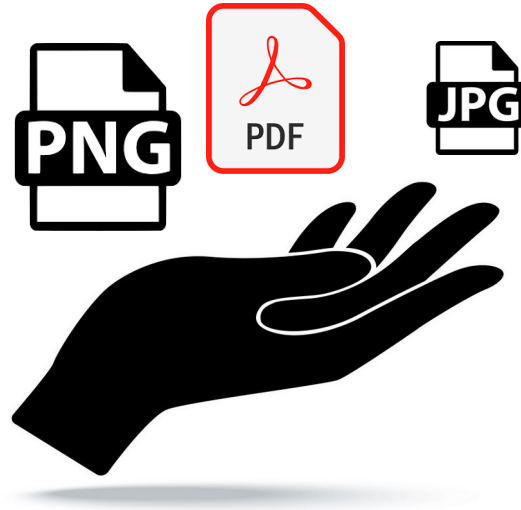


ImageField e FileField



Prof. Dr. João Paulo Lemos Escola
Copyright© 2025

Conteúdo

- Como vincular arquivos a um model;
- Classes ImageField e FileField;
- Configurar a pasta 'media' do servidor;
- Exibir arquivos de imagem;
- Criar link para PDF anexados.



1) ImageField

- Campo de imagem;
- Permite inserir um campo do tipo imagem em um model;
- Podemos incluir parâmetros:
 - null (permite ser nulo);
 - upload_to (permite especificar o diretório destino).

models.py

- Inclua o campo 'foto':

```
5 class Member(models.Model): 9 usages
6     plan = models.ForeignKey(Plan, on_delete=models.CASCADE, default=1)
7     firstname = models.CharField(max_length=255)
8     lastname = models.CharField(max_length=255)
9     phone = models.IntegerField(null=True)
10    joined_date = models.DateField(null=True)
11    foto = models.ImageField(upload_to='images/', null=True)
12
13    def __str__(self):
14        return f"{self.firstname} {self.lastname}"
15
```



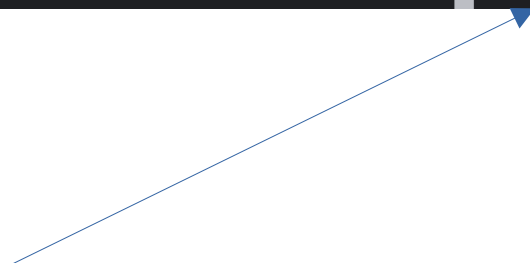
Pacote Pillow

- Caso seja apresentada a mensagem de erro;
- Instale o pacote confirme solicitado.

ERRORS:

members.Member.foto: (fields.E210) Cannot use ImageField because Pillow is not installed.

HINT: Get Pillow at <https://pypi.org/project/Pillow/> or run command "python -m pip install Pillow".



Migrations

- Como houve alteração no model, devemos fazer a migration:
 - `py manage.py makemigrations`
 - `py manage.py migrate`

ERRORS:

members.Member.foto: (fields.E210) Cannot use ImageField because Pillow is not installed.

HINT: Get Pillow at <https://pypi.org/project/Pillow/> or run command "python -m pip install Pillow".

(.venv) jpescola@jpescola-laptop:~/PycharmProjects/Aula09/my_tennis_club\$ pip install Pillow

Collecting Pillow

Using cached pillow-11.1.0-cp311-cp311-manylinux_2_28_x86_64.whl.metadata (9.1 kB)

Using cached pillow-11.1.0-cp311-cp311-manylinux_2_28_x86_64.whl (4.5 MB)

Installing collected packages: Pillow

Successfully installed Pillow-11.1.0

(.venv) jpescola@jpescola-laptop:~/PycharmProjects/Aula09/my_tennis_club\$ python manage.py makemigrations

Migrations for 'members':

members/migrations/0004_member_foto.py

+ Add field foto to member

(.venv) jpescola@jpescola-laptop:~/PycharmProjects/Aula09/my_tennis_club\$ python manage.py migrate

Operations to perform:

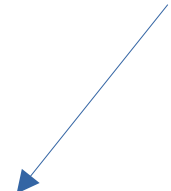
Apply all migrations: admin, auth, contenttypes, members, plans, sessions

Running migrations:


Applying members.0004_member_foto... OK

(.venv) jpescola@jpescola-laptop:~/PycharmProjects/Aula09/my_tennis_club\$

add.html

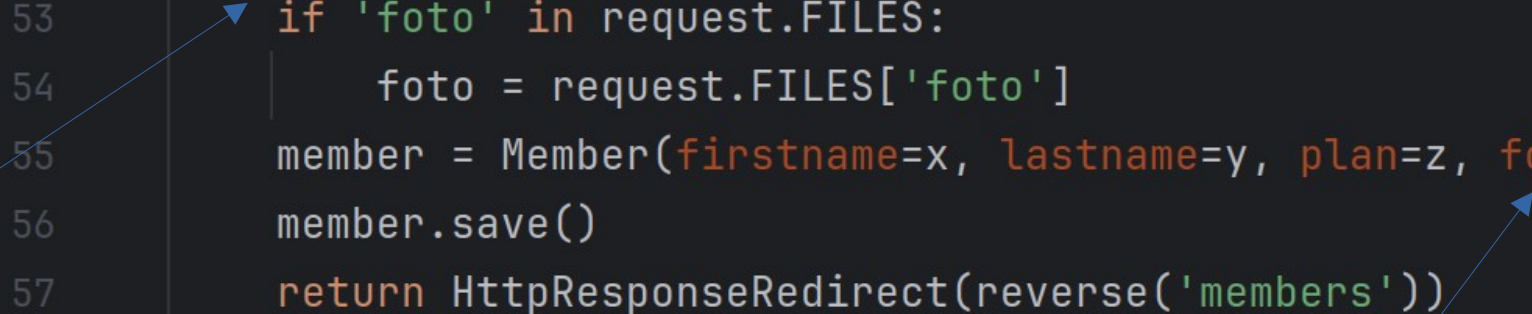


```
16 <form action="addrecord/" method="post" enctype="multipart/form-data">
17   {% csrf_token %}
18   <p>Plan:
19     <select name="plan">
20       {% for x in myplans %}
21         <option value="{{ x.id }}">{{ x.title }}</option>
22       {% endfor %}
23     </select>
24   </p>
25   First Name:<br>
26   <input name="first">
27   <br><br>
28   Last Name:<br>
29   <input name="last">
30   <br><br>
31   Foto:<br>
32   <input name="foto" type="file">
33   <br><br>
34   <input type="submit" value="Submit">
35 </form>
```




views.py

```
48     def addrecord(request): 1 usage
49         x = request.POST['first']
50         y = request.POST['last']
51         z = Plan.objects.filter(id=request.POST['plan']).first()
52         foto = None
53         if 'foto' in request.FILES:
54             foto = request.FILES['foto']
55         member = Member(firstname=x, lastname=y, plan=z, foto=foto)
56         member.save()
57         return HttpResponseRedirect(reverse('members'))
```



Veja o resultado

127.0.0.1:8000/members/add/

 Home members

Novo membro

Plan: Basico ▾

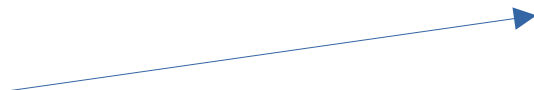
First Name:

Last Name:

Foto:
Procurar... Nenhum arquivo selecionado.

Submit

[Voltar](#)



Novo membro

Plan: Basico ▾

First Name:

Last Name:

Foto:
Procurar... woman.png

Submit

[Voltar](#)

Antes de testar...

- Vamos fazer também os ajustes para alteração, pois ambos permitirão incluir a foto do membro.

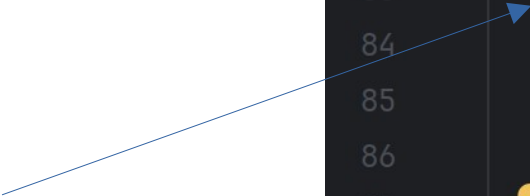
update.html

A propriedade **url** mostra
o endereço da imagem,
na pasta **media** do seu servidor.

```
16 <form action="updaterecord/{{ mymember.id }}" method="post" enctype="multipart/form-data">
17   {% csrf_token %}
18   <p>Plan:
19     <select name="plan">
20       {% for x in myplans %}
21         <option {% if x.id == mymember.plan.id %} selected {% endif %} value="{{ x.id }}" %>{{ x.name }}
22       {% endfor %}
23     </select>
24   </p>
25   First Name:<br>
26   <input name="first" value="{{ mymember.firstname }}">
27   <br><br>
28   Last Name:<br>
29   <input name="last" value="{{ mymember.lastname }}">
30   <br><br>
31   Foto:<br>
32   <input name="foto" type="file">
33   <br><br>
34   {% if mymember.foto %}
35   
36   <br><br>
37   {% endif %}
38
39   <input type="submit" value="Submit">
40 </form>
41
42 <p><a href="/members">Voltar</a></p>
```

views.py

```
75  def updaterecord(request, id): 1 usage
76      first = request.POST['first']
77      last = request.POST['last']
78      plan = request.POST['plan']
79      member = Member.objects.get(id=id)
80      member.firstname = first
81      member.lastname = last
82      member.plan = Plan.objects.get(id=plan)
83      if 'foto' in request.FILES:
84          foto = request.FILES['foto']
85          member.foto = foto
86      member.save()
87      💡 return HttpResponseRedirect(reverse('members'))
```




Ainda não vai funcionar

- Lembra-se que utilizamos a propriedade URL?
- Ela converte o endereço local da imagem, no endereço do domínio do servidor;
- Por isso, precisamos configurar o servidor para servir arquivos de 'media';

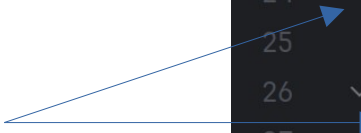
settings.py

```
131     STATIC_URL = 'static/'  
132  
133     MEDIA_URL = '/media/'  
134     MEDIA_ROOT = BASE_DIR / 'media'
```



urls.py do projeto

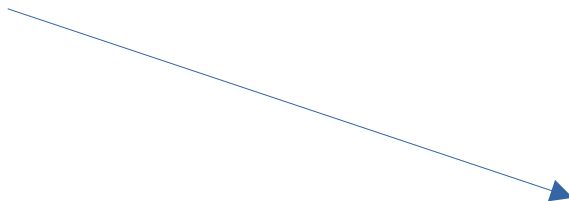
```
19  ▾ urlpatterns = [  
20      path('', include('members.urls')),  
21      path('admin/', admin.site.urls),  
22  ]  
23  
24  from django.conf import settings  
25  from django.conf.urls.static import static  
26  ▾ if settings.DEBUG:  
27      urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```



Esse ajuste não será
necessário em produção.

Agora sim!

- Se tudo estiver correto, você conseguirá ver a imagem ao clicar no botão alterar (lápis).




Atualizando dados do membro

Plan: Basico ▾

First Name:

Last Name:

Foto:
Procurar... Nenhum arquivo selecionado.



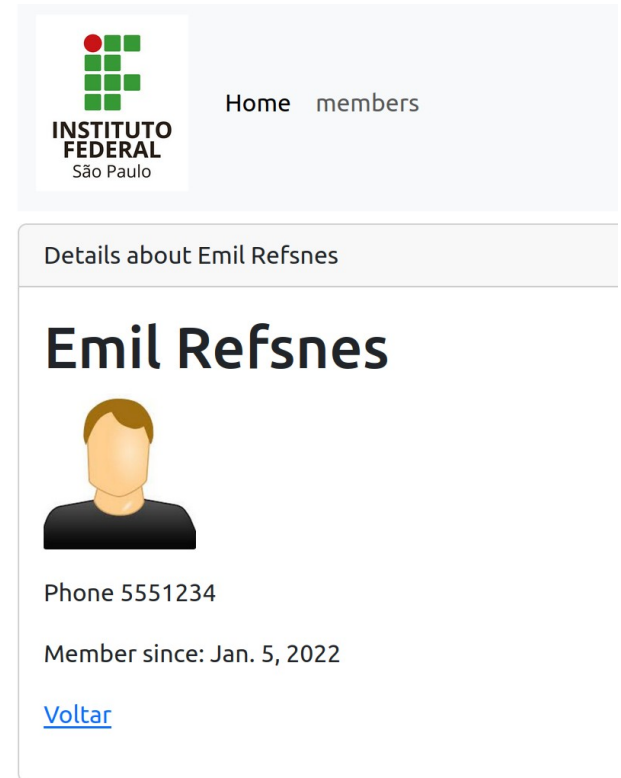
Submit
[Voltar](#)

Pasta 'media'

- Abra a pasta do projeto pelo Windows Explorer para ver as imagens que foram armazenadas no servidor;
- Ela deve ter sido criada automaticamente, dentro da pasta do projeto 'my_tennis_club'.

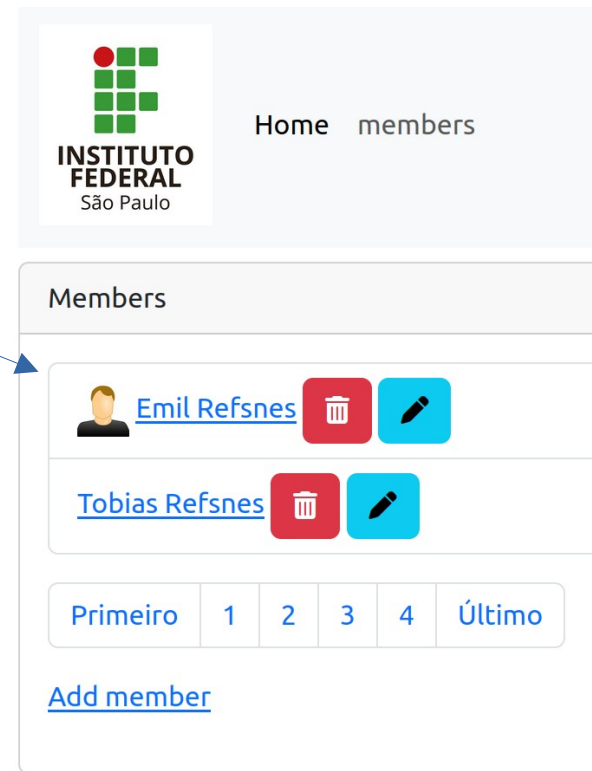
details.html

```
9 <div class="card mt-2">
10
11 <div class="card-header">
12     Details about {{ mymember.firstname }} {{ mymember.lastname }}
13 </div>
14 <div class="card-body">
15
16     <h1>{{ mymember.firstname }} {{ mymember.lastname }}</h1>
17
18     {% if mymember.foto %}
19     <p></p>
20     {% endif %}
21     <p>Phone {{ mymember.phone }}</p>
22     <p>Member since: {{ mymember.joined_date }}</p>
23
24     <p><a href="/members">Voltar</a></p>
25
26 </div>
27 </div>
```



2) Mostrando fotos na listagem

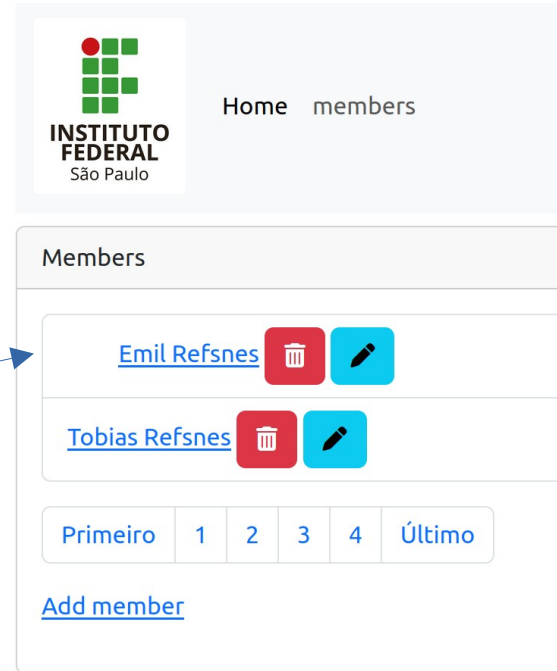
- Vamos incluir a miniatura da foto do membro ao lado do seu nome na listagem;
- Para isso, vamos ajustar a página inicial da rota 'members'.



Inclua o campo img

```
17 <ul class="list-group">
18     {% for x in page_obj %}
19     <li class="list-group-item">
20         {% if x.foto %}
21         
22         {% endif %}
23         <a href="details/{{ x.id }}">{{ x.firstname }} {{ x.lastname }}</a>
24         <a href="delete/{{ x.id }}" class="btn btn-danger"><i class='fa-solid fa-trash-can'></i></a>
25         <a href="update/{{ x.id }}" class="btn btn-info"><i class='fa-solid fa-pen'></i></a>
26     </li>
27     {% endfor %}
28 </ul>
```

BUG: Não mostra as imagens




Veja que o Emil tem foto cadastrada,
mas não aparece...

Corrigindo o bug da propriedade url

- Retire o método **values()** em views.py

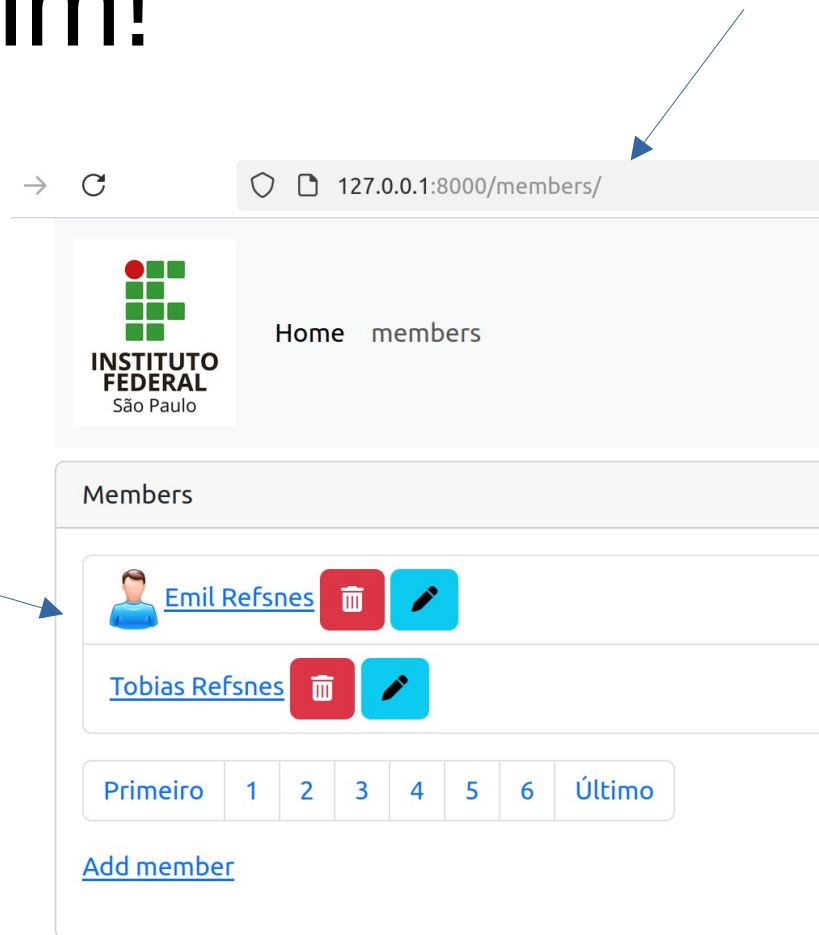
```
9      def members(request):
10          mymembers = Member.objects.all().values()
11          template = loader.get_template('all_members.html')
```

```
9      def members(request):
10          mymembers = Member.objects.all()
11          template = loader.get_template('all_members.html')
```



Agora sim!

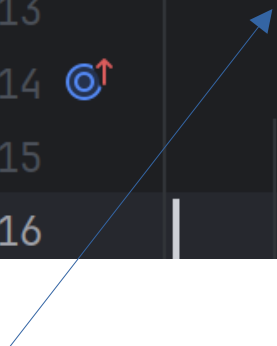
- A listagem exibirá as fotos quando existirem.



3) E no caso de arquivos PDF?

- Os procedimentos aqui descritos servem para qualquer tipo de arquivo;
- Basta alterar o tipo do campo para **FileField**;
- Vamos incluir um campo para upload do documento do membro (Ex: RG, CNH).

```
5 class Member(models.Model): 9 usages
6     plan = models.ForeignKey(Plan, on_delete=models.CASCADE, default=1)
7     firstname = models.CharField(max_length=255)
8     lastname = models.CharField(max_length=255)
9     phone = models.IntegerField(null=True)
10    joined_date = models.DateField(null=True)
11    foto = models.ImageField(upload_to='images/', null=True)
12    documento = models.FileField(null=True)
13
14    def __str__(self):
15        return f"{self.firstname} {self.lastname}"
16
```




Migrações



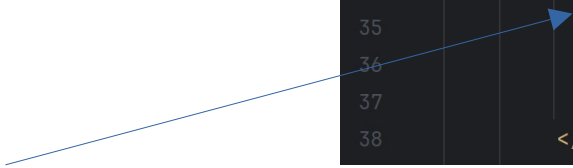
```
(.venv) jpescola@jpescola-laptop:~/PycharmProjects/Aula09/my_tennis_club$ python manage.py makemigrations
Migrations for 'members':
  members/migrations/0005_member_documento.py
    + Add field documento to member
(.venv) jpescola@jpescola-laptop:~/PycharmProjects/Aula09/my_tennis_club$
```

```
^C(.venv) jpescola@jpescola-laptop:~/PycharmProjects/Aula09/my_tennis_club$ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, members, plans, sessions
Running migrations:
  Applying members.0005_member_documento... OK
```



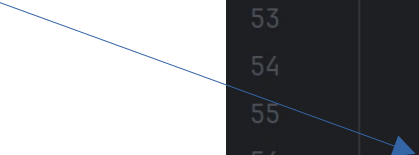
add.html

```
16      <form action="addrecord/" method="post" enctype="multipart/form-data">
17      <p>Plan:
23      </select>
24      </p>
25      First Name:<br>
26      <input name="first">
27      <br><br>
28      Last Name:<br>
29      <input name="last">
30      <br><br>
31      Foto:<br>
32      <input name="foto" type="file">
33      <br><br>
34      Documento:<br>
35      <input name="documento" type="file">
36      <br><br>
37      <input type="submit" value="Submit">
38      </form>
```



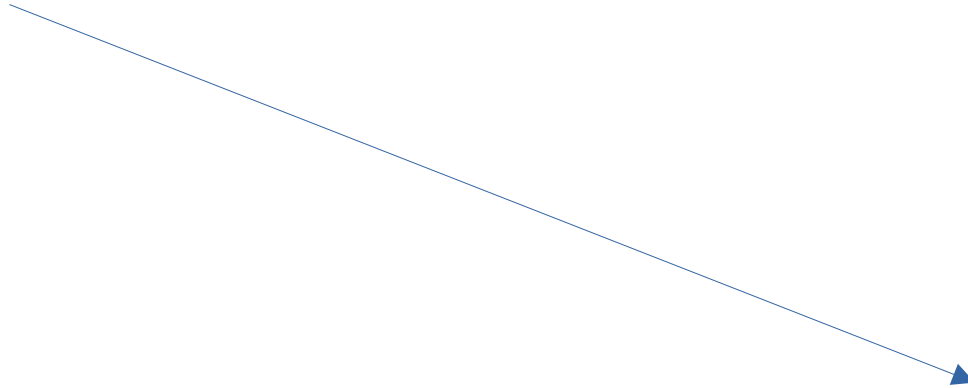
views.py


```
48 def addrecord(request): 1 usage
49     x = request.POST['first']
50     y = request.POST['last']
51     z = Plan.objects.filter(id=request.POST['plan']).first()
52     foto = None
53     documento = None
54     if 'foto' in request.FILES:
55         foto = request.FILES['foto']
56     if 'documento' in request.FILES:
57         documento = request.FILES['documento']
58     member = Member(firstname=x, lastname=y, plan=z, foto=foto, documento=documento)
59     member.save()
60     return HttpResponseRedirect(reverse('members'))
```



Novo membro

- Agora é possível anexar o documento em PDF



Home members

Novo membro

Plan: Básico ▾

First Name:

Last Name:

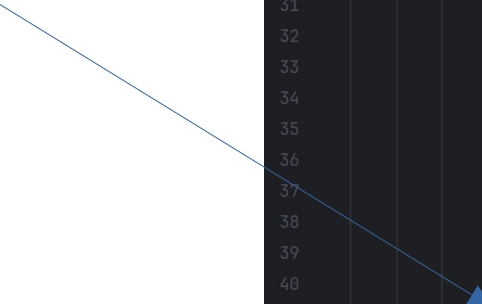
Foto:
 Nenhum arquivo selecionado.

Documento:
 Nenhum arquivo selecionado.

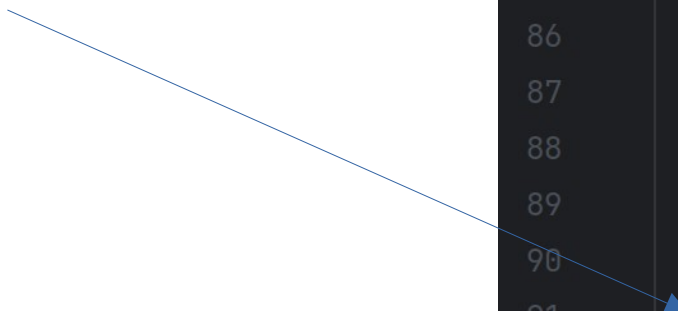
[Voltar](#)

update.html

```
16 <form action="updaterecord/{{ mymember.id }}" method="post" enctype="multipart/form-data">
17   {% csrf_token %}
18   <p>Plan:
19     <select name="plan">
20       {% for x in myplans %}
21         <option {% if x.id == mymember.plan.id %} selected {% endif %} value="{{ x.id }}">{{ x.title }}</option>
22       {% endfor %}
23     </select>
24   </p>
25   First Name:<br>
26   <input name="first" value="{{ mymember.firstname }}">
27   <br><br>
28   Last Name:<br>
29   <input name="last" value="{{ mymember.lastname }}">
30   <br><br>
31   Foto:<br>
32   <input name="foto" type="file">
33   <br><br>
34   {% if mymember.foto %}
35   
36   <br><br>
37   {% endif %}
38   Documento:<br>
39   <input name="documento" type="file">
40   <br><br>
41   {% if mymember.documento %}
42   <a href="{{ mymember.documento.url }}">ver documento</a>
43   <br><br>
44   {% endif %}
45   <input type="submit" value="Submit">
46 </form>
```




views.py



```
80     def updaterecord(request, id): 1 usage
81         first = request.POST['first']
82         last = request.POST['last']
83         plan = request.POST['plan']
84         member = Member.objects.get(id=id)
85         member.firstname = first
86         member.lastname = last
87         member.plan = Plan.objects.get(id=plan)
88         if 'foto' in request.FILES:
89             foto = request.FILES['foto']
90             member.foto = foto
91         if 'documento' in request.FILES:
92             documento = request.FILES['documento']
93             member.documento = documento
94         member.save()
95         return HttpResponseRedirect(reverse('members'))
```


Alterando

- O link do documento é exibido, caso o membro já possua PDF anexado.

Home members


Atualizando dados do membro

Plan: Básico ▾

First Name:

Last Name:

Foto:
 Nenhum arquivo selecionado.



Documento:
 Nenhum arquivo selecionado.

[ver documento](#)

[Voltar](#)

O que vimos?

- Como vincular arquivos a um model;
- Classes ImageField e FileField;
- Configurar a pasta 'media' do servidor;
- Exibir arquivos de imagem;
- Criar link para PDF anexados.

Referências

- <https://www.geeksforgeeks.org/python-uploading-images-in-django/>
- <https://docs.djangoproject.com/en/5.1/ref/models/fields/#django.db.models.FileField>