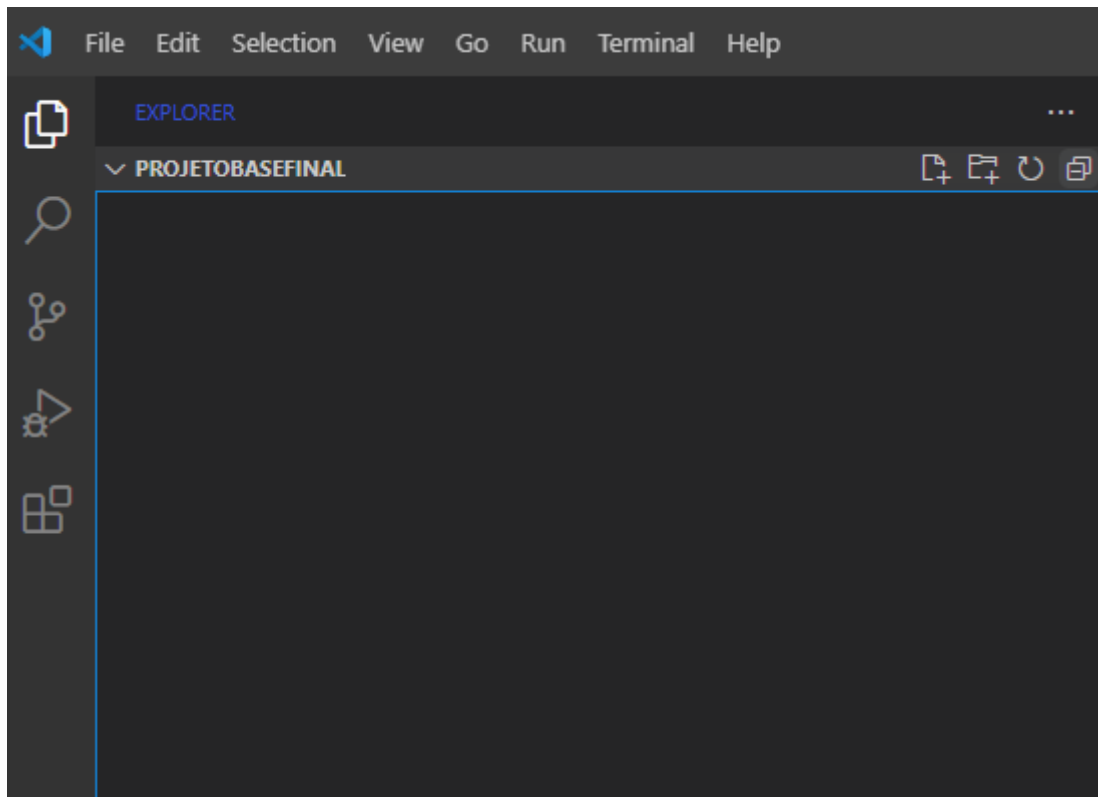


Manual do Programador Python – Parte Final

Criando a funcionalidade de excluir um nível

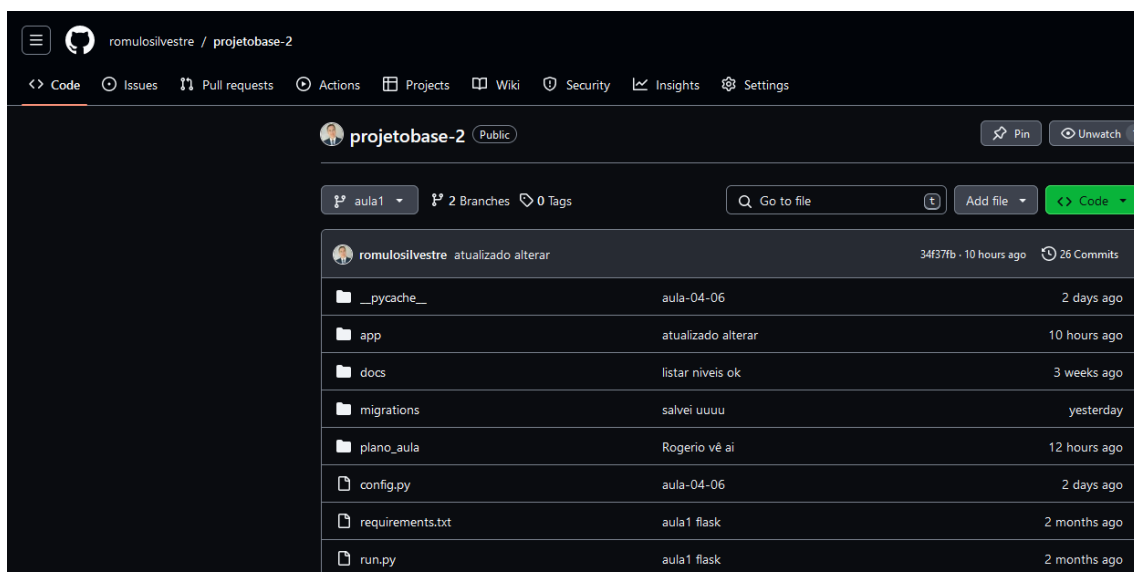
### Passo 1: Abra o Visual Studio Code



Eu criei uma pasta chamada projetobasefinal

Nela vou fazer um clone do projetobase-2 do meu github

<https://github.com/romulosilvestre/projetobase-2>



Copie o link:

<https://github.com/romulosilvestre/projetobase-2.git>

Faça um clone

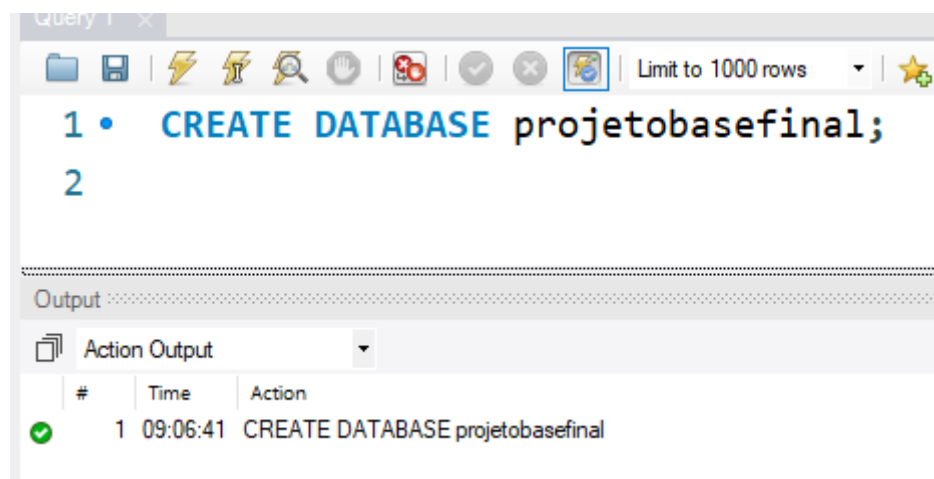
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\BIBLIOTECA\Desktop\projetobasefinal> git clone https://github.com/romulosilvestre/projetobase-2.git

PS C:\Users\BIBLIOTECA\Desktop\projetobasefinal> git clone https://github.com/romulosilvestre/projetobase-2.git
Cloning into 'projetobase-2'...
remote: Enumerating objects: 4069, done.
remote: Counting objects: 100% (4069/4069), done.
remote: Compressing objects: 100% (3087/3087), done.
remote: Total 4069 (delta 930), reused 4055 (delta 916), pack-reused 0
Receiving objects: 100% (4069/4069), 35.00 MiB | 1.23 MiB/s, done.
Resolving deltas: 100% (930/930), done.
PS C:\Users\BIBLIOTECA\Desktop\projetobasefinal> 
```

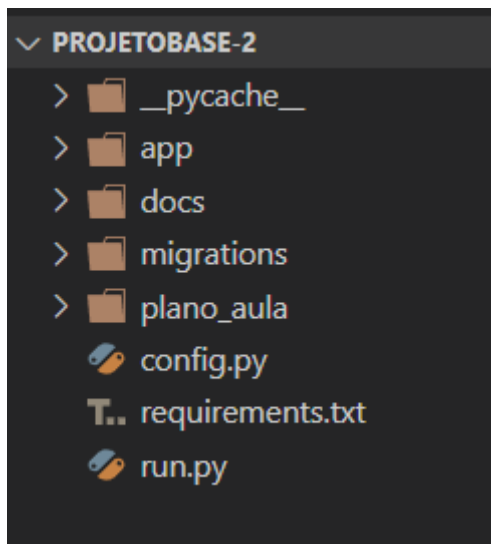
Agora abra o seu banco de dados MySQL

Use o Workbench para gerir o banco de dados



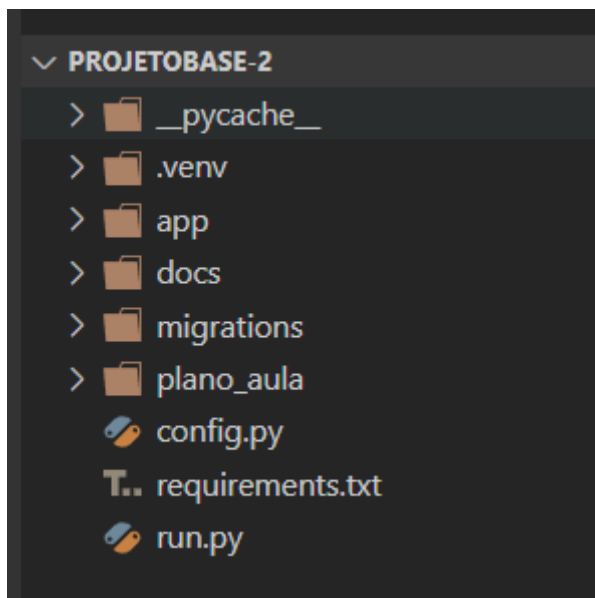
Para um melhor aproveitamento e produtividade do nosso tabalho. Vamos abrir a pasta novamente pelo Visual Studio Code, porém agora a pasta projetobase-2. Vamos?

Fica bem melhor né verdade?

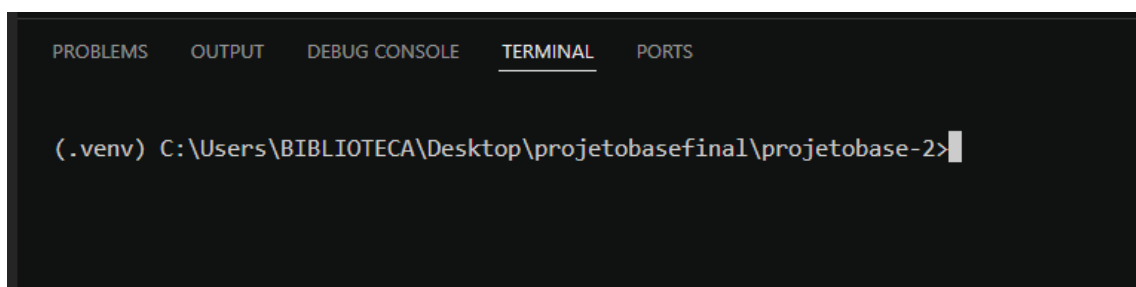


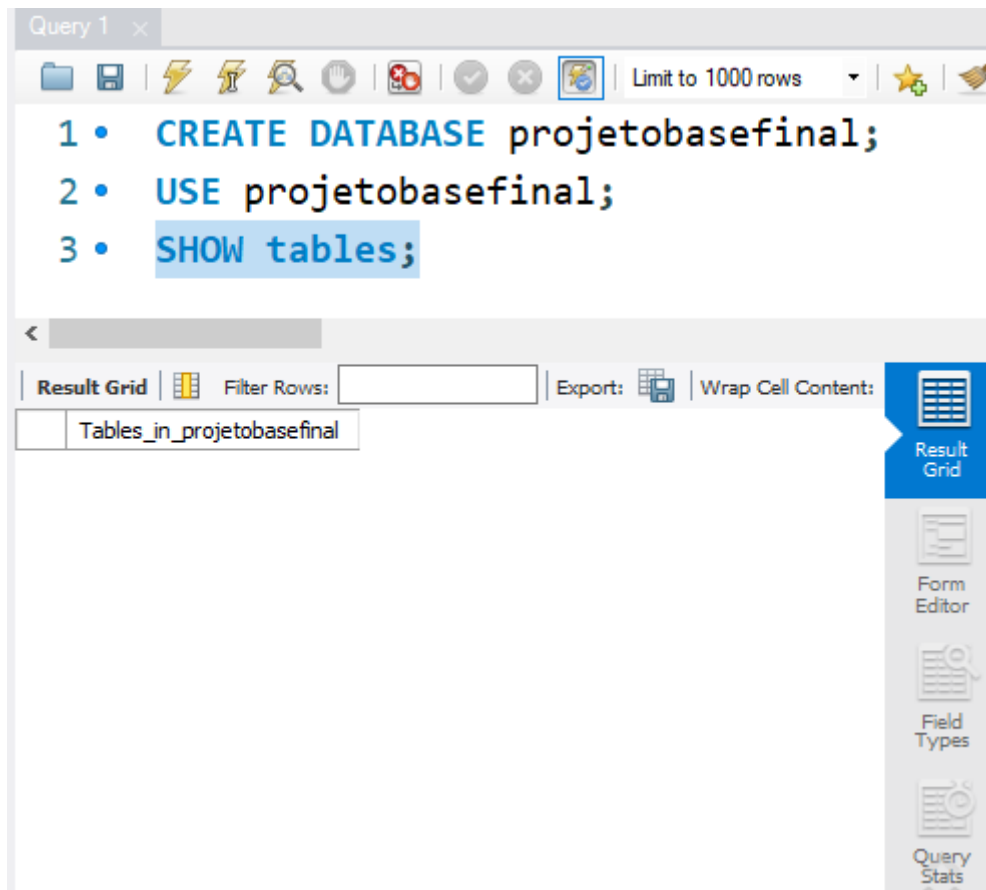
Agora dê um CTRL+SHIFT+P

Defina a .venv



Ative a .venv





No primeiro comando do Flask dá um erro.

```
(.venv) C:\Users\BIBLIOTECA\Desktop\projetobasefinal\projetobase-2>flask db init
Error: While importing 'app', an ImportError was raised:

Traceback (most recent call last):
  File "c:\Users\BIBLIOTECA\Desktop\projetobasefinal\projetobase-2\.venv\Lib\site-packag
```

Erros são uma ocorrência normal durante o desenvolvimento de software.

Atitudes que se deve ter:

- Calma
- Atenção
- Resiliência
- Não desistir
- Ler o erro
- Você é responsável pelos seus erros, a primeira pessoa que deve buscar a solução é você.
- Buscar orientação com o professor
- Buscar uma solução no stackoverflow, de alguém que já passou pelo mesmo problema.
- Buscar uma solução no You Tube, tem muitos vídeos legais e bem explicativos.

Esse erro ocorreu devo não ter instalado o flask\_babel

```
>pip install flask_babel
```

Agora você deve também ajustar a conexão

```
config.py > ...
1  #mostrar os erros em tempo real
2  DEBUG = True
3  #usuário
4  USERNAME = 'root'
5  #senha
6  PASSWORD = 'root'
7  #servidor
8  SERVER = 'localhost'
9  #nome do banco de dados
10 #create database ....
11 # SQLAlchemy
12
13 DB = 'projetobasefinal'
14 #connection string
15 SQLALCHEMY_DATABASE_URI=f'mysql://{USERNAME}:{PASSWORD}@{SERVER}/{DB}'
16 #modificação
17 SQLALCHEMY_TRACK_MODIFICATIONS = True
18 #chave secreta - hash (chave criptografada)
19 #entrar em qualquer site que gere hash - colocar o hash
20 #publicar.
21 SECRET_KEY ="8a4dbb9594173ae2747f9704468a89bd"
22
23 #português
24 BABEL_DEFAULT_LOCALE = 'pt'
25
26
```

## Ações

- flask db init
- flask db migrate -m "projeto base final"
- flask db upgrade

Agora vamos testar a aplicação especificamente:

- python run.py

```
(.venv) C:\Users\BIBLIOTECA\Desktop\projetoabasefinal\projetoabase-2>python run.py
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 774-824-780
```

**Rota:** cadnivel

<http://127.0.0.1:5000/cadnivel>

Nome

dono da marmita

Cadastrar

127.0.0.1:5000/listaniveis

### Lista de Níveis

ID	Nome	Editar	Excluir
1	dono da marmita	<a href="#">alterar</a>	<a href="#">excluir</a>

4 • `SELECT * FROM nivel;`

Result Grid

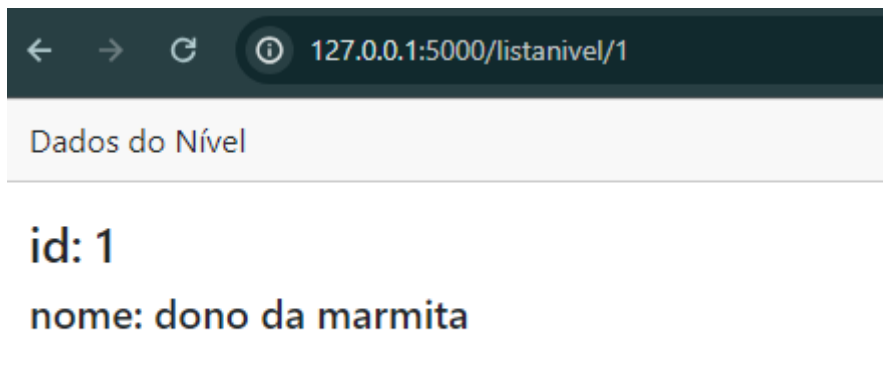
	id	nome
▶	1	dono da marmita
*	NULL	NULL

Result Grid

### Lista de Níveis

ID	Nome	Editar	Excluir
1	dono da marmita	<a href="#">alterar</a>	<a href="#">excluir</a>

Ao clicar no dono da marmita abre informações sobre esse registro.



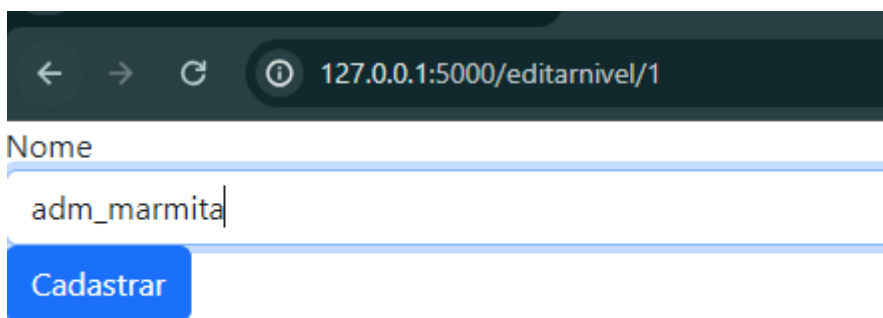
← → ↻ ⓘ 127.0.0.1:5000/listanivel/1

Dados do Nível

**id: 1**

nome: dono da marmita

Agora quero alterar de dono da marmita para adm\_marmita.



← → ↻ ⓘ 127.0.0.1:5000/editarnivel/1

Nome

adm\_marmita

Cadastrar

Pronto agora é só clicar em Cadastrar.

Ele vai redirecionar para a tela de lista de níveis

## Lista de Níveis

ID	Nome	Editar	Excluir
1	adm_marmita	<a href="#">alterar</a>	<a href="#">excluir</a>

Temos um problema no nome. Pois, quando ele cadastra usa um título, agora quando edita continua com o nome cadastrar, temos que mudar para “alterar”.

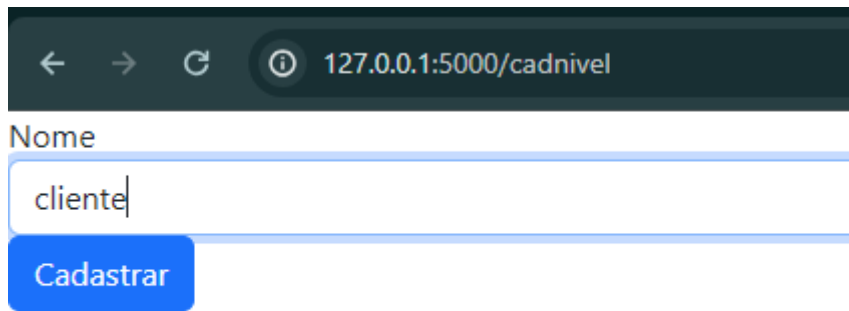
### Cadastrar

```
return render_template("nivel/form_nivel.html", form=form, editar=False)
```

### Alterar

```
return render_template("nivel/form_nivel.html", form=form, editar=True)
```

```
<button type="submit" class="btn btn-primary">{{ 'Editar' if editar else 'Cadastrar' }}</button>
```



Nome

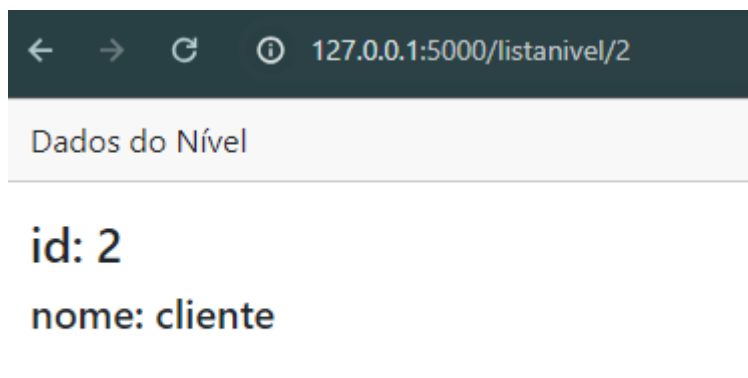
cliente

Cadastrar

## Lista de Níveis

ID	Nome	Editar	Excluir
1	adm_marmita	alterar	excluir
2	cliente	alterar	excluir

Veja as informações do nível cliente cadastrado

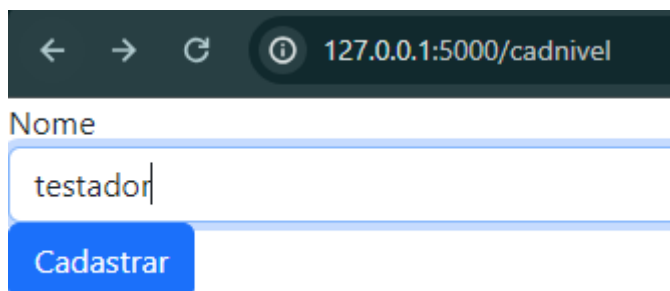


Dados do Nível

id: 2

nome: cliente

Observe que o botão agora esta cadastrar



Nome

testador

Cadastrar



Vamos clicar em alterar

## Lista de Níveis

ID	Nome	Editar	Excluir
1	adm_marmita	alterar	excluir
2	cliente	alterar	excluir
3	testador	alterar	excluir

Ao clicar em alterar

←

→

↺

i

127.0.0.1:5000/editarnivel/3

Nome

testador

Editar

Agora vamos editar

←

→

↺

i

127.0.0.1:5000/editarnivel/3

Nome

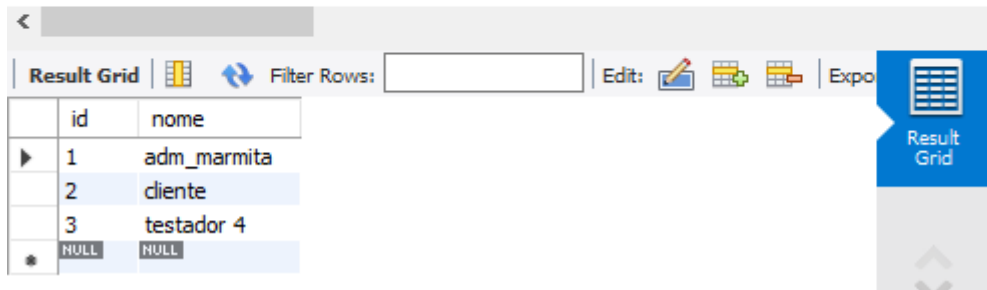
testador 4

Editar

## Lista de Níveis

ID	Nome	Editar	Excluir
1	adm_marmita	alterar	excluir
2	cliente	alterar	excluir
3	testador 4	alterar	excluir

4 • `SELECT * FROM nivel;`

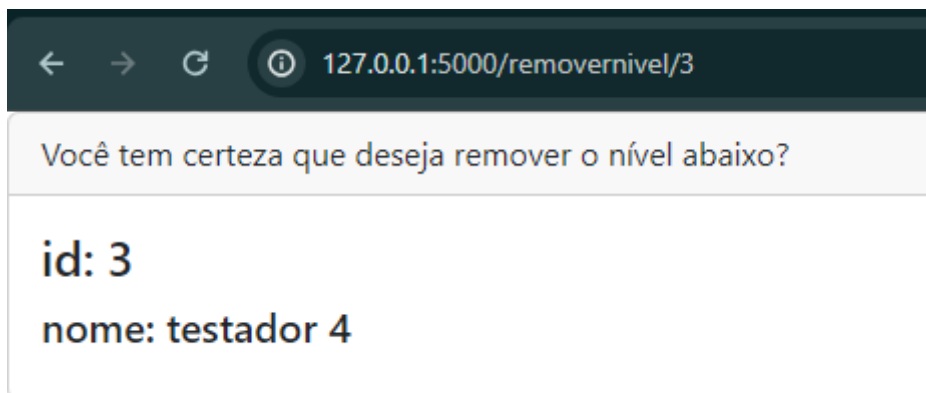


Result Grid

	id	nome
▶	1	adm_marmita
	2	cliente
	3	testador 4
*	NULL	NULL

Result Grid

```
53 @app.route("removernivel/<int:id>", methods=["POST", "GET"])
54 def remover_nivel(id):
55     nivel = nivel_model.Nivel.query.filter_by(id=id).first()
56     return render_template("nivel/remover_nivel.html", nivel=nivel)
57
```



← → ↻ ⓘ 127.0.0.1:5000/removernivel/3

Você tem certeza que deseja remover o nível abaixo?

**id: 3**

**nome: testador 4**

```
{% extends "nivel/base.html" %}
```

```
{% block titulo %}
```

```
Remoção de nivel
```

```
{% endblock titulo %}

{% block conteudo %}
    <div class="card">
        <div class="card-header">
            Você tem certeza que deseja remover o nível
            abaixo?
        </div>

        <div class="card-body">
            <h4 class="card-title"> id: {{ nivel.id }}
        </h4>
            <h5> nome: {{ nivel.nome }}</h5>
        </div>

    </div>
{% endblock conteudo %}
```

```
@app.route("/removernivel/<int:id>", methods=["POST", "GET"])
def remover_nivel(id):
    nivel = nivel_model.Nivel.query.filter_by(id=id).first()
    # vamos indicar que o usuário clicou no botão remover
    # importe request
```

```
from app import app
from flask import render_template, redirect, url_for, request #renderização
from app.forms.alpha import nivel_form
from app.models.alpha import nivel_model
from app import db
```

Colocar agora o csrf\_token para a segurança da aplicação:

```
<form method="post">
    <input type="hidden" name="csrf_token" value="{{ csrf_token() }}">
    <button class="btn btn-danger" type="submit">Remover</button>

</form>
```

Funcionalidades de exemplo entidade: **Nivel**

Listagem dos Códigos:

run.py

```
from app import app

if __name__ == "__main__":
    app.run()
```

config.py

```
#mostrar os erros em tempo real
DEBUG = True
#usuário
USERNAME = 'root'
#senha
PASSWORD = 'root'
#servidor
SERVER = 'localhost'
#nome do banco de dados
#create database ....
# SQLAlchemy

DB = 'projetoabasefinal'
#connection string
SQLALCHEMY_DATABASE_URI=f'mysql://{USERNAME}:{PASSWORD}@{SERVER}/{DB}'
#modificação
SQLALCHEMY_TRACK_MODIFICATIONS = True
#chave secreta - hash (chave criptografada)
#entrar em qualquer site que gere hash - colocar o hash
```

```
#publicar.  
SECRET_KEY ="8a4dbb9594173ae2747f9704468a89bd"  
  
#português  
BABEL_DEFAULT_LOCALE = 'pt'
```

nivel.py

```
from app import db #SQLAlchemy - Migrate:Migrar  
Classe para Tabela  
  
class Nivel(db.Model):  
    __tablename__ = "nivel"  
    #id = db.Column(tipo,chave,auto)  
    id =  
db.Column(db.Integer,primary_key=True,autoincrement=True)  
    nome = db.Column(db.String(200))
```

nível\_view.py

```
from app import app  
from flask import  
render_template,redirect,url_for,request  
#renderização  
from app.forms.alpha import nivel_form  
from app.models.alpha import nivel_model  
from app import db  
@app.route("/",methods=["POST","GET"])  
def cadastrar_nivel():  
    form = nivel_form.NivelForm()  
    if form.validate_on_submit():  
        nome = form.nome.data #capturando o conteúdo  
validado  
        nivel = nivel_model.Nivel(nome=nome)  
        try:  
            #adicionar na sessão
```

```

        db.session.add(nivel)
        #salvar a sessão
        db.session.commit()
        if request.method == 'POST':
            return redirect(url_for('listar_niveis'))
    except:
        print("nivel não cadastrado")
    return
render_template("nivel/form_nivel.html", form=form, editar=False)

@app.route("/listaniveis")
def listar_niveis():
    niveis = nivel_model.Nivel.query.all() #
Consulta todos os registros na tabela Nivel
    return render_template("nivel/lista_nivel.html",
niveis=niveis)

@app.route("/listanivel/<int:id>")
def listar_nivel(id):
    nivel =
nivel_model.Nivel.query.filter_by(id=id).first() #
Consulta todos os registros na tabela Nivel
    return
render_template("nivel/lista_nivel_id.html", nivel=nivel)

@app.route("/editarnivel/<int:id>", methods=["POST", "GET"])
def editar_nivel(id):
    nivel =
nivel_model.Nivel.query.filter_by(id=id).first()
# vamos agora criar o nosso nível de formulário
    form = nivel_form.NivelForm(obj=nivel)

```

```

# verificar se todos os dados estão ok
if form.validate_on_submit():
    nome = form.nome.data
    nivel.nome = nome

    try:
        db.session.commit()
        return redirect(url_for("listar_niveis"))
    except:
        print("o cliente não foi editado")

    return
render_template("nivel/form_nivel.html", form=form, editar=True)

@app.route("/removernivel/<int:id>", methods=["POST", "GET"])
def remover_nivel(id):
    nivel =
    nivel_model.Nivel.query.filter_by(id=id).first()
    # vamos indicar que o usuário clicou no botão
    remover
    # importe request
    if request.method == "POST":
        try:
            db.session.delete(nivel)
            db.session.commit()
            return
        except:
            print("erro ao deletar nível")
    return
render_template("nivel/remover_nivel.html", nivel=nivel)

```

```

<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible"
content="IE=edge">
    <meta name="viewport" content="width=device-
width, initial-scale=1.0">
    <title>Cadastro de Níveis</title>
    <!-- Inclusão do Bootstrap -->
    <link rel="stylesheet" href="{{ url_for('static',
filename='css/bootstrap.css') }}">

</head>
<body>

    {% block conteudo %}


    {% endblock conteudo %}

</body>
</html>

```

Form\_nivel.html

```

{% extends "nivel/base.html" %}

{% block conteudo %}

<form method="post">
    {{ form.csrf_token }}
    <div class="form-group">
        <label for="nome">Nome</label>

```



```
        {{ form.nome(class="form-control", id="nome")
    }}

    </div>
    <button type="submit" class="btn btn-primary">{{
'Editar' if editar else 'Cadastrar' }}</button>
</form>

{% if request.method == 'POST' %}
    <script>
        // Limpa o campo de texto após o envio do
        formulário
        document.getElementById("nome").value = "";
    </script>
{% endif %}

{% endblock conteudo %}
```