Django Python Machine Test

- **Consider the following scenario where we have 3 entities in our system**.
  1. User
  2. Client
  3. Project

  We have the number of users registered in our system.
  You can use Django's default admin template to create/register users but not other entities or you can make REST APIs for users as well if you want.

- **You have to perform the following tasks :**
  1. Register a client
  2. Fetch clients info
  3. Edit/Delete client info
  4. Add new projects for a client and assign users to those projects.
  5. Retrieve assigned projects to logged-in users.

- **Things to consider :**
  1. The system has many users.
  2. The system has many clients.
  3. A client can have many projects
  4. Multiple clients cannot be part of the same project.
  4. A single project can be assigned to many users.

Technical Requirements :

1. Use Django REST framework
2. Use **MySQL** or **Postgres** as DB (**No SQLite**)

Given are the examples of REST APIs you have to build.

➢ **List of all clients**
GET /clients/

```
[
        {
                'id' : 1,
```

```
        'client_name' : 'Nimap',
        'created_at' : '2019-12-24T11:03:55.931739+05:30',
        'created_by' : 'Rohit'
    },
    {
        'id' : 2,
        'client_name' : 'Infotech',
        'created_at' : '2019-12-24T11:03:55.931739+05:30',
        'created_by' : 'Rohit'
    }
]
```

➢ **Create a new client**
POST /clients/

```
Input = {
    'client_name' : 'company A'
}

Output = {
        'id' : 3,
        'client_name' : 'company A',
        'created_at' : '2019-12-24T11:03:55.931739+05:30',
        'created_by' : 'Rohit'
}
```

➢ Retrieve info of a client along with projects assigned to its users
GET /clients/:id

```
{
    'id' : 2,
        'client_name' : 'Infotech',
        'projects' : [
            {
                'id' : 1,
                'name' : 'project A'
            }
        ]
        'created_at' : '2019-12-24T11:03:55.931739+05:30',
        'created_by' : 'Rohit'
```

'updated_at' : '2019-12-24T11:03:55.931739+05:30',
}

➢ **Update info of a client**
PUT-PATCH /clients/:id

Input = {
'client_name' : 'company A'
}

Output = {
'id' : 3,
'client_name' : 'company A',
'created_at' : '2019-12-24T11:03:55.931739+05:30',
'created_by' : 'Rohit',
'updated_at' : '2019-12-24T11:03:55.931739+05:30'
}

➢ **DELETE /clients/:id**

The response status should be 204.

➢ **Create a new project**
POST projects/

Here you do not need to create any new user but assign already registered users.
retrieve client id from the url and assign it to a project.

Input = {
'project_name' : 'Project A',
‘client_id’: 1,
'users' : [1]
}

Output = {
'id' : 3,
'project_name' : 'Project A',

'client' : 'Nimap'
'users' : [
{
'id' : 1,
'name' : 'Rohit'
}
]
'created_at' : '2019-12-24T11:03:55.931739+05:30'
'created_by' : 'Ganesh'
}

➢ **List of all projects assigned to the logged-in user**
GET /projects/

```
[
    {
        'id' : 1,
        'project_name' : 'Project A',
        client_name : Client A',
        'created_at' : '2019-12-24T11:03:55.931739+05:30',
        'created_by' : 'Ganesh'
    },
    {
        'id' : 2,
        'project_name' : 'Project B',
        client_name : Client A',
        'created_at' : '2019-12-24T11:03:55.931739+05:30',
        'created_by' : 'Ganesh'
    }
]
```

➢ **DELETE /projects/:id**

The response status should be 204.
