**ДЪРЖАВЕН ИЗПИТ**

**ЗА ПРИДОБИВАНЕ НА ТРЕТА СТЕПЕН НА ПРОФЕСИОНАЛНА КВАЛИФИКАЦИЯ – ЧАСТ ПО ТЕОРИЯ НА ПРОФЕСИЯТА**

**Д И П Л О М Е Н П Р О Е К Т**

**Тема: „Разработване на уеб базирано приложение за продажба на билети за кино верига“**

**професия: 481030 „Приложен програмист“**

**специалност: 4810301 „Приложно програмиране“**

**Дипломант: Александър Валериев Илиев 12Б клас**

**Ръководител на дипломния проект: инж. Димитрина Делийска**

**Подпис:**

**(дипломант)**

**Подпис:**

**(ръководител)**

**София, 2024 г.**

Съдържание

[1. Въведение 4](#_Toc163902557)

[2. Изложение 5](#_Toc163902558)

[2.1. Избор на технологии 5](#_Toc163902559)

[2.2. Избор на интернет езици 5](#_Toc163902560)

[2.2.1. HTML 5](#_Toc163902561)

[2.2.2. CSS 6](#_Toc163902562)

[2.2.3. JavaScript 7](#_Toc163902563)

[2.2.4. PHP 8](#_Toc163902564)

[2.2.5. MySQL 8](#_Toc163902565)

[2.2.6. Допълнителни библиотеки 9](#_Toc163902566)

[2.3. Разработка на приложението 11](#_Toc163902567)

[2.3.1. Реализация на приложението 11](#_Toc163902568)

[2.3.1.1. MVC архитектура 11](#_Toc163902569)

[2.3.1.2. Dataflow 12](#_Toc163902570)

[2.3.1.3. Страници 13](#_Toc163902571)

[2.3.1.4. Вход и регистрация 13](#_Toc163902572)

[2.3.1.5. Възможности на регистрирал се потребител 13](#_Toc163902573)

[2.3.1.6. Администраторски профил и функции 14](#_Toc163902574)

[2.3.1.7 Имейл след поръчка 15](#_Toc163902575)

[2.3.2. Функционалност на приложението 16](#_Toc163902576)

[2.3.2.1. Таблиците в база данни 16](#_Toc163902577)

[2.3.2.2. Връзките между таблиците 22](#_Toc163902578)

[2.3.2.3. CRUD заявки 23](#_Toc163902579)

[3. Заключение 26](#_Toc163902580)

[3.1 Тестване 26](#_Toc163902581)

[3.2 Постигнати цели 26](#_Toc163902582)

[3.3 Перспективи за развитие 26](#_Toc163902583)

[4. Приложения 28](#_Toc163902584)

[5. Използвани източници 34](#_Toc163902585)

# Въведение

В съвременния свят, където дигиталните иновации проникват във всеки аспект от нашия живот, онлайн платформите играят критична роля в потребителския опит. Сред хората, които предпочитат да избегнат физически контакт, предпочитаният метод за закупуване на кино билети е онлайн. Този вариант осигурява удобство и леснодостъпност за клиентите, които желаят да резервират места за предстоящи филми. Освен това, онлайн платформата може да предложи бърз и лесен начин за намиране на филми и график на прожекциите, избор на места и незабавно потвърждение на поръчката. Това е не само удобно, но и в съответствие със съвременните очаквания за бързо и ефективно обслужване.

Проектът с име ,,Cinemax” се създава с **цел да предостави на потребителите услуга, която да бъде по-бърза и с подобрени функционалности** в сравнение с вече съществуващите алтернативи. В основата си, то ще бъде създадено със стремежа да предложи на потребителите ефективен и удобен начин за използване чрез хубав интерфейс. Потребителите ще могат да достъпват услугата от всяко устройство, което има интернет достъп. Това включва компютри, телефони и таблети. Така се улеснява процеса на достъп и употреба на услугата, като предоставя гъвкавост и удобство на потребителите от вкъщи или навън.

Преди реализацията е важно да се направи анализ на съществуващите онлайн сайтове в тази среда. Това включва проучване на функционалността, дизайна и потребителския опит на водещите платформи.

Примерна алтернатива: [Cinemacity - Мол София (cinemacity.bg)](https://www.cinemacity.bg/cinemas/mallofsofia/1261#/buy-tickets-by-cinema?in-cinema=1261&at=2024-04-05&view-mode=list). ,,Cinemax” ще може да предложи подобрени функционалности и възможности. Едно от основните му предимства е планираната възможност за рейтинг и нотификации, които липсват в другите платформи. В допълнение, той ще предоставя удобството за генериране на код при закупуване на билети, както и притежание на собствена профилна страница с профилна снимка и история.

Резултатите от тази дипломна работа се очаква да включват успешното разработване на уеб базирано приложение, което предлага удобен и ефективен начин за закупуване на билети за кино или просто сърфиране в сайта. Като цяло, тази работа ще допринесе за подобряване на потребителския опит при пазаруването и за развитието на сферата на онлайн кино платформите.

# Изложение

## Избор на технологии

При определянето на технологиите за разработка са предпочетени утвърдени езици и технологии, които да гарантират стабилност и надеждност на проекта. В същото време, са избрани и най-съвременните решения, за да се осигури актуалност за дълго време и възможност за развитие в бъдеще.

Технологиите са разделени на две основни категории: backend (сървърна част) и frontend (клиентска част). Backend е отговорен за основната функционалност на уебсайта, включително управление на потребителски профили, удостоверяване и упълномощаване, обработка на данни и други важни операции, които осигуряват стабилността на уеб приложението (напр. PHP). Frontend, от друга страна, представлява технологиите, които са използвани за създаване на потребителския интерфейс, позволяващ на потребителите да взаимодействат лесно и интуитивно със сървърната функционалност чрез уебсайта (напр. HTML).

## Избор на интернет езици

За проекта са избрани интернет езиците PHP, HTML, CSS и JavaScript, които са изключително подходящи за разработката на уеб базираното приложение и MySQL за базата данни в него.

## 2.2.1. HTML

HTML (HyperText Markup Language) е основен език за маркиране, който се използва за създаване и структуриране на уеб страници. Той е езикът, който определя структурата и съдържанието на уеб страниците, като предоставя средство за изграждане на текстове, графика, мултимедия и връзки.

HTML играе критична роля във всеки уеб сайт, като осигурява следните ползи:

* **Структура и организация**: HTML предоставя ясна структура за уеб страниците, като позволява лесно организиране на различните елементи, като заглавия, параграфи, списъци, таблиц и други. Това прави уеб страниците по-лесни за разбиране и навигация за потребителите.
* **Семантика**: HTML разполага с разнообразие от елементи, които имат семантично значение. Това означава, че не само се определя визуалният вид на елементите, но и техните значимости за търсачки и устройства за адаптивен дизайн. Например, използването на елементи като <header>, <nav>, <article>, <section>, <footer> помага за яснота и семантика на уеб страниците.
* **Достъпност:** Правилното използване на HTML е важно за достъпността на уеб сайтовете. Семантичните елементи и правилната структура на страницата правят уеб съдържанието по-достъпно за хора с увредено зрение или други увреждания.
* **Съвместимост**: HTML е стандартен език за маркиране, който е поддържан от всички основни уеб браузъри. Това означава, че уеб страниците, създадени с HTML, ще бъдат съвместими и достъпни за широк кръг от потребители.
* **Интеграция с други технологии:** HTML лесно се интегрира с други уеб технологии като CSS (Cascading Style Sheets) за стилизиране на страниците и JavaScript за добавяне на интерактивност и динамичност.

## 2.2.2. CSS

CSS (Cascading Style Sheets) е стилов език, който се използва за оформяне и стилизиране на уеб страници. Ето някои ключови аспекти и важна информация за CSS:

Разделяне на съдържанието и визуалния дизайн: CSS позволява ясно разделение между съдържанието и визуалния дизайн на уеб страниците. Това означава, че HTML се използва за структуриране на съдържанието, докато CSS се използва за задаване на визуалните стилове като цветове, шрифтове, разположение на елементите и др.

* **Каскадност**: CSS правилата се прилагат във връзка с каскадата на стиловете. Това означава, че ако има повече от едно правило, което се отнася до даден елемент, те се прилагат в определен ред, като по-специфичните правила имат по-висок приоритет пред по-общите.
* **Селектори**: CSS използва селектори за да определи кои HTML елементи ще бъдат стилизирани с определените правила. Селекторите могат да бъдат базирани на елементи, класове, идентификатори, атрибути и други.
* **Поддръжка на различни медии**: CSS предлага възможност за стилизиране на уеб страниците за различни медии, като екрани на компютри, принтери, мобилни устройства и други. Това позволява разработчиците да създават адаптивни и отзивчиви дизайни, които да се адаптират към различните устройства и екрани.
* **Анимации и преходи**: CSS предоставя възможност за създаване на анимации и преходи, които да подобрят визуалния опит на потребителите. Това включва анимиране на свойства като цветове, размери, позиции и други, както и добавяне на преходи между различни състояния на елементите.
* **Позициониране**: CSS предоставя различни методи за позициониране на елементите върху уеб страницата, включително стандартното поточно позициониране, абсолютно позициониране, фиксирано позициониране и други. Тези техники позволяват създаването на сложни и гъвкави макети за уеб дизайн.

## 2.2.3. JavaScript

JavaScript (JS) е високо-нивен, интерпретиран програмен език, използван за създаване на динамични и интерактивни уеб сайтове. Той е език за скриптове, който се изпълнява в браузъра и дава възможност за взаимодействие с потребителите, манипулиране на HTML и CSS, изпращане на заявки към сървъри и други.

Някои ключови характеристики:

* **Интерактивност**: JavaScript позволява създаването на интерактивни уеб страници, които реагират на действията на потребителя, като кликване, навеждане на мишката и въвеждане на данни в полета за въвеждане.
* **Манипулация на HTML и CSS:** С помощта на JavaScript можем да променяме структурата и стиловете на уеб страниците динамично, като добавяме, премахваме или променяме HTML елементи и CSS свойства.
* **Събития (Events):** JavaScript позволява добавянето на събития към елементи на уеб страницата, като например кликване, преминаване с мишката или въвеждане на клавиши. Това позволява реакция на страницата към действията на потребителя.
* **Асинхронност:** JavaScript поддържа асинхронно програмиране, което позволява изпълнението на код, който не блокира изпълнението на останалата част от приложението. Това е особено полезно при извършване на заявки към сървъри и обработка на данни от външни източници.
* **Обектно-ориентирана природа:** JavaScript е обектно-ориентиран език, който позволява създаването на обекти и работа с тях, което улеснява организацията и управлението на кода.

## 2.2.4. PHP

PHP (Hypertext Preprocessor) е широко използван език за програмиране, предимно за уеб разработка, който се използва за създаване на динамични уеб страници и уеб приложения. Ето някои важни аспекти и информация за PHP:

* **Сървърна страна**: PHP е език за програмиране, който се изпълнява на страната на сървъра. Това означава, че PHP кодът се изпълнява на уеб сървъра, преди да се изпрати резултатът към уеб браузъра на потребителя.
* **Интеграция с HTML и CSS**: PHP се интегрира лесно с HTML и CSS, което позволява разработчиците да създават динамични уеб страници, които да се адаптират и реагират на действията на потребителите.
* **Гъвкавост:** PHP предлага голяма гъвкавост и многофункционалност, като поддържа различни операционни системи и уеб сървъри. Той може да работи с различни бази данни, като MySQL, PostgreSQL, SQLite и други.
* **Динамично съдържание:** PHP позволява генерирането на динамично съдържание на уеб страниците, като например показване на персонализирани съобщения, обработка на формуляри, работа с бази данни и други.
* **Широка употреба:** PHP е един от най-популярните езици за уеб програмиране и се използва от милиони уеб разработчици по целия свят. Той е отворен, безплатен и има голяма общност от разработчици, което прави неговото изучаване и използване достъпно за всички.

## 2.2.5. MySQL

MySQL е мощна и популярна релационна база данни, която се използва широко във всякакви видове приложения. Ето някои ключови аспекти и характеристики на MySQL:

* **Релационна база данни**: MySQL е релационна база данни, което означава, че данните се съхраняват в таблици с редове и колони. Това позволява лесно управление на информацията и извличането й посредством структурирани заявки.
* **Мощност и скалируемост**: MySQL е известен с мощността си и възможността да обслужва големи обеми данни. Той предлага висока производителност и скалируемост, което го прави подходящ избор за различни видове приложения, от малки уеб сайтове до големи корпоративни системи.
* **Отворен код**: MySQL е свободно достъпна база данни с отворен код, което означава, че всеки може да я използва, променя и разпространява според нуждите си, безплатно. Това прави MySQL предпочитан избор за много разработчици и организации, които ценят свободата и гъвкавостта.
* **Съвместимост**: MySQL е съвместима с различни операционни системи като Windows, Linux и macOS, както и с различни програмни езици за програмиране като PHP, Python, Java и други. Това прави MySQL подходящ за разработка на разнообразни видове софтуерни приложения.
* **Сигурност**: MySQL предлага различни механизми за сигурност, включително автентикация на потребителите, права за достъп и криптиране на данните. Това помага за защитата на данните от неоторизиран достъп и за осигуряването на поверителност и цялостност на информацията.
* **Инструменти за администриране**: MySQL предлага различни инструменти за администриране и управление на базата данни, включително конзолни инструменти като MySQL Shell и графични интерфейси като MySQL Workbench. Тези инструменти правят управлението на базата данни по-лесно и удобно за администраторите.

## 2.2.6. Допълнителни библиотеки

Освен изброените интернет езици в чистия им вид, за реализацията могат да помогнат различни видове библиотеки. В проекта са избрани за ползване Sweetalert, Bootstrap, Jquery, Vibrant на някои места и навсякъде Font Awesome, а функционалността на имейла идва от PHPMailer.

* **SweetAlert**: SweetAlert е JavaScript библиотека, която се използва за създаване на красиви и интерактивни модални прозорци за съобщения. Тя предлага по-добър и по-персонализиран опит за потребителите в сравнение със стандартните браузърни съобщения. SweetAlert предоставя различни стилове и възможности за настройка на съобщенията, което я прави популярен избор за усъвършенстване на интерфейса на потребителите.
* **Bootstrap**: Bootstrap е отворен код CSS рамка за уеб разработка, която предоставя готови стилове, компоненти и JavaScript plugins за създаване на отзивчиви уеб приложения. Тя е известна със своята лесна за използване сет система за сети, която помага на разработчиците да създадат уебсайтове, които изглеждат добре на всички устройства и екрани. Bootstrap също така предлага голямо разнообразие от компоненти, като бутони, форми, навигационни менюта и други, които правят разработката на уебсайтове по-лесна и по-бърза.
* **jQuery**: jQuery е библиотека за JavaScript, която улеснява манипулирането на HTML документи, обработването на събития, анимирането на елементи и взаимодействието с AJAX заявки. Тя предоставя опростен синтаксис и мощни функции, които правят JavaScript програмирането по-лесно и по-ефективно. jQuery се използва широко в уеб разработката за създаване на динамични и интерактивни уеб приложения.
* **Vibrant**: Vibrant е JavaScript библиотека, която се използва за извличане на доминиращи цветове от изображения. Тя може да бъде полезна за автоматично генериране на цветове за дизайна на уебсайта въз основа на използваните изображения. Vibrant предоставя API за анализиране на цветовете в изображенията и извличане на палитра от цветове, което може да бъде използвано за създаване на хармоничен и привлекателен дизайн.
* **Font Awesome**: Font Awesome е библиотека с голям набор от векторни икони, които могат да бъдат лесно вграждани в уеб приложения и дизайн. Предлага над 7,000 такива, които покриват широк спектър от категории като социални мрежи, възможности за навигация, медии, технологии, възрастови групи, знакове и много други.
* **PHPMailer**: PHPMailer е мощна библиотека за изпращане на имейли от уебсайтове, изградени с PHP. Тя предоставя лесен и надежден начин за изпращане на електронни писма през SMTP или през локален мейл сървър. PHPMailer позволява на разработчиците да вграждат възможността за изпращане на имейли в техните проекти, като предоставя обширен функционален набор и поддръжка на различни функции като вложени файлове, HTML форматиране, SMTP аутентикация и други.

## 2.3. Разработка на приложението

Разработката на приложение, независимо от неговия характер, е сложен процес. Преди да започнем работата по същинската част, е важно да имаме ясна представа за това как всичко ще изглежда и как бихме се чувствали като потребители, когато използваме приложението.

## 2.3.1. Реализация на приложението

## 2.3.1.1. MVC архитектура

За да се обясни MVC (Model-View-Controller) архитектурния шаблон, е добре да разгледаме неговите компоненти и как те си взаимодействат.

1. **Модел (MySQL)**:

* Моделът представлява данните и бизнес логиката на приложението.
* Той се грижи за обработката, валидацията и манипулирането на данните, които приложението използва.
* Моделът не зависи от визуалния аспект на приложението

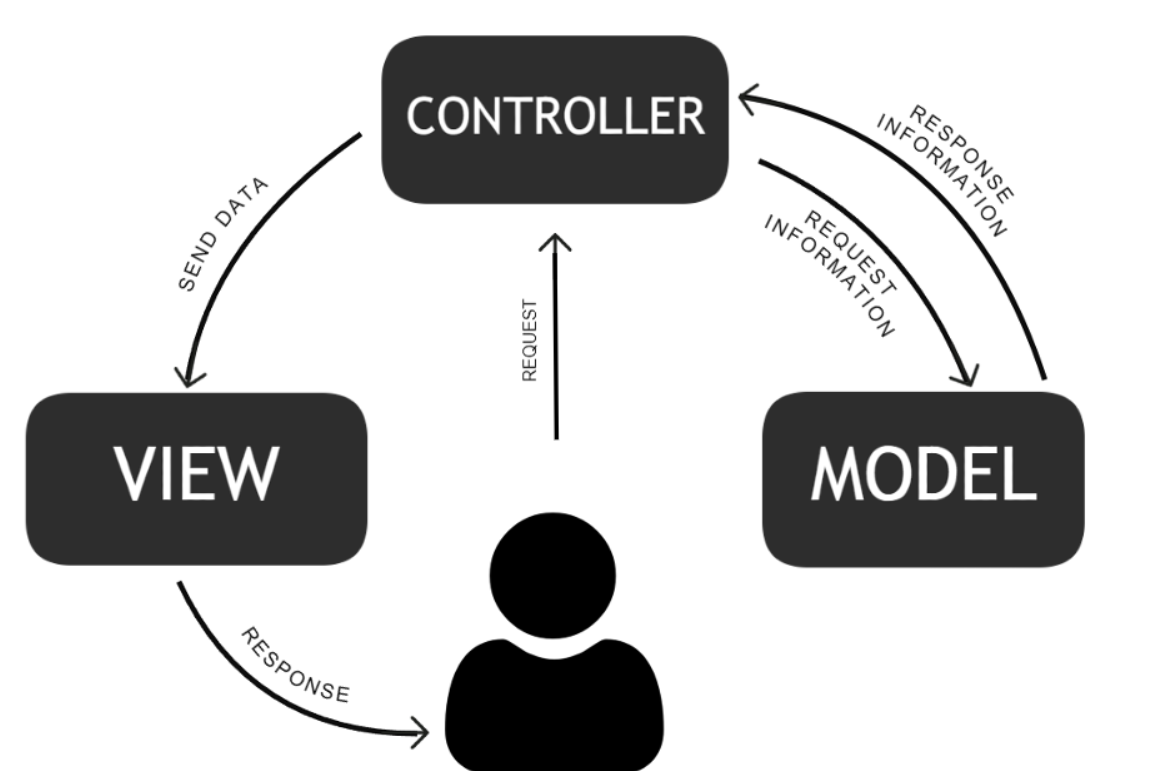
1. **Изглед (HTML):**

* Изгледът представлява визуалното представяне на данните, предоставени от модела.
* Той се отговаря за визуализирането на информацията, която потребителят вижда в браузъра или друго устройство.
* Изгледът обикновено е представен в HTML, CSS и JavaScript и се грижи за интерфейса на потребителя.

1. **Контролер (PHP):**

* Контролерът служи като посредник между модела и изгледа.
* Той приема заявките от потребителите (HTTP заявки, например), обработва ги и определя кой модел и изглед трябва да бъдат използвани за тяхното обслужване.
* Контролерът обработва логиката на приложението и връща резултат към потребителя в подходящия формат.

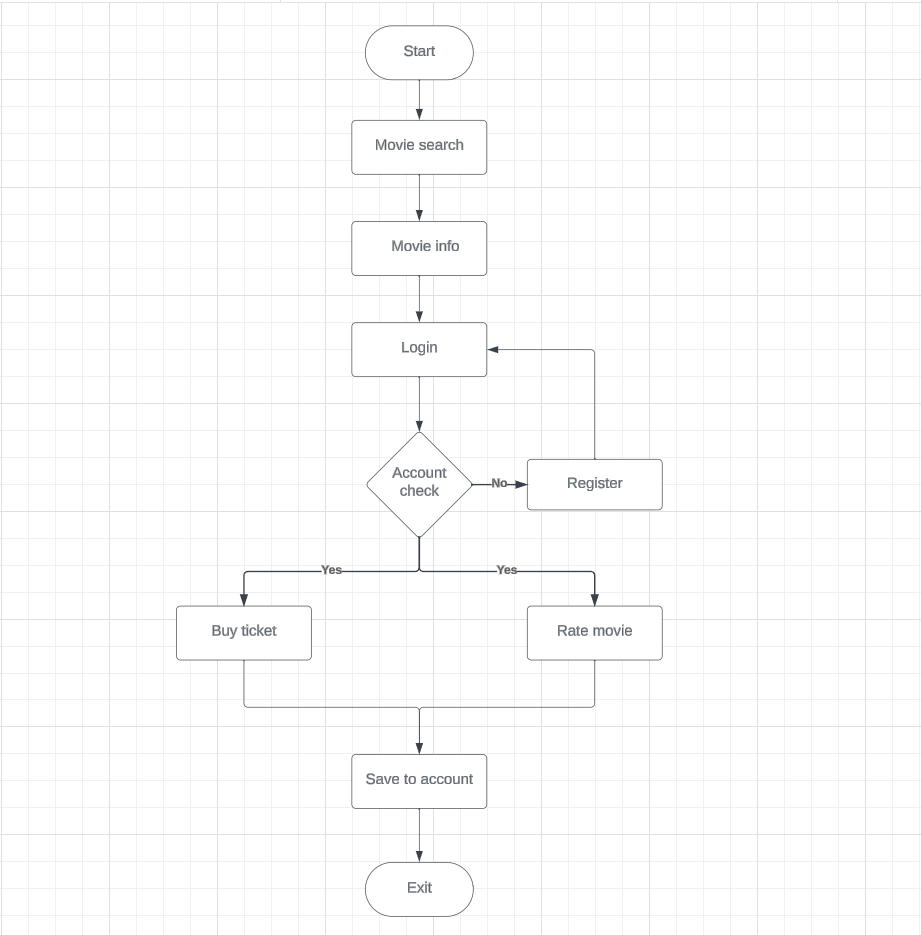
Използвайки тези три компонента в комбинация, MVC архитектурата разделя приложението на логически отделни части, които могат да се разработват и поддържат по-лесно. Моделът се грижи за данните, изгледът за визуализацията им, а контролерът за обработката на заявките и координиране на действията между модела и изгледа.



Фиг. 1: MVC диаграма

## 2.3.1.2. Dataflow

На Фиг. 1 е представено как протича потребителското изживяване. Потребителите започват с избора на филм, който желаят да гледат. Този етап им предоставя възможността да разгледат различни филми, чрез меню или търсачка, за да вземат решение. След като са избрали филм, потребителите могат да прегледат подробна информация за него. Това им помага да се запознаят по-добре с него, преди да вземат окончателно решение. Ако потребителят вече има акаунт, може да продължи напред с покупката на билет или да рейтне филма. Ако не, има възможност да създаде нов акаунт, което също допринася за по-удобен достъп и управление на билетите си в бъдеще. След успешно влизане в системата, потребителите могат да закупят билети за желаната прожекция. Този процес е лесен и бърз, като включва избор на места и плащане. Накрая, информацията за закупените билети се съхранява в потребителския акаунт. Това позволява лесен достъп до билетите и историята на посещенията, както и възможност за рейтване на филмите.



Фиг. 2: Dataflow диаграма

## 2.3.1.3. Страници

**А) Начална страница**

При влизане в началната страница, потребителите имат възможност да прегледат всички филми чрез главното меню или търсачката. Освен това, могат да видят слайдшоу с най-новите филми и да разгледат менюто с храна и напитки плюс цените им. Допълнителна информация може да бъде намерена във долния колонтитул на страницата. Чрез навигационния бар може да се влезе в профил или да се види най-близката премиера.

**Б) Страница на филм**

Всеки филм си има собствена страница. Тя е разделена на няколко сектора от информация, вземана от базата данни. Тук е използван Vibrant, за да може фонът да е различен градиент за всеки филм, като библиотеката калкулира и взима двата най-главни цвята от иконката на филма. Също таблицата с прожекции е под информацията, която автоматично се генерира от базата данни и показва кой ден и час за следващите 7 дни има прожекции. Ако няма прожекции ще изпише това, а ако филмът не е излязъл няма да я има таблицата. ([Фиг. 16](#_Приложения))

**В) Страница за избор на място**

След избор на прожекция се отваря страница за избор на места и заплащане. Секторът за места се генерира автоматично от базата данни чрез информацията за конкретната зала и нейните места. Първо се чете информацията за залата и се генерират броя на местата, които се показват на страницата. След това се извлича информация за прожекцията, и посредством нея се чете таблицата с поръчки. Местата, които вече са резервирани за текущата прожекция, се отбелязват в червен цвят и не могат да бъдат селектирани от потребителя. На дъното на страницата се намира секцията за заплащане, където потребителят може да въведе код за промоция, ако такъв е наличен. При потвърждаване на покупката, поръчката се запазва в базата данни, а потребителят получава потвърждаващ имейл с подробности за транзакцията. ([Фиг. 15](#_Приложения))

**Г) Други**

Влизането и регистрацията са в отделни страници. В долния колонтитул са сложени примерни информативни страници като въпроси, заявка за работа и др., които не се отварят за момента. Потвърждението на поръчка с генериране на код също е отделна страница.

## 2.3.1.4. Вход и регистрация

Разбира се, за да може потребителят да получи специални функционалности, ще трябва да създаде акаунт, ако няма такъв. След като влезе в профила си, той ще стои вписан 1 месец или до натискане на бутона Изход.

## 2.3.1.5. Възможности на регистрирал се потребител

При наличие на профил се откриват различни възможности:

**Профилна страница:** Получаваш достъп до лична страница, в която можеш да си видиш поръчките си, рейтингите, лични данни и възможност за променянето на някои от тях. В тази страницата е и изтриването на профил.

**Рейтинг на филм**: Без значение дали си гледал филма, получаваш възможност да дадеш рейтинг от 1 до 5 в страницата за филма чрез звезди.

**Задаване на напомняне:** За филми, които не са излезли в прожекция се появява бутон камбанка при датата на излизане в страницата на филма. При зададено напомняне, бутонът ще свети в синьо.

**Запазване на билет:** Освен само гледане на графика на филм, вече получаваш възможност при кликване на бутона за прожекция да се появи меню за запазване на билет и от него да отвориш страницата за запазване на място.

**Получаване на код:** След покупка получаваш възможност да генерираш код за промоция, който веднага проверяваш дали е активен.При активен такъв изписва колко % е отстъпката. Не е персонален код и друг може да го генерира също, затова трябва да се използва бързо.

## 2.3.1.6. Администраторски профил и функции

След като потребител стане администратор, той придобива права за управление на базата данни. В навигационния бар се появява ново меню за администратори, което води към специална страница, достъпна само за тях. Ако някой потребител се опита да достъпи тази страница чрез променяне на линка, ще получи грешка 404. На тази страница се предоставя списък с всички филми, включително и тези, които не са активни. Администраторът има възможност да променя статуса на филмите и да добавя нови през менюто в страничната част. Освен това, има и друго меню, което представя всички активни кодове, позволявайки на администратора да добавя или премахва нови. След менюто, той може да преглежда всеки профил в списък по имейл и през него да отвори профилната страница на потребителя. Тя е достъпна само за администраторите и в нея се намира бутонът за триене на профила от базата данни. Ако някой от потребителите е заявил изтриване на профила си, този профил ще бъде маркиран с жълто най-отгоре на списъка. Над този списък е и менюто за създаване на админ. В него се пише имейл и избираш да добавиш или премахнеш такъв. Има проверка, чрез която никой не може да премахне главния админ.

Други администраторски функционалности включват възможността да се редактира цялата информация за филма от информационната страница на филма, чрез специално меню от страничната част. Също така, чрез по-долно меню за прожекции има опция да се добавят прожекции ръчно като се избира дата, час и зала. Тези менюта са достъпни само за потребители с администраторски права и не се виждат от нормален потребител.

Администраторът получава също така и повече видимост за някои елементи. Например това значи, че само админ може да отвори страница на неактивен филм. Също така само той може да види таблицата с прожекции на неизлязъл филм, което служи за подготовката им преди премиерата.

Когато потребителят се опита да отвори страници, в които не трябва да бъде (например страницата за регистрация, имайки вече акаунт или неактивен филм), то автоматично ще бъде изпратен в главната страница. А когато се опита да отвори несъществуваща страница, то ще изпише грешка 404, както и администраторските страници, за да се скрие, че те са съществуващи.

## 2.3.1.7 Имейл след поръчка

Накратко за използването на PHPMailer е нужно инсталиране на файловете. След това трябва да се добавят към php кода чрез:

use PHPMailer\PHPMailer\PHPMailer;

use PHPMailer\PHPMailer\Exception;

require 'emailer/src/Exception.php';

require 'emailer/src/PHPMailer.php';

require 'emailer/src/SMTP.php';

Настройва се smtp host и от кой профил ще се изпраща:

$mail->SMTPDebug = 2; // Enable verbose debug output

      $mail->isSMTP(); // Send using SMTP

**$mail->Host** = 'smtp.gmail.com'; // Set the SMTP server to send through GMAIL

      $mail->SMTPAuth = true; // Enable SMTP authentication

**$mail->Username** = 'alexiliev111@gmail.com'; // SMTP username

  $mail->setFrom('alexiliev111@gmail.com', 'Cinemax');

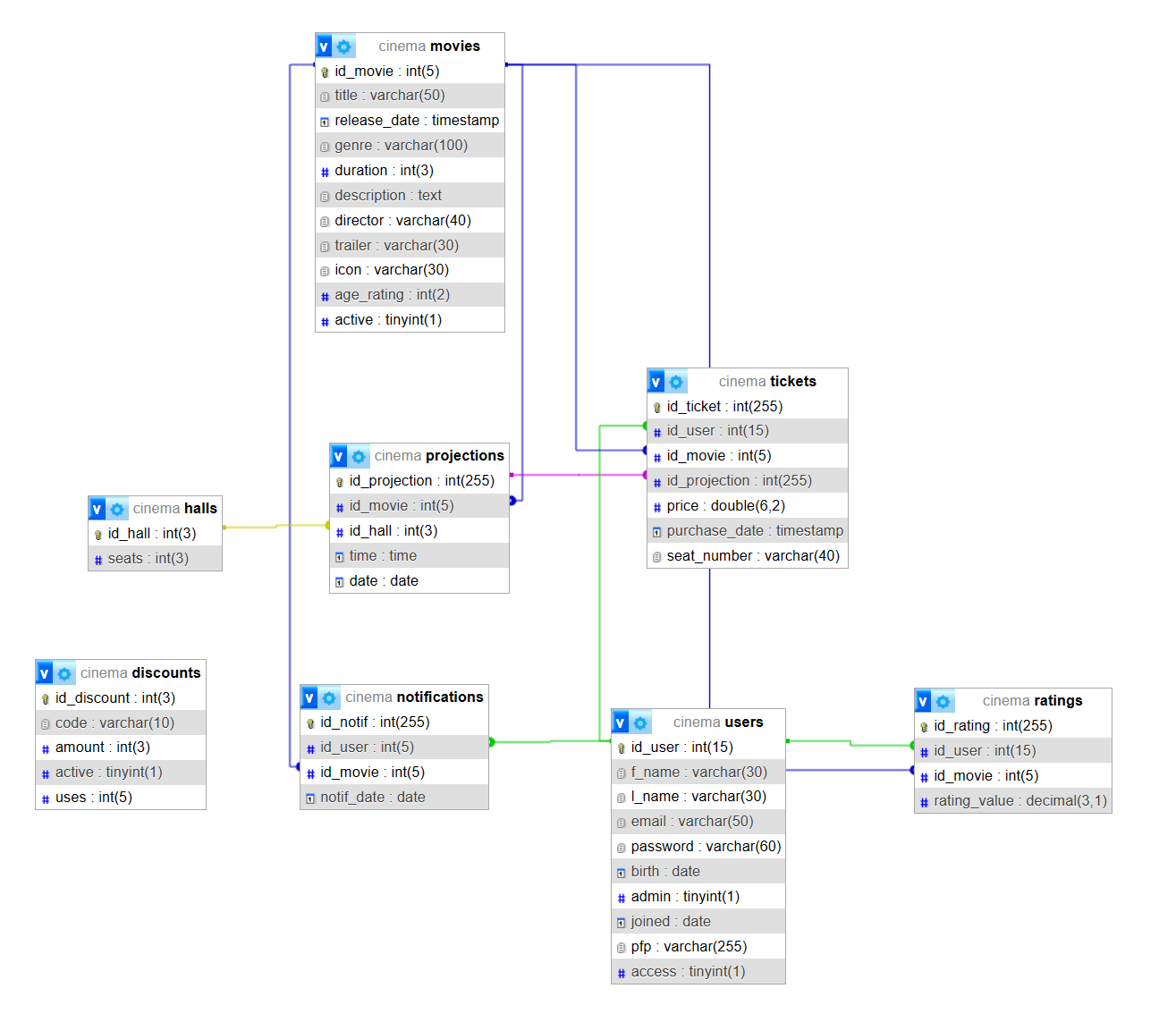
За да може да се изпрати имейлът, трябва да се направи app password на gmail профила, което включва и 2FA. Тя представлява нещо като **$mail->Password** = 'ttim nafm lpfn mzhf'; (SMTP password). Накрая се пише HTML структура на тялото на имейла и се изпраща чрез  **$mail->send();** Така се настройства функционалността и след това в кода са използвани различни вградени функции за допълнителната визия на имейла. Финалното тяло съдържа няколко параграфа за информацията на поръчката и снимка на билет от облак. ([Фиг. 5](#_Приложения))

За заключение, PHPMailer е удобна библиотека за изпращане на имейли в PHP. Този конкретен код обаче работи само локално, когато на компютъра има валиден Gmail профил. За глобално използване ще бъдат необходими допълнителни настройки или използването на друга библиотека за изпращане на имейли. Важно е да се отбележи, че този код е само пример и не съдържа всички необходими проверки и обработки за реално приложение.

## 2.3.2. Функционалност на приложението

## 2.3.2.1. Таблиците в база данни

За проекта са създадени 8 таблици, с възможност за добавяне и на още в бъдеше за нови функционалности или оптимизации.

****

Фиг. 3: ER диаграма на базата данни и връзките

|  |  |  |
| --- | --- | --- |
| Таблица users | | |
| **Име на поле** | **Тип** | **Описание** |
| id\_user | INT, AI | Уникален идентификатор за всеки потребител. Използва за лесното идентифициране на потребителите в системата и за свързване с други таблици в базата данни. |
| f\_name | VARCHAR | Съхранява първото име на потребителя. Това е важно за персонализацията на потребителския опит и за идентификация на потребителите във визуалните елементи на уеб сайта. То ще бъде изписано в навигационния бар. |
| l\_name | VARCHAR | Съхранява фамилното име, което се вижда само в профилната страница. |
| email | VARCHAR | Много важно поле за идентификация на потребителите. Имейл адресът се използва за влизане в системата. Разбира се, не може да се повтаря в таблицата. Използва се за изпращане на имейл при покупка на билет. |
| password | VARCHAR | Съхранява се като хашнат символен низ за повишаване на сигурността на потребителските данни. Паролата трябва да бъде криптирана, за да се предотврати злоупотреба или неоторизиран достъп до профилите на потребителите. Може да бъде сменена през профилната страница. |
| birth | DATE | Информацията за датата на раждане. За момента служи като лична информация в профилната страница. |
| admin | TINYINT | Съхранява 0 или 1 в зависимост дали потребителят е админ. Може да бъде оптимизирано с отделна таблица само за потребители админи, вместо всеки запис да го задържа. Автоматично получава 0 при създаване на акаунт. |
| joined | DATE | Датата на регистриране на потребителя |
| pfp | VARCHAR | Линк към картинка за профилна снимка. При създаване на профил, потребителят автоматично има сложена такава:  https://static.vecteezy.com/system/resources/thumbnails/009/292/244/small/default-avatar-icon-of-social-media-user-vector.jpg |
| access | TINYINT | Полето access може да съхранява информация за заявките на потребителите за изтриване на профилите им. Оптимизацията на този процес може да включва използване на отделна таблица, която да съхранява заявките за изтриване или в полето за админ да се добави възможност за стойност 2 . Нуждата за полето се породи от това, че когато потребител иска да изтрие профила си поради лични причини, това ще доведе то триене и на историята му за покупки. Затова чрез него се следи кой потребител желае да бъде изтрит, а админът е този, който финално трие от базата данни. Потребителя не може да влезе в акаунта си сред тази заявка, но може да се опита да се свърже с компанията за отказ. |

|  |  |  |
| --- | --- | --- |
| Таблица movies | | |
| **Име на поле** | **Тип** | **Описание** |
| id\_movie | INT, AI | Идентификатор на филма |
| title | VARCHAR | Заглавието на филма |
| director | VARCHAR | Режисьорът |
| genre | VARCHAR | Жанрът на филм. Може да се изреждат поджанрове или няколко жанра чрез запетаи, но трябва да има поне 1 главен, ако искаме да се филтрира в менюто с жанрове |
| release\_date | TIMESTAMP | Премиерата на филм. От това поле зависи филмовото меню. Чрез него филмите се делят на „в продажба“, „очаквайте скоро“ и „премиера“. Премиерата е всеки филм излязъл до 14 дена от днешна дата. |
| duration | INT | Съдържа времетраенето в минути (60 за 1 час; 120 за 2 часа) |
| descritpion | TEXT | Описанието за филм |
| trailer | VARCHAR | Съдържа съкратен линк от YouTube, който се използва в iframe. Изглежда като xy8aJw1vYHo (буквите след youtube.com) |
| icon | VARCHAR | Името на изтеглен файл за иконата на филм в папка icons |
| age\_rating | INT | Възрастова оценка на филма |
| active | TINYINT | Съдържа 0 или 1 в зависимост дали филмът е в продажба. Ако е 0, филмът е неактивен и само админ може да го види |

|  |  |  |
| --- | --- | --- |
| Таблица ratings | | |
| **Име на поле** | **Тип** | **Описание** |
| id\_rating | INT, AI | Идентификатор на рейтинг |
| id\_user | INT | Външен ключ за потребителите |
| id\_movie | INT | Външен ключ за филм |
| rating\_value | DECIMAL | Запазва рейтинга на потребител като десетично число, но чрез сайта се използват стойности 1-5. То е decimal за вероятни бъдещи промени в системата за рейтинг за по-конкретно десетично число. |

|  |  |  |
| --- | --- | --- |
| Таблица notifications | | |
| **Име на поле** | **Тип** | **Описание** |
| id\_notif | INT, AI | Идентификатор на нотификацията |
| id\_user | INT | Външен ключ за потребител |
| id\_movie | INT | Външен ключ за филм |
| notif\_date | DATE | Определя кога потребителят ще получи напомняне за предстоящата премиера. При настъпване на указаната дата, потребителят ще бъде уведомен в сайта за наличието на филма |

|  |  |  |
| --- | --- | --- |
| Таблица halls | | |
| **Име на поле** | **Тип** | **Описание** |
| id\_hall | INT | Номер на залата |
| seats | INT | Брой места в нея. От това зависи колко места ще има в страницата за избор на място |

|  |  |  |
| --- | --- | --- |
| Таблица projections | | |
| **Име на поле** | **Тип** | **Описание** |
| id\_projection | INT, AI | Идентификатор на рейтинг |
| id\_movie | INT | Външен ключ за филм |
| id\_hall | INT | Външен ключ за зала |
| time | TIME | Часът на прожекцията |
| date | DATE | Датата на прожекцията. В бъдеще часът и датата могат да се комбинират в timestamp |

|  |  |  |
| --- | --- | --- |
| Таблица discounts | | |
| **Име на поле** | **Тип** | **Описание** |
| id\_discount | INT, AI | Идентификатор на код |
| code | VARCHAR | Наименованието на кода, например буквена комбинация |
| amount | INT | Число от 0-100, което ще бъде използвано като % |
| uses | INT | Брой на използвания. Служи като статистика. При всяко използване се прибавя 1. |
| active | TINYINT | Това поле показва дали отстъпката е активна в момента или не. Ако отстъпката е активна, потребителите могат да я използват при покупка на билети. В противен случай, кодът може да бъде неактивен, ако е използван или е деактивиран от администратор. |

|  |  |  |
| --- | --- | --- |
| Таблица tickets | | |
| **Име на поле** | **Тип** | **Описание** |
| id\_ticket | INT, AI | Идентификатор на поръчката/билета |
| id\_user | INT | Външен ключ за потребител |
| id\_projection | INT | Външен ключ за прожекция |
| id\_movie | INT | Външен ключ за филм. Може да се премахне нуждата от това поле заради ключа от прожекция, но с него по-бързо и лесно се достъпва информация за филма |
| price | DOUBLE | Финалната цена на поръчката |
| purchase\_date | TIMESTAMP | Дата и час на поръчката. Попълва се автоматично веднага при успешна поръчка. Може да бъде използвано като статистика за продажби за период. |
| seat\_number | VARCHAR | Пазят се избраните числа на местата чрез запетаи напр (59, 60, 61). Чрез това поле и ключа за прожекция се генерират свободните места в страницата за избор на място. |

## 2.3.2.2. Връзките между таблиците

Връзките в релационните бази данни са основен аспект от базовия модел на данни и играят важна роля в организацията и управлението на данните. Те позволяват на базите данни да бъдат ефективно структурирани и свързани помежду си, като същевременно запазват целостта и консистентността на данните.

**Основни ключове** (Primary Keys): Основният ключ е уникален идентификатор на запис в таблица. Той се използва за единствено идентифициране на записите в таблицата и осигурява уникалност на данните. Обикновено основният ключ се свързва с външен ключ в друга таблица, за да се създаде връзка между тях. В проекта няма такава връзка.

**Външни ключове** (Foreign Keys): Външният ключ е поле или комбинация от полета в таблица, които се свързват с основния ключ в друга таблица. Това създава връзка между две таблици и определя зависимост между тях.

**Едно към едно** (One-to-One Relationship 1:1): Тук всяко поле в една таблица има точно един запис, свързан с него в друга таблица. В проекта не е използвана тази връзка.

**Едно към много** (One-to-Many Relationship 1:n): Тук всяко едно значение в едно поле в таблица може да бъде свързано с множество стойности в съответното поле в друга таблица. Това създава връзка един към много:

Movies - Notifications: 1:n (Един филм може да има много известия)

Movies - Projections: 1:n (Един филм може да има много прожекции)

Movies - Ratings: 1:n (Един филм може да има много рейтинги)

Movies - Tickets: 1:n (Един филм може да има много билети)

Halls - Projections: 1:n (Една зала може да има много прожекции)

Users - Notifications: 1:n (Един потребител може да има много нотификации)

Users - Ratings: 1:n (Един потребител може да има много рейтинги)

Users - Tickets: 1:n (Един потребител може да има много билети)

**Много към много** (Many-to-Many Relationship n:n): В този случай множество стойности в едно поле в таблица може да бъде свързано с множество стойности в съответното поле в друга таблица. Тази връзка обикновено се постига чрез добавяне на трета таблица, наречена "промеждутъчна таблица" или "таблица за асоциация", която съдържа два външни ключа - към съответните основни ключове. В прокета няма такава връзка.

## 2.3.2.3. CRUD заявки

**А) CRUD**

CRUD (Create, Read, Update, Delete) парадигмата е широко приложимо явление в уеб разработката, тъй като предоставя структуриран начин за създаване на функционалността на приложенията и изграждане на полезни и лесни за използване модели.

В контекста на CRUD, абревиатурата означава следното:

* **Create** (Създаване) операциите позволяват създаването на нови записи, като например нов акаунт или задача. В SQL бази данни, за създаване на записи се използва операцията INSERT.
* **Read** (Четене) операциите позволяват достъп до съществуващите записи чрез потребителския интерфейс. В релационни бази данни за избиране на записи се използва операцията SELECT.
* **Update** (Обновяване) операциите позволяват промяна на съществуващи записи, отново чрез потребителския интерфейс. В релационни бази данни за обновяване на записи се използва операцията UPDATE.
* **Delete** (Изтриване) операциите позволяват премахване на записи от базата данни. В SQL бази данни, за изтриване на записи се използва операцията DELETE.

**Б) Примерни CRUD заявки в проекта**

|  |  |  |
| --- | --- | --- |
| **Вид** | **Заявка** | **Описание** |
| INSERT | INSERT INTO users (f\_name, l\_name, email, password, birth, joined) VALUES ('$first\_name', '$last\_name', '$email', '$hashed\_password', '$birth\_date', NOW()) | Създаването на потребител след регистрация. Може да се види, че не всички полета получават данни от формата например pfp, admin, access, joined, които са автоматични |
| INSERT | INSERT INTO movies (title, genre, director, release\_date, duration, description, trailer, icon, age\_rating) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?) | Създаване на нов филм, който ще бъде автоматично неактивен. Админът го създава чрез бутон, слагащ автоматични временни стойности в полетата |
| SELECT | SELECT \* FROM movies WHERE active = 1 AND release\_date <= CURRENT\_DATE | Функционалността на менюто с филми, които са в продажба става чрез селекция на полетата „активно“ да е 1 и „премиерата“ да е по-малка или равна на днешна дата |
| SELECT | SELECT \* FROM movies WHERE genre LIKE '%Анимация%' AND active = 1 AND release\_date <= CURRENT\_DATE | Филтриране на менюто по жанр анимация |
| UPDATE | UPDATE ratings SET rating\_value = '$user\_rating' WHERE id\_user = '$id\_user' AND id\_movie = '$id\_movie' | Обновяване на рейтинга от потребител, който вече е имал такъв и си е променил мнението |
| UPDATE | UPDATE movies SET active = $status WHERE id\_movie = $id\_movie | Промяна на статуса на филм от админско меню дали е активен или не. $status е променлива, приемаща стойности 0 и 1 |
| DELETE | DELETE FROM ratings WHERE id\_user = $user\_id AND id\_movie = $movieId; | Триене на рейтинг на потребителя за някой филм. Това става чрез бутон, който той клика |
| DELETE | DELETE FROM users WHERE id\_user = ? | Триенето на потребител става само чрез тази заявка, поради автоматичните каскадни операции за всички таблици свързани с потребителя |

# Заключение

## Тестване

Приложението е тествано на няколко различни браузера, включително Chrome, Firefox и Edge. Това помага да се уверим, че функционалността е съвместима с различни браузери и версии. Тестването включва още проверка на приложението на различни устройства с различни разделителни способности на екрана. Това са лаптопи, компютри с различни резолюции и мобилни телефони. Целта е да се уверим, че интерфейсът е отзивчив и добре изглежда на различни устройства. Извършени са функционални тестове, за да се провери дали приложението работи правилно. Това включва тестване на входни форми, навигация, валидация на данни и други ключови функционалности. Стигнато е до извода, че приложението успешно функционира в различните среди.

## Постигнати цели

Всички поставени цели и изисквания са успешно постигнати. Дизайнът на приложението отговаря на очакванията за привлекателно и лесно за използване приложение. Може да се каже, че е също по-добър от други алтернативни сайтове. Менюто има повече опции за филтриране и всеки филм си има собствена страница с най-важната информация, представена най-удобно. Като цяло целият сайт изглежда прост и лесен за използване, но и хубав. Всички функции работят коректно, след като бяха изпробвани, тествани заедно с премахване на проблеми през разработването. За краен резултат, може да заключим, че приложението функционира оптимално и безпроблемно.

## Перспективи за развитие

Въпреки успешното реализиране на всички критерии, винаги едно приложение може да приеме много подобрения или нови функционалности. Например още по-добра защита, нови функции за улесняване на потребителите или редизайн на нещо недостатъчно перфектно. В случая може да се добавят или подобрят следните неща:

1. **Нотификация по имейл при премиера на филм** – значително подобрение на системата за уведомления. Това позволява на потребителите да бъдат информирани, когато нов филм е достъпен, без да е необходимо да посещават сайта, за да проверяват наличността на нови премиери.
2. **Нови видове кодове за промоция** – в момента тази система е доста проста. Може да се добавят нови видове кодове като предметни награди или храни и напитки за следващо посещение. Това ще доведе до промяна на таблицата от бази данни и най-вероятно на начина на генериране на кодове. Може и да се добави кой потребител е използвал кода.
3. **Различни цени на билети** – може да се добавят билети за различни възрасти. Това ще доведе до по-ниски и привлекателни цени за деца, ученици и възрастни. Също е възможно добавяне на автоматична отстъпка при голяма групова поръчка например 10 места.
4. **Коментари за филм** – това ще стимулира по-голямо взаимодействие между потребителите, давайки възможност за споделяне на мнения и впечатления за различните филми, освен само оценяване с рейтинг. Това ще доведе и до включването на имената на потребителите и профилните им снимки в коментарите, но ще се наложи да се обърне внимание на функционалността за качване на профилната снимка. Това е друга функционалност, която не се среща често в алтернативните сайтове.
5. **Промяна в системата за избор на място** – включва промяна към стандартизиран модел с редове и номера на седалките (например, от 1 до 15). Това би улеснило потребителите при избора на места, като въведе по-структуриран и лесно разбираем метод за идентификация на седалките, вместо да има различни индивидуални номера за всяка седалка, но ще доведе до много промени в backend функционалността.
6. **Проверка за валиден имейл** – въпреки че добавянето на проверка за валиден имейл може да удължи процеса на регистрация и да предизвика допълнителни стъпки за потребителите, това е важна функционалност, която ще подобри общата валидация и защита на уеб приложението от спам и нежелана активност.
7. **Забравена парола** – често потребителите могат да забравят своите пароли, и тази функционалност ще им предостави удобен начин да създадат нова парола и да възстановят достъпа до своя акаунт. Когато потребител забрави паролата си ще може да поиска промяна на паролата, като въведе своя имейл адрес. След това системата ще изпрати имейл с инструкции за генериране на нова.

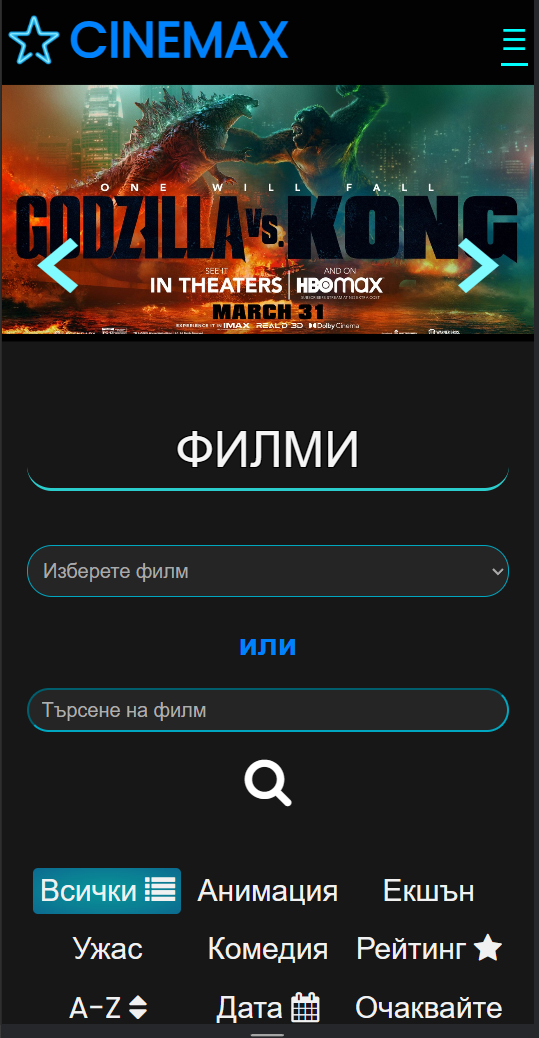
# Приложения



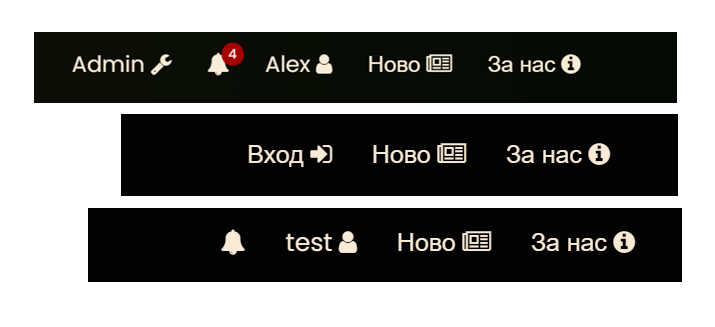
Фиг. 4: Проверки при регистрация



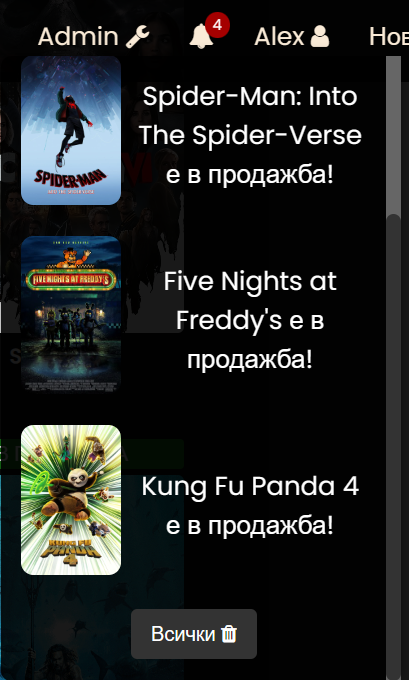
Фиг. 5: Имейлът след покупка



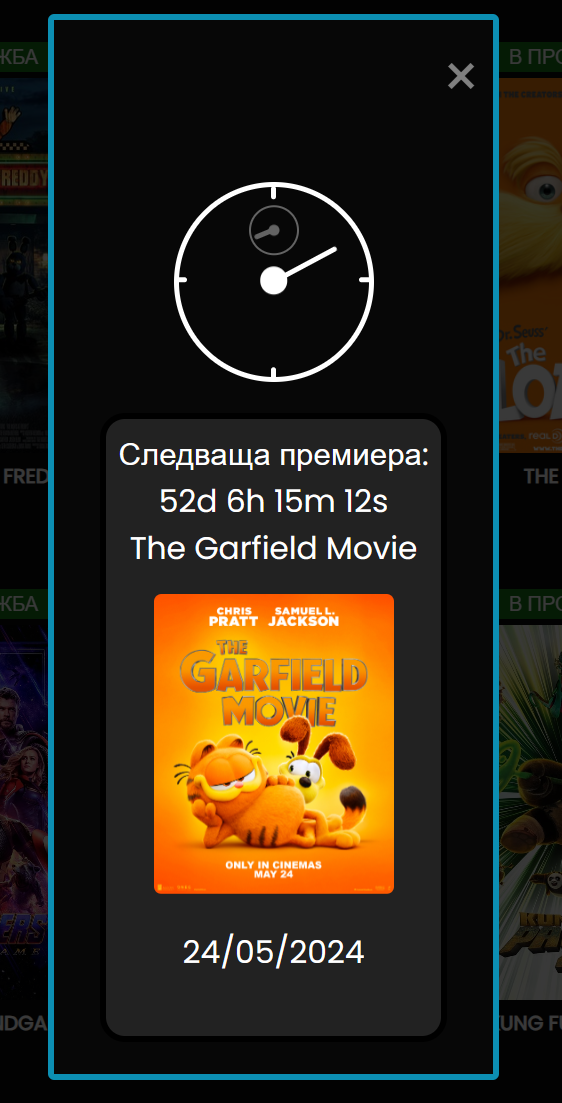
Фиг. 6: Мобилна версия на главната сраница



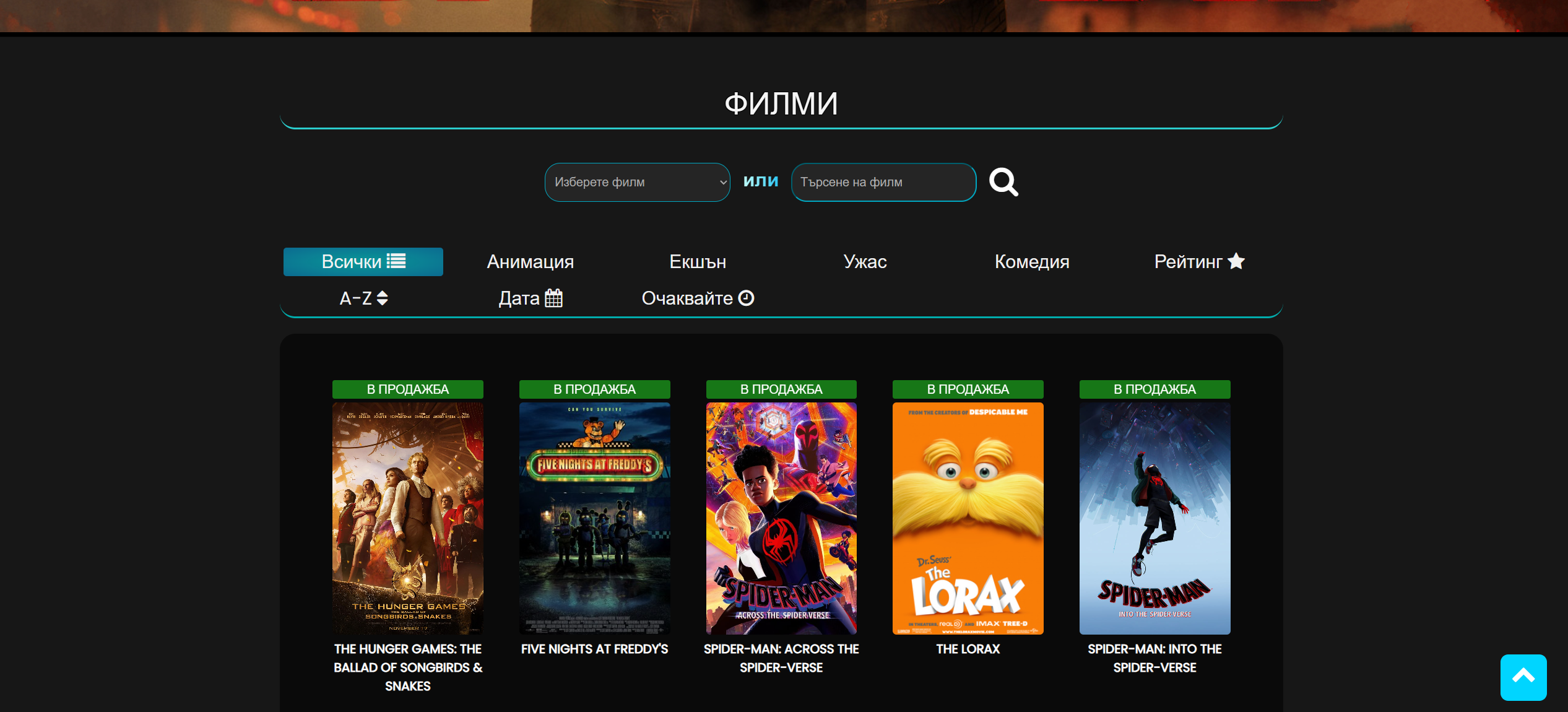
Фиг. 7: Навигационното меню според ролята (админ, гост, потребител)



Фиг. 8: Меню с известия



Фиг. 9: Popup за следваща премиера

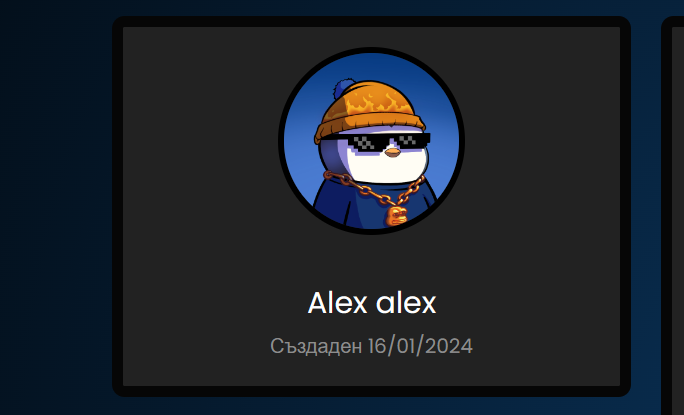


Фиг. 10: Менюто в начална страница

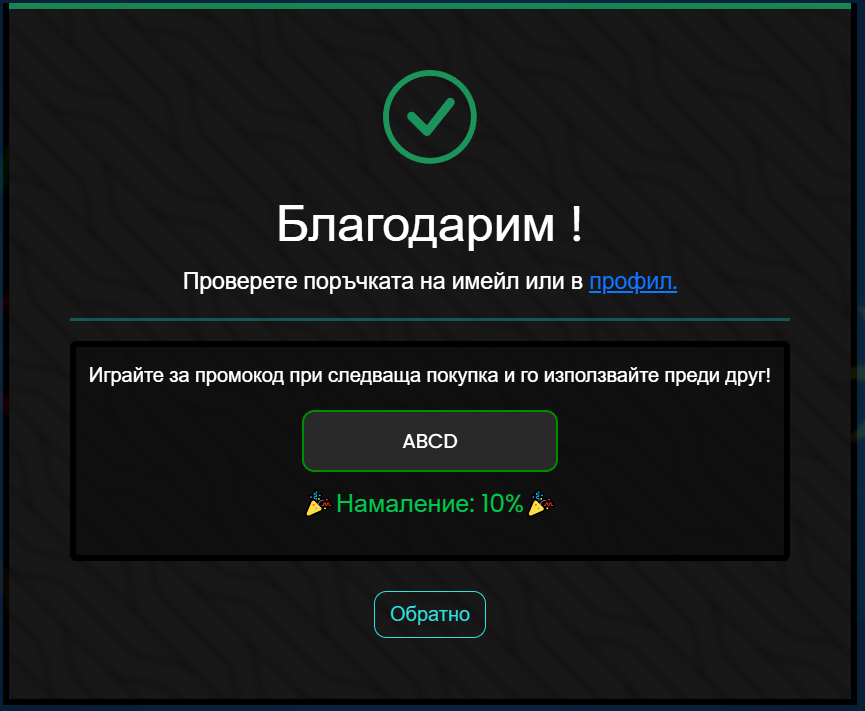




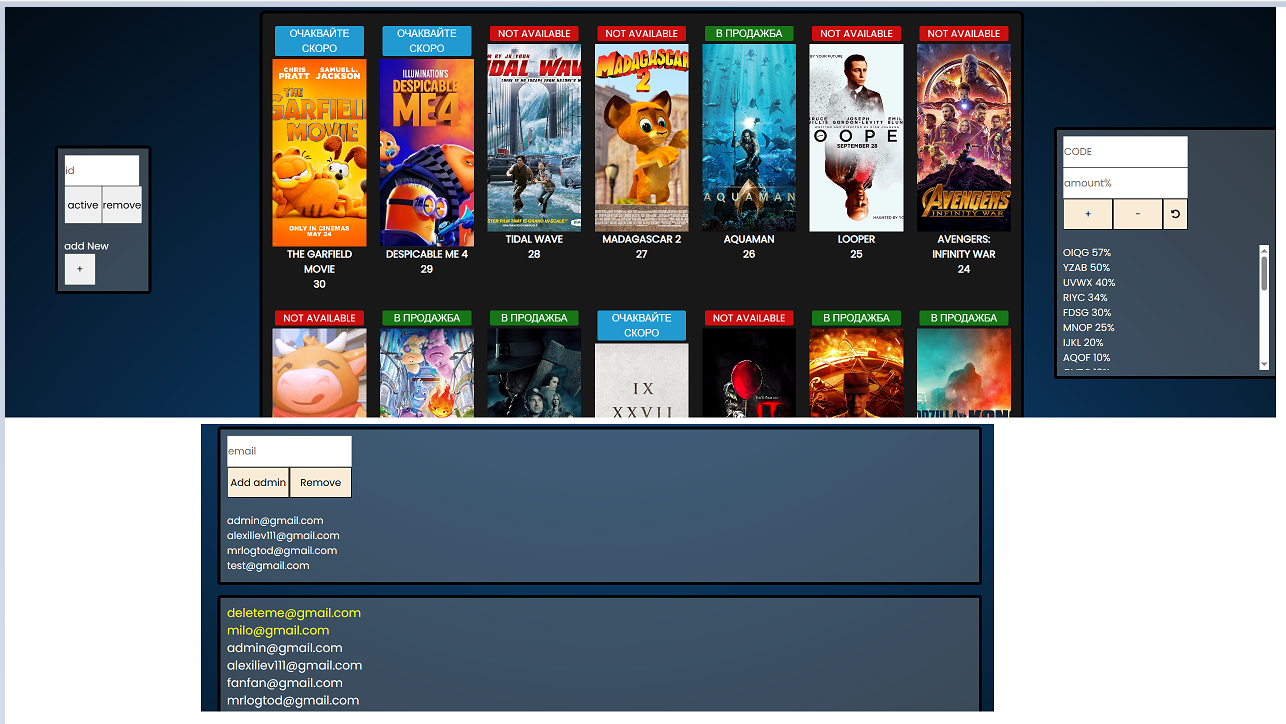
Фиг. 11: Рейтинг на филм и бутон за напомняне



Фиг. 12: Профилна снимка



Фиг. 13: Завършена покупка с печеливш код



Фиг. 14: Админска страница



Фиг. 15: Избор на места



Фиг. 16: Филмовата страница между два филма

# Използвани източници

[Bootstrap · The most popular HTML, CSS, and JS library in the world. (getbootstrap.com)](https://getbootstrap.com/)

[ChatGPT (openai.com)](https://chat.openai.com/)

[CSS Introduction - GeeksforGeeks](https://www.geeksforgeeks.org/css-introduction/)

[Font Awesome](https://fontawesome.com/)

[HTML basics - Learn web development | MDN (mozilla.org)](https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/HTML_basics)

[Overview of ASP.NET Core MVC | Microsoft Learn](https://learn.microsoft.com/en-us/aspnet/core/mvc/overview?view=aspnetcore-8.0)

[PHPMailer/PHPMailer: The classic email sending library for PHP (github.com)](https://github.com/PHPMailer/PHPMailer)

[Relationships in SQL - Complete Guide With Examples - Devart Blog](https://blog.devart.com/types-of-relationships-in-sql-server-database.html)

[YouTube](https://www.youtube.com/)