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 ПРОЕКТ:**

по дисциплине «Объектно-ориентированные технологии программирования и стандарты проектирования»

Тема Программное средство тату-салона «PL TATTOO»

Исполнитель

студент (ка) 2 курса группы 4 Глушеня Полина Андреевна

(Ф.И.О.)

Руководитель работы ассистент Чистякова Ю. А.

(учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой

Председатель Пацей Н.В.

(подпись)

Минск 2023

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования   
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий   
Кафедра программной инженерии

Утверждаю

Заведующий кафедрой ПИ

\_\_\_\_\_\_\_\_\_\_\_\_ \_\_ Н.В. Пацей

подпись инициалы и фамилия

“\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2023г.

**ЗАДАНИЕ**

**к курсовому проектированию**

**по дисциплине** "Объектно-ориентированные технологии программирования и стандарты проектирования"

|  |  |
| --- | --- |
| Специальность: 1-40 01 01Программное обеспечение информационных технологий  Студент: Глушеня Полина Андреевна | Группа:\_\_4\_\_ |
| **Тема: Программное средство «Тату-салон»** | |

**1. Срок сдачи студентом законченной работы**: "20 мая 2023 г."

**2. Исходные данные к проекту:**

**2.1**. Функционально ПС поддерживает:

* Функции администратора сервиса:
  + Поддерживать работу c базой данных;
  + Регистрировать бронирования сеансов;
  + Выполнять поисковые запросы, фильтрацию;
  + Техническая поддержка пользователей.
* Функции клиента:
  + Выполнять регистрацию и авторизацию;
  + Выполнять поисковые запросы, фильтрацию;
  + Добавлять эскиз в избранное;
  + Заполнять форму бронирования сеанса по заданным формам;
  + Оставлять отзывы.

**2.2.** При выполнении курсового проекта необходимо использовать принципы проектирования ООП. Приложение разрабатывается под ОС Windows и представляет собой настольное приложение (desktop). Отображение, бизнес логика должны быть максимально независимы друг от друга для возможности расширения. Диаграммы вариантов использования, классов реализации задачи, взаимодействия разработать на основе UML. Язык разработки проекта – C#. Управление программой должно быть интуитивно понятным и удобным. При разработке использовать несколько наиболее подходящих шаблонов проектирования ПО.

**3. Содержание расчетно-пояснительной записки**

(перечень вопросов подлежащих разработке)

* Введение
* Постановка задачи и обзор литературы (алгоритмы решения, обзор прототипов, актуальность задачи)
* Проектирование архитектуры проекта (структура модулей, классов).
* Разработка функциональной модели и модели данных ПС (выполняемые функции)
* Тестирование
* Заключение
* Список используемых источников
* Приложения

**4. Форма представления выполненной курсовой работы:**

* + Теоретическая часть курсового проекта должны быть представлены в формате docx. Оформление записки должно быть согласно выданным правилам.
  + Листинги программы представляются в приложении.
  + Пояснительную записку, листинги, проект (инсталляцию проекта) необходимо загрузить диск, указанный преподавателем.

***Календарный план***

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Наименование этапов курсового проекта | Срок выполнения этапов проекта | Примечание |
| 1 | Введение | 19.02.2023 |  |
| 2 | Аналитический обзор литературы по теме проекта. Изучение требований, определение вариантов использования | 12.03.2023 |  |
| 3 | Анализ и проектирование архитектуры приложения (построение диаграмм, проектирование бизнес-слоя, представления и данных) | 26.03.2023 |  |
| 4 | Проектирование структуры базы данных. Разработка дизайна пользовательского интерфейса | 2.04.2023 |  |
| 5 | Кодирование программного средства | 23.04.2023 |  |
| 6 | Тестирования и отладка программного средства | 30.04.2023 |  |
| 7 | Оформление пояснительной записки | 7.05.2023 |  |
| 9 | Защита проекта | 20.05.2023 |  |

**5. Дата выдачи задания** 12.02.2023

Руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Чистякова Ю.А.

(подпись)

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Глушеня П.А.

(дата и подпись студента)

**Содержание**

[ВВЕДЕНИЕ 6](#_Toc135489332)

[1 Аналитический обзор литературы и формирование требований 7](#_Toc135489333)

[1.1 Анализ прототипов 7](#_Toc135489334)

[1.2 Требования к проекту 10](#_Toc135489335)

[2 Анализ требований к программному средству и разработка функциональных требований 11](#_Toc135489336)

[2.1 Описание средств разработки 11](#_Toc135489337)

[2.1.1 Microsoft Visual Studio 2022 11](#_Toc135489338)

[2.1.2 Программная платформа .NET 6 11](#_Toc135489339)

[2.1.3 Язык программирования C# 11](#_Toc135489340)

[2.1.4 Технология WPF 11](#_Toc135489341)

[2.1.5 Расширяемый язык разметки XAML 12](#_Toc135489342)

[2.1.6 Технология Entity Framework 12](#_Toc135489343)

[2.1.7 MS SQL Server 12](#_Toc135489344)

[2.1.8 Паттерн MVVM 13](#_Toc135489345)

[2.2 Спецификация функциональных требований к программному средству 13](#_Toc135489346)

[2.3 Спецификация функциональных требований 14](#_Toc135489347)

[3 Проектирование программного средства 15](#_Toc135489348)

[3.1 Общая структура 15](#_Toc135489349)

[3.2 Взаимоотношение между классами 17](#_Toc135489350)

[3.3 Модель базы данных 17](#_Toc135489351)

[3.4 Проектирование архитектуры приложения 20](#_Toc135489352)

[3.5 Проектирование последовательностей проекта 20](#_Toc135489353)

[4 Реализация программного средства 22](#_Toc135489354)

[4.1 Основные классы программного средства 22](#_Toc135489355)

[4.2 Авторизация 22](#_Toc135489356)

[4.3 Регистрация 22](#_Toc135489357)

[4.4 Запись на сеанс 22](#_Toc135489358)

[4.5 Добавление в избранное 23](#_Toc135489359)

[5 Тестирование, проверка работоспособности и анализ полученных результатов 24](#_Toc135489360)

[5.1 Тестирование регистрации 24](#_Toc135489361)

[5.2 Тестирование отправки сообщения и работы с сообщениями 25](#_Toc135489362)

[5.3 Тестирование панели администратора 26](#_Toc135489363)

[6 Руководство по установке и использованию 27](#_Toc135489364)

[ЗАКЛЮЧЕНИЕ 32](#_Toc135489365)

[Список использованных источников 33](#_Toc135489366)

[ПРИЛОЖЕНИЕ А 34](#_Toc135489367)

[ПРИЛОЖЕНИЕ Б 35](#_Toc135489368)

[ПРИЛОЖЕНИЕ В 36](#_Toc135489369)

[ПРИЛОЖЕНИЕ Г 37](#_Toc135489370)

# ВВЕДЕНИЕ

В современном мире татуировки стали неким аксессуаром, указывающим на интересы, хобби, предпочтения человека. Тату приобрело широкую популярность, особенно среди людей, склонных к творчеству. Сейчас рисунки на коже стали классифицировать по стилям, на сегодняшний день существует более двух десятков видов тату. Татуировки стараются превратить не просто в пустую кляксу на теле, а в целую картину со своей историей.

Данное программное устройство незаменимо для желающих сделать тату, поскольку благодаря интуитивно понятному пользовательскому интерфейсу позволяет самостоятельно выбрать желаемый эскиз, предварительно ознакомившись с подробной информацией о процессе и мастерах, не выходя из дома.

Программное приложение позволит авторизированному пользователю просматривать работы и эскизы по интересующим стилям, добавлять работы в избранное, записываться на сеанс онлайн, изучать отзывы других клиентов, получить тех поддержку.

Приложение поможет составлять расписание мастеров администратору, который имеет возможность управлять базой данных из административной части, подтверждать и отменять запись, оказывать техподдержку клиентов, выполнять поиск и сортировку и имеет доступ ко всем записям.

Для успешной реализации курсового проекта необходимо:

* провести анализ соответствующей литературы;
* ознакомиться с прототипами программных средств выбранной мной темы;
* определить функциональные требования;
* продумать структуру базы данных;
* продумать структуру проекта;
* реализовать программное средство;
* протестировать программное средство;
* написать руководство пользователя.

Главная задача данного курсового проектирования – это разработка программного средства, которое реализует все вышеперечисленные функции и решает поставленные задачи. Язык разработки проекта – C#. При выполнении курсового проекта будут использованы принципы и приемы ООП, база данных MS SQL Server, и технология Windows Presentation Foundation (WPF).

Содержание данной пояснительной записки отражает все этапы выполнения курсового проекта.

# 1 Аналитический обзор литературы и формирование требований

## 1.1 Анализ прототипов

Были проанализированы цели и задачи, поставленные в данном курсовом проекте, а также рассмотрены аналогичные примеры их решений. На основании анализа всех достоинств и недостатков данных альтернативных решений были сформулированы требования к данному программному средству.

Первый аналог — интернет-ресурс 365DaysTatoo.

Сайт является интернет-магазином татуировок в Минске, Беларусь. Он предоставляет разнообразный функционал, который помогает пользователям ознакомиться с услугами студии и записаться на них. Интерфейс сайта представлен на рисунке 1.1.

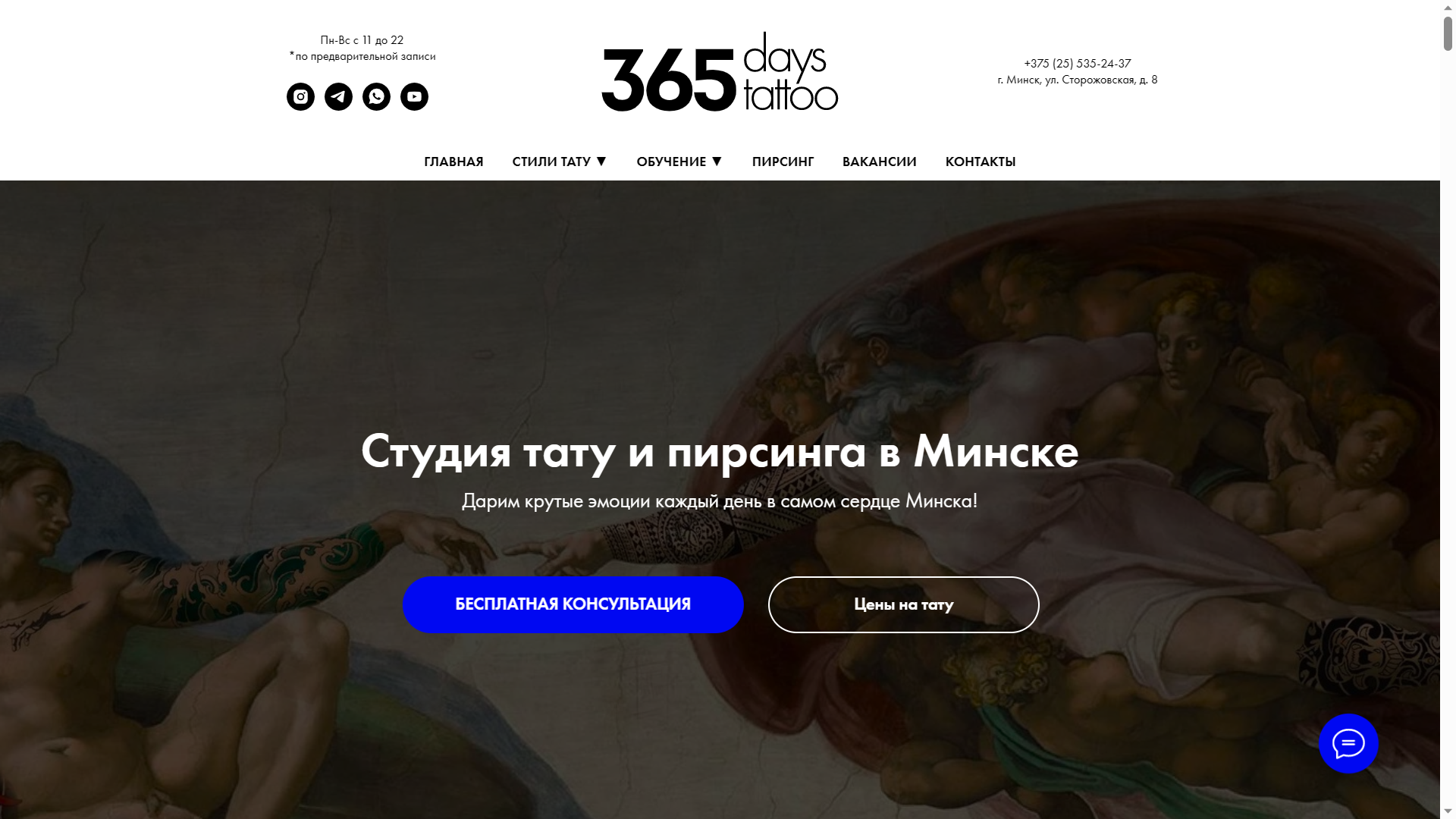


Рисунок 1.1 – Интернет-ресурс «Afisha.me»

На этом сайте пользователь может просмотреть портфолио мастеров и ознакомиться с их работами, чтобы выбрать подходящего для себя специалиста. Помимо этого, у клиента есть возможность записаться на услуги выбранного мастера в удобное для него время, используя функцию онлайн-записи.

Для удобства пользователей сайт также предлагает систему электронных платежей, позволяющую клиенту оплатить услуги онлайн. Это удобно и экономит время клиентов.

Кроме того, на сайте предоставлена подробная информация о различных услугах, предлагаемых мастерами студии, таких как выполнение татуировок и разработка дизайнов. Клиент может ознакомиться с описанием материалов, используемых в работе мастеров, чтобы быть уверенным в их качестве.

Также на сайте приведена информация о самой студии и ее местоположении. Если у клиента возникнут вопросы или он захочет связаться со студией, на сайте указаны контактные данные, включая телефон и адрес электронной почты.

Функционал сфокусирован на представлении работ мастеров и облегчении процесса записи на услуги в салоне. Хотя он не обладает таким широким функционалом, он предоставляет удобство при выборе мастера для татуировки и дает возможность клиенту записаться на услугу.

Сайт имеет несколько недостатков, пользователи могут столкнуться с проблемой долгой загрузки страниц, отсутствием подробной информации и качественных фотографий татуировок. Еще одним недостатком является отсутствие отзывов клиентов и контактной информации для связи.

Второй аналог — это приложение Tattoodo.

Tatoodo - это мобильное приложение, предназначенное для любителей татуировок и поиска профессиональных мастеров. Оно предлагает широкий функционал и удобный интерфейс, чтобы помочь пользователям найти и выбрать подходящего мастера для своей следующей татуировки. Интерфейс приложения представлен на рисунке 1.2.

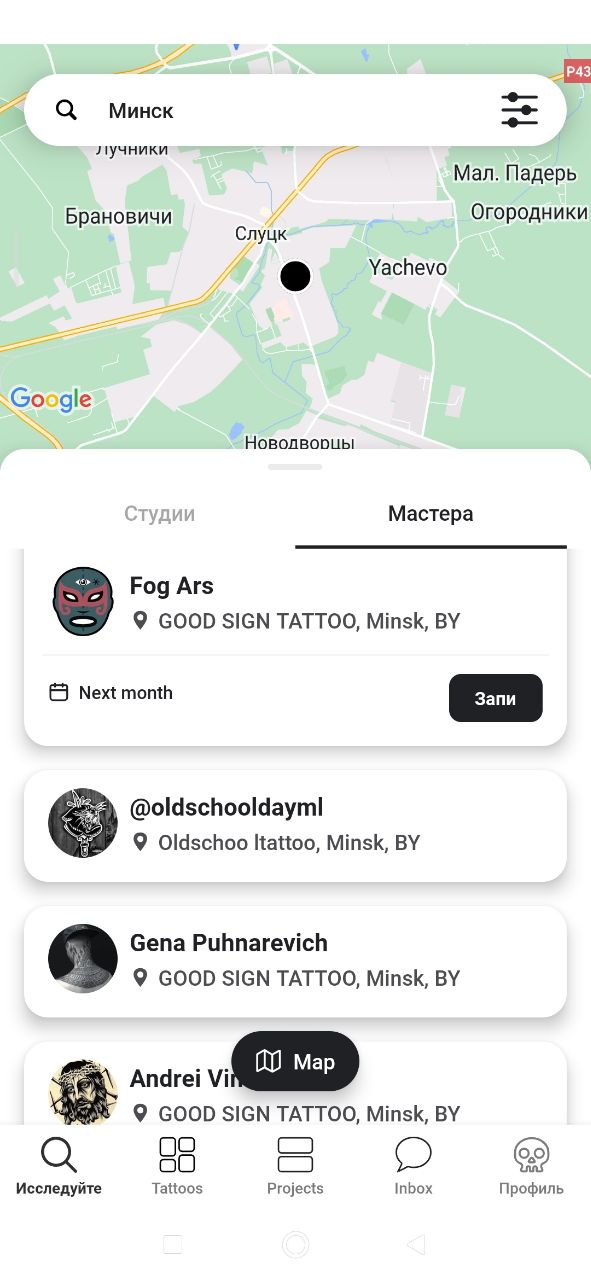


Рисунок 1.2 – Приложение «Tatoodo»

Приложение Tatoodo позволяет пользователям просматривать портфолио мастеров, ознакомиться с их работами и описаниями, чтобы оценить стиль и качество работы. Они могут также воспользоваться функцией поиска, чтобы найти мастера по местоположению, стилю татуировки и рейтингу.

Онлайн-запись является одной из удобных возможностей приложения. Пользователи могут записаться на услуги выбранного мастера, выбрав удобное для них время и дату, без необходимости звонить или отправлять электронные письма для записи.

Через встроенную систему обмена сообщениями пользователи могут связаться с мастером, обсудить детали татуировки, задать вопросы и получить необходимую информацию.

После завершения услуги пользователи могут оставить отзыв о мастере и оценить качество работы, что поможет другим пользователям принять информированное решение и поддержать качество услуг.

Приложение также предоставляет уведомления о подтверждении записи, изменениях в расписании мастера или других важных обновлениях.

В целом, Tatoodo предлагает удобную платформу для поиска татуировщиков, просмотра их работ, записи на услуги и общения с мастером.

Третий аналог — интернет–ресурс Tattoo Club.

Сайт https://tattoo-club.by является официальным сайтом студии татуировки и магазина экипировки Tattoo Club в Беларуси. На этом сайте пользователь найдет все необходимое для ознакомления с работами мастеров, записи на сеанс татуировки, получения информации о стоимости и блоге студии. Интерфейс сайта представлен на рисунке 1.3.

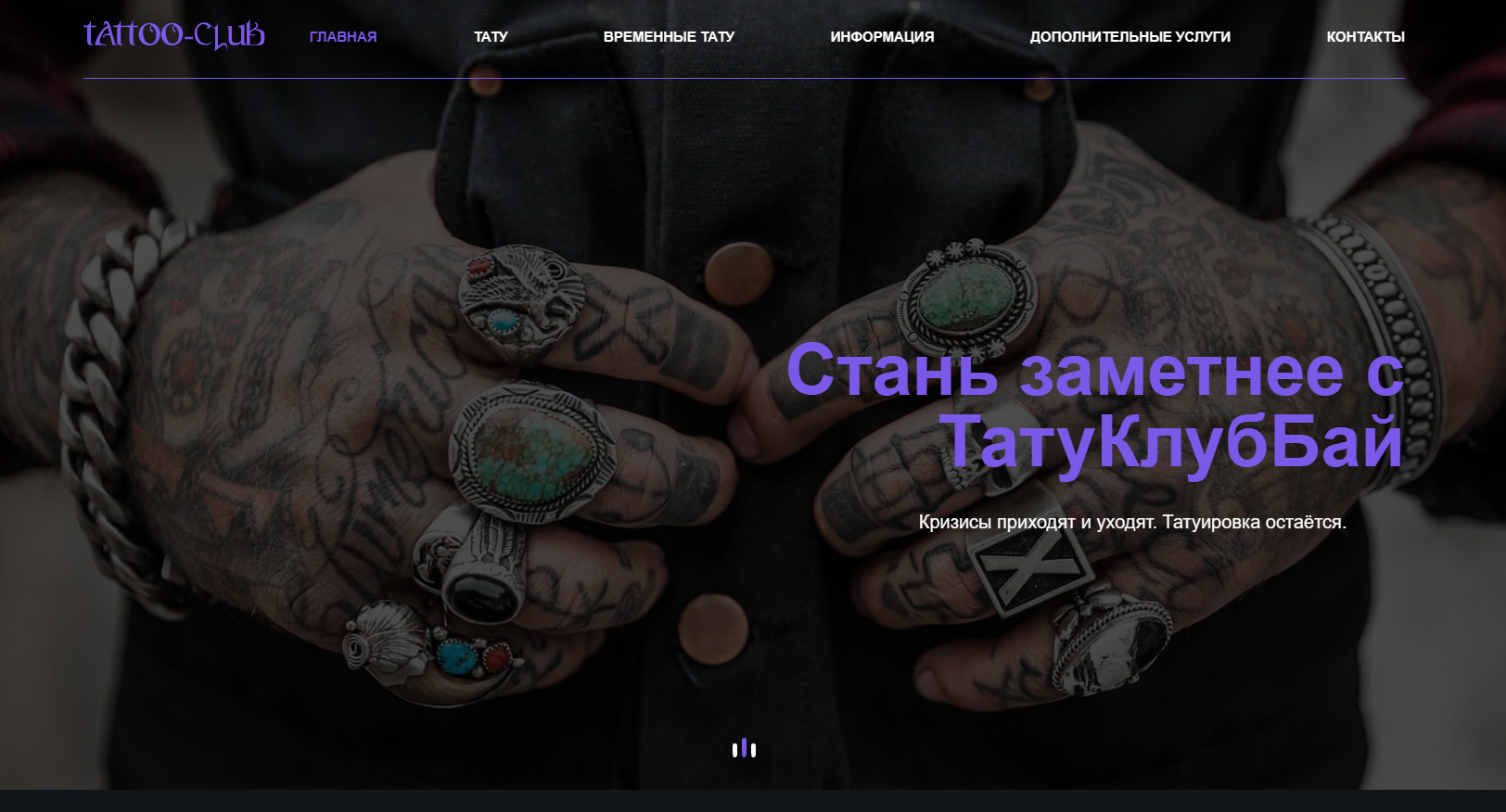


Рисунок 1.3 – Интернет-ресурс «Tattoo Club»

На главной странице сайта пользователь сразу видит карусель с фотографиями различных татуировок, что позволяет быстро ознакомиться с работами мастеров. В меню навигации есть раздел "Татуировки", где представлены категории и подкатегории работ мастеров, такие как "черно-белые татуировки", "колор-татуировки" и другие. Каждая работа сопровождается подробным описанием и информацией о мастере.

Для удобства пользователей есть возможность регистрации на сайте и записи на сеанс татуировки. Пользователь может выбрать студию, мастера, указать дату, время и оставить свои контактные данные.

Дополнительно, на сайте имеется возможность расчета стоимости татуировки в зависимости от размера места на теле. Это позволяет пользователю оценить предполагаемую стоимость своей татуировки.

В разделе "Блог" на сайте пользователь найдет статьи о татуировках, процессе их нанесения, а также новости студии. Раздел предоставляет удобную навигацию по статьям и поиск по ключевым словам.

Интерфейс сайта выполнен в современном стиле, с использованием ярких фотографий и удобным меню навигации. Оформление страниц выполнено в едином стиле, что позволяет легко ориентироваться на сайте. На главной странице сайта расположена карусель с работами мастеров, которая позволяет быстро сделать выбор. В целом, сайт отличается удобным интерфейсом и высоким уровнем функционала.

Анализируя аналоги, можно сделать вывод, что все три приложения предлагают широкий спектр функций, включая каталог работ мастеров, возможность записи на услуги, систему онлайн-оплаты, подробную информацию о услугах, описание используемых материалов, информацию о студии и ее местоположении, контактные данные и возможность связаться со студией. Они также обладают современным интерфейсом и стараются обеспечить удобство и легкость в использовании для клиентов. Однако, каждое приложение может иметь свои особенности и преимущества, которые следует учитывать.

## 1.2 Требования к проекту

Обзор вышеперечисленных аналогов позволяет проанализировать все преимущества и недостатки альтернативных возможностей и позволяет сформулировать список требований, предъявляемых к программному средству, разрабатываемому в данном курсовом проекте.

Программное средство должно обеспечивать возможность выполнения перечисленных ниже функций:

* управление администратором базой данных;
* возможность пользователю зарегистрироваться или войти в существующую учетную запись;
* возможность авторизированным пользователям просматривать каталог эскизов и мастеров;
* возможность авторизированным пользователям записываться на сеанс;
* возможность авторизированным пользователям добавлять эскиз в избранное.

# 2 Анализ требований к программному средству и разработка функциональных требований

# 2.1 Описание средств разработки

При разработке приложения были использованы:

* интегрированная среда разработки Microsoft Visual Studio 2022;
* программная платформа .NET Framework 6.0;
* язык программирования C#;
* расширяемый язык разметки XAML;
* технология WPF;
* технология Entity Framework;
* MS SQL Server.

### 2.1.1 Microsoft Visual Studio 2022

Microsoft Visual Studio 2022 - это интегрированная среда разработки (IDE) от Microsoft, которая обеспечивает разработчиков инструментами и возможностями для создания приложений с использованием Windows Presentation Foundation (WPF).

### 2.1.2 Программная платформа .NET 6

Платформа .NET - это программный фреймворк, разработанный компанией Microsoft, который в основном работает на операционной системе Microsoft Windows. Он предоставляет среду выполнения для запуска приложений и набор библиотек, которые используются для создания программных приложений. Фреймворк .NET состоит из нескольких компонентов, включая общую языковую среду (Common Language Runtime - CLR), библиотеку классов .NET Framework и различные инструменты разработки.

### 2.1.3 Язык программирования C#

В качестве языка программирования используется C# – основной язык разработки в .NET. Язык объектно-ориентированный, имеет строгую статическую типизацию, поддерживает перегрузку операторов, указатели на функции-члены классов, атрибуты, события, свойства, исключения. Используется как основной язык в технологии WPF.

### 2.1.4 Технология WPF

Для предоставления пользовательского интерфейса и разграничения дизайна и бизнес-логики используется технология Microsoft WPF – аналог WinForms, система для построения клиентских приложений Windows с возможностями взаимодействия с пользователем и графическая подсистема в составе .NET, использующая язык разметки XAML.

### 2.1.5 Расширяемый язык разметки XAML

WPF предоставляет средства для создания визуального интерфейса, включая язык XAML (eXtensible Application Markup Language элементы управления, привязку данных, макеты, двухмерную и трёхмерную графику, анимацию, стили, шаблоны, документы, текст, мультимедиа и оформление. XAML представляет собой язык декларативного описания интерфейса, основанный на XML.

### 2.1.6 Технология Entity Framework

Для взаимодействия с базой данных в приложении WPF используется технология Entity Framework Core — кроссплатформерная и расширяемая версия технологии Entity Framework с открытым исходным кодом, которая в текущий момент развивается и дополняется.

Если традиционные средства ADO.NET позволяют создавать подключения, команды и прочие объекты для взаимодействия с базами данных, то Entity Framework представляет собой более высокий уровень абстракции, который позволяет абстрагироваться от самой базы данных и работать с данными независимо от типа хранилища. Entity Framework Core – это последняя версия Entity Framework 7, которая совместима с .NET Core и на данный момент продолжает развиваться. Entity Framework Core предполагает три возможных способа взаимодействия с базой данных:

* Database First: Entity Framework Core создает набор классов, которые отражают модель конкретной базы данных;
* Model First: сначала разработчик создает модель базы данных, по которой затем Entity Framework Core создает реальную базу данных на сервере;
* Code First: разработчик создает класс модели данных, которые будут храниться в базе данных, а затем Entity Framework Core по этой модели генерирует базу данных и ее таблицы.

### 2.1.7 MS SQL Server

Для организации баз данных MS SQL Server использует реляционную модель, которая предполагает хранение данных в виде таблиц, каждая из которых состоит из строк и столбцов. Каждая строка хранит отдельный объект, а в столбцах размещаются атрибуты этого объекта. Для взаимодействия с базой данных применяется язык SQL (Structured Query Language). Клиент (например, внешняя программа) отправляет запрос 10 на языке SQL должным образом интерпретирует и выполняет запрос, а затем посылает клиенту результат выполнения. Основной используемый язык запросов — Transact-SQL — реализован на структурированном языке запросов (SQL) с расширениями.

### 2.1.8 Паттерн MVVM

MVVM (Model-View-ViewModel) - это паттерн архитектуры, используемый в разработке программного обеспечения для разделения пользовательского интерфейса (View) от бизнес-логики (Model) и связывания их через промежуточный слой ViewModel.

В MVVM модель представляет данные и бизнес-логику приложения, которая может включать в себя операции чтения/записи данных из и в источники данных, такие как база данных или веб-сервисы.

Представление (View) отвечает за отображение данных и взаимодействие с пользователем. Оно не содержит бизнес-логику и должно быть максимально независимым от модели.

ViewModel представляет промежуточный слой между View и Model. Он содержит логику, необходимую для обработки пользовательских действий, обновления данных в модели и уведомления View об изменениях. ViewModel предоставляет свойства и команды, которые привязываются к элементам пользовательского интерфейса во View, позволяя им отображать данные и реагировать на действия пользователя.

MVVM позволяет достичь разделения ответственности между компонентами приложения, облегчает тестирование и повышает переиспользуемость кода. Он также способствует лучшей поддержке параллельной разработки пользовательского интерфейса и бизнес-логики, так как разработчики могут работать независимо над своими частями приложения.

## 2.2 Спецификация функциональных требований к программному средству

Программное средство должно предоставлять следующие функциональные возможности:

Для пользователя:

* регистрация;
* авторизация;
* просмотр эскизов;
* просмотр портфолио мастеров;
* фильтрация эскизов;
* поиск мастера
* запись на сеанс;
* просмотр информации о личной записи;
* отмена записи;
* добавление эскиза в избранное;
* удаление эскизы из избранного;
* возможность оставить отзыв;
* возможность посмотреть отзывы на определенного мастера.

Для администратора:

* авторизация;
* просмотр эскизов;
* фильтрация эскизов;
* просмотр мастеров
* просмотр записей;
* удаление мастеров;
* удаление эскизов ;
* удаление записей ;
* удаление пользователей;
* просмотр пользователей;
* поиск пользователей.

## 2.3 Спецификация функциональных требований

Для функциональности ПС необходимо создание базы данных для хранения информации приложения. Подробно о базе данных описано в главе 3.

В программном средстве при запуске необходимо реализовать регистрацию и авторизацию пользователей для дальнейшего использования приложения. Для авторизации входными параметрами являются логин и пароль пользователя, которые содержатся в базе данных. Для того чтобы зарегистрироваться необходимо ввести информацию клиента. Введенные данные, успешно прошедшие валидацию, заносятся в базу данных, а пользователь получает доступ к приложению.

При авторизации пользователь перемещается на главную страницу приложения. На всех страницах размещается меню, с помощью которого клиент может перемещаться по приложению. На странице «Эскиз» пользователь имеет возможность изучить доступные эскизы и добавить их в избранное. На странице «Мастера» он может посмотреть портфолио мастеров, оставить отзыв, посмотреть отзывы, перейти на «Запись», где он может выбрать мастера, дату и время сеанса. В личном кабинете пользователь может просмотреть предстоящие посещения тату – салона.

При авторизации администратора администратор попадает на страницу, где он может переключаться между таблицами базы данных, удалять строки и искать информацию.

Все данные о пользователях, мастерах, эскизах должны заноситься в базу данных в соответствующие таблицы.

# 3 Проектирование программного средства

## 3.1 Общая структура

Программное средство «PL TATTOO» имеет следующею структуру, представленную на рисунке 3.1.



Рисунок 3.1 – Структура проекта

Описание структуры основных папок и файлов проекта и библиотеки классов представлено в таблице 3.1.

Таблица 3.1 – Описание структуры папок и файлов проекта

|  |  |
| --- | --- |
| Имя файла | Содержание |
| App.config | Файл с параметрами проекта. |
| App.xaml | Определение глобальных ресурсов |
| Папка Models | Данные и состояние приложения |
| Папка Font | Шрифты приложения |
| Папка Infrastructure | Интерфейс команд(ICommand), реализация команд (RelayCommand) |
| Папка Resources | Все изображения |
| Папка Services | Функционал для сторонних сервисов |
| Папка Style | Словари ресурсов стилей |
| Папка ViewModels | Классы модели представления |
| Папка Views | XAML разметка окон приложения |

Таким образом, сформированная таблица помогает понять общую стркутура проекта проектируемого программного средства

Cтруктура папки Model (сущностей приложения) представлена на рисунке  3.2.

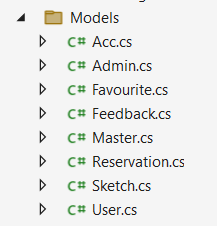


Рисунок 3.1 – Структура папки «Model»

Боллее подробное описание структуры и файлов данной папки представлено в таблице 3.2

Таблица 3.1 – Описание структуры папок и файлов проекта

|  |  |
| --- | --- |
| Имя файла | Содержание |
| Acc.cs | Класс Acc для определения и хранения аккаунтов приложения |
| Admin.cs, User.cs | Классы User и Admin для определения хранения разных типов аккаунта |
| Favourite | Класс Favourite для определения и хранения избранных эскизов |
| Feedback.cs | Класс Feedback для определения и хранения отзывов. |
| Master.cs | Класс Master для пределения и хранения мастеров приложения |
| Reservation.cs | Класс Reservation для определения и хранения сеансов |
| Sketch.cs | Класс Sketch для определения и хранения эскизов приложения |

В целом, описание структуры проекта позволяет лучше понимать, как устроено программное средство и какие компоненты в нем присутствуют.

## 3.2 Взаимоотношение между классами

Для визуализации взаимосвязей между классами используется диаграмма UML – графическое представление набора элементов, изображаемое чаще всего в виде связанного графа с вершинами (сущностями) и ребрами (отношениями).

Для представления внутренней структуры программы в виде классов и связей между ними используется диаграмма классов. Приложение спроектировано таким образом, что каждый класс выполняет свои функции и практически не зависит от других. Диаграмма классов представлена в приложении А.

## 3.3 Модель базы данных

Для реализации поставленной задачи была создана база данных TATOO. Для ее создания использовалась система управления реляционными базами данных MS SQL Server. База данных состоит из таблиц, представленных на рисунке 3.2.

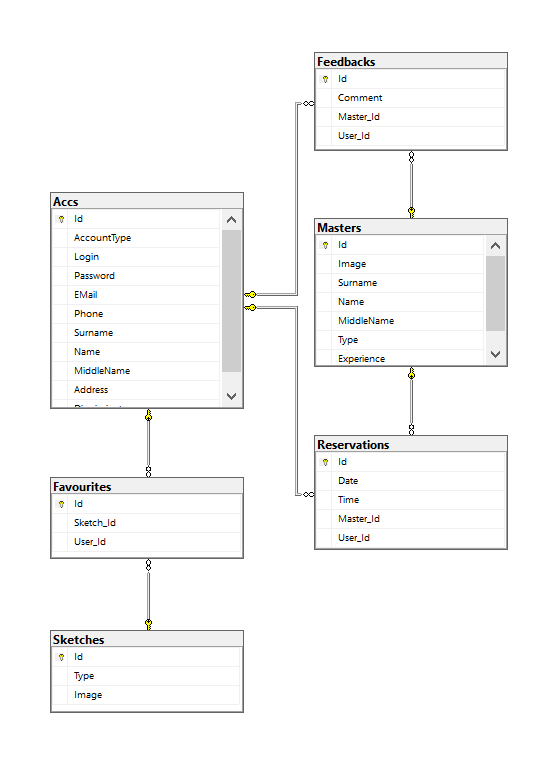


Рисунок 3.2 — База данных TATOO

На рисунке 3.3 проиллюстрирована структура таблицы «Favourite», которая содержит информацию о избранных эскизах. В данной таблице есть Id —целочисленное поле для идентификации записи в базе данных. User — представляющего пользователя, который добавил данный элемент в избранное. Поле Sketch — представляющего элемент, добавленный в избранное.

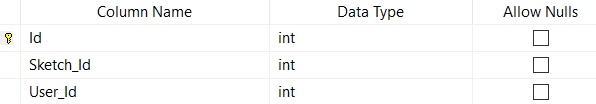


Рисунок 3.3 — Структура таблицы «Favourite»

На рисунке 3.4 изображена структура таблицы «Feedback». В данной таблице хранится информация об отправленных отзывах. Первое поле — «Id». Целочисленное поле для идентификации записи в базе данных. Поле «User» представляющего пользователя, оставившего отзыв. Поле «Master» представляющего мастера, к которому относится отзыв. А поле «Comment» содержит комментарий или отзыв пользователя о мастере.

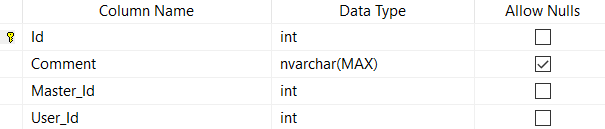


Рисунок 3.4 — Структура таблицы «Feedback»

На рисунке 3.5 изображена структура таблицы «Masters». Данная таблица для хранения информации о мастерах.

Id - поле для идентификации записи в базе данных. Поле Image — содержащее изображение мастера. Поле Surname - содержащее фамилию мастера. Поле Name — содержащее имя мастера. Поле MiddleName — содержащее отчество мастера. Поле обязательно для заполнения. Поле Type — представляющее тип татуировки мастера. И поле Experience — представляющее стаж работы мастера.

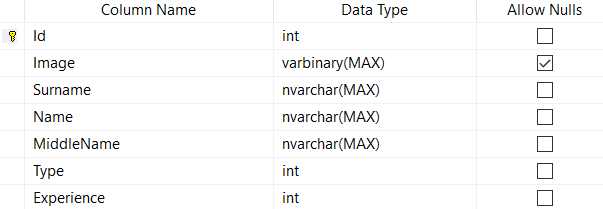


Рисунок 3.5 — Структура таблицы «Masters»

На рисунке 3.6 изображена структура таблицы «Reservation». Данная таблица для хранения информации о сеансах.

Id ­— поле для идентификации записи в базе данных. Поле User — представляющего пользователя, сделавшего бронь. Поле Master — представляющего мастера, к которому относится бронь. Поле Date — представляющее дату бронирования. Time — поле представляющее время бронирования.



Рисунок 3.6 — Структура таблицы «Reservations»

На рисунке 3.6 изображена структура таблицы «Reservation». Данная таблица для хранения информации о сеансах.

Id: целочисленное поле, представляющее уникальный идентификатор аккаунта (первичный ключ). AccountType: поле перечисления AccountType, указывающее тип аккаунта (Admin, User, None). Login: обязательное строковое поле, представляющее логин аккаунта. Password: обязательное строковое поле, представляющее пароль аккаунта. EMail: обязательное строковое поле, представляющее адрес электронной почты аккаунта (должно соответствовать формату электронной почты). Phone: обязательное строковое поле, представляющее номер телефона аккаунта. Surname: обязательное строковое поле, представляющее фамилию пользователя аккаунта. Name: обязательное строковое поле, представляющее имя пользователя аккаунта. MiddleName: обязательное строковое поле, представляющее отчество пользователя аккаунта.

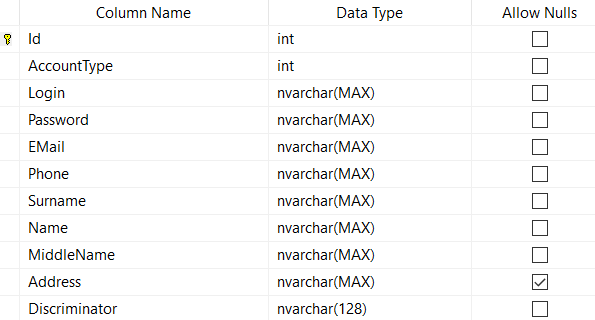


Рисунок 3.6 — Структура таблицы «Reservations»

В целом, структура базы данных "TATOO" предоставляет необходимую основу для хранения и управления информацией о пользователях, мастерах, записях в приложении "PL TATTOO".

## 3.4 Проектирование архитектуры приложения

Для общего представления функционального назначения системы часто используется диаграмма использования. Она позволяет описать, какой функционал доступен каждой группе пользователей в разрабатываемой программной системе. В диаграмме использования выделяются два основных типа сущностей: варианты использования и актёры.

Актёры представляют различные группы пользователей, которые взаимодействуют с системой. Они могут быть представлены любыми сущностями, использующими систему. Варианты использования, с другой стороны, описывают функционал системы. Каждый вариант использования определяет набор действий, которые актёр может выполнить для взаимодействия с системой, а также действия, которые сама система выполняет в ответ.

Диаграмма использования представлена в приложении Б.

## 3.5 Проектирование последовательностей проекта

Для наглядного отображения взаимодействия между объектами системы в различные моменты времени в рамках одного сценария использования применяется диаграмма последовательностей в UML. Эта диаграмма позволяет иллюстрировать, как объекты системы взаимодействуют друг с другом для выполнения определенной функции, а также отображает порядок их взаимодействия в рамках конкретного случая использования.

Для отображения времени на диаграмме используется линия жизни объекта, которая представляет период его существования в системе и изображается вертикальной штриховой линией, направленной вниз. Объекты представлены прямоугольниками, а сообщения, которыми они обмениваются, обозначаются линиями с стрелками.

Диаграмма последовательностей позволяет наглядно представить взаимодействие между объектами и порядок выполнения действий во времени. Она является полезным инструментом для анализа и проектирования системы, а также может использоваться для коммуникации между разработчиками и заинтересованными сторонами.

Пример диаграммы последовательности представлен в приложении В.

# 4 Реализация программного средства

## 4.1 Основные классы программного средства

Для выполнения технических задач программного средства «PL TATTOO» должны быть реализованы следующие функции и соответствующие им классы и методы:

* авторизация;
* регистрация;
* добавление в избранное;
* запись на сеанс.

## 4.2 Авторизация

Необходимый функционал для возможности входа пользователя в аккаунт, а также для проверки корректности данных, реализован в классе Manager.

Основным методом класса является метод Login, выполняющийся при нажатия пользователя на кнопку «Войти». В данном методе вызывается метод HashPassword, преобразующий введённый пользователем пароль в хэш-код для последующей проверки на соответствие с сохранённым в базе данных паролем.

При успешной проверке на существование введённого логина и пароля в базе данных пользователь перенаправляется на главную страницу, а противном случае – получает сообщение о соответствующей ошибке.

Листинг реализации класса Manager представлен в приложении Г.

## 4.3 Регистрация

Для совершения пользователем регистрации нового аккаунта реализован метод Register класс Manager.

При нажатии пользователя на кнопку «Зарегистрироваться» вызывается основной метод класса Validator – Validate(). В начале все введённые пользователем данные проходят проверку на корректность с использованием регулярных выражений, и поля проверяются на пустоту.

После успешного прохождения вышеописанных проверок происходит попытка добавить пользователя в базу данных и перенаправление пользователя на страницу входа. Если логин, адрес электронной почты или номер телефона уже зарегистрирован в базе данных, то пользователь получает соответствующее уведомление об ошибке.

Листинг реализации класса Manager представлен в приложении Г.

## 4.4 Запись на сеанс

Для совершения записи на сеанс пользователь заполняет форму записи и нажимает кнопку «Записаться», после этого вызывается метод SaveReservation.

Данный метод выполняет сохранение записи о бронировании в базе данных. Он проверяет наличие выбранного мастера, выбранного времени и даты, а также проверяет, что выбранная дата больше текущей даты.

Если все проверки пройдены успешно, метод создает экземпляр контекста базы данных и выполняет проверку наличия дубликатов записей о бронировании для выбранного мастера, даты и времени.

Если дубликат обнаружен, выводится сообщение об ошибке. В противном случае, метод создает новую запись о бронировании в базе данных с использованием выбранных параметров и сохраняет изменения.

После успешного сохранения выводится сообщение пользователю о успешном бронировании.

Если какое-либо из полей не заполнено или выбранная дата меньше текущей даты, выводится сообщение с просьбой заполнить все поля корректно.

Листинг данного метода SaveReservation представлен в приложении Г.

## 4.5 Добавление в избранное

На странице эскизов при наведении на определенный рисунок пользователь может добавить его в избранное по нажатию на кнопку «Сердце». Этот функционал обеспечивает метод FavoriteCommand.

Метод ищет запись в таблице Favourites базы данных, где идентификатор пользователя равен идентификатору текущего пользователя и идентификатор эскиза равен идентификатору переданного эскиза. Если запись уже существует, это означает, что эскиз уже находится в избранном.

Если найденная запись существует, она удаляется из таблицы Favourites и изменения сохраняются в базе данных. Если записи не существует, создается новая запись в таблице Favourites с текущим пользователем и переданным эскизом, а затем изменения сохраняются.

Листинг реализаций команды FavoriteCommand представлен в приложении В.

## 5 Тестирование, проверка работоспособности и анализ полученных результатов

## 5.1 Тестирование регистрации

При регистрации пользователь может совершить попытку использования уже занятого логина. Проводится тестирование данной ситуации, в базе данных уже существует запись с идентичным логином.

Результат данного теста представлен на рисунке 5.1.



Рисунок 5.1 — Результат первого тестирования

Так же при регистрации пользователь может ввести некорректные данные в поля формы. Проводится тест данного случая, пользователь ввел некорректный e- mail

Результат данного тестирования представлен на рисунке 5.2.

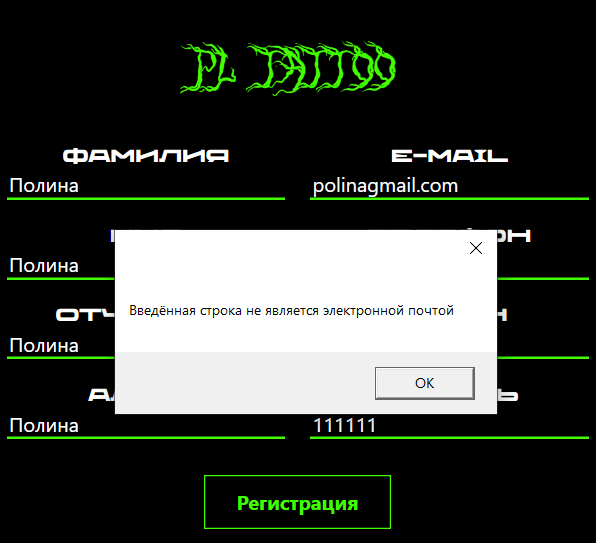


Рисунок 5.2 — Результат второго тестирования

Результаты тестирования регистрации говорят о том, что она работает корректно.

## 5.2 Тестирование отправки сообщения и работы с сообщениями

Самая главная задача мессенджера — это запись на сеанс. Поэтому протестировать данную функцию должно быть первой задачей.

При записи на сеанс возможна ситуация при которой пользователь пытается забронировать уже занятый сеанс. Тестируется данная ситуация, в базе данных уже существует идентичная запись у другого пользователя. Результат данного тестирования представлен на рисунке 5.3.

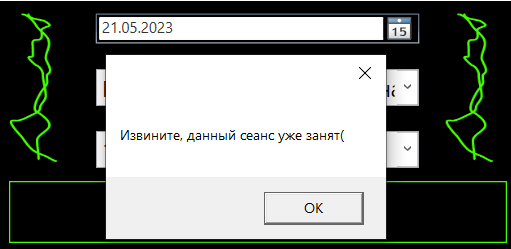


Рисунок 5.3 — Результат третьего тестирования

Также пользователь может совершить попытку записи на дату, которая уже прошла. Результат данного тестирования представлен на рисунке 5.4.

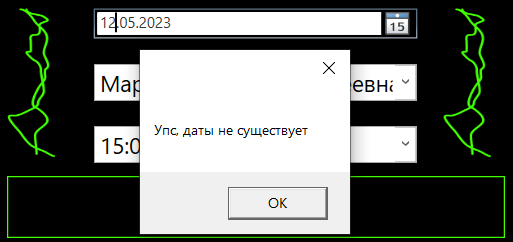


Рисунок 5.4 — Результат четвертого тестирования

Результаты тестирования формы записи говорят о том, что она работает корректно.

## 5.3 Тестирование панели администратора

Администратор должен добавлять корректную информацию в приложение. При заполнении формы для добавления информации мастер может не заполнить поле, проведем тестирование данной ситуации.

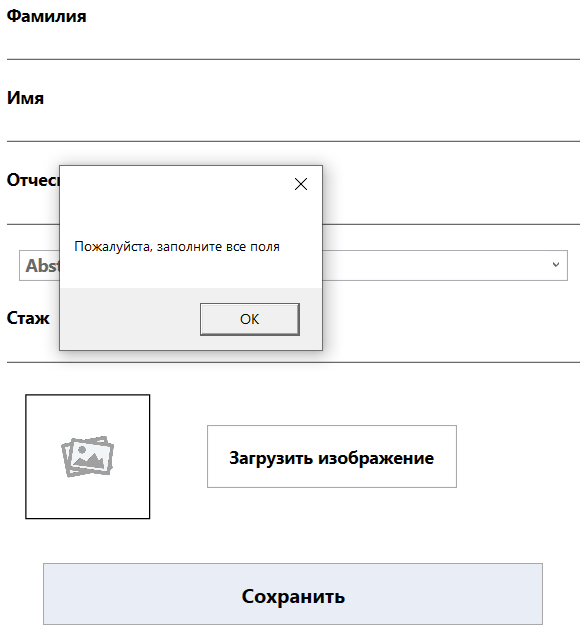


Рисунок 5.5 — Результат первой проверки панели администратора

Администратор не заполнил поле формы. Результат представлен на рисунке  5.5.

# 6 Руководство по установке и использованию

При первом запуске приложения будет открыто окно авторизации, которое показано на рисунке 6.1.



Рисунок 6.1 — Окно авторизации

Если у клиента не аккаунта, он может нажать на надпись «У вас нет аккаунта?» для того, чтобы создать свой аккаунт.

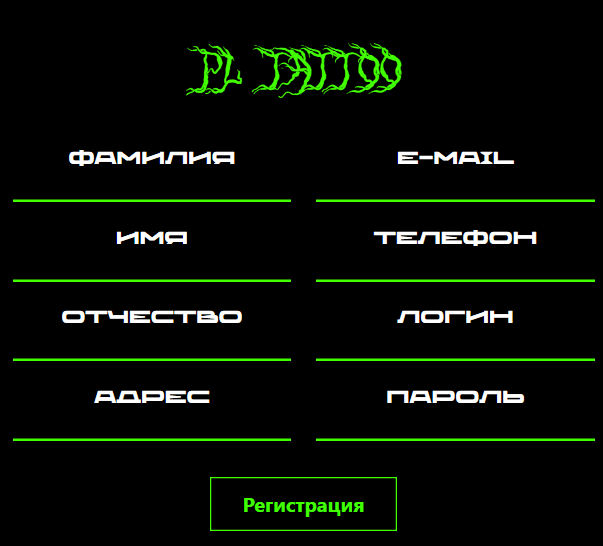


Рисунок 6.2 — Окно регистрации

После нажатия на кнопку регистрации откроется окно «Регистрация». Это окно представлено на рисунке 6.2. Логин должен содержать только латинские буквы (в верхнем или нижнем регистре), цифры, точку, дефис и длиной от 1 до 20 символов. Логин должен начинаться с буквы. Номер телефона должен соответствовать шаблону +375 или 80, за которыми следует код оператора (25, 29, 33 или 44), а затем 7 цифр. Адрес должен содержать только русские или латинские буквы, цифры, точку, запятую, дефис, слеш и пробелы. Адрес электронной почты должен быть в правильном формате, состоящем из локальной части, символа "@" и доменной части.

После успешной регистрации пользователя перекинет на страницу входа.

Когда пользователь совершит успешный вход, он перейдет на главную страницу. На стартовой странице пользователь может записаться на сеанс и начать перемещаться по приложению с помощью навигационного меню, которое описано далее. Главная страница приложения представлена на рисунке 6.3.



Рисунок 6.3 — Главная страница

C помощью навигационного меню пользователь может перемещаться по приложению, просматривать личные записи и свое избранное. Навигационное меню представлено на рисунке 6.4.



Рисунок 6.4 — Навигационное меню

На странице эскизы пользователь может просматривать доступные эскизы, перемещая скроллер, добавлять их в избранное. При наведении курсора на элемент меню «Эскизы» клиент сможет выбрать стиль, по которому он хочет отфильтровать эскизы. Страница эскизов продемонстрирована на рисунке 6.4.

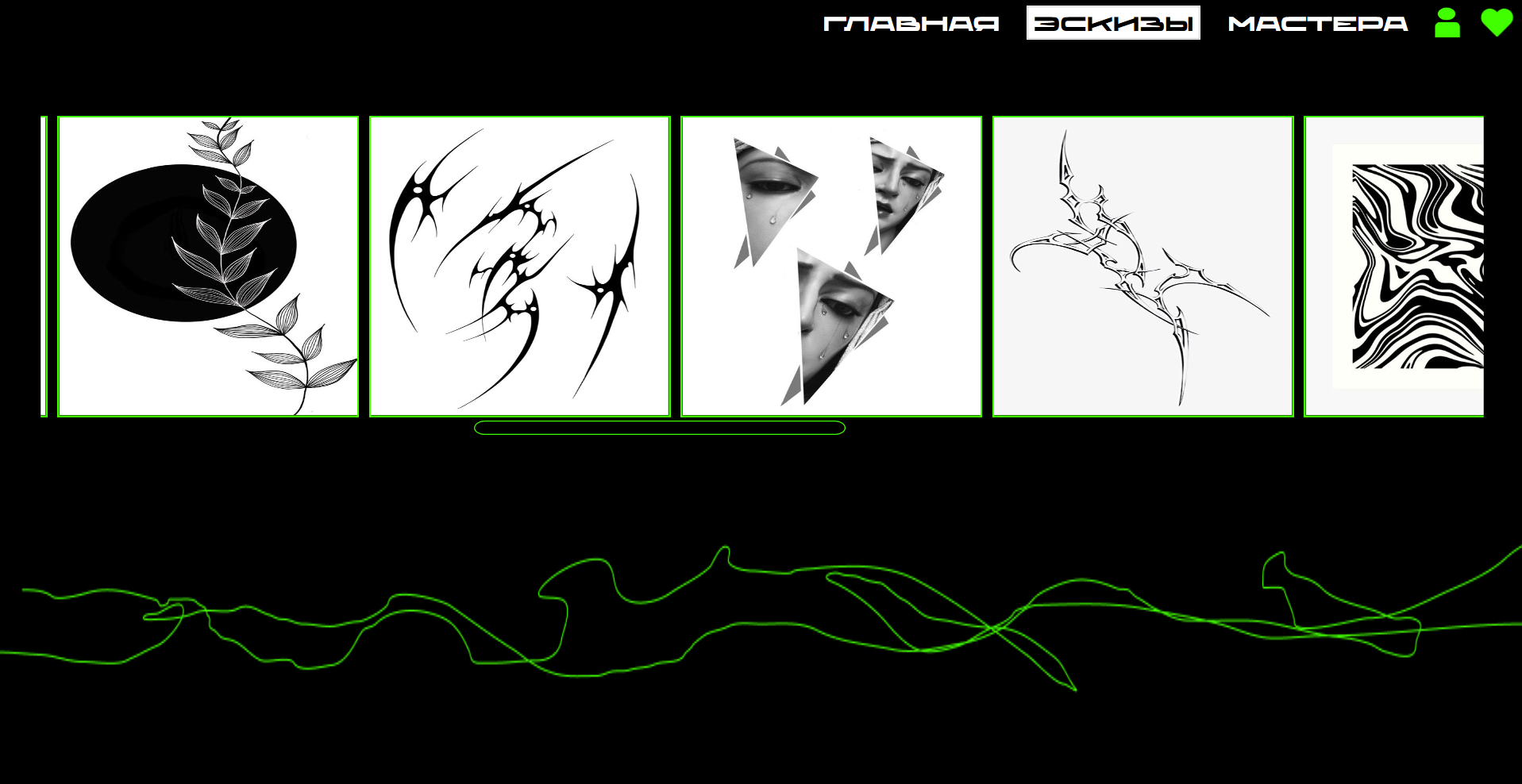


Рисунок 6.4 — Страница «Эскизы»

На странице мастеров пользователь может просматривать список доступных мастеров. При нажатии на определенного мастера открывается подробная информация и отзывы. Так же у пользователя есть возможность оставлять отзывы и искать мастеров. Страница «Мастера» представлена на рисунке 6.5.



Рисунок 6.5 — Страница «Мастера»

Для перехода пользователя на страницу записи, он должен нажать кнопку «Записаться» на главной странице, либо кнопку «Записаться к мастеру» в описании мастера. На данной странице пользователь должен заполнить форму и отправить ее. Окно записи представлено на рисунке 6.6.

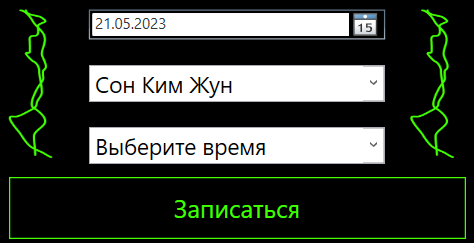


Рисунок 6.6 — Форма записи на сеанс

После записи на сеанс пользователь может ознакомиться с ней в личном кабинете или отменить. Окно личного кабинета представлено на рисунке 6.7.



Рисунок 6.7 — Окна личного кабинета

Так же пользователь может просмотреть свои избранные эскизы. При наведении на рисунок высвечивается кнопка для удаления эскиза из избранного. Окно избранного представлено на рисунке 6.7.

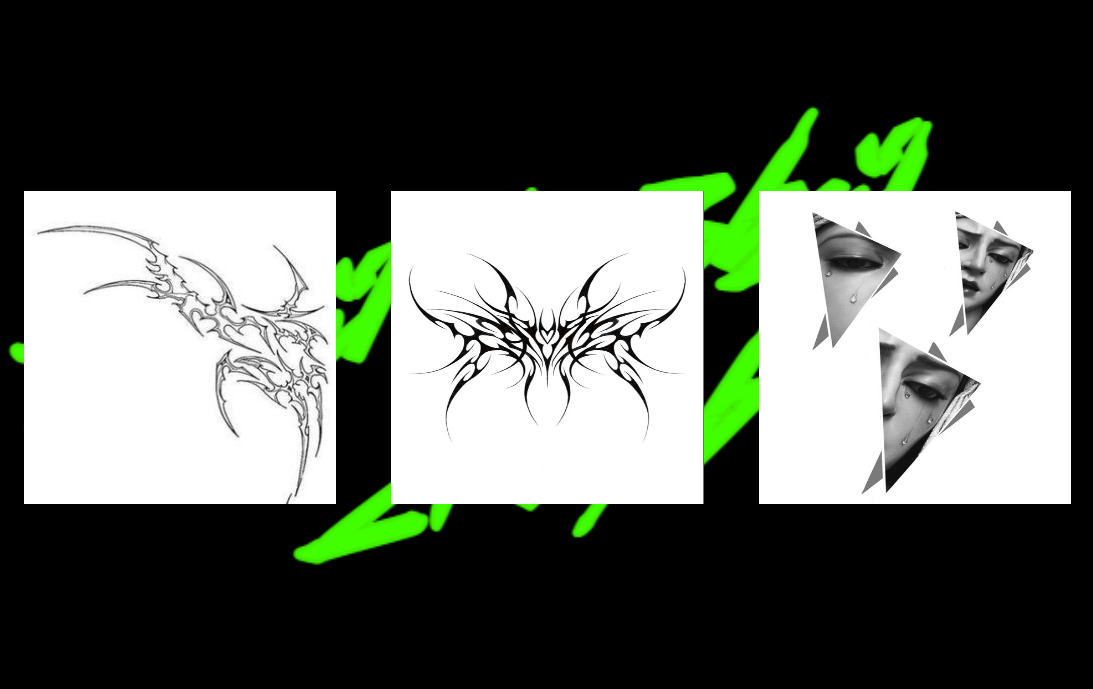


Рисунок 6.7 — Окна «Избранное»

В целом, приложение «PL TATTOO» обеспечивает удобный пользовательский интерфейс, для изучения контента тату-студии.

# ЗАКЛЮЧЕНИЕ

Данное курсовое проектное задание было выполнено с целью разработки программного средства, которое позволяет пользователям выбирать и записываться на сеансы татуировок, а также администратору управлять базой данных и обеспечивать техническую поддержку.

В процессе выполнения проекта были достигнуты следующие результаты:

* разработано интуитивно понятное пользовательское приложение с удобным интерфейсом для просмотра и выбора эскизов татуировок;
* реализована функциональность записи пользователей на сеансы татуировок и обработка данных в базе данных;
* разработана административная часть для управления расписанием мастеров и предоставления технической поддержки клиентам;
* произведено тестирование программного средства для проверки его функциональности и корректности работы.

Для реализации приложения был выбран язык программирования C# и технология Windows Presentation Foundation (WPF), обеспечивающая разработку удобного и современного пользовательского интерфейса. В качестве базы данных была использована MS SQL Server.

В процессе разработки программного средства "PL TATTOO" был активно применен паттерн проектирования MVVM (Model-View-ViewModel). MVVM является одним из наиболее популярных паттернов для разработки пользовательских интерфейсов в технологии WPF.

В процессе разработки программного средства " PL TATTOO " ViewModel была активно использована для связи пользовательского интерфейса с бизнес-логикой и данными. Благодаря MVVM, разработка стала более структурированной и управляемой, а код стал более поддерживаемым и расширяемым.

Таким образом, применение паттерна MVVM в программном средстве " PL TATTOO " обеспечило эффективное управление данными и логикой приложения, улучшенную архитектуру и обеспечило высокую отзывчивость и удобство использования пользовательского интерфейса.

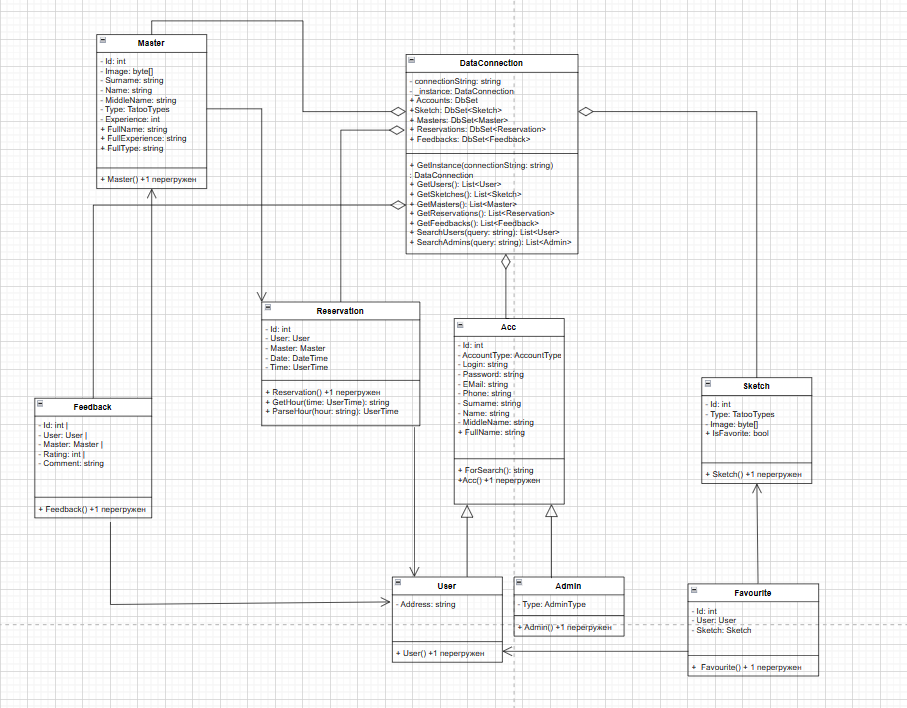
В результате выполнения данного курсового проекта было создано функциональное и удобное программное средство " PL TATTOO . Программа имеет потенциал для дальнейшего развития и расширения функциональности в соответствии с потребностями пользователей.

# Список использованных источников

1. Microsoft Visual Studio [Электронный ресурс] – https://ru.wikipedia.org/wiki/Microsoft\_Visual\_Studio – Дата доступа 23.04.2023
2. Полное руководство по языку программирования C# 7.0 и платформе .NET 4.7. Режим доступа: https://metanit.com/sharp/tutorial/ – Дата доступа: 23.04.2023
3. Пацей, Н.В. Курс лекций по языку программирования C# / Н. В. Пацей. – Минск: БГТУ, 2018. – 175 с.
4. Руководство по WPF // [Электронный ресурс]. – Режим доступа: https://metanit.com/sharp/wpf/ – Дата доступа: 25.04.2023
5. Руководство по XAML // [Электронный ресурс]. – Режим доступа: https://www.tutorialspoint.com/xaml/index.htm– Дата доступа: 25.04.2023
6. Блинова, Е.А. Курс лекций по Базам данным / Е.А. Блинова. – Минск: БГТУ, 2019. – 175 с.

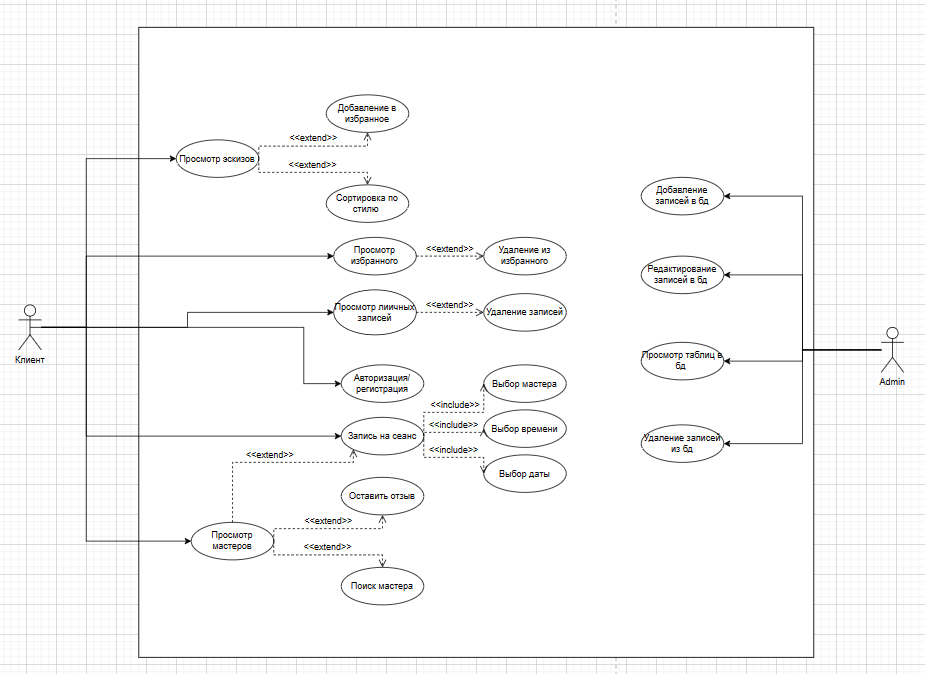
# ПРИЛОЖЕНИЕ А

Диаграмма классов



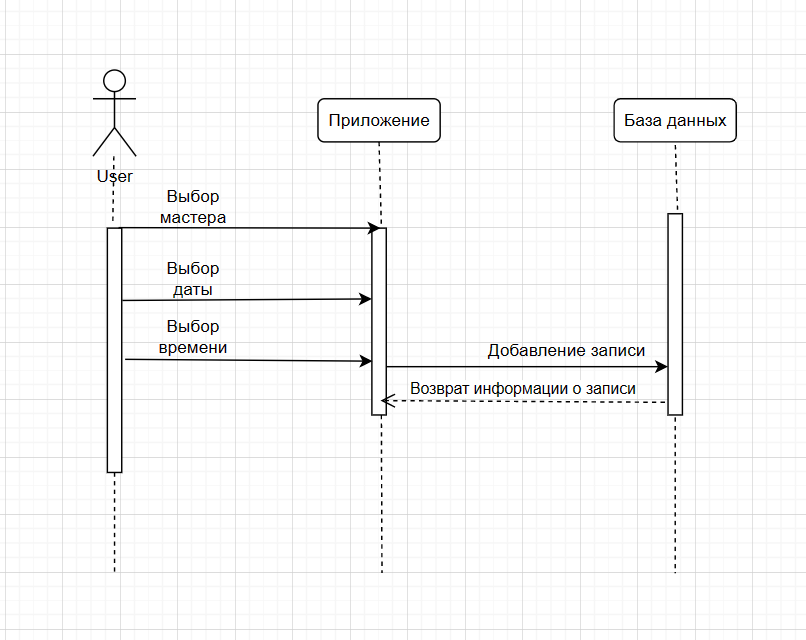
# ПРИЛОЖЕНИЕ Б

Диаграмма использования



# ПРИЛОЖЕНИЕ В

Диаграмма последовательности



# ПРИЛОЖЕНИЕ Г

Листинг класса Manager

|  |
| --- |
| public static class Manager  {  public static Acc CurrentUser { get; private set; }  public static AccountType AccountType { get; private set; }  static Manager()  {  AccountType = AccountType.None;  }  public static bool Login(string login, string password)  {  if (CurrentUser != null)  {  CurrentUser = null;  AccountType = AccountType.None;  }  var hashedPassword = HashPassword(password);  var user = DataConnection.GetInstance().Accounts.ToList()  .Find(item => item.Login == login && item.Password == hashedPassword);  if (user == null)  {  user = DataConnection.GetInstance().Accounts.ToList()  .Find(item => item.Phone == login && item.Password == hashedPassword);  if (user == null)  return false;  }  CurrentUser = user;  AccountType = user.AccountType;  return true;  }  public static (bool result, string error) Register(string login, string password, string email, string phone, string surname, string name, string middleName, string address)  {  try  {  var db = DataConnection.GetInstance();  if (db.Accounts.Any(item => item.Login == login))  return (false, "Логин уже занят");  if (db.Accounts.Any(item => item.EMail == email))  return (false, "Почта уже занята");  if (db.Accounts.Any(item => item.Phone == phone))  return (false, "Телефон уже занят");  var hashedPassword = HashPassword(password);  db.Accounts.Add(new User(login, hashedPassword, email, phone, surname, name, middleName, address));  db.SaveChanges();  return (true, null);  }  catch  {  return (false, "Не удалось зарегистрироваться");  }  }  public static string HashPassword(string password)  {  var hashBytes = MD5.Create().ComputeHash(Encoding.UTF8.GetBytes(password));  string hash = string.Empty;  foreach (var item in hashBytes)  hash += string.Format("{0:x2}", item);  return hash;  }  public static void Logout() => CurrentUser = null;  }  } |

Листинг метода SaveReservation

|  |
| --- |
| private void SaveReservation()  {  if (Master != null && !string.IsNullOrEmpty(SelectedTime) && UserDate != null && UserDate > DateTime.Today)  {  UserTime userTime = Reservation.ParseHour(SelectedTime);  var db = DataConnection.GetInstance();  bool isDuplicateExists = db.Reservations.Any(r =>  r.Master != null &&  r.Master.Id == Master.Id &&  r.Date != null &&  r.Date == UserDate &&  r.Time != null &&  r.Time == userTime  );  if (isDuplicateExists)  {  MessageBox.Show("Извините, данный сеанс уже занят(");  }  else  {  db.Reservations.Add(new Reservation(Manager.CurrentUser as User, Master, UserDate, Reservation.ParseHour(SelectedTime)));  db.SaveChanges();  MessageBox.Show("Вы записались!");  }  }  else  {  MessageBox.Show("Пожалуйста, заполните все поля корректно.");  }  } |

Листинг команды FavoriteCommand

|  |
| --- |
| public class FavoriteCommand : Command  {  public override bool CanExecute(object parameter) => true;  public override void Execute(object parameter)  {  if (parameter is Sketch product)  {  var db = DataConnection.GetInstance();  var currentUser = Manager.CurrentUser as User;  var existingRecord = db.Favourites.FirstOrDefault(f => f.User.Id == currentUser.Id && f.Sketch.Id == product.Id);  if (existingRecord != null)  {  db.Favourites.Remove(existingRecord);  db.SaveChanges();  }  else  {  db.Favourites.Add(new Favourite(currentUser, product));  db.SaveChanges();  }  var window = SketchWindow.Instance;  if (window != null)  {  (window.DataContext as SketchViewModel).SearchSketch();  }  }  else  throw new NotImplementedException();  }  } |

Листинг класса Validator

|  |
| --- |
| public static class Validator  {  public static Regex loginRegex = new Regex(@"^(?=.\*[A-Za-z0-9]$)[A-Za-z][A-Za-z\d.-]{0,19}$");  public static Regex nameRegex = new Regex(@"^[А-Яа-яЁёA-Za-z \-]+$");  public static Regex phoneRegex = new Regex(@"^(\+375|80)(25|29|33|44)(\d{7})$");  public static Regex addressRegex = new Regex(@"^[А-Яа-яЁё\w \-,./]+$");  public static Regex eMailRegex = new Regex(@"^[\w-\.]+@([\w-]+\.)+[\w-]{2,4}$");  public static (bool isValid, List<string> forbiddenSymbols) Validate(string value, Regex regex)  {  if (regex.IsMatch(value))  return (true, null);  return (false, value.ToCharArray()  .Where(symbol => !regex.IsMatch(symbol.ToString()))  .Select(symbol => symbol.ToString()).ToList());  }  public static string JoinSymbols(List<string> symbols) => "\"" + string.Join("\", \"", symbols.Distinct()) + "\"";  } |