```python
import pandas as pd
import numpy as np
movies_df1 = pd.read_csv("C:/Users/LENOVO/Downloads/moviess.csv")
credits_df = pd.read_csv("C:/Users/LENOVO/Downloads/credits.csv")

movies_df = movies_df1.merge(credits_df,on = 'title')

movies_df = movies_df[['id','title','genres']]

import ast

def convert(obj):
    A=[]
    for i in ast.literal_eval(obj):
        A.append(i['name'])
    return A

movies_df['genres'] = movies_df['genres'].apply(convert)

movies_df['genres'] = movies_df['genres'].apply(lambda x:[i.replace(" ","")for i in x])

movies_df['genres'] = movies_df['genres'].apply(lambda x:' '.join(x))

movies_df['genres'] = movies_df['genres'].apply(lambda x:x.lower())

movies_df['title'] = movies_df['title'].apply(lambda x:x.lower())

from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(max_features=5000,stop_words="english")
cv.fit_transform(movies_df['genres']).toarray().shape
vectors = cv.fit_transform(movies_df['genres']).toarray()
```

```python
import nltk

from nltk.stem.porter import PorterStemmer
ps = PorterStemmer()

def stem(text):
    y=[]
    for i in text.split():
        y.append(ps.stem(i))
    return " ".join(y)

movies_df['genres'] = movies_df['genres'].apply(stem)

from sklearn.metrics.pairwise import cosine_similarity

#cosine_similarity(vectors)

similarity = cosine_similarity(vectors)

sorted(list(enumerate(similarity[0])),reverse = True ,key = lambda x:x[1])[1:6]

def recommend(movie):


    movie_index = movies_df[movies_df['title']==movie].index[0]
    dist = similarity[movie_index]
    movies_list=sorted(list(enumerate(dist)),reverse=True,key=lambda x:x[1])[1:6]

    for i in movies_list:
        print(movies_df.iloc[i[0]].title)

a=input("Enter a movie name that you liked: \n ")
recommend(a)
```

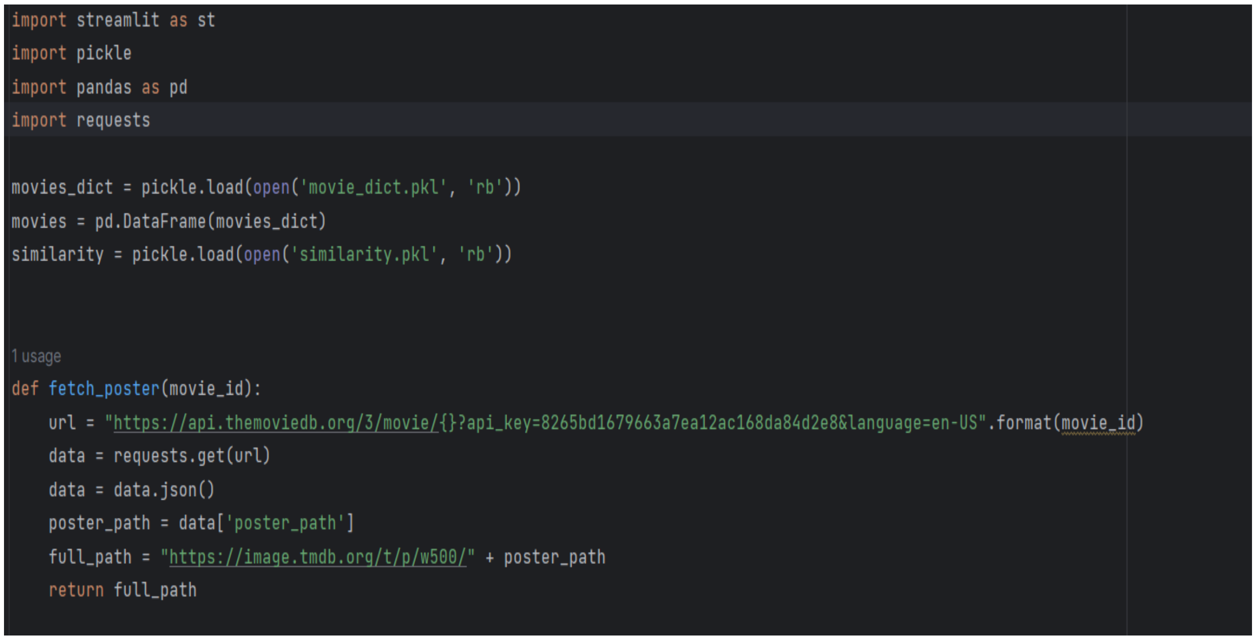
```python
import pickle
```

```python
md=movies_df.to_dict()
```

```python
pickle.dump(md,open('movies.pkl','wb'))
```

```python
pickle.dump(similarity,open('similarity.pkl','wb'))
```

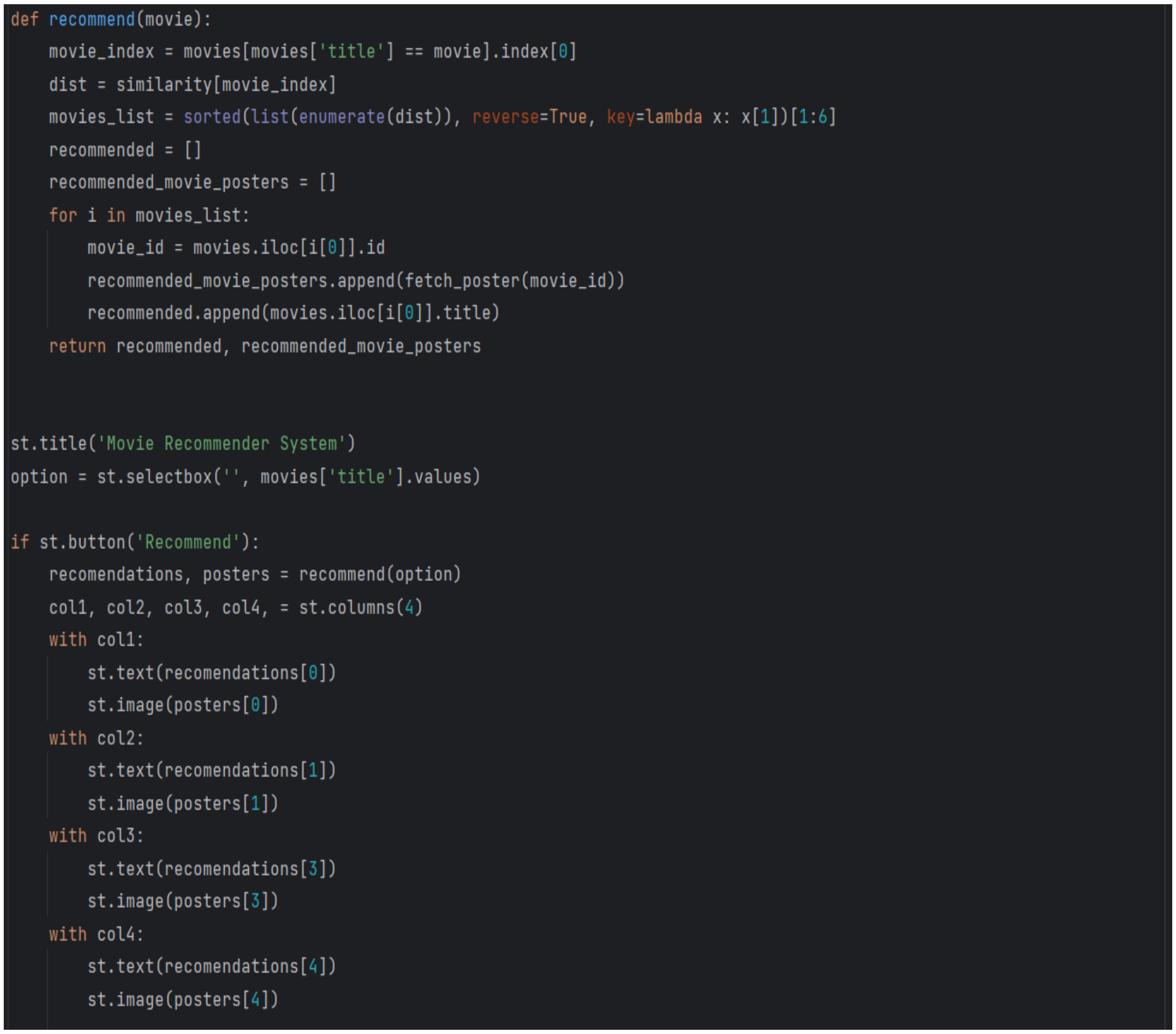
App.py



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

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


1 usage
def fetch_poster(movie_id):
    url = "https://api.themoviedb.org/3/movie/{}?api_key=8265bd1679663a7ea12ac168da84d2e8&language=en-US".format(movie_id)
    data = requests.get(url)
    data = data.json()
    poster_path = data['poster_path']
    full_path = "https://image.tmdb.org/t/p/w500/" + poster_path
    return full_path
```



```python
def recommend(movie):
    movie_index = movies[movies['title'] == movie].index[0]
    dist = similarity[movie_index]
    movies_list = sorted(list(enumerate(dist)), reverse=True, key=lambda x: x[1])[1:6]
    recommended = []
    recommended_movie_posters = []
    for i in movies_list:
        movie_id = movies.iloc[i[0]].id
        recommended_movie_posters.append(fetch_poster(movie_id))
        recommended.append(movies.iloc[i[0]].title)
    return recommended, recommended_movie_posters


st.title('Movie Recommender System')
option = st.selectbox('', movies['title'].values)

if st.button('Recommend'):
    recomendations, posters = recommend(option)
    col1, col2, col3, col4, = st.columns(4)
    with col1:
        st.text(recomendations[0])
        st.image(posters[0])
    with col2:
        st.text(recomendations[1])
        st.image(posters[1])
    with col3:
        st.text(recomendations[3])
        st.image(posters[3])
    with col4:
        st.text(recomendations[4])
        st.image(posters[4])
```