

PROGRAMAÇÃO BACK-END

django

The web framework for
perfectionists with deadlines.

PROGRAMAÇÃO BACK-END

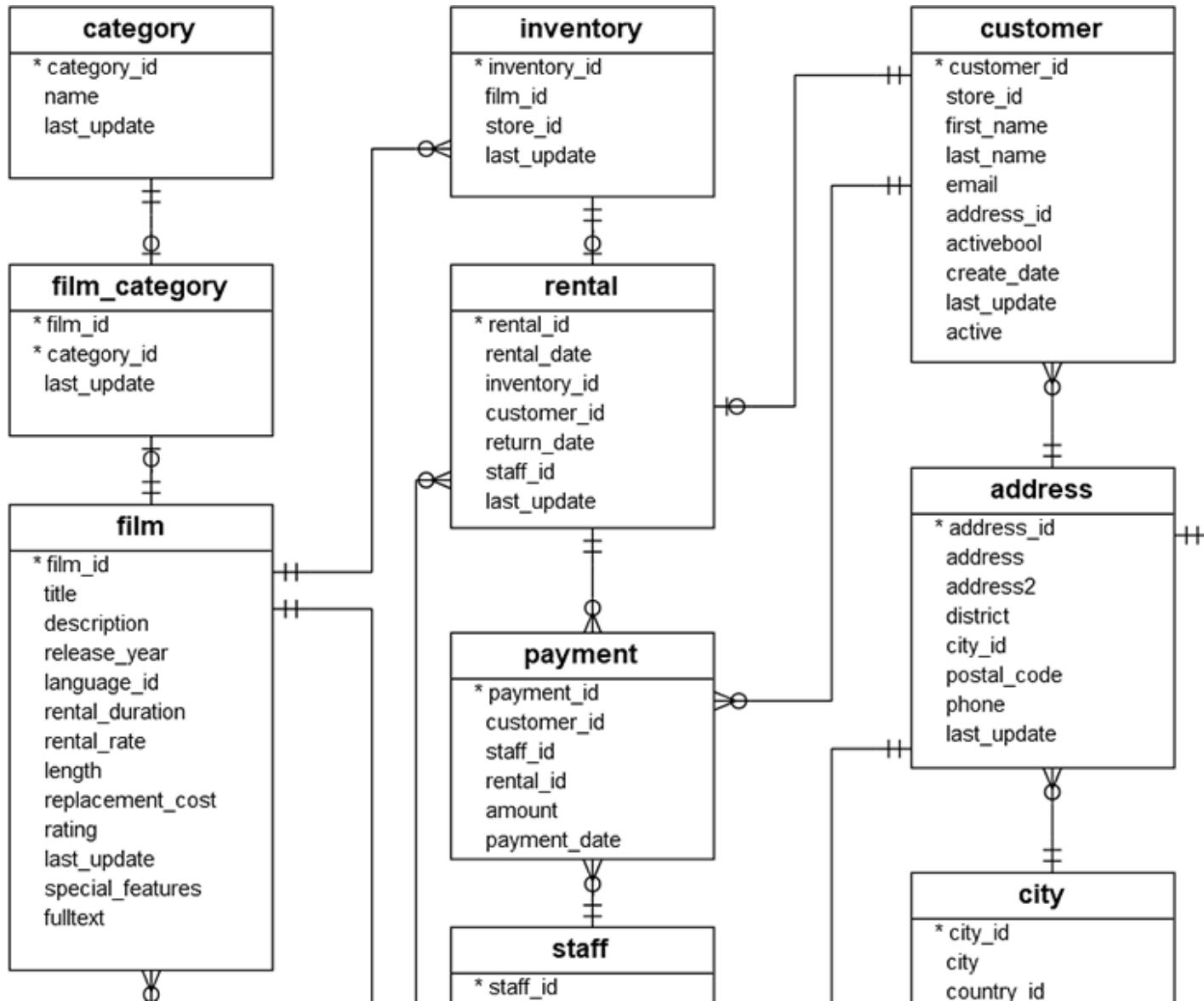
Conectar o Django
no PostgreSQL

PROGRAMAÇÃO BACK-END

Objetivos desta apresentação

Preparar um aplicativo **dvdrentalapp** para conectar no banco **dvdrental** do PostgreSQL e apresentar os dados da tabela **Customer** e quando selecionado, os detalhes da tabela **Rental**.

PROGRAMAÇÃO BACK-END

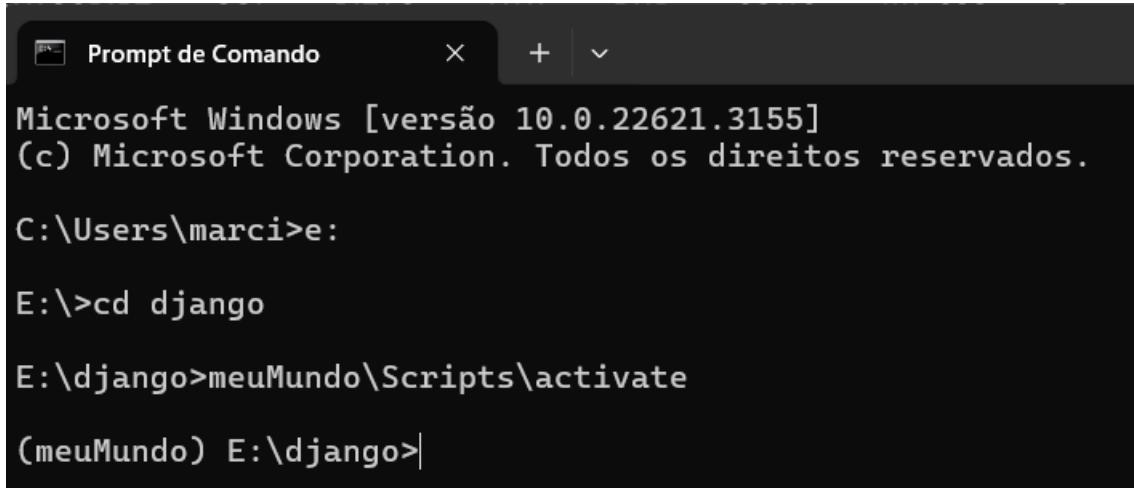


PROGRAMAÇÃO BACK-END

Iniciando

PROGRAMAÇÃO BACK-END

Rodar o script **activate.bat**



```
Prompt de Comando
Microsoft Windows [versão 10.0.22621.3155]
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Users\marci>e:

E:\>cd django

E:\django>meuMundo\Scripts\activate

(meuMundo) E:\django>
```

Este passo é importante para configurar as variáveis de ambiente.

PROGRAMAÇÃO BACK-END

Iniciar Servidor Postgresql

PROGRAMAÇÃO BACK-END

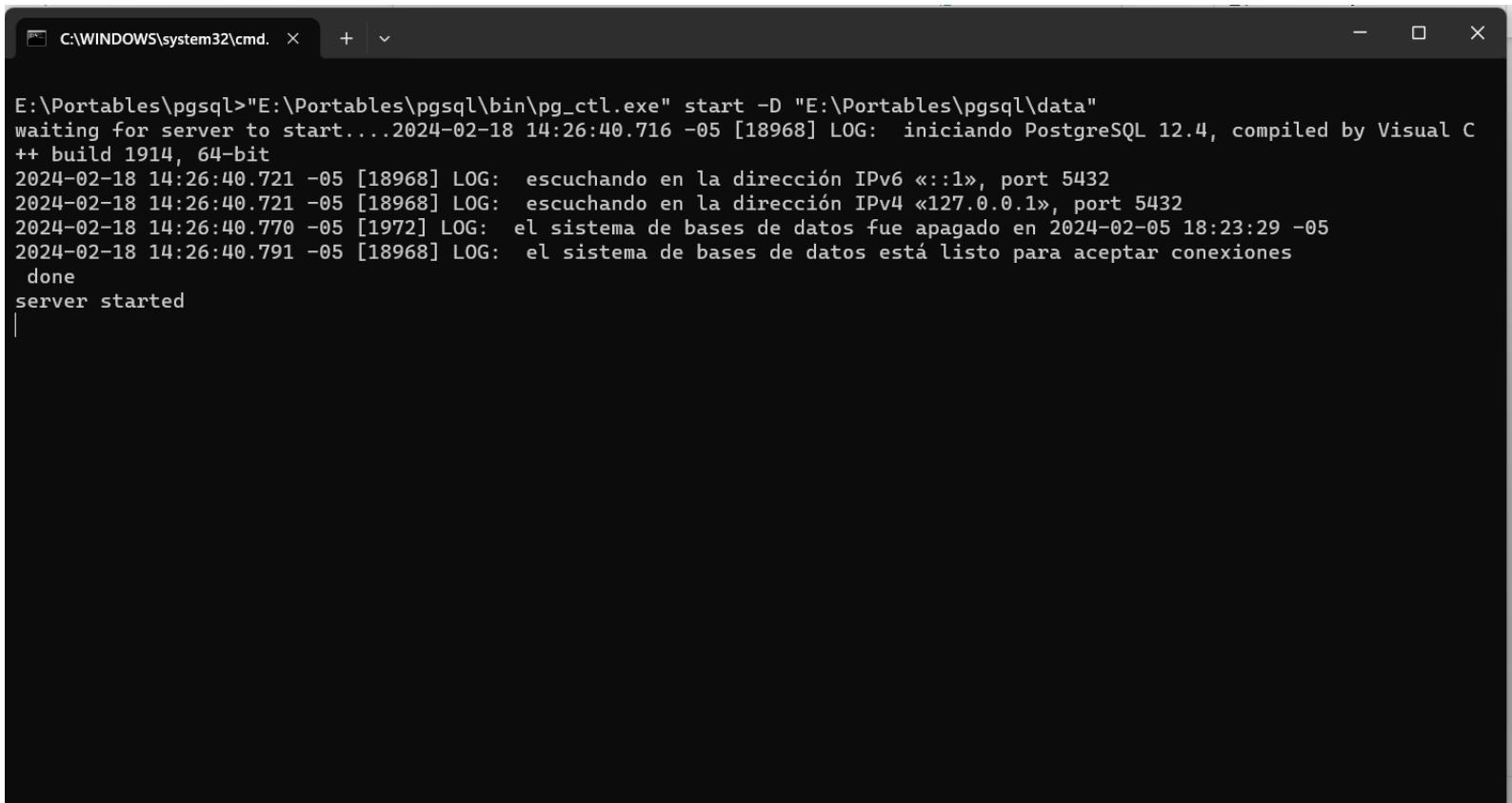
Na pasta `../pgsql` execute o arquivo `PostgreSQL-Start.bat` para iniciar o servidor



Nome	Data de modificação	Tipo	Tamanho
bin	11/08/2020 08:41	Pasta de arquivos	
data	18/02/2024 16:26	Pasta de arquivos	
doc	11/08/2020 08:41	Pasta de arquivos	
include	11/08/2020 08:41	Pasta de arquivos	
lib	11/08/2020 08:41	Pasta de arquivos	
pgAdmin 4	11/08/2020 08:41	Pasta de arquivos	
share	11/08/2020 08:41	Pasta de arquivos	
StackBuilder	11/08/2020 08:41	Pasta de arquivos	
Make Cluster.bat	09/01/2017 14:12	Arquivo em Lotes ...	1 KB
PgAdmin4.bat	28/04/2017 22:29	Arquivo em Lotes ...	1 KB
PostgreSQL-Restart.bat	06/04/2012 09:42	Arquivo em Lotes ...	1 KB
PostgreSQL-Start.bat	28/04/2017 21:46	Arquivo em Lotes ...	1 KB
PostgreSQL-Stop.bat	06/04/2012 09:43	Arquivo em Lotes ...	1 KB
Service - Delete.bat	29/08/2020 23:25	Arquivo em Lotes ...	1 KB
Service - Maker.bat	29/08/2020 23:26	Arquivo em Lotes ...	1 KB

PROGRAMAÇÃO BACK-END

Será aberta uma janela em Texto informando que o servidor iniciou (server started**).
Não feche esta janela**



A screenshot of a Windows Command Prompt window titled "C:\WINDOWS\system32\cmd.". The window contains the following text output from a PostgreSQL server start command:

```
E:\Portables\pgsql>"E:\Portables\pgsql\bin\pg_ctl.exe" start -D "E:\Portables\pgsql\data"
waiting for server to start....2024-02-18 14:26:40.716 -05 [18968] LOG:  iniciando PostgreSQL 12.4, compiled by Visual C
++ build 1914, 64-bit
2024-02-18 14:26:40.721 -05 [18968] LOG:  escuchando en la dirección IPv6 «::1», port 5432
2024-02-18 14:26:40.721 -05 [18968] LOG:  escuchando en la dirección IPv4 «127.0.0.1», port 5432
2024-02-18 14:26:40.770 -05 [1972] LOG:  el sistema de bases de datos fue apagado en 2024-02-05 18:23:29 -05
2024-02-18 14:26:40.791 -05 [18968] LOG:  el sistema de bases de datos está listo para aceptar conexiones
  done
server started
```

PROGRAMAÇÃO BACK-END

Iniciar o ambiente
de gerenciamento pgAdmin

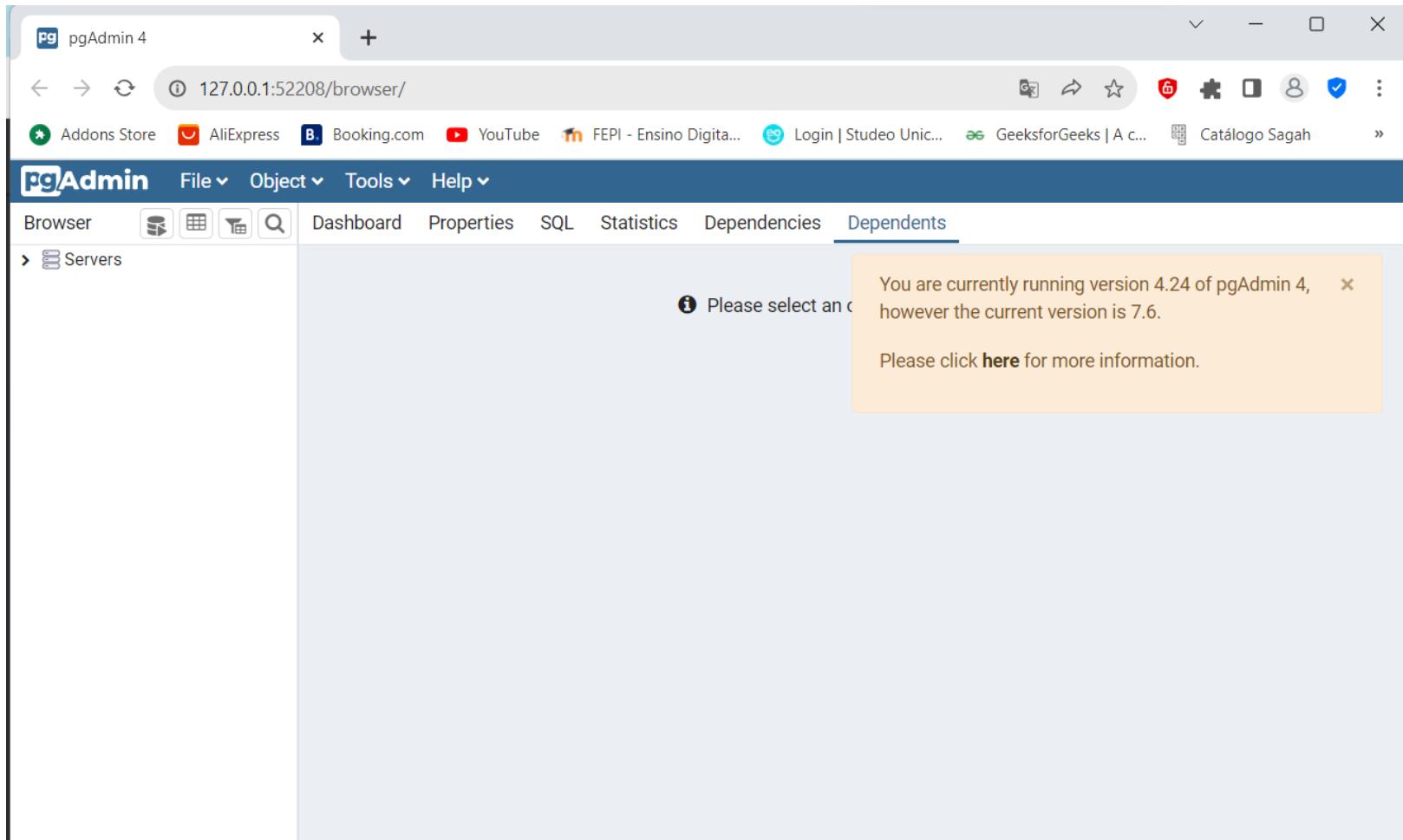
PROGRAMAÇÃO BACK-END

Na pasta ..\pgsql execute o arquivo pgAdmin4.bat

Nome	Data de modificação	Tipo	Tamanho
bin	11/08/2020 08:41	Pasta de arquivos	
data	18/02/2024 16:26	Pasta de arquivos	
doc	11/08/2020 08:41	Pasta de arquivos	
include	11/08/2020 08:41	Pasta de arquivos	
lib	11/08/2020 08:41	Pasta de arquivos	
pgAdmin 4	11/08/2020 08:41	Pasta de arquivos	
share	11/08/2020 08:41	Pasta de arquivos	
StackBuilder	11/08/2020 08:41	Pasta de arquivos	
Make Cluster.bat	09/01/2017 14:12	Arquivo em Lotes ...	1 KB
PgAdmin4.bat	28/04/2017 22:29	Arquivo em Lotes ...	1 KB
PostgreSQL-Restart.bat	06/04/2012 09:42	Arquivo em Lotes ...	1 KB
PostgreSQL-Start.bat	28/04/2017 21:46	Arquivo em Lotes ...	1 KB
PostgreSQL-Stop.bat	06/04/2012 09:43	Arquivo em Lotes ...	1 KB
Service - Delete.bat	29/08/2020 23:25	Arquivo em Lotes ...	1 KB
Service - Maker.bat	29/08/2020 23:26	Arquivo em Lotes ...	1 KB

PROGRAMAÇÃO BACK-END

O Navegador será aberto conforme a imagem.



PROGRAMAÇÃO BACK-END

1. Configurando o arquivo Para conectar no banco

**Caso esta etapa já foi realizada
não é necessário fazer.**

PROGRAMAÇÃO BACK-END

Na pasta `my_ads` altere a configuração do arquivo `settings.py` e coloque o conteúdo conforme a seguir:

#Para conexão ao servidor PostgreSQL, altere para:

```
 DATABASES = {  
     'default': {  
         'ENGINE': 'django.db.backends.postgresql',  
         'NAME': 'dvdrental',  
         'USER': 'postgres',  
         'PASSWORD': '12345',  
         'HOST': '127.0.0.1',  
         'PORT': '5432',  
     }  
 }
```

```
84 #Para conexão ao servidor PostgreSQL, altere para:  
85 DATABASES = {  
86     'default': {  
87         'ENGINE': 'django.db.backends.postgresql',  
88         'NAME': 'dvdrental',  
89         'USER': 'postgres',  
90         'PASSWORD': '12345',  
91         'HOST': '127.0.0.1',  
92         'PORT': '5432',  
93     }  
94 }
```

PROGRAMAÇÃO BACK-END

2. Criando a APP

**Caso esta etapa já foi realizada
não é necessário fazer.**

PROGRAMAÇÃO BACK-END

Criar a APP

Colocaremos o nome do aplicativo de **dvdrentalapp**.

Comece navegando até o local selecionado onde deseja armazenar o aplicativo, no caso a pasta **my_ads**, e execute o comando:

python manage.py startapp dvdrentalapp

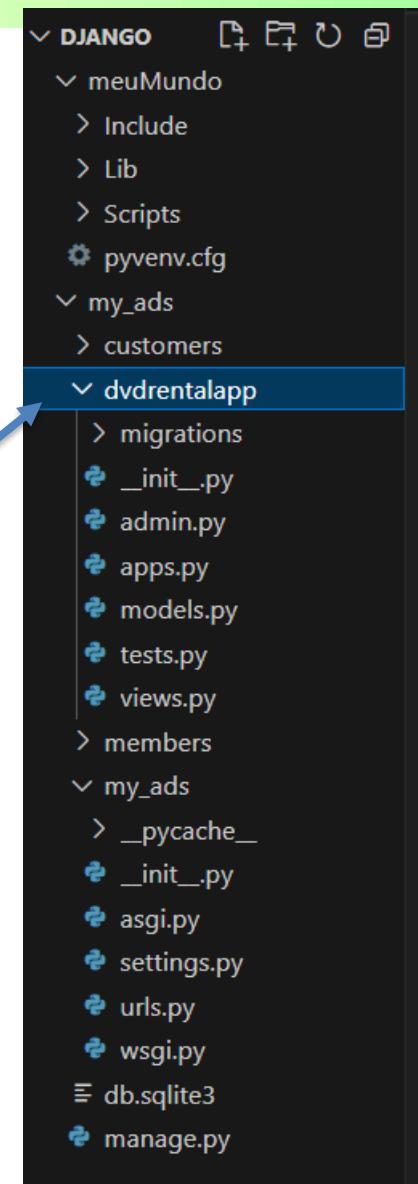
Obs: A(s) outra(s) pasta members criadas para os testes não influenciam a atual criada.

PROGRAMAÇÃO BACK-END

O Comando para criar a app via cmd e seu resultado.

```
(meuMundo) E:\django\my_ads>python manage.py startapp dvdrentalapp  
(meuMundo) E:\django\my_ads>
```

Pasta **dvdrentalapp** criada



PROGRAMAÇÃO BACK-END

3. Mapeando o banco de dados no Django

**Caso esta etapa já foi realizada
não é necessário fazer.**

PROGRAMAÇÃO BACK-END

Para mapear o banco de dados para o arquivo `models.py`, faça o seguinte procedimento:

Na pasta `my_ads` execute o seguinte comando:

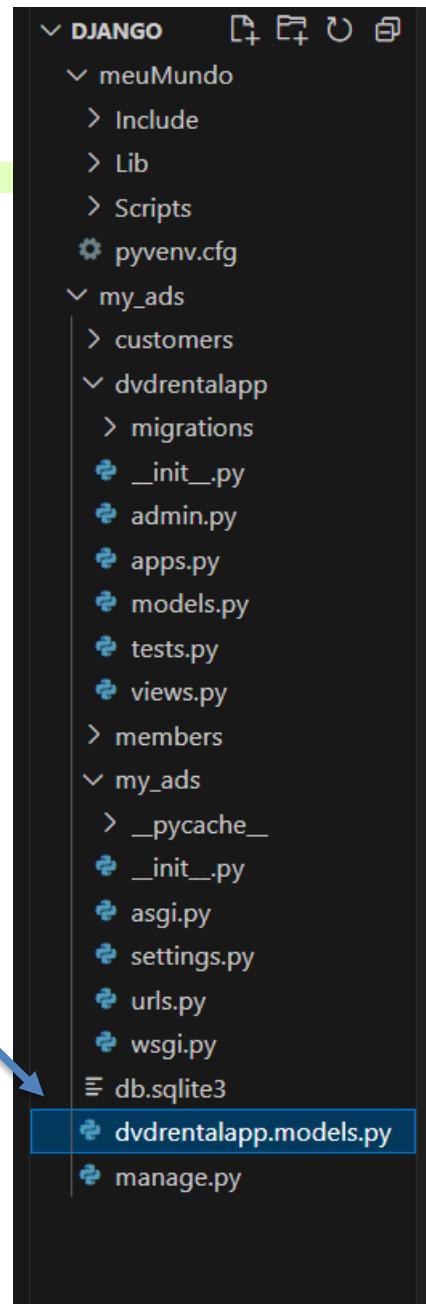
`python manage.py inspectdb > dvdrentalapp.models.py`

```
(meuMundo) E:\django\my_ads>python manage.py inspectdb > dvdrentalapp.models.py
```

```
(meuMundo) E:\django\my_ads>
```

PROGRAMAÇÃO BACK-END

O Comando cria o arquivo:
dvdrentalapp.models.py
na pasta **my_ads**



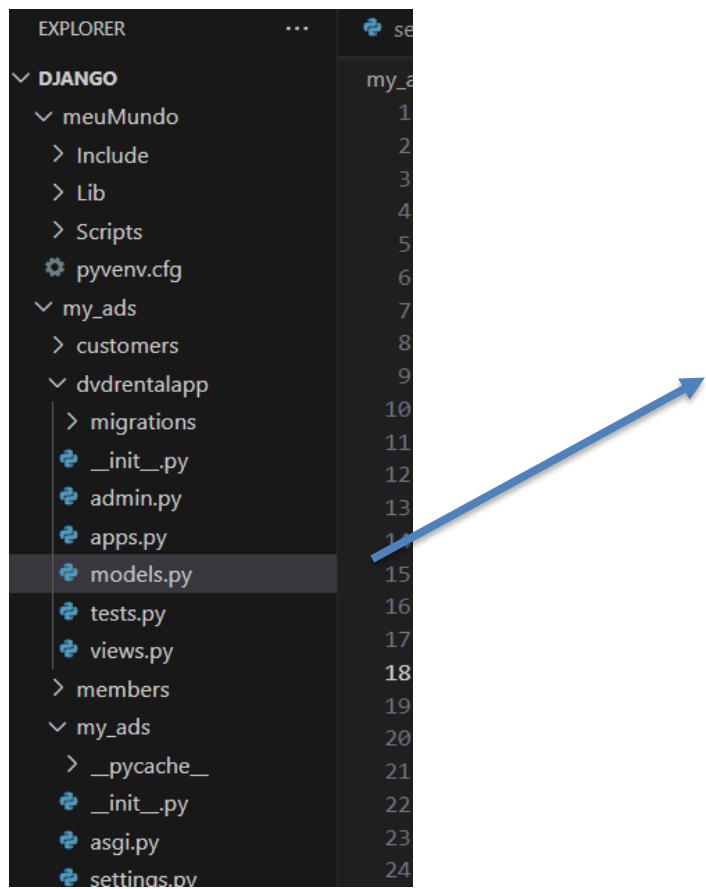
PROGRAMAÇÃO BACK-END

Verificando o conteúdo do arquivo **dvdrentalapp.models.py** pode-se observar as diversas **tabelas** que compõe o banco **dvdrental** e a estrutura de campos de cada tabela.

```
my_ads > dvrentalapp.models.py > ...
1  # This is an auto-generated Django model module.
2  # You'll have to do the following manually to clean this up:
3  #   * Rearrange models' order
4  #   * Make sure each model has one field with primary_key=True
5  #   * Make sure each ForeignKey and OneToOneField has `on_delete` set to the desire
6  #   * Remove `managed = False` lines if you wish to allow Django to create, modify,
7  #     # Feel free to rename the models, but don't rename db_table values or field names.
8  from django.db import models
9
10
11 class Actor(models.Model):
12     actor_id = models.AutoField(primary_key=True)
13     first_name = models.CharField(max_length=45)
14     last_name = models.CharField(max_length=45)
15     last_update = models.DateTimeField()
16
17     class Meta:
18         managed = False
19         db_table = 'actor'
20
21
22 class Address(models.Model):
23     address_id = models.AutoField(primary_key=True)
24     address = models.CharField(max_length=50)
25     address2 = models.CharField(max_length=50, blank=True, null=True)
26     district = models.CharField(max_length=20)
27     city = models.ForeignKey('City', models.DO_NOTHING)
28     postal_code = models.CharField(max_length=10, blank=True, null=True)
29     phone = models.CharField(max_length=20)
30     last_update = models.DateTimeField()
31
32     class Meta:
33         managed = False
34         db_table = 'address'
```

PROGRAMAÇÃO BACK-END

Faça a cópia do conteúdo do arquivo **dvdrentalapp.models.py** para o arquivo **models.py** que está na pasta **dvdrentalapp**.



```
settings.py models.py x dvdrentalapp.models.py
my_ads > dvdrentalapp > models.py > Address
1 # This is an auto-generated Django model module.
2 # You'll have to do the following manually to clean this up:
3 # * Rearrange models' order
4 # * Make sure each model has one field with primary_key=True
5 # * Make sure each ForeignKey and OneToOneField has `on_delete` set to:
6 # * Remove `managed = False` lines if you wish to allow Django to create
7 # Feel free to rename the models, but don't rename db_table values or fie
8 from django.db import models
9
10
11 class Actor(models.Model):
12     actor_id = models.AutoField(primary_key=True)
13     first_name = models.CharField(max_length=45)
14     last_name = models.CharField(max_length=45)
15     last_update = models.DateTimeField()
16
17     class Meta:
18         managed = False
19         db_table = 'actor'
20
21
22 class Address(models.Model):
23     address_id = models.AutoField(primary_key=True)
24     address = models.CharField(max_length=50)
25     address2 = models.CharField(max_length=50, blank=True, null=True)
26     district = models.CharField(max_length=20)
27     city = models.ForeignKey('city', models.DO_NOTHING)
28     postal_code = models.CharField(max_length=10, blank=True, null=True)
29     phone = models.CharField(max_length=20)
30     last_update = models.DateTimeField()
```

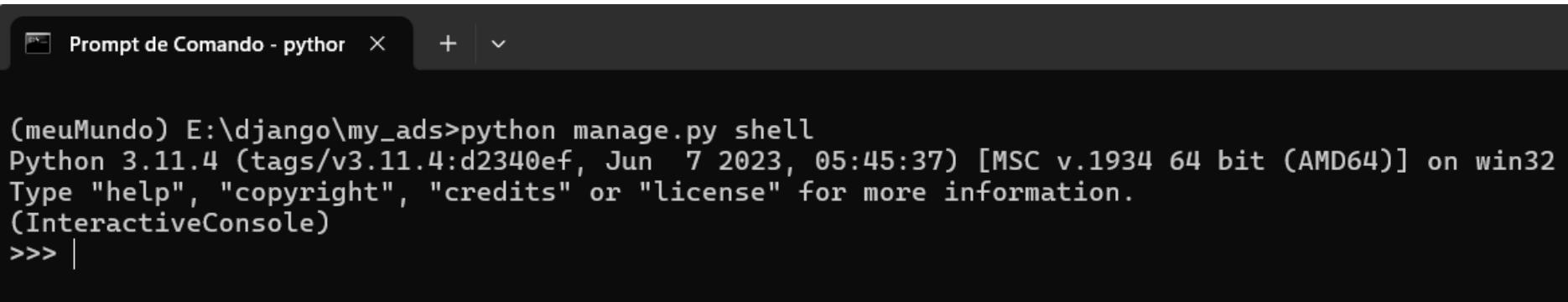
PROGRAMAÇÃO BACK-END

4. Testando a conexão
com o banco

PROGRAMAÇÃO BACK-END

Para testar a conexão, deve-se entrar pelo shell. Execute o seguinte comando:

my_ads> python manage.py shell



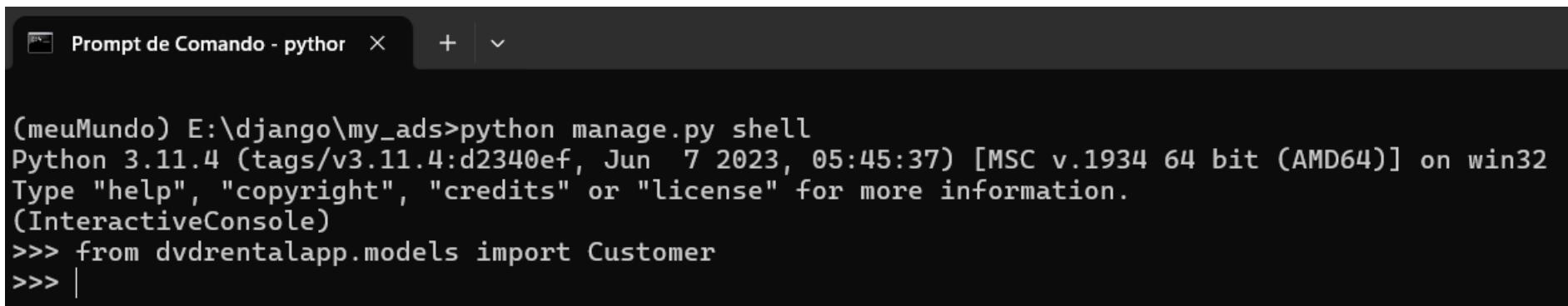
```
(meuMundo) E:\django\my_ads>python manage.py shell
Python 3.11.4 (tags/v3.11.4:d2340ef, Jun  7 2023, 05:45:37) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)
>>> |
```

PROGRAMAÇÃO BACK-END

Execute o seguinte comando:

>>> from dvdrentalapp.models import Customer

Este comando está importando a tabela Customer da APP `dvdrentalapp` no models (que está no arquivo `models.py`)



A screenshot of a terminal window titled "Prompt de Comando - python". The window shows the following command-line session:

```
(meuMundo) E:\django\my_ads>python manage.py shell
Python 3.11.4 (tags/v3.11.4:d2340ef, Jun  7 2023, 05:45:37) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)
>>> from dvdrentalapp.models import Customer
>>> |
```

PROGRAMAÇÃO BACK-END

Execute o seguinte comando:

>>> Customer.objects.all().values()

Este comando irá mostrar todo conteúdo da tabela **Customer conforme mostra a seguir.**

```
>>> Customer.objects.all().values()
<QuerySet [{<!-- customer_id: 524, store_id: 1, first_name: 'Jared', last_name: 'Ely', email: 'jared@sakilacustomer.org', address_id: 530, activebool: True, create_date: datetime.date(2006, 2, 14), last_update: datetime.datetime(2013, 5, 26, 14, 49, 45, 738000), active: 1}, {<!-- customer_id: 1, store_id: 1, first_name: 'Smith', email: 'mary.smith@sakilacustomer.org', address_id: 5, activebool: True, create_date: datetime.date(2006, 2, 14), last_update: datetime.datetime(2013, 5, 26, 14, 49, 45, 738000), active: 1}, {<!-- customer_id: 1, first_name: 'Patricia', last_name: 'Johnson', email: 'patricia.johnson@sakilacustomer.org', activebool: True, create_date: datetime.date(2006, 2, 14), last_update: datetime.datetime(2013, 5, 26, 14, 49, 45, 738000), active: 1}, {<!-- customer_id: 3, store_id: 1, first_name: 'Linda', last_name: 'Lindstrom', activebool: True, create_date: datetime.date(2006, 2, 14), last_update: datetime.datetime(2013, 5, 26, 14, 49, 45, 738000), active: 1}]}>
```

Para sair digite
>>> exit()

PROGRAMAÇÃO BACK-END

5. Criando os arquivos na
pasta templates

PROGRAMAÇÃO BACK-END

5.1. Arquivo all_customers.html

PROGRAMAÇÃO BACK-END

Na pasta `dvdrentalapp` subpasta `templates` crie/altere o arquivo `all_customers.html` com o seguinte conteúdo:

```
all_customers.html × urls.py dvdrentalapp urls.py my_ads detalhes.html views.py
dvdrentalapp > templates > all_customers.html > html
1  <!DOCTYPE html>
2  <html>
3  <body>
4
5  <h1>Customers</h1>
6
7  <ul>
8  |  {% for x in mycustomer %}
9  |  |  <li><a href="/detalhes/{{x.customer_id}}">{{ x.first_name }} {{ x.last_name }}</a></li>
10 |  |  {% endfor %}
11 </ul>
12
13 </body>
14 </html>
```

PROGRAMAÇÃO BACK-END

Verifique que no arquivo `all_customers.html`, foi acrescentado a tag `href`, do html, que permite direcionar para um nova chamada html, passando o `customer_id` que server de filtro da tabela **Rental que será utilizada.**

PROGRAMAÇÃO BACK-END

5.2. Arquivo detalhes.html

PROGRAMAÇÃO BACK-END

Na pasta **dvdrentalapp**
subpasta **templates**
crie o arquivo
detalhes.html com
o seguinte
conteúdo:

```
all_customers.html views.py detalhes.html X urls.py
my_ads > dvdrentalapp > templates > detalhes.html > html
1  <!DOCTYPE html>
2  <html>
3
4  <body>
5      <h2>Detalhes para {{ customer_name }}</h2>
6
7  <p>Back to <a href="/customer">Customers</a></p>
8
9  <table border="1">
10     <tr>
11         <th>inventory_id</th>
12         <th>rental_date</th>
13     </tr>
14     {% for x in myDetalhe %}
15     <tr>
16         <td>{{ x.inventory_id }}</td>
17         <td>{{ x.rental_date }}</td>
18     </tr>
19     {% endfor %}
20  </table>
21
22 </body>
23 </html>
```

PROGRAMAÇÃO BACK-END

Verifique que no arquivo **detalhes.html**, foi acrescentado a tag **href=“customer”**, que permite retornar a página anterior.

Serão apresentados somente dois campos (**inventory_id** e **rental_date**) como exemplo, mostrados na forma de tabela.

PROGRAMAÇÃO BACK-END

Preparando os arquivos
view.py, urls.py

PROGRAMAÇÃO BACK-END

**Na pasta
dvdrental
app altere
o arquivo
views.py
com o
seguinte
conteúdo:**

```
all_customers.html views.py detalhes.html urls.py

my_ads > dvdrentalapp > views.py > detalhes
1 # Create your views here.
2 from django.http import HttpResponseRedirect
3 from django.template import loader
4 from .models import Customer, Rental
5 from django.shortcuts import get_object_or_404
6
7 def customer(request):
8     mycustomers = Customer.objects.all().values()
9     template = loader.get_template('all_customers.html')
10    context = {
11        'mycustomer': mycustomers,
12    }
13    return HttpResponseRedirect(template.render(context, request))
14
15 def detalhes(request, id):
16     myDetalhes = Rental.objects.filter(customer_id=id)
17     customer = get_object_or_404(Customer, pk=id)
18
19     template = loader.get_template('detalhes.html')
20     context = {
21         'myDetalhe': myDetalhes,
22         'customer_name' : f'{customer.first_name} {customer.last_name}',
23     }
24     return HttpResponseRedirect(template.render(context, request))
```

PROGRAMAÇÃO BACK-END

Observe o conteúdo da função **detalhes**:

Na linha 4 o comando está importando as tabelas **Customer** e **Rental** do modelo.

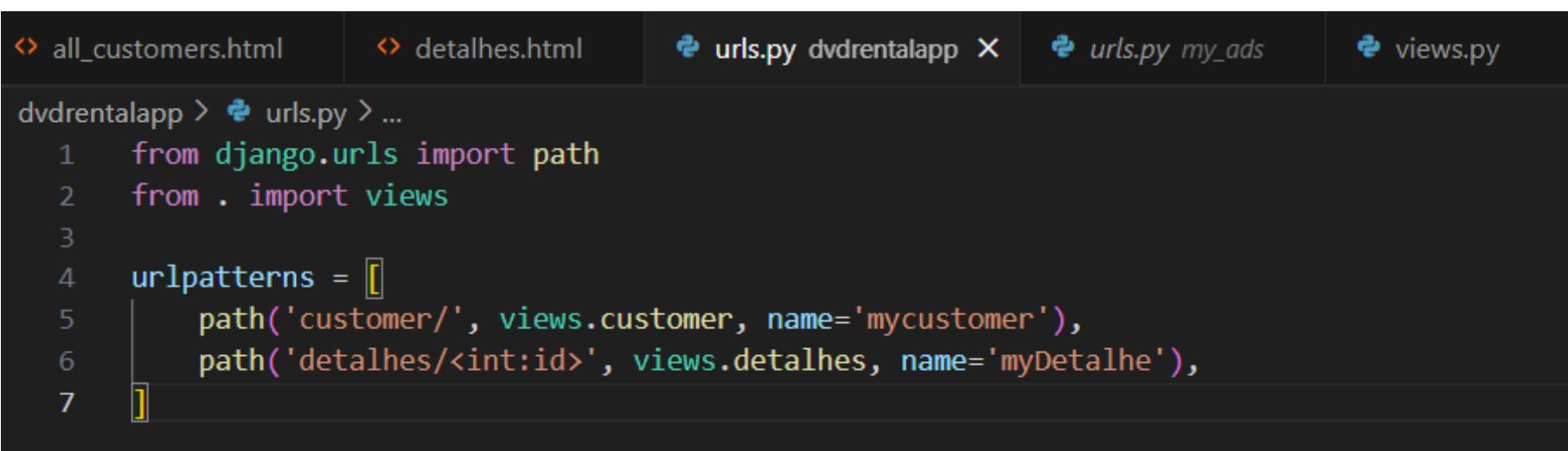
```
15 def detalhes(request, id):
16     myDetalhes = Rental.objects.filter(customer_id=id)
17     customer = get_object_or_404(Customer, pk=id)
18     template = loader.get_template('detalhes.html')
19     context = {
20         'myDetalhe': myDetalhes,
21         'customer_name' : f'{customer.first_name} {customer.last_name}',
22     }
23     return HttpResponseRedirect(template.render(context, request))
```

Na linha 16 cria-se a variável **myDetalhes** que importa da tabela **Rental** aplicando um filtro (filter) sendo **customer_id = id**. Este **id** é recebido da página html **all_customer.html**:

`href="/detalhes/{{x.customer_id}}"`

PROGRAMAÇÃO BACK-END

Na pasta `dvdrentalapp` crie um arquivo `urls.py` com o seguinte conteúdo:



The screenshot shows a code editor with multiple tabs at the top: 'all_customers.html', 'detalhes.html', 'urls.py dvdrentalapp', 'urls.py my_ads', and 'views.py'. The 'urls.py dvdrentalapp' tab is active. The code in the editor is as follows:

```
1  from django.urls import path
2  from . import views
3
4  urlpatterns = [
5      path('customer/', views.customer, name='mycustomer'),
6      path('detalhes/<int:id>', views.detalhes, name='myDetalhe'),
7 ]
```

Verifique que `detalhes/<int:id>` será passado um id para filtragem.

PROGRAMAÇÃO BACK-END

Na pasta `my_ads` altere/verifique o arquivo `urls.py` com o seguinte conteúdo:

```
17 from django.contrib import admin
18 from django.urls import include, path
19
20 urlpatterns = [
21     path('', include('dvdrentalapp.urls')),
22     path('admin/', admin.site.urls),
23 ]
```

PROGRAMAÇÃO BACK-END

Na pasta `my_ads` altere/verifique o arquivo `settings.py` acrescentando o APP `dvdrentalapp`

```
33     INSTALLED_APPS = [
34         'django.contrib.admin',
35         'django.contrib.auth',
36         'django.contrib.contenttypes',
37         'django.contrib.sessions',
38         'django.contrib.messages',
39         'django.contrib.staticfiles',
40         'dvdrentalapp',
41     ]
```

PROGRAMAÇÃO BACK-END

Execute na pasta my_ads o comando:

python manage.py migrate

```
(meuMundo) E:\django\my_ads>python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying sessions.0001_initial... OK
```

```
(meuMundo) E:\django\my_ads>
```

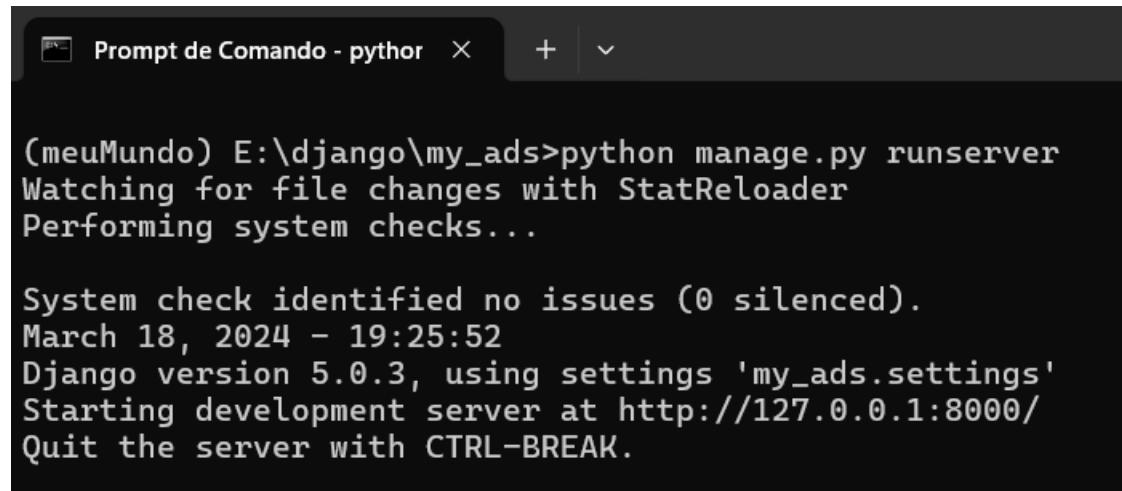
PROGRAMAÇÃO BACK-END

Iniciando o servidor
e testando

PROGRAMAÇÃO BACK-END

Execute na pasta my_ads o comando:

python manage.py runserver



```
(meuMundo) E:\django\my_ads>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
March 18, 2024 - 19:25:52
Django version 5.0.3, using settings 'my_ads.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

Vá ao navegador e digite:

<http://127.0.0.1:8000/customer/>

PROGRAMAÇÃO BACK-END

O resultado no navegador



Customers

- [Jared Ely](#)
- [Mary Smith](#)
- [Patricia Johnson](#)
- [Linda Williams](#)
- [Barbara Jones](#)
- [Elizabeth Brown](#)
- [Jennifer Davis](#)
- [Maria Miller](#)
- [Susan Wilson](#)
- [Margaret Moore](#)
- [Dorothy Taylor](#)
- [Lisa Anderson](#)
- [Nancy Thomas](#)
- [Karen Jackson](#)
- [Betty White](#)
- [Helen Harris](#)
- [Sandra Martin](#)
- [Donna Thompson](#)
- [Carol Garcia](#)
- [Ruth Martinez](#)



Detalhes para Jared Ely

Back to [Customers](#)

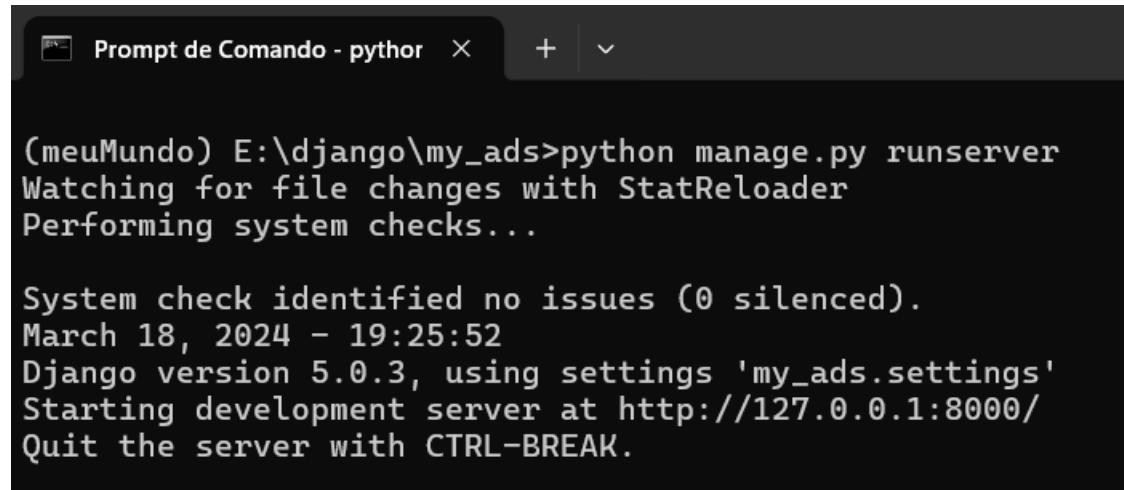
inventory_id	rental_date
872	May 25, 2005, 7:31 p.m.
3560	May 30, 2005, 10:15 p.m.
2132	June 15, 2005, 9:59 a.m.
4444	June 16, 2005, 9:24 a.m.
3009	June 21, 2005, 9:12 p.m.
2961	July 7, 2005, 7:48 p.m.
435	July 9, 2005, 2:59 a.m.
3681	July 11, 2005, 10:11 a.m.
640	July 11, 2005, 2:32 p.m.
1331	July 12, 2005, 2:30 p.m.
714	July 27, 2005, 2:14 a.m.
470	July 27, 2005, 3:17 a.m.
3273	July 29, 2005, 9:29 a.m.
594	Aug. 20, 2005, 6:55 a.m.
2352	Aug. 20, 2005, 9:53 p.m.
2178	Aug. 21, 2005, 3:13 a.m.
4011	Aug. 21, 2005, 9:31 a.m.
4469	Aug. 21, 2005, 8:19 p.m.
3901	Aug. 22, 2005, 4:33 p.m.

PROGRAMAÇÃO BACK-END

Desafio

PROGRAMAÇÃO BACK-END

Crie outra APP que acesse outra tabela de detalhes do banco, mostrando os seus dados.



```
(meuMundo) E:\django\my_ads>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
March 18, 2024 - 19:25:52
Django version 5.0.3, using settings 'my_ads.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```