

**Facultad de Ingeniería de Sistemas, Universidad Internacional San Isidro  
Labrador.**

**Curso:**

**(ISB-32) PROGRAMACIÓN AVANZADA**

**Proyecto Final Programado.**

**Nombre del proyecto:**

**PortalAsociado.**

**Estudiante/Desarrollador:**

**Michael Molina Castillo.**

**Profesora: ESTEFANIA BOZA VILLALOBOS**

**Noviembre 2025.**

## Tabla de Contenido

Índice de Tablas.....	3
Índice de Figuras.....	4
Parte I: Información General del Proyecto.....	5
Objetivo General.....	5
Objetivos Específicos .....	5
Justificación del Proyecto .....	5
Alcances Esperados .....	6
Requerimientos del Proyecto.....	6
Parte II: Diseño del Sistema (Pre-diseño visual).....	8
Elementos de Diseño.....	8
Explicación .....	13
Parte III: Base de Datos .....	14
Definición:.....	14
Elementos .....	14
Parte IV: Desarrollo del Backend.....	31
1. Capa de Presentación (ASP.NET MVC). .....	31
2. Capa de Lógica de Negocio (LogicaNegocios). .....	32
3. Capa de Acceso a Datos (AccesoDatos). .....	32
4. Capa de Entidades (Entidades). .....	33
5. Control de Autenticación y Autorización.....	33
6. Diagrama Real de Flujo del Backend.....	34

## Índice de Tablas

Tabla 1. Colores.....	12
Tabla 2. Tabla Usuarios. ....	14
Tabla 3. Asociados.....	15
Tabla 4. Solicitudes.....	15
Tabla 5. Pagos.....	16
Tabla 6. HistorialAportes. ....	16
Tabla 7. Notificaciones. ....	17
Tabla 8. Roles.....	17
Tabla 9. PermisosPorRol.....	18
Tabla 10. Relación General Entre Tablas.....	18

## **Índice de Figuras**

Ilustración 1. Prediseño Login. ....	8
Ilustración 2. Prediseño Menú de navegación. ....	9
Ilustración 3. Prediseño Formulario de registro/solicitud.....	10
Ilustración 4. Prediseño Reportes.....	11
Ilustración 5. Diagrama de Base de Datos. ....	24

## **Parte I: Información General del Proyecto**

*PortalAsociado* es una solución web diseñada para fortalecer la relación entre Coopeuvita R.L. y sus asociados, facilitando la gestión de trámites, pagos y consultas de manera segura, eficiente y accesible. El sistema permite a los usuarios realizar solicitudes, efectuar pagos de capital social, y consultar el historial de aportes desde cualquier dispositivo con conexión a internet.

### **Objetivo General**

Desarrollar una plataforma digital intuitiva y segura que centralice los servicios clave para los asociados de Coopeuvita R.L., promoviendo la transparencia, la autonomía del usuario y la modernización de los procesos internos.

### **Objetivos Específicos**

- Permitir a los asociados realizar solicitudes en línea (ej. afiliación, trámites administrativos, consultas).
- Facilitar el pago de capital social mediante integración con pasarelas de pago seguras.
- Proporcionar acceso al historial de aportes y movimientos financieros personales.
- Mejorar la trazabilidad y eficiencia de los procesos administrativos.
- Garantizar la protección de datos conforme a la legislación costarricense vigente.

### **Justificación del Proyecto**

La transformación digital de los servicios cooperativos es una necesidad estratégica para fortalecer la relación entre Coopeuvita R.L. y sus asociados. Actualmente, muchos de los trámites relacionados con el capital social, solicitudes administrativas y consultas de aportes se realizan de forma presencial o mediante canales no integrados, lo que genera demoras, duplicidad de esfuerzos y limitaciones en la trazabilidad de la información.

PortalAsociado surge como respuesta a esta necesidad, ofreciendo una plataforma web segura, accesible y orientada al usuario, que centraliza los servicios clave para los asociados. Este sistema permitirá realizar pagos de capital social, gestionar solicitudes,

y consultar el historial de aportes de manera autónoma, desde cualquier dispositivo con conexión a internet.

La implementación de este proyecto no solo optimiza los procesos internos de la cooperativa, sino que también promueve la transparencia, la eficiencia operativa y el empoderamiento del asociado. Además, se alinea con los principios cooperativos de participación activa, educación financiera y acceso equitativo a la información.

En un entorno regulado y en constante evolución, contar con una herramienta digital que facilite el cumplimiento normativo, la generación de reportes auditables y la atención oportuna a los asociados representa una ventaja competitiva y una mejora sustancial en la calidad del servicio.

### **Alcances Esperados**

- El sistema permitirá a los asociados registrar, consultar y gestionar solicitudes, pagos de capital social y visualizar el historial de aportes de manera autónoma y segura.
- Se habilitará la generación de reportes y estadísticas consolidadas, orientadas a la toma de decisiones administrativas, auditorías internas y cumplimiento regulatorio.
- Los usuarios contarán con acceso en tiempo real a su información financiera y transaccional, desde cualquier dispositivo con conexión a internet.
- La arquitectura del sistema estará diseñada para permitir integración modular con futuras funcionalidades, tales como notificaciones automatizadas, según las necesidades de Coopeuvita R.L.

### **Requerimientos del Proyecto**

- **Requerimientos funcionales:**
  - Registro y autenticación de usuarios con credenciales seguras y control de acceso por roles.
  - Gestión de solicitudes administrativas por parte de los asociados (afiliación, trámites, consultas).
  - Procesamiento de pagos de capital social, con integración a pasarelas de pago.
  - Consulta de historial de aportes, movimientos y estado de cuenta personal.

- Generación de reportes y estadísticas para uso administrativo, auditorías y toma de decisiones.
- Sistema de notificaciones para alertas de vencimientos, confirmaciones y novedades.
- Panel administrativo para gestión de usuarios, solicitudes y configuración del sistema.
- **Requerimientos no funcionales:**
  - Seguridad: Cifrado de datos sensibles, validación de sesiones, cumplimiento con normativa de protección de datos.
  - Rendimiento: Respuesta eficiente bajo carga moderada, con tiempos de espera mínimos.
  - Escalabilidad: Arquitectura modular que permita incorporar nuevas funcionalidades sin afectar el rendimiento.
  - Usabilidad: Interfaz intuitiva, accesible desde dispositivos móviles y adaptada a distintos niveles de alfabetización digital.
  - Disponibilidad: Acceso continuo con tolerancia a fallos y respaldo periódico de la información.
- **Infraestructura mínima:**
  - Servidor web con capacidad para alojar aplicaciones .NET, Node.js o equivalente, con mínimo 4 vCPU, 8 GB RAM y almacenamiento SSD.
  - Base de datos: SQL Server o MySQL, con soporte para transacciones, integridad referencial y respaldo automatizado.
  - Framework de desarrollo: Preferiblemente .NET Core, aunque puede adaptarse a otras tecnologías según el equipo de desarrollo.
  - Certificado SSL para navegación segura (HTTPS).
  - Sistema de respaldo y recuperación ante fallos o pérdida de datos.

## Parte II: Diseño del Sistema (Pre-diseño visual)

El pre-diseño visual representa una propuesta inicial de la interfaz del sistema PortalAsociado. Su objetivo es ofrecer una guía clara y comprensible para los usuarios, desarrolladores y partes interesadas, facilitando la validación temprana del enfoque visual antes de la implementación técnica. Esta etapa permite anticipar decisiones de diseño, accesibilidad y navegación, alineando la experiencia del usuario con los objetivos funcionales del sistema.

### Elementos de Diseño

- **Estructura de pantallas:**

El sistema contempla las siguientes secciones clave:

- **Login:** Acceso mediante correo electrónico y contraseña, con opción de recuperación.



The image shows a pre-design of a login page for 'PortalAsociado'. The browser's address bar shows 'https://'. The page header features the 'CoopeUvita R.L.' logo and navigation links: 'Inicio', 'Nosotros', and 'Mi Portal'. The main content area is titled 'Iniciar Sesión.' and contains two input fields for 'Usuario:' and 'Contraseña:'. Below these is a link for 'Recuperar Datos de Ingreso'. There are two buttons: a green 'Ingresar' button and a light green 'Registrarse' button. The footer includes links for 'Home', 'Products', 'Company', and 'Blog', along with social media icons for Facebook, Twitter, Instagram, and TikTok.

Ilustración 1. Prediseño Login.



- *Menú de navegación: disposición lateral fija o superior desplegable, con íconos representativos.*



Ilustración 2. Prediseño Menú de navegación.

- *Formulario de registro/solicitud: campos dinámicos según tipo de trámite, validación en tiempo real.*



The image shows a web browser window with the address bar displaying "PortalAsociado" and "https://". The page header features the CoopeUvita R.L. logo, the company name, and navigation buttons for "Inicio", "Nosotros", and "Mi Portal". The main content area is titled "Crear Nuevo Usuario" and contains four input fields labeled "Correo:", "Nombre Completo:", "Telefono:", and "Contraseña:". A green "Guardar" button is positioned below the fields. The footer includes links for "Home", "Products", "Company", and "Blog", along with social media icons for Facebook, Twitter, Instagram, and TikTok.

Ilustración 3. Prediseño Formulario de registro/solicitud.

- *Reportes: visualización de gráficos, tablas filtrables y opción de exportación en PDF.*


PortalAsociado

←

→

↺

https://



CoopeUvita R.L.

Inicio

Nosotros

Mi Portal

Información A

Pagos

Solicitudes

Historial

OCTOBER 2025



S	M	T	W	T	F	S
28	29	30	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	1



Historial Aportes

Mes	Monto	Fecha Pago
enero 2025	5,000.00	02/01/2025
febrero 2025	5,000.00	01/02/2025
marzo 2025	5,000.00	03/03/2025
abril 2025	5,000.00	02/04/2025
mayo 2025	5,000.00	02/05/2025
junio 2025	5,000.00	01/06/2025
julio 2025	5,000.00	01/07/2025
agosto 2025	5,000.00	31/07/2025
septiembre 2025	5,000.00	30/08/2025
octubre 2025	5,000.00	29/09/2025
noviembre 2025	5,000.00	29/10/2025
<b>Total Aportado</b>	<b>55,000.00</b>	

[Home](#) | 
 [Products](#) | 
 [Company](#) | 
 [Blog](#)




Ilustración 4. Prediseño Reportes.

- **Colores:** Se propone una combinación basada en verdes y gris claro, que transmite modernidad, simplicidad y accesibilidad:

Elemento	Color HEX	Uso
<i>Primario</i>	#349a0b	Botones, enlaces activos
<i>Secundario</i>	#CAF0F8	Fondos suaves, tarjetas
<i>Texto principal</i>	#343A40	Títulos, párrafos
<i>Fondo general</i>	b6d7a8	Fondo de pantallas

Tabla 1. Colores.

- **Tipografía:**

Se selecciona una fuente moderna y legible:

- **Principal:** Segoe UI
- **Alternativas compatibles:** Roboto, Arial

- **Imágenes:**

Se utilizará iconografía sencilla y representativa para cada módulo, facilitando la navegación intuitiva:

- **Solicitudes** → *fa-file-alt*
- **Aportes** → *fa-credit-card*
- **Reportes** → *fa-chart-bar*
- **Configuración** → *fa-cog*
- **Perfil** → *fa-user-circle*

Las imágenes complementarias serán minimalistas, evitando sobrecarga visual.

- **Íconos:**

Se hará uso de librerías estándar como:

- **FontAwesome**
- **Lucide-React**
- **Heroicons**
- **Bootstrap Icons**
- **Material Icons**

- **Slogan:**

**“Tecnología al servicio de nuestros asociados.”**

## **Explicación**

El pre-diseño visual funciona como una guía preliminar que permitirá a los usuarios y al equipo de desarrollo comprender la distribución, apariencia y navegación general del sistema antes de proceder con su codificación. Este enfoque facilita la retroalimentación temprana, la validación de decisiones de diseño y la alineación con los objetivos funcionales del proyecto.

## Parte III: Base de Datos

### Definición:

- **Tipo:** Relacional
- **Propósito:** Gestionar de forma segura y eficiente los datos de usuarios, solicitudes, pagos y reportes del sistema PortalAsociado.
- **Motor elegido:** SQL Server (por compatibilidad con .NET Core y soporte robusto para transacciones y seguridad).
- **Conexión con la Aplicación:** ADO.NET con Entity Framework Core

### Elementos

#### 1. Tabla Usuarios:

La tabla **Usuarios** gestiona la información de acceso al sistema. Contiene los datos necesarios para autenticar y autorizar a los usuarios, como nombre, correo, contraseña cifrada y rol asignado.

- **Propósito:** Controlar el acceso al sistema y definir el tipo de usuario (Asociado, Administrador, etc.).
- **Relaciones:** Se vincula con Solicitudes, Pagos, HistorialAportes, Notificaciones, y opcionalmente con Asociados.
- **Justificación:** Separa la lógica de autenticación de los datos institucionales del asociado.

Campo	Tipo de Dato	Descripción
UsuarioID	INT (PK)	Identificador único del usuario
NombreCompleto	VARCHAR(100)	Nombre completo del asociado
CorreoElectronico	VARCHAR(100)	Correo electrónico único
Telefono	VARCHAR(20)	Número de contacto
ContraseñaHash	VARCHAR(255)	Contraseña cifrada
FechaRegistro	DATETIME	Fecha de creación del usuario
Rol	VARCHAR(20)	Rol del usuario (Asociado, Admin, etc.)
Activo	BIT	Estado lógico del usuario

Tabla 2. Tabla Usuarios.

## 2. Asociados:

La tabla **Asociados** almacena la información institucional de los miembros de la cooperativa, independientemente de si tienen acceso digital.

- **Propósito:** Registrar datos personales y administrativos del asociado, como cédula, dirección, estado civil y fecha de afiliación.
- **Relaciones:** Puede vincularse opcionalmente con Usuarios mediante UsuarioID.
- **Justificación:** Permite que existan asociados sin cuenta digital, manteniendo la base normalizada y flexible

Campo	Tipo de Dato	Descripción
AsociadoID	INT (PK)	Identificador único del asociado
Cedula	VARCHAR(20)	Número de cédula o identificación
NombreCompleto	VARCHAR(100)	Nombre completo del asociado
FechaNacimiento	DATE	Fecha de nacimiento
EstadoCivil	VARCHAR(20)	Estado civil del asociado
Direccion	TEXT	Dirección física
Telefono	VARCHAR(20)	Número de contacto
FechaAfiliacion	DATE	Fecha de afiliación al sistema
UsuarioID	INT (FK, NULL)	Usuario vinculado (si tiene cuenta)

Tabla 3. Asociados.

## 3. Solicitudes:

La tabla **Solicitudes** registra los trámites realizados por los usuarios, como afiliaciones, consultas o gestiones administrativas.

- **Propósito:** Llevar control del tipo de solicitud, su estado y fecha de creación.
- **Relaciones:** Cada solicitud está asociada a un usuario mediante UsuarioID.
- **Justificación:** Permite trazabilidad de gestiones individuales y facilita el seguimiento administrativo.

Campo	Tipo de Dato	Descripción
SolicitudID	INT (PK)	Identificador único de la solicitud
UsuarioID	INT (FK)	Usuario que realiza la solicitud
TipoSolicitud	VARCHAR(50)	Tipo de trámite
Descripcion	TEXT	Detalle de la solicitud
Estado	VARCHAR(20)	Estado de la solicitud
FechaSolicitud	DATETIME	Fecha de creación de la solicitud

Tabla 4. Solicitudes.

#### 4. Pagos:

La tabla **Pagos** almacena las transacciones financieras realizadas por los usuarios, como aportes al capital social.

- Propósito: Registrar montos, fechas, métodos de pago y referencias asociadas.
- Relaciones: Cada pago está vinculado a un usuario mediante UsuarioID.
- Justificación: Permite control contable y generación de reportes financieros

Campo	Tipo de Dato	Descripción
PagoID	INT (PK)	Identificador único del pago
UsuarioID	INT (FK)	Usuario que realiza el pago
Monto	DECIMAL(10.2)	Monto aportado
FechaPago	DATE	Fecha del pago
MetodoPago	VARCHAR(50)	Método utilizado
Referencia	VARCHAR(100)	Código de referencia del pago

Tabla 5. Pagos.

#### 5. HistorialAportes:

La tabla **HistorialAportes** consolida los aportes mensuales de cada usuario, permitiendo análisis por periodo.

- **Propósito:** Registrar el monto aportado por mes y la fecha efectiva del pago.
- **Relaciones:** Cada registro está vinculado a un usuario mediante UsuarioID.
- **Justificación:** Facilita auditorías, reportes y seguimiento del cumplimiento de aportes.

Campo	Tipo de Dato	Descripción
AporteID	INT (PK)	Identificador del registro de aporte
UsuarioID	INT (FK)	Usuario asociado al aporte
Mes	VARCHAR(20)	Mes del aporte
Monto	DECIMAL(10.2)	Monto aportado ese mes
FechaPago	DATE	Fecha efectiva del pago

Tabla 6. HistorialAportes.

#### 5. Notificaciones:

La tabla **Notificaciones** gestiona los mensajes enviados a los usuarios, como alertas de vencimientos o confirmaciones.



- **Propósito:** Mejorar la comunicación entre el sistema y el usuario.
- **Relaciones:** Cada notificación está asociada a un usuario mediante UsuarioID.
- **Justificación:** Permite trazabilidad de la comunicación y mejora la experiencia del usuario.

Campo	Tipo de Dato	Descripción
NotificacionID	INT (PK)	Identificador único
UsuarioID	INT (FK)	Usuario destinatario
Mensaje	VARCHAR(255)	Contenido de la notificación
FechaEnvio	DATETIME	Fecha y hora de envío
Leido	BIT	Indicador de lectura

Tabla 7. Notificaciones.

## 6. Roles:

La tabla **Roles** define los tipos de usuario dentro del sistema, como Asociado, Administrador o Supervisor.

- **Propósito:** Establecer categorías de acceso y responsabilidades.
- **Relaciones:** Se vincula con Usuarios y con PermisosPorRol.
- **Justificación:** Permite escalabilidad y control granular de privilegios

Campo	Tipo de Dato	Descripción
RolID	INT (PK)	Identificador único del rol
NombreRol	VARCHAR(50)	Nombre del rol (ej. Asociado, Administrador)
Descripcion	TEXT	Descripción funcional del rol

Tabla 8. Roles.

## 7. PermisosPorRol:

La tabla **PermisosPorRol** asigna permisos específicos a cada rol, como ver solicitudes, editar usuarios o generar reportes.

- **Propósito:** Controlar qué acciones puede realizar cada tipo de usuario.
- **Relaciones:** Cada permiso está vinculado a un rol mediante RolID.
- **Justificación:** Refuerza la seguridad y permite personalizar el comportamiento del sistema según el perfil.

Campo	Tipo de Dato	Descripción
PermisoID	INT (PK)	Identificador único del permiso
RolID	INT (FK)	Rol al que se asigna el permiso
NombrePermiso	VARCHAR(50)	Nombre del permiso (ej. VerSolicitudes, EditarUsuarios)
Descripcion	TEXT	Descripción funcional del permiso

Tabla 9. PermisosPorRol.

### Relación General Entre Tablas:

Entidad Origen	Entidad Destino	Relación	Descripción
Usuarios	Solicitudes	1:N	Un usuario puede tener varias solicitudes
Usuarios	Pagos	1:N	Un usuario puede realizar varios pagos
Usuarios	HistorialAportes	1:N	Un usuario puede tener múltiples registros de aportes
Usuarios	Notificaciones	1:N	Un usuario puede recibir múltiples notificaciones
Roles	PermisosPorRol	1:N	Un rol puede tener múltiples permisos asignados
Roles	Usuarios	1:N	Un rol puede estar asignado a varios usuarios
Asociados	Usuarios	1:1 (opcional)	Un asociado puede tener un usuario vinculado, pero no es obligatorio

Tabla 10. Relación General Entre Tablas.

### Triggers:

#### 1. TRG\_EvitarPagoDuplicado

- **Tabla afectada:** Pagos
- **Evento:** FOR INSERT
- **Propósito:** Este trigger valida que un usuario no registre más de un pago en la misma fecha. Si se detecta un duplicado, se lanza un error y se cancela la transacción.
- **Justificación:** Evita errores contables y duplicidad de registros que podrían afectar reportes financieros y el historial de aportes

#### 2. TRG\_RegistrarAporteDesdePago

- **Tabla afectada:** Pagos
- **Evento:** AFTER INSERT

- **Propósito:** Cada vez que se registra un pago, este trigger crea automáticamente un registro en la tabla HistorialAportes, extrayendo el mes desde la fecha de pago.
- **Justificación:** Automatiza el proceso de consolidación mensual de aportes, asegurando que cada pago quede reflejado en el historial sin intervención manual

### 3. TRG\_MarcarNotificacionLeida

- **Tabla afectada:** Notificaciones
- **Evento:** AFTER UPDATE
- **Propósito:** Detecta cuando una notificación ha sido marcada como leída (Leido = 1) y puede activar acciones complementarias como auditoría o limpieza de bandeja.
- **Justificación:** Mejora la trazabilidad de la interacción del usuario con el sistema y permite futuras extensiones como métricas de lectura.

### 4. TRG\_AuditarUsuarios

- **Tabla afectada:** Usuarios
- **Evento:** AFTER INSERT, UPDATE, DELETE
- **Propósito:** Registra en una tabla de auditoría (AuditoriaUsuarios) los cambios realizados sobre los usuarios, incluyendo nombre, correo y tipo de acción.
- **Justificación:** Refuerza la seguridad y trazabilidad del sistema, permitiendo revisar quién modificó qué y cuándo, útil para control interno y cumplimiento

### 5. TRG\_ValidarCorreoUnico

- **Tabla afectada:** Usuarios
- **Evento:** FOR INSERT, UPDATE
- **Propósito:** Refuerza la restricción de unicidad del correo electrónico, incluso si el constraint fue omitido o desactivado.
- **Justificación:** Garantiza que cada usuario tenga un correo único, evitando conflictos en autenticación y duplicidad de cuentas.

## Stored Procedures:

### 1. sp\_RegistrarUsuario

- **Propósito:** Inserta un nuevo usuario en el sistema, validando previamente que el correo electrónico no esté duplicado.
- **Parámetros:** Nombre completo, correo electrónico, teléfono, contraseña cifrada, ID de rol.
- **Tabla afectada:** Usuarios
- **Justificación:** Este procedimiento encapsula la lógica de validación y registro, asegurando la unicidad del correo y la correcta asignación de roles. Mejora la seguridad y evita errores de duplicidad.

### 2. sp\_RegistrarPago

- **Propósito:** Registra un pago realizado por un usuario y genera automáticamente el correspondiente registro en HistorialAportes.
- **Parámetros:** ID de usuario, monto, fecha de pago, método de pago, referencia.
- **Tablas afectadas:** Pagos, HistorialAportes
- **Justificación:** Automatiza el vínculo entre pagos y aportes mensuales, reduciendo errores humanos y asegurando la trazabilidad financiera
- 

### 3. sp\_RegistrarSolicitud

- **Propósito:** Inserta una nueva solicitud asociada a un usuario, especificando el tipo de trámite y su descripción.
- **Parámetros:** ID de usuario, tipo de solicitud, descripción.
- **Tabla afectada:** Solicitudes
- **Justificación:** Centraliza la lógica de registro de trámites, facilitando el seguimiento administrativo y la generación de reportes.

#### 4. sp\_EnviarNotificacion

- **Propósito:** Envía una notificación al usuario, almacenando el mensaje y la fecha de envío.
- **Parámetros:** ID de usuario, mensaje.
- **Tabla afectada:** Notificaciones
- **Justificación:** Mejora la comunicación interna del sistema, permitiendo alertas, confirmaciones y recordatorios automatizados

#### 5. sp\_RegistrarAsociado

- **Propósito:** Registra un nuevo asociado en el sistema, con posibilidad de vincularlo a un usuario si ya tiene cuenta.
- **Parámetros:** Cédula, nombre completo, fecha de nacimiento, estado civil, dirección, teléfono, ID de usuario (opcional).
- **Tabla afectada:** Asociados
- **Justificación:** Permite mantener un registro institucional de los miembros, incluso si no tienen acceso digital. Refuerza la normalización y flexibilidad del modelo

#### 6. sp\_AsignarPermisoARol

- **Propósito:** Asigna un permiso específico a un rol determinado, definiendo qué acciones puede realizar ese tipo de usuario.
- **Parámetros:** ID de rol, nombre del permiso, descripción.
- **Tabla afectada:** PermisosPorRol
- **Justificación:** Refuerza el control de acceso y permite personalizar el comportamiento del sistema según el perfil del usuario

## Vistas:

### 1. vw\_HistorialAportesDetallado

- **Propósito:** Presenta un historial consolidado de aportes mensuales por asociado, incluyendo nombre, mes, monto y fecha de pago.
- **Tablas involucradas:** HistorialAportes, Usuarios, Asociados
- **Justificación:** Facilita la generación de reportes financieros y administrativos, vinculando pagos con datos institucionales del asociado.

### 2. vw\_SolicitudesPendientes

- **Propósito:** Muestra todas las solicitudes que se encuentran en estado "Pendiente", junto con el nombre del usuario y tipo de trámite.
- **Tablas involucradas:** Solicitudes, Usuarios
- **Justificación:** Permite a administradores y supervisores visualizar rápidamente los trámites en espera, optimizando la gestión operativa.

### 3. vw\_ResumenPagosPorMes

- **Propósito:** Agrupa los pagos realizados por mes, mostrando el total de transacciones y el monto acumulado.
- **Tabla involucrada:** Pagos
- **Justificación:** Proporciona una vista agregada útil para análisis contable, planificación financiera y visualización en dashboards

## Funciones:

### 1. fn\_TotalAportesPorUsuario

- **Propósito:** Calcula el total acumulado de aportes registrados por un usuario en la tabla HistorialAportes.
- **Parámetro:** @UsuarioID
- **Retorno:** DECIMAL(10,2)

- **Justificación:** Permite obtener métricas individuales para reportes, validaciones o comparaciones entre asociados.

## 2. fn\_EdadDesdeNacimiento

- **Propósito:** Calcula la edad actual de un asociado a partir de su fecha de nacimiento.
- **Parámetro:** @FechaNacimiento
- **Retorno:** INT
- **Justificación:** Útil para segmentar asociados por edad, validar requisitos de afiliación o generar estadísticas demográficas

## Diagrama de Base de Datos:

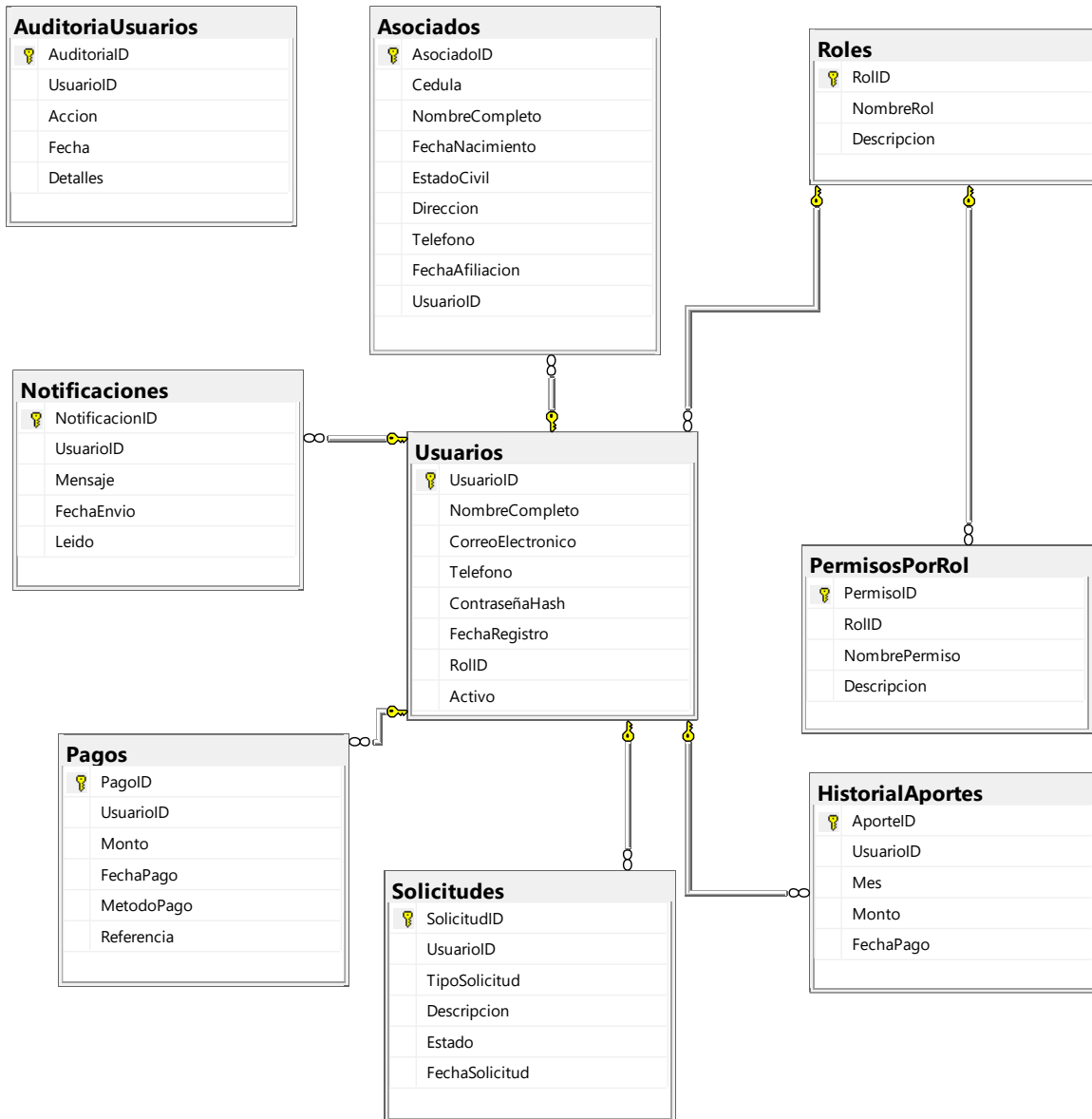


Ilustración 5. Diagrama de Base de Datos.



## **Casos de Uso.**

### **Caso de Uso 1: Baja de Usuario**

- 1. Nombre del Caso de Uso:** Desactivar cuenta de usuario.
- 2. Actores:** Administrador.
- 3. Objetivo:** Inhabilitar el acceso de un usuario sin eliminar sus registros históricos.
- 4. Precondiciones:**
  - El usuario debe existir en la base de datos.
  - El administrador debe tener permisos para modificar usuarios.
- 5. Flujo Principal:**
  1. El administrador accede al módulo de gestión de usuarios.
  2. Selecciona el usuario que desea desactivar.
  3. El sistema actualiza el campo Activo a 0 en la tabla Usuarios.
  4. El trigger TRG\_AuditarUsuarios registra la acción en AuditoriaUsuarios.
- 6. Flujos Alternos / Excepciones:**
  - Si el usuario no existe, se muestra un mensaje de error.
  - Si el administrador no tiene permisos, se bloquea la acción.
- 7. Postcondiciones:**
  - El usuario queda inactivo y no puede iniciar sesión.
  - Todos sus registros históricos permanecen intactos.
  - La acción queda registrada en la auditoría.

## **Caso de Uso 2: Anulación de Aporte**

- 1. Nombre del Caso de Uso:** Anular aporte mensual.
- 2. Actores:** Administrador.
- 3. Objetivo:** Eliminar o marcar como inválido un aporte registrado por error.
- 4. Precondiciones:**
  - El aporte debe existir en la tabla HistorialAportes.
  - El administrador debe tener permisos para modificar aportes.
- 5. Flujo Principal:**
  5. El administrador accede al historial de aportes.
  6. Localiza el aporte por UsuarioID, Mes o FechaPago.
  7. Ejecuta la acción de anulación (soft delete o hard delete).
  8. El sistema actualiza el campo Anulado o elimina el registro.
  9. Se registra la acción en AuditoriaAportes con motivo y fecha.
- 6. Flujos Alternos / Excepciones:**
  - Si el aporte no existe, se muestra un mensaje de error.
  - Si el aporte ya fue anulado, se bloquea la acción.
  - Si el administrador no tiene permisos, se deniega la operación.
- 7. Postcondiciones:**
  - El aporte queda marcado como anulado o eliminado.
  - El historial financiero se actualiza.
  - La acción queda registrada para trazabilidad.

### **Caso de Uso 3: Registro de Nuevo Asociado sin Usuario**

1. **Nombre del Caso de Uso:** Registrar asociado sin cuenta digital.
2. **Actores:** Administrador.
3. **Objetivo:** Ingresar los datos institucionales de un nuevo asociado sin necesidad de vincularlo a un usuario del sistema.
4. **Precondiciones:**
  - El asociado no debe existir previamente (validación por cédula).
  - El administrador debe tener permisos de registro.
5. **Flujo Principal:**
  1. El administrador accede al módulo de asociados.
  2. Completa el formulario con los datos personales del nuevo asociado.
  3. El sistema valida que la cédula no esté duplicada.
  4. Se inserta el registro en la tabla Asociados con UsuarioID = NULL.
6. **Flujos Alternos / Excepciones:**
  - Si la cédula ya existe, se muestra un error y se cancela el registro.
  - Si el administrador no tiene permisos, se bloquea la acción.
7. **Postcondiciones:**
  - El nuevo asociado queda registrado en el sistema.
  - No tiene acceso digital, pero puede ser vinculado a un usuario en el futuro.

### **Caso de Uso 4: Generación Automática de Aportes desde Pagos**

1. **Nombre del Caso de Uso:** Registrar aporte mensual desde pago.
2. **Actores:** Usuario, Sistema.
3. **Objetivo:** Automatizar el registro del aporte mensual al realizar un pago.

#### **4. Precondiciones:**

- El usuario debe estar activo.
- El pago debe ser válido y no duplicado en la misma fecha.

#### **5. Flujo Principal:**

1. El usuario realiza un pago desde la aplicación web.
2. El sistema valida que no exista otro pago en la misma fecha.
3. Se registra el pago en la tabla Pagos.
4. El trigger TRG\_RegistrarAporteDesdePago crea automáticamente el registro en HistorialAportes.

#### **6. Flujos Alternos / Excepciones:**

- Si el pago ya existe, se lanza un error y se cancela la transacción
- Si el usuario está inactivo, se bloquea la operación

#### **7. Postcondiciones:**

- El pago queda registrado.
- El aporte mensual queda reflejado en el historial sin intervención manual.

### **Caso de Uso 5: Asignación de Permisos por Rol**

**1. Nombre del Caso de Uso:** Asignar permiso a rol.

**2. Actores:** Administrador.

**3. Objetivo:** Definir qué acciones puede realizar cada tipo de usuario en el sistema.

#### **4. Precondiciones:**

- El rol debe existir en la tabla Roles.
- El administrador debe tener privilegios para modificar permisos.

## 5. Flujo Principal:

1. El administrador accede al módulo de configuración de roles
2. Selecciona el rol al que desea asignar permisos
3. Ingresa el nombre y descripción del permiso
4. El sistema registra el permiso en la tabla PermisosPorRol

## 6. Flujos Alternos / Excepciones:

- Si el rol no existe, se muestra un error.
- Si el permiso ya está asignado, se evita duplicidad.

## 7. Postcondiciones:

- El rol queda actualizado con nuevos permisos.
- Los usuarios con ese rol heredan automáticamente las nuevas capacidades.

## Caso de Uso 6: Consulta de Solicitudes Pendientes por Supervisor

1. **Nombre del Caso de Uso:** Revisar solicitudes pendientes.

2. **Actores:** Supervisor.

3. **Objetivo:** Visualizar todas las solicitudes que aún no han sido procesadas.

### 4. Precondiciones:

- El supervisor debe tener permisos de consulta.
- Deben existir solicitudes en estado "Pendiente".

## 5. Flujo Principal:

1. El supervisor accede al módulo de solicitudes
2. El sistema ejecuta la vista vw\_SolicitudesPendientes
3. Se muestran las solicitudes con estado "Pendiente" y datos del solicitante
4. El supervisor puede filtrar por tipo, fecha o usuario

## **6. Flujos Alternos / Excepciones:**

- Si no hay solicitudes pendientes, se muestra un mensaje informativo.
- Si el supervisor no tiene permisos, se bloquea el acceso.

## **7. Postcondiciones:**

- El supervisor obtiene una vista clara de los trámites en espera
- Puede tomar decisiones o asignar tareas de seguimiento

## Parte IV: Desarrollo del Backend

El backend de PortalAsociado está construido bajo una **arquitectura por capas**, siguiendo el patrón **ASP.NET MVC** tradicional. Esta estructura permite separar de manera clara la lógica de presentación, negocio y datos, lo que facilita el mantenimiento, la escalabilidad y la incorporación de nuevas funcionalidades sin afectar la estabilidad del sistema.

La arquitectura real está organizada en las siguientes capas:

### 1. Capa de Presentación (ASP.NET MVC).

Corresponde al proyecto **WebPortalAsociado**, el cual contiene:

- **Controladores (Controllers):**

Administran el flujo de la aplicación, reciben solicitudes del usuario, validan datos, llaman a la lógica de negocio y devuelven vistas.

Ejemplos:

- AccountController
- UsuariosController
- AsociadosController
- PagosController
- SolicitudesController

- **Vistas (Razor Views):**

Implementan la interfaz del portal, incluyendo formularios, tablas, reportes y menús dinámicos.

- **ViewModels:**

Se utilizan para enviar información específica a las vistas (ejemplo: Dashboard, Formularios de registro, Reportes PDF).

- **Scripts y estilos:**

Incluyen funcionalidades como el menú colapsable, validaciones en cliente y manejo de mensajes de éxito/error.

Esta capa integra directamente herramientas como **Rotativa** para generar PDFs y control de sesión/autenticación propio de ASP.NET.

## 2. Capa de Lógica de Negocio (LogicaNegocios).

La capa **LN** contiene la lógica funcional del sistema. Se encarga de:

- Validar información antes de enviarla a la base de datos.
- Encapsular reglas de negocio (por ejemplo, evitar duplicados).
- Gestionar flujos completos:  
creación de asociados, solicitudes, pagos, notificaciones.
- Coordinar la comunicación entre controladores y capa de datos.

Ejemplos de clases de negocio:

- UsuariosLN
- AsociadosLN
- PagosLN
- SolicitudesLN

Esta capa garantiza que la aplicación mantenga integridad y consistencia independientemente de la interfaz utilizada.

## 3. Capa de Acceso a Datos (AccesoDatos).

La capa **AD** interactúa directamente con **SQL Server** a través de:

- **Entity Framework tradicional**
- **Procedimientos almacenados (SP)**
- **Mapeo de resultados a entidades** (sp\_ObtenerUsuarioPorCorreo\_Result, etc.)
  - Sus responsabilidades incluyen:
  - Ejecutar SP para consultas, inserciones y actualizaciones.
  - Manejar transacciones según la lógica definida en SQL Server.
  - Administrar excepciones y conexión con la base de datos.

Ejemplos de SP utilizados:

- sp\_ObtenerUsuarios
- sp\_modUsuarioPorCorreo
- sp\_RegistrarAsociado



- `sp_RegistrarPago`

Esta capa evita duplicar lógica SQL en el código y garantiza integridad mediante triggers y constraints de la BD.

#### **4. Capa de Entidades (Entidades).**

Incluye las clases que representan:

- **Tablas reales de la base de datos**
- **Resultados de procedimientos almacenados**
- **DTOs específicos para transporte de datos**

Ejemplos:

- Usuarios
- Asociados
- Solicitudes
- Pagos
- `sp_ObtenerUsuarioPorCorreo_Result`
- `DashboardInicioViewModel`

Estas entidades permiten que cada capa comparta modelos consistentes y seguros.

#### **5. Control de Autenticación y Autorización.**

El proyecto implementa:

##### **Autenticación**

- Uso de FormsAuthentication
- Contraseñas cifradas con BCrypt.Net
- Manejo seguro de sesión y expiración

##### **Autorización basada en permisos**

El sistema incorpora un atributo personalizado:

*[PermisoAuthorize("GestionAsociados")]*

Este filtro evalúa dinámicamente si el usuario autenticado posee los permisos requeridos según su **RoIID** y los permisos asignados desde BD.

Si no cumple los permisos:

- Se bloquea el acceso.
- Se devuelve un 403 amigable con mensaje de advertencia.

## 6. Diagrama Real de Flujo del Backend

Usuario → Controlador MVC → Lógica de Negocio (LN)  
→ AccesoDatos (AD) → SQL Server (SP / Triggers)  
← Respuesta procesada ← LN ← Controlador ← Vista

Este flujo refleja cómo cada solicitud atraviesa la arquitectura, asegurando validaciones, seguridad y trazabilidad en cada etapa.

La implementación actual del backend de PortalAsociado se caracteriza por:

- Separación clara de responsabilidades.
- Consistencia entre lógica y datos.
- Seguridad basada en roles y permisos.
- Integración con servicios externos (Tilopay).
- Escalabilidad para agregar nuevos módulos en el futuro.

Esta arquitectura está alineada con buenas prácticas empresariales para aplicaciones MVC robustas, facilitando mantenimiento y evolución incremental del sistema.

## **Conclusión:**

El desarrollo del sistema PortalAsociado representa una solución integral orientada a modernizar la relación entre Coopeuvita R.L. y sus asociados mediante una plataforma web segura, accesible y funcional. A lo largo del proyecto se mantuvo un enfoque centrado en la eficiencia operativa, la autonomía del usuario y la trazabilidad de la información, respondiendo a necesidades reales de gestión administrativa y financiera dentro de la organización.

Desde la fase de análisis se identificaron requerimientos clave como la trazabilidad de aportes, el registro y seguimiento de solicitudes, la administración de usuarios y la digitalización de pagos. Estos elementos permitieron establecer objetivos claros y orientados a mejorar la experiencia del asociado y la eficiencia interna. El diseño visual planteado en las etapas iniciales sirvió como referencia para construir vistas intuitivas y coherentes con los principios de accesibilidad y usabilidad.

En la Parte III se consolidó una estructura de datos robusta basada en SQL Server, complementada con procedimientos almacenados, triggers y validaciones que garantizan integridad, consistencia y control sobre las operaciones críticas del sistema. Esta base de datos respalda funcionalidades como el registro automático de aportes, la prevención de pagos duplicados, la gestión de roles y permisos, y la generación de reportes auditables para la toma de decisiones.

La Parte IV presentó la implementación real del backend bajo una arquitectura por capas utilizando ASP.NET MVC. La separación entre presentación, lógica de negocio y acceso a datos permite un desarrollo modular y escalable, favoreciendo el mantenimiento y la evolución futura del sistema. Se integraron mecanismos de autenticación mediante contraseñas cifradas, autorización basada en permisos definidos en base de datos, validaciones tanto en servidor como en cliente y procesos automáticos de auditoría. Además, se incorporó la integración con la pasarela de pagos Tilopay, permitiendo a los asociados realizar aportes de forma digital y segura.

En conjunto, PortalAsociado se consolida como una herramienta tecnológica alineada con los valores cooperativos de transparencia, participación y servicio. Su diseño modular, su enfoque centrado en el usuario y su arquitectura escalable lo convierten en

una plataforma capaz de crecer, adaptarse y responder a las necesidades presentes y futuras de la organización y sus asociados.