

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ  
Учреждения образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ  
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий  
Кафедра программной инженерии  
Специальность 1-40 01 01 Программное обеспечение информационных технологий  
Направление специальности 1-40 01 01 10 Программное обеспечение  
информационных технологий (программирование интернет приложений)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА  
КУРСОВОГО ПРОЕКТА:**

по дисциплине «Объектно-ориентированные технологии программирования и стандарты проектирования»  
Тема Программное средство «Автошкола»

Исполнитель  
студент 2 курса группы 6 Подобед Владислав Георгиевич  
(Ф.И.О.)

Руководитель работы преп.-стажёр Чистякова Ю.А.  
(учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой \_\_\_\_\_  
Председатель Пацей Н.В.  
(подпись)

## Содержание

Введение .....	3
1 Анализ прототипов, литературных источников и формирование требований к проектируемому программному средству.....	4
2 Анализ требований к программному средству и разработка функциональных требований .....	8
2.1 Описание инструментов для разработки программного средства .....	8
2.2 Описание функциональности программного средства.....	9
3 Проектирование программного средства .....	10
3.1 Модель базы данных .....	10
3.2 Структура проекта .....	12
3.3 Структура классов программного средства .....	13
3.4 Работа с базой данных.....	14
3.5 Аутентификация пользователей в системе .....	14
3.6 Окна приложения.....	14
4 Реализация программного средства.....	17
4.1 Реализация MVVM и других шаблонов .....	17
4.2 Оставление отзывов.....	21
4.3 Запись на занятие .....	22
5 Тестирование, проверка работоспособности и анализ полученных результатов .	24
6 Методика использования программного средства .....	28
6.1 Регистрация и авторизация .....	28
6.2 Домашнее представление.....	29
6.3 Администрирование .....	32
6.4 Смена пользователя .....	34
Заключение .....	35
Список литературы .....	36
Приложение А .....	37
Приложение Б.....	38

## Введение

В настоящее время наблюдается бурное развитие информационных технологий и программных средств в сфере обучения водительских навыков. Все существующие автошколы созданы только ради одной благородной цели – помочь вам, уважаемые автолюбители в приобретении водительских прав. Однако не стоит заблуждаться в том, что вы просто придете в одно из учебных заведений и получите вожделенную пластиковую карточку. Ведь сами подумайте, какой толк от машины, если вы не знаете, как ей правильно пользоваться? Вы, конечно, можете возразить, что в принципе, опыт, который вы можете получить от других людей, пусть даже и отрывочный. Но почему бы нам тогда и не учиться в школе или в университете? Дело в том, что в любом обучении очень важен подход, важна некоторая систематизация знаний, накопленных многими годами. Поэтому обучение в автошколе — это такой же необходимый этап в стремлении научиться управлять автомобилем, как собственно и его покупка. Остается только выбрать лучшую автошколу.

Целью данного курсового проекта является разработка программного средства, которое позволит курсанту автошколы в любое время в комфортных для него условиях, быстро и удобно, планировать свои занятия с инструкторами. Автоинструктору – легко просматривать своё расписание занятий и информацию, связанную с предстоящим занятием. Также программное средство позволяет администратору просматривать информацию о пользователях, зарегистрированных в системе, о всех занятиях и управлять необходимой для функционирования системы информацией.

Достоинства программы:

- удобный графический интерфейс в сравнении с конкурентами;
- бесплатный доступ;

## 1 Анализ прототипов, литературных источников и формирование требований к проектируемому программному средству

На сегодняшний день существует множество программ для автошкол. Системы программирования дают возможность удобно работать с базами данных.

Использование современных технологий – естественный выбор для создания приложений на современной технологической базе.

В качестве одного из аналогов возьмем программу «ПРОФТЕХ»[1]. Данная программа была разработана под мобильную операционную систему Android. Имеет следующие возможности:

- видеть расписание своего инструктора по вождению;
- осуществлять онлайн-запись на занятия к инструктору;
- выставлять оценки инструктору после каждого занятия;
- получать оперативные уведомления об отмене занятий, изменении в расписании.

Интерфейс программы представлен на рисунке 1.1.

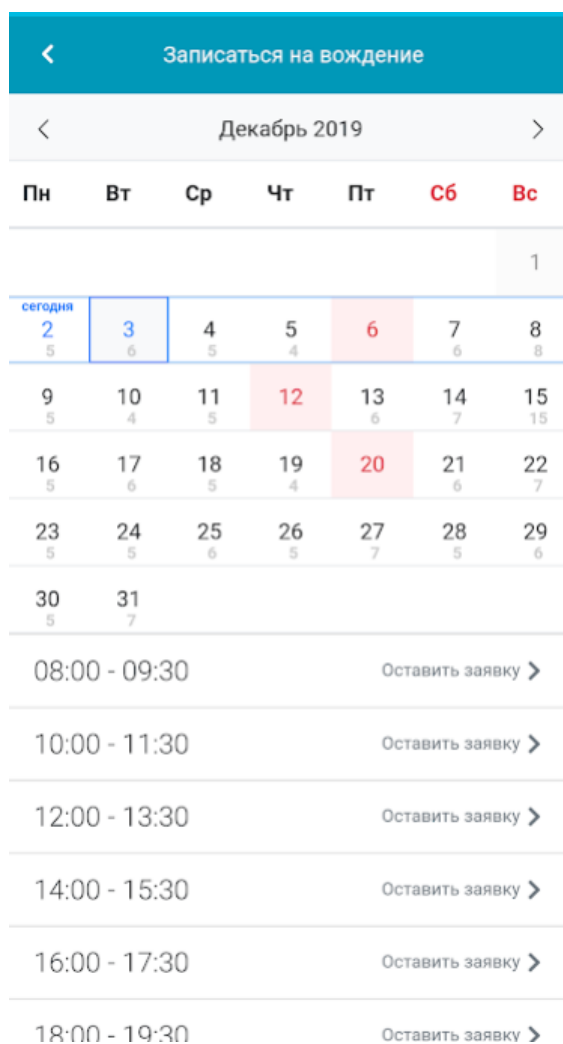


Рисунок 1.1 – ПРОФТЕХ

### Недостатки «ПРОФТЕХ»:

- курсант должен быть зарегистрирован в определенной автошколе;
- инструктор закрепляется за курсантом, т.е. записаться на занятия можно только к нему.

Следующая программа для анализа – «Hesus»[2]. Имеет следующие возможности:

- осуществлять онлайн-запись на занятия к инструктору;
- получать уведомления об отмене занятий, изменении в расписании;
- отслеживание динамики учебы.

Интерфейс программы представлен на рисунке 1.2.

The screenshot displays the 'Hesus' program interface for booking lessons. It is divided into three main sections:

- 1. Выберите свободный день месяца**: This section shows three monthly calendars for May 2022, June 2022, and July 2022. Days are color-coded: green for available and orange for unavailable. In May, the 12th is orange. In June, the 1st-5th are green. In July, the 1st-3rd are green.
- 2. Выберите свободное время. Всего записей: 7**: This section lists seven time slots: 07:00 – 09:00, 09:00 – 11:00, 11:00 – 13:00, 13:30 – 15:30, 15:30 – 17:30, 17:30 – 19:30, and 19:30 – 21:00.
- 3. Укажите информацию для связи с Вами**: This section contains input fields for 'Имя/ФИО', 'Телефон', 'Email', and 'Группа'. There is a dropdown menu for 'Где вождение?' and a checkbox for 'Согласен на обработку персональных данных'. A blue 'Бронировать' button is at the bottom right. A small logo for 'Квалификация - Условия использования' is in the bottom right corner.

Рисунок 1.2 – Hesus

### Недостатки «Hesus»:

- нельзя выбирать инструктора/ автомобиль для занятий;
- для подтверждения занятия вам будут звонить или писать в другом мессенджере.

Рассмотрим программу «SignMeUp»[4]. Имеет следующие возможности:

- назначать и отменять практические занятия;
- оставлять отзывы инструктору.

Запись к инструктору: **Василевский И. П.**

Выберите удобное время. Можно записаться сразу на несколько занятий.

26.02.14 Среда	27.02.14 Четверг	28.02.14 Пятница	01.03.14 Суббота	02.03.14 Воскресенье	03.03.14 Понедельник
06:00 - 07:30	06:00 - 07:30	06:00 - 07:30	07:00 - 08:30	07:00 - 08:30	06:00 - 07:30
07:30 - 09:00	07:00 - 08:30	07:30 - 09:00	<input checked="" type="checkbox"/> 08:30 - 10:00	08:30 - 10:00	07:30 - 09:00
09:30 - 11:00	08:30 - 10:00	08:30 - 10:00	10:00 - 11:30	10:00 - 11:30	09:30 - 11:00
11:30 - 13:00	10:00 - 11:30	10:00 - 11:30	11:30 - 13:00	11:30 - 13:00	11:30 - 13:00
13:30 - 15:00	11:30 - 13:00	11:30 - 13:00	13:00 - 14:30	13:00 - 14:30	13:30 - 15:00
15:30 - 17:00	13:30 - 15:00	13:30 - 15:00	<input checked="" type="checkbox"/> 14:30 - 16:00	14:30 - 16:00	14:30 - 16:00
17:30 - 19:00	15:00 - 16:30	15:00 - 16:30	16:00 - 17:30	16:00 - 17:30	16:30 - 18:00
19:30 - 21:00	17:00 - 18:30	17:00 - 18:30	17:30 - 19:00	17:30 - 19:00	17:30 - 19:00
		19:30 - 21:00			19:00 - 20:30

Записать

Отмена

Рисунок 1.4 – SignMeUp

Недостатки «SignMeUp»:

- нельзя поменять инструктора;
- скудный функционал.

Рассмотрим программу «keneo»[5]. Имеет следующие возможности:

- осуществлять онлайн-запись на занятия к инструктору;
- получать уведомления об отмене занятий, изменении в расписании;
- отслеживание динамики учебы.

Интерфейс программы представлен на рисунке 1.5.

ГРАФИК

СТАТИСТИКА

НАСТРОЙКИ

Запись на вождение

Отображение графика

февраль

март

апрель

май

июнь

июль

август

Стандартный

Компактный

Список инструкторов

Иванов Сергей  
Х5 х555хх

Воробьев Михаил  
а999бб

Конушкин Иван  
Х5 х555хх

Григорьев Константин  
2115 о000оо

Абрамчук Геннадий  
х000хх

Павел Гусев  
в391рв

Ианов Иан  
а999бб

Сюсин Владимир  
Васильевич

1, ВС						
1 08:30 - 10:00						
2 10:00 - 11:30						
3 11:30 - 13:00						
4 13:00 - 14:30						
5 14:30 - 16:00						
6 16:30 - 18:00						
7 18:00 - 19:30						
2, ПН	3, ВТ	4, СР	5, ЧТ	6, ПТ	7, СБ	8, ВС
1 08:30 - 10:00	1 08:30 - 10:00	1 08:30 - 10:00	1 08:30 - 10:00	1 08:30 - 10:00	1 08:30 - 10:00	1 08:30 - 10:00
2 10:00 - 11:30	2 10:00 - 11:30	2 10:00 - 11:30	2 10:00 - 11:30	2 10:00 - 11:30	2 10:00 - 11:30	2 10:00 - 11:30
3 11:30 - 13:00	3 11:30 - 13:00	3 11:30 - 13:00	3 11:30 - 13:00	3 11:30 - 13:00	3 11:30 - 13:00	3 11:30 - 13:00
4 13:00 - 14:30	4 13:00 - 14:30	4 13:00 - 14:30	4 13:00 - 14:30	4 13:00 - 14:30	4 13:00 - 14:30	4 13:00 - 14:30
5 14:30 - 16:00	5 14:30 - 16:00	5 14:30 - 16:00	5 14:30 - 16:00	5 14:30 - 16:00	5 14:30 - 16:00	5 14:30 - 16:00
6 16:30 - 18:00	6 16:30 - 18:00	6 16:30 - 18:00	6 16:30 - 18:00	6 16:30 - 18:00	6 16:30 - 18:00	6 16:30 - 18:00
7 18:00 - 19:30	7 18:00 - 19:30	7 18:00 - 19:30	7 18:00 - 19:30	7 18:00 - 19:30	7 18:00 - 19:30	7 18:00 - 19:30

Рисунок 1.5 – keneo

Недостатки «keneo»:

- пустой интерфейс;
- скудный функционал.

Еще одна программа для анализа – «Автошкола-контроль»[6]. Имеет следующие возможности:

- онлайн запись на практические занятия с автоинструктором;
- просмотр своего расписания;
- просмотр профиль инструкторов.

Интерфейс программы представлен на рисунке 1.6.

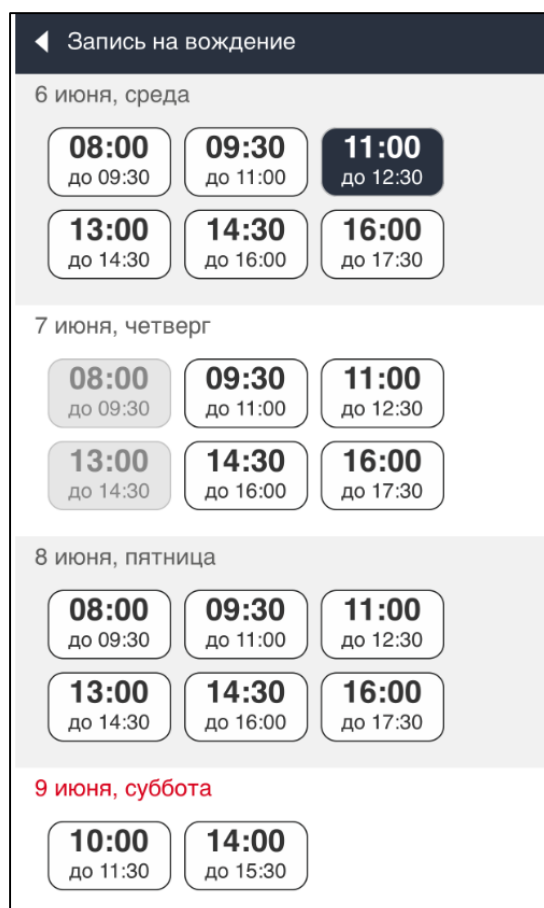


Рисунок 1.6 – Автошкола-контроль

Недостатки «Автошкола-контроль»:

- перегруженный дизайн Автошколы-контроль.

Исходя из анализа аналогов, были выделены требования к программному обеспечению:

- удобный и понятный интерфейс для пользователей с любой компьютерной грамотностью;
- разделение бизнес-логики приложения на клиентскую часть и часть администратора.

## **2 Анализ требований к программному средству и разработка функциональных требований**

### **2.1 Описание инструментов для разработки программного средства**

В ходе разработки данного программного средства используются следующие инструменты:

- объектно-ориентированный язык программирования C#;
- платформа для кроссплатформенной разработки с открытым исходным кодом .NET Framework;
- расширяемый язык разметки XAML;
- система управления базами данных Microsoft SQL Server;
- шаблон проектирования MVVM;
- фреймворк для работы с базой данных Entity Framework 6;
- интегрированная среда разработки Microsoft Visual Studio 2022.

C# — это язык программирования, предназначенный для разработки самых разнообразных приложений, предназначенных для выполнения в среде .NET Framework. Visual C# — это реализация языка C# корпорацией Майкрософт. Поддержка Visual C# в Visual Studio обеспечивается с помощью полнофункционального редактора кода, компилятора, шаблонов проектов, конструкторов, мастеров кода, мощного и удобного отладчика и многих других средств. Библиотека классов .NET Framework предоставляет доступ ко многим службам операционной системы и другим полезным, правильным классам, что существенно ускоряет цикл разработки.

XAML — это декларативный язык разметки. С точки зрения модели программирования .NET Core язык XAML упрощает создание пользовательского интерфейса для приложения .NET Core. Можно создать видимые элементы пользовательского интерфейса в декларативной XAML-разметке, а затем отделить определение пользовательского интерфейса от логики времени выполнения, используя файлы кода программной части, присоединенные к разметке с помощью определений разделяемых классов. Язык XAML напрямую представляет создание экземпляров объектов в конкретном наборе резервных типов, определенных в сборках.

Microsoft SQL Server — система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка.

SQL Server — это основа платформы обработки данных Майкрософт, которая предоставляет надежную и устойчивую производительность (в том числе благодаря технологиям обработки данных в памяти) и помогает быстрее извлечь ценную информацию из любых данных, расположенных как в локальной среде, так и в облаке.



MVVM — шаблон проектирования архитектуры приложения, который позволяет отделить логику приложения от визуальной части, что упрощает тестирование и поддержку приложения.

## **2.2 Описание функциональности программного средства**

На основе анализа были составлены следующие функциональные требования для программного средства для клиента:

- просмотр расписания и запись на занятия;
- просмотр своих занятий и их отмена;
- просмотр профилей инструкторов и оставление отзывов о них;
- просмотр своего профиля и его изменение.

Для администратора:

- просмотр всех занятий и информации о них;
- добавление инструкторов;
- просмотр профилей пользователей и их удаление;
- добавление машины в автопарк.

Для автоинструктора:

- просмотр своих занятий и информации о них;
- просмотр своего профиля и его редактирование.

В графическом материале приведена схема Use-case, описывающая функциональность программного средства для каждого из пользователей.

Из данной схемы видно, что клиент может выполнять операции по поиску свободных занятий, их бронированию и изменений своего профиля, инструктор в свою очередь просматривает расписание и редактирует свой профиль, а администратор может изменять автопарк, просматривать и удалять пользователей, а также добавлять и удалять инструкторов.

### 3 Проектирование программного средства

#### 3.1 Модель базы данных

Для создания программного средства была разработана база данных Planner, состоящая из 8 таблиц. Описания таблиц базы данных представлены в таблицах 3.1 – 3.8:

Таблица 3.1 – Таблица Cars

Имя столбца	Тип данных	Описание
CarID	int	Содержит идентификатор машины
CarName	nvarchar(20)	Содержит название марки машины
ImageIndex	int	Содержит индекс картинки
CategoryName	nvarchar(2)	Содержит название категории машины
CarYear	int	Содержит год выпуска авто

Таблица ClassIntervals предназначена для хранения временных интервалов занятий.

Таблица 3.2 – Таблица ClassIntervals

Имя столбца	Тип данных	Описание
IntervalNumber	int	Порядковый номер интервала времени
TimeInterval	nvarchar(MAX)	Строка с описанием интервала времени

В таблице Categories хранится информация о категориях авто.

Таблица 3.3 – Таблица Categories

Имя столбца	Тип данных	Описание
CategoryName	nvarchar(2)	Имя категорий авто
CategoryDescription	nvarchar(30)	Описание категорий авто

Таблица FeedBacks предназначена для хранения отзывов курсантов, оставленные определенному инструктору.

Таблица 3.4 – Таблица FeedBacks

Имя столбца	Тип данных	Описание
FeedBackID	int	ID отзыва
FeedBackMessage	nvarchar(100)	Содержит сообщение в отзыве
UserID	int	Содержит ID пользователя
InstructorID	int	Содержит ID инструктора

Таблица Users хранит личные данные пользователей.

Таблица 3.5 – Таблица Users

Имя столбца	Тип данных	Описание
UserID	int	Идентификатор пользователя
FIO	nvarchar(30)	ФИО пользователя
BirthDate	datetime	Дата рождения
Login	nvarchar(12)	Логин пользователя
UserEmail	nvarchar(20)	Почта пользователя
ImageIndex	int	Индекс картинки пользователя
UserVK	nvarchar(20)	ВК пользователя
UserPhone	nvarchar(20)	Номер телефона
HashPass	nvarchar(100)	Хэш пароля

Таблица Instructors хранит личные данные инструкторов, а также авто, которые к ним относятся.

Таблица 3.6 – Таблица Instructors

Имя столбца	Тип данных	Описание
InstructorID	int	ID инструктора
FIO	nvarchar(30)	ФИО инструктора
ImageIndex	int	Индекс фотографии
CarID	int	ID автомобиля
InstructorBirth	datetime	Дата рождения
InstructorEMAIL	nvarchar(20)	Почта пользователя
InstructorVK	nvarchar(20)	ВК инструктора
InstructorPhone	nvarchar(20)	Номер телефона
Login	nvarchar(20)	Логин
HashPass	nvarchar(100)	Хэш пароля

Таблица TimeTables хранит информацию о записи пользователя на занятие.

Таблица 3.7 – Таблица TimeTables

Имя столбца	Тип данных	Описание
ClassID	int	ID занятия
DateOfClass	date	Дата
IntervalCode	int	Дата рождения
UserID	int	Почта пользователя
InstructorID	int	ВК инструктора

В таблицу Admins вносится информация о действующих администраторах.

Таблица 3.8 – Таблица Admins

Имя столбца	Тип данных	Описание
AdminID	int	ID Администратора
Login	nvarchar(20)	Логин
Name	nvarchar(30)	Имя
AdminEMAIL	nvarchar(20)	Почта администратора
HashPass	nvarchar(100)	Хэш пароля

Диаграмма базы данных, отображающая связи между таблицами, представлена в графическом материале.

## 3.2 Структура проекта

Программное средство выполнено одним проектом структура которого показана на рисунке 3.2

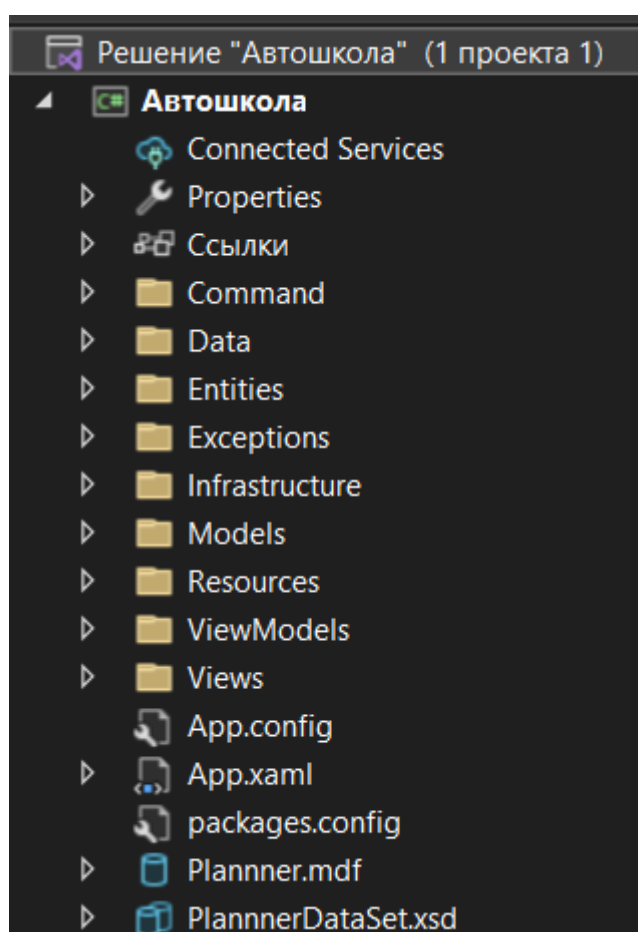


Рисунок 3.2 – Структура проекта

Описание хранимых в папках проекта элементов представлено в таблице 3.9.

Таблица 3.9 – Описание структуры проекта

Элемент	Описание
Папка «Infrastructure»	Содержит элементы инфраструктуры проекта, такие как, конвертеры, атрибуты валидации, валидаторы, классы шифраторы пароля
Папка «Models»	Содержит в себе модели для хранения данных
Папка «Resources»	Содержит шрифты, словари ресурсов
Папка «ViewModels»	Содержит ViewModels для MVVM паттерна
Папка «Views»	Содержит представления, которые отвечают за визуальную часть для MVVM паттерна
Папка «Exceptions»	Содержит пользовательские исключения, генерируемые на сервисе
Папка «Data»	Содержит контекст базы данных
Папка «Command»	Содержит команды
Папка «Entities»	Содержит классы для работы с базой данных
Папка «Resources»	Содержит шрифт, стили, иконки, использующиеся в приложении
App.config	Содержит строки конфигурации приложения
App.xaml	Содержит стартовую точку, с которой начинается выполнение приложения

### 3.3 Структура классов программного средства

Структура классов программного средства представлена в виде диаграммы классов.

Диаграмма классов определяет типы классов системы и различного рода статические связи, которые существуют между ними. На диаграммах классов изображаются также атрибуты классов, операции классов и ограничения, которые накладываются на связи между классами. Вид и интерпретация диаграммы классов существенно зависит от точки зрения (уровня абстракции): классы могут представлять сущности предметной области (в процессе анализа) или элементы программной системы (в процессах проектирования и реализации).

Основными элементами являются классы и связи между ними. Классы характеризуются при помощи атрибутов и операций.

Атрибуты описывают свойства объектов класса. Большинство объектов в классе получают свою индивидуальность из-за различий в их атрибутах и взаимосвязи с другими объектами. Однако, возможны объекты с идентичными значениями атрибутов и взаимосвязей. Т.е. индивидуальность объектов определяется самим фактом их существования, а не различиями в их свойствах. Имя атрибута должно быть уникально в пределах класса. За именем атрибута может следовать его тип и значение по умолчанию.

В данном пункте будут рассматриваться только классы модели приложения без структуры модели представления.

Модель приложения представлена 8 основными классами.

Класс Car определяет автомобиль, который имеет свой идентификатор, фото и название, год выпуска, а также связан с инструктором.

Класс Categorie определяет категорию транспортного средства и предоставляет описание категорий и краткое имя.

Класс Intervals определяет интервалы времени для занятия. Имеет порядковый номер интервала и полный интервал времени.

Класс Feedback определяет отзыв об инструкторе и имеет ссылку на курсанта и инструктора, содержит сообщение и персональный идентификатор.

Класс Users определяет курсантов, имеет ФИО, дату рождения, логин и хеш пароля. Наследуется от BaseUser.

Класс Instructors определяет инструктора, имеет ссылку на Car. Содержит ФИО, дату рождения, логин и хеш пароля. Наследуется от BaseUser.

Класс Admin определяет администратора приложения. Содержит логин и хеш пароля. Наследуется от BaseUser.

Класс TimeTable определяет расписание занятий. Имеет ссылку на ClassInterval, User, Instructor. Имеет поле даты занятия.

На основании данных классов была создана база данных и ее таблицы. База данных с таблицами создавалась с помощью Entity Framework 6.

Диаграмма классов представлена в графической части.

### **3.4 Работа с базой данных**

Вся работа с базой данных целиком возложена на WCF сервис. При разработке приложения использовался Entity Framework 6. Клиент взаимодействует с сервисом посредством вызова методов сервиса. При создании базы данных были заполнены несколько таблиц.

### **3.5 Аутентификация пользователей в системе**

Программное средство подразумевает использование несколькими пользователями с разными правами. Для этого необходима регистрация и аутентификация пользователей в системе. В системе может зарегистрироваться каждый, однако инструкторов может регистрировать только администратор. Данные, введенные пользователями, должны проходить валидацию, а пароль – хешироваться для безопасности.

Схема работы алгоритма регистрации приведена в приложении А, а схема работы алгоритма авторизации – в приложении Б.

### **3.6 Окна приложения**

Набор окон приложения доступных каждому пользователю, будет отличаться, т. к. отличаются их возможности в системе. Для этого необходимо разработать схему переходов для каждого пользователя. Запуск ПО будет начинаться с окна логина, откуда пользователь может перейти к регистрации, или войти и перейти к основному меню. Далее он может записываться на занятия,

просматривать профили и расписание и изменять свои профили и отменять занятия, а также оставлять отзывы, а администратор, в свою очередь, может управлять различными данным. Схемы пользователей представлены на рисунках 3.3-3.5.

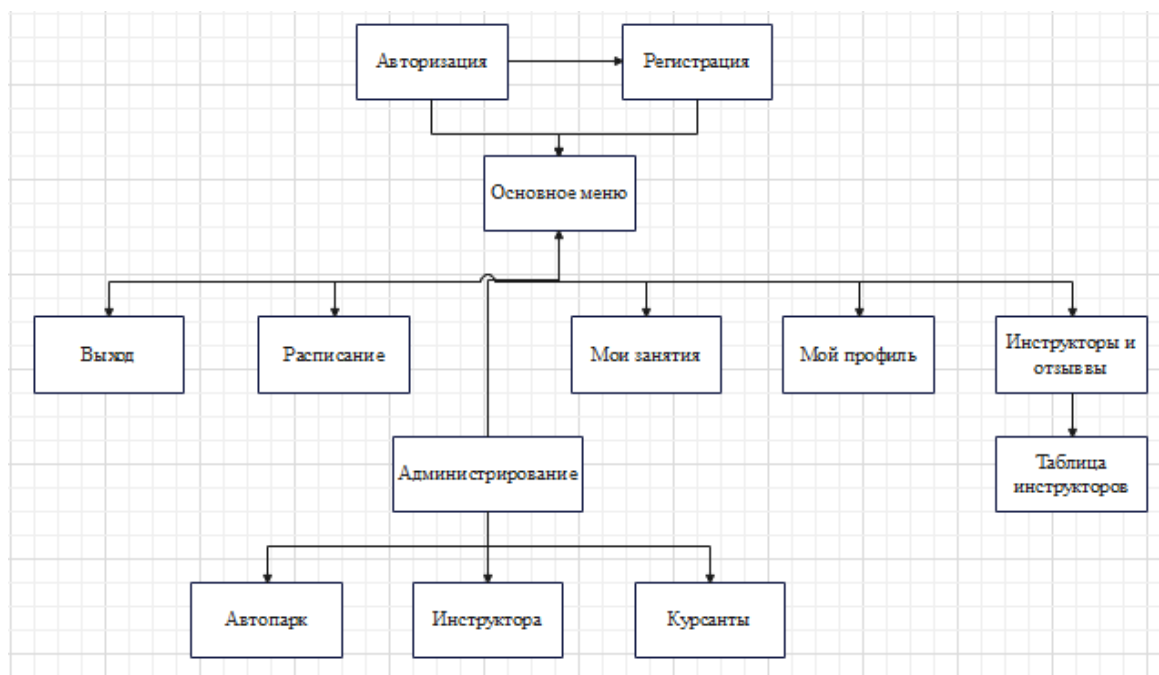


Рисунок 3.3 – Схема окон администратора

Самым привилегированным пользователем является администратор. Так как он может управлять аккаунтами остальных пользователей.

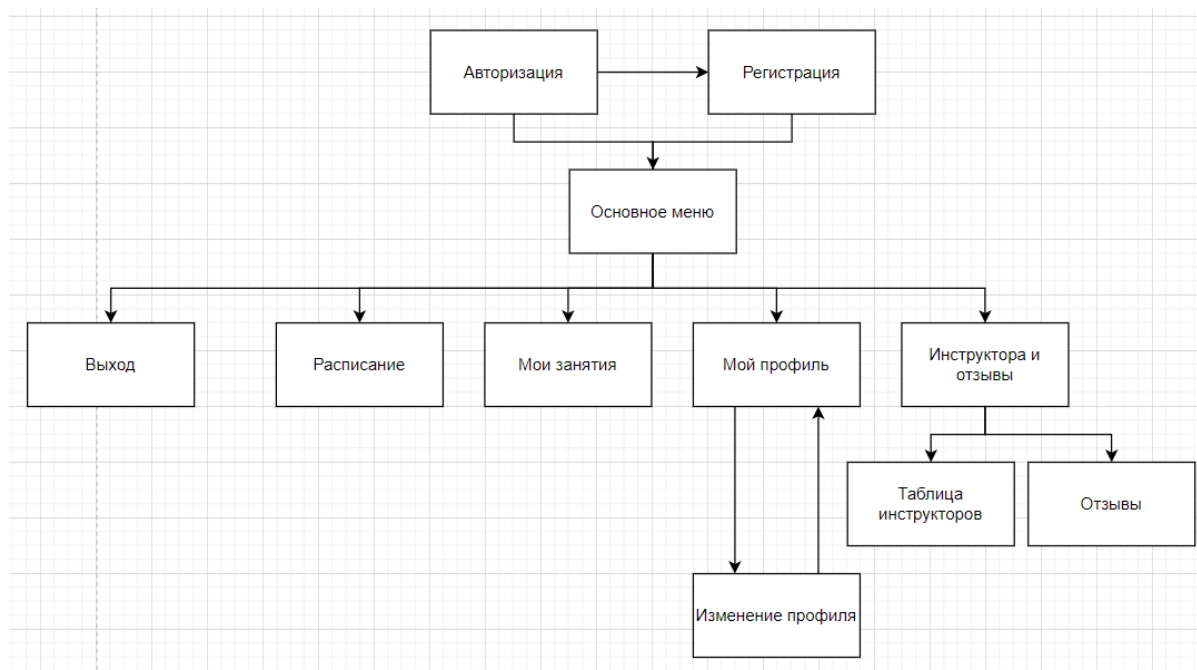


Рисунок 3.4 – Схема окон пользователя

Меньше всего функционала у инструктора.



Рисунок 3.5 – Схема окон инструктора

Диаграмма последовательностей для записи на вождение приведена в графическом материале.



## 4 Реализация программного средства

### 4.1 Реализация MVVM и других шаблонов

Паттерн MVVM реализуется через базовый класс ViewModel (листинг 4.1), наследующий интерфейс INotifyPropertyChanged. Данный интерфейс используется для уведомления представления об изменениях свойств объекта. Реализация классом интерфейса предполагает генерацию события PropertyChanged каждый раз, когда значение свойства объекта изменяется. Такое поведение позволяет привязкам данных отслеживать состояние объекта и обновлять данные пользовательского интерфейса при изменении значения связанного свойства. Также в этом классе определен виртуальный метод Set, который сравнивает значение «до» и значение «после» и, если они совпадают, не переопределяет поле. Это позволяет предотвратить кольцевые обновления свойств и переполнение стека. На основе класса ViewModel сделаны все ViewModels, присутствующие в данном проекте.

```
internal class ViewModel : INotifyPropertyChanged
{
    public event PropertyChangedEventHandler PropertyChanged;

    protected void OnPropertyChanged([CallerMemberName] string
propertyName = null)
    {
        PropertyChanged?.Invoke(this, new
PropertyChangedEventArgs(propertyName));
    }

    protected bool Set<T>(ref T field, T value, [CallerMemberName] string
propertyName = null)
    {
        if (Equals(field, value)) return false;
        field = value;
        OnPropertyChanged(propertyName);
        return true;
    }
}
```

Листинг 4.1 – структура класса ViewModel

Примером реализации потомка данного класса может служить класс OverviewInstructorViewModel (листинг 4.2).

```

class OverviewInstructorViewModel : ViewModel
{
    public OverviewInstructorViewModel(Instructor instance)
    {
        Instance = instance;
        ContentVisibility = Visibility.Visible;
    }
    public OverviewInstructorViewModel()
    {
        ContentVisibility = Visibility.Hidden;
    }

    #region Props

    private Visibility _contentVisibility;

    public Visibility ContentVisibility
    {
        get { return _contentVisibility; }
        set { Set(ref _contentVisibility, value); }
    }

    private Instructor _instance;
    public Instructor Instance
    {
        get { return _instance; }
        set { Set(ref _instance, value); }
    }
    #endregion
}

```

Листинг 4.2 – структура класса OverviewInstructorViewModel

В приложении используется паттерн Command который позволяет инкапсулировать запрос на выполнение определенного действия в виде отдельного объекта. В WPF команды представлены интерфейсом ICommand. В приложении он представлен в виде собственной базовой команды Command (листинг 4.3), от которой наследуются остальные основные команды.

```

internal abstract class Command : ICommand
{
    public event EventHandler CanExecuteChanged
    {
        add => CommandManager.RequerySuggested += value;
        remove => CommandManager.RequerySuggested -= value;
    }
    public abstract bool CanExecute(object parameter);
    public abstract void Execute(object parameter);
}

```

Листинг 4.3 – базовая команда Command

Основной командой, реализованной в проекте, является команда действия LambdaCommand (листинг 4.4), конструктор которой принимает два параметра: первый – это делегат, который вызывается при выполнении команды (например, в ответ на нажатие кнопки). Второй параметр – это делегат, который возвращает логическое значение, указывающее, может ли команда выполниться; если параметр не передан, то команду можно выполнить всегда. Если делегат возвращает false, он отключает кнопку.

```

internal class LambdaCommand : DriverPlanner.Command.Base.Command
{
    private readonly Action<object> _execute;
    private readonly Func<object, bool> _canExecute;
    public LambdaCommand(Action<object> _execute, Func<object, bool>
_canExecute = null)
    {
        this._execute = _execute;
        this._canExecute = _canExecute;
    }
    public override bool CanExecute(object parameter) =>
_canExecute?.Invoke(parameter) ?? true;

    public override void Execute(object parameter) => _execute(parameter);
}

```

Листинг 4.4 – LambdaCommand

Еще одной командой, которая присутствует в проекте, является команда навигации между представлениями – UpdateViewCommand (листинг 4.5), которая принимает в конструктор параметр ViewModel главного окна.

```

class UpdateViewCommand : ICommand
{
    public event EventHandler CanExecuteChanged;
    private MainViewModel _viewModel;
    public UpdateViewCommand(MainViewModel viewModel)
    {
        this._viewModel = viewModel;
    }
    public bool CanExecute(object parameter) => true;
    public void Execute(object parameter)
    {
        if (parameter.ToString() == "Login")
        {
            _viewModel.CurrentViewModel = new
LoginViewModel(_viewModel);
        }
        else if (parameter.ToString() == "Register")
        {
            _viewModel.CurrentViewModel = new
RegisterViewModel(_viewModel);
        }
        else if (parameter.ToString() == "WorkingSpace")
        {
            _viewModel.CurrentViewModel = new
WorkingViewModel(_viewModel);
        }
    }
}

```

Листинг 4.5 – класс UpdateViewCommand

В классе модели базы данных CurrentUserSingleton, который показан на листинге 4.6, реализован шаблон «Одиночка», благодаря чему мы получаем объект контекста только 1 раз, а после при обращении просто его возвращает. Для решения этой проблемы паттерн Singleton возлагает контроль над созданием единственного объекта на сам класс. Доступ к этому объекту осуществляется через статическую функцию-член класса, которая возвращает указатель или ссылку на него. Этот объект будет создан только при первом обращении к методу, а все последующие вызовы просто возвращают его адрес. Для обеспечения уникальности объекта, конструкторы и оператор присваивания объявляются закрытыми.

```

class CurrentUserSingleton
{
    private static ERole _currentRole;
    public static ERole CurrentRole
    {
        get
        {
            return _currentRole;
        }
        set
        {
            _currentRole = value;
        }
    }
    public static BaseUser CurrentUser { get; set; }
    private static readonly CurrentUserSingleton UserSingleton;

    public CurrentUserSingleton GetCurrentUserSingleton() =>
UserSingleton;

    static CurrentUserSingleton() => UserSingleton = new
CurrentUserSingleton();

    private CurrentUserSingleton() { }
}

```

Листинг 4.6 – класс CurrentUserSingleton

## 4.2 Оставление отзывов

Оставление отзывов происходит следующим образом: авторизованный пользователь во вкладке «Наша команда» выбирает понравившегося инструктора, пролистывает профиль в низ и заполняет окошко для записи, далее пользователь нажимает кнопку «Добавить» и отзыв, который он написал, вносится в базу данных в таблицу Feedbacks.

На листинге 4.7 находится код, реализующий добавление отзыва.

```

public ICommand AddFeedbackCommand { get; }
    private bool CanExecuteAddFeedbackCommand(object p)
    {
        return (CurrentUserSingleton.CurrentRole == ERole.USER &&
CurrentInstructor != null && !String.IsNullOrEmpty(FeedbackMsg)
        && !String.IsNullOrWhiteSpace(FeedbackMsg));
    }

    private void OnExecuteAddFeedbackCommand(object p)
    {
        using (var dps = new DriverPlannerService())
        {
            string textFeedback = FeedbackMsg;

            FeedBacks fb = new FeedBacks();
            fb.FeedBackMessage = textFeedback;
            fb.UserID =
((User)CurrentUserSingleton.CurrentUser).UserID;
            fb.InstructorID = CurrentInstructor.InstructorID;

            if (dps.AddFeedback(fb))
            {
                FeedbackList = new
ObservableCollection<FeedBacks>(dps.GetFeedbacks());
                CurrentFeedbacks = new
ObservableCollection<FeedBacks>(FeedbackList.
                Where(t => t.InstructorID ==
CurrentInstructor.InstructorID).Reverse().ToList());
            }
        }
    }
}

```

Листинг 4.7 – Команда, реализующая добавление отзыва

### 4.3 Запись на занятие

Запись на занятия происходит следующим образом: курсант выбирает из выпадающего списка инструктора, в календаре выбирается удобная дата, затем появляется список свободных мест у инструктора с указанным временем, курсант, нажав на подходящее время и кликнув кнопку «Добавить» отправляет запрос в базу данных на проверку доступности записи на эту неделю, если все успешно запись вносится в базу данных в таблицу TimeTables.

На листинге 4.8 находится код реализации записи на занятие.

```

public ICommand PlanTaskCommand { get; }
    private bool CanExecutePlanTaskCommand(object p)
    {
        return (SelectedClass != null && TaskList.Count != 0 &&
SelectedIndexOfTask != -1 && SelectedDate > DateTime.Today) ? true : false;
    }

    private void OnExecutePlanTaskCommand(object p)
    {
        try
        {
            var dps = new DriverPlannerService();
            SelectedClass.UserID =
((User)CurrentUserSingleton.CurrentUser).UserID;
            if
(dps.CheckForTaskLimits(((User)CurrentUserSingleton.CurrentUser).UserID,
SelectedClass.DateOfClass))
            {
                if (dps.PickTask(SelectedClass,
CurrentUserSingleton.CurrentUser as User))
                {
                    SelectedDate = SelectedDate;
                }
            }
        }
        catch (FaultException<InvalidOperationException> ex)
        {
            MessageBox.Show(ex.Message);
        }
        catch(Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
}

```

Листинг 4.8 – Команда, реализующая запись на занятие

Свои занятия можно проверить и отменить в разделе меню «Мои занятия».

## 5 Тестирование, проверка работоспособности и анализ полученных результатов

В курсовом проекте задействуется обработка ошибок, таким образом, что пользователь будет уведомлен о неудачном выполнении запроса к базе данных, или недоступности данных в формах.

При запуске приложения открывается представление с формой для авторизации, куда следует ввести свой логин и пароль. Если логин или пароль введены неверно, то в TextBox будет выведена информация о произошедшей ошибке (рис. 5.1).

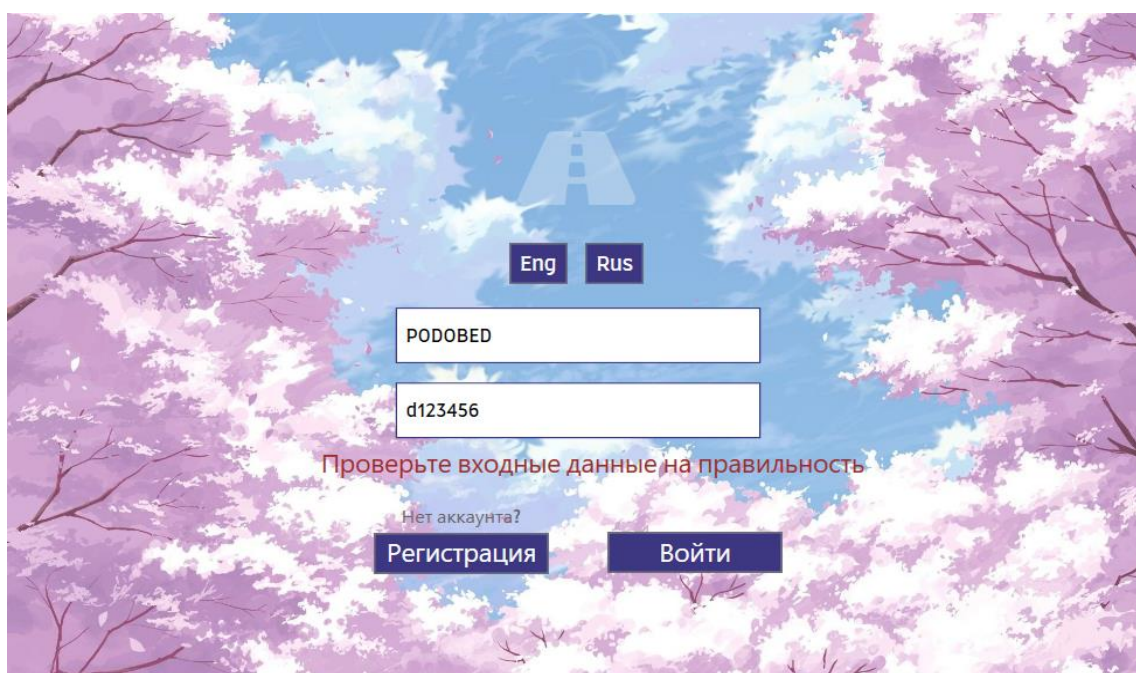


Рисунок 5.1 – При авторизации введены ошибочные данные

Далее рассмотрим представление для регистрации. Оно имеет 6 полей для ввода: ФИО, логин, пароль, повторный ввод пароля, дату рождения и email.

В случае незаполненного поля, пользователь также получит предупреждение о необходимости заполнения данного поля. Это позволяет гарантировать, что все необходимые данные будут предоставлены перед завершением процесса регистрации.

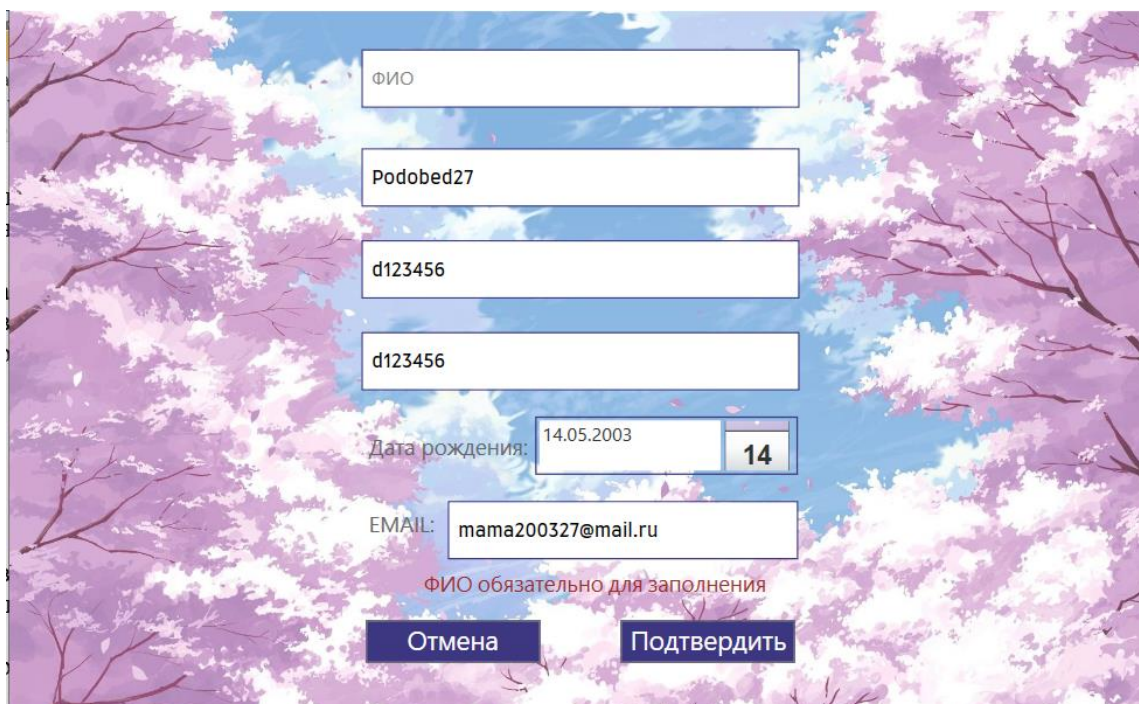
Например, при вводе некорректного формата даты рождения или email адреса, пользователь получит соответствующее уведомление о необходимости исправить ошибку. Это помогает предотвратить возможные проблемы при последующей обработке и использовании введенных данных.

Таким образом, представление для регистрации обеспечивает пользовательскую дружелюбность и безопасность данных, предупреждая о возможных ошибках и гарантируя корректное заполнение всех необходимых полей перед завершением регистрационного процесса.



В приложении предусмотрена обработка следующих ошибок:

Если пользователь оставил поле незаполненным, он увидит сообщение с рисунка 5.2.

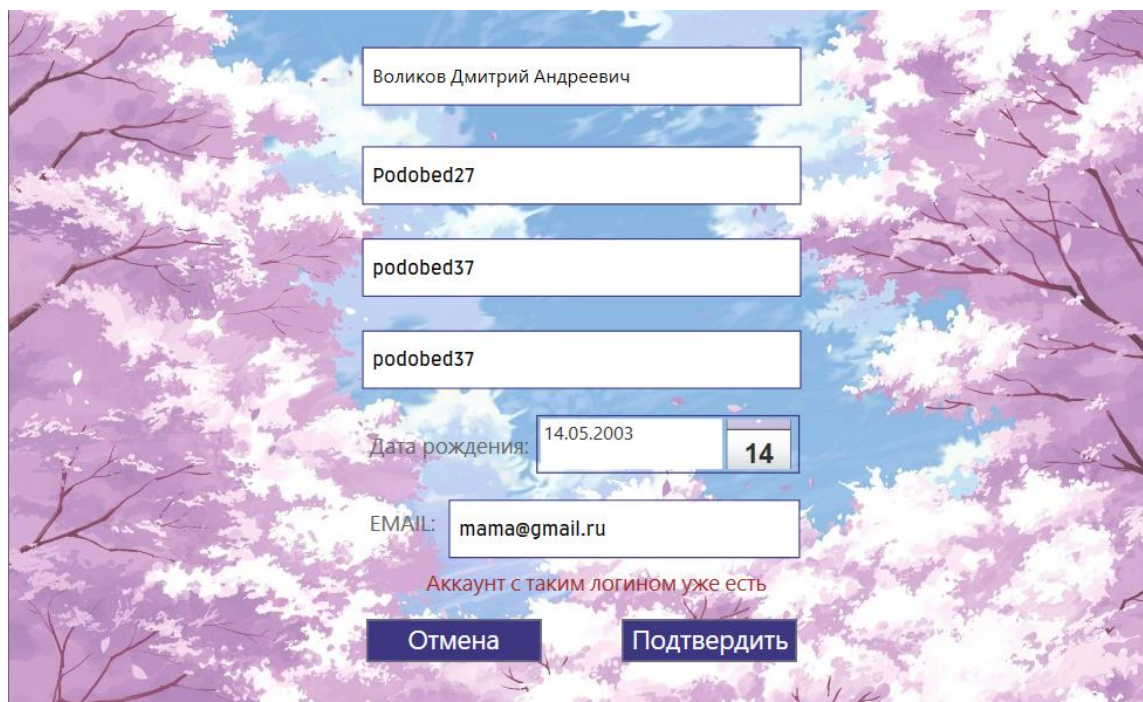


The image shows a registration form with a pink cherry blossom background. The form contains the following fields and elements:

- Field for "ФИО" (Full Name) which is empty.
- Field for "Подобед27" (Username) which contains the text "Подобед27".
- Field for password which contains "d123456".
- Field for password confirmation which contains "d123456".
- Date of birth field with "14.05.2003" in the date picker and "14" in the year dropdown.
- Email field with "mama200327@mail.ru".
- A red error message: "ФИО обязательно для заполнения" (Full name is required for completion).
- Buttons: "Отмена" (Cancel) and "Подтвердить" (Confirm).

Рисунок 5.2 – При регистрации есть пустые поля

Если пользователь попытается занять логин, который уже существует в базе данных, он увидит сообщение с рисунка 5.3.

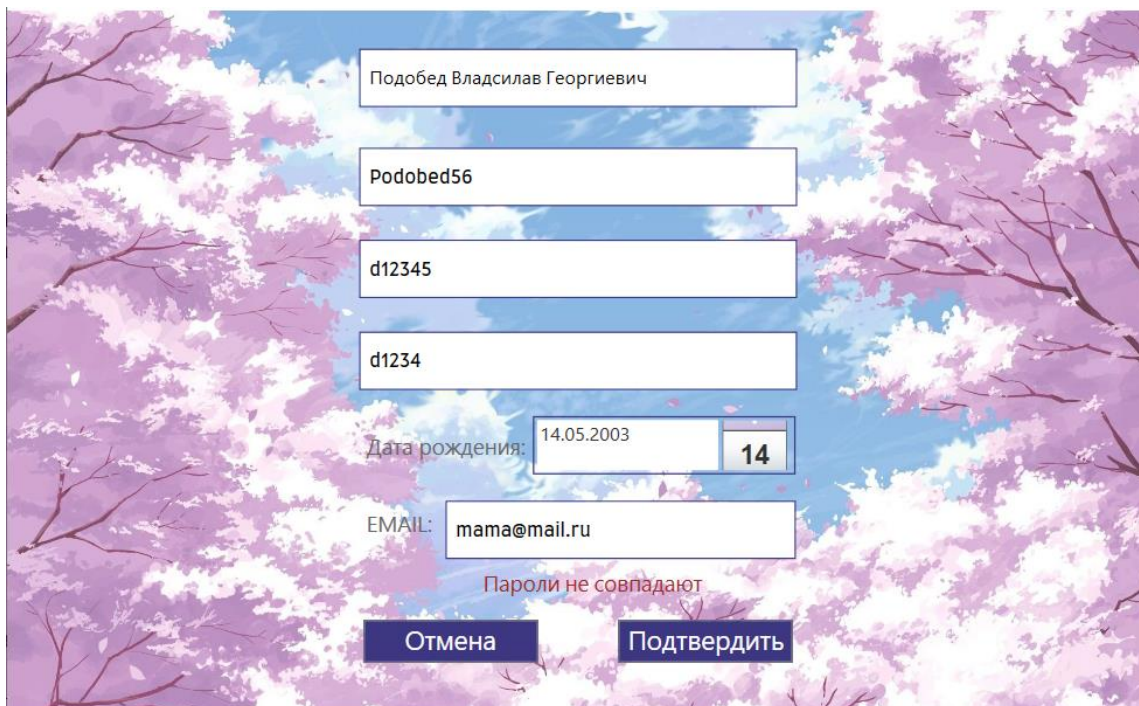


The image shows the same registration form as in Figure 5.2, but with the following changes:

- The "ФИО" field now contains "Воликов Дмитрий Андреевич".
- The password field now contains "podobed37".
- The password confirmation field now contains "podobed37".
- The email field now contains "mama@gmail.ru".
- A red error message: "Аккаунт с таким логином уже есть" (Account with this login already exists).
- The "Подтвердить" button is disabled.

Рисунок 5.3 – При регистрации выбран уже занятый логин

Если пользователь при регистрации введет пароли, которые не совпадают, приложение ему подскажет сообщением с рисунка 5.4.



Подобед Владислав Георгиевич

Podobed56

d12345

d1234

Дата рождения: 14.05.2003 14

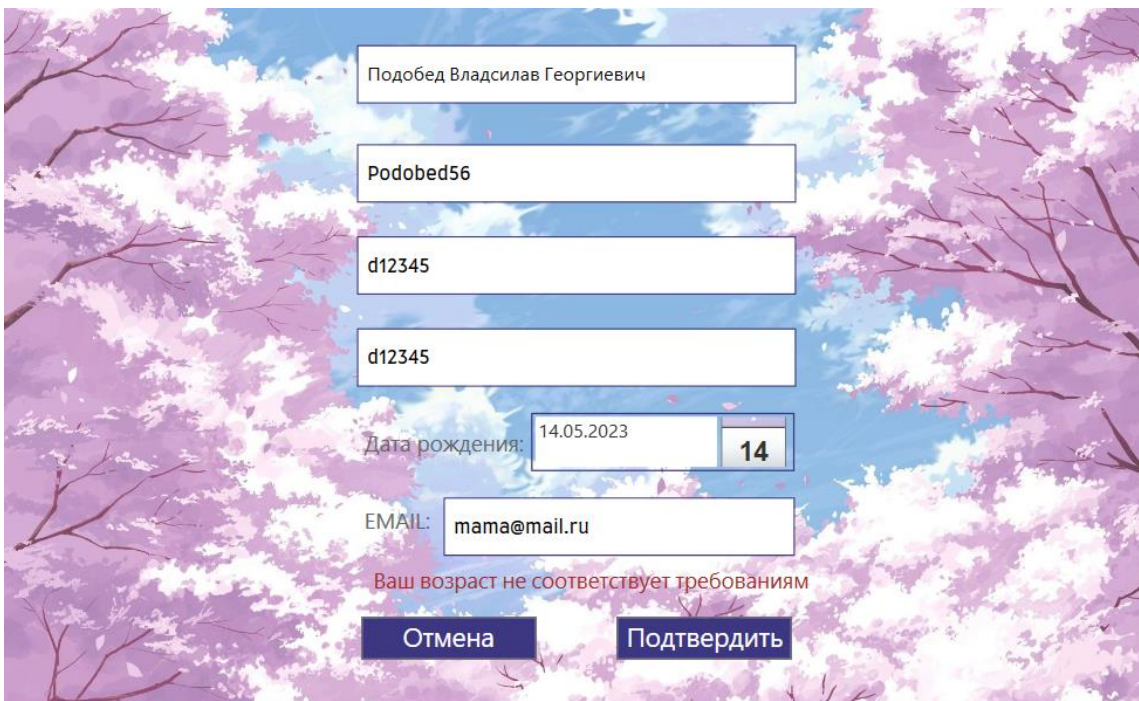
EMAIL: mama@mail.ru

Пароли не совпадают

Отмена Подтвердить

Рисунок 5.4 – При регистрации пользователь ошибся в паролях

Если пользователь при регистрации введет возраст младше 16 или старше 100 лет, приложение ему подскажет сообщением с рисунка 5.5.



Подобед Владислав Георгиевич

Podobed56

d12345

d12345

Дата рождения: 14.05.2023 14

EMAIL: mama@mail.ru

Ваш возраст не соответствует требованиям

Отмена Подтвердить

Рисунок 5.5 – При регистрации пользователь не подходит по возрасту



Если пользователь при записи на занятие выберет больше 3 занятий на одной неделе, приложение ему подскажет сообщением с рисунка 5.6.

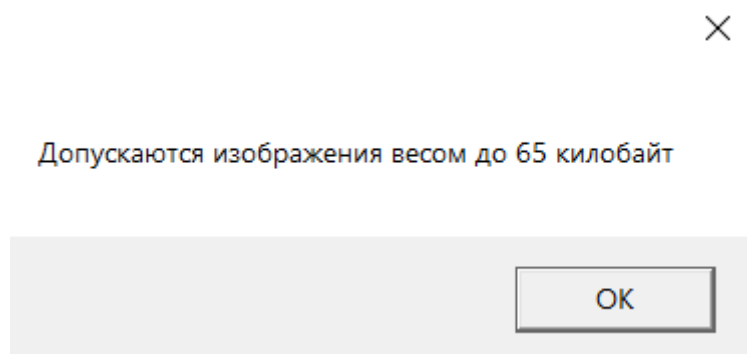


Рисунок 5.6 – При записи превышен лимит занятий

Если пользователь в профиле решит сменить фото, и оно большое по размеру, приложение ему подскажет сообщением с рисунка 5.7.

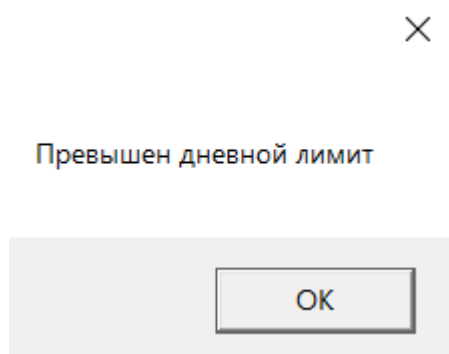


Рисунок 5.7 – При записи превышен лимит занятий

Так же были проведены другие проверки, которые прошли успешно.

## 6 Методика использования программного средства

### 6.1 Регистрация и авторизация

При запуске приложения запускается главное окно, в которое выводится представление с формой для авторизации. Если у пользователя еще нет аккаунта, ему следует нажать на кнопку «Регистрация», которая его перенаправит на представление с формой для регистрации. Представление авторизации представлено на рисунке 6.1.

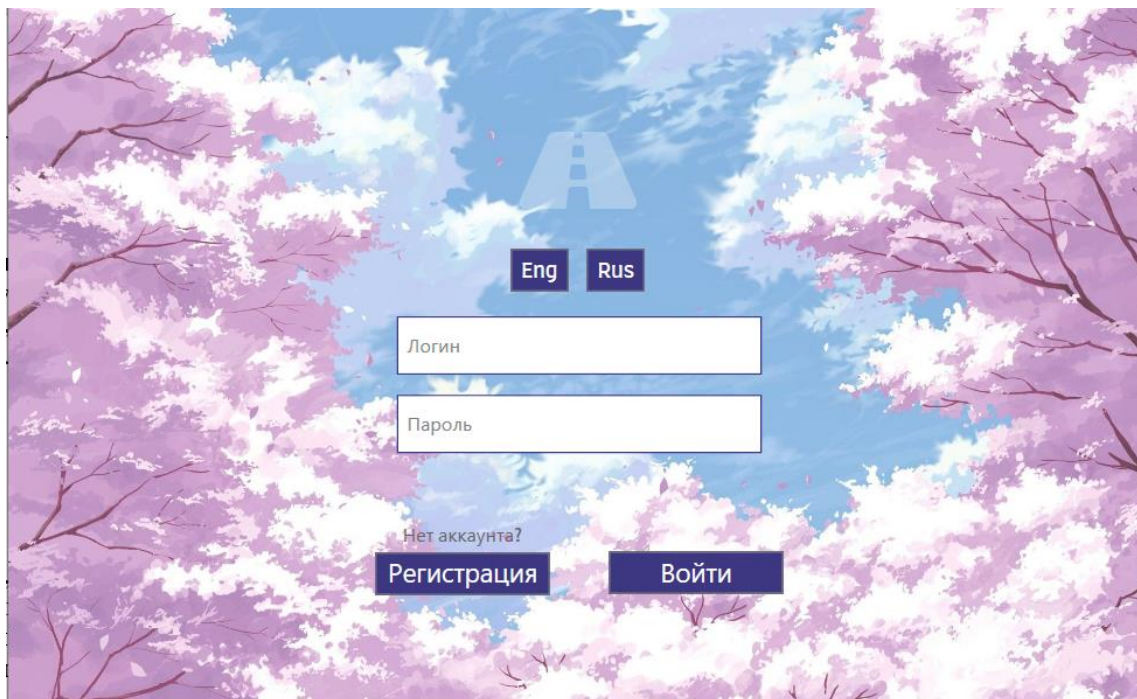
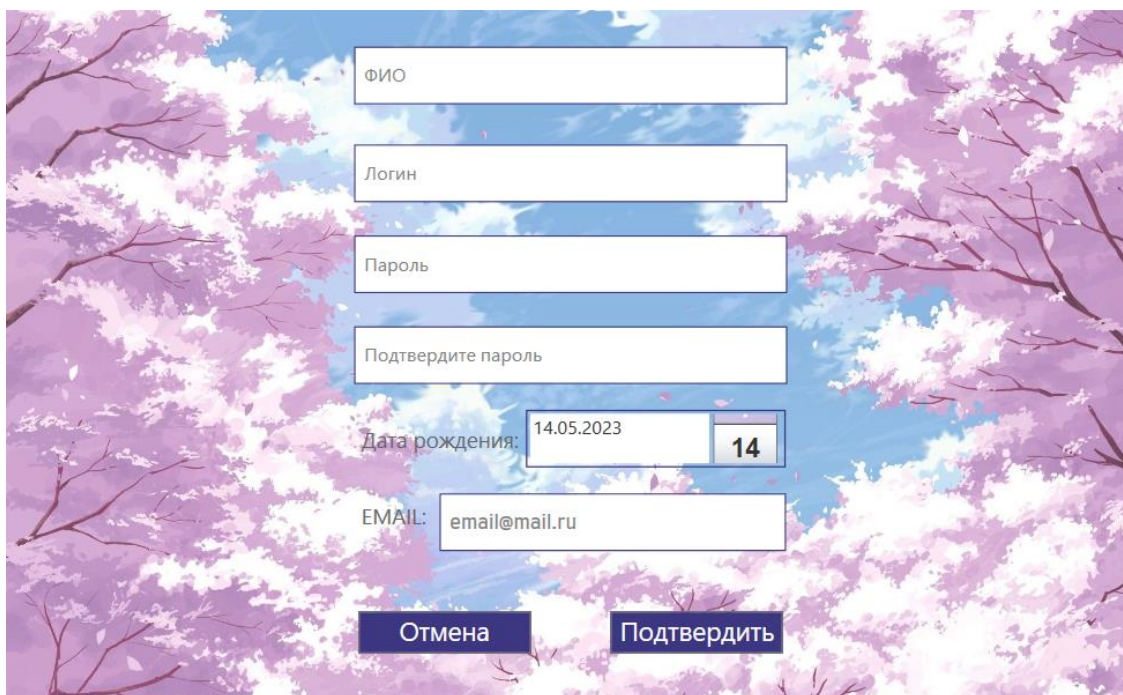


Рисунок 6.1 – Представление авторизации

Если была нажата кнопка «Регистрация», в открывшемся представлении следует ввести все данные о регистрируемом пользователе.

В случае, если кнопка «Регистрация» была нажата ошибочно или пользователь передумал регистрироваться, есть возможность нажать на кнопку «Отмена». Это позволит вернуться к представлению авторизации, где пользователю будет предоставлена возможность войти в систему существующей учетной записью.

Для более наглядного представления процесса регистрации, на рисунке 6.2 изображено соответствующее представление, которое может содержать поля ввода для различных данных о пользователе, а также кнопки для подтверждения регистрации или отмены операции. Это представление может варьироваться в зависимости от конкретной системы или приложения, однако его основная цель - обеспечить удобный интерфейс для регистрации новых пользователей.



Registration form with the following fields and buttons:

- ФИО
- Логин
- Пароль
- Подтвердите пароль
- Дата рождения: 14.05.2023 (Day selector: 14)
- EMAIL: email@mail.ru
- Buttons: Отмена, Подтвердить

Рисунок 6.2 – Представление регистрации

## 6.2 Домашнее представление

При успешной авторизации пользователя приложение переключит представление с авторизации на домашнее. На рисунке 6.3. показано то, что новый пользователь увидит после своей первой авторизации.

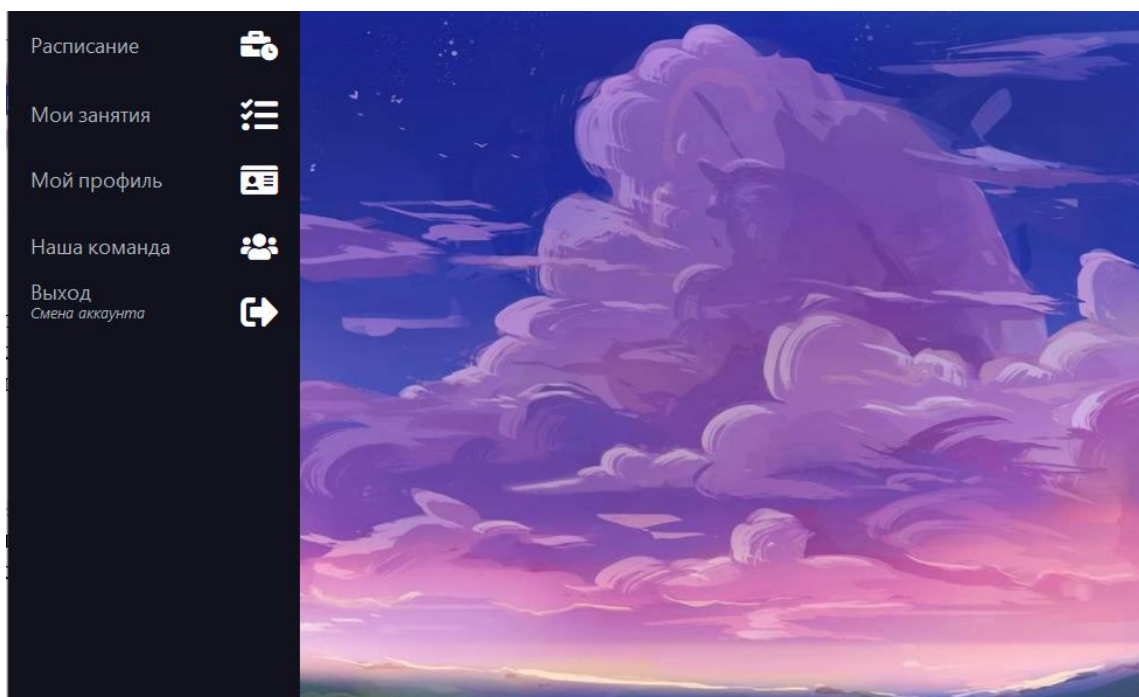


Рисунок 6.3 – Домашнее представление после авторизации

Если пользователь желает записаться на занятие, ему следует нажать в меню на кнопку «Расписание», ему откроется окно записи, как можно увидеть на рисунке 6.4.

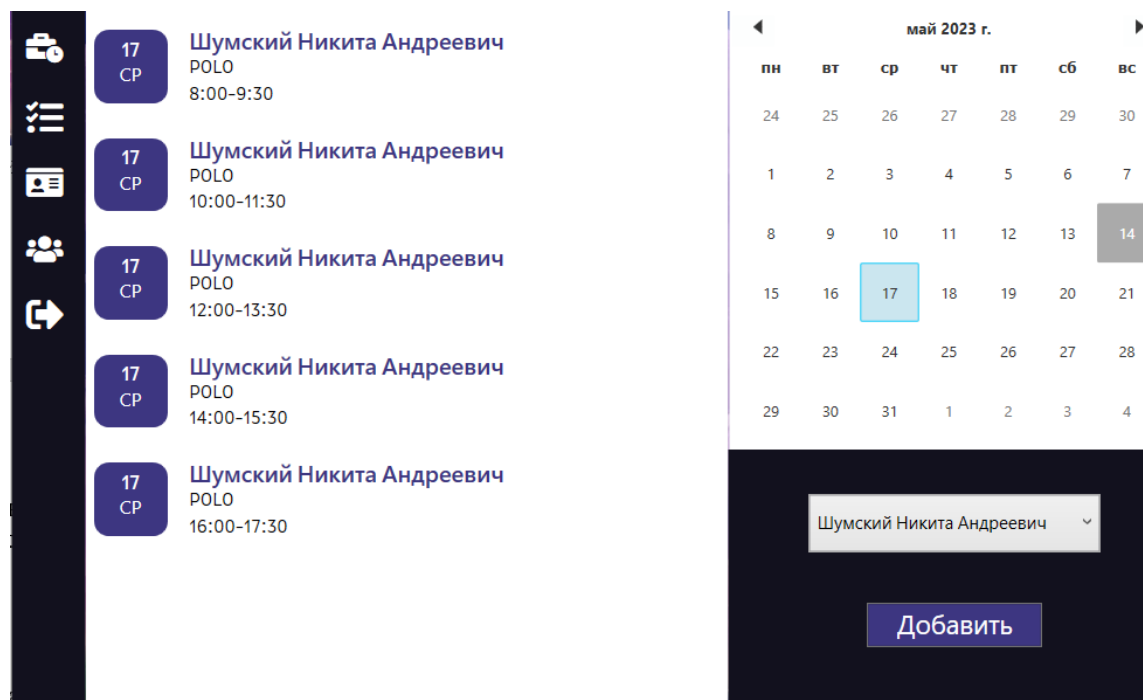


Рисунок 6.4 – Расписание

Для записи на занятие ему нужно выбрать инструктора из выпадающего списка, подходящую дату в календаре, после этого откроется список «окошек» инструктора. Нужно нажать на интересующее нас время и нажать кнопку «Добавить», если выбранное время пропадёт из списка – вы успешно записаны.

Чтобы увидеть все ваши записи на занятия в меню нажмите «Мои занятия», откроется список всех ваших записей. Если нужно отменить занятие, то при нажатии на него откроется информация об этом занятии, где внизу есть кнопка «Удалить», нажав на нее занятие отменится. Это можно увидеть на рисунке 6.5.



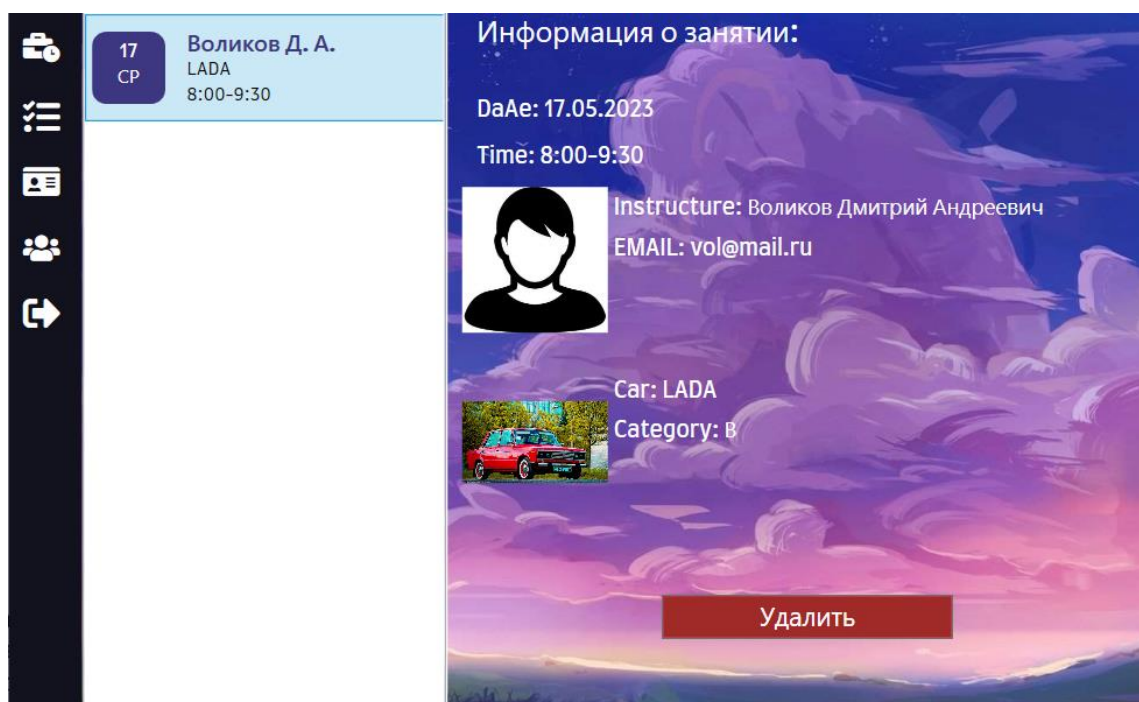


Рисунок 6.5 – Мои занятия

Чтобы увидеть свой профиль, в меню нужно выбрать «Мой профиль». Если нужно, пользователь может изменить свой профиль нажав кнопку «Изменить». Это изображено на рисунке 6.6.

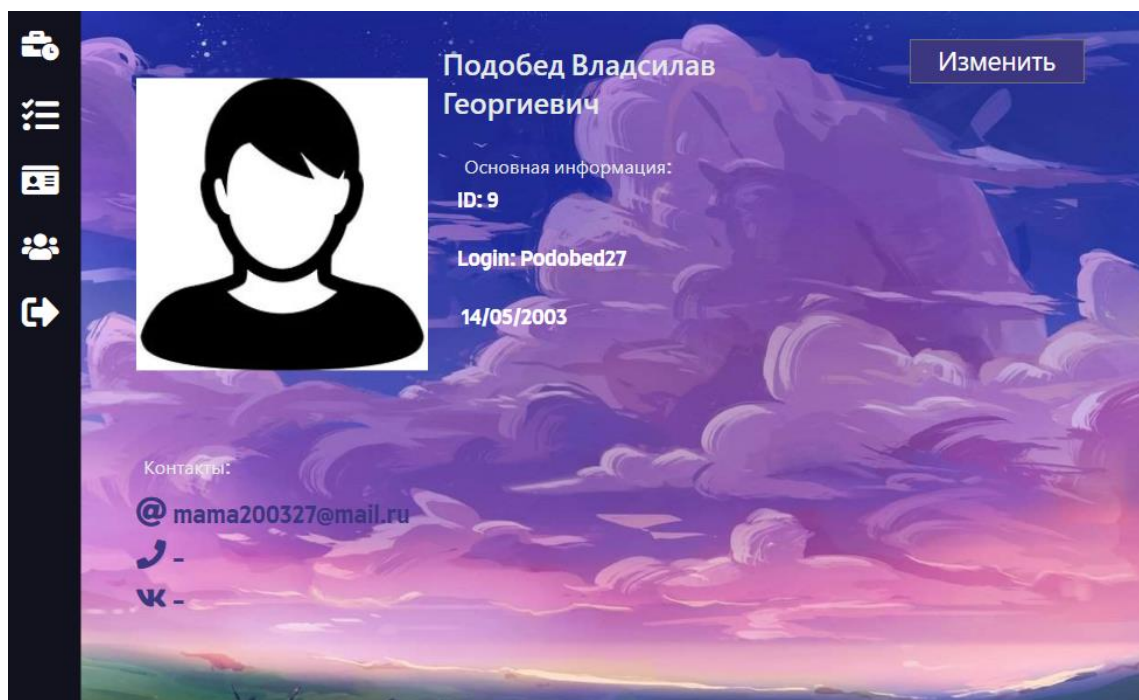


Рисунок 6.6 – Мой профиль

Чтобы просмотреть всех инструкторов и оставить отзыв, в меню выберите «Наша команда». Для удобства визуализации этого шага, на рисунке 6.7 представлено, как найти соответствующий раздел в меню.

Затем проскрольте вниз до тех пор, пока не увидите раздел под названием «Наша команда». Нажмите на этот раздел, чтобы открыть страницу, на которой будут представлены все доступные инструкторы. Вы сможете просмотреть информацию о каждом инструкторе. После того, как вы ознакомитесь с инструкторами, вы также можете оставить свой отзыв о них, чтобы поделиться своими впечатлениями с другими пользователями.

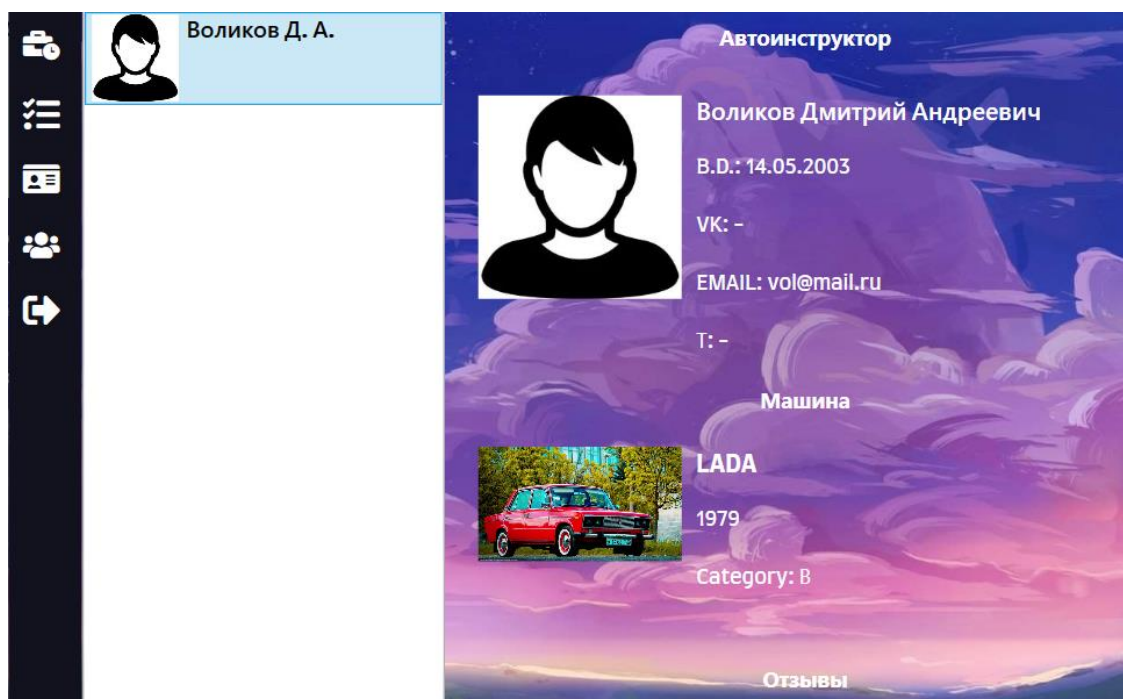


Рисунок 6.7 – Наша команда

### 6.3 Администрирование

При успешной авторизации администратора приложение переключит представление с авторизации на домашнее. В домашнем представлении для администратора доступны все возможности обычного пользователя, а также кнопки «Пользователи», «Инструкторы», «Автопарк» в меню. При нажатии на кнопку «Пользователи» откроется представление, в котором администратор сможет увидеть список всех пользователей, имеющих в базе данных. Администратор также может с помощью клика мышкой по строке с нужным пользователем увидеть профиль и удалить его.

В представлении списка пользователей администратору должна быть доступна информация о каждом пользователе, такая как имя, фамилия, дата рождения, адрес электронной почты, телефон и другие данные, которые могут быть полезны для администрирования приложения.

Если авторизация администратора не удалась, приложение должно отобразить соответствующее сообщение об ошибке и оставаться на странице



авторизации, давая возможность пользователю повторить попытку. Это проиллюстрировано на рисунке 6.8.

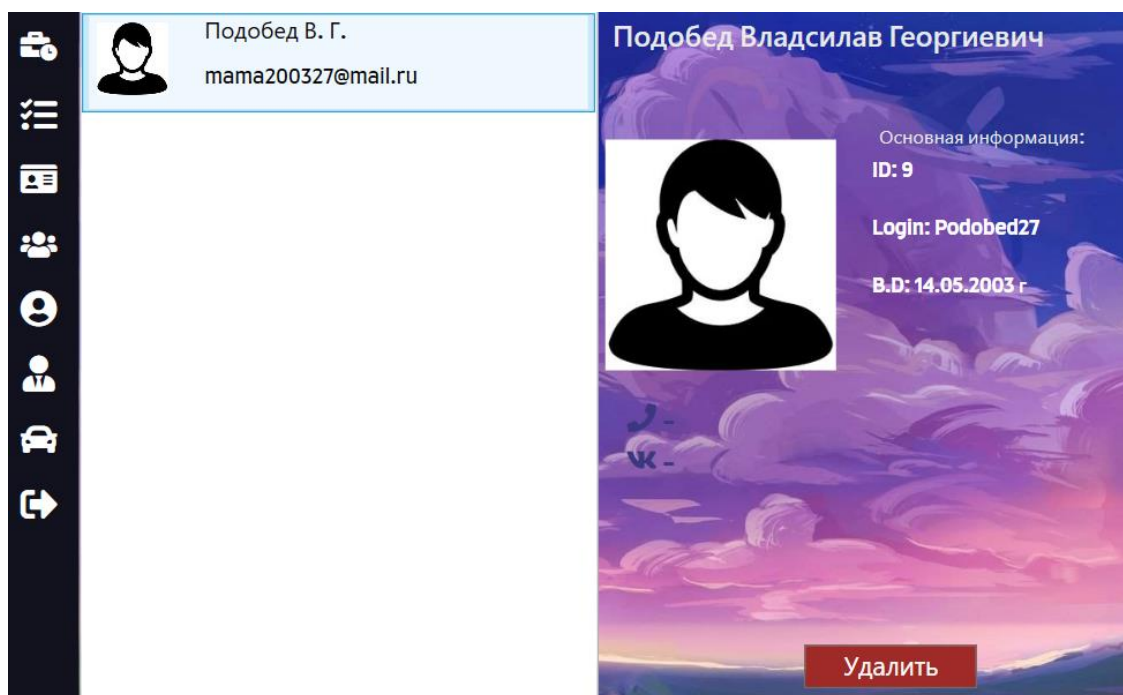


Рисунок 6.8 – Список пользователей, видимый только администратору

При нажатии на кнопку «Инструкторы» откроется представление, в котором можно увидеть список всех автоинструкторов, также их можно добавлять и удалять. Это изображено на рисунке 6.9.

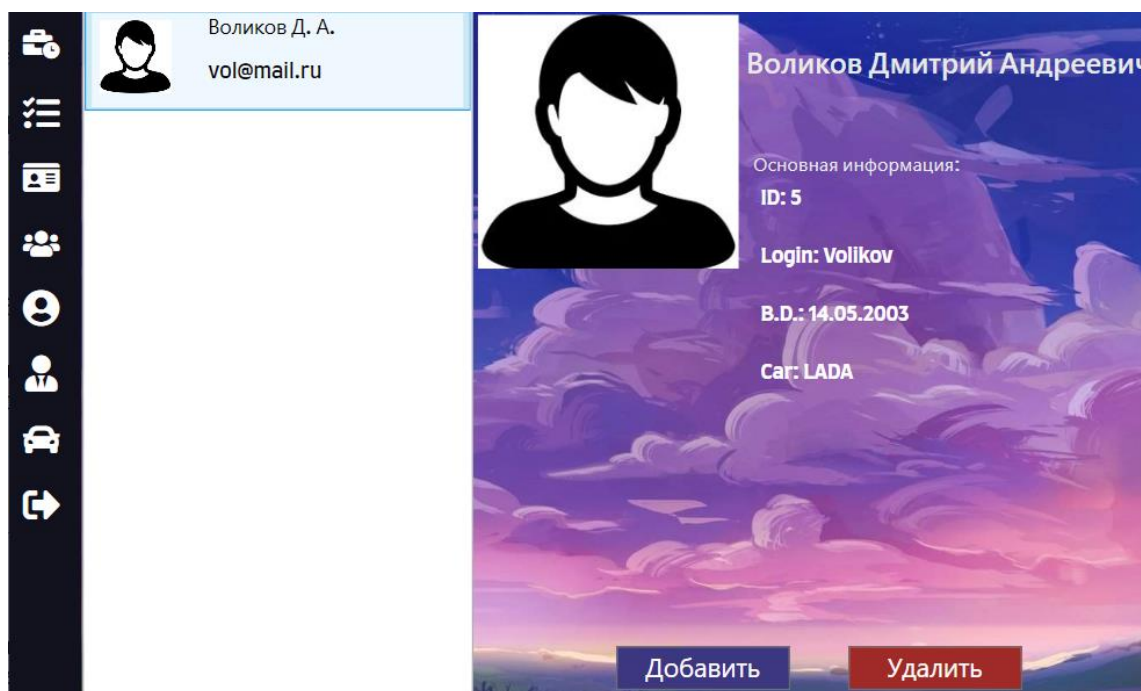


Рисунок 6.9 – Инструкторы

При нажатии на кнопку «Автопарк» откроется представление, в котором можно увидеть список всех авто, также их можно добавлять и удалять. Это изображено на рисунке 6.10.

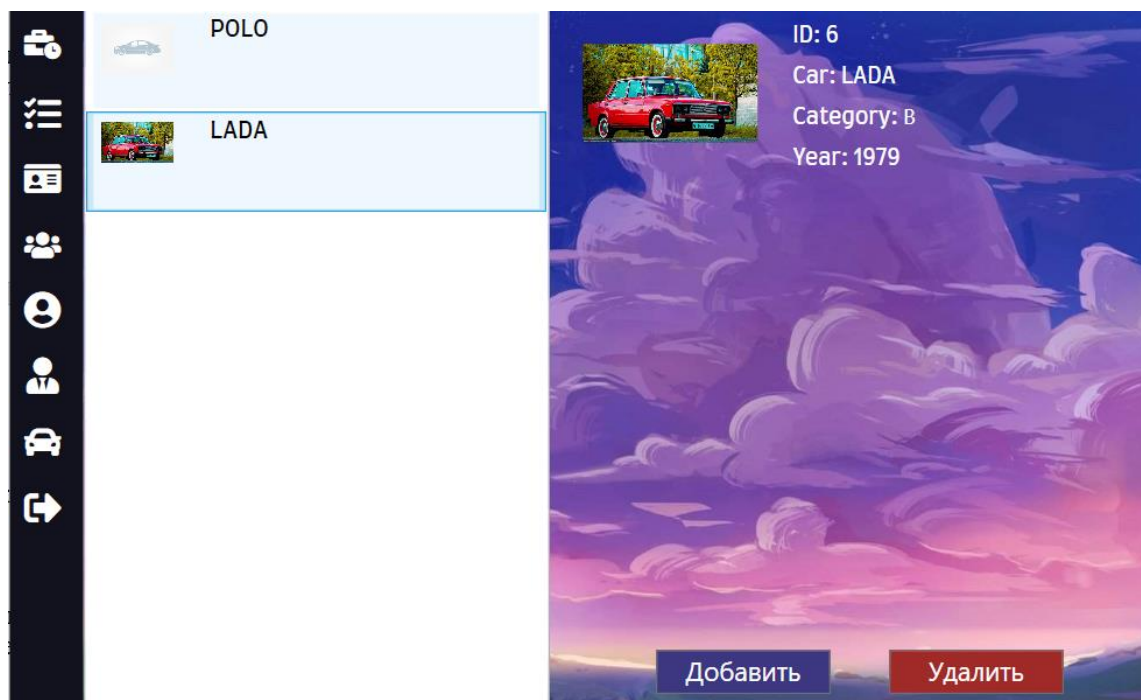


Рисунок 6.10 – Автопарк

#### 6.4 Смена пользователя

При желании сменить пользователя или выйти из аккаунта, пользователю стоит нажать на кнопку «Выход» в меню.

## Заключение

При выполнении данного курсового проекта было разработано программное средство для автошкол, которое позволяет пользователям записываться на практические занятия по вождению, без контакта с инструктором в виде звонка или переписки в мессенджере. Это позволяет пользователю экономить свое время и деньги, а вся нужная информация находится в одном приложении. Все данные упорядочены и визуализированы, что помогает ориентироваться при планировании своего дня, чтобы было время для вождения. Для разработки приложения, которое полностью бы удовлетворяло запросы пользователя, был проведен анализ аналогов для выявления сильных и слабых сторон других программных средств и разработки функциональных требований, проектирование и разработка приложения.

Исходя из анализа аналогов, были выделены требования к программному обеспечению:

- удобный и понятный интерфейс для пользователей с любой компьютерной грамотностью;
- разделение бизнес-логики приложения на клиентскую часть и часть администратора.

Приложение было протестировано, а также проверено на полноту выполняемых операций.

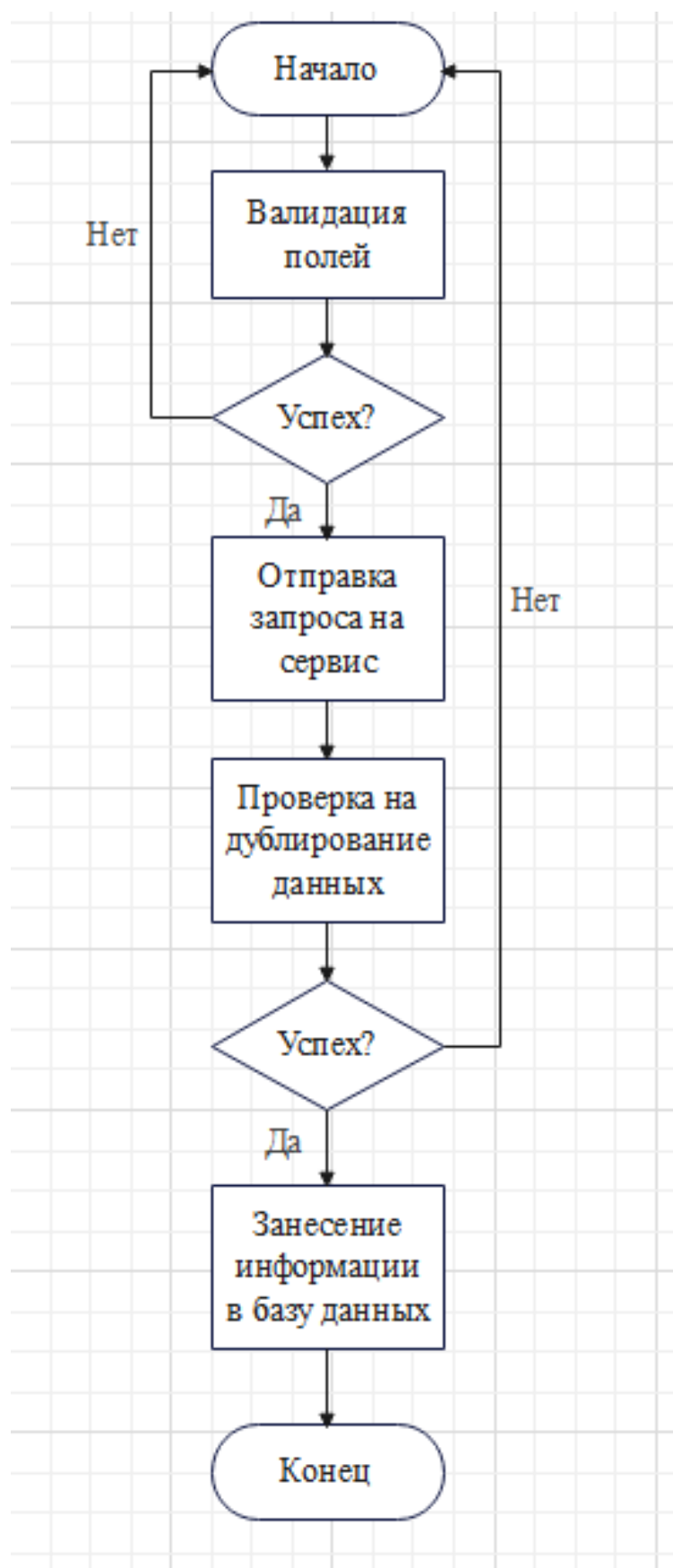
Разработанное приложение удовлетворяет всем требованиям, предъявленным в задании. В дальнейшем возможно оставление отзывов о инструкторе. Ошибок на стадии разработки выявлено не было, так как приложение работает без ошибок.

Также в процессе выполнения данного курсового проекта были закреплены навыки в программировании на объектно-ориентированном языке C#, усовершенствованы навыки в работе с платформой для кроссплатформенной разработки с открытым исходным кодом .NET Framework, ознакомление с созданием десктопных приложений на расширяемом языке разметки WPF, использовании фреймворка для работы с базой данных Entity Framework Core 6.1.0, работа с современными паттернами MVVM и Command, проектирование базы данных и реализация их в системе управления базами данных Microsoft SQL Server 2019.

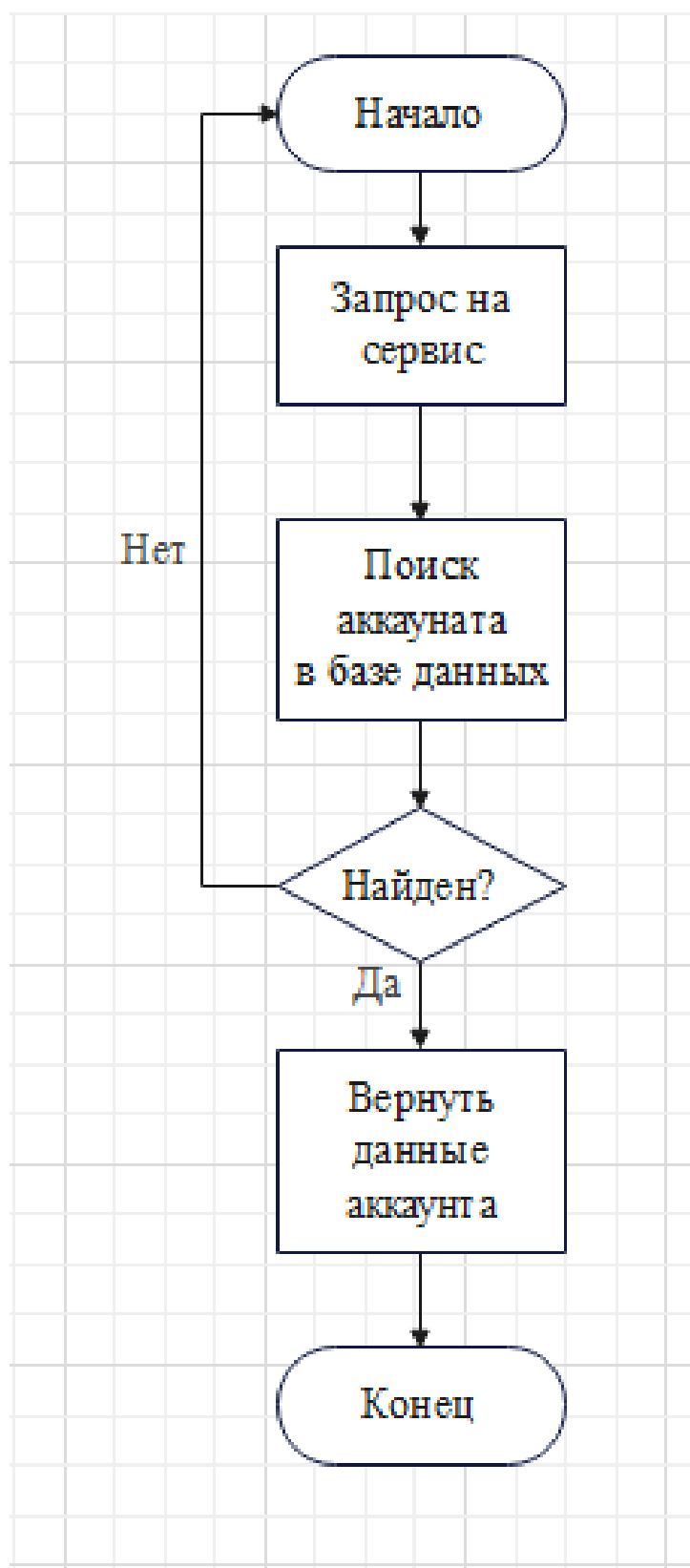
## Список литературы

1. docs.microsoft.com Сайт о программировании [Электронный ресурс] / Режим доступа: <https://docs.microsoft.com>.
2. Пацей, Н.В. Курс лекций по языку программирования С# / Н.В. Пацей. – Минск: БГТУ, 2016. – 175 с.
3. METANIT.COM Сайт о программировании [Электронный ресурс] / Режим доступа: <https://metanit.com>.

## Приложение А



## Приложение Б



## Use-case диаграмма

