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

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Цель проекта

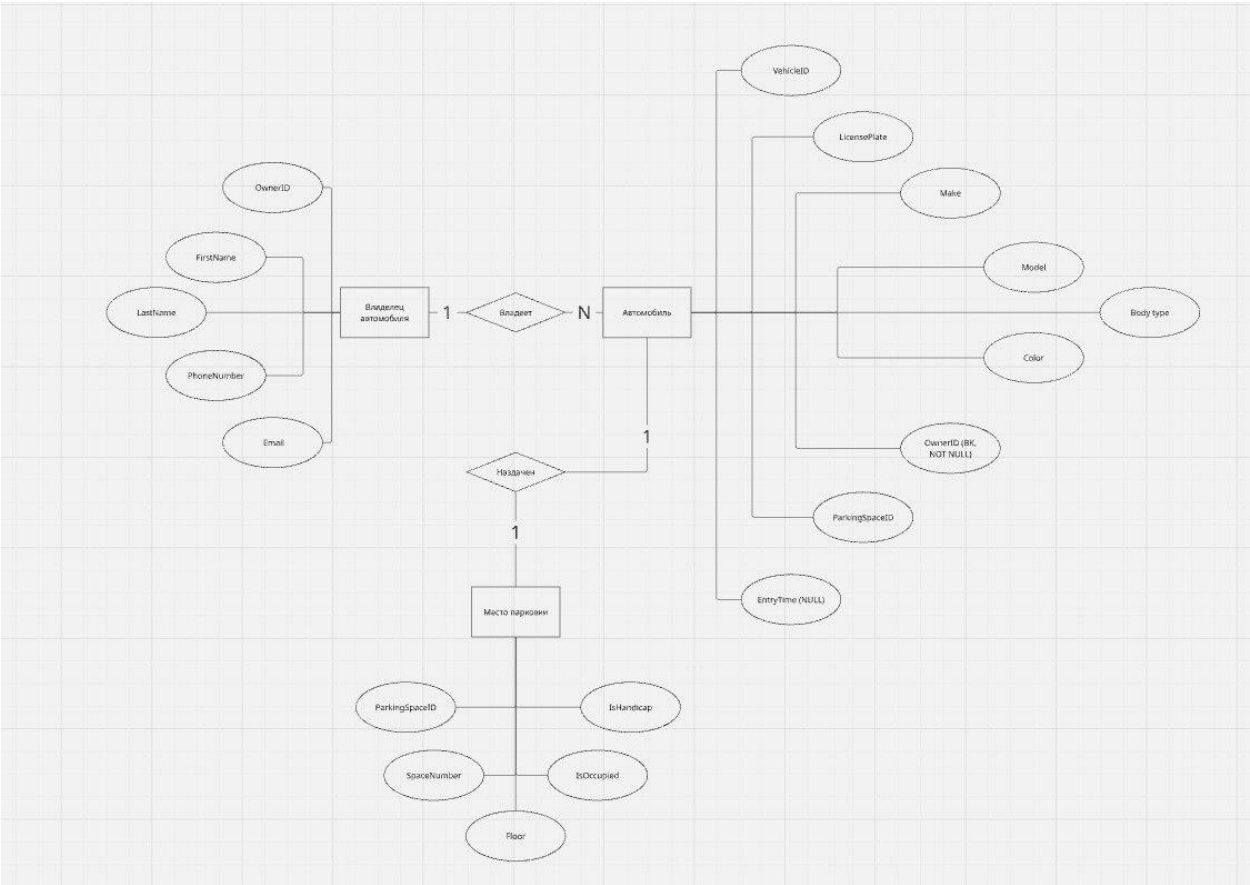
Разработать проект консольной системы учета автомобилей на парковке, позволяющий отображать и управлять данными о парковочных местах и автомобилях.

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.

НОРМАЛИЗАЦИЯ ТАБЛИЦ

Ненормализованная форма

ParkingSpace	
SpaceNumber	A1
IsOccupied	0
IsHandicap	0

VehicleOwner	
Name	Lebron James
PhoneNumber	18632963292
Email	lebron@gmail.com

Vehicle	
LicensePlate	LEBRON
Make	Lamborghini Urus
Color	Black
EntryTime	01.06.2025

Нормализация в 1NF

ParkingSpace	
SpaceNumber	A1
IsOccupied	0
IsHandicap	0

VehicleOwner	
FirstName	Lebron
LastName	James
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, заместо нее, строк FirstName и LastName для атомарности значений

Нормализация в 2NF

ParkingSpace	
ParkingSpaceID	0
SpaceNumber	A1
IsOccupied	0
IsHandicap	0

VehicleOwner	
OwnerID	1
FirstName	Lebron
LastName	James
PhoneNumber	18632963292
Email	lebron@gmail.com

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

Что было изменено:

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

Нормализация в 3NF

ParkingSpace	
ParkingSpaceID	0
SpaceNumber	A1
IsOquppied	0
IsHandicap	0

VehicleOwner	
OwnerID	1
FirstName	Lebron
LastName	James
PhoneNumber	18632963292
Email	lebron@gmail.com

Vehicle	
VehicleID	2
LicensePlate	LEBRON
Make	Lamborghini
Model	Urus
Color	Black
OwnerID	1
ParkingSpaceID	0

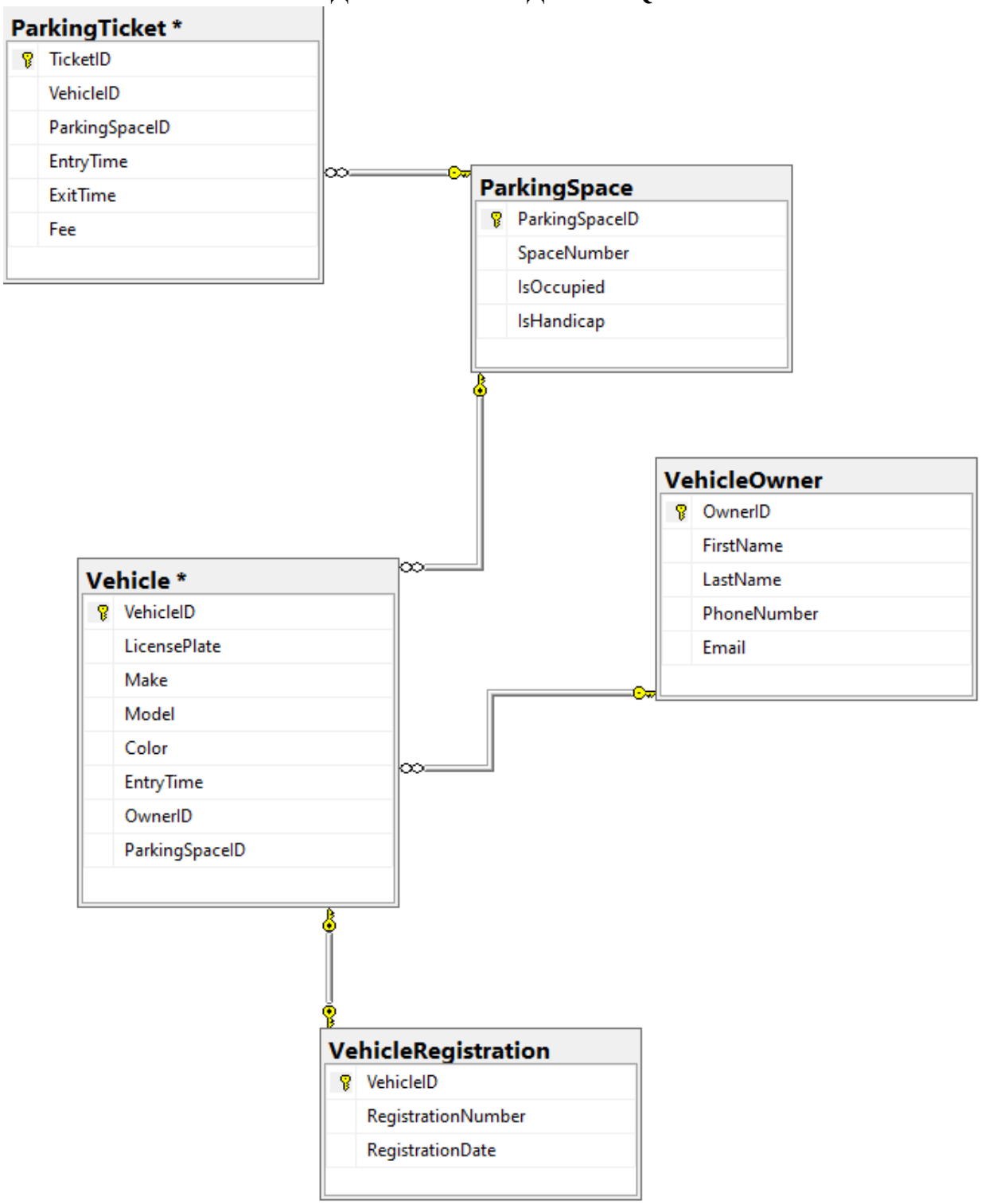
ParkingTicket	
TicketID	3
VehicleID	2
ParkingSpaceID	0
EntryTime	01.06.2025
ExitTime	02.06.2025
Fee	

VehicleRegistration	
VehicleID	2
RegistrationNumber	TICK-001
RegistrationDate	01.06.2025

Что было изменено:

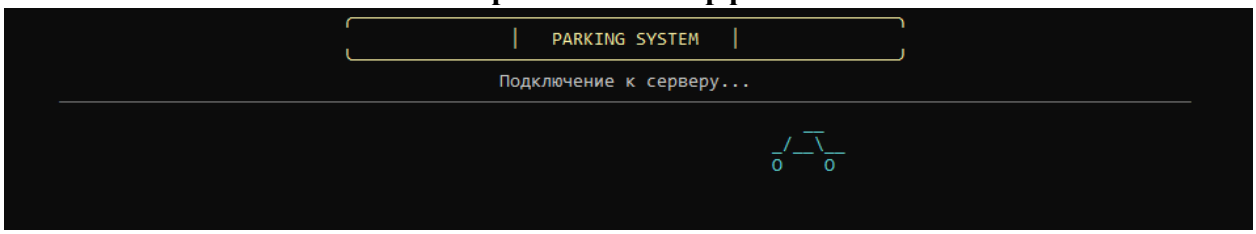
- Была добавлена новая таблица ParkingTicket из атрибутов других таблиц для:
  - Разделения постоянных и временных данных
    - В таблице Vehicle смешивались Make, Model(постоянные данные) и EntryTime(временные данные)
  - Удаления транзитивых зависимостей с помощью добавления внешних ключей
- Добавлена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

ДИАГРАММА БД В MS SQL



ДОКУМЕНТАЦИЯ

Скриншоты интерфейса



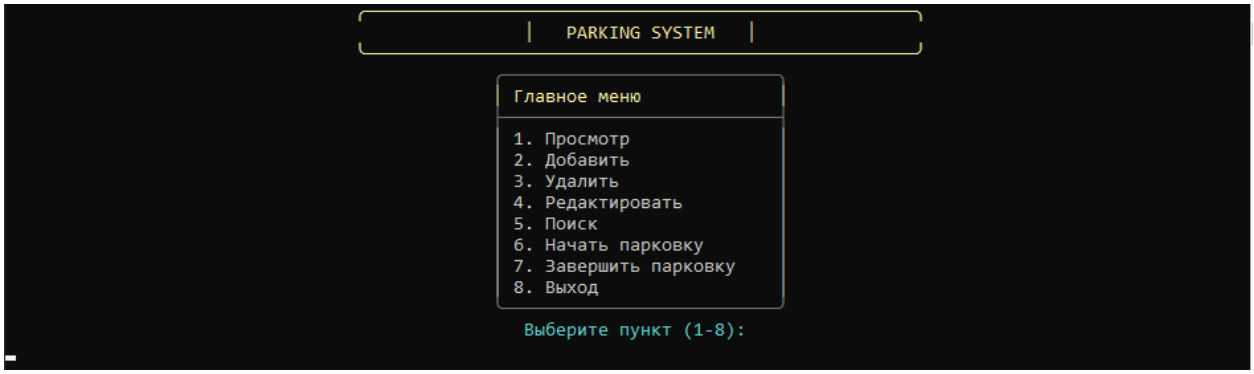


Рис. №1, 2 Подключение к базе данных и выборка действий

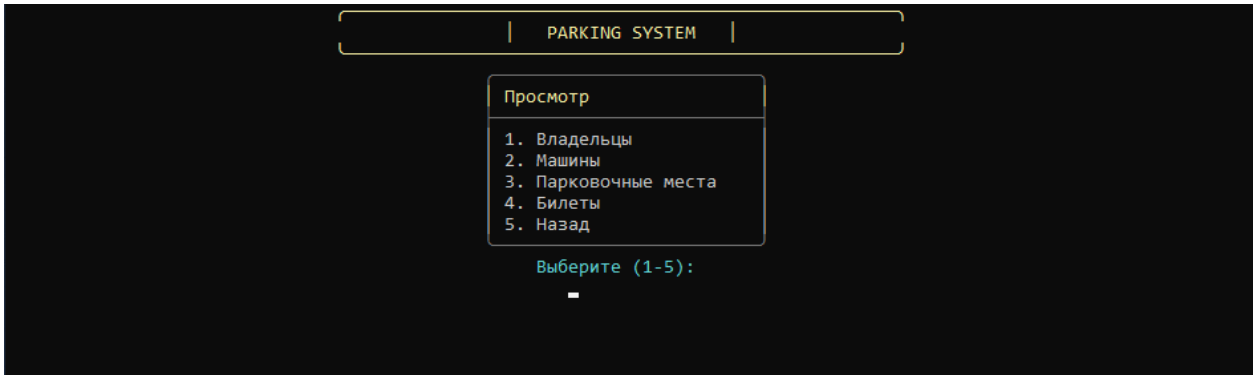


Рис №3 Просмотр данных

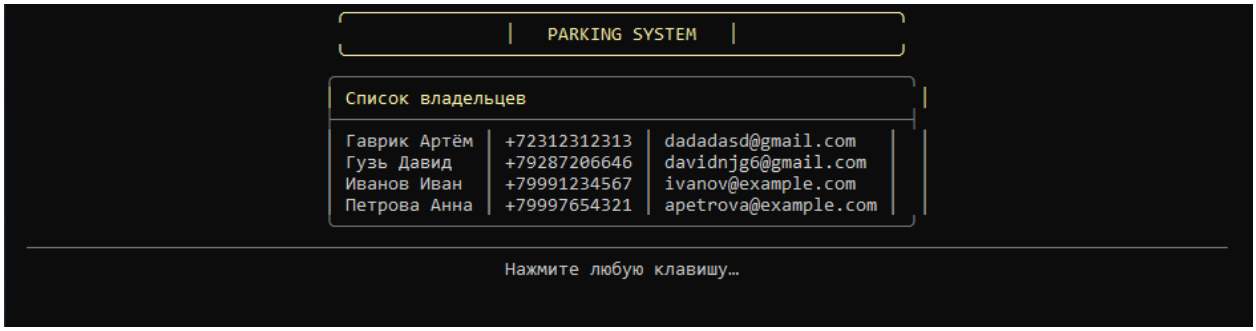


Рис №3.1 Просмотр владельццев

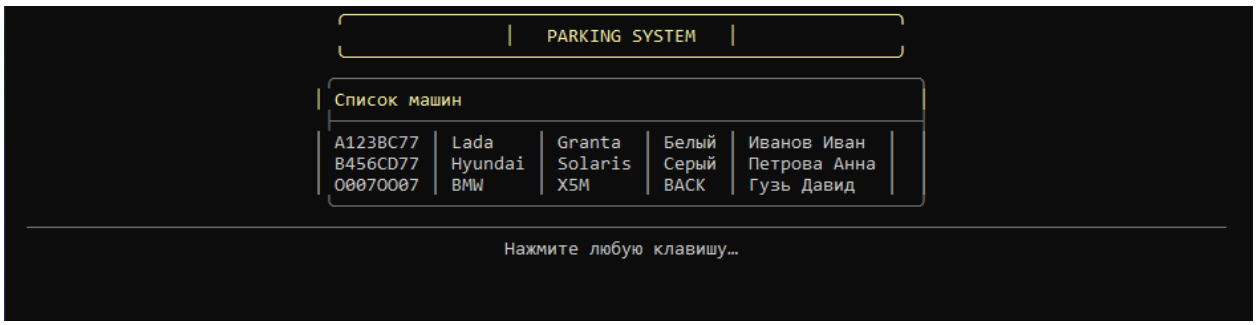


Рис №3.2 Просмотр автомобилей и их владельцев

Рис №3.3 Просмотр статус занятости парковочных мест

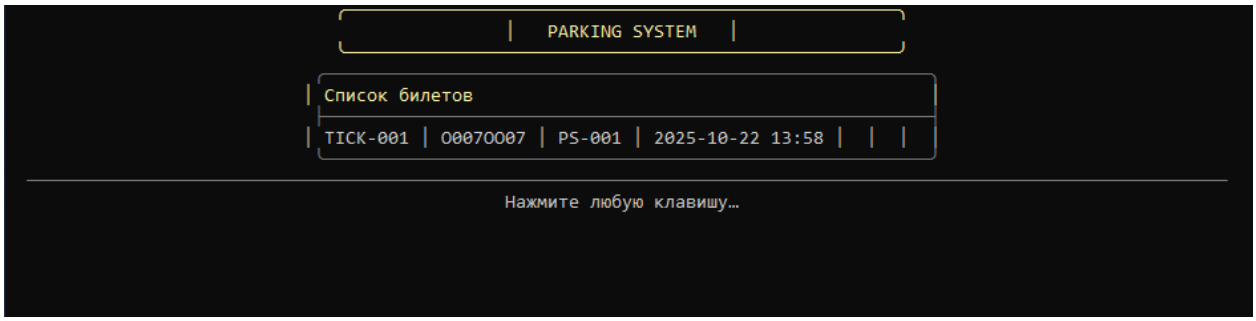


Рис №3.4 Просмотр билетов

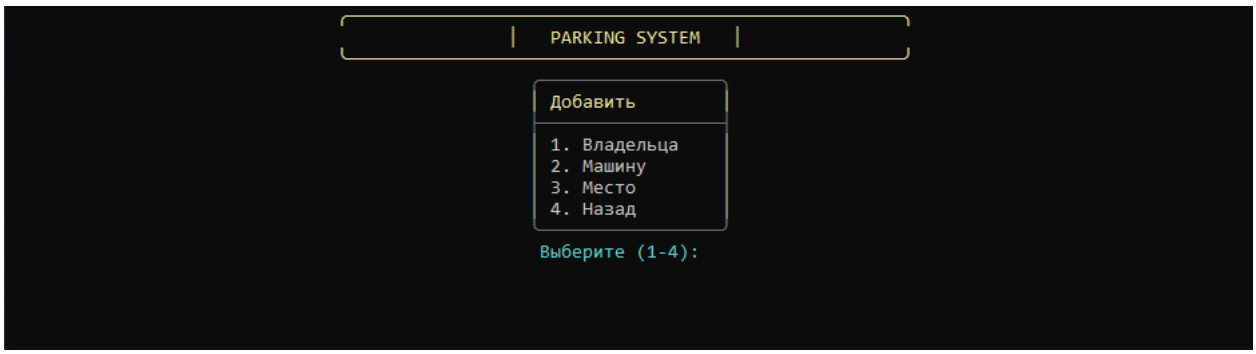
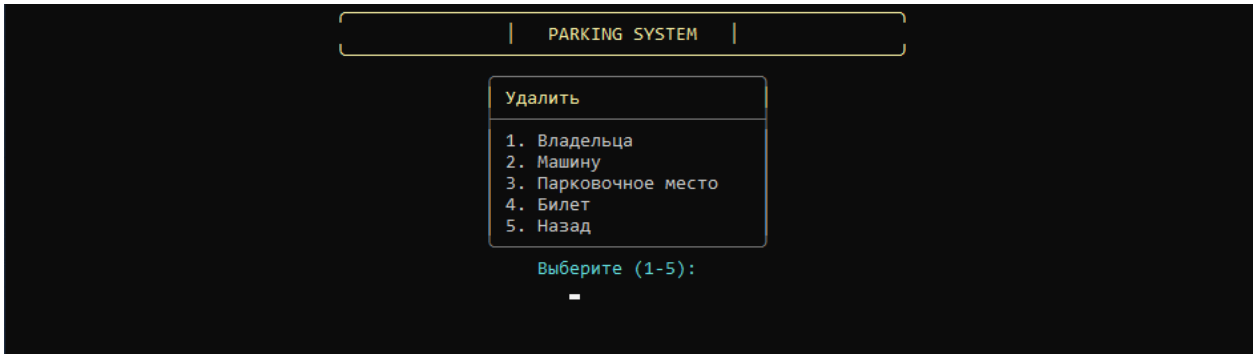
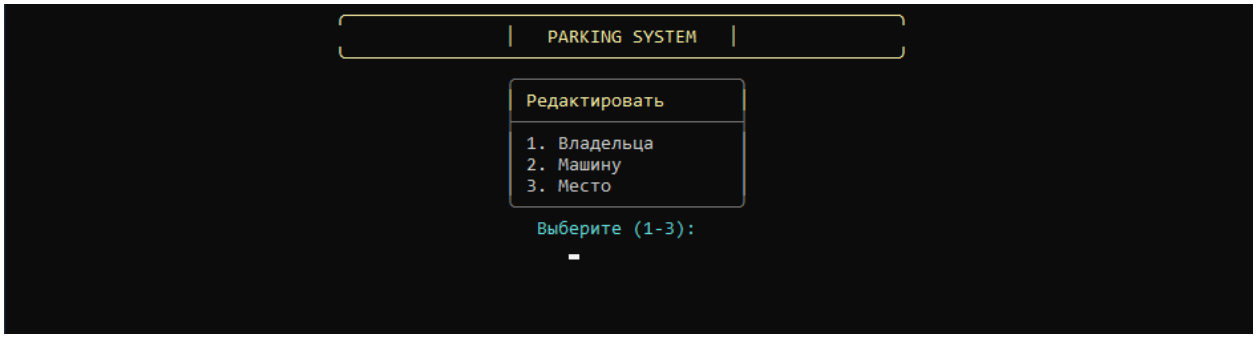


Рис. №4 Выборка добавления данных



Рис№5 Выборка удаления данных



Рис№6 Выборка редактирования данных

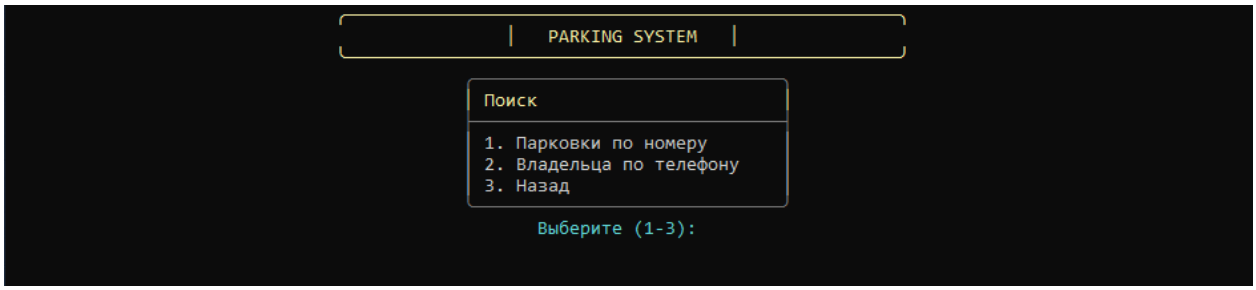


Рис №7 Поиск данных

TEST-CASES

ID-теста	Описание	Шаги	Обработка	Сведения об ошибках
TC-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 доожны иметь хотя бы 2 буквы. 2. Неверный формат email. 3. Только буквы(рус/англ), дефис
TC-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
TC-003	Удаление владельца	1. Выбрать опцию 3 -> выбрать опцию 1. 2. Ввести ФИ владельца	1. Если данные владельца введены корректно - владелец удаляется из системы. 2. Если ошибка в полях ввода данных - сообщить об ошибке.	1. Нельзя удалить владельца, у которого есть машины 2. Владелец с таким ФИ не найден
TC-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
TC-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 2. Место не найдено
TC-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 3. Если ошибка в поле ввода билета – сообщить об ошибке	1. Нельзя удалить активный билет. 2. Неверный формат билета
TC-007	Начало парковки	1. Выбрать опцию 6. 2. Ввести госномер авто.	1. Если госномер авто существует - начать парковку.	1. Машина с таким госномером не найдена

			2. Если госномера авто не существует – не давать ввести госномер.	
TC-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 – не давать вводить билет	1.Билет уже закрыт. 2. Билет не найден.
TC-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	
TC-010	Поиск места	1. Выбрать опцию 5 -> выбрать опцию 1. 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
TC-011	Поиск владельца	1. Выбрать опцию 5 -> выбрать опцию 2. 2. Ввести номер владельца.	1. Если номер телефона существует - отобразить владельца. 2. Если ошибка в поле ввода телефона - сообщить об ошибке	1. Владелец с таким телефоном не найден

ПРОГРАММНЫЙ КОД(ОСНОВНЫЕ МЕТОДЫ)

Добавление владельца

```
public void AddOwner(string first, string last, string phone, string email)
{
   ValidateNames(first, last);

   phone = NormalizePhone(phone);
   if (string.IsNullOrWhiteSpace(email))
      throw new Exception("Email обязателен");
   if (email.Length > 100)
      throw new Exception("Email не должен превышать 100 символов");
   if (!Regex.IsMatch(email, @"^[^@\s]+@[^@\s]+\.[^@\s]+$"))
      throw new Exception("Неверный формат email (например, user@example.com)");
```

```
    if (_ctx.VehicleOwners.Any(o => o.PhoneNumber == phone))
        throw new Exception("Пользователь с таким телефоном уже существует");

    if (_ctx.VehicleOwners.Any(o => o.Email != null && o.Email == email))
        throw new Exception("Пользователь с таким email уже существует");

    _ctx.VehicleOwners.Add(new VehicleOwner
    {
        FirstName = first.Trim(),
        LastName = last.Trim(),
        PhoneNumber = phone,
        Email = 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.IsNullOrWhiteSpace(newPhone))
    {
        var phone = NormalizePhone(newPhone);
        if (_ctx.VehicleOwners.Any(o => o.OwnerID != owner.OwnerID && o.PhoneNumber == phone))
            throw new Exception("Этот телефон уже используется другим владельцем");
        owner.PhoneNumber = phone;
    }

    if (!string.IsNullOrWhiteSpace(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");
        var mail = newEmail.Trim();
        if (_ctx.VehicleOwners.Any(o => o.OwnerID != owner.OwnerID && o.Email != null && o.Email == mail))
            throw new Exception("Этот email уже используется другим владельцем");
        owner.Email = mail;
    }

    _ctx.SaveChanges();
}
```

Добавление автомобиля

```
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 автомобиля

```
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();
}
```

Добавление парковочного места

```
public void AddSpace(string spaceNumber)
{
    if (string.IsNullOrWhiteSpace(spaceNumber) || spaceNumber.Length > 10)
        throw new Exception("Номер места обязателен и не должен превышать 10 символов");

    if (_ctx.ParkingSpaces.Any(p => p.SpaceNumber == spaceNumber))
        throw new Exception("Место с таким номером уже существует");

    _ctx.ParkingSpaces.Add(new ParkingSpace
    {
        SpaceNumber = spaceNumber.Trim(),
        IsOccupied = false
    });
    _ctx.SaveChanges();
}
```

Редактирование парковочного места

```
public void EditSpace(string currentNumber, string newNumber)
{
    if (string.IsNullOrWhiteSpace(currentNumber))
        throw new Exception("Текущий номер места не указан");
    if (string.IsNullOrWhiteSpace(newNumber) || newNumber.Length > 10)
        throw new Exception("Новый номер обязателен и не должен превышать 10 символов");

    var space = _ctx.ParkingSpaces.FirstOrDefault(p => p.SpaceNumber == currentNumber)
        ?? throw new Exception("Место не найдено");

    if (_ctx.ParkingSpaces.Any(p => p.SpaceNumber == newNumber && p.ParkingSpaceID != space.ParkingSpaceID))
        throw new Exception("Место с таким номером уже существует");

    space.SpaceNumber = newNumber.Trim();
    _ctx.SaveChanges();
}
```