**Diseño y Desarrollo de una Aplicación Web para Control de Migración en C# MVC**

Jennifer Rebeca García Moncada

Universidad Politécnica Internacional

Programación III

Profesor: Alexander Benjamin Curling

Diciembre, 2024

# Introducción

**Objetivo del diseño:** El objetivo principal de este proyecto es desarrollar una aplicación web para gestionar los datos de pasajeros, documentos y transacciones migratorias, con un enfoque inicial en el manejo básico de datos sin diferenciar accesos según roles.

**Alcance del sistema:** El sistema actualmente permite:

* Gestionar información de pasajeros.
* Registrar documentos asociados a los pasajeros.
* Administrar transacciones migratorias.
* Autenticarse mediante un login.

# Arquitectura del Sistema

## Patrón de diseño utilizado:

**Modelo-Vista-Controlador (MVC):** La aplicación sigue el patrón MVC, separando responsabilidades entre:

* **Modelo:** Contiene las clases generadas por Entity Framework para representar los datos.
* **Vista:** Presenta la interfaz de usuario utilizando Razor.
* **Controlador:** Gestiona la interacción entre el usuario, la vista y el modelo.

## Herramientas y tecnologías:

**ASP.NET MVC 5:** Framework utilizado para la implementación de la aplicación.

**Entity Framework:** Herramienta para mapear y gestionar la base de datos.

**Bootstrap:** Para un diseño responsivo y moderno.

**SQL Server:** Base de datos utilizada para almacenar los datos.

**Visual Studio:** IDE utilizado para el desarrollo.

# Decisiones de Diseño

## Navegación y Menú

**Decisión:** Implementar un menú estático en \_Layout.cshtml que permita la navegación a las secciones principales del sistema.

**Razonamiento:** Un menú estático es más simple de implementar en la etapa inicial del proyecto y facilita la navegación básica.

**Ejecución:** El menú utiliza Html.ActionLink para navegar entre las acciones y controladores principales.

## Autenticación

**Decisión:** Implementar un login básico para autenticar usuarios.

**Razonamiento:** Esto proporciona una base inicial para la seguridad y permite la futura expansión hacia la gestión de roles.

Ejecución: El controlador LoginController verifica las credenciales contra la tabla USUARIOS.

Se almacena la información del usuario en la sesión (Session["UsuarioId"] y Session["Nombre"]).

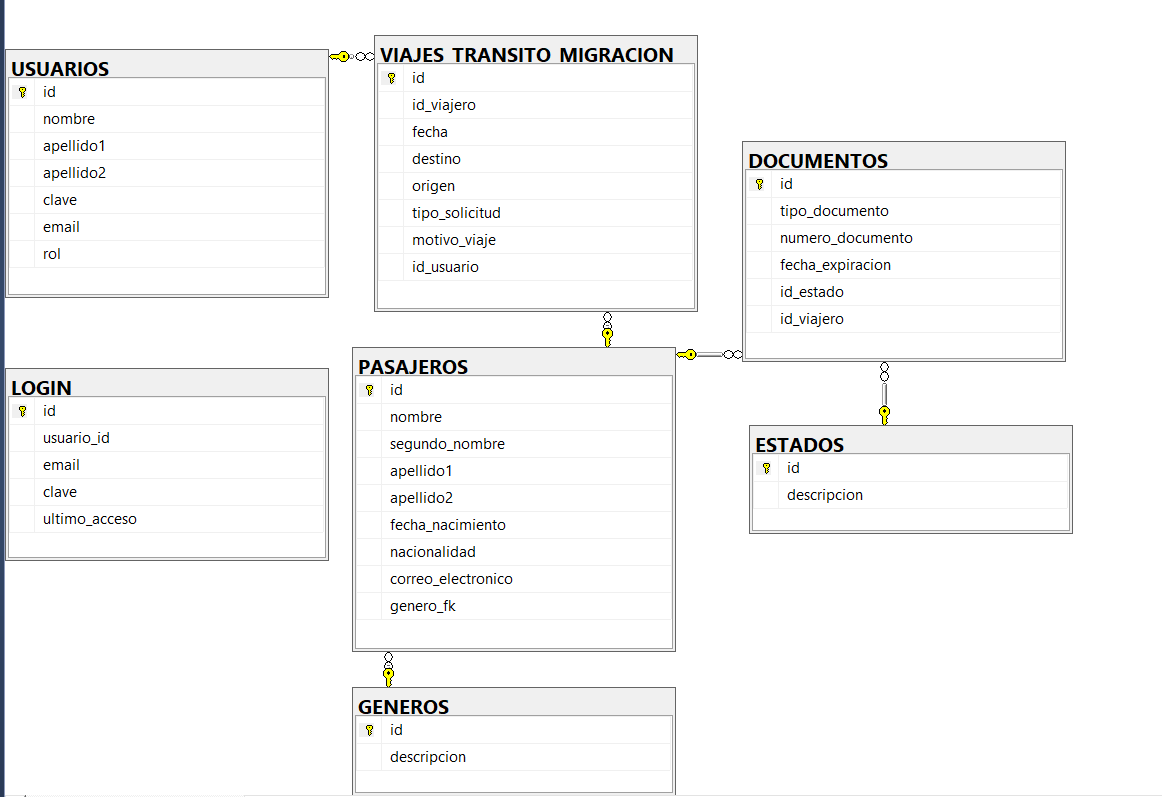
## Diseño de Base de Datos

**Decisión:** Diseñar y normalizar la base de datos hasta la 3FN.

**Razonamiento:** La normalización asegura la consistencia y evita redundancias en los datos.

**Ejecución:** Las tablas incluyen relaciones con claves foráneas (PASAJEROS, DOCUMENTOS, VIAJES\_TRANSITO\_MIGRACION, etc.).

Claves primarias y restricciones aseguran la integridad de los datos.



## Interfaz de Usuario

**Decisión:** Diseñar una interfaz simple pero funcional utilizando Bootstrap.

**Razonamiento:** Bootstrap facilita la creación de diseños responsivos sin esfuerzo adicional.

**Ejecución:** Formularios, tablas y la barra de navegación (navbar) utilizan clases de Bootstrap.

# Estructura del Proyecto

## Carpetas Principales:

**Controllers:** Contiene los controladores responsables de manejar las solicitudes del usuario (LoginController, PASAJEROSController, etc.).

**Models:** Contiene las entidades generadas por Entity Framework para representar las tablas de la base de datos.

**Views:** Contiene las vistas asociadas a cada controlador, así como \_Layout.cshtml para la estructura compartida.

**App\_Start:** Contiene configuraciones como rutas y filtros.

# Flujo del Sistema

## Flujo de autenticación:

1. El usuario accede al formulario de login (Login/Index).
2. El controlador LoginController valida las credenciales del usuario contra la base de datos.
3. Si las credenciales son correctas:

* La información del usuario se almacena en la sesión.
* El usuario es redirigido a la página principal (Home/Index).

## Flujo de navegación:

Desde la página principal, el usuario puede acceder a las secciones principales del sistema (Pasajeros, Documentos, etc.) a través del menú en \_Layout.cshtml.

Cada sección permite al usuario realizar operaciones básicas como ver, crear, editar y eliminar registros.

# Pruebas

## Escenarios probados:

**Inicio de sesión:**

* Se verifica que usuarios con credenciales válidas pueden iniciar sesión.
* Se verifica que credenciales inválidas muestran un mensaje de error.

**Operaciones CRUD:**

* Crear, leer, actualizar y eliminar registros en las tablas principales.

**Navegación:**

* Acceso a las diferentes secciones del menú y redirecciones correctas.

**Validaciones de datos:**

* Se asegura que los campos obligatorios en formularios están validados.
* Manejo de errores en datos duplicados o inválidos.

# Consideraciones para la Expansión

**Gestión de Roles:** En futuras versiones, se puede agregar un sistema basado en roles para restringir el acceso a ciertas secciones.

**Seguridad Avanzada:** Se debe considerar la encriptación de contraseñas y medidas adicionales para proteger la autenticación.

**Optimización del Menú:** Implementar un menú dinámico que cambie según el tipo de usuario.

# Conclusión

El diseño actual proporciona una base sólida para gestionar la información del sistema de migración. Las decisiones tomadas son simples y eficaces para esta etapa inicial, dejando abierta la posibilidad de mejoras y expansión en futuras iteraciones.