

Lab: 12

Student Task:

Code:

Views.py

```
views.py x models.py x serializers.py x admin.py x settings.py x basic_api\urls.py x
1 from django.shortcuts import render
2 from rest_framework import generics
3 from basic_api.models import DRFPost
4 from basic_api.serializers import DRFPostSerializer
5
6 # Create your views here.
7
8 class API_objects(generics.ListCreateAPIView):
9     queryset = DRFPost.objects.all()
10    serializer_class = DRFPostSerializer
11
12 class API_objects_details(generics.RetrieveUpdateDestroyAPIView):
13     queryset = DRFPost.objects.all()
14     serializer_class = DRFPostSerializer
15
```

Models.py

```
views.py x models.py x serializers.py x admin.py x settings.py x basic_api\urls.py x DRFtuto
1 from django.db import models
2 # list
3 Grade = [
4     ('excellent', 1),
5     ('average', 0),
6     ('bad', -1)
7 ]
8 # DataFlair
9 class DRFPost(models.Model):
10     name = models.CharField(max_length=100)
11     author = models.CharField(max_length=100)
12     uploaded = models.DateTimeField(auto_now_add=True)
13     rating = models.CharField(choices=Grade, default='average', max_length=50)
14
15     class Meta:
16         ordering = ['uploaded']
17
18     def __str__(self):
19         return self.name
```

Serializers.py

```
views.py × models.py × serializers.py × admin.py × settings.py ×
1      # DataFlairs
2
3      from rest_framework import serializers
4      from basic_api.models import DRFPost
5
6      class DRFPostSerializer(serializers.ModelSerializer):
7          class Meta:
8              model = DRFPost
9              fields = '__all__'
```

Admin.py

```
views.py × models.py × serializers.py × admin.py ×
1      from django.contrib import admin
2      from basic_api.models import DRFPost
3
4      # Register your models here.
5
6      # DataFlair
7      admin.site.register(DRFPost)
8      |
```

Basi_api\urls.py

```
views.py × models.py × serializers.py × admin.py × settings.py × basic_api\urls.py ×
1 from rest_framework.urlpatterns import format_suffix_patterns
2 from django.urls import path
3 from basic_api import views
4
5 urlpatterns = [
6     path('basic/', views.API_objects.as_view()),
7     path('basic/<int:pk>/', views.API_objects_details.as_view()),
8 ]
9
10 urlpatterns = format_suffix_patterns(urlpatterns)
```

DRFtutorial\urls.py

```
views.py × models.py × serializers.py × admin.py × settings.py × basic_api\urls.py × DRFtutorial\urls.py ×
1 urlpatterns configuration
2
3 The 'urlpatterns' list routes URLs to views. For more information please see:
4 https://docs.djangoproject.com/en/3.1/topics/http/urls/
5 Examples:
6 Function views
7 1. Add an import: from my_app import views
8 2. Add a URL to urlpatterns: path('', views.home, name='home')
9 Class-based views
10 1. Add an import: from other_app.views import Home
11 2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
12 Including another URLconf
13 1. Import the include() function: from django.urls import include, path
14 2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
15 """
16 from django.contrib import admin
17 from django.urls import path, include
18
19 urlpatterns = [
20     path('admin/', admin.site.urls),
21     path('', include('basic_api.urls')),
22 ]
23
```

Outputs:

```
Vary: Accept

[
  {
    "id": 1,
    "name": "DataFlair",
    "author": "saad",
    "uploaded": "2021-01-28T11:22:11.568346Z",
    "rating": "average"
  },
  {
    "id": 2,
    "name": "Django REST framework",
    "author": "saad",
    "uploaded": "2021-01-28T11:36:03.752162Z",
    "rating": "excellent"
  },
  {
    "id": 3,
    "name": "REACT",
    "author": "saad",
    "uploaded": "2021-01-28T11:40:31.152384Z",
    "rating": "excellent"
  },
  {
    "id": 4,
    "name": "Django",
    "author": "saad",
    "uploaded": "2021-02-05T09:55:18.905603Z",
    "rating": "excellent"
  }
]
```

API Objects

POST /basic/?format=api

HTTP 201 Created
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

```
{
  "id": 4,
  "name": "Django",
  "author": "saad",
  "uploaded": "2021-02-05T09:55:18.905603Z",
  "rating": "excellent"
}
```

Raw data HTML form

Name	<input type="text" value="Django"/>
Author	<input type="text" value="saad"/>
Rating	<input type="text" value="1"/>

POST

The above output was from - <http://127.0.0.1:8000/basic/>

Calling unique ID's:

<http://127.0.0.1:8000/basic/1/>

The screenshot shows a web browser window with the address bar displaying `127.0.0.1:8000/basic/1/`. The browser tabs include WhatsApp, Ap I Objects Details - Django RE, and others. The page title is "Django REST framework". The main content area shows "Ap I Objects Details" with a breadcrumb "Ap I Objects / Ap I Objects Details". There are three buttons: "DELETE" (red), "OPTIONS" (blue), and "GET" (blue). Below these is a text input field containing "GET /basic/1/". The response area shows the following details:

```
HTTP 200 OK
Allow: GET, PUT, PATCH, DELETE, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
  "id": 1,
  "name": "DataFlairDjango",
  "author": "Ali",
  "uploaded": "2021-01-28T11:10:14.299675Z",
  "rating": "excellent"
}
```

Below the response is a "Raw data" tab and an "HTML form" tab. The "HTML form" tab is active, showing a form with three fields: "Name" (DataFlairDjango), "Author" (Ali), and "Rating" (1). There is a "PUT" button at the bottom right of the form.

<http://127.0.0.1:8000/basic/2/>

Ap I Objects / Ap I Objects Details

DELETE OPTIONS GET

GET /basic/2/

HTTP 200 OK
Allow: GET, PUT, PATCH, DELETE, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

```
{
  "id": 2,
  "name": "Django REST Framework",
  "author": "Mughal",
  "uploaded": "2021-01-28T11:10:48.843651Z",
  "rating": "excellent"
}
```

Raw data HTML form

Name Django REST Framework

Author Mughal

Rating 1

PUT

<http://127.0.0.1:8000/basic/3/>

Ap I Objects / Ap I Objects Details

DELETE OPTIONS GET

GET /basic/3/

HTTP 200 OK
Allow: GET, PUT, PATCH, DELETE, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

```
{
  "id": 3,
  "name": "React native",
  "author": "Ahsan",
  "uploaded": "2021-01-28T11:11:22.093701Z",
  "rating": "average"
}
```

Raw data HTML form

Name React native

Author Ahsan

Rating 0

PUT

Finally, the one which in actual doesn't exists:

<http://127.0.0.1:8000/basic/4/>

The screenshot shows a web browser window with the address bar displaying `127.0.0.1:8000/basic/4/`. The browser's tab bar includes links to WhatsApp, Reddit, Oracle Application..., Solutions to Calculu..., Course Progress, Slack | general | CS..., Judge System | Has..., GDG Kolachi (Karac..., and Google. The page title is "Django REST framework". The main content area shows the "Ap I Objects Details" page. At the top right of this page are buttons for "DELETE", "OPTIONS", and "GET". Below these is a "GET /basic/4/" button. The main content area displays the following HTTP response:

```
HTTP 404 Not Found
Allow: GET, PUT, PATCH, DELETE, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
  "detail": "Not found."
}
```

Below the response, there are tabs for "Raw data" and "HTML form". The "HTML form" tab is selected, showing a form with three fields: "Name", "Author", and "Rating". The "Rating" field has a dropdown menu with the value "1" selected. A "PUT" button is located at the bottom right of the form.