Хмельницький національний університет
Факультет інформаційних технологій
Кафедра інженерії програмного забезпечення

ДИПЛОМНИЙ ПРОЕКТ

Інформаційна система моніторингу діяльності автосалону

Рівень вищої освіти Перший (бакалаврський)

Галузь знань 12 «Інформаційні технології»

Спеціальність 121 «Інженерія програмного забезпечення»

Освітня програма Освітньо-професійна програма «Інженерія програмного забезпечення»

Шифр ДППІЗ.190158.01.11.ПЗ

Виконав студент IV курсу група ІПЗс-19-1 _____ Р.Р.Савкін
Підпис Ініціали, прізвище

Керівник канд. техн. наук, доцент _____ Ю.В.Форкун
Науковий ступінь, звання Підпис Ініціали, прізвище

Нормоконтролер канд. техн. наук, доцент _____ І.В. Гурман
Підпис Ініціали, прізвище

До захисту допускаю:

Завідувач кафедри інженерії
програмного забезпечення _____ Л. П. Бедратюк
Підпис Ініціали, прізвище

16 червня 2022 р.

Хмельницький 2022

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет Інформаційних технологій
Кафедра Інженерії програмного забезпечення
Рівень вищої освіти Перший (бакалаврський)
Галузь знань 12 «Інформаційні технології»
Спеціальність 121 «Інженерія програмного забезпечення»
Освітня програма Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри ІПЗ
Л. П. Бедратюк
05 02 2022 р.

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ)

Савкіну Роману Руслановичу
Прізвище, ім'я, по батькові студента

1. Тема проєкту (роботи) Інформаційна система моніторингу діяльності автосалону ------

Керівник проєкту (роботи) Форкун Юрій Вікторович, канд. техн. наук, доцент
Прізвище, ім'я, по батькові, науковий ступінь, вчене звання

Затверджена наказом ректора університету від 01.03.2022 р. № 18

2. Строк подання студентом проєкту (роботи) на кафедру 01.06.2022 р.

3. Вихідні дані до проєкту (роботи) Матеріали переддипломної практики, Методичні вказівки до виконання кваліфкаційної роботи

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
Дослідження предметної області та постановка задачі, проектування програмного забезпечення, програмна реалізація системи, тестування програмної системи

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслень)
Презентаційні матеріали (слайди)

6. Консультанти розділів дипломного проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Гурман І.В., доцент кафедри ІПЗ		
Антиплагіат	Гурман І. В., доцент кафедри ІПЗ		

7. Дата видачі завдання «_05_» лютого_ 2021 р.

КАЛЕНДАРНИЙ ПЛАН

Назва етапів (розділів) дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітки
1 Ознайомлення з тематикою дипломного проектування (ДП), визначення та узгодження індивідуальних там ДП	01.12 – 30.12.2021	
2 Дослідження предметної області, в якій планується використання програмного засобу (ПЗ), визначення задач та вимог, розробка технічного завдання	02.01 – 31.01.2022	
3 Проектування програмного забезпечення	01.02 – 28.02.2022	
4 Програмна реалізація	01.03 – 10.04.2022	
5 Тестування програмного забезпечення	11.04 – 30.04.2022	
6 Написання вступу, загальних висновків, оформлення джерел посилання та додатків. Оформлення пояснювальної записки ДП згідно вимог стандартів	01.05 – 25.05.2022	
7 Попередній захист ДП	Травень 2022 (згідно графіка)	
8 Перевірка ДП на плагіат, нормконтроль, отримання відгуків та рецензій. Брошурування (зшиття) пояснювальної записки)	26.05 – 30.05.2022	
9 Підготовка до захисту та захист ДП	з 01.06.2022	

Студент _____ Савкін Д. Р.
Підпис Ініціали, прізвище

Керівник проекту (роботи) _____ Гурман І.В.
Підпис Ініціали, прізвище

АНОТАЦІЯ

Тема дипломного проекту: «Інформаційна система моніторингу діяльності автосалону».

Автор проекту: Савкін Роман Русланович

Керівник проекту: Форкун Юрій Вікторович.

Пояснювальна записка: 66 с., 33 рис., 8 табл., 7 дод., 12 джерел.

Графічна частина: 18 презентаційних слайдів.

Ключові слова: API, MVC, HTML, CSS, DATABASE, MYSQL, SQL, C#, VS XAMPP, .NET, SATC.

Метою проекту є дослідження предметної області, аналіз, реалізація методів та розробка інформаційної системи моніторингу діяльності автосалону.

У дипломному проекті проведено аналіз предметної області, змісту і методів програмного забезпечення, визначено функціональні вимоги до програмного продукту, розроблено архітектуру програмного продукту, спроектовано структуру та розроблено базу даних, та здійснено конструювання програмного продукту на клієнт-серверній технології.

В якості реалізації програмного продукту було використано мову програмування C# та СКБД MySQL та середовище розробки додатків Microsoft Visual Studio 2019.

06.06.2022

ВІДОМІСТЬ ДОКУМЕНТІВ

№ рядка	Формат	Позначення документа	Найменування документа	К-сть аркушів	№ екз.	Примітка
			Текстові документи			
1	А	ДПІПЗ.190158.01.11.ПЗ	Пояснювальна записка	65		
2	А		Завдання на дипломний	1		
			проект			
3	А		Анотація	1		
			Графічні документи			
4	А		Презентаційні матеріали	18		

Змн.	Арк.	№ докум.	Підпис	Дата	ДПІПЗ.190158.01.11.ВД		
Виконав		Савкін Р.Р.		6.06	Інформаційна система моніторингу діяльності автосалону **Відомість документів**	Літ.	Арк. 1 / Акрушів 1
Керівник		Форкун Ю.В.		6.06			
Рецензент						ХНУ, ІПЗс-19-1	
Н. Контр.		Гурман І.В.		12.06			
Зав. каф.		Бедратюк Л.П.		16.06			

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

БД -	– база даних;
ІЛМ	– інформаційно-логічна модель;
ПЗ	–програмне забезпечення;
СКБД	– система керування базами даних;
ASP	– технологія створення веб-застосунків і веб-сервісів компанії Майкрософт
DBMS	– система керування базами даних;
CSS	–мова формлення стилю сторінок;
HTML	– мова розмітки веб-сторінок;
MVC	– архітектурний шаблон Model-View-Controler;
MYSQL	– сервер баз даних MySQL;
JS	– скриптова мова програмування;
SQL	– мова запитів до баз даних;
SATC	– Software Assurance Technology Center;
XAMPP	– багатоплатформова збірка веб-сервера.

ВСТУП

Розвиток технологій дає можливість різноманітним підприємствам та установам використовувати автоматизовані інформаційні системи.

Інформаційна система – сукупність організаційних і технічних засобів для збереження та обробки інформації з метою забезпечення інформаційних потреб користувачів системи.

В будь-якій інформаційній системі управління вирішуються задачі трьох різних типів:

− задачі оцінки ситуації (інколи їх називають задачами для розпізнавання образів та рисунків);

− задачі перетворення опису ситуації (так звані розрахункові задачі, задачі моделювання тощо);

− задачі прийняття рішень, в тому числі і оптимізаційні.

Загалом будь-яка інформаційна система повинна виконувати основні задачі поставлені перед нею:

- реєстрація та зберігання даних;
- надання прав доступу до даних;
- модерація даних.

Метою і призначенням даного проекту є розробка бази даних, архітектури та компонентів модуля моніторингу діяльності автосалону з метою визначення особливостей маркетингової діяльності автосалону, та відповідно, надання рекомендацій, щодо підвищення ефективності маркетингової діяльності автосалону з урахуванням впливу як внутрішнього так і зовнішнього середовища.

Зараз практично жодне підприємством та жодна інформаційна система, технічна система, інженерна мережа та система транспорту не можуть існувати, якщо в них недостатньо інформації, або не в повній мірі відбуваються самі процеси передачі та обміну цією інформацією. Це можна побачити при розгляді функціоналу будь-якої системи (до прикладу деякої державної установи,

виробничого процесу, технологічного процесу тощо). Автотранспортні та автосалони не можуть ефективно працювати, якщо інформаційні процеси в них недостатньо врегульовані і, відповідно відбувається втрата інформації. Інформаційні технології тут стають саме тією продуктивною силою. Різноманітні комп'ютерні мережі та системи зв'язку дозволяють докорінно змінити не тільки сферу виробництва, але суспільство загалом, зокрема, поширюється зв'язок між людьми та різними країнами, світ та суспільство стають більш глобальнішими як для кожної окремої людини так і загалом. Саме тут виникає така ситуація, коли різна інформація та інформаційні технології стають загальнодоступними для усіх.

Враховуючи вище наведене, метою нашого проекту, є створення інформаційної системи, яка б не тільки дозволяла вести облік проданих автомобілів, їх технічного стану та супутніх товарів і послуг, умови гарантійного обслуговування автомобілів, які здійснюються менеджерами автосалону але й надавати рекомендації щодо самої діяльності автосалону, для покращення його роботи загалом.

Об'єктом нашого дослідження виступає процес продажу автомобілів та маркетингової діяльності автосалону.

Предметом дослідження є теоретичні, методичні та практичні завдання використання інформаційних технологій маркетингової діяльності автосалону.

Новизна одержаних результатів полягає у тому, що дана система дозволяє здійснювати ефективне керування процесу моніторингу діяльності автосалону шляхом визначення особливостей маркетингової діяльності автосалону та розробки рекомендацій, щодо підвищення ефективності його маркетингової діяльності з урахуванням впливу внутрішнього та зовнішнього та середовища, та застосуванням якісного використання сучасних інформаційних технологій.

Практичне значення отриманих результатів полягає у максимальному спрощенні процесу інформаційної діяльності маркетингової діяльності автосалону, де однією з важливих переваг системи є те, що вона самостійно здійснюватиме оновлення інформаційної системи. Також, практична значимість

проекту полягає в групуванні та автоматизації вхідних даних. Значимість програмного забезпечення полягатиме у зручності роботи з базою даних та реалізація інтерфейсу, що забезпечить процес оформлення замовлення, перегляд, використання, оновлення інформації з бази даних та додавання нової інформації у інформаційну систему.

1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Аналіз та визначення видів, змісту та методів інформаційного забезпечення предметної області

Автосалон – комерційно експлуатаційне підприємство, що займається технічним продажем та обслуговуванням автомобілів, забезпечує продажу в необхідній кількості, укомплектоване штатом водіїв, диспетчерів та ремонтних робітників.

В ході виконання цієї роботи необхідно автоматизувати роботу підприємства, що здійснює продаж, обслуговування та моніторинг продаж автомобілів працівниками.

При замовлені автомобіля потрібно оформляти замовлення на купівлю автомобіля, а також зберігати дані про замовлений автомобіль, клієнта, що робить замовлення та продавця-консультанта, що приймає замовлення. Періодично потрібно оновляти дані про наявні автомобілі, про звільнених та прийнятих на роботу нових продавців консультантів, формувати звіти про кількість проданих автомобілів.

При покупці автомобіля формується замовлення на купівлю. Воно складається з інформації про автомобіль(автомобіль обирається із списку автомобілів, які поставляє автосалон), інформації про клієнта-покупця(його паспортні дані, адреса, номер телефонна), інформації про продавця-консультанта, що оформляв замовлення і оформляє замовлення, дати замовлення та стану замовлення

Дана інформаційна система підприємства повинна враховувати відомості про персонал підприємства, наявні автомобілі, додаткові послуги, сервісне обслуговування та моніторинг роботи менеджерів. Вона повинна надавати оператору звіти, необхідні для діяльності автовсалону та роботи менеджерів загалом, а саме:

- список персоналу підприємства;
- список автомобілів;
- список додаткових послуг;
- список додаткових пропозицій;
- відомості, про які підлягають гарантійному ремонту автомобілів (період експлуатації);
- моніторинг діяльності підприємства;
- моніторинг роботи менеджерів;
- інші звіти.

Наша інформаційна система повинна володіти простим та зручним для користувача інтерфейсом, повинна бути забезпечена можливість коригування всіх даних системи.

Після впровадження використання інформаційної системи на автобусному парку з'явиться ряд переваг для працівників:

- автоматизований процес ведення інформації про транспортні засоби;
- зручний облік штату менеджерів, продавців, слюсарів та персоналу;
- покращення процесу проведення гарантійних ремонтних робіт;
- зручна звітність.

Розглянемо основні завдання, які вирішуються інформаційною системою.

- аналіз даних. Тут під інтерпретацією та аналізом даних розуміється процес визначення змісту даних, результати якого мають бути погодженими і коректними. Зазвичай передбачається багатоваріантний аналіз даних;
- діагностика. Під діагностикою розуміється процес співвідношення об'єкту з деяким класом об'єктів і виявлення несправності в деякій системі. Несправність - це відхилення від норми. Таке трактування дозволяє з єдиних теоретичних позицій розглядати і несправність обладнання та устаткування в технічних системах, але й захворювання живих організмів та інші природні явища і аномалії;

− моніторинг. Основне завдання моніторингу - безперервна інтерпретація даних в реальному часі і сигналізація про вихід тих або інших параметрів за допустимі межі;

− проектування. Проектування полягає в підготовці специфікацій на створення «об'єктів» із заздалегідь визначеними властивостями. Під специфікацією розуміється весь набір необхідних документів − креслення, пояснювальна записка тощо. Основні проблеми тут здобуття чіткого структурного опису знань про об'єкт і проблема «сліду»;

− прогнозування. Прогнозування дозволяє передбачати наслідки деяких подій або явищ на підставі аналізу наявних даних. Прогнозують системи логічно виводять вірогідні наслідки з заданих ситуацій;

− планування. Під плануванням розуміється знаходження планів дій, що відносяться до об'єктів, здатних виконувати деякі функції. У таких ЕС використовуються моделі поведінки реальних об'єктів з тим, щоб логічно вивести наслідки планованої діяльності;

− навчання. Під навчанням розуміється використання комп'ютера для навчання деякої дисципліни або предмету. Системи вчення діагностують помилки при вивченні якої-небудь дисципліни за допомогою ЕОМ і підказують правильні рішення;

− управління. Під керуванням розуміється функція організованої системи, що підтримує певний режим діяльності. Такого роду ЕС здійснюють управління поведінкою складних систем відповідно до заданих специфікацій;

− підтримка прийняття рішень. Підтримка прийняття рішень - це сукупність процедур, що забезпечує особу, яка приймає рішення, необхідною інформацією і рекомендаціями, що полегшують процес ухвалення рішення. Ці ЕС допомагають фахівцям вибрати і сформувати потрібну альтернативу серед нескінченного числа виборів при ухваленні відповідальних рішень керівництвом, менеджерами та працівниками автосалонує.

1.2 Аналіз інформаційного забезпечення предметної області

Інформаційна система автосалону потребує збереження та опрацювання досить значних обсягів інформації. Існує багато способів зберігання інформації, серед яких, можна для прикладу навести текстові файли, типізовані файли, бази даних тощо. Найкраще на нашу думку для вирішення проблеми зберігання та оброки даних при роботі автосалону підходять бази даних. Вони є найзручнішими і надають більше можливостей, необхідних для роботи з даними.

База даних, яка складається із подібних таблиць називається реляційною. Реляційна модель зберігає дані у вигляді таблиць, вона є простою в роботі і реалізації, також дозволяє досить швидко створювати працюючі системи.

Сутності зручно представляти у формі таблиць, де кожен рядок є кортежем, а кожен стовпець - атрибутом, який визначений на певному домені. Такий неформальний підхід до поняття відношення надає більш звичну для програмістів, розробників та користувачів форму представлення, де реляційна база даних являє собою кінцевий набір таблиць. Слід відмітити, що також є інші моделі БД, які в певних випадках значно ефективніші, ніж реляційні моделі:

– ієрархічна модель - може бути представлена як дерево, що складається з об'єктів різних рівнів. Між об'єктами тут існують зв'язки типу «батько-нащадок». При цьому можлива ситуація, коли об'єкт не має нащадків, або має їх декілька, тоді як у об'єкта-нащадка має бути обов'язково тільки один батьківський об’єкт;

– мережева модель - подібна до ієрархічної моделі, за винятком того, що кожен вузол БД може мати взаємодію з іншими вузлами завдяки складній системі зв’язків;

– об’єктна модель - зберігаються не лише дані, а і методи їх обробки у вигляді програмного коду.

Для побудови баз даних є багато програм таких як Microsoft Acces, Oracle, DataBase, MySql Server.

Для даного проекту було обрано клієнт-серверну архітектуру, тому що вона ідеально підходить для вирішення завдання – автоматизовувати обробку запитів великої кількості користувачів і рівномірно розподілювати навантаження в мережі та між користувачами.

1.3 Постановка задачі

Реалізація поставленої мети обумовила такі завдання:

- розглянути сутність та зміст маркетингової діяльності підприємства;
- розглянути методичні аспекти аналізу маркетингової діяльності;
- визначити специфіку маркетингової діяльності підприємств автодилерів салонів;
- дослідити складові маркетингової діяльності автосалону;
- проаналізувати маркетингове середовище автосалонів;
- визначити цінові сегментів ринку автомобілів в Україні;
- окреслити альтернативи маркетингової діяльності автосалону;
- запропонувати засоби управління поведінкою споживачів підприємства на ринку легкових автомобілів.

Предметною областю цього проекту є діяльність автосалону. Вона має такі функції:

- перегляд обліку автомобілів та інформацію про них;
- перегляд покупців та інформацію про них;
- перегляд співробітників та інформацію про них;
- перегляд продажу автомобілів;
- перегляд поставок автомобілів;
- перегляд запропонованих супутніх товарів та послуг;
- перегляд спецпропозицій;
- перегляд кредитних пропозицій.

Інформація буде заноситись у базу менеджерами та продавцями салону та буде доступна для перегляду керівництву та менеджерам.

Проектована інформаційна система дасть змогу збільшити ефективність отримання усієї необхідної інформації, як то про наявність автомобілів у салоні, покупців автомобілів, співробітників, продажі та поставки автомобілів, супутні товари, спецпропозиції тощо. Також частково позбавить працівників необхідності роботи з паперовими документами, та при цьому дозволить підвищити ефективність роботи з оброблювальними даними.

Проект проектується на основі поставленої задачі, яка має назву «Інформаційна система моніторингу автосалону». Дана задача призначена для надання користувачу інформації автосалону та автоматизованого отримання результатних даних про наявність та продаж автомобілів, як в натуральних та вартісних показниках.

Основною метою розв’язування задачі є створення автоматизованої системи обслуговування бази даних автосалону та проведення маркетингового дослідження. Необхідність автоматизованого розв'язання задачі визначається зменшенням трудових затрат на обробку інформації, підвищенням якості обліку і контролю за наявністю та рухом автомобілів в автосалоні, оперативним отриманням різнобічної результатної інформації, збільшенням її достовірності, точності тощо.

Для проведення маркетингового дослідження роботи автосалону слід рахувати, що у сучасних умовах усі автосалони стикаються з значною проблемою конкуренції на ринку. Одним із головних факторів є імідж є автосалону, що впливає досить суттєво на досягнення конкурентоспроможності та його ефективної діяльності на ринку. Під поняттям іміджу слід розуміти певний цілеспрямований сформований, який враховує певні кількісні характеристики, який забезпечує емоційно та психологічно впливати на людей з певною метою просування компанії на ринку.

Так, зокрема, зорієнтуємось на виборі автосалону споживачами за такими

показниками, які визначають ціни на автомобілі, різні послуги, а також репутацією автосалону (Таблиця 1.1.)

Таблиця 1.1. – Показники оцінки маркетингової діяльності автосалону споживачами

Характеристика уподобань	Кількісна характеристика
Ціни	4
Сервіс	4
Якість обслуговування	4
Кваліфікація персоналу	4
Індивідуальність підходу	2
Асортимент автомобілів	3
Спецпропозиції	2
Додаткові послуги	2
Внутрішній комфорт автосалону	4
Репутація	3
Кредитні пропозиції	3

На рисунку 1.1 наведено суб'єктивну оцінку позиціонування автосалону певним експертом за споживчими характеритиками.

Рисунок 1.1 - Споживчі уподобання автосалону

Умови, за яких припиняється розв'язання задачі: виявлені порушення в інформаційній системі та базі даних (внаслідок несанкціонованого або помилкового доступу до даних), відсутність інформації, необхідної для розв'язання задачі, вихід з ладу апаратно-програмних засобів.

Загалом, в розділі здійснено аналіз предметної області, визначення видів, змісту та методів інформаційного забезпечення предметної області. Проаналізовано різні типи рішень, останні досягнення та перспективи розвитку існуючих рішень. Здійснено постановку задачі в загальному вигляді.

2 ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Проектування архітектури та структури системи

Архітектура клієнт – сервер є одним із архітектурних шаблонів програмного забезпечення та є домінуючою концепцією у створенні розподілених мережних засобів і передбачає взаємодію та обмін даними між ними. Вона передбачає такі основні компоненти:

- набір серверів, які надають інформацію або інші послуги програмам, які звертаються до них;
- набір клієнтів, які використовують сервіси, що надаються їм відповідними серверами;
- мережа , яка забезпечує взаємодію між клієнтами та серверами.

В даній системі вирішено використовувати архітектуру «клієнт-сервер», тому що, у сучасних мережевих інформаційних системах для роботи із загальною базою даних використовують цю архітектуру.

Дана система буде реалізована згідно до концепції Model – View – Controler, яка забезпечує зручне розподілення функціоналу, самих даних та, відповідно, відображення даних (рисунок 2.1)

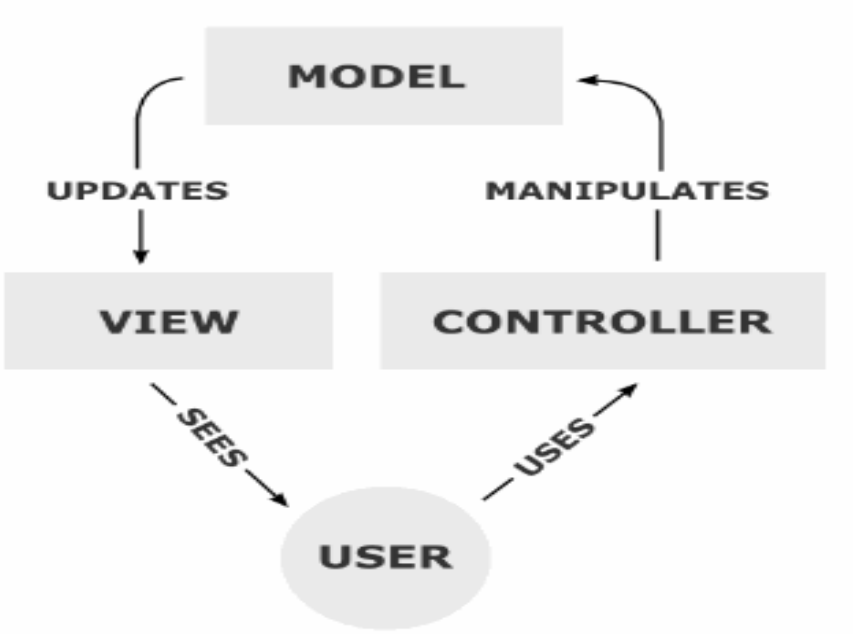


Рисунок 2.1 – Концепція Model – View – Controler

Модель MVC – архітектурний шаблон, який використовується під час проектування та розробки програмного забезпечення.

Цей шаблон поділяє систему на три частини: модель даних, подання даних та керування даними.

MVC застосовується для відокремлення даних (Model) від інтерфейсу користувача (View) так, щоб зміни в інтерфейсі користувача мали мінімальний вплив на роботу з самими даними, а, відповідно, зміни в моделі даних могли здійснюватися без змін в інтерфейсі інформаційної системи.

Мета даного шаблону — гнучкий дизайн програмного забезпечення, який повинен полегшувати подальші зміни чи розширення програм, а також надавати можливість повторного використання окремих компонентів програми. Крім того використання цього шаблону у великих системах призводить до певної впорядкованості їх структури і робить їх зрозумілішими завдяки зменшенню складності.

2.2 Опис декомпозиції

Вихідна система нашого додатку розташовується на нульовому рівні, де після її розділення виходять підсистеми наступних рівнів. Розділ підсистем першого рівня, або деяких з них, приводить до появи підсистем другого рівня, які в свою чергу поділяються на системи третього рівня і так далі.

Декомпозиція – це науковий метод, що використовує структуру завдання і дозволяє замінити вирішення одного великого завдання рішенням серії менших завдань, нехай і взаємопов'язаних, але більш простих. Декомпозиція, як процес розділення, дозволяє розглядати будь-яку досліджувану систему як складну, що складається з окремих взаємопов'язаних підсистем, які, в свою чергу, також можуть бути розділеними на частини. Як системи можуть виступати не тільки матеріальні об'єкти, а й процеси, явища і поняття.

Проектовану інформаційну систему нам необхідно розбити на окремі модулі, які будуть працювати максимально незалежно один від одного згідно концепції MVC з мінімальною зв’язністю. Так система буде складатись з наступних модулів:

– загальнодоступна відкрита частина, яку будуть бачити всі користувачі, в цьому модулі буде розміщена інформація яка призначена для ознайомлення клієнтів з часом відбуття та прибуття автобусу , наявним квитком;

– клієнтський модуль адміністрування – закрита частина, яка доступна тільки авторизованим клієнтам, в якому буде здійснюватись керування обліковим записом, налаштування особистих параметрів, внесення і редагування інформації в базу даних;

– модуль «конструктор сутностей» – закрита частина, доступна тільки для авторизованих користувачів з клієнтського модуля, конструктор призначений для створення структури об’єктів для наповнення бази даних;

– адміністративний модуль керування сервісом має закритий доступ, який є тільки в модераторів сервісу, призначений для керування основними процесами в системі - налаштування облікових записів клієнтів, обмеження кількості запитів, налаштування тарифних планів, обробка загальної статистики активності клієнтів на сервісі.

Декомпозиція дозволить спростити розробку загалом, а також покращити подальшу підтримку продукту і буде надавати можливість зручної розробки нових модулів та підтримки існуючих.

Спрощене графічне представлення декомпозиційованої системи називається її ієрархічною структурою.

Ієрархічна структура може бути зображена у вигляді розгалуженої блок-схеми модулів.

2.3 Опис залежностей

Діаграма компонентів відображає залежності між компонентами нашого програмного забезпечення, що включає компоненти вихідного коду, бінарні компоненти, а також компоненти, які виконуються. Залежності зображаються у вигляді діаграм UML (Component diagram).

Сюди входять такі представники як адміністратор, покупець, клієнт, адміністратор, менеджер. Також є пакети: додавання в базу даних нового клієнта, оформлення квитка для клієнта і компонент для реєстру. Діаграма компонентів наведено на рисунку 2.2.

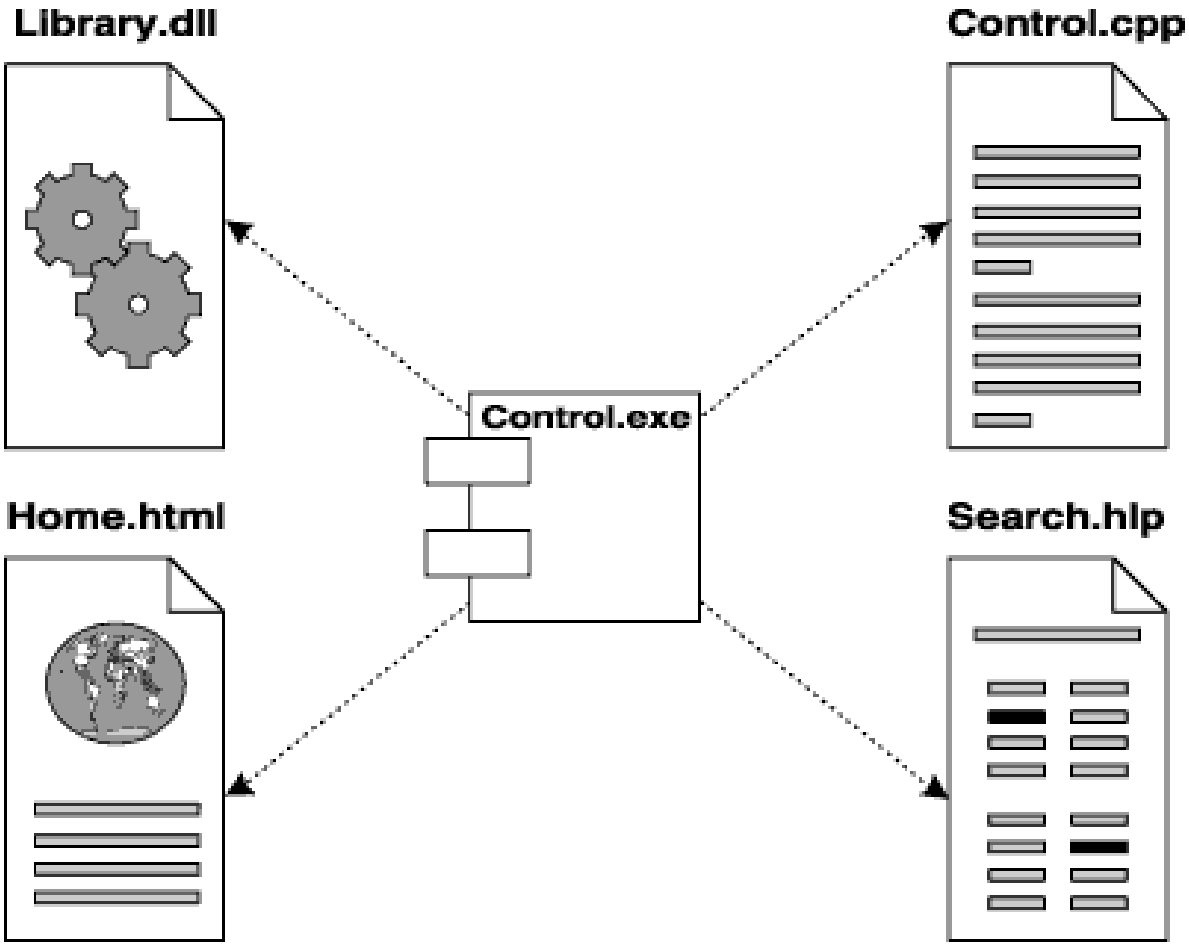


Рисунок 2.2 – Залежність між компонентами

Схема роботи з даними зображена на рисунку 2.3.

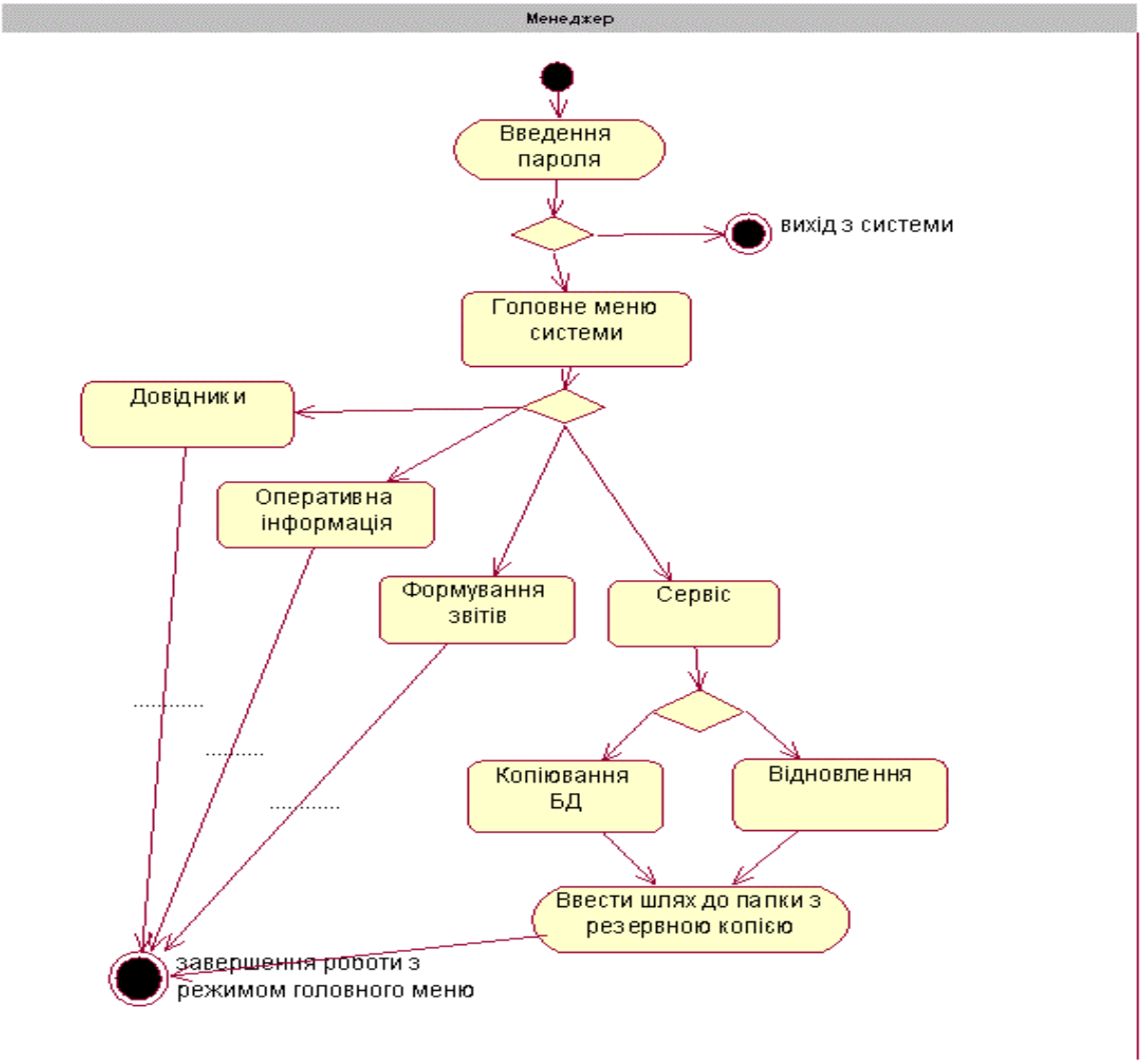


Рисунок 2.3 - Схема роботи з даними

2.4 Опис інтерфейсів

Для інтерфейсу буде підпорядкована сутність, а сутності будуть підпорядковуватися контролери, модулі, граничні сутності та сервіс.

Діаграма наведена на рисунку 2.4

Опис інтерфейсів зображено у вигляді «компонентної діаграми». До інтерфейсів буде належати маршрут, номер автобусу, ремонтні роботи, випуск автобусів на лінію, облік співробітників, створення звітності.

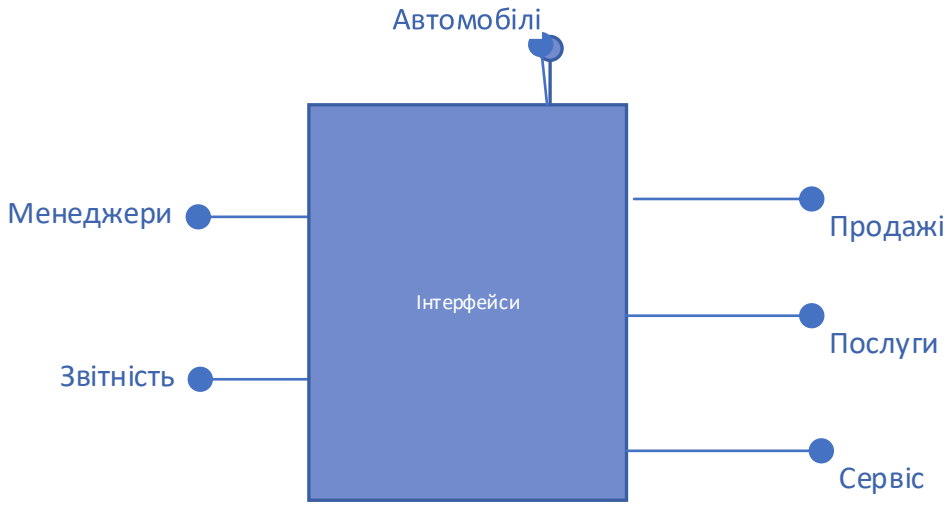


Рисунок 2.4 – Інтерфейси інформаційної системи

Загалом, у даному розділі здійснено проектування архітектури та структури системи, здійснено декомпозицію модулів з детальним описом декомпозиції, наведено опис залежностей модулів та подано опис інтерфейсів інформаційної системи діяльності автосалону .

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Детальне проектування модулів та реалізація бази даних

Процес побудови включає в себе уточнення моделі бази даних для уточнення структури користувальницького інтерфейсу; генерації SQL-пропозицій; побудова структурних схем, які відображають логіку роботи призначеного для користувача інтерфейсу. На етапі детального проектування будується модульна модель. Під модульною моделю розуміється реальна модель проектованої прикладної системи. Алгоритм роботи модуля програмного продукту представлений у вигляді «діаграми послідовності» на рисунку 3.1.

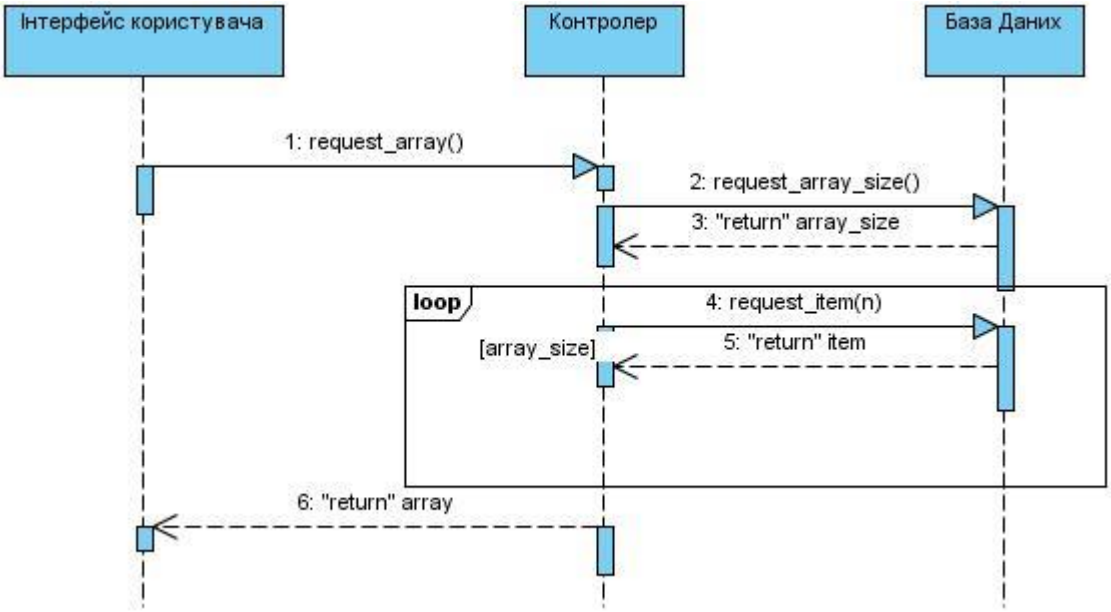


Рисунок 3.1 – Детальне проектування модуля

Маючи детальний опис предметної області можна переходити до виділення сутностей предметної області.

Неформальну модель даного проекту можна представити таким текстом:

— у салоні ведеться облік автомобілів у спеціальному журналі;

— у разі відстуності автомобіля у салоні виконується запит на поставку даного автомобіля;

— облік співробітників разом ведеться у спеціальному журналі;

— при здійсненні продажу співробітник збирає необхідну інформацію про покупця;

— інформація про продаж заноситься у спеціальний журнал.

При розробці слід враховувати такі дані:

— про кожен автомобіль слід зберігати код автомобіля, марку автомобіля, модель, назву виробника, колір автомобіля, вартість, дату виробництва, дату продажу;

— про кожного покупця слід зберігати код покупця, ПІП, адреса та контактні дані;

— про кожного продавця слід зберігати код продавця, ПІП, посаду;

— про виробника слід зберігати код виробника, назву виробника, країну виробника;

— про запчастини слід зберігати код запчастини, її назву та код автомобіля;

— про аксесуари слід зберігати код аксесуару, його назву, назву виробника та код певного автомобіля.

Виділимо сутності нашої предметної області:

– автомобіль – сутність, що містить інформацію про автомобіль, його марку, виробника, модель, ціну, дату випуску, колір;

– клієнт – сутність, що містить інформацію про клієнта;

– менеджер – сутність, що містить інформацію про менеджера консультанта;

– замовлення – сутність, що містить інформацію про замовлений автомобіль, клієнта-замовника та продавця-консультанта, дату замовлення, стан замовлення.

У сутності «Автомобіль» можна виділити наступні атрибути:

– марка;

– виробник;
– модель;
– ціна;
– колір;
– дата випуску.

У сутності «Клієнт» можна виділити наступні атрибути:

– прізвище;
– ім’я;
– по батькові;
– адреса;
– телефон;
– паспортні дані.

У сутності «менеджер» можна виділити наступні атрибути:

– прізвище;
– ім’я;
– по батькові;
– вік;
– телефон.

У сутності «Замовлення» можна виділити наступні атрибути:

– автомобіль;
– клієнт;
– менеджер;
– дата;
– стан.

Проводячи аналіз виділених сутностей предметної області видно, що один клієнт може оформити одне чи багато замовлень, один консультант може оформляти одне чи багато замовлень, один автомобіль може бути замовлений один або багато разів. Отже можна побудувати загальну схему отриманих відношень, як це зображено на рисунку 3.2.

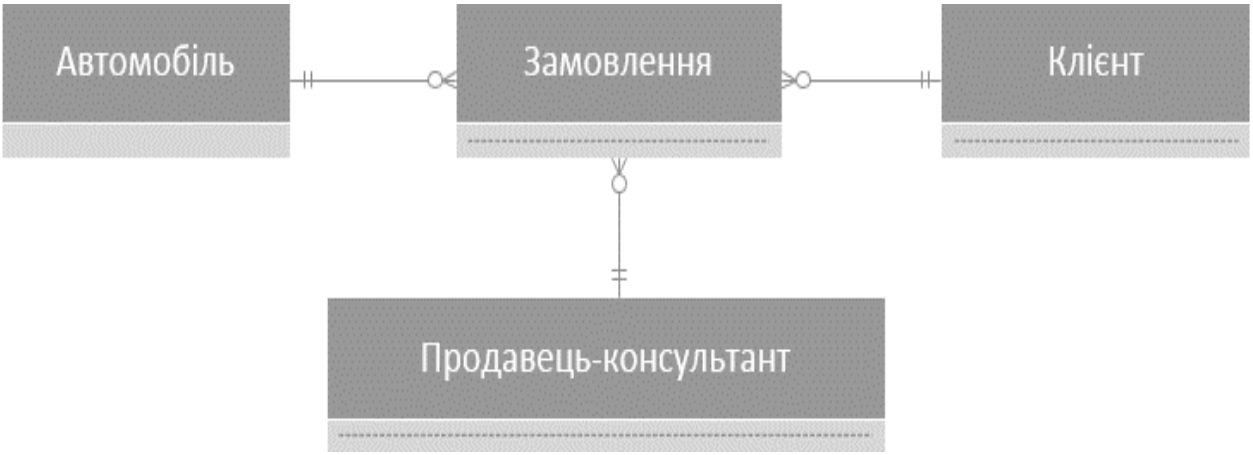


Рисунок 3.2– Схема початкових відношень сутностей

3.2 Приведення відношень до 3-ї нормальної форми

Задля забезпечення застосування кращих методів додавання, зміни та видалення даних, необхідно провести процес нормалізації.

Спочатку потрібно привести БД до 1-ї нормальної форми. Відношення знаходиться в 1НФ, якщо:

– відношення має унікальний первинний ключ;
– у відношенні немає однакових кортежів;
– кортежі не впорядковані;
– атрибути не упорядковані і мають різні назви.

Виходячи із цього означення нам потрібно виконати декілька наступних дій для створення бази даних.

Додаємо до кожної сутності унікальні первинні ключі: «Автомобіль» - номер_автомобіля; «Клієнт» - номер_клієнта; «Консультант» - номер_консультанта; «Замовлення» - номер_замовлення.

Атрибут «модель» винесемо як окрему сутність з атрибутами: «номер_моделі», «назва», «об’єм двигуна», «трансмісія», «привід», «тип_кузова»,

«тип_пального». «Номер_ моделі» - унікальний первинний ключ даного відношення.

Атрибут «Виробник» винесемо як окрему сутність з атрибутами: «номер_виробника», «назва», «країна», «адреса», «телефон», з первинним ключем «номер_виробника».

Тепер схема відношень отримала наступний вигляд, що приведений на рисунку 3.3.

Рисунок 3.3 – Схема змінених відношень сутностей.

Виходячи із предметної області, можна стверджувати, що одна модель може належати багатьом автомобілям. Також, можна сказати, що один виробник може випускати багато автомобілів.

Тепер відношення бази даних відповідають першій нормальній формі.

На наступному етапі потрібно привести базу даних до 2-ї нормальної форми. Відношення знаходиться в 2НФ тоді і тільки тоді, коли відношення знаходиться в 1НФ і немає не ключових атрибутів, залежних від частини складного ключа. Оскільки наші відношення вже знаходиться в 1НФ та первинні ключі прості, то відношення знаходяться в 2НФ.

Відношення знаходиться в 3-й нормальній формі, тоді і тільки тоді, коли воно знаходиться в 2НФ і всі не ключові атрибути взаємно незалежні. Аналізуючи модель, визначаємо, що відношення знаходяться у 3НФ.

До початкової моделі було додано такі сутності:

– модель – сутність, що містить інформацію про модель автомобіля, його характеристики, комплектацію;

– виробник – сутність, що містить інформацію про виробника автомобіля, країну виробника, його контактну інформацію.

Розглянемо повний опис предметної області, отриманий в процесі нормалізації. Клієнт здійснює купівлю автомобіля в автосалоні шляхом оформлення замовлення. Замовлення оформлюється лише на одного клієнта, але один клієнт може оформити багато замовлень. Замовлення складається з даних про автомобіль, даних про продавця-консультанта, даних про клієнта та дати замовлення. Замовлення оформляє продавець консультант. В одному замовлені містяться дані лише про одного продавця консультанта. Один продавець-консультант може оформляти безліч замовлень. Данні про автомобіль складаються з даних про марку, модель, виробника, колір, дату випуску, ціну. Один автомобіль має лише одну модель та одного виробника, але одна модель відповідає декільком автомобілям та один виробник може випустити безліч автомобілів.

3.3 Побудова концептуальної ER-діаграми бази даних

Інфологічний рівень являє собою інформаційно-логічну модель предметної області, в якій виключена надмірність даних і відображені інформаційні особливості об’єкта управління, без урахування особливостей і специфіки конкретної СУБД. Він може бути самостійним або функціонувати як складова зовнішнього рівня. Інтеграція всіх зовнішніх представлень даних виконується на інфологічному рівні. На цьому рівні формується інфологічна (канонічна) модель даних.

Мета інфологічного проектування — створити структуровану інформаційну модель предметної області, для якої розроблятиметься БД. Під час

проектування на інфологічному рівні створюється інформаційно-логічна модель, яка має відповідати таким вимогам:

– коректність схеми БД, тобто адекватне відображення модельованої предметної області;

– простота і зручність використання на наступних етапах проектування, тобто ІЛМ має легко відображатися в моделі БД, що підтримується відомими СКБД (мережеві, ієрархічні, реляційні);

– ІЛМ має бути описана мовою, зрозумілою проектувальникам бази даних, програмістам, адміністратору і майбутнім користувачам.

Основною складовою інфологічної моделі є атрибути, які потрібно проаналізувати і деяким чином згрупувати для подальшого зберігання в БД.

Сутність інфологічного моделювання полягає у виокремленні інформаційних об’єктів ПО (сутностей), які підлягають зберіганню в БД, а також визначення характеристик об’єктів і зв’язків між ними. Характеристиками чи властивостями об’єктів є атрибути.

В реальному проектуванні БД застосовується метод семантичного моделювання, що являє собою моделювання структури даних спираючись на зміст даних. В якості інструменту використовується діаграма «Сутність-зв’язок» чи «ER – Entity Relationship».

Отже на основі вище приведеного опису здійснюємо побудову ER – діаграми використовуючи нотації Баркера. Результат розробки приведений на рисунку 3.4.

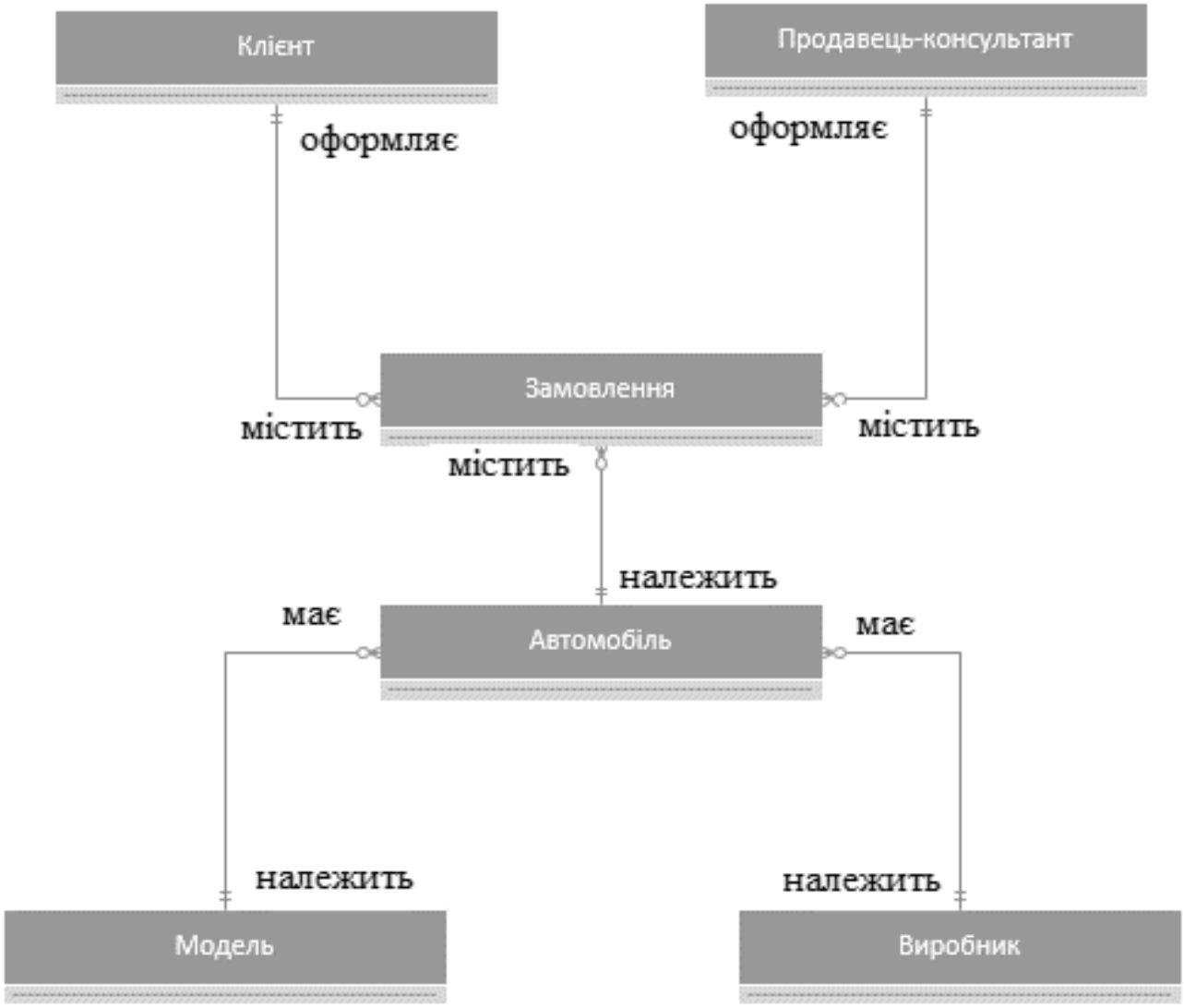


Рисунок 3.4 – Концептуальна ER – діаграма БД.

3.4 Вибір та обґрунтування СКБД

Перед вибором СКБД необхідно визначитися із моделлю даних. На сьогодні розрізняють такі моделі: ієрархічні, мережеві, реляційні. Ми будемо використовувати реляційну модель даних. Для реляційної моделі характерна структура основана на реляційних таблицях. В реляційній таблиці дані зберігаються у вигляді двовимірних таблиць, які називаються відношеннями або

плоскими файлами. Реляційні бази даних стали найбільш поширеними завдяки таким своїм перевагам:

– простота nf доступність для розуміння користувачем;

– строгі правила проектування, які базуються на математичному апараті;

– дані в таблиці є незалежними одне від одного, що дозволяє оперативно змінювати структуру бази даних, внаслідок чого всі зв'язки в цій моделі легко змінюються;

– розширення структури баз даних здійснюється простим додаванням нової таблиці.

Виходячи з отриманої інформаційно-логічної моделі, а також характеристик існуючих моделей баз даних, спираючись на значні перевари, було обрано реляційну модель баз даних.

Система керування базами даних (СКБД) – комп'ютерна програма або комплекс програм, що забезпечують користувачам можливість збереження, створення, оновлення, пошуку інформації та контролю доступу в базах даних.

Серед існуючих СКБД будо обрано MySql 5.5. Це вільна система керування реляційними базами даних. MySql характеризується високою швидкістю, стійкістю та простотою використання. Вона вважається гарним рішенням для малих та середніх застосувань.

Можливості MySql:

– простота у встановлені та використанні;

– підтримується необмежена кількість користувачів, що одночасно працюють з БД.

– кількість рядків у таблицях може досягати 50 мільйонів;

– висока швидкість виконання команд;

– наявність простої та ефективної системи безпеки.

Проаналізувавши ці дані, було вирішено, що дана СКБД повністю підходить під потреби проектованої нами інформаційної системи моніторингу діяльності автосалону.

3.5 Побудова фізичної моделі даних

Логічна модель - відображає інформацію про предметну область у вигляді, незалежному від СКБД, що використовується. В нашому випадку для побудови фізичної моделі ми будемо використовувати Erwin Data Modeler.

Для розробки даної моделі нам потрібно мати інформацію про список сутностей, зв’язки між ними та список атрибутів.

Сутність - будь-який конкретний або абстрактний об'єкт в предметної області. Кожна сутність повинна мати найменування, виражене іменником в однині. Прикладами сутностей можуть бути: «Продавець», «Покупець».

Атрибут - це властивість сутності у предметній області[5]. Найменування атрибута має бути виражене іменником в однині (можливо, з прикметниками). Прикладами атрибутів сутності «Покупець» можуть бути: «Прізвище», «Номер».

Зв'язок - взаємозв'язок між сутностями в предметній області[6]. Одна сутність може бути пов'язана з іншою сутністю або сама з собою. Зв'язки дозволяють по одній сутності знаходити інші суті, пов'язані з нею.

Існує декілька типів зв’язків:

— один-до-одного (1:1) - зв’язок, при якому одна сутність в першій таблиці відповідає тій самій сутності в другій таблиці;

— один-до-багатьох (1:б) – зв’язок, при якому одна сутність однієї таблиці відповідає багатьом сутностям другої;

— багато-до-багатьох (б:б) – зв’язок, при якому декілька екземплярів однієї сутності може відповідати декільком екземплярам другої.

Логічна ER-діаграма бази даних наведена на рисунку 3.5.

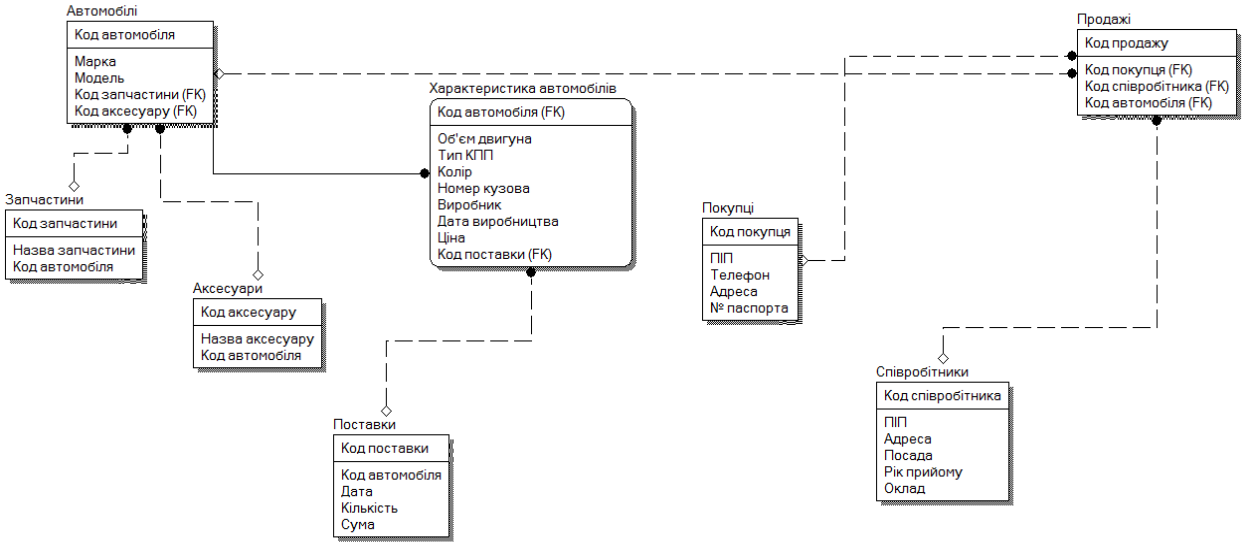


Рисунок 3.5 – Логічна ER-діаграма

Для того, щоб створити базу даних в MySQL потрібно зробити наступне:

— спочатку потрібно створити базу даних та дати їй назву. Це робиться наступним чином: при запуску MySQL на вкладці Створення вибираємо Порожня база даних. В наступному вікні нам діється змога задати назву нашої бази даних та вказати місце збереження;

— далі потрібно створити таблиці та додати до них поля наступним чином: переходимо у вкладку Створення. В ній вибираємо Таблиця. Після цього в меню об’ктів з’явиться наша таблиця. Натискаємо на неї правою кнопкою миші та вибираємо пункт Перейменувати, після чого вказуємо необхідну назву таблиці;

— далі додаємо до таблиця необхідні поля. Для зручності будемо використовувати Конструктор. Натискаємо на тиаблицю правою кнопкою миші та переходимо у вкладку Конструктор. Далі додаємо назви необхідних полів та встановлюємо їх тип даних;

— наступним кроком буде встановлення зв’язків між створеними таблицями. На вкладці Робота з базами даних вибираємо Схема даних». У відкрившомуся вікні вибираємо таблиці, з якими ми будемо працювати. Далі

перетягуємо поле одної таблиці (як правило, поле первинного ключа) на поле іншої (зовнішній ключ). У наступному вікні перевірте дані та натисніть Ок;

— далі потрібно заповнити таблиці необхідною інформацією. Натискаємо правою кнопкою миші по таблиці та вибираємо Режим таблиці. Далі заповнюємо необхідні колонки інформацією.

Фізична модель – описує те, як дані зберігаються в комп'ютері, представляючи інформацію про структуру записів, їх впорядкованість і про існуючі шляхи доступу до даних. У життєвому циклі проекту вона типово походить від логічної моделі даних, хоча вона може бути зворотно розроблена з даної реалізації бази даних (рисунок 3.6).

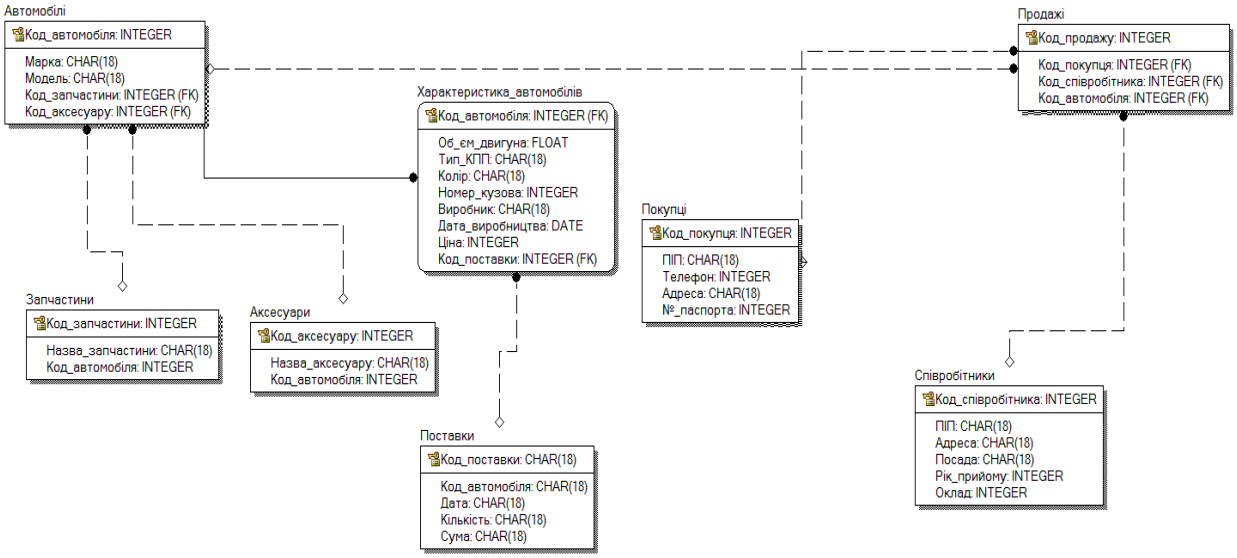


Рисунок 3.6 – Фізична ER-діаграма

Завершена фізична модель даних включає всі артефакти бази даних, необхідні для створення відношень між таблицями чи для досягнення мети продуктивності, як індексів, визначень обмежень, зв'язаних і таблиць.

Опишемо нашу модель у обраній СКБД. Покажемо властивості атрибутів, сутностей бази даних, що необхідно створити відповідно до опису даних обраної СКБД. Назви полів атрибутів, типи, обмеження, призначення як ключового поля приведено в таблиці 3.1.

Таблиця 3.1 - Поля таблиць БД.

Поле	Тип	Обмеження	Ключ	Таблиця
Model_id	Int	not null	primary key	Model
Model_name	Varchar(50)	not null		Model
Motor_value	Double	not null		Model
Fuel	Varchar(50)	not null		Model
Driver	Varchar(50)	not null		Model
Transmission	Varchar(50)	not null		Model
Body_type	Varchar(50)	not null		Model
Car_id	Int	not null	primary key	Car
Marka	Varchar(50)	not null		Car
Model_id	Int	not null	foreign key	Car
Maker_id	Int	not null	foreign key	Car
Year_of	Int	not null		Car
Colour	Varchar(50)	not null		Car
Price	Double	not null		Car
Maker_id	Int	not null	primary key	Maker
Maker_name	Varchar(50)	not null		Maker
Country	Varchar(50)	not null		Maker
City	Varchar(50)	not null		Maker
Address	Varchar(50)	not null		Maker
Phone	Varchar(50)	not null		Maker
Order_id	Int	not null	primary key	Order
Client_id	Int	not null	foreign key	Order
Advisor_id	Int	not null	foreign key	Order
Car_id	Int	Not null	foreign key	Order
Fleg	Varchar(50)	not null		Order

Date	Date	not null		Order
Client_id	Int	not null	primary key	Client
First_name	Varchar(50)	not null		Client
Surname	Varchar(50)	not null		Client
Middle_name	Varchar(50)	not null		Client
Phone	Varchar(50)	not null		Client
Address	Varchar(50)	not null		Client
Passport_data	Varchar(50)	not null		Client
Advisor_id	Int	not null	primary key	Advisor
Surname	Varchar(50)	not null		Advisor
First_name	Varchar(50)	not null		Advisor
Middle_name	Varchar(50)	not null		Advisor
Year	Int	not null		Advisor
Phone	Varchar(50)	not null		Advisor

Адміністрування СКБД MySQL будемо проводити за допомогою PhpMyAdmin. PhpMyAdmin — веб-застосунок з відкритим кодом на мові PHP із графічним веб-інтерфейсом для адміністрування СКБД MySQL. PhpMyAdmin дозволяє через браузер здійснювати адміністрування сервера MySQL, запускати запити SQL, переглядати та редагувати вміст таблиць баз даних. Ця програма користується великою популярністю у веб-розробників, оскільки дозволяє керувати СКБД MySQL без безпосереднього вводу SQL команд в коді, а через дружній інтерфейс і з будь-якого комп'ютера під'єднаного до Інтернету без необхідності встановлення додаткового програмного забезпечення на цей комп’ютер.

Створювати таблицю за допомогою цього інструменту можна здійснювати двома способами: скриптом, використовуючи спеціальну мову Transact-SQL або графічним конструктором.

Створення бази даних зробимо за допомогою наступного SQL коду:

```
CREATE  DATABASE `car_show`;
```

Створимо таблицю «Client» за допомогою скрипту, з використанням мови Transact-SQL:

```
CREATE TABLE IF NOT EXISTS `client` (
  `client_id` int(11) NOT NULL AUTO_INCREMENT,
  `surname` varchar(255) NOT NULL,
  `first_name` varchar(255) NOT NULL,
  `middle_name` varchar(255) NOT NULL,
  `address` varchar(255) NOT NULL,
  `phone` varchar(255) NOT NULL,
  `passport_data` varchar(255) NOT NULL,
  PRIMARY KEY (`client_id`)
) ENGINE=MyISAM  DEFAULT CHARSET=utf8 AUTO_INCREMENT=1;
```

В першому рядку виконується створення таблиці «client». Наступні сім рядків виконують додавання полів таблиці, вони мають таку структуру: «назва поля», «тип даних», «обмеження», «дозвіл на null значення». Поле AUTO_INCREMENT забезпечує автоматичне заповнення ключового поля в таблиці. Запис PRIMARY KEY (`client_id`) означає, що поле «client_id» є ключовим.

Також можна створювати таблиці за допомогою графічного конструктора PhpMyAdmin. Всі інші таблиці створимо в графічному редакторі. Зовнішні ключі цьому разі додаються при створенні діаграми, шляхом з'єднання відповідних полів.

Таблицю «Продавець-консультант» створимо в графічному конструкторі, з відповідними полями. Результат створення можна побачити на рисунку 3.7.

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно
1	advisor_id	int(11)			Нет	*Нет*	AUTO_INCREMENT
2	surname	varchar(255)	utf8_general_ci		Нет	*Нет*	
3	first_name	varchar(255)	utf8_general_ci		Нет	*Нет*	
4	middle_name	varchar(255)	utf8_general_ci		Нет	*Нет*	
5	year	int(11)			Нет	*Нет*	
6	phone	varchar(255)	utf8_general_ci		Нет	*Нет*	

Рисунок 3.7 – Структура таблиці «Продавець-консультант»

Таблицю «Автомобіль» створимо в графічному конструкторі, з

відповідними полями. Результат створення можна побачити на рисунку 3.8.

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно
1	car_id	int(11)			Нет	Нем	AUTO_INCREMENT
2	marka	varchar(255)	utf8_general_ci		Нет	Нем	
3	model_id	int(11)			Нет	Нем	
4	maker_id	int(11)			Нет	Нем	
5	year_of	int(4)			Нет	Нем	
6	color	varchar(255)	utf8_general_ci		Нет	Нем	
7	price	int(11)			Нет	Нем	

Рисунок 3.8 – Структура таблиці «Автомобіль»

Таблицю «Клієнт» створимо в графічному конструкторі, з відповідними полями. Результат створення можна побачити на рисунку 3.9.

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно
1	client_id	int(11)			Нет	Нем	AUTO_INCREMENT
2	surname	varchar(255)	utf8_general_ci		Нет	Нем	
3	first_name	varchar(255)	utf8_general_ci		Нет	Нем	
4	middle_name	varchar(255)	utf8_general_ci		Нет	Нем	
5	address	varchar(255)	utf8_general_ci		Нет	Нем	
6	phone	varchar(255)	utf8_general_ci		Нет	Нем	
7	passport_data	varchar(255)	utf8_general_ci		Нет	Нем	

Рисунок 3.9 – Структура таблиці «Клієнт»

Таблицю «Виробник» створимо в графічному конструкторі, з відповідними полями. Результат створення можна побачити на рисунку 3.10.

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно
1	maker_id	int(11)			Нет	Нем	AUTO_INCREMENT
2	maker_name	varchar(255)	utf8_general_ci		Нет	Нем	
3	country	varchar(255)	utf8_general_ci		Нет	Нем	
4	city	varchar(255)	utf8_general_ci		Нет	Нем	
5	address	varchar(255)	utf8_general_ci		Нет	Нем	
6	phone	varchar(255)	utf8_general_ci		Нет	Нем	

Рисунок 3.10 – Структура таблиці «Maker»

Таблицю «Модель» створимо в графічному конструкторі, з відповідними полями. Результат створення можна побачити на рисунку 3.11.

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно
1	model_id	int(11)			Нет	Нем	AUTO_INCREMENT
2	model_name	varchar(255)	utf8_general_ci		Нет	Нем	
3	motor_valume	double			Нет	Нем	
4	fuel	varchar(255)	utf8_general_ci		Нет	Нем	
5	drive	varchar(255)	utf8_general_ci		Нет	Нем	
6	body_type	varchar(255)	utf8_general_ci		Нет	Нем	
7	transmission	varchar(255)	utf8_general_ci		Нет	Нем	

Рисунок 3.11 – Структура таблиці «Модель»

Таблицю «Замовлення» створимо в графічному конструкторі, з відповідними полями. Результат створення можна побачити на рисунку 3.12.

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно
1	order_id	int(11)			Нет	Нем	AUTO_INCREMENT
2	car_id	int(11)			Нет	Нем	
3	client_id	int(11)			Нет	Нем	
4	advisor_id	int(11)			Нет	Нем	
5	date	date			Нет	Нем	
6	fleg	varchar(255)	utf8_general_ci		Нет	Нем	

Рисунок 3.12 – Структура таблиці «Замовлення»

Після створення всіх таблиць, можемо переходити до їх заповнення. Заповнення будемо проводити за допомогою за допомогою запиту.

Приклад заповнення таблиці «Client» даними про нового клієнта:

INSERT INTO `car_show`.`client` (`client_id`, `surname`, `first_name`, `middle_name`, `address`, `phone`, `passport_data`) VALUES (NULL, 'Осійчук', 'Тетяна', 'Вікторівна', 'м. Хмельницький, вул. Інститутська 12', '0965487963', 'НВ 458963');

Запити в інформаційній системі використовуються для вибірки, додавання, оновлення або видалення даних в одній або декількох таблицях. Перед тим як створювати запити потрібно створити необхідні таблиці та заповнити їх даними. Далі ми розглянемо на прикладах основні типи запитів.

Основні типи запитів:

— запит на вибірку – цей запит вибирає з однієї або декількох таблиць дані, що відповідають заданій умові. Наприклад, вибиремо із таблиці «Характеристика автомобілів» інформацію про автомобілі з механічною КПП (Рисунок 3.13)

Поле:	Код автомобіля	Об'єм двигуна	Тип КПП	Колір	Виробник	Дата виробництва	Ціна
Имя таблицы:	Характеристика авто	Характеристика авто	Характеристика авто	Характеристика авто	Характеристика авто	Характеристика автомо	Характеристика авто
Сортировка:							
Вывод на экран:	☑	☑	☑	☑	☑	☑	☑
Условие отбора:			"Механіка"				
или:							

Рисунок 3.13 – Запит на вибірку

— запит з параметром – цей запит використовується для введення з клавіатури умови на початку його виконання. Наприклад, створимо запит з параметром для таблиці «Поставки». У ній ми зробимо запит на виведення даних про поставки, здійснені після введеної нами дати (Рисунок 3.14).

Поле:	Код поставки	Код автомобіля	Дата	Кількість	Сума
Имя таблицы:	Поставки	Поставки	Поставки	Поставки	Поставки
Сортировка:					
Вывод на экран:	☑	☑	☑	☑	☑
Условие отбора:			> [Введіть дату поставки:]		
или:					

Рисунок 3.14 – Запит з параметром

— запит на видалення – цей запит видаляє необхідні дані з однієї або декількох таблиць. Для прикладу створимо запит, при якому введиться код автомобіля і видаляється рядок з відповідним кодом із таблиці «Автомобілі». (Рисунок 3.15)

Поле:	Код Автомобіля
Имя таблицы:	Автомобілі
Удаление:	Условие
Условие отбора:	[Введіть код автомобіля]
или:	

Рисунок 3.15 – Запит на видалення

— запит на оновлення – даний запит оновлює дані в одній або декількох таблицях. Для прикладу створимо запит, який зменшить ціни автомобілів на 10% (Рисунок 3.16).

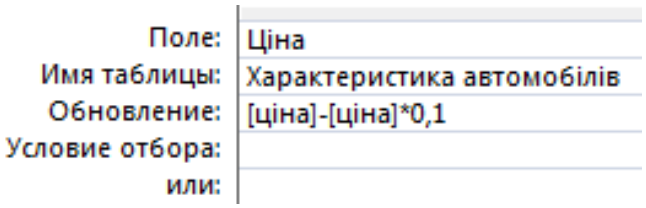


Рисунок 3.16 – Запит на оновлення

— запит на додавання – цей запит додає відповідні дані в одну або декілька таблиць. Наприклад, додамо в таблицю «Автомобілі» новий автомобіль. Для цього створимо запит на додавання, який дозволить користувачу власноруч додати нові записи (Рисунок 3.18).

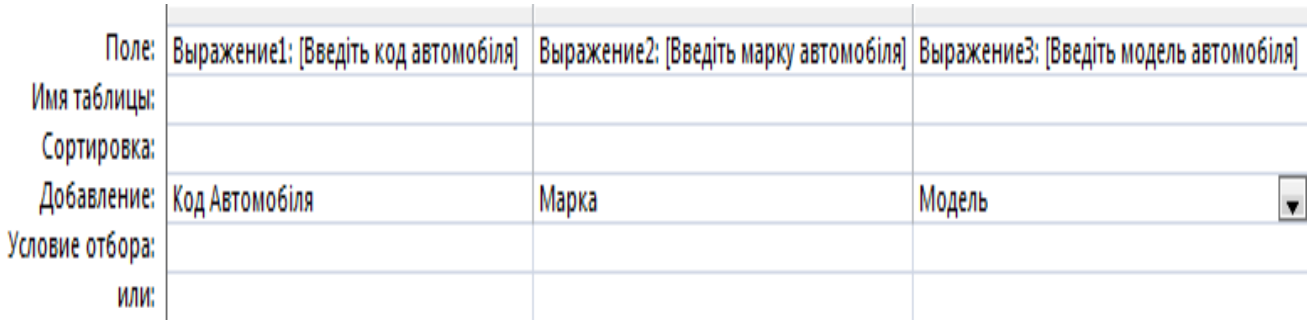


Рисунок 3.18 – Запит на додавання

3.6 Створення інтерфейсу для роботи з базою даних

Для реалізації інтерфейсу програми скористаємось утилітою MYSQL CONNECTOR NET, яка дозволяє розробляти .NET-додатки, які вимагають безпечних, високоефективних можливостей з'єднання даних з MySQL. він здійснює необхідні інтерфейси ADO.NET і об'єднується з інструментами ADO.NET. Можна створити додатки, використовуючи мови .NET. Connector/NET

керований провайдер даних ADO.NET, написаний на мові C# . Сам інтерфейс створимо в з допомогою середовища розробки MS Visual Studio та MS Access.

Connector/NET включає повну підтримку:

– Особливостей, які забезпечені MySQL Server включаючи MySQL 8.0.

– MySQL як сховище документів (NoSQL), поряд зі зв'язком X-протоколу, щоб отримати доступ до даних MySQL, використовуючи порти X Plugin.

– Підтримка великого пакета відправлення та отримання рядків та значень BLOB до 2 гігабайт у розмірі.

– Стиснення протоколу, яке дозволяє стиснути потік даних між самим клієнтом та сервером.

– Зв'язки з використанням сокетів TCP/IP, іменовані канали та загальну пам'ять Windows.

– Зв'язки з використанням сокетів TCP/IP та сокетів Unix.

– Зашифроване використання зв'язків: TLSv1.2 через TCP/IP з Connector/NET 8.0.11 та вище.

– Зашифроване використання зв'язків:TLSv1.3 через TCP/IP з Connector/NET 8.0.20 та вище.

– .NET Standard та виконання на реалізації .NET для Universal Windows Platform (UWP).

– Entity Framework 6 и Entity Framework Core, для міграції данних таблиць данних MySQL.

– Open Source Mono framework від Novell.

– Connector/NET співпрацює з Microsoft Visual Studio

Алгоритм створення інтерфейсу для бази даних:

— спочатку слід створити форму для кожної таблиці. Для цього на вкладці Створення вибираємо Майстер форм. У наступному вікні ми повинні вибрати таблицю та необхідні поля, для яких буде створена форма. Після цього слід вибрати зовнішній вигляд форми (у нашому випадку – в один стовпчик) та натиснути Готово;

— далі переходимо у режим Конструктора та виконуємо редагування створеної форми. Також, при необхідності, додаємо до нашої форми кнопки. В нашому випадку ми додали кнопки для переходу між записами, створення та видалення запису, пошуку запису, оновлення введених даних та виходу із форми (Рисунок 3.19);

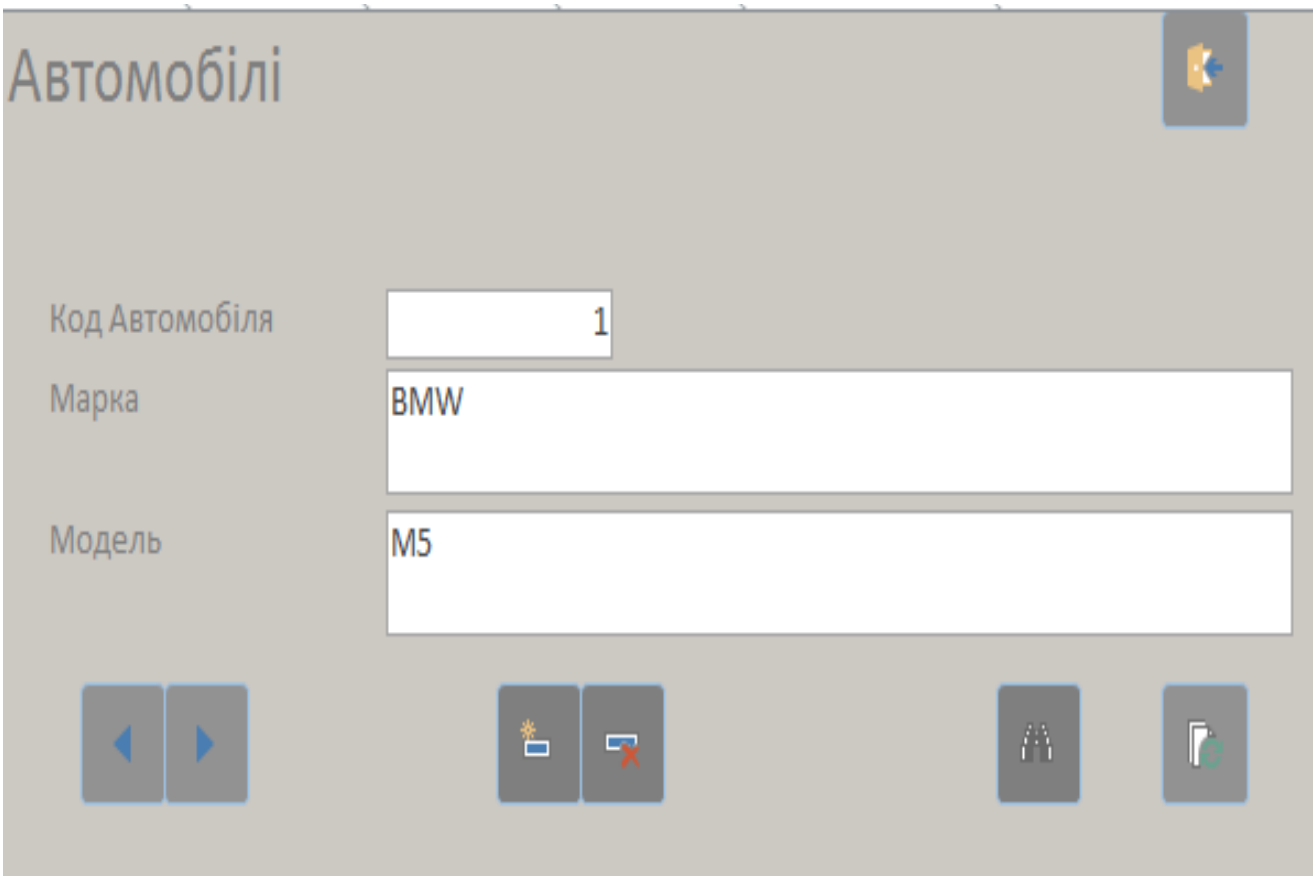


Рисунок 3.19 – Форма таблиці «Автомобілі»

— на наступному етапі створюємо безпосередньо головну форму нашого інтерфейсу. Для цього переходимо на вкладку Створення та вибираємо Конструктор форм. Для даної форми створюємо кнопки для відповідних таблиць. Для цього в меню Конструктор вибираємо об’єкт Кнопка та розміщуємо її у необхідному місці. Після цього відкриється меню в якому слід вибрати пункт Робота з формами та Відкрити форму. Далі вибираємо створену нами форму таблиці. Ці ж дії виконуємо для всіх інших форм таблиць (Рисунок 3.20).

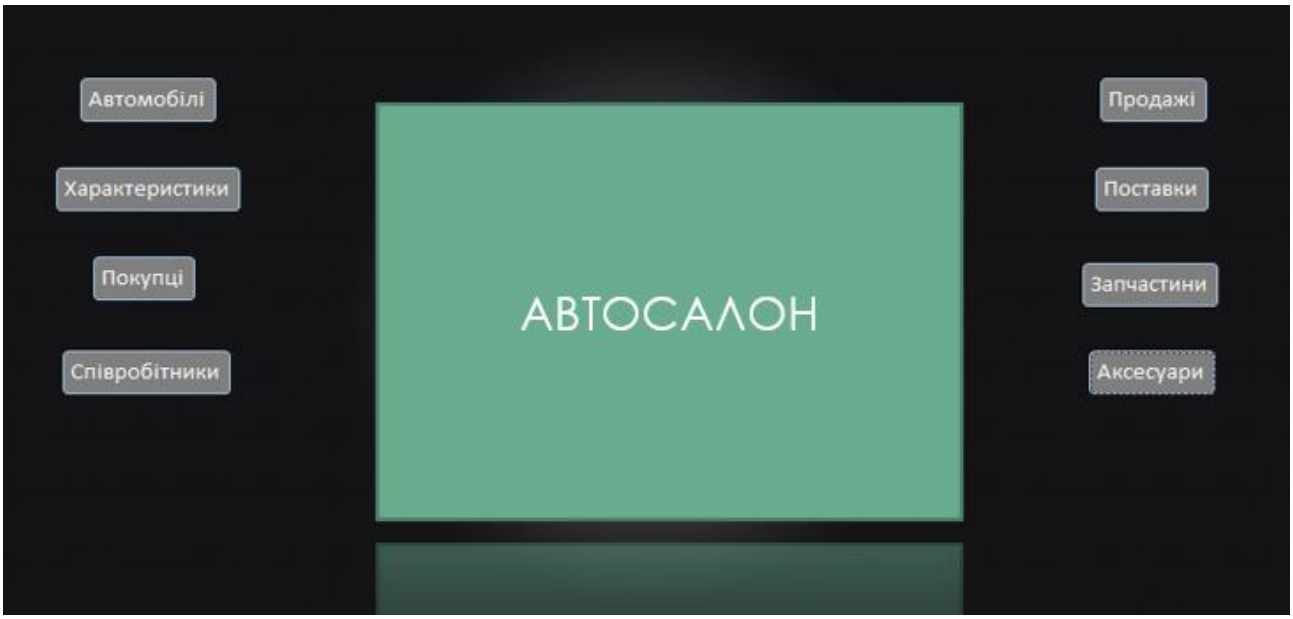


Рисунок 3.20 – Головна форма

Виконання програми відбулося без помилок, тому можемо перейти безпосередньо до її тестування:

— додавання автомобіля (Рисунок 3.21, 3.22):

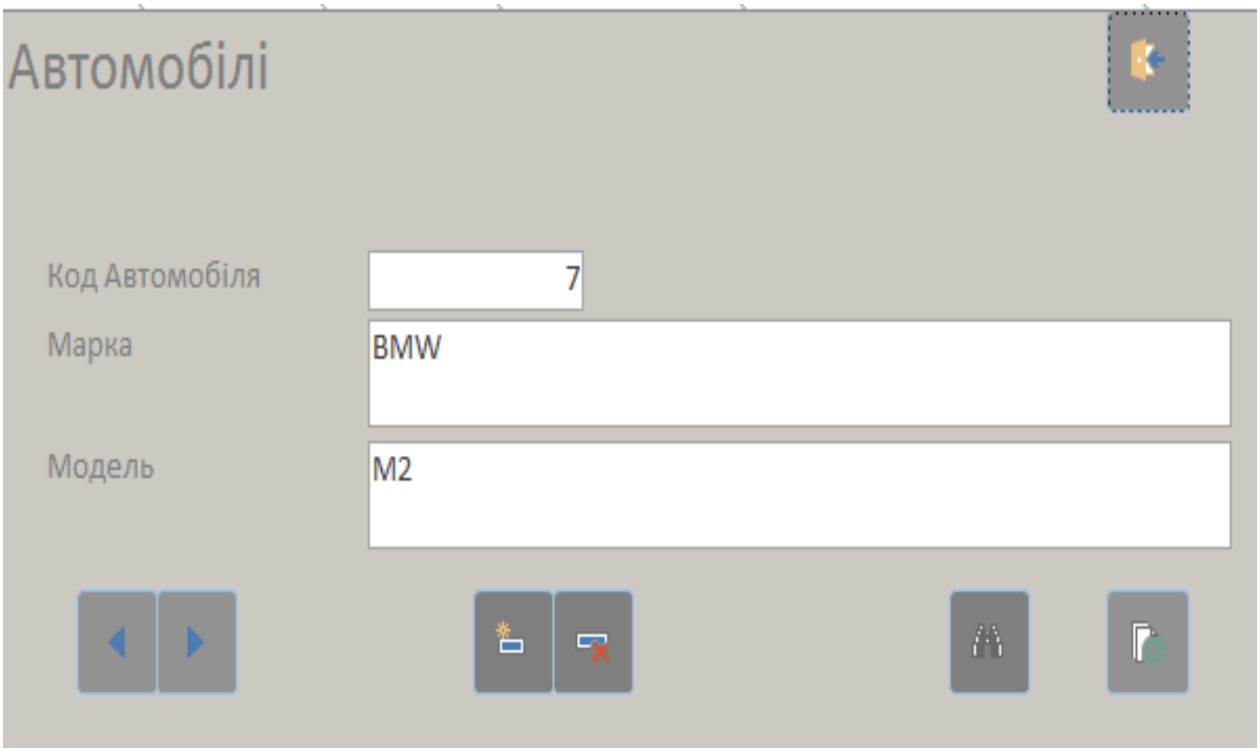


Рисунок 3.21 – Додавання автомобіля

Код Автомобіля	Марка	Модель
1	BMW	M5
2	BMW	X5
3	BMW	X5M
4	BMW	M3
5	BMW	X6
6	BMW	X1
7	BMW	M2

Рисунок 3.22 – Результат додавання

Видалення даних про автомобіль (Рисунок 3.23, 3.24):

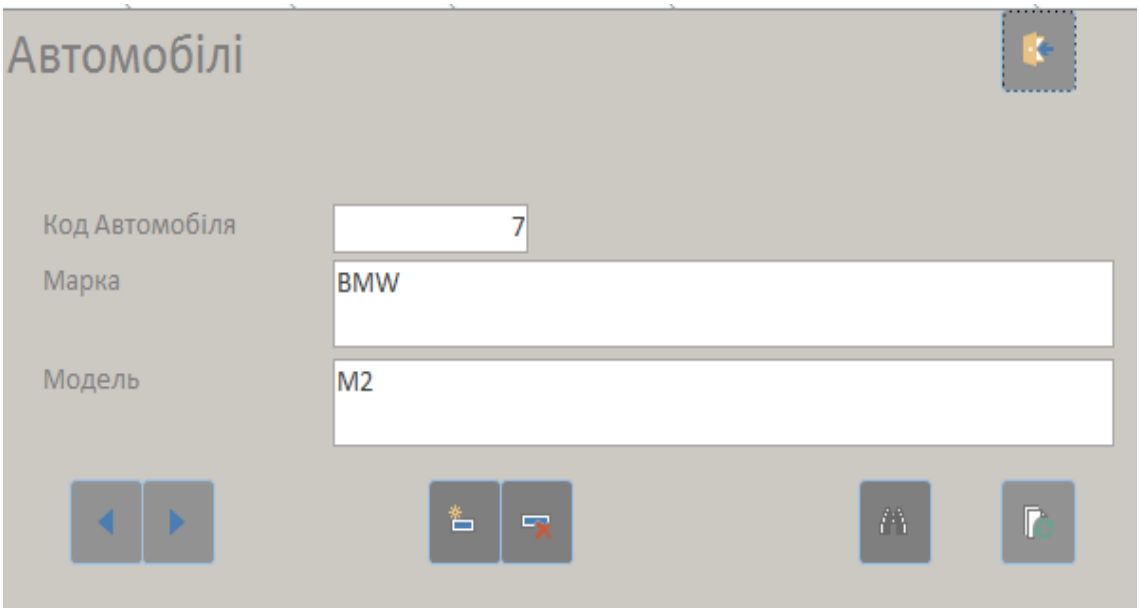


Рисунок 3.23 – Видалення автомобіля

Код Автомобіля	Марка	Модель
1	BMW	M5
2	BMW	X5
3	BMW	X5M
4	BMW	M3
5	BMW	X6
6	BMW	X1
#Удалено	#Удалено	#Удалено

Рисунок 3.24 – Результат видалення

Пошук інформації про автомобіль з механічною КПП зображено на рисунку 3.25 :

Рисунок 3.25 – Введення умови

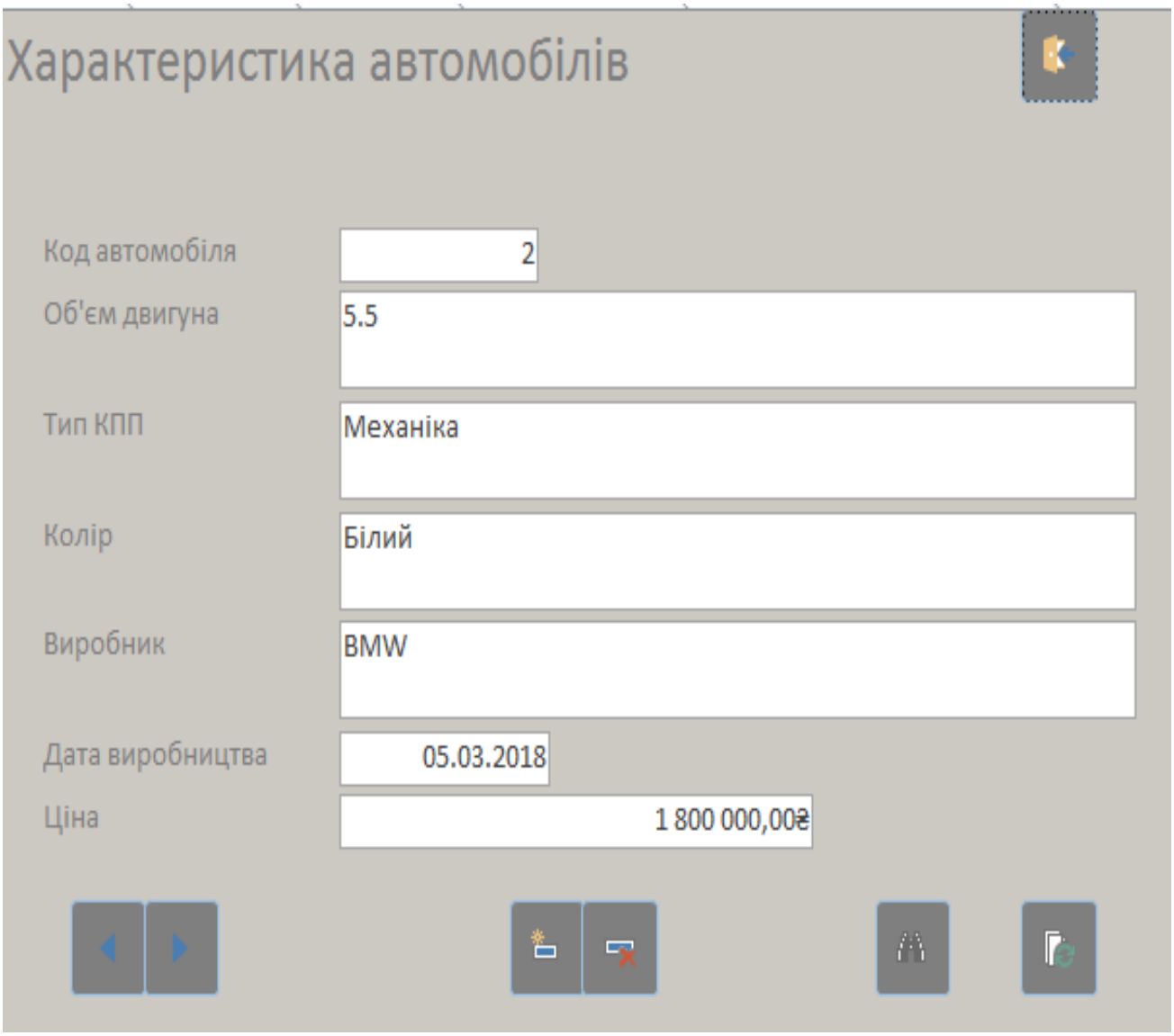


Рисунок 3.26 – Результат пошуку

Результати проведеного моніторингу менеджерами різних автосалонів міста Хмельницького наведено на рисунку 3.27

Рисунокку 3.27 – Результати моніторингу

3.7 Вимоги до технічних засобів

Вимоги до надійності. Програмний комплекс повинен стійко функціонувати і не приводити до збоїв системи.

Вимоги до технічних засобів можна поділити на такі категорії:

1) вимоги до програмного забезпечення:
   - має бути встановлена та налаштована сама інформаційна система моніторингу діяльності автосалону;
   - повинен бути встановлений конектор MySQL;
   - MySQL 8.0. або вище;
2) вимоги до апаратного забезпечення:
   - під’єднання до локальної мережі чи мережі Internet;
   - обсяг оперативної пам'яті від 256 MB;
   - процесор не нижче Pentium IV;

- вільне місце на жорсткому диску для зберігання проекту, сервера баз даних;
- вільне місце на жорсткому диску для зберігання бази даних;

Таким чином, у даному розділі здійснено детальне проектування модулів та реалізація бази даних з приведенням її таблиць до третьої нормальної форми. Здійснено вибір та обгрунтування СКБД та засобів розробки, наведено процес розробки інформаційної системи та продемонстровано її роботу в тестовому режимі з підтвердженням її роботи. Наведено вимоги до програмних та технічних засобів для роботи системи

4 НАЛАГОДЖЕННЯ ТА ТЕСТУВАННЯ СИСТЕМИ

4.2 Вибір та обґрунтування методів тестування системи

Однак для підвищення якості програмного забезпечення та, відповідно задоволення потреб користувачів інформаційної системи досить важливо врахувати всі різні характеристики самої інформаційної системи. Такі системи характеристик прийнято називати моделями якості програмного забезпечення.

Згідно до прийнятих норм та стандартів тестування програмного забезпечення виділять наступні найважливіші його характеристики програмного забезпечення:

– functionality – функціональність, яка визначається здатністю ПЗ вирішувати різні задачі, які відповідають вимогам користувача, при певних наперед замовлених умов його експлуатації. Саме ця характеристика відповідає вірність роботи ПЗ, відповідає усім стандартам галузі і, відповідно має захист від сторонніх осіб.

– reliabiity - надійність ПЗ. тобто його здатність виконувати поставлені на нього завдання у визначений термін, або заявлену кількість необхідних дій. Тут важливими параметрами є цілісність самої системи та ступінь завершеності, а також коректність її роботи незважаючи на некоректність її завершення після збоїв і відмовостійкість.

– usability - зручність використання ПЗ Тут мається на увазі легкість її використання, зручність в роботі, легке засвоєння функціоналу ПЗ для користувачів ПЗ тощо.

– efficiency– ефективність ПЗ забезпечувати потрібний рівень продуктивності у відповідності з виділеними ресурсами та умовами використання

– maintainability - зручність експлуатації, можливість зміни, аналізу зміни його версій, модифікації та адаптації до нових вимог.

– portability – дана характеристика означає можливість легкості переносу з одного різні системи та відповідну техніку.

Розглянемо сучасні моделі якості ПЗ, які використовуються на даний час.

Першою моделлю є модель якості програмного забезпечення Маккола.

Найбільш широко відомою моделлю якості програмного забезпечення є модель якості ПЗ запропонована Макколом. В цій моделі характеристики якості поділені на три групи:

factors - фактори, які описують ПЗ з точки зору користувача і заданих вимог до нього;

criteria - критерії, які описують ПЗ з точки зору розробників і є головною метою для реалізації

metrics - метрики, які використовуються для кількісного опису вимірювання якості ПЗ.

Факторів якості тут виділять одинадцять, які, в свою чергу, групують на три різних групи .

Критерії якості - це різні числові рівні визначених факторів, поставлені за мету при розробці ПЗ . Оцінюються такі фактори за деситибальною шкалою.

Маккол виділив наступні метрики якості:

– audiability - зручність перевірки на відповідність прийнятим стандартам якості;

– accuracy - точність;

– communication commonality – стандартність інтерфейсів;

– completeness - повнота ПЗ;

– consistency - однорідність правил проектування супровідної документації ПЗ;

– data commonality – міра до стандарну формату даних;

– error tolerance _-_ стійкість до помилок;

– execution efficiency - ефективність роботи ПЗ;

– expandability – можливість до масштабованості та розширювання;

– generality – можливість до широкого використання;

– hardware independence - незалежність від апаратних засобів;

– software system independence – незалежність від програмних засобів;
– instrumentation - повнота реєстрації подій та помилок;
– modularity – модульність ПЗ;
– operability - зручність при роботі з ПЗ;
– security - захищеність;
– selfdocumentation – повнота документації;
– simplicity – легкість у використанні;
– traceability - можливість порівняння проекту з вимогами;
– training - зручність при виччені ПЗ.

При вимірюванні за цією метрикою коефіцієнти визначаються відповідно до різних типів програмного забезпечення.

Наступною моделлю якості ПЗ є модель Боема. Яка представляє розширену модель Маккола. Тут визначають 19 атрибутів, включаючи всі атрибути якості за Макколом, та класифікуються за способами використання ПЗ, а саме:

– functionality - функціональність ПЗ;
– clarity - зрозумілість;
– modifiability - зручність при певних змінах ПЗ;
– documentation - документованість;
– resilience - здатність до відновлення;
– understandability - зрозумілість;
– validity - валідованість;
– generality - універсальність;
– economy - економічна ефективність.

Наступна модель - модель якості FURPS, запропонована компанією Hewlett Packard, що являє собою інтерпретацію моделі Маккола та моделі Боема.

Вона містить п’ять наступних атрибутів:

– functionality - функціональність;
– usability - зручність при використанні;

– reliability - надійність;

– performance - продуктивність;

– supportability – можливість супровду.

Розширена сучасна модель FURPS+ включає в себе додаткові атрибути:

– Design rquirements – вимоги розробки;

– Implementation requirements - вимоги щодо реалізації;

– Interface requirements – вимоги реалізації до інтерфейсу;

– Physical requirements - вимоги ізичних компонентів.

Як бачимо, дана модель якості складається з двох компонентів:

– перший компонент визначає характеристики ПЗ

– другий компонент оперує атрибутами.

До недоліків цієї моделі якості відносять:

– немає атрибутів якості;

– не приділяється уваги мобільності ПЗ.

Ще однією сучасною моделлю якості ПЗ є модель Карло Гецци. У цій моделі Гецци враховуються такі характеристики ПЗ:

– цілісність;

– стійкість та надійність;

– продуктивність;

– практичність;

– верифікованість;

– супроводжуваність;

– багаторазове використання;

– мобільність;

– зрозумілість;

– взаємодіяз іншим ПЗ

– ефективність ПЗ;

– ступінь реагування системи;

– видимість самого процесу проектування та розробки.

Ще однією моделлю оцінки якості програмного забезпечення є система SATC, яка була розроблена як програма метрик в Центрі забезпечення якості програмного забезпечення NASA (Software Assurance Technology Center, SATC). Вона розроблена з метою оцінки ризиків проекту, якості ПЗ та ефективності у ньому. Ця програма рекомендує здійснювати оцінку якості вимог, якість самого ПЗ документації та якість тестування. Дана модель визначає певний набір цілей, які пов’язані із ПЗ та атрибути процесів у відповідності до моделі якості програмного забезпечення ISO 9126-1. У цій моделі оцінка якості ПЗ спирається на наступні характеристики:

– goals – глобальна мета, яка визначає, що має бути загалом в програмному забезпеченні;

– attributes - атрибути властивостей програмного забезпечення, які забезпечують мету;

– metrics - кількісні характеристики атрибутів.

– За головну мету в модулі виділяють шість особливостей:

– functionality - функціональність ();

– realibility - надійність ();

– usability - практичність або зручність використання ();

– efficiency - ефективність);

– maintainability - супроводжуваність ();

– portability - мобільність.

4.2 Тестування інформаційної системи

Для ефективного проведення тестування нашої інформаційної системи, розробимо план тестування, який буде містити в собі опис стратегій, що мають використовуватись для проведення такого тестування, а також сформуємо

тестовий набір даних для нього. Всі роботи будуть відбуватися на цьому етапі проектування бази даних та проектування самого додатку. Значну увагу при цьому приділимо формуванню тестового набору даних, оскільки внесення в даних БД та їх зміна і модифікація може спровокувати за собою додаткові фінансові, матеріальні та людські ресурси. Для автоматизації такого процесу використовують широкий набір різного програмного забезпечення для тестування. Це в свою чергу дає можливість не лише скоротити час на тестування, а також дозволить покращити та пришвидшити процес тестування бази даних та інформаційної системи.

На початковому етапі розробки інформаційної системи найбільша увага приділяється тестуванню саме бази даних, яка розробляється поряд з самою інформаційною системою. Саме цей етап дає можливість найбільш перевірити різноманітними тестами функціонал усієї системи, що включає в себе усі звернення до бази даних – читання, запис та модифікацію даних. Такі тести дають можливість виявляти критичні місця, як в самій базі даних, наприклад, невідповідність приведення до третьої нормальної форми (так зване тестування інфологічної моделі), так і виявляти складні запити до БД, які суттєво сповільнюють роботу інформаційної системи та окремих модулів.

В даний момент при проведенні тестування найчастіше використовують еволюційні підходи до розробки програмного забезпечення, яке ще називають адаптивним. Такі методики використовуються, як для тестування БД, коли сама база даних розробляється не на початку розробки проекту інформаційної системи, а нарощується з часом, паралельно до розробки самої системи. Процес тестування бази даних нашої інформаційної системи діяльності автосалону показано на рисунку 4.1.

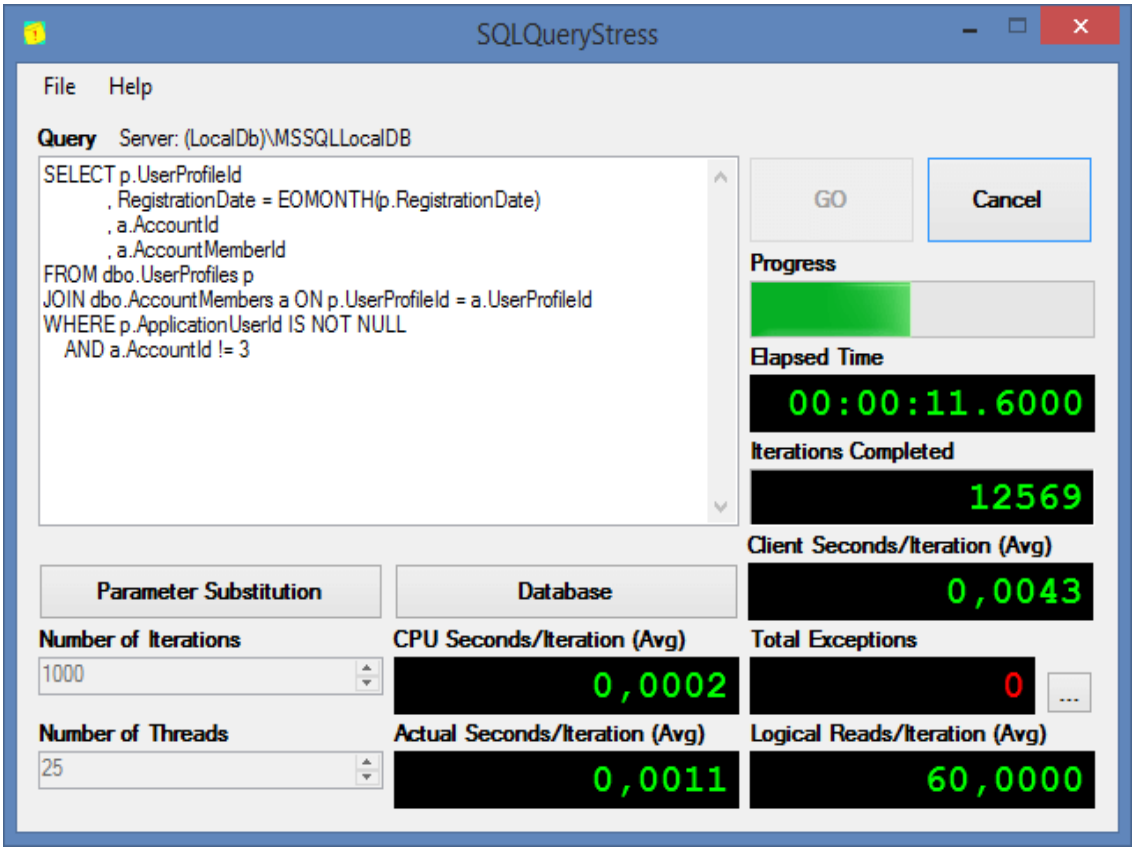


Рисунок 4. 1 – Тестування бази даних інформаційної системи

У даному розділі здійснено вибір та обґрунтування методів тестування системи та наведено процес тестування розробленої інформаційної системи.

ВИСНОВКИ

В ході проведеного аналізу предметної області та сучасного стану досліджуваної проблеми, нами були розглянуто основні підходи до проектування та створення інформаційної системи .

У якості предметної сфери була обрана предметна область купівлі, продажу та пошуку автомобілів, роботу менеджерів та діяльність автосалону загалом. Тому у якості об’єкту дослідження були обрані системи, що спеціалізуються на продажу автомобілів. Проведено їх аналіз та виділено основні можливості та принципи функціонування. В ході роботи було вирішено проблему проектування та конструювання інформаційного забезпечення такої системи. Здійснено проектування інформаційної системи, яка спеціалізується на діяльності роботи менеджерів автосалону.

Зокрема, при проектуванні системи, в першому розділі було проведено досліджено предметну область та здійснено її аналіз. Розкрито поняття рекомендаційних систем. Здійснено постановку завдання та описано математичну модель. Також у ході аналізу предметної області було розроблено технічне завдання до проекту.

В другому розділі здійснено проектування архітектури та структури системи. Запропоновано вибір технології проектування, зокрема у якості архітектурного шаблону сервісу була обрана MVC модель. Здійснено аналіз та вибір СКБД та здійснено проектування логічної структури бази даних.

В третьому розділі здійснено програмну реалізацію та детальне проектування модулів, а також реалізація бази даних автосалону та самої інформаційної системи. В розділі детально описано реалізацію модулів системи з застосуванням сучасних технологій розробки, проектування з застосуванням сучасних засобів програмування. Описано процедуру розгортання та встановлення системи.

У четвертому розділі здійснено вибір та обґрунтування методів тестування системи та проведено тестування бази даних і інформаційної ситеми загалом обраним методом.

В загальному розроблена інформаційна система моніторингу діяльності автосалону має практичну значимість та готова до використання менеджерами з продажу автомобілів для покращення роботи автосалону та його іміджу в сучасних ринкових реаліях.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Бедратюк Л. П. Дипломний проект : методичні вказівки щодо його виконання для студентів спеціальності 121 «Інженерія програмного забезпечення» / Л. П. Бедратюк, Г. І. Радельчук, Ю. В. Форкун, О. М. Яшина. – Хмельницький: ХНУ, 2020. – 77 с.
2. Шаблон проектування Спостерігач(Observer). [Електронний ресурс]. – Режим доступу: http://www.codeproject.com/Articles/12183/Design-Your-Soccer-Engine-and-Learn-How-To-Apply-D (дата звернення: 02.04.2022) - Назва з екрану
3. Шаблон проектування Ітератор. [Електронний ресурс]. – Режим доступу: http://www.znannya.org/?view=behavioral-patterns (дата звернення: 21.14.2022) - Назва з екрану
4. Авраменко А.С. Тестування программного забезпечення. Навчальний посібник / А.С. Авраменко, В.С. Авраменко, Г.В. Косенюк.– Черкаси: ЧНУ імені Богдана Хмельницького, 2017. 284с.
5. Алексенко О.В. Технології програмування та створення програмних продуктів: конспект лекцій / О.В. Алексенко. – Суми: Сумський державний університет, 2013. 133с.
6. Канер К. Тестирование программного обеспечения. Фундаментальные концепции менеджмента бизнес-приложений / К.Канер, Дж.Фолк, Е.Нгуен. –Киев: ДиаСофт, 2001. 544с.
7. Крепич С.Я. Якість програмного забезпечення та тестування: базовий курс. Навчальний посібник / С.Я.Крепич, І.Я. Співак / для бакалаврів галузі знань 12 «Інформаційні технології» спеціальності 121 «Інженерія програмного забезпечення». – Тернопіль: ФОП Паляниця В.А., 2020. – 478с.
8. Developing Database Applications with ADO.NET [Електронний ресурс]. // Сайт docs.embarcadero.com – Режим доступу:

http://docs.embarcadero.com/products/rad_studio/radstudio2007/RS2007_helpupdates/HUpdate4/EN/html/devnet/databasepart_xml.html (дата звернення: 01.06.2022) - Назва з екрану

9. Реляційні бази даних [Електронний ресурс]. // Методичні матеріали з інженерії програмного забезпечення. – Режим доступу: http://www.ua5.org/database/117-reljacjjn-bazi-danikh.html (дата звернення: 16.05.2012) - Назва з екрану

10. Твердохлеб Н. Г. Организация машинной обработки / Н. Г. Твердохлеб, Н. И. Татарчук, М.А Сендзюк – Киев : Головное издательство издательского объединения «Вища школа», 2014. – 512 с.

11. Принципи встановлення вимог розробки системи [Електронний ресурс] –Режим доступу до ресурсу: http://org2.knuba.edu.ua/pluginfile.php/28593/mod_resource/content/1 - Назва з екрану.

12. Савчук Т.О. Організація баз даних і знань. / Савчук Т.О Вінниця: ВДТУ, 2012. – 244 с.

ДОДАТОК А

(обов'язковий)

ТЕХНІЧНЕ ЗАВДАННЯ

2.4 Технічне завдання на розробку програмного забезпечення за темою ДП

Введення

Робота виконується в рамках розробки інформаційної системи моніторингу діяльності автосалону.

1. Найменування та область застосування

Найменування програмна «Інформаційна система моніторингу діяльності автосалону». Данний програмний продукт розробляється для створення системи моніторингу просцеів автосалону.

2. Підстави для розробки

Підставою для розробки є “Завдання на дипломний проект”, затверджене завідуючим ректором ХНУ.

3. Призначення розробки
4.

Функціональне призначення: робота менеджера щодо моніторингу діяльності автосалону.

Експлуатаційне призначення: встановити додаток можна на будь-який комп’ютер і операційну систему

5. Вимоги до програми

При реалізації мають враховані вимоги до функціональних характеристик, умов експлуатації, складу параметрів і технічних засобів, апаратної і програмної сумісності.

5.1 Вимоги до функціональних характеристик

Додатки повинні забезпечувати наступні функції:

- Пошук автомобілів у базі даних;
- Пошук ремонтних робіт;
- Засоби моніторингу;
- Редагування даних про автомобілі;

4.2 Вимоги до надійності

- забезпечувати надійність та безпеку передачі даних по мережі.

- обмеження доступу до даних;

-

4.3 Умови експлуатації

Умови експлуатації повинні відповідати санітарним і технічним нормам експлуатації компютерів. Основні вимоги для інсталяції додатку на ПК відповідають мінімальним вимогам до експлаутації операційної системи, що використовується на пристрої..

4.4 Вимоги до інформаційної та програмної сумісності

Засіб реалізації Python, Ruby, Javascript, C#, C++, Java, PHP. Для розробки баз даних можна використати технологію MySQL, PostgreSQL, MongoDB.4.5 Спеціальні вимоги

Програма повинна мати дружній інтерфейс, розрахований на користувача середньої кваліфікації.

6. Вимоги до програмної документації

У момент здачі проекту замовнику надається наступний набір документів:

- Методика випробувань;
- Програмна система;
- Керівництво користувача;
- Посібник для користувача;
- Опис функціоналу програми;
- Текс програми.

ДОДАТОК Б

(обов’язковий)

ДАТАЛОГІЧНА МОДЕЛЬ

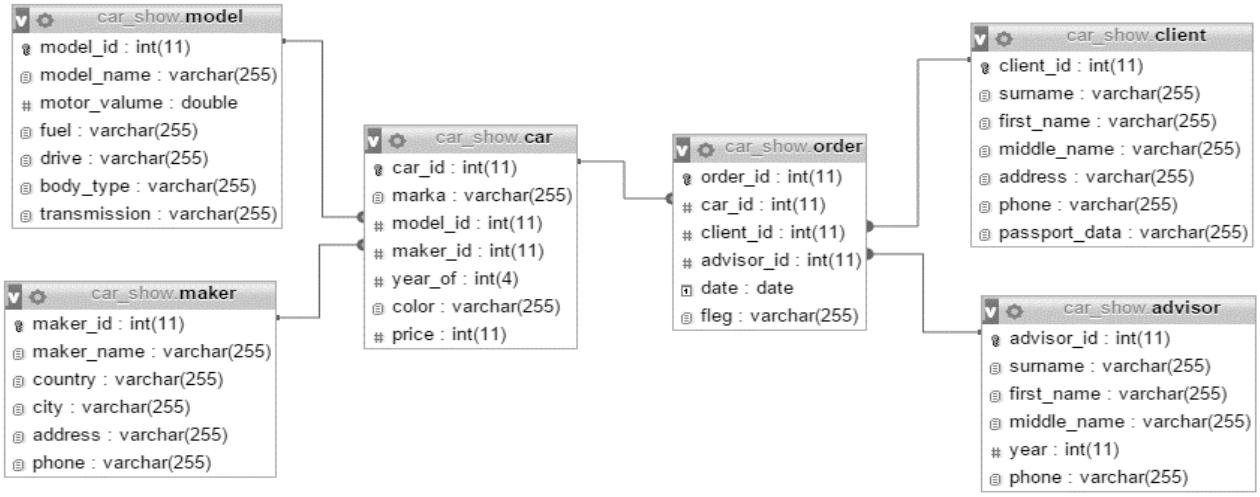


Рисунок Б.1 – Даталогічна модель бази даних

ДОДАТОК В

(обов'язковий)

ФРАГМЕНТИ КОДУ ПРОГРАМНОЇ СИСТЕМИ

Файл main.cs

```
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;
using MySql.Data;
using Word = Microsoft.Office.Interop.Word;

namespace Курсова_БД
{
    public partial class Form1 : Form
    {
        MySqlConnection Connection;
        public Form1()
        {
            InitializeComponent();
        }
        private DataTable comand(string str)
        {
            MySqlCommand command = new MySqlCommand(str);//команда на запит для додавання
            command.Connection = Connection;
            MySqlDataAdapter adapter = new MySqlDataAdapter(command);
```

```
    DataTable dataTable = new DataTable();
    adapter.Fill(dataTable);
    return dataTable;
}

private void button1_Click(object sender, EventArgs e)
{
    if(comboBox1.SelectedIndex == 0)
    {
        Maker mk = new Maker(Connection);
        mk.Show();
    }
    if (comboBox1.SelectedIndex == 1)
    {
        Car one = new Car(Connection);
        one.Show();
    }
    if (comboBox1.SelectedIndex == 2)
    {
        Model mod = new Model(Connection);
        mod.Show();
    }
    if (comboBox1.SelectedIndex == 3)
    {
        Order or = new Order(Connection);
        or.Show();
    }
    if (comboBox1.SelectedIndex == 4)
    {
```

```
            Client cl = new Client(Connection);
            cl.Show();
        }
        if (comboBox1.SelectedIndex == 5)
        {
            Advisor ad = new Advisor(Connection);
            ad.Show();
        }
    }

    private void Form1_Load(object sender, EventArgs e)
    {
        // данные соединения
        string MySQL_host = "127.0.0.1";
        string MySQL_port = "3306";
        string MySQL_uid = "root";
        string MySQL_pw = "";
        string MySQL_database = "car_show";
```

Connection = new MySqlConnection("Database=" + MySQL_database + ";Data Source=" + MySQL_host + ";Port=" + MySQL_port + ";User Id=" + MySQL_uid + ";Password=" + MySQL_pw + ";charset=utf8"); // Создаем соединение. Формат строки соединения подробно описан в прилагающейся документации.

MySqlCommand Query = new MySqlCommand(); // С помощью этого объекта выполняются запросы к БД

Query.Connection = Connection; // Присвоим объекту только что созданное соединение

try

```
        {
            MessageBox.Show("Зєднуюсь з сервером бази даних...");
            Connection.Open();// Соединяемся
        }
        catch (MySqlException SSDB_Exception)
        {
            // Ошибка - выходим
            MessageBox.Show("Перевірти настройки зєднання, не можу зєднатися з базою даних \nОшибка: " + SSDB_Exception.Message);
            this.Close();
        }
        MessageBox.Show("Зєднання встановлено");

        try
        {
            InitializDGW();
        }
        catch { MessageBox.Show("Error...Збій при вивидені списку активних замовлень."); }

    }

    private void button2_Click(object sender, EventArgs e)
    {
        AddOrder addOrd = new AddOrder(Connection);
        addOrd.Show();
    }

    private void InitializDGW()
```

```
    {
        dataGridView1.DataSource = comand("SELECT order_id, `order`.car_id, model_name, client.surname, client.first_name, advisor.surname, advisor.first_name, date, price FROM `order`, car, client, advisor, model WHERE car.car_id = `order`.car_id AND client.client_id = `order`.client_id AND advisor.advisor_id = `order`.advisor_id AND model.model_id = car.model_id AND fleg = \"Виконується\"");
        dataGridView1.DefaultCellStyle.Font = new Font("Arial", 12F, GraphicsUnit.Pixel);
        renameColum();
    }

    private void button3_Click(object sender, EventArgs e)
    {
        try
        {
            comand("UPDATE `order` SET fleg = \"Виконано\" WHERE order_id=" + dataGridView1[0, dataGridView1.CurrentRow.Index].Value.ToString());
            InitializDGW();
        }
        catch { MessageBox.Show("Некоректно введені дані"); }
    }

    private void button4_Click(object sender, EventArgs e)
    {
        try
        {
            comand("UPDATE `order` SET fleg = \"Відмінено\" WHERE order_id=" + dataGridView1[0, dataGridView1.CurrentRow.Index].Value.ToString());
```

```
            InitializDGW();
        }
        catch { MessageBox.Show("Некоректно введені дані"); }
    }

    private void button5_Click(object sender, EventArgs e)
    {
        try
        {
            InitializDGW();
        }
        catch { MessageBox.Show("Error...Збій при вивидені списку активних замовлень."); }
    }
    public void renameColum()
    {
        dataGridView1.Columns[0].AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCells;
        dataGridView1.Columns[0].HeaderText = "Номер замовлення";
        dataGridView1.Columns[1].HeaderText = "Номер авто";
        dataGridView1.Columns[2].HeaderText = "Модель";
        dataGridView1.Columns[3].HeaderText = "Прізвище клієн.";
        dataGridView1.Columns[4].HeaderText = "Ім'я клієн.";
        dataGridView1.Columns[5].HeaderText = "Прізвище конс.";
        dataGridView1.Columns[6].HeaderText = "Ім'я конс.";
        dataGridView1.Columns[7].HeaderText = "Дата";
        dataGridView1.Columns[8].HeaderText = "Ціна";
    }
```

```
private void button6_Click(object sender, EventArgs e)
{
    DataTable bd;
    string date = " day(date)>=" + dateTimePicker1.Value.Day.ToString() + " and month(date)>=" + dateTimePicker1.Value.Month.ToString() + " and year(date)>=" + dateTimePicker1.Value.Year.ToString() + " and day(date)<=" + dateTimePicker2.Value.Day.ToString() + " and month(date)<=" + dateTimePicker2.Value.Month.ToString() + " and year(date)<=" + dateTimePicker2.Value.Year.ToString()+" and `order`.fleg =\"Виконано\"";
    string [] srez = new string[8];

    for (int i = 0; i < srez.Length; i++)
        srez[i] = string.Empty;

    // show count of the orders
    bd = comand("select count(*) from `order` where " + date);
    srez[0] = bd.Rows[0][0].ToString();

    // select 1
    bd = comand("select surname, first_name, middle_name, count(*) kil from advisor, `order` where `order`.advisor_id = advisor.advisor_id and " + date + " group by `order`.advisor_id");
    for (int i = 0; i < bd.Rows.Count; i++)
    {
        srez[1] += bd.Rows[i][0].ToString() + " " + bd.Rows[i][1].ToString() + " " + bd.Rows[i][2].ToString()+" : кількість проданих автомобілів : " + bd.Rows[i][3].ToString()+"\n";
    }
```

```
// select 2
bd = comand("select marka, model_name, color, price from car, `order`, model where `order`.car_id = car.car_id and car.model_id = model.model_id and "+date);
for (int i = 0; i < bd.Rows.Count; i++)
{
   srez[2] += bd.Rows[i][0].ToString()+" "+bd.Rows[i][1].ToString()+", колір : "+bd.Rows[i][2].ToString()+", ціна : "+bd.Rows[i][3].ToString()+"\n";
}

// select
bd = comand("select sum(price) from car, `order` where `order`.car_id = car.car_id and "+date);
srez[3] = bd.Rows[0][0].ToString();

srez[4] = dateTimePicker1.Text;
srez[5] = dateTimePicker2.Text;

bd = comand("select count(*) from `order` where "+ date);
srez[6] = bd.Rows[0][0].ToString();

bd = comand("select count(*) from `order` where fleg=\"Відмінено\" and day(date)>=" + dateTimePicker1.Value.Day.ToString() + " and month(date)>=" + dateTimePicker1.Value.Month.ToString() + " and year(date)>=" + dateTimePicker1.Value.Year.ToString() + " and day(date)<=" + dateTimePicker2.Value.Day.ToString() + " and month(date)<=" + dateTimePicker2.Value.Month.ToString() + " and year(date)<=" + dateTimePicker2.Value.Year.ToString());
srez[7] = bd.Rows[0][0].ToString();
```

```
            ShowOrder ord1 = new ShowOrder(srez);
            ord1.Show();

        }
    }
}
```

Файл AddClient.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

namespace Курсова_БД
{
    public partial class AddClient : Form
    {
        public MySqlConnection  connect;
        public AddClient(MySqlConnection connect)
        {
            InitializeComponent();
```

```
            this.connect = connect;
        }

        private void comand(string str)
        {
            MySqlCommand command = new MySqlCommand(str);//команда на запит для додавання
            command.Connection = connect;
            MySqlDataAdapter adapter = new MySqlDataAdapter(command);
            DataTable dataTable = new DataTable();
            adapter.Fill(dataTable);
        }

        private void button1_Click(object sender, EventArgs e)
        {
            try
            {
                comand("INSERT INTO client (surname, first_name, middle_name, address, phone, passport_data) values ('" + textBox1.Text + "','" + textBox2.Text + "','" + textBox3.Text + "','" + textBox4.Text + "','" + textBox5.Text + "','" + textBox6.Text + "')");
                MessageBox.Show("Дані успішно додано");
                this.Close();
            }
            catch { MessageBox.Show("Некоректно введені дані"); }
        }
    }
}
```

Файл showOrders.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Курсова_БД
{
    public partial class ShowOrder : Form
    {
        public string [] str;
        public string stringToPrint;
        public ShowOrder(string [] str)
        {
            InitializeComponent();
            this.str = str;
        }

        private void ShowOrder_Load(object sender, EventArgs e)
        {
            // Clear all text from the RichTextBox;
            richTextBox1.Clear();
            // Set the foreground color of the text.
            richTextBox1.ForeColor = Color.Black;
```

```
            // Set the alignment of the text that follows.
            richTextBox1.SelectionAlignment = HorizontalAlignment.Center;
            // Set the font for the text.
            richTextBox1.SelectionFont = new Font("Lucinda Console", 20);
            // Set the text within the control.
            richTextBox1.SelectedText = "Звіт\n";

            richTextBox1.SelectionFont = new Font("Lucinda Console", 16);
            richTextBox1.SelectedText = "Автосалон \"Energy\"\n";

            richTextBox1.SelectionFont = new Font("Lucinda Console", 16);
            richTextBox1.SelectedText = "За період з "+ str[4]+" по " + str[5];

            richTextBox1.SelectedText = "\n\n\n";

            richTextBox1.SelectionAlignment = HorizontalAlignment.Left;

            richTextBox1.SelectionFont = new Font("Lucinda Console", 14);
            richTextBox1.SelectedText = "Загальна кількість проданих автомобілів : ";
            richTextBox1.SelectedText = str[0];

            richTextBox1.SelectionFont = new Font("Lucinda Console", 14);
            richTextBox1.SelectedText = "\n\nКількість проданих автомобілів для кожного продавця-консультанта : \n";
            richTextBox1.SelectionFont = new Font("Lucinda Console", 10);
            richTextBox1.SelectedText = str[1];

            richTextBox1.SelectionFont = new Font("Lucinda Console", 14);
            richTextBox1.SelectedText = "\n\nВсі продані автомобілі : \n";
```

```
            richTextBox1.SelectionFont = new Font("Lucinda Console", 10);
            richTextBox1.SelectedText = str[2];

            richTextBox1.SelectionFont = new Font("Lucinda Console", 14);
            richTextBox1.SelectedText = "\n\nДохід автосалону : ";
            richTextBox1.SelectionFont = new Font("Lucinda Console", 10);
            richTextBox1.SelectedText = str[3] + " $";

            richTextBox1.SelectionFont = new Font("Lucinda Console", 14);
            richTextBox1.SelectedText = "\n\nКількість виконаних замовлень  : ";
            richTextBox1.SelectionFont = new Font("Lucinda Console", 10);
            richTextBox1.SelectedText = str[6];

            richTextBox1.SelectionFont = new Font("Lucinda Console", 14);
            richTextBox1.SelectedText = "\nКількість відмінених замовлень  : ";
            richTextBox1.SelectionFont = new Font("Lucinda Console", 10);
            richTextBox1.SelectedText = str[7];

        }

        private void printDocument1_PrintPage(object sender,
System.Drawing.Printing.PrintPageEventArgs e)
        {
            int charactersOnPage = 0;
            int linesPerPage = 0;
            stringToPrint = richTextBox1.Text;

            // Sets the value of charactersOnPage to the number of characters
            // of stringToPrint that will fit within the bounds of the page.
```

```
            e.Graphics.MeasureString(stringToPrint, this.Font,
                e.MarginBounds.Size, StringFormat.GenericTypographic,
                out charactersOnPage, out linesPerPage);

            // Draws the string within the bounds of the page.
            e.Graphics.DrawString(stringToPrint, this.Font, Brushes.Black,
            e.MarginBounds, StringFormat.GenericTypographic);

            // Remove the portion of the string that has been printed.
            stringToPrint = stringToPrint.Substring(charactersOnPage);

            // Check to see if more pages are to be printed.
            e.HasMorePages = (stringToPrint.Length > 0);

            // If there are no more pages, reset the string to be printed.
            if (!e.HasMorePages)
                stringToPrint = richTextBox1.Text;
        }

        private void роздрукуватиToolStripMenuItem_Click(object sender, EventArgs e)
        {
            printPreviewDialog1.Document = printDocument1;
            printPreviewDialog1.ShowDialog();
        }
    }
}
```

Файл Addorder.cs

```
using System;
```

```
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data;
using MySql.Data.MySqlClient;
using System.IO;

namespace Курсова_БД
{
    public partial class AddOrder : Form
    {
        MySqlConnection connect;
        public int carId, clientId, advisorId;
        public AddOrder(MySqlConnection connect)
        {
            this.connect = connect;
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            AddClient addCl = new AddClient(connect);
            addCl.Show();
        }
```

```
public DataTable comand(string str)
{
    MySqlCommand command = new MySqlCommand(str);//команда на запит для додавання
    command.Connection = connect;
    MySqlDataAdapter adapter = new MySqlDataAdapter(command);
    DataTable dataTable = new DataTable();
    adapter.Fill(dataTable);
    return dataTable;
}

public void addItems(ComboBox combobox, DataTable dataTable)
{
    combobox.Items.Clear();
    for (int i = 0; i < dataTable.Rows.Count; i++)
    {
        combobox.Items.Add(dataTable.Rows[i][0].ToString());
    }
}

private void AddOrder_Load(object sender, EventArgs e)
{
    string str = "SELECT car_id, maker_name, marka, model_name, motor_valume, fuel, drive, body_type, transmission, year_of, color, price FROM car, maker, model WHERE maker.maker_id = car.maker_id AND model.model_id = car.model_id";
    dataGridView1.DataSource = comand(str);
}
```

```
private void comboBox2_Click(object sender, EventArgs e)
{
    addItems(comboBox2, comand("SELECT first_name FROM client WHERE surname=\""+comboBox1.Text+"\""));
    comboBox3.Text = string.Empty;
}

private void comboBox3_Click(object sender, EventArgs e)
{
    addItems(comboBox3, comand("SELECT passport_data FROM client WHERE first_name=\"" + comboBox2.Text + "\" AND surname =\"" + comboBox1.Text + "\""));
}

private void comboBox1_Click(object sender, EventArgs e)
{
    addItems(comboBox1, comand("SELECT surname FROM client"));
    comboBox2.Text = string.Empty;
    comboBox3.Text = string.Empty;
}

private void comboBox15_Click(object sender, EventArgs e)
{
    addItems(comboBox15, comand("SELECT first_name FROM advisor WHERE surname=\"" + comboBox14.Text + "\""));
    comboBox16.Text = string.Empty;
    comboBox17.Text = string.Empty;
}
```

```
private void comboBox14_Click(object sender, EventArgs e)
{
    addItems(comboBox14, comand("SELECT surname FROM advisor"));
    comboBox15.Text = string.Empty;
    comboBox16.Text = string.Empty;
    comboBox17.Text = string.Empty;
}

private void comboBox16_Click(object sender, EventArgs e)
{
    addItems(comboBox16, comand("SELECT middle_name FROM advisor WHERE first_name=\"" + comboBox15.Text + "\" AND surname =\"" + comboBox14.Text + "\""));
    comboBox17.Text = string.Empty;
}

private void comboBox17_Click(object sender, EventArgs e)
{
    addItems(comboBox17, comand("SELECT phone FROM advisor WHERE first_name=\"" + comboBox15.Text + "\" AND surname =\"" + comboBox14.Text +"\" AND middle_name =\""+comboBox16.Text+"\""));
}

private void button4_Click(object sender, EventArgs e)
{
    selectAdvisorId();
    selectClientId();
    try
```

```
        {
            if ((this.clientId != 0) && (this.advisorId != 0))
            {
                comand("INSERT INTO `order`(car_id, client_id, advisor_id, date, fleg) value (" + this.carId + "," + this.clientId + "," + this.advisorId + ",\"" + dateTimePicker1.Text + "\",\"Виконується\")");
                MessageBox.Show("Замовлення успішно створене.");
            }
            else { MessageBox.Show("Error...Перевірти дані для оформлення замовлення."); }
        }
        catch { MessageBox.Show("Error...Перевірти дані для оформлення замовлення."); }
    }

    private void button2_Click(object sender, EventArgs e)
    {
        int carId;
        if (int.TryParse(dataGridView1[0, dataGridView1.CurrentRow.Index].Value.ToString(), out carId))
        {
            MessageBox.Show("Автомобіль успішно обрано.\nId_номер автомобіля : " + carId.ToString());
            this.carId = carId;
        }
        else { MessageBox.Show("Error...Неправильно обрано автомобіль зі списку."); }
    }
```

```
private void selectClientId()
{
    DataTable dt;
    int clientId;

    dt = comand("SELECT client_id  FROM client WHERE passport_data=\"" + comboBox3.Text + "\"");
    try
    {
        if (int.TryParse(dt.Rows[0][0].ToString(), out clientId))
        {
            this.clientId = clientId;
        }
        else { MessageBox.Show("Error...Неправильно введені дані про клієнта."); }
    }
    catch { MessageBox.Show("Error...Неправильно введені дані про клієнта.");
}
}

private void selectAdvisorId()
{
    DataTable dt;
    int advisorId;

    dt = comand("SELECT advisor_id FROM advisor WHERE phone = \"" + comboBox17.Text +"\" AND surname = \"" + comboBox14.Text + "\"");
    try
    {
```

```
            if (int.TryParse(dt.Rows[0][0].ToString(), out advisorId))
            {
               this.advisorId = advisorId;
            }
            else { MessageBox.Show("Error...Неправильно введені дані про продавця-консультанта"); }
         }
         catch { MessageBox.Show("Error...Неправильно введені дані про продавця-консультанта"); }
      }

   }
}
```

Файл Car.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;
using MySql.Data;

namespace Курсова_БД
```

```
{
   public partial class Car : Form
   {
      MySqlConnection connect;
      public Car(MySqlConnection connect)
      {
         InitializeComponent();
         this.connect = connect;
      }

      public void show() // Метод для вивидення всієї таблиці на екран
      {
         MySqlCommand command = new MySqlCommand("SELECT * FROM car");//команда на виведення
         command.Connection = connect;
         MySqlDataAdapter adapter = new MySqlDataAdapter();//створення адаптера
         adapter.SelectCommand = command;
         DataTable dataTable = new DataTable();//створенння порпожньої таблиці для подальшого відображення результатів
         adapter.Fill(dataTable);//метод після якого команда відправляється в БД та повертає результат у таблицю dataTable1
         dataGridView1.DataSource = dataTable;
         dataGridView1.DataMember = dataTable.TableName;
      }

      public void changeName()
      {
         dataGridView1.Columns[0].AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCells;
```

```
            dataGridView1.Columns[0].HeaderText = "Номер авто";
            dataGridView1.Columns[1].HeaderText = "Марка";
            dataGridView1.Columns[2].HeaderText = "Номер моделі";
            dataGridView1.Columns[3].HeaderText = "Номер виробника";
            dataGridView1.Columns[4].HeaderText = "Дата випуску";
            dataGridView1.Columns[5].HeaderText = "Колір";
            dataGridView1.Columns[6].HeaderText = "Ціна($)";
        }

        public void comand(string str)
        {
            MySqlCommand command = new MySqlCommand(str);//команда на запит для додавання
            command.Connection = connect;
            MySqlDataAdapter adapter = new MySqlDataAdapter(command);
            DataTable dataTable = new DataTable();
            adapter.Fill(dataTable);
        }

        public void textClear()
        {
            textBox1.Text = string.Empty;
            comboBox2.Text = string.Empty;
            comboBox3.Text = string.Empty;
            textBox4.Text = string.Empty;
            textBox5.Text = string.Empty;
            textBox6.Text = string.Empty;
        }
```

```
public void addItem()
{
    comboBox2.Items.Clear();
    MySqlCommand command = new MySqlCommand("SELECT model_id FROM `model`");//команда на виведення
    command.Connection = connect;
    MySqlDataAdapter adapter = new MySqlDataAdapter();//створення адаптера
    adapter.SelectCommand = command;
    DataTable dataTable = new DataTable();//створеннння порпожньої таблиці для подальшого відображення результатів
    adapter.Fill(dataTable);//метод після якого команда відправляється в БД та повертає результат у таблицю dataTable1

    for (int i = 0; i < dataTable.Rows.Count; i++)
    {
       comboBox2.Items.Add(dataTable.Rows[i][0].ToString());
    }

    comboBox3.Items.Clear();
    MySqlCommand command2 = new MySqlCommand("SELECT maker_id FROM `maker`");//команда на виведення
    command2.Connection = connect;
    MySqlDataAdapter adapter2 = new MySqlDataAdapter();//створення адаптера
    adapter2.SelectCommand = command2;
    DataTable dataTable2 = new DataTable();//створеннння порпожньої таблиці для подальшого відображення результатів
    adapter2.Fill(dataTable2);//метод після якого команда відправляється в БД та повертає результат у таблицю dataTable1
```

```
    for (int i = 0; i < dataTable2.Rows.Count; i++)
    {
        comboBox3.Items.Add(dataTable2.Rows[i][0].ToString());
    }
}

private void Car_Load(object sender, EventArgs e)
{
    show();
    changeName();
    addItem();
}

private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    if (comboBox1.SelectedIndex == 0) // Вставка
    {
        textClear();
        groupBox1.Visible = true;
        groupBox2.Visible = false;
        button1.Text = "Додати";
    }
    if (comboBox1.SelectedIndex == 1) // Редагування
    {
        groupBox1.Visible = true;
        groupBox2.Visible = false;
        button1.Text = "Редагувати";
    }
```

```
        if (comboBox1.SelectedIndex == 2) // Видалення
        {
            groupBox1.Visible = false;
            groupBox2.Visible = true;
        }
    }

    private void button1_Click(object sender, EventArgs e)
    {
        switch (comboBox1.SelectedIndex)
        {
            case 0:
                {
                    try
                    {
                        comand("INSERT INTO car (marka ,model_id, maker_id, year_of, color, price) values ('"+textBox1.Text +"'," + int.Parse(comboBox2.Text) + "'," + int.Parse(comboBox3.Text) + "'," + int.Parse(textBox4.Text) + "'," + textBox5.Text +"'," + int.Parse(textBox6.Text) + "')");
                    }
                    catch { MessageBox.Show("Некоректно введені дані"); }
                    break;
                }
            case 1:
                {
                    try
                    {
                        comand("UPDATE car SET marka='"+textBox1.Text+"', model_id='" + comboBox2.Text + "', maker_id='" + comboBox3.Text + "',year_of='"
```

```
+ textBox4.Text + "', color='" + textBox5.Text + "',price='" + textBox6.Text + "'
WHERE car_id=" + dataGridView1[0,
dataGridView1.CurrentRow.Index].Value.ToString());
                }
                catch { MessageBox.Show("Некоректно введені дані"); }
                break;
            }
        }
        show();
        textClear();
    }

    private void dataGridView1_MouseClick(object sender, MouseEventArgs e)
    {
        if (comboBox1.SelectedIndex == 1)
        {
            textBox1.Text = dataGridView1[1,
dataGridView1.CurrentRow.Index].Value.ToString();
            comboBox2.Text = dataGridView1[2,
dataGridView1.CurrentRow.Index].Value.ToString();
            comboBox3.Text = dataGridView1[3,
dataGridView1.CurrentRow.Index].Value.ToString();
            textBox4.Text = dataGridView1[4,
dataGridView1.CurrentRow.Index].Value.ToString();
            textBox5.Text = dataGridView1[5,
dataGridView1.CurrentRow.Index].Value.ToString();
            textBox6.Text = dataGridView1[6,
dataGridView1.CurrentRow.Index].Value.ToString();
        }
```

```
        }

        private void button2_Click(object sender, EventArgs e)
        {
            try
            {
                comand("DELETE FROM car WHERE car_id = " + dataGridView1[0,
dataGridView1.CurrentRow.Index].Value.ToString());
                show();
            }
            catch { MessageBox.Show("Некоректно введені дані"); }
        }


    }
}
```

Файл advisor.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
```

```
using MySql.Data;
using MySql.Data.MySqlClient;

namespace diplom
{
   public partial class Advisor : Form
   {
      public MySqlConnection connect;
      public Advisor(MySqlConnection connect)
      {
         InitializeComponent();
         this.connect = connect;
      }

      public void show() // Метод для вивидення всієї таблиці на екран
      {
         MySqlCommand command = new MySqlCommand("SELECT * FROM advisor");//команда на виведення
         command.Connection = connect;
         MySqlDataAdapter adapter = new MySqlDataAdapter();//створення адаптера
         adapter.SelectCommand = command;
         DataTable dataTable = new DataTable();//створенння порпожньої таблиці для подальшого відображення результатів
         adapter.Fill(dataTable);//метод після якого команда відправляється в БД та повертає результат у таблицю dataTable1
         dataGridView1.DataSource = dataTable;
         dataGridView1.DataMember = dataTable.TableName;
      }
```

```
        public void changeName()
        {
            dataGridView1.Columns[0].AutoSizeMode =
DataGridViewAutoSizeColumnMode.AllCells;
            dataGridView1.Columns[0].HeaderText = "Номер консультанта";
            dataGridView1.Columns[1].HeaderText = "Прізвище";
            dataGridView1.Columns[2].HeaderText = "Ім'я";
            dataGridView1.Columns[3].HeaderText = "По батькові";
            dataGridView1.Columns[4].HeaderText = "Вік";
            dataGridView1.Columns[5].HeaderText = "Телефон";
        }

        public void comand(string str)
        {
            MySqlCommand command = new MySqlCommand(str);//команда на запит
для додавання
            command.Connection = connect;
            MySqlDataAdapter adapter = new MySqlDataAdapter(command);
            DataTable dataTable = new DataTable();
            adapter.Fill(dataTable);
        }

        public void textClear()
        {
            textBox1.Text = string.Empty;
            textBox2.Text = string.Empty;
            textBox3.Text = string.Empty;
            textBox4.Text = string.Empty;
            textBox5.Text = string.Empty;
```

```
}

private void Advisor_Load(object sender, EventArgs e)
{
   show();
   changeName();
}

private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
{
   if (comboBox1.SelectedIndex == 0) // Вставка
   {
      textClear();
      groupBox1.Visible = true;
      groupBox2.Visible = false;
      button1.Text = "Додати";
   }
   if (comboBox1.SelectedIndex == 1) // Редагування
   {
      groupBox1.Visible = true;
      groupBox2.Visible = false;
      button1.Text = "Редагувати";
   }
   if (comboBox1.SelectedIndex == 2) // Видалення
   {
      groupBox1.Visible = false;
      groupBox2.Visible = true;
   }
}
```

```
private void dataGridView1_MouseClick(object sender, MouseEventArgs e)
{
    textBox1.Text = dataGridView1[1, dataGridView1.CurrentRow.Index].Value.ToString();
    textBox2.Text = dataGridView1[2, dataGridView1.CurrentRow.Index].Value.ToString();
    textBox3.Text = dataGridView1[3, dataGridView1.CurrentRow.Index].Value.ToString();
    textBox4.Text = dataGridView1[4, dataGridView1.CurrentRow.Index].Value.ToString();
    textBox5.Text = dataGridView1[5, dataGridView1.CurrentRow.Index].Value.ToString();
}

private void button1_Click(object sender, EventArgs e)
{
    switch (comboBox1.SelectedIndex)
    {
        case 0:
            {
                try
                {
                    comand("INSERT INTO advisor (surname, first_name, middle_name, year, phone) values ('" + textBox1.Text + "','" + textBox2.Text + "','" + textBox3.Text + "','" + textBox4.Text + "','" + textBox5.Text + "')");
                }
                catch { MessageBox.Show("Некоректно введені дані"); }
                break;
```

```
                }
            case 1:
                {
                    try
                    {
                        comand("UPDATE advisor SET surname='" + textBox1.Text + "', first_name='" + textBox2.Text + "',middle_name='" + textBox3.Text + "',year='" + textBox4.Text + "', phone='" + textBox5.Text + "' WHERE advisor_id=" + dataGridView1[0, dataGridView1.CurrentRow.Index].Value.ToString());
                    }
                    catch { MessageBox.Show("Некоректно введені дані"); }
                    break;
                }
            }
            show();
            textClear();
        }
        private void button2_Click(object sender, EventArgs e)
        {
            try
            {
                comand("DELETE FROM advisor WHERE advisor_id = " + dataGridView1[0, dataGridView1.CurrentRow.Index].Value.ToString());
                show();
            }
            catch { MessageBox.Show("Некоректно введені дані"); }
        }
    }
}
```

ДОДАТОК Д

(обов'язковий)

ПРЕЗЕНТАЦІЯ

1

ДИПЛОМНИЙ ПРОЕКТ

«Інформаційна система моніторингу діяльності автосалону»

Виконав: студент IV курсу, групи ІПЗс-19-1 Савкін Р.Р.

Керівник: канд. тех. наук, доцент Форкун Ю.В.

2

Вступ

Розвиток технологій дає можливість різноманітним підприємствам та установам використовувати автоматизовані інформаційні системи.

Інформаційна система – сукупність організаційних і технічних засобів для збереження та обробки інформації з метою забезпечення інформаційних потреб користувачів системи.

В будь-якій інформаційній системі вирішуються задачі трьох типів:

- задачі оцінки ситуації (деколи їх називають задачами розпізнавання образів);
- задачі перетворення опису ситуації (розрахункові задачі, задачі моделювання тощо);
- задачі прийняття рішень (в тому числі і оптимізаційні).

3

Мета дослідження

Метою і призначенням цього проекту є розробка архітектури і компонентів модуля моніторингу діяльності автосалону для визначення особливостей маркетингової діяльності автосалону та розробка рекомендацій щодо підвищення ефективності маркетингової діяльності підприємства з урахуванням впливу внутрішнього та зовнішнього середовища. Об'єктом дослідження роботи виступають веб-сервіси, що спеціалізуються на моніторингу продажу та обслуговування мобільних телефонів менеджерами магазинів.

Об'єкт та предмет дослідження

Об'єкт дослідження – процеси маркетингової діяльності автосалону.

Предмет дослідження – теоретичні, методичні та практичні завдання використання інформаційних технологій при маркетингової діяльності автосалону.

Постановка задачі 4

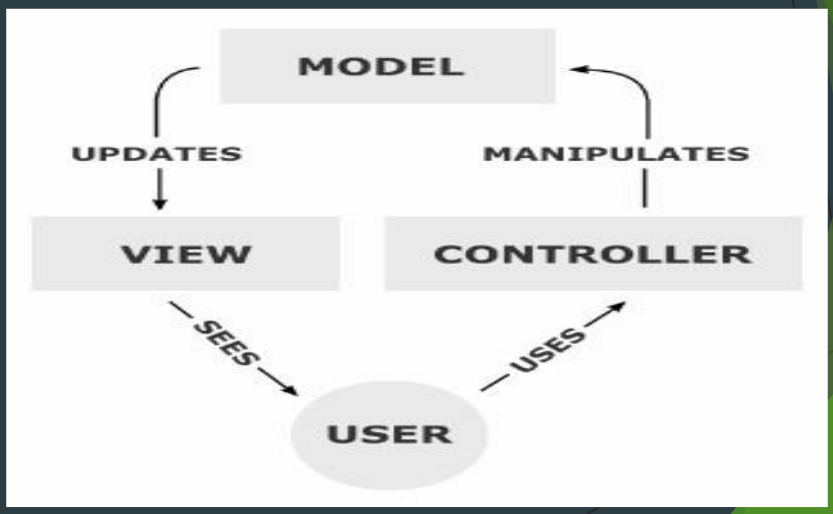
Реалізація поставленої мети обумовила такі завдання:
- розглянути сутність та зміст маркетингової діяльності підприємства;
- розглянути методичні аспекти аналізу маркетингової діяльності;
- визначити специфіку маркетингової діяльності підприємств автодилерів;
- дослідити складові маркетингової діяльності автосалону;
- проаналізувати маркетингове середовище автосалону «CITROËN Запоріжжя»;
- визначити цінові сегментів ринку автомобілівв Україні;
- окреслити альтернативи маркетингової діяльності автосалону;
- запропонувати засоби управління поведінкою споживачів підприємства на ринку легкових автомобілів.

Предметною областю цього проекту є діяльність автосалону. Вона має такі функції:
- перегляд обліку автомобілів та інформації про них;
- перегляд покупців та інформації про них;
- перегляд співробітників та інформації про них;
- перегляд продажів;
- перегляд поставок;
- моніторинг діяльності підприємства;
- моніторинг роботи менеджерів автосалону.

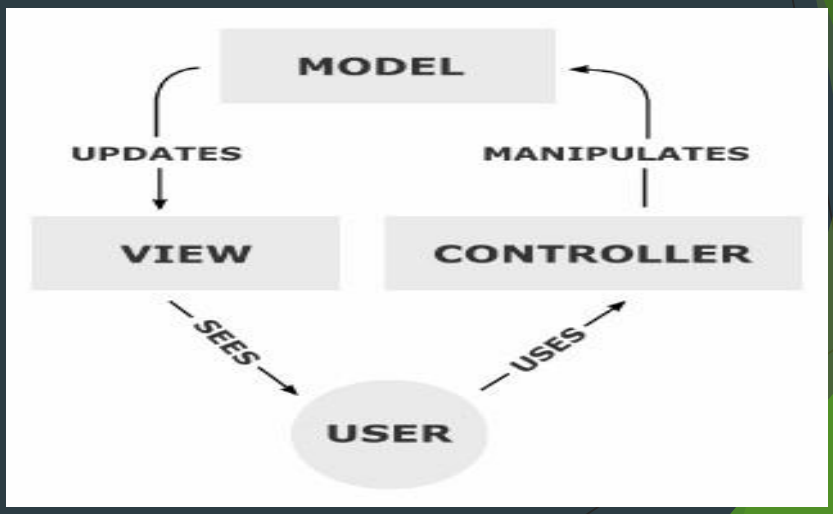
Показники оцінки маркетингової діяльності автосалону споживачами 8

Характеристика уподобань	Кількісна характеристика
Ціни	4
Сервіс	4
Якість обслуговування	4
Кваліфікація персоналу	4
Індивідуальність підходу	2
Асортимент автомобілів	3
Спецпропозиції	2
Додаткові послуги	2
Внутрішній комфорт автосалону	4
Репутація	3
Кредитні пропозиції	3

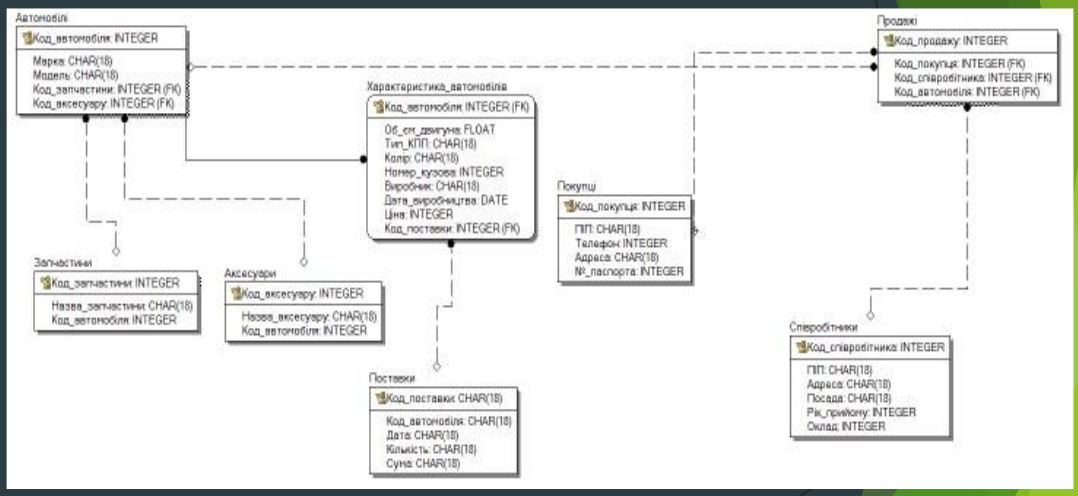
Концепція Model – View – Controler



Концепція Model – View – Controler



Фізична ER-діаграма



12 Вибір технологій і методів реалізації системи

Для реалізації нашого проекту була обрана СКБД MY SQL. Вона є однією з найбільш популярних систем керування базами даних в світі. Дана СКБД підходить для різних проектів: від невеликих застосунків до високонавантаженихпроектів.

Для роботи з базою даних використовуєтьсяфреймворк Entity Framework

Вибір технологій і методів реалізації системи

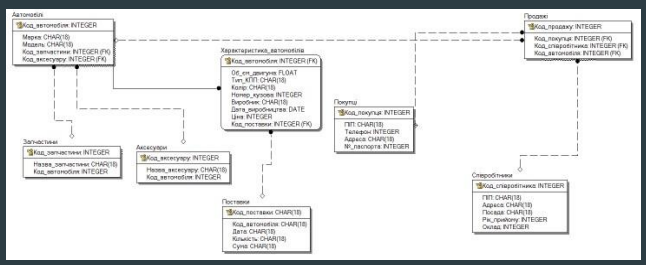
14

Для розробки даного проекту я обрав мову програмування C#. C# має багато можливостей для зручної роботи з різними СКБД.

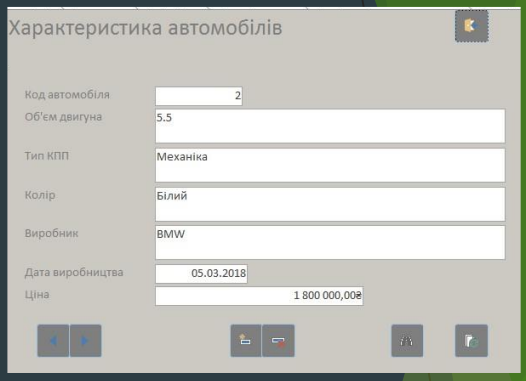
C# використовується, як для розробки програм для настільних ПК, так і для розробки веб-сервісів, сайтів та інших застосунків. Дана мова є універсальною. На ній можна розробляти, як і прості проекти, так і досить потужні додатки.

Реалізація модулів системи

18



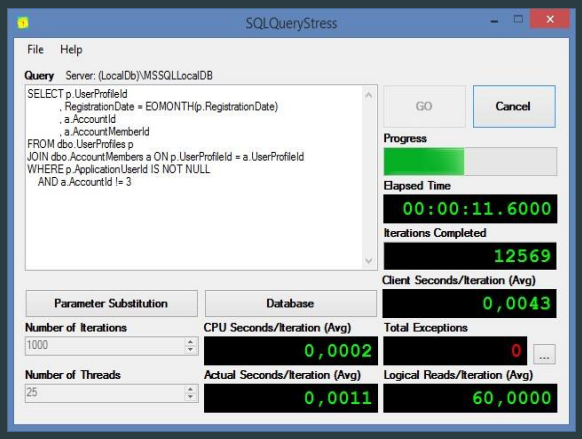
Фізична ER-діаграма



Код Автомобіля	Марка	Модель
1	BMW	M5
2	BMW	X5
3	BMW	X5M
4	BMW	M3
5	BMW	X6
6	BMW	X1
7	BMW	M2

Сорінка роботи з марками автомобіля

Тестування БД ІС



Висновки

В ході проведеного аналізу предметної області та сучасного стану досліджуваної проблеми, нами були розглянуто основні підходи до проектування та створення інформаційних систем.

У якості предметної сфери була обрана предметна область купівлі, продажу та пошуку автомобілів та діяльність автосалону. Тому у якості об’єкту дослідження були обрані системи, що спеціалізуються на продажу автомобілів. Проведено їх аналіз та виділено основні можливості та принципи функціонування. В ході роботи було вирішено проблему проектування та конструювання інформаційного забезпечення такої систем та здійснено проектування інформаційної системи, яка спеціалізується на діяяльності роботи автосалону.

Зокрема, при проектуванні системи, в першому розділі було проведено досліджено предметну область та здійснено її аналіз. Розкрито поняття рекомендаційних систем. Здійснено постановку завдання та описано математичну модель. Також у ході аналізу предметної області було розроблено технічне завдання.

В другому розділі здійснено проектування архітектури та структури системи. Запропоновано вибір технології проектування, зокрема у якості архітектурного шаблону сервісу була обрана MVC модель. Здійснено аналіз та вибір СКБД та здійснено проектування логічної структури бази даних.

В третьому розділі здійснено програмну реалізацію та детальне проектування модулів та реалізація бази даних. Детально описано реалізацію модулів системи з застосуванням сучасних технологій проектування та програмування. Описано процедуру розгортання та встановлення системи.

Завідувачу кафедри інженерії програмного забезпечення проф. Бедратюку Л. П.

здобувача вищої освіти
Савкіна Р.Р.
Прізвище, ініціали

факультет ІТ, 4 курс, група ІПЗс-19-1

ЗАЯВА

З правилами чинного Положення «Про дотримання академічної доброчесності в Хмельницькому національному університеті» від 26.09.2020 (зі змінами від 26.11.2020), згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування заходів дисциплінарної та академічної відповідальності, ознайомлений. Про використання програмно-технічних засобів для перевірки кваліфікаційних робіт здобувачів вищої освіти на плагіат оповіщений та надаю свою згоду на обробку та збереження університетом моєї роботи в інституційному репозитарії університету.

Також надаю університету право на передачу моєї роботи для обробки та збереження в базах даних програмно-технічних засобів (Unicheck та Anti-Plagiarism) та використання роботи для виявлення плагіату в інших роботах, які перевіряються програмно-технічними засобами та користувачами, що мають доступ до цих програмно-технічних засобів, виключно в обмежених цілях для виявлення плагіату в текстах робіт.

Робота для перевірки університетом надається в друкованому та електронному варіанті. Електронна версія моєї роботи збігається (ідентична) з друкованою.

31.05.2022
дата

підпис

Thu Jun 16 11:10:33 EEST 2022 Хавріч Володимир Ростиславович Хмельницький національний університет XHY

Anti-Plagiarism v-15.257

Максимальне співпадіння з одним документом 32.0%

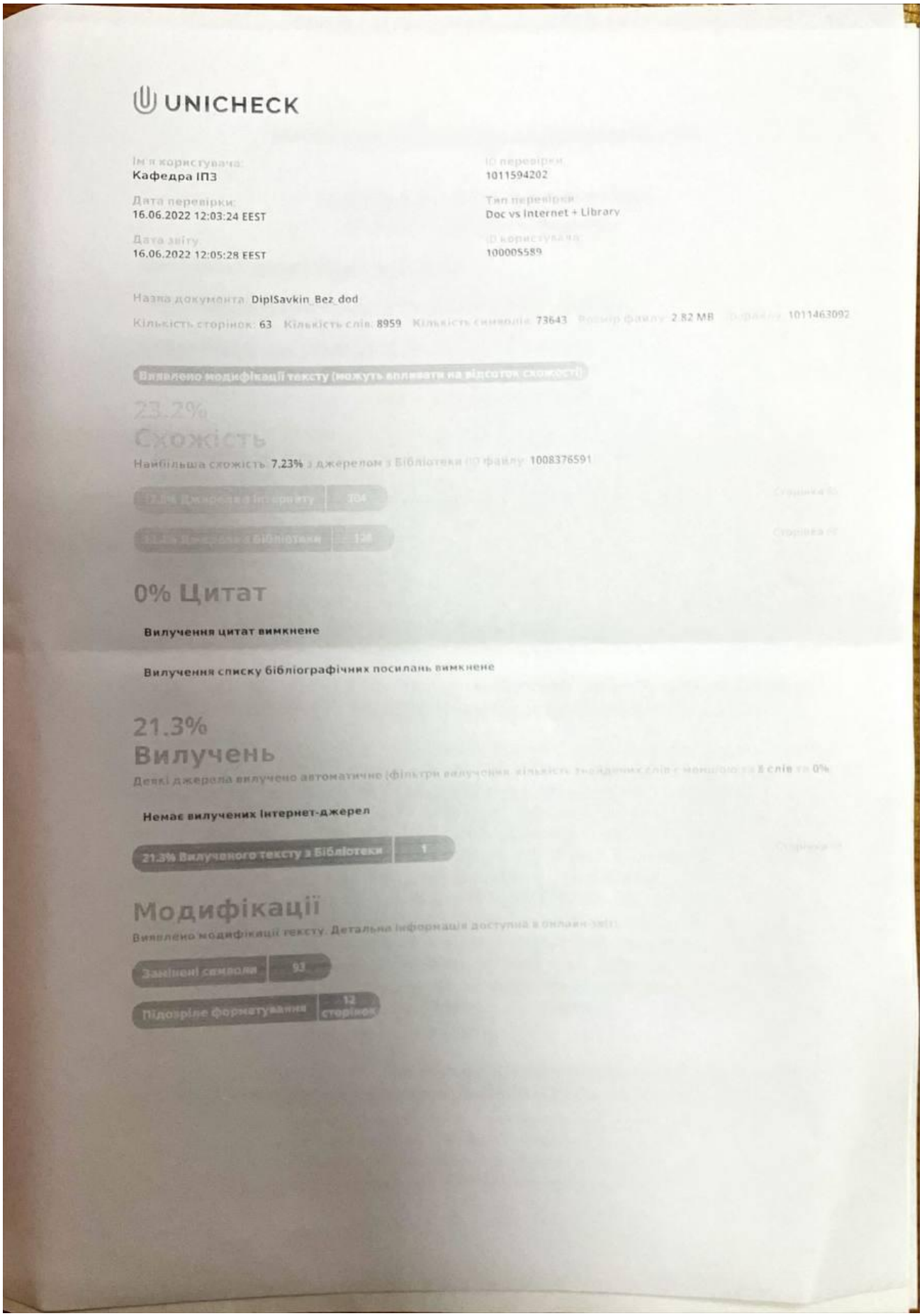
Словники перевірки: en_US ru_RU ua_UA **Помилок в документах: 9%**

ID: 105672
Назва: Інформаційна система моніторингу діяльності автосалону
Додано в БД: 2022-06-16
Автора: Р Р Савкін
Керівники: Ю В Форкун
Консультанти:
Опоненти:

Документ		Сумарний збіг по Базі Даних	
Символи	Лексеми	Символи	Лексеми
55462	540	20029 (36%)	226 (42%)

Джерело плагіату

ID	Опис	Наявність плагіату в документі: Символи	Лексеми
102093	Назва: Курсовий проект з дисципліни "Бази даних" Додано в БД: 2022-02-09 Автора: Висоброшна А С Керівники: Форкун Ю В Консультанти: Опоненти:	5511 (10.0%)	72 (13.0%)
104482	Назва: Інформаційна система моніторингу діяльності автосалону Додано в БД: 2022-06-05 Автора: Р Р Савкін Керівники: Ю В Форкун Консультанти: Опоненти:	17671 (32.0%)	200 (37.0%)
103278	Назва: Звіт з переддипломної практики Додано в БД: 2022-05-04 Автора: Р. Р. Савкін Керівники: Форкун Ю В Консультанти: Опоненти:	5999 (11.0%)	52 (10.0%)



UNICHECK

Ім'я користувача:
Кафедра ІПЗ

Дата перевірки:
16.06.2022 12:03:24 EEST

Дата звіту:
16.06.2022 12:05:28 EEST

ID перевірки:
1011594202

Тип перевірки:
Doc vs Internet + Library

ID користувача:
100005589

Назва документа: DiplSavkin_Bez_dod

Кількість сторінок: 63 Кількість слів: 8959 Кількість символів: 73643 Розмір файлу: 2.82 MB ID файлу: 1011463092

Виявлено модифікації тексту (можуть впливати на відсоток схожості)

23.2%
Схожість

Найбільша схожість: 7.23% з джерелом з Бібліотека (ID файлу: 1008376591

7.23% Джерела з Інтернету 304

3.14% Джерела з Бібліотеки 128

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

21.3%
Вилучень

Деякі джерела вилучено автоматично (фільтри вилучення: кількість знайдених слів є меншою за 8 слів та 0%

Немає вилучених Інтернет-джерел

21.3% Вилученого тексту з Бібліотеки 1

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Заміжені символи 93

Підозріле форматування 12 сторінок

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА ДИПЛОМНИЙ ПРОЕКТ
освітнього ступеня «Бакалавр»

Дипломник Савкін Роман Русланович

Тема Інформаційна система моніторингу діяльності автосалону

Спеціальність 121 – Інженерія програмного забезпечення

Обсяг дипломного проекту:

Кількість листів креслень 18 ; кількість сторінок записки 110

1.Короткий зміст пояснювальної записки та прийнятих рішень У дипломному проекті проведено аналіз предметної області, змісту і методів програмного забезпечення, визначено функціональні вимоги до програмного продукту, розроблено архітектуру програмного продукту, спроектовано структуру та розроблено базу даних, та здійснено конструювання програмного продукту на клієнт-серверній технології. В якості реалізації програмного продукту було використано мову програмування C# та СКБД MySQL та середовище розробки додатків Microsoft Visual Studio 2019.. Було здійснено тестування програми, за результатами якого доведено, що розроблена інформаційна система працює коректно та готова до використання.

2. Висновок про відповідність проекту поставленому завданню Дипломний проект виконаний відповідно до поставленого завдання та з дотриманням всіх вимог.

3. Характеристика виконання кожного розділу проекту, ступінь використання останніх досягнень науки і техніки та передових методів роботи У вступі наведено актуальність теми, визначено мету та завдання дипломного проектування. У першому розділі здійснено аналіз предметної області. у якості об'єкту дослідження були обрані системи, що спеціалізуються на продажу автомобілів. Проведено їх аналіз та виділено основні можливості та принципи функціонування. В ході роботи було вирішено проблему проектування та конструювання інформаційного забезпечення такої системи. В другому розділі здійснено проектування архітектури та структури системи. Запропоновано вибір технології проектування.У четвертому розділі здійснено вибір та обґрунтування методів тестування системи та проведено тестування бази даних і інформаційної ситеми загалом обраним методом Далі було виконано тестування інформаційної системи та проведено його у відповідності до функціональних вимог, в результаті було підтверджено коректну роботу веб-сайту.

4. Позитивні сторони проекту Тематика дипломного проекту є актуальною, оскільки на робота сисмами моніторингу діяльності роботи персоналу є досить затребуваною

5. Негативні сторони проекту У проекті досить мало уваги приділено тестуванню роботи системи

6. Оцінка графічного оформлення та пояснювальної записки проекту Графічне оформлення виконано відповідно до теми дипломного проекту та подано у вигляді діаграм і рисунків. Пояснювальна записка оформлена згідно вимог чинних стандартів.

7. Відгук про дипломний проект в цілому Дипломний проект заслуговує позитивної оцінки. Матеріал пояснювальної записки структурований, послідовний, чіткий та простий, що дозволяє чітко зрозуміти викладений матеріал у рамках тематики дипломного проекту. Графічний матеріал дає можливість наочно побачити деталі проектування системи.

8. Інші зауваження

9. Оцінка дипломного проекту Дипломний проект виконаний на достатньому рівні, відповідає поставленій задачі та заслуговує на оцінку «задовільно».

РЕЦЕНЗЕНТ Д. Ш. Н., професор Говорущенко Тетяна Олександрівна

“ 16 ” 06 2021 р. (підпис)

РІШЕННЯ ЕКСПЕРНОЇ КОМІСІЇ
КАФЕДРИ ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ
ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: «Інформаційна система моніторингу діяльності автосалону»
Автор: Савкін Роман Русланович
Спеціальність: 121 – Інженерія програмного забезпечення
Освітня програма: Освітньо-професійна програма «Інженерія програмного забезпечення»
Науковий керівник: Форкун Юрій Вікторович, кандидат технічних наук, доцент
Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом. Робота приймається до захисту.	**відповідає**
2	Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи. Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи	
3	Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	
5	Інше:	

Підтвердження:

Запозичення, виявлені в роботі, є законними і не є плагіатом, оскільки:

1) у тексті дипломної роботи системами перевірки на плагіат виявлено схожість з деякими документами в частині загальновживаних обов'язкових словосполучень у стандартних бланках (титулка, бланк завдання), у структурі змісту, назвах розділів/підрозділів тощо та в назвах публікацій у переліку джерел посилання;

2) в якості запозичень системою було зафіксовано деякі послідовності вихідного коду і посиланя на бібліотеки, які є стандартними мовними конструкціями програмування та не можуть розглядатися як об'єкт авторських прав і, відповідно, їх порушення;

3) усі запозичення є фрагментарними або мають належним чином оформленні посилання;

4) виявлені модифікації тексту не впливають на відсоток схожості.

Сумарний обсяг всіх запозичень, визначений системою виявлення збігів ідентичності/ схожості, складає 23,2% і адресується до 1 джерела, а саме звіту з передипломниої практики самого студента, що, з урахуванням характеру звіту з переддипломної практики, який включає в себе частину дипломного проекту відповідає характеру теми і свідчить на користь дипломної роботи.

Керівник _____ Ю.В. Форкун

Гарант ОП _____ Л. П. Бедратюк

Завідувач кафедри _____ Л. П. Бедратюк