МГТУ им. Н.Э. Баумана

Факультет «Информатика и системы управления»

ДИСЦИПЛИНА:

«Разработка интернет-приложений»

Отчет по домашней работе

«Разработка веб-приложения»

Выполнил:

Студент 3 курса

Факультет ИУ

Группа ИУ5-51Б

Фонканц Р.В.

Преподаватель:

Гапанюк Ю.Е.

**Задание**

В рамках домашнего задания студент разрабатывает веб-приложение на основе технологий Python, Django, JavaScript, MySQL.

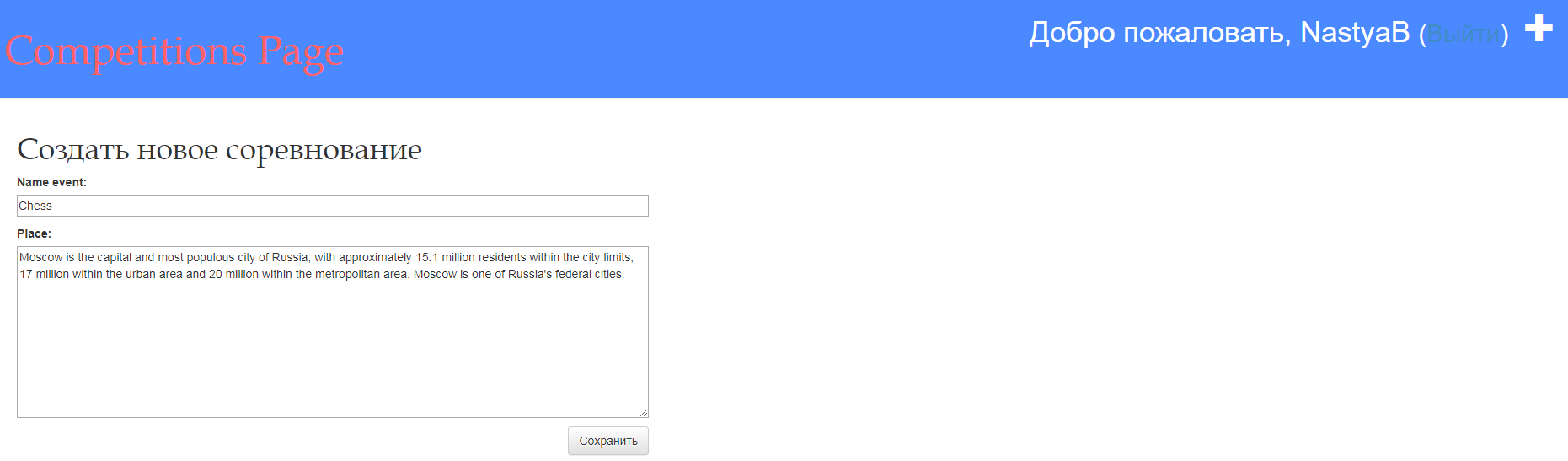
Предметная область домашнего задания содержит субъект, отношение и объект. Необходимые атрибуты субъекта и объекта выбираются студентом самостоятельно.

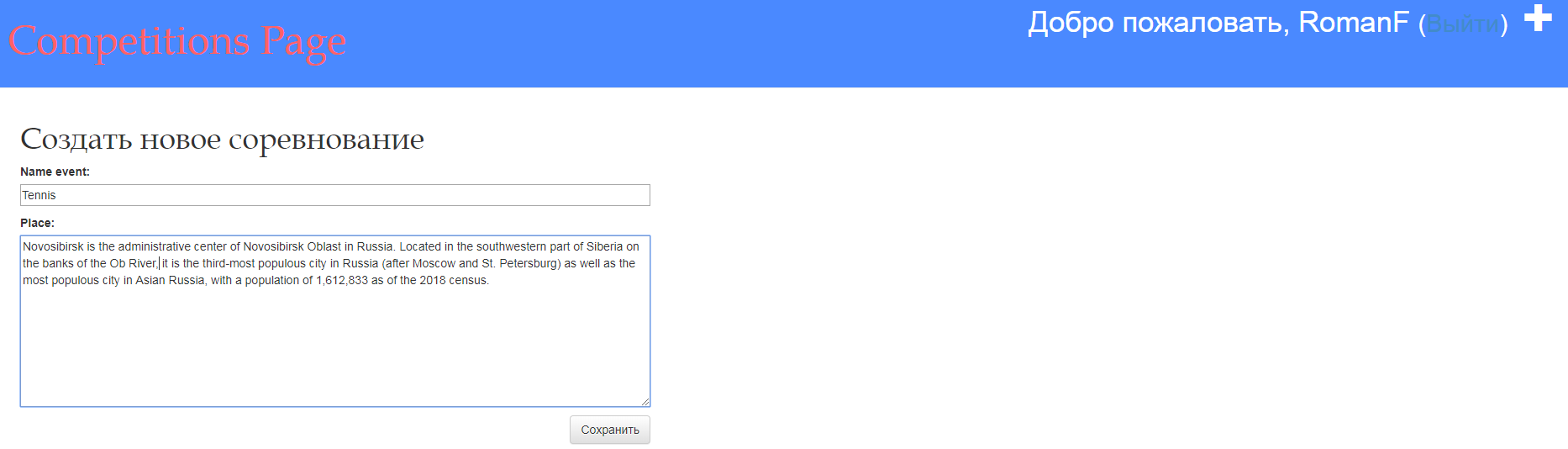
**Скриншоты страниц разработанного веб-приложения**

Страница логина:

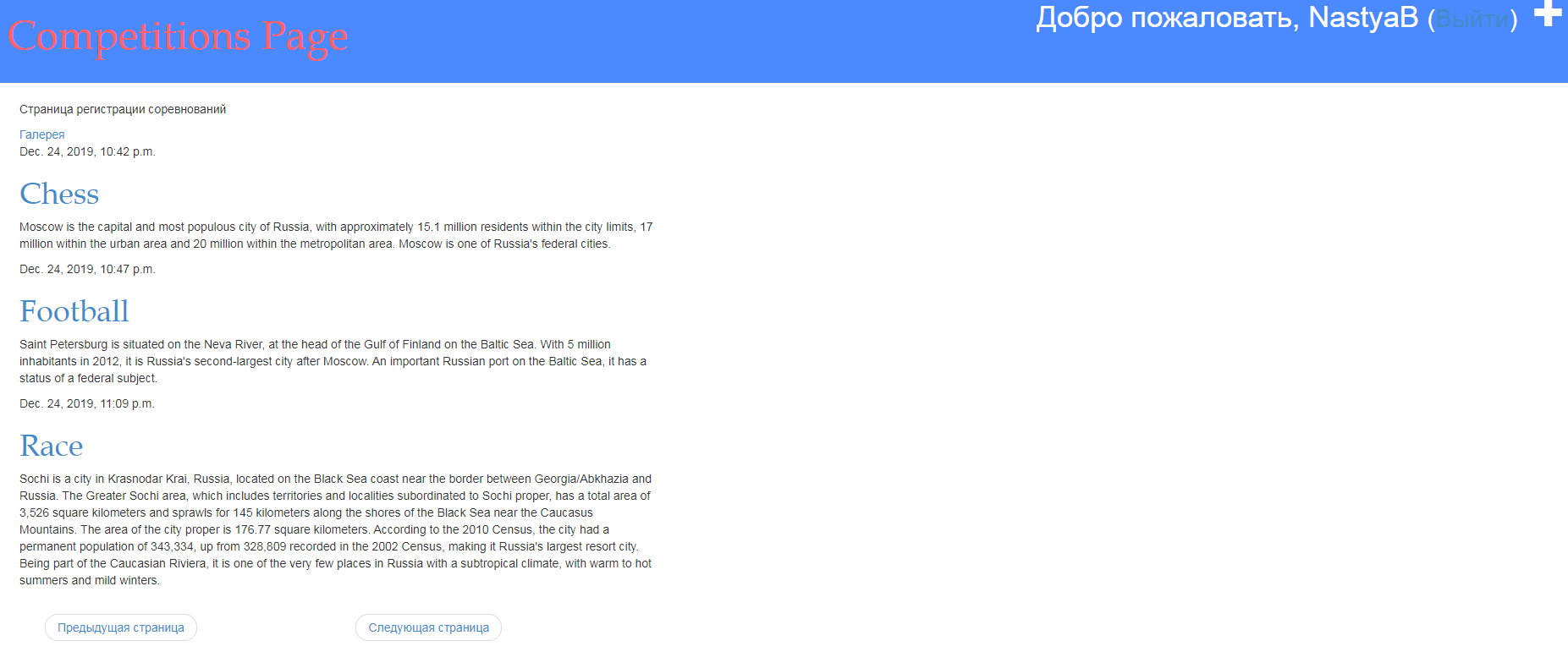


Страница создания новых соревнований:



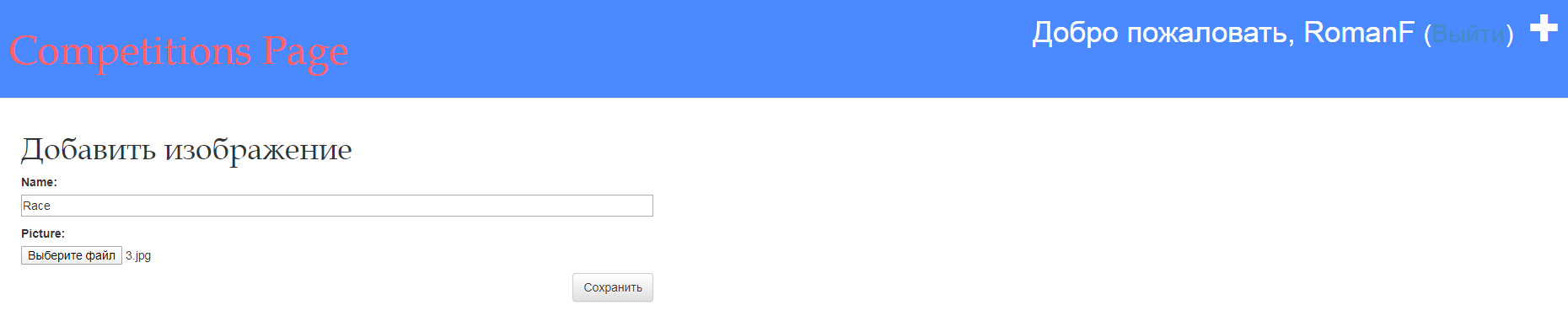


Страницы списка действующих соревнований:

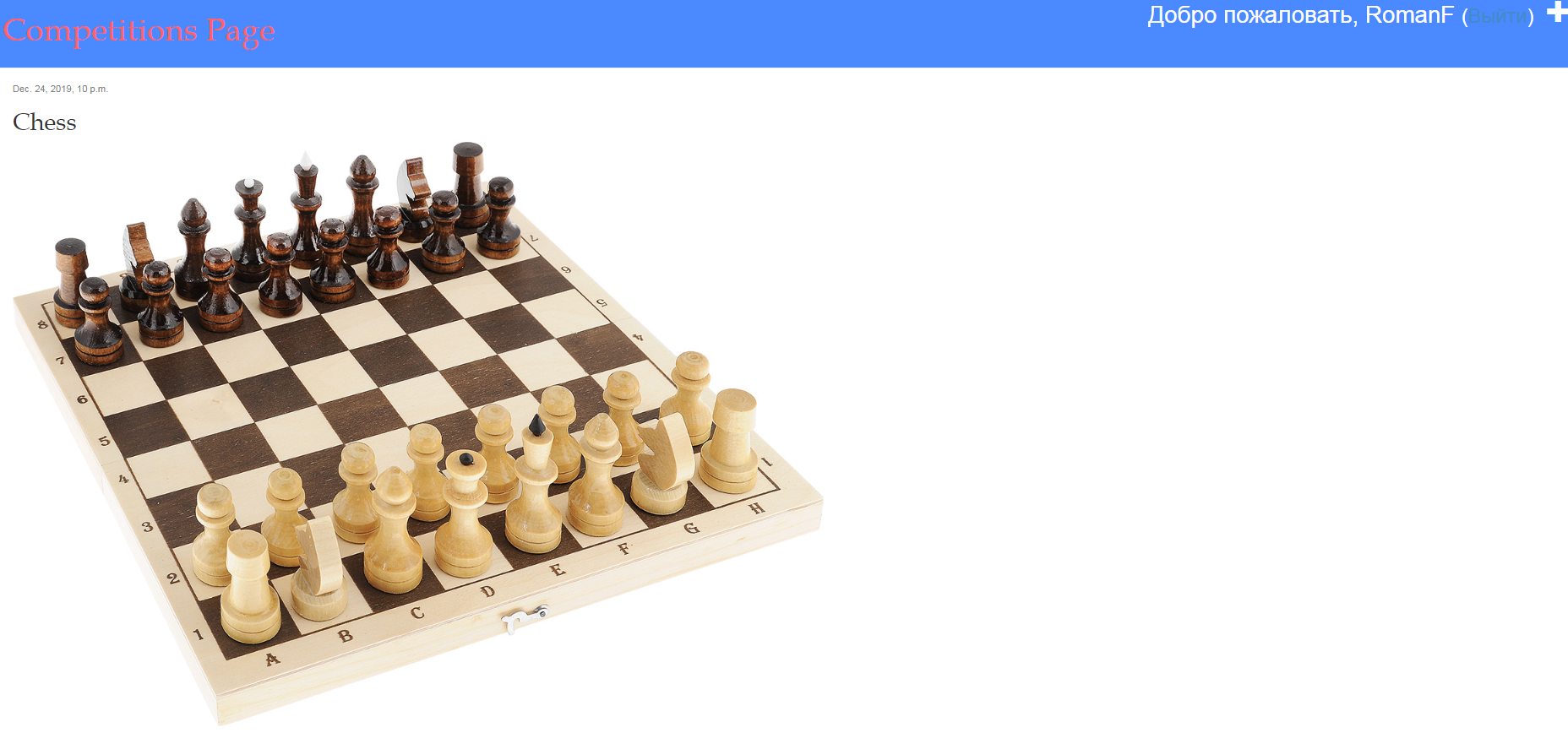


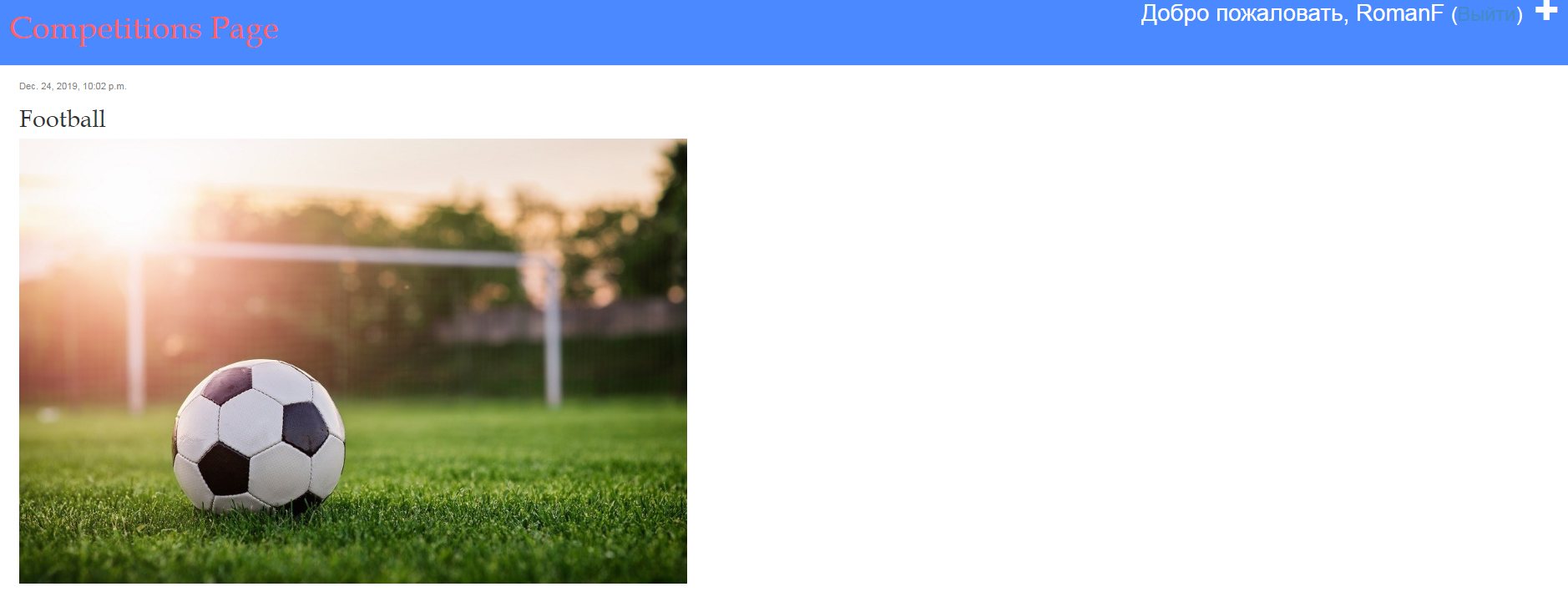


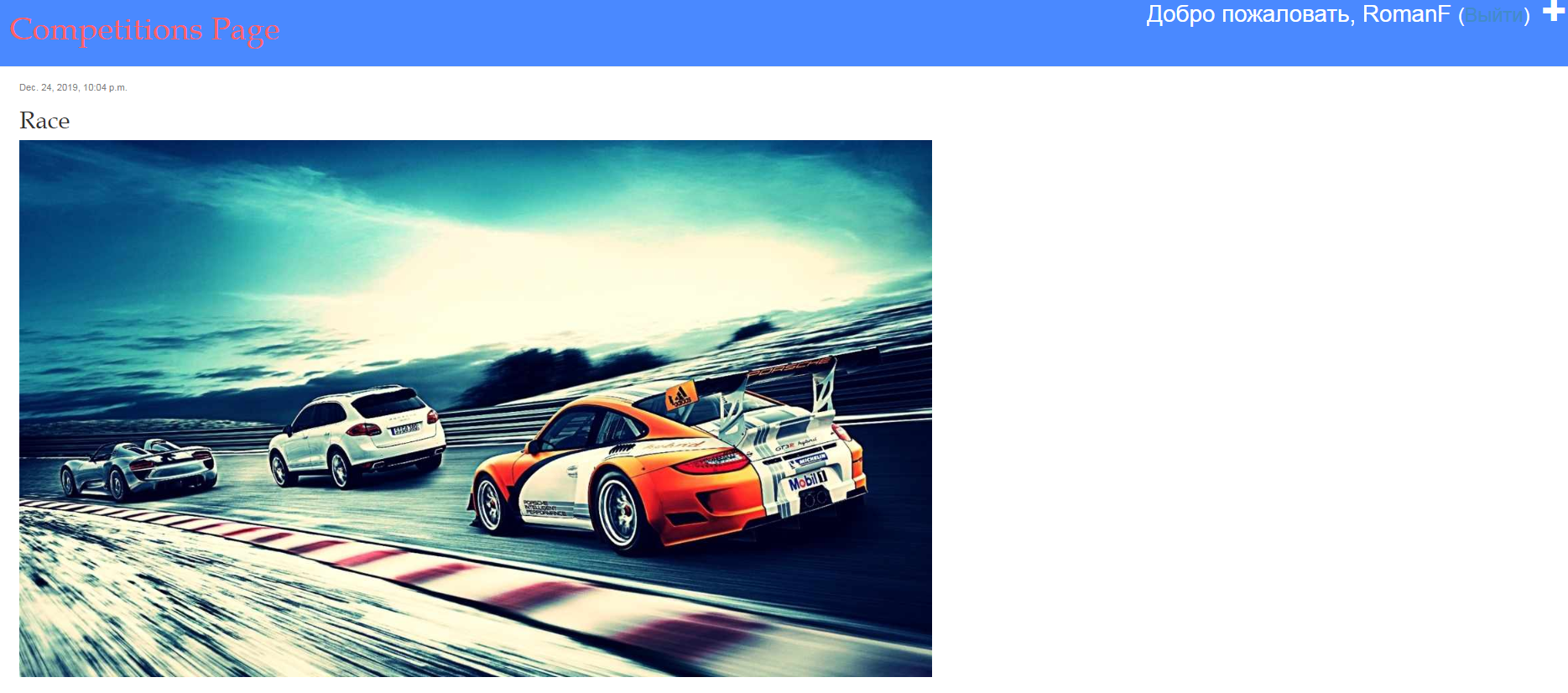
Страница добавления изображения соревнований:



Страницы с изображениями соревнований:







**Исходный код**

Файл models.py:

from django.db import models  
from django.utils import timezone  
from django.contrib.auth.models import AbstractUser  
  
  
  
class Event(models.Model):  
 player = models.ForeignKey('auth.User', on\_delete=models.CASCADE)  
 name\_event = models.CharField(max\_length=200)  
 place = models.TextField()  
 reg\_date = models.DateTimeField(default=timezone.now)  
 event\_date = models.DateTimeField(blank=True, null=True)  
  
 def participation(self):  
 self.event\_date = timezone.now()  
 self.save()  
  
 def \_\_str\_\_(self):  
 return self.name\_event  
  
  
class Picture(models.Model):  
 name = models.CharField(max\_length=200)  
 published\_date = models.DateTimeField(blank=True, null=True)  
 picture = models.ImageField(upload\_to='image', null=True, blank=True)  
  
 def publish(self):  
 self.published\_date = timezone.now()  
 self.save()  
  
 def \_\_str\_\_(self):  
 return self.name

Файл forms.py:

from django import forms  
from .models import Event, Picture  
  
  
class EventForm(forms.ModelForm):  
 class Meta:  
 model = Event  
 fields = ('name\_event', 'place')  
  
  
class PictureForm(forms.ModelForm):  
 class Meta:  
 model = Picture  
 fields = ('name', 'picture')

Файл views.py:

from .models import Event, Picture  
from django.utils import timezone  
from django.shortcuts import render, get\_object\_or\_404  
from .forms import EventForm, PictureForm  
from django.shortcuts import redirect  
from django.contrib.auth.decorators import login\_required  
from django.core.paginator import Paginator, PageNotAnInteger, EmptyPage  
  
  
def event\_list(request):  
 queryset\_list = Event.objects.all().order\_by("id")  
 paginator = Paginator(queryset\_list, 3) # posts per page  
 page = request.GET.get('page')  
 try:  
 queryset = paginator.page(page)  
 except PageNotAnInteger:  
 queryset = paginator.page(1)  
 except EmptyPage:  
 queryset = paginator.page(paginator.num\_pages)  
  
 context = {  
 "object\_list": queryset,  
 }  
 return render(request, 'page/event\_list.html', context)  
  
  
def event\_detail(request, pk):  
 event = get\_object\_or\_404(Event, pk=pk)  
 return render(request, 'page/event\_detail.html', {'event': event})  
  
  
@login\_required  
def event\_new(request):  
 if request.method == "POST":  
 form = EventForm(request.POST)  
 if form.is\_valid():  
 event = form.save(commit=False)  
 event.player = request.user  
 event.event\_date = timezone.now()  
 event.save()  
 return redirect('event\_detail', pk=event.pk)  
 else:  
 form = EventForm()  
 return render(request, 'page/event\_edit.html', {'form': form})  
  
  
@login\_required  
def event\_edit(request, pk):  
 post = get\_object\_or\_404(Event, pk=pk)  
 if request.method == "POST":  
 form = EventForm(request.POST, instance=post)  
 if form.is\_valid():  
 event = form.save(commit=False)  
 event.player = request.user  
 event.event\_date = timezone.now()  
 event.save()  
 return redirect('event\_detail', pk=event.pk)  
 else:  
 form = EventForm(instance=post)  
 return render(request, 'page/event\_edit.html', {'form': form})  
  
  
def picture\_list(request):  
 queryset\_list = Picture.objects.all().order\_by("id")  
 paginator = Paginator(queryset\_list, 3) # posts per page  
 page = request.GET.get('page')  
 try:  
 queryset = paginator.page(page)  
 except PageNotAnInteger:  
 queryset = paginator.page(1)  
 except EmptyPage:  
 queryset = paginator.page(paginator.num\_pages)  
  
 context = {  
 "object\_list": queryset,  
 }  
 return render(request, 'page/picture\_list.html', context)  
  
  
def picture\_detail(request, pk):  
 picture = get\_object\_or\_404(Picture, pk=pk)  
 return render(request, 'page/picture\_detail.html', {'picture': picture})  
  
  
@login\_required  
def picture\_new(request):  
 if request.method == "POST":  
 form = PictureForm(request.POST)  
 if form.is\_valid():  
 picture = form.save(commit=False)  
 picture.published\_date = timezone.now()  
 picture.save()  
 return redirect('picture\_detail', pk=picture.pk)  
 else:  
 form = PictureForm()  
 return render(request, 'page/picture\_edit.html', {'form': form})  
  
  
@login\_required  
def picture\_edit(request, pk):  
 picture = get\_object\_or\_404(Picture, pk=pk)  
 if request.method == "POST":  
 form = PictureForm(request.POST, instance=picture)  
 if form.is\_valid():  
 picture = form.save(commit=False)  
 picture.published\_date = timezone.now()  
 picture.save()  
 return redirect('event\_detail', pk=picture.pk)  
 else:  
 form = PictureForm(instance=picture)  
 return render(request, 'page/picture\_edit.html', {'form': form})

Файлы urls.py:

from django.contrib import admin  
from django.urls import path  
from django.conf.urls import include, url  
from django.contrib.auth import views  
  
  
urlpatterns = [  
 path('admin/', admin.site.urls),  
 path('accounts/login/', views.LoginView.as\_view(), name='login'),  
 path('accounts/logout/', views.LogoutView.as\_view(), name='logout'),  
 url(r'', include('DZ.urls'))  
]

Файлы DZ/urls.py:

from django.conf.urls import url  
from . import views  
from django.conf import settings  
from django.conf.urls.static import static  
  
urlpatterns = [  
 url(r'^$', views.post\_list, name='post\_list'),  
 url(r'^post/(?P<pk>\d+)/$', views.post\_detail, name='post\_detail'),  
 url(r'^post/new/$', views.post\_new, name='post\_new'),  
 url(r'^post/(?P<pk>\d+)/edit/$', views.post\_edit, name='post\_edit'),  
 url(r'^picture\_list', views.picture\_list, name='picture\_list'),  
 url(r'^picture/(?P<pk>\d+)/$', views.picture\_detail, name='picture\_detail'),  
 url(r'^picture/new/$', views.picture\_new, name='picture\_new'),  
 url(r'^picture/(?P<pk>\d+)/edit/$', views.picture\_edit, name='picture\_edit'),  
 ] + static(settings.MEDIA\_URL, document\_root=settings.MEDIA\_ROOT)

Файл page.css:

.page-header {  
 background-color: #4a89ff;  
 margin-top: 0;  
 padding: 20px 20px 20px 40px;  
}  
  
.page-header h1, .page-header h1 a, .page-header h1 a:visited, .page-header h1 a:active {  
 color: #ff6166;  
 font-size: 36pt;  
 text-decoration: none;  
}  
  
.content {  
 margin-left: 40px;  
}  
  
h1, h2, h3, h4 {  
 font-family: 'Book Antiqua', fantasy;  
}  
  
.date {  
 color: #828282;  
}  
  
.save {  
 float: right;  
}  
  
.post-form textarea, .post-form input {  
 width: 100%;  
}  
  
.top-menu, .top-menu:hover, .top-menu:visited {  
 color: #ffffff;  
 float: right;  
 font-size: 26pt;  
 margin-right: 20px;  
}  
  
.post {  
 margin-bottom: 70px;  
}  
  
.post h1 a, .post h1 a:visited {  
 color: #000000;  
}