Manual do Programador Python – Parte Final

Criando a funcionalidade de excluir um nível

**Passo 1: Abra o Visual Studio Code**

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

**Eu criei uma pasta chamada projetobasefinal**

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

[**https://github.com/romulosilvestre/projetobase-2**](https://github.com/romulosilvestre/projetobase-2)

Tela de computador com jogo

Descrição gerada automaticamente

**Copie o link:**

[**https://github.com/romulosilvestre/projetobase-2.git**](https://github.com/romulosilvestre/projetobase-2.git)

**Faça um clone**

Texto

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

Agora abra o seu banco de dados MySQL

Use o Workbench para gerir o banco de dados

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

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?

Uma imagem contendo Texto

Descrição gerada automaticamente

Agora dê um CTRL+SHIFT+P

Defina a .venv

Tela preta com letras brancas

Descrição gerada automaticamente

Ative a .venv

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

No primeiro comando do Flask dá um erro.

Texto

Descrição gerada automaticamente

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



Agora você deve também ajustar a conexão

Texto

Descrição gerada automaticamente

**Ações**

* flask db init
* flask db migrate -m “projeto base final”
* flask db upgrade

Agora vamos testar a aplicação especificamente:

* python run.py

Texto

Descrição gerada automaticamente

**Rota:** cadnivel

<http://127.0.0.1:5000/cadnivel>

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente com confiança baixa

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Padrão do plano de fundo

Descrição gerada automaticamente

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

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

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

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Pronto agora é só clicar em Cadastrar.

Ele vai redirecionar para a tela de lista de níveis

Padrão do plano de fundo

Descrição gerada automaticamente

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**



**Alterar**



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

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Uma imagem contendo Tabela

Descrição gerada automaticamente

Veja as informações do nível cliente cadastrado

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Observe que o botão agora esta cadastrar

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Vamos clicar em alterar

Interface gráfica do usuário

Descrição gerada automaticamente

Ao clicar em alterar

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Agora vamos editar

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Tabela

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

{% 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 %}

Texto

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

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

Texto

Descrição gerada automaticamente

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 = 'projetobasefinal'

*#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 redirect(url\_for("listar\_niveis"))

        except:

             print("erro ao deletar nível")

     return render\_template("nivel/remover\_nivel.html",nivel=nivel)

base.html

<!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 %}