|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  Калужский филиал  федерального государственного бюджетного  образовательного учреждения высшего образования  ***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»***  ***(КФ МГТУ им. Н.Э. Баумана)*** |

**ФАКУЛЬТЕТ** ***ИУК «Информатика и управление»***

**КАФЕДРА** ***ИУК5 «Системы обработки информации»***

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к курсовой работе на тему:**

***Веб-приложение для автоматизации продажи билетов в кинотеатре***

по дисциплине ***Базы данных***

Студент: гр. ИУК5-51Б \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ( Поляков Р. А. )

(подпись) (Ф.И.О.)

Руководитель: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ( Кириллов В. Ю. )

(подпись) (Ф.И.О.)

Оценка руководителя \_\_\_\_\_ баллов \_\_\_\_\_\_\_\_\_\_\_

30-50 (дата)

Оценка защиты \_\_\_\_\_ баллов \_\_\_\_\_\_\_\_\_\_\_

30-50 (дата)

Оценка проекта \_\_\_\_\_ баллов \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оценка по пятибалльной шкале)

Комиссия: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

Калуга, 2020

Калужский филиал   
федерального государственного бюджетного образовательного учреждения высшего образования

***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»   
(КФ МГТУ им. Н.Э. Баумана)***

УТВЕРЖДАЮ

Заведующий кафедрой **\_\_ИУК5-КФ\_\_\_**

\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2020 г.

**ЗАДАНИЕ**

**на выполнение курсовой работы**

по дисциплине ***Базы данных***

Студент *Поляков Р.А., ИУК5-51Б*

(фамилия, инициалы, индекс группы)

Руководитель *Кириллов В.Ю.*

(фамилия, инициалы)

График выполнения проекта: 25% к 4 нед., 50% к 7 нед., 75% к 10 нед., 100% к 14 нед.

***1. Тема курсовой работы***

***Веб-приложение для автоматизации продажи билетов в кинотеатре.***

***2. Техническое задание***

*Разработать веб-приложение для автоматизации продажи билетов в кинотеатре и интегрировать его со спроектированной базой данных.*

***3. Оформление курсовой работы***

3.1. Расчетно-пояснительная записка на\_\_\_\_\_\_\_\_ листах формата А4.

3.2. Перечень графического материала КП (плакаты, схемы, чертежи и т.п.)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата выдачи задания «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2020 г.

Руководитель курсовой работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

(подпись) (Ф.И.О.)

Задание получил\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_2020 г.

(подпись) (Ф.И.О.)

Примечание:

Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

**ОГЛАВЛЕНИЕ**

[1. Техническое задание 5](#_Toc56010604)

[1.1 Наименование системы 5](#_Toc56010605)

[1.2 Основания для разработки 5](#_Toc56010606)

[1.3 Исполнитель 5](#_Toc56010607)

[1.4 Назначение и цель разработки системы 5](#_Toc56010608)

[1.5 Содержание работы 5](#_Toc56010609)

[1.5.1. Задачи, подлежащие решению 5](#_Toc56010610)

[1.5.2. Требования к архитектуре АСОИ 6](#_Toc56010611)

[1.5.3. Требования к составу программных компонентов 6](#_Toc56010612)

[1.5.4. Требования к прикладным программам 6](#_Toc56010613)

[1.5.5. Требования к входным/выходным данным 6](#_Toc56010614)

[1.5.6. Требования к временным характеристикам 6](#_Toc56010615)

[1.5.7. Требования к составу технических средств 7](#_Toc56010616)

[1.6 Этапы разработки 7](#_Toc56010617)

[1.7 Техническая документация, предъявляемая по окончании работы 7](#_Toc56010618)

[1.8 Дополнительные условия 8](#_Toc56010619)

[2. исследовательская часть 9](#_Toc56010620)

[2.1. Постановка задачи 9](#_Toc56010621)

[2.2. Перечень задач, подлежащих решению в процессе разработки 9](#_Toc56010622)

[2.3. Анализ существующих аналогов 10](#_Toc56010623)

[2.4. Предлагаемые функции приложения для реализации 16](#_Toc56010624)

[2.5. Обоснование выбора среды разработки 17](#_Toc56010625)

[2.6. Обоснование выбора языка разработки 17](#_Toc56010626)

[2.7. Обоснование выбора СУБД 18](#_Toc56010627)

[2.8. Выводы 18](#_Toc56010628)

[3. Проектно - конструкторская часть 19](#_Toc56010629)

[3.1 Общие сведения 19](#_Toc56010633)

[3.2 Структура веб-приложения 19](#_Toc56010634)

[3.3 Концептуальная модель базы данных 19](#_Toc56010635)

[3.4 Логическая модель базы данных 21](#_Toc56010636)

[3.5 Физическое проектирование базы данных 23](#_Toc56010637)

[3.6 Архитектура приложения 28](#_Toc56010638)

[3.7 Архитектура приложения 28](#_Toc56010639)

[3.8 Архитектура приложения 28](#_Toc56010640)

[3.9 Архитектура приложения 28](#_Toc56010641)

[3.10 Архитектура приложения 28](#_Toc56010642)

[3.11 Архитектура приложения 28](#_Toc56010643)

[4. Проектно – технологическая часть 29](#_Toc56010644)

[4.1 Требования к аппаратной платформе 29](#_Toc56010646)

[4.2 Тестирование и отладка макета рабочей программы 29](#_Toc56010647)

[4.3 Руководство администратора 29](#_Toc56010648)

[4.4 Руководство пользователя 29](#_Toc56010649)

[4.5 Экспериментальные данные тестирования процедур и функциональных задач 29](#_Toc56010650)

[4.6 Экспериментальные данные тестирования процедур и функциональных задач 29](#_Toc56010651)

[4.7 Экспериментальные данные тестирования процедур и функциональных задач 29](#_Toc56010652)

[4.8 Экспериментальные данные тестирования процедур и функциональных задач 29](#_Toc56010653)

[4.9 Экспериментальные данные тестирования процедур и функциональных задач 29](#_Toc56010654)

[4.10 Экспериментальные данные тестирования процедур и функциональных задач 29](#_Toc56010655)

[4.11 Экспериментальные данные тестирования процедур и функциональных задач 29](#_Toc56010656)

[4.12 Экспериментальные данные тестирования процедур и функциональных задач 29](#_Toc56010657)

[Заключение 30](#_Toc56010658)

[Список использованных источников 31](#_Toc56010659)

[Приложение 32](#_Toc56010660)

# Техническое задание

## Наименование системы

Настоящее Техническое задание определяет требования и порядок создания веб-приложения для автоматизации продажи билетов в кино.

## Основания для разработки

Основанием для разработки веб-приложения для автоматизации продажи билетов в кино является задание в соответствии с учебным планом дисциплины «Базы данных»

## Исполнитель

Исполнителем проекта является студент Калужского филиала МГТУ им. Н. Э. Баумана, факультета ИУК «Информатики и управления», кафедры ИУК5 «Системы обработки информации», группы ИУК5-51Б, Поляков Роман Андреевич.

## Назначение и цель разработки системы

Разрабатываемое приложение предназначено для демонстрации современных подходов в веб-разработке.

Целью создания системы является закрепление полученных навыков работы с СУБД PostgreSQL и изучение принципов её взаимодействия с технологиями веб-программирования.

## **Содержание работы**

### Задачи, подлежащие решению

В ходе реализации программного продукта необходимо предусмотреть интерфейс веб-приложения, разработать модель данных и установить взаимодействие между frontend и backend частями.

### Требования к архитектуре АСОИ

Архитектура программного продукта представлена на рисунке 1.



Рисунок 1  Архитектура АСОИ

В качестве основного стека технологий использовать «PERN» (PostgreSQL, Express, React and Node.js).

### Требования к составу программных компонентов

Программный продукт должен представлять собой протестированное веб-приложение, размещенное на удаленном сервере.

### **Требования к прикладным программам**

Максимальная функциональность в конкретной предметной области.

### Требования к входным/выходным данным

Входные данные должны быть логичными и отображать то, что пользователь хочет получить в результате работы программы.

Выходные данные должны максимально правильно и полностью соответствовать действиям конечных пользователей.

### Требования к временным характеристикам

Требования к временным характеристикам программы не предъявляются.

### Требования к составу технических средств

Для построения программного продукта необходимы следующие аппаратно-технические и программные средства:

* OC: Windows 10;
* Visual Studio Code;
* веб-браузер на основе Сhromium;
* база данных PostgreSQL;
* СУБД pgAdmin.

## Этапы разработки

На стадии разработки технического задания должен быть выполнен этап разработки, согласования и утверждения настоящего технического задания.

На исследовательской стадии должен быть выполнен обзор существующих аналогов программному продукту; выбор программного обеспечения, библиотек для создания продукта.

На стадии проектирования компонентов программного продукта выполняется проектирование архитектуры системы; состава и методов взаимодействия компонентов; алгоритмов обработки и представления данных.

На стадии реализации производится разработка и тестирование спроектированной программы, ввод ее в эксплуатацию, а также оформление технической документации.

## Техническая документация, предъявляемая по окончании работы

Должны быть разработаны следующие программные документы:

1. Расчетно-пояснительная записка:

* техническое задание;
* научно-исследовательская часть;
* проектирований компонентов программного продукта;
* проектно-технологическая часть.

2. Графическая часть.

## Дополнительные условия

Клиентская часть представляет собой сайт, на котором размещена афиша фильмов, идущих в кино. Список фильмов и сеансов возвращать из базы данных по прописанным SQL-запросам. Для фильма в сетке отображается миниатюра фото, заголовок, возрастной рейтинг. Реализовать возможность просмотреть описания конкретного фильма подробно. Реализовать функцию покупки билета. На билете отображать только необходимую информацию о сеансе. Реализовать возможность авторизации для пользователя и просмотра купленных им билетов.

Клиентская часть приложения должна быть реализована с использованием следующих библиотек: React, ReactFinalForm, React-Router, Lodash, Material-UI. Для сборки проектов использовать Webpack, Babel или CRA, NextJS. Для линтинга проекта и соблюдения код-стайла использовать: ESLint, Eslint-config-airbnb.

Для приложения использовать монорепозиторий на GitHub. Для хостинга приложения использовать Heroku.

# исследовательская часть

## Постановка задачи

Киноиндустрия в целом является одним из главных развлечений для человека. Миллионы людей каждый день смотрят фильмы как в обычных кинотеатрах, так и онлайн-кинотеатрах, горячо обсуждают текущие кинопроекты в соцсетях, с нетерпением ждут выхода новых.

Обычный человек, когда идет в кинотеатр в первую очередь хочет получить удовольствие. Чтобы обеспечить комфорт зрителям владельцы кинотеатров используют системы кондиционирования, комфортные кресла, большие и качественные экраны, мощные аудиосистемы, но зачастую, по разным причинам не обращают внимание на такой важный процесс как покупка билета на киносеанс. До сих пор во многих кинотеатрах мы можем наблюдать очереди у касс перед началом сеанса, неудобные киноафиши и невозможность купить билет заранее через интернет. С решением этих проблем может справится веб приложение, автоматизирующее примитивные процессы. Оно позволит зрителю посмотреть афишу выбрать необходимый фильм, место в зале и купить билет, а главное сделать это за несколько минут с помощью интуитивно понятного интерфейса.

Главной задачей курсовой работы является проектирование и создание веб приложения с использованием базы данных для реализации возможностей просмотра киноафиши и покупки билета в кинотеатр.

## Перечень задач, подлежащих решению в процессе разработки

Для выполнения поставленной задачи необходимо определить оптимальную архитектуру для построения веб приложения и средства реализации программного продукта, соответствующего данной архитектуре. Также необходимо исследовать существующие аналоги приложений для кинотеатров, разработать концептуальные и физические модели проектирования базы данных для приложения, разработать базу данных для хранения необходимой информации, спроектировать и разработать визуальную часть приложения и провести её тестирование.

## Анализ существующих аналогов

Одним из вариантов веб-приложения для автоматизации работы кинотеатра является продукт компании «Платформа 24». Кинотеатр адаптирует готовый шаблон под свой бренд как конструктор. Это прекрасное решение для небольших кинотеатров, в котором реализованы все необходимые функции: показ афиши, покупка и просмотр уже купленных билетов, просмотр информации о кинотеатре. Из минусов я бы отметил отсутствие идентичности у сайта и не эргономичное размещение сетки фильмов.

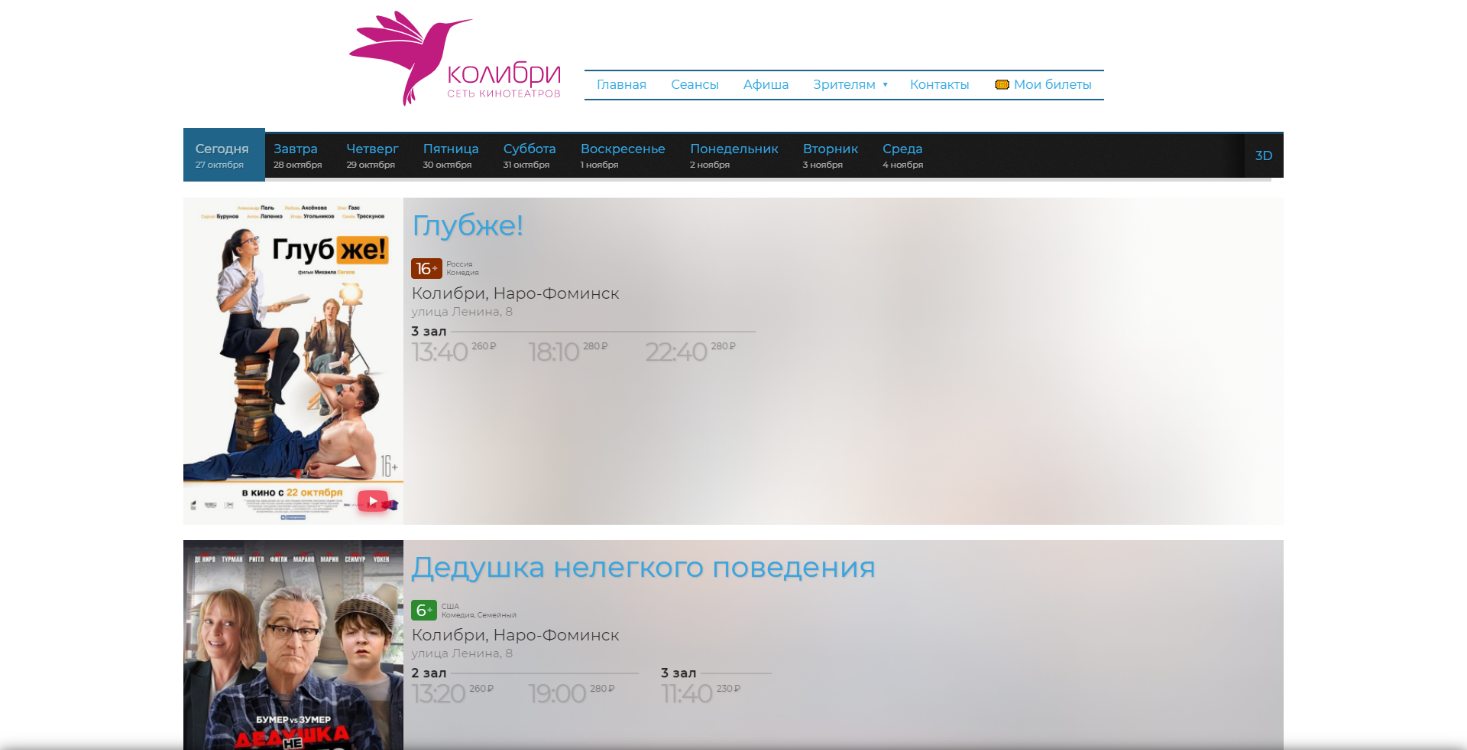


Рисунок 1 — Сеансы кинотеатра

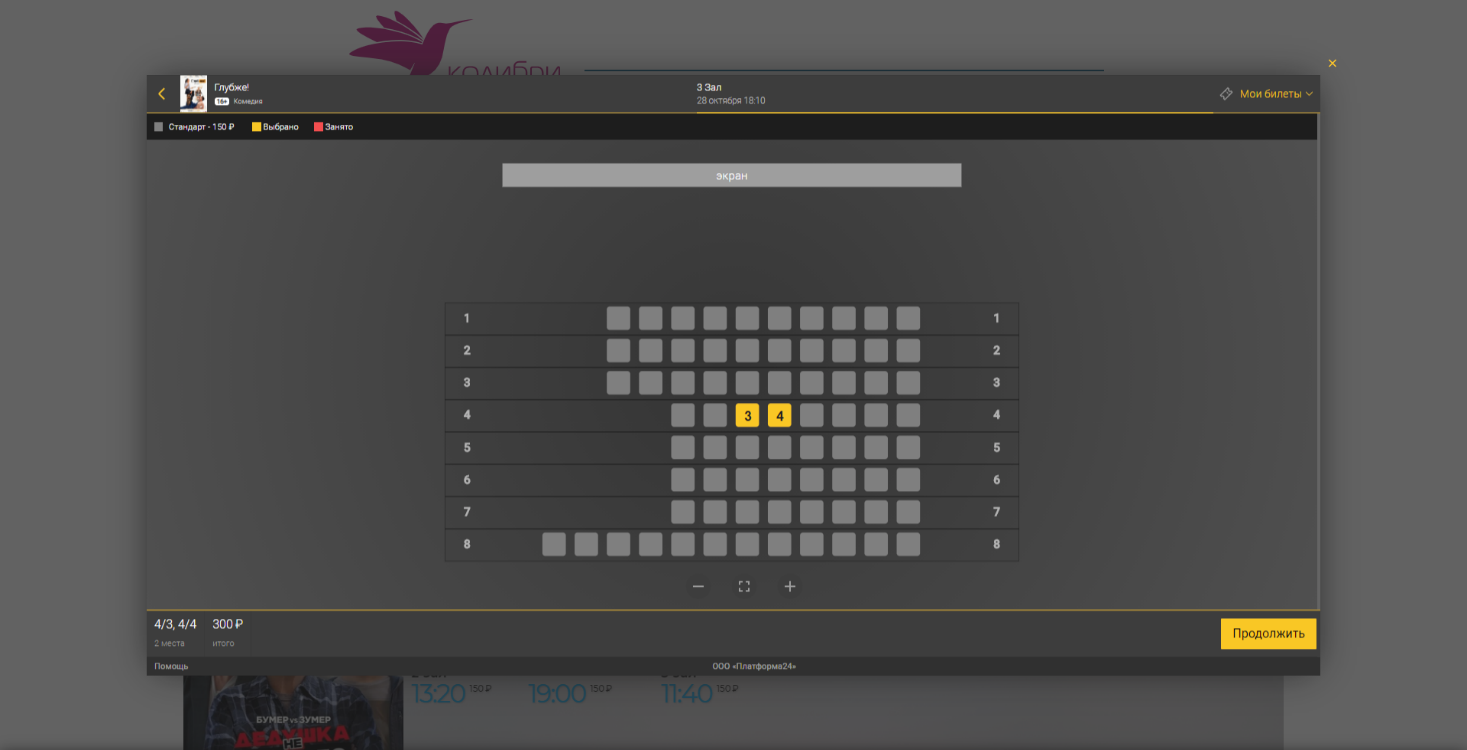


Рисунок 2 — Выбор места и ряда

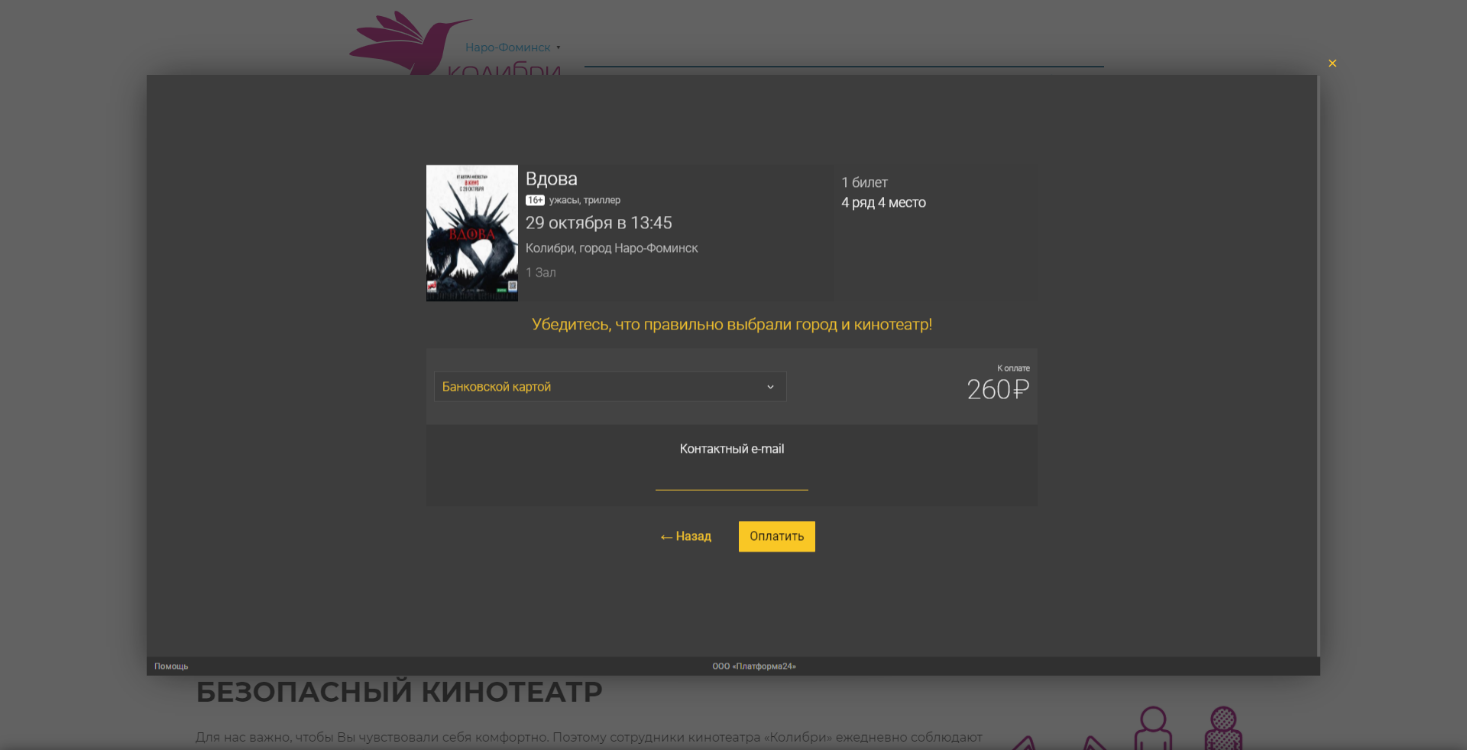


Рисунок 3 — Оплата билета

По-иному ситуация обстоит с крупными сетями кинотеатров, они могут себе позволить собственный сайт со всеми вытекающими преимуществами. Как правило, прослеживается фирменный стиль оформления, присутствует собственная реализация виджетов для оформления билетов либо интеграция сторонних виджетов, например, от компании Rambler.

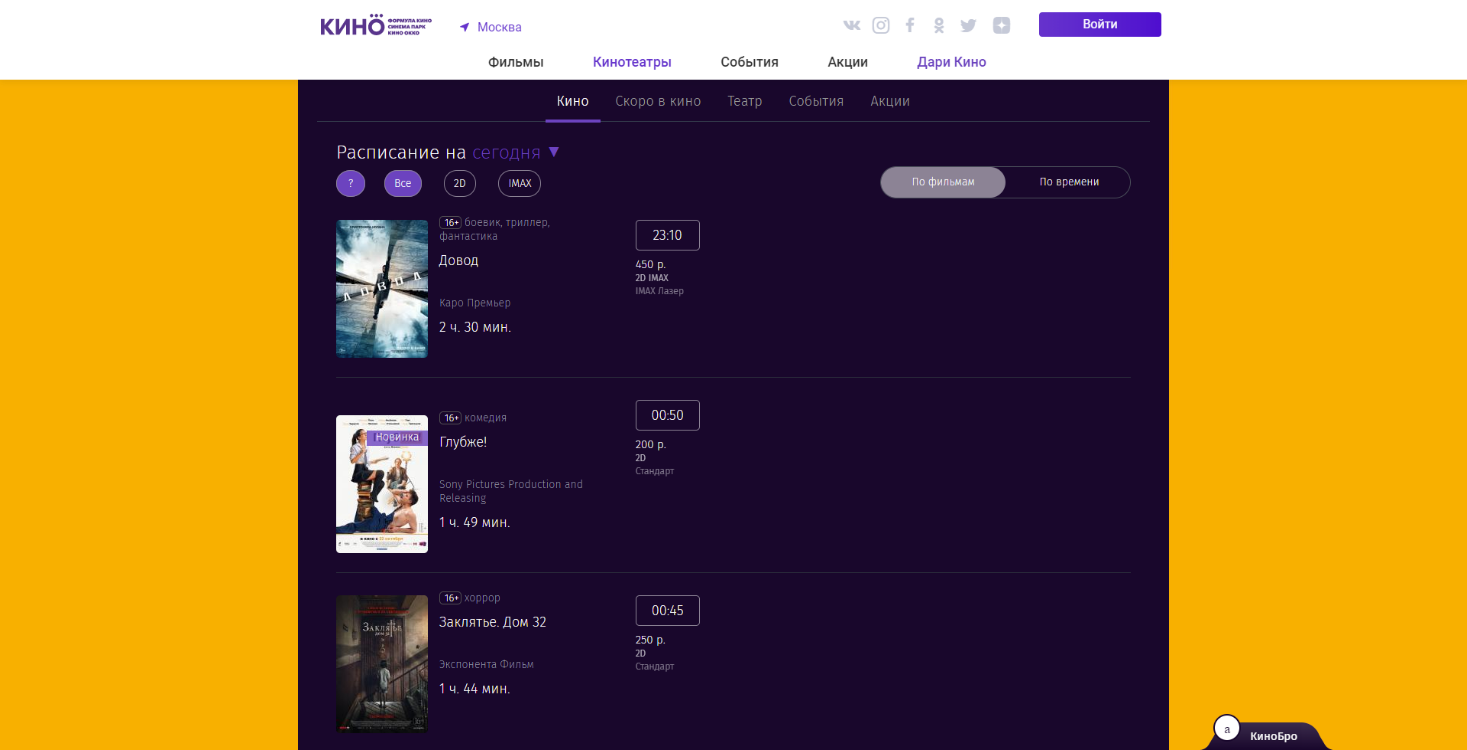


Рисунок 4 — Сеансы кинотеатра «Формула Кино»



Рисунок 5 — Виджет оформления билета

Стоит также рассмотреть одну из крупнейших сетей кинотеатров в США — «CineMark»

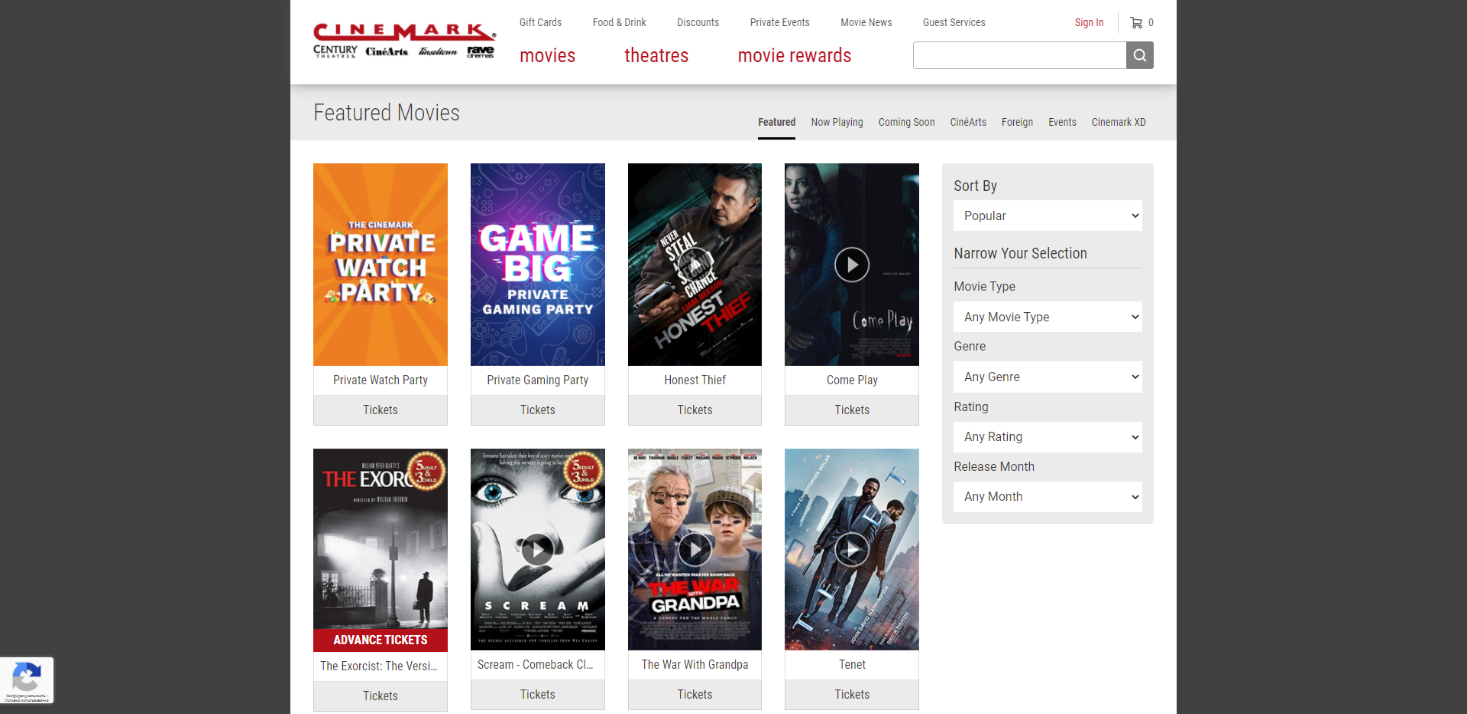


Рисунок 6 — Сетка фильмов в прокате

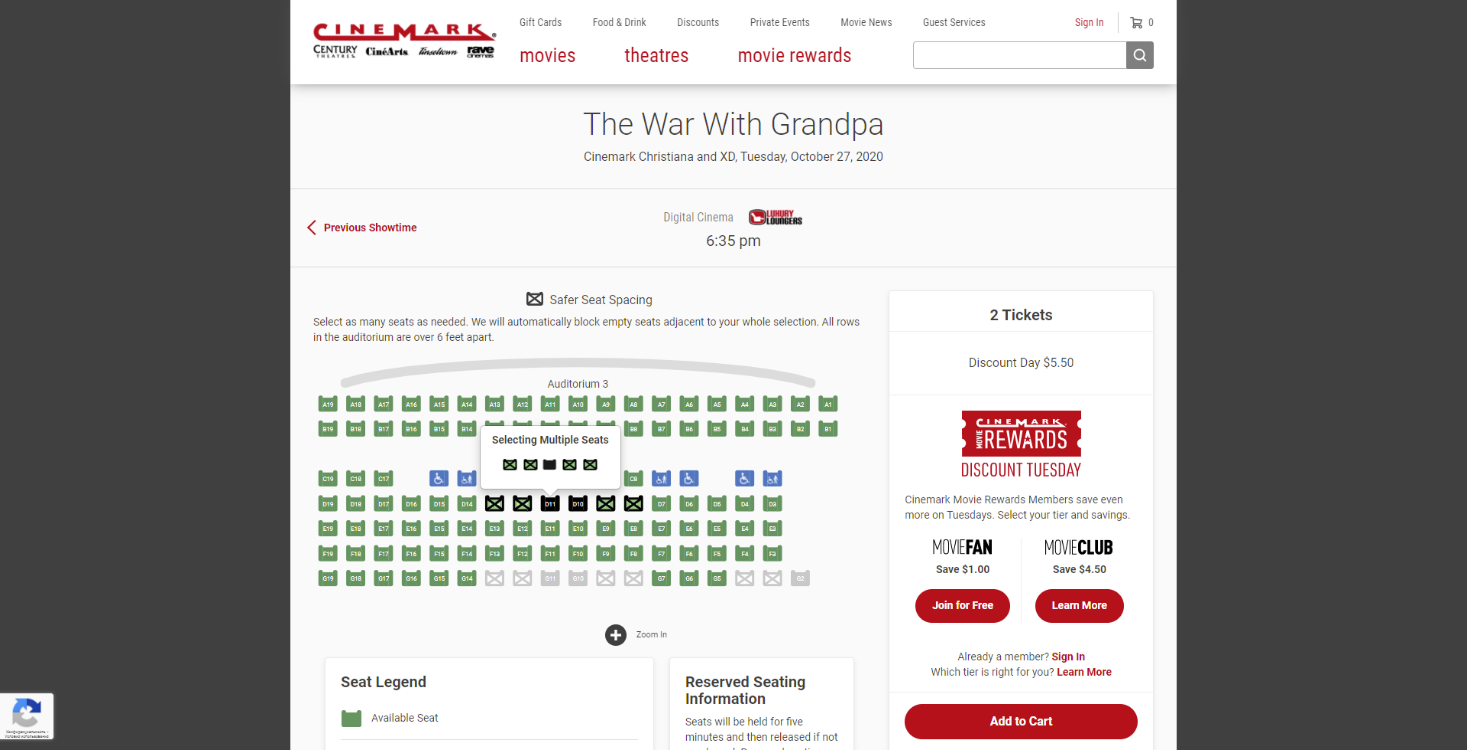


Рисунок 7 — Выбор места

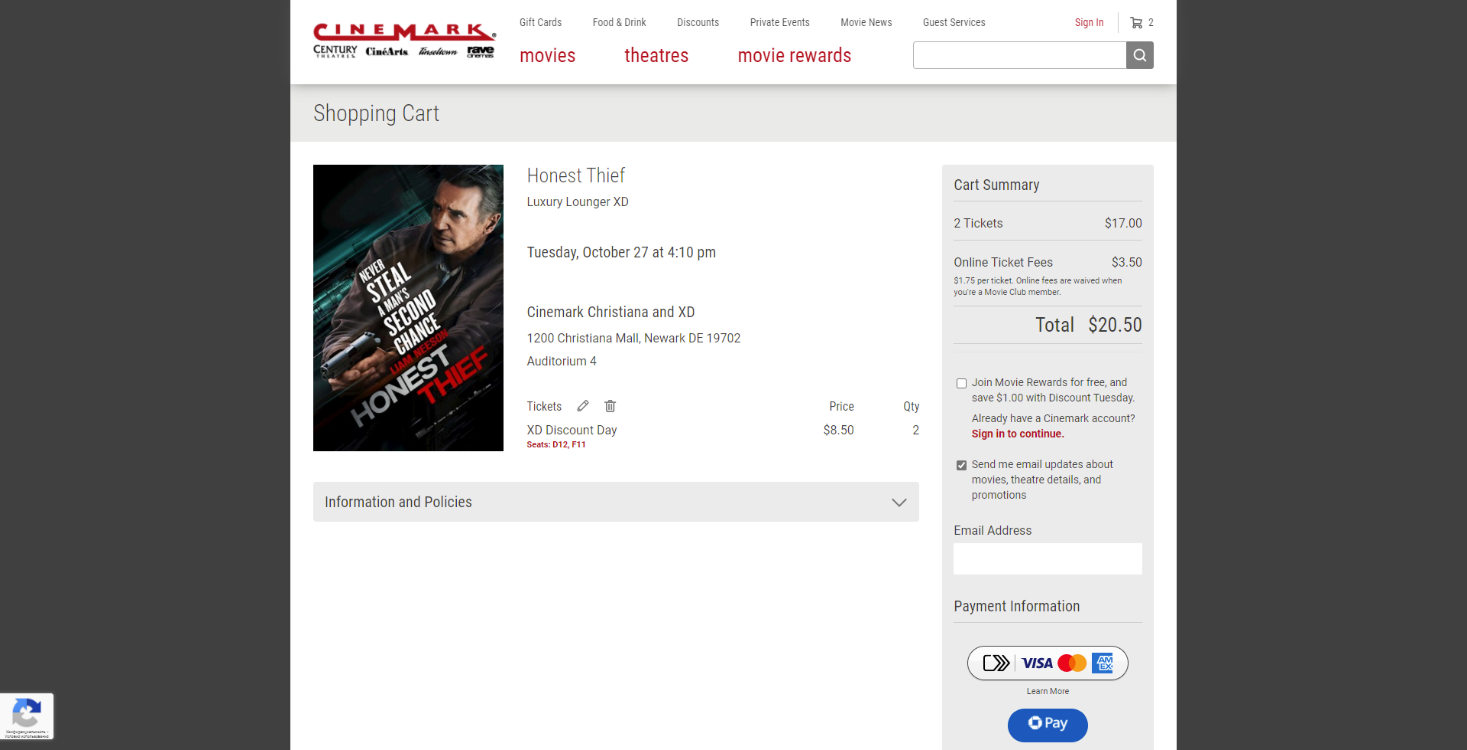


Рисунок 8 — Завершение оформления билета

Как видно, подход к реализации другой: оформление реализовано поэтапно без лишних всплывающих окон. Это несомненно является достоинством данного веб-приложения. Однако устаревший перегруженный дизайн оставляет желать лучшего.

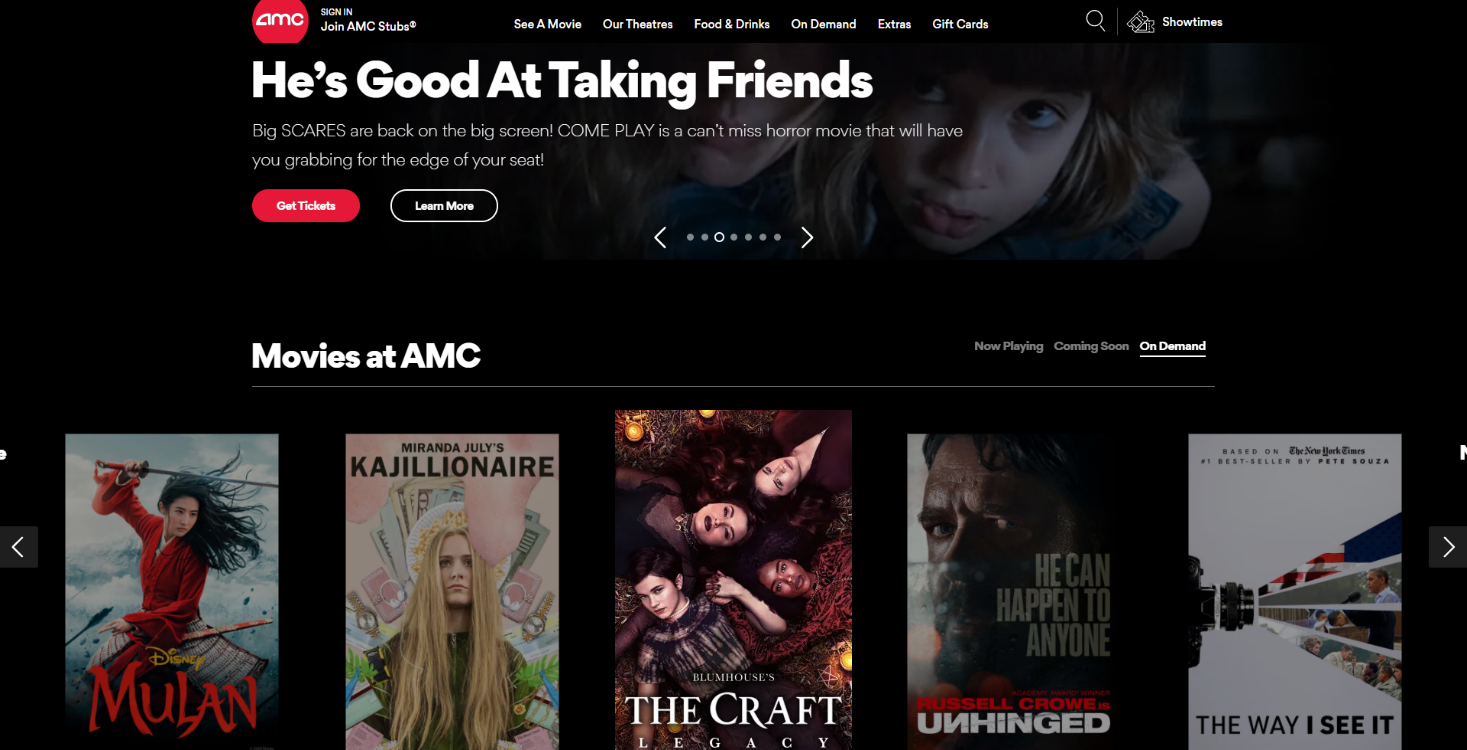
При этом сайт другой сети «AMC» выглядит современно и в общем очень удобен.  


Рисунок 9 — Главная страница веб приложения

Процесс оформления билета также пошаговый.



Рисунок 10 — Выбор места и ряда

## Предлагаемые функции приложения для реализации

Прежде чем подходить к этапу реализации программных функций, необходимо понять, нужны ли они конечному потребителю, если речь идет о клиентских фишках, или работоспособен ли программный продукт без них, если мы говорим о серверных и других «скрытых» методах.

На основании разобранных выше аналогов разрабатываемой автоматизированной системы было выделено два основных подхода реализации функций веб-приложения: с помощью встраиваемых виджетов и использование перерисовки компонентов страницы. Использование второго подхода более целесообразно по следующим причинам: библиотека React позволяет не обновлять страницу веб-приложения целиком, а только необходимые элементы и с точки зрения предполагаемых действий конечного пользователя наличие бесшовного пошагового процесса работы с сайтом считаю приоритетным над реализацией всплывающих виджетов.

Для реализации MVP:

Со стороны клиента предлагаются следующие функции:

* регистрация с помощью почты и пароля;
* авторизация с помощью почты и пароля;
* просмотр купленных билетов;
* просмотр карусели фильмов;
* открытие карточки фильма с возможностью покупки билетов.

Со стороны серверной части предлагаются следующие функции:

* реализация регистрации и авторизации пользователя (создание соответствующих записей в БД и проверка наличия пользователя в БД);
* функция занесения информации о купленных билетах в БД;
* получение информации о фильмах и привязанных к ним сеансам из БД.

Основная задача — разработка веб приложения. Среди современных инструментов для веб-разработки можно рассмотреть следующие IDE:

* + Brackets;
  + Atom;
  + Microsoft Visual Studio Code;
  + Eclipse;
  + VIM.

Идеальным вариантом для решения поставленных мною задач является легкий редактор, обладающий функциями IDE — Microsoft Visual Studio Code по следующим причинам:

* + удобная и простая кастомизация интерфейса;
  + расширяемая библиотека плагинов;
  + хорошие инструменты отладки и рефакторинга;
  + поддержка множества языков;
  + встроенные инструменты интеграции с GitHub, GIT.

## Обоснование выбора языка разработки

В качестве основного языка разработки был выбран JavaScript. Это гибкий язык с огромным количеством различных библиотек и фреймворков. Он работает со всеми распространёнными веб-браузерами, позволяет писать код серверной части приложения, используя Node.js, и разрабатывать одностраничные и многостраничные приложения с помощью Express.js.

## Обоснование выбора СУБД

Так как разработка веб приложения будет производится с использованием стека технологий PERN, системой управления базами данных была выбрана PostgreSQL — самая продвинутая свободная объектно-реляционная система управления базами данных, обладающая большим рядом преимуществ.

## Выводы

Таким образом, исходя из требований к реализуемой системе, рассмотрения возможностей наиболее подходящих инструментов, вариантов разработки и последующего их сравнения было решено использовать следующие решения:

Основной язык программирования — JavaScript. Для разработки веб-приложения была выбрана интегрированная среда разработки Microsoft Visual Studio Code. В качестве базы данных — PostgreSQL, СУБД — PgAdmin (psql). Для отладки функций серверной части — Postman.

# Проектно - конструкторская часть



## Общие сведения

Данная курсовая работа представляет собой интеграцию базы данных и веб-приложения вместе предназначенные для решения вопроса автоматизации покупки билета в кинотеатр.

## Структура веб-приложения

Разрабатываемая система представляет собой одностраничное приложение с клиентской и серверной частями, написанное на языке JavaScript c использованием библиотек необходимых для реализации предполагаемых функций.

## Концептуальная модель базы данных

Концептуальная модель является входными данными для процесса проектирования логической и физической, поэтому необходимо на начальном этапе определить сущности базы данных, их атрибуты и отношения между сущностями.

Для будущей базы данных сущностями будут выступать:

* сеанс;
* фильм;
* зритель (пользователь);
* билет;
* зал.

Каждая сущность должна обладать уникальным идентификатором, который будет обеспечиваться генератором случайных идентификаторов. Каждый экземпляр сущности должен однозначно идентифицироваться и отличаться от всех других экземпляров данного типа сущности. Также сущность должна обладать следующими свойствами:

* к одному и тому же имени должна всегда применяться одна и та же интерпретация;
* иметь один или несколько атрибутов, которые либо принадлежат сущности, либо наследуются через связь;
* иметь один или несколько атрибутов, которые однозначно идентифицируют каждый экземпляр сущности.

Каждая сущность может обладать любым количеством связей с другими сущностями модели.

На основе описания предметной области смоделируем данные с помощью методологии Питера Чена на рисунке 11.

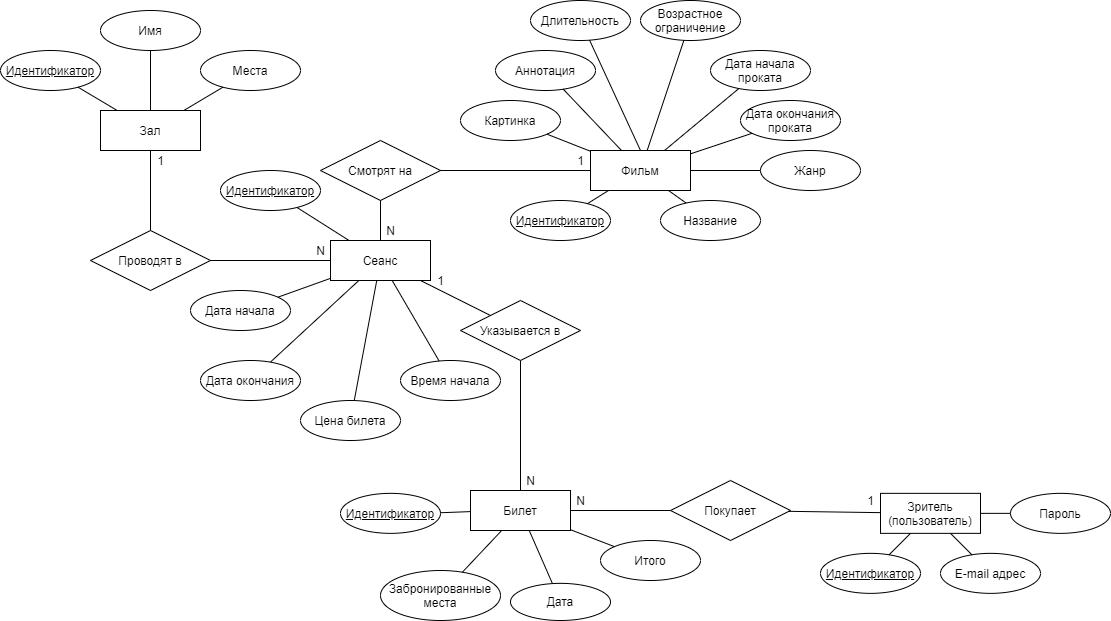


Рисунок 11 — Концептуальная схема проектируемой БД

## Логическая модель базы данных

При организации отношений на логическом уровне необходимо помнить о нормализации. Нормальная форма — свойство отношения в реляционной модели данных, характеризующее его с точки зрения избыточности, потенциально приводящей к логически ошибочным результатам выборки или изменения данных. Нормальная форма определяется как совокупность требований, которым должно удовлетворять отношение. Среди них:

* адекватность базы данных предметной области;
* легкость разработки и сопровождения базы данных;
* скорость выполнения операций обновления данных (вставка, обновление, удаление кортежей);
* скорость выполнения операций выборки данных.

Общее назначение процесса нормализации заключается в следующем:

* исключение некоторых типов избыточности;
* устранение некоторых аномалий обновления;
* разработка проекта базы данных, который является достаточно «качественным» представлением реального мира, интуитивно понятен и может служить хорошей основой для последующего расширения;
* упрощение процедуры применения необходимых ограничений целостности.

Учитывая всевозможные особенности реализации интерфейса в веб-приложении предлагается логическая схема указанная на рисунке 12.

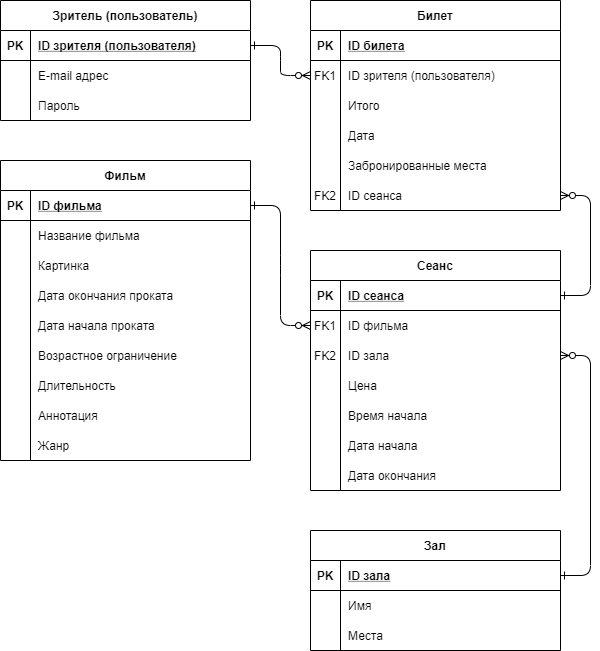


Рисунок 12 — Логическая схема проектируемой БД

## Физическое проектирование базы данных

Физическая модель данных зависит от конкретной СУБД, в ней содержится информация обо всех объектах БД. В физической модели важно описать всю информацию о конкретных физических объектах — таблицах, доменах, индексах, процедурах и т.д.

Проектируемая база данных должна удовлетворять следующим эксплуатационным требованиям:

* Оптимальная производительность;
* Расширяемость при реорганизации, гибкость при изменении, отказоустойчивость при удалении связей, атрибутов и сущностей;
* Загруженные в базу данных данные должны оставаться корректными;
* Загружаемые данные должны проверятся на корректность.

Определим назначение атрибутов сущностей, их формат и свойства:

Таблица 1  
Атрибуты сущности «Зритель (пользователь)»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Атрибут | Ключ | Формат | Размер | Свойства |
| user\_id | PRIMARY KEY | uuid | - | DEFAULT uuid\_generate\_v4() |
| user\_email | - | VARCHAR | 255 | NOT NULL |
| user\_password | - | VARCHAR | 255 | NOT NULL |

Таблица 2  
 Атрибуты сущности «Билет»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Атрибут | Ключ | Формат | Размер | Свойства |
| reservation\_id | PRIMARY KEY | uuid | - | DEFAULT uuid\_generate\_v4() |
| user\_id | FOREIGN KEY | uuid | - | NOT NULL REFERENCES users(user\_id) ON DELETE CASCADE |
| showtime\_id | FOREIGN KEY | uuid | - | NOT NULL REFERENCES showtimes(showtime\_id) ON DELETE CASCADE |
| booked\_seats | - | INTEGER[][] | - | NOT NULL |
| start\_date | - | DATE | - | NOT NULL |
| total | - | INTEGER | - | NOT NULL |

Таблица 3  
 Атрибуты сущности «Сеанс»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Атрибут | Ключ | Формат | Размер | Свойства |
| showtime\_id | PRIMARY KEY | uuid | - | DEFAULT uuid\_generate\_v4() |
| ticket\_price | - | INTEGER | - | NOT NULL |
| start\_at | - | VARCHAR | 255 | NOT NULL |
| start\_date | - | TIMESTAMP | - | NOT NULL |
| end\_date | - | TIMESTAMP | - | NOT NULL |
| movie\_id | FOREIGN KEY | uuid | - | NOT NULL REFERENCES movies(movie\_id) ON DELETE CASCADE |
| hallscheme\_id | FOREIGN KEY | uuid | - | NOT NULL REFERENCES hallschemes(hallscheme\_id) ON DELETE CASCADE |

Таблица 4  
 Атрибуты сущности «Фильм»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Атрибут | Ключ | Формат | Размер | Свойства |
| movie\_id | PRIMARY KEY | uuid | - | DEFAULT uuid\_generate\_v4() |
| movie\_title | - | VARCHAR | 255 | NOT NULL |
| movie\_description |  | VARCHAR | 255 | NOT NULL |
| image\_url | - | VARCHAR | 255 | NOT NULL |
| back\_image\_url | - | VARCHAR | 255 | NOT NULL |
| movie\_director | - | VARCHAR | 100 | NOT NULL |
| movie\_duration | - | VARCHAR | 255 | NOT NULL |
| movie\_genre | - | TEXT[] | - | NOT NUL |
| release\_date | - | DATE | - | NOT NULL |
| end\_date | - | DATE | - | NOT NULL |

Таблица 5  
 Атрибуты сущности «Зал»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Атрибут | Ключ | Формат | Размер | Свойства |
| hallscheme\_id | PRIMARY KEY | uuid | - | DEFAULT uuid\_generate\_v4() |
| hall\_name | - | VARCHAR | 100 | NOT NULL |
| seats | - | INTEGER[][] | - | NOT NULL |

## Архитектура базы данных

CREATE TABLE IF NOT EXISTS users(

user\_id uuid PRIMARY KEY DEFAULT uuid\_generate\_v4(),

user\_email VARCHAR(255) NOT NULL,

user\_password VARCHAR(255) NOT NULL

);

CREATE TABLE IF NOT EXISTS movies

(

movie\_id uuid PRIMARY KEY DEFAULT uuid\_generate\_v4(),

movie\_title VARCHAR(255) NOT NULL,

movie\_description TEXT NOT NULL,

image\_url VARCHAR(255) NOT NULL,

back\_image\_url VARCHAR(255) NOT NULL,

movie\_duration VARCHAR(255) NOT NULL,

movie\_director VARCHAR(100) NOT NULL,

movie\_genre TEXT[] NOT NULL,

release\_date DATE NOT NULL,

end\_date DATE NOT NULL

);

CREATE TABLE IF NOT EXISTS hallschemes (

hallscheme\_id uuid PRIMARY KEY DEFAULT uuid\_generate\_v4(),

hall\_name VARCHAR(100) NOT NULL,

seats INTEGER[][] NOT NULL

);

CREATE TABLE IF NOT EXISTS showtimes

(

showtime\_id uuid PRIMARY KEY DEFAULT uuid\_generate\_v4(),

ticket\_price INTEGER NOT NULL,

start\_at VARCHAR(255) NOT NULL,

start\_date TIMESTAMP NOT NULL,

end\_date TIMESTAMP NOT NULL ,

movie\_id uuid NOT NULL REFERENCES movies(movie\_id) ON DELETE CASCADE,

hallscheme\_id uuid NOT NULL REFERENCES hallschemes(hallscheme\_id) ON DELETE CASCADE

);

CREATE TABLE IF NOT EXISTS reservations (

reservation\_id uuid PRIMARY KEY DEFAULT uuid\_generate\_v4(),

user\_id uuid NOT NULL REFERENCES users(user\_id) ON DELETE CASCADE,

showtime\_id uuid NOT NULL REFERENCES showtimes(showtime\_id) ON DELETE CASCADE,

booked\_seats INTEGER[][] NOT NULL,

start\_date DATE NOT NULL,

total INTEGER NOT NULL

);

# Проектно – технологическая часть



## Требования к аппаратной платформе

Исходя из размера разработанного веб-приложения и ресурсов, требуемых для его исправной работы необходимо, чтобы компьютер обладал следующими минимальными характеристиками:

* интернет-соединение на основе протокола TCP/IP с пропускной способностью 50/100 Мбит/с;
* Двухъядерный процессор с тактовой частотой не менее 1.2 ГГц;
* 4Gb оперативной памяти;
* ОС Windows 8/macOS 10.12 Sierra;

Для взаимодействия с приложением необходимо наличие, компьютерной мыши и монитора.

Также возможно использование веб приложения на портативных устройствах: ноутбуках, планшетных компьютерах, смартфонах.

## Тестирование и отладка рабочей программы

Тестирование приложения производилось с использованием персонального компьютера с установленным браузером на основе Chromium. Тестирование включало в себя проверку всех функций проверку всех API функций приложения для работы с БД, корректного отображения запрашиваемой информации в браузере и отрисовки пользовательского интерфейса.

## Руководство администратора

Чтобы управлять веб-приложением администратору необходимо произвести следующие действия:

Установить IDE MS Visual Studio Code или его аналог

## Руководство пользователя

## Экспериментальные данные тестирования процедур и функциональных задач

## Экспериментальные данные тестирования процедур и функциональных задач

# Заключение

В результате выполнения курсовой работы было создано веб-приложение для автоматизации продажи билетов в кинотеатр на языке JavaScript с интегрированной базой данных.

# Список использованных источников

1.

2.

# Приложение