

PROGRAMAÇÃO BACK-END

The Django logo, featuring the word "django" in a white, lowercase, sans-serif font with a small dot over the 'i', set against a dark green rectangular background.

The web framework for
perfectionists with deadlines.

Django Admin

PROGRAMAÇÃO BACK-END

Administrador Django

Django Admin é uma ferramenta no Django, ou seja, é uma interface de usuário CRUD de todos os seus modelos!

CRUD significa: Criar, Ler, Atualizar, Excluir.

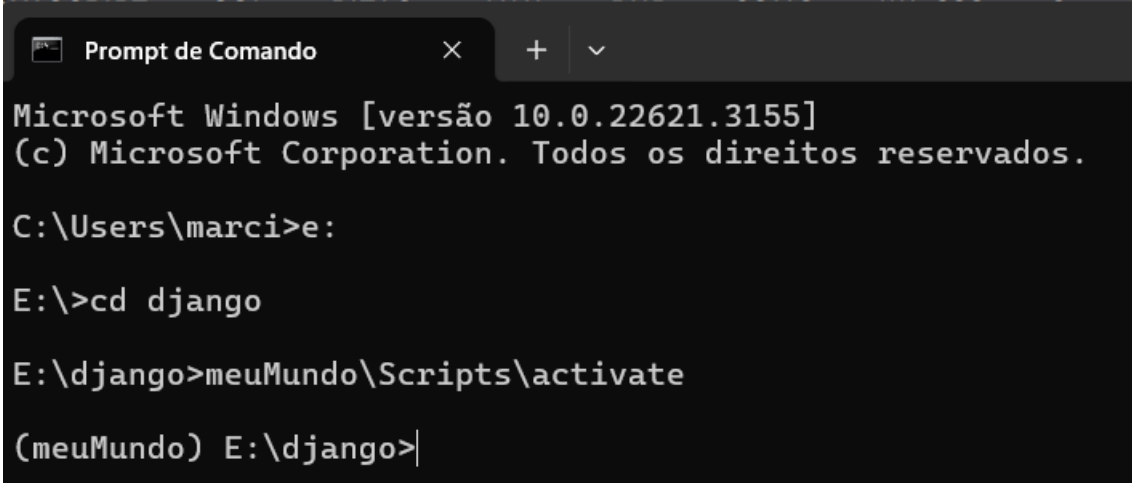
(Create, Read, Update, Delete)

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

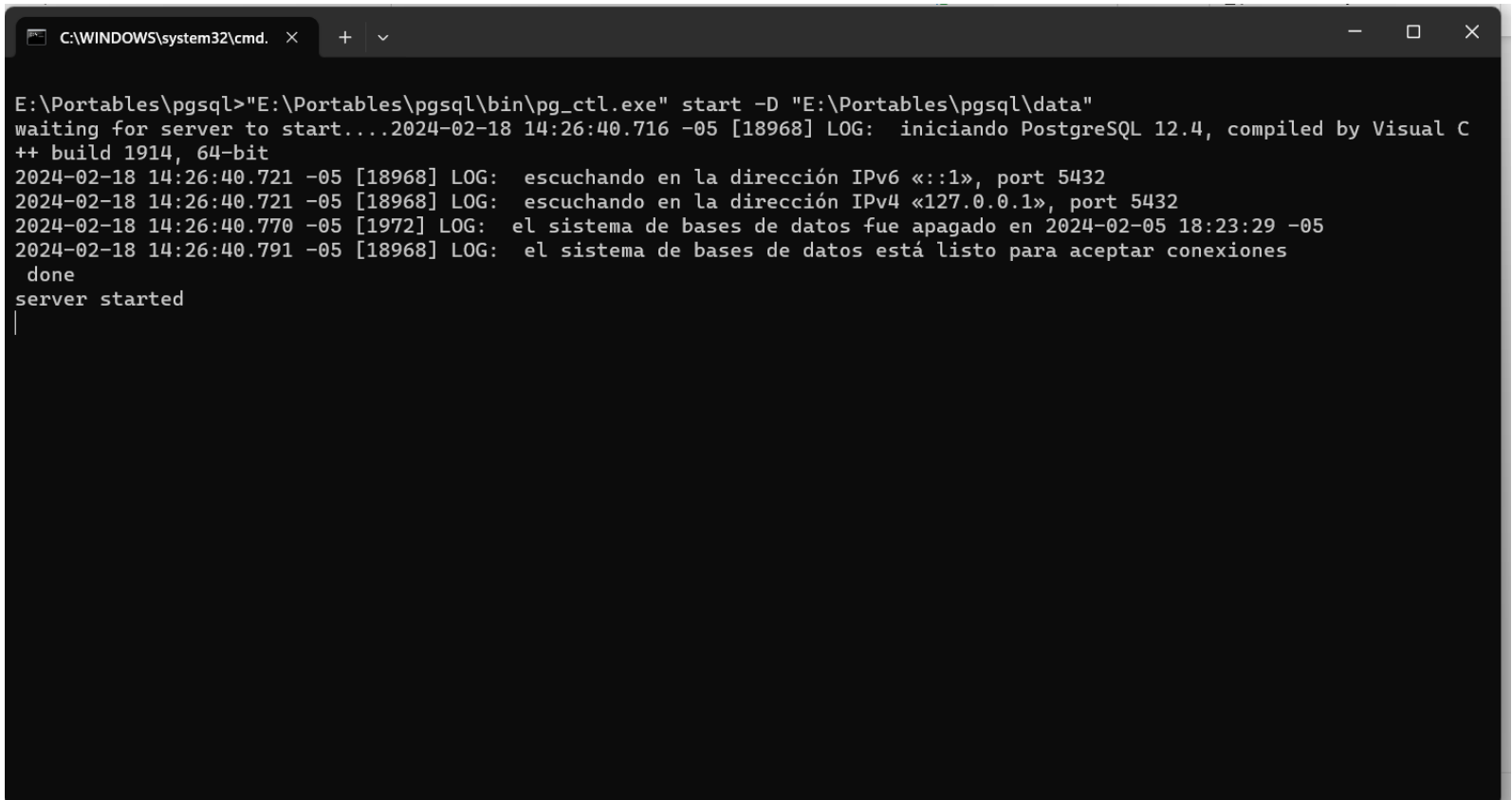
Na pasta `../pgsql` execute o arquivo **PostgreSQL-Start.bat** para iniciar o servidor



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. The title bar shows 'C:\WINDOWS\system32\cmd.' with standard window controls. The command prompt shows the execution of 'pg_ctl.exe' to start a PostgreSQL server. The output includes log messages indicating the server is starting, listening on IPv6 and IPv4, and that the database system is ready for connections. The final output is 'server started' followed by a cursor.

```
C:\WINDOWS\system32\cmd. x + v

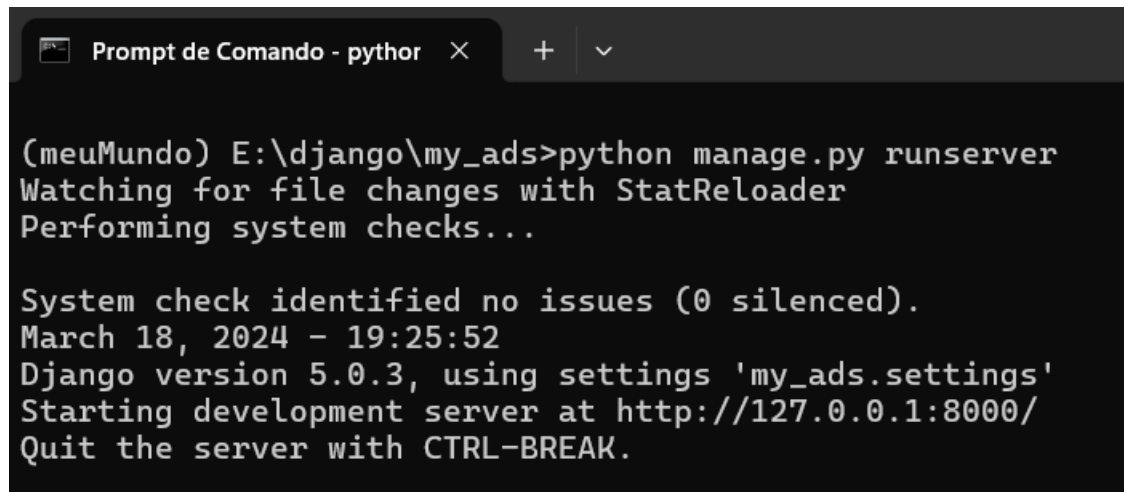
E:\Portables\pgsql>"E:\Portables\pgsql\bin\pg_ctl.exe" start -D "E:\Portables\pgsql\data"
waiting for server to start...2024-02-18 14:26:40.716 -05 [18968] LOG:  iniciando PostgreSQL 12.4, compiled by Visual C
++ build 1914, 64-bit
2024-02-18 14:26:40.721 -05 [18968] LOG:  escuchando en la direcci3n IPv6 «::1», port 5432
2024-02-18 14:26:40.721 -05 [18968] LOG:  escuchando en la direcci3n 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 est1 listo para aceptar conexiones
done
server started
|
```


**Iniciando o servidor
e testando**

PROGRAMAÇÃO BACK-END

Execute na pasta my_ads o comando:

python manage.py runserver



```
Prompt de Comando - pythor  X  +  v

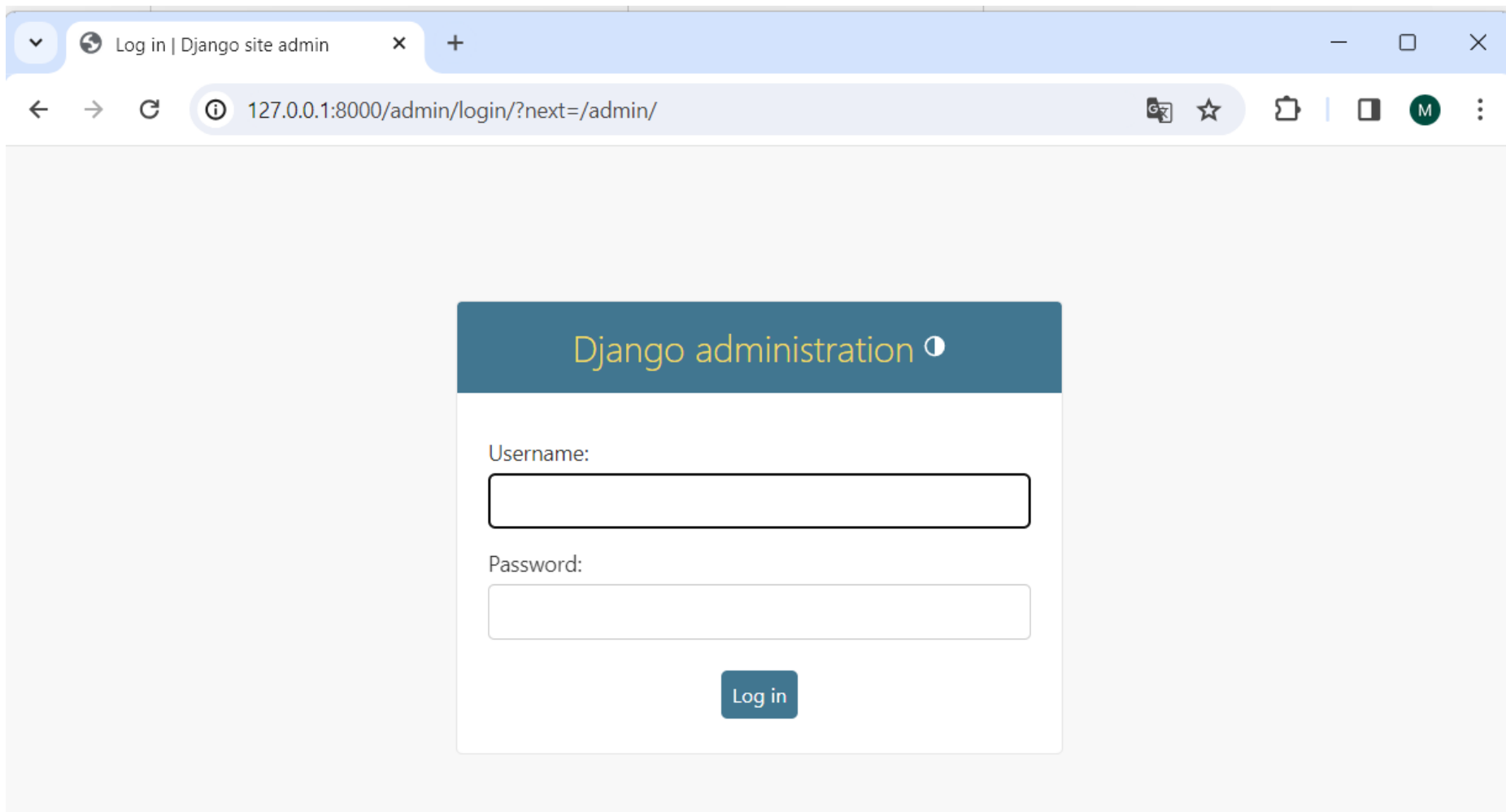
(meuMundo) E:\django\my_ads>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

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/admin/>

PROGRAMAÇÃO BACK-END



The image shows a web browser window with the title "Log in | Django site admin". The address bar displays the URL "127.0.0.1:8000/admin/login/?next=/admin/". The main content area features a login form titled "Django administration". The form includes a "Username:" label, a text input field, a "Password:" label, another text input field, and a "Log in" button.

Log in | Django site admin

127.0.0.1:8000/admin/login/?next=/admin/

Django administration

Username:

Password:

Log in

É necessário ter um **username** e **password** para logar. A seguir será criado um usuário admin.

PROGRAMAÇÃO BACK-END

A razão pela qual está URL vai para a página de login do administrador do Django pode ser encontrada no arquivo **urls.py** do projeto (pasta **my_ads**).

```
..  <> all_customers.html  urls.py dvdrentalapp  urls.py my_ads X  views.py
my_ads >  urls.py > ...
1  from django.contrib import admin
2  from django.urls import include, path
3
4  urlpatterns = [
5      path('', include('dvdrentalapp.urls')),
6      path('admin/', admin.site.urls),
7  ]
```

PROGRAMAÇÃO BACK-END

Criando o usuário

PROGRAMAÇÃO BACK-END

Criar usuário

Para poder fazer login no aplicativo administrativo, precisamos criar um usuário.

Isso é feito digitando este comando na visualização de comando:

`python manage.py createsuperuser`

Digite os dados solicitados: username, email, password, e confirme com y

PROGRAMAÇÃO BACK-END

```
Prompt de Comando

(meuMundo) E:\django\my_ads>code .

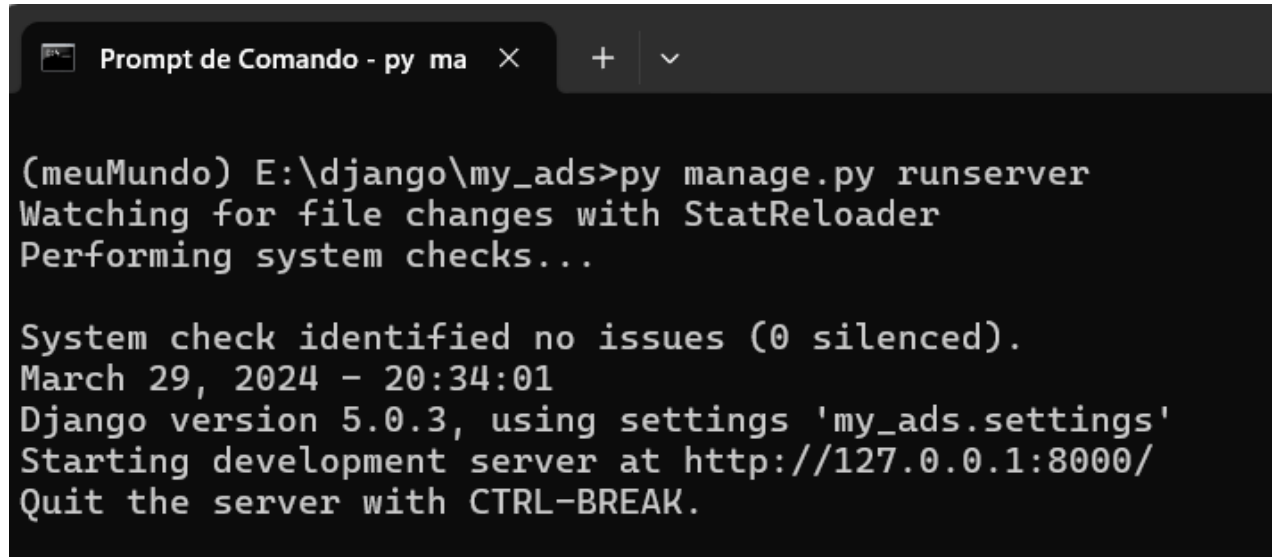
(meuMundo) E:\django\my_ads>py manage.py createsuperuser
Username (leave blank to use 'marci'): admin
Email address: eng.mecanica.fepi@gmail.com
Password:
Password (again):
This password is too short. It must contain at least 8 characters.
This password is too common.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.

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

PROGRAMAÇÃO BACK-END

Inicie o servidor. Execute na pasta my_ads o comando:

python manage.py runserver



```
Prompt de Comando - py ma  ×  +  v

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

System check identified no issues (0 silenced).
March 29, 2024 - 20:34:01
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.
```

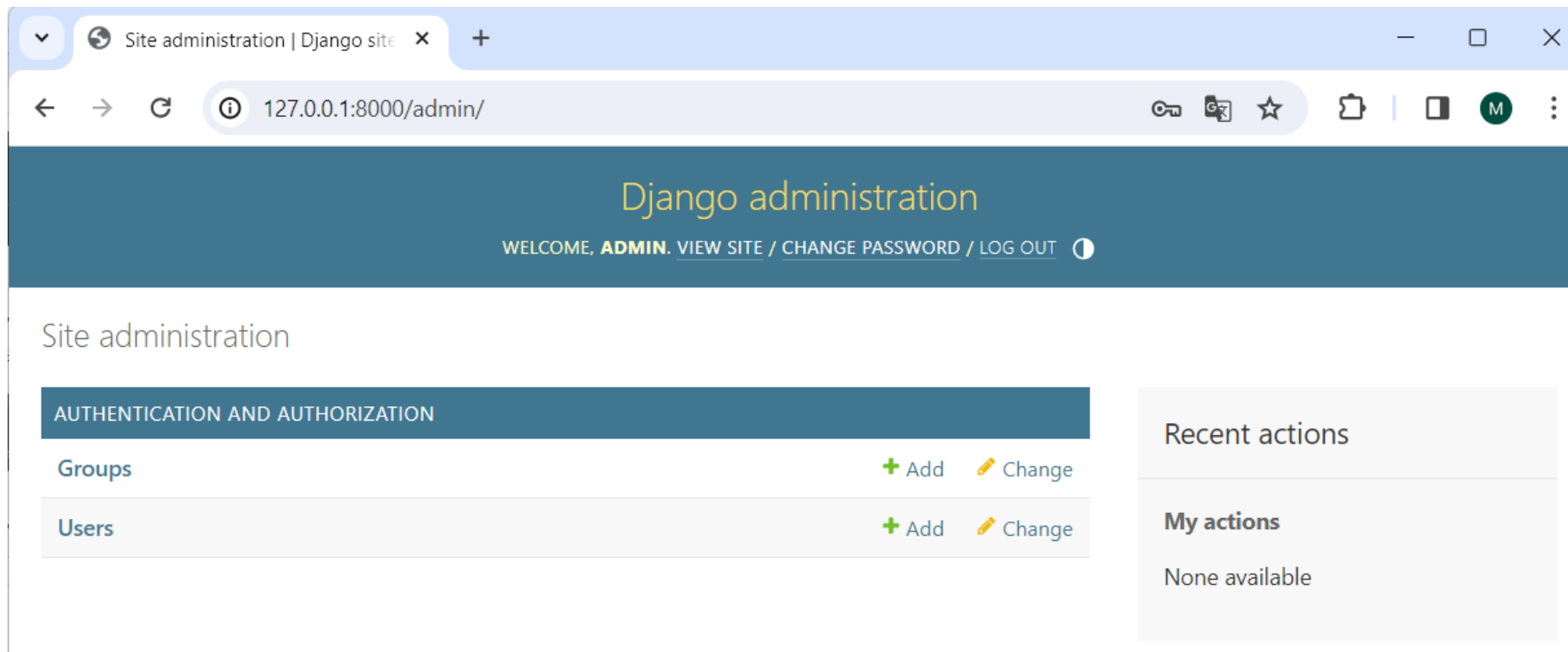
Vá ao navegador e digite:

<http://127.0.0.1:8000/admin/>

Logar com o usuário criado

PROGRAMAÇÃO BACK-END

O que deve resultar nesta interface de usuário:



Modelo ausente

Deve-se informar ao Django quais modelos devem estar visíveis na interface administrativa.

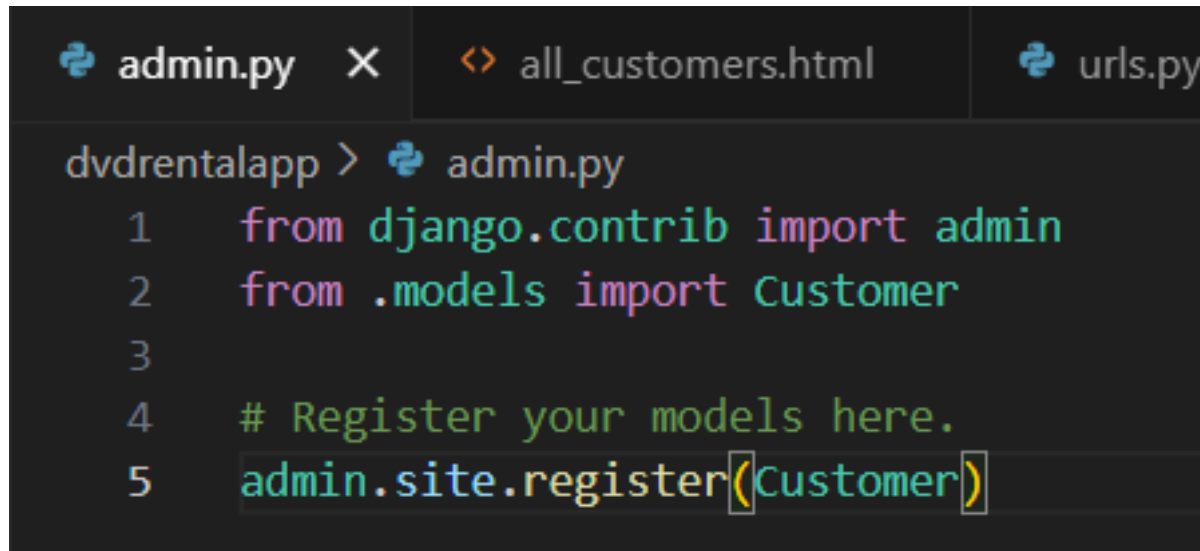
PROGRAMAÇÃO BACK-END

Para incluir o modelo **Customer** na interface administrativa, temos que dizer ao Django que este modelo deve estar visível na interface administrativa.

Isso é feito em um arquivo chamado **admin.py**, e está localizado na pasta do seu aplicativo, que no nosso caso é a pasta **dvdrentalapp**.

Abra-o e deve ficar assim:

PROGRAMAÇÃO BACK-END



The screenshot shows a code editor with three tabs at the top: 'admin.py', 'all_customers.html', and 'urls.py'. The 'admin.py' tab is active. The code in the editor is as follows:

```
dvdrentalapp > admin.py
1  from django.contrib import admin
2  from .models import Customer
3
4  # Register your models here.
5  admin.site.register(Customer)
```

Após a alteração, inicie o servidor. Execute na pasta **my_ads** o comando:

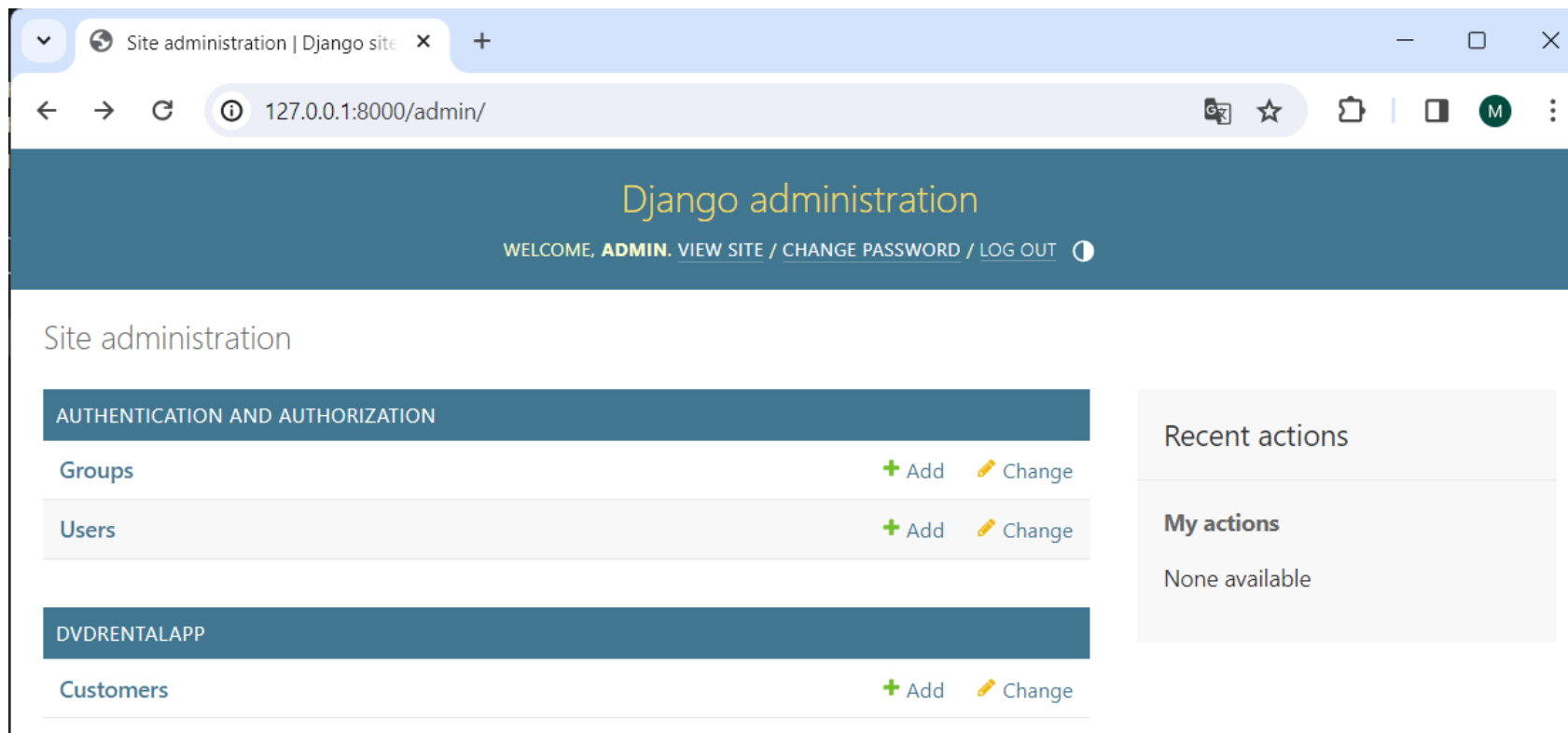
python manage.py runserver

Vá ao navegador e digite:

<http://127.0.0.1:8000/admin/>

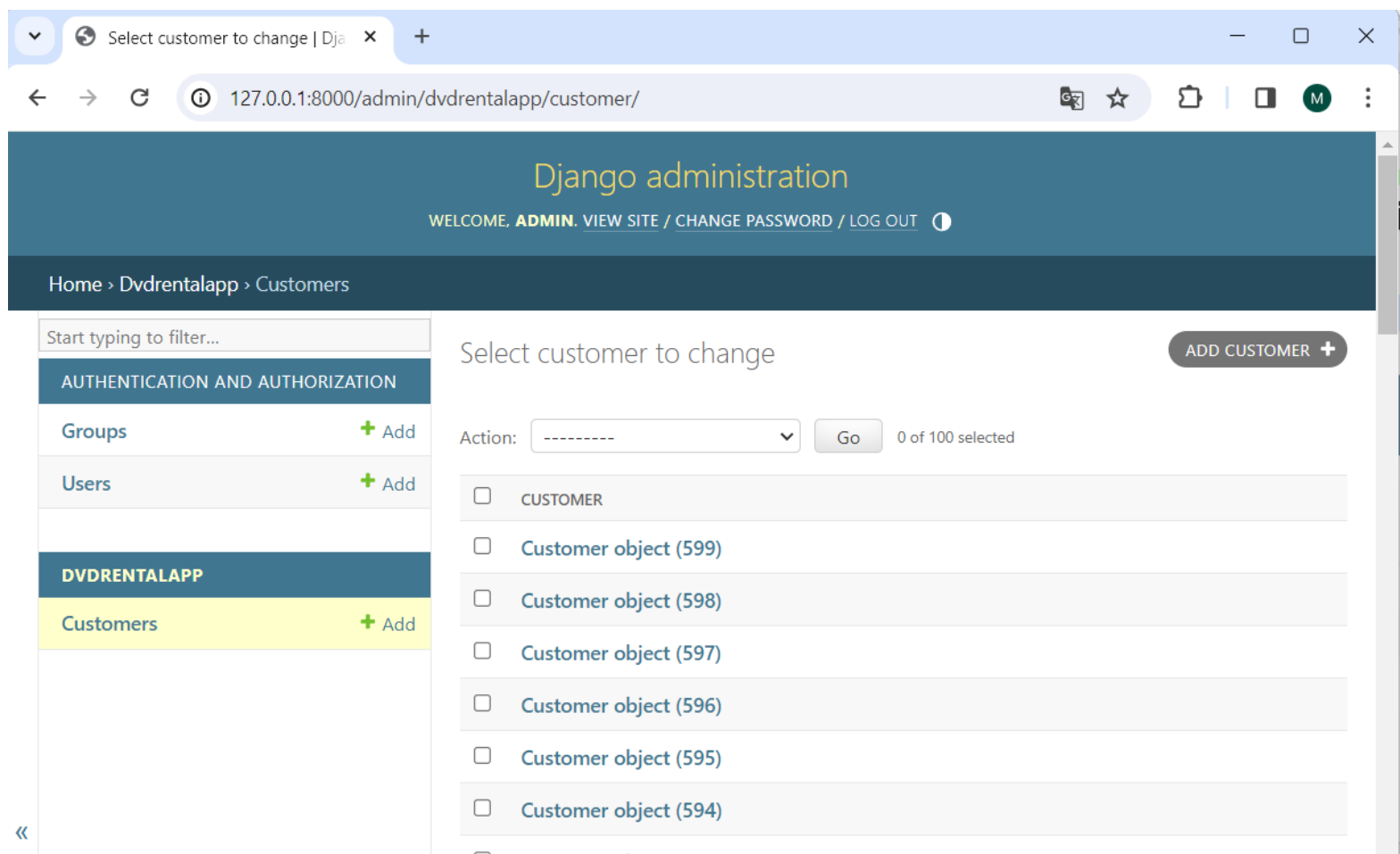
PROGRAMAÇÃO BACK-END

O que deve resultar nesta interface de usuário:



PROGRAMAÇÃO BACK-END

Clique em **Customer** e veja os registros da tabela no banco.



PROGRAMAÇÃO BACK-END

Na lista da captura de tela acima, vemos **Customer Objeto (599)**, **Customer Objeto (598)** etc, que podem não ser os dados que se deseja que sejam exibidos na listagem.

Seria melhor exibir **first_name** e **last_name**.

Verifica-se também que podem ser acrescentados novos Customers, deletados e alterados por meio da interface.

PROGRAMAÇÃO BACK-END

Para mudar num formato mais fácil de ler, temos duas opções:

- 1. Alterar a função de representação de string `__str__()` do modelo de Customer.**
- 2. Defina a propriedade `list_details` do modelo de Customer.**

PROGRAMAÇÃO BACK-END

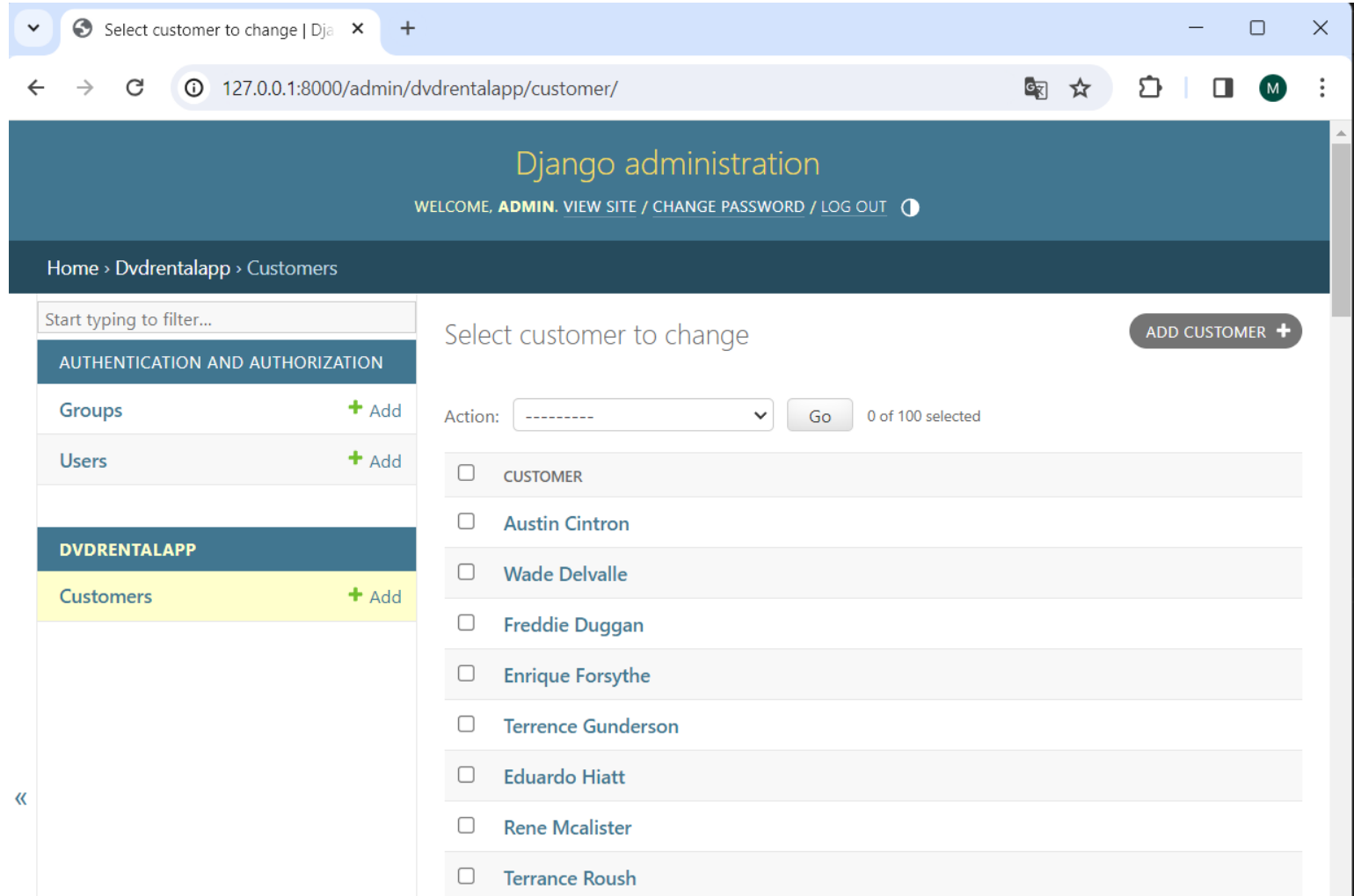
1. Alterar a função de representação de string

Para alterar a representação da string, temos que definir a função `__str__()` do Model **Customer** em **models.py**, assim:

```
86 class Customer(models.Model):
87     customer_id = models.AutoField(primary_key=True)
88     store_id = models.SmallIntegerField()
89     first_name = models.CharField(max_length=45)
90     last_name = models.CharField(max_length=45)
91     email = models.CharField(max_length=50, blank=True, null=True)
92     address = models.ForeignKey(Address, models.DO_NOTHING)
93     activebool = models.BooleanField()
94     create_date = models.DateField()
95     last_update = models.DateTimeField(blank=True, null=True)
96     active = models.IntegerField(blank=True, null=True)
97
98     class Meta:
99         managed = False
100         db_table = 'customer'
101
102     def __str__(self):
103         return f"{self.first_name} {self.last_name}"
104
```


PROGRAMAÇÃO BACK-END

O resultado das alterações são:



PROGRAMAÇÃO BACK-END

2. Definir `list_display`

Podemos controlar os campos a serem exibidos especificando-os em uma propriedade `list_display` do arquivo `admin.py`.

Primeiro crie uma classe `CustomerAdmin()` e especifique a tupla `list_display`, assim:

PROGRAMAÇÃO BACK-END

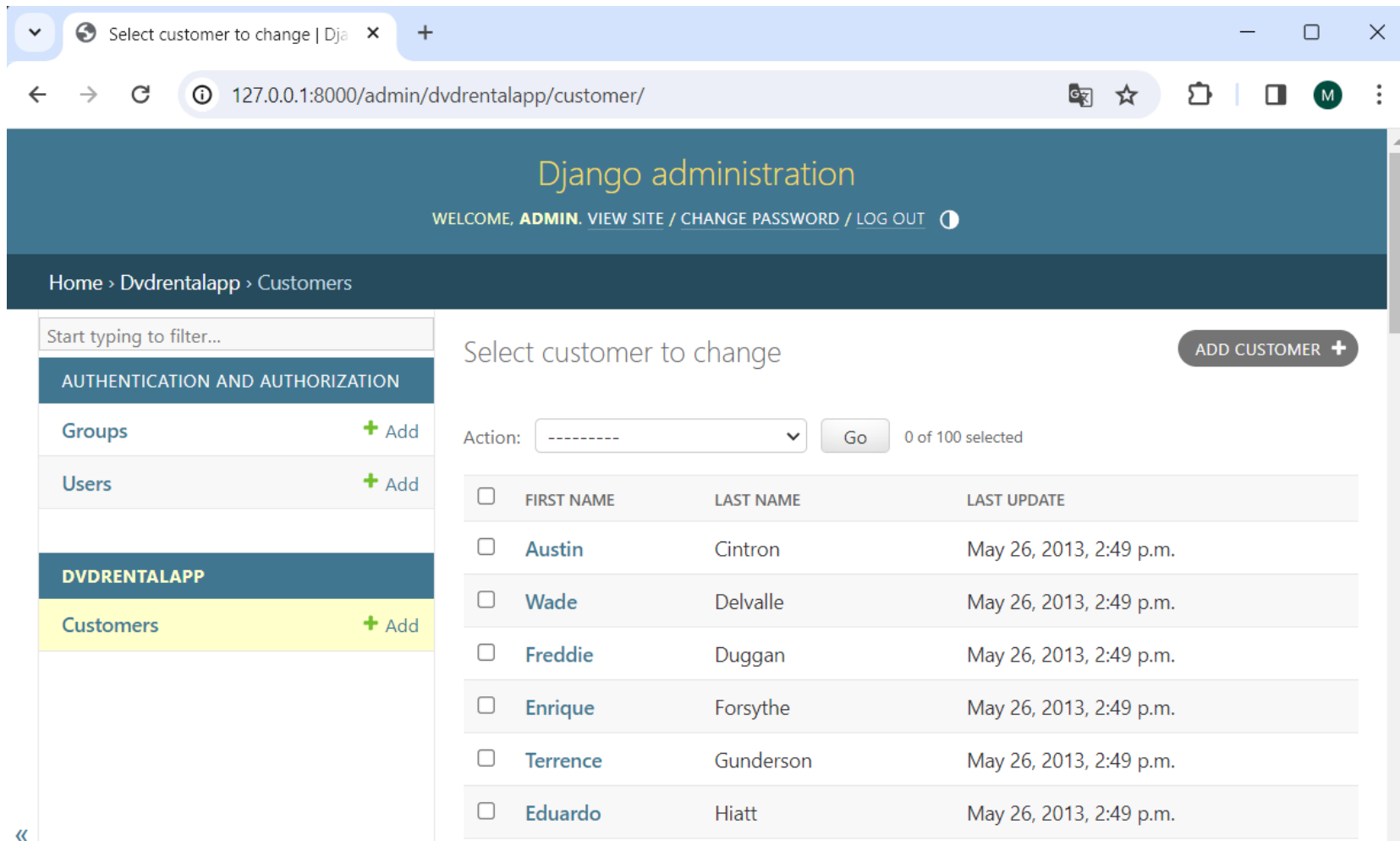
Conteúdo do arquivo admin.py:

```
admin.py ×  models.py  all_customers.html  urls.py dvdrentalapp

dvdrentalapp > admin.py > ...
1  from django.contrib import admin
2  from .models import Customer
3
4  # Register your models here.
5  class CustomerAdmin(admin.ModelAdmin):
6      list_display = ("first_name", "last_name", "last_update",)
7
8  admin.site.register(Customer, CustomerAdmin)
```

PROGRAMAÇÃO BACK-END

O resultado das alterações são:



The screenshot shows the Django administration interface for the 'DVDRENTALAPP' application. The page title is 'Select customer to change'. The sidebar on the left contains a search bar and a list of models: 'AUTHENTICATION AND AUTHORIZATION' (Groups, Users) and 'DVDRENTALAPP' (Customers). The 'Customers' model is highlighted. The main content area displays a table of customers with columns for 'FIRST NAME', 'LAST NAME', and 'LAST UPDATE'. The table lists six customers: Austin Cintron, Wade Delvalle, Freddie Duggan, Enrique Forsythe, Terrence Gunderson, and Eduardo Hiatt. All customers were last updated on May 26, 2013, at 2:49 p.m. An 'ADD CUSTOMER' button is located in the top right corner of the main content area.

Django administration

WELCOME, **ADMIN**. [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)

Home > Dvdrentalapp > Customers

Start typing to filter...

AUTHENTICATION AND AUTHORIZATION

- Groups + Add
- Users + Add

DVDRENTALAPP

- Customers + Add

Select customer to change

ADD CUSTOMER +

Action: Go 0 of 100 selected

<input type="checkbox"/>	FIRST NAME	LAST NAME	LAST UPDATE
<input type="checkbox"/>	Austin	Cintron	May 26, 2013, 2:49 p.m.
<input type="checkbox"/>	Wade	Delvalle	May 26, 2013, 2:49 p.m.
<input type="checkbox"/>	Freddie	Duggan	May 26, 2013, 2:49 p.m.
<input type="checkbox"/>	Enrique	Forsythe	May 26, 2013, 2:49 p.m.
<input type="checkbox"/>	Terrence	Gunderson	May 26, 2013, 2:49 p.m.
<input type="checkbox"/>	Eduardo	Hiatt	May 26, 2013, 2:49 p.m.

PROGRAMAÇÃO BACK-END

The screenshot shows the Django administration interface for the 'DVDRENTALAPP'. The left sidebar contains a tree view with 'AUTHENTICATION AND AUTHORIZATION' (Groups, Users) and 'DVDRENTALAPP' (Customers). The main content area is titled 'Select customer to change' and features a table of customers. A blue arrow points to the 'ADD CUSTOMER +' button in the top right corner. Another blue arrow points to the 'FIRST NAME' column header in the table.

<input type="checkbox"/>	FIRST NAME	LAST NAME	LAST UPDATE
<input type="checkbox"/>	Austin	Cintron	May 26, 2013, 2:49 p.m.
<input type="checkbox"/>	Wade	Delvalle	May 26, 2013, 2:49 p.m.
<input type="checkbox"/>	Freddie	Duggan	May 26, 2013, 2:49 p.m.
<input type="checkbox"/>	Enrique	Forsythe	May 26, 2013, 2:49 p.m.
<input type="checkbox"/>	Terrence	Gunderson	May 26, 2013, 2:49 p.m.
<input type="checkbox"/>	Eduardo	Hiatt	May 26, 2013, 2:49 p.m.

Pode-se adicionar um customer clicando sobre o botão.

Pode-se alterar os dados de um customer clicando sobre o first_name.

PROGRAMAÇÃO BACK-END

Start typing to filter...

AUTHENTICATION AND AUTHORIZATION

Groups + Add

Users + Add

DVDRENTALAPP

Customers + Add

Change customer

Austin Cintron HISTORY

Store id: 2

First name: Austin

Last name: Cintron

Email: austin.cintron@sakilacustomer.org

Address: Address object (605) ▾

☒ Activebool

Create date: 2006-02-14 Today | 📅
Note: You are 3 hours behind server time.

Last update: Date: 2013-05-26 Today | 📅
Time: 14:49:45 Now | ⌚
Note: You are 3 hours behind server time.

Active: 1

SAVE Save and add another Save and continue editing Delete

Tela de Alteração

Pode-se deletar um customer clicando sobre o botão.

Pode-se salvar os dados alterados de um customer clicando sobre os botões

PROGRAMAÇÃO BACK-END

Desafio

PROGRAMAÇÃO BACK-END

Altere as configurações para administrar outra tabela do banco dvdrental.

```
Prompt de Comando - pythor × + ▾  
  
(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.
```