1. О сервисе

Сервис предоставляет пользователю возможности регистрации и авторизации. А также следующие возможности:

- отправить одному пользователю заявку в друзья другому;
- принять/отклонить пользователю заявку в друзья от другого пользователя;
- посмотреть пользователю список своих входящих/исходящих и принятых заявок в друзья;
- посмотреть пользователю список своих друзей;
- получит пользователю статус дружбы с каким-то другим пользователем;
- удалить пользователю другого пользователя из своих друзей.

Сервис использует две сущности: Пользователь (User) и Заявка в друзья (Friendship) (рисунок 1).

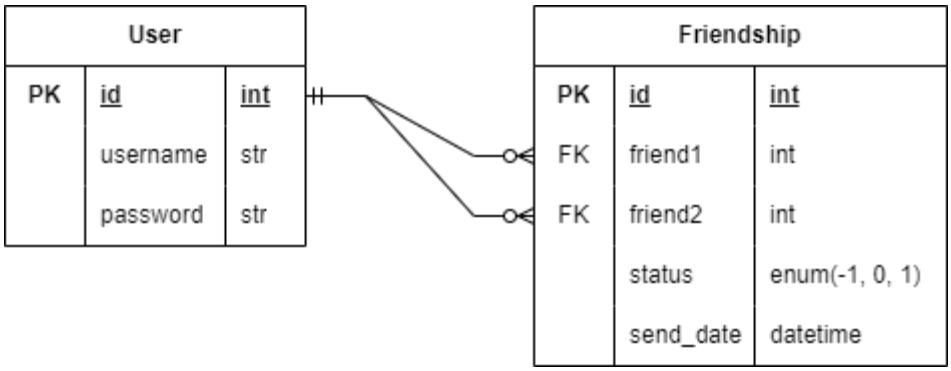

Рисунок 1 – ERD-диаграмма сущностей и их ключевых полей

Поле status сущности Friendship хранит статус заявки в друзья: -1 (пользователь1 отправил заявку пользователю2), 1 (пользователь1 получил заявку от пользователя2) и 0 (пользователь1 принял заявку в друзья от пользователя2).

Сущность Friendship неявно переопределяет функционал симметричного ManyToMany поля Django-модели. При отправке пользователем1 заявки в друзья пользователю2 в БД создаются две симметричные записи о заявках: пользователь1-пользователь2 (статус=-1) и пользователь2-пользователь1 (статус=1). При одобрении полученной заявки

статусы обеих заявок меняются на 0. Такой подход обеспечивает более быстрый и простой доступ к данным, однако увеличивает объем хранимой информации.

2. Запуск сервиса

Для запуска проекта необходимо клонировать репозиторий проекта в интересующую папку.

Затем создать и запустить контейнер docker на основе файлов docker и docker-compose в проекте.

После чего функционал сервиса будет доступен на htttp://localhost/. По умолчанию для удобства демонстрации сервис открывается в формате Django REST Framework API, который позволяет работать с запросами к сервису в удобном графическом формате (рисунок 2).

Рисунок 2 – Интерфейс Django REST Framework API

3. Конечные точки

3.1. Регистрация

Регистрация доступна любым пользователям через POST запрос на адрес htttp://localhost/register/. В теле запроса передаются имя пользователя и его пароль.

Сущность пользователя реализована с помощью переопределения Django-пользователя и наследует от него валидаторы для username и password (см. рисунок 3).

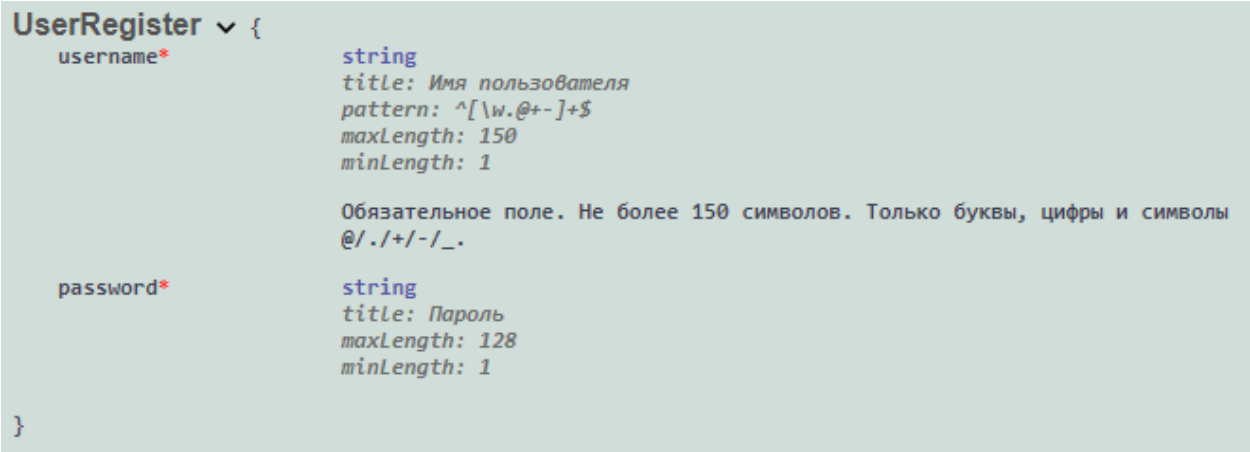

Рисунок 3 – Сериализатор для регистрации пользователя

При удачной регистрации пользователя от сервера приходит ответ со статусом 201 и данными имени и пароля пользователя (см. рисунок 1).

Если данные отправленные на сервер не прошли валидацию, то от сервера приходит ответ со статусом 400, в теле которого указаны замечания по содержимому не прошедших валидацию полей (см. пример таблица 1).

Таблица 1 – Пример запроса-ответа на регистрацию

Запрос	**Ответ**
Позитивный 201	

{
"username": "Mixail",
"password": "123"
}
	{
	"username": "Mixail",
	"password":
	"pbkdf2_sha256$600000$ZEwU7mF2KpsQK448CZbNaf$Ay7KEhtW2Mj2zouvCpMWuFv0BsFzS5rrhPsJVjFwH6s="
	}

Негативный 400	

{
"username": "Mixail^",
"password": ""
}
	{
	"username": [
	"Введите правильное имя пользователя. Оно может содержать только буквы, цифры и знаки @/./+/-/_."
	],
	"password": [
	"Это поле не может быть пустым."
	]
	}

3.2. Авторизация

Вход и выход из аккаунта выполнены с помощью встроенных инструментов Django REST framework и доступны пользователям по адресам htttp://localhost/api-auth/login/ и htttp://localhost/api-auth/logout/.

В теле POST запроса на авторизацию передаются имя пользователя и пароль (см. рисунок 1).

Далее все описанные функции предоставляются только авторизованным пользователям. При отсутствии в запросе заголовка авторизации в запросе сервер отдает ответ со статусом 403 и телом: { "detail": "Учетные данные не были предоставлены."}.

3.3. Получение списка друзей

Получение списка друзей пользователя происходит по GET запросу на адрес htttp://localhost/friends/. На запрос сервер отправляет ответ со статусом 200 и телом, в котором указан массив пользователей с их уникальными идентификаторами и именами (см. пример таблица 2).

Таблица 2 – Пример запроса-ответа на получение списка друзей

Запрос	**Ответ**
Позитивный 200	
	[{"id":2,"username":"Василий"},{"id":4,"username":"Екатерина"},{"id":5,"username":"Татьяна"}]

3.4. Отправка заявки в друзья

Отправка заявки в друзья также доступна только авторизованным пользователям по POST запросу на адрес htttp://localhost/friends/. В теле запроса обязательно передается уникальный идентификатор пользователя, которому нужно отправить заявку в друзья (см. таблицу 3). В ответ на запрос сервер отправляет данные заявки в друзья: ID, ID пользователя, которому отправлена заявка, статус заявки и дату-время отправки заявки (см. рисунок 4).

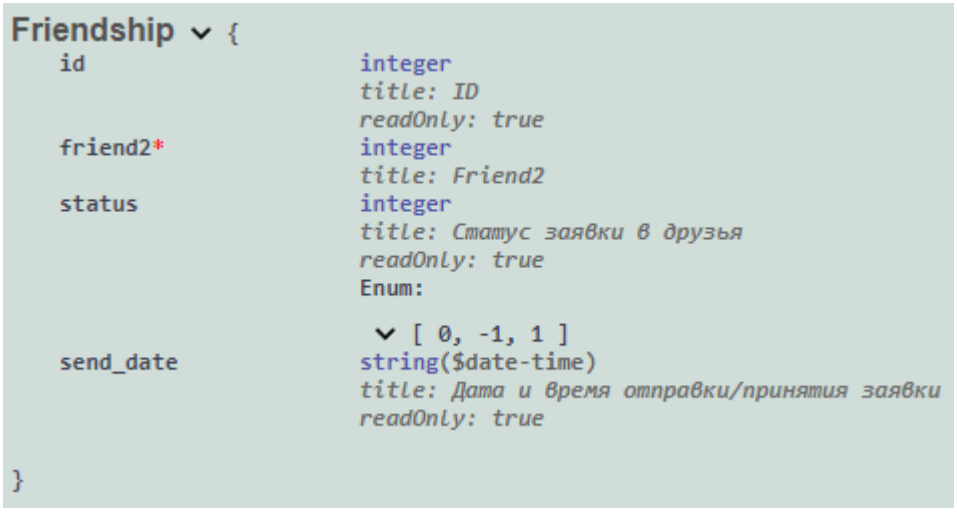

Рисунок 4 – Сериализатор заявки в друзья

Таблица 3 – Пример запроса-ответа на отправку заявки в друзья

Запрос	Ответ
Позитивный 200 Пользователь 5 ранее уже отправлял заявку в друзья текущему пользователю	
{"friend2": 5}	{"id":5,"friend2":5,"status":0,"send_date":"2023-05-08T15:44:06.689794Z"}
Позитивный 201 Пользователь отправил новую заявку в друзья	
{"friend2": 3}	{"id":13,"friend2":3,"status":-1,"send_date":"2023-05-08T15:44:16.429339Z"}
Позитивный 208 Пользователь ранее уже отправлял заявку в друзья этому пользователю	
{"friend2": 4}	{"id":3,"friend2":4,"status":-1,"send_date":"2023-05-08T16:00:40.388022Z"}
Позитивный 208 Пользователи уже являются друзьями	
{"friend2": 2}	{"id":1,"friend2":2,"status":0,"send_date":"2023-05-08T16:00:40.388022Z"}
Негативный 400	
{"friend2": 15}	{"friend2": ["Недопустимый первичный ключ \"15\" - объект не существует."]}
Негативный 422	
{"friend2": 1}	{"detail": "Нельзя Отправить Заявку Самому Себе."}

3.5. Получение статуса дружбы

Для получения статуса дружбы с пользователем авторизованный пользователь должен отправить GET запрос на адрес htttp://localhost/friends/{id}/ с указанием уникального идентификатора интересующего пользователя. В ответ сервер пришлет данные в формате, представленном на рисунке 5.

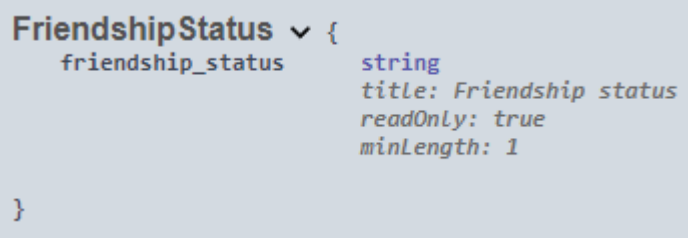

Рисунок 5 – Сериализатор статуса дружбы

В таблице 4 представлены примеры запросов на получение статуса дружбы.

Таблица 4 – Пример запроса-ответа на получение статуса дружбы

Запрос	Ответ
Позитивный 200	
GET /friends/2/	{"friendship_status":"Вы друзья."}
Позитивный 200	
GET /friends/4/	{"friendship_status":"Вы подали заявку на дружбу."}
Позитивный 200	
GET /friends/5/	{"friendship_status":"Пользователь отправил Вам заявку в друзья."}
Позитивный 200	
GET /friends/3/	{"friendship_status":"Вы еще не друзья."}
Позитивный 200	
GET /friends/1/	{"friendship_status":"Это вы."}
Негативный 404	
GET /friends/15/	{"detail": "Страница Не Найдена."}

3.6. Удаление из друзей

Удаление из друзей происходит по DELETE запросу на адрес htttp://localhost/friends/{id}/ с указанием уникального идентификатора интересующего пользователя (текущий пользователь берется из заголовка

авторизации). Удачный запрос вернет ответ со статусом 204 и без тела, даже если с указанным пользователем текущего ничего не связывало (не были ни друзьями, ни отправляли друг другу заявок в друзья).

Если пользователя с указанным id не существует, то сервер вернет ответ 400 {"detail": "Страница Не Найдена."}.

3.7. Получение списка заявок в друзья

Получение списка заявок в друзья происходит через отправку GET запроса на адрес htttp://localhost/friendship/?status=. В параметре status указываются: -1 для получения списка отправленных заявок; 0 для получения списка принятых заявок; 1 для получения списка полученных заявок. Запрос без указания параметра status вернет список отправленных заявок.

Ответ сервера на полученный запрос состоит из массива заявок в друзья (см. рисунок 4). В таблице 5 приведены примеры работы данной функции.

Таблица 5 – Пример запроса-ответа на получение списка заявок в друзья

Запрос	Ответ
Позитивный 200	
GET /friendship/?status=1	[{"id":5,"friend2":5,"status":1,"send_date":"2023-05-08T16:45:21.555129Z"}]
Позитивный 200	
GET /friendship/?status=-1	[{"id":3,"friend2":4,"status":-1,"send_date":"2023-05-08T16:45:21.555129Z"}]
Позитивный 200	
GET /friendship/?status=0	[{"id":1,"friend2":2,"status":0,"send_date":"2023-05-08T16:45:21.555129Z"}]
Негативный 400	
GET /friendship/?status=12	{"status":["Выберите корректный вариант. 12 нет среди допустимых значений."]}

3.8. Принятие заявки

Для того, чтобы авторизованному пользователю принять поступившую заявку в друзья ему необходимо отправить GET запрос на адрес htttp://localhost/friendship/{id}/ с указанием уникального идентификатора заявки. Удачный запрос вернет статус 200 с телом, в котором указаны данные заявки (см. рисунок 4).

В таблице 6 представлены примеры запросов на принятие заявки.

Таблица 6 – Пример запроса-ответа на принятие заявки

Запрос	Ответ
Позитивный 200	
GET /friendship /5/	{"id":5,"friend2":5,"status":0,"send_date":"2023-05-08T17:28:52.685048Z"}
Позитивный 400	
GET /friendship /1/	{"detail":"Попытка доступа к чужой или неактуальной заявке."}
Позитивный 404	
GET /friends/66/	{"detail":"Страница не найдена."}

3.9. Отклонение заявки в друзья

Для того, чтобы пользователю отклонить поступившую заявку в друзья ему необходимо отправить DELETE запрос на адрес htttp://localhost/friendship/{id}/ с указанием уникального идентификатора заявки. Удачный запрос вернет статус 204 без тела. Не удачные запросы аналогичны запросам на принятие заявки.

3.10. Дополнительно

Конечная точка htttp://localhost/admin/ дает доступ к админ-панели Django для управления данными в БД.

Конечная точка htttp://localhost/swagger/ дает доступ к swagger-панели автособираемой документации.