

САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ИТМО
ФАКУЛЬТЕТ ИКТ

Дисциплина: Основы Web-программирования

Отчет

по лабораторной работе №2-3

Тема: РЕАЛИЗАЦИЯ WEB-СЕРВИСОВ СРЕДСТВАМИ Django REST
framework, Vue.js, Muse-UI

Выполнила:

Цыбаева А.

Группа К3343

Проверил:

Говоров А.И.

Санкт-Петербург

2020 г.

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

Программное обеспечение: Python 3.6, Django REST framework, Vue.js, Muse-UI (или аналогичная библиотека), PostgreSQL *.

Практическое задание:

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

Необходимо использовать те же варианты практических заданий, которые выполнялись в рамках предмета «Базы данных», либо предложить преподавателю согласовать свой (индивидуальный) вариант.

Был предложен индивидуальный вариант.

Условия практического задания:

Создать программную систему, предназначенную для работников эко-такси. Такая система должна обеспечивать хранение сведений о закрытых и предстоящих заказах, собранном и сданном мусоре, о заказчиках и водителях такси.

Для каждого заказа в БД должны содержаться следующие сведения: id заказчика, адрес заказчика, вид мусора, который он предпочитает сдать, масса мусора, стоимость заказа, выполненность заказа, id водителя, время и дата исполнения заказа (если заказ еще не выполнен, то указывается промежуток исполнения).

Для каждого заказчика в БД должны быть ФИО, адрес, телефон, email, дата рождения.

Для каждого водителя в БД должны содержаться ФИО, телефон, email, дата рождения, марка машины, номер машины, номер паспорта, количество выполненных заказов.

В эко-такси можно сдать следующие виды мусора: “Компост”, “Пластик Твердый”, “Пластик Мягкий”, “Макулатура”, “Стекло”, “ТэтраПак”, “Пенопласт”, “Лампочки и батарейки”.

После выполнения заказа мусор вывозиться на склад, где сортируется и передается на перерабатывающие заводы. Если мусор сдан на завод, то он списывается со склада. В бд склада должна содержаться следующая информация : Вид мусора, масса мусора, стоимость сдачи мусора на завод. Работа с системой предполагает получение следующей информации:

- 1) О заказах, выполненным данным водителем за сегодня;
- 2) О количестве каждого вида мусора на складе;
- 3) Список самых сдаваемых видов мусора (3 позиции);
- 4) Сколько клиентов моложе 40;
- 5) Сколько клиентов в процентном соотношении пользовались эко-такси в каждый из дней недели?

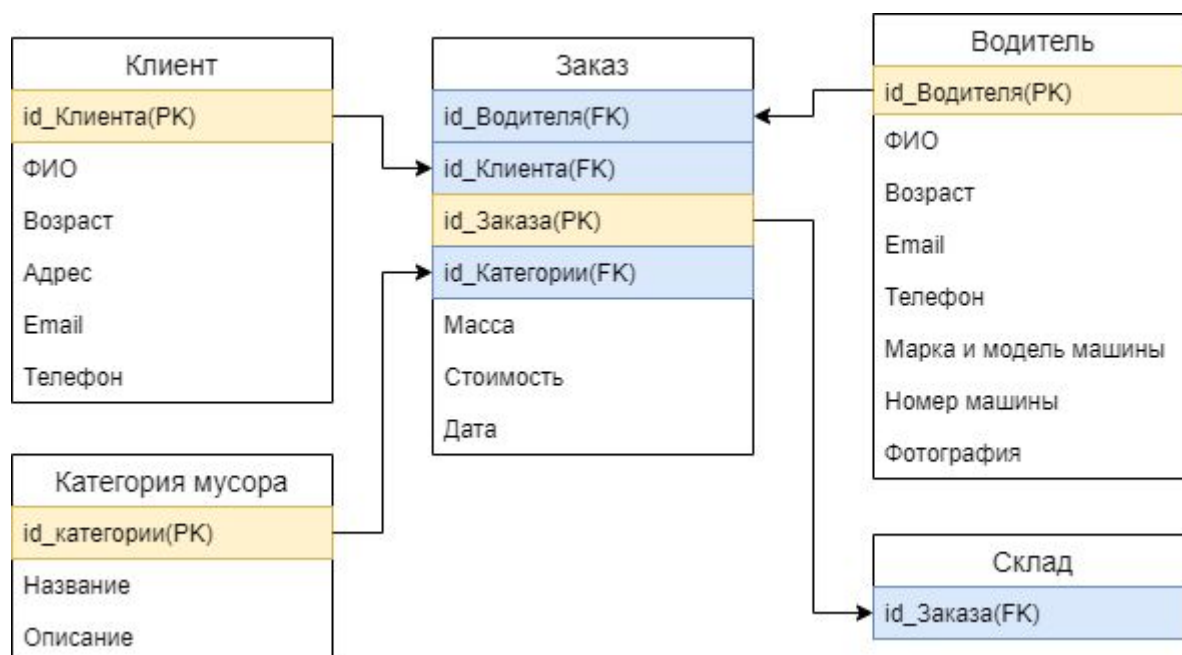
Администратор должен иметь возможность выполнить следующие операции:

1. Нанять или уволить водителя;
2. Списать мусор со склада;

Необходимо предусмотреть возможность выдачи отчета о работе такси в течение месяца. Отчет должен включать в себя следующую информацию: количество забранного у заказчика и количество сданного на завод мусора, количество обслуженных клиентов, итоговый доход за отчетный месяц.

Выполнение работы:

- 1) Построение базы данных:



2) Реализация построенной базы данных в файле models.py Django.

Пример добавления сущности “Клиент”:

```
class Client(models.Model):
    """Клиент"""
    name = models.CharField("ФИО", max_length=100)
    age = models.IntegerField("Возраст")
    email = models.EmailField("Email")
    telephone = models.CharField("Телефон", max_length=15)
    address = models.TextField("Адрес", max_length=100)

    def __str__(self):
        return self.name

    class Meta:
        verbose_name = "Клиент"
        verbose_name_plural = "Клиенты"
```

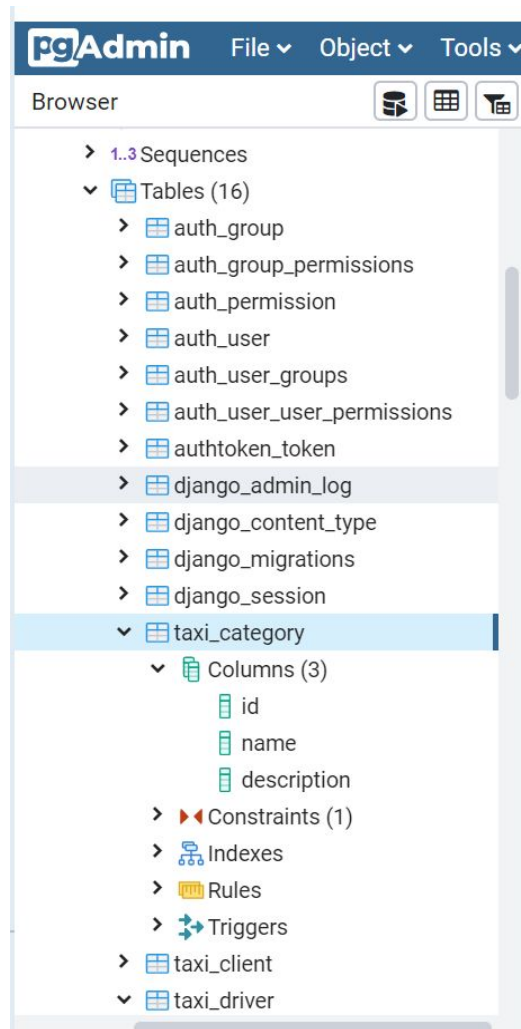
3) Установка необходимых пакетов для работы с Django Rest Framework:

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
    'djoser',  
    'rest_framework.auth_token',  
    'rest_framework',  
    'django_filters',  
    'corsheaders',  
    'taxi'  
]
```

4) Настройка базы данных:

```
DATABASES = {  
    'default': {  
        'ENGINE': 'django.db.backends.sqlite3',  
        'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),  
    }  
}
```

5) Применение миграций, проверка правильности функционирования БД:



- 6) Создание сериализеров в соответствии с условием практического задания. Пример вывода топ 3 категорий сдаваемого мусора:

```
class TopCategorySerializer(serializers.Serializer):
    """Топ 3 категории сдаваемого мусора"""
    amount = serializers.IntegerField()
    category = serializers.CharField(source='category__name')

    class Meta:
        model = Order
        fields = ("amount", "category")
```

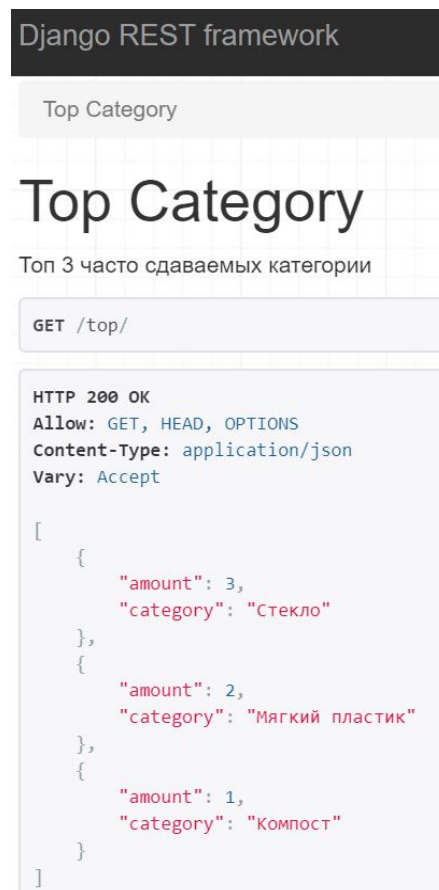
- 7) Создание views и urls для отображения необходимой информации на сайте. Использована библиотека DRF generics.

```

class TopCategoryView(generics.ListAPIView):
    """Топ 3 часто сдаваемых категории"""
    def get_queryset(self):
        order = Order.objects.values('category__name').annotate(amount=Count('id')).order_by('-amount')[:3]
        return order
    serializer_class = TopCategorySerializer
    permission_classes = [permissions.IsAuthenticatedOrReadOnly]

```

8) Проверка вывода нужной информации:



9) Создание фильтров для вывода нужной информации. Пример фильтрации клиентов моложе 40:

```

class ClientFilter(filters.FilterSet):
    age = filters.RangeFilter()

    class Meta:
        model = Client
        fields = ['age']

```

10) Аналогичное создание views и urls для использования фильтра:

```

class YoungListView(generics.ListAPIView):
    """Вывод списка клиентов моложе 40"""
    queryset = Client.objects.all()
    serializer_class = YoungClientSerializer
    permission_classes = [permissions.IsAuthenticatedOrReadOnly]
    filterset_class = ClientFilter
    filter_backends = (DjangoFilterBackend,)

```

11) Пример фильтрации клиентов по возрасту:

Вывод списка клиентов моложе 40

GET /client/?age_max=40

```

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

[
  {
    "name": "Цыбаева Арина Олеговна",
    "age": 20
  },
  {
    "name": "Куликов Артем Васильевич",
    "age": 40
  },
  {
    "name": "Чужакова",
    "age": 32
  },
  {
    "name": "ofis",
    "age": 23
  },
  {
    "name": "Alisa",
    "age": 9
  }
]

```

12) Создание регистрации и аутентификации пользователя, добавление прав доступа с помощью библиотеки DRF permissions.


```
DJOSER = {
    'PASSWORD_RESET_CONFIRM_URL': '#/password/reset/confirm/{uid}/{token}',
    'USERNAME_RESET_CONFIRM_URL': '#/username/reset/confirm/{uid}/{token}',
    'ACTIVATION_URL': '#/activate/{uid}/{token}',
    'SEND_ACTIVATION_EMAIL': True,
    'SERIALIZERS': {},
}
```

```
REST_FRAMEWORK = {
    'DEFAULT_AUTHENTICATION_CLASSES': (
        'rest_framework.authentication.TokenAuthentication',
        'rest_framework_simplejwt.authentication.JWTAuthentication'
    ),
    'DEFAULT_SCHEMA_CLASS': 'rest_framework.schemas.coreapi.AutoSchema',
    'DEFAULT_FILTER_BACKENDS': ('django_filters.rest_framework.DjangoFilterBackend',),
}
```

- 13) С помощью приложения Postman произведена регистрация пользователя, с помощью админовской панели предоставлены необходимые права доступа для штата, а также для администратора.

Пример регистрации через приложение Postman:

POST http://127.0.0.1:8000/auth/users/ Send

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings

☐ none
 ☒ form-data
 ☐ x-www-form-urlencoded
 ☐ raw
 ☐ binary
 ☐ GraphQL

	KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/>	username	admin	
<input checked="" type="checkbox"/>	password	123456Test	
<input checked="" type="checkbox"/>	email	testarina38@gmail.com	

Пример добавление прав доступа администратору через админовскую панель Django:

Personal info

First name:

Last name:

Email address:

Permissions

☒ Active

Designates whether this user should be treated as active. Unselect this instead of deleting accounts.

☒ Staff status

Designates whether the user can log into this admin site.

☒ Superuser status

Designates that this user has all permissions without explicitly assigning them.

Пример аутентификации через приложение Postman:

The screenshot shows a Postman interface for a POST request to `http://127.0.0.1:8000/auth/token/login`. The request body is set to 'form-data' and contains two fields: 'password' with value '123456Test' and 'username' with value 'admin'. The response status is 200 OK, with a time of 650 ms and a size of 303 B. The response body is displayed in JSON format, showing an 'auth_token'.

KEY	VALUE	DESCRIPTION
password	123456Test	
username	admin	

Key	Value	Description
auth_token	"558828e815556c13b1d935f0cb43530fe3ad7924"	

```
1 {
2   "auth_token": "558828e815556c13b1d935f0cb43530fe3ad7924"
3 }
```

Как мы видим, аутентификация прошла успешно, был выдан токен для дальнейшей работы.

Необходимый бэкэнд на Django Rest Framework прописан, переходим к написанию фронтенда на Vue.js.

- 1) Устанавливаем плагин Vuetify для удобной и быстрой “сборки” сайта.
- 2) Добавляем необходимые разделы в App.vue (navigation-drawer, app-bar, list) и также методы для кнопок “login”, “Register”, “logout”.

Пример метода:

```
methods:{
  goLogin() {
    this.$router.push({name:"login"})
  },
}
```

- 3) Далее прописываем необходимые компоненты в соответствии с одобренным интерфейсом.

Пример создания страницы “Home” (главной):

На главной странице отображается 3 запроса: Клиенты моложе 40, процентная востребованность эко-такси в каждый день недели, топ 3 сдаваемых категории мусора. Для отображения информации были выбраны таблицы vuetify. Методы прописаны следующие:

```
methods: {
  async fetchClients() {
    const response = await fetch( input: 'http://127.0.0.1:8000/client/?age_max=40');
    this.clients = await response.json()
  },
  async fetchOrders() {
    const response = await fetch( input: 'http://127.0.0.1:8000/week/');
    this.visuals = await response.json()
  },
  async fetchTrash() {
    const response = await fetch( input: 'http://127.0.0.1:8000/top/');
    this.trash = await response.json()
  },
},
```

Пример реализации метода login:

```
methods:{  
  
  setLogin() {  
    this.$axios.post( url: "http://127.0.0.1:8000/auth/token/login", this.login )  
    .then(response =>{  
      console.log(response.data.auth_token)  
      let auth_token = response.data.auth_token;  
      localStorage.setItem('auth_token', auth_token);  
      alert("Спасибо, что с нами!");  
      this.$router.push({name: "home"});  
    })  
  }  
}}
```

Результат выполнения работы:

1) Главная страница:

Фамилия	Возраст
Цыбаева Арина Олеговна	20
Куликов Артем Васильевич	40
Чужакова	32
ofis	23
Alisa	9

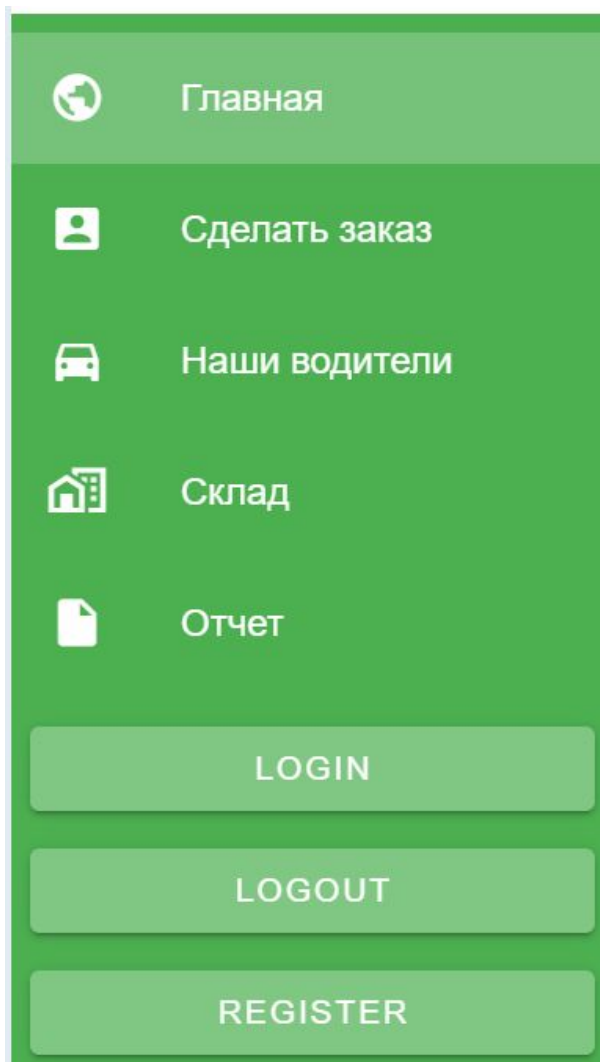
Количество заказов в зависимости от дня недели:

День недели	Процентная занятость
1	11 %
2	55 %
5	22 %
6	11 %



Топ-3 самых сдаваемых категорий мусора:

Стекло
Мягкий пластик
Компост

2) Выпадающее меню:



3) Создание заказа:

  Эко Такси

Создать заказ

Масса мусора

200

Цена

300

Дата

2020-06-15

Водитель

18

Клиент

1

Категория

2

СОЗДАТЬ

4) Список водителей:

Эко Такси		
Наши водители		
Фамилия	Возраст	
Маркелов	35	УВОЛИТЬ
Завъялова	29	УВОЛИТЬ
Новикова	21	УВОЛИТЬ

Авторизована, как администратор, поэтому могу добавить водителя, а также уволить.

Добавление водителя:

Добавить водителя	
Фамилия	Любимова
Возраст	29
Email	fnfh@gmail.com
Телефон	48294588
Модель и марка машины	Lexus 1283
Номер машины	13jkd23
ДОБАВИТЬ	

Проверка, видим, что новый водитель появился (последняя строка):

Наши водители		
Фамилия	Возраст	
Маркелов	35	УВОЛИТЬ
Завъялова	29	УВОЛИТЬ
Новикова	21	УВОЛИТЬ
Любимова	29	УВОЛИТЬ



Уволим последнего водителя, нажав кнопку уволить:

Наши водители

Фамилия	Возраст	
Маркелов	35	УВОЛИТЬ
Завьялова	29	УВОЛИТЬ
Новикова	21	УВОЛИТЬ

Водитель удален из базы данных.

- 5) Склад. Здесь можно принять заказ (администратору), а также списать заказ (сдать на переработку на завод). Выводится количество и наименования категорий, хранящихся на складе:

  Эко Такси		
Склад		
ПРИНЯТЬ ЗАКАЗ СПИСАТЬ ЗАКАЗ		
Хранится на складе:		
	Категория	Килограммы
	3	20
	5	8
	1	5
Справка:		
	Наименование	Название категории
	1	Компост
	2	Твердый пластик
	3	Мягкий пластик
	4	Макулатура
	5	Стекло
	6	Тетра-Пак
	7	Металл
	8	Лампочки и батарейки

- 6) Принятие заказа. Выводятся все существующие заказы, необходимо ввести номер заказа для добавления мусора на склад.

Добавить заказ на склад

Номер заказа

ДОБАВИТЬ

ВЕРНУТЬСЯ НА СКЛАД

Номер заказа	Категория	Масса	Дата	Водитель
141	Стекло	2	2020-05-25	Маркелов
142	Макулатура	12	2020-05-25	Маркелов
143	Мякий пластик	30	2020-05-25	Маркелов
144	Мякий пластик	20	2020-05-25	Маркелов
145	Твердый пластик	200	2020-05-25	Завьялова
146	Твердый пластик	200	2020-06-15	Новикова
10	Компост	5	2020-05-17	Завьялова

7) Списание заказа со склада. Заказы списываются с помощью кнопки “Списать”:

ВЕРНУТЬСЯ НА СКЛАД

Списать заказ со склада

Выберите заказ, который хотите списать:

Номер заказа	
1	СПИСАТЬ
10	СПИСАТЬ
11	СПИСАТЬ
12	СПИСАТЬ
12	СПИСАТЬ
143	СПИСАТЬ
144	СПИСАТЬ

Спишем заказ №144(последняя строка):

Списать заказ со склада

Выберите заказ, который хотите списать:

Номер заказа	
1	СПИСАТЬ
10	СПИСАТЬ
11	СПИСАТЬ
12	СПИСАТЬ
12	СПИСАТЬ
143	СПИСАТЬ
146	СПИСАТЬ

Как мы видим, заказ больше не выводится в списке заказов, хранящихся на складе.

8) Отчет.

На данной странице выводится отчетность за месяц в соответствии с индивидуальным заданием:

Отчет за текущий месяц			
Количество забранного мусора (кг)	Количество сданного мусора (кг)	Количество заказов	Доход
212	12	2	336
Посмотреть активность водителя за определенную дату			
Введите фамилию и дату через пробел			
НАЙТИ			
Заказы:			
Водитель	Дата	Клиент	Стоимость
Маркелов	2020-05-25	Цыбаева Арина Олеговна	200
Маркелов	2020-05-25	Куликов Артем Васильевич	200

Также можно найти все заказы водителя за определенную дату:

Поиск только по фамилии водителя:

Посмотреть активность водителя за определенную дату

Введите фамилию и дату через пробел
Маркелов

НАЙТИ

Заказы:

Водитель	Дата	Клиент	Стоимость
Маркелов	2019-10-24	Вальтов Добрыня Ильич	4
Маркелов	2020-06-12	Куликов Артем Васильевич	36
Маркелов	2020-05-25	Куликов Артем Васильевич	100

Поиск по фамилии водителя и по дате:

Посмотреть активность водителя за определенную дату

Введите фамилию и дату через пробел
Завъялова 2020-05-14

НАЙТИ

Заказы:

Водитель	Дата	Клиент	Стоимость
Завъялова	2020-05-14	Куликов Артем Васильевич	200

Страница logout:


localhost:8080/#/logout

Эко Такси

Вы вышли из системы!

ВОЙТИ

Страница регистрации:

 Эко Такси

Регистрация

Фамилия

Возраст

Email

Телефон

Адрес

ЗАРЕГИСТРИРОВАТЬСЯ

Вывод:

В ходе работы были получены практические навыки и умения реализации web-сервисов средствами Django Rest Framework, Vue.js, Vuetify. Был реализован сайт в соответствии с практическим заданием.