

Титульный лист

Тема

Разработка консольной системы учета автомобилей на парковке

Участники проекта:

Гузь Давид Иванович, группа [РПО 23/9/1]

Титов Данил Романович, группа [РПО 23/9/1]

Дата: 01 октября 2025 года

Введение

Цель проекта

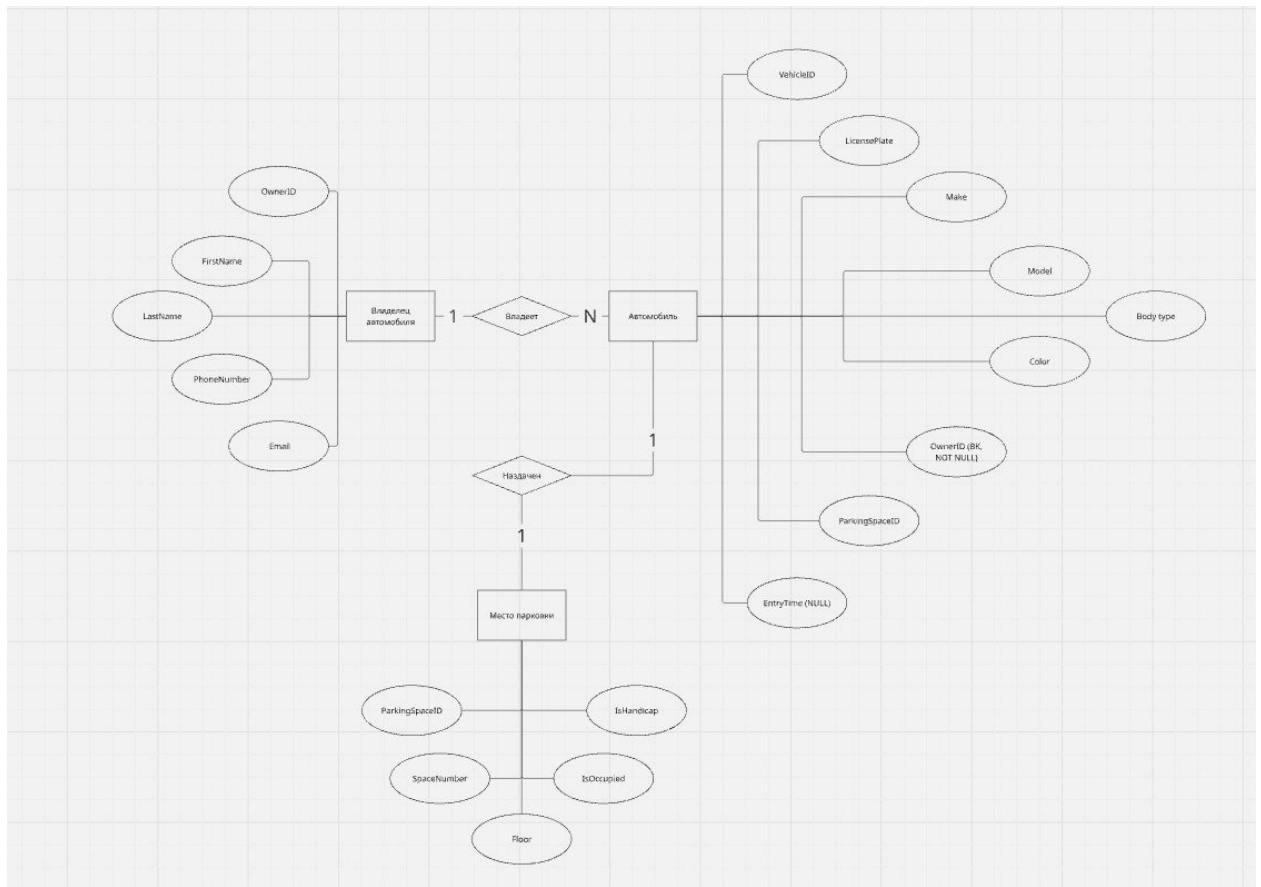
Разработать проект консольной системы учета автомобилей на парковке, позволяющий отображать и управлять данными о парковочных местах и автомобилях.

Задачи проекта

1. Разработать базу данных для хранения информации о парковке.
 - a. Указать объекты:
 - i. Место парковки
 - ii. Владелец автомобиля
 - iii. Автомобиль
 - b. Описать таблицы в ER-диаграмме(написать атрибуты, типы данных, ключи, ограничения)
 - c. Нормализовать БД до 3NF
 - d. Создать таблицы, используя СУБД MS SQL
 - e. Сделать диаграмму БД из SSMS
 - f. Заполнить таблицы базы данных тестовыми данными.
2. Реализовать консольный интерфейс с функциями добавления, редактирования, удаления и просмотра данных, просмотр данных используя фильтрацию и сортировку
 - a. Использовать ЯП C#
 - b. Использовать БД технологию EF Core
 - c. Использовать IDE Visual Studio
3. Провести тестирование программы для проверки корректности работы.
4. Подготовить отчет о результатах продельной работы.
5. Подготовить презентацию для защиты проекта.

База данных

ER-диаграмма



Описание таблиц (атрибуты, типы данных, ключи, ограничения)

1. Таблица: Место парковки (ParkingSpace)

Атрибуты, типы данных и ограничения

Атрибут	Тип данных	Описание	Ключи и ограничения
ParkingSpaceID	INT	Уникальный идентификатор места парковки	Первичный ключ (PK), автоинкремент (IDENTITY(1,1))
SpaceNumber	VARCHAR(10)	Номер парковочного места (например, "A1")	Уникальный (UNIQUE), NOT NULL
IsOccupied	BIT	Статус занятости места (0 = свободно, 1 = занято)	По умолчанию 0 (DEFAULT 0)
IsHandicap	BIT	Индикатор места для людей с ограниченными возможностями (0 = нет, 1 = да)	По умолчанию 0 (DEFAULT 0)

Описание и обоснование

- **ParkingSpaceID**: Используется как уникальный идентификатор для каждого парковочного места. Автоинкремент упрощает добавление новых записей.
- **SpaceNumber**: Уникальный номер места, который должен быть задан при создании записи (NOT NULL) и не может повторяться (UNIQUE).
- **IsOccupied**: Бинарный атрибут для отслеживания занятости места, по умолчанию свободно (0).
- **IsHandicap**: Указывает, предназначено ли место для инвалидов, по умолчанию не предназначено (0).
- **Отсутствие внешних ключей**: Эта таблица является независимой и не ссылается на другие таблицы.

2. Таблица: Владелец автомобиля (VehicleOwner)

Атрибуты, типы данных и ограничения

Атрибут	Тип данных	Описание	Ключи и ограничения
OwnerID	INT	Уникальный идентификатор владельца	Первичный ключ (PK), автоинкремент (IDENTITY(1,1))
FirstName	VARCHAR(50)	Имя владельца	NOT NULL
LastName	VARCHAR(50)	Фамилия владельца	NOT NULL
PhoneNumber	VARCHAR(15)	Номер телефона владельца	Уникальный (UNIQUE), NOT NULL
Email	VARCHAR(100)	Электронная почта владельца	Уникальный (UNIQUE), NULL разрешен

Описание и обоснование

- **OwnerID:** Уникальный идентификатор владельца, генерируется автоматически.
- **FirstName и LastName:** Обязательные поля (NOT NULL), так как имя и фамилия являются ключевыми данными владельца.
- **PhoneNumber:** Уникальный контактный номер, обязательный для связи с владельцем (NOT NULL, UNIQUE).
- **Email:** Необязательный атрибут (NULL разрешен), но если указан, должен быть уникальным для предотвращения дублирования.
- **Отсутствие внешних ключей:** Эта таблица независима, но используется как родительская для таблицы "Автомобиль".

3. Таблица: Автомобиль (Vehicle)

Атрибуты, типы данных и ограничения

Атрибут	Тип данных	Описание	Ключи и ограничения
VehicleID	INT	Уникальный идентификатор автомобиля	Первичный ключ (PK), автоинкремент (IDENTITY(1,1))
LicensePlate	VARCHAR(20)	Номерной знак автомобиля	Уникальный (UNIQUE), NOT NULL
Make	VARCHAR(50)	Марка автомобиля (например, Toyota)	NOT NULL
Model	VARCHAR(50)	Модель автомобиля (например, Camry)	NOT NULL
Color	VARCHAR(30)	Цвет автомобиля	NOT NULL
OwnerID	INT	Идентификатор владельца	Внешний ключ (FK), ссылается на VehicleOwner (OwnerID), NOT NULL
ParkingSpaceID	INT	Идентификатор занятого парковочного места	Внешний ключ (FK), ссылается на ParkingSpace (ParkingSpaceID), NULL разрешен
EntryTime	DATETIME	Время въезда автомобиля на парковку	NULL разрешен

Описание и обоснование

- **VehicleID:** Уникальный идентификатор автомобиля, генерируется автоматически.
- **LicensePlate:** Уникальный номерной знак, обязательный для идентификации автомобиля (NOT NULL, UNIQUE).
- **Make, Model, Color:** Обязательные атрибуты (NOT NULL), описывающие характеристики автомобиля.
- **OwnerID:** Внешний ключ, ссылающийся на OwnerID в таблице "Владелец автомобиля". Обязателен (NOT NULL), так как каждый автомобиль должен иметь владельца.
- **ParkingSpaceID:** Внешний ключ, ссылающийся на ParkingSpaceID в таблице "Место парковки". Может быть NULL, если автомобиль не припаркован.

- **EntryTime:** Время въезда, устанавливается при занятии места и может быть NULL, если автомобиль не на парковке.
- **Ограничения референциальной целостности:** При удалении владельца или места парковки необходимо определить поведение (например, CASCADE или RESTRICT), чтобы избежать "висячих" записей.

Ненормализованная форма

[illegible]

Нормализация в 1NF

ParkingSpace		VehicleOwner	
SpaceNumber	A1	FirstName	Lebron
IsOccupied	0	LastName	James
IsHandicap	0	PhoneNumber	18632963292
		Email	lebron@gmail.com
	Vehicle		
	LicensePlate	LEBRON	
	Make	Lamborghini	
	Model	Urus	
	Color	Black	
	EntryTime	01.06.2025	

Что было изменено:

- Были добавлены строки:
 - Таблица Vehicle:
 - Добавлена строка Model для удаления избыточности данных
 - Таблица VehicleOwner:
 - Удаление строки Name и добавление, вместо нее, строк FirstName и LastName для атомарности значений

Нормализация в 2NF

ParkingSpaceID	0
SpaceNumber	A1
IsOccupied	0
IsHandicap	0

Vehicle	
VehicleID	2
LicensePlate	LEBRON
Make	Lamborghini
Model	Urus
Color	Black
EntrvTime	01.06.2025

Что было изменено:

- Были добавлены первичные ключи на каждую таблицу, чтобы неключевые поля зависели от всего ключа

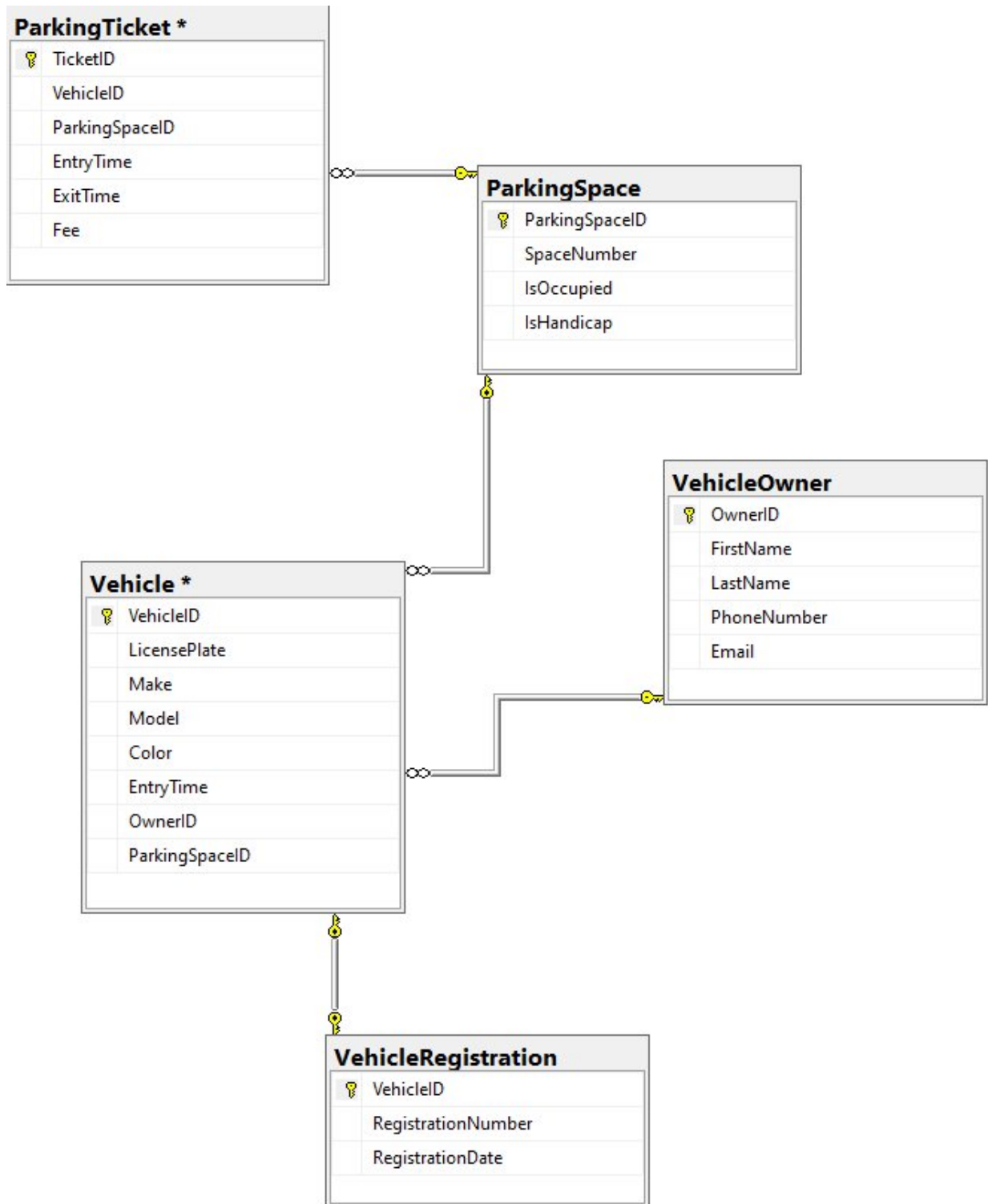
Нормализация в 3NF

ParkingSpace		VehicleOwner	
ParkingSpaceID	0	OwnerID	1
SpaceNumber	A1	FirstName	Lebron
IsOuppped	0	LastName	James
IsHandicap	0	PhoneNumber	18632963292
		Email	lebron@gmail.com
Vehicle		ParkingTicket	
VehicleID	2	TicketID	3
LicensePlate	LEBRON	VehicleID	2
Make	Lamborghini	ParkingSpaceID	0
Model	Urus	EntryTime	01.06.2025
Color	Black	ExitTime	02.06.2025
OwnerID	1	Fee	
ParkingSpaceID	0		
		VehicleRegistration	
		VehicleID	2
		RegistrationNumber	TICK-001
		RegistrationDate	01.06.2025

Что было изменено:

- Была добавлена новая таблица ParkingTicket из атрибутов других таблиц для:
 - Разделения постоянных и временных данных
 - В таблице Vehicle смешивались Make, Model(постоянные данные) и EntryTime(временные данные)
 - Удаления транзитивных зависимостей с помощью добавления внешних ключей
- Добавлена таблица VehicleRegistration для добавления регистрации автомобиля

Диаграмма БД в MS SQL



ДОКУМЕНТАЦИЯ

Скриншоты интерфейса

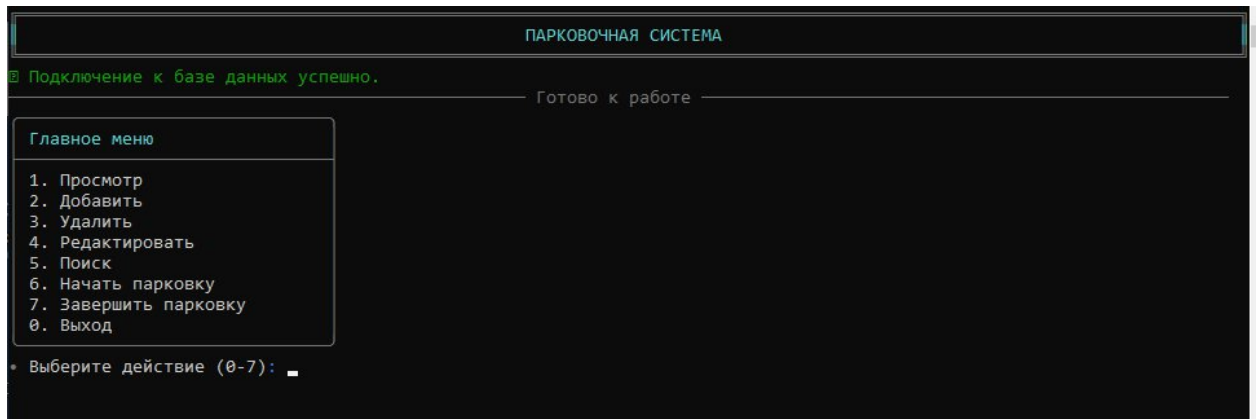


Рис. №1 Подключение к базе данных и выборка действий

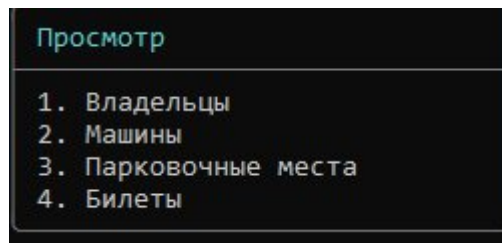


Рис №2 Просмотр данных

FullName	PhoneNumber	Email
Гаврик Артём	+79287374546	1dadada2fga@gmail.com
Гузь Давид	+77777777777	dadad1313@gmail.com
Иван Иванов	+79864345453	dasda@gmail.com
Панов Илья	+79275644323	fafsa@gmail.com
Петров Пётр	+79990002222	petr@example.com
Романов Сергей	+79423608749	serrom11@gmail.com
Селиванов Матвей	88005553535	petyapetyapetya@mail.ru

Рис №2.1 Просмотр владельцев

LicensePlate	Make	Model	Color	Owner
M555MM78	BMW	M5	BLUE	Гузь Давид
O003CM78	Daewoo	Matiz	Розовый	Гаврик Артём
A00100198	Subaru	Forester	Blue	Романов Сергей
B456KT77	Lada	Vesta	White	Петров Пётр
C453BA198	Toyota	Celica	White	Селиванов Матвей
T565CC22	Pajero	V4	Black	Селиванов Матвей

Рис №2.2 Просмотр автомобилей и их владельцев

SpaceNumber	IsOccupied
A1	Свободно
A10	Свободно
A11	Свободно
A12	Свободно
A13	Свободно
A14	Свободно
A15	Свободно
A16	Свободно
A17	Свободно
A18	Свободно
A19	Свободно
A20	Свободно
A4	Свободно
A5	Свободно
A6	Свободно
A7	Свободно
A8	Свободно
A9	Свободно
B1	Занято

Рис №2.3 Просмотр статус занятости парковочных мест

UniqueNumber	LicensePlate	SpaceNumber	EntryTime	ExitTime	Fee
TICK-006	C453BA198	A4	2025-10-15 10:27	2025-10-15 10:28	100,00
TICK-005	O003CM78	B1	2025-10-15 09:33		
TICK-004	M555MM78	A1	2025-10-15 09:20	2025-10-15 10:28	200,00
TICK-002	B456KT77	B1	2025-10-02 09:30	2025-10-08 14:20	14900,00

Рис №2.4 Просмотр билетов

Добавить

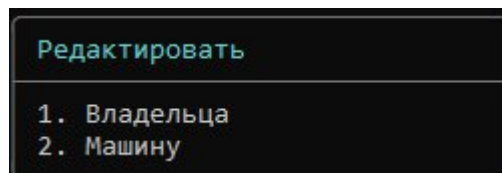
1. Владельца
2. Машину

Рис. №3 Выборка добавления данных

Удалить

1. Владельца
2. Машину
3. Парковочное место
4. Билет

Рис.№4 Выборка удаления данных



Рис№5 Выборка редактирования данных

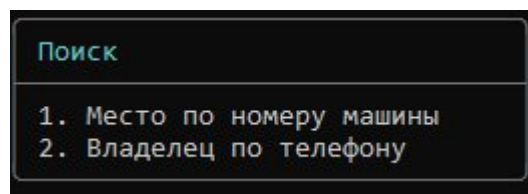


Рис №6 Поиск данных

Test-cases

ID-теста	Описание	Шаги	Обработка	Сведения об ошибках
ТС-001	Добавление владельца	1. Выбрать опцию 2 -> выбрать опцию 1. 2. Ввести данные владельца(Имя, Фамилия, телефон, email)	1. Если данные владельца введены корректно - владелец добавляется в систему. 2. Если ошибка в полях ввода данных - сообщить об ошибке	1. Фамилия/имя дожны иметь хотя бы 1 букву 2. Неверный формат email.
ТС-002	Добавление автомобиля	1. Выбрать опцию 2. -> выбрать опцию 2. 2. Ввести данные авто(госномер, марка, модель, цвет) и ФИ владельца	1. Если данные авто и ФИ владельца введены корректно - автомобиль добавляется в систему 2. Если ошибка в полях ввода - сообщить об ошибке	1. Машина с таким госномером уже существует
ТС-003	Удаление владельца	1. Выбрать опцию 3 ->	1. Если данные	1. Нельзя удалить

		выбрать опцию 1. 2. Ввести ФИ владельца	владельца введены корректно - владелец удаляется из системы. 2. Если ошибка в полях ввода данных - сообщить об ошибке	владельца, у которого есть машины
ТС-004	Удаление автомобиля	1. Выбрать опцию 3 -> опцию 2. 2. Ввести госномер авто	1. Если госномер авто существует, автомобиль удаляется из системы. 2. Если ошибка в полях ввода данных - сообщить об ошибке	1. Машина с таким госномером не найдена. 2. Нельзя удалить машину с активным билетом
ТС-005	Удаление парковочного места	1. Выбрать опцию 3 -> выбрать опцию 3. 2. Ввести номер места	1. Если номер места свободен - место удаляется из системы. 2. Если номер места занят - сообщить об ошибке	1. Нельзя удалить занятое место
ТС-006	Удаление билета	1. Выбрать опцию 3 -> опцию 4. 2. Ввести номер билета	1. Если номер билета неактивный - билет удаляется из системы. 2. Если номер билета активный - сообщить об ошибке	1. Нельзя удалить активный билет
ТС-007	Начало парковки	1. Выбрать опцию 6. 2. Ввести госномер авто	1. Если госномер авто существует - начать парковку. 2. Если	1. Машина с таким госномером не найдена

			госномера авто не существует - сообщить об ошибке.	
ТС-008	Завершение парковки	1. Выбрать опцию 7. 2. Ввести номер билета	1. Если номер билета введен корректно - завершить парковку. 2. Если ошибка в поле ввода номера билета - сообщить об ошибке	1. Билет уже закрыт. 2. Билет не найден.
ТС-009	Вывод данных	1. Выбрать опцию 1 2. Далее выбрать любую из опций.	1. Если владельцы существуют - вывод владельцев. 2. Если автомобили существуют - вывод автомобилей. 3. Если парковочные места существуют - вывод парковочных мест. 4. Если билеты существуют - вывод билетов	
ТС-010	Поиск места	1. Выбрать опцию 5 -> выбрать опцию 1 или . 2. Ввести госномер авто	1. Если госномер авто существует - отобразить место. 2. Если ошибка в поле ввода госномера - сообщить об ошибке	1. Не найдено активных парковок для этого номера
ТС-011	Поиск владельца	1. Выбрать опцию 5 -> выбрать опцию 2. 2. Ввести	1. Если номер телефона существует -	1. Владелец с таким телефоном

		номер владельца	отобразить владельца. 2. Если ошибка в поле ввода телефона - сообщить об ошибке	не найден
--	--	--------------------	---	-----------

Программный код(основные методы)

Добавление владельца

```
public void AddOwner(string first, string last, string phone, string? email)
{
    if (string.IsNullOrEmpty(first) || string.IsNullOrEmpty(last))
        throw new Exception("Имя и фамилия обязательны");
    if (first.Any(char.IsDigit) || last.Any(char.IsDigit))
        throw new Exception("Имя и фамилия не должны содержать цифры");

    // длины под типовую схему БД
    if (first.Length > 50) throw new Exception("Имя не должно превышать 50 символов");
    if (last.Length > 50) throw new Exception("Фамилия не должна превышать 50 символов");
    if (!string.IsNullOrEmpty(email) && email.Length > 100)
        throw new Exception("Email не должен превышать 100 символов");

    // Нормализация телефона: всегда +7XXXXXXXXXX
    phone = NormalizePhone(phone);

    if (!string.IsNullOrEmpty(email) &&
        !Regex.IsMatch(email, @"^[^@\s]+@[^@\s]+\.[^@\s]+$"))
        throw new Exception("Неверный формат email (например, user@example.com)");

    _ctx.VehicleOwners.Add(new VehicleOwner
    {
```



```

        FirstName = first.Trim(),
        LastName = last.Trim(),
        PhoneNumber = phone,
        Email = string.IsNullOrEmpty(email) ? null : email.Trim()
    });
    _ctx.SaveChanges();
}

```

Редактирование владельца

```

public void EditOwner(string fullName, string newPhone, string? newEmail)
{
    var owner = _ctx.VehicleOwners
        .FirstOrDefault(o => (o.LastName + " " + o.FirstName) == fullName)
        ?? throw new Exception("Владелец не найден");

    if (!string.IsNullOrEmpty(newPhone))
        owner.PhoneNumber = NormalizePhone(newPhone);

    if (!string.IsNullOrEmpty(newEmail))
    {
        if (newEmail.Length > 100) throw new Exception("Email не должен превышать 100 символов");
        if (!Regex.IsMatch(newEmail, @"^[^@\s]+@[^\s]+\.[^\s]+$"))
            throw new Exception("Неверный формат email");
        owner.Email = newEmail.Trim();
    }

    _ctx.SaveChanges();
}

```

Добавление авто

```

public void AddVehicle(string plate, string make, string model, string color, string
ownerFullName)
{

```

```

        if (string.IsNullOrWhiteSpace(plate) || plate.Length < 6 || plate.Length > 10)
            throw new Exception("Госномер обязателен и должен содержать от 6 до 10 символов");

        if (string.IsNullOrWhiteSpace(make)) throw new Exception("Марка автомобиля обязательна");

        if (string.IsNullOrWhiteSpace(model)) throw new Exception("Модель автомобиля обязательна");

        if (string.IsNullOrWhiteSpace(color)) throw new Exception("Цвет автомобиля обязателен");

```

```

var owner = _ctx.VehicleOwners
    .FirstOrDefault(o => (o.LastName + " " + o.FirstName) == ownerFullName)
    ?? throw new Exception("Владелец с таким ФИО не найден");

```

```

if (_ctx.Vehicles.Any(v => v.LicensePlate == plate))
    throw new Exception("Машина с таким госномером уже существует");

```

```

_ctx.Vehicles.Add(new Vehicle
{
    LicensePlate = plate.Trim(),
    Make = make.Trim(),
    Model = model.Trim(),
    Color = color.Trim(),
    OwnerID = owner.OwnerID
});
_ctx.SaveChanges();
}

```

Редактирование авто

```

public void EditVehicle(string plate, string? make, string? model, string? color)
{
    var vehicle = _ctx.Vehicles.FirstOrDefault(v => v.LicensePlate == plate)
        ?? throw new Exception("Машина не найдена");
}

```

```
if (!string.IsNullOrWhiteSpace(make)) vehicle.Make = make.Trim();  
if (!string.IsNullOrWhiteSpace(model)) vehicle.Model = model.Trim();  
if (!string.IsNullOrWhiteSpace(color)) vehicle.Color = color.Trim();  
  
_ctx.SaveChanges();  
}
```