1. **Code with Comments:**

import customtkinter, subprocess

from selenium import webdriver

from selenium.webdriver.common.by import By

from selenium.webdriver.chrome.options import Options

import pandas as pd

from tkinter import ttk

from tkinter import messagebox

# GUI to display top 10 movies with their ratings n a table formaat

def display\_details(df, file\_name):

def close():

display\_root.destroy()

def extract\_more():

display\_root.destroy()

select\_genre\_language()

def open\_file():

subprocess.run(['start', file\_name], shell=True)

display\_root = customtkinter.CTk()

display\_root.title("Movie suggestor")

table = ttk.Treeview(display\_root, columns = ('slno', 'name', 'rating'), show = 'headings')

table.heading('slno', text = 'Sl. No.')

table.heading('name', text = 'Movie Name')

table.heading('rating', text = 'Movie Rating')

table.grid(row = 0, columnspan = 3, sticky="nsew", pady=10, padx=10)

for i, row in df.iterrows(): # itterate through the data frame to get the values aand insert them in table format

table.insert(parent='', index=i, values=(i+1, row['Title'], row['Rating']))

close\_button = customtkinter.CTkButton(display\_root, text="Close", command=close)

close\_button.grid(row=1, column=0, padx=10, pady=10, sticky="nsew")

extract\_more\_button = customtkinter.CTkButton(display\_root, text="New Extraction", command=extract\_more)

extract\_more\_button.grid(row=1, column=1, padx=10, pady=10, sticky="nsew")

view\_more\_button = customtkinter.CTkButton(display\_root, text="View More", command=open\_file)

view\_more\_button.grid(row=1, column=2, padx=10, pady=10, sticky="nsew")

display\_root.mainloop()

# GUI to select genre and language

def select\_genre\_language():

root = customtkinter.CTk()

root.title("Movie suggestor")

root.attributes('-topmost', True)

def close():

root.destroy()

def validity(genre\_option, language\_option): # varify if no genre is selected and give a error message

if genre\_option == '':

messagebox.showinfo('Genre Validation', 'Please select a genre and try again.')

else:

root.destroy()

scrape\_movies(genre\_option, language\_option)

genre\_frame = customtkinter.CTkFrame(root)

genre\_frame.grid(row = 0, column = 0, sticky="nsew", pady=10, padx=10)

language\_frame = customtkinter.CTkFrame(root)

language\_frame.grid(row = 0, column = 1, sticky="nsew", pady=10, padx=10)

def open\_input\_genre():

genre\_dialog = customtkinter.CTkInputDialog(text="Type a Genre:", title="Enter Genre")

genre\_option.set(genre\_dialog.get\_input())

def open\_input\_language():

language\_dialog = customtkinter.CTkInputDialog(text="Type a Language:", title="Enter Language")

language\_option.set(language\_dialog.get\_input())

genre\_lebel = customtkinter.CTkLabel(genre\_frame, text="Select Genre:")

genre\_lebel.grid(row=0, column=0, pady=10, padx=10, sticky="nsew")

genre\_option = customtkinter.CTkComboBox(genre\_frame, values=["action", "comedy", "adventure", "thriller", "horror", "animation", "sci-fi", "drama", "mystery"])

genre\_option.set("")

genre\_option.grid(row=1, column=0, pady=10, padx=10, sticky="nsew")

enter\_genre\_btn = customtkinter.CTkButton(genre\_frame, text="Enter a genre", command=open\_input\_genre)

enter\_genre\_btn.grid(row=2, column=0, pady=10, padx=10, sticky="nsew")

language\_lebel = customtkinter.CTkLabel(language\_frame, text="Select Language:")

language\_lebel.grid(row=0, column=0, pady=10, padx=10, sticky="nsew")

language\_option = customtkinter.CTkComboBox(language\_frame, values=["English", "Hindi", "Telegu", "Malayalam", "Oriya", "Tamil", "Kannada"])

language\_option.set("")

language\_option.grid(row=1, column=0, pady=10, padx=10, sticky="nsew")

enter\_language\_btn = customtkinter.CTkButton(language\_frame, text="Enter a language", command=open\_input\_language)

enter\_language\_btn.grid(row=2, column=0, pady=10, padx=10, sticky="nsew")

ext\_button = customtkinter.CTkButton(root, text="Extract", command=lambda:validity(genre\_option.get(), language\_option.get()))

ext\_button.grid(row=1, column=0, padx=10, pady=10, sticky="nsew")

close\_button = customtkinter.CTkButton(root, text="Close", command=close)

close\_button.grid(row=1, column=1, padx=10, pady=10, sticky="nsew")

root.mainloop()

# Scraping data from IMDB using selenium

def scrape\_movies(genre, language):

options = Options()

options = webdriver.ChromeOptions()

driver = webdriver.Chrome(options=options)

# dict to get the language tag for url

language\_tags = {

'Hindi': 'hi',

'English': 'en',

"Telegu":'te',

"Malayalam":'ml',

"Oriya":'or',

"Tamil":'ta',

"Kannada":'kn',

}

if language == '':

url = f'https://www.imdb.com/search/title/?genres={genre}'

else:

lang\_tag = language\_tags [language]

url = f'https://www.imdb.com/search/title/?genres={genre}&languages={lang\_tag}'

titles = []

years = []

ratings = []

description = []

driver.get(url)

try:

movies\_list = driver.find\_elements(By.XPATH, '/html/body/div[2]/main/div[2]/div[3]/section/section/div/section/section/div[2]/div/section/div[2]/div[2]/ul')[0]

movies\_number = movies\_list.find\_elements(By.TAG\_NAME, 'li')

for i in range ((len(movies\_number))):

try:

title\_xpath = f'/html/body/div[2]/main/div[2]/div[3]/section/section/div/section/section/div[2]/div/section/div[2]/div[2]/ul/li[{i+1}]/div/div/div/div[1]/div[2]/div[1]/a/h3'

movie\_title = driver.find\_element(By.XPATH, title\_xpath).text

titles.append(movie\_title.split('. ')[1])

except:

titles.append("Cannot find title")

try:

year\_of\_rel\_xpath = f'/html/body/div[2]/main/div[2]/div[3]/section/section/div/section/section/div[2]/div/section/div[2]/div[2]/ul/li[{i+1}]/div/div/div/div[1]/div[2]/div[2]/span[1]'

yr\_of\_rel = driver.find\_element(By.XPATH, year\_of\_rel\_xpath).text.split('–')[0]

years.append(yr\_of\_rel)

except:

years.append("Cannot find year of release")

try:

rating\_xpath = f'/html/body/div[2]/main/div[2]/div[3]/section/section/div/section/section/div[2]/div/section/div[2]/div[2]/ul/li[{i+1}]/div/div/div/div[1]/div[2]/span/div/span'

rating = driver.find\_element(By.XPATH, rating\_xpath).text

ratings.append(rating.split()[0] + " ★ " + rating.split()[1])

except:

ratings.append("Cannot find rating")

try:

movie\_description\_xpath = f'/html/body/div[2]/main/div[2]/div[3]/section/section/div/section/section/div[2]/div/section/div[2]/div[2]/ul/li[{i+1}]/div/div/div/div[2]/div/div'

movie\_description = driver.find\_element(By.XPATH, movie\_description\_xpath).text

description.append(movie\_description)

except:

description.append("Cannot find description")

except Exception as e:

print(f"An error occurred while scraping: {e}")

finally:

driver.quit()

data = {'Title': titles, 'Year': years, 'Rating': ratings, 'Description': description}

df = pd.DataFrame(data) # converting extracted data into daata frame

if language == '':

file\_name = f'movies\_{genre}.xlsx'

else:

file\_name = f'movies\_{genre}\_{language}.xlsx'

df.to\_excel(file\_name, index=False) # saaving output as excel using pd

display\_details(df.loc[:9, ['Title', 'Rating']], file\_name)

print(f"Scraped {len(titles)} movies for genre {genre}")

if \_\_name\_\_ == "\_\_main\_\_":

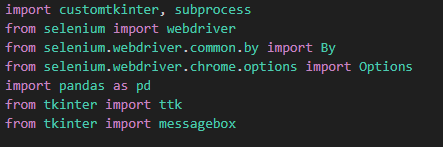
customtkinter.set\_appearance\_mode("Light") # Modes: "System" (standard), "Dark", "Light"

customtkinter.set\_default\_color\_theme("dark-blue") # Themes: "blue" (standard), "green", "dark-blue"

select\_genre\_language()

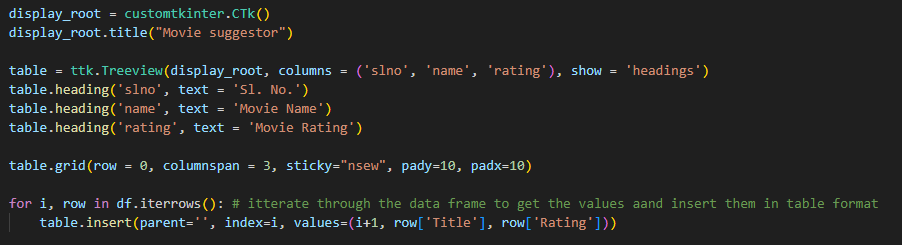
1. **Code Explanation Step by step:**

**Importing Required Libraries and Modules**

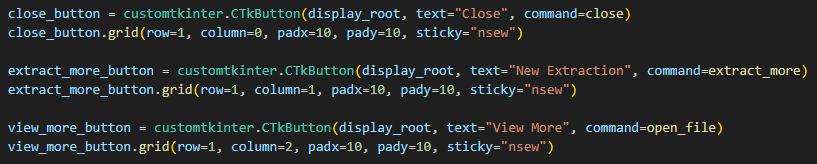
****

* customtkinter: Custom libraries for creating GUIs in Python.
* subprocess: Used here to open files using system commands.
* selenium: Required for automating web interactions, used here to scrape IMDb.
* pandas: Essential for data manipulation, specifically handling data frames.
* tkinter and ttk: Standard libraries for creating GUIs in Python.

**display\_details(df, file\_name) Function**

****

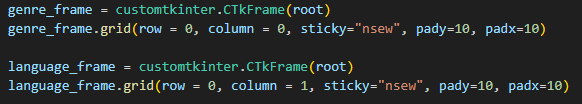
* Function Purpose: Displays a GUI window (display\_root) showing the top 10 movies with their ratings in a table format.
* Components:
  + Treeview (table): Displays the data in a tabular format with columns for serial number (slno), movie name (name), and movie rating (rating).
  + Buttons:



* + - Close Button: Destroys the current window (display\_root).
    - New Extraction Button: Closes the current window and opens the genre and language selection window (select\_genre\_language()).
    - View More Button: Opens the saved Excel file using the default system application.

**select\_genre\_language() Function**

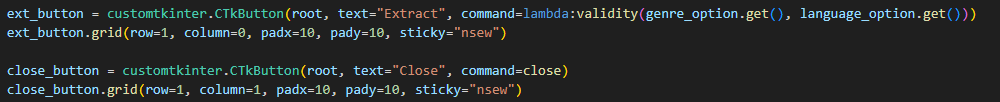
* Function Purpose: Displays a GUI window (root) for selecting a movie genre and language.
* Components:



* + Frames (genre\_frame and language\_frame): Organize widgets for genre selection and language selection.

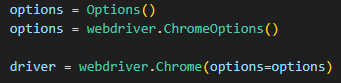


* + Labels (genre\_lebel and language\_lebel): Display text labels for indicating the purpose of dropdowns.
  + Combo Boxes (genre\_option and language\_option): Dropdown menus for selecting genre and language.
  + Buttons:
    - Enter Genre Button (enter\_genre\_btn) and Enter Language Button (enter\_language\_btn): Allow users to enter custom genres and languages.

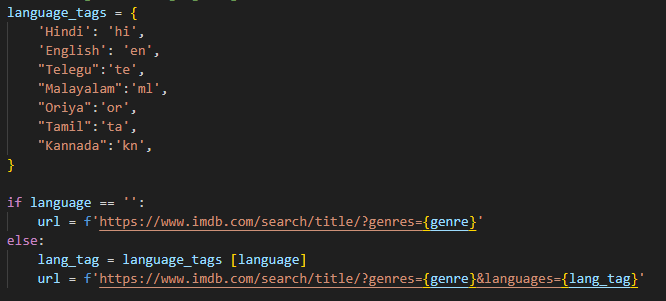


* + - Extract Button (ext\_button): Initiates data extraction based on selected genre and language.
    - Close Button (close\_button): Closes the current window (root).

**scrape\_movies(genre, language) Function**



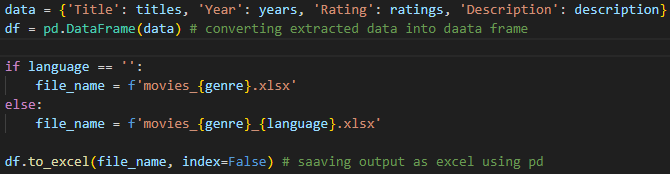
* Selenium Setup: Initializes Selenium WebDriver with Chrome options.



* Language Tags: Dictionary to map languages to their respective IMDb URL tags.
* URL Construction: Constructs the URL based on selected genre and language.



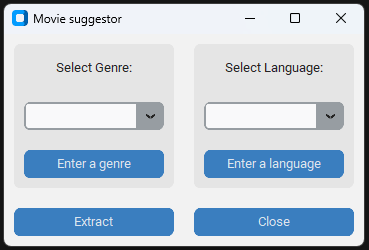
* Web Scraping:
  + Opens the constructed URL.
  + Extracts movie titles, years, ratings, and descriptions using XPath.
  + Handles missing data with try-except blocks.
* Data Handling:



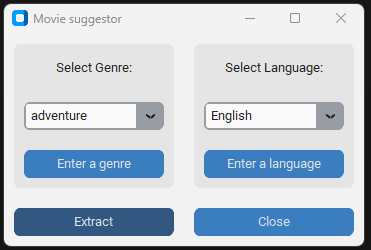
* Stores extracted data in a Pandas DataFrame.
* Saves the DataFrame to an Excel file.
* Calls display\_details() to show the top 10 movies.

1. **Output**

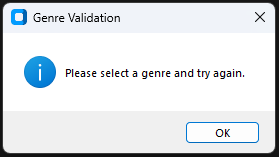
Initial dialog box pops up after running the code



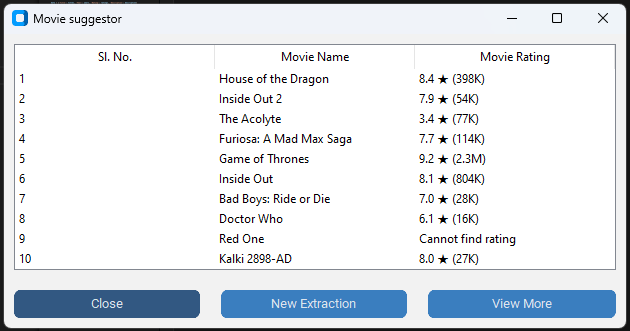
Select the genre and language from the dropdowns. You can enter them manually by clicking on the “enter a genre” and “enter a language” buttons respectively.



Genre selection is mandatory as proceeding without it will show error



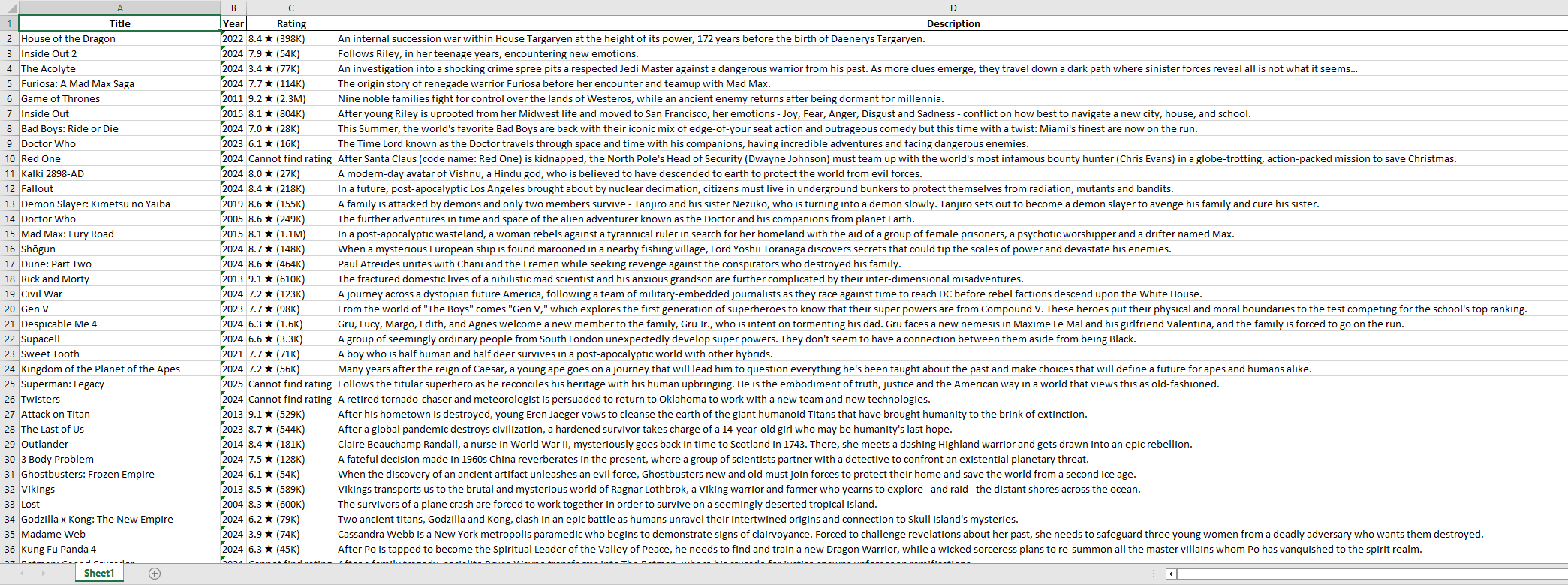
The first out put contains a tabular view of the top 10 movies from the selected genre.



The “New Extraction” button will close the output table and open the extraction popup to select genre and language.

The “View more” button will open the excel sheet in which the data is stored.

**Excel Output:**

****