**FoodWise**

**Teste de Sistema:**

class Usuario:

def \_\_init\_\_(self, nome):

self.nome = nome

self.posts = [ ]

def publicar\_post(self, conteudo):

if conteudo.strip():

self.posts.append(conteudo)

return f"Post publicado: {conteudo}"

else:

return "Erro: O conteúdo do post não pode estar vazio."

**// Função de teste de sistema**

def teste\_sistema\_publicar\_post():

**// Passo 1: Criar um usuário**

usuario = Usuario("João")

print(f"Usuário criado: {usuario.nome}")

**// Passo 2: Publicar um post de doação**

conteudo\_post = "Doação de 2 pacotes de feijão em Belo Horizonte."

resultado\_post = usuario.publicar\_post(conteudo\_post)

print(resultado\_post)

**// Passo 3: Verificar se o post foi armazenado corretamente**

assert conteudo\_post in usuario.posts, "Erro: O post não foi salvo corretamente."

print("\nTeste de sistema concluído com sucesso!")

**// Executar o teste de sistema**

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

teste\_sistema\_publicar\_post()

**Fluxo do Teste**

1. **Criar o Usuário:**

* Simular a criação de um usuário no app

1. **Publicar um Post de Doação:**

* O usuário coloca um texto no post.
* O conteúdo é armazenado em uma lista de posts.

1. **Validação do Post:**

* Verifica se o post foi corretamente salvo.

**Teste de Aceitação:**

class Usuario:

def \_\_init\_\_(self, nome):

self.nome = nome

self.posts = [ ]

def publicar\_post(self, conteudo):

if conteudo.strip():

self.posts.append(conteudo)

return f"Post publicado: {conteudo}"

else:

return "Erro: O conteúdo do post não pode estar vazio."

class TesteAceitacaoPublicarPost(unittest.TestCase):

def setUp(self):

**// Configuração inicial para os testes**

self.usuario = Usuario("João")

def test\_publicar\_post\_valido(self):

**// Cenário: Publicação de um post válido**

conteudo\_post = "Doação de 2 pacotes de feijão em Belo Horizonte."

resultado = self.usuario.publicar\_post(conteudo\_post)

self.assertIn(conteudo\_post, self.usuario.posts, "O post não foi salvo corretamente.")

self.assertEqual(resultado, f"Post publicado: {conteudo\_post}", "Mensagem de sucesso incorreta.")

def test\_publicar\_post\_vazio(self):

**// Cenário: Tentativa de publicação com conteúdo vazio**

conteudo\_post = " "

resultado = self.usuario.publicar\_post(conteudo\_post)

self.assertNotIn(conteudo\_post, self.usuario.posts, "Um post vazio foi salvo erroneamente.")

self.assertEqual(resultado, "Erro: O conteúdo do post não pode estar vazio.", "Mensagem de erro incorreta.")

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

unittest.main()