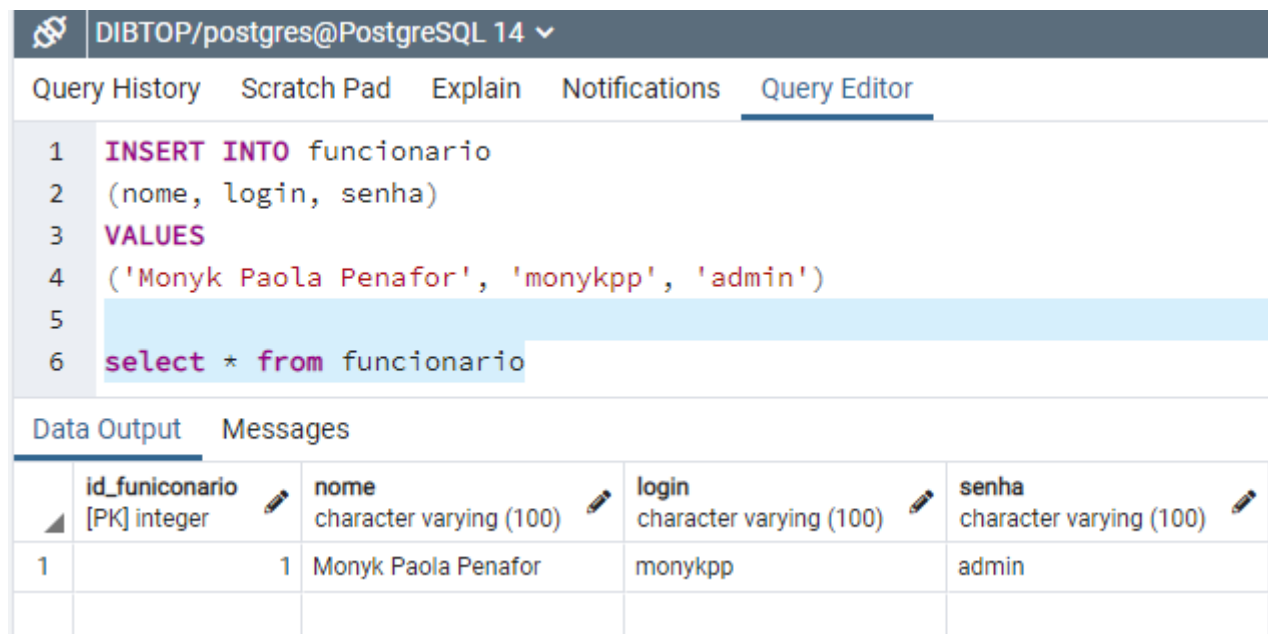


Projeto PBD - Etapa 4: Preparação do ambiente de desenvolvimento + teste de conexão + tela login

Aluno: Monyk Paola Penafor

Este projeto está sendo desenvolvido em python com kivy (biblioteca python), e banco de dados postgresql. As ferramentas utilizadas são Pycharm e PGAdmin.

1. no meu banco de dados eu fiz o primeiro cadastro pra conseguir entrar na aplicação



The screenshot shows the PGAdmin interface with the 'Query Editor' tab selected. The SQL query is as follows:

```
1 INSERT INTO funcionario
2 (nome, login, senha)
3 VALUES
4 ('Monyk Paola Penafor', 'monykpp', 'admin')
5
6 select * from funcionario
```

Below the query editor, the 'Data Output' tab shows the results of the query in a table format:

	id_funiconario [PK] integer	nome character varying (100)	login character varying (100)	senha character varying (100)
1	1	Monyk Paola Penafor	monykpp	admin

2. importações usadas até o momento:

```
from kivy.lang import Builder
from kivy.uix.screenmanager import ScreenManager
from kivymd.uix.screen import MDScreen
from kivy.core.window import Window
from kivymd.app import MDApp
from kaki.app import App
from kivymd.uix.toolbar import MDTopAppBar
import psycopg2
from kivymd.toast import toast
```

3. Fazer uma função que conecte com o banco de dados :

```
def conectar():
    conn = psycopg2.connect(
        host="localhost",
        database="DIBTOP",
        user="postgres",
        password="postgres"
    )
    return conn
```

- antes de colocar o app para funcionar é preciso fazer as telas, a tela de LOGIN com os campos que vão receber o login e a senha e os botões, e a tela PRINCIPAL(falta fazer o design), que vai aparecer quando o login e a senha forem validados:
- o código de estilização é feito em kv:

```
Builder.load_string('''

<LoginScreen>:
    FloatLayout:
        MDTextField:
            id: idlogin
            hint_text:"Login"
            pos_hint: {'center_x':0.5,'center_y':0.8}
            size_hint_x: 0.75
            max_text_length: 25
            #line_color_normal: [0,1,1,1]
            #line_color_focus: [0,1,0,1]
        MDTextField:
            id: idsenha
            hint_text:"Senha"
            pos_hint: {'center_x':0.5,'center_y':0.6}
            size_hint_x: 0.75
            password: True
            #line_color_normal: [0,1,1,1]
            #line_color_focus: [0,1,0,1]
```

```
<MainScreen>:
    FloatLayout:
        MDTextButton:
            text: "TELA PRINCIPAL"
            pos_hint: {'center_x':0.5,'center_y':0.4}
            font_size: 30
```

- Depois criamos as classes que vão receber as telas, e a main class com o screen manager para admn as telas e a função build para construir o app:

```
1 usage  Monyk Paola Penafor *
class LoginScreen(MDScreen):
    Window.size = (350, 275)

    Monyk Paola Penafor *
    def get_data(self):...

1 usage  Monyk Paola Penafor
class MainScreen(MDScreen):
    # Window.size = (700, 550)
    pass
```

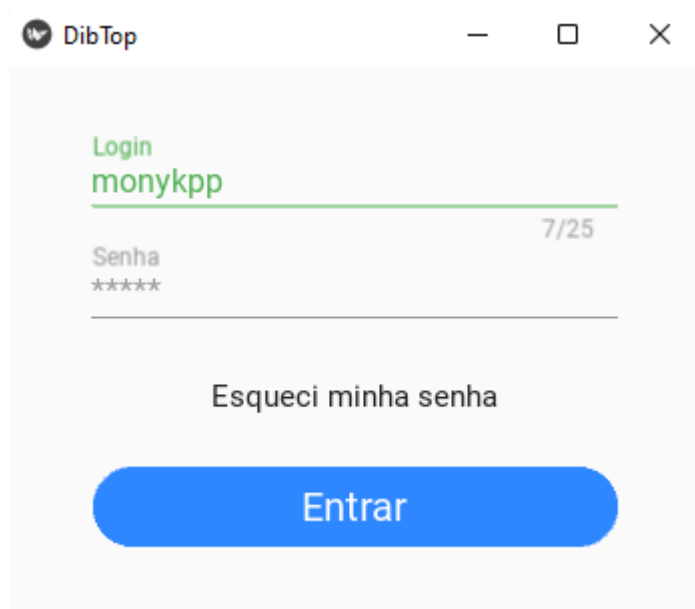
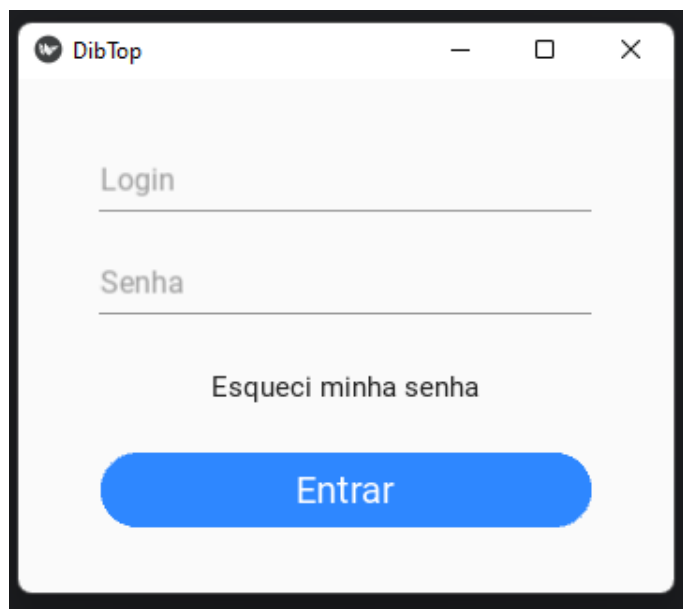
```
class DibTopApp(MDApp, App):

    Monyk Paola Penafor *
    def build_app(self, **kwargs):
        # Window.maximize()
        self.theme_cls.primary_palette = "Green"

        sm = ScreenManager()
        main_screen = MainScreen(name='principal')
        login_screen = LoginScreen(name='login')
        sm.add_widget(login_screen)
        sm.add_widget(main_screen)

        return sm
```

7. A minha tela de login:



8. Quando clicar o botão a função `get_data` vai ser acionada, dentro dela a função `validar_login` também será acionada:

```
def get_data(self):
    login = self.ids.idlogin.text
    senha = self.ids.idsenha.text
    if validar_login(login, senha):
        toast("Login e senha válidos", duration=2)
        self.manager.current = 'principal'
        # toast('Bem vindo', duration=3)
    else:
        toast("Login ou senha inválidos", duration=10)
```

1 usage Momyk Paola Penafor *

```
def validar_login(login, senha):
    try:
        conn = conectar()
        cur = conn.cursor()
        cur.execute("SELECT * FROM funcionario WHERE login = %s AND senha = %s", (login, senha))
        resultado = cur.fetchone()
        if resultado:
            return True
        else:
            return False
    except Exception as e:
        print(f"Error ao validar login: {e}")
        return False
```

9. Se o login for valido, escreve uma mensagem na tela e passa para a tela principal, se for invalido, informa o usuario e ele pode tentar outra vez:

