

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
«КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА»
навчально-науковий інститут
«Інститут інформаційних технологій в економіці»**

Кафедра інформаційних систем в економіці

галузь знань 12 «Інформаційні технології»
спеціальність 122 «Комп'ютерні науки»

Форма навчання: денна

КВАЛІФІКАЦІЙНИЙ БАКАЛАВРСЬКИЙ ПРОЄКТ

на тему

**Проектування інформаційної системи підтримки роботи автомобільних
стоянок**

студента Васильєва Івана Юрійовича _____

Науковий керівник: к.е.н., проф.

_____ Ситник Н.В.

**кваліфікаційний бакалаврський проєкт
допущений до захисту в Екзаменаційній
комісії з атестації здобувачів вищої
освіти**

В.о. завідувача кафедри: к.е.н., доцент.

_____ Тішков Б.О.

Київ 2022

Міністерство освіти і науки України
Державний вищий навчальний заклад
«Київський національний економічний університет
імені Вадима Гетьмана»
Навчально-науковий інститут
«Інститут інформаційних технологій в економіці»
Кафедра інформаційних систем в економіці
денна форма навчання
Затверджую:
в.о. завідувача кафедри _____ Тішков Б.О.
“ ____ ” _____ 2022 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на каліфікаційний бакалаврський проєкт
на тему: Проєктування інформаційної системи підтримки роботи
автомобільних стоянок

Васильєва Івана Юрійовича
Тему кваліфікаційного бакалаврського проєкту затверджено наказом ректора
від «21» грудня 2021 р. № 1916-ст.

Кваліфікаційний бакалаврський проєкт виконується на матеріалах
Мережі автомобільних стоянок

План кваліфікаційного бакалаврського проєкту

Розділ I Характеристика та аналіз предметної області.

Розділ II Розробка вимог і моделювання інформаційної системи.

Розділ III Проєктування та реалізація компонентів системи.

Об’єкт дослідження інформаційні процеси, що характеризують підтримку
роботи автомобільних стоянок

Предмет дослідження методи, підходи та інформаційні технології розробки
інформаційних систем для підтримки роботи автомобільних стоянок

Мета кваліфікаційного бакалаврського проєкту розробка інформаційну
систему для підтримки роботи автомобільних стоянок

Конкретні завдання, які студент повинен виконати для досягнення поставленої мети:

У розділі I Дослідити та характеризувати інформаційні процеси обліку та аналізу роботи автомобільних стоянок.

Провести аналіз існуючих підходів та систем до створення інформаційних систем для підтримки роботи автомобільних стоянок

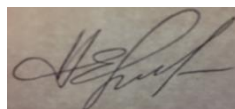
У розділі II Провести вивчення та аналіз бізнес-вимог до проєктуємої системи з розробки інформаційних систем для підтримки роботи автомобільних стоянок

Розробити постановку задачі, навівши її характеристику, перелік вхідних і вихідних повідомлень та алгоритм вирішення. Провести моделювання системи з використанням інструментарію побудови UML-діаграм.

У розділі III Запроектувати базу даних інформаційної системи в середовищі реляційної СКБД, визначити основні вимоги до технічного забезпечення та представити його конфігурацію. Розробити програмне забезпечення для реалізації задачі з підтримки роботи автомобільних стоянок та проілюструвати його роботу на контрольному прикладі.

Завдання підготував

науковий керівник



(підпис)

Ситник Ніна Василівна

(прізвище, ім'я, по-батькові)

“22” грудня 2021 р.

**Завдання держав
студент**

(підпис)

Васильєв Іван Юрійович

(прізвище, ім'я, по-батькові)

“ _____ ” _____ 2022 р.

АНОТАЦІЯ

Кваліфікаційного бакалаврського проєкту студента 4 курсу
Навчально-наукового інституту «Інститут інформаційних технологій в
економіці»

Васильєва Івана Юрійовича, виконаної на тему:
«Проектування інформаційної системи підтримки роботи автомобільних
стоянок»

Київ: кафедра інформаційних систем в економіці, 2022 р.

Кваліфікаційний бакалаврський проєкт присвячений актуальній проблемі автоматизації обліку даних на автостоянках, яку пропонується розв'язувати та удосконалювати з використанням сучасних інформаційних технологій. Проєкт складається з трьох розділів, логічно пов'язаних між собою.

В першому розділі дана характеристика предметної галузі й об'єкта дослідження, наведено аналіз задач, що розв'язуються, а також наводиться перелік існуючих програмних засобів для даної предметної галузі.

Другий розділ є проектним і присвячений обґрунтуванню методу проектування системи, розробленню її архітектури, виконанню постановки та розробленню алгоритма розв'язання задач.

Третій розділ – конструктивний. Тут наведено інформаційне, технічне програмне та організаційне забезпечення для підсистеми підтримки прийняття рішень з фінансового моніторингу операцій комерційного банку. Висвітлені питання організації інформаційного забезпечення: розроблена структура інформаційного забезпечення, описані та наведені в додатках форми вхідних та вихідних документів. Обґрунтований комплекс технічних засобів, а також програмне забезпечення, що використовується для створення системи підтримки роботи автомобільних стоянок. Вказані структури інформаційних масивів, які використовуються під час вирішення задач.

Висновки містять рекомендації щодо доцільності розроблення та впровадження системи підтримки роботи автомобільних стоянок.

РЕФЕРАТ

Кваліфікаційний бакалаврський проєкт містить 76 сторінок, 14 таблиць, 40 рисунка, список літератури з 20 найменувань, 4 додатка.

ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ПІДТРИМКИ РОБОТИ АВТОМОБІЛЬНИХ СТОЯНОК

Перелік ключових слів: ІС, інформаційна система, проєктування, ООП, автостоянка, контракти, оренда, microsoft visual studio, windows forms, .NET framework, C#, БД, база даних, СУБД, таблиці, дані, облік.

Предметом дослідження є автоматизоване рішення завдань з обліку роботи даних на автостоянці.

Об'єктом дослідження виступає організація роботи автостоянок.

Мета бакалаврського проєкту полягає в використанні набутих навичок для проєктування та реалізації інформаційної системи.

Завданнями бакалаврського проєкту є проєктування та реалізації інформаційної системи для підтримки роботи автостоянок.

Апаратні та програмні засоби, що використовувались при проєктуванні: Microsoft Visual Studio, Microsoft SQL Server, SQL Server Management Studio, draw.io, Enterprise Architect.

Результати досягнуті в процесі роботи – Спроєктовано та реалізовано ІС для підтримки роботи автостоянок на основі аналізу існуючих систем.

Одержані результати можуть бути використані – Данна ІС може використовуватися автостоянками, які не мають ніякої автоматизації процесів обліку даних у своїй роботі. ІС має перспективи до розширення та поповнення новими функціями та ідеями.

Рік виконання бакалаврського проєкту 2022

Рік захисту бакалаврського проєкту 2022

ЗМІСТ

Перелік умовних позначень, символів, одиниць величин і термінів	8
Вступ	9
Розділ 1. Характеристика та аналіз предметної галузі	10
1.1. Характеристика предметної галузі та об'єкта дослідження	10
1.2. Аналіз літературних джерел та практичного досвіду використання ІС і технологій в предметній галузі	12
Розділ 2. Розробка вимог і моделювання інформаційної системи/підсистеми.	17
2.1. Аналіз і специфікація вимог до інформаційної системи/підсистеми	17
2.2. Постановка задачі	21
2.3. Моделювання інформаційної системи/підсистеми	24
Розділ 3. Проектування та реалізація компонентів системи/підсистеми	33
3.1. Інформаційне забезпечення	33
3.2. Технічне забезпечення	40
3.3. Програмне забезпечення	42
3.4. Результати реалізації інформаційної системи/підсистеми	43
Висновок	45
Перелік використаних джерел	46
Додатки	47

Перелік умовних позначень, символів, одиниць величин і термінів

Інформаційна система – ІС,

База даних – БД ,

Транспортний засіб – ТЗ,

Система контролю бази даних – СКБД,

structured query language «Мова Структурованих Запитів» – SQL,

рисунок – рис,

Microsoft – MS,

гігагерц – ГГц,

гігабайт – Гб,

електронна обчислювальна машина – ЕОМ,

Application Programming Interface – «програмний інтерфейс програми» – API.

ВСТУП

За останні 20-30 років все більше різних галузей намагаються створити, або вже створити інформаційні системи та програми для своїх потреб. Підвищена ефективність роботи, автоматизація певних процесів, простіший спосіб обмін інформацією та інші причини є хорошим підґрунтям для створення предметно-орієнтованої ІС майже у будь-якій галузі.

Автомобільні стоянки, як і автомобілі існують доволі давно, але інформаційні технології у даній галузі розвинені не рівномірно. Нові підземні або наземні паркінги, зазвичай мають інформаційні системи, а автостоянки, які відкрилися та працюють 10 років тому чи пізніше у більшості своїй до сих пір ведуть паперову документацію та не мають ніяких засобів автоматизації.

Слід зазначити, що парковкою або паркінгом вважається територія для паркування автомобільного транспорту, на якій охороняється територія паркінгу і власник даної території не несе відповідальності за автомобілі, які припарковані на території. Автомобільна стоянка або автостоянка – це територія для паркування транспортних засобів і власник даної несе відповідальність за причинену шкоду транспортним засобам, або за їх викрадення.

Черговий на автостоянці – це людина, яка відповідальна за основні процеси, а саме: заключення контрактів, облік машин, фіксація в'їзду/виїзду та інші.

Створення ІС перенесе облік даних з різних папок та журналів до єдиної системи, чим полегшить роботу та збільшить її ефективність, виключить з роботи чергового деякі розрахунки та додасть можливість швидкого переносу звітності до паперового формату за допомогою принтера.

Розділ 1. Характеристика та аналіз предметної галузі

1.1. Характеристика предметної галузі та об'єкта дослідження

Автомобільною стоянкою вважається будівля, споруда (частина будівлі, споруди) або спеціальний відкритий майданчик, призначений для зберігання (стоянки/парковки) автомобілів.

Типи автомобільних стоянок за тривалістю знаходження на них автомобілів поділяються на:

- Автостоянки для постійного зберігання автомобілів біля житлових будинків, житлових кварталів, на міжрайонних територіях. Тривалість зберігання понад 1 добу. Ці автостоянки використовують для розміщення автомобілів, що належать громадянам. Залежно від рівня обслуговування такі стоянки можуть бути платними із закріпленням місць за громадянами та безплатними, вільного користування;
- Автостоянки великої тривалості зберігання у підприємств, установ та міських комплексів для розміщення автомобілів, що належать робітникам, службовцям та відвідувачам, тривалістю понад 8 год. Ці автостоянки в залежності від типу установи можуть бути загального користування або лише для службових автомобілів. Останнє виправдане лише у частині міста зі сформованою тісною забудовою, як правило, у центральній або старій частині міста;
- Автостоянки середньої тривалості зберігання біля будівель та споруд, що періодично збирають великі маси людей (стадіони, театри, кіноконцертні зали, ресторани, великі торгові центри), на період 2—4 год.

За типом територіального розміщення автомобільні стоянки та парковки поділяються на:

- Вуличні – зазвичай це парковки, які розміщені вздовж тротуару;
- Підземні – автостоянки або парковки, основна частина яких знаходиться під землею.

- Наземні – автостоянки або парковки, які знаходяться у приміщенні. Можуть бути одно- або багатоповерховими.
- Позавуличні – автостоянки або парковки, які знаходяться недалеко від дороги та мають виїзд до неї. Зазвичай територія є огороженою.

В даному проєкті ми розглядаємо платну позавуличну, наземну або підземну автомобільну стоянку для постійного зберігання транспортних засобів. На такій автостоянці власник ТЗ, він же клієнт, проводить оплату за оренду місця проводиться 1 раз в певний період. Період може варіюватися від 1 дня до 1 року та зазначається клієнтом, тобто перший раз клієнт може заплатити за неділю, а другий за 2 місяці. Але клієнт не зможе користуватися орендованим раніше місцем, якщо дата кінця оренди, зазначена у контракті про оренду місця пройшла і новий контракт не був складений та оплачений.

Система, яка пропонується в бакалаврському дипломному проєкті має виключити велику кількість паперових журналів, веденням яких займається черговий. Розробити технологію опрацювання даних, яка дозволить автоматизувати всі операції обліку та аналізу роботи автомобільної стоянки.

Організаційна структура автостоянки представлена на рисунку 1.1.



Рис. 1.1. Організаційна структура автостоянки

При такій організаційній структурі робітники виконують такі функції:

1. Директор:
 - 1.1. Контролює роботу усіх інших працівників та керує ними;
 - 1.2. Приймає майже всі рішення, які стосуються автостоянки.
2. Черговий:

- 2.1. Слідкує за входом/виходом з автостоянки;
- 2.2. Складає контракти з клієнтами та приймає оплату;
- 2.3. Веде облік складених контрактів;
- 2.4. Спілкується з клієнтами.
- 3. Бухгалтер:
 - 3.1. Ведення бухгалтерського обліку;
 - 3.2. Начислення заробітної плати робітникам;
 - 3.3. Ведення касових операцій, спілкування з банками;
 - 3.4. Прийом прибутку від контролюючих;
 - 3.5. Внесення прибутку до банку.
- 4. Різноробочий:
 - 4.1. Прибирання автостоянки;
 - 4.2. Ремонтні роботи.
- 5. Охорона;
 - 5.1. Охорона автостоянки.

Виходячи з аналізу предметної області можна виділити наступні бінарні зв'язки:

- 1. "Місце" та "Контракт" зі зв'язком "записується в";
- 2. "Клієнт" та "Машина" зі зв'язком "має";
- 3. "Контракт" та "Машина" зі зв'язком "записується в";
- 4. "Черговий" та "Документ про оренду" зі зв'язком "підписує";
- 5. "Машина" та "В'їзд/виїзд" зі зв'язком "виконала";

1.2. Аналіз літературних джерел та практичного досвіду використання ІС і технологій в предметній галузі

Було проведено пошук існуючих інформаційних систем для обраної предметної області. У відкритому доступі існує дуже маленька кількість систем, які відносяться до обраної області. В результаті пошуку було виділено

З системи, які найбільш схожі на систему, яка запропонована в даному проєкті.

Програма «Avtojanka» призначена для автоматизації роботи автостоянки [1]. Програма веде електронні журнали обліку реєстрації та вибуття автотранспортних засобів автостоянки. Наявність довідників, що автоматично заповнюються, дозволяє декількома рухами мишки швидко зареєструвати автомобіль. А також випустити його з території автостоянки. Інтерфейс даної програми продемонстрований на рисунку 1.3.

Програма формує такі види звітів:

1. виторг каси;
2. звіт про боржників;
3. повний звіт про автотранспортний засіб (власника);
4. звіт про охорону автостоянки.

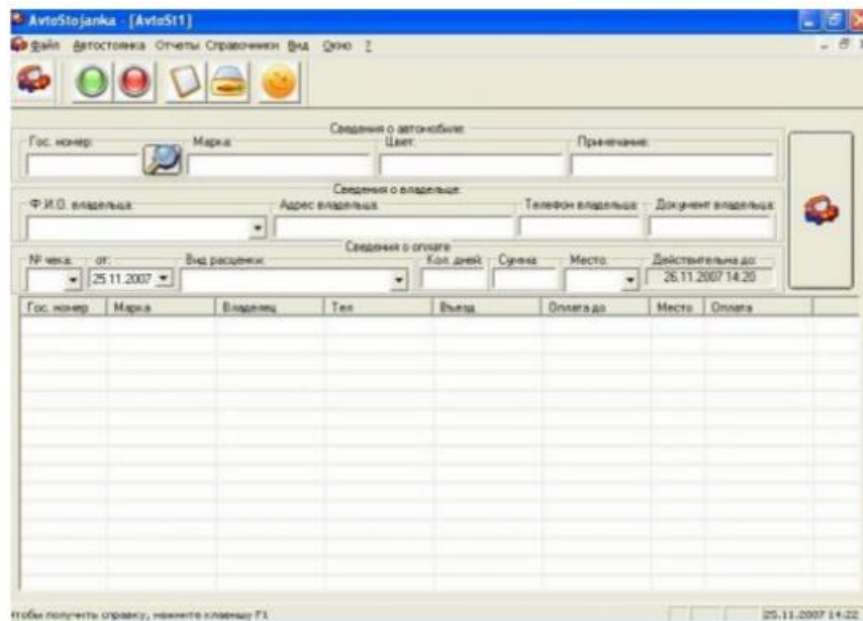


Рис. 1.2 Інтерфейс програми Avtojanka

Програма «AllStojanka», яка продемонстрована на рисунку 1.4. дозволяє [2]:

1. вести облік в електронному вигляді в базі даних з простим і зрозумілим інтерфейсом;
2. отримувати достовірну та оперативну інформацію по взаєморозрахунках;

3. наочно відобразити схему автостоянки з поділом на цікаві і вільні місця;
4. показати на схемі розташування конкретного транспортного засобу;
5. швидко оформляти операцію постановки і зняття транспортного засобу з стоянкового місця;
6. приймати і оформляти отримання грошових коштів з печаткою товарного чека;
7. автоматично розраховувати вартість послуг по стоянці використовуючи різні варіанти і тарифи;
8. вести облік знижок;
9. автоматично розраховувати і нараховувати пеню за прострочення;
10. контролювати дні і час прострочення.

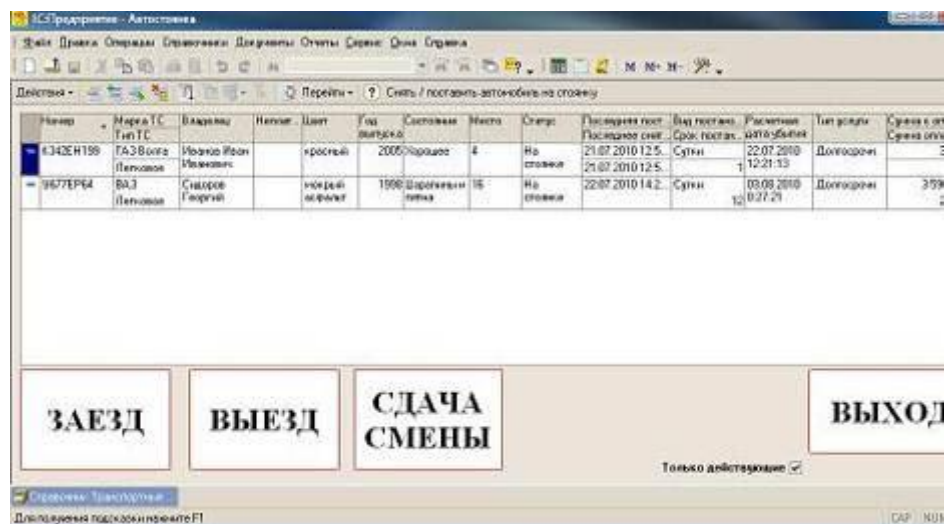


Рис. 1.3 Інтерфейс програми AllStojanka

Автоматизована система контролю оплати паркування «АСКОП SEA», зображення інтерфейсу якої представлено на рисунку 1.4 [3]. Призначення даної системи контролю парковок:

1. Автоматизація процесів оплати за паркування;
 - 1.1. готівкою або банківською картою через паркомати;
 - 1.2. за допомогою мобільного додатку SEA Parking;
2. Диспетчеризація паркоматів;
3. Контроль оплат водіїв за допомогою мобільного додатку Parking Inspector;

4. Формування фінансових та інших статистичних звітів;
5. Створення та друк протоколу за сплату штрафу;
6. Відправлення протоколу правопорушникові;

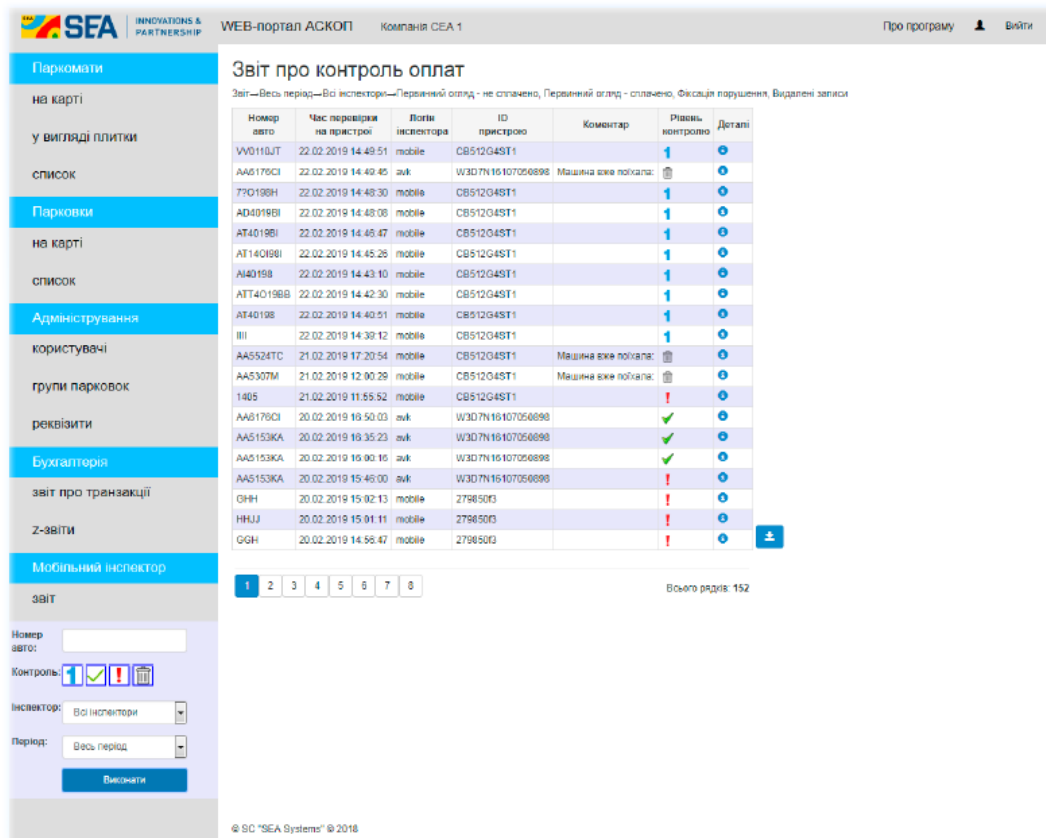


Рис. 1.4 Інтерфейс програми АСКОП SEA

Обмін інформацією між мобільними пристроями і сервером АСКОП захищений за допомогою сервісу Mobile ID. На сервері з базою даних зберігається та обробляється вся інформація про контрольовані парковки, доступ до якої (згідно з визначеними правами) власники та обслуговуючий персонал паркувальних систем отримують через інтернет за допомогою веб-порталу АСКОП і мобільного додатку Parking Inspector, а водії використовують програму SEA Parking.

Можливості системи:

1. Ведення обліку та архівів, генерування статистики і звітів

2. Підтримка безготівкових способів оплати за паркування
3. Мобільні програми SEA Parking для клієнтів і Parking Inspector персоналу
4. Високий рівень адаптації та адміністрування системи
5. Можливість інтеграції з іншими інформаційними системами

Таблиця 1.1. Порівняння систем

Назва системи	Ліцензія	Підтримка системи	Недоліки	Особливості використання системи
Avtojanka	2000 грн.	Відсутня	Завелика кількість можливостей. Неможливість зв'язку с розробником системи.	—
AllStojanka	Безкоштовна	Відсутня	Завелика кількість можливостей. Неможливість зв'язку с розробником системи.	—
АСКОП SEA	Пропрієтарна	Присутня	1. Необхідність встановлення додаткового обладнання. 2. Система більше підходить для паркінгу, ніж для позавуличних автостоянок. 3. Завелика кількість можливостей.	1. Необхідне постійне підключення до інтернету. 2. Контролюючий не бере участь у процесі паркування.

На основі проведеного аналізу існуючих систем, можна сказати, що існуючі системи не можна вважати ідеальним рішенням. АСКОП SEA можна вважати вже готовою системою, навколо якої легше побудувати нову автостоянку, ніж впровадити дану систему у уже існуючий робочий процес поза вуличних автостоянок. Avtojanka та AllStojanka мають великий функціонал, але немає впевненості, що автостоянки, які існують вже кілька років будуть повністю використовувати функціонал даних програм.

Розділ 2. Розробка вимог і моделювання інформаційної системи/підсистеми

2.1. Аналіз і специфікація вимог до інформаційної системи/підсистеми

Особами, які зацікавлені в успішному завершенні проєкту виступають чергові та клієнти автостоянок. Бізнес цілями чергових на автостоянці є:

- Автоматизування всіх операцій обліку та аналізу;

- Зручний облік даних.

Бізнес ціллю клієнтів є зменшення часу очікування.

З цього можна визначили, що бізнес вимогами до системи є:

- **Автоматизування всіх операцій обліку та аналізу.** Чергові мають записувати багато інформації. Фіксування в'їзду та виїзду транспортного засобу на територію автостоянки, укладення нових контрактів на оренду, додавання даних про транспортні засоби та їх власників. Це все можливо перенести до ІС.
- **Зручний облік даних.** Система повинна зберігати велику кількість даних, що може замінити декілька десятків папок з папером і підвищити швидкість і зручність користування інформацією.
- **Зменшення часу очікування.** У клієнтів автостоянки не завжди є час чекати доки черговий заповнить контракт.

Система повинна мати таку функціональність:

- можливість фіксувати кожен в'їзд та виїзд машин на територію автостоянки та заносити цю інформацію до відповідного списку, який за необхідності можливо буде переглянути ;
- виводити на екран список усіх місць на автостоянці та надавати інформацію щодо статусу місця. Чи пусте/зайняте/заброньоване місце;
- функцію створення нового контракту на оренду місця, перегляд контрактів;
- можливість перегляду прибутку за обраний користувачем період часу;
- функцію авторизації користувача;
- функцію запису, редагування та видалення записів БД про власників транспортних засобів, самих транспортних засобів та місць на автостоянці;

Розроблена діаграма вимог. Вона зображена на рисунку 2.1.

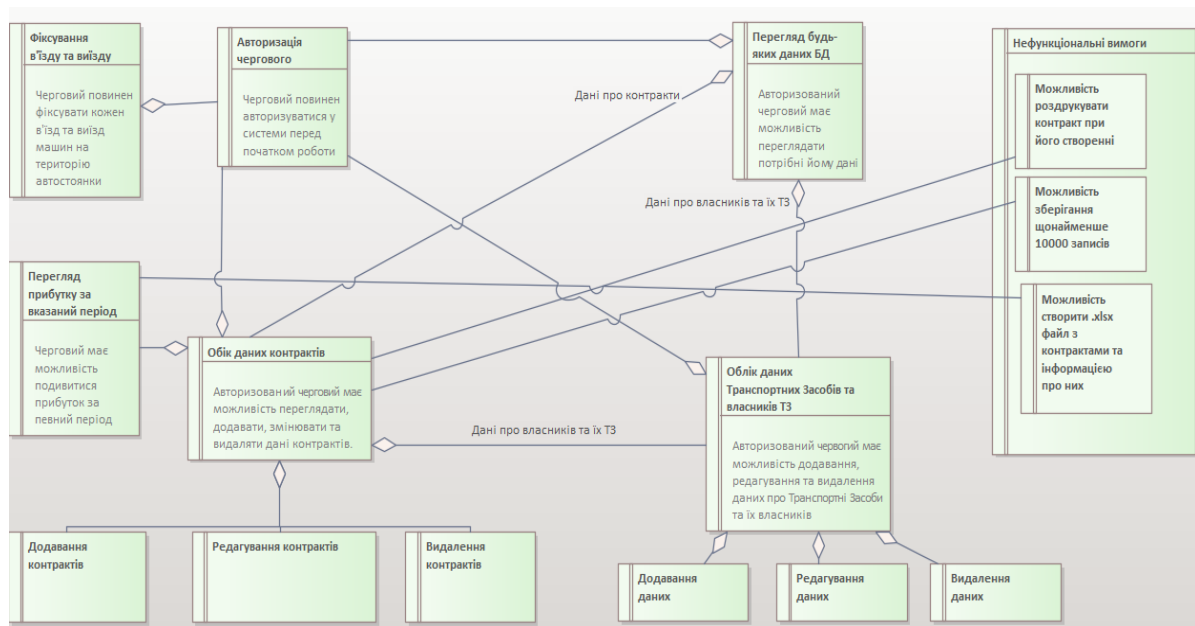


Рис. 2.1. Діаграма вимог

До нефункціональних вимог системи відносяться:

- Система повинна коректно працювати при кількості записів бази даних 10000;
- Система повинна мати можливість роздрукувати контракт при його створенні;
- Система повинна мати можливість експорту даних до файлу формату XLSX.

Специфікація вимог зображена на рисунку 2.2. та 2.3.

Item	Stereotype	Status	Difficulty	Priority
☒ Авторизація чергового Черговий повинен авторизуватися у системи перед початком роботи	FunctionalRequir...	Proposed	Medium	Medium
☒ Облік даних контрактів Авторизований черговий має можливість переглядати, додавати, змінювати та видаляти дані контрактів.	Functional	Proposed	Medium	Medium
☒ Перегляд прибутку за вказаний період Черговий має можливість подивитися прибуток за певний період	Functional	Proposed	Medium	Medium
☒ Додавання контрактів	Functional	Approved	Medium	Medium
☒ Редагування контрактів	Functional	Validated	Medium	Medium
☒ Видалення контрактів	Functional	Proposed	Medium	Medium
☒ Облік даних Транспортних Засобів та власників ТЗ Авторизований черговий має можливість додавання, редагування та видалення даних про Транспортні Засоби та їх власників	Functional	Approved	Medium	Medium

Рис. 2.2. Специфікація вимог

✓ <u>Видалення даних</u>	Functional	Proposed	Medium	Medium
✓ <u>Додавання даних</u>	Functional	Proposed	Medium	Medium
✓ <u>Редагування даних</u>	Functional	Proposed	Medium	Medium
✓ <u>Перегляд будь-яких даних БД</u> Авторизований черговий має можливість переглядати потрібні йому дані	Functional	Proposed	Medium	Medium
✓ <u>Фіксування в'їзду та виїзду</u> Черговий повинен фіксувати кожен в'їзд та виїзд машин на територію	Functional	Proposed	Medium	Medium
✓ <u>Нефункціональні вимоги</u>	Functional	Proposed	Medium	Medium
✓ <u>Можливість зберігання щонайменше 10000 записів</u>	FunctionalRequir...	Proposed	Medium	Medium
✓ <u>Можливість роздрукувати контракт при його створенні</u>	Functional	Proposed	Medium	Medium
✓ <u>Можливість створити .xlsx файл з контрактами та інформацією про них</u>	Functional	Proposed	Medium	Medium

Рис. 2.3. Специфікація вимог

2.2. Постановка задачі

Характеристика задачі

Основною задачею інформаційної системи підтримки роботи автомобільних стоянок є здійснення обліку даних.

Клієнти автостоянки мають оплачувати контракти на оренду місця для того, щоб зберігати свій ТЗ на території автостоянки. Для постійних клієнтів оплата проводиться 1 раз в певний період. Даний період залежить від потреб самого клієнта і може варіюватися від 1 дня до 1 року. Чим більша різниця у дати початку та кінця контракту, тим меншим є тариф. Маючи діючий контракт про оренду місця, клієнт має можливість забирати або залишати свій ТЗ коли забажає.

Періодичність вирішення задачі залежить від функцій, які потрібно виконувати в той чи інший момент часу. Введення даних про вїзд та виїзд виконується цілодобово, Заключення контрактів на користування автостоянкою обумовлена частотою звертань клієнтів за послугою

довготривалого зберігання транспортних засобі і є випадковою величиною. Формування звітності про роботу автостоянки виконується кожного місяця.

При розв'язанні задачі здійснюється керування наступними об'єктами: бухгалтерія, черговий та клієнт.

Бухгалтер використовує вихідну інформацію для ведення обліку усіх оплат, оплати податків та іншого.

Черговий в свою чергу використовує вихідну інформацію ІС для моніторингу стану автостоянки.

Клієнт на виході роботи системи отримує роздрукований контракт, який свідчить про можливість користування послугами автостоянки.

Інформаційна модель задачі представлена на рисунку 2.4

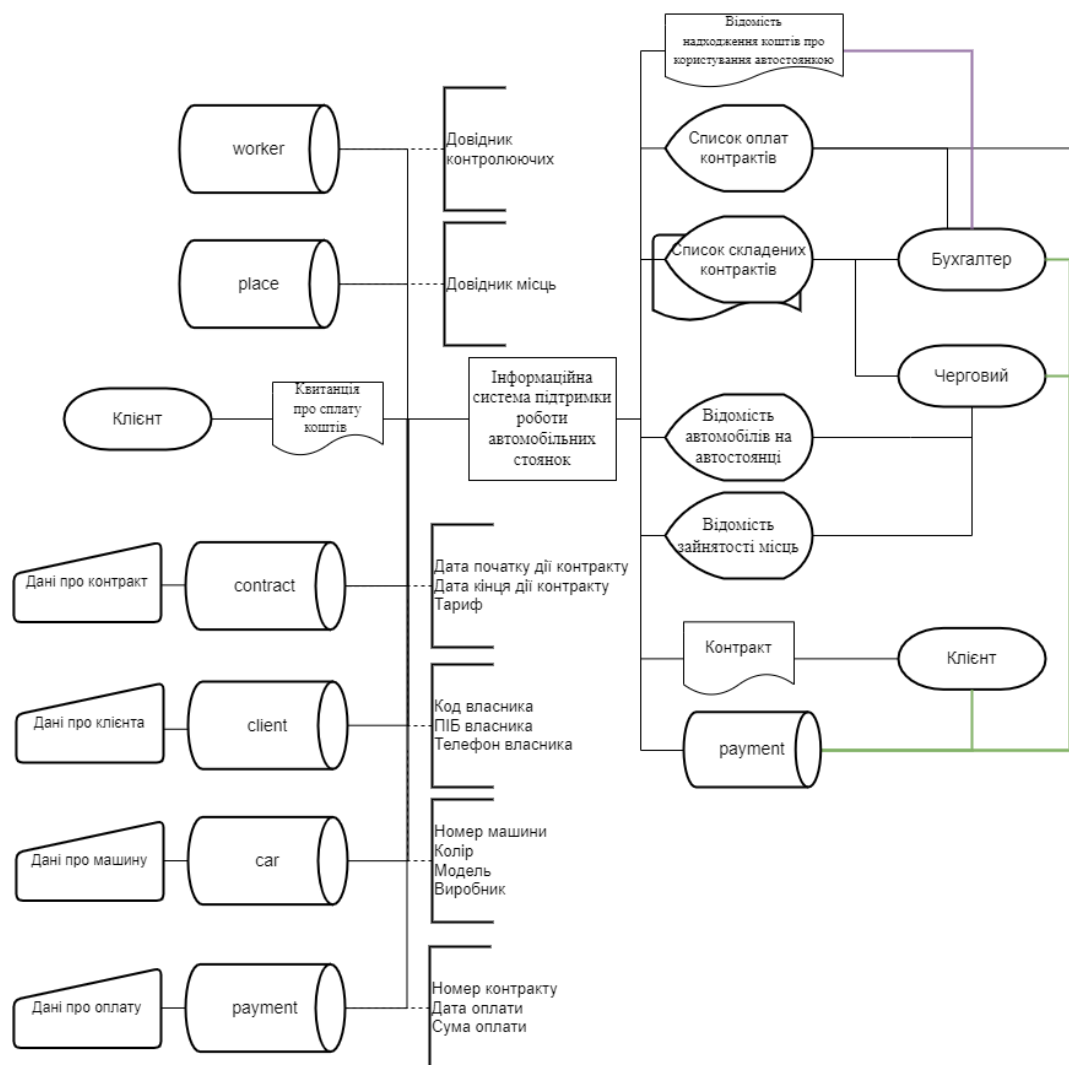


Рис. 2.4. Інформаційна модель задачі

Вихідна інформація

Вихідними даними ІС є список контрактів, який використовується контролюючими та бухгалтером. Контролюючий використовує список для того, щоб знати чи має право власник користуватись послугами стоянки на даний момент при втраті контракту клієнтом, а бухгалтер для підрахунку прибутку. Черговий використовує вихідні повідомлення з відомостями про місця та контракти для контролю ситуації на автостоянці.

Клієнт використовує контракт, який є вихідним повідомленням для користування послугами автостоянки.

Вихідні повідомлення описані у таблиці 2.1.

Структурними одиницями інформації є:

1. Номера машин, які використовується у повідомленні «Відомість автомобілів на автостоянці».
2. Номера місць, які використовується у повідомленні «Відомість зайнятих місць».
3. Номер контракту, який використовується у повідомленні «Контракт» та «Список складених контрактів».

Таблиця 2.1 Перелік і опис вихідних повідомлень

№ з/п	Назва вихідного повідомлення	Ідентифікатор	Форма подання і вимоги до неї	Періодичність видання	Термін видання і допустимий час затримки	Користувачі інформації
1	Список складених контрактів	contracts	Екранна форма, документ, файл	При необхідності	1 хвилина	Черговий, бухгалтер
2	Відомість автомобілів на автостоянці	cars	Екранна форма	При необхідності	1 хвилина	Черговий
3	Відомість зайнятих місць	places	Екранна форма	При необхідності	1 хвилина	Черговий
4	Контракт	contract	документ	При необхідності	5 хвилин	Клієнт

5	Чек оплати	check	документ	При необхідності	1 хвилина	Клієнт, Черговий, Бухгалтер
6	Відомість надходження коштів про користування автостоянкою	notice_rec eipt	документ	При необхідності	5 хвилин	Бухгалтер

Формуляр-зразок контракту представлений у додатку Г.

Вхідна інформація

Вхідними даними ІС є інформація, яка формує контракт, а саме:

1. Особисті дані клієнта та дані про ТЗ, які клієнт надає один раз при складанні свого першого контракту;
2. Номер місця, дані про час початку та кінця контракту та тариф контракту, який контролюючий записує у контракт при складанні кожного контракту;

Вхідні повідомлення описані у таблиці 2.2.

Структурними одиницями інформації є:

1. Дата та час, подається з точністю до 1 хвилини, використовується у повідомленні «Дата та час початку контракту» та «Дата та час кінця контракту».
2. Тариф – подається з точністю до 0.01 гривні, використовується у повідомленні «Тариф».
3. Номер місця, подається з точністю до одиниці, використовується у повідомленні «Довідник місць».

Таблиця 2.2 Перелік і опис вхідних повідомлень

№ з/п	Назва вхідного повідомлення	Ідентифікатор	Форма подання	Термін і частота надходження	Джерело
1	Довідник місць	place	Файл	1 раз в певний період	Контролюючі
2	Паспорт клієнта	client_data	документ	1 раз	Клієнт
3	Технічний паспорт ТЗ	car_data	документ	1 раз	Клієнт
4	Дата та час початку контракту	start_date	Екранна форма	1 раз в певний період	Контролюючі
5	Дата та час кінця контракту	end_date	Екранна форма	1 раз в певний період	Контролюючі
6	Тариф	tarif	Екранна форма	1 раз в певний період	Контролюючі
7	Квитанція про сплату коштів	payment_receipt	Документ	1 раз в певний період	Клієнт

Алгоритм розв'язання задачі

Використовувана інформація

В процесі роботи системи використовуються масиви інформації, які представлені на таблиці 2.3.

Таблиця 2.3 Перелік масивів використовуваної інформації

Масив	Ідентифікатор	Максимальна кількість записів
Масив паролів	passwords	50
Масив місць автостоянки	places	500
Масив клієнтів	clients	2000
Масив ТЗ	cars	2500
Масив контрактів	contracts	10000

Масив паролів використовуються в ІС для перевірки пароля при авторизації користувача. Масив місць автостоянки використовуються на головному для показання зайнятості місць на стоянці. Масив клієнтів на ТЗ використовується при створенні контракту. Масив контрактів виводиться у екранній формі «Контракти».

Результати розв'язання

В результаті роботи системи заповнюються масиви інформації, які представлені на таблиці 2.4.

Таблиця 2.4 Перелік масивів результатної інформації

Масив	Ідентифікатор	Максимальна кількість записів
Масив контрактів	contracts	10000
Масив в'їзду/виїзду	InOutputs	50000

Масив контрактів заповнюється при створенні нового контракту. Масив в'їзду/виїзду заповнюється черговим при в'їзді/виїзді на територію автостоянки.

Математичний опис

Атрибути, які використовуються у математичному описі описані у таблиці 2.5.

Таблиця 2.5. Опис математичних атрибутів

Назва атрибута	Ідентифікатор	Умовне позначення	Характеристика атрибута
Тариф (грн/ день)	<i>TARIF</i>	<i>P</i>	Кількісний, факт
Сума до оплати контракту	<i>SUM</i>	<i>S</i>	Кількісний, розрах
Код контракту	<i>CONTRACT_ID</i>	<i>d</i>	якісний, груповий
Код клієнта	<i>CLIENT_ID</i>	<i>l</i>	якісний, груповий
Дані клієнта	<i>CLIENT_DATA</i>	-	якісний, груповий
Дата початку контракту	<i>DATE _START</i>	<i>x</i>	якісний, груповий
Дата кінця контракту	<i>DATE _END</i>	<i>y</i>	якісний, груповий
Код ТЗ	<i>CAR_ID</i>	<i>v</i>	якісний, груповий
Код місця	<i>PLACE_ID</i>	<i>q</i>	якісний, груповий
Початок періоду	<i>PERIOD_START</i>	<i>a</i>	Кількісний, факт
Кінець періоду	<i>PERIOD_END</i>	<i>B</i>	Кількісний, факт
Кількість днів для разового користування послугами автостоянки	<i>QUANTITY_ONCE</i>	<i>o</i>	Кількісний, факт

Тариф може змінюватися як від взаємовідношень між автостоянкою та клієнтів, так і від потреб самої автостоянки.

Показники, які необхідно розрахувати:

1. Кількість днів, які варіюють у контракті – T_d ;
2. Сума оплати i -го контракту l -м клієнтом q -го місця – S_i^{lq} ;
3. Прибуток за період – G .
4. Сума до сплати за разове користування стоянкою q -го місця – W_q

Формули для розрахунку показників:

1. Кількість днів, які варіюють у контракті = дата кінця контракту – дата початку контракту.

$$T_d = y_d - x_d; \quad (2.1)$$

Де d – код контракту.

2. Сума до сплати = Тариф за контрактом $\times$ Кількість днів, які варіюють у контракті.

$$S_d^{lq} = P_d \times T_d; \quad (2.2)$$

Де l – код клієнта, d – код контракту, q – код місця.

3. Прибуток за період = сумі прибутку контрактів, які склались від дати початку періоду до дати кінця періоду.

$$G = \sum_{i=b}^a T_i^d = T_b^d + T_{b+1}^d + T_{b+2}^d + \dots + T_{a-1}^d + T_a^d \quad (2.3)$$

Де d – код контракту, a – кінець періоду, b – початок періоду.

4. Сума до сплати за разове користування стоянкою = кількість днів разового користування послугами автостоянки $\times$ тариф оплати місця за день.

$$W_q = o \times P \quad (2.4)$$

Де q – код місця.

Алгоритм розв'язання задачі на ЕОМ

Сформовані системою результати можуть подаватися у різному вигляді на вибір користувача системи. Сформовані системою контракти можуть

подаватися у вигляді друкованого документу, xlsx файлу або у екранній формі.

Інформація про в'їзд/виїзд подається у вигляді екранної форми.

Алгоритм розв'язання задачі подано на рисунку 2.8.

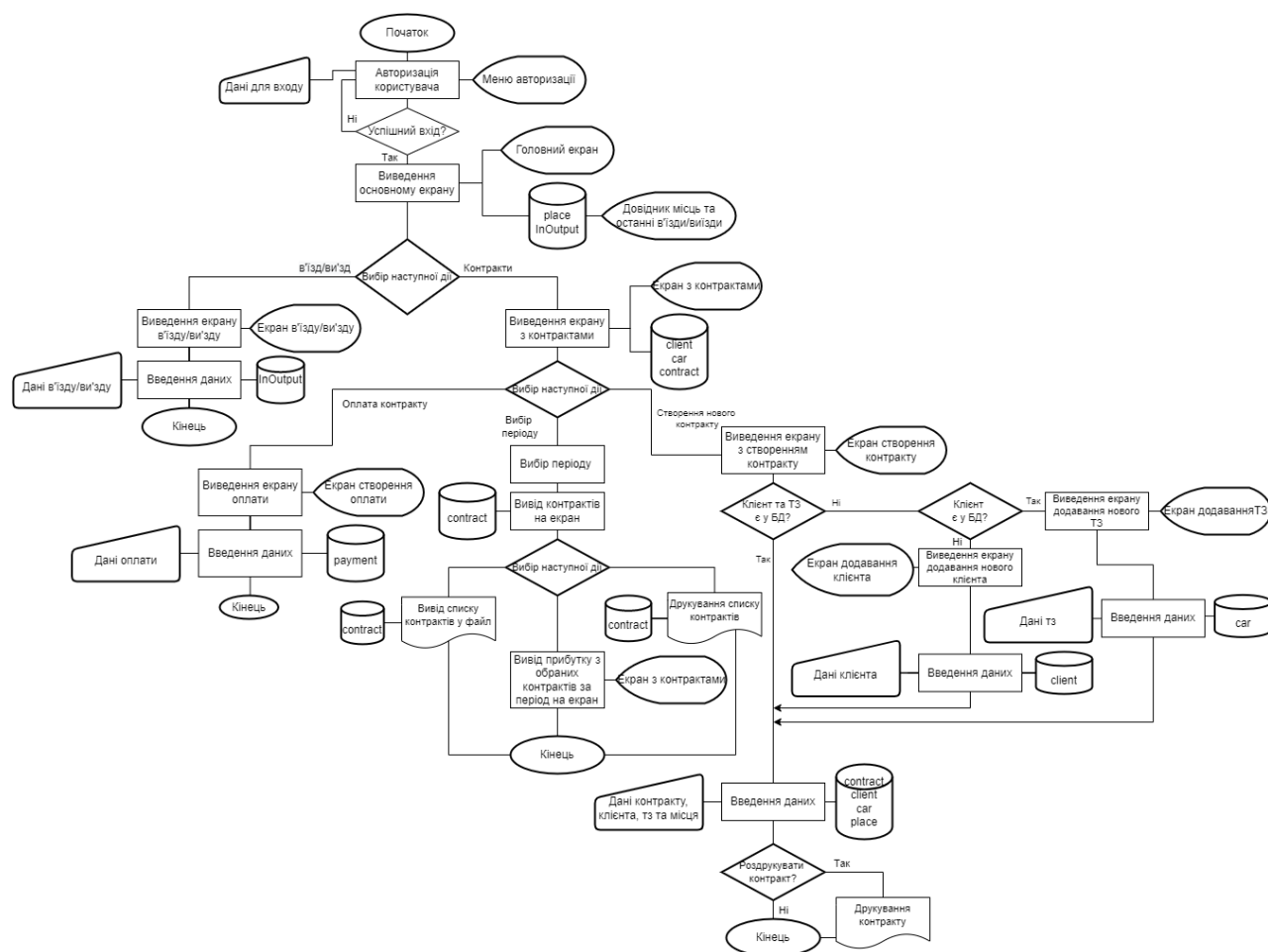


Рис. 2.8. Алгоритм вирішення задачі

2.3. Моделювання інформаційної системи/підсистеми

Для більш узагальненого вигляду функціональних вимог була створена діаграму прецедентів, яка зображена на рисунку 2.9 та діаграми послідовності для кожного випадку використання, які зображені на рисунках 2.10. – 2.14.

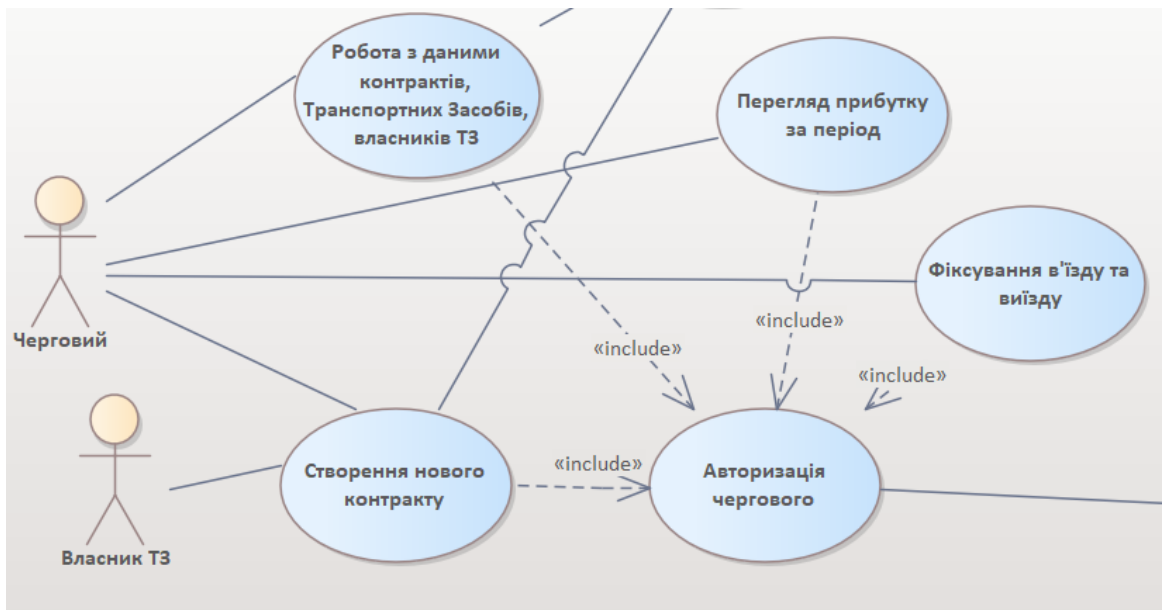


Рис. 2.9. Діаграма прецедентів

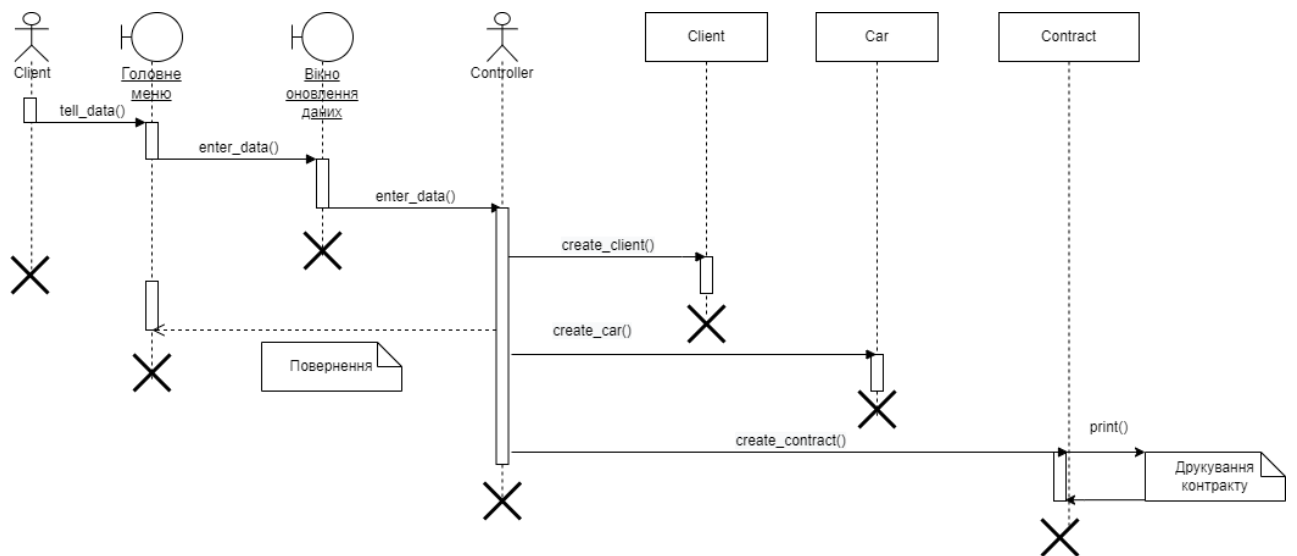


Рис. 2.10. Діаграма послідовності для випадку використання «Створення нового контракту»

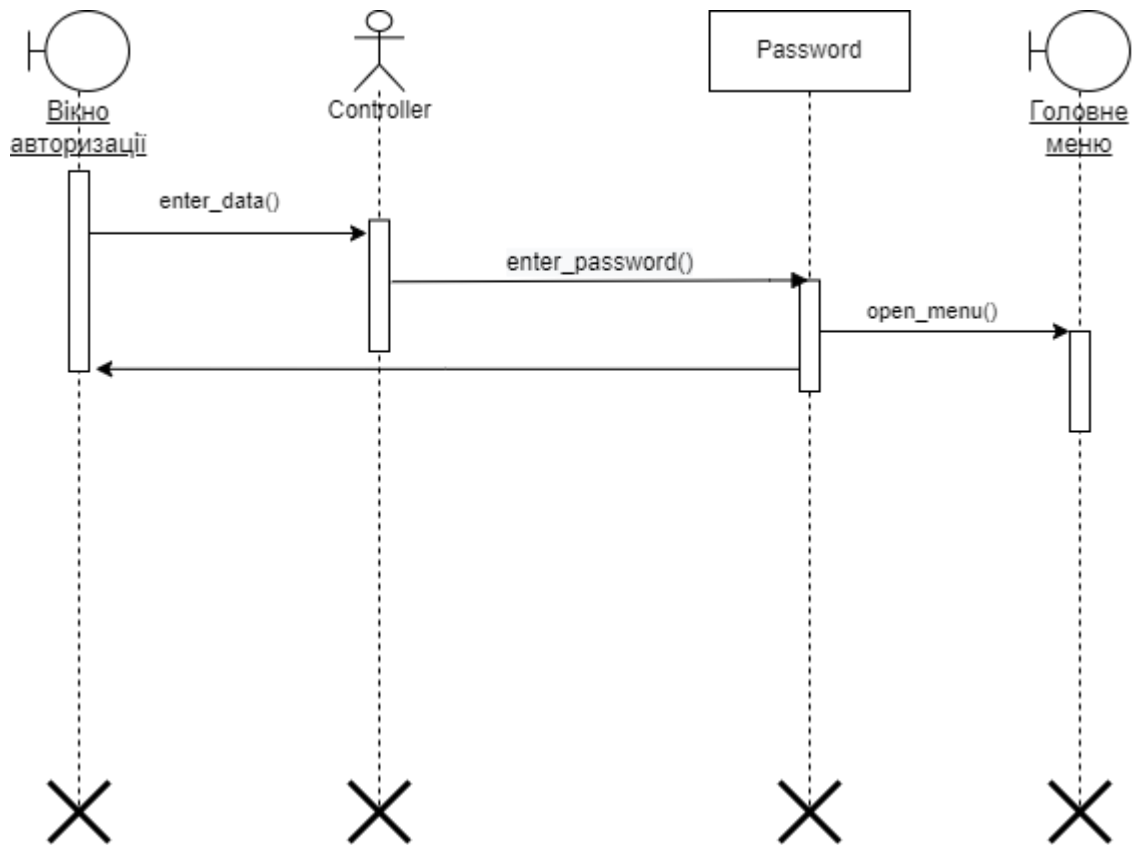


Рис. 2.11. Діаграма послідовності для випадку використання «Авторизація чергового»

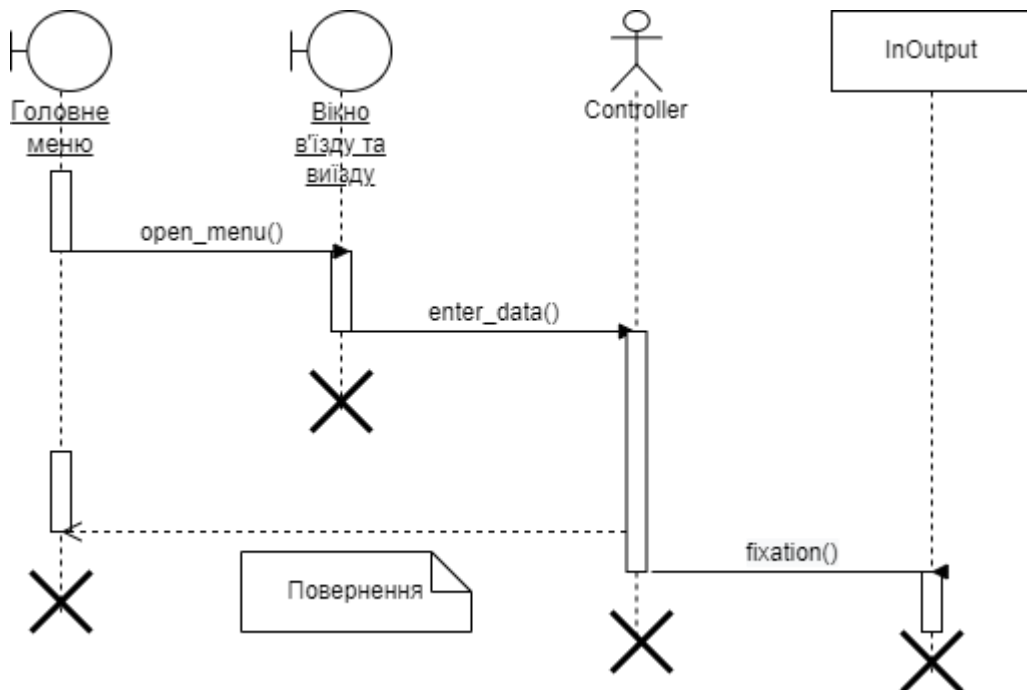


Рис. 2.12. Діаграма послідовності для випадку використання «Фіксування в'їзду та виїзду»

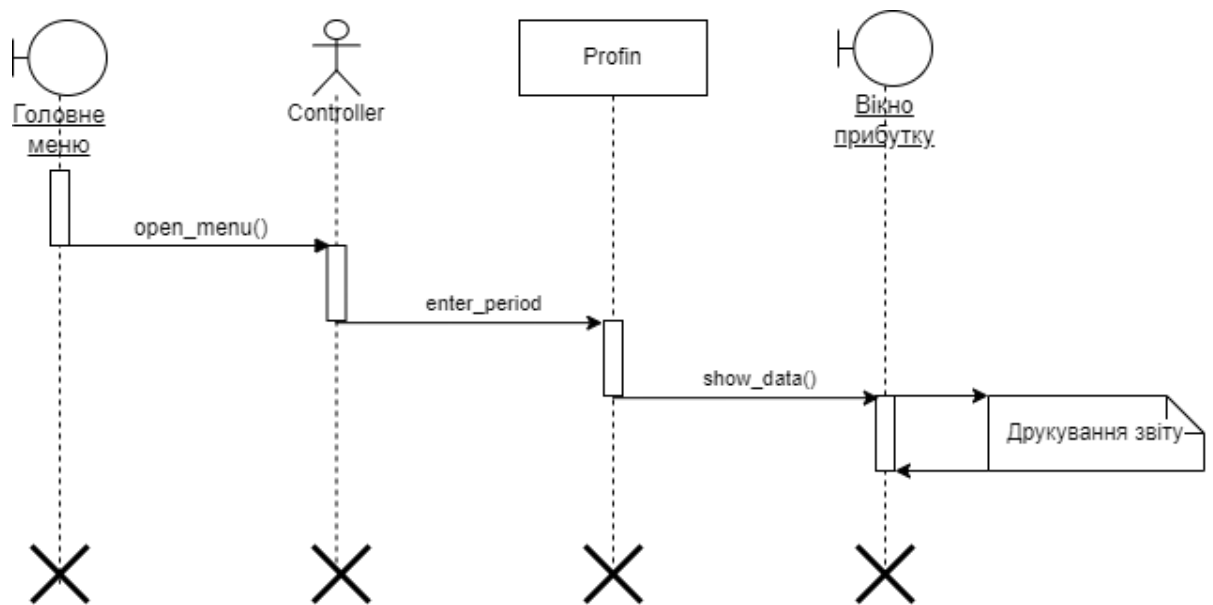


Рис. 2.13. Діаграма послідовності для випадку використання «Перегляд прибутку за період»

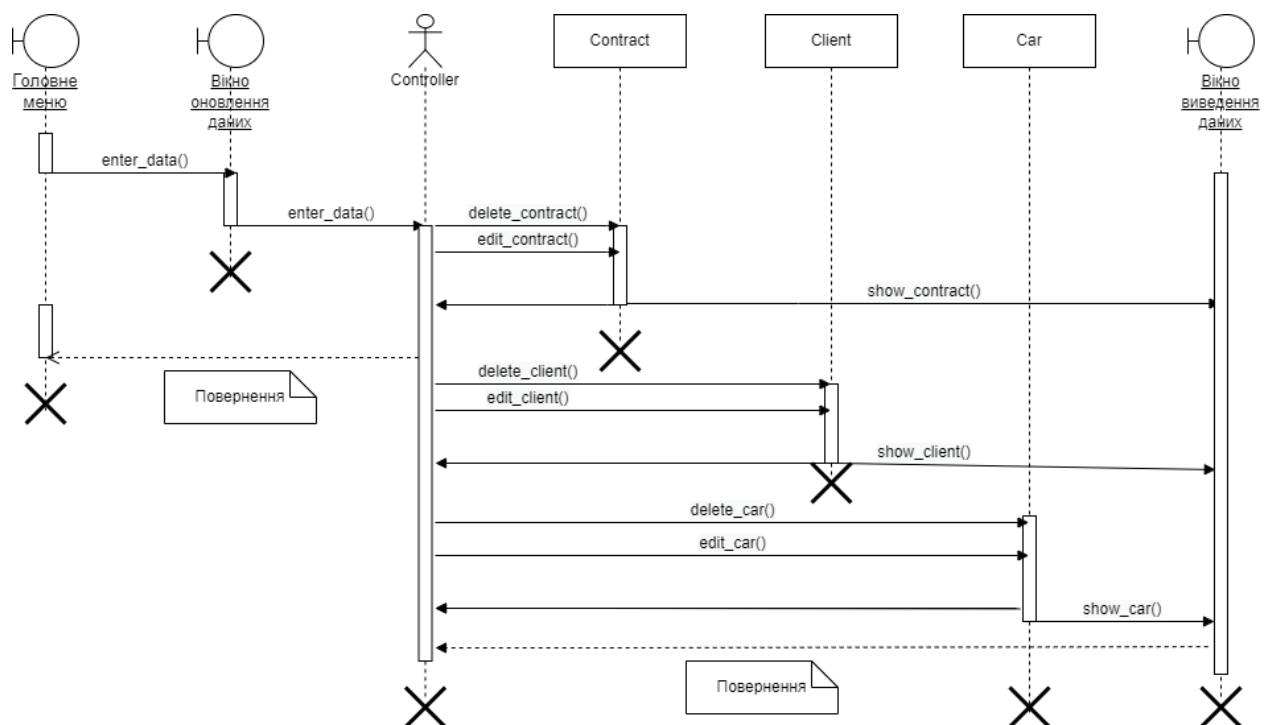


Рис. 2.14. Діаграма послідовності для випадку використання «Робота з даними»

Модель структури класів

Була розроблена модель структури класів ІС. Вона подана на рисунку 2.15.

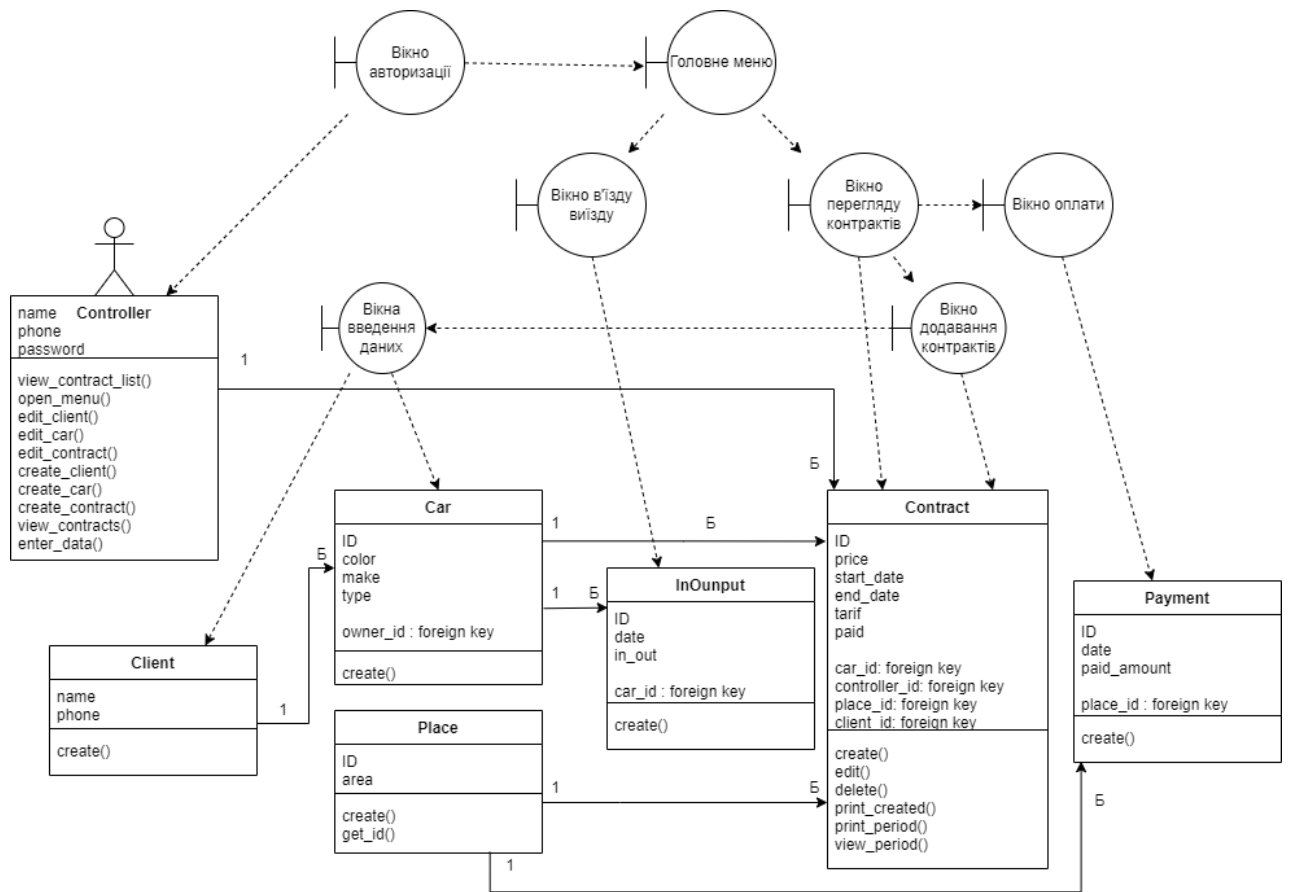


Рис. 2.15. Модель структури класів

Розподіл вимог за компонентами системи

Була розроблена діаграма трасування, яка представлена на рисунку 2.16 . Та матриця взаємозв'язків для доповнення діаграми трасування. Матриця представлена на Таблиці 2.6.

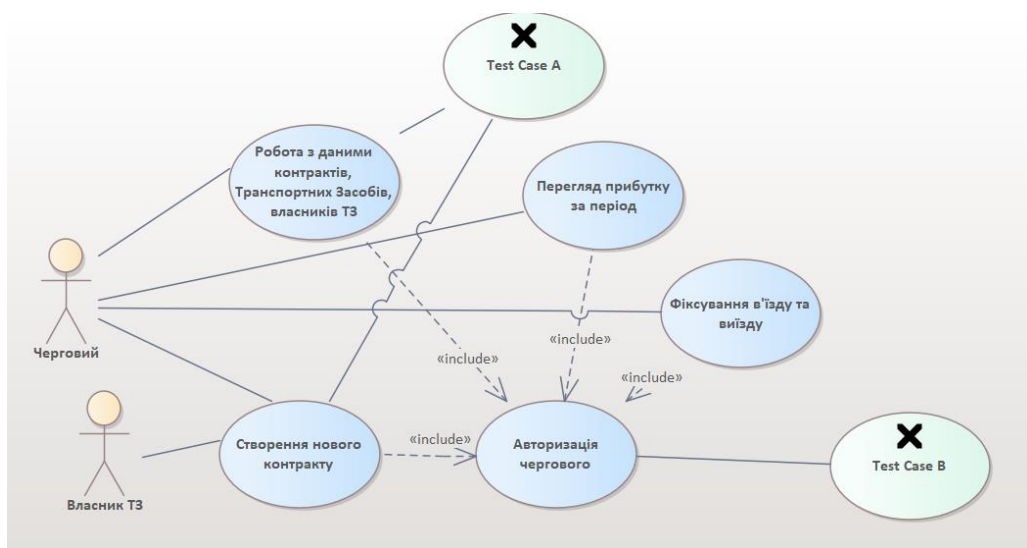


Рис. 2.16. Діаграма трасування

Таблиця 2.6. «Матриця взаємозв'язків»

	Створення нового контракту	Авторизація чергового	Фіксування в'їзду та виїзду	Перегляд прибутку за період	Робота з даними контрактів, Транспортних Засобів, власників ТЗ
Черговий	Так	Так	Так	Так	Так
Власник ТЗ	Так	-	-	-	-

Розділ 3. Проектування та реалізація компонентів системи/підсистеми

3.1. Інформаційне забезпечення

Для ведення і підтримки бази даних була обрана СКБД SQL Server 2019, бо вона є однією з найпопулярніших СКБД та її функцій та можливостей достатньо для реалізації даної ІС.

SQL Server є однією з найпопулярніших систем управління базами даних (СУБД) у світі. Ця СУБД підходить для різних проектів: від невеликих додатків до великих високонавантажених проектів.

SQL Server тривалий час був виключно системою керування базами даних для Windows, однак, починаючи з версії 16, ця система доступна і на Linux.

SQL Server характеризується такими особливостями як:

- Продуктивність. SQL Server працює дуже швидко.
- Надійність та безпека. SQL Server надає шифрування даних.
- Простота. З цієї СУБД щодо легко працювати та вести адміністрування.

Центральним аспектом у MS SQL Server, як і будь-який СУБД, є база даних. База даних представляє сховище даних, організованих певним способом. Нерідко фізично база даних представляє файл на жорсткому диску, хоча така відповідність необов'язково. Для зберігання та адміністрування баз даних застосовуються системи управління базами даних (database management system) або СУБД (DBMS). І саме MS SQL Server є однією з таких СУБД.

Для організації бази даних MS SQL Server використовує реляційну модель. Ця модель баз даних була розроблена ще в 1970 Едгаром Коддом. А на сьогодні вона фактично є стандартом для організації бази даних.

Реляційна модель передбачає зберігання даних у вигляді таблиць, кожна з яких складається з рядків та стовпців. Кожен рядок зберігає окремий об'єкт, а стовпці розміщуються атрибути цього об'єкта.

Для ідентифікації кожного рядка у межах таблиці застосовується первинний ключ (primary key). Як первинний ключ може виступати один або кілька стовпців. Використовуючи первинний ключ, ми можемо посилатися на

певний рядок у таблиці. Відповідно два рядки не можуть мати один і той же первинний ключ.

Через ключі одна таблиця може бути пов'язана з іншою, тобто між двома таблицями можуть бути організовані зв'язки. А сама таблиця може бути представлена у вигляді відношення ("relation").

Для взаємодії з базою даних використовується мова SQL (Structured Query Language). Клієнт (наприклад, зовнішня програма) надсилає запит мовою SQL за допомогою спеціального API. СКБД належним чином інтерпретує та виконує запит, а потім надсилає клієнту результат виконання [4].

Інформаційна система разом з БД знаходиться на 1 носії даних – ЕОМ, яка знаходиться у кімнаті чергового. Для функціонування ІС на ЕОМ повинно бути встановлено SQL Server та .NET Framework версії 4.0 або вище, який потрібен для запуску та подальшої роботи з ІС.

Структура таблиць БД подана на таблицях 3.1 – 3.7.

Таблиця 3.1. Структура таблиці "Клієнт"

Назва атрибута	Ідентифікатор	Формат	Обов'язковий (необов'язковий)	Обмеження на право звертання	Первинний (вторинний) ключ	Виводимість значень	Дублювання значень, %	Індексне поле
№ Клієнта	ID_Owner	Int	Обов'язковий	Має право черговий	ПК	—	Ні	
ПІБ	Owner_Name	varchar(50)	Обов'язковий			Так	Так	ІНД
Телефон	Owner_Phone	varchar(15)	Обов'язковий			Так	Так	

Таблиця 3.2. Структура таблиці "Черговий"

Назва атрибута	Ідентифікатор	Формат	Обов'язковий (необов'язковий)	Обмеження на право звертання	Первинний (вторинний) ключ	Виводимість значень	Дублювання значень, %	Індексне поле
№ Чергового	ID_Worker	int	Обов'язковий	Має право черговий	ПК	—	Ні	
ПІБ	Name_Worker	varchar (50)	Обов'язковий			Так	Так	ІНД
Телефон	Phone_Worker	varchar (15)	Обов'язковий			Так	Так	
Пароль	Password_Worker	varchar (20)	Обов'язковий			—	Так	

Таблиця 3.3. Структура таблиці "Місце"

Назва атрибута	Ідентифікатор	Формат	Обов'язковий (необов'язковий)	Обмеження на право звертання	Первинний (вторинний) ключ	Виводимість значень	Дублювання значень, %	Індексне поле
№ Місця	ID_Place	int	Обов'язковий	Має право черговий	ПК	Так	Ні	ІНД
Розмір місця	Size_Place	int	Обов'язковий			—	Так	
Зайнятість	busy_Place	bit	Обов'язковий			Так	Так	

Таблиця 3.4. Структура таблиці "Машина"

Назва атрибута	Ідентифікатор	Формат	Обов'язковий (необов'язковий)	Обмеження на право звертання	Первинний (вторинний) ключ	Виводимість значень	Дублювання значень, %	Індексне поле
№ ТЗ	ID_Car	varchar(12)	Обов'язковий	Має право черговий	ПК	Так	Ні	ІНД
Виробник	Car_Maker	varchar(30)				Так	Так	
Модель	Car_Model	varchar(40)				Так	Так	
Колір	Car_Color	varchar(20)	Обов'язковий			Так	Так	
№ Власника	ID_Car_Owner	int	Обов'язковий		ВК	—	Так	

Таблиця 3.5. Структура таблиці "Контракт"

Назва атрибута	Ідентифікатор	Формат	Обов'язковий (необов'язковий)	Обмеження на право звертання	Первинний (вторинний) ключ	Виводимість значень	Дублювання значень, %	Індексне поле
№ Контракту	ID_Document	int	Обов'язковий	Має право черговий	ПК	Так	Ні	
№ Машини	ID_Car	varchar(12)	Обов'язковий		ВК	Ні	Так	
№ Чергового	ID_Worker	int	Обов'язковий		ВК	Ні	Так	
№ Місця	ID_Place	int	Обов'язковий		ВК	Ні	Так	
Дата	take_date	smalld	Обов'язковий			Так	Так	ІНД

початку контракту	te	atetim e	ковий					
Дата кінця контракту	back_da te	smalld atetim e	Обов'яз- ковий			Так	Так	
Тариф	tarif	float	Обов'яз- ковий			Так	Так	
Оплачена сума	paid_am ount	float				Так	Так	
Повна вартість контракту	cost	float				Так	Так	

Таблиця 3.6. Структура таблиці "Оплата"

Назва атрибута	Ідентифікатор	Формат	Обов'язковий (необов'язковий)	Обмеження на право звертання	Первинний (вторинний) ключ	Виводимість значень	Дублювання значень, %	Індексне поле
№ Оплати	ID_Payment	int	Обов'язковий	Має право черговий	ПК	Так	Ні	
Дата та час оплати	Date_Payment	datetime	Обов'язковий			Так	Так	
Сума оплати	Amount_Payment	float	Обов'язковий			Так	Так	
№ Місяця	Password_Worker	int	Обов'язковий		ВК	Так	Так	ІНД

Таблиця 3.7. Структура таблиці "В'їзд/Виїзд"

Назва атрибута	Ідентифікатор	Формат	Обов'язковий (необов'язковий)	Обмеження на право звертання	Первинний (вторинний) ключ	Виводимість значень	Дублювання значень, %	Індексне поле
№ В'їзду/Виїзду	ID_InOutput	int	Обов'язковий	Має право черговий	ПК	Так	Ні	
Дата та час оплати	Date_InOutput	datetime	Обов'язковий			Так	Так	
В'їзд/Виїзд	InOut	bit	Обов'язковий			Так	Так	
№ Місяця	ID_InOutput_Car	varchar(12)	Обов'язковий		ВК	Так	Так	ІНД

Діаграма база даних представлена на рисунку 3.1.

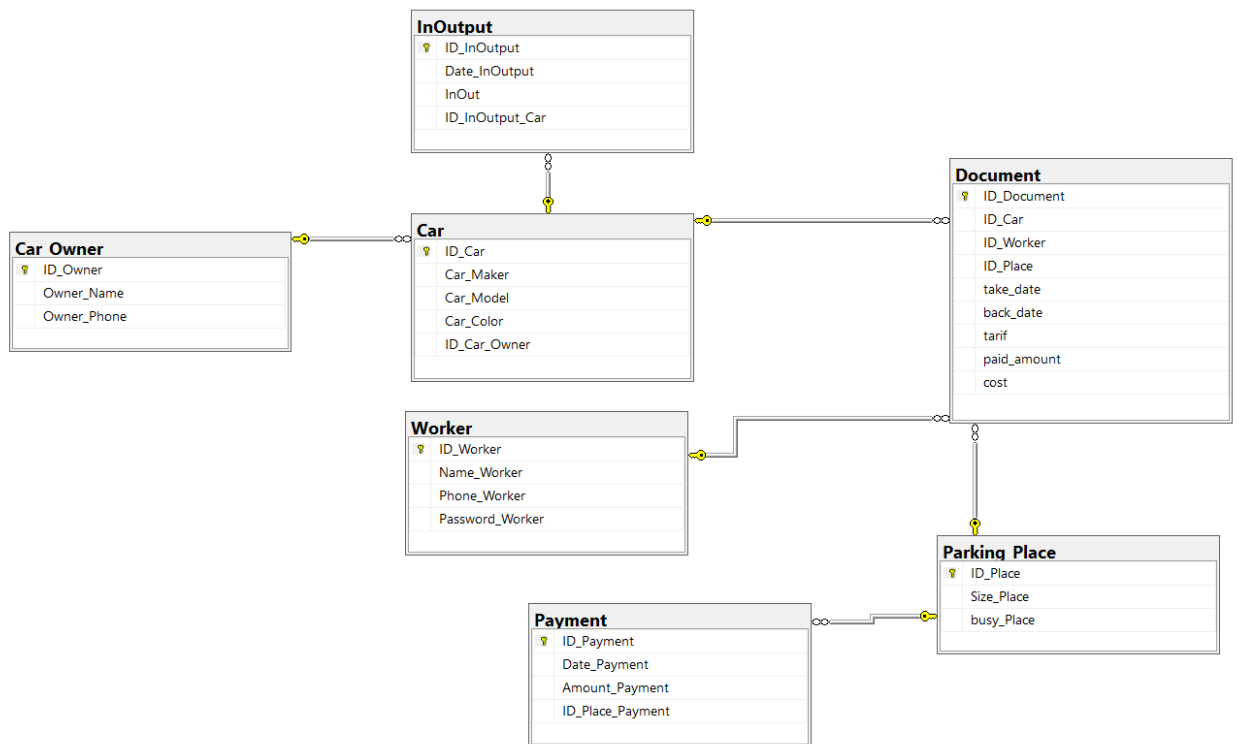


Рис. 3.1. Діаграма БД

Контрольні приклади таблиць БД представлені у додатку В на рисунках В.1 – В.7.

3.2. Технічне забезпечення

Загальні положення та схема автоматизації.

Інформаційна система підтримки роботи автомобільних стоянок буде функціонувати на комп'ютерах чергового та бухгалтера, які знаходяться на робочих місцях чергового та бухгалтера відповідн. Загальна схема автоматизації ІС показана на рисунку 3.2.

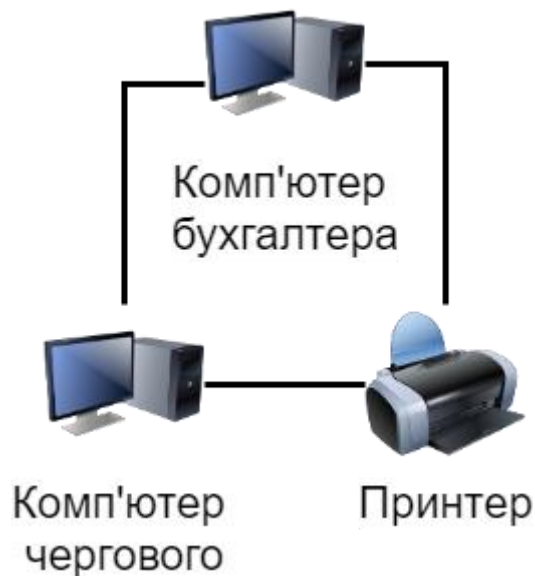


Рис. 3.2. Загальна схема автоматизації

До чергового комп'ютеру на якому знаходиться виконавчий файл ІС та БД, повинен бути підключений принтер для друкування контрактів та звітів та комп'ютер бухгалтера для доступу до БД. Також принтер повинен бути підключений до принтеру для друкування звітів.

Усією ІС може керувати 1 людина – черговий, а за потреби нею може скористуватися бухгалтер.

Опис автоматизованого робочого місця.

Запропонована ІС не є вимогливою щодо ресурсів комп'ютера, тому мінімально допустимі системні вимоги для її роботи такі:

- Центральний процесор – не менше за 1.5 ГГц;
- Оперативна пам'ять – не менше 2 Гб;
- Доступне місце на диску – не менше 100 Мб (не враховуючи місце, зарезервоване операційною системою);
- Відеоадаптер, який підтримує DirectX 9;

- Екран з роздільною здатністю 800*600, але рекомендується мати екран не менш ніж 1024*600 точок.

З периферійних засобів для функціонування ІС достатньо: монітору, миші, клавіатури та принтеру.

Інформаційна система не передає та не отримує дані до/з інших пристроїв тому Ethernet або інший тип підключення для функціонування ІС не потрібен.

3.3. Програмне забезпечення

Запропонована інформаційна система розроблена від операційну систему Windows 7 та вище. Тому для запуску та роботи ІС на комп'ютері повинна бути встановлена операційна система Windows 7 або вище. Для запуску виконавчого файлу системи потрібні такі встановлені пакети: Visual C++ Redistributable 2013 та .NET Framework версії 4.0, або більш нові їх версії. Для функціонування БД на комп'ютері повинен бути встановлений Microsoft SQL Server версії 2019 випуску Developer або Express.

Була розроблена діаграма компонентів, яка відображає залежності між компонентами програми, а саме екранними формами під час роботи програми. Діаграма компонентів зображена на рисунку 3.3.

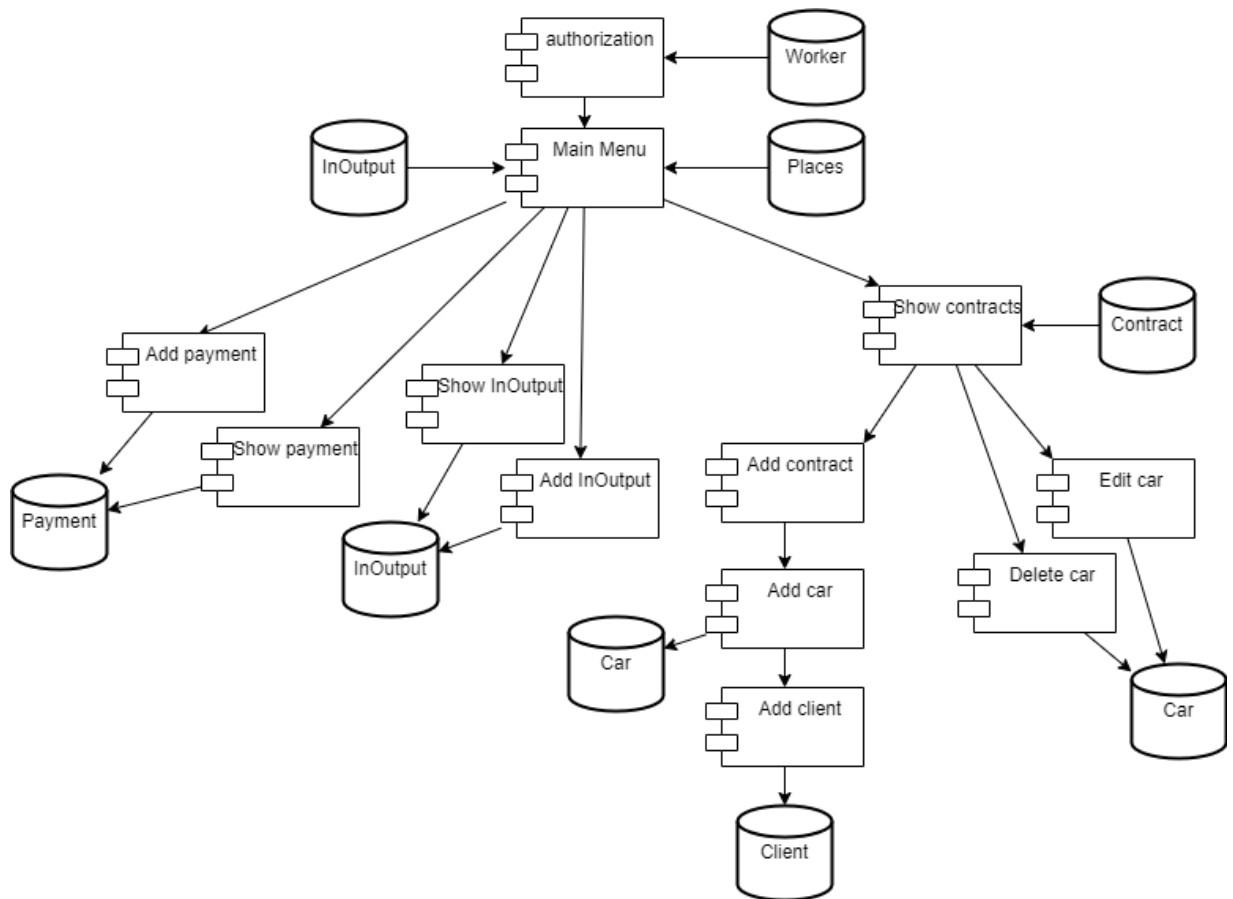


Рис. 3.3. Діаграма компонентів програми

Для роботи з запропонованою ІС достатньо лише встановити вищеповисані пакети для коректної роботи ІС та мати виконавчий файл.

3.4. Результати реалізації інформаційної системи/підсистеми

Екрані форми, які з'являються при роботі з ІС представлені у додатку А.

Форма авторизації містить поле для вводу паролю та список чергових автостоянки (додаток А, рисунок А.1). При успішному введенні паролю з'являється форма головного меню. Дана форма містить інформацію про останні оплати та місця на автостоянці (додаток А, рисунок А., 2). Також вона має багато кнопок, які відкривають інші екранні форми, такі як: форма додання в'їзду або виїзду, форма додання оплати за місцем, форма перегляду контрактів, форма перегляду оплат за місцем, форма перегляду в'їздів та виїздів.

Екранна форма додавання оплати за місцем містить поля для вводу номера місця та оплаченої суми, а також елемент для вибору дати та часу оплати і прапорець, що відповідає за друкування чеку (додаток А, З).

Екранна форма перегляду в'їздів та виїздів містить таблицю зафіксованих в'їздів та виїздів на територію автостоянки (додаток А, рисунок А.4).

Форма оплати за місцем містить таблицю, яка відображає усі оплати за обраним місцем (додаток А, рисунок А.5).

Екранна форма фіксування в'їздів та виїздів, яка зображена на рисунку А.6, містить лише поле для вводу номеру ТЗ.

Екранна форма контрактів містить таблицю з даними контрактів декілька елементів управління, які дозволяються показали, роздрукувати та вивести контракти за обраним періодом у файл (додаток А, рисунок А.7.). Також є можливість Перейти до форми створення нового контракту.

Форма додавання нового контракту містить поля для вводу даних (додаток А, рисунок А.8). Якщо ТЗ не був знайдений у БД користувач може перейти до форми занесення нового ТЗ до БД.

Форми занесення нового ТЗ до БД містить елементи для вводу даних (додаток А, рисунок А.9). Якщо власник не був знайдений у БД користувач може перейти до форми занесення нового власника до БД.

Екранна форма форми занесення нового власника до БД містить поля для вводу даних власника.

Висновок

Було виконано дослідження предметної галузі. Досліджені процеси роботи автостоянок, а саме обліку їх роботи. Також було досліджено структуру автомобільних стоянок, проаналізовано та порівняно ІС, які вже розроблені у даній предметній області.

Розроблені та змодельовані бізнес цілі, функціональні на нефункціональні вимоги до ІС. Було надано характеристику задачі та описано вихідну, вихідну інформацію, алгоритм розв'язання задачі розроблено інформаційну модель задачі та модель структури ІС. Також було наведено математичний опис алгоритму формування основних розрахункових показників.

Було надано структуру таблиць БД, описано інформаційне та технічне забезпечення, яке потрібне для реалізації запропонованої ІС. Та описані результати реалізації системи.

В ході виконання кваліфікаційного бакалаврського проєкту була розроблена інформаційна система для підтримки роботи автостоянок. Порівнюючи запропоновану систему з існуючими продуктами, які були проаналізовані у розділі 1.2, дана система має такі переваги. З її допомогою автостоянки, на яких ще не впроваджені ІС будуть мати можливість отримати ІС та працювати з нею.

Перевагами ІС є зменшення часу очікування для клієнтів автостоянки та фактично повне виключення паперової документації з роботи автостоянки.

Недоліком даної ІС є те, що для її впровадження на автостоянці повинен бути комп'ютер та принтер. Також при наявності проблем з подачею електроенергії ІС не зможе працювати на час відсутності електроенергії.

Дана система не є ідеальною, до неї можливо додати певні функції, наприклад аналітику. Також не всі автостоянки можуть облаштовувати ще

одне робоче місце для бухгалтера та підтримувати локальну обчислювальну мережу, так як це вимагає додаткових фінансових витрат.

Перелік використаних джерел

1. Програма «Avtojanka» – [Електронний ресурс]
http://www.alloyourprog.narod.ru/avtosojanka_v.htm
2. Програма «AllStojanka» – [Електронний ресурс]
<http://www.softplaneta.ru/programmy-dlya-windows/delovye-programmy/allstojanka-1-1.html>
3. Автоматизована система «АСКОП SEA» – [Електронний ресурс]
https://www.sea.com.ua/ua/parkovochnoe_oborudovanie/web-portal-askop-sea/
4. Характеристика SQL Server – [Електронний ресурс]
<https://metanit.com/sql/sqlserver/1.1.php>
5. Інформаційні системи і технології в економіці: навч. посіб. - М.А. Сендзюка, М.Б. Вітер К. :КНЕУ, 2011. – 422с.
6. Системи оброблення економічної інформації: навч. посіб. /за заг. ред. М.А. Сендзюка, М.І. Татарчука. – К. :КНЕУ, 2010. – 455с.
7. Ситник Н.В. Проектування баз і сховищ даних: Нав. посіб. – К.:2004. – 348 с.
8. Береза А.М., Козак І.А. Проектування систем оброблення інформації: Навч. посіб. – К.: КНЕУ, 2008. – 448 с.
9. Галузевий стандарт вищої освіти України з напрямку підготовки 6.050101 "Комп'ютерні науки" : збірник нормативних документів вищої освіти. – К.: Видавнича група BVH, 2011. – 85 с.
10. ISO/IEC/IEEE 29148:2018 Systems and software engineering - Life cycle processes - Requirements engineering.
11. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений на C++. Пер. с англ. – М.: СПб.: “Издательство БИНОМ”-“Невский Диалект”.

12. Глушаков С.В., Коваль А.В., Смирнов С.В. Язык программирования С++: Учебный курс.- Харьков: Фолио; М.: «Видавництво АСТ»,
13. Р. Лафоре. Объектно-ориентированное программирование в С++. 4-е издание. Издательство: Питер. Серия: Классика computer science,
14. Шилдт Г. Самоучитель С++, 3-е издание: пер. с англ.-СПб.:ВНУ-СанктПетербург,1999.
15. Петрик О., Петрик М. Технологія програмування в середовищі С++.– Тернопіль: ТДТУ ім. Ів. Пулюя, 2005.
16. Шилдт Г. Полный справочник по С++. 4-е издание. - М.: Изд. дом «Вильямс», 2003г.
17. Седжвик Р. Фундаментальные алгоритмы на С++. –К.: Изд-во «ДиаСофт», 2001.
18. Шилдт Г. Теория и практика С++: пер. с англ.-СПб.: ВНУ-СанктПетербург,1999.
19. Подбельский В.В. Язык С++.-М.: Финансы и статистика,1966.
20. Дейтл Н., Уимз Ч., Хедингтон М. Программирование на С++. Пер. С англ. - М.:ДМК,2000.

Додатки

Додаток А

Демонстрація результатів реалізації ІС

The image shows a software window titled 'Авторизація' (Authorization) with a dark blue header bar. The main area has a light yellow background. At the top center is the stylized logo 'Автостопанка'. Below it, the text 'Контролюючий' (Controlling) is followed by a dropdown menu showing 'Васильєв Андрій Георгійов' with a downward arrow. Underneath is the label 'Пароль' (Password) above a password input field. The field contains ten black dots and a vertical cursor line, with a small square icon to its right. At the bottom center is a grey button labeled 'Вхід' (Login).

Рис. А.1. Екрана форма авторизації

Головне меню

Вїзд

Виїзд

Місця

Додати оплату місця

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	

Останні оплати

06.03.2022 18:35:39 було оплачено місце №1 на суму 20 гривень.
23.11.2021 14:56:00 було оплачено місце №1 на суму 300 гривень.
21.10.2021 19:57:13 було оплачено місце №2 на суму 2000 гривень.
21.07.2021 7:10:55 було оплачено місце №1 на суму 1000 гривень.

Контролюючий: Васильєв Андрій Георгійович

Вихід до вікна авторизації

Переглянути контракти

Переглянути оплати за місцем

Переглянути усі вїзди ти виїзди

Рис. А.2. Екрана форма головного меню

Додавання оплати

Додавання оплати

Номер місця

Тариф

Дата початку оренди місця

Дата кінця оренди місця

Зафіксувати оплату

Повернутися назад

Рис. А.3. Форма оплати місця

В'їзди та виїзди

В'їзди та виїзди з території

	№ машини	Дата	В'їзд або виїзд
	AA1111AA	03.06.2022 20:24:40	В'їзд
	AA1111AA	03.06.2022 20:30:17	Виїзд
	AA1111AA	03.06.2022 20:36:15	В'їзд
»*			

Повернутися

Рис. А.4. Форма в'їздів та виїздів

Плати за місцем №1

	№ оплати	Дата	Сума оплати
	4	06.03.2022 18:35:39	20
	3	23.11.2021 14:56:00	300
	1	21.07.2021 7:10:55	1000
»»			

Повернутися

Рис. А.5 Форма виводу оплат за місцем

Додавання виїзду

Номер ТЗ

AA4124BG

Зафіксувати

Повернутися

Рис. А.6. Екранна форма фіксування в'їздів та виїздів

Контракти

	№	Дата початку	Дата кінця	Тариф	Сплачено	Сума по контракту	№ місця	Власник ТЗ	№ ТЗ
	14	03.06.2022 19:36:00	04.06.2022 19:36:00	123	0	123	10	Комаров Олександр ...	AA1111AA
	13	06.03.2022 19:31:00	08.03.2022 19:31:00	123	0	7503	10	Комаров Олександр ...	123123
	11	06.03.2022 18:26:00	06.05.2022 18:26:00	20	0	40	5	Сафонов Данило Євг...	AA1111AA
	10	21.07.2021 7:11:00	21.12.2021 7:11:00	20	2000	2000	10	Гришина Анастасія С...	AA7777AA
	9	28.04.2021 11:35:00	28.09.2021 11:35:00	30	1000	1200	9	Синіцина Аріна Степа...	AA6666AA
	8	21.10.2021 19:57:00	21.11.2021 19:57:00	30	800	800	8	Максимова Єлизаве...	AA5555AA
	7	12.07.2021 8:55:00	12.09.2021 8:55:00	25	3200	3400	7	Максимова Єлизаве...	AA4444AA
	6	22.05.2021 20:26:00	22.06.2021 20:26:00	40	1200	1230	6	Максимова Єлизаве...	AA2222AA
	5	23.07.2021 14:56:00	23.11.2021 14:56:00	10	4000	4000	5	Максимова Єлизаве...	BB1111BB
	4	16.08.2021 19:57:00	16.09.2021 19:57:00	12	0	1293	4	Васильєв Федір Арт...	AA8888AA
	3	26.04.2021 2:23:00	26.07.2021 2:23:00	25	3000	3000	3	Белова Мілана Іллініч...	BB2222BB

Період з 4 люня 2022 г. по 4 люня 2022 г.

Показати контракти за періодом

Роздрукувати контракти за періодом

Вивести контракти за періодом в Excel

Додати новий контракт

Повернутися назад

Рис. А.7. Екранна форма контрактів

InOutput

Додавання нового контракту

Номер ТЗ

AI3421DS

Номер місця

23

Тариф

30

Дата початку контракту

4 люня 2022 г.

Дата кінця контракту

4 мая 2023 г.

Зафіксувати

Повернутися

Рис. А.8. Екранна форма додавання нових контрактів

Додавання машини

Номер машини*

AI3421DS

Виробник машини

BMW

Модель машини

I3

Колір машини

Чорний

ПІБ Власника*

Васильєв Іван Юрійович

* - обов'язково до заповнення

Закріпити

Повернутися

Рис. А.9. Екранна форма додавання нових ТЗ

Вїзд

Ім'я клієнта

Васильєв Іван Юрійович

Телефон клієнта

+380112223344

Закріпити

Повернутися

Рис. А.10. Екранна форма додавання нових власників

Додаток Б
Код програми

Login_form.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 System.IO;
using System.Data.SqlClient;

namespace Avtostoyanka
{
    public partial class login_form : Form
    {
        private List<string> passwords = new List<string>();
        public login_form()
        {
            InitializeComponent();
        }

        private void login_button_Click(object sender, EventArgs e)
        {
            //тут проверка на правильность логина и пароля
            if (login_comboBox1.SelectedIndex < 0 ||
!passwords[login_comboBox1.SelectedIndex].Equals(password_textBox1.Text))
            {
                MessageBox.Show("Неправильный пароль.");
                return;
            }

            //имя берётся с логина
            string controller_name = login_comboBox1.SelectedItem.ToString();

            password_textBox1.Text = string.Empty;

            func.controller_id = login_comboBox1.SelectedIndex + 1;

            //открытие следующей формы
            main_form n_form = new main_form(controller_name, this);
            n_form.Show();
            this.Hide();
        }

        private void login_form_Load(object sender, EventArgs e)
        {
            //берём имя и пароли дежурных
            using (SqlConnection mySqlConnection = new SqlConnection("Data Source=(local);"
+ "Initial Catalog=DB_AVTOSTOYANKA;" + "Integrated Security=True"))
            {
                try
                {
                    mySqlConnection.Open();
                    SqlCommand MySQL_Command = new SqlCommand("SELECT * FROM [Worker]",
mySqlConnection);

                    using (SqlDataReader reader = MySQL_Command.ExecuteReader())
                    {
                        while (reader.Read())
                        {

```

```

        login_comboBox1.Items.Add(reader["Name_Worker"].ToString());
        passwords.Add(reader["Password_Worker"].ToString());
    }
}
}
catch (Exception ex)
{
    //MessageBox.Show(ex.ToString());
    MessageBox.Show("Проблеми з підключенням до бази даних. Перевірте чи
увімкнені SQL служби.");
}
}
}

e) private void password_visible_checkBox1_CheckStateChanged(object sender, EventArgs
{
    if (password_textBox1.UseSystemPasswordChar)
        password_textBox1.UseSystemPasswordChar = false;
    else
        password_textBox1.UseSystemPasswordChar = true;
}

private void button1_Click(object sender, EventArgs e)
{
    //char a = (char) 65;
    //MessageBox.Show(Convert.ToChar(Convert.ToInt32(a)+1).ToString());
    string b = func.getDate(DateTime.Now);
    MessageBox.Show(b);
}
}
}
}

```

Main_form.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 System.Data.SqlClient;

namespace Avtostoyanka
{
    public partial class main_form : Form
    {
        data_base db = data_base.getDB();
        string controller_name;
        List<int> places_id = new List<int>();
        List<bool> places_busy = new List<bool>();
        login_form back_form;

        public main_form(string c_name, login_form b)
        {
            back_form = b;
            InitializeComponent();

            controller_name = c_name;
        }
    }
}

```

```

private void main_form_Load(object sender, EventArgs e)
{
    //берём номера мест и занятость
    using (SqlConnection mySqlConnection = new SqlConnection("Data Source=(local);"
+ "Initial Catalog=DB_AVTOSTOYANKA;" + "Integrated Security=True"))
    {
        try
        {
            mySqlConnection.Open();
            SqlCommand MySQL_Command = new SqlCommand("SELECT * FROM
[Parking_Place]", mySqlConnection);

            using (SqlDataReader reader = MySQL_Command.ExecuteReader())
            {
                while (reader.Read())
                {
                    places_id.Add(Convert.ToInt32(reader["ID_Place"]));
                    places_busy.Add(Convert.ToBoolean(reader["busy_Place"]));
                }
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.ToString());
        }
    }

    //добавляем на экран имя дежурного
    controller_name_label1.Text += controller_name;

    //добавляем колонки
    for (int i = 0; i <= 9; i++)
    {
        DataGridViewColumn t_collum = new DataGridViewColumn
        {
            Name = (i + 1).ToString(),
            Width = 50,
            //чтобы не выбивало ошибку
            CellTemplate = new DataGridViewTextBoxCell()
        };
        places_dataGridView1.Columns.Add(t_collum);
    }
    places_dataGridView1.Rows.Add();

    //переименования для просчёта айди места, которое мы заполняем

    int temp_place_id = 0;
    //высчитываем нужное количество строк
    int ceiling;
    if (places_id.Count % 10 == 0)
        ceiling = Convert.ToInt32(Math.Ceiling(Convert.ToDouble(places_id.Count /
10)) - 1);
    else
        ceiling = Convert.ToInt32(Math.Ceiling(Convert.ToDouble(places_id.Count /
10)));

    //добавляем и заполняем столбцы
    for (int i = 0; i <= ceiling; i++)
    {
        DataGridViewRow t_row;
        //временное решение (наверное :)
        if (i == 0)
        {

```

```

        //создаём строчку
        t_row = (DataGridViewRow)places_dataGridView1.Rows[0].Clone();
        places_dataGridView1.Rows.RemoveAt(0);
    }
    else
        t_row = (DataGridViewRow)places_dataGridView1.Rows[0].Clone();

    for (int j = 0; j <= 9; j++)
    {
        //значение ячейки
        t_row.Cells[j].Value = places_id[temp_place_id].ToString();
        if(places_busy[temp_place_id])
            t_row.Cells[j].Style.BackColor = Color.OrangeRed;

        if (!(temp_place_id + 1 >= places_id.Count))
            temp_place_id++;
        else
            break;
    }
    places_dataGridView1.Rows.Add(t_row);
}
//прячем скролбары
//places_dataGridView1.ScrollBars = ScrollBars.None;

richTextBox1.Text = string.Empty;
List<payment> pays = new List<payment>();

pays = db.get_payment();
for (int i = 0; i < 5 && i < pays.Count; i++)
{
    richTextBox1.Text += pays[i].date + " було оплачено місце №";
    richTextBox1.Text += pays[i].id_place + " на суму ";
    richTextBox1.Text += pays[i].sum + " гривень." + Environment.NewLine;
}
//убираем столбцы и строчки элемента управления
places_dataGridView1.ColumnHeadersVisible = false;
places_dataGridView1.RowHeadersVisible = false;
}

private void main_form_FormClosed(object sender, FormClosedEventArgs e)
{
    //Application.Exit();
}

private void in_car_button1_Click(object sender, EventArgs e)
{
    add_InOutput newform = new add_InOutput(this, true);
    newform.Show();
    this.Hide();
}

private void out_car_button1_Click(object sender, EventArgs e)
{
    add_InOutput newform = new add_InOutput(this, false);
    newform.Show();
    this.Hide();
}

private void relogin_button2_Click(object sender, EventArgs e)
{
    back_form.Show();
}

```

```

        this.Close();
    }

    private void login_button_Click(object sender, EventArgs e)
    {
        //richTextBox1.Text = string.Empty;
        List<payment> pays = new List<payment>();
        pays =
        db.get_payment_by_place(Convert.ToInt32(places_dataGridView1.CurrentCell.Value));

        unusual_forms.payments_by_place ioform = new
        unusual_forms.payments_by_place(this, pays,
            Convert.ToInt32(places_dataGridView1.CurrentCell.Value));

        this.Hide();
        ioform.Show();
        /*
        for (int i = 0; i < pays.Count; i++)
        {
            richTextBox1.Text += pays[i].date + " було оплачено місце №";
            richTextBox1.Text += pays[i].id_place + " на суму ";
            richTextBox1.Text += pays[i].sum + " гривень." + Environment.NewLine;
        }
        */
    }

    private void show_contracts_Click(object sender, EventArgs e)
    {
        contracts nform = new contracts(this);
        nform.Show();
        this.Hide();
    }

    private void inout_show_button_Click(object sender, EventArgs e)
    {
        unusual_forms.in_out_show nform = new unusual_forms.in_out_show(this);
        this.Hide();
        nform.Show();
    }

    private void payment_button_Click(object sender, EventArgs e)
    {
        add_payment aform = new add_payment(this);
        this.Hide();
        aform.Show();
    }
}
}
}

```

Func.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Avtostoyanka
{
    class func
    {
        public static int controller_id;
        public static string getDate(DateTime a)
    }
}

```

```

    {
        //2021-23-07 14:56:00
        string date = string.Empty;

        date += a.Year + "-";
        date += a.Day + "-";
        date += a.Month + " ";
        date += a.Hour + ":";
        date += a.Minute + ":";
        date += a.Second;

        return date;
    }
    public static string date_to_start_period(DateTime a)
    {
        //меняем время на 00:00:00
        TimeSpan t = new TimeSpan(0, 0, 0);
        a = a.Date + t;
        return getDate(a);
    }
    public static string date_to_end_period(DateTime a)
    {
        //меняем время на 00:00:00
        TimeSpan t = new TimeSpan(23, 59, 59);
        a = a.Date + t;
        return getDate(a);
    }
}
}
}

```

Data_base.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace Avtostoyanka
{
    public struct payment
    {
        public int id;
        public string date;
        public double sum;
        public int id_place;
    }
    public struct doc
    {
        public int id;
        public string car_id;
        public int worker_id;
        public int place_id;
        public string start;
        public string end;
        public double tarif;
        public double paid_amount;
        public double cost;
    };
    class data_base
    {
        private static data_base db;
    }
}

```

```

public SqlConnection data_base_SQL_Connection;

public static data_base getDB()
{
    if (db == null)
    {
        db = new data_base();
    }
    return db;
}

public void open_sql_connection()
{
    try
    {
        this.data_base_SQL_Connection = new SqlConnection("Data Source=(local);" +
            "Initial Catalog=DB_AVTOSTOYANKA;" +
            "Integrated Security=True");
        this.data_base_SQL_Connection.Open();
    }
    catch (Exception ex)
    {
        MessageBox.Show("Невдала спроба підключення до бази даних." +
            Environment.NewLine
            + ex.Message);
    }
}

public void close_sql_connection()
{
    try
    {
        this.data_base_SQL_Connection.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show("Невдала спроба відключення від бази даних." +
            Environment.NewLine
            + ex.Message);
    }
}

public void add_universal(string table_name, List<string> values)
{
    this.open_sql_connection();
    string command_text = string.Empty;
    //65 это A, 66 это B...
    char command_char = (char) 65;
    //названия переменных для вставки значений в СКЛ команду
    List<string> value_names = new List<string>();

    try
    {
        for (int i = 0; i < values.Count; i++)
        {
            //генерация названия переменной
            char temp = Convert.ToChar(Convert.ToInt32(command_char) + i);
            value_names.Add(temp.ToString());

            command_text += "@";
            command_text += temp.ToString();
        }
    }
}

```

```

        if ((i + 1) != values.Count)
            command_text += ",";
    }

    SqlCommand myCommand = new SqlCommand("INSERT INTO " + table_name
        + " VALUES("
        + command_text
        + ")"
        , this.data_base_SQL_Connection);

    for (int i = 0; i < values.Count; i++)
        myCommand.Parameters.AddWithValue(value_names[i], values[i]);

    myCommand.ExecuteNonQuery();
}
catch (Exception ex)
{
    MessageBox.Show("Помилка додавання даних!\n" + ex.Message, "Додавання
даних", MessageBoxButtons.OK, MessageBoxIcon.Error);
}
finally {
    this.close_sql_connection();
}
MessageBox.Show("Успішне додавання даних.");
}

public List<payment> get_payment()
{
    List<payment> pays = new List<payment>();

    this.open_sql_connection();
    //select* from tbl_name order by id desc limit N

    SqlCommand MySQL_Command = new SqlCommand("SELECT * FROM [payment] order by
ID_Payment desc", this.data_base_SQL_Connection);
    SqlDataReader mySqlReader = null;
    try
    {
        mySqlReader = MySQL_Command.ExecuteReader();
        while (mySqlReader.Read())
        {
            payment pay = new payment();
            pay.id = Convert.ToInt32(mySqlReader["ID_Payment"]);
            pay.date = mySqlReader["Date_Payment"].ToString();
            pay.sum = Convert.ToDouble(mySqlReader["Amount_Payment"]);
            pay.id_place = Convert.ToInt32(mySqlReader["ID_Place_Payment"]);

            pays.Add(pay);
        }
    }
    catch (Exception ex)
    {
        // вивод помилки
        MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
        MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    finally
    {
        if (mySqlReader != null)
            mySqlReader.Close();
    }
}

```

```

    }

    this.close_sql_connection();
    return pays;
}

public List<payment> get_payment_by_place(int place_id)
{
    List<payment> pays = new List<payment>();

    this.open_sql_connection();

    SqlCommand MySQL_Command = new SqlCommand("SELECT * FROM [payment] WHERE
ID_Place_Payment = @p1 order by ID_Payment desc ", this.data_base_SQL_Connection);
    SqlDataReader mySqlReader = null;
    try
    {
        MySQL_Command.Parameters.AddWithValue("p1", place_id);
        mySqlReader = MySQL_Command.ExecuteReader();
        while (mySqlReader.Read())
        {
            payment pay = new payment();
            pay.id = Convert.ToInt32(mySqlReader["ID_Payment"]);
            pay.date = mySqlReader["Date_Payment"].ToString();
            pay.sum = Convert.ToDouble(mySqlReader["Amount_Payment"]);
            pay.id_place = Convert.ToInt32(mySqlReader["ID_Place_Payment"]);

            pays.Add(pay);
        }
    }
    catch (Exception ex)
    {
        // вывод ошибки
        MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
        MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    finally
    {
        if (mySqlReader != null)
            mySqlReader.Close();
    }

    this.close_sql_connection();
    return pays;
}

public List<doc> get_contracts_by_period(DateTime start, DateTime end)
{
    List<doc> contracts = new List<doc>();

    this.open_sql_connection();

    SqlCommand MySQL_Command = new SqlCommand
        ("SELECT * FROM [Document] WHERE take_date >= @td AND back_date <= @bd
order by ID_Document desc "
        , this.data_base_SQL_Connection);
    SqlDataReader mySqlReader = null;
    try

```

```

        {
            MySQL_Command.Parameters.AddWithValue("td",
func.date_to_start_period(start));
            MySQL_Command.Parameters.AddWithValue("bd", func.date_to_end_period(end));

            mySqlReader = MySQL_Command.ExecuteReader();
            while (mySqlReader.Read())
            {
                doc d = new doc();
                d.id = Convert.ToInt32(mySqlReader["ID_Document"]);
                d.car_id = mySqlReader["ID_Car"].ToString();
                d.worker_id = Convert.ToInt32(mySqlReader["ID_Worker"]);
                d.place_id = Convert.ToInt32(mySqlReader["ID_Place"]);
                d.start = mySqlReader["take_date"].ToString();
                d.end = mySqlReader["back_date"].ToString();
                d.tarif = Convert.ToDouble(mySqlReader["tarif"]);
                d.paid_amount = Convert.ToDouble(mySqlReader["paid_amount"]);
                d.cost = Convert.ToDouble(mySqlReader["cost"]);

                contracts.Add(d);
            }
        }
        catch (Exception ex)
        {
            // вывод ошибки
            if (mySqlReader != null)
                mySqlReader.Close();
            MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
        finally
        {
            if (mySqlReader != null)
                mySqlReader.Close();
            this.close_sql_connection();
        }

        return contracts;
    }
}
}
}

```

Contracts.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 System.Reflection;
using Excel = Microsoft.Office.Interop.Excel;
using System.Data.SqlClient;
using System.Threading;

namespace Avtostoyanka
{
    public partial class contracts : Form
    {
        main_form back_form;
    }
}

```

```

data_base db = data_base.getDB();
public contracts(main_form bform)
{
    back_form = bform;
    InitializeComponent();
}

private void fill_contracts(List<doc> contracts)
{
    foreach (doc d in contracts)
    {
        int row_id = dataGridView1.Rows.Add();

        DataGridViewRow row = dataGridView1.Rows[row_id];
        row.Cells[0].Value = d.id;
        row.Cells[1].Value = d.start;
        row.Cells[2].Value = d.end;
        row.Cells[3].Value = d.tarif;
        row.Cells[4].Value = d.paid_amount;
        row.Cells[5].Value = d.cost;
        row.Cells[6].Value = d.place_id;
        row.Cells[8].Value = d.car_id;
    }
}

struct tz {
    public string car;
    public int owner;
}

private void fill_owners()
{
    //мне супер не нравится эта часть, да и эта функция в принципе
    List<int> owners = new List<int>();
    List<tz> tzs = new List<tz>();
    List<string> cars = new List<string>();

    List<string> ready_owners = new List<string>();
    foreach (DataGridViewRow row in dataGridView1.Rows)
        if(row.Cells[1].Value!=null)
            cars.Add(row.Cells[8].Value.ToString());

    db.open_sql_connection();

    string temp = string.Empty;

    SqlCommand MySQL_Command = new SqlCommand("SELECT * FROM [Car] ",
db.data_base_SQL_Connection);
    SqlDataReader mySqlReader = null;
    try
    {
        mySqlReader = MySQL_Command.ExecuteReader();
        while (mySqlReader.Read())
        {
            tz t;
            t.car = mySqlReader["ID_Car"].ToString();
            foreach(string c in cars)
                if (c.Equals(t.car))
                {
                    t.owner = Convert.ToInt32(mySqlReader["ID_Car_Owner"]);
                    tzs.Add(t);
                }
            }
        }
    }

```

```

        }
    }
}
catch (Exception ex)
{
    // вывод ошибки
    MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
        MessageBoxButtons.OK, MessageBoxIcon.Error);
}
finally
{
    if (mySqlReader != null)
        mySqlReader.Close();
}

//2:25 ночи. мне уже всё равно, что можно сделать умнее и оптимальнее

foreach (tz a in tzs)
    owners.Add(a.owner);

MySQL_Command = new SqlCommand("SELECT * FROM [Car_Owner] ",
db.data_base_SQL_Connection);
mySqlReader = null;
try
{
    mySqlReader = MySQL_Command.ExecuteReader();
    while (mySqlReader.Read())
    {
        int i = Convert.ToInt32(mySqlReader["ID_Owner"]);
        foreach (int o in owners)
            if (o.Equals(i))
            {
                ready_owners.Add(mySqlReader["Owner_Name"].ToString());
            }
    }
}
catch (Exception ex)
{
    // вывод ошибки
    MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
        MessageBoxButtons.OK, MessageBoxIcon.Error);
}
finally
{
    if (mySqlReader != null)
        mySqlReader.Close();
}

db.close_sql_connection();
int j = 0;
foreach (DataGridViewRow row in dataGridView1.Rows)
{
    if (row.Cells[1].Value != null)
    {
        row.Cells[7].Value = ready_owners[j];
        j++;
    }
    else break;
}
}
}

```

```

private void contracts_Load(object sender, EventArgs e)
{
    DateTime start = (DateTime.Now).AddYears(-20);
    DateTime end = (DateTime.Now).AddYears(+20);
    fill_contacts(db.get_contracts_by_period(start, end));
    //Thread.Sleep(1000);
    fill_owners();

    dataGridView1.ReadOnly = true;
    end_dateTimePicker2.Value.AddDays(1);
}

private void button4_Click(object sender, EventArgs e)
{
    this.Close();
    back_form.Show();
}

private void show_period_button_Click(object sender, EventArgs e)
{
    dataGridView1.Rows.Clear();
    fill_contacts(db.get_contracts_by_period(start_dateTimePicker1.Value,
end_dateTimePicker2.Value));
    //Thread.Sleep(1000);
    fill_owners();
}

private void add_contract_button3_Click(object sender, EventArgs e)
{
    add_contract nform = new add_contract(this);
    this.Hide();
    nform.Show();
}

private void exel_period_button2_Click(object sender, EventArgs e)
{
    /*
        SaveFileDialog sfd = new SaveFileDialog();
        DialogResult dr = sfd.ShowDialog();
        if (dr == DialogResult.OK)
        {
            Excel._Application ExcelApp = new Excel.Application();
            ExcelApp.Visible = true;
            if (ExcelApp == null)
            {
                MessageBox.Show("Excel is not properly installed!!");
                return;
            }
            Excel._Workbook workbook =
(Excel._Workbook)(ExcelApp.Workbooks.Add(Missing.Value));

            Excel.Worksheet worksheet = workbook.ActiveSheet;

            for (int i = 1; i < dataGridView1.RowCount * 1; i++)
            {
                for (int j = 1; j < dataGridView1.ColumnCount * 1; j++)
                {
                    worksheet.Rows[i].Columns[j] = dataGridView1.Rows[i -
1].Cells[j - 1].Value;
                }
            }
        }
    */
}

```

```

        ExcelApp.AlertBeforeOverwriting = false;
        workbook.SaveAs(sfd.FileName + ".xls");
        ExcelApp.Quit();
        MessageBox.Show("Records Successfully Exported");
    }
    else if (dr != DialogResult.OK)
    {
        MessageBox.Show("Not Exported");
    }
}
*/
}
}
}

```

Add_payment.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 Avtostoyanka
{
    public partial class add_payment : Form
    {
        main_form back_form;
        public add_payment(main_form b)
        {
            back_form = b;
            InitializeComponent();
        }

        private void add_payment_Load(object sender, EventArgs e)
        {
            checkBox1.Checked = true;
        }

        private void back_button4_Click(object sender, EventArgs e)
        {
            back_form.Show();
            this.Close();
        }

        private void enter_button_Click(object sender, EventArgs e)
        {
            List<string> val = new List<string>();

            val.Add(func.getDate(dateTimePicker1.Value).ToString());
            val.Add(paid_amout_textBox1.Text);
            val.Add(place_id_textBox1.Text);

            data_base db = data_base.getDB();
            if (place_id_textBox1.Text.Length > 0 && paid_amout_textBox1.Text.Length > 0)
                db.add_universal("Payment", val);
            else
                MessageBox.Show("Заповніть усі поля!");
        }
    }
}

```

Add_owner.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 System.Data.SqlClient;

namespace Avtostoyanka
{
    public partial class add_owner : Form
    {
        data_base my_db = data_base.getDB();
        add_car back_form;
        string owner_name;
        public add_owner(add_car mf, string name)
        {
            back_form = mf;
            owner_name = name;
            InitializeComponent();
        }

        private void InOutput_Load(object sender, EventArgs e)
        {
            name_textBox1.Text = owner_name;
        }

        private void enter_button_Click(object sender, EventArgs e)
        {
            List<string> values = new List<string>();

            if (name_textBox1.Text.Length > 0 && phone_textBox1.Text.Length > 0)
            {
                values.Add(name_textBox1.Text);
                values.Add(phone_textBox1.Text);
            }
            else
            {
                MessageBox.Show("Заповніть усі поля.");
                my_db.add_universal("Car_Owner", values);
            }
        }

        private void back_button_Click(object sender, EventArgs e)
        {
            back_form.Show();
            this.Close();
        }
    }
}
```

Add_inouts.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 System.Data.SqlClient;

namespace Avtostoyanka
{
    public partial class add_InOutput : Form
    {
        //0 out 1 in
        bool inout;
        data_base my_db = data_base.getDB();
        main_form back_form;
        public add_InOutput(main_form mf, bool iot)
        {
            back_form = mf;
            inout = iot;
            InitializeComponent();
        }

        private void InOutput_Load(object sender, EventArgs e)
        {
            if (inout)
            {
                inoutput_label1.Text = "Додавання в'їзду";
                this.Text = "Додавання в'їзду";
            }
            else
            {
                inoutput_label1.Text = "Додавання виїзду";
                this.Text = "Додавання виїзду";
            }
        }

        private void enter_button_Click(object sender, EventArgs e)
        {
            List<string> values = new List<string>();
            string date = func.getDate(DateTime.Now);

            values.Add(date);
            values.Add(inout.ToString());
            values.Add(car_id_textBox1.Text);

            my_db.add_universal("InOutput", values);
        }

        private void back_button_Click(object sender, EventArgs e)
        {
            back_form.Show();
            this.Close();
        }
    }
}

```

Add_contracts.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 System.Data.SqlClient;

```

```

namespace Avtostoyanka
{
    public partial class add_contract : Form
    {
        data_base my_db = data_base.getDB();
        contracts back_form;
        public add_contract(contracts mf)
        {
            back_form = mf;
            InitializeComponent();
        }

        private void InOutput_Load(object sender, EventArgs e)
        {
            end_dateTimePicker2.Value.AddDays(1);
        }

        private void enter_button_Click(object sender, EventArgs e)
        {
            List<string> values = new List<string>();

            my_db.open_sql_connection();

            string temp = string.Empty;

            SqlCommand MySQL_Command = new SqlCommand("SELECT * FROM [Car] where ID_Car =
@ic", my_db.data_base_SQL_Connection);
            SqlDataReader mySqlReader = null;
            try{
                MySQL_Command.Parameters.AddWithValue("ic", car_id_textBox1.Text);
                mySqlReader = MySQL_Command.ExecuteReader();
                while (mySqlReader.Read())
                {
                    temp = mySqlReader["ID_Car"].ToString();
                }
            }
            catch (Exception ex){
                // Вывод ошибки
                MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
                MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            finally{
                if (mySqlReader != null)
                    mySqlReader.Close();
                my_db.close_sql_connection();
            }

            if (temp.Length > 0)
            {
                values.Add(car_id_textBox1.Text);
                values.Add(func.controller_id.ToString());
                values.Add(place_id_textBox1.Text);
                values.Add(func.getDate(start_dateTimePicker1.Value));
                if (start_dateTimePicker1.Value.CompareTo(end_dateTimePicker2.Value) < 0)
                    values.Add(func.getDate(end_dateTimePicker2.Value));
                else
                    MessageBox.Show("Дата кінця контракту не может бути меншою за дату
початку.");
                values.Add(tarif_textBox1.Text);
            }
        }
    }
}

```

```

        //разница в днях
        System.TimeSpan diff =
end_dateTimePicker2.Value.Subtract(start_dateTimePicker1.Value);

        //Оплаченная сумма в данный момент
        values.Add("0");

        values.Add((Convert.ToInt32(tarif_textBox1.Text) * (diff.Days +
1)).ToString());

        my_db.add_universal("Document", values);
        MessageBox.Show("Успішне додавання даних.");
    }
    else
    {
        DialogResult dialogResult = MessageBox.Show("ТЗ немає у БД. Додати його?",
" ", MessageBoxButtons.YesNo);
        if (dialogResult == DialogResult.Yes)
        {
            add_car nform = new add_car(this, car_id_textBox1.Text);
            this.Hide();
            nform.Show();
        }
    }
}

private void back_button_Click(object sender, EventArgs e)
{
    back_form.Show();
    this.Close();
}

private void label1_Click(object sender, EventArgs e)
{
}
}
}

```

Add_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 System.Data.SqlClient;

namespace Avtostoyanka
{
    public partial class add_car : Form
    {
        data_base my_db = data_base.getDB();
        add_contract back_form;
        string car_id;
        public add_car(add_contract mf, string car)
        {
            back_form = mf;
            car_id = car;
            InitializeComponent();
        }
    }
}

```

```

    }

    private void InOutput_Load(object sender, EventArgs e)
    {
        car_id_textBox1.Text = car_id;
    }

    private void enter_button_Click(object sender, EventArgs e)
    {
        List<string> values = new List<string>();

        values.Add(car_id_textBox1.Text);
        if (maker_textBox1.Text.Length > 0)
            values.Add(maker_textBox1.Text);
        else
            values.Add("none");
        if (model_textBox2.Text.Length > 0)
            values.Add(model_textBox2.Text);
        else
            values.Add("none");
        if (color_textBox3.Text.Length > 0)
            values.Add(color_textBox3.Text);
        else
            values.Add("none");

        my_db.open_sql_connection();

        string id_owner = string.Empty;

        SqlCommand MySQL_Command = new SqlCommand("SELECT * FROM [Car_Owner] where
Owner_Name = @ic", my_db.data_base_SQL_Connection);
        SqlDataReader mySqlReader = null;
        try
        {
            MySQL_Command.Parameters.AddWithValue("ic", owner_name_textBox4.Text);
            mySqlReader = MySQL_Command.ExecuteReader();
            while (mySqlReader.Read())
            {
                id_owner = mySqlReader["ID_Owner"].ToString();
            }
        }
        catch (Exception ex)
        {
            // вывод ошибки
            MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
            MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
        finally
        {
            if (mySqlReader != null)
                mySqlReader.Close();
            my_db.close_sql_connection();
        }

        if (id_owner.Length > 0)
        {
            values.Add(id_owner);
            my_db.add_universal("Car", values);
        }
        else
        {

```

```

        DialogResult dialogResult = MessageBox.Show("Клієнта немає у БД. Додати  
його?", " ", MessageBoxButtons.YesNo);
        if (dialogResult == DialogResult.Yes)
        {
            add_owner nform = new add_owner(this, owner_name_textBox4.Text);
            this.Hide();
            nform.Show();
        }
    }

    private void back_button_Click(object sender, EventArgs e)
    {
        back_form.Show();
        this.Close();
    }
}

```

Payment_by_place.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 Avtostoyanka.unusual_forms
{
    public partial class payments_by_place : Form
    {
        List<payment> pays;
        main_form back_form;
        int place_id;
        public payments_by_place(main_form bform, List<payment> p, int p_id)
        {
            pays = p;
            back_form = bform;
            place_id = p_id;
            InitializeComponent();

            this.Text += place_id.ToString();
            title_label1.Text += place_id.ToString();
        }

        private void InOutputShow_Load(object sender, EventArgs e)
        {
            foreach (payment a in pays)
            {
                int row_id = dataGridView1.Rows.Add();

                DataGridViewRow row = dataGridView1.Rows[row_id];
                row.Cells[0].Value = a.id;
                row.Cells[1].Value = a.date;
                row.Cells[2].Value = a.sum;
            }
        }
    }
}

```

```

        private void back_button_Click(object sender, EventArgs e)
        {
            back_form.Show();
            this.Close();
        }
    }
}

```

In_outs_show.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 System.Data.SqlClient;

namespace Avtostoyanka.unusual_forms
{
    public partial class in_out_show : Form
    {
        data_base db = data_base.getDB();
        main_form back_form;
        public in_out_show(main_form bform)
        {
            back_form = bform;
            InitializeComponent();

            private void InOutputShow_Load(object sender, EventArgs e)
            {
                db.open_sql_connection();

                string temp = string.Empty;

                SqlCommand MySQL_Command = new SqlCommand("SELECT * FROM [InOutput] ",
db.data_base_SQL_Connection);
                SqlDataReader mySqlReader = null;
                try
                {
                    mySqlReader = MySQL_Command.ExecuteReader();
                    while (mySqlReader.Read())
                    {
                        int row_id = dataGridView1.Rows.Add();

                        DataGridViewRow row = dataGridView1.Rows[row_id];
                        row.Cells[0].Value = mySqlReader["ID_InOutput_Car"].ToString();
                        row.Cells[1].Value = mySqlReader["Date_InOutput"].ToString();
                        if (Convert.ToBoolean(mySqlReader["InOut"].ToString()))
                        {
                            row.Cells[2].Value = "В'їзд";
                            row.Cells[2].Style.BackColor = Color.LightGreen;
                        }
                        else
                        {
                            row.Cells[2].Value = "Виїзд";
                            row.Cells[2].Style.BackColor = Color.LightCoral;
                        }
                    }
                }
            }
        }
    }
}

```

```

    }
    catch (Exception ex)
    {
        // Вывод ошибки
        MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
        MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    finally
    {
        if (mySqlReader != null)
            mySqlReader.Close();
        db.close_sql_connection();
    }
}

private void back_button_Click(object sender, EventArgs e)
{
    back_form.Show();
    this.Close();
}
}
}

```

Додаток В

Тестові приклади таблиць БД

Результаты		Сообщения	
	ID_Owner	Owner_Name	Owner_Phone
1	1	Комаров Олександр Григорович	380551341539
2	2	Сафонов Данило Євгенович	380557141300
3	3	Гришина Анастасія Семенівна	380555682002
4	4	Синіцина Аріна Степанівна	380559325103
5	5	Максимова Єлизавета Артемівна	380553749133
6	6	Васильєв Федір Артемович	380554898806
7	7	Богданов Михайло Володимирович	380556567590
8	8	Карасьов Тимофій Олександрович	380552204246
9	9	Белова Мілана Іллінічна	380552153492
10	10	Бабушкін Тихон Михайлович	380557285259
11	11	Федорова Майя Анатоліївна	380550791601
12	12	Казакова Юлія Ярославівна	380559384274
13	13	anton	380111111111

Рис. В.1. Тестовий приклад таблиці Власник ТЗ

Результаты		Сообщения			
	ID_Car	Car_Maker	Car_Model	Car_Color	ID_Car_Owner
1	123123	w	r	t	13
2	AA0000AA	Autoar	A64B	Black	8
3	AA1111AA	Hispano-Argentina	Takar	Blue	5
4	AA2222AA	IAME	Falls	White	1
5	AA3333AA	Koller	Boller	Blue	1
6	AA4444AA	IKA	Nika	White	2
7	AA5555AA	IAME	Opasa 4	Black	3
8	AA6666AA	IAVA	S2	Black	4
9	AA7777AA	Hispano-Argentina	Suiza	Blue	5
10	AA8888AA	Eniak	Durango	White	6
11	AA9999AA	Crespi	Milano	Black	7
12	BB1111BB	AutoLatina	Carros	White	9
13	BB2222BB	Anasagasti	X2	Black	10

Рис. В.2. Тестовий приклад таблиці Машина

Результаты				Сообщения			
	ID_Place	Size_Place	busy_Place				
1	1	10	0				
2	2	10	0				
3	3	10	0				
4	4	10	0				
5	5	10	0				
6	6	10	0				
7	7	10	0				
8	8	10	0				
9	9	10	0				
10	10	10	0				
11	11	10	0				
12	12	10	0				
13	13	10	0				
14	14	10	0				
15	15	10	0				
16	16	10	0				
17	17	10	0				

Рис. В.3. Тестовий приклад таблиці Місце

Результаты					Сообщения				
	ID_Worker	Name_Worker	Phone_Worker	Password_Worker					
1	1	Головін Іван Михайлович	380554897536	123					
2	2	Васильєв Андрій Георгійович	380550755699	123					
3	3	Комаров Пилип Іванович	380557884902	123					

Рис. В.4. Тестовий приклад таблиці Черговий

	ID_Document	ID_Car	ID_Worker	ID_Place	take_date	back_date	tarif	paid_amount	cost
1	1	AA1111AA	2	1	2021-05-24 03:22:00	2021-06-24 03:22:00	10	1000	1200
2	2	AA3333AA	3	2	2021-11-18 11:13:00	2021-12-19 11:13:00	30	4300	4500
3	3	BB2222BB	2	3	2021-04-26 02:23:00	2021-07-26 02:23:00	25	3000	3000
4	4	AA8888AA	1	4	2021-08-16 19:57:00	2021-09-16 19:57:00	12	0	1293
5	5	BB1111BB	1	5	2021-07-23 14:56:00	2021-11-23 14:56:00	10	4000	4000
6	6	AA2222AA	1	6	2021-05-22 20:26:00	2021-06-22 20:26:00	40	1200	1230
7	7	AA4444AA	2	7	2021-07-12 08:55:00	2021-09-12 08:55:00	25	3200	3400
8	8	AA5555AA	3	8	2021-10-21 19:57:00	2021-11-21 19:57:00	30	800	800
9	9	AA6666AA	2	9	2021-04-28 11:35:00	2021-09-28 11:35:00	30	1000	1200
10	10	AA7777AA	1	10	2021-07-21 07:11:00	2021-12-21 07:11:00	20	2000	2000
11	11	AA1111AA	2	5	2022-03-06 18:26:00	2022-05-06 18:26:00	20	0	40
12	13	123123	2	10	2022-03-06 19:31:00	2022-03-08 19:31:00	123	0	7503
13	14	AA1111AA	2	10	2022-06-03 19:36:00	2022-06-04 19:36:00	123	0	123

Рис. В.5. Тестовий приклад таблиці Контракт

Результаты					Сообщения				
	ID_Payment	Date_Payment	Amount_Payment	ID_Place_Payment					
1	1	2021-07-21 07:10:55.000	1000	1					
2	2	2021-10-21 19:57:13.000	2000	2					
3	3	2021-11-23 14:56:00.000	300	1					
4	4	2022-03-06 18:35:39.000	20	1					
5	5	2022-06-04 20:38:48.000	50	10					
6	6	2022-05-06 20:38:48.000	100	15					
7	7	2022-09-25 20:38:48.000	70	80					
8	8	2022-11-25 20:38:48.000	200	46					
9	9	2022-03-13 20:38:48.000	340	34					
10	10	2022-01-18 20:38:48.000	750	67					

Рис. В.6. Тестовий приклад таблиці Оплата

Результаты		Сообщения		
	ID_InOutput	Date_InOutput	InOut	ID_InOutput_Car
1	1	2022-06-03 20:24:40.000	1	AA1111AA
2	3	2022-06-03 20:30:17.000	0	AA1111AA
3	4	2022-06-03 20:36:15.000	1	AA1111AA
4	6	2022-06-04 20:41:06.000	1	AA3333AA
5	7	2022-06-04 20:41:14.000	1	AA5555AA
6	8	2022-06-04 20:41:20.000	0	AA5555AA
7	9	2022-06-04 20:41:23.000	0	AA1111AA

Рис. В.7. Тестовий приклад таблиці В'їзд/Виїзд

Додаток Г

Формуляр-зразок контракту

ДОГОВІР ЗБЕРІГАННЯ ТРАНСПОРТНОГО ЗАСОБУ

М. _____

« ____ » _____

202_р.

_____ в особі _____,

дійсного на підставі _____, далі «**Власник**

автостоянки», з одної сторони, і Громадянин

_____, дані «**Автовласник**» з іншої сторони,

іменовані надалі як "**Сторони**", заключили даний договір, надалі «**Договір**», о нижчеописаному:

1. Власник автостоянки приймає, а Автовласник передає на відповідне зберігання транспортний засіб з номерним знаком _____.
2. Розмір оплати за зберігання транспортного засобу на охороняє мій автостоянці в місяць визначається при оформленні контракту по собівартості послуг або по договірним цінам.
3. Власник автостоянки несе відповідальність:
 - за безпеку транспортного засобу ;
 - за пошкодження транспортного засобу, прийнятого на автостоянку під охорону;
 - за розкомплектацію транспортного засобу, який знаходиться на автостоянці, яка охороняється і крадіжку комплектуючих.
4. Автовласник несе відповідальність за нанесену матеріальну шкоду автостоянці у розмірі нанесеної матеріальної шкоди, за пошкодження транспортних засобів, які знаходяться від охороною стоянки.
5. При рішенні питань, які не передбачені даним договором, сторони керуються дійсним законодавством та Правилами користування платними автомобільними стоянками.

6. Договір оформлений в двох екземплярах, один з яких знаходиться у Власника автостоянки, другий у Автовласник

ЮРИДИЧНІ АДРЕСИ І ПЛАТІЖНІ РЕКВІЗИТИ СТОРІН

Власник автостоянки – Юридична адреса: _____

Почтовий адрес: _____

ЄДРПОУ _____