

Nhóm 1 - R7

BÁO CÁO CUỐI KÌ



THÀNH VIÊN NHÓM



BẢO PHƯƠNG

Nhóm trưởng



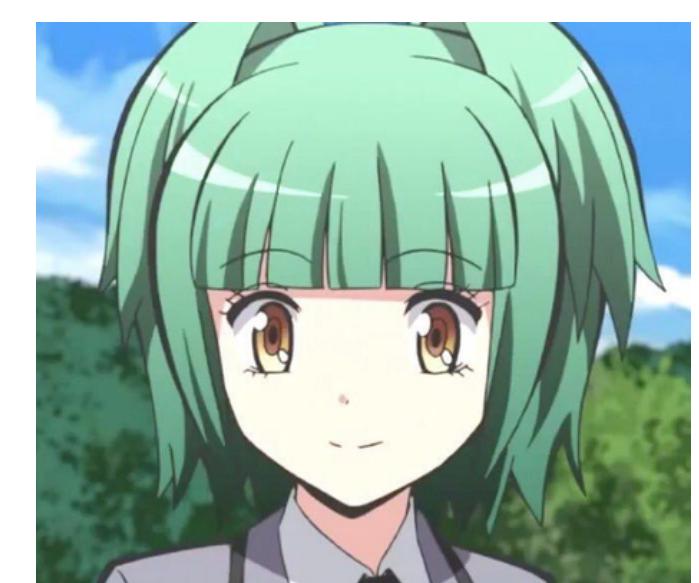
THẢO VY

Thành viên



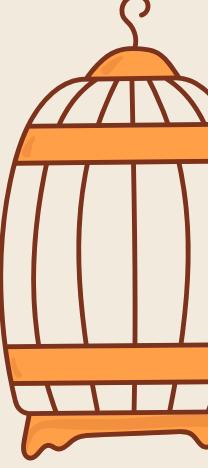
NHƯ Ý

Thành viên



MINH NGỌC

Thành viên



PLAN

Xây dựng hệ thống để xuất Anime

BƯỚC 01

BƯỚC 02

Cào dữ liệu

Đưa dữ liệu vào CSDL

BƯỚC 03

Tiền xử lý dữ liệu

BƯỚC 04

Thực hiện backup dữ liệu

BƯỚC 05

Phân quyền cho cơ
sở dữ liệu

BƯỚC 06

Trực quan dữ liệu

Xây dựng hệ thống
để xuất Anime



TỔNG QUAN DỰ ÁN

HỆ THỐNG ĐỀ XUẤT ANIME



PREVIEW

MyAnimeList
myanimelist.net

R1 : Cào dữ liệu tại MyAnimelist



```
CREATE PROCEDURE RemoveRatingComments
AS
BEGIN
    -- Cập nhật cột Rating, giữ lại các giá trị chính và xóa
    UPDATE MN
    SET Rating = CASE
        WHEN Rating LIKE 'G%' THEN 'G'
        WHEN Rating LIKE 'PG-13%' THEN 'PG-13'
        WHEN Rating LIKE 'R+%' THEN 'R+'
        ELSE Rating
    END
END
```

R3: Tiền xử lý dữ liệu

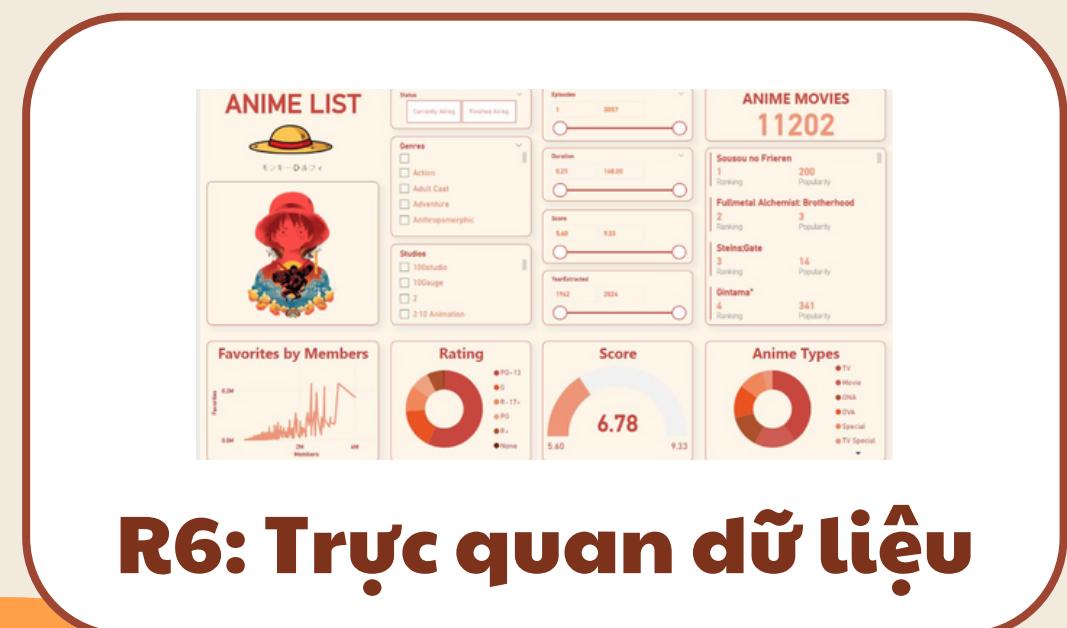
Agent Job Activity:

Name	Enabled	Status	Last Run	Next Run	Category	Runns
BackupDB	yes	Idle	Unknown	never	11/10/2023	[Uncate...]
BackupLog	yes	Idle	Unknown	never	11/30/2023	[Uncate...]
syspolicy_purge...	yes	Idle	Unknown	never	11/3/2023	[Uncate...]

R4: Thực hiện backup

Quyền	Admin	DE	DA
Create/Delete table	✓		
Alter table	✓		
Backup	✓	✓	

R5: Phân quyền



1. Import thư viện

```
import pyodbc  
import pandas as pd  
import matplotlib.pyplot as plt
```

3. Kết nối

```
# Tạo kết nối  
conn = pyodbc.connect(  
    f"DRIVER={{ODBC Driver 17 for SQL Server}};SERVER={server};DATABASE={database};UID={username};PWD={password}")
```

4. Câu lệnh SQL truy vấn tương ứng

```
# Câu truy vấn SQL  
query = ""  
SELECT TOP 10 A.AnimeName, S.Members  
FROM Anime A  
JOIN Statistic S ON A.StatisticID = S.StatisticID  
ORDER BY S.Members DESC;  
...  
  
# Đọc dữ liệu vào DataFrame  
df = pd.read_sql(query, conn)  
  
# Kiểm tra dữ liệu  
print(df.head())
```

2. Kết nối SQL server

```
# Thông tin kết nối  
server = r'DESKTOP-6LJT5VT\NHUYXI' # Tên server  
database = 'Anime_CauTruc' # Tên database  
username = 'sa' # Username (nếu cần)  
password = 'ttny01122004' # Password (nếu cần)
```

5. Trực quan hóa

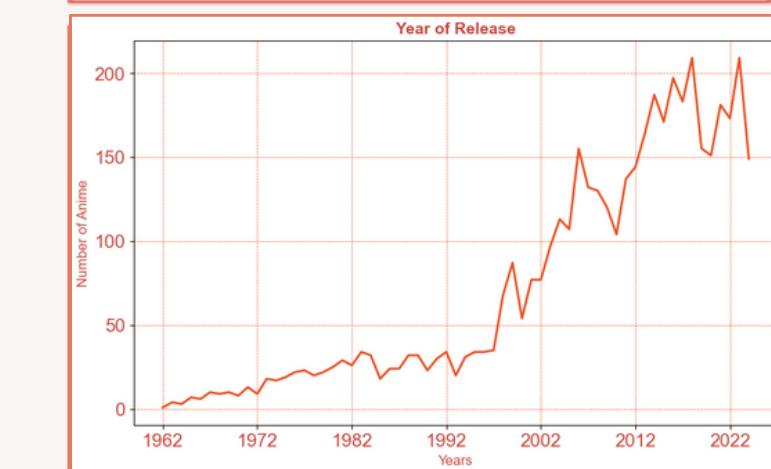
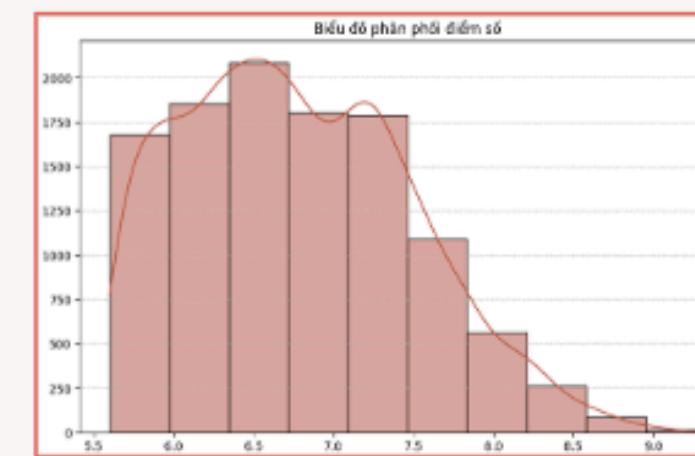
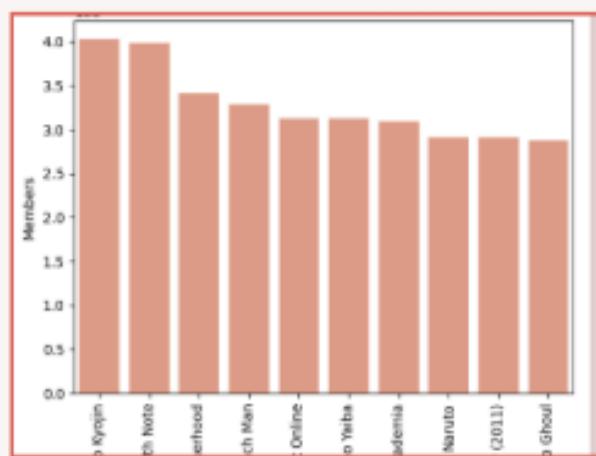
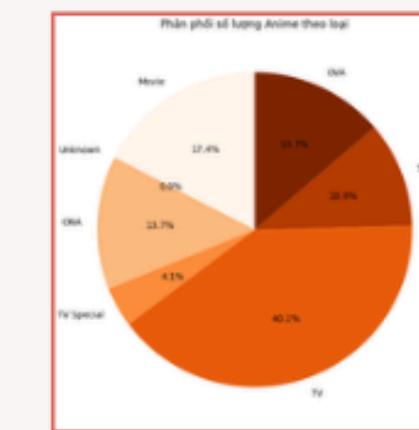
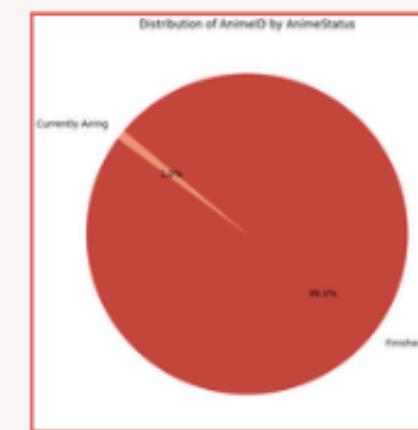
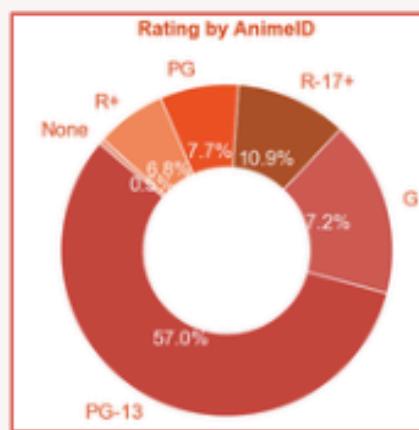
```
import seaborn as sns  
import matplotlib.pyplot as plt  
  
# Màu HEX duy nhất cho tất cả các cột  
single_color = '#ee9679' # mã màu  
  
# Vẽ biểu đồ  
sns.barplot(x='AnimeName', y='Members', data=df, color=single_color)  
  
# Tùy chỉnh biểu đồ  
plt.xticks(rotation=90)  
plt.title('Top 10 AnimeNames with the highest Members (Single Color)')  
plt.show()
```

6. Đóng kết nối

```
# Đóng kết nối  
conn.close()
```

DASHBOARD (PYTHON)

ANIME DATA



XÂY DỰNG HỆ THỐNG

HỆ THỐNG ĐỂ XUẤT ANIME



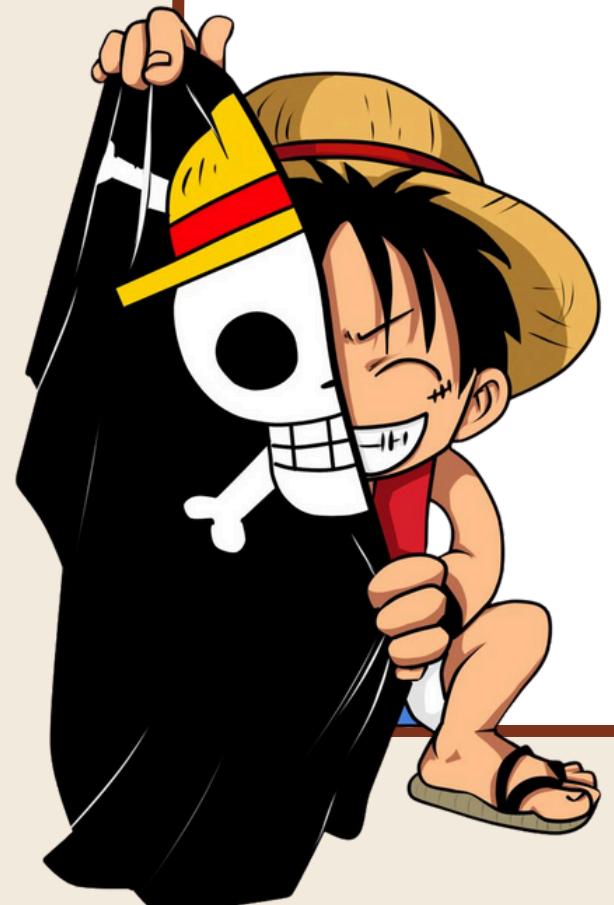


Ý TƯỞNG THỰC HIỆN

Hệ thống đề xuất Anime

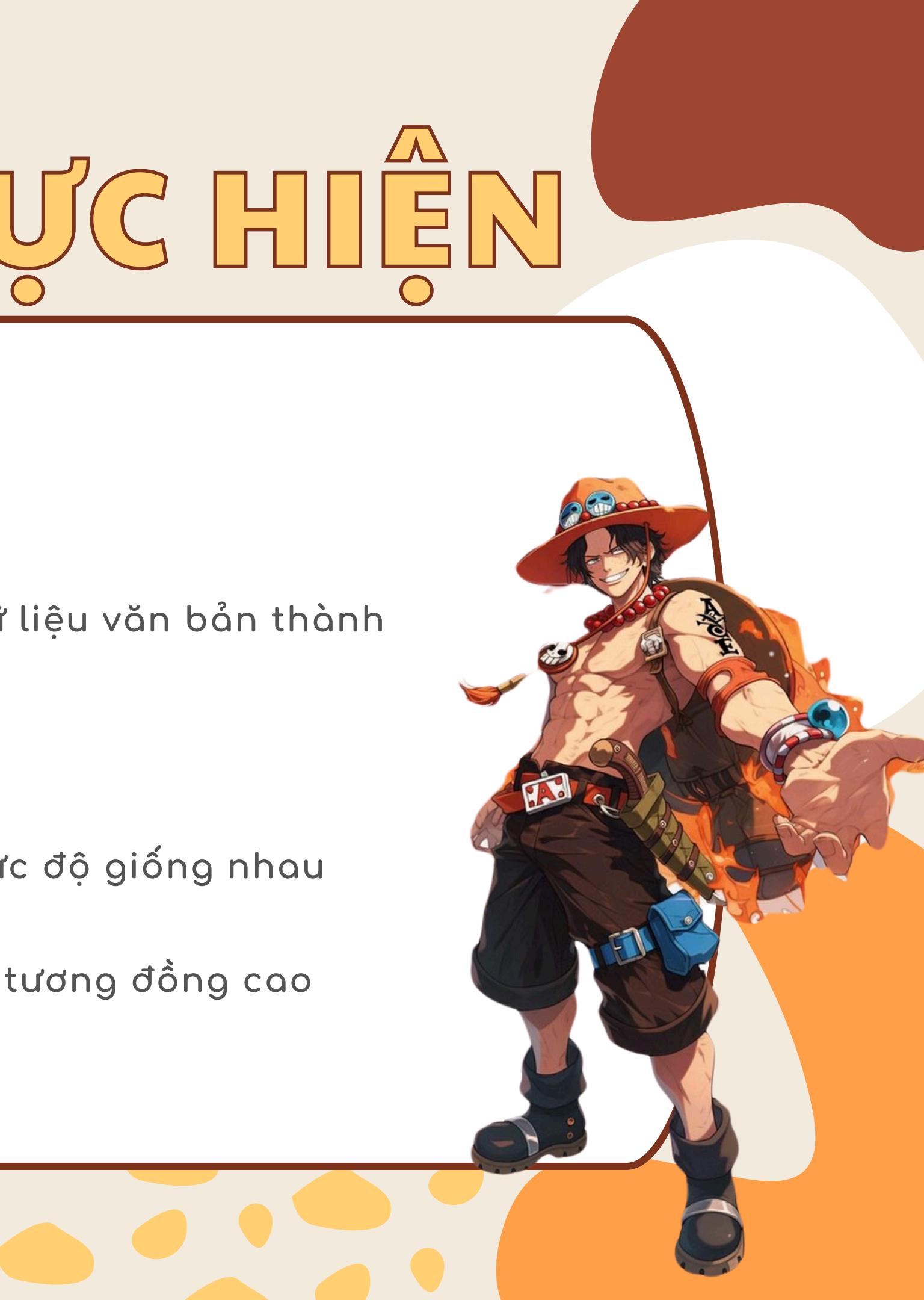
TF-IDF Vectorizer

- Sử dụng TF-IDF Vectorizer để biến đổi dữ liệu văn bản thành vectơ đặc trưng



Cosine Similarity

- Sử dụng Cosine Similarity để đo lường mức độ giống nhau giữa các bộ phim.
- Sắp xếp và chọn ra những bộ phim có độ tương đồng cao nhất để đề xuất.



CODE

Import thư viện

```
import pandas as pd
import pyodbc
import numpy as np
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
```

Kết nối tới CSDL

```
# Kết nối tới SQL Server
conn = pyodbc.connect(
    'DRIVER={ODBC Driver 17 for SQL Server};'
    'SERVER=DESKTOP-IFEAUMD;'
    'DATABASE=Anime_CauTruc;'
    'UID=sa;'
    'PWD=3110'
)
```



Truy vấn dữ liệu

```
# Kết nối tới SQL Server
conn = pyodbc.connect(
    'DRIVER={ODBC Driver 17 for SQL Server};'
    'SERVER=DESKTOP-IFEAUMD;'
    'DATABASE=Anime_CauTruc;'
    'UID=sa;'
    'PWD=3110'
)
# Truy vấn dữ liệu
query = """
SELECT a.AnimeName, a.Synopsis, a.Episodes, a.AnimeStatus, a.Duration_min, g.GenreName,
       s.StudioName, r.Rating, p.Premiered, t.AnimeType, st.Ranking, st.Popularity
FROM   Anime a
JOIN   Anime_genres ag ON a.AnimeID = ag.AnimeID
JOIN   Genres g ON ag.GenresID = g.GenresID
JOIN   Anime_Studio asd ON a.AnimeID = asd.AnimeID
JOIN   Studio s ON asd.StudioID = s.StudioID
JOIN   Rating r ON a.RatingID = r.RatingID
JOIN   Statistic st ON a.StatisticID = st.StatisticID
JOIN   Premiered p ON a.PremieredID = p.PremieredID
JOIN   AnimeType t ON a.AnimeTypeID = t.AnimeTypeID
"""
df_f = pd.read_sql_query(query, conn)

# Đóng kết nối
conn.close()
```

Đóng kết nối

```
# Đóng kết nối
conn.close()

0.2s
```



Tiền xử lí dữ liệu

```
import re
df['Genres'] = df['Genres'].apply(lambda x: re.sub(r',', ' ', x.lower()) if isinstance(x, str) else '')

# Tạo cột mô tả tổng hợp
df['Description'] = df['Studio'] + ' ' + \
                     df['Premiered'] + ' ' + \
                     df['Ranking'].astype(str) + ' ' + \
                     df['Genres'] + ' ' + \
                     df['Episodes'].astype(str) + ' ' + \
                     df['AnimeStatus'] + ' ' + \
                     df['Duration_min'].astype(str) + ' ' + \
                     df['Popularity'].astype(str) + ' ' + \
                     df['Rating'].astype(str)

# Tạo DataFrame mới chỉ gồm các cột cần thiết
new_df = df[['AnimeName', 'Description', 'Synopsis']]
```



Chuyển đổi văn bản thành dạng số (vector)

```
# Sử dụng TfidfVectorizer
tfidf_vec = TfidfVectorizer(stop_words='english', max_features=1500)
genre_vec = tfidf_vec.fit_transform(new_df['Description'])
```

Hàm để xuất phim dựa trên cosine similarity

```
def get_recommend(title, cosine_sim, length=10):
    # Tạo một series để ánh xạ tên phim với chỉ số
    movie_title_series = pd.Series(df.index, index=new_df['AnimeName']).drop_duplicates()

    # Lấy chỉ số của bộ phim dựa trên tên
    if title not in movie_title_series:
        raise ValueError(f"'{title}' không tồn tại trong dữ liệu.")
    movie_idx = movie_title_series[title]

    # Tính toán điểm tương đồng
    sim_scores = list(enumerate(cosine_sim[movie_idx]))
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
    sim_scores = sim_scores[1:length+1] # Bỏ chính phim được chọn (phim đầu tiên)

    # Lấy tên phim và nội dung
    movie_indices = [i[0] for i in sim_scores]
    recommendations = new_df.iloc[movie_indices][['AnimeName', 'Synopsis']]
    return recommendations
```



Đề xuất dựa trên danh sách phim người dùng đã chọn

```
def get_recommend_by_user_list(user_list, length=10):
    # Kiểm tra đầu vào
    if not user_list:
        raise ValueError("Danh sách phim người dùng không được để trống.")
    # Tạo vector người dùng bằng cách lấy trung bình vectơ của các phim
    user_vec = np.mean(
        [genre_vec[new_df[new_df['AnimeName']] == movie].index[0]].toarray()
        for movie in user_list if movie in new_df['AnimeName'].values,
        axis=0
    )
    # Tính toán độ tương đồng giữa vector người dùng và các bộ phim
    cosine = cosine_similarity(user_vec, genre_vec)
    # Sắp xếp các bộ phim theo độ tương đồng giảm dần
    sim_scores = list(enumerate(cosine[0]))
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
    # Loại bỏ các phim đã chọn trong danh sách người dùng
    sim_scores = [score for score in sim_scores if new_df.iloc[score[0]]['AnimeName'] not in user_list]
    sim_scores = sim_scores[:length] # Chỉ lấy số lượng phim cần đề xuất
    # Lấy tên phim và nội dung
    movie_indices = [i[0] for i in sim_scores]
    recommendations = new_df.iloc[movie_indices][['AnimeName', 'Synopsis']]
    return recommendations
```



Thực hiện đề xuất

```
name = input('Nhập tên phim bạn đã coi:')

# Tính độ tương đồng cosine
cos_similar = cosine_similarity(genre_vec, genre_vec)

# từ 1 phim đề xuất ra
# Ví dụ sử dụng:
print(f'Kết quả đề xuất cho phim {name}:')
recommendation_1 = get_recommend(name, cos_similar, length=10)
display(recommendation_1)
```



Kết quả đề xuất cho phim Naruto:

	AnimeName	Synopsis
9738	Street Fighter II: Yomigaeru Fujiwara-Kyou - T...	An educational anime featuring Ryu, Ken, Chun...
5381	Naruto: Takigakure no Shitou - Ore ga Eiyuu Da...	After safely escorting the cowardly Takigakure...
5148	Yuu☆Yuu☆Hakusho (Movie)	Yusuke Urameshi is not exactly what you'd call...
2356	Boruto: Naruto the Movie	The spirited Boruto Uzumaki, son of Seventh Ho...
3362	Yuu☆Yuu☆Hakusho: Eizou Hakusho - Ankoku Bujuts...	These are two recap specials that focus on Tea...
2050	Cang Yuan Tu: Dongning Fu Fanwai Pian	No synopsis information has been added to this...
2157	Feng Ling Yu Xiu	It is rumored in the martial world that a myst...
9753	Ninkuu: Knife no Bohyou	After the war, Fuusuke, Aicho and Touji head t...
2456	Naruto x UT	All-new animation offered throughout UNIQLO cl...
3452	Yuu☆Yuu☆Hakusho: Meikai Shitou-hen - Honoo no ...	Millennia ago, a war was fought between the Ne...

Thực hiện đề xuất dựa trên danh sách người dùng

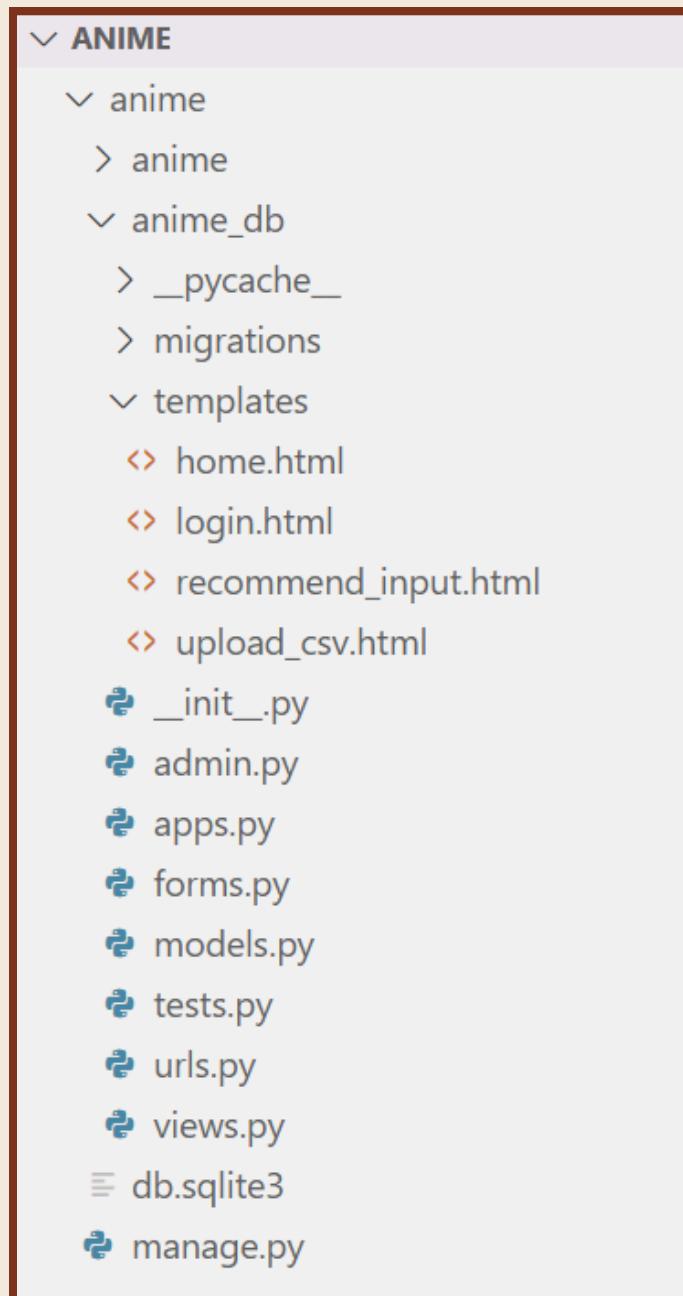
```
print("\nKết quả đề xuất dựa trên danh sách người dùng:")
user_selected_movies = ['Dragon Ball', 'Bleach', 'Boku no Hero Academia']
recommendations = get_recommend_by_user_list(user_selected_movies, length=10)
display(recommendations)
```

A smaller illustration of a boy with short dark hair, wearing a grey t-shirt and blue pants, holding a small white cat against his chest. He is smiling and looking down at the cat.

Kết quả đề xuất dựa trên danh sách người dùng:		
	AnimeName	Synopsis
2153	Boku no Hero Academia the Movie 3: World Heroe...	Adaptation of "No.XXX Hawks: SOOTHE" side-chap...
2484	Boku no Hero Academia the Movie 1: Futari no H...	Episode 1 is based on one-shot manga released ...
2861	Boku no Hero Academia: Hero Note	Recap of Boku no Hero Academia that aired a we...
3472	Boku no Hero Academia: Ikinokore! Kesshi no Su...	The stage this time is a survival training cou...
6158	Boku no Hero Academia: Memories	Recap of Boku no Hero Academia leading up to t...
3148	Boku no Hero Academia: Training of the Dead	Returning from their internships, the students...
3147	Boku no Hero Academia: Sukue! Kyuujo Kunren!	UA High School must regain the public's confid...
2448	Kekkai Sensen: Ousama no Restaurant no Ousama	For a rare occasion, the members of Libra have...
3256	Hunter x Hunter Pilot	Gon Freecss, a young boy living on Whale Islan...
298	Boku no Hero Academia 6th Season	With Tomura Shigaraki at its helm, the former ...

UPLOAD TO WEBSITE

Sử dụng thư viện Django



```
# models.py
from django.db import models

class Anime(models.Model):
    name = models.CharField(max_length=255)
    studio = models.CharField(max_length=255, blank=True, null=True)
    premiered = models.CharField(max_length=255, blank=True, null=True)
    ranking = models.CharField(max_length=255, blank=True, null=True)
    genres = models.TextField(blank=True, null=True)
    episodes = models.IntegerField(blank=True, null=True)
    status = models.CharField(max_length=255, blank=True, null=True)
    duration_min = models.FloatField(blank=True, null=True)
    popularity = models.FloatField(blank=True, null=True)
    rating = models.FloatField(blank=True, null=True) # Giữ lại kiểu FloatField cho rating
    synopsis = models.TextField(blank=True, null=True)

    def __str__(self):
        return self.name
```

Cấu trúc file

Tạo model -> Lưu dữ liệu



UPLOAD TO WEBSITE

Tạo view -> xử lý yêu cầu từ user

```
from django.shortcuts import render, redirect
from django.contrib.auth import authenticate, login, logout
from django.contrib import messages

# Create your views here.

def home(request):
    return render(request, 'home.html')

def log_in(request): ...

def log_out(request):
    logout(request)
    messages.success(request, "Bạn đã đăng xuất!")
    return redirect('home')

import pandas as pd ...

def upload_csv(request): ...
def recommend_movie(request): ...
```

Tạo form up data

```
from django import forms

class UploadCSVForm(forms.Form):
    file = forms.FileField()
```

Tạo đường dẫn

```
from django.urls import path
from . import views

urlpatterns = [
    path('', views.home, name='home'),

    path('login/', views.log_in, name='login'),
    path('logout/', views.log_out, name='logout'),

    path('recommend/', views.recommend_movie, name='recommend_movie'),
    path('upload/', views.upload_csv, name='upload_csv'),
]
```

UPLOAD TO WEBSITE

Tạo template



Template DashBoard

The screenshot shows the 'ANIME RECOMMENDATIONS' dashboard for the anime 'Gintama'. The interface includes a sidebar with dropdown menus for 'Anime Name' (set to 'Gintama'), 'Statistic' (set to 'Favorites'), and filters for 'Premiered' (Spring 2006), 'Studio' (Sunrise), and 'Rating' (PG-13). The main content area displays the anime's status as 'Finished Airing' and type as 'TV'. It also shows a synopsis: 'Edo is a city that was home to the vigor and ambition of samurai across the country. However, following feudal Japan's surrender to powerful aliens known as the "Amanto," those aspirations now seem unachievable. With the once-influential shogunate rebuilt as a puppet government, a ne...'. Below this, there are sections for 'TOTAL ANIME' (11,210) and 'Statistic' (5), showing a chart with 213,808 data points. At the bottom, there are filters for 'Duration' (24 min. per ep.) and 'Anime Type' (TV Special 4.11%, Movie 0.04%, TV 95.85%).

Template Recommendation System

The screenshot shows the 'Template Recommendation System' interface. It features a logo for 'TAKU AI ANIME RECOMMENDATIONS' and a search bar with the placeholder 'Enter the movie name:'. Below the search bar is a 'Get Recommendations' button. At the bottom left is a 'Back to Home' button.



KẾT LUẬN

HỆ THỐNG ĐỀ XUẤT ANIME



HỆ THỐNG ĐỀ XUẤT ANIME

ƯU ĐIỂM

Khả năng linh hoạt cao

Gọi ý đa dạng

Tiện lợi cho người dùng

Không cần dữ liệu đầu vào phức tạp, chi tiết

NHƯỢC ĐIỂM

Phụ thuộc vào dữ liệu người dùng

Hạn chế hiểu gu người xem

Dễ bị lặp lại các đề xuất

Không phân biệt được chất lượng phim



ĐỀ XUẤT CẢI TIẾN



Tăng tính cá nhân hoá

Thêm thông tin như lịch sử xem, thời gian xem, hành vi click để hiểu rõ hơn sở thích của người dùng



Quan tâm tới giao diện người dùng

Visualization các dữ liệu giúp người dùng dễ dàng tiếp cận và hiểu các đề xuất hơn



Tự động cập nhật dữ liệu

Tạo pipeline tự động để cập nhật anime mới và tính toán lại các đặc trưng khi có dữ liệu mới



QUÁ TRÌNH THỰC HIỆN DỰ ÁN

THUẬN LỢI

Hiểu rõ về chủ đề Anime

Dữ liệu phong phú, rõ ràng

Công cụ, thư viện hỗ trợ

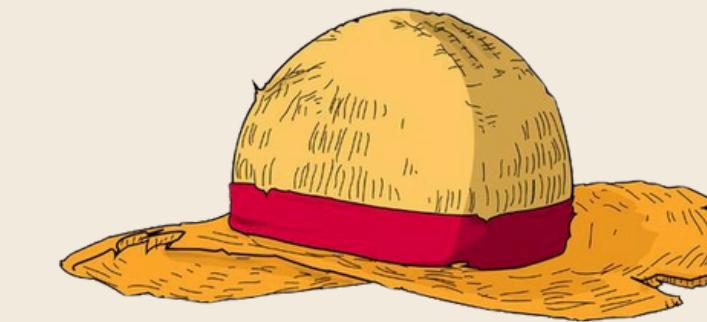
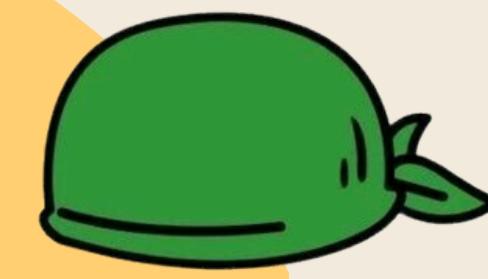
Thành viên nhóm có trách nhiệm,
luôn cố gắng phát triển

KHÓ KHĂN

Dung lượng, tài nguyên

Sự thay đổi trong yêu cầu

Hạn chế về thời gian
mở rộng, phát triển



CẢM ƠN THẦY

vì ĐÃ LẮNG NGHE

