

Nome: Matthaussouza Siqueira de Souza

TSI 5V

## Documento de Implantação

### 1. Descrição do Projeto:

O atual documento mostrará como foi feita a implantação de um projeto que consiste na construção de uma rede social utilizando do *framework Django*.

### 2. Enviando os arquivos do projeto para o repositório no *Git*:

Inicialmente os arquivos do projeto foram enviados para o repositório do projeto no *GitHub*. Para isso utilizou-se do *GitHub Desktop* e apenas foram atualizados os arquivos que já estavam no repositório. Após isso o repositório ficou assim:

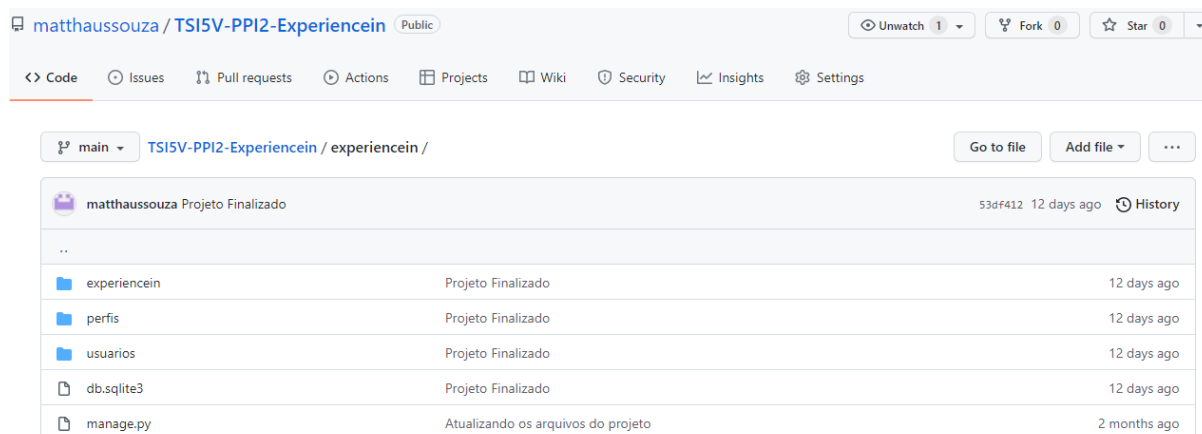


Figura 1: Repositório no *GitHub*;

### 3. Enviando os arquivos do repositório para o servidor:

Logo após atualizar o repositório, foi a vez de colocar esses dados em um servidor, no caso o *pythonanywhere*. Tendo já criado a conta no *pythonanywhere*, partiu-se então para a clonagem do repositório dentro do servidor. Para isso foi acessada a opção \$ Bash dentro da seção *new console*:

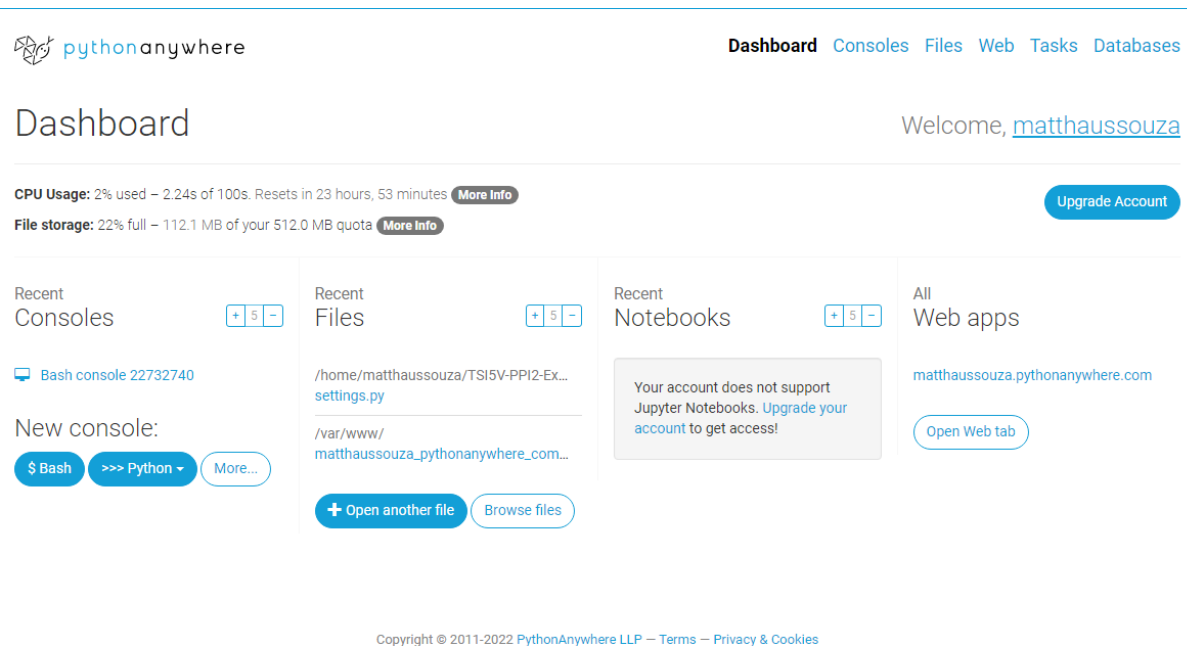


Figura 2: Aba *Dashboard* do *pythonanywhere*;

Já dentro do *console* digitou-se o seguinte comando para poder clonar o repositório:

```
git clone https://github.com/matthaussoza/TSI5V-PPI2-Experiencein.git
```

Após executar esse comando os arquivos já estão no servidor:

```
20:03 ~ $ ls
README.txt  TSI5V-PPI2-Experiencein
20:26 ~ $ cd TSI5V-PPI2-Experiencein
20:26 ~/TSI5V-PPI2-Experiencein (main)$ ls
README.md  experiencein
20:27 ~/TSI5V-PPI2-Experiencein (main)$ cd experiencein
20:27 ~/TSI5V-PPI2-Experiencein/experiencein (main)$ ls
db.sqlite3  experiencein  manage.py  perfis  static  usuarios
20:27 ~/TSI5V-PPI2-Experiencein/experiencein (main)$
```

Figura 3: Verificando os arquivos no console;

#### 4. Criando um ambiente virtual *Python*:

Com os arquivos já no servidor, criou-se um ambiente virtual. Para isso utilizou-se o seguinte comando dentro da pasta do projeto:

```
mkvirtualenv --python=/usr/bin/python3.7 experiencein-virtualenv
```

Após esse comando o ambiente virtual será criado e a linha de comando já começará dentro do ambiente virtual criado, com isso foi feita a instalação do *django* 2.2 com o seguinte comando:

```
pip install django==2.2
```

Ao dar o comando *python -m django version* verificamos que o *django* foi instalado com sucesso.

```
(experiencein-virtualenv) 20:52 ~ $ python -m django version
2.2
(experiencein-virtualenv) 20:58 ~ $ █
```

Figura 4: Verificando se o *django* foi instalado;

## 5. Criando e configurando a aplicação *web* com *WSGI*:

O próximo passo foi criar uma aplicação *web* como mostrado a seguir:

Passo 1:

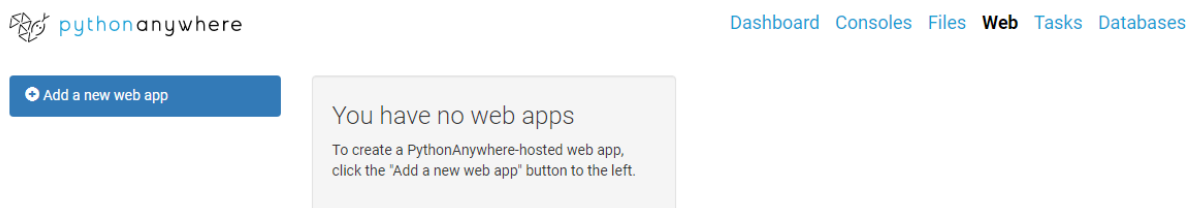


Figura 5: Aba *web*, opção *Add a new web app* do *pythonanywhere*;

Passo 2:

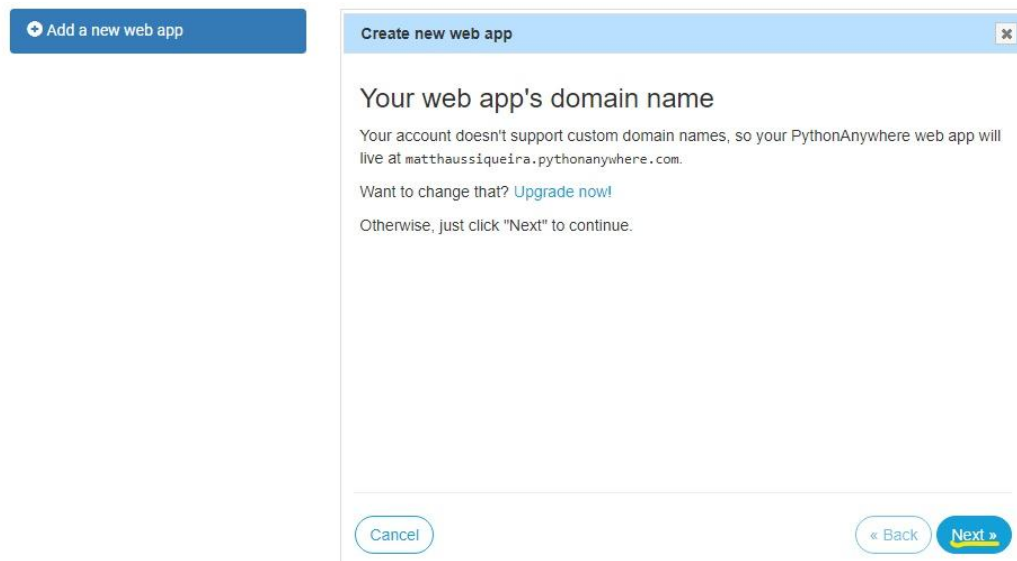


Figura 6: Aba *web*, opção *Add a new web app* - botão *Next*;

Passo 3:

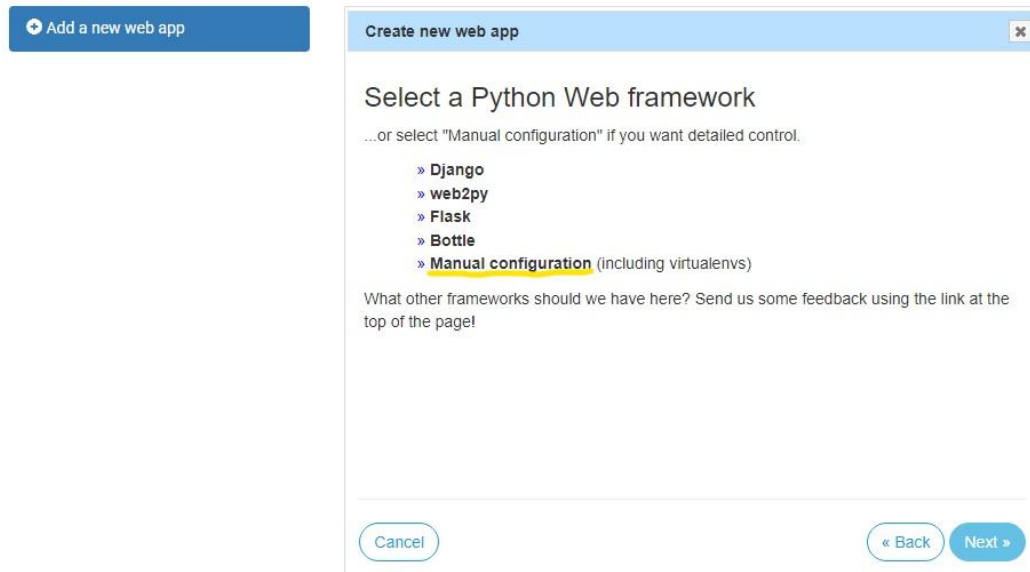


Figura 7: Aba *web*, opção *Add a new web app* - opção *Manual configuration*;

Passo 4:

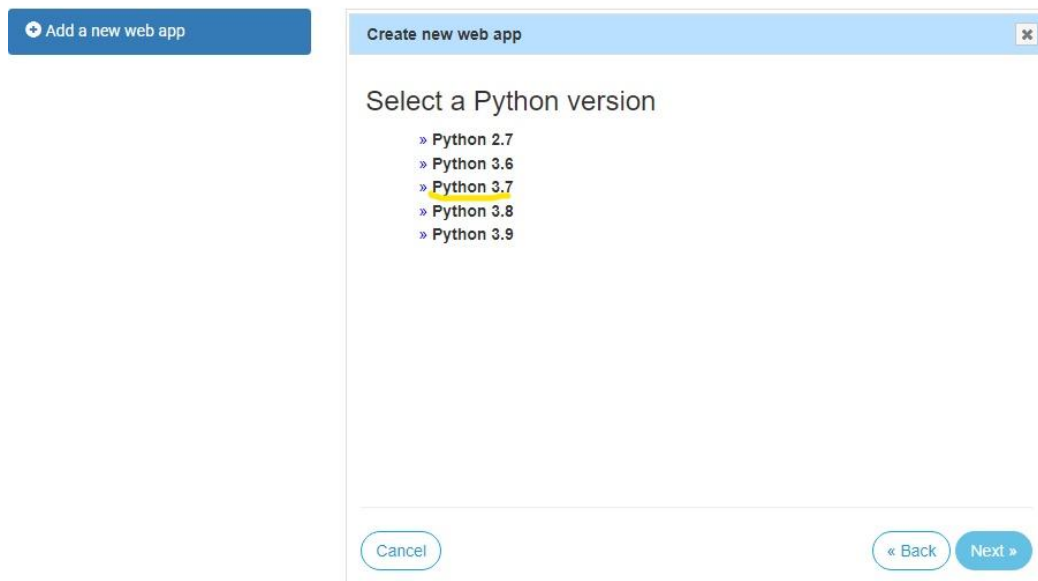


Figura 8: Aba *web*, opção *Add a new web app* - opção *Python 3.7*;

Passo 5:

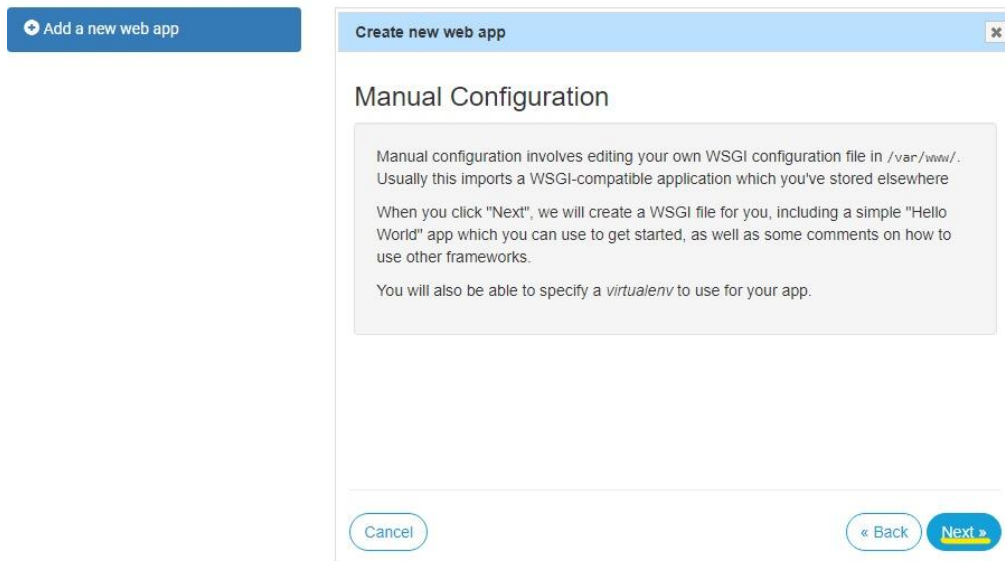


Figura 9: Aba web, opção Add a new web app - botão Next;

Após a realização desse processo a aplicação web foi criada com sucesso.

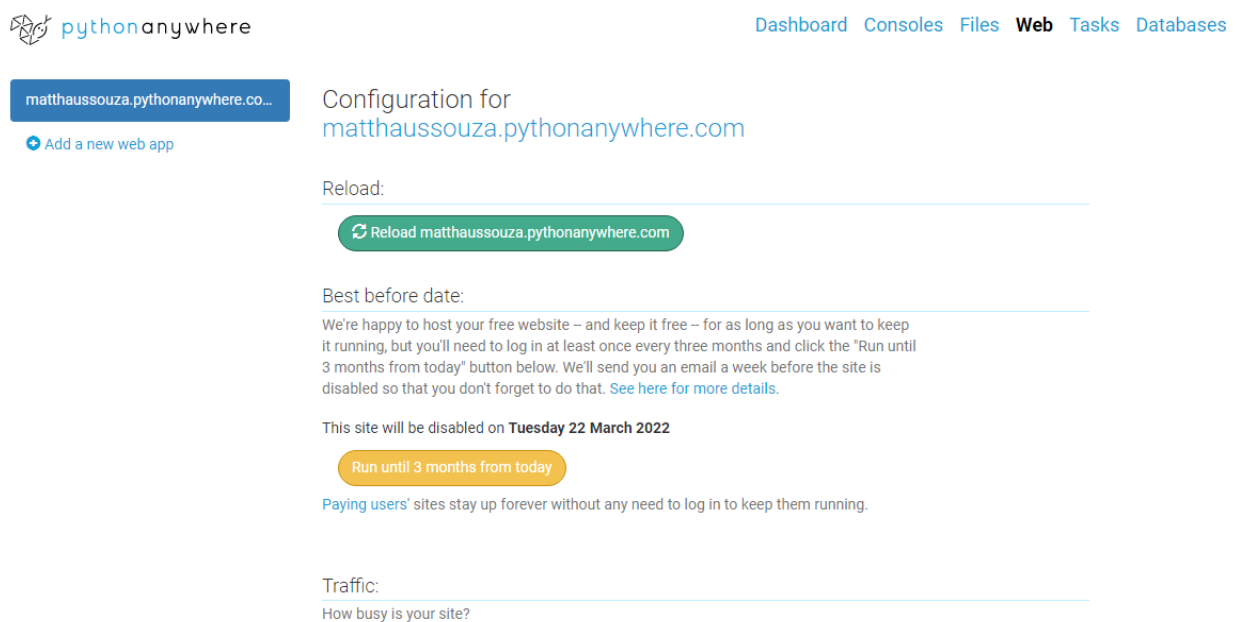


Figura 10: Aba web, aplicação já criada;

Com a aplicação web já criada, foi feita a vinculação do ambiente virtual com a aplicação web.

## Virtualenv:


Use a virtualenv to get different versions of flask, django etc from our default system ones. [More info here](#). You need to **Reload your web app** to activate it; NB - will do nothing if the virtualenv does not exist.

</home/matthausso Souza/.virtualenvs/experiencein-virtualenv>

 [Start a console in this virtualenv](#)

Figura 11: Aba web em *Virtualenv*;

Após vincular, passou-se para o próximo passo, configurar o *WSGI*. Onde se editou o arquivo de configuração *WSGI*, tendo ele ficado desta forma:




</var/www/matthausso Souza/pythonanywhere.com/wsgi.py>

```
1 # This file contains the WSGI configuration required to serve up your
2 # web application at http://matthausso Souza.pythonanywhere.com/
3 # It works by setting the variable 'application' to a WSGI handler of some
4 # description.
5 #
6
7 # ++++++ GENERAL DEBUGGING TIPS ++++++
8 # getting imports and sys.path right can be fiddly!
9 # We've tried to collect some general tips here:
10 # https://help.pythonanywhere.com/pages/DebuggingImportError
11
12 # Below are templates for Django and Flask. You should update the file
13 # appropriately for the web framework you're using, and then
14 # click the 'Reload /yourdomain.com/' button on the 'Web' tab to make your site
15 # live.
16
17 # ++++++ DJANGO ++++++
18 # To use your own django app use code like this:
19 import os
20 import sys
21
22 ## assuming your django settings file is at '/home/matthausso Souza/mysite/mysite/settings.py'
23 ## and your manage.py is at '/home/matthausso Souza/mysite/manage.py'
24 path = '/home/matthausso Souza/TSI5V-PPI2-Experiencein/experiencein'
25 if path not in sys.path:
26     sys.path.append(path)
27
28 os.environ['DJANGO_SETTINGS_MODULE'] = 'experiencein.settings'
29
30 ## then:
31 from django.core.wsgi import get_wsgi_application
32 application = get_wsgi_application()
33
```

Figura 12: Arquivo de configuração *WSGI* editado;

Após esse processo, recarregar o servidor e tentar abrir a aplicação ela dará erro. Esse erro foi resolvido adicionando o domínio da aplicação web no item *ALLOWED\_HOSTS* do arquivo *settings.py* dentro da aba *Files* no *pythonanywhere* da seguinte forma:

 /home/matthaus Souza/TSI5V-PPI2-Experiencein/experiencein/experiencein/settings.py

```

15 # Build paths inside the project like this: os.path.join(BASE_DIR, ...)
16 BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
17
18
19 # Quick-start development settings - unsuitable for production
20 # See https://docs.djangoproject.com/en/2.2/howto/deployment/checklist/
21
22 # SECURITY WARNING: keep the secret key used in production secret!
23 SECRET_KEY = 'b_b7i1vn3wk_&whdh1ey!gyo7zezkwz#z4=0zywr#f*vcan_d7'
24
25 # SECURITY WARNING: don't run with debug turned on in production!
26 DEBUG = False
27
28 ALLOWED_HOSTS = ['matthaus Souza.pythonanywhere.com']
29
30


```

Figura 13: Arquivo de *settings.py* editado;

## 6. Configurando os arquivos estáticos:

Após realizar o tópico anterior e abrir a aplicação vemos que os arquivos estáticos não foram carregados. Isso foi resolvido da seguinte forma:

Primeiramente foi criada uma nova constante chamada de *STATIC\_ROOT* dentro do arquivo *settings.py*, como mostrado na linha 123:

 /home/matthaus Souza/TSI5V-PPI2-Experiencein/experiencein/experiencein/settings.py
Keyboard shortcuts: Normal ▼

```

94 # django.contrib.auth.password_validation.MinimumLengthValidator',
95 },
96 {
97     'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
98 },
99 {
100     'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
101 },
102 ]
103
104
105 # Internationalization
106 # https://docs.djangoproject.com/en/2.2/topics/i18n/
107
108 LANGUAGE_CODE = 'en-us'
109
110 TIME_ZONE = 'UTC'
111
112 USE_I18N = True
113
114 USE_L10N = True
115
116 USE_TZ = True
117
118
119 # Static files (CSS, JavaScript, Images)
120 # https://docs.djangoproject.com/en/2.2/howto/static-files/
121
122 STATIC_URL = '/static/'
123 STATIC_ROOT = '/home/matthaus Souza/TSI5V-PPI2-Experiencein/experiencein/static'
124

```

Figura 14: Adicionando a constante *STATIC\_ROOT* em *settings.py*;

Após adicionar e salvar o arquivo, dentro do *console* foi dado o seguinte comando para coletar e organizar os arquivos estáticos dentro da pasta static que ele criará:

```
python manage.py collectstatic
```

```
(experiencein-virtualenv) 23:09 ~/TSI5V-PPI2-Experiencein/experiencein (main)$ ls
db.sqlite3 experiencein manage.py perfis static usuarios
(experiencein-virtualenv) 23:10 ~/TSI5V-PPI2-Experiencein/experiencein (main)$ cd static
(experiencein-virtualenv) 23:15 ~/TSI5V-PPI2-Experiencein/experiencein/static (main)$ ls
admin fonts img scripts styles
(experiencein-virtualenv) 23:15 ~/TSI5V-PPI2-Experiencein/experiencein/static (main)$
```

Figura 15: Arquivo static criado;

Por configurou se o menu *Static files* na aba *Web* do *pythonanywhere*, tendo ele ficado da seguinte forma:

#### Static files:

Files that aren't dynamically generated by your code, like CSS, JavaScript or uploaded files, can be served much faster straight off the disk if you specify them here. You need to **Reload your web app** to activate any changes you make to the mappings below.


URL	Directory	Delete
<a href="#">/static/</a>	<a href="#">/home/matthaus Souza/TSI5V-PPI2-Experiencein/experiencein/static</a>	
<a href="#">Enter URL</a>	<a href="#">Enter path</a>	

Figura 16: Menu *Static files* editado;

A última configuração feita foi editar a constante *DEBUG* no arquivo *settings.py* de *True* para *False*. E pronto a aplicação *web* está funcionando.

matthaus Souza.pythonanywhere.com/login/?next=/

## Login

Login

[registre-se](#)

Figura 17: Aplicação *web*;