**Microservices Architecture**

**Practical – 02**

**Aim:** Create a Django application to demonstrate CRUD for the registration form for an event in college.

**Description:**

1. **Django:**

Django is a free and open-source, Python-based web framework that follows the model–template–views architectural pattern.

1. **Django REST framework:**

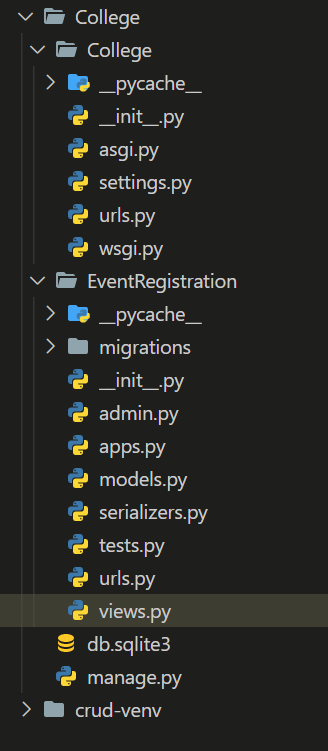
Django REST framework is a powerful and flexible toolkit for building Web APIs.

REST is a loosely defined protocol for listing, creating, changing, and deleting data on your server over HTTP. The Django REST framework (DRF) is a toolkit built on top of the Django web framework that reduces the amount of code you need to write to create REST interfaces.

1. **Postman API platform:**

Postman is an API client that makes it easy for developers to create, share, test and document APIs. With this open-source solution, users can create and save simple and complex HTTP/s requests, as well as read their responses.

**Project’s Files Structure:**



1. **urls.py :** A request in Django first comes to urls.py and then goes to the matching function in views.py.
2. **views.py:** Python functions in views.py take the web request from urls.py and give the web response to templates.
3. **Serializers.py:** Serializers in Django REST Framework are responsible for converting objects into data types understandable by javascript and front-end frameworks.
4. **Models.py:** Model is defined in Models.py file. This file can contain multiple models. Let's see an example here, we are creating a model Employee which has two fields first\_name and last\_name. The first\_name and last\_name fields are specified as class attributes and each attribute maps to a database column.

**Code:**

**NOTE:** The complete code is availible at my github repository:

<https://github.com/vishal8github/Django-ReST-API-for-college-event-registration-form>

1. view.py

from django.shortcuts import render

from rest\_framework import generics

from .serializers import \*

from .models import \*

#CRUD operations

*class* ListEventRegister(generics.ListAPIView):

    queryset = EventRegister.objects.all()

    serializer\_class = EventRegisterSerializer

*class* DetailEventRegister(generics.RetrieveUpdateAPIView):

    queryset = EventRegister.objects.all()

    serializer\_class = EventRegisterSerializer

*class* CreateEventRegister(generics.CreateAPIView):

    queryset = EventRegister.objects.all()

    serializer\_class = EventRegisterSerializer

*class* UpdateEventRegister(generics.UpdateAPIView):

    queryset = EventRegister.objects.all()

    serializer\_class = EventRegisterSerializer

*class* DeleteEventRegister(generics.DestroyAPIView):

    queryset = EventRegister.objects.all()

    serializer\_class = EventRegisterSerializer

**Output:**

