Python

Цель:

Реализовать сервис, который принимает и отвечает на HTTP запросы.

Функционал:

1. В случае успешной обработки сервис должен отвечать статусом 200, в случае любой ошибки — статус 400.
2. Сохранение всех объектов в базе данных.
3. Запросы:
    a. GET /city/ — получение всех городов из базы;
    b. GET /city//street/ — получение всех улиц города; (city_id — идентификатор города)
    c. POST /shop/ — создание магазина; Данный метод получает json с объектом магазина, в ответ возвращает id созданной записи.
    d. GET /shop/?street=&city=&open=0/1 — получение списка магазинов;
        i. Метод принимает параметры для фильтрации. Параметры не обязательны. В случае отсутствия параметров выводятся все магазины, если хоть один параметр есть , то по нему выполняется фильтрация.
        ii. Важно!: в объекте каждого магазина выводится название города и улицы, а не id записей.
        iii. Параметр open: 0 - закрыт, 1 - открыт. Данный статус определяется исходя из параметров «Время открытия», «Время закрытия» и текущего времени сервера.

Объекты:

Магазин:

- Название
- Город
- Улица
- Дом
- Время открытия
- Время закрытия

Город:

- Название

Улица:

- Название
- Город

!! Замечание: поле id у объектов не указаны, но подразумевается что они есть.
!! Важно: Выстроить связи между таблицами в базе данных.

Инструменты:

- Фреймворк для обработки http запросов Django + Django Rest Framework
- Реляционная БД (PostgreSQL - предпочтительно, MySQL и тд)
- Запросы в базу данных через ORM (ORM на выбор).

Сдача задания:

Ссылка на репозиторий, который содержит ваш проект и README:

- Фамилия Имя
- Тестовое задание на Python
- Описание проекта
- Подготовительные действия (установки, настройки и т.д) для успешной работы проекта *
- Информация о доступах (логины/пароли и т.д.) *
- Описание, как запустить ваш проект

* если требуется