**Министерство науки и высшего образования Российской Федерации**
**Федеральное государственное автономное образовательное учреждение**

**высшего образования**
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Факультет:** инфокоммуникационных технологий

**Направление подготовки:** интеллектуальные системы в гуманитарной сфере

# ОТЧЕТ

**Тема задания:** Реализация простого сайта средствами django

**Выполнил:** Селезнев Илья К33412

**Проверил:** Говоров А.И.

Санкт-Петербург 2020

**Цель**: овладеть практическими навыками и умениями реализации webсервисов средствами Django 2.2.

**Описание варианта No6 (Табло победителей автогонок):**

Табло должно отображать информацию об участниках автогонок: ФИО участника, название команды, описание автомобиля, описание участника, опыт и класс участника.

Необходимо реализовать следующий функционал:
- Регистрация новых пользователей.
- Просмотр автогонок и регистрацию гонщиков. Пользователь должен иметь возможность редактирования и удаления своих регистраций.

- Написание отзывов и комментариев к автогонкам. Предварительно комментатор должен зарегистрироваться. При добавлении комментариев должны сохраняться даты заезда, текст комментария, тип комментария (вопрос о сотрудничестве, вопрос о гонках, иное), рейтинг (1-10), информация о комментаторе.

- Администратор должен иметь возможность указания времени заезда и результата средствами Django-admin.

- В клиентской части должна формироваться таблица всех заездов и результатов конкретной гонки.

**Описание модели базы данных:**

```python
from django.db import models
from datetime import datetime
import django
from django.contrib.auth.models import User

class Driver(models.Model):
    COUNTRIES = (
        ('ES', 'Spain'),
        ('IT', 'Italy'),
        ('FR', 'France'),
        ('DE', 'Deutschland'))

    RACER_TYPES = (
        ('Drag', 'Drag racer'),
        ('Sport', 'Sports car racer'),
        ('Off', 'Off-road racer'))
    user = models.OneToOneField(User, null=True, on_delete=models.CASCADE)
    name = models.CharField(max_length=200, null=True)
    surname = models.CharField(max_length=200, null=True)
    team = models.CharField(max_length=200, null=True)
    country = models.CharField(max_length=2, choices=COUNTRIES, null=True)
    driver_class = models.CharField(max_length=5, choices=RACER_TYPES, null=True)
    age = models.IntegerField(null=True)
    experience = models.IntegerField(null=True)

    def __str__(self):
        return '{} {}'.format(self.name, self.surname)

class Car(models.Model):
    CAR_TYPES = (
        ('Drag', 'Drag'),
        ('Sport', 'Sportscar'),
        ('Off', 'Off-road'))
    car_model = models.CharField(max_length=200, null=True)
    number = models.IntegerField(null=True, unique=True)
    car_class = models.CharField(max_length=200, choices=CAR_TYPES, null=True)
    speed = models.IntegerField(null=True)
```

```python
class Race(models.Model):
    RACE_TYPES = (
        ('Drag', 'Drag racing'),
        ('Sport', 'Sports car racing'),
        ('Off', 'Off-road racing'))
    race_date = models.DateField(null=True)
    race_type = models.CharField(max_length=200, choices=RACE_TYPES, null=True
    length = models.IntegerField(null=True)
    name = models.CharField(max_length=200, null=True)
    registrations = models.ManyToManyField(Driver, through='Registration', bla
    time = models.TimeField(null=True, blank=True)

    def __str__(self):
        return '{} - {}'.format(self.name, self.id)


class Comment(models.Model):
    TYPE = (
        ('Cooperation', 'Cooperation'),
        ('Race', 'Race'),
        ('Other', 'Other'))
    RATE = (
        ('1', '1'),
        ('2', '2'),
        ('3', '3'),
        ('4', '4'),
        ('5', '5'))
    race = models.ForeignKey(Race, null=True, on_delete=models.SET_NULL)
    driver = models.ForeignKey(Driver, null=True, on_delete=models.SET_NULL)
    text = models.TextField()
    com_type = models.CharField(max_length=11, choices=TYPE, null=True)
    grade = models.CharField(max_length=1, choices=RATE, null=True)
    def __str__(self):
        return '{}: {}'.format(self.driver, self.com_type)
```

```python
class Registration(models.Model):
    race = models.ForeignKey(Race, null=True, on_delete=models.SET_NULL)
    driver = models.ForeignKey(Driver, null=True, on_delete=models.SET_NULL)
    reg_date = models.DateField(null=True, blank=True, default=django.utils.timezone.now)
    place = models.IntegerField(null=True, blank=True)
    car = models.ForeignKey(Car, null=True, on_delete=models.SET_NULL)

    def __str__(self):
        return '{} {}'.format(self.race, self.driver)
```

**Примеры описания представлений:**

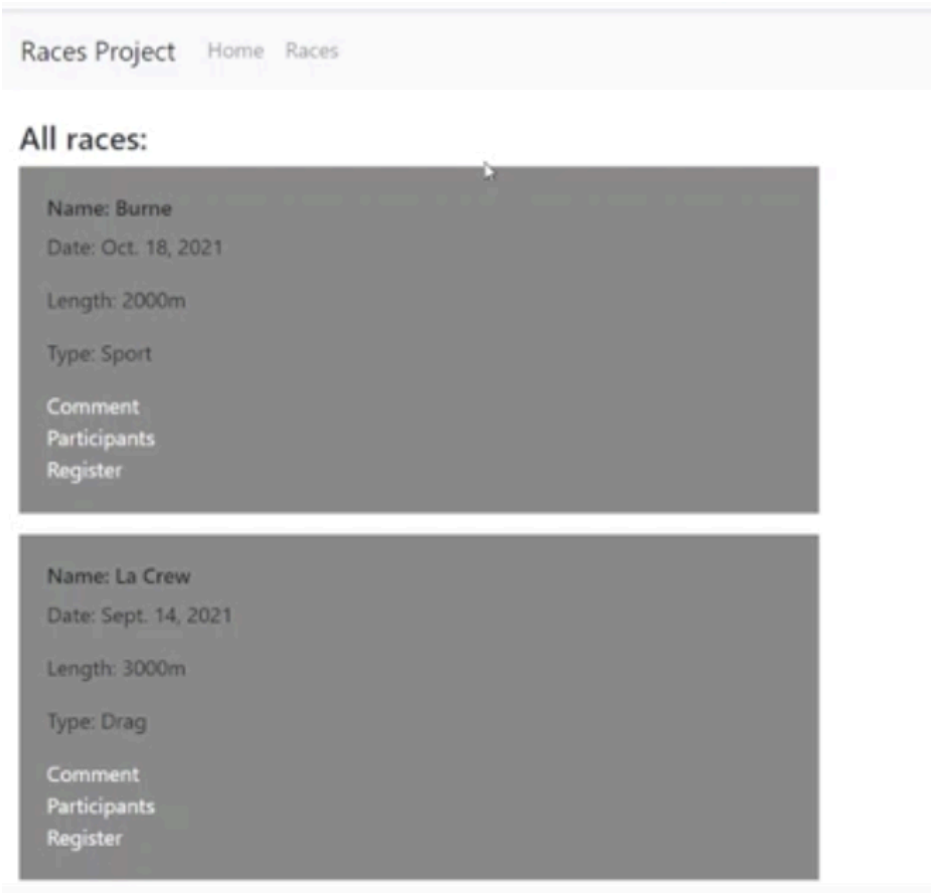
```python
def drivers(request, pk_test):
    drivers = []
    cars = []

    registrations = Registration.objects.filter(race=pk_test).order_by('place')
    for reg in registrations:
        drivers.append(reg.driver)
        cars.append(reg.car)
    print(drivers)

    return render(request, 'drivers.html', {'drivers': drivers, 'cars': cars, 'regs': registrations})
```

```python
def loginPage(request):
    if request.method == "POST":
        username = request.POST.get('username')
        password = request.POST.get('password')

        user = authenticate(request, username=username, password=password)

        if user is not None:
            login(request, user)
            return redirect('/races')
        else:
            messages.info(request, 'Username or password is not correct')

    context = {}
    return render(request, 'login_page.html', context)
```

**Описание роутеров:**

```python
urlpatterns = [
    path('', views.races),
    path('races/<int:pk_test>/drivers', views.drivers, name="race"),
    path('races/', views.races, name="races"),
    path('home/', views.home, name="home"),
    path('registration/create', views.createReg, name="registration_cr"),
    path('registration/<int:pk>/change', views.changeReg, name="registration_ch"),
    path('registration/<int:pk>/delete', views.deleteReg, name="registration_dl"),
    path('car/create', views.createCar, name="car_cr"),
    path('car/<int:pk>/change', views.changeCar, name="car_ch"),
    path('register', views.regPage, name="register"),
    path('login', views.loginPage, name="login"),
    path('logout', views.logoutUser, name="logout"),
    path('driver_cr', views.regDriver, name="driver_cr"),
    path('race/<int:pk>/registrate', views.raceReg, name="race_reg"),
    path('race/<int:pk>/comment', views.writeComment, name="comment"),
    path('race/create', views.createRace, name="race_cr"),
    path('race/<int:pk>/change', views.changeRace, name="race_ch"),
    path('race/<int:pk>/results', views.results, name="results"),
    path('race/<int:pk>/comments', views.comments, name="comments"),
    path('race/<int:pk>/delete', views.delRace, name="race_dl"),
]
```

**Страницы получившегося сайта:**

**Выводы: в результате выволнения лабораторной работы были получены практические навыки и умения реализации веб-серверов средствами Django.**