Backend Coding Assignment

Create a small Django REST API serving vector features in a GeoJSON compatible format.

The main goal of this assignment is to assess how you structure your code and workflow.

The Django server should offer the following:

- A **CRUD** REST API for GeoJSON feature objects. A real **FeatureCollection** object does not allow for paging (limiting the amount of features returned), so a result page may look like this:

```
{
    “next”: “http://example.com/features/?page=3”
    “prev”: “http://example.com/features/?page=1”
    “results”: [<GeoJSON Feature Object>, ...]
}
```

- The features should be able to filtered by a boundingbox (e.g. you are only viewing part of the map) and the results should be paged (100 features per request)
- The API should use json web tokens for authorization
- A **html** frontend page to visualize (using OpenLayers e.g.) the data (no feature editing required, but you are free to implement this)
- A second frontend page where you show the list of features and allow you to edit the properties (POST/PATCH) of the features (name, color e.g.). Bonus points: use basic styling (Bootstrap etc.) for this page.
- Design how you would store and query the feature data using a PostGIS database or, alternatively, implement it in the Django API to store the data in a PostGIS database.
- Bonus points: provide a **Dockerfile** and **docker-compose.yml** to build and test your application easily into a docker container.

Deliverables:
A link to a **public** git repository (github.com, bitbucket.org or gitlab.com)
A zip archive containing the following:

- All your source code and other assets required to run and build the app
- A short README.md with instructions on how to run and build the app