

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
ФАКУЛЬТЕТ ИНФОКОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ**

Отчет по лабораторной работе №2-3
по дисциплине «Основы Web-программирования»
по теме «РЕАЛИЗАЦИЯ WEB-СЕРВИСОВ СРЕДСТВАМИ Django REST
framework, Vue.js, Muse-UI»

Выполнил:
Лукина А.С
Студент группы К3343

Проверил:
Говоров А.И.

Санкт-Петербург
2020

Цель: овладеть практическими навыками и умениями реализации web-сервисов средствами Django REST framework, Vue.js, Muse-UI.

Задача: Реализовать сайт используя вышеуказанные технологии, в соответствии с практическим заданием.

Вариант: (4) Выставка собак

Описание задания по варианту: Создать программную систему, предназначенную для организаторов ежегодных выставок собак. Выставки могут быть моно- и полипородные. Она должна обеспечивать хранение сведений о собаках - участниках выставок и экспертах. Участие может быть индивидуальным или от клуба. У выставки могут быть спонсоры, которые могут спонсировать разные выставки. Для каждой собаки в БД должны храниться сведения, о том, к какому клубу она относится, кличка, порода и возраст, классность, сведения о родословной (номер документа, клички родителей), дата последней прививки, фамилия, имя, отчество и паспортные данные хозяина. Перед соревнованиями собаки должны пройти обязательный медосмотр. Т.к. участие является платным, то хозяин обязан после регистрации до прохождения медосмотра должен оплатить счет и предоставить его организаторам. Собака допускается до соревнований, если она успешно прошла медосмотр. Сведения об эксперте должны включать фамилию и имя, номер ринга, который он обслуживает, клуб, название клуба, в котором он состоит. Каждый ринг могут обслуживать несколько экспертов. Каждая порода собак выступает на своем ринге, но на одном и том же ринге в разное время могут выступать разные породы. Каждая собака должна выполнить 3 упражнения, за каждое из которых она получает баллы от каждого эксперта. Итогом выставки является определение медалистов по каждой породе по итоговому рейтингу.

Организатор выставки должен иметь возможность добавить в базу нового участника или нового эксперта, снять эксперта с судейства, заменив его другим, отстранить собаку от участия в выставке.

Организатору выставки могут потребоваться следующие сведения;

- На каком ринге выступает заданный хозяин со своей собакой?
- Какими породами представлен заданный клуб?
- Сколько собак были отстранены от участия в выставке?
- Какие эксперты обслуживают породу?
- Количество участников по каждой породе?

Необходимо предусмотреть возможность выдачи отчета о результатах заданной выставки (сколько всего участников, какие породы, сколько медалей по каждой породе).

Этапы выполнения работы:

В первую очередь была спроектирована схема базы данных, представленная на рисунке 1. На схеме отображены все функциональные характеристики, необходимые для реализации программной системы.

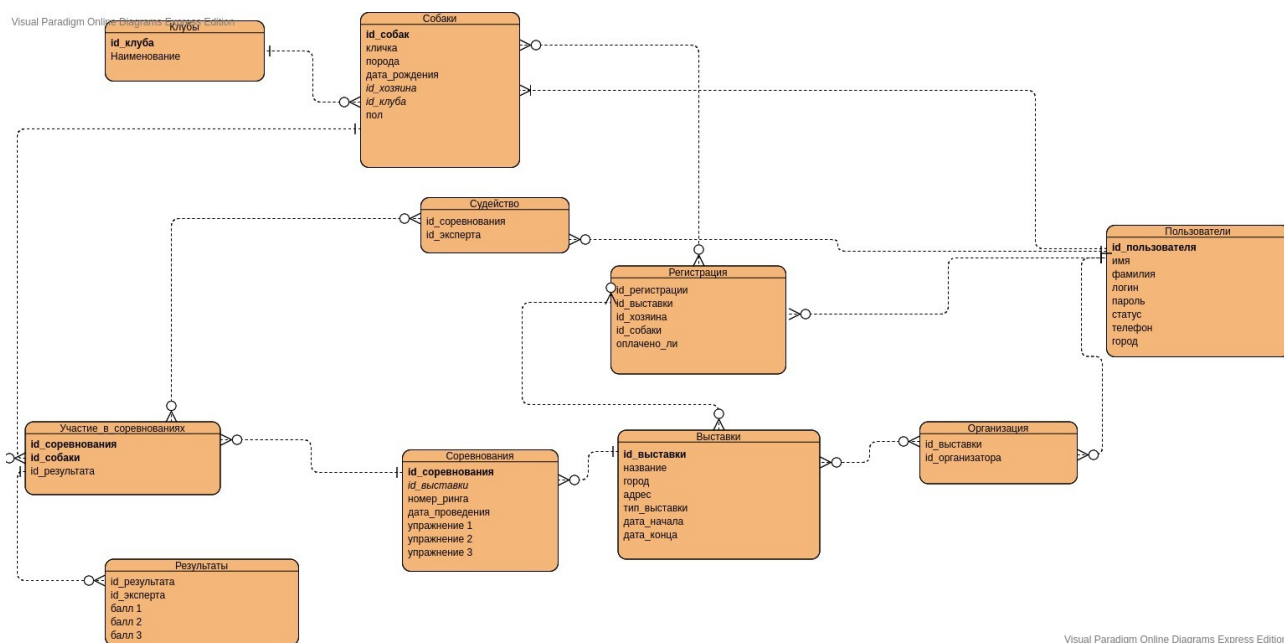


Рисунок 1 - ER-диаграмма для программной системы "Выставка собак"

Программная система состоит из двух главных частей: бэкенд и фронтэнд, которые реализованы с помощью Django REST Framework и Vue JS соответственно. Главное преимущество использования данных технологий состоит в том, что серверную и клиентскую часть можно модернизировать параллельно, что позволяет сервису, написанному с указанными технологиями быть легко масштабируемым.

Использование Django REST Framework позволило работать с Django API, что упростило работу с отслеживанием состояния системы и формированию запросов. Приведем пример такого использования для функционала пользователей (Users).

Для начала было необходимо описать сериализатор в файле **serializers.py**, который выглядит следующим образом:

```
class ProfileSer(serializers.ModelSerializer):
    """Сериализация профиля"""
    user = UserSerializer()
    follow = UserSerializer(many=True)

    class Meta:
        model = Profile
        fields = ('__all__',)
```

Рисунок 2 - Сериализация профиля пользователя

Данный сериализатор был использован для написания представления в файле **views.py**.
Ниже представлены два представления на основе сериализатора профиля: вывод и редактирование профиля.

```
class ProfileUser(APIView):
    """Вывод профиля пользователя"""
    permission_classes = [permissions.IsAuthenticated]

    def get(self, request):
        ser = ProfileSer(Profile.objects.get(user=request.user))
        return Response(ser.data)

class UpdateProfile(APIView):
    """Редактирование профиля"""
    permission_classes = [permissions.IsAuthenticated]

    def post(self, request):
        prof = Profile.objects.get(user=request.user)
        ser = EditAvatar(prof, data=request.data)
        if ser.is_valid():
            if "avatar" in request.FILES:
                ser.save(avatar=request.FILES["avatar"])
                return Response(status=201)
            else:
                return Response(status=400)
```

Рисунок 3 - Представления для вывода и редактирования профиля пользователя

Для того, чтобы выйти на страницу API и просмотреть полученную информацию, необходимо прописать путь в файле **urls.py**:

```
urlpatterns = [
    path('', ProfileUser.as_view()),
    path('update-ava/', UpdateProfile.as_view()),
]
```

Рисунок 4 - Пути для доступа

Используя любой из указанных путей можно просмотреть «серверную» информацию о конкретной модели, описанной в сериализаторе. Например, на рисунке ниже представлено отображение информации для модели пользователя:

Profile User

Вывод профиля пользователя

```
GET /api/v1/profile/
```

```
HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
  "id": 3,
  "user": {
    "id": 3,
    "username": "eviekevie"
  },
  "avatar": null,
  "name": "Анастасия",
  "surname": "Лукина"
}
```

Рисунок 5 - Вывод информации о пользователях через API

На рисунке 5 показано отображение одного из пользователей, уже зарегистрированного в системе, с указанием его информации. С каждой новой регистрацией информация о пользователях будет пополняться.

Описанным выше способом был реализован и дальнейший функционал системы, описывающий все модели, отображенные в файле **models.py**. (Реализованная база данных). Файлы **models.py** всех приложений содержат следующие «таблицы»:

Название модели	Назначение
Exhibition	Информация о выставках
Competition	Информация о соревнованиях
DogOwner	Информация о хозяинах собак
Expert	Информация об экспертах
ExpertCompetition	Информация о судействе (какие эксперты назначены на соревнования)
Dog	Информация о собаках
DogRegistration	Информация о регистрации собаки на выставку
CompParticipation	Информация о участии на соревновании
Result	Информация о результатах конкретного соревнования
Club	Информация о клубах
ClubParticipation	Информация о членстве в клубах
Dismissed	Информация об отстранённых собаках
User	Информация о пользователе

Файлы **views.py** представлен следующими представлениями:

Название представления	Описание
exhibition_info	Информация о всех выставках
exhibition_add	Добавление выставки
one_exhibition_info	Информация о конкретной выставке и отчёта по ней
experts_output	Информация обо всех экспертах
set_experts	Назначение и снятие экспертов на выставку
competition_add	Добавление соревнования к выставке
dog_to_comp	Назначение и снятие собак с соревнования
dog_reg	Регистрация собак на выставку
query	Представление для формирования пяти запросов
ProfileView	Вывод информации о пользователе
ProfileEditView	Редактирование пользователя

Помимо прочего были определены формы для внесения изменений в существующую базу данных, которые хранятся в файлах **forms.py**.

Форма	Назначение
ExhibitionForm	Форма добавления выставки
SetExpertForm	Форма назначения эксперта на выставку
DelExpertForm	Форма снятия эксперта с выставки
CompetitionForm	Форма добавления соревнования
DogToCompForm	Форма назначения собаки на соревнования
DelDogFromCompForm	Форма снятия собаки с соревнования
DogRegForm	Форма регистрации собаки на участие в выставке
Query2Form	Форма запроса №2
Query3Form	Форма запроса №3
ProfileForm	Форма редактирования профиля

Таким образом в данной работе представлен бэкэнд.

Фронтэнд представлен шаблонами и файлами с расширением .vue. Все функциональные vue файлы хранятся в директории **components** и состоят из трёх частей: шаблона, скриптовой части и части стилей. Приведем пример описания одного из таких файлов — Login.vue (авторизация). Авторизация представляет собой модальное окно, которое было реализовано посредством технологии AJAX, что отображено в части скриптов.

Первая часть — часть шаблонов — ограничена тэгом `template` и описывает внешний структурный вид модального окна:

```
<template>
  <div class="" id="loginModal">
    <div class="modal-dialog modal-dialog-centered auth-modal">
      <div class="modal-content">
        <!-- Modal Header -->
        <div class="modal-header">
          <h4 class="modal-title">Вход</h4>
          <button @click="close" type="button" class="close" data-dismiss="modal">
            &times;
          </button>
        </div>

        <!-- Modal body -->
        <div class="modal-body">
          <p>{{mess}}</p>
          <input type="text" placeholder="Логин" value="" v-model="user.username">
          <input type="password" placeholder="Пароль" value="" v-model="user.password">
          <button type="button" @click="setLogin">Войти</button>
        </div>
      </div>
    </div>
  </div>
</template>
```

Рисунок 6 - Часть шаблонов в файле Login.vue

Вторая часть — часть скриптов — ограничена тэгом `script` и описывает функции, срабатывающие при нажатии на кнопки.

```
<script>
  export default {
    name: "Login",
    data() {
      return {
        user: {
          username: "",
          password: ""
        },
        mess: '',
      }
    },
    methods: {
      setLogin() {
        $.ajax({
          url: this.$store.getters.get_url_server + 'auth/token/login/',
          type: "POST",
          data: {
            username: this.user.username,
            password: this.user.password
          },
          success: (response) => {
            sessionStorage.setItem("token", response.auth_token)
            this.$store.commit("set_auth", true)

            $.ajaxSetup({
              headers: {'Authorization': "Token " + sessionStorage.getItem('token')},
            });
            this.close()
          },
          error: (response) => {
            if (response.status === 400) {
              this.mess = response.responseJSON.non_field_errors[0]
            }
          }
        })
      },
      close() {
        this.$emit("hideLogin")
      }
    }
  }
</script>
```

Рисунок 7 - Часть скриптов в файле Login.vue

И последняя часть — часть стилей — ограничена тэгом `style` и описывает внешний вид шаблона (css стили):

```
<style scoped>
  #loginModal {
    position: fixed;
    z-index: 1000;
    top: -150px;
    left: 40%;
  }
</style>
```

Рисунок 8 - Часть стилей в файле Login.vue

Таким образом в данной работе представлен **фронтэнд**. К этому блоку также можно отнести все назначенные html-шаблоны. Ниже будут представлены основные интерфейсы с логикой работы программы.

В меню представлено пять опций: «Эксперты», «Выставки», «Запросы», аккаунт и «Выход» или «Вход», если вход в системы не был совершен. На рисунке 9 представлен пример выпадающего списка из опции «Выставки» для доступа к основному функционалу.

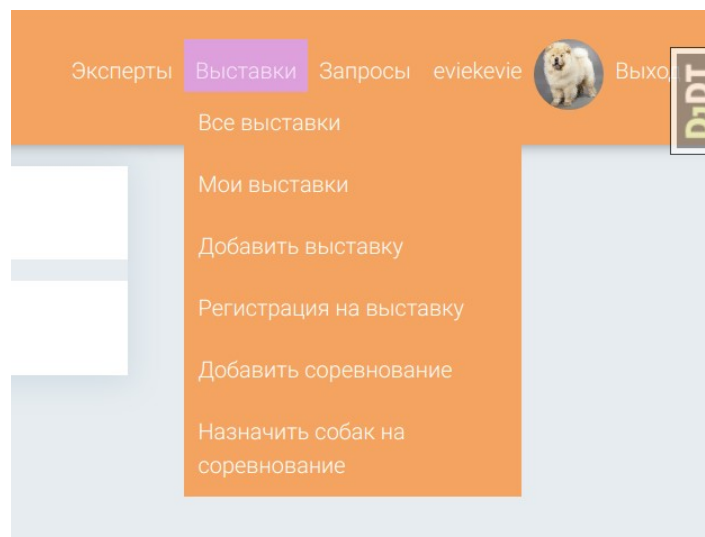


Рисунок 9 - Главное меню сайта

Вывод всех выставок представляет собой список «карточек», при нажатии на которые раскрывается полная информация о выставке. Пример представлен на рисунке 10.

Выставка пупсиков и лапусек

Симпозиум умничек и красавчиков



Дата проведения: с 14 мая 2020 г. по 14 мая 2020 г.


Место проведения: Санкт-Петербург, Лимонная ул., 672

Организатор: Анастасия Лукина

Соревнование чудесных малышей

Рисунок 10 - Вывод всех выставок

Информация о конкретной выставке включает себя её название, сроки проведения, организатора и отчётной информации (результаты). Пример представлен на рисунке 11.



Дата проведения: с 20 января 2020 г. по 22 января 2020 г.

Место проведения: Москва, Красный проспект, 36

Количество участников: 3

Список пород: Чихуа-хуа, Чау-чау, Немецкая овчарка,

Номер ринга	Дата соревнования	Упражнение 1	Упражнение 2	Упражнение 3	Результаты
1	20 января 2020 г.	Показательный ринг	Полоса препятствий	Выполнение команд	Результаты
2	20 января 2020 г.	Команда "лежать"	Команда "сидеть"	Команда "дай лапу"	Результаты
	Кличка собаки	Балл 1	Балл 2	Балл 3	Эксперт
	Боня	30	30	30	Денис Семёнов

Рисунок 11 - Пример вывода информации о выбранной выставке

Интерфейс регистрации собаки на выставку содержит поля с выбором выставки, хозяина и его собаки. Поле ввода собаки предусмотрено таким образом, чтобы были доступны только собаки выбранного хозяина. Пример регистрации представлен на рисунке 12.

Это поле обязательно.
Выставка:

Выставка пупсиков и лапусек ▼

Это поле обязательно.
Хозяин:

Ирина Власова ▼

Это поле обязательно.
Собака:

Мухтар ▼

Оплата:

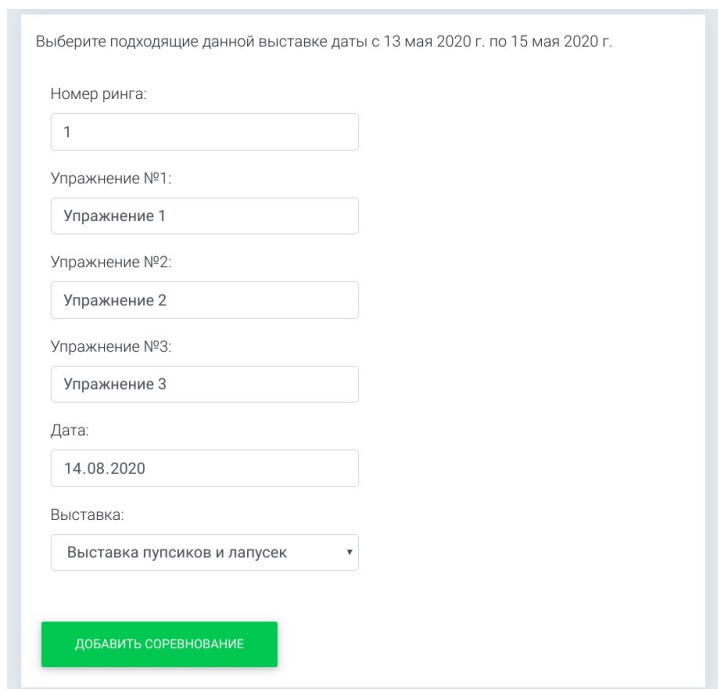
☐

[ЗАРЕГИСТРИРОВАТЬ СОБАКУ](#)

Рисунок 12 - Пример регистрации собаки на выставку

Добавление соревнования на выставку подразумевает ввод номера ринга, названия трёх упражнений, дату и саму выставку. Список выставок включает только выставки

организатора, который произвел вход в систему. При этом, если организатор по случайности укажет неверную дату, то систему ему об этом сообщит. Пример приведен на рисунке 13.



Выберите подходящие данной выставке даты с 13 мая 2020 г. по 15 мая 2020 г.

Номер ринга:

Упражнение №1:

Упражнение №2:

Упражнение №3:

Дата:

Выставка:

Рисунок 13 - Пример добавления соревнования на выставку

Назначение и снятие собаки с соревнования представлены на одной странице с возможностью переключения предусмотренными кнопками. При назначении собаки организатору предоставлена возможность выбора только тех собак, которые были зарегистрированы на участие в выставке, на которой рассматриваемое соревнование проходит. В свою очередь для снятия собак предоставлен выбор собак, которые ранее были назначены на соревнование. Пример представлен на рисунке 14.

Назначить Снять

Это поле обязательно.
Соревнование:

----- ▾

Это поле обязательно.
Собака:

----- ▾

НАЗНАЧИТЬ СОБАКУ

Рисунок 14 - Пример страницы с назначением и снятием собак с соревнований

Назначение и снятие экспертов с выставки реализовано похожим способом. Список соревнований включает себя только те соревнования, которые являются частью выставок того пользователя, который вошел в систему. Пример представлен на рисунке 15.

Назначить Снять

Это поле обязательно.
Соревнование:

Симпозиум умничек и красавчиков 2020-05-14 1 ринг ▾

Это поле обязательно.
Эксперт:

Валерия Павлова ▾

НАЗНАЧИТЬ ЭКСПЕРТА

Рисунок 15 - Пример страницы с назначением и снятием экспертов с соревнований

Помимо прочего в системе реализованы формы для пяти запросов. Первый запрос представляет собой форму для ответа на вопрос «На каком ринге выступает заданный хозяин со своей собакой?» Для того, чтобы узнать интересующую информацию необходимо указать только релевантную информацию, в противном случае систему сообщит об ошибке (например, о несовпадении принадлежности собаки хозяину). Пример успешного запроса представлен на рисунке 16, а на рисунке 17 — не успешного.

На каком ринге выступает заданный хозяин со своей собакой?

Введите имя хозяина:

Введите кличку собаки:

УЗНАТЬ

Мария Мышкина с собакой Боня выступает на ринге № 2

Рисунок 16 - Пример успешно выполненного запроса №1

На каком ринге выступает заданный хозяин со своей собакой?

Введите имя хозяина:

Введите кличку собаки:

УЗНАТЬ

Такого хозяина нет в базе данных или у указанного хозяина нет указанной собаки

Рисунок 17 - Пример ошибки при некорректном запросе №1

Форма запроса №2 представляет собой одно поле с выбором всем доступных клубов для того, чтобы ответить на вопрос «Какими породами представлен заданный клуб?». Пример выполнения этого запроса представлен на рисунке 18.

Какими породами представлен заданный клуб?

Клуб:

Все псы попадут в рай ▼

УЗНАТЬ

Немецкая овчарка

Чихуа-хуа

Рисунок 18 - Пример выполнения запроса №2

Запрос №3 служит для ответа на вопрос «Сколько собак были отстранены от участия в выставке?». Система ищет по каждому соревнованию для указанной выставки и выводит информацию о количестве и список отстранённых собак. Пример представлен на рисунке 19.

Сколько собак были отстранены от участия в выставке?

Выставка:

Соревнование чудесных малышей ▼

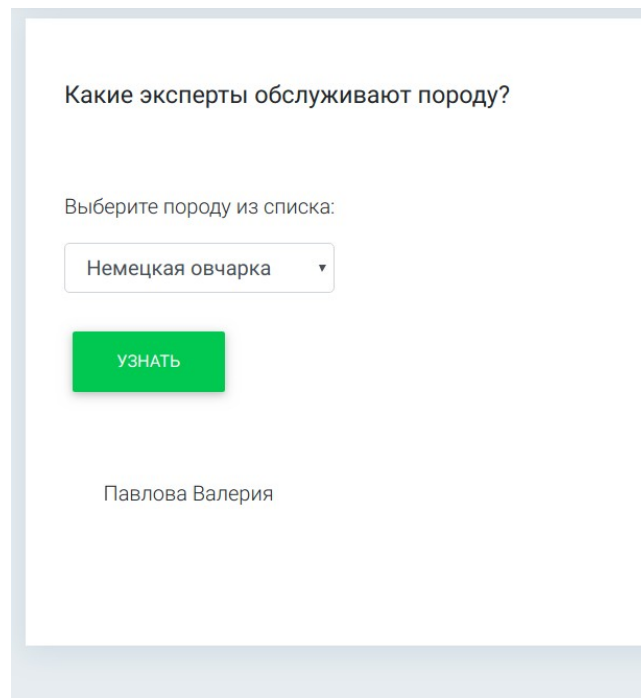
УЗНАТЬ

Всего отстранённых собак: 1

Подробнее: Рекс

Рисунок 19 - Пример выполнения запроса №3

Запрос №4 отвечает на вопрос «Какие эксперты обслуживают породу?». Для этого организатору необходимо из списка выбрать породу, и в случае успеха, система выводит список экспертов. Пример представлен на рисунке 20.



Какие эксперты обслуживают породу?

Выберите породу из списка:

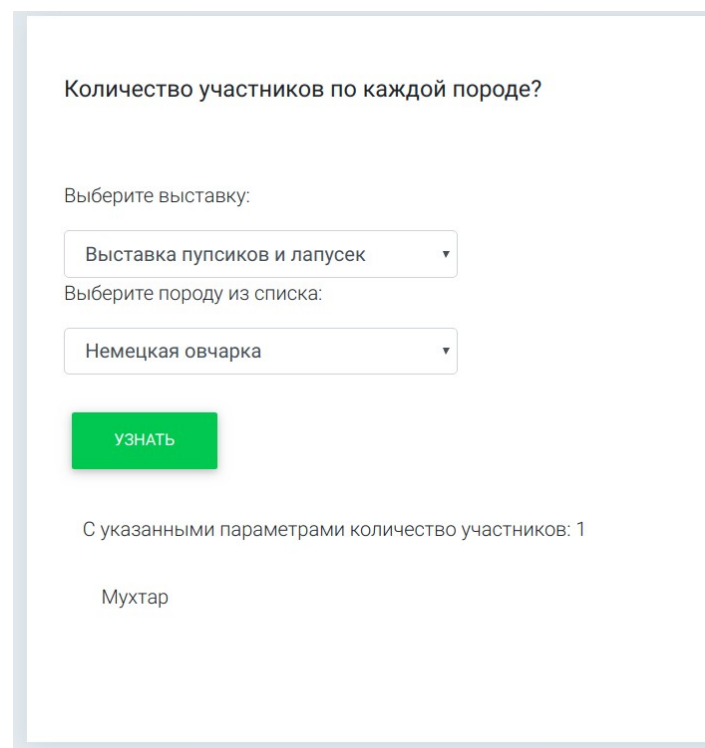
Немецкая овчарка ▼

УЗНАТЬ

Павлова Валерия

Рисунок 20 - Пример выполнения запроса №4

Последний запрос ищет количество участников по каждой породе. Для этого нужно указать выставку и интересующую породу. В случае успеха система выводит количество собак, а также список собак. Пример представлен на рисунке 21.



Количество участников по каждой породе?

Выберите выставку:

Выставка пупсиков и лапусек ▼

Выберите породу из списка:

Немецкая овчарка ▼

УЗНАТЬ

С указанными параметрами количество участников: 1

Мухтар

Рисунок 21 - Пример выполнения запроса №5

Помимо рассмотренных функций пользователь (организатор) имеет возможность просмотреть и отредактировать свои личные данные. При регистрации пользователю

устанавливает стандартный аватар, который при желании можно поменять на любой другой. При успешном редактировании система выводит оповещение о сохранённых изменениях. Пример просмотра информации представлен на рисунке 22, редактирования — на рисунке 23.

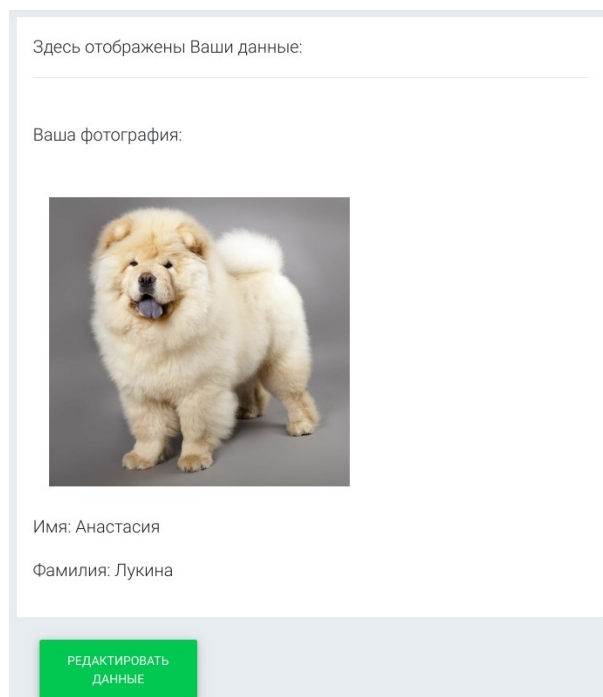


Рисунок 22 - Просмотр личной информации в профиле

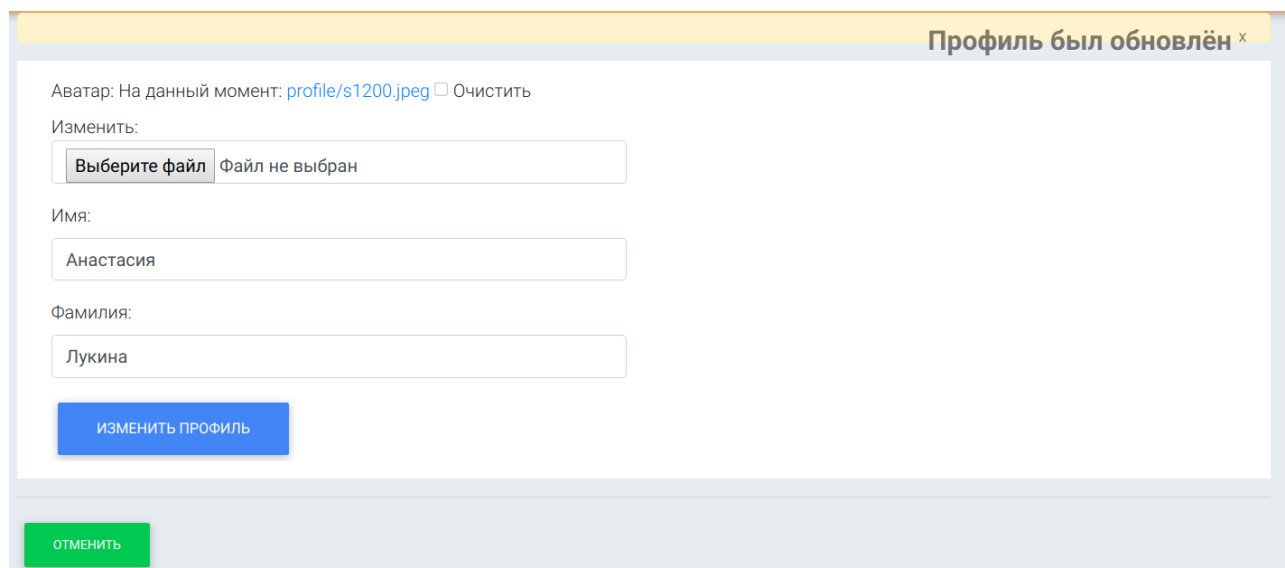
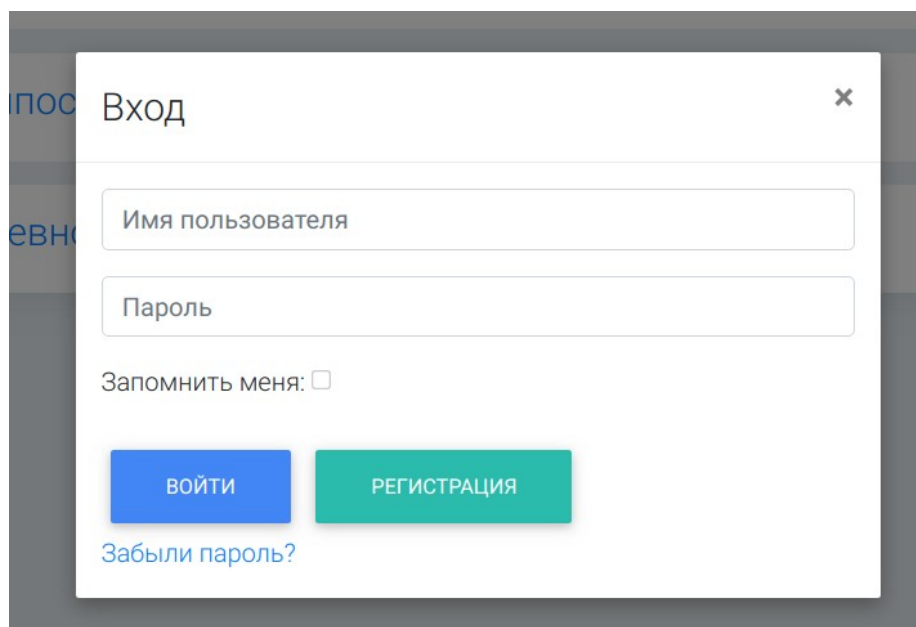


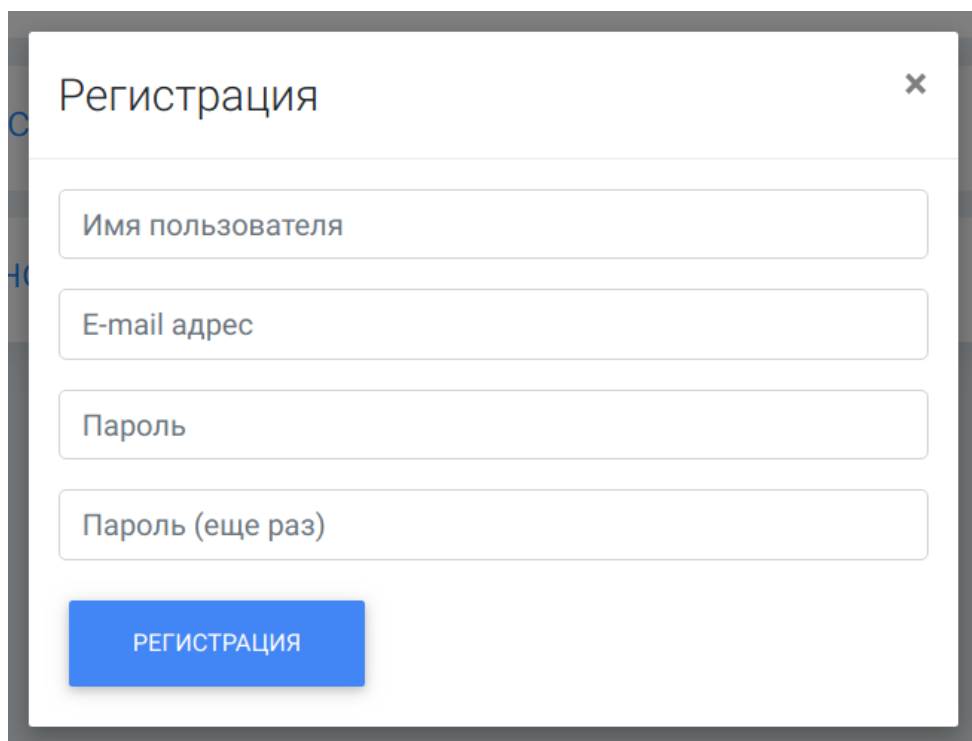
Рисунок 23 - Пример редактирование личной информации в профиле

Новые пользователи также имеют возможность использовать систему, но перед этим им необходимо пройти процесс регистрации (рисунок 24) и авторизации (рисунок 25).



Modal window titled "Вход" (Login) with a close button (X) in the top right corner. It contains two input fields: "Имя пользователя" (Username) and "Пароль" (Password). Below the password field is a checkbox labeled "Запомнить меня:" (Remember me:). At the bottom, there are two buttons: "ВОЙТИ" (Login) in blue and "РЕГИСТРАЦИЯ" (Registration) in green. A link "Забыли пароль?" (Forgot password?) is located below the buttons.

Рисунок 24 - Модальное окно для авторизации



Modal window titled "Регистрация" (Registration) with a close button (X) in the top right corner. It contains four input fields: "Имя пользователя" (Username), "E-mail адрес" (Email address), "Пароль" (Password), and "Пароль (еще раз)" (Password (again)). At the bottom, there is a blue button labeled "РЕГИСТРАЦИЯ" (Registration).

Рисунок 25 - Модальное окно для регистрации

Авторизованные пользователи имеют возможность выхода из системы. В этом случае они становятся «гостями» и имеют возможность только просматривать уже существующие выставки. Пример представлен на рисунке 26, на котором показано, что у гостей нет никаких других прав.

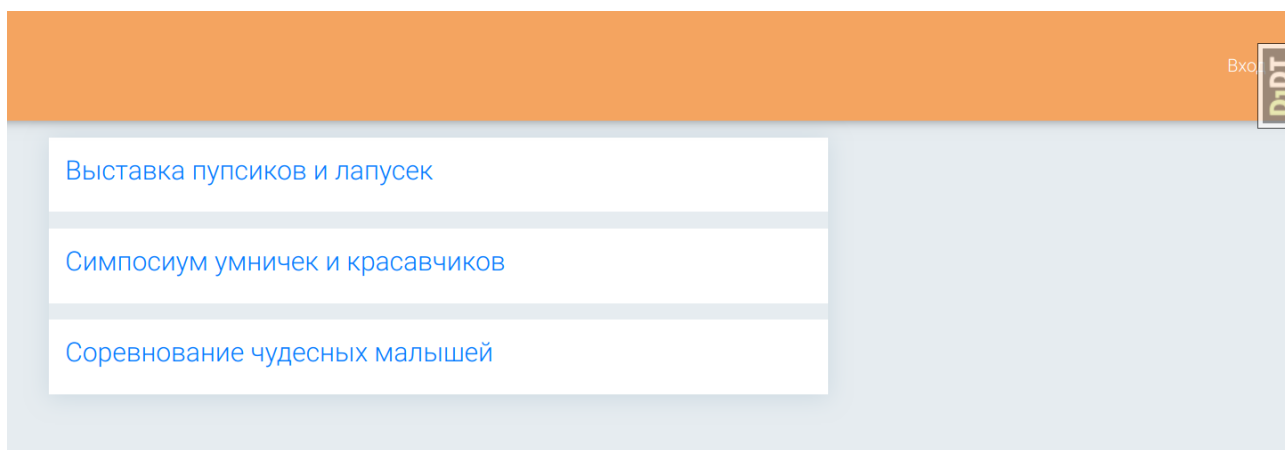


Рисунок 26 - Главная страница для гостей с возможностью входа

Выводы по работе

В результате выполнения лабораторной работы были получены навыки работы с технологиями Django REST Framework, а также с Django API, Vue JS и другими библиотеками для клиентской стороны. Помимо этого был спроектирован и реализован веб-сервис согласно варианту.