SENAI - Raimundo Franco Texeira

ANA PAULA, HIANA, MARIA BEATRIZ, NICOLY E YGOR CAINÃ

PROGRAMAÇÃO DE APLICATIVO

São Luis-MA

2024

ANA PAULA, HIANA, MARIA BEATRIZ, NICOLY E YGOR CAINÃ

PROGRAMAÇÃO DE APLICATIVO

Relatório de pesquisa referente ao tema programação de aplicativo, apresentado ao curso Desenvolvimento de sistema da instituição Senai Raimundo Franco Teixeira para obtenção de nota.

Orientador: Anderson Bruno

SÃO LUÍS-MA

2024

Sumário

[1 Introdução 4](#_Toc156833337)

[2 DESENVOLVIMENTO 5](#_Toc156833338)

[3 CONCLUSÃO 6](#_Toc156833339)

[4 CÓDIGO 7](#_Toc156833340)

# 1 Introdução

# Este trabalho foi desenvolvido por nossa equipe com o objetivo de criar um sistema de gerenciamento de biblioteca utilizando Python e SQLite. O projeto surgiu da necessidade de facilitar a organização e o acesso a informações sobre livros em uma biblioteca, proporcionando uma experiência eficiente tanto para os administradores quanto para os usuários A partir da análise das funcionalidades necessárias, decidimos implementar um sistema que permite o cadastro, consulta, atualização e exclusão de livros de forma simples e intuitiva. O código foi estruturado em uma classe chamada Livro, que encapsula todas as operações necessárias, garantindo uma organização clara e modular do programa. A seguir, serão apresentadas as principais características e funcionalidades do sistema.

# 2 DESENVOLVIMENTO

# Hiana: foi a principal responsável pela execução do trabalho. Fez a parte do código e implementou o projeto

# Beatriz: ficou responsável pela pesquisa e Coleta de Dados além de fazer a parte do principal.

# Estrutura do trabalho: Hiana organizou todo o conteúdo em uma estrutura lógica, dividindo o trabalho em seções claras, como introdução, desenvolvimento, conclusão e referências.

# Trabalho escrito: Ana Paula participou ativamente do trabalho escrito, contribuindo com a pesquisa e a organização das informações.

# Apresentação do trabalho(Nicoly e Ygor): Os alunos participaram de forma colaborativa no trabalho, contribuindo de maneira ativa e ajudando em tudo que precisaram. Eles se mostraram disponíveis para discutir ideias e colaborar nas diferentes etapas do projeto, assegurando que as tarefas fossem realizadas com eficiência.

# 3 CONCLUSÃO

# Concluímos que todos os integrantes aprenderam significativamente durante o processo, e a experiência foi muito importante para nós. O desenvolvimento do sistema de gerenciamento de biblioteca nos proporcionou uma compreensão prática de conceitos de programação e manipulação de bancos de dados. Além disso, o trabalho em equipe fortaleceu nossas habilidades de colaboração e comunicação. Esse projeto não apenas ampliou nosso conhecimento técnico, mas também nos ensinou a importância de trabalhar juntos para alcançar um objetivo comum. A experiência adquirida será valiosa para nossos futuros desafios acadêmicos e profissionais.

# 4 CÓDIGO

import sqlite3

class Livro:

def \_init\_(self, db\_name="biblioteca\_db"):

self.connection = sqlite3.connect(db\_name)

self.create\_table()

def create\_table(self):

with self.connection:

self.connection.execute("""

CREATE TABLE IF NOT EXISTS livros (

codigo INTEGER PRIMARY KEY AUTOINCREMENT,

titulo TEXT NOT NULL,

autor TEXT NOT NULL,

ano\_publicacao INTEGER NOT NULL

)

""")

def cadastrarLivro(self, titulo, autor, ano\_publicacao):

with self.connection:

self.connection.execute("INSERT INTO livros (titulo, autor, ano\_publicacao) VALUES (?, ?, ?)",

(titulo, autor, ano\_publicacao))

def consultarLivros(self):

cursor = self.connection.cursor()

cursor.execute("SELECT \* FROM livros")

for livro in cursor.fetchall():

print(f"Código: {livro[0]}, Título: {livro[1]}, Autor: {livro[2]}, Ano de Publicação: {livro[3]}")

def deletarLivros(self, codigo):

with self.connection:

cursor = self.connection.cursor()

cursor.execute("DELETE FROM livros WHERE codigo = ?", (codigo,))

if cursor.rowcount > 0:

print("Livro deletado com sucesso.")

else:

print("Erro: Livro não encontrado.")

def atualizarLivros(self, codigo, titulo=None, autor=None, ano\_publicacao=None):

with self.connection:

if titulo:

self.connection.execute("UPDATE livros SET titulo = ? WHERE codigo = ?", (titulo, codigo))

if autor:

self.connection.execute("UPDATE livros SET autor = ? WHERE codigo = ?", (autor, codigo))

if ano\_publicacao:

self.connection.execute("UPDATE livros SET ano\_publicacao = ? WHERE codigo = ?", (ano\_publicacao, codigo))

def consultarLivroIndividual(self, codigo):

cursor = self.connection.cursor()

cursor.execute("SELECT \* FROM livros WHERE codigo = ?", (codigo,))

livro = cursor.fetchone()

if livro:

print(f"Código: {livro[0]}, Título: {livro[1]}, Autor: {livro[2]}, Ano de Publicação: {livro[3]}")

else:

print("Livro não encontrado.")

def close(self):

self.connection.close()

def main():

biblioteca = Livro(

while True:

print("\nMenu de Gerenciamento da Biblioteca:")

print("1. Cadastrar Livro")

print("2. Consultar Livros")

print("3. Deletar Livro")

print("4. Atualizar Livro")

print("5. Consultar Livro Individual")

print("6. Sair")

opcao = input("Escolha uma opção: ")

if opcao == '1':

titulo = input("Título: ")

autor = input("Autor: ")

ano\_publicacao = int(input("Ano de Publicação: "))

biblioteca.cadastrarLivro(titulo, autor, ano\_publicacao)

print("Livro cadastrado com sucesso.")

elif opcao == '2':

biblioteca.consultarLivros()

elif opcao == '3':

codigo = int(input("Código do Livro a ser deletado: "))

biblioteca.deletarLivros(codigo)

elif opcao == '4':

codigo = int(input("Código do Livro a ser atualizado: "))

titulo = input("Novo Título (deixe em branco para não alterar): ")

autor = input("Novo Autor (deixe em branco para não alterar): ")

ano\_publicacao = input("Novo Ano de Publicação (deixe em branco para não alterar): ")

biblioteca.atualizarLivros(codigo, titulo if titulo else None, autor if autor else None, int(ano\_publicacao) if ano\_publicacao else None)

print("Livro atualizado com sucesso.")

elif opcao == '5':

codigo = int(input("Código do Livro a ser consultado: "))

biblioteca.consultarLivroIndividual(codigo)

elif opcao == '6':

biblioteca.close()

print("Saindo do sistema.")

break

else:

print("Opção inválida. Tente novamente.")

if \_name\_ == "\_main\_":

main()

import sqlite3

class Livro:

def \_init\_(self, db\_name="biblioteca\_db"):

self.connection = sqlite3.connect(db\_name)

self.create\_table()

def create\_table(self):

with self.connection:

self.connection.execute("""

CREATE TABLE IF NOT EXISTS livros (

codigo INTEGER PRIMARY KEY AUTOINCREMENT,

titulo TEXT NOT NULL,

autor TEXT NOT NULL,

ano\_publicacao INTEGER NOT NULL

)

""")

def cadastrarLivro(self, titulo, autor, ano\_publicacao):

with self.connection:

self.connection.execute("INSERT INTO livros (titulo, autor, ano\_publicacao) VALUES (?, ?, ?)",

(titulo, autor, ano\_publicacao))

def consultarLivros(self):

cursor = self.connection.cursor()

cursor.execute("SELECT \* FROM livros")

for livro in cursor.fetchall():

print(f"Código: {livro[0]}, Título: {livro[1]}, Autor: {livro[2]}, Ano de Publicação: {livro[3]}")

def deletarLivros(self, codigo):

with self.connection:

cursor = self.connection.cursor()

cursor.execute("DELETE FROM livros WHERE codigo = ?", (codigo,))

if cursor.rowcount > 0:

print("Livro deletado com sucesso.")

else:

print("Erro: Livro não encontrado.")

def atualizarLivros(self, codigo, titulo=None, autor=None, ano\_publicacao=None):

with self.connection:

if titulo:

self.connection.execute("UPDATE livros SET titulo = ? WHERE codigo = ?", (titulo, codigo))

if autor:

self.connection.execute("UPDATE livros SET autor = ? WHERE codigo = ?", (autor, codigo))

if ano\_publicacao:

self.connection.execute("UPDATE livros SET ano\_publicacao = ? WHERE codigo = ?", (ano\_publicacao, codigo))

def consultarLivroIndividual(self, codigo):

cursor = self.connection.cursor()

cursor.execute("SELECT \* FROM livros WHERE codigo = ?", (codigo,))

livro = cursor.fetchone()

if livro:

print(f"Código: {livro[0]}, Título: {livro[1]}, Autor: {livro[2]}, Ano de Publicação: {livro[3]}")

else:

print("Livro não encontrado.")

def close(self):

self.connection.close()

def main():

biblioteca = Livro()

while True:

print("\nMenu de Gerenciamento da Biblioteca:")

print("1. Cadastrar Livro")

print("2. Consultar Livros")

print("3. Deletar Livro")

print("4. Atualizar Livro")

print("5. Consultar Livro Individual")

print("6. Sair")

opcao = input("Escolha uma opção: ")

if opcao == '1':

titulo = input("Título: ")

autor = input("Autor: ")

ano\_publicacao = int(input("Ano de Publicação: "))

biblioteca.cadastrarLivro(titulo, autor, ano\_publicacao)

print("Livro cadastrado com sucesso.")

elif opcao == '2':

biblioteca.consultarLivros()

elif opcao == '3':

codigo = int(input("Código do Livro a ser deletado: "))

biblioteca.deletarLivros(codigo)

elif opcao == '4':

codigo = int(input("Código do Livro a ser atualizado: "))

titulo = input("Novo Título (deixe em branco para não alterar): ")

autor = input("Novo Autor (deixe em branco para não alterar): ")

ano\_publicacao = input("Novo Ano de Publicação (deixe em branco para não alterar): ")

biblioteca.atualizarLivros(codigo, titulo if titulo else None, autor if autor else None, int(ano\_publicacao) if ano\_publicacao else None)

print("Livro atualizado com sucesso.")

elif opcao == '5':

codigo = int(input("Código do Livro a ser consultado: "))

biblioteca.consultarLivroIndividual(codigo)

elif opcao == '6':

biblioteca.close()

print("Saindo do sistema.")

break

else:

print("Opção inválida. Tente novamente.")

if \_name\_ == "\_main\_":

main()