**Auto generated document and swagger request:**

**Link for page :** <https://drf-swagger.readthedocs.io/en/1.1.2/readme.html>

**step 1:** pip install drf-yasg

**step 2:** settings.py

INSTALLED\_APPS = [

'drf\_yasg',

]

**step 3:** urls.py

from drf\_yasg.views import get\_schema\_view

from drf\_yasg import openapi

schema\_view = get\_schema\_view(

openapi.Info(

title="Snippets API",

default\_version='v1',

description="Test description",

terms\_of\_service="https://www.google.com/policies/terms/",

contact=openapi.Contact(email="contact@snippets.local"),

license=openapi.License(name="BSD License"),

),

validators=['ssv', 'flex'],

public=True,

permission\_classes=(permissions.AllowAny,),

)

urlpatterns = [

path('swagger(?P<format>.json|.yaml)', schema\_view.without\_ui(cache\_timeout=None), name='schema-json'),

path('swagger/', schema\_view.with\_ui('swagger', cache\_timeout=None), name='schema-swagger-ui'),

path('redoc/', schema\_view.with\_ui('redoc', cache\_timeout=None), name='schema-redoc'),

]

**Customized document and swagger request:**

**Sample for creating employee record:**

@swagger\_auto\_schema(

method="post",

request\_body=openapi.Schema(

type=openapi.TYPE\_OBJECT,

properties={

"employee\_id": openapi.Schema(type=openapi.TYPE\_STRING),

"first\_name": openapi.Schema(type=openapi.TYPE\_STRING),

"last\_name": openapi.Schema(type=openapi.TYPE\_STRING, nullable=True),

"phone\_number": openapi.Schema(type=openapi.TYPE\_STRING),

"email": openapi.Schema(type=openapi.TYPE\_STRING),

"city": openapi.Schema(type=openapi.TYPE\_STRING, nullable=True),

},

required=["employee\_id", "phone\_number", "email"]

),

responses={200: openapi.Response("Created", schema=openapi.Schema(

type=openapi.TYPE\_OBJECT,

properties={

"record\_id": openapi.Schema(type=openapi.TYPE\_STRING),

"record\_created\_time": openapi.Schema(type=openapi.TYPE\_STRING,

format=openapi.FORMAT\_DATETIME),

"record\_created\_by": openapi.Schema(type=openapi.TYPE\_STRING),

"record\_modified\_time": openapi.Schema(type=openapi.TYPE\_STRING,

format=openapi.FORMAT\_DATETIME),

"record\_modified\_by": openapi.Schema(type=openapi.TYPE\_STRING),

"employee\_id": openapi.Schema(type=openapi.TYPE\_STRING),

"first\_name": openapi.Schema(type=openapi.TYPE\_STRING),

"last\_name": openapi.Schema(type=openapi.TYPE\_STRING, nullable=True),

"phone\_number": openapi.Schema(type=openapi.TYPE\_STRING),

"email": openapi.Schema(type=openapi.TYPE\_STRING),

"city": openapi.Schema(type=openapi.TYPE\_STRING, nullable=True),

},

required=["employee\_id"]

))},

operation\_id="Create",

)

@api\_view(["POST"])

def employee\_create(request):

employee\_id = request.data.get('employee\_id')

first\_name = request.data.get('first\_name')

last\_name = request.data.get('last\_name')

phone\_number = request.data.get('phone\_number')

email = request.data.get('email')

city = request.data.get('city')

record = M0002.objects.create(

employee\_id=employee\_id,

first\_name=first\_name,

last\_name=last\_name,

phone\_number=phone\_number,

email=email,

city=city,

)

response\_data = {

'message': 'Employee Record has been Created.',

'record\_id': record.id,

'record\_created\_time': record.record\_created\_time,

'record\_created\_by': record.record\_created\_by,

'record\_modified\_time': record.record\_modified\_time,

'record\_modified\_by': record.record\_modified\_by,

'employee\_id': record.employee\_id,

'first\_name': record.first\_name,

'last\_name': record.last\_name,

'phone\_number': record.phone\_number,

'email': record.email,

'city': record.city,

}

return JsonResponse(response\_data, status=200)

**Sample for deleting employee by id:**

@swagger\_auto\_schema(

method="delete",

responses={

200: openapi.Response(

"OK",

schema=openapi.Schema(

type=openapi.TYPE\_OBJECT,

properties={

"message": openapi.Schema(

type=openapi.TYPE\_STRING,

default="Employee Record has been deleted."

)

}

)

),

404: openapi.Response(

"Not Found",

schema=openapi.Schema(

type=openapi.TYPE\_OBJECT,

properties={

"detail": openapi.Schema(type=openapi.TYPE\_STRING,

default="Employee list not found."),

},

),

),

},

operation\_id="Delete",

manual\_parameters=[

openapi.Parameter(

name="employee\_id",

in\_=openapi.IN\_QUERY,

type=openapi.TYPE\_STRING,

description="Delete record by employee ID.",

required=False,

),

],

)

@api\_view(["DELETE"])

def employee\_delete(request):

employee\_id = request.GET.get('employee\_id')

try:

record = M0002.objects.get(employee\_id=employee\_id)

except M0002.DoesNotExist:

raise Http404("Employee record does not exist")

record.delete()

response\_data = {"message": "Employee Record has been deleted."}

return JsonResponse(response\_data, status=200)