# #STARTANDO O PROJETO

$Projetos> django-admin startproject loja

$Projetos> cd loja

$loja> pip install virtualenv

$loja> python -m venv venv

$loja> .\venv\Scripts\activate

(venv) $loja> pip install django

(venv) $loja> pip install pillow

# #STARTANTO A APLICAÇÃO

## Primeiro: Cria a aplicação

(venv) $loja> python manage.py startapp catalogo

## Segundo: Adiciona a aplicação nas configurações do projeto loja/settings.py

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'catalogo.apps.CatalogoConfig',

]

## Terceiro: Cria o arquivo urls.py da aplicação catalogo/urls.py

from django.urls import path

from. import views

urlpatterns = [

    path('', views.index, name='index'),

]

## Quarto: Adiciona as urls da aplicação nas do projeto loja/urls.py

from django.contrib import admin

from django.urls import include,path

from django.conf import settings

from django.conf.urls.static import static

from django.views.generic import RedirectView

urlpatterns = [

    path('admin/', admin.site.urls),

    path('catalogo/', include('catalogo.urls')),

    path('', RedirectView.as\_view(url='/catalogo/')),

]

urlpatterns += static(settings.STATIC\_URL, document\_root=settings.STATIC\_ROOT)

## Quinto: Nas views da aplicação catalogo adiciona o método index catalogo/views.py

from django.shortcuts import render

from django.http import HttpResponse

def index(request):

    return HttpResponse("Hello, world. You're at the polls index.")

## Sexto: Testa a aplicação

(venv) $loja> python manage.py runserver

# CRIANDO OS MODELOS DA APLICAÇÃO

## Primeiro: Nos models do catalogo adicionar Tipo e Produto catalogo/models.py

from django.db import models

class Tipo(models.Model):

    descricao = models.CharField("Descrição", max\_length=100)

    dt\_cad = models.DateTimeField("Data de cadastro")

    class Meta:

        ordering = ['dt\_cad']

    def \_\_str\_\_(self):

        return self.descricao

UNIDADES\_CHOICES = [

        (1, "Kilos"),

        (2, "Litros"),

        (3, "Und"),

        (4, "Metro"),

]

class Produto(models.Model):

    descricao = models.CharField("Descrição", max\_length=260, null=False, blank = False)

    unidade = models.IntegerField("Unidade", default=3, choices = UNIDADES\_CHOICES)

    preco = models.DecimalField(decimal\_places=2, max\_digits=8)

    estoque = models.DecimalField(decimal\_places=2, max\_digits=8)

    foto = models.ImageField(upload\_to='catalogo\_img')

    tipo = models.ForeignKey(Tipo, on\_delete=models.CASCADE)

    dt\_cad = models.DateTimeField("Data de cadastro")

    class Meta:

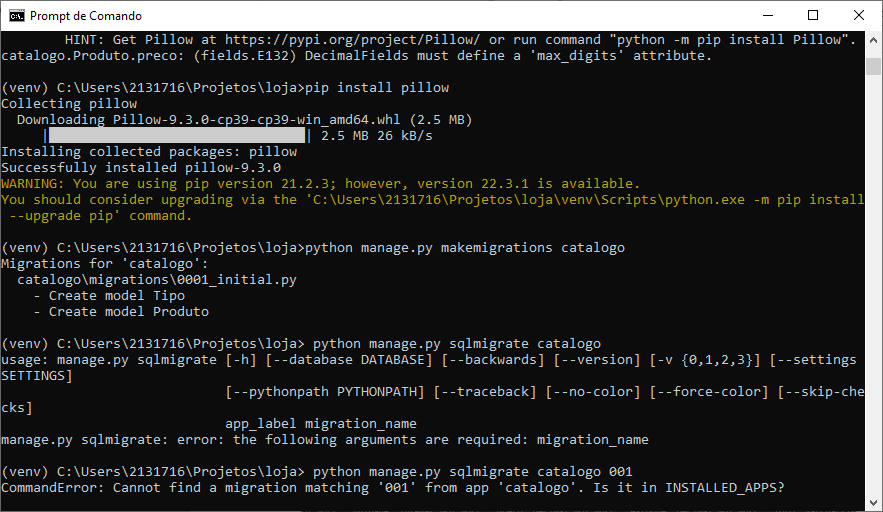
        ordering = ['dt\_cad']

    def \_\_str\_\_(self):

        return self.descricao

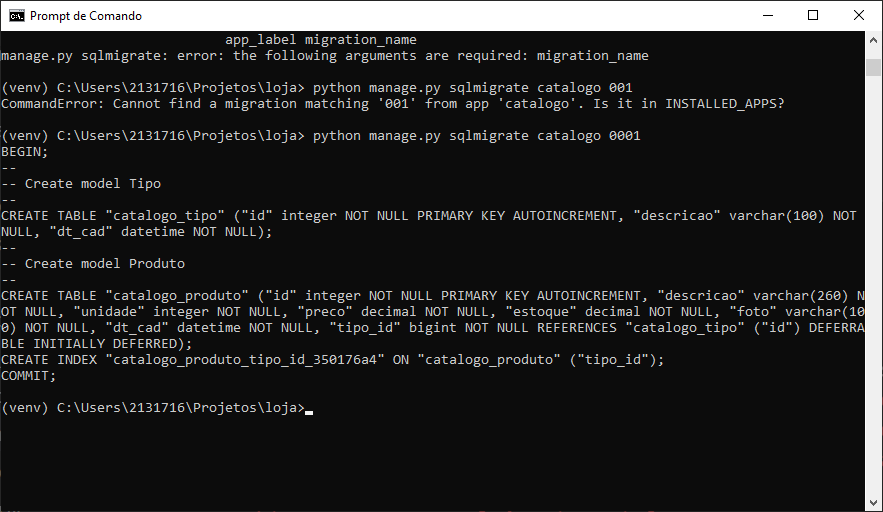
## Segundo: Comunicar ao Django mudança no modelo da aplicação

(venv) $loja> python manage.py makemigrations catalogo



## Terceiro: Ver o sql criado pela migração

(venv) $loja> python manage.py sqlmigrate catalogo 0001



## Quarto: Executar migrações não aplicadas à base de dados

(venv) $loja> python manage.py migrate

# TESTANDO A API DE COMUNICAÇÃO COM O BANCO

(venv) $loja> python manage.py shell

## # Importa os modelos recém-criados

>> from catalogo.models import Tipo, Produto

## # Importa classe timezone para extrair data atual

>> from django.utils import timezone

## # Seleciona todos os tipos salvos

>> Tipo.objects.all ()

## #Cria um novo tipo

>> t = Tipo(descricao="Smartphone", dt\_cad= timezone.now())

## # Salva o tipo criado

>> t.save()

## # Verificando o id do tipo criado

>> t.id

## # Acessando os atributos do tipo criado

>> t.descricao

>> t.dt\_cad

## # Mudando os atributos do tipo criado

>> t.descricao = "Celular"

>> t.save()

## #Seleciona todos os tipos salvos

>> Tipo.objects.all()

## #Filtrando os tipos pelo ID

>> Tipo.objects.filter(id=1)

## #Filtrando os tipos que a descricao começa com a (não case-sensitive)

>> Tipo.objects.filter(descricao\_\_istartswith= ”a”)

## #Filtrando os tipos que a descricao terminam com a (não case-sensitive)

>> Tipo.objects.filter(descricao\_\_iendswith= ”a”)

## #Filtrando os tipos que a descricao contem a (não case-sensitive)

>> Tipo.objects.filter(descricao\_\_icontains= ”a”)

## #Filtrando os tipos que a descricao é exatamente “smartphone” (não case-sensitive)

>> Tipo.objects.filter(descricao\_\_iexact= ”smartphone”)

## #Filtrando os tipos que a descricao é exatamente “smartphone” (case-sensitive)

>> Tipo.objects.filter(descricao\_\_exact= ”smartphone”)

## #Filtrando tipos cadastrados este ano

>> from django.utils import timezone

>> ano\_atual = timezone.now().year

>> Tipo.objects.filter(dt\_cad\_\_year= ano\_atual)

## #Criando produtos

>> p1 = Produto(descricao = “Iphone 13”, unidade = 1, preco = 13000.00, estoque = 10.0, tipo = t, dt\_cad = timezone.now())

>> p2 = Produto(descricao = “Moto OneHyper”, unidade = 1, preco = 1300.00, estoque = 100.0, tipo = t, dt\_cad = timezone.now())

## #Salvando produtos

>> p1.salve()

>> p2.salve()

## #Filtrando todos os produtos por tipo e descricao

>> Produto.objects.filter(descricao\_\_icontains=”i”, tipo\_\_descricao\_\_iexact=”smartphone”)

## #Filtrando todos os produtos por tipo e preço menor que 1500

>> Produto.objects.filter(tipo\_\_descricao\_\_iexact=”smartphone”, preco\_\_lte = 1500.0)

## #Filtrando todos os produtos por tipo e preço maior que 1500

>> Produto.objects.filter(tipo\_\_descricao\_\_iexact=”smartphone”, preco\_\_gte = 1500.0)

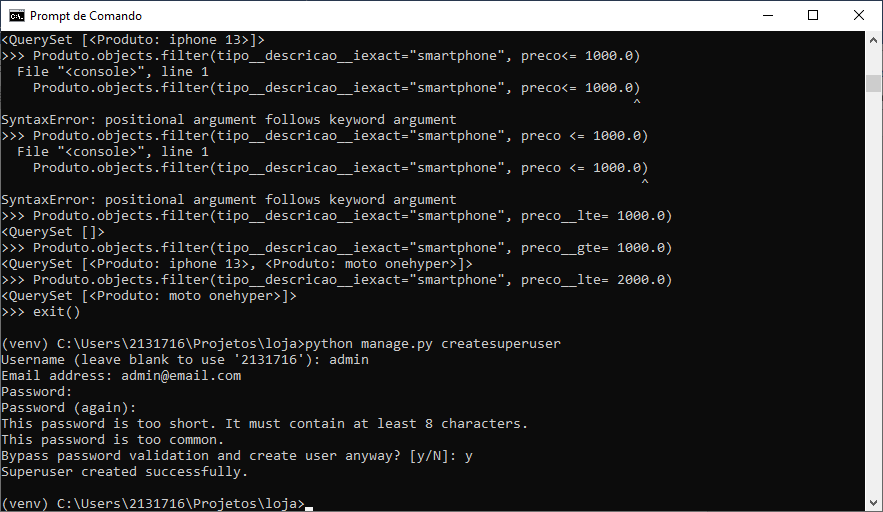
## #Sair do shell

>> exit()

# INTRODUÇÃO AO DJANGO ADMIN

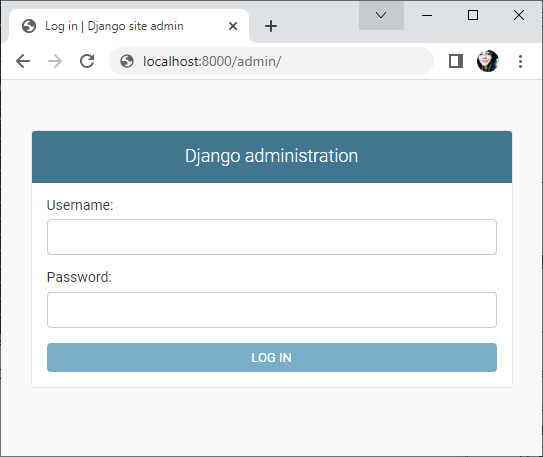
## Primeiro: Criar superuser

(venv) $loja> python manage.py createsuperuser

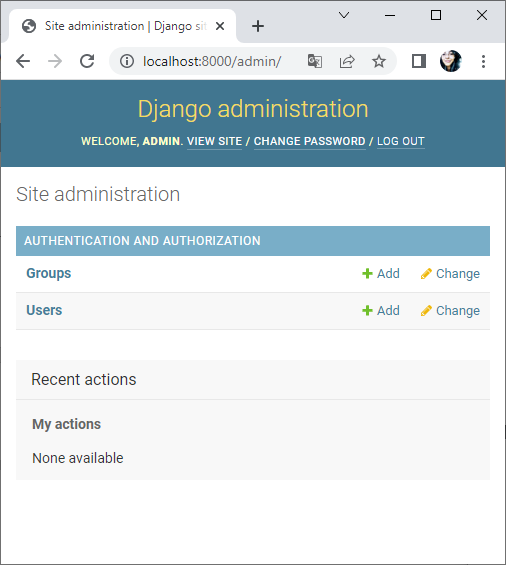


## Segundo: Acessar módulo admin

(venv) $loja> python manage.py runserver



## Terceiro: Realizar Login



## Quarto: Modificar idioma do ADMIN em loja/settings.py

LANGUAGE\_CODE = 'pt-br'

LANGUAGES = (

    ('pt-br', u'Português'),

    ('en', u'Inglês')

)

# ADICIONANDO UMA APLICAÇÃO A INTERFACE DO DJANGO ADMIN

## Primeiro: Adicionar o módulo admin.py na aplicação catalogo/admin.py

from django.contrib import admin

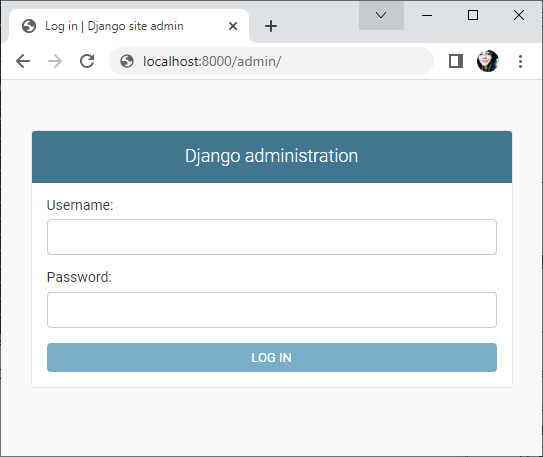
from catalogo.models import Tipo, Produto

admin.site.register(Tipo)

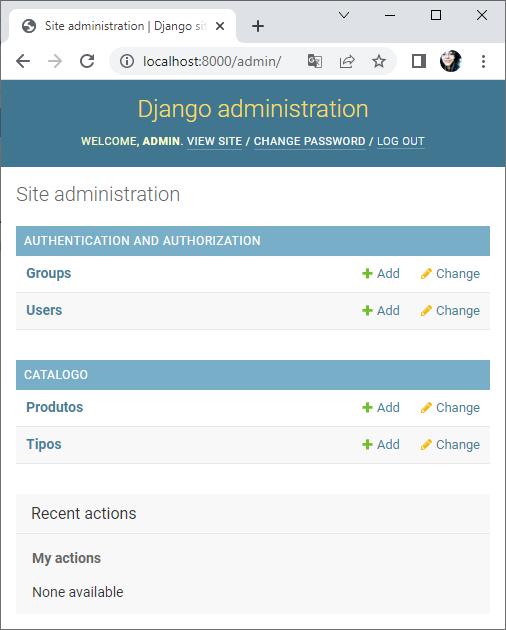
admin.site.register(Produto)

## Segundo: Acessar módulo admin

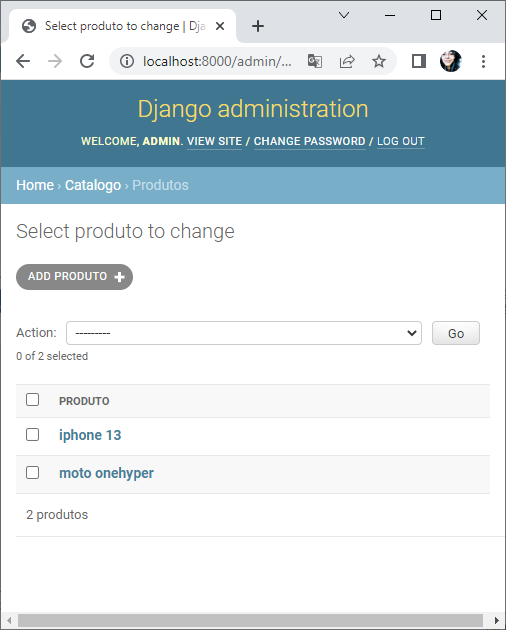
(venv) $loja> python manage.py runserver



## Terceiro: Realizar Login



## Quarto: Gerenciar Produtos



## Quarto: Gerenciar Tipos

