
Django ORM

Um guia prático para conexão e manipulação de banco de dados com Django e SQLite

git clone

<https://github.com/romulodf-cesar/django-orm>



1. Introdução

Django é um framework e possui dentro dele o recurso de ORM. .

→ **Python**

Dominar o python é importante para usufruir do potencial do Django.

→ **POO**

Invista em programação orientada a objetos, ela é um diferencial.

→ **Mapeamento**

Entenda o mapeamento de Classes para Tabelas, aplicando Design Patterns e tecnologia do Framework.

Code First

Classes
de
Entidades
(Entity)



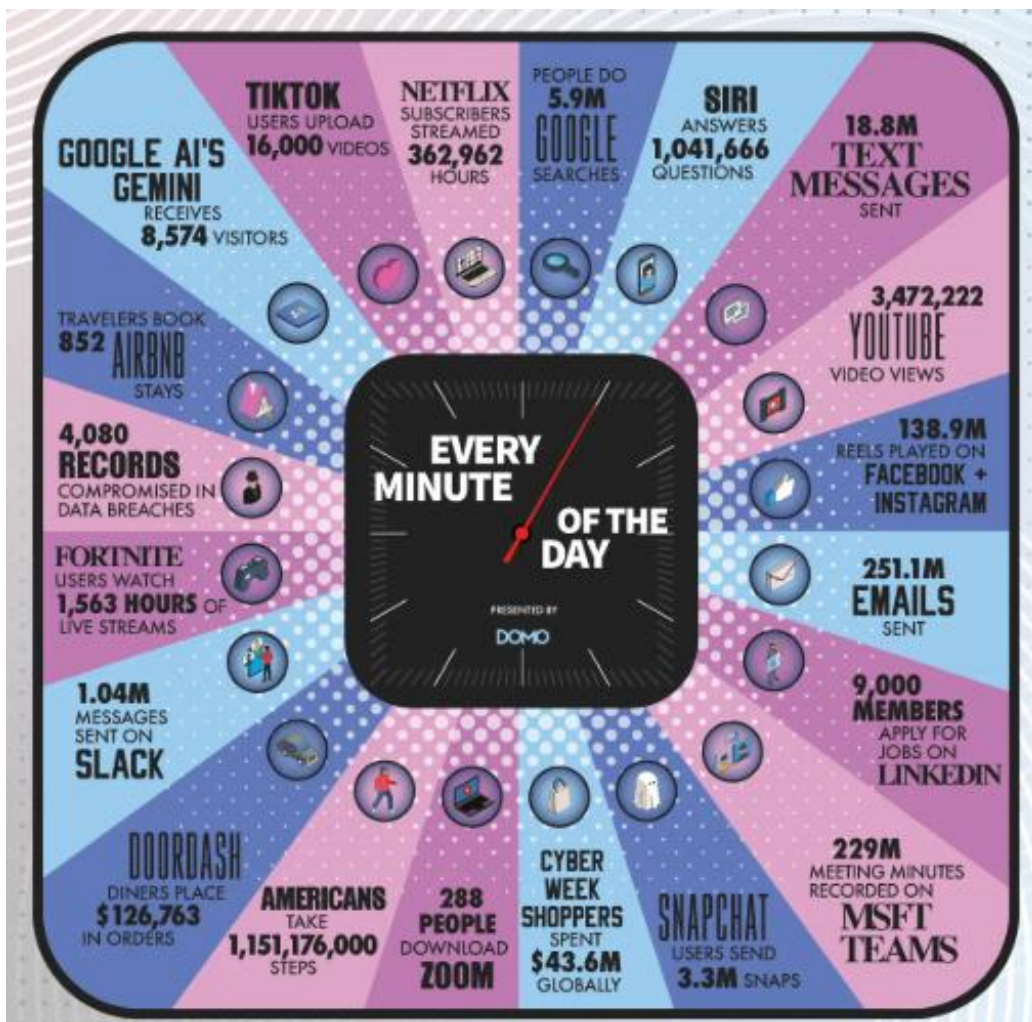
Tabela

O programador cria uma classe em Python



Django ORM

Tabela em um banco de dados Relacional





EXPLORADOR



EDITORES ABERTOS



requirements.txt

M

DJANGO-ORM



.venv

galeria

setup

static

templates

.gitignore

manage.py

requirements.txt

M



requirements.txt M X



requirements.txt

```
1 asgiref==3.8.1
...
2 Django==4.1
3 dotenv==0.9.9
...
4 python-dotenv==1.1.0
...
5 sqlparse==0.5.3
...
6 tzdata==2025.1
...
7
```

—

Faça o clone, ative a venv, instale o dotenv e execute a aplicação?



Dica

```
python manage.py  
runserver
```

```
pip install -r  
requirements.txt
```

```
pip freeze >  
requirements.txt
```



Home



Mais vistas



Novas



Surpreenda-me

A galeria mais
completa de fotos
do espaço!

Busque por tags:

Nebulosa

Estrela

Galáxia

Planeta

Faça! Uma mudança.

(deve aparecer apenas um card na nova versão)



Dica

Retire os cads e deixe apenas um. Observe que os cards estão em uma lista html , então???

Abra o views.py da galeria..



Dica

Além do banco de dados podemos também inserir informações em listas, tuplas, conjuntos e dicionários no Python..



2. Na views.py

Entre na função `index()`:

→ **Dicionários**

Crie dicionários

→ **Paramêtros**

Passe os parâmetros (valores) pelo render.

```
4  def index(request):
7      "legenda": "www.jaspion.com.br"},
8      2: {"nome": "Planeta Cripton",
9          "legenda": "www.superman.com.br"},
10     }
11     return render(request, 'galeria/index.html', {"cards": dados})
12
```

```
<section class="conteudo">
```

```
<section class="galeria">
```

```
<ul class="cards_lista">
```

```
{% for foto_id,info in cards.items %}
```

```
<li class="card">
```

```
<a href="{% url 'imagem' %}">
```

```

```

```
</a>
```

```
<span class="card_tag">Estrelas</span>
```

```
<div class="card_info">
```

```
<section class="conteudo">
```

```
  <section class="galeria">
```

```
    <span class="card__tag">Estrelas</span>
```

```
    ...
```

```
    <div class="card__info">
```

```
      <p class="card__titulo">{{info.nome}}</p>
```

```
      <div class="card__texto">
```

```
        <p class="card__descricao">{{info.legendas}}</p>
```

```
        ...
```

```
      <span>
```

```
        
```

```
          ...
```

```
          ...
```

```
      </span>
```

`alt="ícone de coraç`

`...`

`...`

``

`</div>`

`</div>`

``

`{% endfor %}`



``

`</div>`

`</section>`



Dica

Use o SQLite para o tempo de projeto.

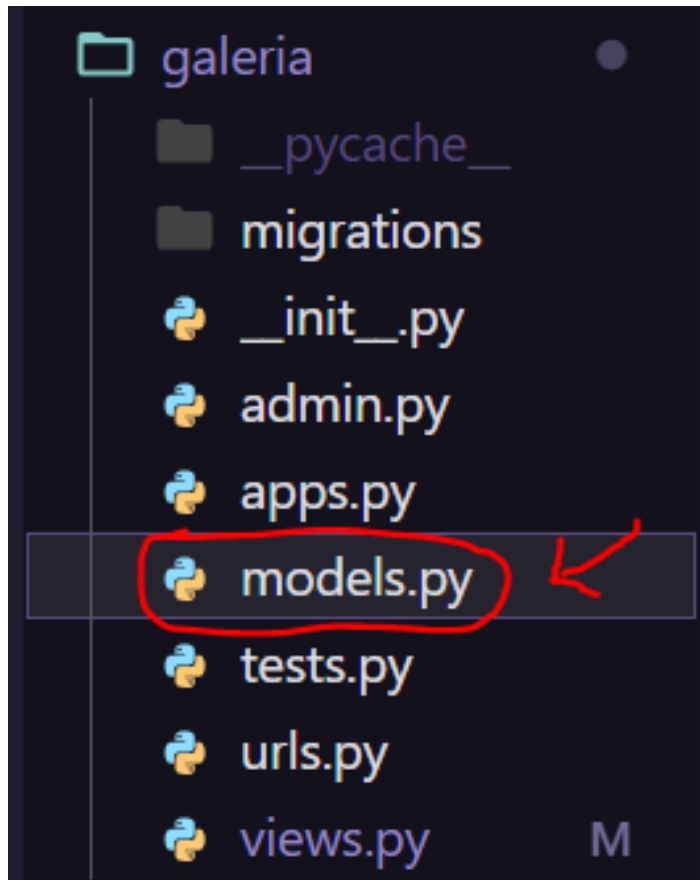
Que tal usarmos ...

Um banco de dados.

models

Classe

Orientação a objetos



```
3  # Create your models here.
4  class Fotografia(models.Model): # herança
5      nome = models.CharField(max_length=100, null=False, blank=False)
6      # string , máximo 100,  null um campo que não tem informação, blank é uma string vazia
7      legenda = models.CharField(max_length=150, null=False, blank=False)
8      descricao = models.TextField(null=False, blank=False)
9      foto = models.CharField(max_length=100, null=False, blank=False)
10     # boa prática
11     def __str__(self):
12         return f"Fotografia [nome={self.nome}]"
13
```

```
(.venv) C:\Users\ALUNO\OneDrive\Desktop\projetos\django-orm>python manage.py help
```

```
Type 'manage.py help <subcommand>' for help on a specific subcommand.
```

```
Available subcommands:
```

```
[auth]
```

```
    changepassword  
    createsuperuser
```

```
[contenttypes]
```

```
    remove_stale_contenttypes
```

```
[django]
```

Python manage.py help

check
compilemessages
createcachetable
dbshell
diffsettings
dumpdata
flush
inspectdb
loaddata
makemessages
makemigrations
migrate
optimizemigration
sendtestemail
shell
showmigrations
sqlflush
sqlmigrate
sqlsequencereset
squashmigrations

(.venv) C:\Users\ALUNO\OneDrive\Desktop\projetos\django-orm>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations: admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.

April 01, 2025 - 16:32:42

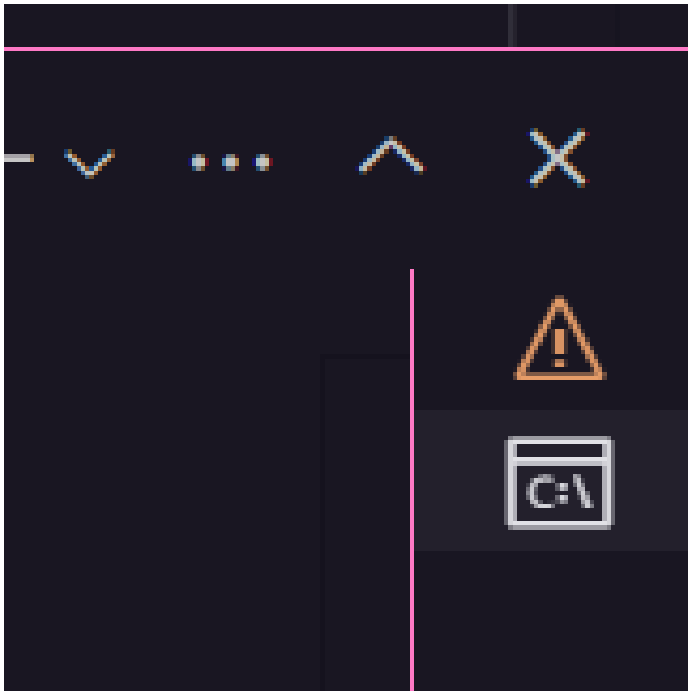
Django version 4.1, using settings 'setup.settings'


Starting development server at http://127.0.0.1:8000/

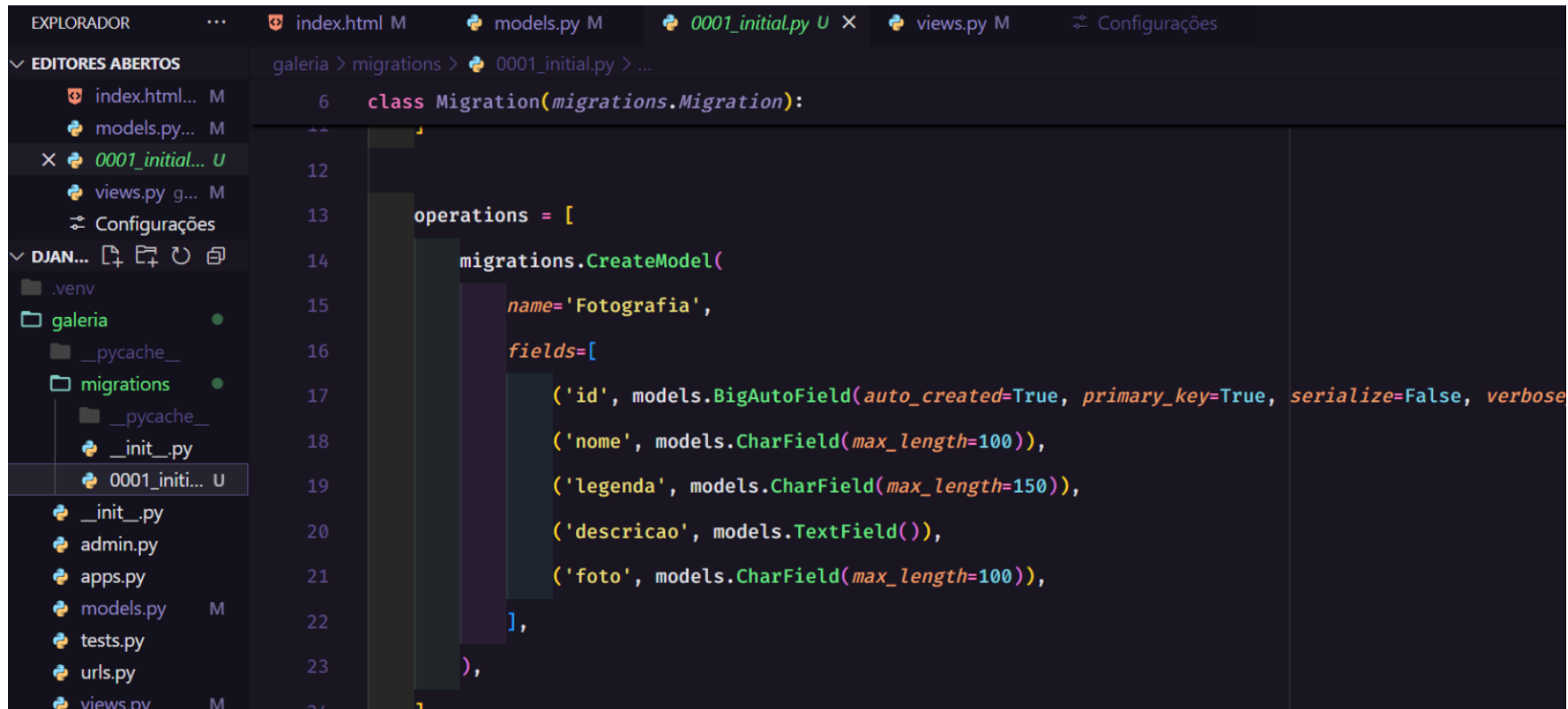
Quit the server with CTRL-BREAK.

□

—



```
(.venv) C:\Users\ALUNO\OneDrive\Desktop\projetos\django-orm>python manage.py makemigrations
Migrations for 'galeria':
  galeria\migrations\0001_initial.py 
    - Create model Fotografia
```

(.venv) C:\Users\ALUNO\OneDrive\Desktop\projetos\django-orm>python manage.py migrate

Operations to perform:

Apply all migrations: admin, auth, contenttypes, galeria, 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

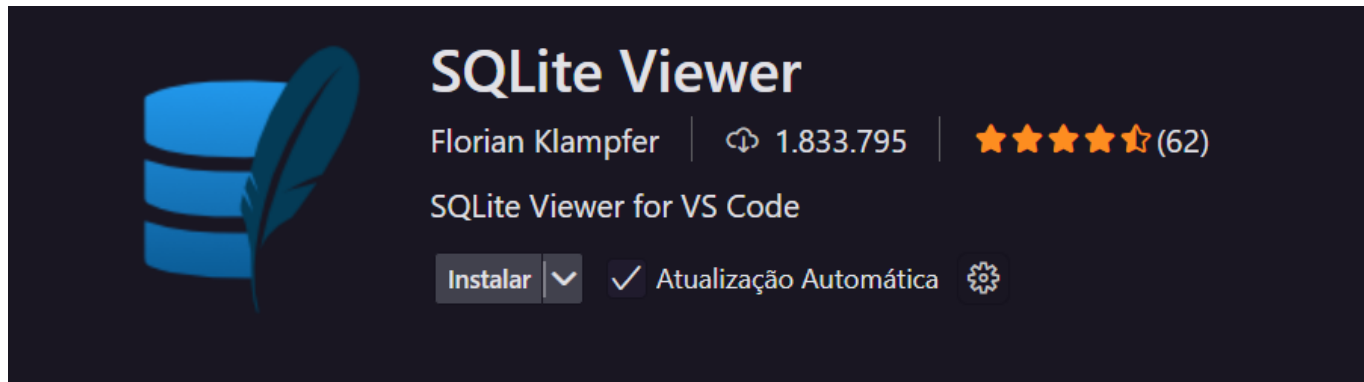
Applying auth.0007_alter_validators_add_error_messages... OK

Applying auth.0008_alter_user_username_max_length... OK

Eu tenho um banco de dados

SQLite - embutido no Django.

Extensão SQLite no Visual Studio Code



The image shows a dark-themed card for the 'SQLite Viewer' extension in Visual Studio Code. On the left is a blue icon of a database cylinder with a feather. To the right of the icon, the text 'SQLite Viewer' is displayed in a large, bold font. Below this, the author 'Florian Klampfer' is listed, followed by a download icon and the number '1.833.795'. To the right of the download count is a row of five orange stars and the text '(62)'. Below the author and download information, the text 'SQLite Viewer for VS Code' is shown. At the bottom of the card, there is a button labeled 'Instalar' with a dropdown arrow, a checkmark icon, the text 'Atualização Automática', and a gear icon for settings.

SQLite Viewer

Florian Klampfer | 1.833.795 | ★★★★★ (62)

SQLite Viewer for VS Code

Instalar ▼ ✓ Atualização Automática ⚙

ArquivoEditorSeleçãoVer...

←→

🔍 django-orm

EXPLORADOR

EDITORES ABERTOS

✕ db.sqlite3

DJANGO-ORM

.venv

galeria

setup

static

templates

.gitignore

db.sqlite3

manage.py

requirements.txt

db.sqlite3

db.sqlite3

Filtrar 12 t

Linhas: 0

TABELAS

auth_group

auth_group_permi...

auth_permission

auth_user

auth_user_groups

auth_user_user_pe...

django_admin_log

django_content_ty...

django_migrations

django_session

galeria_fotografia

sqlite_sequence

id

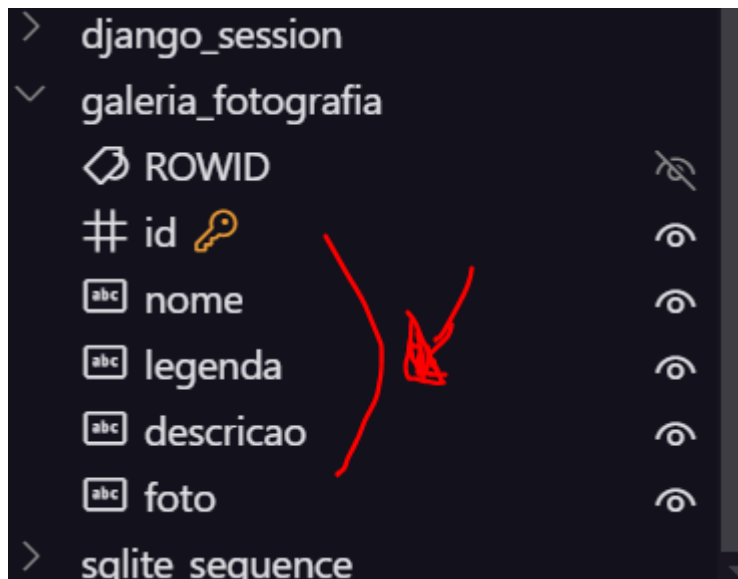
name

Filtrar

Filtrar

1

Visualize o seu banco



Dica

Se um exemplo não for suficiente para ajudar as pessoas a entender o escopo da sua ideia, escolha alguns exemplos.

python manage.py shell



Dica

Vai abrir uma shell
imperativo.

```
(.venv) C:\Users\ALUNO\OneDrive\Desktop\projetos\django-orm>python manage.py shell
Python 3.13.2 (tags/v3.13.2:4f8bb39, Feb  4 2025, 15:23:48) [MSC v.1942 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)
>>> from galeria.models import Fotografia
>>> foto = Fotografia(nome="Planeta Odium",legenda="www.jaspion.com",foto='jaspion.png')
>>> foto.save()
>>> Fotografia.objects.all()
<QuerySet [<Fotografia: Fotografia [nome=Planeta Odium]>]>
>>> █
```

db.sqlite3

db.sqlite3

Filtrar 12 tabelas...

Linhas: 1

Filtrar 1 linhas...

TABELAS

auth_group

auth_group_permissions

auth_permission

auth_user

auth_user_groups

auth_user_user_permissions

django_admin_log

django_content_type

django_migrations

django_session

galeria_fotografia

id	nome	legenda	descricao	foto
1	Planeta Odim	www.jaspion.com	"	jaspion.png
2				



3. Boa prática

Existe uma pasta chamada apps.py

→ **Puxar o arquivo de configuração do Django.**

```
apps.py ×
galeria > apps.py > ...
1  from django.apps import AppConfig
2
3
4  class GaleriaConfig(AppConfig):
5      default_auto_field = 'django.db.models.BigAutoField'
6      name = 'galeria'
7
```

—

Melhor prática para
chamar o arquivo de
configuração...

settings.py M X

setup > settings.py > ...

```
24  INSTALLED_APPS = [  
25      'django.contrib.admin',  
26      'django.contrib.auth',  
27      'django.contrib.contenttypes',  
28      'django.contrib.sessions',  
29      'django.contrib.messages',  
30      'django.contrib.staticfiles',  
31      'galeria.apps.GaleriaConfig',  
32  ]  
33  
34  MIDDLEWARE = [
```

—

Vamos executar...

Ainda usamos dicionários...



Dica

Precisamos agora mostrar os itens que vem do banco de dados.

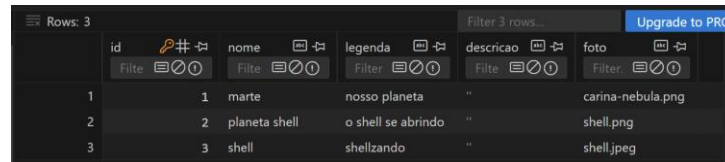


4. Apagar o dict

Vá para Views.py e apague o dict

ORM

```
class Fotografia(models.Model): # herança
    nome = models.CharField(max_length=100,
        # string , máximo 100, null um campo q
```



Rows: 3

id	nome	legenda	descricao	foto
1	1	marté	nosso planeta	carina-nebula.png
2	2	planeta shell	o shell se abrindo	shell.png
3	3	shell	shellizando	shell.jpeg

Upgrade to PRO



M- mapeamento

O – Objeto

R- Relacional

Lista de ORMs

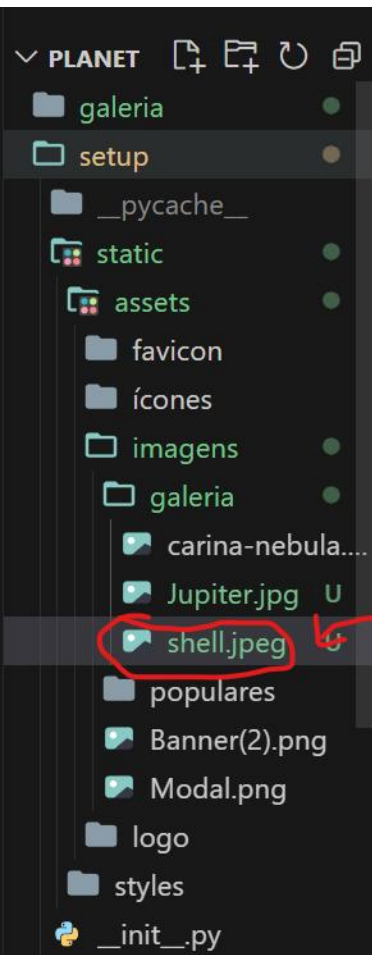
SQLAlchemy (Python e Flask)
Django ORM (Python e Django)
Hibernate (Java)
Entity Framework (C#)
Doctrine (PHP)
Sequelize (Java Script)

shell.png

Pasta de Imagens

No banco de dados
shell.png

		nome	legenda	descricao	foto
		Filter	Filter	Filter	Filter
1	1	marte	nosso planeta	"	carina-nebula.png
2	2	planeta shell	o shell se abrindo	"	shell.png
	3	shell	shellzando	"	shell.jpeg



setup > static > assets > imagens > galeria > shell.jpeg



```

1  from django.shortcuts import render
2  from django.http import HttpResponse
3  from galeria.models import Fotografia
4  def index(request):
5      # o mesmo comando do Shell
6      # diferenciar shell de prompt de comando
7      fotografias = Fotografia.objects.all() # lista de todos os objetos que repr
8      dados.
9      return render(request, 'galeria/index.html', {"cards": fotografias})

```

```
<div class="pagina-inicial">
```

```
<main class="principal">
```

```
<section class="conteudo">
```

```
<h2 class="cards__titulo">Navegue pela galeria</h2>
```

```
<ul class="cards__lista">
```

```
{% if cards %}
```

```
{% for fotografia in cards %}
```

```
<li class="card">
```

```
<a href="{% url 'imagem' %}">
```

```

```

```
src="{% static '/assets/imagens/galeria/'%}{{fotografia.foto}}"
```



```
5      <main class="principal">
```

```
4          <section class="conteudo">
```

```
0              </li>
```

```
1              {% endfor%}
```

```
2              {% else %}
```

```
3              {% endif %}
```

```
4          </ul>
```

```
...  
<div class="card__info">
```

```
  <p class="card__titulo">{{fotografia.nome}}</p>
```

```
  <div class="card__texto">
```

```
    <p class="card__descricao">{{fotografia.legenda}}</p>
```

```
    <span>
```

```
      
```


Planeta Odim

www.jaspion.com



fantástico jaspion

destrua o santangoz



Desafios

Criar novas imagens

Criar novos cards

Ajustar o arquivo
imagem.html

Viva o Django.

The background of the slide shows the silhouettes of four people sitting at a table in a dimly lit room, looking out a large window. Outside the window, a city skyline is visible, featuring a prominent domed building, likely St. Paul's Cathedral in London, under a hazy sky.

Dica

Repita as operações.

Agora você vai ser desafiado.

CRUD

- Inserir
 - Deletar
 - Pesquisar
 - Atualizar
-

```
>>> help
Type help() for interactive help, or help(object) for help about object.
>>> from galeria.models import Fotografia
>>> foto_objeto = Fotografia.objects.get(pk=4)
>>> foto_objeto.nome = "martelo.jpg"
>>> foto_objeto.save()
>>> foto_objeto.foto = "martelo.jpg"
>>> foto_objeto.save()
>>> foto_objeto.nome = "o martelo é de teste"
>>> foto_objeto.save()
>>> □
```



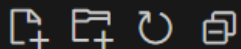
Projeto Integrador

- WIREFRAME (**Thamires**)
 - .png e um editável
- PROTOTIPO FIGMA (**Gustavo**)
 - Link
- HTML (**Raposo**)
 - .html
- BASE EM DJANGO (**Walter** – criar a base Django)
 - django
- (**Vivi – Documentacao Junto com o Robson**)
- Lista de Concorrentes (**Suzane**)
 - **Texto explicando sobre os concorrentes (pontos de melhoria)**
- Robson (Documentação Escrita)
 - O que esta falando:
- Diagramas UML – Denilson e Daniel
- Rafael (FRONT END Manager)

Urls das Imagens

manutenção

PLANET



galeria



__pycache__

migrations

__init__.py

admin.py

apps.py

models.py

tests.py

urls.py



views.py

galeria > urls.py > ...

```
1 from django.urls import path
2 from galeria.views import index
3 from galeria.views import imagem
4
5 urlpatterns = [
6     path('', index, name='index'),
7     path('imagem/<int:foto_id>', imagem, name='imagem'),
8 ]
```



galeria

__pycache__

migrations

__init__.py

admin.py

apps.py

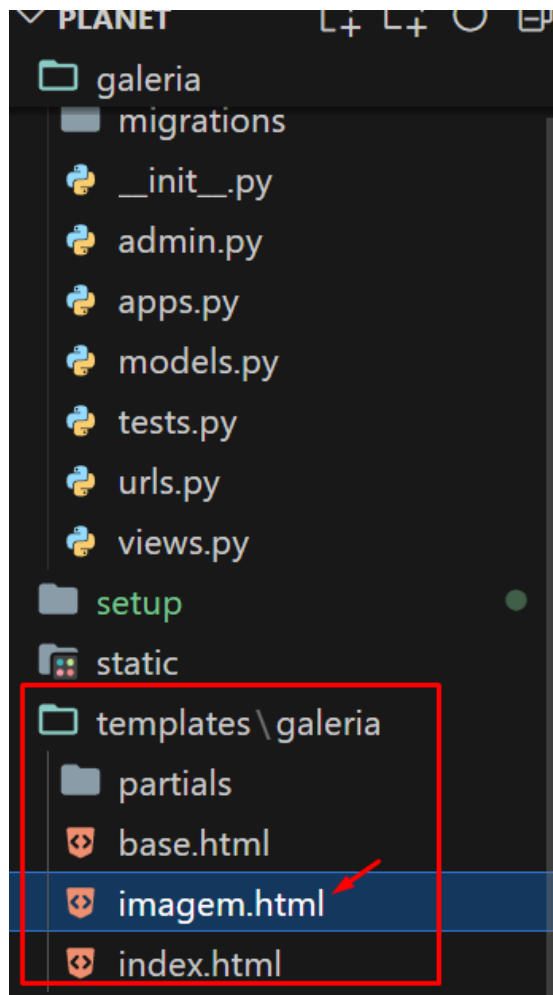
models.py

tests.py


urls.py

views.py

```
def imagen(request, foto_id):  
    fotografia = get_object_or_404(Fotografia, pk=foto_id)  
    return render(request, 'galeria/imagen.html', {"fotografia": fotografia})
```



```
class="conteudo">
on class="imagem">
div class="imagem__conteudo">
  
  <div class="imagem__info">
    <div class="imagem__texto">
      <p class="imagem__titulo">{{fotografia.nome}}</p>
      <p class="imagem__descricao">{{fotografia.legenda}}</p>
      <p class="imagem__texto">{{fotografia.descricao}}</p>
    </div>
  </div>
</div>
```



```
v class="imagem__conteudo">

<div class="imagem__info">
  <div class="imagem__texto">
    <p class="imagem__titulo">{{fotografia.nome}}</p>
    <p class="imagem__descricao">{{fotografia.legenda}}</p>
    <p class="imagem__texto">{{fotografia.descricao}}</p>
  </div>
</div>
```

github

GitHub - romulodf-cesar/planet


```
python manage.py shell
```

Hoje

- Manutencao do Bug da Imagem (20-50)
 - Entrega parcial de atividades PI (ate 21:40)
-

11/04/2024

- ADMIN (no Code)