EJERCICIO 1 (PHP – 50% de la nota)

Aplicación CLI de Gestión de Tareas con Persistencia en JSON

Desarrolla una aplicación en **PHP CLI** que permita gestionar una lista de tareas desde la consola.
Debe funcionar **únicamente desde terminal**, sin HTML, sin frameworks.

Requisitos funcionales

La aplicación debe permitir:

1. Agregar tarea

php gestor.php add "Título" "Descripción" "2025-12-01"

- Debe generar automáticamente un **ID único**.
- Debe validar la fecha (no puede ser anterior al día actual).

2. Eliminar tarea

php gestor.php delete <id>

3. Editar tarea

php gestor.php edit <id> "Nuevo título" "Nueva descripción" "Nueva fecha"

4. Marcar como completada

php gestor.php complete <id>

5. Listar tareas

php gestor.php list
php gestor.php list --completed
php gestor.php list --pending

Persistencia de datos

- Las tareas deben guardarse en tareas.json.

- Si el archivo no existe, debe crearse automáticamente.

Formato del JSON obligatorio

```
{
   "tasks": [
      {
         "id": 1,
         "title": "Comprar pan",
         "description": "Pan integral",
         "due_date": "2025-12-02",
         "completed": false
      }
   ]
}
```

Validaciones obligatorias

- ID inexistente → mostrar mensaje: *"Error: Tarea no encontrada."* ●

  Fecha inválida → *"Error: La fecha no es válida."*

README.md obligatorio

Debe incluir:

- Cómo ejecutar cada comando
- Ejemplos
- Explicación del JSON
- Requisitos del sistema

EJERCICIO 2 (C# – 50% de la nota)

Código base.

```
using System.Net.Http;
using System.Text.Json;
```

```
HttpClient client = new HttpClient();

async Task<JsonDocument?> GetCountryAsync(string name) {
    try
    {
        var url = $"https://restcountries.com/v3.1/name/{name}"; var response =
        await client.GetAsync(url);

        if (!response.IsSuccessStatusCode)
            return null;

        var content = await response.Content.ReadAsStringAsync(); return
        JsonDocument.Parse(content);
    }
    catch
    {
        return null;
    }
}
```

ENUNCIADO

Crear un programa que muestre un menú en consola:

1. Buscar país por nombre
2. Listar mis países guardados
3. Salir

Cuando el usuario elija **1**, pedir el nombre del país y llamar a la API:

Endpoint:

https://restcountries.com/v3.1/name/{nombre}

Se debe mostrar **mínimo**:

- Nombre común → name.common
- Capital → capital[0]
- Región → region
- Población → population

Ejemplo de salida:

Nombre: Spain

Capital: Madrid
Región: Europe
Población: 47351567

Si el país no existe:

País no encontrado.

Después de buscar un país válido debéis preguntar:

¿Quieres guardarlo en tu lista? (s/n)

Debe guardarse en una List<Country> (Debéis crear esta clase con las propiedades necesarias). Enseñar los países guardados de la siguiente manera:

=== MIS PAÍSES ===
1. Spain (Madrid) - Europe - 47351567 hab.
2. France (Paris) - Europe - 65273511 hab.

Si no hay países guardados:

No tienes países guardados todavía.

Ejercicio extra (1 punto adicional)

Agregar una opción al menú:

4. Guardar lista en archivo JSON

Y guardar el contenido de la lista en:

mis_paises.json.

Entrega

Un **link a GitHub**, corregiré desde la última versión realizada durante el examen.