ViewSet и Router

Рассмотрим использование класса ViewSet для создания общего представления для API запросов.
В DRF есть класс ModelViewSet, который позволяет использовать внутренние настройки этого класса для создания представлений для всех типов запросов. Изменим файл views.py

```python
from rest_framework import viewsets
class MenViewSet(viewsets.ModelViewSet):
    queryset = Men.objects.all()
    serializer_class = MenSerializer
```

Достаточно указать что все поля берутся из нашей модели и для преобразований используется сериализатор, который сейчас тоже завязан на нашу модель.

В URL рекомендуется указывать какие конкретно методы работают при вызове того или иного адреса.

```python
urlpatterns = [
    path('', views.MenViewSet.as_view({'get': 'list'})),
    path('<int:pk>/', views.MenViewSet.as_view({'put': 'update', 'delete': 'destroy', 'get': 'retrieve'})),
]
```

Но организовывать и прописывать такие маршруты не совсем верно, т.к. они взаимодействуют с одним представлением. Для этого можно использовать роутер

Роутеры (routers) для viewsets

Для начала необходимо импортировать роутер

```
from rest_framework import routers
```

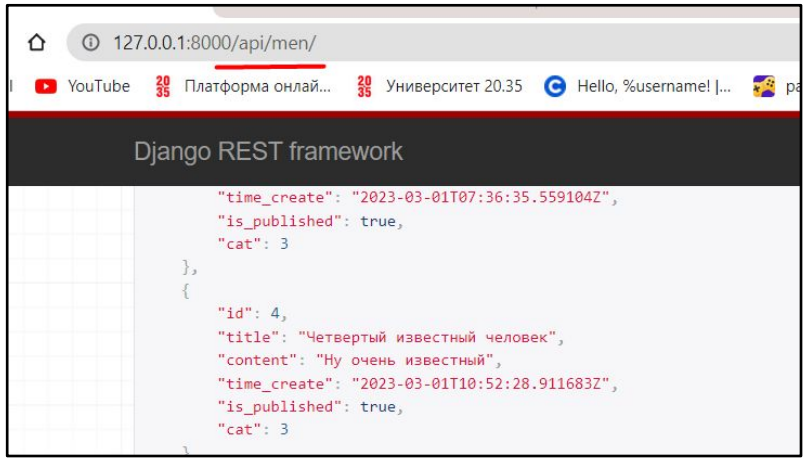
Далее создадим объект простого роутера (SimpleRouter) и связать этот роутер с представлением

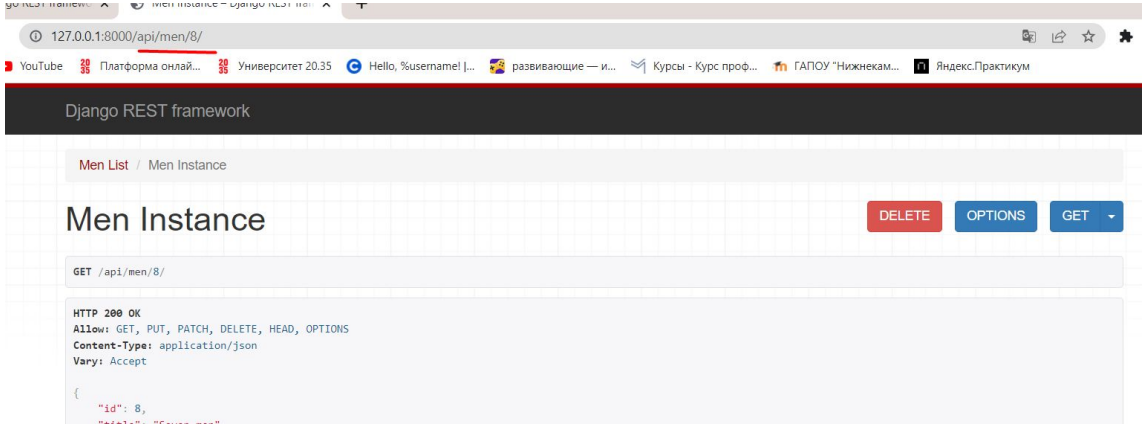
```
router = routers.SimpleRouter()

router.register(r'men', views.MenViewSet)
```

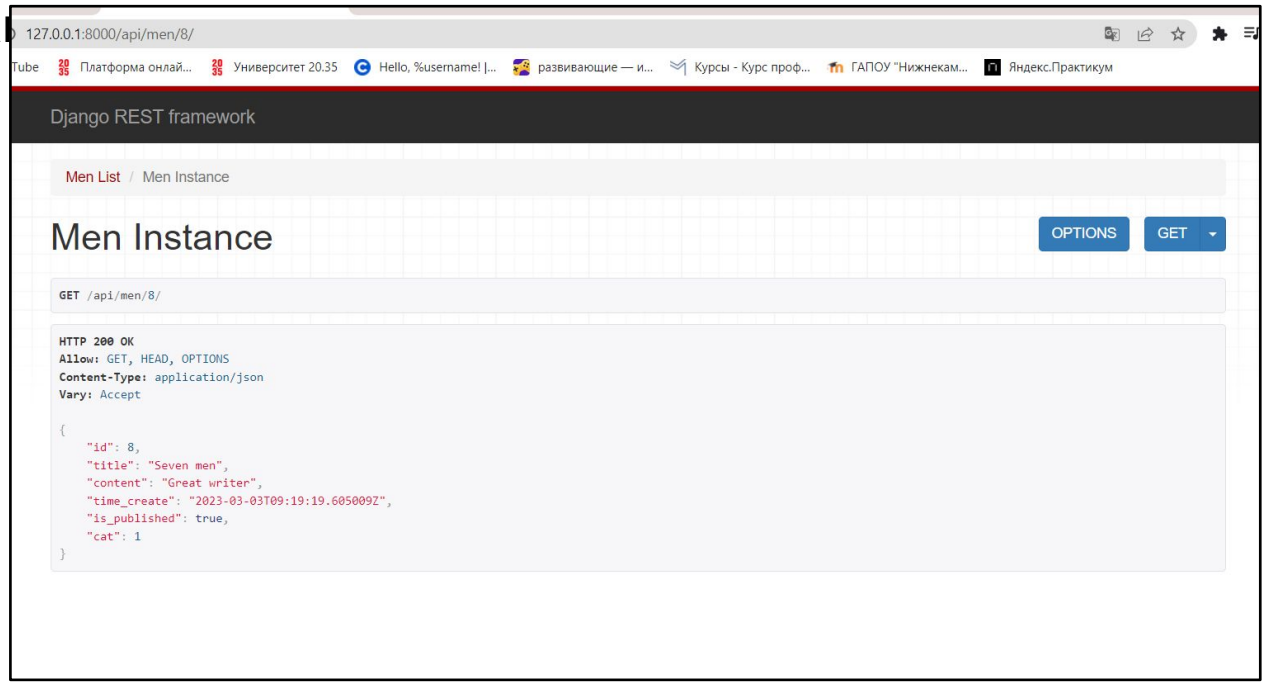
Все что нам остается это прописать URL для запроса этих маршрутов. Подключим их с помощью метода include

```
urlpatterns = [
    path('', include(router.urls))
]
```

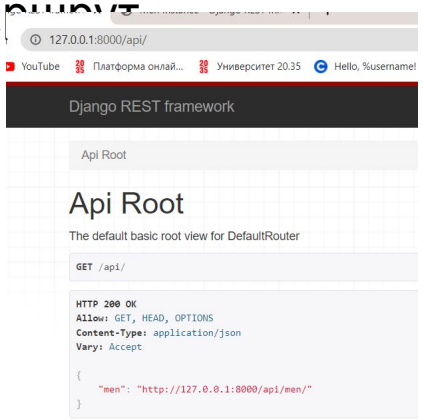




Изменив базовый класс на ReadOnlyModelViewSet мы получим представления, в которых можно только читать дан



Существует 2 типа роутеров это SimpleRouter и DefaultRouter. Они практически идентичны по функционалу, единственное отличие – это то, что DefaultRouter дополнительно создает корневой маршрут



Задание

Предлагаю переделать API с питомцами на продажу. На сколько вы помните там была возможность получить общий список питомцев, добавить питомца на продажу, продать его (изменив его внутренний статус). Ну и, как результат, со временем его можно удалить из базы.
Данные о питомцах содержали следующую информацию: Категория, Имя, Фото, Тип животного и Статус. Категория – это доп. таблица с данными о существующих категориях. Тип животного – также попадает из доп. таблицы. Статус – доп. таблица.

Также была информация о заказах. Их можно добавить, получить о них инфу и удалить. В таблице с заказом есть информация о Питомце, которого заказали, Количество, Дата продажи, Статус, Выполнен (Булевое поле). Статусы не совпадают со статусами питомцев. У Заказа статусы: размещен, обработан, выполнен.