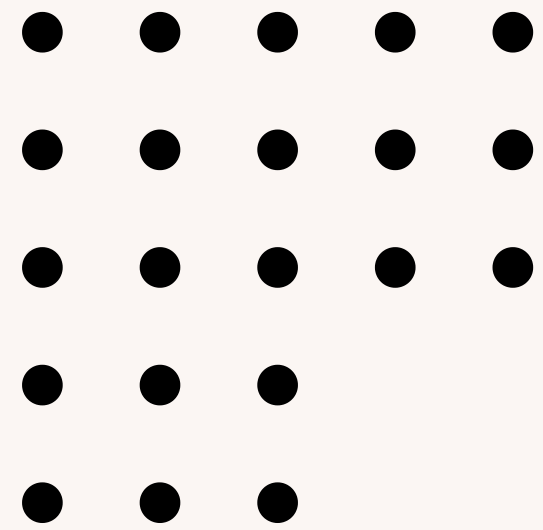
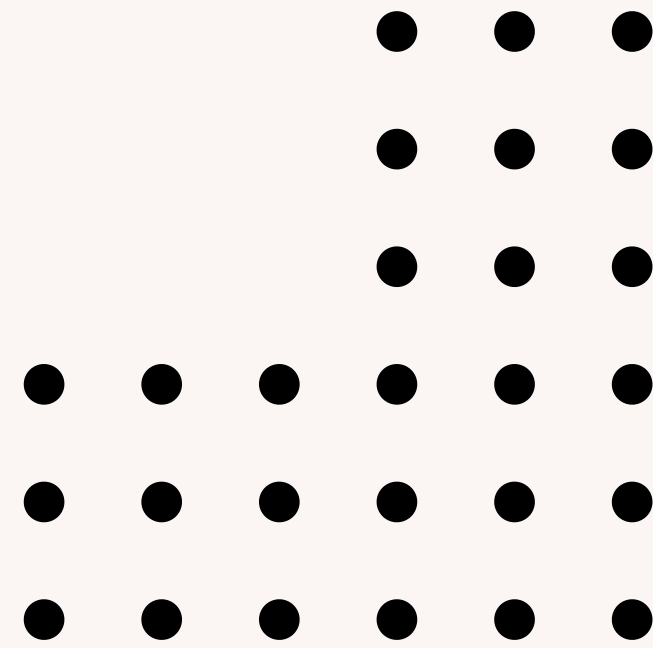


# ATIVIDADE AF11

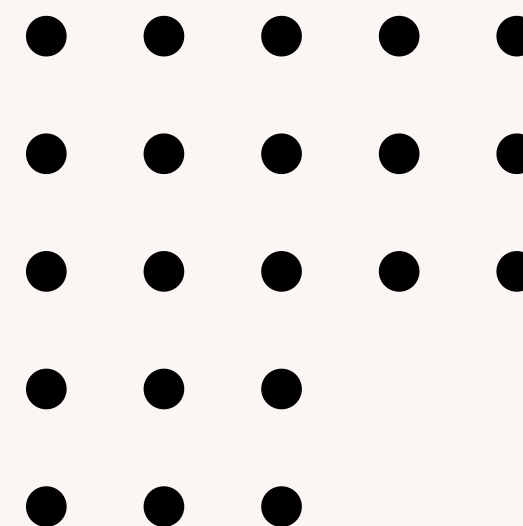
Higor rocha, Julio cesar, Matheus Paiva, Mateo Fabio, Gabriel Cabreira,  
Nicolas Medina

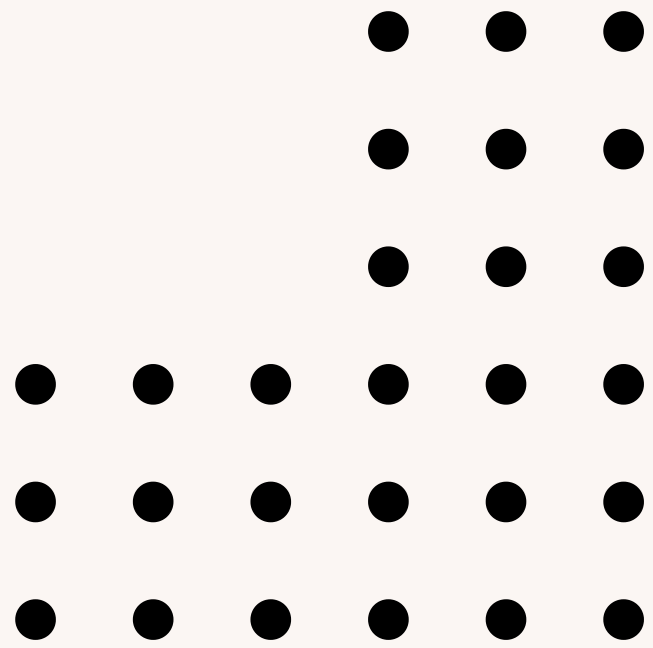




# Problema

Dificuldade na localização de arquivos,  
dentro de uma grande base de dados





# Solução

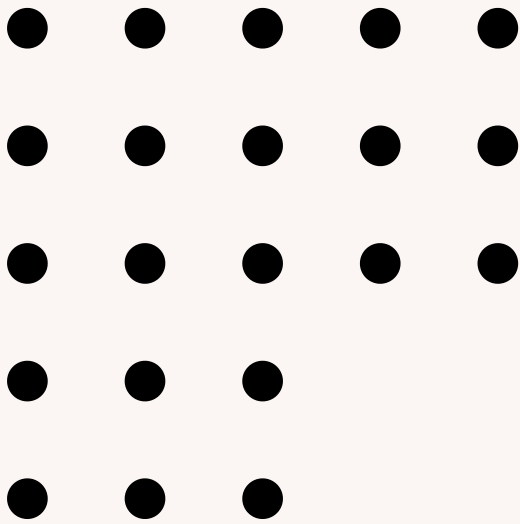
HuggingFace (para gerar embeddings)

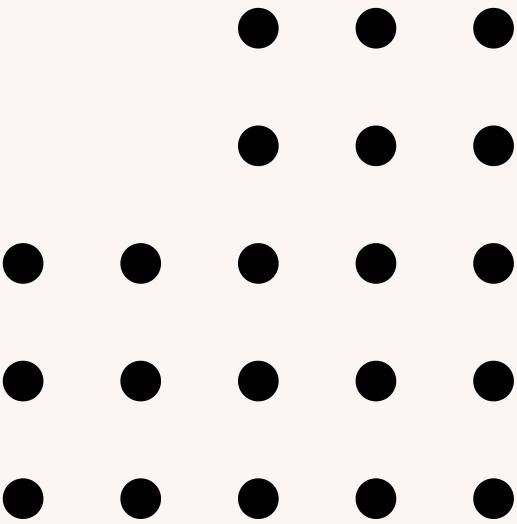
- Transformamos nomes de arquivos em vetores numéricos que representam o significado do texto.

FAISS (Vector Database)

- Criamos um índice vetorial para fazer busca rápida por similaridade.

O usuário pode:  
digitar o que procura  
o modelo gera embeddings  
o FAISS encontra o arquivo mais semântico  
o sistema pergunta se ele quer continuar





```
1 # Instalar:
2 # pip install transformers sentence-transformers faiss-cpu
3
4 from transformers import AutoTokenizer, AutoModel
5 import torch
6 import faiss
7 import numpy as np
8
9 # 1. Carregar modelo HuggingFace
10 model_name = "sentence-transformers/all-MiniLM-L6-v2"
11
12 tokenizer = AutoTokenizer.from_pretrained(model_name)
13 model = AutoModel.from_pretrained(model_name)
14
15 # Função para gerar embeddings
16 def encode(texts):
17     tokens = tokenizer(texts, padding=True, truncation=True, return_tensors="pt")
18     with torch.no_grad():
19         output = model(**tokens)
20     embeddings = output.last_hidden_state.mean(dim=1)
21     return embeddings.numpy()
22
23 # 2. Base de documentos
24 docs = [
25     "trabalho_vector_database.py",
26     "projeto_busca_semantica_vector_db.py",
27     "trabalho_huggingface_faiss.py",
28     "atividade_pratica_vector_db.py",
29     "implementacao_vector_database.py",
30 ]
31
32 # 3. Criar índice FAISS
33 embeddings = encode(docs)
34 index = faiss.IndexFlat(embeddings.shape[1])
35 index.add(embeddings)
36
37 # 4. Função para buscar documentos similares
38 def buscar_similares(query):
39     query_embedding = encode([query])[0]
40     distances, indices = index.search(query_embedding.reshape(1, -1), k=5)
41     return indices[0].tolist()
42
43 # 5. Interface de usuário
44 def main():
45     while True:
46         pergunta = input("Digite sua pergunta: ")
47         if pergunta == "":
48             continue
49         indices = buscar_similares(pergunta)
50         for i, index in enumerate(indices):
51             documento = docs[index]
52             distancia = distances[0][i]
53             print(f"{i+1} Pergunta: {pergunta}")
54             print(f"{i+1} Documento mais parecido: {documento}")
55             print(f"{i+1} Distância: {distancia}")
56         print()
57         resposta = input("Deseja fazer outra pergunta? (s/n): ")
58         if resposta.lower() != "s":
59             break
60
61 if __name__ == "__main__":
62     main()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\mathe\OneDrive\Documentos\ATIVIDADE-IA> & C:/Users/mathe/AppData/Local/Programs/Python/Python311/python.exe c:/Users/mathe/OneDrive/Documentos/ATIVIDADE-IA/atividadeIA.py

Digite sua pergunta: trabalho\_IA

1 Pergunta: trabalho\_IA  
2 Documento mais parecido: trabalho\_huggingface\_faiss.py  
3 Distância: 25.490952

Deseja fazer outra pergunta? (s/n):

