**Django Rest Framework**

**REST-Representational state transfer used to create a web service.**

**They are two types:**

1)Restful Api(stateful)

2)Restless Api(stateless)

**API:(Application programming interface)**

**Application:** It is like a software used to store data

**Programming:** It is used to tell a computer to perform a certain task.

**Interface**: It is used to communicate between two applications.

**Create a Django Rest Api:**

**Requirements**:

1)python(3.6,3.7,3.8)

2)Django

**Install rest\_framework** :

In command prompt

pip install django

pip install djangorestframework

**HTTP Methods:**

GET,POST,PUT,DELETE.

**Start Project:**

**In command prompt---**

C:\desktop>django-admin startproject EventRegistration **//project name**

**The project folder has-**

**\_\_init\_\_.py , asgi.py ,settings.py , urls.py , wsgi.py.**

C:\desktop>cd EventRegistration

C:\desktop\EventRegistration>python manage.py startapp EventApp //**APP(application)**

**EventApp has-**

**\_\_init\_\_.py, admin.py, apps.py, models.py, tests.py, views.py.**

**Configure EventRegistration\settings.py**

**Open settings.py file**

**Add a EventApp, rest\_framework.**

NSTALLED\_APPS = [  
 'django.contrib.admin',  
 'django.contrib.auth',  
 'django.contrib.contenttypes',  
 'django.contrib.sessions',  
 'django.contrib.messages',  
 'django.contrib.staticfiles',  
 'rest\_framework',  
 'EventApp',

]

**Create a Model or Event:**

**EventApp\models.py**

from django.db import models  
#model for event registration  
class EventRegistration(models.Model):  
 firstname=models.CharField(max\_length=10)  
 lastname=models.CharField(max\_length=10)  
 email=models.EmailField()  
 mobile=models.CharField(max\_length=10)  
 address=models.CharField(max\_length=50)  
 def \_\_str\_\_(self):  
 return self.firstname+" "+self.lastname

**To register model into admin site**

**EventApp\admin.py**

from django.contrib import admin  
from . models import EventRegistration  
# Register your models here.  
admin.site.register(EventRegistration)

**Apply 3 operations in command prompt**

1)C:/desktop/EventRegistration>python manage.py migrate

2)C:/ desktop/EventRegistration>python manage.py makemigrations EventApp

2)C:/desktop/EventRegistration>python manage.py sqlmigrate EventApp 0001

**Run a server:**

C:/desktop/EventRegistration>python manage.py runserver 8090

8090 is port number

Note: In any manipulation in project to re-run the server

**Create a Admin(superuser)**

C:/desktop/EventRegistration>python manage.py createsuperuser

Enter username , email –id, password.

Open the browser-

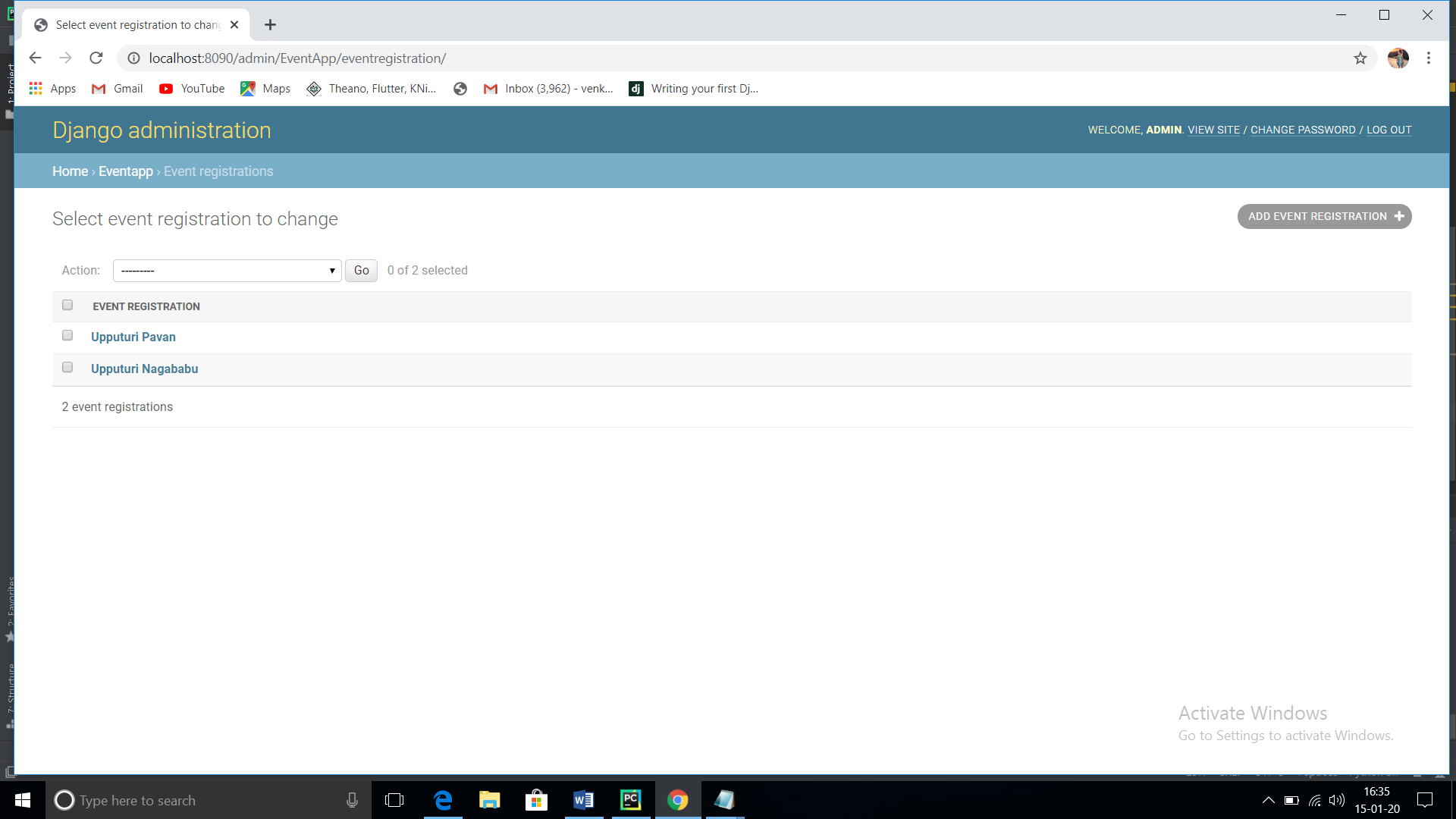
In url enter the below address.

<http://localhost:8090/admin/>

it asks username and password

admin site is open see **EventApp** has **EventRegistration** model is created.

--**To Click Add EventRegistration button add the Registration**



**RESTFRAMEWORK API :**

**EventApp\views.py**

**It returns json format,api format**

from django.shortcuts import render,get\_object\_or\_404  
from django.http import JsonResponse  
from .models import EventRegistration  
#from rest\_framework import generics  
  
  
import EventApp  
  
def EventRegistration\_list(request):  
 polls=EventRegistration.objects.all()  
 data={"results":list(polls.values('firstname','lastname','email','mobile','address'))}  
 return JsonResponse(data)

**EventApp\serializer.py**

from rest\_framework import serializers  
from .models import EventRegistration  
from django.contrib.auth.models import User  
class EventSerializer(serializers.ModelSerializer):  
 class Meta:  
 model=EventRegistration()  
 fields='\_\_all\_\_'

**EventApp\apiviews.py**

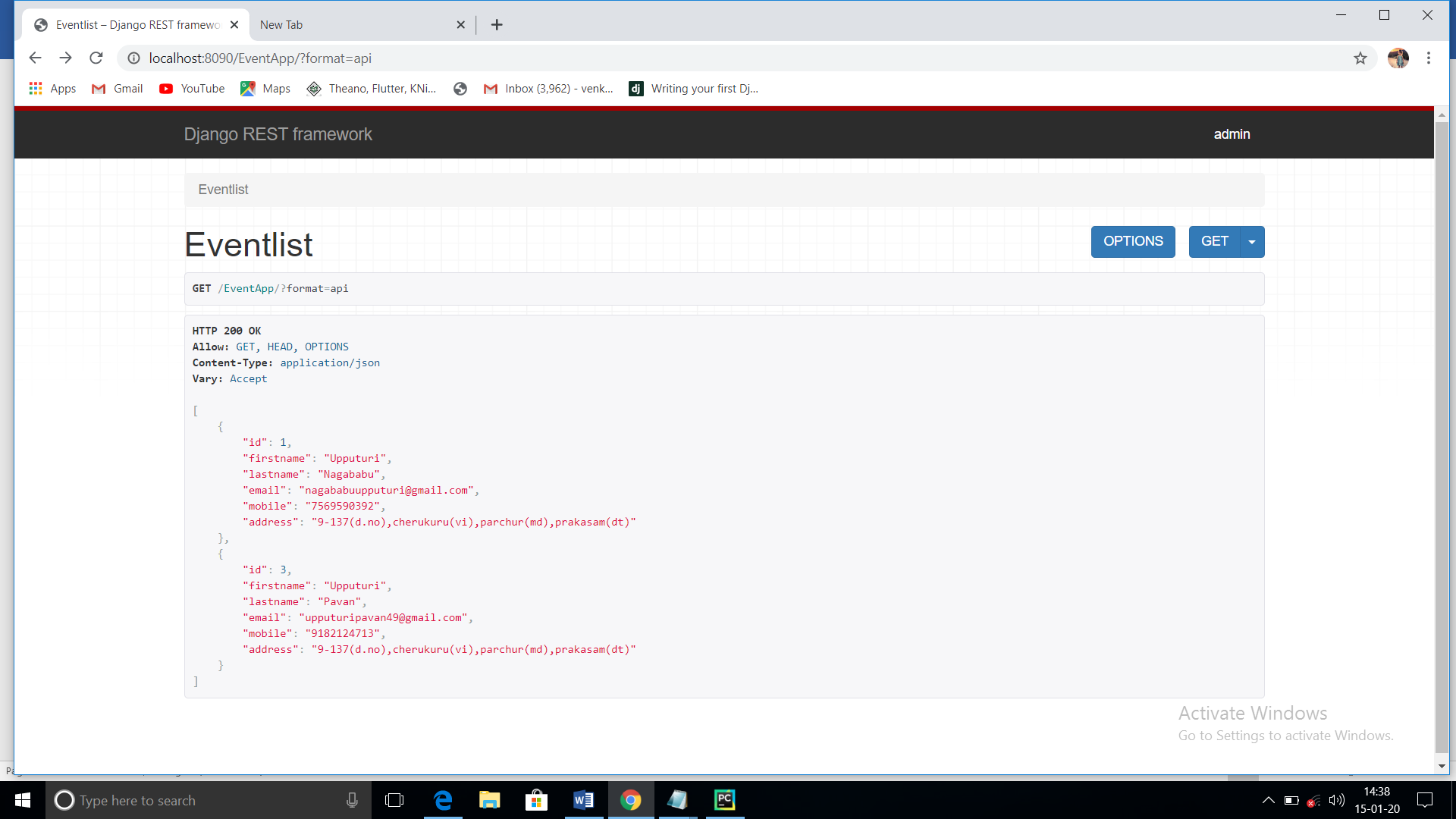
from rest\_framework.views import APIView  
from rest\_framework.response import Response  
from django.shortcuts import get\_object\_or\_404  
from . models import EventRegistration  
from .serializer import EventSerializer  
  
class Eventlist(APIView):  
  
 def get(self,request):  
 maxdata=20  
 pools = EventRegistration.objects.all()[:maxdata]  
 data=EventSerializer(pools,many=True).data  
 return Response(data)  
  
class EventDetails(APIView):  
 def get(self,request):  
 global data  
 poll=get\_object\_or\_404(EventRegistration)  
 data=EventSerializer(poll).data  
 return Response(data)

**EventApp\urls.py**

from django.urls import path,include  
from .views import EventRegistration\_list  
from . apiviews import Eventlist,EventDetails   
urlpatterns=[  
  
 path('EventApp/',Eventlist().as\_view(), name="Event Registered list"),  
 path('EventApp/',EventDetails().as\_view(),name="Event Details"),

In the uel enter the below address

<http://localhost:8090/EventApp/>



**Json format:**

[{"id":1,"firstname":"Upputuri","lastname":"Nagababu","email":"nagababuupputuri@gmail.com","mobile":"7569590392","address":"9-137(d.no),cherukuru(vi),parchur(md),prakasam(dt)"},

{"id":3,"firstname":"Upputuri","lastname":"Pavan","email":"upputuripavan49@gmail.com","mobile":"9182124713","address":"9-137(d.no),cherukuru(vi),parchur(md),prakasam(dt)"}]

**//Registration with RESTAPI**

**EventApp\serializer.py**

from django.contrib.auth.models import User

class UserSerializer(serializers.ModelSerializer):  
 class Meta:  
 model=User  
 fields=['username','email','password']  
 kwargs={'password':{'write\_only':True}}  
 def create(self,validated\_data):  
 user=User(email=validated\_data['email'],username=validated\_data['username'])  
 user.set\_password(validated\_data['password'])  
 user.save()  
 return user

**EventApp\apiviews.py**

from rest\_framework import generics

from .serializer import EventSerializer,UserSerializer

class Usercreate\_or\_Registeruser(generics.CreateAPIView):  
 authentication\_classes = ()  
 permission\_classes = ()  
 serializer\_class = UserSerializer

**Serializationis the process of making a streamable representation of the data which we can transfer over the network.**

I use ModelSerializer(serializers.ModelSerializer)

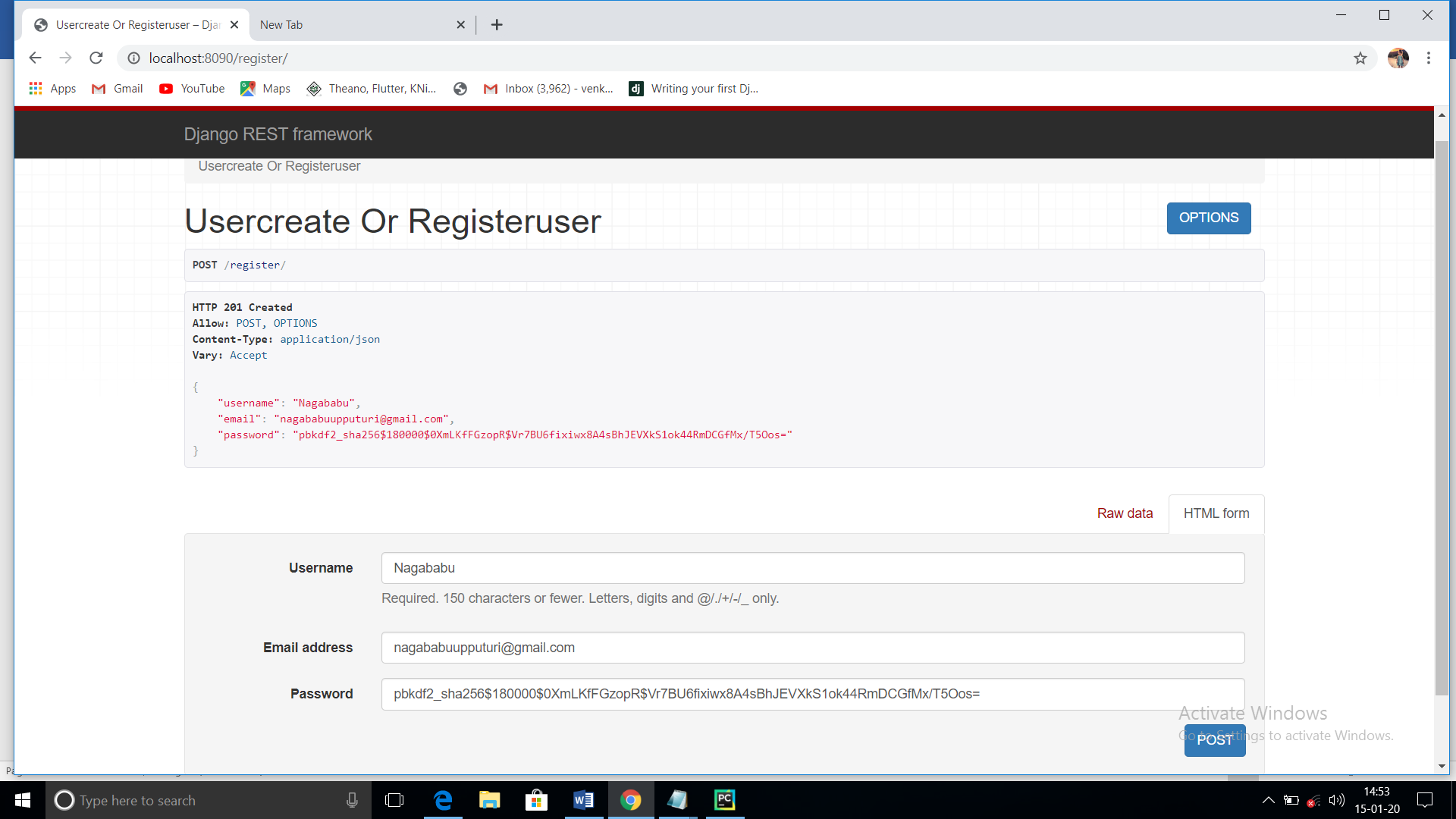
**EventApp\urls.py**

from . apiviews import Eventlist,EventDetails,Usercreate\_or\_Registeruser

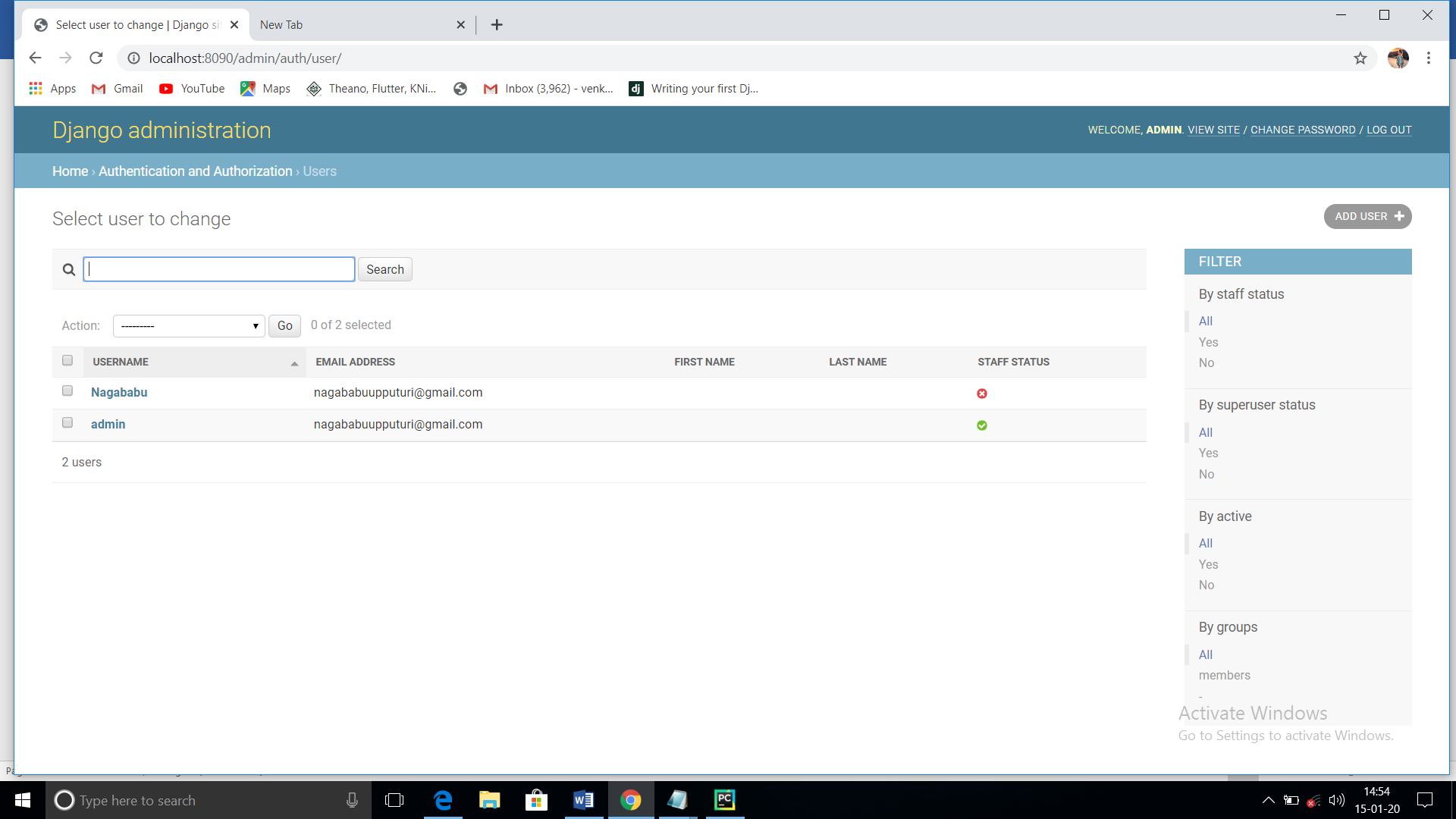
path('register/',Usercreate\_or\_Registeruser.as\_view(),name="User Registration"),

**Then enter**

<http://localhost:8090/register/>



**In admin site Users has**



**Login for Specified Users:(only registred users access the admin site)**

**EventApp\apiviews.py**

from django.contrib.auth import login

from django.contrib.auth import authenticate

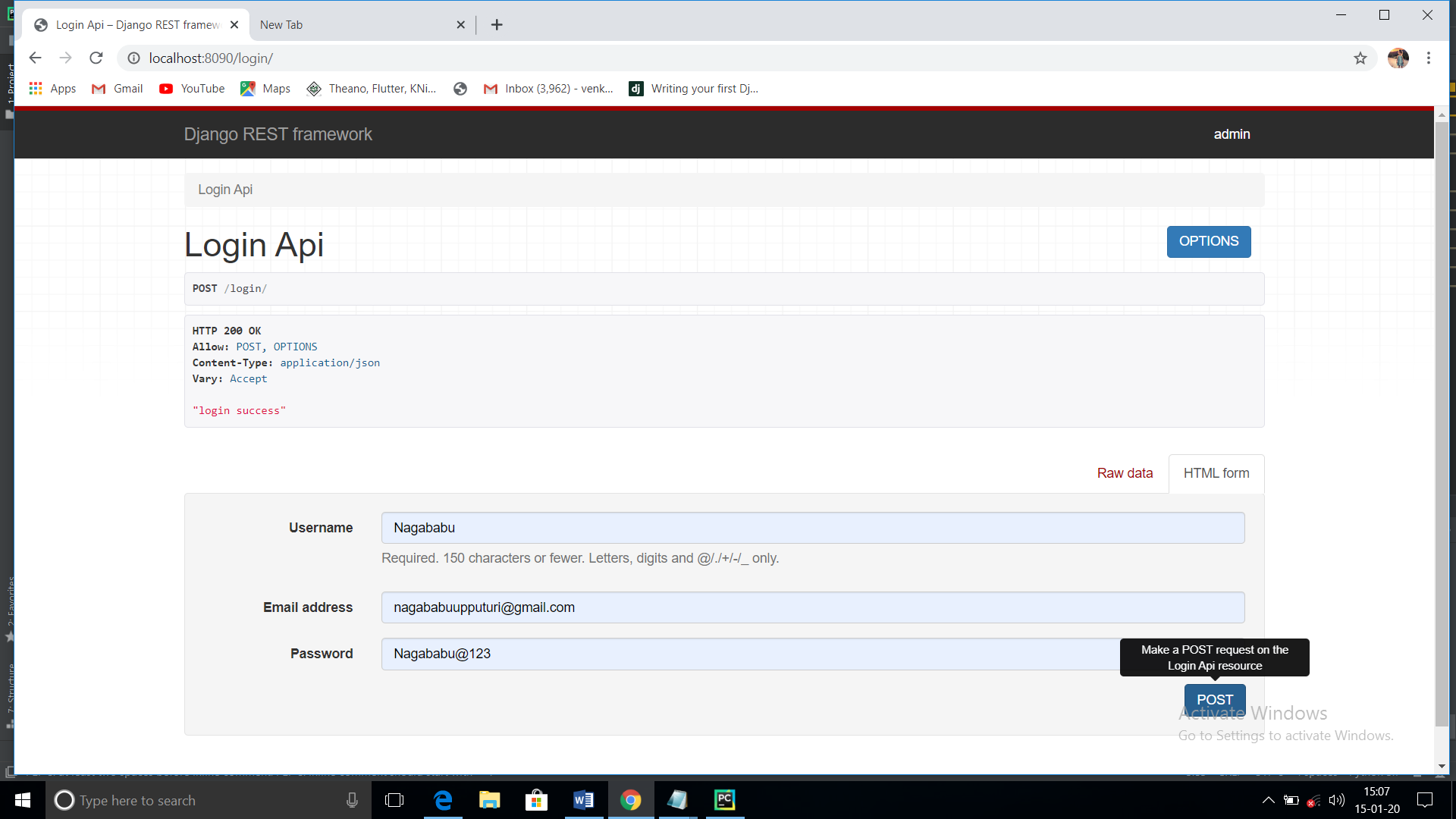
class LoginApi(generics.CreateAPIView):  
 serializer\_class = UserSerializer  
 permission\_classes = ()  
 def post(self,request,):  
 username=request.data.get("username")  
 password=request.data.get("password")  
 user=authenticate(username=username,password=password)  
 if user:  
 return Response("login success")  
 else:  
 return Response({"error":"wrong credentials"},status=status.HTTP\_400\_BAD\_REQUEST)

**EventApp\urls.py**

from . apiviews import Eventlist,EventDetails,Usercreate\_or\_Registeruser,LoginApi

path('login/',LoginApi().as\_view(),name="Login"),

<http://localhost:8090/login/>



**If the Username is already exists in admin site to display ERROR message.**

**Delete Event:**

**EventApp\serializer.py**

class AccountSeliazier(serializers.ModelSerializer):  
 password=serializers.CharField(write\_only=True,required=False)  
 class Meta:  
 model=EventRegistration  
 fields='\_\_all\_\_'  
 def creates(self,validated\_data):  
 return EventRegistration.objects.create\_user(request\_data=validated\_data)

**EventApp**\**apiviews.py**

from .serializer import EventSerializer,UserSerializer,AccountSeliazier

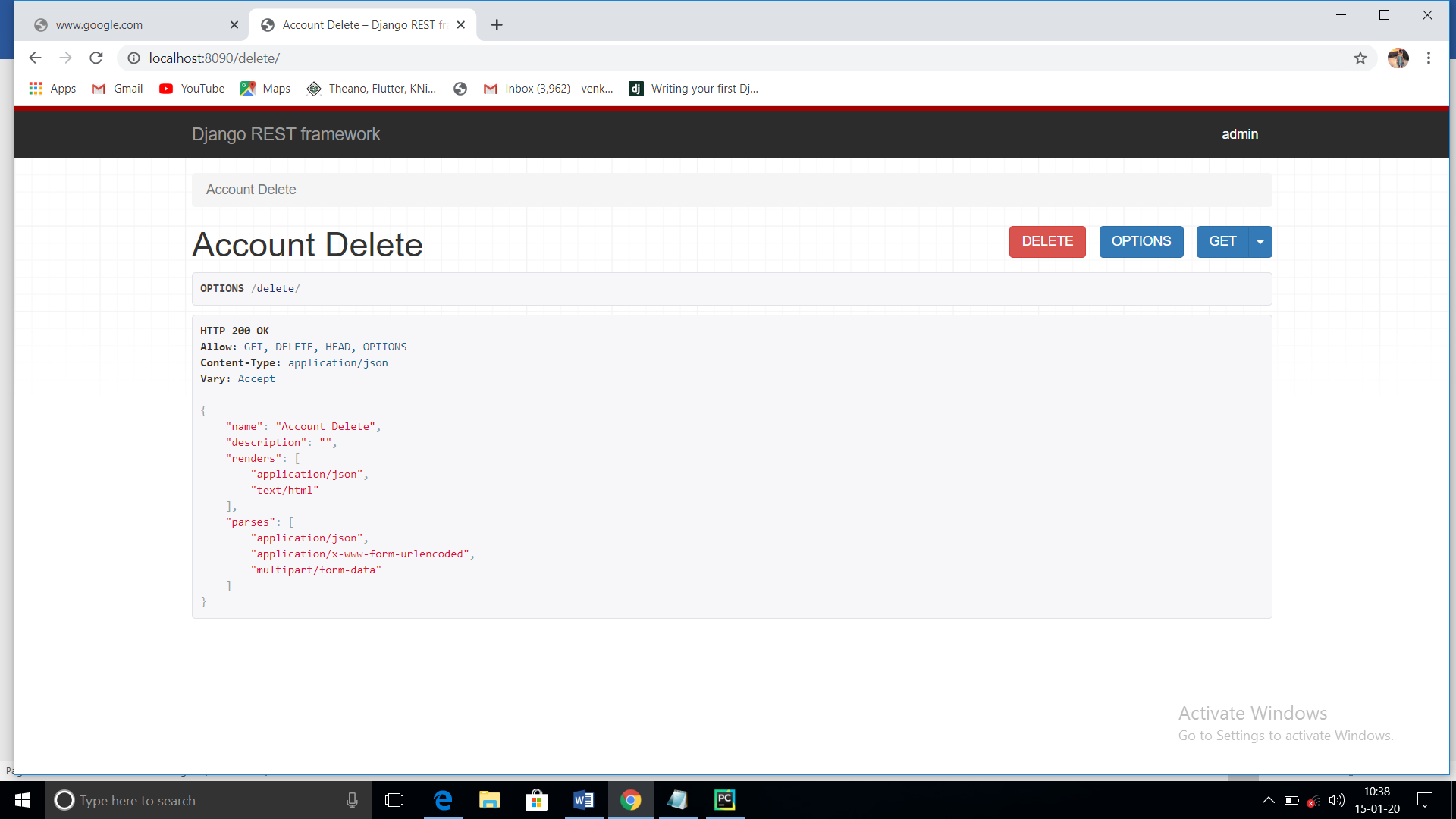
class AccountDelete(generics.DestroyAPIView):  
 serializer\_class = AccountSeliazier  
 lookup\_field = "email"  
 queryset = EventRegistration.objects.all()  
 def get(self,request):  
 try:  
 instance=self.queryset.get(email=self.request.user)  
 return instance  
 except EventRegistration.DoesNotExist:  
 raise Http404

**EventApp\urls.py:**

from . apiviews import Eventlist,EventDetails,Usercreate\_or\_Registeruser,LoginApi,AccountDelete

path('delete/',AccountDelete().as\_view(),name="account delete")

**enter in url:**<http://localhost:8090/delete/>



[AUTHENTICATION AND AUTHORIZATION](http://localhost:8090/admin/auth/)