```python
import streamlit as st
import pickle
import pandas as pd
import requests


def fetch_poster(movie_id):
    response = requests.get("https://api.themoviedb.org/3/movie/{}?api_key=ce07f8f5c01025d46d1d86d7b3e63fa3&language=en-US".format(movie_id))
    data = response.json()
    print(data)
    return "https://image.tmdb.org/t/p/w500/" + data['poster_path']

def recommend(option):
    if option in movies['title'].values:
        movie_index = movies[movies['title'] == option].index[0]
        distance = similarity[movie_index]
        movie_list = sorted(list(enumerate(distance)), reverse=True, key=lambda x: x[1])[1:6]
        recommended_movies = []
        recommended_movies_posters=[]
        for i in movie_list:
            movie_id=movies.iloc[i[0]].movie_id
            recommended_movies.append(movies.iloc[i[0]])
            # fetch poster from api
            recommended_movies_posters.append(fetch_poster(movie_id))
    elif option in unique_genres:
        mask = movies.genres.apply(lambda x: option in x)
        filtered_movie = movies[mask]
        filtered_movie = filtered_movie.sort_values(by='popularity', ascending=False)[0:5]
        suggested_movies = [(idx, row['movie_id']) for idx, row in filtered_movie.iterrows()]

        recommended_movies = []
        recommended_movies_posters=[]
        for i in suggested_movies:
            movie_id = i[1]
            recommended_movie = movies[movies['movie_id'] == movie_id].iloc[0]
            recommended_movies.append(recommended_movie)
            recommended_movies_posters.append(fetch_poster(movie_id))
    elif option in actors:
        mask = movies.cast.apply(lambda x: option in x)
        filtered_movie = movies[mask]
        filtered_movie = filtered_movie.sort_values(by='popularity', ascending=False)[0:5]
        suggested_movies = [(idx, row['movie_id']) for idx, row in filtered_movie.iterrows()]

        recommended_movies = []
        recommended_movies_posters=[]
        for i in suggested_movies:
            movie_id = i[1]
            recommended_movie = movies[movies['movie_id'] == movie_id].iloc[0]
            recommended_movies.append(recommended_movie)
            recommended_movies_posters.append(fetch_poster(movie_id))
    else:
        mask = movies.crew.apply(lambda x: option in x)
        filtered_movie = movies[mask]
        filtered_movie = filtered_movie.sort_values(by='popularity', ascending=False)[0:5]
        suggested_movies = [(idx, row['movie_id']) for idx, row in filtered_movie.iterrows()]

        recommended_movies = []
        recommended_movies_posters=[]
        for i in suggested_movies:
            movie_id = i[1]
            recommended_movie = movies[movies['movie_id'] == movie_id].iloc[0]
            recommended_movies.append(recommended_movie)
            recommended_movies_posters.append(fetch_poster(movie_id))
    return recommended_movies,recommended_movies_posters


movie_dict=pickle.load(open('movie_dict2.pkl','rb'))
movies = pd.DataFrame(movie_dict)

similarity=pickle.load(open('similarity2.pkl','rb'))

unique_genres=pickle.load(open('unique_genres2.pkl','rb'))

actors=pickle.load(open('actors.pkl','rb'))

directers=pickle.load(open('directer.pkl','rb'))

st.title('Movie Recommender system')

options = {
    'Genre': unique_genres,
    'Movie Title': movies['title'].values,
    'Actor':actors,
    'Directer':directers}

select_filter=st.selectbox("Select a filter : ",list(options.keys()))

if select_filter:
    select_option=st.selectbox(f'Select your {select_filter}: ', options[select_filter])
    if st.button('Recommend'):
        rec_movies,posters=recommend(select_option)
        if select_option in actors or select_option in directers:
```

```
            n=len(posters)
        else:
            n=5
        for i in range(n):
            col1, col2= st.columns(2)
            with col1:
                st.image(posters[i])
            with col2:
                st.subheader(rec_movies[i]["title"])
                st.markdown(f"Director: {rec_movies[i]['crew'][0]} ")
                st.markdown(f"Cast: {', '.join(rec_movies[i]['cast'])} ")
                st.caption(f"Genre: {', '.join(rec_movies[i]['genres'])}")
                st.caption(f"Year: {rec_movies[i]['release_date']}")
                st.write(f"{rec_movies[i]['overview']}")
                st.text(f"Raiting: {round(float(rec_movies[i]['score']),2)}")
                st.progress(float(rec_movies[i]['score'])/10)
```