МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА 44

КУРСОВАЯ РАБОТА (ПРОЕКТ)   
ЗАЩИЩЕНА С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| старший преподаватель |  |  |  | А.В.Аксенов |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОЙ РАБОТЕ (ПРОЕКТУ) |
| САЙТ КИНОТЕАТРА. |
| по дисциплине: Базы данных |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4641 |  |  |  | А.А.Антонов |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2019

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА 44

РУКОВОДИТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| старший преподаватель |  |  |  | А.В.Аксенов |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ТЕХНИЧЕСКОЕ ЗАДАНИЕ |
| САЙТ КИНОТЕАТРА. |
| по дисциплине: Базы данных |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4641 |  |  |  | А.А.Антонов |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2019

Цель курсового проекта: закрепление теоретических и практических знаний, полученных во время лекционных, лабораторных и самостоятельных занятий.   
В ходе данной работы будет разработан сайт кинотеатра. Он будет состоять из следующих страниц:

1. Главная страница. На ней слева будет расположено вертикальное меню, через которое можно будет зарегистрироваться, авторизоваться или выйти с аккаунта. А также перейти на вкладку расписание сеансов и киноафишу.
2. Киноафиша. Здесь будет располагаться информация о фильмах, которые идут в кинотеатре в этот период. У каждого фильма будет присутствовать название, картинка и краткое описание. Для получение полного описания фильма, нужно нажать на название фильма, которое является гиперссылкой.
3. Расписание сеансов. При переходе в данный раздел, пользователю будет предложено выбрать дату. После выбора даты будут показаны все фильмы на данный день и время их показа. При нажатии на время сеанса выбранного фильма, пользователь будет направлен в меню выбора мест.
4. Меню выбора мест. В данном разделе будет развёрнута карта мест зала, выкупленные места будут отмечены синим цветом, а забронированные желтым. Пользователь может выбрать незанятое место и нажать купить или забронировать.

В зависимости от условий страницы могу выглядеть по-разному. Если до начала сеанса меньше или равно часу, то не будет возможности забронировать билет и вся бронь сбрасывается, а если меньше 30 минут, то пропадает функция купить билет онлайн. Пользователь не может перейти в меню выбора мест пока не авторизуется.

Работа будет выполнена на языке программирования Python с использованием фреймворка Django. После завершения, сайт будет развернут на Heroku.

**1.Реляционная схема разработанной БД**

Все данные кинотеатра будут храниться в базе данных. Воспользуемся реляционной схемой для ее проектирования. Реляционная схема изображена на рисунке 1.

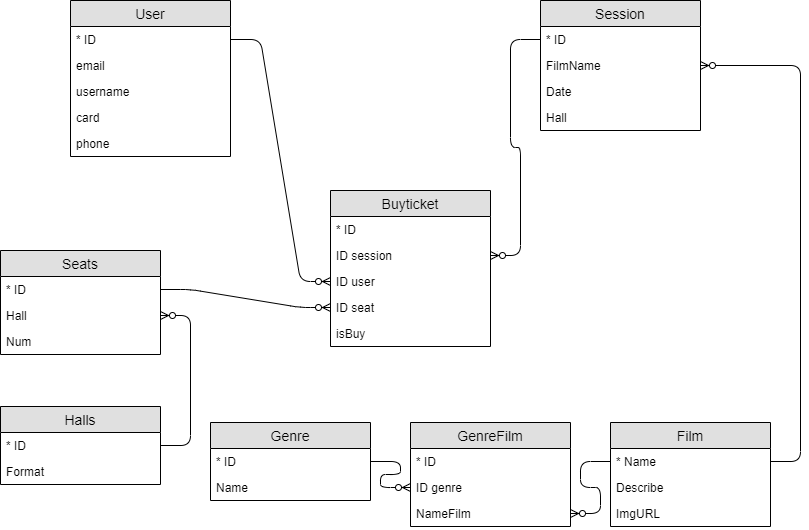


Рисунок 1 Реляционная схема

**2.** **Описание выбранных технологий реализации.**

Для разработки веб-сайта кинотеатра необходимы следующие дополнительные инструментальные программные средства:

— язык программирования Python и его фреймворк Django,

—облачная платформа Heroku

— система управления базами данных PostgreSQL.

Django — свободный фреймворк для веб-приложений на языке Python, использующий шаблон проектирования MVC. Он содержит огромное количество функциональности для решения большинства задач веб-разработки. Сайт на Django строится из одного или нескольких приложений, которые рекомендуется делать отчуждаемыми и подключаемыми. Это одно из существенных архитектурных отличий этого фреймворка от некоторых других.

Некоторые возможности Django:

1. встроенный интерфейс администратора, с уже имеющимися переводами на многие языки
2. диспетчер URL на основе регулярных выражений
3. расширяемая система шаблонов с тегами и наследованием[[17]](https://ru.wikipedia.org/wiki/Django#cite_note-17)
4. система кеширования
5. интернационализация
6. подключаемая архитектура приложений, которые можно устанавливать на любые Django-сайты
7. «generic views» — шаблоны функций контроллеров
8. авторизация и аутентификация, подключение внешних модулей аутентификации: LDAP, OpenID и проч.
9. система фильтров («middleware») для построения дополнительных обработчиков запросов, как например включённые в дистрибутив фильтры для кеширования, сжатия, нормализации URL и поддержки анонимных сессий
10. библиотека для работы с формами (наследование, построение форм по существующей модели БД)
11. встроенная автоматическая документация по тегам шаблонов и моделям данных, доступная через административное приложение

PostgreSQL - это свободно распространяемая объектно-реляционная система управления базами данных (ORDBMS), наиболее развитая из открытых СУБД в мире. PostgreSQL базируется на языке SQL и поддерживает многие из возможностей стандарта SQL:2011.

Сильными сторонами PostgreSQL считаются:

1. высокопроизводительные и надёжные механизмы [транзакций](https://ru.wikipedia.org/wiki/%D0%A2%D1%80%D0%B0%D0%BD%D0%B7%D0%B0%D0%BA%D1%86%D0%B8%D1%8F_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) и [репликации](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BF%D0%BB%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F_(%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%82%D0%B5%D1%85%D0%BD%D0%B8%D0%BA%D0%B0));
2. расширяемая система встроенных языков программирования: в стандартной поставке поддерживаются [PL/pgSQL](https://ru.wikipedia.org/wiki/PL/pgSQL), [PL/Perl](https://ru.wikipedia.org/wiki/PL/Perl), [PL/Python](https://ru.wikipedia.org/wiki/PL/Python) и [PL/Tcl](https://ru.wikipedia.org/wiki/PL/Tcl); дополнительно можно использовать [PL/Java](https://ru.wikipedia.org/w/index.php?title=PL/Java&action=edit&redlink=1), [PL/PHP](https://ru.wikipedia.org/w/index.php?title=PL/PHP&action=edit&redlink=1), [PL/Py](https://ru.wikipedia.org/w/index.php?title=PL/Py&action=edit&redlink=1), [PL/R](https://ru.wikipedia.org/w/index.php?title=PL/R&action=edit&redlink=1), [PL/Ruby](https://ru.wikipedia.org/w/index.php?title=PL/Ruby&action=edit&redlink=1), [PL/Scheme](https://ru.wikipedia.org/wiki/PL/Scheme), [PL/sh](https://ru.wikipedia.org/w/index.php?title=PL/sh&action=edit&redlink=1) и [PL/V8](https://ru.wikipedia.org/w/index.php?title=PL/V8&action=edit&redlink=1), а также имеется поддержка загрузки модулей расширения на языке [C](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8_(%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F));
3. [наследование](https://ru.wikipedia.org/wiki/%D0%9D%D0%B0%D1%81%D0%BB%D0%B5%D0%B4%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5));
4. возможность индексирования геометрических объектов и наличие базирующегося на ней расширения [PostGIS](https://ru.wikipedia.org/wiki/PostGIS" \o "PostGIS);
5. встроенная поддержка [слабоструктурированных данных](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D0%B0%D0%B1%D0%BE%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D0%B5_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D0%B5) в формате [JSON](https://ru.wikipedia.org/wiki/JSON) с возможностью их индексации;
6. расширяемость (возможность создавать новые типы данных, типы индексов, языки программирования, модули расширения, подключать любые внешние источники данных).

Heroku — облачная PaaS-платформа, поддерживающая ряд языков программирования. Platform as a Service (PaaS, «платформа как услуга») — модель предоставления облачных вычислений, при которой потребитель получает доступ к использованию информационно-технологических платформ: операционных систем, систем управления базами данных, связующему программному обеспечению, средствам разработки и тестирования, размещённым у облачного провайдера. В этой модели вся информационно-технологическая инфраструктура, включая вычислительные сети, серверы, системы хранения, целиком управляется провайдером, провайдером же определяется набор доступных для потребителей видов платформ и набор управляемых параметров платформ, а потребителю предоставляется возможность использовать платформы, создавать их виртуальные экземпляры, устанавливать, разрабатывать, тестировать, эксплуатировать на них прикладное программное обеспечение, при этом динамически изменяя количество потребляемых вычислительных ресурсов.

**3.Описание используемых методов взаимодействия с БД.**

База данных подключается в файле settings.py нашего проекта. Для подключения необходимо указать название СУБД, имя пользователя, пароль, название базы данных и ip адрес. Связь проекта с данными осуществляется посредством моделей. Модели в Django описывают структуру используемых данных. Для того, чтобы получить модели уже созданных таблиц в БД, воспользуемся утилитой inspectdb. Утилита inspectdb просматривает базу данных, указанную в нашем конфигурационном файле, определяет представление модели для каждой таблицы и выводит код модели на стандартный вывод.

**4. Описание архитектуры приложения**

Весь проект будет состоять из 4 приложений.

1. ManinDb – это приложение в моделях которого будут храниться, уже созданные таблицы нашей БД.
2. Accounts. В нем будут происходить такие процессы, как регистрация, авторизация и выход с аккаунта.
3. Listfilms. В этом приложении обработка страниц связных с выводом киноафиш, это и список фильмов на неделю и каждый фильм по-отдельности.
4. Listsession. Последнее приложение отвечает за покупку билетов и просмотр сеансов. Он отображает календарь, список сеансов на выбранный день, схему зала с отмеченными купленными или забронированными билетами.
5. Cinemamax. Главное приложение, которое создается вместе с проектом. В нем происходит обработка главной страницы. Одной из главных его функций является отображения билетов для каждого пользователя в меню мои билеты.

Все html документы храниться в папке templates. Все страницы наследуются от главного шаблона base. Файлы css и js храниться в static.

**5.Листинги модулей приложения с комментариями**

1)Cinemamax/views.py

def index(request):

if request.method == "POST":

typetick=request.POST.get("type")

print(typetick,type(typetick))

if(typetick=="1"):

typetick=True

else:

typetick=False

print("Куплено или",typetick)

sid=Buyticket.objects.last().buyid+1

un=request.user.username

uid=Users.objects.get(uname=un).user\_id

numses=request.POST.get("nums")

zal=request.POST.get("zal")

lasttick=Buyticket.objects.last().sessionid

lasttick.idsession=numses

lastseat=Buyticket.objects.last().seatid

lastuser=Buyticket.objects.last().userid

lastuser.user\_id=uid

listtick=[]

listtick= request.POST.get("listticketbuy").split(",")

for i in range(len(listtick)-1):

sed=Seats.objects.get(hallid=zal,num=listtick[i]).idseats

lastseat.idseats=sed

ticket=Buyticket(buyid=sid,sessionid=lasttick,seatid=lastseat,

userid=lastuser,isbuy=typetick)

ticket.save()

sid+=1

if request.user.is\_authenticated:

return render(request,"baseauth.html")

else:

return render(request,"base.html")

2) accounts/views.py

def signup(request):

if request.method == 'POST':

user\_form = UserRegistrationForm(request.POST)

if user\_form.is\_valid():

new\_user = user\_form.save(commit=False)

new\_user.set\_password(user\_form.cleaned\_data['password'])

unam=user\_form.cleaned\_data.get('username')

uid=Users.objects.last().user\_id+1

umail=user\_form.cleaned\_data.get('email')

ucard=user\_form.cleaned\_data.get('card')

uphone=user\_form.cleaned\_data.get('phone')

u=Users(user\_id=uid,uname=unam,email=umail,card=ucard,phone=uphone)

u.save()

new\_user.save()

return render(request, 'registration/register\_done.html', {'new\_user': new\_user})

else:

user\_form = UserRegistrationForm()

return render(request, 'registration/signup.html', {'user\_form': user\_form})

3)listfils/views.py

def index(request):

films=Film.objects.all()

return render(request,"filmlist/index.html",{"film\_lst":films})

def onefilm(request,filmid):

film=Film.objects.get(namefilm=filmid)

print(film.namefilm)

return render(request,"filmlist/kino.html",{"film":film})

4)listsession/views.py

def index(request):

if request.method == "POST":

listtick = request.POST.get("List")

return render(request,"seslist/calendar.html",{"datenow":datetime.date.today()})

else:

ourday=request.GET.get("calendar")

if ourday!=None:

beginourday=request.GET.get("calendar")+" 00:00:00.0"

endourday=request.GET.get("calendar")+" 23:59:59.0"

#ourday=request.GET.get("calendar")

timeBegin=datetime.datetime.strptime(beginourday,'%Y-%m-%d %H:%M:%S.%f')

timeEnd=datetime.datetime.strptime(endourday,'%Y-%m-%d %H:%M:%S.%f')

sestoday=Sessions.objects.filter(datasession\_\_gte=timeBegin,datasession\_\_lte=timeEnd).order\_by('fillname')

cnt=Sessions.objects.filter(datasession\_\_gte=timeBegin,datasession\_\_lte=timeEnd).order\_by('fillname').values('fillname').annotate(dcount=Count('fillname')).count()

sesgroup=Sessions.objects.filter(datasession\_\_gte=timeBegin,datasession\_\_lte=timeEnd).distinct('fillname')

listses=[]

listnamefilm=[]

listimg=[]

for i in range(cnt):

listses.append(Sessions.objects.select\_related().filter(datasession\_\_gte=timeBegin,datasession\_\_lte=timeEnd,fillname=sesgroup[i].fillname))

print(listses[i][0].fillname.imgurl)

for i in range(cnt):

listimg.append(Film.objects.get(namefilm=listses[i][0].fillname.namefilm).imgurl)

return render(request,"seslist/rasp.html",{"listses":listses,"imglist":listimg})

#cur\_user=request.user

#if (cur\_user.is\_authenticated):

#print("Hellow user",request.user.username)

#return render(request,"seslist/calendar.html")

else:

x=Sessions.objects.select\_related ('fillname')

for s in x:

print(s.fillname.imgurl)

return render(request,"seslist/calendar.html",{"datenow":datetime.datetime.now().strftime("%Y-%m-%d")})

def printzal(request,idzal,numses):

if request.user.is\_authenticated:

if((Users.objects.get(uname=request.user.username).card)is None):

isbuy=0

else:

isbuy=1

datetimeses=Sessions.objects.get(idsession=numses).datasession

isres=1

endbuy=1

print(datetime.datetime.now())

print(datetimeses)

deltatime=(datetimeses-datetime.datetime.now()).seconds/60

prosh=(datetime.datetime.now()-datetimeses).seconds/60

if (abs(deltatime)<=60 and datetime.datetime.now().day==datetimeses.day and datetime.datetime.now().month==datetimeses.month ):

Buyticket.objects.filter(sessionid=numses,isbuy=False).delete()

isres=0

if (abs(deltatime)<=30 and datetime.datetime.now().day==datetimeses.day and datetime.datetime.now().month==datetimeses.month):

endbuy=0

if(datetime.datetime.now().day>datetimeses.day and datetime.datetime.now().month==datetimeses.month or ((datetime.datetime.now()-datetimeses).seconds/60 >0) and (datetime.datetime.now().day==datetimeses.day)):

isres=0

endbuy=0

Select\_numses=Buyticket.objects.filter(sessionid=numses,isbuy=True)

Select\_numsesreserv=Buyticket.objects.filter(sessionid=numses,isbuy=False)

reslist=[]

buylist=[]

sizelist=0

sizeres=0

for i in Select\_numses:

buylist.append(Seats.objects.get(idseats=i.seatid.idseats).num)

sizelist+=1

for i in Select\_numsesreserv:

reslist.append(Seats.objects.get(idseats=i.seatid.idseats).num)

sizeres+=1

#return render(request,zal,{"buylist":buylist,"sizelist":sizelist,"numses":numses,"zal":idzal,"reslist":reslist,"sizeres":sizeres})

return render(request,"seslist/zal1.html",{"buylist":buylist,"sizelist":sizelist,"numses":numses,"zal":idzal,"reslist":reslist,"sizeres":sizeres,"isb":isbuy,"isress":isres,"endbuy":endbuy})

else:

return render(request,"seslist/logged\_out.html")

def endbuy(request):

return render(request,"seslist/endbuy.html")

**6.Описание применения приложения со снимками экрана**

В зависимости от того авторизирован ли пользователь будет отображаться разное боковое меню.



Рисунок 2 Главная страница авторизированного пользователя

Если пользователь перейдет на страницу киноафиши ему отобразиться фильмы, которые идут в кинотеатре.

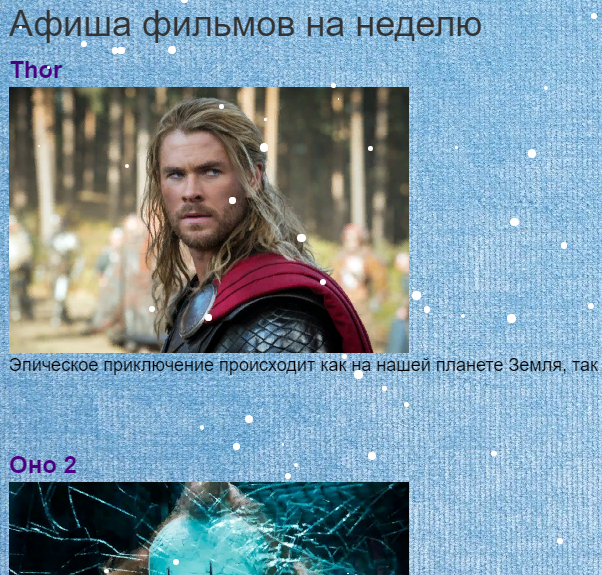


Рисунок 3 Страница киноафиши

Для просмотра расписания фильмов пользователь должен выбрать интересующий его день, а после отобразиться сеансы.

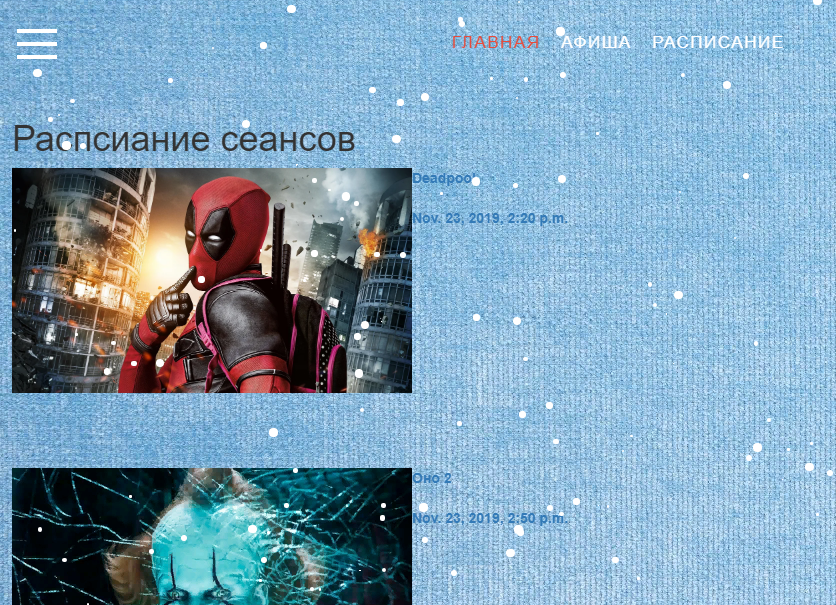


Рисунок 4 Сеансы на 23 ноября 2019 г.

После того как пользователь выберет сеанс, ему отобразиться схема зала, на которой отмечены купленные и забронированные места, и он может выбрать интересующие его места.

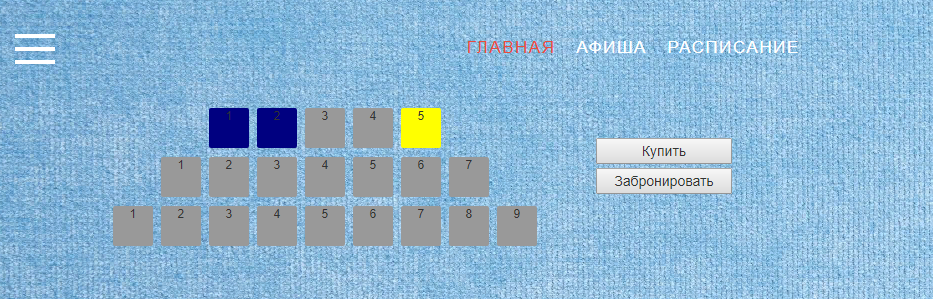


Рисунок 5 Схема 2 зала.

Все купленные и забронированные пользователем билеты отображаются в меню Мои билеты.

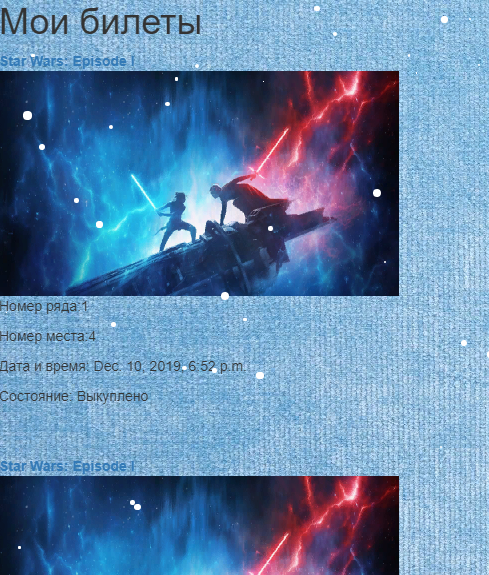


Рисунок 6 Мои билеты

**6.Вывод**

В ходе данной работы был разработан сайт кинотеатра. Войдя на сайт, пользователь может зарегистрироваться, авторизоваться, просмотреть сеансы на текущую неделю, а также купить или забронировать места на желаемый фильм. После покупки пользователь может проверить купленные или забронированные им билеты в меню Мои билеты. Так же имеется возможность посмотреть какие фильмы недавно показывали в кинотеатре. При выполнении курсовой работы были закреплены теоретические и практические знания, полученных во время лекционных, лабораторных и самостоятельных занятий. 