

Для зберігання фінансової інформації у фізичній моделі визначені таблиці "invoices" (рахунки) та "payments" (оплати). Таблиця "invoices" має наступні характеристики: первинний ключ id є цілим числом з автоінкрементом. Зовнішній ключ repair_id є цілим числом, що посилається на поле id таблиці "repairs", з обмеженням ON DELETE CASCADE. Поле amount має тип DECIMAL(10,2) для точного зберігання грошових значень. Поле status використовує тип ENUM з

можливими значеннями "виставлений", "частково оплачений", "повністю оплачений". Поля issue_date та due_date мають тип DATE для зберігання дат виставлення та очікуваної оплати рахунку.

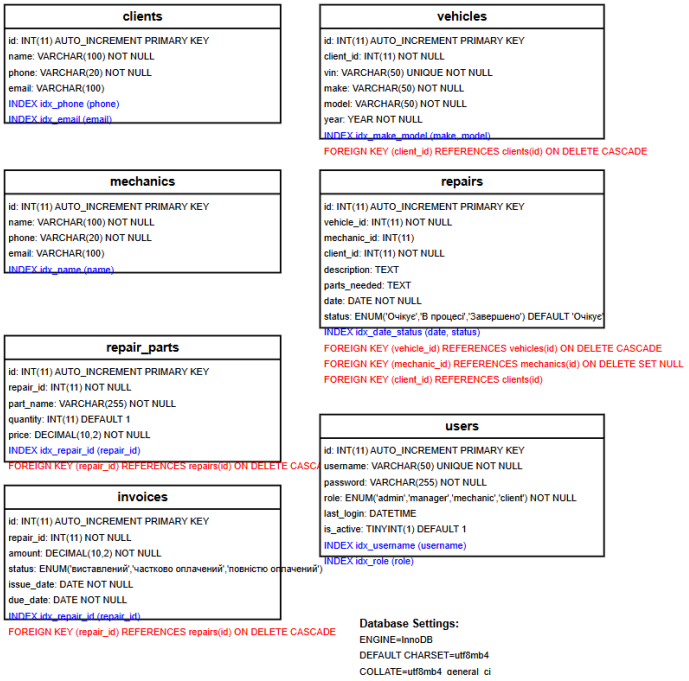


Рис. 2.3 - Фізична модель даних автосервісу

Для оптимізації продуктивності бази даних у фізичній моделі створюються індекси для полів, що часто використовуються у пошукових запитах та для з'єднання таблиць. Наприклад, у таблиці "repairs" створюється складений індекс по полях date та status для прискорення вибірки ремонтів за певний період з певним статусом. У таблиці "repair_parts" створюється індекс по полю repair_id для швидкого отримання всіх запчастин, використаних у конкретному ремонті. Створення оптимальних індексів є важливим аспектом фізичного проектування, оскільки індекси прискорюють виконання запитів, але збільшують розмір бази даних та сповільнюють операції вставки та оновлення.

Для забезпечення цілісності даних у фізичній моделі використовуються обмеження цілісності, такі як PRIMARY KEY, FOREIGN KEY, UNIQUE, NOT NULL. Наприклад, первинні ключі забезпечують унікальність записів у кожній таблиці, зовнішні ключі гарантують, що записи, які посилаються на інші таблиці, посилаються на існуючі записи, обмеження UNIQUE запобігають дублюванню важливих даних, а обмеження NOT NULL забезпечують наявність значень у

обов'язкових полях. Використання цих обмежень дозволяє перенести частину логіки валідації даних з програмного коду на рівень бази даних, що підвищує надійність системи та зменшує ймовірність помилок.

Важливим аспектом фізичної моделі є визначення стратегії резервного копіювання та відновлення даних. Для веб-застосунку з управління автосервісом рекомендується використовувати щоденне повне резервне копіювання бази даних та інкрементальне копіювання кожні кілька годин (див. рисунок 2.1).

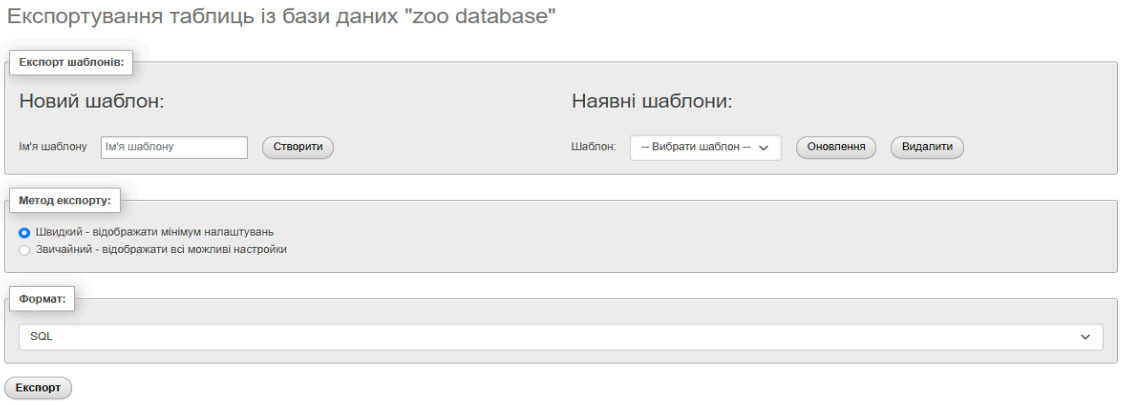


Рис. 2.1 – Вікно експорту бази даних у форматі SQL через phpMyAdmin

Це дозволяє забезпечити мінімальні втрати даних у разі збоїв системи та швидке відновлення роботи автосервісу. Розроблена фізична модель передбачає можливість легкого створення резервних копій завдяки стандартним інструментам MySQL/MariaDB, таким як mysqldump. Також модель підтримує можливість реплікації даних між кількома серверами, що може бути корисним для великих автосервісів з розподіленою інфраструктурою.

У фізичній моделі також визначаються параметри налаштування сервера бази даних для оптимальної продуктивності. Зокрема, налаштовуються розміри буферів, кеші запитів, параметри з'єднань, таймаути та інші параметри, що впливають на швидкодію та стабільність роботи бази даних. Наприклад, для таблиць, що часто оновлюються, таких як "repairs" та "repair_parts", рекомендується використовувати більші розміри буферів для зменшення навантаження на дисковий ввід-вивід. Також налаштовуються параметри журналювання транзакцій для забезпечення цілісності даних навіть у разі раптового відключення електроживлення або збою операційної системи.

Остаточна фізична модель даних для веб-застосунку з управління автосервісом є результатом ретельного аналізу бізнес-вимог, особливостей предметної області та технічних можливостей обраної системи управління базами даних. Модель забезпечує ефективне зберігання та обробку даних, підтримку всіх необхідних бізнес-процесів автосервісу, захист інформації від несанкціонованого доступу та втрати, а також можливість подальшого розвитку та масштабування системи відповідно до зростання бізнесу. Правильно спроектована фізична модель є основою для успішної реалізації веб-застосунку та його ефективного використання в реальних умовах роботи автосервісу.

2.4. Проектування інтерфейсу користувача

2.4.1. Прототипи інтерфейсу

Проектування інтерфейсу користувача для веб-застосунку з управління автосервісом починається з розробки прототипів, які дозволяють візуалізувати структуру та функціональність системи, а також протестувати взаємодію користувачів з нею ще до початку повноцінної розробки. Прототипи розроблялися з урахуванням принципів зручності використання (usability), доступності (accessibility) та адаптивності (responsiveness), щоб забезпечити комфортну роботу з системою для всіх категорій користувачів на різних пристроях.

Першим етапом розробки прототипів було створення низькорівневих ескізів (wireframes), які відображають базову структуру сторінок, розміщення основних елементів інтерфейсу та навігаційних компонентів. Для головної сторінки веб-застосунку був розроблений ескіз, що включає заголовок з логотипом та назвою автосервісу, навігаційне меню з основними розділами системи, інформаційні блоки з ключовими показниками (кількість активних замовлень, клієнтів на черзі, завершених робіт за день) та швидкі посилання на найчастіше використовувані функції (додати клієнта, створити замовлення, переглянути розклад). Ескізи розроблялися з використанням спеціалізованого програмного забезпечення для прототипування Figma, яке дозволяє швидко створювати та редагувати макети сторінок.

Для основних функціональних сторінок системи, таких як сторінка управління клієнтами, сторінка управління замовленнями на ремонт, сторінка планування розкладу робіт, були розроблені детальні прототипи, які включають всі необхідні елементи управління та відображення даних. Наприклад, для сторінки управління замовленнями прототип включає фільтри для пошуку замовлень за різними параметрами (дата, статус, клієнт, автомобіль), таблицю з списком замовлень, яка відображає ключову інформацію про кожне замовлення (номер, дата, клієнт, автомобіль, статус), та панель інструментів для виконання дій з замовленнями (перегляд деталей, редагування, зміна статусу, друк документів). Особлива увага приділялася розробці форм введення даних, таких як форма створення нового клієнта, форма створення замовлення на ремонт, форма планування робіт, які повинні бути інтуїтивно зрозумілими та зручними у використанні.

Для мобільної версії веб-застосунку були розроблені окремі прототипи, які враховують особливості взаємодії на пристроях з сенсорними екранами та меншими розмірами дисплеїв. В цих прототипах було передбачено адаптивне меню, яке згортається на малих екранах у «бургер-меню», збільшені розміри елементів управління для зручності натискання пальцем, спрощені версії таблиць з можливістю горизонтального скролінгу або відображення у форматі карток. Також були розроблені прототипи для специфічних мобільних функцій, таких як сканування штрих-кодів запчастин, фотографування пошкоджень автомобіля, використання GPS для визначення місцезнаходження при виїзному обслуговуванні.

Після доопрацювання низькорівневих прототипів були створені високорівневі макети (mockups), які включають не лише структурні елементи, але й візуальний дизайн: кольорову гаму, шрифти, іконки, елементи оформлення. Високорівневі макети надають більш повне уявлення про те, як буде виглядати готовий інтерфейс, і дозволяють оцінити естетичні аспекти дизайну [17]. Для створення високорівневих макетів також використовувався інструмент Figma, що забезпечує широкі можливості для роботи з візуальним дизайном, включаючи бібліотеки компонентів, стилів та ефектів.

2.4.2. Розробка дизайну інтерфейсу

Розробка дизайну інтерфейсу веб-застосунку з управління автосервісом є важливим етапом проектування, що безпосередньо впливає на зручність використання системи та ефективність роботи персоналу. На основі раніше розроблених прототипів був створений повноцінний дизайн користувацького інтерфейсу, який поєднує функціональність, естетичну привабливість та ергономічність. При розробці дизайну були враховані вимоги всіх категорій користувачів: адміністраторів, менеджерів, слюсарів та клієнтів, а також особливості роботи в умовах автосервісу, де важливою є швидкість доступу до інформації та простота виконання типових операцій.

Кольорова схема веб-застосунку була розроблена з урахуванням психології кольорів та їх впливу на сприйняття інформації. Основними кольорами були обрані синій та білий, які асоціюються з надійністю, професіоналізмом та чистотою. Акцентними кольорами стали зелений для позитивних дій та статусів (наприклад, «Завершено»), жовтий для проміжних статусів (наприклад, «В процесі») та червоний для критичних повідомлень і статусів (наприклад, «Затримка»). Така кольорова схема дозволяє користувачам швидко оцінювати стан системи та замовлень на ремонт за допомогою кольорових індикаторів, не вчитуючись у текстову інформацію. Для забезпечення доступності інтерфейсу для користувачів з порушеннями зору був забезпечений достатній контраст між текстом та фоном, а також передбачена можливість збільшення розміру шрифту.

Типографіка веб-застосунку базується на використанні сучасних шрифтів без засічок (sans-serif), які забезпечують гарну читабельність на різних пристроях. Для заголовків використовується шрифт Helvetica, а для основного тексту — Open Sans. Ієрархія текстових елементів (заголовки різних рівнів, підписи, основний текст) чітко визначена за допомогою різних розмірів, насиченості та кольорів шрифту, що дозволяє користувачам швидко сканувати сторінки та знаходити потрібну інформацію. Всі текстові елементи інтерфейсу розроблені з урахуванням правил української типографіки, включаючи використання правильних лапок, тире та інших специфічних символів.

Для забезпечення інтуїтивно зрозумілої навігації у веб-застосунку розроблений єдиний принцип організації меню та панелей інструментів. Головне навігаційне меню розташоване у верхній частині сторінки та містить посилання на

основні розділи системи: «Головна», «Клієнти», «Автомобілі», «Замовлення», «Розклад», «Слюсарі», «Звіти», «Налаштування». Для покращення орієнтації користувача у системі передбачено візуальне виділення активного пункту меню та «хлібні крихти» (breadcrumbs), що показують шлях до поточної сторінки. Додатково розроблені панелі швидкого доступу з найбільш часто використовуваними функціями для кожного розділу системи, які розміщуються у верхній частині контентної області. Такий підхід забезпечує логічну структуру навігації та дозволяє користувачу швидко знаходити потрібні функції.

Інтерфейс розроблений з урахуванням принципів адаптивного дизайну, який забезпечує коректне відображення на різних пристроях (комп'ютерах, планшетах, смартфонах) та при різних розмірах екрану. Для цього використовуються гнучкі сітки (flexible grids), відносні одиниці виміру, медіа-запити (media queries) та інші технології адаптивного дизайну. На великих екранах інтерфейс розкривається на всю ширину, надаючи максимальний огляд даних, на середніх — адаптується для збереження зручності використання, а на малих — перебудовується для мобільного перегляду, змінюючи розташування елементів та розмір шрифтів. Це забезпечує комфортну роботу з системою незалежно від пристрою, що використовується, та місця розташування користувача (офіс, ремонтна зона, виїзд до клієнта).

Для основних компонентів інтерфейсу, таких як таблиці, форми, кнопки, випадаючі списки, календарі, були розроблені стандартизовані шаблони, які забезпечують єдиний стиль оформлення по всій системі. Такий підхід не лише покращує естетичне сприйняття інтерфейсу, але й спрощує процес навчання користувачів, оскільки однотипні елементи інтерфейсу працюють однаково в різних частинах системи. Наприклад, всі форми введення даних мають стандартне розташування полів, підписів, кнопок підтвердження та скасування, а всі таблиці з даними мають єдиний стиль оформлення заголовків стовпців, рядків, елементів навігації та кнопок дій [18]. Такий підхід також спрощує процес подальшої розробки та підтримки системи, оскільки дозволяє використовувати готові компоненти для нових функцій без необхідності розробки нових елементів інтерфейсу.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ВЕБ-ЗАСТОСУНКУ З УПРАВЛІННЯ АВТОСЕРВІСОМ

3.1. Вибір технологій та інструментів розробки

3.1.1. Серверна частина: PHP, MySQL

Модуль управління клієнтами є одним з ключових компонентів веб-застосунку з управління автосервісом, оскільки забезпечує функціональність для роботи з інформацією про клієнтів та їх автомобілі. Цей модуль надає засоби для додавання нових клієнтів, перегляду та редагування інформації про існуючих клієнтів, пошуку клієнтів за різними критеріями, а також видалення клієнтів з бази даних. Розробка модуля управління клієнтами базувалася на вимогах до функціональності веб-застосунку, визначених на етапі проектування, та на аналізі бізнес-процесів автосервісу, пов'язаних з роботою з клієнтами.

Основним компонентом модуля є файл add_client.php, який реалізує функціональність додавання нових клієнтів та їх автомобілів до бази даних. Файл містить HTML-форму для введення даних клієнта (ім'я, телефон, email) та його автомобіля (VIN-код, марка, модель, рік випуску), а також PHP-код для обробки даних форми при її відправці. Форма має поля для введення всіх необхідних даних з відповідними підписами та підказками щодо формату введення. Для поля "Рік випуску" встановлені обмеження на мінімальне та максимальне значення (1900-2100), що запобігає введенню неправильних даних. При відправці форми дані перевіряються на серверній стороні для забезпечення їх коректності та відповідності вимогам бази даних. Якщо дані пройшли валідацію, вони вставляються в базу даних за допомогою підготовлених SQL-запитів, що забезпечує захист від SQL-ін'єкцій.

Для перегляду списку клієнтів та їх автомобілів розроблений файл view_clients.php, який відображає таблицю з інформацією про всіх клієнтів автосервісу. Таблиця містить колонки з іменем клієнта, телефоном, email, а також інформацією про його автомобіль (VIN-код, марка, модель, рік випуску). Для отримання цих даних з бази виконується SQL-запит, який об'єднує таблиці "clients" та "vehicles" за допомогою операції JOIN. Результати запиту виводяться у вигляді

HTML-таблиці з використанням циклу, який генерує рядок таблиці для кожного клієнта. Для кожного клієнта також відображаються кнопки для швидких дій: редагування інформації та видалення клієнта.

Функціональність пошуку клієнтів реалізована за допомогою форми з полями для введення критеріїв пошуку: імені клієнта, телефону, марки та моделі автомобіля. При відправці форми пошуку формується SQL-запит з умовами, що відповідають введеним критеріям. Наприклад, якщо користувач ввів ім'я "Іван" та марку автомобіля "Toyota", запит буде шукати клієнтів з іменем, що містить "Іван", та автомобілями марки "Toyota". Результати пошуку відображаються у такому ж форматі, як і загальний список клієнтів, але містять лише записи, що відповідають критеріям пошуку [19]. Для редагування інформації про клієнта та його автомобіль реалізований файл edit_client.php, який дозволяє змінювати всі дані, крім VIN-коду автомобіля (який є унікальним ідентифікатором і зазвичай не змінюється). Функціональність видалення клієнта реалізована з використанням механізму каскадного видалення, який забезпечується обмеженнями цілісності, визначеними в структурі бази даних.

3.1.2. Клієнтська частина: HTML, CSS

Клієнтська частина веб-застосунку з управління автосервісом розроблена з використанням сучасних технологій веб-розробки — HTML5 (HyperText Markup Language) та CSS3 (Cascading Style Sheets). HTML5 забезпечує структурну основу інтерфейсу, визначаючи елементи сторінки та їх організацію, в той час як CSS3 відповідає за візуальне оформлення цих елементів, включаючи кольори, шрифти, відступи, анімації та інші аспекти презентації. Такий підхід до розробки клієнтської частини дозволяє розділити зміст та представлення, що значно спрощує подальшу підтримку та розвиток системи.

Структура HTML-документів веб-застосунку розроблена відповідно до принципів семантичної верстки, що передбачає використання HTML-тегів відповідно до їх смислового призначення. Наприклад, для заголовків різних рівнів використовуються теги h1-h6, для навігаційного меню — тег nav, для основного контенту — тег main, для підвалу сторінки — тег footer. Такий підхід покращує

доступність сайту для пошукових систем та користувачів з обмеженими можливостями, які використовують спеціальні технології для роботи з веб-сторінками, такі як програми читання з екрану. Для кожної сторінки веб-застосунку розроблена чітка структура HTML-документа з визначеними секціями та елементами, що забезпечує логічну організацію контенту та спрощує сприйняття інформації користувачами.

Для організації елементів на сторінці та створення адаптивного дизайну використовується CSS-сітка (grid layout) та флексбокс (flexbox). Ці сучасні CSS-технології дозволяють ефективно розміщувати елементи на сторінці, створювати складні макети та забезпечувати їх коректне відображення на різних пристроях — від настільних комп'ютерів до мобільних телефонів. Наприклад, на головній сторінці адміністративної панелі використовується CSS-сітка для розміщення інформаційних блоків з ключовими показниками в кілька колонок на великих екранах та в одну колонку на мобільних пристроях. Для забезпечення адаптивності дизайну використовуються медіа-запити (media queries), які дозволяють застосовувати різні стилі в залежності від характеристик пристрою, на якому відображається сайт. Зокрема, визначені контрольні точки для різних розмірів екрану: мобільні телефони (до 767px), планшети (768px-1023px) та настільні комп'ютери (від 1024px).

Для забезпечення єдиного стилю оформлення по всьому веб-застосунку розроблені CSS-змінні (CSS Custom Properties), які визначають основні параметри дизайну: кольорову палітру, розміри шрифтів, відступи, радіуси заокруглення кутів тощо. Ці змінні використовуються в усіх CSS-правилах, що забезпечує консистентність дизайну та спрощує внесення глобальних змін. Наприклад, зміна значення змінної --primary-color призведе до автоматичного оновлення кольору всіх елементів, для яких використовується цей колір, без необхідності редагувати кожне CSS-правило окремо [20]. Для покращення користувацького досвіду в клієнтській частині використовуються різноманітні CSS-ефекти та анімації. Наприклад, при наведенні курсора на кнопки та посилання застосовуються ефекти зміни кольору та тіні, що візуально підкреслює інтерактивність елементів. Для форм введення даних реалізовані ефекти фокусу, які виділяють активне поле та покращують орієнтацію користувача.

3.2.1. Створення таблиць та зв'язків

Процес створення структури бази даних для веб-застосунку з управління автосервісом базувався на розробленій раніше фізичній моделі даних. Для реалізації бази даних використовувалася система управління реляційними базами даних MySQL/MariaDB, яка є частиною набору програм XAMPP. Створення таблиць та зв'язків між ними здійснювалося за допомогою SQL-запитів мови визначення даних (DDL — Data Definition Language), які виконувалися через веб-інтерфейс інструменту phpMyAdmin, що спрощує управління базою даних та надає зручні інструменти для створення, редагування та видалення таблиць, індексів, зв'язків тощо.

Першим кроком було створення бази даних з назвою "autoservice_db" з кодуванням utf8mb4, яке забезпечує підтримку Unicode символів, включаючи емодзі, що важливо для коректного зберігання та відображення різних мов та спеціальних символів. Команда створення бази даних виглядає наступним чином: CREATE DATABASE autoservice_db CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;. Після створення бази даних були розроблені SQL-запити для створення всіх необхідних таблиць відповідно до фізичної моделі даних. Центральною таблицею бази даних є "clients", яка містить інформацію про клієнтів автосервісу. Таблиця має наступну структуру: первинний ключ id (цілочисельний тип з автоінкрементом), ім'я клієнта name (рядковий тип VARCHAR з обмеженням на 100 символів), телефон phone (рядковий тип VARCHAR з обмеженням на 20 символів) та електронна пошта email (рядковий тип VARCHAR з обмеженням на 100 символів).

Таблиця "vehicles" містить інформацію про автомобілі клієнтів і має наступну структуру: первинний ключ id, зовнішній ключ client_id, який посилається на таблицю "clients" та забезпечує зв'язок між клієнтами та їх автомобілями, VIN-код автомобіля vin (рядковий тип VARCHAR з обмеженням на 50 символів та обмеженням UNIQUE для забезпечення унікальності VIN-кодів), марка make та модель model (рядкові типи VARCHAR з обмеженням на 50 символів), рік випуску year (спеціальний тип YEAR). Зовнішній ключ client_id має обмеження ON DELETE

CASCADE, яке забезпечує автоматичне видалення всіх автомобілів клієнта при видаленні самого клієнта. Таблиця "mechanics" містить інформацію про слюсарів автосервісу і має структуру, подібну до таблиці "clients": первинний ключ id, ім'я слюсаря name, телефон phone та електронна пошта email. Ця таблиця є важливою для реалізації функціоналу призначення слюсарів на виконання ремонтних робіт та аналізу їх завантаженості.

Таблиця "repairs" є ключовою для відображення бізнес-процесів автосервісу і має наступну структуру: первинний ключ id, зовнішній ключ vehicle_id, який посилається на таблицю "vehicles", опис проблеми або необхідних робіт description (текстовий тип TEXT), необхідні запчастини parts_needed (текстовий тип TEXT), дата створення або планована дата ремонту date (тип DATE), зовнішній ключ mechanic_id, який посилається на таблицю "mechanics", статус ремонту status (тип ENUM з можливими значеннями "Очікує", "В процесі", "Завершено"), зовнішній ключ client_id, який посилається на таблицю "clients" та дублює зв'язок з клієнтом через автомобіль для оптимізації запитів [21]. Для зовнішніх ключів визначені відповідні обмеження цілісності: для vehicle_id — ON DELETE CASCADE, для mechanic_id — ON DELETE SET NULL.

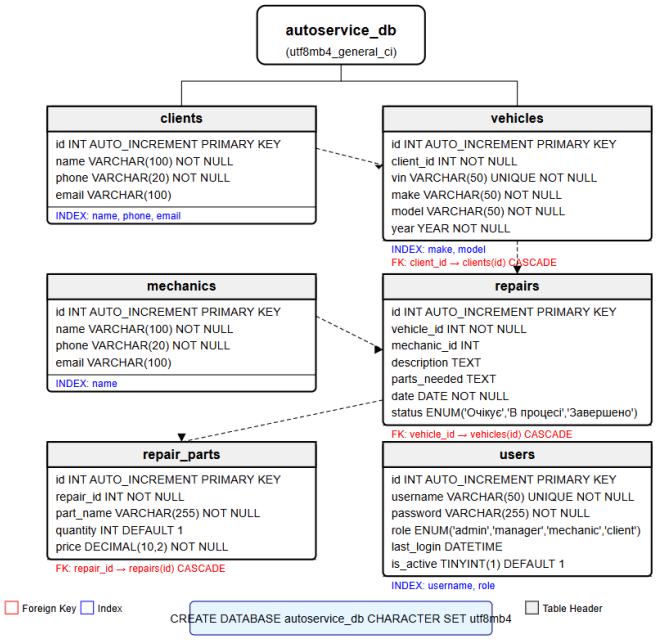


Рис. 3.1 - Створення таблиць та зв'язків бази даних автосервісу

Після створення всіх необхідних таблиць були створені індекси для полів, що часто використовуються у пошукових запитах та для з'єднання таблиць, що оптимізує продуктивність бази даних при виконанні складних запитів.

3.2.2. Наповнення тестовими даними

Після створення структури бази даних важливим етапом розробки веб-застосунку з управління автосервісом є наповнення її тестовими даними. Це необхідно для перевірки коректності роботи програмного коду, оцінки продуктивності системи, демонстрації функціональних можливостей застосунку та проведення навчання персоналу. Для наповнення бази даних тестовими даними були розроблені SQL-скрипти, які виконують вставку записів у всі таблиці бази даних, забезпечуючи при цьому зв'язність даних відповідно до визначених обмежень цілісності (див. рисунок 3.1).

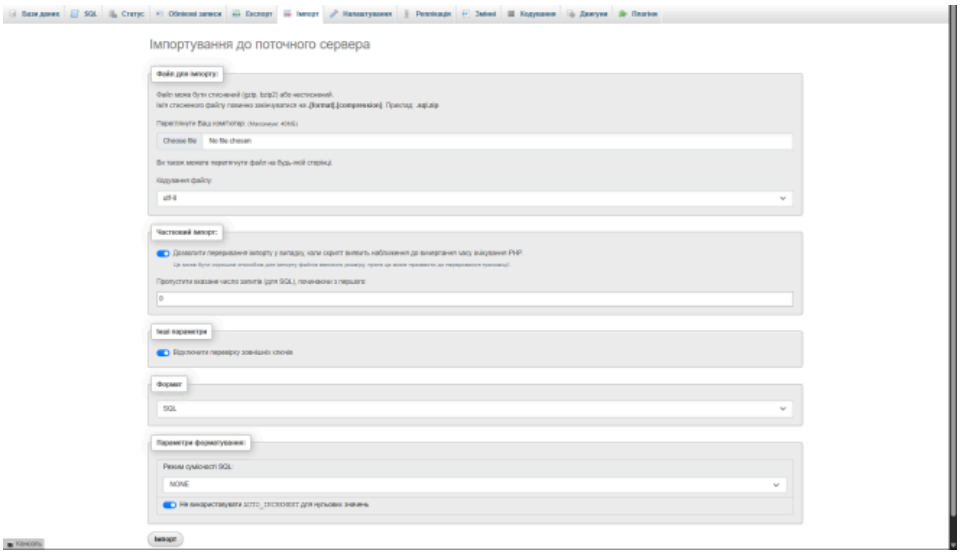


Рис. 3.1 – Вікно імпорту SQL-файлу до бази даних через phpMyAdmin

Першим кроком було наповнення таблиці "clients" даними про клієнтів автосервісу. Для цього був розроблений SQL-запит, який вставляє в таблицю п'ять записів з реалістичними, але вигаданими даними: INSERT INTO clients (id, name, phone, email) VALUES (1, 'Іван Іванов', '0971234567', 'ivan.ivanov@example.com'), (2, 'Петро Петренко', '0987654321', 'petro.petrenko@example.com'), (3, 'Марія Коваль', '0951122334', 'maria.koval@example.com'), (4, 'Олександр Сидоренко', '0934455667',

'oleksandr.sydorenko@example.com'), (5, 'Наталія Власенко', '0967788990', 'natalia.vlasenko@example.com');. Імена, телефони та електронні адреси були обрані таким чином, щоб вони виглядали реалістично, але не співпадали з реальними людьми.

Наступним кроком було наповнення таблиці "vehicles" даними про автомобілі клієнтів. Для кожного клієнта був створений один запис про автомобіль з вказанням VIN-коду, марки, моделі та року випуску. Наприклад, для клієнта з id=1 був створений запис про автомобіль Toyota Corolla 2015 року випуску з VIN-кодом 1HGBH41JXMN109186. VIN-коди для тестових даних були згенеровані випадковим чином, але з дотриманням формату, характерного для справжніх VIN-кодів. Марки та моделі автомобілів були обрані з числа найбільш популярних, а роки випуску – в діапазоні від 2015 до 2020 року.

Таблиця "mechanics" була наповнена даними про п'ятьох слюсарів автосервісу з вказанням їх імен, телефонів та електронних адрес. Імена були обрані таким чином, щоб представити різноманітність персоналу, включаючи чоловіків та жінок. Телефони та електронні адреси, як і у випадку з клієнтами, були створені за реалістичним шаблоном, але не відповідають реальним контактам. Ці дані є важливими для тестування функціоналу призначення слюсарів на виконання ремонтних робіт та аналізу їх завантаженості.

Для таблиці "repairs" були створені тестові записи про ремонти автомобілів з різними статусами: "Очікує", "В процесі", "Завершено". Кожен запис містить інформацію про автомобіль (через зовнішній ключ vehicle_id), опис проблеми або необхідних робіт, перелік необхідних запчастин, дату ремонту, призначеного слюсаря (через зовнішній ключ mechanic_id) та статус. Описи проблем та переліки запчастин були створені на основі типових ситуацій, що виникають при обслуговуванні автомобілів: заміна гальмівних колодок, ремонт кондиціонера, заміна масла в двигуні, ремонт трансмісії, технічне обслуговування. Дати ремонтів були розподілені на часовій шкалі таким чином, щоб деякі ремонти були заплановані на минулі дати (зі статусом "Завершено"), деякі – на поточну дату (зі статусом "В процесі"), а деякі – на майбутні дати (зі статусом "Очікує").

Для забезпечення цілісності даних при наповненні бази використовувалися правильні зв'язки між записами в різних таблицях. Наприклад, при створенні запису

про ремонт в таблиці "repairs" поле vehicle_id містить значення, яке відповідає існуючому запису в таблиці "vehicles", а поле mechanic_id – існуючому запису в таблиці "mechanics". Це забезпечує відповідність даних реальним бізнес-процесам автосервісу та дозволяє коректно відображати взаємозв'язки між різними сутностями при тестуванні веб-застосунку [22].

Для наповнення бази даних великим обсягом тестових даних, необхідним для тестування продуктивності системи, були розроблені додаткові SQL-скрипти, які генерують значну кількість записів з випадковими, але реалістичними значеннями. Наприклад, для генерації 1000 клієнтів використовувався цикл, в якому для кожного клієнта створювалося унікальне ім'я, телефон та електронна адреса за певним шаблоном, а потім для кожного клієнта створювалося від одного до трьох автомобілів з випадковими, але реалістичними характеристиками.

Після наповнення бази даних тестовими даними була проведена перевірка їх коректності та відповідності бізнес-правилам. Зокрема, були перевірені зв'язки між таблицями, унікальність значень для полів з обмеженням UNIQUE (наприклад, VIN-кодів автомобілів), відповідність значень перерахованих типів (ENUM) допустимим варіантам (наприклад, статусів ремонтів). Також були виконані тестові запити на вибірку даних, які моделюють типові операції веб-застосунку, для перевірки коректності результатів та оцінки продуктивності системи при роботі з тестовими даними.

Тестові дані, створені на цьому етапі, були використані в процесі розробки та тестування всіх модулів веб-застосунку, а також для демонстрації функціональних можливостей системи замовнику та навчання персоналу автосервісу роботі з новою системою. За необхідності тестові дані можуть бути видалені з бази перед початком реальної експлуатації системи за допомогою SQL-запитів TRUNCATE або DELETE, які очищають таблиці, але зберігають їх структуру, готову для наповнення реальними даними.

3.3. Реалізація основних функціональних модулів

3.3.1. Модуль управління клієнтами

Модуль управління клієнтами є одним з ключових компонентів веб-застосунку з управління автосервісом, оскільки забезпечує функціональність для роботи з інформацією про клієнтів та їх автомобілі. Цей модуль надає засоби для додавання нових клієнтів, перегляду та редагування інформації про існуючих клієнтів, пошуку клієнтів за різними критеріями, а також видалення клієнтів з бази даних. Розробка модуля управління клієнтами базувалася на вимогах до функціональності веб-застосунку, визначених на етапі проектування, та на аналізі бізнес-процесів автосервісу, пов'язаних з роботою з клієнтами.

Основним компонентом модуля є файл add_client.php, який реалізує функціональність додавання нових клієнтів та їх автомобілів до бази даних. Файл містить HTML-форму для введення даних клієнта (ім'я, телефон, email) та його автомобіля (VIN-код, марка, модель, рік випуску), а також PHP-код для обробки даних форми при її відправці. Форма має поля для введення всіх необхідних даних з відповідними підписами та підказками щодо формату введення. Для поля "Рік випуску" встановлені обмеження на мінімальне та максимальне значення (1900-2100), що запобігає введенню неправильних даних.

При відправці форми додавання нового клієнта дані перевіряються на серверній стороні для забезпечення їх коректності та відповідності вимогам бази даних. Зокрема, перевіряється заповнення обов'язкових полів (ім'я клієнта, телефон, VIN-код), формат введених даних (наприклад, формат email), унікальність VIN-коду автомобіля (для запобігання дублювання автомобілів в базі даних). Якщо дані пройшли валідацію, вони вставляються в базу даних за допомогою підготовлених SQL-запитів, що забезпечує захист від SQL-ін'єкцій. Спочатку створюється запис про клієнта в таблиці "clients", а потім – запис про його автомобіль в таблиці "vehicles" з посиланням на створеного клієнта.

Для перегляду списку клієнтів та їх автомобілів розроблений файл view_clients.php, який відображає таблицю з інформацією про всіх клієнтів автосервісу. Таблиця містить колонки з іменем клієнта, телефоном, email, а також інформацією про його автомобіль (VIN-код, марка, модель, рік випуску). Для отримання цих даних з бази виконується SQL-запит, який об'єднує таблиці "clients" та "vehicles" за допомогою операції JOIN. Результати запиту виводяться у вигляді HTML-таблиці з використанням циклу, який генерує рядок таблиці для кожного

клієнта. Для кожного клієнта також відображаються кнопки для швидких дій: редагування інформації та видалення клієнта.

Функціональність пошуку клієнтів реалізована за допомогою форми з полями для введення критеріїв пошуку: імені клієнта, телефону, марки та моделі автомобіля. При відправці форми пошуку формується SQL-запит з умовами, що відповідають введеним критеріям. Наприклад, якщо користувач ввів ім'я "Іван" та марку автомобіля "Toyota", запит буде шукати клієнтів з іменем, що містить "Іван", та автомобілями марки "Toyota". Результати пошуку відображаються у такому ж форматі, як і загальний список клієнтів, але містять лише записи, що відповідають критеріям пошуку. Це дозволяє швидко знаходити потрібних клієнтів навіть у великій базі даних.

Для редагування інформації про клієнта та його автомобіль реалізований файл edit_client.php, який дозволяє змінювати всі дані, крім VIN-коду автомобіля (який є унікальним ідентифікатором і зазвичай не змінюється). При переході на сторінку редагування з параметром id клієнта з бази даних отримується поточна інформація про клієнта та його автомобіль, яка заповнює поля форми. Користувач може внести необхідні зміни та зберегти їх, після чого дані оновлюються в базі даних за допомогою SQL-запитів UPDATE. Як і при додаванні нового клієнта, всі дані проходять валідацію перед збереженням для забезпечення їх коректності.

Функціональність видалення клієнта реалізована з використанням механізму каскадного видалення, який забезпечується обмеженнями цілісності, визначеними в структурі бази даних. При кліку на кнопку "Видалити" для конкретного клієнта користувачу відображається діалогове вікно з підтвердженням дії, оскільки видалення є необоротною операцією. Якщо користувач підтверджує видалення, виконується SQL-запит DELETE, який видаляє запис про клієнта з таблиці "clients". Завдяки обмеженню ON DELETE CASCADE, визначеному для зовнішнього ключа client_id в таблиці "vehicles", автоматично видаляються також всі записи про автомобілі цього клієнта [23]. Така логіка забезпечує цілісність даних та запобігає появі "висячих" записів в базі даних.

Модуль управління клієнтами також включає функціональність для перегляду історії звернень клієнта до автосервісу. Для кожного клієнта можна переглянути список всіх ремонтів його автомобілів з інформацією про дату, тип робіт, статус

виконання та вартість. Ця функціональність реалізована за допомогою SQL-запиту, який об'єднує таблиці "clients", "vehicles" та "repairs" та вибирає всі ремонти для автомобілів конкретного клієнта. Результати відображаються у вигляді таблиці з можливістю сортування за різними параметрами (дата, статус, вартість). Така інформація є цінною для аналізу історії взаємодії з клієнтом та для планування майбутніх робіт.

Для забезпечення зручності роботи з модулем управління клієнтами реалізовані додаткові функції, такі як підказки при введенні даних (автодоповнення для полів марки та моделі автомобіля на основі списку популярних варіантів), валідація даних на стороні клієнта з використанням JavaScript (для швидкого виявлення помилок введення без необхідності відправки форми на сервер), форматування телефонних номерів та інших полів відповідно до прийнятих стандартів. Всі ці функції спрямовані на покращення користувацького досвіду та зменшення кількості помилок при роботі з системою.

3.3.2. Модуль управління автомобілями

Модуль управління автомобілями є важливою складовою веб-застосунку з управління автосервісом, що забезпечує функціональність для роботи з інформацією про транспортні засоби клієнтів. Цей модуль тісно інтегрований з модулем управління клієнтами, оскільки кожен автомобіль належить певному клієнту, але містить специфічні функції для роботи саме з автомобілями. Модуль надає можливості для додавання нових автомобілів для існуючих клієнтів, перегляду детальної інформації про автомобілі, редагування технічних характеристик, пошуку автомобілів за різними параметрами та відстеження історії обслуговування конкретного транспортного засобу.

Ключовим компонентом модуля є файл add_vehicle.php, який реалізує функціональність додавання нового автомобіля для існуючого клієнта. Форма додавання автомобіля містить випадаючий список для вибору клієнта (заповнюється даними з таблиці "clients"), а також поля для введення інформації про автомобіль: VIN-код, марка, модель, рік випуску. Поле VIN-код має валідацію на правильність формату (17 символів, що складаються з цифр та літер, за винятком літер I, O, Q), а

також перевірку на унікальність, щоб уникнути дублювання автомобілів у базі даних. Для полів марка та модель реалізовано автодоповнення на основі списку популярних марок та моделей, що спрощує введення даних та зменшує ймовірність помилок.

Для перегляду списку всіх автомобілів, зареєстрованих в системі, розроблений файл view_vehicles.php, який відображає таблицю з інформацією про кожен автомобіль: VIN-код, марка, модель, рік випуску, а також ім'я та контактні дані власника. Для отримання цієї інформації з бази даних виконується SQL-запит, який об'єднує таблиці "vehicles" та "clients" за допомогою операції JOIN. Результати запиту виводяться у вигляді таблиці з можливістю сортування за різними параметрами (марка, модель, рік випуску, власник). Для кожного автомобіля також відображаються кнопки для швидких дій: перегляд деталей, редагування інформації, перегляд історії обслуговування.

Функціональність пошуку автомобілів реалізована через форму з полями для введення різних критеріїв: VIN-код, марка, модель, рік випуску, ім'я власника. При відправці форми пошуку формується динамічний SQL-запит, який включає умови, що відповідають введеним критеріям. Для детального перегляду інформації про конкретний автомобіль розроблений файл vehicle_details.php, який відображає всю доступну інформацію про вибраний транспортний засіб: VIN-код, марка, модель, рік випуску, дата реєстрації в системі, дані про власника, а також технічні характеристики та історію обслуговування. Технічні характеристики можуть включати тип двигуна, об'єм двигуна, тип коробки передач, пробіг та інші параметри, важливі для планування обслуговування та ремонту [24].

Модуль управління автомобілями також включає функціональність для відстеження історії обслуговування автомобіля та формування електронної сервісної книжки. Сервісна книжка формується у форматі PDF з використанням бібліотеки FPDF для PHP і включає інформацію про автомобіль, його власника, а також таблицю з усіма записами про проведені ремонти та технічні обслуговування. Кожен запис містить дату, тип робіт, перелік замінених запчастин, пробіг на момент обслуговування, виконавця робіт та печатку автосервісу. Така сервісна книжка є важливим документом для власника автомобіля і може підвищити вартість

автомобіля при його продажу, оскільки підтверджує регулярне та якісне обслуговування.

3.3.3. Модуль управління замовленнями на ремонт

Модуль управління замовленнями на ремонт є центральним компонентом веб-застосунку з управління автосервісом, оскільки він забезпечує автоматизацію основного бізнес-процесу – процесу прийому, виконання та видачі замовлень на ремонт та технічне обслуговування автомобілів. Цей модуль інтегрує дані про клієнтів, автомобілі, слюсарів, запчастини та роботи, забезпечуючи повний цикл обробки замовлення від моменту його створення до закриття після виконання всіх робіт та розрахунку з клієнтом.

Основним компонентом модуля є файл add_repair.php, який реалізує функціональність створення нового замовлення на ремонт. Форма створення замовлення містить випадаючий список для вибору автомобіля клієнта (заповнюється даними з таблиць "clients" та "vehicles", відображаючи ім'я клієнта, марку, модель та рік випуску автомобіля), поле для опису проблеми або необхідних робіт, поле для вказання необхідних деталей, календар для вибору бажаної дати ремонту (див.рисунок 3.2).

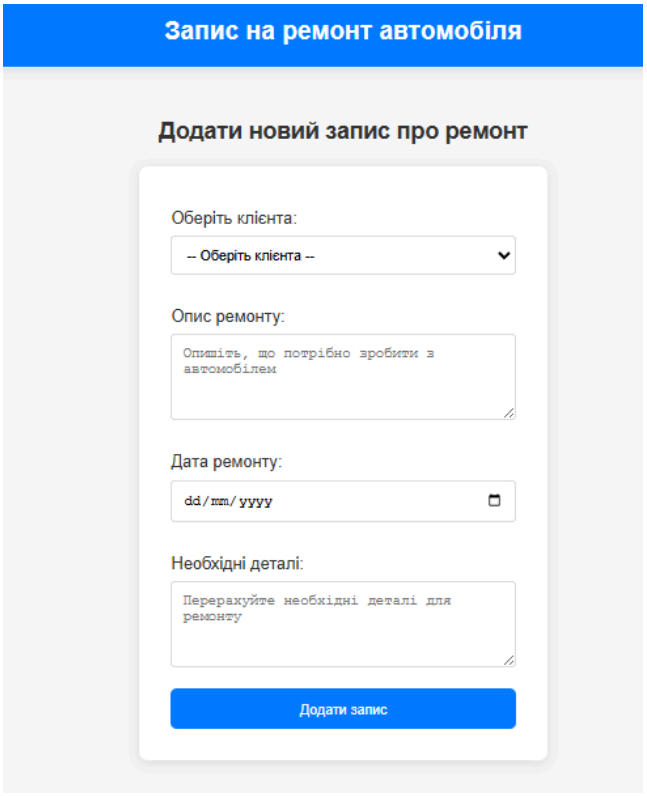


Рис. 3.2 – Форма створення нового замовлення на ремонт автомобіля

Також передбачена можливість завантаження фотографій або інших документів, пов'язаних з замовленням (наприклад, фотографій пошкоджень автомобіля). При відправці форми дані зберігаються в таблиці "repairs" з початковим статусом "Очікує", а завантажені файли зберігаються на сервері з посиланнями на них у базі даних.

Для перегляду всіх замовлень на ремонт розроблений файл view_repairs.php, який відображає таблицю зі списком замовлень з можливістю фільтрації за різними параметрами: статус, дата, клієнт, автомобіль, слюсар. Кожен рядок таблиці містить ключову інформацію про замовлення: номер, дату створення, клієнта, автомобіль, короткий опис робіт, статус, призначеного слюсаря. Для наочного відображення статусу використовується кольорове кодування: жовтий для статусу "Очікує", синій для "В процесі", зелений для "Завершено". Для кожного замовлення також відображаються кнопки для швидких дій: перегляд деталей, редагування, зміна статусу, видалення.

Для детального перегляду інформації про конкретне замовлення на ремонт розроблений файл repair_details.php, який відображає всі дані про замовлення: клієнта та його контактні дані, автомобіль та його характеристики, опис проблеми,

результати діагностики, перелік необхідних робіт та запчастин з вартістю, призначеного слюсаря, поточний статус, історію змін статусу, а також всі прикріплені файли. На цій сторінці також доступні кнопки для генерування та друку необхідних документів: акту прийому-передачі автомобіля, наряду-замовлення, акту виконаних робіт, рахунку. Документи генеруються у форматі PDF з використанням бібліотеки FPDF для PHP і включають всю необхідну інформацію з автоматичним заповненням даними з бази даних.

Функціональність зміни статусу замовлення реалізована через інтерфейс з кнопками для переходу між статусами: "Прийнято" -> "Діагностика" -> "Очікування запчастин" -> "В роботі" -> "Готово" -> "Видано". При зміні статусу в базі даних фіксується не лише новий статус, але й дата та час зміни, а також користувач, який виконав зміну. Це дозволяє відстежувати історію виконання замовлення та аналізувати час, витрачений на кожен етап. Також при зміні статусу на "Готово" система автоматично формує рахунок на основі виконаних робіт та використаних запчастин, а при зміні на "Видано" – фіксує факт оплати та закриття замовлення.

Для управління переліком робіт в рамках замовлення розроблена функціональність додавання, редагування та видалення окремих робіт. Кожна робота характеризується типом (діагностика, ремонт, заміна деталей, регулювання тощо), описом, нормативним часом виконання, фактичним часом виконання, вартістю, виконавцем. Сумарна вартість замовлення розраховується як сума вартостей усіх робіт та використаних запчастин. Для зручності використання передбачено можливість вибору робіт зі списку типових операцій з автоматичним заповненням нормативного часу та вартості відповідно до встановлених тарифів.

Для управління запчастинами, необхідними для виконання ремонту, реалізована функціональність додавання, редагування та видалення запчастин в рамках замовлення. Для кожної запчастини вказується найменування, кількість, ціна за одиницю. При додаванні запчастини система перевіряє її наявність на складі і, якщо запчастина в наявності, автоматично резервує її під це замовлення, зменшуючи доступну кількість. Якщо запчастини немає в наявності або її кількість недостатня, система пропонує оформити замовлення постачальнику з можливістю вказання очікуваної дати поставки, що впливає на загальний термін виконання замовлення.

Модуль управління замовленнями на ремонт також включає функціональність для комунікації з клієнтом щодо статусу виконання замовлення. При зміні статусу замовлення або при появі важливої інформації (наприклад, виявлення додаткових несправностей під час діагностики, затримка поставки запчастин) система може автоматично надсилати повідомлення клієнту електронною поштою або SMS з актуальною інформацією. Також реалізована можливість для клієнта перевіряти статус свого замовлення через спеціальну форму на сайті автосервісу, вказавши номер замовлення та контактний телефон або email.

Для забезпечення ефективної роботи автосервісу модуль включає аналітичні функції для аналізу замовлень на ремонт: статистика за типами робіт, середній час виконання різних типів робіт, середня вартість ремонту для різних марок та моделей автомобілів, завантаженість слюсарів та постів. Ця аналітика відображається у вигляді графіків та діаграм на спеціальній сторінці, доступній для адміністраторів та менеджерів автосервісу, і дозволяє приймати обґрунтовані управлінські рішення щодо оптимізації роботи, перегляду тарифів, навчання персоналу тощо.

3.3.4. Модуль планування розкладу робіт

Модуль планування розкладу робіт є важливим компонентом веб-застосунку з управління автосервісом, який забезпечує оптимальне використання ресурсів (слюсарів, ремонтних постів, обладнання) та ефективне планування робіт. Цей модуль дозволяє візуалізувати завантаженість автосервісу, планувати нові роботи на оптимальний час, контролювати виконання робіт відповідно до графіку та швидко реагувати на зміни у розкладі. Такий підхід до планування робіт допомагає уникнути простоїв обладнання та персоналу, а також забезпечити виконання робіт у строки, погоджені з клієнтами.

Основним компонентом модуля є файл schedule.php, який реалізує функціональність перегляду та управління розкладом робіт у формі інтерактивного календаря. Календар дозволяє переглядати завантаженість автосервісу в різних режимах: день, тиждень, місяць. У кожній комірці календаря відображаються заплановані роботи з ключовою інформацією: клієнт, автомобіль, тип робіт, призначений слюсар. Роботи з різними статусами відображаються різними

кольорами для наочності. Наприклад, заплановані роботи — жовтим, роботи в процесі — синім, завершені роботи — зеленим. При наведенні курсора на роботу відображається спливаюча підказка з детальною інформацією, а при кліку — відкривається модальне вікно з повними деталями замовлення та можливістю редагування.

Для планування нових робіт у календарі використовується механізм "drag-and-drop" (перетягування), який дозволяє вибрати вільний слот у розкладі і створити в ньому нове замовлення. При перетягуванні відображається підказка про доступність вибраного часу та відповідність кваліфікації слюсаря типу робіт. Також можна змінювати терміни вже запланованих робіт шляхом перетягування їх на інший час або день. При спробі перепланувати роботу на час, який вже зайнятий іншою роботою для того ж слюсаря або поста, система відображає попередження та пропонує альтернативні варіанти. Тривалість роботи можна змінювати шляхом перетягування нижньої межі блоку роботи в календарі.

Для оптимізації планування розкладу реалізована функціональність автоматичного розподілу робіт з урахуванням різних факторів: кваліфікації слюсарів, необхідного обладнання, пріоритетності замовлень, доступності запчастин, побажань клієнтів щодо часу виконання робіт. Алгоритм оптимізації знаходить найкращий варіант розподілу робіт, який мінімізує час простою слюсарів та обладнання, забезпечує вчасне виконання всіх замовлень, враховує технологічні особливості різних типів робіт. Наприклад, роботи, які вимагають використання підйомника, плануються тільки на пости, обладнані підйомниками, а роботи, які вимагають високої кваліфікації (наприклад, діагностика та ремонт електронних систем), призначаються слюсарям відповідної спеціалізації.

Для контролю виконання робіт відповідно до графіку реалізована функціональність відстеження фактичного часу початку та завершення робіт у порівнянні з плановим. Для кожної роботи фіксується плановий час початку та завершення, а також фактичний час. При значному відхиленні від графіку (наприклад, якщо робота не розпочата протягом 30 хвилин після запланованого часу або її тривалість перевищує планову більш ніж на 50%), система генерує повідомлення для менеджера автосервісу, що дозволяє оперативно реагувати на проблеми та перепланувати інші роботи за необхідності. Такий підхід забезпечує

ефективне використання ресурсів автосервісу та мінімізує затримки у виконанні замовлень.

3.3.5. Модуль призначення слюсарів

Модуль призначення слюсарів є спеціалізованим компонентом веб-застосунку з управління автосервісом, який відповідає за розподіл замовлень на ремонт між технічним персоналом – слюсарями. Цей модуль забезпечує оптимальне використання кадрових ресурсів автосервісу, враховуючи спеціалізацію, кваліфікацію та завантаженість кожного слюсаря, що дозволяє підвищити якість та швидкість виконання ремонтних робіт. Модуль тісно інтегрований з модулями управління замовленнями на ремонт та планування розкладу робіт, забезпечуючи комплексний підхід до організації роботи автосервісу.

Основним компонентом модуля є файл assign_mechanic.php, який реалізує функціональність призначення слюсаря для виконання конкретного замовлення на ремонт. Форма призначення слюсаря містить випадаючий список для вибору замовлення (з відображенням інформації про клієнта, автомобіль, дату та опис робіт) та випадаючий список для вибору слюсаря (з відображенням інформації про його ім'я, спеціалізацію та поточну завантаженість). При виборі замовлення автоматично відображається рекомендований список слюсарів, відсортований за відповідністю їх спеціалізації типу робіт, зазначеному в замовленні. Це спрощує процес вибору оптимального виконавця для кожного замовлення (див. рисунок 3.3).

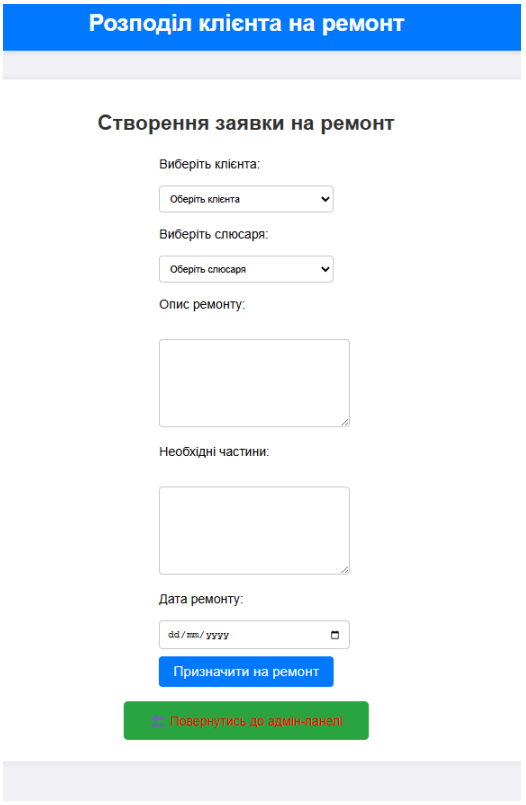


Рис. 3.3 – Форма призначення клієнта на ремонт та вибору слюсаря

Для ефективного управління персоналом автосервісу розроблений функціонал ведення бази даних слюсарів з детальною інформацією про кожного працівника. Для кожного слюсаря зберігається не лише контактна інформація (ім'я, телефон, email), але й професійні дані: спеціалізація, кваліфікаційний рівень, досвід роботи, сертифікати, графік роботи. Ця інформація використовується при призначенні слюсарів на виконання замовлень для забезпечення відповідності кваліфікації працівника складності робіт. Наприклад, складні діагностичні роботи призначаються слюсарям з високою кваліфікацією та відповідними сертифікатами, а простіші роботи можуть виконуватися слюсарями з меншим досвідом (див. рисунок 3.4-3.5).

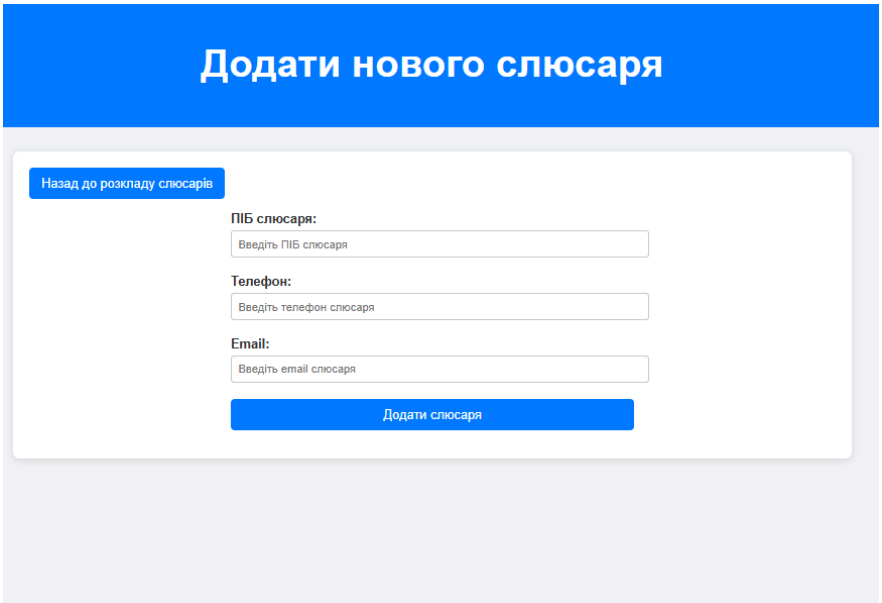


Рис. 3.4 – Форма додавання нового слюсаря до бази персоналу автосервісу

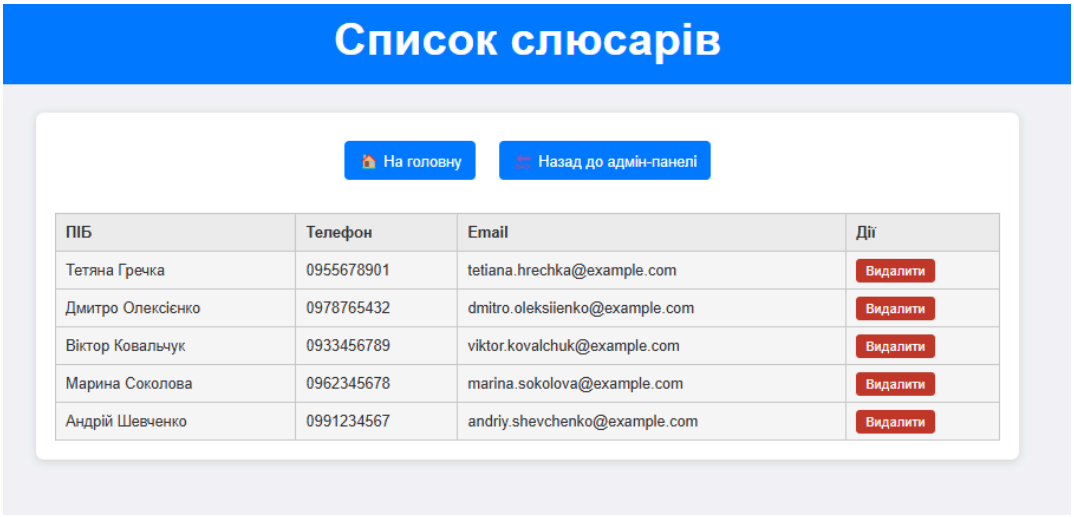


Рис. 3.5 – Таблиця зі списком слюсарів з контактною інформацією та можливістю керування записами

Для забезпечення справедливого розподілу робіт між слюсарями реалізована функціональність відстеження загальної завантаженості кожного працівника. Система аналізує кількість та складність замовлень, призначених кожному слюсарю, а також фактичний час виконання робіт у порівнянні з нормативним. На основі цих даних розраховується показник завантаженості, який використовується при призначенні нових замовлень для забезпечення рівномірного розподілу робіт. Такий підхід дозволяє уникнути ситуацій, коли одні слюсарі перевантажені роботою, а інші мають багато вільного часу, що підвищує загальну ефективність роботи автосервісу.

Для контролю якості виконання робіт модуль призначення слюсарів включає функціональність оцінки результатів роботи кожного слюсаря. Після завершення ремонту клієнт може залишити відгук та оцінку якості виконаних робіт за шкалою від 1 до 5. Також враховуються фактори, такі як дотримання термінів виконання, необхідність повторного виконання робіт через виявлені недоліки, скарги клієнтів. На основі цих даних формується рейтинг слюсарів, який враховується при розподілі замовлень – слюсарі з вищим рейтингом отримують пріоритет при призначенні на складні або термінові роботи, а також при роботі з VIP-клієнтами.

Для оптимізації використання робочого часу слюсарів реалізована функціональність планування їх завантаження з урахуванням технологічної послідовності робіт. Наприклад, якщо для виконання ремонту необхідно спочатку провести діагностику, потім замінити деталі, а потім провести регулювання, система планує ці роботи з урахуванням доступності слюсарів відповідної спеціалізації та необхідного обладнання. Також враховується можливість паралельного виконання різних робіт різними слюсарями, якщо це дозволяє технологічний процес. Такий підхід дозволяє мінімізувати загальний час виконання замовлення та забезпечити ефективне використання робочого часу всіх працівників.

Для обліку фактично відпрацьованого часу слюсарів модуль включає функціональність електронних табелів обліку робочого часу. Кожен слюсар при початку роботи над замовленням відмічає це в системі (через спеціальний інтерфейс на планшеті або комп'ютері в цеху), а при завершенні роботи також робить відмітку. Система автоматично розраховує час, витрачений на виконання роботи, порівнює його з нормативним часом та формує звіт про ефективність використання робочого часу. Ці дані використовуються не лише для контролю продуктивності праці, але й для розрахунку заробітної плати, якщо в автосервісі застосовується відрядна система оплати праці.

Для підвищення кваліфікації персоналу модуль призначення слюсарів включає функціональність планування навчання та сертифікації. На основі аналізу типів замовлень, що надходять до автосервісу, система визначає, які спеціалізації та навички є найбільш затребуваними, та рекомендує відповідні навчальні курси для слюсарів. Також відстежуються терміни дії сертифікатів та формуються нагадування про необхідність проходження повторної сертифікації. Ця функціональність

допомагає підтримувати високий рівень кваліфікації персоналу та забезпечувати якісне виконання всіх типів ремонтних робіт, що підвищує конкурентоспроможність автосервісу на ринку.

Модуль призначення слюсарів є ключовим елементом системи управління автосервісом, який забезпечує ефективне використання кадрових ресурсів, підвищення якості обслуговування та оптимізацію бізнес-процесів. Завдяки інтеграції з іншими модулями системи, зокрема з модулями управління замовленнями на ремонт та планування розкладу робіт, модуль призначення слюсарів створює комплексне рішення для організації роботи автосервісу, що дозволяє підвищити продуктивність праці, задоволеність клієнтів та прибутковість бізнесу.

3.4. Реалізація користувацького інтерфейсу

3.4.1. Сторінка авторизації

Сторінка авторизації є точкою входу до адміністративної частини веб-застосунку з управління автосервісом та забезпечує контроль доступу до системи. Ця сторінка реалізована у файлі login.php і представляє собою форму для введення облікових даних користувача – логіна та пароля.

Дизайн сторінки виконаний у мінімалістичному стилі з фокусом на функціональність та зручність використання. На сторінці розміщено логотип автосервісу, назву системи та форму авторизації на контрастному фоні, що привертає увагу користувача (див. рисунок 3.6).

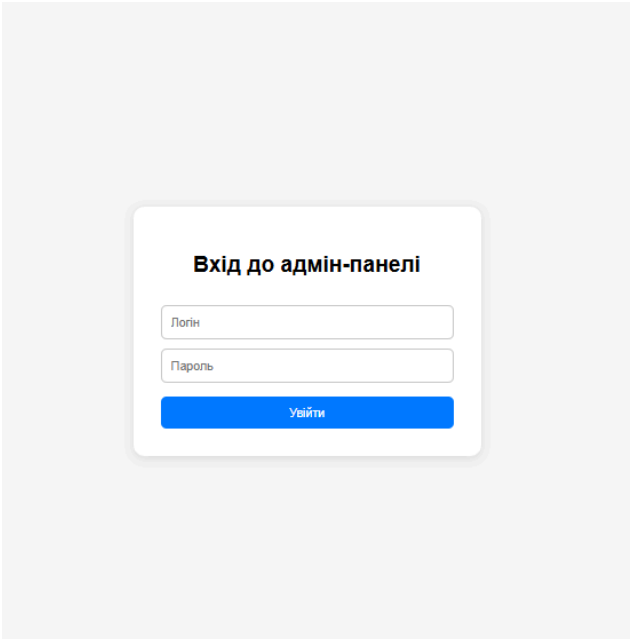


Рис. 3.6 – Сторінка авторизації для входу до адміністративної панелі

Форма авторизації містить два поля введення: для логіна (атрибут name="username") та пароля (атрибут name="password" з типом "password", який забезпечує приховування введених символів). Обидва поля мають атрибут required, що забезпечує базову валідацію на стороні клієнта та запобігає відправці форми з порожніми полями. Поля оформлені таким чином, щоб їх зручно було заповнювати на різних пристроях – від настільних комп'ютерів до мобільних телефонів. Під полями розміщена кнопка "Увійти" (тип "submit"), при натисканні на яку дані форми відправляються на сервер для перевірки.

Процес авторизації відбувається на серверній стороні у PHP-коді файлу login.php. При отриманні даних форми (перевірка умови $_SERVER["REQUEST_METHOD"] == "POST") виконується порівняння введених користувачем логіна та пароля з еталонними значеннями. У поточній реалізації використовується спрощений підхід з фіксованими значеннями логіна "autoserviceAdmin" та пароля "autoserviceAdmin". У повноцінній системі доцільно використовувати перевірку облікових даних з бази даних з використанням хешування паролів для забезпечення вищого рівня безпеки.

У разі успішної авторизації (збігу введених логіна та пароля з еталонними) створюється сесія для збереження стану авторизації. Для цього використовується механізм сесій PHP, який дозволяє зберігати дані між різними HTTP-запитами користувача. Спочатку за допомогою функції session_start() ініціалізується сесія, а

потім у масив $_SESSION записується значення "logged_in" = true, що свідчить про успішну авторизацію користувача. Після цього користувач перенаправляється на головну сторінку адміністративної панелі (admin_dashboard.php) за допомогою функції header() з параметром "Location".

У разі невдалої авторизації (неправильного логіна або пароля) користувачу відображається повідомлення про помилку. Для цього використовується змінна $error, яка заповнюється текстом повідомлення "❌ Невірний логін або пароль!" у разі невдалої авторизації. Це повідомлення відображається під формою авторизації в елементі з класом "error", який стилізований таким чином, щоб привертати увагу користувача (червоний колір тексту, напівжирний шрифт). Важливо, що повідомлення про помилку не розкриває конкретну причину невдачі (неправильний логін чи пароль), що підвищує безпеку системи, оскільки не дає потенційному зловмиснику інформації про те, яка частина облікових даних неправильна.

Для забезпечення безпеки сторінки авторизації використовуються різні механізми захисту. По-перше, це використання методу POST для відправки форми, що запобігає відображенню облікових даних в URL. По-друге, застосування механізму сесій PHP, який дозволяє зберігати стан авторизації в безпечному вигляді на сервері, а не в куках браузера. По-третє, обмеження кількості спроб авторизації з однієї IP-адреси для запобігання атакам підбору пароля. Також важливим аспектом безпеки є використання HTTPS для шифрування передачі даних між клієнтом і сервером, що запобігає перехопленню облікових даних.

Стилізація сторінки авторизації реалізована за допомогою CSS у файлі css/login.css. Для форми авторизації використовується стильний контейнер з класом "login-box", який має м'які кути (border-radius), тінь (box-shadow) та відступи (padding). Поля введення мають збільшену висоту та розмір шрифту для зручності використання, а також змінюють стиль при фокусі для візуального підтвердження активності. Кнопка "Увійти" виділена кольором, що відповідає фірмовому стилю автосервісу, та має ефект зміни стилю при наведенні для підкреслення інтерактивності.

Для зручності користувачів на сторінці авторизації реалізовані додаткові функції, такі як запам'ятовування логіна (але не пароля) для наступних сеансів роботи, перевірка стану клавіші Caps Lock при введенні пароля з відображенням

відповідного попередження, можливість відновлення забутого пароля через відправку посилання на електронну пошту. Також передбачена можливість автоматичного перенаправлення на сторінку, яку користувач намагався відвідати до авторизації, що покращує користувацький досвід.

Сторінка авторизації є важливим елементом безпеки веб-застосунку з управління автосервісом, оскільки вона контролює доступ до адміністративної частини системи, де здійснюється управління всіма аспектами роботи автосервісу. Правильна реалізація механізмів авторизації та захисту від несанкціонованого доступу є запорукою безпечної та надійної роботи всієї системи.

3.4.2. Головна панель управління

Головна панель управління є центральним елементом інтерфейсу адміністративної частини веб-застосунку з управління автосервісом, який надає доступ до всіх функціональних модулів системи та відображає ключові показники діяльності автосервісу. Ця сторінка реалізована у файлі admin_dashboard.php і доступна тільки для авторизованих користувачів. Перед відображенням вмісту сторінки виконується перевірка наявності сесії та статусу авторизації користувача (перевірка умови isset($_SESSION["logged_in"]) && $_SESSION["logged_in"] === true). Якщо користувач не авторизований, відбувається автоматичне перенаправлення на сторінку авторизації.

Дизайн головної панелі управління розроблений з урахуванням принципів інформаційної ієрархії та зручності використання. Сторінка розділена на логічні блоки, кожен з яких відповідає певній функціональній області. У верхній частині сторінки розміщені елементи навігації: логотип автосервісу, який є посиланням на головну сторінку, та головне меню з посиланнями на основні розділи системи: "Клієнти", "Автомобілі", "Замовлення", "Розклад", "Слюсарі", "Звіти", "Налаштування". Меню має горизонтальну орієнтацію на великих екранах та згортається у "бургер-меню" на мобільних пристроях для забезпечення зручності використання на різних типах пристроїв.

Центральна частина сторінки відображає інформаційні блоки з ключовими показниками діяльності автосервісу, які дозволяють керівництву швидко оцінити

поточний стан справ. Ці блоки організовані у вигляді інтерактивних карток з яскравими заголовками та числовими показниками. До ключових показників відносяться: кількість активних замовлень (зі статусами "Очікує" та "В процесі"), кількість завершених замовлень за поточний день та місяць, середній час виконання замовлень, загальна сума виручки за день та місяць, завантаженість слюсарів (відсоток використання робочого часу), найпопулярніші типи робіт за поточний період. Кожен показник супроводжується відповідною іконкою та кольоровим індикатором (зелений — позитивна динаміка, червоний — негативна), що покращує сприйняття інформації.

Під блоками з ключовими показниками розміщений розділ "Швидкі дії", який містить кнопки для найбільш часто використовуваних функцій: "Додати клієнта", "Створити замовлення", "Призначити слюсаря", "Переглянути розклад на сьогодні", "Сформувати звіт". Ці кнопки забезпечують швидкий доступ до відповідних функцій без необхідності навігації через меню, що підвищує ефективність роботи користувачів. Нижче розміщений список останніх подій та повідомлень, який інформує користувача про важливі зміни в системі: нові замовлення, зміни статусу існуючих замовлень, завершення робіт, наближення термінів виконання, низький рівень запасів запчастин тощо. У правій частині сторінки (або в нижній частині на мобільних пристроях) розміщені графіки та діаграми, які відображають ключові метрики в динаміці: кількість замовлень за останні 30 днів, виручка по днях тижня, розподіл замовлень за типами робіт, розподіл виручки за слюсарями.

3.4.3. Сторінка управління клієнтами

Сторінка управління клієнтами є важливим елементом інтерфейсу веб-застосунку з управління автосервісом, який забезпечує взаємодію користувачів з базою даних клієнтів. Ця сторінка реалізована у файлі view_clients.php і надає функціональність для перегляду, пошуку, додавання, редагування та видалення інформації про клієнтів автосервісу. Доступ до сторінки обмежений для авторизованих користувачів, що забезпечується перевіркою сесії на початку виконання PHP-скрипту.

У верхній частині сторінки розміщені елементи навігації та управління: заголовок сторінки "Управління клієнтами", кнопка "Додати нового клієнта", яка відкриває форму для створення нового запису, та форма пошуку, яка дозволяє фільтрувати список клієнтів за різними параметрами. Форма пошуку містить поля для введення критеріїв пошуку: імені клієнта, телефону, email, марки та моделі автомобіля. При введенні тексту в будь-яке з цих полів відбувається автоматична фільтрація списку клієнтів без перезавантаження сторінки, що реалізовано за допомогою JavaScript та AJAX-запитів до сервера.

Основну частину сторінки займає таблиця зі списком клієнтів, яка містить наступні колонки: ПІБ, телефон, email, інформація про автомобіль (VIN, марка, модель, рік) та колонка "Дії" з кнопками для управління записом. Таблиця має заголовки стовпців, при натисканні на які відбувається сортування списку за відповідним параметром. Наприклад, при натисканні на заголовок "ПІБ" список сортується за алфавітом імен клієнтів, при повторному натисканні - в зворотному порядку. Таке інтерактивне сортування реалізовано за допомогою JavaScript та AJAX, що дозволяє змінювати порядок відображення даних без перезавантаження сторінки (рис.3.7).

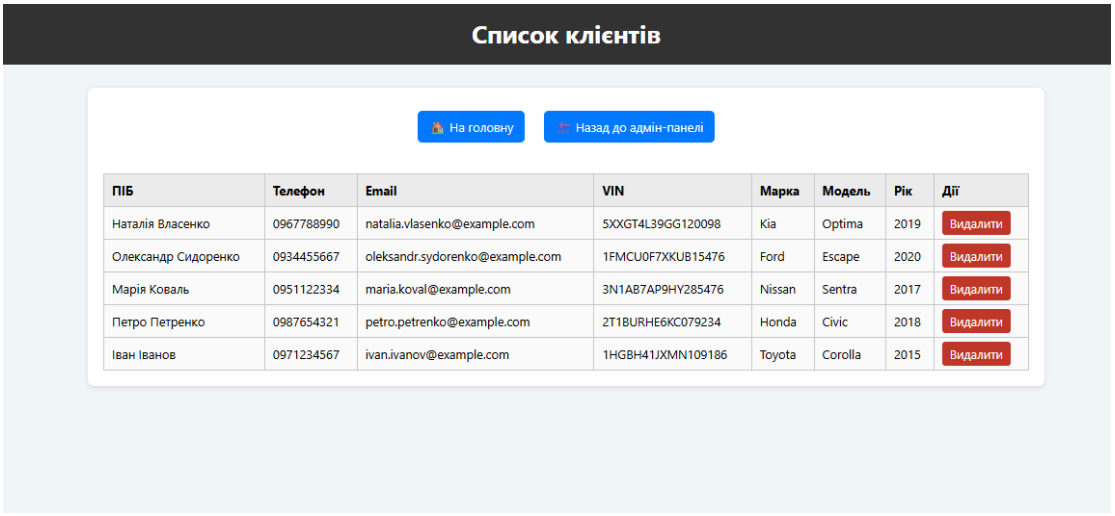


Рис. 3.7 – Таблиця зі списком клієнтів та їх автомобілів з можливістю видалення записів

Кожен рядок таблиці відповідає одному клієнту та містить його основні дані. У колонці "Дії" розміщені кнопки для управління записом: "Переглянути" (відкриває сторінку з детальною інформацією про клієнта та історією його обслуговування),

"Редагувати" (відкриває форму для зміни даних клієнта), "Видалити" (видаляє запис про клієнта після підтвердження). Кнопки оформлені у вигляді іконок з підказками, що з'являються при наведенні курсора, що робить інтерфейс інтуїтивно зрозумілим та економить місце в таблиці.

Для клієнтів, які мають кілька автомобілів, в таблиці відображається інформація про один автомобіль, а при натисканні на спеціальний індикатор розкривається список всіх автомобілів клієнта. Це реалізовано за допомогою вкладених рядків таблиці та JavaScript-коду, який керує їх відображенням. Такий підхід дозволяє компактно представити інформацію про всі автомобілі клієнта, не перевантажуючи основну таблицю.

Під таблицею розміщені елементи пагінації, які дозволяють переходити між сторінками списку клієнтів, якщо їх кількість перевищує встановлений ліміт (наприклад, 20 записів на сторінку). Пагінація включає кнопки переходу до першої, попередньої, наступної та останньої сторінки, а також індикатор поточної сторінки та загальної кількості сторінок. Перехід між сторінками відбувається без перезавантаження всього вмісту сторінки, що реалізовано за допомогою AJAX-запитів, які отримують тільки нову порцію даних для відображення.

Для додавання нового клієнта використовується модальне вікно з формою, яке відкривається при натисканні на кнопку "Додати нового клієнта". Форма містить поля для введення всіх необхідних даних про клієнта (ПІБ, телефон, email) та його автомобіль (VIN, марка, модель, рік випуску). Всі поля мають валідацію на стороні клієнта, яка перевіряє коректність введених даних перед відправкою форми на сервер. Наприклад, для поля телефону перевіряється відповідність формату українського номера, для email - відповідність стандартному формату електронної пошти, для VIN - відповідність стандартному формату VIN-коду (17 символів, що складаються з цифр та літер, за винятком літер I, O, Q) (див. рисунок 3.8).

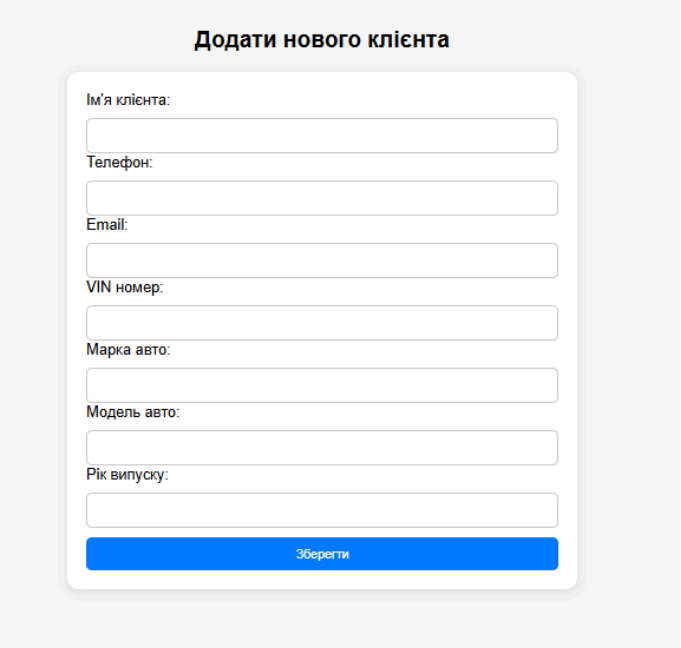


Рис. 3.8 – Інтерфейс форми додавання нового клієнта.

Для редагування даних клієнта використовується аналогічне модальне вікно з формою, яке відкривається при натисканні на кнопку "Редагувати" для конкретного запису. Форма заповнюється поточними даними клієнта, які користувач може змінити. Валідація даних відбувається так само, як і при додаванні нового клієнта. Після збереження змін оновлені дані відображаються в таблиці без перезавантаження сторінки, що реалізовано за допомогою AJAX-запитів та маніпуляцій з DOM-деревом за допомогою JavaScript.

Сторінка управління клієнтами також включає функціональність для експорту даних у різні формати: CSV, Excel, PDF. Це дозволяє користувачам створювати резервні копії даних, готувати звіти та обмінюватися інформацією з іншими системами. Експорт даних реалізований за допомогою спеціалізованих PHP-бібліотек, таких як PHPExcel для створення файлів Excel та FPDF для створення файлів PDF. Користувач може вибрати, які колонки включати в експортований файл, а також застосувати фільтри для експорту тільки певної підмножини записів.

3.4.4. Сторінка управління автомобілями

Сторінка управління автомобілями є спеціалізованим інтерфейсом для роботи з інформацією про транспортні засоби клієнтів автосервісу. Ця сторінка реалізована у файлі view_vehicles.php і забезпечує функціональність для перегляду, пошуку, додавання, редагування та видалення інформації про автомобілі, а також для відстеження історії їх обслуговування. Як і інші сторінки адміністративної частини, вона доступна тільки для авторизованих користувачів, що перевіряється на початку виконання PHP-скрипту.

Структура сторінки управління автомобілями подібна до структури сторінки управління клієнтами, але з фокусом на інформацію про транспортні засоби. У верхній частині сторінки розміщені заголовок "Управління автомобілями", кнопка "Додати новий автомобіль" та форма пошуку з полями для фільтрації за різними параметрами: VIN-код, марка, модель, рік випуску, ім'я власника. Пошук працює в режимі реального часу за допомогою AJAX-запитів, які відправляються при введенні тексту в будь-яке з полів форми.

Основну частину сторінки займає таблиця зі списком автомобілів, яка містить наступні колонки: VIN-код, марка, модель, рік випуску, ім'я власника, контактний телефон власника та колонка "Дії" з кнопками для управління записом. Таблиця має інтерактивні заголовки стовпців, які дозволяють сортувати список за відповідними параметрами.

Особливістю цієї таблиці є візуальне представлення додаткової інформації про автомобіль за допомогою кольорових індикаторів: зелений — для автомобілів, які мають актуальне технічне обслуговування, жовтий — для автомобілів, яким скоро потрібно проходити ТО, червоний — для автомобілів з простроченим ТО. Для кожного автомобіля в колонці "Дії" доступні кнопки: "Переглянути" (відкриває сторінку з детальною інформацією про автомобіль та історією його обслуговування), "Редагувати" (відкриває форму для зміни даних автомобіля), "Сервісна книжка" (відкриває сторінку з електронною сервісною книжкою автомобіля), "Видалити" (видаляє запис про автомобіль після підтвердження). Кнопки оформлені у вигляді іконок з підказками, що з'являються при наведенні курсора, для забезпечення інтуїтивно зрозумілого інтерфейсу.

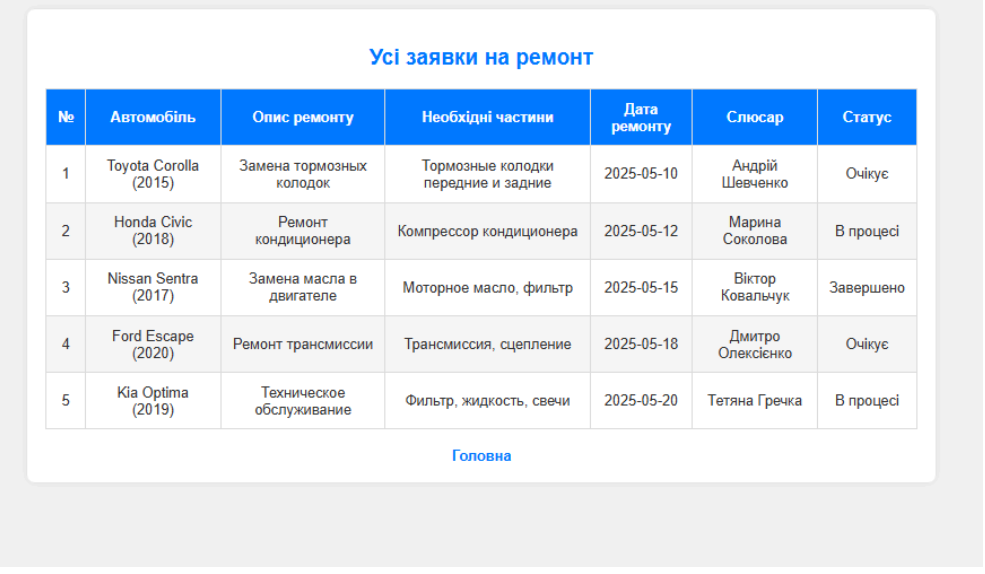
Для додавання нового автомобіля використовується модальне вікно з формою, яке відкривається при натисканні на кнопку "Додати новий автомобіль". Форма

містить випадаючий список для вибору власника (клієнта) та поля для введення інформації про автомобіль: VIN-код, марка, модель, рік випуску, а також додаткові технічні характеристики, такі як тип двигуна, об'єм двигуна, тип коробки передач, тип кузова тощо. Поле VIN-код має спеціальну функцію автоматичного декодування, яка при введенні валідного VIN-коду автоматично заповнює поля марки, моделі та року випуску на основі інформації, отриманої з зовнішнього API. Сторінка детального перегляду інформації про автомобіль (vehicle_details.php) відкривається при натисканні на кнопку "Переглянути" і містить розширену інформацію про автомобіль, включаючи технічні характеристики, фотографії, історію обслуговування, рекомендації щодо планового ТО.

3.4.5. Сторінка управління замовленнями

Сторінка управління замовленнями є ключовим інтерфейсом веб-застосунку з управління автосервісом, оскільки вона забезпечує контроль за основним бізнес-процесом – виконанням замовлень на ремонт та технічне обслуговування автомобілів.

Ця сторінка реалізована у файлі view_repairs.php і надає функціональність для перегляду, фільтрації, створення, редагування та видалення замовлень, а також для відстеження їх статусу та контролю виконання (див. рисунок 3.9).



Усі заявки на ремонт

№	Автомобіль	Опис ремонту	Необхідні частини	Дата ремонту	Слюсар	Статус
1	Toyota Corolla (2015)	Замена тормозных колодок	Тормозные колодки передние и задние	2025-05-10	Андрій Шевченко	Очікує
2	Honda Civic (2018)	Ремонт кондиционера	Компрессор кондиционера	2025-05-12	Марина Соколова	В процесі
3	Nissan Sentra (2017)	Замена масла в двигателе	Моторное масло, фильтр	2025-05-15	Віктор Ковальчук	Завершено
4	Ford Escape (2020)	Ремонт трансмиссии	Трансмиссия, сцепление	2025-05-18	Дмитро Олексієнко	Очікує
5	Kia Optima (2019)	Техническое обслуживание	Фильтр, жидкость, свечи	2025-05-20	Тетяна Гречка	В процесі

Головна

Рис. 3.9 – Таблиця перегляду заявок на ремонт із зазначенням авто, деталей, дати та статусу виконання

У верхній частині сторінки розміщені елементи навігації та управління: заголовок "Управління замовленнями", кнопка "Створити нове замовлення" та розширена форма пошуку, яка дозволяє фільтрувати список замовлень за різними параметрами: номер замовлення, дата створення, статус, клієнт, автомобіль, тип робіт, слюсар.

Форма пошуку реалізована з використанням випадаючих списків та полів введення з автодоповненням, що спрощує процес фільтрації. При зміні будь-якого параметра пошуку таблиця замовлень автоматично оновлюється за допомогою AJAX-запитів, що забезпечує швидкий відгук інтерфейсу без перезавантаження сторінки.

Основну частину сторінки займає таблиця зі списком замовлень, яка містить наступні колонки: номер замовлення, дата створення, клієнт, автомобіль (марка, модель), короткий опис робіт, статус, призначений слюсар, планована дата завершення та колонка "Дії" з кнопками для управління замовленням. Таблиця має інтерактивні заголовки стовпців для сортування та використовує кольорове кодування для візуального виділення замовлень з різними статусами: жовтий для "Очікує", синій для "В процесі", зелений для "Завершено", червоний для замовлень з простроченим терміном виконання.

Для кожного замовлення в колонці "Дії" доступні кнопки: "Переглянути" (відкриває сторінку з детальною інформацією про замовлення), "Редагувати" (відкриває форму для зміни параметрів замовлення), "Змінити статус" (дозволяє швидко змінити статус замовлення через випадаючий список), "Друк документів" (відкриває меню для вибору та друку документів, пов'язаних з замовленням), "Видалити" (видаляє замовлення після підтвердження). Кнопки оформлені у вигляді іконок з підказками для забезпечення інтуїтивно зрозумілого інтерфейсу (див. рисунок 3.10).

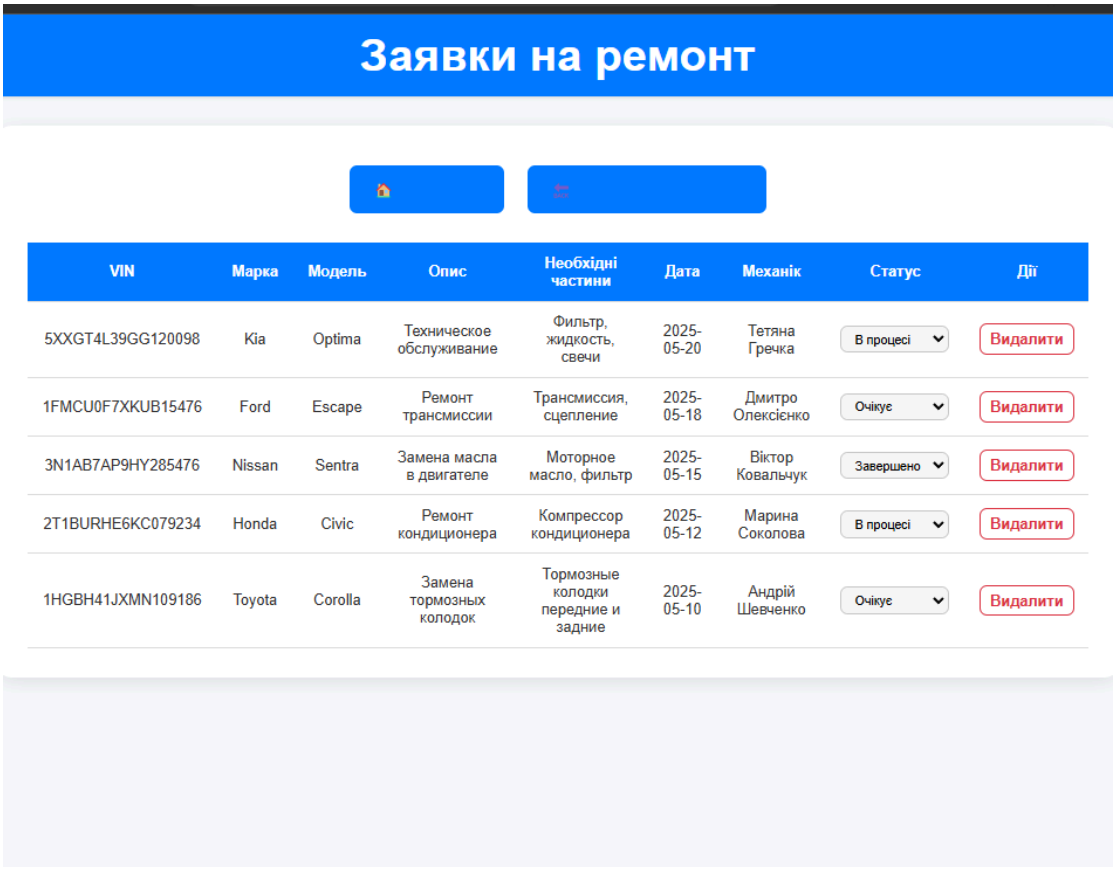


Рис. 3.10 – Інтерфейс керування заявками на ремонт із можливістю зміни статусу та видалення

Сторінка детального перегляду замовлення (repair_details.php) відкривається при натисканні на кнопку "Переглянути" і містить повну інформацію про замовлення, розділену на логічні блоки: дані про клієнта та автомобіль, опис проблеми, результати діагностики, перелік необхідних робіт та запчастин з вартістю, призначений слюсар, поточний статус, історія змін статусу, коментарі працівників автосервісу. Для кожного блоку передбачені можливості редагування та додавання нової інформації безпосередньо на цій сторінці, що дозволяє оперативно оновлювати дані про хід виконання замовлення.

Особливу увагу на сторінці управління замовленнями приділено функціональності відстеження статусу замовлення та контролю його виконання. Для цього реалізована система статусів, яка відображає поточний етап виконання замовлення: "Прийнято", "Діагностика", "Очікування запчастин", "В роботі", "Готово", "Видано". При зміні статусу в системі фіксується дата та час зміни, а також користувач, який виконав зміну, що дозволяє відстежувати всю історію виконання замовлення.

Додатково реалізована система сповіщень, яка автоматично надсилає повідомлення відповідальним особам при критичних змінах статусу або при наближенні терміну виконання замовлення.

Для ефективного управління замовленнями реалізована функціональність групових операцій, яка дозволяє виконувати дії над кількома замовленнями одночасно.

Користувач може вибрати кілька замовлень за допомогою чекбоксів у таблиці та виконати над ними групову операцію: змінити статус, призначити слюсаря, запланувати на певну дату, надіслати повідомлення клієнтам, експортувати дані. Ця функціональність особливо корисна для масової обробки замовлень при зміні робочих змін, плануванні завантаження на наступний день, закритті виконаних замовлень.

Сторінка управління замовленнями також включає функціональність для аналізу ефективності виконання замовлень та контролю якості роботи автосервісу. Для цього реалізовані інструменти для моніторингу ключових показників: середній час виконання замовлень різних типів, відсоток замовлень, виконаних у строк, кількість повторних звернень клієнтів з тією ж проблемою, рейтинг задоволеності клієнтів. Ці дані відображаються у вигляді графіків та діаграм на спеціальній панелі аналітики, доступній з головного меню сторінки управління замовленнями.

Для оптимізації процесу виконання замовлень реалізована інтеграція з іншими модулями системи, зокрема з модулем управління запчастинами та модулем управління персоналом. При створенні або редагуванні замовлення система автоматично перевіряє наявність необхідних запчастин на складі та доступність слюсарів потрібної спеціалізації на вказану дату. Якщо виявлені обмеження (відсутність запчастин або завантаженість персоналу), система відображає відповідні попередження та пропонує альтернативні варіанти, наприклад, іншу дату або можливість замовлення відсутніх запчастин у постачальників.

3.4.6. Сторінка планування розкладу

Сторінка планування розкладу є спеціалізованим інтерфейсом для візуалізації та управління розподілом робіт між слюсарями та ремонтними постами автосервісу.

Ця сторінка реалізована у файлі schedule.php і забезпечує функціональність для перегляду поточного розкладу, планування нових робіт, редагування існуючих записів, а також для оптимізації використання ресурсів автосервісу. Центральним елементом сторінки планування розкладу є інтерактивний календар, який відображає заплановані роботи в різних режимах: день, тиждень, місяць. Календар реалізований з використанням JavaScript-бібліотеки FullCalendar, яка забезпечує багатий функціонал для роботи з подіями.

У верхній частині сторінки розміщені елементи управління календарем: перемикачі режимів відображення (день, тиждень, місяць), кнопки навігації по датах (вперед, назад, сьогодні), випадаючий список для вибору типу ресурсів для відображення (всі слюсарі, конкретний слюсар, всі пости, конкретний пост), поле для вибору дати, кнопка "Додати нову роботу".

Ці елементи дозволяють користувачу швидко налаштувати відображення календаря відповідно до своїх потреб та отримати доступ до функцій управління розкладом. По вертикалі календар розділений на рядки, які відповідають ресурсам (слюсарі або ремонтні пости), а по горизонталі — на колонки, які відповідають часовим інтервалам (години для режиму "день", дні для режиму "тиждень", дні для режиму "місяць" з групуванням подій).

Кожна робота в календарі відображається у вигляді блоку з інформацією про замовлення: номер, клієнт, автомобіль, тип робіт. Колір блоку відповідає статусу роботи: жовтий для запланованих, синій для робіт у процесі, зелений для завершених. При наведенні курсора на блок з'являється спливаюча підказка з детальною інформацією про роботу, а при кліку — контекстне меню з опціями: "Переглянути деталі", "Редагувати", "Змінити статус", "Перепризначити", "Видалити".

Ці опції дозволяють швидко виконувати основні операції з управління роботами безпосередньо з календаря. Для планування нової роботи користувач може або натиснути на кнопку "Додати нову роботу", або клікнути на вільний слот в календарі.

В обох випадках відкривається модальне вікно з формою для створення нової роботи, яка містить поля для вибору клієнта та автомобіля, опису робіт, вибору слюсаря, вказання дати та часу початку та завершення, вибору ремонтного поста.

Особливою функцією сторінки планування розкладу є можливість перетягування блоків робіт (drag-and-drop) для зміни їх часу, тривалості або призначеного ресурсу. Користувач може перетягнути блок роботи на інший час або день, змінити його тривалість шляхом перетягування нижньої межі блоку, або перемістити роботу від одного слюсаря до іншого.

При перетягуванні блоку система в реальному часі перевіряє доступність вибраного ресурсу на новий час та відображає підказки про можливі конфлікти.

Після завершення перетягування зміни автоматично зберігаються в базі даних через AJAX-запит, що забезпечує миттєве оновлення розкладу без перезавантаження сторінки.

Для оптимізації використання ресурсів автосервісу реалізована функціональність автоматичного планування робіт.

Користувач може вибрати опцію "Оптимізувати розклад", і система проаналізує поточний розклад та запропонує оптимальний варіант розподілу робіт з урахуванням пріоритетності замовлень, кваліфікації слюсарів, технологічних вимог до послідовності операцій тощо.

РОЗДІЛ 4. ТЕСТУВАННЯ ТА ВПРОВАДЖЕННЯ ВЕБ-ЗАСТОСУНКУ

4.1. Методологія тестування

Методологія тестування веб-застосунку з управління автосервісом розроблена з метою забезпечення високої якості кінцевого продукту, виявлення та усунення дефектів на ранніх стадіях розробки, а також підтвердження відповідності системи функціональним та нефункціональним вимогам. При розробці методології враховувалися особливості предметної області автосервісу, обрані технології розробки (PHP, MySQL, HTML, CSS), а також обмеження проекту за часом та ресурсами. Тестування проводилося на всіх етапах життєвого циклу розробки, починаючи від аналізу вимог і закінчуючи впровадженням системи в експлуатацію.

Процес тестування веб-застосунку був структурований відповідно до рівнів тестування: модульне тестування (unit testing), інтеграційне тестування, системне тестування та приймальне тестування. Модульне тестування виконувалося розробниками в процесі створення окремих компонентів системи і було спрямоване на перевірку коректності роботи кожного модуля окремо. Інтеграційне тестування проводилося після об'єднання окремих модулів і було спрямоване на виявлення дефектів взаємодії між компонентами. Системне тестування перевіряло роботу системи в цілому на відповідність функціональним та нефункціональним вимогам. Приймальне тестування проводилося за участю представників замовника для підтвердження готовності системи до впровадження в реальне робоче середовище.

Для забезпечення систематичного підходу до тестування був розроблений план тестування, який включав графік проведення тестів, розподіл відповідальності, критерії входу та виходу для кожного етапу, а також методи та інструменти тестування. План тестування був узгоджений з загальним планом проекту та враховував залежності між різними етапами розробки. Для кожного модуля системи були розроблені тест-кейси, які охоплювали всі можливі сценарії використання та включали позитивні та негативні тести, тести граничних значень, тести продуктивності тощо.

У рамках методології тестування були визначені основні типи тестів, які застосовувалися до веб-застосунку: функціональне тестування, тестування

користувацького інтерфейсу, тестування продуктивності, тестування безпеки, тестування доступності, тестування сумісності з різними браузерами та пристроями. Кожен тип тестування мав свою специфіку та був спрямований на виявлення певних категорій дефектів. Для автоматизації процесу тестування використовувалися різні інструменти та фреймворки: PHPUnit для модульного тестування PHP-коду, Selenium WebDriver для тестування користувацького інтерфейсу, Apache JMeter для навантажувального тестування, OWASP ZAP для тестування безпеки.

Важливою складовою методології тестування було регресійне тестування, яке проводилося після внесення змін у код для перевірки, що нова функціональність не порушила роботу існуючих компонентів. Для ефективного проведення регресійного тестування були визначені набори тест-кейсів, які охоплювали ключову функціональність системи, та автоматизовані за допомогою відповідних інструментів [25]. Регресійне тестування проводилося як після виправлення дефектів, так і після додавання нових функцій, що дозволяло підтримувати високу якість коду протягом всього життєвого циклу розробки.

4.2. Тестування функціональності

Тестування функціональності веб-застосунку з управління автосервісом проводилося з метою перевірки відповідності системи функціональним вимогам, визначеним на етапі аналізу та проектування. Цей вид тестування фокусувався на перевірці коректності роботи всіх функцій системи, включаючи обробку даних, взаємодію з базою даних, логіку бізнес-процесів та інтеграцію між різними модулями. Функціональне тестування проводилося на різних рівнях: модульне тестування окремих компонентів, інтеграційне тестування взаємодії між компонентами та системне тестування веб-застосунку в цілому.

Першим етапом функціонального тестування було модульне тестування окремих компонентів системи. Для кожного модуля (управління клієнтами, управління автомобілями, управління замовленнями на ремонт, планування розкладу робіт, призначення слюсарів тощо) були розроблені набори тест-кейсів, які перевіряли коректність роботи всіх функцій модуля. Наприклад, для модуля управління клієнтами були створені тести для перевірки функцій додавання нового

клієнта, редагування інформації про клієнта, видалення клієнта, пошуку клієнтів за різними критеріями. Тести включали як позитивні сценарії (коректне виконання операцій), так і негативні (обробка неправильних вхідних даних, обробка помилок).

Після успішного проходження модульних тестів проводилося інтеграційне тестування, яке перевіряло взаємодію між різними модулями системи. Інтеграційні тести фокусувалися на перевірці коректності передачі даних між модулями, узгодженості інтерфейсів та правильності сумісної роботи компонентів. Наприклад, тестувалася взаємодія між модулем управління замовленнями та модулем планування розкладу при створенні нового замовлення: правильність відображення замовлення в розкладі, коректність розрахунку часу виконання, правильність призначення слюсаря тощо.

Особлива увага при функціональному тестуванні приділялася перевірці бізнес-логіки системи, яка відображає специфічні процеси роботи автосервісу. Були розроблені тест-кейси для перевірки таких аспектів, як правильність розрахунку вартості ремонту на основі використаних запчастин та виконаних робіт, коректність процесу зміни статусу замовлення, правильність розподілу робіт між слюсарями з урахуванням їх кваліфікації та завантаженості, коректність формування звітів та аналітичних даних. Ці тести виконувалися як з використанням автоматизованих інструментів, так і вручну, для забезпечення повного покриття всіх аспектів бізнес-логіки [26].

Важливим аспектом функціонального тестування була перевірка взаємодії системи з базою даних. Були розроблені тести для перевірки коректності SQL-запитів, правильності обробки результатів запитів, дотримання цілісності даних при виконанні операцій вставки, оновлення та видалення, ефективності індексів та оптимізації запитів. Для тестування взаємодії з базою даних використовувалася тестова база даних, яка була структурно ідентична робочій, але містила контрольовані тестові дані. Перед кожним запуском тестів база даних приводилася до відомого початкового стану, що забезпечувало повторюваність результатів тестування.

4.3. Тестування користувацького інтерфейсу

Тестування користувацького інтерфейсу веб-застосунку з управління автосервісом проводилося з метою перевірки зручності використання системи, коректності відображення елементів інтерфейсу, адаптивності до різних пристроїв та відповідності дизайну вимогам та очікуванням користувачів. Цей вид тестування є критично важливим, оскільки користувацький інтерфейс є точкою взаємодії між системою та користувачами, і його якість безпосередньо впливає на загальне враження від роботи з системою, ефективність виконання завдань та задоволеність користувачів.

Першим етапом тестування користувацького інтерфейсу була перевірка коректності відображення елементів інтерфейсу у різних браузерах та на різних пристроях. Були проведені тести на сумісність з найбільш популярними браузерами (Google Chrome, Mozilla Firefox, Safari, Microsoft Edge) та операційними системами (Windows, macOS, iOS, Android). Для кожної комбінації браузера та операційної системи перевірялася коректність відображення всіх сторінок системи, правильність роботи JavaScript-функцій, коректність стилів CSS, правильність відображення шрифтів та зображень. Особлива увага приділялася тестуванню адаптивності дизайну, тобто здатності інтерфейсу коректно перебудовуватися та відображатися на екранах різних розмірів, від настільних комп'ютерів до мобільних телефонів.

Важливим аспектом тестування користувацького інтерфейсу була перевірка зручності використання (usability testing), яка проводилася з залученням реальних користувачів різних ролей: адміністраторів, менеджерів, слюсарів. Користувачам пропонувалося виконати типові завдання в системі (створити нового клієнта, зареєструвати замовлення на ремонт, призначити слюсаря, переглянути розклад тощо) та оцінити зручність виконання цих завдань, зрозумілість інтерфейсу, логічність розташування елементів, інформативність повідомлень та підказок. Під час тестування фіксувалися час виконання завдань, кількість помилок, коментарі та пропозиції користувачів. Ці дані аналізувалися для виявлення проблем зручності використання та розробки рекомендацій щодо покращення інтерфейсу.

Особлива увага при тестуванні користувацького інтерфейсу приділялася перевірці форм введення даних, які є ключовим елементом взаємодії користувача з системою. Для кожної форми перевірялася коректність валідації даних на стороні клієнта, зрозумілість повідомлень про помилки, зручність навігації між полями за

допомогою клавіатури, коректність роботи автодоповнення та інших допоміжних функцій. Наприклад, для форми додавання нового клієнта перевірялася валідація обов'язкових полів, формату телефонного номера та email, унікальності VIN-коду автомобіля, а також коректність відображення повідомлень про помилки та підказок щодо формату введення даних.

Важливим аспектом тестування користувацького інтерфейсу була перевірка доступності (accessibility testing), яка спрямована на забезпечення можливості користування системою людьми з обмеженими можливостями. Були проведені тести на відповідність стандартам доступності WCAG 2.1 (Web Content Accessibility Guidelines), які включали перевірку контрастності кольорів, наявності альтернативного тексту для зображень, можливості навігації за допомогою клавіатури, сумісності з програмами читання з екрану, коректності структури заголовків та інших аспектів доступності [27]. Для автоматизації цього процесу використовувалися спеціалізовані інструменти, такі як WAVE (Web Accessibility Evaluation Tool) та Axe, які дозволяли сканувати веб-сторінки на наявність проблем доступності.

4.4. Виправлення помилок та оптимізація

Процес виправлення помилок та оптимізації веб-застосунку з управління автосервісом є важливим етапом розробки, який забезпечує високу якість кінцевого продукту, його надійність, продуктивність та зручність використання. Цей процес базувався на результатах різних видів тестування, аналізі зворотного зв'язку від користувачів та моніторингу роботи системи в реальних умовах. Метою цього етапу було не лише усунення виявлених дефектів, але й покращення загальної якості коду, оптимізація роботи з базою даних, підвищення продуктивності системи та удосконалення користувацького інтерфейсу.

Для ефективного управління процесом виправлення помилок використовувалася система відстеження дефектів (bug tracking system) на базі інструменту Mantis Bug Tracker. Кожен виявлений дефект фіксувався в системі з детальним описом, кроками відтворення, очікуваним та фактичним результатами, серйозністю, пріоритетом та іншими атрибутами. Дефекти класифікувалися за

типами (функціональні, інтерфейсні, продуктивності, безпеки тощо) та розподілялися між розробниками відповідно до їх спеціалізації та завантаженості. Такий підхід дозволяв організувати систематичний процес виправлення помилок, контролювати його прогрес та забезпечувати, що всі виявлені дефекти будуть належним чином оброблені.

При виправленні функціональних помилок важливо було не лише усунути симптоми проблеми, але й виявити та виправити її корінну причину. Для цього використовувався підхід глибокого аналізу дефекту, який включав вивчення коду, логування проміжних результатів, аналіз стеків викликів та використання інструментів налагодження (debugging). Наприклад, при виправленні помилки з некоректним розрахунком вартості ремонту аналізувалася не лише функція розрахунку, але й усі пов'язані з нею компоненти: отримання даних про запчастини та роботи, застосування знижок та податків, округлення результатів тощо. Такий підхід дозволяв виявити не лише конкретну помилку, але й потенційні проблеми в суміжних областях.

Оптимізація роботи з базою даних була важливим напрямком покращення веб-застосунку, оскільки ефективність взаємодії з базою даних безпосередньо впливає на продуктивність системи в цілому. Були проведені заходи з оптимізації SQL-запитів, включаючи аналіз та покращення складних запитів, оптимізацію індексів, кешування результатів часто виконуваних запитів. Для аналізу ефективності запитів використовувався інструмент EXPLAIN, який дозволяв отримати інформацію про план виконання запиту, використання індексів, кількість сканованих рядків тощо. На основі цього аналізу були визначені запити, які потребували оптимізації, та розроблені більш ефективні варіанти цих запитів.

Оптимізація продуктивності клієнтської частини веб-застосунку була спрямована на покращення швидкості завантаження сторінок, зменшення часу до інтерактивності (time to interactive), оптимізацію роботи JavaScript-коду та CSS-стилів. Були проведені заходи з мінімізації та об'єднання CSS та JavaScript файлів, оптимізації зображень, впровадження ледачого завантаження (lazy loading) для ресурсів, які не потрібні при початковому відображенні сторінки. Для аналізу продуктивності клієнтської частини використовувалися інструменти, такі як Google PageSpeed Insights та Chrome DevTools Performance, які дозволяли виявити проблеми

продуктивності та розробити рекомендації щодо їх вирішення [28]. Наприклад, було виявлено, що сторінка планування розкладу робіт завантажує всі JavaScript-бібліотеки при початковому відображенні, що збільшувало час завантаження. Це було оптимізовано шляхом впровадження асинхронного завантаження бібліотек та відкладеного виконання некритичного JavaScript-коду.

4.5. Рекомендації щодо впровадження та використання

Успішне впровадження та ефективне використання веб-застосунку з управління автосервісом вимагає системного підходу, який враховує технічні, організаційні та людські фактори. На основі досвіду розробки, тестування та попереднього впровадження системи були сформульовані рекомендації, які допоможуть забезпечити плавний перехід на нову систему, її ефективне використання персоналом автосервісу та отримання максимальної віддачі від інвестицій у автоматизацію бізнес-процесів.

Перш за все, рекомендується провести ретельну підготовку до впровадження системи, яка включає аналіз існуючих бізнес-процесів автосервісу, визначення точок інтеграції нової системи з існуючими процесами та системами, підготовку технічної інфраструктури та навчання персоналу. Важливо провести аудит поточних процесів управління клієнтами, автомобілями, замовленнями на ремонт, планування робіт, щоб виявити можливі невідповідності між цими процесами та функціональністю нової системи. На основі цього аналізу може бути прийнято рішення про адаптацію системи до існуючих процесів або, навпаки, про зміну процесів для більш ефективного використання можливостей системи. Також необхідно визначити, які дані з існуючих систем або паперових документів необхідно буде перенести в нову систему, та розробити план міграції даних.

Технічне впровадження системи рекомендується проводити поетапно, починаючи з розгортання системи на тестовому середовищі для остаточної перевірки її функціональності та відповідності вимогам конкретного автосервісу. На цьому етапі важливо провести налаштування системи відповідно до специфіки автосервісу: налаштування довідників (типи робіт, марки та моделі автомобілів, види запчастин), налаштування тарифів на роботи та послуги, налаштування шаблонів

документів (акти, рахунки, сервісні книжки), налаштування прав доступу для різних категорій користувачів. Після завершення налаштування та тестування можна переходити до розгортання системи на робочому середовищі, при цьому рекомендується забезпечити можливість швидкого повернення до попередньої системи у разі виникнення критичних проблем.

Навчання персоналу є критично важливим фактором успішного впровадження системи. Рекомендується розробити програму навчання, яка включає загальне ознайомлення з системою для всіх співробітників та спеціалізовані навчальні модулі для різних категорій користувачів (адміністратори, менеджери, слюсарі, бухгалтери). Навчання може проводитися у формі групових семінарів, індивідуальних консультацій, навчальних відео, письмових інструкцій. Важливо не лише навчити персонал використовувати функції системи, але й пояснити переваги нової системи, її вплив на бізнес-процеси автосервісу, очікувані результати від її впровадження. Це допоможе подолати можливий опір змінам та мотивувати співробітників до активного використання системи. Також рекомендується призначити "чемпіонів" системи серед співробітників — людей, які швидко освоюють нову систему та можуть допомагати іншим у її використанні.

Після початку використання системи в робочому режимі рекомендується організувати період активної підтримки, протягом якого забезпечується оперативна допомога користувачам у вирішенні проблем, відповіді на питання, виправлення помилок та внесення необхідних корективів у налаштування системи. Для цього може бути організована "гаряча лінія" підтримки, система збору та обробки звернень користувачів, регулярні зустрічі для обговорення досвіду використання системи. Також важливо організувати моніторинг роботи системи для раннього виявлення можливих технічних проблем: продуктивності, доступності, цілісності даних [29]. Для забезпечення безпеки даних та стабільної роботи системи рекомендується впровадити регулярне резервне копіювання бази даних та файлів системи, а також розробити план відновлення у разі аварійних ситуацій. Резервне копіювання повинно проводитися з періодичністю, яка відповідає інтенсивності змін даних в системі, та включати повне копіювання бази даних, файлів системи, налаштувань сервера.

ВИСНОВКИ

У результаті виконання дипломної роботи було розроблено веб-застосунок для управління автосервісом, який забезпечує автоматизацію ключових бізнес-процесів підприємства та створює ефективну інформаційну систему для підтримки прийняття управлінських рішень.

Система відповідає поставленим вимогам щодо функціональності, продуктивності, безпеки та зручності використання, а також враховує специфіку діяльності автосервісів в Україні.

В ході дослідження було проаналізовано сучасний стан автосервісної галузі та виявлено основні потреби в автоматизації бізнес-процесів. Встановлено, що ефективне управління автосервісом потребує комплексного підходу, який охоплює всі аспекти діяльності: від роботи з клієнтами до контролю виконання технічних робіт та обліку запчастин.

Проведений огляд існуючих рішень для управління автосервісом дозволив виявити їх переваги та недоліки, що було враховано при розробці власного веб-застосунку.

На етапі проектування веб-застосунку було детально проаналізовано бізнес-процеси автосервісу, сформовано функціональні вимоги до системи, розроблено концептуальну, логічну та фізичну моделі бази даних, а також спроектовано користувацький інтерфейс з урахуванням принципів зручності використання та адаптивного дизайну.

Розроблена архітектура системи забезпечує модульність, масштабованість та гнучкість, що дозволяє легко адаптувати систему до конкретних потреб підприємства та розширювати її функціональність у майбутньому.

Реалізація веб-застосунку базувалася на сучасних технологіях веб-розробки: PHP для серверної частини, MySQL для бази даних, HTML, CSS та JavaScript для клієнтської частини. Такий технологічний стек забезпечує оптимальне співвідношення функціональності, продуктивності та вартості розробки.

Розроблена система включає всі необхідні функціональні модулі: управління клієнтами, управління автомобілями, управління замовленнями на ремонт, планування розкладу робіт, призначення слюсарів, формування звітності. Кожен

модуль забезпечує повний цикл операцій CRUD (Create, Read, Update, Delete) та інтегрований з іншими модулями системи.

Особлива увага в роботі була приділена тестуванню веб-застосунку. Розроблена методологія тестування включала різні види та рівні тестів: модульне тестування окремих компонентів, інтеграційне тестування взаємодії між компонентами, функціональне тестування, тестування користувацького інтерфейсу, тестування продуктивності та безпеки. За результатами тестування було виявлено та усунено ряд дефектів, а також проведено оптимізацію роботи системи, що дозволило забезпечити її стабільну та ефективну роботу.

На основі результатів розробки та тестування були сформульовані рекомендації щодо впровадження та використання веб-застосунку в реальних умовах роботи автосервісу.

Ці рекомендації охоплюють технічні аспекти розгортання системи, міграцію даних з існуючих систем, навчання персоналу, забезпечення безпеки даних, а також подальший розвиток та підтримку системи. Дотримання цих рекомендацій дозволить забезпечити успішне впровадження та ефективне використання розробленого веб-застосунку.

Практична значимість роботи полягає в тому, що розроблений веб-застосунок є готовим до впровадження рішенням, яке може бути використане автосервісами різного масштабу для автоматизації бізнес-процесів та підвищення ефективності діяльності.

Система має модульну структуру, що дозволяє гнучко налаштовувати її функціональність відповідно до потреб конкретного підприємства, а відкритий код забезпечує можливість подальшого розвитку та модифікації системи.

Результати дослідження показують, що впровадження розробленого веб-застосунку дозволяє досягти значних покращень у діяльності автосервісу: підвищити якість обслуговування клієнтів, оптимізувати використання ресурсів підприємства, забезпечити повний контроль над усіма аспектами діяльності, підвищити прозорість бізнес-процесів та забезпечити керівництво актуальною аналітичною інформацією для прийняття управлінських рішень.

У перспективі подальшого розвитку веб-застосунку можливе розширення його функціональності за рахунок додавання нових модулів, таких як мобільний додаток

для клієнтів, система лояльності, інтеграція з бухгалтерськими системами, онлайн-каталог запчастин, а також впровадження технологій штучного інтелекту для прогнозування потреб у запчастинах, оптимізації розкладу робіт та персоналізації обслуговування клієнтів.

Розроблений у рамках дипломної роботи веб-застосунок з управління автосервісом є ефективним інструментом для автоматизації бізнес-процесів автосервісних підприємств, який відповідає сучасним вимогам галузі та має потенціал для подальшого розвитку та вдосконалення.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Писаревський А. В., Компанець К. А. Система якості послуг автосервісних підприємств України: монографія. Київ, 2018.
2. Городецький М. Я. Управління конкурентоспроможністю підприємств автосервісу: монографія. Львів, 2020.
3. Бурлака К. А. Розроблення інформаційної системи для управління автосервісом: кваліфікац. робота бакалавра. Київ, 2024.
4. Філатова М. В. Інформаційна технологія машинного навчання для детектування та розпізнавання тексту в веб-системі для автосервісу: магіст. дис. Сумський державний університет, 2022.
5. Лістов С. І. Система GRM для контролю за роботою персоналу СТО за рахунок оптимізації процесів взаємодії з клієнтами: кваліфікац. робота бакалавра. Харків, 2022.
6. Харута А. О. Інформаційний веб-портал станції технічного обслуговування «Автограф»: кваліфікац. робота бакалавра. Київ, 2023.
7. Бєльський М. С., Сайківська Л. Ф. Особливості розробки веб-застосунків для електронної комерції: дис. ... канд. техн. наук. Київ, 2024.
8. Тененський М. В., Галюк М. В. Дослідження та розробка застосунку для автоматизації бізнес-процесів з розширеними інтеграційними можливостями: магіст. дис. Київ, 2024.
9. Журило І. В., Лазарєва Ю. Г. Організація системи логістичних бізнес-процесів на підприємствах автосервісу: дис. ... канд. екон. наук. Кропивницький: РВЛ ЦНТУ, 2020.
10. Довга Ю. О. Автоматизація бізнес-процесів з застосуванням SaaS-технологій: кваліфікац. робота бакалавра. Київ, 2020.
11. Сатко С. Управління бізнес-процесами на підприємстві (за матеріалами ПрАТ «Українська автомобільна корпорація», філія «Автомобільний центр Київ», м. Київ): кваліфікац. робота бакалавра. Київ, 2019.
12. Козаченко М. Р., Ліщинська Л. Б. Аналіз вимог до інтелектуального програмного застосунку для адаптивного керування параметрами екрана // Молодь в науці: дослідження, проблеми, перспективи (МН-2025): матеріали конф. 2025.

13. Джрагацпанян Н. В. Дослідження та реалізація функціональних вимог до вебзастосунків онлайн шкіл задля забезпечення інклюзивних підходів навчання: магіст. дис. Національний університет "Запорізька політехніка", 2022.

14. Редько І. В., Яганов П. О. Концептуальна модель технологічного середовища програмування: кваліфікац. робота бакалавра. Київ, 2020.

15. Гребеннік І. В., Коваленко О. А. Логічна модель оцінки знань // Science, innovations and education: problems and prospects: Proc. of the 12th Int. sci. and pract. conf. Tokyo, Japan: CPN Publishing Group, 2022.

16. Пундик С. Е. Р. Г. І. Й., Кармаліта А. Н. А. Т. О. Л. І. Й. Фізична модель взаємодії струменів повітря з рельєфом плоскої поверхні: кваліфікац. робота. Харків, 2024.

17. Кривошеєнко Ю. Проектування та розробка користувацького інтерфейсу веб-системи: кваліфікац. робота бакалавра. Київ, 2021.

18. Годько Ю. С. Розробка дизайну інтерфейсу мобільного додатку «JIIL»: кваліфікац. робота. Харків, 2024.

19. Рисак Д. С. Серверна частина платформи дистанційного навчання: кваліфікац. робота. Київ, 2022.

20. Теницька А. С. Клієнтська частина веб-сайту за допомогою JavaScript: кваліфікац. робота. Київ, 2020.

21. Тітаренко І. І. Педагогічні умови формування конфліктологічної компетентності майбутніх фахівців з реклами і зв’язків з громадськістю: автореф. дис. … канд. пед. наук. Київ, 2018. 23 с. URL: http://ipood.com.ua/data/avtoreferaty_i_dysertatsii/2018/avtoref_TITARENKO_pas.pdf (дата звернення: 04.09.2019)/

22. Мазурець О. В. Інформаційна технологія автоматизованого створення тестів до навчальних матеріалів: кваліфікац. робота. Київ, 2019.

23. Герасименко І. Дослідження методології розробки системи управління взаємин з клієнтами: кваліфікац. робота. Київ, 2023.

24. Чолан С. М. Інформаційна система управління орендою автомобілів: кваліфікац. робота. Київ, 2024.

25. Поліщук О. П. Методологія наукових досліджень: базові поняття, тести та інструктивно-методичні вказівки до їх виконання. Київ, 2023.

26. Крепич С. Я., Співак І. Я. Якість програмного забезпечення та тестування: базовий курс. Київ, 2020.

27. Томачинська В. С. Моделі даних автоматизованого тестування користувацького інтерфейсу веб-сайтів: кваліфікац. робота. Львів, 2023.

28. Невкипілова О. Я., Амерідзе О. С. Оптимізація письмового контролю знань, умінь і навичок з фахової англійської мови студентів технічних спеціальностей. Київ, 2018.

29. Цуранова О., Шевченко І., Романенко Л. Методичні рекомендації щодо впровадження технологій гейміфікації в дистанційній освіті. Київ, 2022.

ДОДАТКИ

ДОДАТОК А ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Розробка web-застосунку для управління автосервісом

Виконав студент 4 курсу
групи ПД-42
Чернодуб Владислав Олегович
Керівник роботи
Кандидат технічних наук, доцент, доцент кафедри ІПЗ Яскевич Владислав Олександрович

Київ – 2025

МЕТА, ОБ’ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

- **Мета роботи** - Підвищення ефективності роботи автосервісів за рахунок автоматизації бізнес-процесів у сфері обслуговування транспортних засобів.
- **Об’єкт дослідження** - Процеси автоматизації роботи автосервісу за допомогою веб-застосунку. Особлива увага приділяється розробці функціоналу для управління записом клієнтів, обліком виконаних робіт і взаємодії з працівниками сервісу.
- **Предмет дослідження** - Веб-застосунок призначений для управління процесами автосервісу, включаючи запис клієнтів, ведення історії обслуговування, контроль за виконанням ремонтних робіт та управління персоналом

ЗАДАЧІ ДИПЛОМНОЇ РОБОТИ

1. Проаналізувати існуючі рішення для автоматизації автосервісів і визначити їхні переваги та недоліки.
2. Визначити функціональні та нефункціональні вимоги до веб-застосунку.
3. Розробити архітектуру системи та обрати відповідні технології для реалізації.
4. Реалізувати основний функціонал веб-застосунку, включаючи запис клієнтів, облік робіт і управління співробітниками.
5. Провести загальне тестування системи з метою виявлення помилок і визначення відповідності додатку функціональних вимог.

ВИМОГИ ДО ДОДАТКУ

Функціональні вимоги:

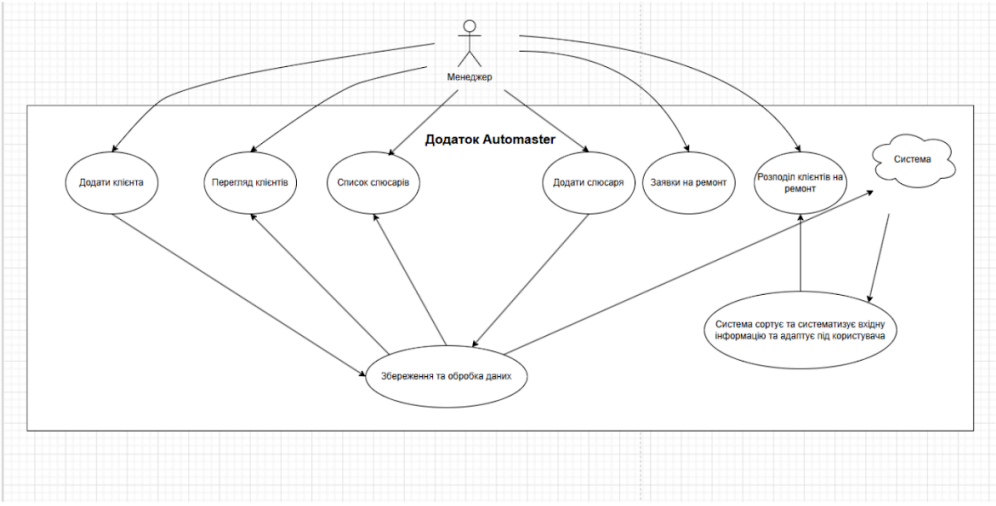
1. Користувач повинен мати можливість створювати, редагувати та видаляти записи клієнтів і автомобілів.
2. Система повинна надавати адміністратору змогу призначати заявки на конкретних майстрів та вести облік виконаних робіт.

Нефункціональні вимоги:

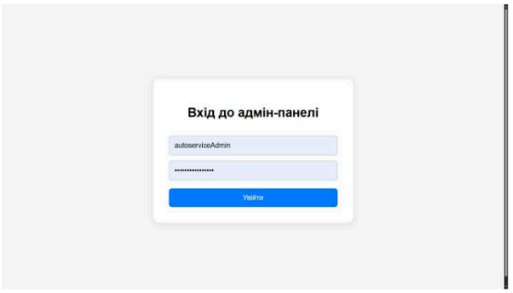
1. Інтерфейс системи має бути інтуїтивно зрозумілим, з логічною структурою та зручною навігацією між основними розділами.
2. Веб-застосунок повинен коректно працювати на різних типах пристроїв таких як ПК, ноутбуках, планшетах і смартфонах - через сучасні браузери.
3. Система має бути доступною в будь-який час доби за наявності інтернет-з`єднання.

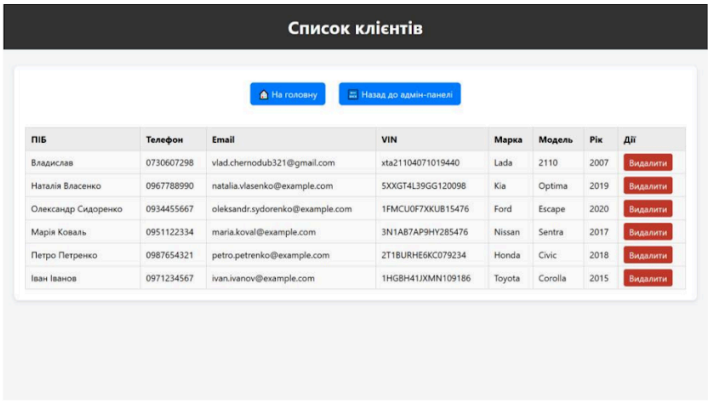
ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ

ДІАГРАМА ПРЕЦЕДЕНТІВ

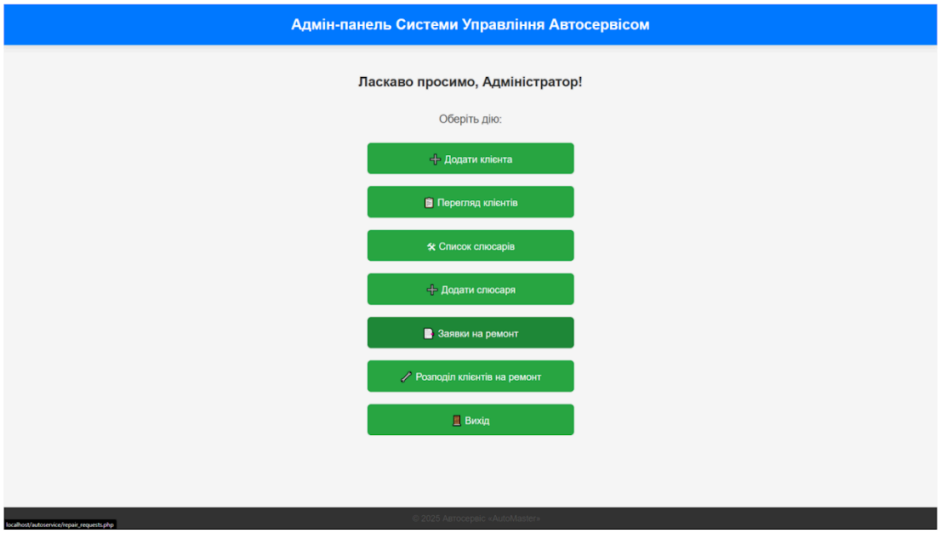


ЕКРАННІ ФОРМИ



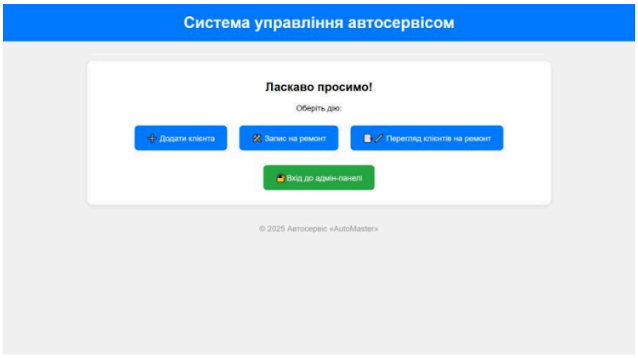


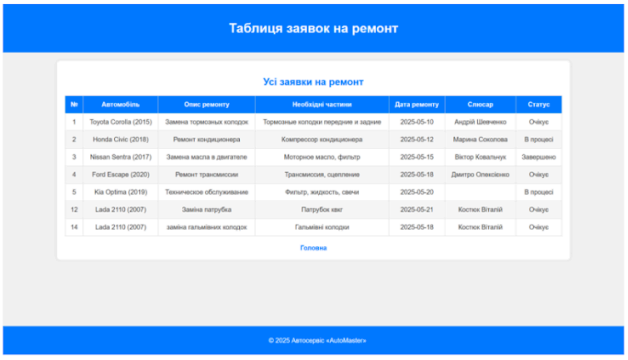
ЕКРАННІ ФОРМИ



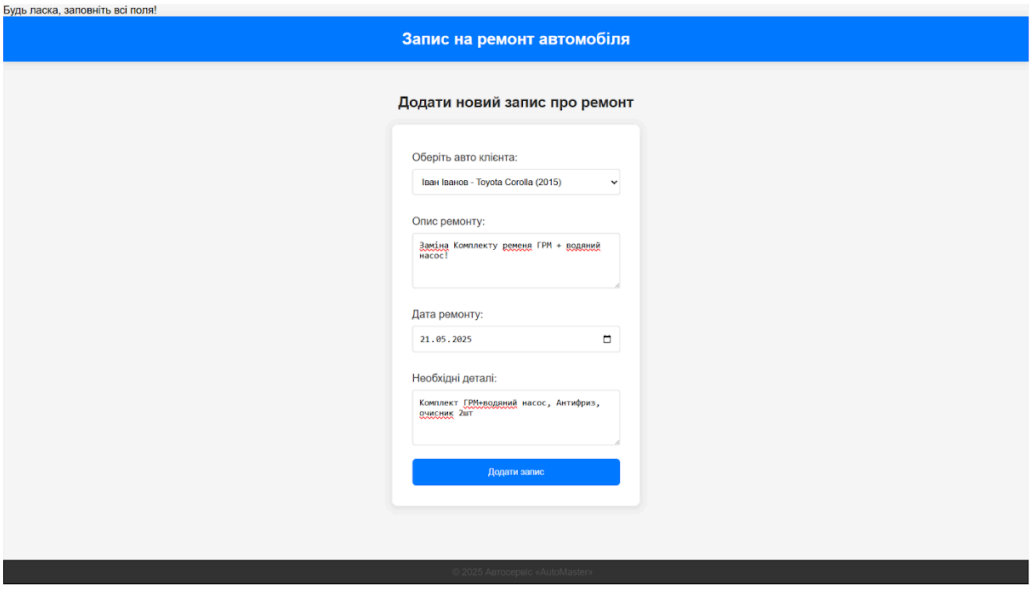
Головна сторінка Адміністратора

ЕКРАННІ ФОРМИ

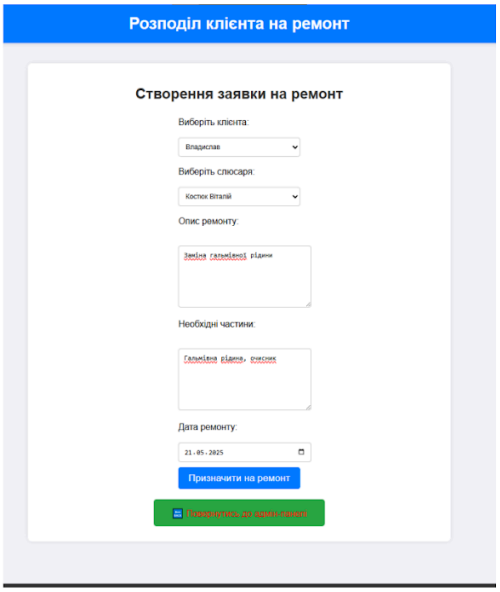


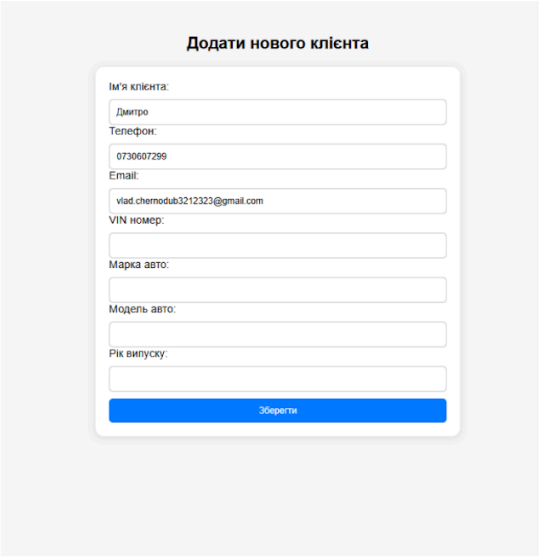


ЕКРАННІ ФОРМИ



ЕКРАННІ ФОРМИ





РЕЗУЛЬТАТИ ТЕСТУВАННЯ

- Проведено тестування веб-застосунку на автосервісі
- Отримано позитивні відгуки після тестування
- Делегування роботи на сервісі стало комфортніше

ВИСНОВКИ

1. Реалізований функціональний веб-застосунок для управління автосервісом.
2. Забезпечений UI/UX
3. Спроєктований інтерфейс користувача з урахуванням зручності в використанні.
4. Реалізувано основний функціонал веб-застосунку, включаючи запис клієнтів, облік робіт і управління співробітниками.
5. Проведене тестування системи та оцінено її ефективність.

ДЯКУЮ ЗА УВАГУ!

ДОДАТОК Б ЛІСТИНГ ПРОГРАМНИХ МОДУЛІВ

```
<?php
session_start();
if (!isset($_SESSION["logged_in"])) {
    header("Location: login.php");
    exit();
}
```

```
?>

<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="UTF-8">
  <title>Адмін-панель | Автосервіс</title>
  <link rel="stylesheet" href="css/admin_dashboard.css">
</head>
<body>

  <header>
    <h1>Адмін-панель Системи Управління Автосервісом</h1>
  </header>

  <div class="container">
    <h2>Ласкаво просимо, Адміністратор!</h2>
    <p>Оберіть дію:</p>

    <a href="add_client.php" class="nav-link">➕ Додати клієнта</a>
    <a href="clients_list.php" class="nav-link">📋 Перегляд клієнтів</a>
    <a href="schedule.php" class="nav-link">🛠 Список слюсарів</a>
    <a href="add_mechanic.php" class="nav-link">➕ Додати слюсаря</a>

    <a href="repair_requests.php" class="nav-link">📄 Заявки на ремонт</a>

        <a href="assign_repair.php" class="nav-link">🔧 Розподіл клієнтів на ремонт</a>

    <a href="logout.php" class="nav-link">🚪 Вихід</a>
  </div>
```

```
  <footer>
    <p>© 2025 Автосервіс «AutoMaster»</p>
  </footer>

</body>
</html>

<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "autoservice_db";

$conn = mysqli_connect($servername, $username, $password, $dbname);

if (!$conn) {
    die("Помилка підключення: " . mysqli_connect_error());
}

mysqli_set_charset($conn, "utf8");
?>
<?php

include('db_connection.php');

$query = "
    SELECT
        repairs.id,
            CONCAT(vehicles.make, ' ', vehicles.model, ' (', vehicles.year, ')') AS
vehicle_name,
```

```
        repairs.description,
        repairs.parts_needed,
        repairs.date,
        mechanics.name AS mechanic_name,
        repairs.status
    FROM repairs
    LEFT JOIN vehicles ON repairs.vehicle_id = vehicles.id
    LEFT JOIN mechanics ON repairs.mechanic_id = mechanics.id
";

$result = mysqli_query($conn, $query);
?>

<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <title>Перегляд заявок на ремонт</title>
    <link rel="stylesheet" href="css/view_repairs.css">
</head>
<body>

<header>
    <h1>Таблиця заявок на ремонт</h1>
</header>

<div class="container">
    <h2>Усі заявки на ремонт</h2>
```

```
<table class="repairs-table">
    <thead>
        <tr>
            <th>№</th>
            <th>Автомобіль</th>
            <th>Опис ремонту</th>
            <th>Необхідні частини</th>
            <th>Дата ремонту</th>
            <th>Слюсар</th>
            <th>Статус</th>
        </tr>
    </thead>
    <tbody>
        <?php if (mysqli_num_rows($result) > 0): ?>
            <?php while ($repair = mysqli_fetch_assoc($result)): ?>
                <tr>
                    <td><?= $repair['id'] ?></td>
                    <td><?= $repair['vehicle_name'] ?></td>
                    <td><?= $repair['description'] ?></td>
                    <td><?= $repair['parts_needed'] ?></td>
                    <td><?= $repair['date'] ?></td>
                    <td><?= $repair['mechanic_name'] ?></td>
                    <td><?= $repair['status'] ?></td>
                </tr>
            <?php endwhile; ?>
        <?php else: ?>
            <tr>
                <td colspan="7">Немає заявок на ремонт.</td>
            </tr>
        <?php endif; ?>
```

```
        </tbody>
    </table>

    <a href="index.php" class="nav-link">Головна</a>
</div>

<footer>
    <p>© 2025 Автосервіс «AutoMaster»</p>
</footer>

</body>
</html>
<?php
include('db_connection.php');

if (isset($_GET['delete_id'])) {
    $delete_id = intval($_GET['delete_id']);

    $stmt = $conn->prepare("DELETE FROM clients WHERE id = ?");
    if ($stmt) {
        $stmt->bind_param("i", $delete_id);
        $stmt->execute();
        $stmt->close();
    }
    header("Location: clients_list.php");
    exit;
}

$query = "
    SELECT c.id AS client_id, c.name, c.phone, c.email,
```

```
        v.vin, v.make, v.model, v.year
    FROM clients c
    LEFT JOIN vehicles v ON c.id = v.client_id
    ORDER BY c.id DESC
";

$result = mysqli_query($conn, $query);
?>

<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <title>Клієнти | Автосервіс</title>
    <link rel="stylesheet" href="css/index.css">
    <link rel="stylesheet" href="css/clients_list.css">
    <style>
        table {
            width: 100%;
            border-collapse: collapse;
            margin-top: 20px;
            background: #f9f9f9;
        }

        th, td {
            padding: 10px;
            border: 1px solid #ccc;
            text-align: left;
        }
```

```
        th {
            background: #eee;
        }

        .delete-btn {
            background-color: #c0392b;
            color: white;
            padding: 5px 10px;
            text-decoration: none;
            border-radius: 4px;
        }

        .delete-btn:hover {
            background-color: #e74c3c;
        }
    </style>
</head>
<body>

<header>
    <h1>Список клієнтів</h1>
</header>
<div class="container">
    <a href="index.php" class="nav-link">🏠 На головну</a>
        <a href="admin_dashboard.php" class="nav-link">🔙 Назад до адмін-панелі</a>

    <table>
        <thead>
            <tr>
```

```
                <th>ПІБ</th>
                    <th>Телефон</th>
                    <th>Email</th>
                    <th>VIN</th>
                    <th>Марка</th>
                    <th>Модель</th>
                    <th>Рік</th>
                    <th>Дії</th>
                </tr>
            </thead>
            <tbody>
            <?php if (mysqli_num_rows($result) > 0): ?>
                <?php while ($row = mysqli_fetch_assoc($result)): ?>
                    <tr>
                        <td><?= htmlspecialchars($row['name']) ?></td>
                        <td><?= htmlspecialchars($row['phone']) ?></td>
                        <td><?= htmlspecialchars($row['email']) ?></td>
                        <td><?= htmlspecialchars($row['vin']) ?></td>
                        <td><?= htmlspecialchars($row['make']) ?></td>
                        <td><?= htmlspecialchars($row['model']) ?></td>
                        <td><?= htmlspecialchars($row['year']) ?></td>
                        <td>
                                <a href="?delete_id=<?= $row['client_id'] ?>" class="delete-btn" onclick="return confirm('Ви впевнені, що хочете видалити цього клієнта?');">Видалити</a>
                        </td>
                    </tr>
                <?php endwhile; ?>
            <?php else: ?>
                <tr><td colspan="8">Клієнтів не знайдено.</td></tr>
            <?php endif; ?>
```

```
        </tbody>
    </table>
</div>

</body>
</html>
```