

**"НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО"**

Факультет «Инфокоммуникационных технологий»  
Направление подготовки «09.03.03 Прикладная информатика»  
Бакалаврская программа «Мобильные и сетевые технологии»

**ОТЧЁТ**  
**по лабораторной работе №3**

**по дисциплине «Web-программирование»**

**Тема: «Реализация серверной части приложения средствами  
Django и Django Rest Framework»**

**Выполнил:**

\_\_\_\_\_ / Шугинин Ю. А. (К33402)

**Проверил:**

\_\_\_\_\_ / Говоров А. И.

**Дата: 18.03.2022**

**Санкт-Петербург**  
**2022**

**Цель:** овладеть практическими навыками и умениями реализации web-сервисов средствами Django и Django Rest Framework.

## Ход работы.

Выбран вариант с созданием сайта прогноза погоды.

### Описание модели данных:

User (id, username, password, email, first\_name, last\_name, favourite\_cities), где favourite\_cities – many to many поле с таблицей City через таблицу Favourite.

City (id, name)

Favourite (id, user\_id (FK), city\_id (FK))

### Описание контроллеров в файле views.py:

class UserAPIView() – получение данных о пользователе;

class CityAPIView() – получение данных о городах в базе;

class CityCreateView() – создание города в базе;

class FavouriteAPIView() – получение данных об избранных городах пользователя;

class FavouriteCreateView() – добавление города в избранные;

class FavouriteDeleteView() – удаление города из избранного.

### Описание сериализаторов в файле serializers.py:

class CitySerializer() – для получения данных о городах в таблице City.

class CityCreateSerializer() – для добавления нового города в таблицу City.

class FavouriteSerializer() – для получения данных об избранных городах в таблице Favourite.

class FavouriteCreateSerializer() – для добавления новой записи пользователь-город в таблицу Favourite.

class UserSerializer() – для получения данных о пользователях (используется SlugRelatedField для отображения списка названий избранных городов в поле favourites\_cities, а не их id).

## Документирование. Swagger:

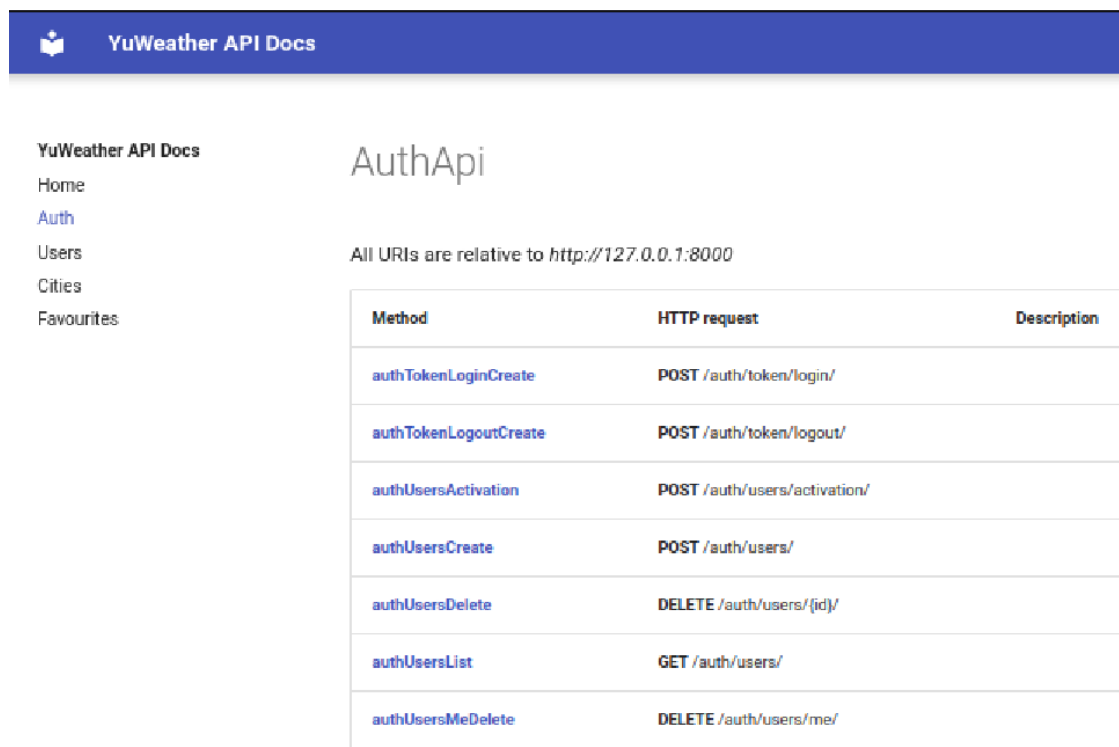
auth			⌵
POST	/auth/token/login/	auth_token_login_create	🔒
POST	/auth/token/logout/	auth_token_logout_create	🔒
GET	/auth/users/	auth_users_list	🔒
POST	/auth/users/	auth_users_create	🔒
POST	/auth/users/activation/	auth_users_activation	🔒
GET	/auth/users/me/	auth_users_me_read	🔒
PUT	/auth/users/me/	auth_users_me_update	🔒
PATCH	/auth/users/me/	auth_users_me_partial_update	🔒
DELETE	/auth/users/me/	auth_users_me_delete	🔒
POST	/auth/users/resend_activation/	auth_users_resend_activation	🔒
POST	/auth/users/reset_password/	auth_users_reset_password	🔒
POST	/auth/users/reset_password_confirm/	auth_users_reset_password_confirm	🔒
POST	/auth/users/reset_username/	auth_users_reset_username	🔒



Рисунок 1 – Swagger

## Документирование. MkDocs:

Документация создана с помощью библиотеки **openapi-generator**.



#### YuWeather API Docs

[Home](#)

[Auth](#)

[Users](#)

[Cities](#)

[Favourites](#)

## UsersApi

All URIs are relative to `http://127.0.0.1:8000`

Method	HTTP request	Description
<a href="#">usersList</a>	<b>GET</b> /users/	

#### YuWeather API Docs

[Home](#)

[Auth](#)

[Users](#)

[Cities](#)

[Favourites](#)

## CitiesApi

All URIs are relative to `http://127.0.0.1:8000`

Method	HTTP request	Description
<a href="#">citiesCreateCreate</a>	<b>POST</b> /cities/create	
<a href="#">citiesList</a>	<b>GET</b> /cities/	

#### YuWeather API Docs

[Home](#)

[Auth](#)

[Users](#)

[Cities](#)

[Favourites](#)

## FavouritesApi

All URIs are relative to `http://127.0.0.1:8000`

Method	HTTP request	Description
<a href="#">favouritesCreateCreate</a>	<b>POST</b> /favourites/create	
<a href="#">favouritesDeleteDelete</a>	<b>DELETE</b> /favourites/delete	
<a href="#">favouritesList</a>	<b>GET</b> /favourites/	

Рисунок 2 – MkDocs

**Вывод:** в процессе выполнения лабораторной работы были получены практические навыки реализации сайта с использованием фреймворка Django и Django Rest Framework. Реализованы endpoint'ы, необходимые для корректной работы интерфейсов выбранного варианта (сайта с прогнозом погоды). Изучены и использованы методы сериализации. Реализованы регистрация, логин и изменение данных пользователя с помощью библиотеки Djoser. Освоено документирование API средствами Swagger и MkDocs.