Intellify Test Task for Django

The purpose behind this task is to understand your knowledge about system design thought process as well as evaluate your skillset regarding Python, Django, DRF and ability to document written code.

* use PostgreSQL database
* use Django
* extra points for async code
* prepare setup instructions

Section 1. Django

1. Setup DRF application
2. Implement JWT authentication
3. Implement HTTP POST mechanisms to create new Django user by **email/password**
    1. Setup model schema so that each user got one "organization project"
    2. To each "organization project" it is possible to attach multiple subprojects - "organizations objects"
    3. To each "organization object" there is a possibility to create *datapoints* where to store time series data from multiple datasources
    4. To each "organization project " it is possible to attach users with 3 permission levels
        1. admin – can access all "organizations" data (project details, object details, timeseries data, user in projects etc.)
        2. moderator – can access only one "organization project" data (every object as well)
        3. simple user – can access only one "organization object" data
4. Create two "organization projects", for each create 3 "oraganization objects" and multiple users with random permission types.
5. Generate random timeseries data for each "organization objects" *datapoint*
6. Built http GET endpoint where authenticated users can access all datapoint IDs
7. Built http POST endpoint where authenticated users can give additional information to each datapoint (datapoint name, datapoint creation datetime, "organization project name","organization object name"
8. Built http GET endpoint where authenticated users can access timeseries data by datapoint id
9. Built http GET endpoint where authenticated users can access **aggregated** timeseries data by datapoint id (for example hourly, daily, monthly average values)
10. Built http POST endpoint where user can store configuration in JSON format
11. Built http PUT endpoint where user can edit individual section of previously saved configuration JSON
12. Setup secure settings for deployment

* additional extra bonus cookies for using REDIS as database cache
* additional extra bonus cookies for using Django 3.1 with ASYNC where possible
* additional extra bonus cookies for clear and comprehensive documentation
* additional extra bonus cookies for Section 3

Section 2. Python

1. Setup MQTT broker (mosquitto or other)
2. Build python data generator that will post random timeseries data to MQTT broker
3. Implement ACL editable from Djagno application, so that there is a possibility to create new datasources with credentials
4. Build python MQTT data consumer, that will consume data and put in into a database

* additional extra bonus cookies for building with MQTTs

Section 3. Docker

1. Create Dockerfile for application done 1.section deployment
2. Create docker-compose file where are included
    1. Build command for 1. point to build Django app with gunicorn as server
    2. Setup nginx to serve static and media files for the application
    3. Use PostgreSQL
3. Use the enviroment variable to configure Django:
    1. Application key
    2. Create superuser
    3. Setup database details
    4. Switch from to/from debugging
4. Buill and populate database from 2.section task