python manage.py migrate

python manage.py createsuperuser

###

Play

Entrar na app

Entrar no admin

Mostrar usuarios, login, permissões

import uuid  
  
from django.db import models  
  
# Create your models here.  
  
STATUS = (  
 ('1', 'Proposal'),  
 ('2', 'Accepted'),  
 ('3', 'Work in Progress'),  
 ('4', 'Done'),  
 ('5', 'Archived'),  
)  
  
class Customer(models.Model):  
 name = models.CharField(max\_length=200, verbose\_name="Customer Name")  
 address = models.CharField(max\_length=200)  
  
 def \_\_str\_\_(self):  
 return self.name  
  
class Phone(models.Model):  
 number = models.CharField(max\_length=20, verbose\_name="Phone number")  
 customer = models.ForeignKey(Customer, on\_delete=models.CASCADE, related\_name='contacts')

def \_\_str\_\_(self):  
 return self.number  
  
class Order(models.Model):  
 order\_id = models.UUIDField(default=uuid.uuid4)  
 date = models.DateTimeField('date')  
 customer = models.ForeignKey(Customer, on\_delete=models.CASCADE)  
 status = models.CharField(max\_length=5, choices=STATUS, default='1')  
  
 def \_\_str\_\_(self):  
 return '%s - %s' % (self.customer.name, str(self.order\_id))  
  
  
class Service(models.Model):  
 name = models.CharField(max\_length=200, verbose\_name="Service Name")  
  
 def \_\_str\_\_(self):  
 return self.name  
  
class Order\_Item(models.Model):  
 order = models.ForeignKey(Order, on\_delete=models.CASCADE)  
 service = models.ForeignKey(Service, on\_delete=models.CASCADE)  
 cost = models.DecimalField(max\_digits=6,decimal\_places=2)  
 payed = models.BooleanField(blank=True)  
 done = models.BooleanField(blank=True)  
  
 def \_\_str\_\_(self):  
 return self.service.name

####create model

python manage.py makemigrations orderbackoffice

python manage.py migrate

python manage.py shell

from orderbackoffice.models import \*

Service.objects.all()

s = Service(name='Coding')

s.save()

s = Service(name='Deploy')

s.save()

Service.objects.all()

Service.objects.filter(name\_\_startswith='co')

from django.contrib import admin

from .models import \*

admin.site.register(Service)

admin.site.register(Customer)

admin.site.register(Order)

**veja como esta simples**

**e esta sem os detalhes**

Mostrar usuarios, login, permissões

**veja o service simples**

class ServiceAdmin(admin.ModelAdmin):  
 list\_display = ['name']  
 search\_fields = ['name']  
  
admin.site.register(Service, ServiceAdmin)

**Customer sem telephone**

class CustomerAdmin(admin.ModelAdmin):  
 list\_display = ['name', 'address']  
 search\_fields = ['name', 'address']  
  
admin.site.register(Customer,CustomerAdmin)

class PhoneInline(admin.TabularInline):  
 model = Phone  
 extra = 1

inlines = [PhoneInline]

**Order sem items e sem servicos**

**Prox pagina**

**Order sem items e sem servicos**

class ItemInline(admin.TabularInline):  
 model = Order\_Item  
 extra = 1  
  
class OrderAdmin(admin.ModelAdmin):  
 list\_display = ['order\_id', 'date', 'customer', 'status']  
 search\_fields = ['name', 'address', 'customer\_\_name']  
 date\_hierarchy = 'date'  
 list\_filter = ['status']  
 inlines = [ItemInline]

admin.site.register(Order, OrderAdmin)

**marcar como done**

def change\_done(modeladmin, request, queryset):  
 queryset.update(status='4')  
change\_done.short\_description = "Mark as DONE"

actions = [change\_done]

**custom column**

def pending\_services(self, obj):  
 return Order\_Item.objects.filter(order=obj).filter(done=False).count()  
pending\_services.allow\_tags = True  
pending\_services.short\_description = u'Pending Services'

list\_display = ['order\_id', 'date', 'customer', 'status', 'pending\_services']

**outro filtro, cliente**

list\_filter = ['status','customer']

**settings**

'rest\_framework',

**pip install djangorestframework**

serializers.py

from .models import \*  
from rest\_framework import serializers

class CustomerSerializer(serializers.HyperlinkedModelSerializer):  
 class Meta:  
 model = Customer  
 fields = ['id', 'name', 'address']

class ServiceSerializer(serializers.HyperlinkedModelSerializer):  
 class Meta:  
 model = Service  
 fields = ['id', 'name']

**Views**

from .models import \*  
from rest\_framework import viewsets  
from rest\_framework import permissions  
from .serializers import CustomerSerializer, ServiceSerializer  
  
  
class CustomerViewSet(viewsets.ModelViewSet):  
 queryset = Customer.objects.all()  
 serializer\_class = CustomerSerializer  
 permission\_classes = [permissions.IsAuthenticated]  
  
  
class ServiceViewSet(viewsets.ModelViewSet):  
 queryset = Service.objects.all()  
 serializer\_class = ServiceSerializer  
 permission\_classes = [permissions.IsAuthenticated]

**urls**

**urls**

from django.contrib import admin  
from django.urls import include, path  
from rest\_framework import routers  
from orderbackoffice import views  
  
router = routers.DefaultRouter()  
router.register(r'customer', views.CustomerViewSet)  
router.register(r'service', views.ServiceViewSet)  
  
urlpatterns = [  
 path('admin/', admin.site.urls),  
 path('', include(router.urls)),  
 path('api-auth/', include('rest\_framework.urls', namespace='rest\_framework'))  
]

**pagination ? settings**

REST\_FRAMEWORK = {

'DEFAULT\_PAGINATION\_CLASS': 'rest\_framework.pagination.PageNumberPagination',

'PAGE\_SIZE': 10

}

**Add nestead (serializers)**

class ContactSerializer(serializers.ModelSerializer):  
 class Meta:  
 model = Phone  
 fields = ('id', 'number')

contacts = ContactSerializer(many=True, read\_only=True)

fields = ['id', 'name', 'address', 'contacts']

**suggestao**

**suggestao**

* **rest backend (mobile app) + (pode ser tb App web angular, react ou vue.js)**
* **app\_internas para small companies (freelancers)**
* **personal controls**
* **app para controle tabelas de dominios de aplicações existentes (enum\_tables, properties\_tables, tipical selects in screens, tipo MDM)**

**https://github.com/bilouro/django-example-admin-restframework**