

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ
БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ
ГОСУДАРСТВЕННЫЙ
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий
Кафедра Программной инженерии
Специальность 6-05-0612-01 «Программная инженерия»
Направление специальности 6-05-0612-01 «Программная инженерия»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
КУРСОВОГО ПРОЕКТА:**

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

Исполнитель
Студент(ка) 2 курса группы 8 Мамонько Денис Александрович
(Ф.И.О.)

Руководитель работы доцент Белодед Н.И.
(учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой

Председатель Смелов В.В.
(подпись)

фамилия

ЗАДАНИЕ

К курсовому проектированию

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

Специальность: 6-05-0612-01 Программная инженерия

Группа: 8

Студент: Мамонько Денис Александрович

Тема: Программное средство «Кинотеатр»

1. Срок сдачи студентом законченной работы: «21 мая 2025 г.»

2. Исходные данные к проекту:

2.1. Функціонально ПС підтримує:

- Функции администратора:
 - Регистрировать заказ билетов.
 - Оповещать клиента о бронировании заказа.
 - Выполнять поисковые запросы, фильтрацию.
 - Выполнять сортировку списка фильмов.
- Функции клиента:
 - Заполнять форму для заказа билетов.
 - Просматривать информацию о статусе заказа.
 - Выполнять поисковые запросы, фильтрацию.
 - Оставлять и просматривать рейтинг/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

- Введение
- Постановка задачи и обзор литературы

- Проектирование архитектуры проекта
- Разработка функциональной модели и модели данных ПС
- 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	12.03.2025	
2	Аналитический обзор литературы по теме проекта. Изучение требований, определение вариантов использования	19.03.2025	
3	Анализ и проектирование архитектуры приложения (построение диаграмм, проектирование бизнес-слоя, представления и данных)	31.03.2025	
4	Проектирование структуры базы данных. Разработка дизайна пользовательского интерфейса	09.04.2025	
5	Кодирование программного средства	23.04.2025	
6	Тестирования и отладка программного средства	07.05.2025	
7	Оформление пояснительной записки	14.05.2025	
8	Защита проекта	21.05.2025	

5. Дата выдачи задания 05.03.2025

Руководитель _____ доц. Белодед Н.И.
(подпись)

Задание принял к исполнению _____ Мамонько Д.А.
(дата и подпись студента)

Оглавление

Введение	6
1 Постановка задачи и обзор литературы	7
1.1 Обзор аналогов	7
1.1.1 Анализ сайта Кинопоиск.ru	7
1.1.2 Анализ сайта mooon.by.....	8
1.1.3 Анализ сайта skyline.by	9
1.2 Постановка задачи	11
2 Анализ требований к программному средству и разработка функциональных требований.....	12
2.1 Описание средств разработки.....	12
2.2 Описание разрабатываемой функциональности программного средства.....	13
2.2.1 Пользовательские роли в системе	13
2.2.2 Функциональные требования	13
2.2.2.1 Общие функции системы.....	13
2.2.2.2 Функции администратора.....	14
2.2.2.3 Функции клиента	14
2.2.3 Диаграмма вариантов использования	14
2.2.4 Спецификация функциональных требований	15
2.2.4.1 Требования к регистрации и авторизации	15
2.2.4.2 Требования к работе с фильмами	15
2.2.4.3 Требования к работе с уведомлениями.....	15
2.2.4.4 Требования к администрированию	15
3 Проектирование программного средства	16
3.1 Архитектура системы.....	16
3.2 Взаимоотношения между классами	17
3.3 Проектирование модели базы данных	17
3.4 Проектирование последовательности взаимодействия.....	18
4 Реализация программного средства	19
4.1 Реализация архитектуры проекта на основе MVVM	19
4.2 Реализация ключевых функций	20
4.2.1 Реализация функции авторизации и регистрации	20
4.2.2 Реализация функции добавления фильмов	20
4.2.3 Реализация функции бронирования мест	21
4.3 Реализация модели данных.....	21
5 Тестирование, проверка работоспособности и анализ полученных результатов.....	23
5.1 Тестирование сервиса регистрации и авторизации.....	23
5.2 Тестирование функций администратора.....	25
5.2.1 Тестирование функции добавления фильма.....	25
5.2.2 Тестирование функции добавления расписания.....	26
5.3 Тестирование функций клиента	26
5.3.1 Функция изменения адреса электронной почты.....	26
5.3.2 Функция изменения пароля.....	27

6 Руководство по установке и использованию	28
6.1 Авторизация и регистрация	28
6.2 Руководство по использованию для клиента.....	29
6.3 Руководство по использованию для администратора	33
Заключение.....	36
Список использованных источников	37
Приложение А.....	38
Приложение Б	39
Приложение В.....	40
Приложение Г	41
Приложение Д.....	43
Приложение Е.....	44
Приложение Ж.....	45
Приложение З.....	47

Введение

В современном мире многие люди на регулярной основе посещают культурно-развлекательные мероприятия, такие как пьесы, кукольные театры и кинопоказы. Однако не каждый человек имеет возможность заблаговременно зарезервировать фиксированное количество билетов на то или иное мероприятие, тем самым обезопасить себя от ситуации, когда по приходу в заведение кинотеатра все билеты на желаемый фильм оказались распроданы. Более того, многие киноманы сталкиваются с проблемой просмотра на большом экране фильмов, выходявших многие годы назад, так как в современных кинотеатрах практически отсутствует такая возможность.

Данное программное устройство незаменимо для любителей реальных кинопоказов, поскольку благодаря интуитивно понятному пользовательскому интерфейсу позволяет самостоятельно выбрать желаемый фильм, предварительно ознакомившись с подробной информацией по каждому из них, не выходя из дома.

Не посещая специализированных учреждений и кинотеатров, пользователь может самостоятельно создать или войти в существующую учетную запись. Это дает возможность подробно ознакомиться с аннотациями к фильмам, расписанием сеансов, а также совершить заказ определенного числа билетов, выбрав удобную для себя дату показа.

Авторизированный пользователь имеет возможность просмотреть личный кабинет, подробную информацию о фильме, заказать билеты на определённый сеанс и оставить рейтинг после просмотра фильма.

Данное программное устройство незаменимо и с точки зрения составителя расписания кинофильмов, который имеет возможность управлять базой данных из административной части, создавать и удалять фильмы и расписание к ним, выполнять поиск и сортировку и иметь доступ ко всем заказам.

Главная задача данного курсового проектирования – это разработка программного средства, которое реализует все вышеперечисленные функции и решает поставленные задачи.

1 Постановка задачи и обзор литературы

1.1 Обзор аналогов

В данном разделе будет проведен обзор и анализ веб-сайтов, посвященных кинотеатрам, с целью выявления их особенностей, функциональных возможностей и уровня удобства для пользователей.

Этот анализ является ключевым этапом перед постановкой задачи для разработки программного средства «Кинотеатр».

Основные критерии, которые будут учитываться при анализе, включают функциональность платформы, интерфейс. Сравнив существующие решения, мы сможем определить сильные и слабые стороны конкурентов и на основе этого спроектировать более эффективный и удобный продукт.

1.1.1 Анализ сайта Кинопоиск.ru

Сайт «Кинопоиск» [1] является одним из самых популярных интернет-ресурсов в СНГ, собравшим полноценную информацию о мировой киноиндустрии. Главными преимуществами данного ресурса является его наполненность информацией и постоянное её обновление. Пользователь данного ресурса имеет возможность детально и подробно ознакомиться с каждым фильмом. Ещё одним немаловажным достоинством данного программного средства является возможность пользователей ознакомиться с рецензиями как обычных пользователей, так и профессиональных критиков, а также оставить свою рецензию. Данное программное средство предоставляет пользователю возможность выбрать фильм по определенным критериям и составить полноценное мнение. Также, помимо теоретического ознакомления с информацией о фильмах, пользователь имеет возможность узнать, в каких кинотеатрах города в определенный день показывается интересующий его фильм. Интерфейс интернет-ресурса представлен на рисунке 1.1.

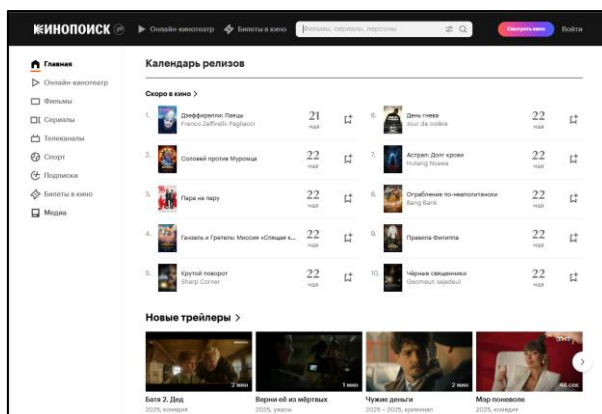


Рисунок 1.1 – Интерфейс Кинопоиск

Из выявленных недостатков данного ресурса необходимо выделить невозможность забронировать фиксированное количество билетов на реальный кинопоказ в кинотеатре.

1.1.2 Анализ сайта mooon.by

Мооон [2] – это полное переосмысление понятия «кинотеатр» и совершенно новый подход к созданию мультифункционального пространства для ярких впечатлений, комфорта и незабываемого времяпровождения для каждого из зрителей.

Интерфейс данного сайт представлен на рисунке 1.2.

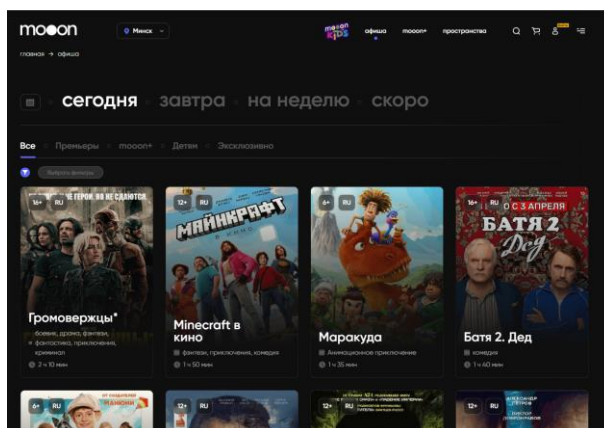


Рисунок 1.2 – Интерфейс мооон

Данный кинотеатр предлагает широкую линейку доступных фильмов для просмотра как от текущего дня, так и до дня премьер, которые скоро появятся в прокате. Также есть огромное количество фильтров, которые позволяют выбрать фильм по определенным категориям, например для детей, премьеры.

Также кинотеатр создает эксклюзивные проекты, позволяющие увидеть в кинопространстве большее, чем кино. Пример эксклюзивного контента кинотеатра мооон представлен на рисунке 1.3.

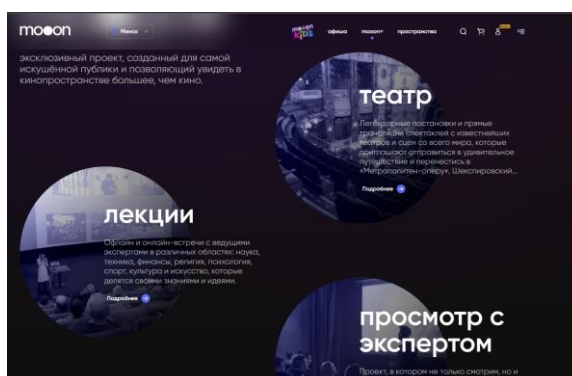


Рисунок 1.3 – Эксклюзивный контент кинотеатра мооон

В отличие от предыдущего аналога, система позволяет бронировать билеты на реальные сеансы с фиксированным количеством мест. Кроме этого, доступно множество вариантов по датам и залам, что делает процесс бронирования гибким и удобным.

Пример выбора сеанса в кинотеатре мооон представлен на рисунке 1.4.

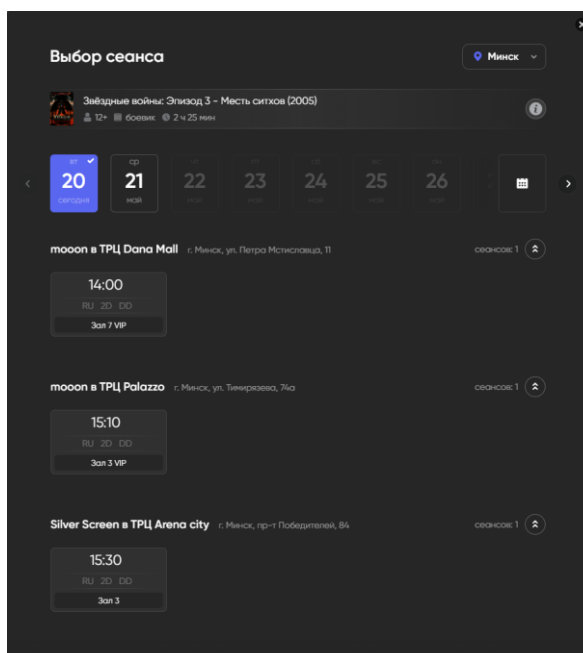


Рисунок 1.4 – Выбор сеанса на фильм в кинотеатре mooon

После того, как пользователь выберет фильм, который хочет посмотреть, ему нужно будет выбрать дату и зал, как в форме, которая представлена выше. После того, как он выберет дату и зал, ему открывается схема зала, где можно выбрать конкретное место и завершить бронирование. Пример схемы выбранного зала представлен на рисунке 1.5.

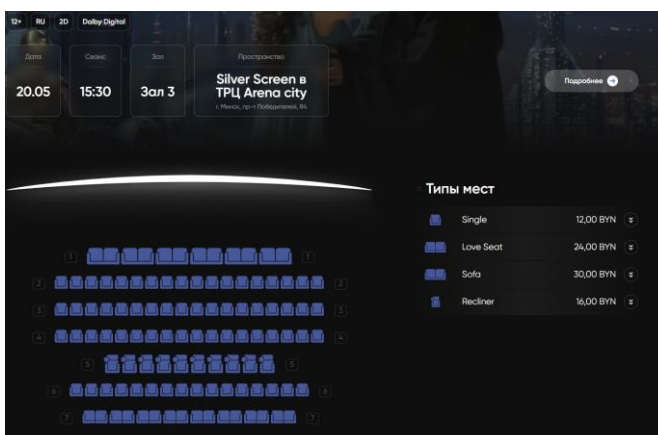


Рисунок 1.5 – Схема зала для брони в кинотеатре mooon

Из данного сайта можно взять пример выбора сеанса для определенного фильма, а также загрузку схемы определенного зала, где пользователь может выбрать место и завершить бронирование, после того как он выберет дату, время и зал.

1.1.3 Анализ сайта skyline.by

Skyline [3] – это современный кинотеатр, который сочетает в себе высокотехнологичные кинопоказы с продуманным комфортом и

дополнительными сервисами, делая посещение кино особым событием.

Интерфейс данного сайта чистый, интуитивно понятный и с удобной навигацией. Есть возможность просматривать афишу на ближайшие дни, а также предварительный выбор будущих премьер. Доступна фильтрация по жанрам, форматам (2D, 3D) и специальным категориям. Интерфейс данного сайта представлен на рисунке 1.6.

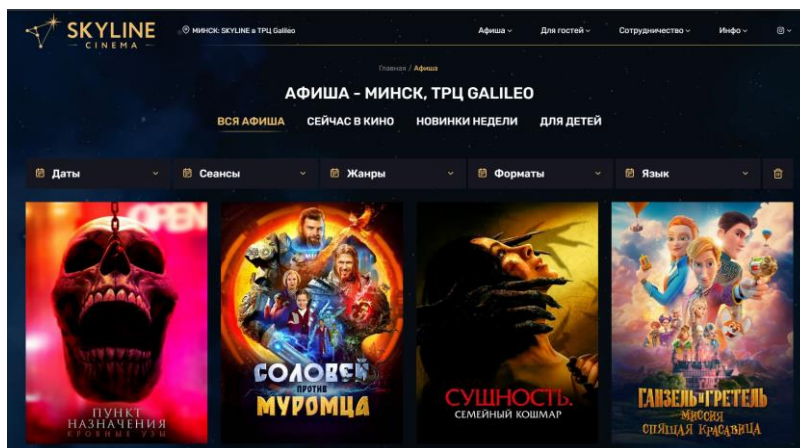


Рисунок 1.6 – Интерфейс skyline

Как и предыдущий аналог, данный сайт позволяет также бронировать билеты на реальные сеансы с фиксированным количеством мест. Пример странички выбора даты и зала в кинотеатре skyline представлен на рисунке 1.7.

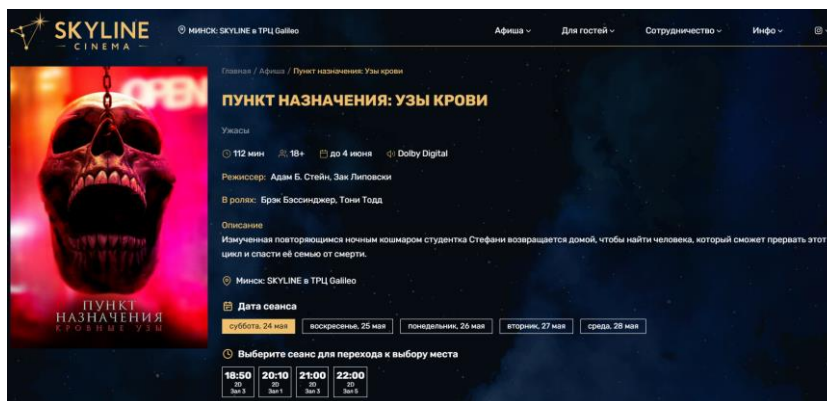


Рисунок 1.7 – Выбор сеанса на фильм в кинотеатре Skyline

Аналогично предыдущему аналогу, после выбора даты и зала, пользователю будет загружена схема соответствующего зала, где он может выбрать место и завершить бронирование. Пример схемы выбранного зала представлен на рисунке 1.8.

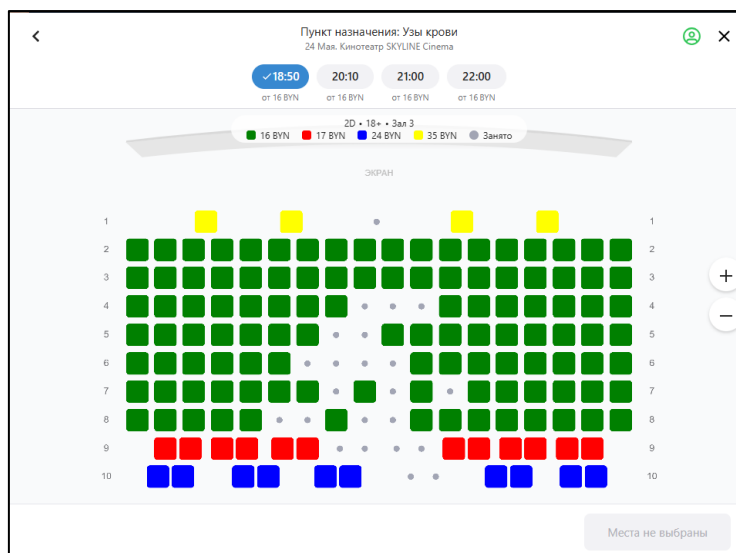


Рисунок 1.8 – Схема зала для бронирования в кинотеатре Skyline

Из данного сайта можно взять пример оформления схем залов, где определенные места раскрашены разными цветами и рядом указано сразу же стоимость за одно место, чтобы пользователь понимал, какие места лучше выбрать и сколько он за них заплатит.

1.2 Постановка задачи

Обзор вышеперечисленных известных 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:

- регистрация и авторизация: система регистрации и авторизации для доступа к функционалу приложения и управления личным кабинетом;
- оценки пользователей: функция оставления оценок после просмотра фильмов;
- уведомление пользователя о заказе билета;
- поиск и фильтрация: удобная система поиска фильмов по названию и фильтрация по жанрам, чтобы пользователи могли найти нужный им фильм;
- панель администратора: функционал для управления фильмами, заказами и расписанием с возможностью изменения контента.

2 Анализ требований к программному средству и разработка функциональных требований

2.1 Описание средств разработки

При разработке программного средства были использованы следующие технологии и инструменты:

- интегрированная среда разработки Microsoft Visual Studio 2022;
- платформа .NET 8.0;
- язык программирования C#;
- декларативный язык разметки XAML;
- технология WPF;
- архитектурный паттерн MVVM;
- entity Framework 9.0.3;
- Microsoft SQL Server.

Microsoft Visual Studio 2022 – интегрированная среда разработки с полной поддержкой платформы .NET и технологии WPF. В процессе работы активно использовались встроенный визуальный редактор XAML, система отладки с точками останова и мощный инструмент Intellisense для ускорения написания кода.

.NET 8.0 – современная высокопроизводительная платформа для разработки приложений. Обеспечивает доступ к обширному набору библиотек классов.

C# – объектно-ориентированный язык программирования, являющийся основным для разработки на платформе .NET. Отличается строгой типизацией и поддержкой различных парадигм программирования. В проекте активно применялись возможности LINQ для работы с коллекциями данных, асинхронное программирование с использованием ключевых слов `async/await` и система событий на основе делегатов.

XAML (eXtensible Application Markup Language) – декларативный язык разметки для определения пользовательского интерфейса в приложениях WPF. Позволяет четко разделить презентационную логику от бизнес-логики приложения.

WPF (Windows Presentation Foundation) – технология для создания настольных клиентских приложений под Windows, ориентированных на богатый пользовательский интерфейс.

Entity Framework 6.2.0 – объектно-реляционный маппер (ORM), позволяющий работать с базой данных через объекты .NET. В проекте применялся подход Code First, при котором структура базы данных определяется через классы C#. Преимуществами данного фреймворка стали автоматическая генерация SQL-запросов и система миграций для управления структурой БД.

Microsoft SQL Server – система управления базами данных, использующая реляционную модель. Используется в качестве сервера баз

данных для хранения информации, необходимой в работе программного средства.

Архитектурный паттерн MVVM (Model-View-ViewModel) – подход к проектированию приложений с пользовательским интерфейсом. Обеспечивает разделение приложения на три основных компонента: Model (модель данных и бизнес-логика), View (представление, пользовательский интерфейс) и ViewModel (модель представления, связующее звено между Model и View).

2.2 Описание разрабатываемой функциональности программного средства

На основе проведенного анализа аналогов и требований, указанных в постановке задач, была определена функциональность разрабатываемого программного средства «Кинотеатр». В данном разделе описаны основные функции системы, пользовательские роли и требования к взаимодействию пользователей с системой.

2.2.1 Пользовательские роли в системе

В программном средстве «Кинотеатр» выделены следующие пользовательские роли:

Администратор системы – пользователь с расширенными правами, осуществляющий контроль за работой платформы, модерацию фильмов, расписаний и управление заказами пользователей.

Клиент – зарегистрированный пользователь, имеющий возможность просматривать фильмы, осуществлять поисковые запросы и фильтрацию, бронировать места для просмотра фильмов, отслеживать статус заказов и изменять их, оставлять оценки после просмотра фильма.

Указанные роли определяют перечень доступных функций для каждого типа пользователя, обеспечивают разграничение прав доступа и способствуют структурированной работе всей системы.

2.2.2 Функциональные требования

2.2.2.1 Общие функции системы

Регистрация и авторизация пользователей:

- регистрация новых пользователей с указанием электронной почты, пароля и персональных данных;

- авторизация зарегистрированных пользователей.

Афиша фильмов:

- просмотр доступных фильмов с фильтрацией по жанрам;
- просмотр всей доступной информации о фильме, а именно: постер фильма, описание, возрастной рейтинг, название, продолжительность, режиссёр;

- поиск фильмов, которые имеют схожие признаки, а именно: схожие название, или один режиссер.

Залы:

- просмотр залов, которые предоставляет кинотеатр.

Система уведомлений:

- отправка уведомлений на электронную почту, вводимую при регистрации, при успешном бронировании фильма пользователем

2.2.2.2 Функции администратора

Управление фильмами. Администратор может просматривать все фильмы, которые есть в кинотеатре, добавлять фильмы, редактировать уже существующие и удалять, если есть такая необходимость.

Управление заказами пользователей. Предусмотрена возможность менять статусы заказов пользователей, а также удалять.

Управление расписанием для фильмов. Администратор может добавлять расписание на определенную дату, время и зал. При необходимости можно удалить расписание.

Оповещение клиентов. Система отправляет пользователям уведомление по электронной почте при бронировании места в кинотеатре на определенный сеанс. Также предусмотрены внутренние системные уведомления, отображающиеся в интерфейсе приложения.

2.2.2.3 Функции клиента

Просмотр фильмов. Клиент может просматривать все фильмы, которые доступны в кинотеатре, может выполнять функции поиска фильма по названию или по режиссеру, выполнять фильтрацию по жанрам.

Бронирование мест. После выбора фильма, даты и времени, пользователь может перейти к схеме зала и забронировать фиксированное количество мест для просмотра фильма.

Управление активностью. Клиент имеет доступ к истории всех своих заказов, включая забронированные, отмененные и просмотренные заказы.

Работа с оценками фильма. Клиент может поставить оценку фильму после того, как его просмотрит.

2.2.3 Диаграмма вариантов использования

Для наглядного представления функциональных требований была разработана диаграмма вариантов использования, отражающая взаимодействие пользователей с системой. Данная диаграмма вариантов использования представлена в приложении А.

2.2.4 Спецификация функциональных требований

2.2.4.1 Требования к регистрации и авторизации

Система должна предоставлять форму для регистрации, которая включает поля: имя пользователя, фамилию пользователя, электронную почту, пароль и подтверждение пароля. Пароль должен быть длиной не менее 8 и не более 20 символов, содержать буквы верхнего и нижнего регистра, цифры и спецсимволы. Для букв можно использовать только латинский алфавит. После заполнения формы и ее отправки система должна проверить уникальность указанного адреса электронной почты.

Для авторизации пользователя система должна предоставлять форму с полями для ввода электронной почты и пароля. При успешной авторизации пользователь будет перенаправлен на главную страницу. В случае неудачной попытки авторизации система должна выводить сообщение о ошибке.

2.2.4.2 Требования к работе с фильмами

Для добавления фильма система должна предоставлять форму с полями для ввода и выбора: название фильма, режиссёр, продолжительность, жанр, возрастной рейтинг, описание фильма. Также необходимо предусмотреть возможность загрузки одной фотографии, которая будет выступать в качестве постера к фильму. Эта фотография должна автоматически масштабироваться до приемлемого размера.

Пользователи должны иметь возможность просматривать фильмы, которые доступны в кинотеатре. При выборе конкретного фильма должна открываться страница с его детальным описанием, постером фильма и сеансами, которые актуальны для данного фильма.

Для бронирования мест для просмотра фильма необходимо выбрать интересующий пользователя фильм, затем выбрать удобное расписание среди доступных и перейти к бронированию мест в зале, который загрузится после выбора даты и времени. По завершении бронирования на почту пользователя придет уведомление о том, что он забронировал себе место/места на определенный фильм, дату и время.

2.2.4.3 Требования к работе с уведомлениями

Система должна поддерживать два типа уведомлений: системные уведомления внутри приложения и уведомления по электронной почте для важных событий.

2.2.4.4 Требования к администрированию

Для модерации фильмов администратор должен иметь интерфейс для просмотра всех фильмов. Он должен иметь возможность удалять фильмы.

В разделе управления заказами администратор должен иметь возможность просматривать список заказов пользователей с возможностью фильтрации по статусу и с помощью кнопок управления или менять статусы заказов, или вовсе удалять их.

В разделе управления сеансами администратор должен иметь возможность просматривать список сеансов всех фильмов, которые являются актуальными и активными в данный момент времени. Он должен иметь возможность удалять сеансы с помощью кнопок управления.

Система должна предоставлять интерфейс для редактирования фильма.

Система должна предоставлять интерфейс отправки уведомлений пользователям от лица системы.

3 Проектирование программного средства

3.1 Архитектура системы

Программное средство «Кинотеатр» разработано с использованием архитектурного паттерна MVVM (Model-View-ViewModel), который обеспечивает четкое разделение ответственностей между компонентами приложения, что способствует улучшению структуры кода, упрощает тестирование и обеспечивает легкую поддержку.

Архитектура MVVM состоит из трех основных компонентов:

- Model (Модель) – представляет данные и бизнес-логику приложения;
- View (Представление) – определяет структуру, компоновку и внешний вид того, что пользователь видит на экране;
- ViewModel (Модель представления) - выступает в качестве посредника между представлением и моделью.

Структура шаблона MVVM представлена на рисунке 3.1.

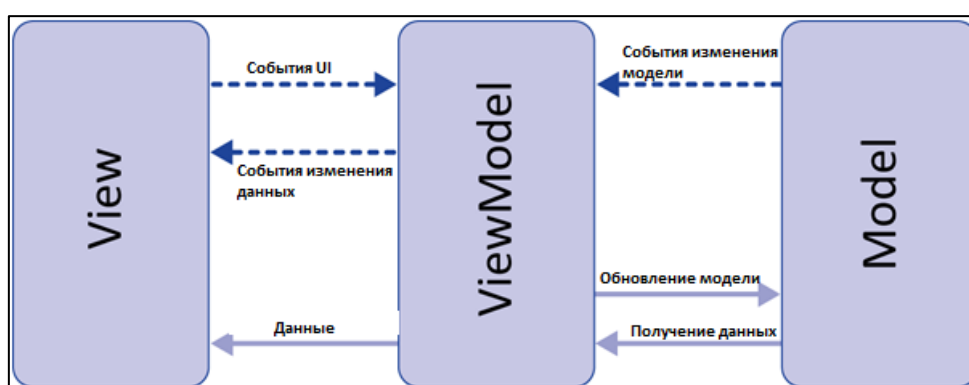


Рисунок 3.1 – Структура шаблона MVVM

Помимо основного паттерна MVVM, в архитектуре приложения применяются следующие паттерны проектирования:

- команда (Command) – для инкапсуляции действий пользовательского интерфейса в виде объектов;

– наблюдатель (Observer) – реализованный через механизм привязки данных WPF, обеспечивает автоматическое обновление пользовательского интерфейса при изменении модели данных.

3.2 Взаимоотношения между классами

Для визуализации структуры системы и отображения взаимосвязей между её основными компонентами была построена диаграмма классов, представленная в Приложении Б.

Диаграмма классов является одним из ключевых элементов UML и служит для описания классов, их свойств, методов, а также связей между ними – таких как ассоциации, наследование, агрегация и композиция.

Созданная диаграмма отражает архитектуру приложения, включая основные сущности, такие как пользователь, фильм, зал, заказ и расписание, а также характер их взаимодействия. Она демонстрирует применённые принципы объектно-ориентированного проектирования и помогает лучше понять структуру системы, что облегчает её сопровождение и возможное расширение в будущем.

3.3 Проектирование модели базы данных

Для хранения данных программного средства «Кинотеатр» разработана реляционная модель базы данных, которая позволяет эффективно организовать информацию о пользователях, фильмах, залах, расписаниях и заказах. Логическая модель базы данных, отражающая основные таблицы, их атрибуты и связи между ними, представлена на рисунке 3.2.

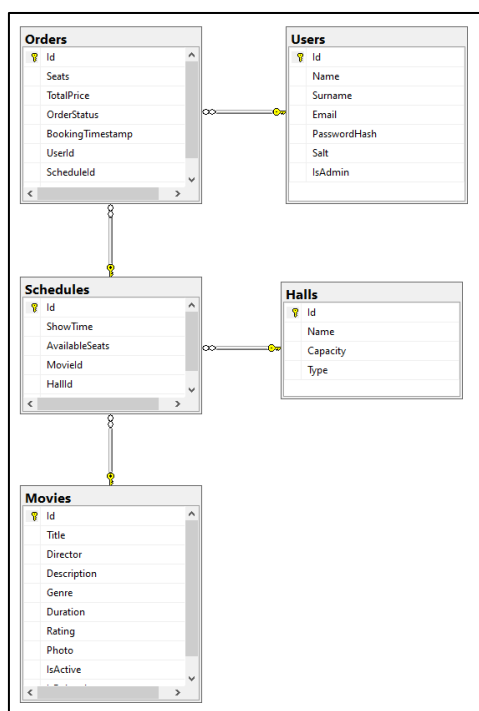


Рисунок 3.2 – Логическая модель базы данных

Разработанная модель базы данных охватывает основные сущности программного средства «Кинотеатр» и отражает логические взаимосвязи между ними – пользователями, фильмами, залами, расписаниями и заказами. Реляционный подход обеспечивает структурированное хранение данных и поддержку бизнес-логики. Ограничения целостности (первичные и внешние ключи, уникальность, значения по умолчанию) способствуют сохранению согласованности данных и корректной работе системы.

3.4 Проектирование последовательности взаимодействия

В процессе проектирования приложения для отображения взаимодействия между его объектами была использована диаграмма последовательности, являющаяся одним из основных инструментов языка UML.

Этот тип диаграмм позволяет отразить порядок обмена сообщениями между участниками процесса и широко применяется для описания логики работы системы, взаимодействия между её компонентами, пользовательских сценариев и процессов обработки данных.

Диаграмма последовательности, демонстрирующая взаимодействие пользователя с системой во время авторизации, просмотра лотов и создания ставки, приведена в Приложении В.

4 Реализация программного средства

4.1 Реализация архитектуры проекта на основе MVVM

Программное средство «Кинотеатр» реализовано в соответствии с архитектурным паттерном MVVM, который обеспечивает четкое разделение ответственности между компонентами приложения, описание паттерна представлено в главе 3.1. Структура проекта показана на рисунке 4.1.

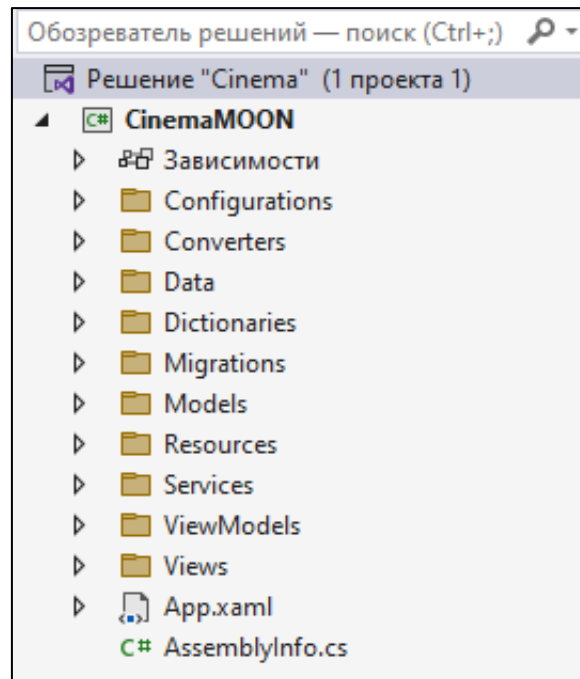


Рисунок 4.1 – Структура проекта

Программное средство структурировано в виде отдельных модулей, каждый из которых выполняет определённую роль в системе:

- Configurations – содержит конфигурации классов сущностей;
- Converters – содержит классы преобразования данных между различными форматами для корректного отображения в интерфейсе;
- Data – модуль для работы с данными, включающий классы доступа к базе данных, определения контекста данных и конфигурации подключений;
- Dictionaries – содержит словари ресурсов для языка;
- Migrations – содержит файлы миграций базы данных, обеспечивающие управление версионированием схемы БД и возможность обновления структуры данных;
- Models – классы, описывающие основные сущности базы данных;
- Resources – содержит статические ресурсы проекта (изображения, темы), используемые в пользовательском интерфейсе;
- Services – классы, предоставляющие различные сервисные функции;
- ViewModels – содержит классы, реализующие бизнес-логику приложения и обработку взаимодействия с пользователем;

– Views – XAML-файлы, реализующие визуальные компоненты интерфейса приложения.

4.2 Реализация ключевых функций

В данном разделе представлены ключевые функции программного обеспечения «Кинотеатр», реализованные в соответствии с требованиями, изложенными в предыдущих главах. Каждая функция была разработана с учетом архитектурного паттерна MVVM.

4.2.1 Реализация функции авторизации и регистрации

Функции авторизации и регистрации являются базовыми для программного средства, поскольку обеспечивают доступ к системе и определяют права пользователя. Процесс регистрации включает в себя следующие этапы:

- ввод регистрационных данных (имя, фамилия, электронная почта, пароль, подтверждение пароля);
- валидация введенных данных;
- проверка уникальности электронной почты в системе;
- хэширование пароля для повышения безопасности;
- сохранение данных о пользователе в базе данных.

Процесс авторизации включает следующие шаги:

- ввод пользователем электронной почты и пароля;
- поиск пользователя в базе данных по электронной почте;
- верификация пароля путем сравнения хешей.

За процесс авторизации и регистрации в проекте отвечает класс User, он хранит информацию о текущем пользователе и использует сервис PasswordHasher, для хэширования пароля.

Реализация функции регистрации представлена в приложении Г, а функция авторизация в приложении Д.

4.2.2 Реализация функции добавления фильмов

Функция добавления фильмов позволяет пользователям с ролью «Администратор» размещать фильмы в афише кинотеатра. Процесс добавления реализован в несколько этапов:

- заполнение формы с информацией о фильме;
- загрузка постера фильма;
- валидация данных;
- сохранение фильма в базе данных.

Реализация функции добавления фильма вместе с функцией загрузки изображения представлена в приложении Е.

4.2.3 Реализация функции бронирования мест

Функция бронирования мест на определенный сеанс является одной из ключевых для системы кинотеатра и реализована следующим образом:

- выбор сеанса, который доступен для определенного фильма;
- выбор места на схеме зала, который загружается следом за датой и временем сеанса;
- проверка возможности забронировать место (место может быть занято);
- выбор фиксированного количества мест для брони;
- подтверждение бронирования;
- система уведомлений.

Система уведомлений посылает сообщение клиенту о том, что он забронировал место/места на определенный сеанс.

Реализация функции бронирования мест на определенный сеанс представлена в приложении Ж.

4.3 Реализация модели данных

Модель данных в системе кинотеатра представляет собой ключевую часть архитектуры, отвечающую за хранение и управление всей информацией, связанной с фильмами, пользователями, расписаниями и другими сущностями. В системе используется объектно-реляционное отображение.

Данная модель реализована с использованием Entity Framework Core по принципу Code First, что позволило описать структуру базы данных через классы. Структура базы данных была подробно описана в разделе 3.3.

Пример подхода Code First представлен в листинге 4.1.

```
namespace CinemaMOON.Models
{
    public class User
    {
        public Guid Id { get; set; }
        public string Name { get; set; }
        public string Surname { get; set; }
        public string Email { get; set; }
        public string PasswordHash { get; set; }
        public string Salt { get; set; }
        public bool IsAdmin { get; set; }
        public ICollection<Order> Orders { get; set; } = new
List<Order>();
    }
}
```

Листинг 4.1 – Реализация модели данных пользователя

Также, для реализации принципа Code First был создан класс ApplicationContext, который наследуется от DbContext. Этот класс служит для

описания структуры базы данных и управления взаимодействием с ней. Он определяет таблицы, связи между сущностями и настройки для каждой модели. Через `ApplicationContext` осуществляется выполнение операций с базой данных, таких как добавление, изменение, удаление и чтение данных. Класс также используется для применения миграций, что позволяет автоматически обновлять структуру базы данных при изменении модели данных.

Реализация данного класса представлена в приложении 3.

5 Тестирование, проверка работоспособности и анализ полученных результатов

5.1 Тестирование сервиса регистрации и авторизации

Сервис регистрации и авторизации является важным компонентом системы, обеспечивающим безопасность пользователей и контроль доступа к функционалу приложения кинотеатра. Для проверки работоспособности данного сервиса были проведены несколько ручных тестов.

В ходе тестирования были проверены как типичные случаи, так и нестандартные.

При регистрации нового аккаунта пользователь обязан ввести email, соответствующий установленному формату. В случае ввода некорректного адреса электронной почты система автоматически выявляет ошибку и выводит соответствующее уведомление с пояснением. Такая проверка осуществляется на этапе валидации данных и предотвращает возможность регистрации с недействительным или ошибочным email. Обработка данной представлена на рисунке 5.1.

The screenshot shows a registration form with the following fields and values:

- Имя**: Иван
- Фамилия**: Александрович
- Email**: examplegmail.com
- Пароль**: [masked with dots]
- Подтверждение пароля**: [masked with dots]

Below the email field, there is a red error message: "Неверный формат email". At the bottom of the form, there are two buttons: "Назад" (Back) and "Зарегистрироваться" (Register).

Рисунок 5.1 – Обработка неверного формата email

При вводе пароля, не соответствующего установленным требованиям (недостаточная длина, отсутствие заглавных букв, цифр или специальных символов), система должна корректно обрабатывать данную ситуацию. Пользователю выводится информативное сообщение об ошибке. Это исключение предотвращает отправку некорректных данных и повышает безопасность и удобство использования системы. Пример обработки такого случая показан на рисунке 5.2.

Регистрация

Имя
Иван

Фамилия
Александрович

Email
example@gmail.com

Пароль
●●●●●●●●

Пароль: 8-20 симв., заглавная, строчная, цифра, спецсимвол
(Пример: Password123!)

Подтверждение пароля
●●●●●●●●

Пароли не совпадают

Назад Зарегистрироваться

Рисунок 5.2 – Обработка некорректного пароля

При вводе пароля в процессе авторизации система выполняет сравнение хэша введённого значения с хранимым в базе данных. Это означает, что введённый пароль преобразуется в хэш с использованием того же алгоритма, что и при регистрации, и только затем происходит сверка. Если полученные хэши не совпадают, пользователю отображается сообщение об ошибке с пояснением, что введён неверный пароль. Это позволяет обеспечить безопасность и защиту пользовательских данных. Пример отображения такой ошибки показан на рисунке 5.3.

Добро пожаловать в кинотеатр!

Email
denismmnk@gmail.com

Пароль
●●●●●●●●

Неверный пароль

Войти Зарегистрироваться

Рисунок 5.3 – Обработка неправильного пароля

Тестирование сервисов регистрации и авторизации подтвердило стабильную работу системы, корректную обработку ошибок и надёжную защиту данных пользователей.

5.2 Тестирование функций администратора

5.2.1 Тестирование функции добавления фильма

При добавлении фильма необходимо предусмотреть обработку разных ситуаций, например, когда администратор пытается установить продолжительность для нового фильма, которая выходит за определенный диапазон. Пример отображение такой ошибки представлен на рисунке 5.4.

The screenshot shows a web form titled "Добавить фильм". It contains several input fields: "Название фильма" (movie title) with "asdfasdf", "Режиссёр" (director) with "sadsadf", "Жанр" (genre) set to "Боевик" (Action), and "Продолжительность (мин)" (duration in minutes) with "1234". A red error message below the duration field reads: "Длительность должна быть от 1 до 300 минут". The "Возрастной рейтинг" (age rating) is set to "6+". To the right, there is a description field with "asdfsafsafa" and a movie poster for "Pulp Fiction". Buttons at the bottom include "Добавить фильм" (disabled) and "Очистить форму" (Reset form).

Рисунок 5.4 – Обработка ввода некорректной продолжительности

Также стоит отметить, что кнопка добавления фильма находится неактивной до тех пор, пока администратор не исправит введенное значение в поле на корректное.

Дополнительно можно рассмотреть вариант ошибки, когда администратор пытается добавить постер к фильму, который уже закреплен за каким-то другим фильмом, находящемся в афише кинотеатра. Пример отображения данной ошибки представлен на рисунке 5.5.

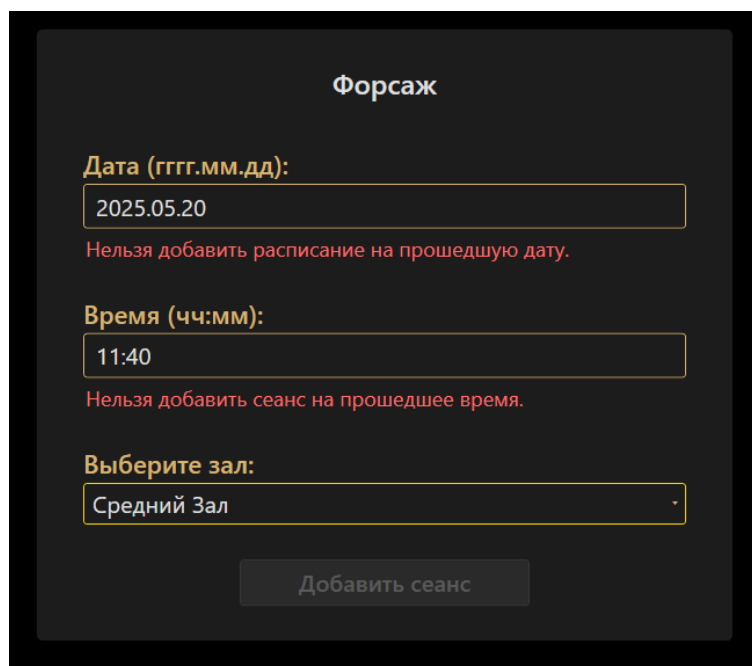
The screenshot shows the same "Добавить фильм" form as in Figure 5.4, but with the duration field set to "123". The "Продолжительность (мин)" field now has "123" instead of "1234". The "Добавить фильм" button remains disabled. The error message below the duration field is: "Это изображение уже используется для другого фильма" (This image is already used for another movie). The movie poster for "Pulp Fiction" is still visible on the right.

Рисунок 5.5 – Обработка добавления используемого постера к фильму

При попытке вставить фотографию, которая уже используется в другом фильме, система сообщит о том, что данный постер уже используется.

5.2.2 Тестирование функции добавления расписания

При добавлении расписания к фильму, следует учесть несколько важных проверок, а именно корректность даты и времени. Пример ошибок, если пользователь введет некорректные значения даты и времени, представлен на рисунке 5.6.



The screenshot shows a dark-themed interface for the movie 'Форсаж' (Fast & Furious). It contains a form with three input fields and a button. The first field, labeled 'Дата (гггг.мм.дд):', contains '2025.05.20' and has a red error message below it: 'Нельзя добавить расписание на прошедшую дату.' The second field, labeled 'Время (чч:мм):', contains '11:40' and has a red error message below it: 'Нельзя добавить сеанс на прошедшее время.' The third field, labeled 'Выберите зал:', is a dropdown menu with 'Средний Зал' selected. At the bottom is a button labeled 'Добавить сеанс'.

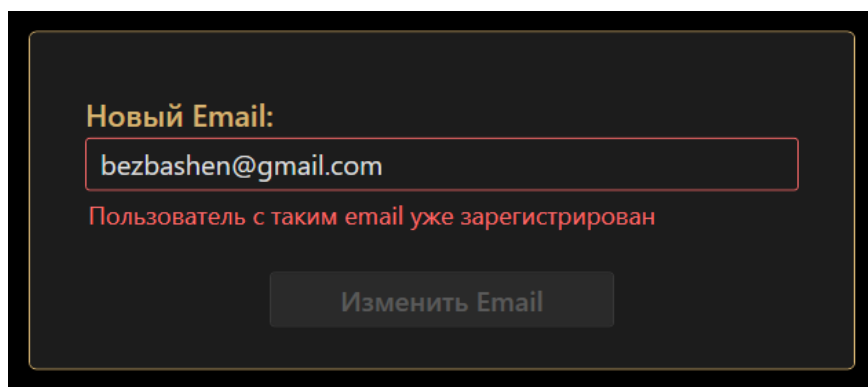
Рисунок 5.6 – Обработка ввода некорректной даты и времени

Кнопка будет неактивна до тех пор, пока пользователь не введет корректные значения даты и времени. Как только он это исправит, он сможет добавить сеанс для определенного фильма и для зала, который также выбирается при добавлении расписания.

5.3 Тестирование функций клиента

5.3.1 Функция изменения адреса электронной почты

Клиенту доступна возможность изменения адреса электронной почты. Данная функция должна обрабатывать ошибки по типу ввода неверного формата email, изменение адреса электронной почты на текущий или на тот, который уже имеет другой пользователь. Пример ошибки, если пользователь попытается изменить адрес электронной почты на тот, под которым уже зарегистрирован другой пользователь, представлен на рисунке 5.7.



Новый Email:

bezbashen@gmail.com

Пользователь с таким email уже зарегистрирован

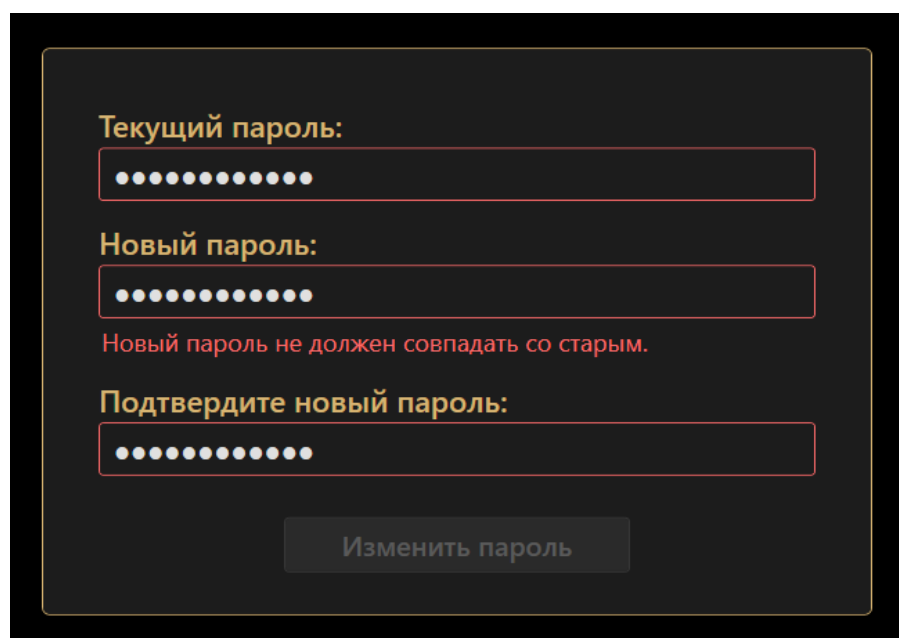
Изменить Email

Рисунок 5.7 – Обработка email, который занят другим пользователем

Тут также есть кнопка, которая остается неактивной до тех пор, пока пользователь не введет другой адрес электронной почты.

5.3.2 Функция изменения пароля

Клиент также имеет возможность поменять свой текущий пароль на новый. Для этого ему нужно заполнить простую форму, где нужно заполнить три поля: текущий пароль, новый пароль и подтверждение нового пароля. Данная функция должна обрабатывать следующие ошибки: пароль не соответствует корректному формату, нельзя изменить пароль в том случае, если он совпадает со старым. Пример ошибки, если пользователь хочет поменять пароль, который совпадает с текущим, представлен на рисунке 5.8.



Текущий пароль:

Новый пароль:

Новый пароль не должен совпадать со старым.

Подтвердите новый пароль:

Изменить пароль

Рисунок 5.8 – Обработка нового пароля, который совпадает с текущим

Тут также есть кнопка, которая остается неактивной до тех пор, пока пользователь не введет другой пароль, отличающийся от текущего, и который будет соответствовать корректному формату.

6 Руководство по установке и использованию

6.1 Авторизация и регистрация

Пользователь, впервые заходя в приложение, должен пройти процесс регистрации. На этапе регистрации ему нужно будет ввести следующие данные: имя, фамилию, адрес электронной почты, пароль и подтверждение пароля. Окно регистрации представлено на рисунке 6.1.

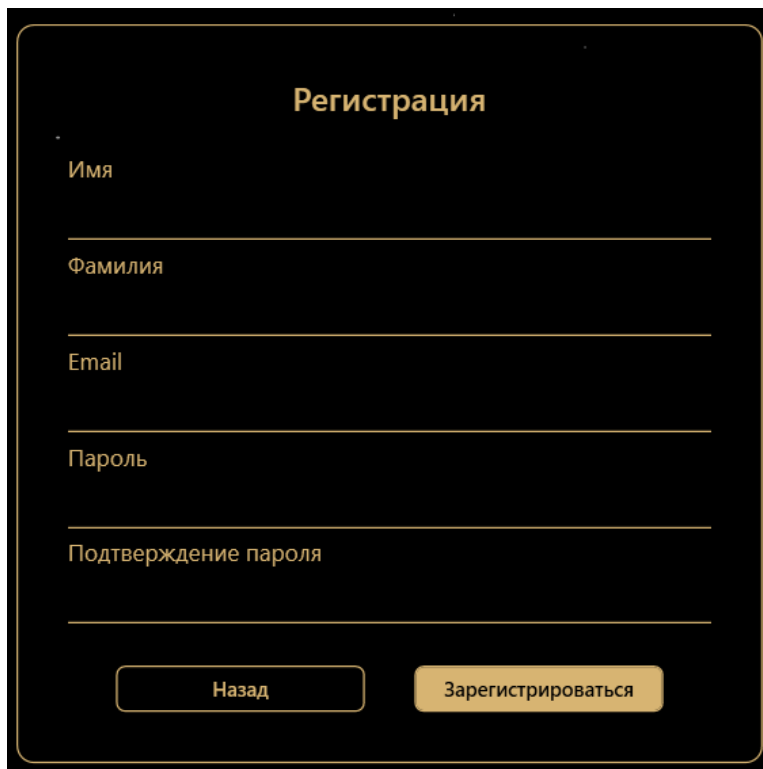
The image shows a registration form titled "Регистрация" (Registration) in a dark-themed window. The form contains five input fields with labels in Russian: "Имя" (Name), "Фамилия" (Surname), "Email", "Пароль" (Password), and "Подтверждение пароля" (Confirm password). Below the fields are two buttons: "Назад" (Back) and "Зарегистрироваться" (Register).

Рисунок 6.1 – Окно регистрации

При регистрации должны быть выполнены следующие проверки:

- имя должно содержать от 2 до 20 символов;
- фамилия должна содержать от 2 до 25 символов (дефис допускается);
- адрес электронной почты должен быть корректным;
- пароль должен соответствовать определенному формату;

После успешного прохождения всех проверок и завершения регистрации, пользователь автоматически перенаправляется на окно авторизации, где он может войти в систему, используя указанные при регистрации учетные данные. Интерфейс окна авторизации представлен на рисунке 6.2 и обеспечивает удобный и безопасный вход в систему для дальнейшей работы.

Добро пожаловать в кинотеатр!

Email

Пароль

Войти Зарегистрироваться

Рисунок 6.2 – Окно авторизации

В окне авторизации пользователь должен ввести те же данные, которые были указаны при регистрации – адрес электронной почты и пароль. Если введенные данные совпадают с теми, которые были сохранены при регистрации, пользователь будет успешно авторизован и получит доступ к основным функциям приложения.

Также реализованы кнопки перехода между окнами регистрации и авторизации для более лучшего взаимодействия пользователя и системы.

6.2 Руководство по использованию для клиента

После успешной авторизации пользователь с ролью «Клиент» попадает на главный экран приложения. С данного экрана доступны основные разделы, необходимые для взаимодействия с кинотеатром:

- переход к афише со всеми фильмами – позволяет просмотреть все фильмы, которые доступны в кинотеатре, отсортированные по алфавиту;
- переход на страницу с залами – просмотр залов, которые предлагает кинотеатр и соответствующих мест залов, а также описание этих мест;
- переход на страницу личного кабинета – просмотр всех заказов пользователя, а также просмотр информации о пользователе и возможность изменить свои учетные данные.

Каждый фильм можно открывать для подробного просмотра: пользователь увидит описание фильма, постер фильма, название фильма, продолжительность, возрастной рейтинг, режиссёра, жанр, а также загруженные сеансы, если такие имеются. Иначе пользователь увидит сообщение о том, что расписание к данному фильму будет загружено позже.

Интерфейс главного экрана для пользователя с ролью «Клиент» разработан с учётом удобства и простоты взаимодействия с системой. Он предоставляет доступ ко всем необходимым функциям и разделам.

Главная страница показана на рисунке 6.3.

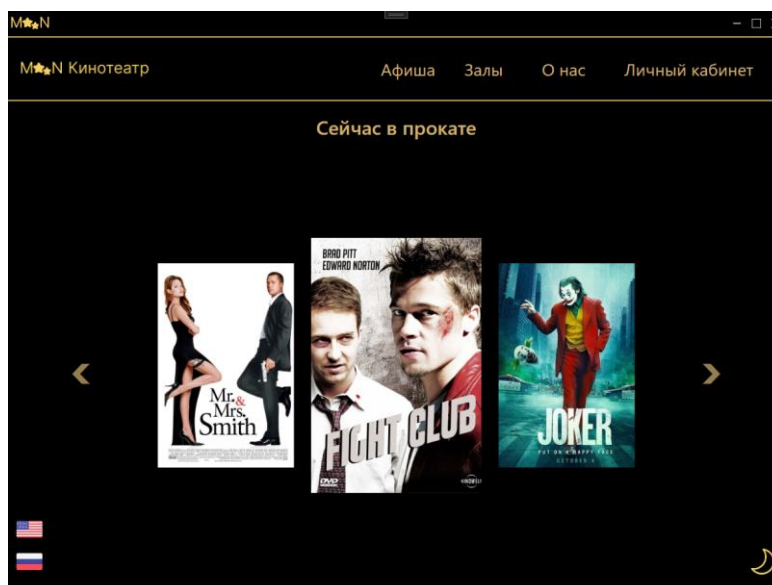


Рисунок 6.3 – Главная страница

Также на главном окне у клиента в верхней панели присутствует 5 следующих разделов:

- главная страница;
- афиша;
- страница с залами;
- о нас;
- личный кабинет;

Данная модель помогает в быстрой навигации для приложения.

При нажатии на раздел «Афиша» мы попадаем на новую страницу, на которой отображаются постеры фильмов и под каждой из них есть кнопка подробнее, при нажатии на которую пользователь перейдет на новую страницу и увидит всю подробную информацию о фильме, который он выбрал. Также на странице с афишей можно фильтровать фильмы по жанрам и осуществлять поисковые запросы по названию фильма или по режиссёру. Страница с афишей фильмов представлена на рисунке 6.4.

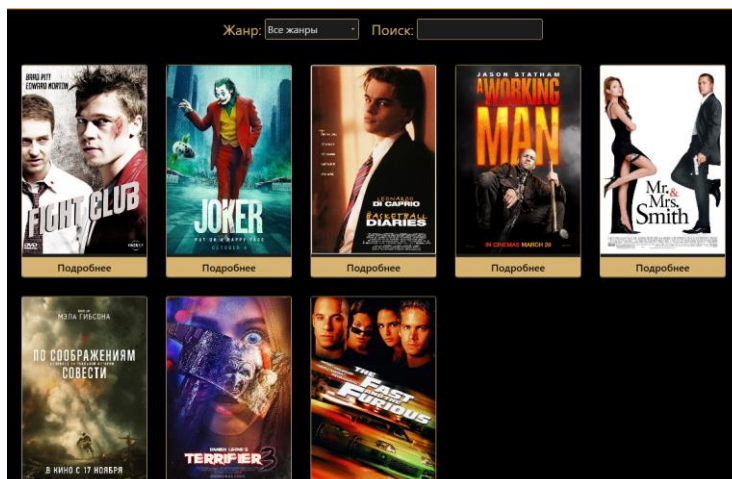


Рисунок 6.4 – Страница с афишей фильмов

При нажатии на кнопку подробнее под одним из фильмов, открывается страница с подробным описанием фильма. Данная страница отображена на рисунке 6.5.

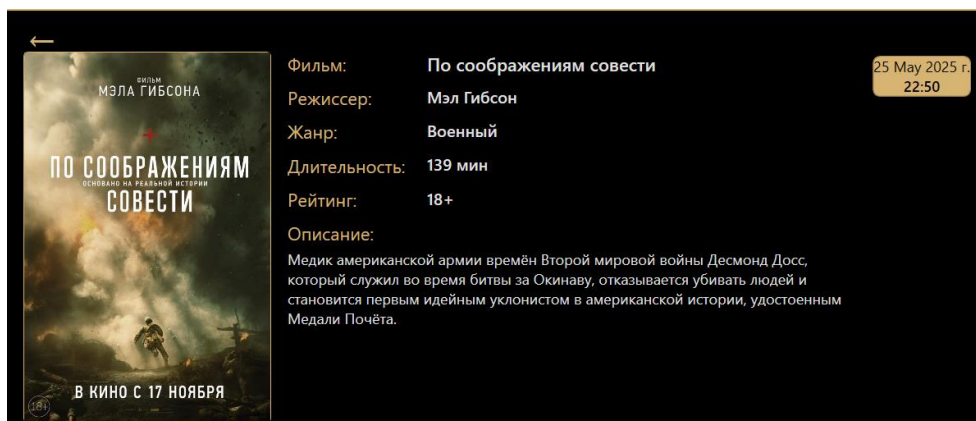


Рисунок 6.5 – Страница с подробным описанием фильма

На странице с описанием фильма отображаются активные сеансы, на которые можно забронировать места для просмотра фильма. Сеанс представлен в виде кнопки, на которой указаны дата и время сеанса. При нажатии на один из доступных сеансов, пользователю открывается новая страничка, на которой отображена схема зала, в котором будут показывать данный фильм. Данная страница отображена на рисунке 6.6.

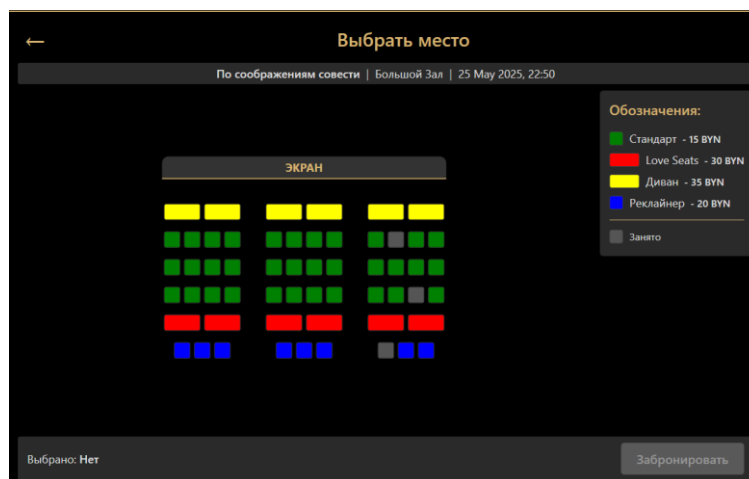


Рисунок 6.6 – Страница для бронирования мест

Места на данной схеме являются кликабельными за исключением тех мест, которые обозначены серым цветом. Это значит, что место занято и недоступно для выбора. В каждом из залов можно выбирать фиксированное число мест для брони: три места. Выбранные места будут отображаться в нижней панели в следующем формате: P1 M1. Данная запись означает, что пользователь выбрал первый ряд и первое место в данном ряду. Если пользователь выбрал несколько мест, то они будут выводиться перечислением через запятую в точно таком же формате.

После того, как пользователь выбрал места, ему будет доступна кнопка забронировать. После того, как он её нажмет, система его оповестит о том, что пользователь забронировал места для просмотра и ему придет оповещение о успешном бронировании на почту. После чего, пользователь в своем личном кабинете сможет увидеть свой забронированный заказ. Также этот заказ будет виден в окне заказов для администратора.

В личном кабинете помимо забронированных заказов, пользователю видны другие заказы, которые находятся в статусе просмотрено или отменено. Страница личного кабинета представлена на рисунке 6.7.

Фильм	Дата	Время	Зал	Места	Цена	Статус заказа	Ваша оценка
По соображениям совести	25.05.2025	22:50	Большой Зал	P2 M10, P4 M11, P6 M7	50.00 BYN	Забронировано	
Мистер и миссис Смит	29.05.2025	22:40	Большой Зал	P2 M10	15.00 BYN	Просмотрено	10

Рисунок 6.7 – Личный кабинет пользователя

Также можно обратить внимание на то, что пользователь может сам управлять своими заказами, например он может отменить забронированный заказ или поставить оценку тому фильму, который находится в статусе просмотрено. Если пользователь захочет поставить оценку фильму после его просмотра, то ему нужно будет выбрать тот фильм, который он просмотрел и нажать на кнопку поставить оценку. Затем ему откроется форма для заполнения, где будет одно поле, в которое нужно ввести оценку от 1 до 10 для соответствующего фильма. Страница для оценки фильма представлена на рисунке 6.8.

Рисунок 6.8 – Страница с оценкой фильма

Все что останется сделать пользователю – это оставить оценку для просмотренного фильма. И когда он это сделает, поставленная оценка отобразится в заказах клиента и администратора. И после того, как он поставит оценку, изменить он ее сможет. Для того, чтобы это сделать, пользователю нужно будет пересмотреть этот фильм еще раз, если будут доступны сеансы или ожидать, когда данный фильм появится в прокате снова.

Представлены все ключевые возможности интерфейса: навигация по фильмам, фильтрация и поиск, система уведомлений, работа с личным кабинетом и история заказов. Описанные функции демонстрируют удобство и полноту взаимодействия с системой кинотеатра, обеспечивая клиенту доступ к необходимой информации и действиям в интуитивно понятной форме.

6.3 Руководство по использованию для администратора

При входе за администратора системы у нас открывается админ панель, в которой имеются следующие разделы:

- главная страница;
- афиша;
- залы;
- о нас;
- панель администратора.

При переходе на раздел «Панель администратора», нам открывается страница, в которой представлен список добавленных фильмов, возможность сортировки, фильтрации и поиска, а также ряд кнопок для взаимодействия с фильмами и расписанием. Данная страница представлена на рисунке 6.9.

Поиск	Название	Режиссёр	Жанр	Длительность	Рейтинг
<div></div>	Бойцовский клуб	Дэвид Финчер	Триллер	139 мин	18+
<div>Добавить фильм</div>	Джокер	Тодд Филлипс	Триллер	122 мин	18+
<div>Редактировать фильм</div>	Дневник баскетболиста	Скотт Килверт	Драма	102 мин	18+
<div>Удалить фильм</div>	Мастер	Дэвид Эйр	Боевик	116 мин	18+
<div>Добавить расписание</div>	Мистер и миссис Смит	Дэв Лайман	Боевик	120 мин	18+
<div>Удалить расписание</div>	По сообщениям совести	Мэл Гибсон	Военный	139 мин	18+
	Ужасающий 3	Дэмиен Леоне	Ужасы	125 мин	18+
	Форсаж	Роб Коэн	Боевик	106 мин	18+
<div>Заказы</div>					
<div>Расписания</div>					
<div>Обновить список</div>					
<div>Показать все фильмы</div>					
<div>Очистить список фильмов</div>					
<div>Фильтр по жанру</div>					
<div>Все жанры</div>					
<div>Сортировка</div>					
<div>Без сортировки</div>					
<div>Выйти</div>					

Рисунок 6.9 – Страница управления фильмами

При нажатии на кнопку «Заказы» администратор переходит на страницу заказов и получает доступ к списку всех заказов.

На данной странице реализованы следующие функции:

- фильтрация по статусу – помогает отфильтровать заказы по определенному статусу (забронировано, отменено, просмотрено).

– удалять заказы – удаляет заказы, которые, например отменены самими пользователями или находятся в статусе просмотрено и не нужны для хранения в базе данных.

– управлять заказами – может менять статус заказов пользователей с помощью кнопок управления.

Интерфейс данного раздела выполнен в виде таблицы с краткой информацией по каждому заказу и кнопками управления. Данная страница представлена на рисунке 6.10.

Email	Фильм	Дата	Время	Зал	Места	Цена	Статус	Рейтинг
gerastatko@gmail.com	Мистер и миссис Смит	29.05.2025	22:40	Большой Зал	P4 M11	15.00 BYN	Просмотрено	9
gerastatko@gmail.com	Мистер и миссис Смит	29.05.2025	22:40	Большой Зал	P2 M11	15.00 BYN	Отменен	
denismnk@gmail.com	По соображениям совести	25.05.2025	22:50	Большой Зал	P2 M10, P4 M11, P6 M7	50.00 BYN	Просмотрено	
denismnk@gmail.com	Мистер и миссис Смит	29.05.2025	22:40	Большой Зал	P2 M10	15.00 BYN	Просмотрено	10

Отменить заказ Удалить заказ Пометить как "Просмотрено"

Рисунок 6.10 – Страница управления заказами

При нажатии на кнопку «Расписания» администратор переходит на страницу расписаний и получает доступ к списку всех расписаний.

На данной странице реализованы следующие функции:

– фильтрация по залу – помогает отфильтровать все расписания по определенному залу (большой, средний, малый);

– удалять расписания – удаляет расписания из базы данных.

Интерфейс данного раздела выполнен в виде таблицы с краткой информацией по каждому расписания и кнопкой управления. Данная страница представлена на рисунке 6.11.

Название фильма	Дата	Время	Зал
По соображениям совести	25.05.2025	22:50	Большой Зал
Мистер и миссис Смит	29.05.2025	22:40	Большой Зал

Удалить расписание

Рисунок 6.11 – Страница управления расписаниями

Раздел администратора предоставляет все необходимые инструменты для управления фильмами, расписанием и заказами. Интерфейс интуитивно понятен и позволяет эффективно контролировать работу системы, обеспечивая порядок, безопасность и взаимодействие с пользователями.

Заключение

В рамках выполнения курсового проекта была разработана и реализована система кинотеатра, предназначенная для взаимодействия пользователей с различными ролями: клиент и администратор.

В ходе работы:

- изучены подходы к построению клиентских приложений с использованием WPF и архитектуры MVVM;
- исследована организация баз данных и реализована структура хранения данных в Microsoft SQL Server;
- разработана модель ролевого доступа с учётом функциональных различий между пользователями;
- реализованы пользовательские интерфейсы клиента с возможностями взаимодействия с фильмами;
- создана админ панель с функциями управления фильмами, расписаниями и заказами;

Проект демонстрирует модель кинотеатра и может служить основой для дальнейшего развития. В перспективе возможно расширение приложения за счёт внедрения клиент-серверной архитектуры, интеграции с платёжными системами и реализации дополнительных механизмов.

Список использованных источников

- 1 Кинопоиск [Электронный ресурс] – Режим доступа: <https://www.kinopoisk.ru/> Дата доступа: 17.03.2025.
- 2 Мооон [Электронный ресурс] – Режим доступа: <https://mooon.by/> Дата доступа: 17.03.2025.
- 3 Skyline [Электронный ресурс] – Режим доступа: <https://skyline.by/> Дата доступа: 17.03.2025.
- 4 Руководство по WPF [Электронный ресурс] – Режим доступа: <https://metanit.com/sharp/wpf/> Дата доступа: 23.04.2025.
- 5 Документация Entity Framework [Электронный ресурс] – Режим доступа: <https://learn.microsoft.com/ru-ru/ef/> Дата доступа: 24.04.2025.
- 6 MVVM полное понимание (+WPF) [Электронный ресурс] – Режим доступа: <https://habr.com/ru/articles/338518/> Дата доступа: 01.05.2025.
- 7 Курс лекций по ООП Мущук А.Н.

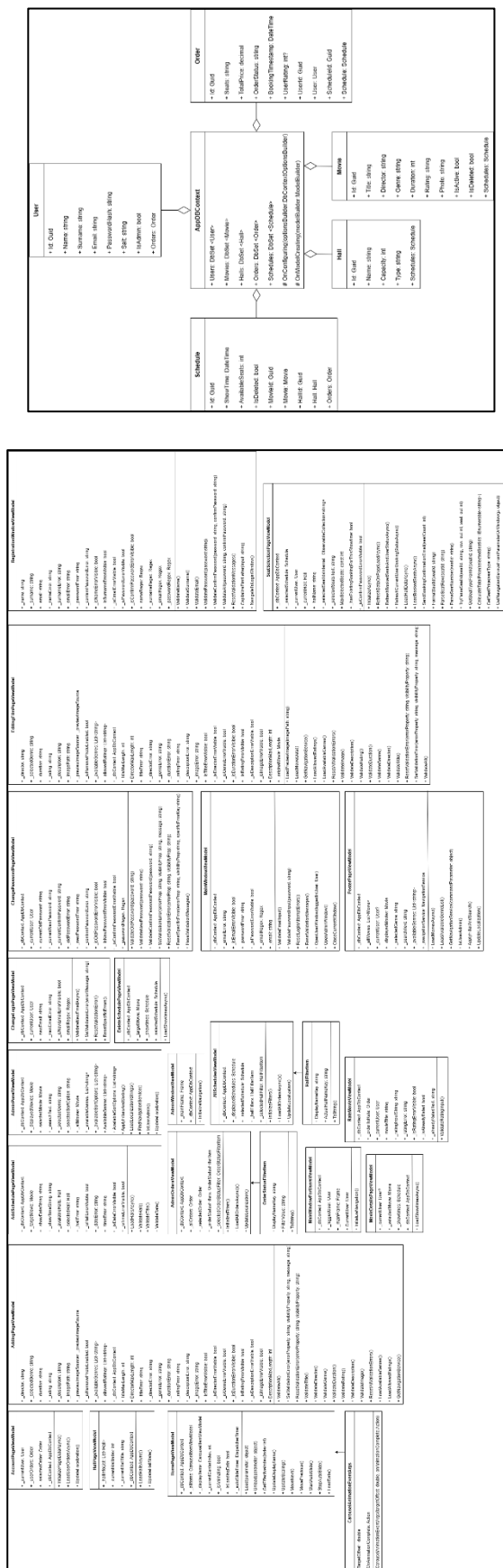
```
graph TD
    subgraph Actors
        Client[Клиент]
        Admin[Администратор]
    end

    subgraph UseCases
        UC1(Зарегистрироваться)
        UC2(Заказать билет на сеанс)
        UC3(Произвести поиск по фильмам или по фильмам режиссера)
        UC4(Выполнить фильтрацию фильмов по жанру)
        UC5(Просмотреть историю заказов)
        UC6(Выполнить ввод)
        UC7(Просматривать фильмы)
        UC8(Просмотреть информацию о кинотеатре)
        UC9(Просмотреть информацию о заказах)
        UC10(Просмотреть доступные сеансы)
        UC11(Выполнить регистрацию фильмов)
        UC12(Отменить заказ)
        UC13(Просмотреть панель администратора)
        UC14(Удалить фильм)
        UC15(Работа со списком фильмов)
        UC16(Добавить расписание)
        UC17(Добавить фильм)
        UC18(Редактировать фильм)
        UC19(Изменить статус заказов на просмотрено)
        UC20(Фильтрация заказов по статусу)
        UC21(Удалить заказы)
        UC22(Отменить заказы)
        UC23(Просмотреть список всех сеансов)
        UC24(Фильтрация сеансов по жанру)
        UC25(Просмотреть список всех заказов)
        UC26(Изменить описание фильма)
        UC27(Изменить возрастную рейтинг фильма)
        UC28(Изменить режиссера фильма)
        UC29(Изменить жанр фильма)
        UC30(Изменить название фильма)
        UC31(Изменить список фильмов)
        UC32(Очистить список фильмов)
        UC33(Произвести фильтрацию списка фильмов по жанру)
        UC34(Сортировку списка фильмов)
        UC35(Обновить список фильмов)
    end

    Client --> UC1
    Client --> UC2
    Client --> UC3
    Client --> UC4
    Client --> UC5
    Client --> UC6
    Client --> UC7
    Client --> UC8
    Client --> UC9
    Client --> UC10
    Client --> UC11
    Client --> UC12

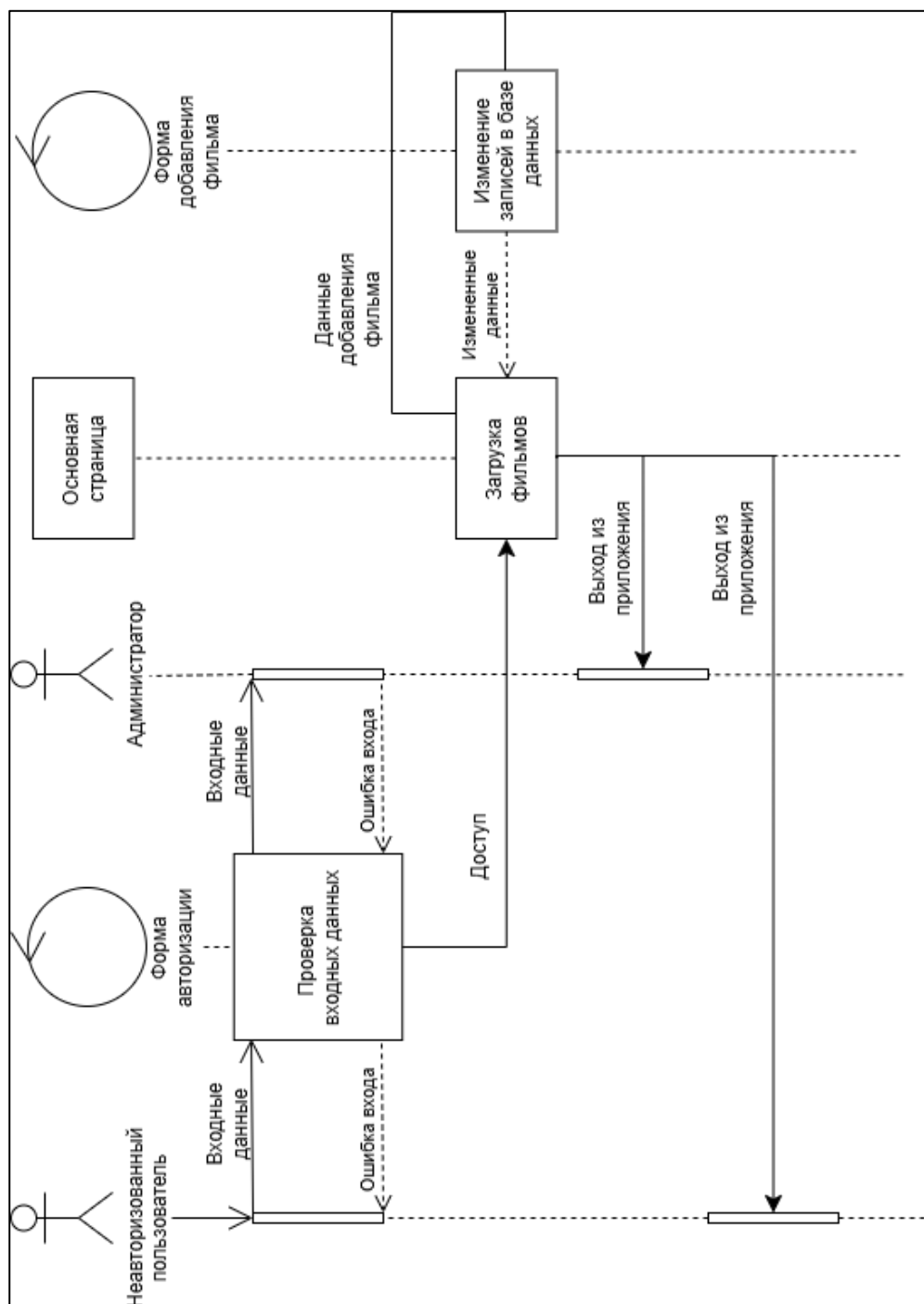
    Admin --> UC13
    Admin --> UC14
    Admin --> UC15
    Admin --> UC16
    Admin --> UC17
    Admin --> UC18
    Admin --> UC19
    Admin --> UC20
    Admin --> UC21
    Admin --> UC22
    Admin --> UC23
    Admin --> UC24
    Admin --> UC25
    Admin --> UC26
    Admin --> UC27
    Admin --> UC28
    Admin --> UC29
    Admin --> UC30
    Admin --> UC31
    Admin --> UC32
    Admin --> UC33
    Admin --> UC34
    Admin --> UC35

    UC1 -.-> UC12
    UC2 -.-> UC12
    UC3 -.-> UC4
    UC4 -.-> UC3
    UC5 -.-> UC13
    UC6 -.-> UC13
    UC7 -.-> UC13
    UC8 -.-> UC13
    UC9 -.-> UC13
    UC10 -.-> UC13
    UC11 -.-> UC13
    UC12 -.-> UC13
    UC13 -.-> UC14
    UC14 -.-> UC15
    UC15 -.-> UC16
    UC16 -.-> UC17
    UC17 -.-> UC18
    UC18 -.-> UC19
    UC19 -.-> UC20
    UC20 -.-> UC21
    UC21 -.-> UC22
    UC22 -.-> UC23
    UC23 -.-> UC24
    UC24 -.-> UC25
    UC25 -.-> UC26
    UC26 -.-> UC27
    UC27 -.-> UC28
    UC28 -.-> UC29
    UC29 -.-> UC30
    UC30 -.-> UC31
    UC31 -.-> UC32
    UC32 -.-> UC33
    UC33 -.-> UC34
    UC34 -.-> UC35
```



Приложение В

Диаграмма последовательности



Приложение Г

Листинг – Функция регистрация пользователя

```
private async Task ExecuteRegister(object parameter)
{
    if (!(parameter is object[] controls) || controls.Length <
2 ||
        !(controls[0] is PasswordBox passBox) ||
!(controls[1] is PasswordBox confirmPassBox))
    {
        MessageBox.Show(

            GetString("ErrorRegistrationGeneral").Replace("{0}
", "Internal error: Password controls missing."),
            "Error", MessageBoxButton.OK,
MessageBoxImage.Error);
        return;
    }

    string password = passBox.Password;
    string confirmPassword = confirmPassBox.Password;

    bool isValidForm = ValidateAll(password, confirmPassword);

    if (!isValidForm)
    {
        return;
    }

    try
    {
        bool isEmailTaken = await _dbContext.Users.AnyAsync(u
=> u.Email == Email);
        if (isEmailTaken)
        {
            EmailError =
GetString("ErrorEmailExists");
            IsEmailErrorVisible = true;
            return;
        }

        var (hash, salt) =
PasswordHasher.HashPassword(password);

        var newUser = new User
        {
            Id = Guid.NewGuid(),
            Name = Name,
            Surname = Surname,
            Email = Email.Trim(),
            PasswordHash = hash,
```

```

        Salt = salt,
        IsAdmin = false
    };

    await _dbContext.Users.AddAsync(newUser);
    await _dbContext.SaveChangesAsync();

    MessageBox.Show(
        GetString("SuccessRegistrationMessage"),
        GetString("SuccessRegistrationTitle"),
        MessageBoxButton.OK,
        MessageBoxImage.Information);

    NavigateToLoginWindow();
}
catch (DbUpdateException dbEx)
{
    MessageBox.Show(
        string.Format(GetString("ErrorRegistrationGeneral"),
            $"Database error: {dbEx.InnerException?.Message} ?? dbEx.Message"),
        "Error", MessageBoxButton.OK,
        MessageBoxImage.Error);
}
catch (Exception ex)
{
    MessageBox.Show(
        string.Format(GetString("ErrorRegistrationGeneral"), ex.Message),
        "Error", MessageBoxButton.OK,
        MessageBoxImage.Error);
}
}

```

Приложение Д

Листинг – Функция авторизации пользователя

```
private async Task ExecuteLogin(object parameter)
{
    var passwordBox = parameter as PasswordBox;
    if (passwordBox == null) return;
    string password = passwordBox.Password;
    ResetLoginAttemptErrors();
    bool isEmailValid = ValidateEmailInput();
    bool isPasswordValid = ValidatePasswordInput(password);
    if (!isEmailValid || !isPasswordValid) return;
    try
    {
        string emailToCompare = Email.ToLower();
        var user = await
        _dbContext.Users.FirstOrDefaultAsync(u => u.Email.ToLower() ==
        emailToCompare);
        if (user == null)
        {
            EmailError =
            GetString("ErrorEmailNotFound");
            IsEmailErrorVisible = true;
            return;
        }
        if (!PasswordHasher.VerifyPassword(password,
        user.PasswordHash, user.Salt))
        {
            PasswordError =
            GetString("ErrorPasswordIncorrect");
            IsPasswordErrorVisible = true;
            return;
        }
        if (user.IsAdmin)
        {
            OpenAdminWindow();
        }
        else
        {
            OpenUserWindow(user);
        }
        CloseCurrentWindow();
    }
    catch (Exception ex)
    {
        PasswordError =
        string.Format(GetString("ErrorLoginGeneral"),
        ex.Message);
        IsPasswordErrorVisible = true;
    }
}
```

Приложение Е

Листинг – Функция добавления фильма

```
private async void ExecuteAddFilm(object parameter)
{
    if (!await ValidateAll())
    {
        ShowMessage("AddFilmPage_Error_ValidationFails",
            "AddFilmPage_Title_InputError", MessageBoxImage.Warning);
        return;
    }

    int.TryParse(Duration, out int durationMinutes);
    try
    {
        var newMovie = new Movie
        {
            Title = Title.Trim(),
            Director = Director.Trim(),
            Genre = SelectedGenre,
            Duration = durationMinutes,
            Rating = Rating,
            Description = Description.Trim(),
            Photo = ImagePath
        };

        var existingMovie = await
            _dbContext.Movies.FirstOrDefaultAsync(m => !m.IsDeleted &&
                m.Title == newMovie.Title);
        if (existingMovie != null)
        {
            SetValidationError(nameof(TitleError),
                nameof(IsTitleErrorVisible),
                GetString("AddFilmPage_Error_TitleDuplicate"));
            return;
        }
        _dbContext.Movies.Add(newMovie);
        await _dbContext.SaveChangesAsync();

        ShowMessageFormat("AddFilmPage_Success_MovieAdded",
            "AddFilmPage_Title_Success", MessageBoxImage.Information,
            newMovie.Title);
        ExecuteClearForm(null);
    }
    catch (Exception ex)
    {
        ShowMessageFormat("AddFilmPage_Error_AddingFailed",
            "AddFilmPage_Title_Error", MessageBoxImage.Error, ex.Message);
    }
}
```

Приложение Ж

Листинг – Функция бронирования мест

```
private async Task ExecuteConfirmBookingAsync()
{
    if (_currentUser == null)
    {
        ShowMessage("Booking_Error_UserNotIdentified",
"ErrorTitle", MessageBoxImage.Warning);
        return;
    }

    await RefreshCurrentUserBookingStatusAsync();

    if (!CanConfirmBooking)
    {
        if (_hasExistingBookingForThisShowtime)
        {
            ShowMessage("Booking_Error_AlreadyBookedThisShowtime",
"ErrorTitle", MessageBoxImage.Warning);
        }
        else if (_selectedSeatIdsInternal == null ||
!_selectedSeatIdsInternal.Any())
        {
            ShowMessage("SeatSelectionPage_NoSeatsSelected",
"ErrorTitle", MessageBoxImage.Warning);
        }
        return;
    }

    var currentSelection = new
List<string>(_selectedSeatIdsInternal);
    if (!currentSelection.Any())
    {
        ShowMessage("SeatSelectionPage_NoSeatsSelected",
"ErrorTitle", MessageBoxImage.Warning);
        return;
    }

    try
    {
        var newOrder = new Order
        {
            Id = Guid.NewGuid(),
            Seats = string.Join(", ",
currentSelection.OrderBy(s => s)),
            TotalPrice =
CalculateTotalPrice(currentSelection),
            OrderStatus = "OrderStatus_Booked",
            BookingTimestamp = DateTime.Now,
```

```

        UserId = _currentUser.Id,
        ScheduleId = _selectedSchedule.Id
    };

    _dbContext.Orders.Add(newOrder);
    await _dbContext.SaveChangesAsync();

    try
    {
        await Task.Run(() =>
SendBookingConfirmationEmail(currentSelection.Count));
    }
    catch (Exception emailEx)
    {
    }

    ShowMessage("Booking_Success", "Success_Title",
MessageBoxImage.Information);

    _hasExistingBookingForThisShowtime = true;

    bool newItemAddedToGlobalBooked = false;

    foreach (var bookedId in currentSelection)
    {
        if (BookedSeatIds.Add(bookedId))
newItemsAddedToGlobalBooked = true;
    }

    if (newItemsAddedToGlobalBooked)
    {
        BookedSeatIds = new
HashSet<string>(BookedSeatIds);
        OnPropertyChanged(nameof(BookedSeatIds));
    }
    _selectedSeatIdsInternal.Clear();
    UpdateSelectedSeatsText();
    CommandManager.InvalidateRequerySuggested();
}
catch (DbUpdateException dbEx)
{
    ShowMessageFormat("Booking_Error_DbSave",
"ErrorTitle", MessageBoxImage.Error,
dbEx.InnerException?.Message ?? dbEx.Message);
    await RefreshBookedSeatsAndUserStatusAsync();
}
catch (Exception ex)
{
    ShowMessageFormat("Booking_Error_General",
"ErrorTitle", MessageBoxImage.Error, ex.Message);
}
}

```

Приложение 3

Листинг – Реализация класса ApplicationDbContext

```
using Microsoft.EntityFrameworkCore;
using CinemaMOON.Configurations;
using System;
using CinemaMOON.Models;

namespace CinemaMOON.Data
{
    public class AppDbContext : DbContext
    {
        public DbSet<User> Users { get; set; }
        public DbSet<Movie> Movies { get; set; }
        public DbSet<Hall> Halls { get; set; }
        public DbSet<Schedule> Schedules { get; set; }
        public DbSet<Order> Orders { get; set; }

        public AppDbContext(DbContextOptions<AppDbContext>
options) : base(options)
        {
        }

        protected override void OnModelCreating(ModelBuilder
modelBuilder)
        {
            modelBuilder.ApplyConfiguration(new
UserConfiguration());
            modelBuilder.ApplyConfiguration(new
MovieConfiguration());
            modelBuilder.ApplyConfiguration(new
HallConfiguration());
            modelBuilder.ApplyConfiguration(new
ScheduleConfiguration());
            modelBuilder.ApplyConfiguration(new
OrderConfiguration());

            modelBuilder.Entity<Hall>().HasData(
                new Hall
                {
                    Id = Guid.Parse("DA291A5E-71C2-46A6-965C-
02D3E2C59431"),
                    Name = "HallPage_SmallHallName",
                    Capacity = 45,
                    Type = "small"
                },
                new Hall
                {
                    Id = Guid.Parse("B68871D6-35E5-432C-B0E0-
3178586333E9"),
                    Name = "HallPage_MediumHallName",
```

```

        Capacity = 60,
        Type = "medium"
    },
    new Hall
    {
        Id = Guid.Parse("14EC82A0-460F-4BE2-9BC9-
1C56E949A17E"),
        Name = "HallPage_LargeHallName",
        Capacity = 70,
        Type = "large"
    }
    );
}

protected override void
OnConfiguring(DbContextOptionsBuilder optionsBuilder)
{
    if (!optionsBuilder.IsConfigured)
    {
optionsBuilder.UseSqlServer("Server=(localdb)\\MSSQLLocalDB;Dat
abase=CinemaMOON;Trusted_Connection=True;TrustServerCertificate
=True;");
    }
}
}
}

```