Práctica semana 9

Caso 1: “Gestión de Libros – Biblioteca Campus”

Objetivo General

Desarrollar una aplicación web en **ASP.NET Core MVC** que permita registrar y visualizar información de libros en una biblioteca universitaria, aplicando el patrón MVC, validaciones de datos y separación de responsabilidades, almacenando los datos en memoria.

Historia del Cliente: “Biblioteca Campus”

La biblioteca “Campus U” necesita una pequeña aplicación web para registrar los libros que se agregan a su catálogo.

El sistema debe permitir ingresar los siguientes datos de cada libro:

- Título
- Autor
- Categoría (Programación, Redes, Bases de Datos, IA, Otro)
- Año de publicación
- Número de páginas
- Código interno (ej. LIB-001)
- Disponible (Sí/No – tipo booleano)

Al enviar el formulario, la información debe almacenarse en una **lista temporal en memoria** y mostrarse en una **tabla** en la misma página (o en una vista de listado).

Requisitos de Validación

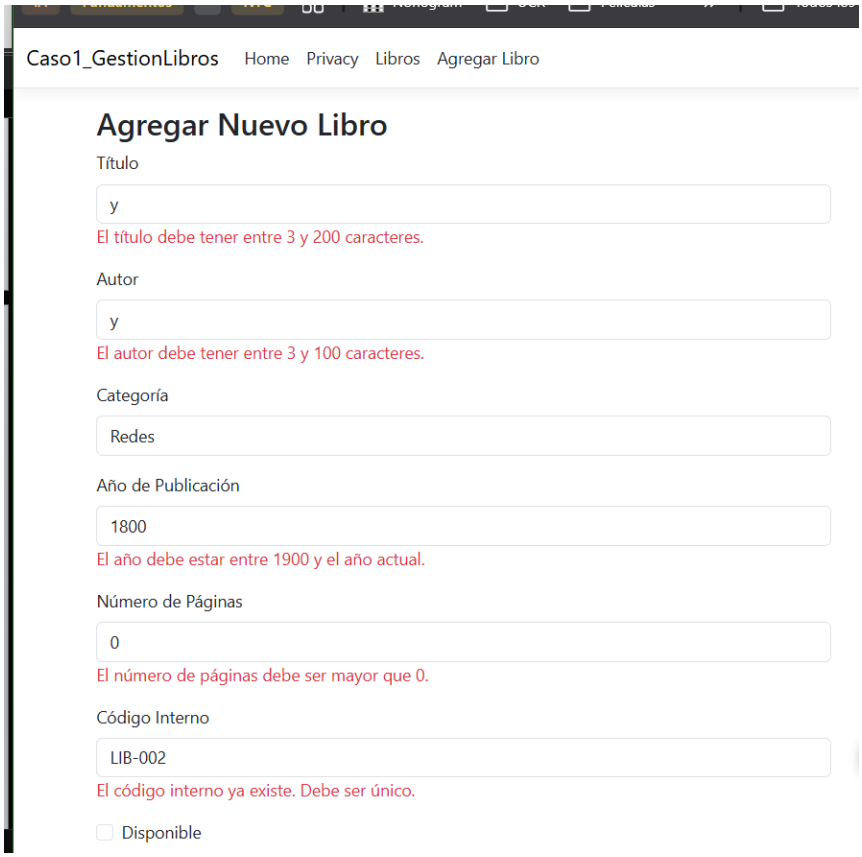
- **Título**
  - Obligatorio.
  - Longitud mínima de 3 caracteres.

- **Autor**
  - Obligatorio.
  - Longitud mínima de 3 caracteres.
- **Categoría**
  - Campo obligatorio (no se permite la opción por defecto “Seleccione...”).
- **Año de publicación**
  - Obligatorio.
  - Debe estar entre **1900** y el año actual.
  - No puede ser un valor futuro.
- **Número de páginas**
  - Obligatorio.
  - Debe ser un número entero **mayor que 0**.
  - No se permiten valores negativos.
- **Código interno**
  - Obligatorio.
  - Debe seguir un formato, por ejemplo: LIB-### (usar [RegularExpression]).
  - No debe repetirse dentro de la lista (validación manual en el repositorio/controlador).
- **Disponible**
  - Obligatorio (true/false).

Entregables

Cada grupo deberá entregar:

- Proyecto **ASP.NET Core MVC** completo en GitHub.
- Evidencias del uso de GIT:
  - Commits frecuentes y con mensajes claros.
  - Uso de al menos una rama adicional (por ejemplo: feature/registro-libros).
- Capturas de pantalla:
  - Formulario de registro de libros.
  - Tabla/listado de libros.
  - Mensajes de validación.



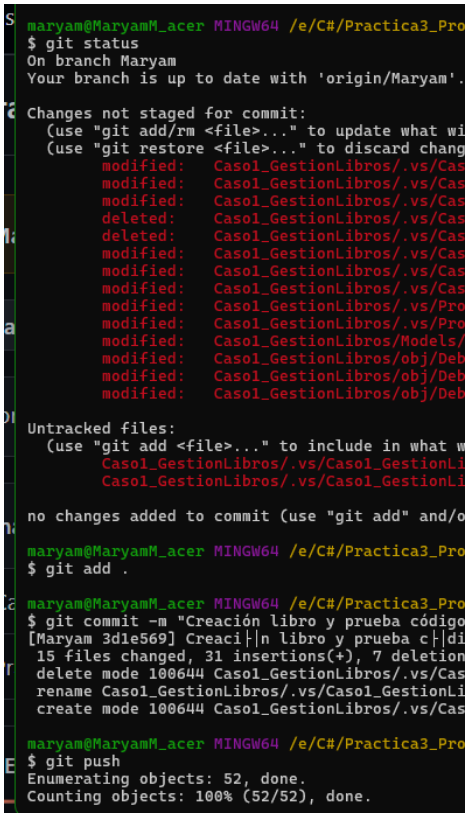
Pasos de Desarrollo

Fase 1 – Creación del Proyecto

- Crear un nuevo proyecto **ASP.NET Core MVC** en Visual Studio o VS Code.
- Inicializar un repositorio **Git** y hacer el primer commit.

Fase 2 – Creación del Modelo

- Crear la clase Libro.cs en la carpeta **Models** con las propiedades: Titulo, Autor, Categoria, AnioPublicacion, NumeroPaginas, CodigoInterno, Disponible.
- Agregar anotaciones de validación:
    - [Required], [StringLength], [Range], [RegularExpression], etc.



Fase 3 – Creación del Repositorio

- Crear la clase LibroRepository.cs en la carpeta **Data**.
- Mantener una **lista en memoria**:
    - private static List<Libro> _libros = new List<Libro>();
- Métodos sugeridos:
    - AgregarLibro(Libro libro)
    - ObtenerLibros()

.vs	Fase 3 – Creación del Repositorio y cambio en la clase Libro
Controllers	Fase1: Creación del Proyecto
Data	Fase 3 – Creación del Repositorio y cambio en la clase Libro
Models	Fase 3 – Creación del Repositorio y cambio en la clase Libro
Properties	Fase1: Creación del Proyecto
Views	Fase1: Creación del Proyecto
obj	Fase 3 – Creación del Repositorio y cambio en la clase Libro
wwwroot	Fase1: Creación del Proyecto

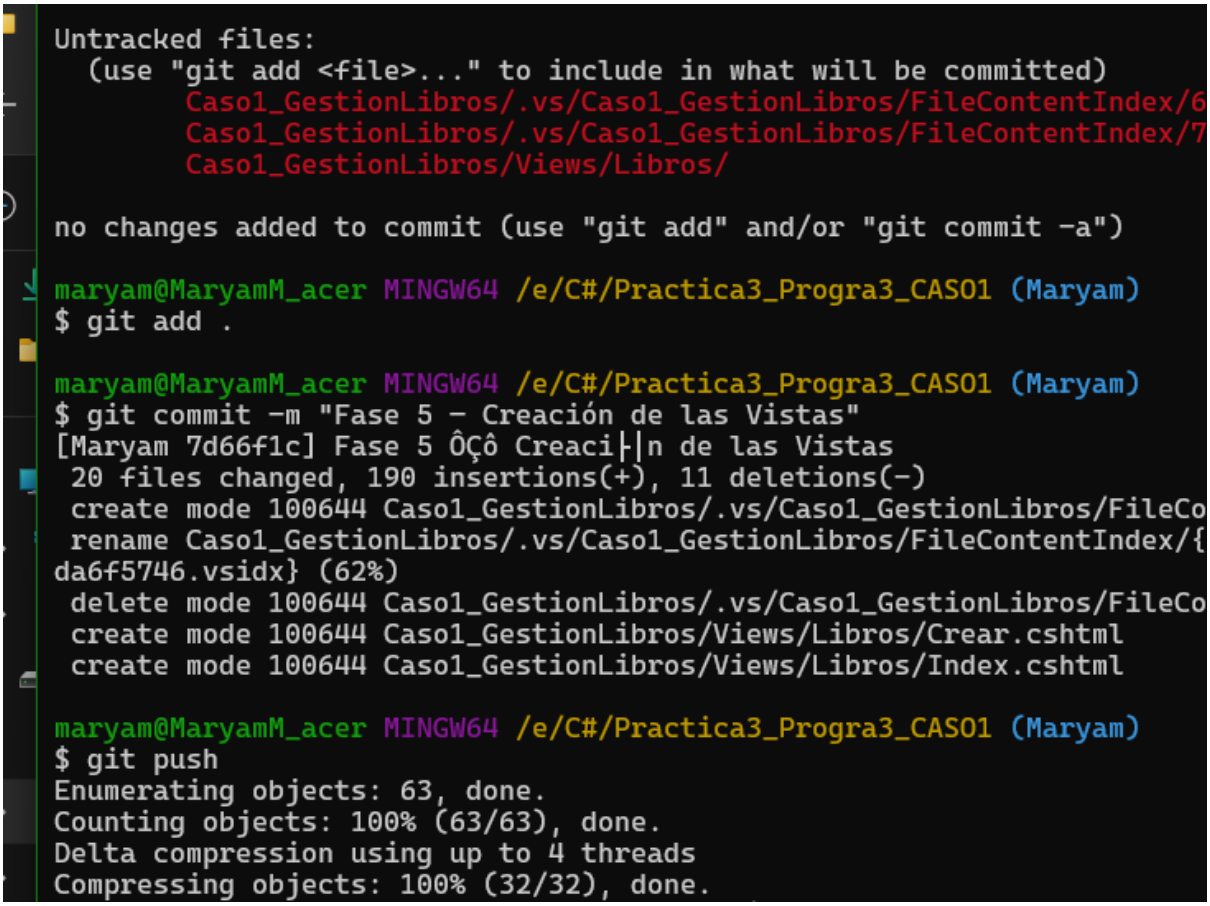
Fase 4 – Creación del Controlador

- Crear LibroController.cs en la carpeta **Controllers**.
- Métodos sugeridos:
  - Index() → Mostrar listado de libros.
  - Crear() (GET) → Mostrar formulario.
  - Crear(Libro libro) (POST) → Validar y guardar en el repositorio; si hay errores, regresar a la vista con mensajes.

```
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Caso1_GestionLibros/.vs/Caso1_GestionLibros/FileContentIndex/7ab67914-fc49-4287-becd-13e0397e
        Caso1_GestionLibros/.vs/Caso1_GestionLibros/FileContentIndex/d37e6012-9217-40cd-82a9-3e88ea4a
        Caso1_GestionLibros/.vs/Caso1_GestionLibros/FileContentIndex/fc7b0d3e-0cfe-4859-aed6-3920ef5f
        Caso1_GestionLibros/Controllers/LibroController.cs

no changes added to commit (use "git add" and/or "git commit -a")

maryam@MaryamM_acer MINGW64 /e/C#/Practica3_Progra3_CASO1 (Maryam)
$ git add .

maryam@MaryamM_acer MINGW64 /e/C#/Practica3_Progra3_CASO1 (Maryam)
$ git commit -m "Fase 4 – Creación del Controlador"
[Maryam c3778f0] Fase 4 ÔÇô Creaci├│n del Controlador
 19 files changed, 75 insertions(+), 54 deletions(-)
 delete mode 100644 Caso1_GestionLibros/.vs/Caso1_GestionLibros/FileContentIndex/39d29cc0-06ea-41c1-a
 create mode 100644 Caso1_GestionLibros/.vs/Caso1_GestionLibros/FileContentIndex/7ab67914-fc49-4287-b
 rename Caso1_GestionLibros/.vs/Caso1_GestionLibros/FileContentIndex/{70c81b19-892d-492a-8535-15565fa
(62%)
 create mode 100644 Caso1_GestionLibros/.vs/Caso1_GestionLibros/FileContentIndex/fc7b0d3e-0cfe-4859-a
 create mode 100644 Caso1_GestionLibros/Controllers/LibroController.cs

maryam@MaryamM_acer MINGW64 /e/C#/Practica3_Progra3_CASO1 (Maryam)
$ git push
Enumerating objects: 60, done.
Counting objects: 100% (59/59), done.
Delta compression using up to 4 threads
Compressing objects: 100% (30/30), done.
Writing objects: 100% (33/33), 490.30 KiB | 4.30 MiB/s, done.
Total 33 (delta 14), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (14/14), completed with 13 local objects.
To https://github.com/maryammf2/Practica3_Progra3_CASO1.git
   df44eef..c3778f0  Maryam -> Maryam
```

Fase 5 – Creación de las Vistas

- En Views/Libro crear:
  - Index.cshtml → Tabla con el listado de libros.
  - Crear.cshtml → Formulario para registrar un nuevo libro.



Fase 6 – Personalización de las Páginas

- Aplicar estilos usando **Bootstrap**.
- Mostrar mensajes de validación y un mensaje de confirmación al registrar un libro.
- Incluir navegación básica:
  - Enlace desde Index hacia Crear y viceversa.

Caso 2: “Registro de Inscripciones – Talleres de Tecnología”

Objetivo General

Desarrollar una aplicación web en **ASP.NET Core MVC** que permita registrar y visualizar inscripciones a talleres de tecnología, aplicando el patrón MVC, validaciones de datos y buenas prácticas de separación de responsabilidades.

Historia del Cliente: “TechWorkshops”

La empresa “TechWorkshops” organiza talleres cortos de programación y tecnología para estudiantes.
Necesitan una aplicación web donde puedan registrar a los participantes de cada taller.

El sistema debe permitir ingresar los siguientes datos de cada inscripción:

- Nombre del participante
- Apellidos
- Correo electrónico
- Teléfono
- Taller seleccionado (C#, Python, Web, Bases de Datos, Otro)
- Nivel de experiencia (Principiante, Intermedio, Avanzado)
- Fecha del taller
- Acepta términos y condiciones (checkbox)

La información debe almacenarse en una **lista temporal en memoria** y mostrarse en una **tabla** en la página de listado.

Requisitos de Validación

- **Nombre del participante**
  - Obligatorio.
  - Longitud mínima de 2 caracteres.

- **Apellidos**
  - Obligatorio.
  - Longitud mínima de 2 caracteres.
- **Correo electrónico**
  - Obligatorio.
  - Formato válido de correo ([EmailAddress]).
- **Teléfono**
  - Obligatorio.
  - Debe seguir un formato mayor que 8 dígitos.
- **Taller seleccionado**
  - Obligatorio.
  - No se permite la opción por defecto “Seleccione un taller...”.
- **Nivel de experiencia**
  - Obligatorio.
- **Fecha del taller**
  - Obligatorio.
  - No puede ser una fecha pasada (debe ser hoy o una fecha futura).
- **Acepta términos y condiciones**
  - Obligatorio (debe ser true).
  - Si no se selecciona, mostrar mensaje: “Debe aceptar los términos y condiciones para completar la inscripción”.

Entregables

Cada pareja/grupo deberá entregar:

- Proyecto **ASP.NET Core MVC** completo en GitHub.
- Evidencias del uso de GIT:
  - Commits por cada fase (modelo, repositorio, controlador, vistas, estilos).
  - Uso de ramas para nuevas funcionalidades (por ejemplo: feature/validaciones-taller).
- Capturas de pantalla:
  - Formulario de registro con validaciones visibles.
  - Listado de inscripciones.
  - Mensaje de confirmación de registro exitoso.

Pasos de Desarrollo

Fase 1 – Creación del Proyecto

- Crear un nuevo proyecto **ASP.NET Core MVC**.
- Configurar el repositorio Git:
    - git init, primer commit, y conectar con GitHub.

Fase 2 – Creación del Modelo

- Crear la clase Inscripcion.cs en la carpeta **Models** con las propiedades: Nombre, Apellidos, Correo, Telefono, Taller, NivelExperiencia, FechaTaller, AceptaTerminos.
- Agregar anotaciones de validación:
    - [Required], [EmailAddress], [StringLength], [RegularExpression], [DataType(DataType.Date)].

Fase 3 – Creación del Repositorio

- Crear InscripcionRepository.cs en la carpeta **Data**.
- Mantener una lista en memoria:
    - private static List<Inscripcion> _inscripciones = new List<Inscripcion>();
- Métodos sugeridos:
    - AgregarInscripcion(Inscripcion inscripcion)
    - ObtenerInscripciones()

Fase 4 – Crear el Controlador

- Crear InscripcionController.cs en la carpeta **Controllers**.
- Métodos sugeridos:
    - Index() → Mostrar la lista de inscripciones.
    - Crear() (GET) → Mostrar formulario.
    - Crear(Inscripcion inscripcion) (POST) → Validar el modelo; si ModelState.IsValid es true, guardar y redirigir; si no, retornar la vista con mensajes de error.

Fase 5 – Crear las Vistas

- En la carpeta Views/Inscripcion crear:

- Index.cshtml → Tabla con las inscripciones y botón para “Nueva inscripción”.
- Crear.cshtml → Formulario con asp-for para cada campo y validation-summary.

Fase 6 – Personalización de las Páginas

- Agregar estilos usando **Bootstrap** (cards, botones, tabla responsive).
- Agregar mensajes:
  - Mensaje de éxito al completar la inscripción.
  - Mensajes de validación junto a cada campo.
- Incluir enlaces de navegación:
  - Menú simple en el *Layout* (Inicio, Inscripciones, etc.).