**OVERVIEW**

$pipenv install django~=4.2.3

$pipenv shell

$django-admin startproject project\_name .

$exit

Create Dockerfile

Create docker-compose.yml file

$docker compose up -d —build

Check if its working

$docker compose exec <service\_name> \_\_\_\_\_\_command goes here\_\_\_\_

From here on wards every command preceed with docker compose exec web

$python manage.py startapp app\_name

Register the app in settings.py

Change the type of the database in settings.py /which ever you want to use

For postgres use the following command

$pipenv install psycopg2-binary==2.9.6

Then Create a model in the models.py

$python manage.py makemigrations <app name>

$python manage.py migrate <app name>

Now register the app into admin.py . create a custom admin class if necessary

$python manage.py createsuperuser

now login to the admin panel see if you can see the model

Tillhere everything is same

$pipenv install djangorestframework~=3.14.0

Add ‘rest\_framework’ to installed apps in settings.py of the project

Configure urls.py in project

add urls.py in the app

then add a serializer

then add views

$pipenv install django-cors-headers~=4.2.0

add ‘corsheaders’ to INSTALLED\_APPS

add ‘corsheaders.middleware.CorsMiddleware’ to the middleware section above common middleware

create cors\_allowed\_origin at the bottom

For csrf just add the trusted origins directly at the bottom in settings.py

**DJANGO REST FRAMEWORK SETTINGS**

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

<https://www.django-rest-framework.org/api-guide/settings/>

We can configure django’s rest framework in settings.py. The confirguration is namespaced inside a single django setting named REST\_FRAMEWORK

#settings.py

from rest\_framework.settings import api\_settings

REST\_FRAMEWORK={

‘DEFAULT\_RENDERER\_CLASSES’:[

‘rest\_framework.renderers.JSONRenderer’

],

‘DEFAULT\_PARSER\_CLASSES’:[

‘rest\_framework.parsers.JSONParser’

]

‘DEFAULT\_PERMISSION\_CLASSES’:[

‘rest\_framework.permissions.AllowAny”

]

}

The api\_settings object will check for user cusomized settings. Any setting that uses string import paths to refer to a class will automatically import and return the referenced class instead of the string literal.

DEFAULT\_RENDERE\_CLASSES: list or tuples of renderer class determining teh default set of renderers used when returning a response object

DEFAULT\_PARSER\_CLASSES: parsers used when accessing request.data property

DEFAULT\_AUTHENTICATION\_CLASSES: list of authenticators used when accessing the request.user or request.auth properties

DEFAULT\_PERMISSION\_CLASSES: default set of permissions checked at the start of the view. permissions must be granted by every class in the list.

**[django admin module](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)**

[from django.contrib import admin](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[admin allows us to configure the admin interface](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[admin interface allows us to interact with application data.](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[admin.site.register():](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[admin.site object provides method to register models and their corresponding admin classes.](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[Once you create a model register it in the admin.py of that particular app. You can also create an admin class so that you can customize it.](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

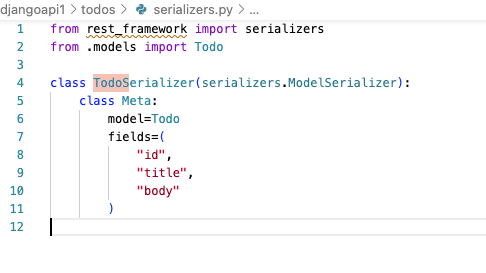
[Here TodoAdmin is a custom admin class that sets the list\_display attribute. That way in the admin panel data is shown as a list with 2 fields title and body in the admin panel.](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

**[SERIALIZERS](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)**

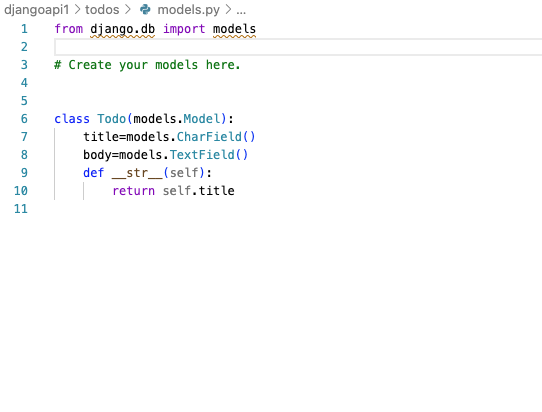
[Serializers are used to parse the data. For example I need to send the response as json data. But what I have is a model data. Serializers parse that model data into json data.](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[serializers.ModelSerializer provides a built in serializer for django models.](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[It consists of a nested class Meta which specifies which model it should serialize and what fields.](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[The id field is automatically created by django even though our model just consists of title and body as given below](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

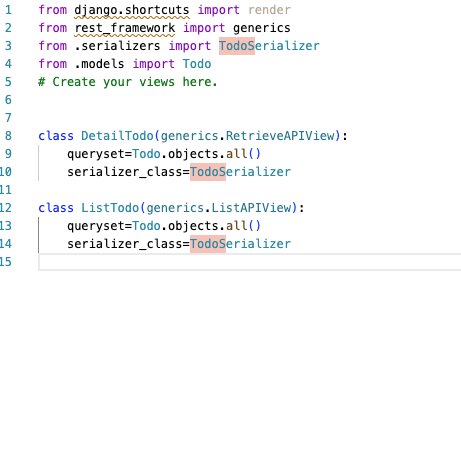
While django creates id automatically, it expects a *pk* value in the url to retrieve a particular record.They are both same at a particular level.

[https://docs.djangoproject.com/en/4.0/ref/models/instances/#the-pk-property](https://docs.djangoproject.com/en/4.0/ref/models/instances/" \l "the-pk-property)

[https://docs.python.org/3.10/library/functions.html#id](https://docs.python.org/3.10/library/functions.html" \l "id)

**[THE FOLLOWING ARE THE VIEWS FOR THE ABOVE](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)**

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

[](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[In Django REST framework, the `RetrieveAPIView` is a built-in generic view that is used to handle the retrieval of a single object from the database and serialize it into a response. It is specifically designed for read-only operations that fetch a single instance of a model.](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[In the above code queryset and serializer\_class are pre-defined attributes.](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[How does DetailTodo know it should get only one of the data?](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[Its because we specified a](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings) *pk* in the url. It will obtain the parameters and search for it. The urls.py of the todos app is as follows:

[](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[The as\_view is used to convert a class based view into a functin based view. It is used to generate a callable view function.](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

**[CORS](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)**

https://www.django-rest-framework.org/topics/ajax-csrf-cors/

<https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS>

<https://github.com/adamchainz/django-cors-headers>

[django-cors-headers is the default choice.](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[after installing cors:](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[add it to installed apps,](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[add it to middlware,](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[add CORS\_ALLOWED\_ORIGINS at the bottom](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)

[All the above in settings.py](https://www.django-rest-framework.org/api-guide/settings/" \l "accessing-settings)



**CSRF protection**

Given that we are building an SPA, csrf protection is important.

Add CSRF trusted origins to the settings.py

