In [ ]:

```python
import streamlit as st
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer
from sklearn.metrics.pairwise import cosine_similarity
import joblib

# Define global variables to store history and chat data
history = []
chat_data = []

def load_data(csv_file):
    data = pd.read_csv(csv_file)
    return data

def load_models():
    count_vectorizer = joblib.load(r"D:\internship\Data science\TASKS\search e
    tfidf_transformer = joblib.load(r"D:\internship\Data science\TASKS\search
    tfidf_matrix = joblib.load(r"D:\internship\Data science\TASKS\search engin
    return count_vectorizer, tfidf_transformer, tfidf_matrix

def retrieve_similar_documents(query, count_vectorizer, tfidf_transformer, tfi
    query_vector = count_vectorizer.transform([query])
    query_tfidf = tfidf_transformer.transform(query_vector)
    similarity_scores = cosine_similarity(query_tfidf, tfidf_matrix)
    top_indices = similarity_scores.argsort()[0][::-1]
    retrieved_documents = [data['clean_file_content'][idx] for idx in top_indi
    retrieved_subtitle_names = [data['name'][idx] for idx in top_indices[:top_
    retrieved_subtitle_nums = [data['num'][idx] for idx in top_indices[:top_n]

    return retrieved_documents, retrieved_subtitle_names, retrieved_subtitle_n

def main():
    global history, chat_data

    # Customizing title and header
    st.title('🎬🔍 FilmFinder')
    st.subheader('🍿In the realm of movies,allow SeekSpot to guide you through

    # Sidebar navigation
    st.sidebar.title('🌟 Navigation')
    if st.sidebar.button('🏠 Home'):
        st.sidebar.text('Go to Home')
        # Clear chat data and history
        history = []
        chat_data = []
    if st.sidebar.button('📜 History'):
        st.sidebar.text('View Search History')
        # Display search history
        st.sidebar.write(history)
    if st.sidebar.button('💾 Export'):
        st.sidebar.text('Export Data')
        # Export chat data to a file
        export_data(chat_data)
    if st.sidebar.button('⚙️ Settings'):
        st.sidebar.text('Change Settings')

    # Search functionality
    query = st.text_input('Enter your query:', '')
    if st.button('🔎 Search'):
        if query:
            # Add query to history
            history.append(query)
            # Retrieve similar documents
            retrieved_documents, retrieved_subtitle_names, retrieved_subtitle_
            st.subheader('📄 Top 5 documents similar to the query:')
            for i, (doc, subtitle_name, subtitle_num) in enumerate(zip(retriev
                st.write(f"**Document {i}:**")
                st.write(f"Subtitle Name: {subtitle_name}")
                st.write(f"Subtitle Number: {subtitle_num}")
                st.write("Summary:", doc)
            # Add search results to chat data
            chat_data.append((query, retrieved_documents, retrieved_subtitle_n

def export_data(data):
    # Export chat data to a file
    pass  # Placeholder for actual export functionality

if __name__ == '__main__':
    data = load_data(r"D:\internship\Data science\TASKS\search engine\data\eng
    count_vectorizer, tfidf_transformer, tfidf_matrix = load_models()
    main()
```