## django rest framework ابزاری قدرتمند و انعطاف پذیر برای ساخت Web Api است. Api مخفف Application Programming Interface است. Apiها استانداردی برای تبادل اطلاعات بین دو یا چند ماشین هستند. ماشین ها میتوانند اطلاعاتی که توسط Api به اشتراک گذاشته میشوند را گرفته و به هر شکلی که مایل هستند استفاده کنند. همچنین در صورت نیاز Apiها میتوانند اجازه تغییر در اطلاعات را به ماشین ها دهند. با استفاده از فریمورک Dajngo Rest میتوانید به سادگی اقدام به ساخت Api کنید. Django Rest دارای مستندات فوق العاده قابل فهمی میباشد. یکی از نقاط قوت Django Rest سادگی در serialize کردن model objects میباشد.

================================================================================

api  
api مخفف Application Programming Interface است. api یک رابط برای اتصال چندین ماشین بکدیگر برای تبادل اطلاعات است. با استفاده از یک برنامه میتواند در چند پلتفرم مختلف کار کند.

================================================================================

response  
زمانی که درخواست از سمت کاربر به برنامه ارسال میشود برای پاسخ دادن به درخواست ها میتوان از کلاس Response استفاده کرد. نوع پاسخی که از طریق کلاس Response ارسال میکنید باید یک دیکشنری پایتونی باشد.

https://www.django-rest-framework.org/api-guide/responses/

متد response پارامترهای گوناگونی به عنوان پارامتر دریافت میکند که مهمترین آنها data میباشد که باید به صورت دیکشنری باشد و پاسخی که متد response ارسال میکند به صورت json میباشد   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

هنگامیکه میخواهید از طریق response به ارسال پاسخ بپردازید لازم است اگر از cbv استفاده میکنید از APIView ارث بری کنید یا اگر از fbv استفاده میکنید از دکوریتور api\_view@ استفاده کنید

[https://www.django-rest-framework.org/api-guide/views/#class-based-views](https://www.django-rest-framework.org/api-guide/views/" \l "class-based-views)

[https://www.django-rest-framework.org/api-guide/views/#function-based-views](https://www.django-rest-framework.org/api-guide/views/" \l "function-based-views)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

دکوریتور api\_view متدهایی که باید این فانکشن قبول کند را به عنوان ورودی دریافت میکند اگر چیزی قرار ندهید به صورت پیش فرض فقط get میباشد

========================================================================

request  
با استفاده از آبجکت request میتوانید به پارامترهای ارسالی توسط کاربر دسترسی پیدا کنید. همچنین میتوانید به وضعیت کاربر, متد درخواستی یا نوع دیتای قابل قبول دسترسی داشته باشید

https://www.django-rest-framework.org/api-guide/requests/

این request همان requestی هست که ما به هر متدی میدهیم

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

نکته قابل توجه :

هنگامیکه از متد get استفاده میکنید لازم نیست که حتما (/) قرار دهید اما اگر از متد post استفاده میکنید باید حتما (/) قرار دهید

========================================================================

serializer  
زمانی که نیاز به ارسال اطلاعات دیتابیس داشته باشید باید اطلاعات خود را به json تبدیل کرده و ارسال کنید. برای تبدیل model objects به json میتوانید از serializer استفاده کنید. serializerها میتوانند با دو کلاس Serializer و ModelSerializer ایجاد شوند.

https://www.django-rest-framework.org/api-guide/serializers/

========================================================================

هنگامیکه تعداده زیادی object رو به serializer ارسال میکنید لازم است که many=true را قرار دهید

ser\_data = PersonSerializer(persons, many=True)

اما اگر یک object رو ارسال میکردید دیگر لازم نبود عبارت بالا رو قرار دهید

اگر در serializer از serializer.Serializer ارث بری کردید لازم است برای هر فیلدی که میخواهید ارسال شود یک بازنویسی انجام شود

========================================================================

ModelSerializer  
ModelSerializer نوع دیگری از serializer است که به صورت مستقیم میتواند اطلاعات داخل یک model را به json تبدیل کند. استفاده از کلاس ModelSerializer نسبت به کلاس Serializer مرسوم تر است

[https://www.django-rest-framework.org/api-guide/serializers/#modelserializer](https://www.django-rest-framework.org/api-guide/serializers/" \l "modelserializer)

در ModelSeerializer میتوانید تمام فیلدی هارو به وسیله\_\_all\_\_ ارسال کنید اما جنگو پیشنهاد میکند به صورت صریح هر فیلدی را بنویسید

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

نکته ای که وجود دارد این است که بعضی از مواقع نشان دادن ایمیل افراد کار مناسبی نمیباشد یکی از راه کارها حذف email از fields میباشد اما موقع ساخت person جدید به مشکل میخوریم یک راه مناسب استفاده از optionهای ModelSerializer میباشد که در این مورد write\_only میباشد :

[https://www.django-rest-framework.org/api-guide/fields/#core-arguments](https://www.django-rest-framework.org/api-guide/fields/" \l "core-arguments)

نکته ی مهمی که وجود دارد که اگر در کلاسی که از ModelSerializer ارث بری میکند فیلد جدیدی ساخته شود باید در همان فیلد optionها ساخته شود

out\_of\_iran = serializers.NullBooleanField(required=False, default=False)

اما اگر فیلدی ساخته نشد و به همان فیلدهایی که در fields قرار گرفته بود option اضافه شد از   
extra\_kwargs استفاده میکنیم :

fields = ('id', 'name', 'age', 'email', 'car')  
extra\_kwargs = {'email':{'write\_only':True}}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_