using System;

using System.Net.Http;

using System.Threading.Tasks;

using System.Xml.Linq;

using Newtonsoft.Json.Linq;

namespace Lab5FinalTest;

class Program

{

static async Task Main()

{

while (true)

{

Console.WriteLine("Star Wars Info Viewer");

Console.WriteLine("1. View character information");

Console.WriteLine("2. films information");

Console.WriteLine("3. View planets information");

Console.WriteLine("4. View multiple characters");

Console.WriteLine("5. Exit");

Console.Write("Enter your choice: ");

string choice = Console.ReadLine();

switch (choice)

{

case "1":

await ViewCharacterInformation();

break;

case "2":

await ViewEpisodeInformation();

break;

case "3":

await ViewLocationInformation();

break;

case "4":

await ViewMultipleCharacters();

break;

case "5":

Console.WriteLine("Goodbye!");

return;

default:

Console.WriteLine("Invalid choice. Please try again.");

Console.WriteLine();

break;

}

}

}

static async Task ViewMultipleCharacters()

{

Console.WriteLine("Enter character IDs with white spaces:");

string input = Console.ReadLine();

List<int> characterIds = input.Split(' ').Select(int.Parse).ToList();

using (HttpClient client = new HttpClient())

{

try

{

foreach (int peopleId in characterIds)

{

HttpResponseMessage response = await client.GetAsync($" https://swapi.dev/api/people/{peopleId}");

if (response.IsSuccessStatusCode)

{

string responseBody = await response.Content.ReadAsStringAsync();

JObject characterData = JObject.Parse(responseBody);

string name = characterData["name"].ToString();

string height = characterData["height"].ToString();

string mass = characterData["mass"].ToString();

//string hairColor = characterData["hair\_Color"].ToString();

//string skinColor = characterData["skin color"].ToString();

// string eyeColor = characterData["eyeColor"].ToString();

string birth\_year = characterData["birth\_year"].ToString();

string gender = characterData["gender"].ToString();

string homeworld = characterData["homeworld"].ToString();

string films = characterData["films"].ToString();

string species = characterData["species"].ToString();

string vehichles = characterData["vehicles"].ToString();

string starShips = characterData["starships"].ToString();

string created = characterData["created"].ToString();

string edited = characterData["edited"].ToString();

string url = characterData["url"].ToString();

Console.WriteLine($"Name: {name}");

Console.WriteLine($"Species: {species}");

Console.WriteLine($"birth year: {birth\_year}");

//Console.WriteLine($"Status: {eyeColor}");

Console.WriteLine($"films: {films}");

Console.WriteLine($"gender: {gender}");

//Console.WriteLine($"Status: {hairColor}");

Console.WriteLine($"homeworld: {homeworld}");

Console.WriteLine($"mass: {mass}");

Console.WriteLine($"height: {height}");

//Console.WriteLine($"Status: {skinColor}");

Console.WriteLine($"vehicles: {vehichles}");

Console.WriteLine($"StarShips: {starShips}");

Console.WriteLine($"created: {created}");

Console.WriteLine($"edited: {edited}");

Console.WriteLine($"url: {url}");

Console.WriteLine("----------");

}

else

{

Console.WriteLine($"Error for Character ID {peopleId}: {response.StatusCode} - {response.ReasonPhrase}");

}

}

}

catch (Exception ex)

{

Console.WriteLine($"Error: {ex.Message}");

}

}

}

static async Task ViewCharacterInformation()

{

string peopleId;

Console.Write("Enter character ID: ");

if (!int.TryParse(Console.ReadLine(), out int nameId) || nameId <= 0)

{

Console.WriteLine("Invalid character ID. Please enter a positive integer.");

return;

}

using (HttpClient client = new HttpClient())

{

try

{

HttpResponseMessage response = await client.GetAsync($"https://swapi.dev/api/people/{nameId}");

if (response.IsSuccessStatusCode)

{

string responseBody = await response.Content.ReadAsStringAsync();

JObject characterData = JObject.Parse(responseBody);

string name = characterData["name"].ToString();

string height = characterData["height"].ToString();

string mass = characterData["mass"].ToString();

//string hairColor = characterData["hair\_Color"].ToString();

//string skinColor = characterData["skin color"].ToString();

// string eyeColor = characterData["eyeColor"].ToString();

string birth\_year = characterData["birth\_year"].ToString();

string gender = characterData["gender"].ToString();

string homeworld = characterData["homeworld"].ToString();

string films = characterData["films"].ToString();

string species = characterData["species"].ToString();

string vehichles = characterData["vehicles"].ToString();

string starShips = characterData["starships"].ToString();

string created = characterData["created"].ToString();

string edited = characterData["edited"].ToString();

string url = characterData["url"].ToString();

Console.WriteLine($"Character ID: {nameId}");

Console.WriteLine($"Name: {name}");

Console.WriteLine($"Species: {species}");

Console.WriteLine($"birth year: {birth\_year}");

//Console.WriteLine($"Status: {eyeColor}");

Console.WriteLine($"films: {films}");

Console.WriteLine($"gender: {gender}");

//Console.WriteLine($"Status: {hairColor}");

Console.WriteLine($"homeworld: {homeworld}");

Console.WriteLine($"mass: {mass}");

Console.WriteLine($"height: {height}");

//Console.WriteLine($"Status: {skinColor}");

Console.WriteLine($"vehicles: {vehichles}");

Console.WriteLine($"StarShips: {starShips}");

Console.WriteLine($"created: {created}");

Console.WriteLine($"edited: {edited}");

Console.WriteLine($"url: {url}");

Console.WriteLine("----------");

}

else

{

Console.WriteLine($"Error: {response.StatusCode} - {response.ReasonPhrase}");

}

}

catch (Exception ex)

{

Console.WriteLine($"Error: {ex.Message}");

}

}

}

static async Task ViewEpisodeInformation()

{

Console.Write("Enter episode ID: ");

if (!int.TryParse(Console.ReadLine(), out int filmId) || filmId <= 0)

{

Console.WriteLine("Invalid episode ID. Please enter a positive integer.");

return;

}

using (HttpClient client = new HttpClient())

{

try

{

HttpResponseMessage response = await client.GetAsync($"https://swapi.dev/api/films/{filmId}");

if (response.IsSuccessStatusCode)

{

string responseBody = await response.Content.ReadAsStringAsync();

JObject episodeData = JObject.Parse(responseBody);

string title = episodeData["title"].ToString();

string episode\_id = episodeData["episode\_id"].ToString();

string director = episodeData["director"].ToString();

string producer = episodeData["producer"].ToString();

string release\_date = episodeData["release\_date"].ToString();

string characters = episodeData["characters"].ToString();

string planets = episodeData["planets"].ToString();

string starships = episodeData["starships"].ToString();

string vehicles = episodeData["vehicles"].ToString();

string created = episodeData["created"].ToString();

string edited = episodeData["edited"].ToString();

string url = episodeData["url"].ToString();

Console.WriteLine("----------");

Console.WriteLine("Episode Information:");

Console.WriteLine($"Name: {episodeData["title"]}");

Console.WriteLine($"Director: {episodeData["director"]}");

Console.WriteLine($"Episode Id: {episodeData["episode\_id"]}");

Console.WriteLine($"diretor: {episodeData["director"]}");

Console.WriteLine($"producer: {episodeData["producer"]}");

Console.WriteLine($"release\_date: {episodeData["release\_date"]}");

Console.WriteLine($"characters: {episodeData["characters"]}");

Console.WriteLine($"planets: {episodeData["planets"]}");

Console.WriteLine($"starships: {episodeData["starships"]}");

Console.WriteLine($"vehicles: {episodeData["vehicles"]}");

Console.WriteLine($"created: {episodeData["created"]}");

Console.WriteLine($"edited: {episodeData["edited"]}");

Console.WriteLine($"Name: {episodeData["url"]}");

Console.WriteLine("----------");

}

else

{

Console.WriteLine($"Error: {response.StatusCode} - {response.ReasonPhrase}");

}

}

catch (Exception ex)

{

Console.WriteLine($"Error: {ex.Message}");

}

}

}

static async Task ViewLocationInformation()

{

Console.Write("Enter location ID: ");

if (!int.TryParse(Console.ReadLine(), out int planetId) || planetId <= 0)

{

Console.WriteLine("Invalid location ID. Please enter a positive integer.");

return;

}

using (HttpClient client = new HttpClient())

{

try

{

HttpResponseMessage response = await client.GetAsync($" https://swapi.dev/api/planets/{planetId}");

if (response.IsSuccessStatusCode)

{

string responseBody = await response.Content.ReadAsStringAsync();

JObject locationData = JObject.Parse(responseBody);

string name = locationData["name"].ToString();

// string roational\_period = locationData["Rotaional\_period"].ToString();

string orbital\_period = locationData["orbital\_period"].ToString();

string diameter = locationData["diameter"].ToString();

string climate = locationData["climate"].ToString();

string gravity = locationData["gravity"].ToString();

string terrain = locationData["terrain"].ToString();

string surface\_water = locationData["surface\_water"].ToString();

string population = locationData["population"].ToString();

string residents = locationData["residents"].ToString();

string films = locationData["films"].ToString();

string created = locationData["created"].ToString();

string edited = locationData["edited"].ToString();

string url = locationData["url"].ToString();

Console.WriteLine("----------");

Console.WriteLine("Location Information:");

Console.WriteLine($"Name: {locationData["name"]}");

// Console.WriteLine($"Rotaional\_period: {locationData["Rotational\_period"]}");

Console.WriteLine($"orbital\_Period: {locationData["orbital\_Period"]}");

Console.WriteLine($"diameter: {locationData["diameter"]}");

Console.WriteLine($"climate: {locationData["climate"]}");

Console.WriteLine($"gravity: {locationData["gravity"]}");

Console.WriteLine($"terrain: {locationData["terrain"]}");

Console.WriteLine($"surface\_water: {locationData["surface\_water"]}");

Console.WriteLine($"population: {locationData["population"]}");

Console.WriteLine($"residents: {locationData["residents"]}");

Console.WriteLine($"films: {locationData["films"]}");

Console.WriteLine($"created: {locationData["created"]}");

Console.WriteLine($"edited: {locationData["edited"]}");

Console.WriteLine($"url: {locationData["url"]}");

Console.WriteLine("----------");

}

else

{

Console.WriteLine($"Error: {response.StatusCode} - {response.ReasonPhrase}");

}

}

catch (Exception ex)

{

Console.WriteLine($"Error: {ex.Message}");

}

}

}

}