

Swagger

API testing with swagger:-

- Django Rest swagger is used to provide communication for all APIs which are used in our application with brief description about each API individually
- Swagger is a open source API testing module, which is accepting all different kind of APIs and it returns response on the browser, with status codes for every request
- it is not focusing requested API is developed by which platform and which domain, it focus only that API is existing or not
- swagger is a set of open source tools built around the open API specification, that can help you to design, design, document and consume the rest APIs
- if we want to work with swagger, we need to install one third party module like below

```
cmd> pip install django-rest-swagger
```

- After installing add "rest-framework-swagger" application inside INSTALLED_APPS in settings.py

```
INSTALLED_APPS = [
```

```
.....  
'rest-framework-swagger'  
.....  
]
```

→ Open `url's.py` file and importing the 'get-swagger-view' for generating the swagger Apis and importing the 'include-docs-urls' for generating the readymade documentation for our Apis

```
from rest_framework_swagger.views import get_swagger_view
```

```
swagger-view = get_swagger_view(title = 'Employee API details')  
↓  
user defined name
```

```
from rest_framework.documentation import include_docs_urls
```

```
swagger-docs = include_docs_urls(title = 'swagger documentation  
for Employee')  
↓  
user defined name
```

→ Create some url's to execute `swagger-view`, `swagger-docs` variables

```
url patterns = [
```

```
url(r'^swagger-api/', swagger-view; name='swagger-api'),  
url(r'^swagger-docs/', swagger-docs),
```

Project:-

step-1: create a project:

project name: swagger-project

step-2: application name: swagger-app

step-3: database name: swagger-db

step-4: Add application name, rest-framework, rest-framework-swagger, in INSTALLED_APPS, and add database configurations in settings.py file

Add this line for api presentation

```
REST_FRAMEWORK = {
```

```
    'DEFAULT_SCHEMA_CLASS': 'rest_framework.schemas.coreapi.  
AutoSchema'
```

```
    'LOGIN_REDIRECT_URL': 'swagger-api'
```

```
    'LOGOUT_REDIRECT_URL': 'login'
```

step-5: open models.py and write the below code

```
from django.db import models
```

```
class Employee(models.Model):
```

```
    eno = models.IntegerField(primary_key=True)
```

```
    ename = models.CharField(max_length=100)
```

```
    esal = models.DecimalField(max_digits=10, decimal_places=2)
```

```
    eaddr = models.CharField(max_length=100)
```

step-6: open serializers.py

```
from rest_framework import serializers
```

```
from swagger_app.models import Employee
```

```
class EmployeeSerializer(serializers.ModelSerializer):
```

```
    class Meta:
```

```
        model = Employee
```

```
        fields = '__all__'
```


step-6: open admin.py

```
from django.contrib import admin
from swagger_app.models import Employee

class EmployeeAdmin(admin.ModelAdmin):
    list_display = ['eno', 'ename', 'esal', 'eaddr']

admin.site.register(Employee, EmployeeAdmin)
```

step-7: open views.py

```
from django.shortcuts import render
from swagger_app.models import Employee
from swagger_app.serializers import EmployeeSerializer
from rest_framework import generics
from rest_framework.authentication import BasicAuthentication
from rest_framework.permissions import IsAuthenticated

class EmployeeListSwaggerView(generics.ListCreateAPIView):
    """
    list :
        Return a list of all Employees.
    create :
        create a new Employee
    """
    queryset = Employee.objects.all()
    serializer_class = EmployeeSerializer
    authentication_classes = (BasicAuthentication,)
    permission_classes = (IsAuthenticated,)
```

class Employee - Detail - swagger - view (generics: Retrive Update DestroyAPIView):

"""

retrive:

Return the given Employee

destroy:

Delete a given Employee

update:

update a given Employee

partial - update:

partial update a given employee

"""

queryset = Employee.objects.all()

serializer_class = EmployeeSerializer

step-8: open urls.py

from django.contrib import admin

from django.urls import path

from django.conf.urls import url, include

from swagger_app import views

from rest_framework_swagger.views import get_swagger_view

swagger_view = get_swagger_view(title='Employee API details')

from rest_framework.documentation import include_docs_urls

swagger_docs = include_docs_urls(title='swagger documentation for Employee')

from django.contrib.auth.views import LoginView, LogoutView

urlpatterns = [

path('admin/', admin.site.urls),

path('accounts/login/', loginview.as_view(template_name='login.html',
name='login'),

```

path ('logout/', logoutview.as_view(template_name='logout.html'),
      name='logout'),
url (r'^swagger-api/', swagger-view, name='swagger-api'),
url (r'^swagger-docs/', swagger-docs),
url (r'^emp/$', views.Employee-list-swagger-view.as_view()),
url (r'^emp/(?p<pk>[0-9]+)/$', views.Employee-Detail-swagger-
      view.as_view())
]

```

login.html

```

<html>
<head>
    { % load static % }
    <!DOCTYPE html>
    <html lang='en'>
    <head>
        <meta [copy Bootstrap lines]>
    <style>
        form { background-color: green; }
        th { padding: 10px; }
    </style>
</head>
<body>
    <div class='container'>
        <div class='row'>
            <div class='col-md-6 offset-3'>
                <form method='post'>
                    { % csrf-token % }
                    <table>
                        { % form % }
                    </table>
                    <input type='submit' value='login' style='margin-left: 130px'>

```



```
</form>
<div>
</div>
</div>
<body>
</html>
```

logout.html

```
<h1> Thanks for visiting our site </h1>
<a href = "/login" > login Again </a>
```