



UNIVERSIDAD TECNOLÓGICA DE PANAMÁ



FACULTAD DE INGENIERÍA EN SISTEMAS

LICENCIATURA EN GESTION Y DESARROLLO DE SOFTWARE

Parcial 3 De Desarrollo De Software IV

PROFESOR:

Regis Rivera

ASIGNATURA:

Desarrollo de Software VI

Estudiante:

CRISTHIAN GUEVARA 8-1034-1549

GRUPO 1GS221

1. ¿Cómo resolvería los problemas del bufete con una solución tecnológica?

El bufete enfrenta problemas graves debido al uso de hojas de cálculo y documentos de texto para gestionar sus casos legales. Esto genera pérdida de información, duplicidad de datos, problemas de acceso, mala comunicación interna y retrasos en plazos legales.

Para resolver estos problemas se implementará una aplicación web centralizada desarrollada en ASP.NET Core con Razor Pages, con la lógica de negocio en C# y almacenamiento en SQL Server.

La solución tecnológica permitirá:

1.1 Centralización de la información

Toda la información estará en una sola plataforma accesible desde cualquier dispositivo dentro del bufete.

1.2 Automatización

El sistema gestionará:

- Fechas de vencimiento
- Alertas
- Documentos relacionados a cada caso

1.3 Seguridad

Se empleará autenticación con Login de usuarios para garantizar que solo abogados y personal autorizado accedan a los casos.

1.4 Orden y estructura

Cada módulo de la aplicación tendrá su pantalla:

- Clientes
- Casos
- Documentos
- Eventos importantes (audiencias, reuniones, vencimientos)

1.5 Búsquedas rápidas

El sistema permitirá buscar por:

- Cliente
- Número de caso
- Abogado asignado
- Estado del caso

1.6 Prevención de pérdida de datos

SQL Server garantiza integridad, respaldo, rendimiento y relaciones ordenadas.

2. Pasos para desarrollar el sistema

FASE 1 — PLANIFICACIÓN

1. Análisis del problema: Se revisan los procesos actuales del bufete y se detectan los puntos débiles: desorden, falta de control de fechas, mala gestión documental.

2. Identificación de requisitos: Se definen requisitos funcionales (qué debe hacer el sistema) y técnicos (cómo debe funcionar).

3. Diseño preliminar del sistema

Se establecen los módulos principales:

- Usuarios (login)
- Clientes
- Casos
- Documentos
- Eventos

FASE 2 — DISEÑO DE BASE DE DATOS

4. Crear la base de datos CristihianGuevaraDB en SQL Server: El proyecto usará SQL Server por estabilidad, seguridad y capacidad de manejar datos estructurados.

5. Diseñar las tablas con iniciales CG_

Ej:

- CG_Usuarios
- CG_Clientes
- CG_Casos
- CG_Eventos
- CG_Documentos

6. Definir relaciones entre tablas:

Relaciones tipo:

- Cliente tiene muchos casos
- Caso tiene muchos documentos
- Caso tiene muchos eventos
- Usuario puede crear casos

7. Especificar claves primarias y foráneas:

Claves:

- UsuarioId
- ClienteId
- CasoId
- EventoId
- DocumentoId

FASE 3 — DESARROLLO

8. Crear el proyecto web en Visual Studio 2022:

Tipo de proyecto:

- ASP.NET Core Web App
- Interfaz: Razor Pages

9. Configurar la cadena de conexión: En appsettings.json apuntando a SQL Server.

10. Implementar Entity Framework Core: Permite mapear tablas como clases C#.

11. Crear el sistema de Login:

Funcionalidad incluida en el prototipo:

- Registro
- Login
- Logout

12. Crear pantallas para gestión de casos:

Incluye:

- Crear casos
- Editar
- Ver lista
- Filtrar

13. Crear pantallas para gestión de clientes:

Formulario para:

- Crear
- Listar
- Editar

14. Crear módulo de eventos y fechas importantes:

Ejemplo:

- Audiencias
- Reuniones
- Vencimientos de plazos

15. Crear módulo para documentos:

Subida de archivos:

- PDF, imágenes, Word, etc.

FASE 4 — PRUEBAS Y MEJORA

16. Pruebas unitarias y funcionales:

Validar:

- Login correcto
- Guardado de datos
- Relaciones correctas

17. Pruebas con datos reales: Simular casos reales del bufete.

18. Optimizar rendimiento:

Ajustar:

- Índices
- Consultas

19. Implementar roles: Administrador y abogado.

20. Preparar despliegue: Servidor IIS o Azure.

3. ¿Qué estructura tendría la base de datos?

Base de datos: CristhianGuevaraDB

Prefijo de tablas: CG_

3.1 Tablas principales

Tabla CG_Usuarios

Campo	Tipo
UsuarioId	int
NombreUsuario	nvarchar(50)
ContrasenaHash	nvarchar(255)
Rol	nvarchar(20)

Tabla CG_Clientes

Campo	Tipo
ClienteId	int
NombreCompleto	nvarchar(100)
Telefono	nvarchar(20)
Email	nvarchar(100)

Tabla CG_Casos

Campo	Tipo
CasoId	int
CienteId	int
UsuarioId	int
Titulo	nvarchar(100)
Descripcion	nvarchar(MAX)
FechaInicio	datetime
FechaVencimiento	datetime
Estado	nvarchar(50)

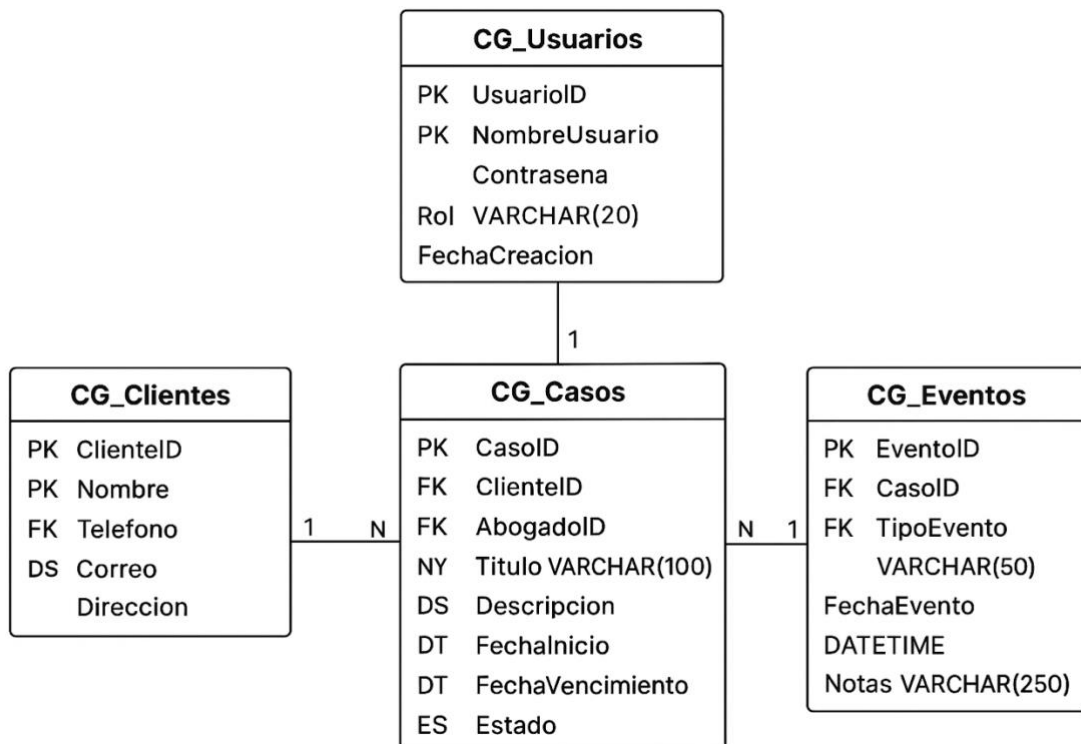
Tabla CG_Documentos

Campo	Tipo
DocumentoId	int
CasoId	int
NombreArchivo	nvarchar(200)
RutaArchivo	nvarchar(300)

Tabla CG_Eventos

Campo	Tipo
EventoId	int
CasoId	int
FechaEvento	datetime
TipoEvento	nvarchar(50)
Descripcion	nvarchar(MAX)

Diagrama Entidad Relacion agregado como extra:



RELACIONES

CG Clientes 1 — N CG Casos

Un cliente puede tener muchos casos.

Un caso pertenece a un solo cliente.

CG Usuarios 1 — N CG Casos

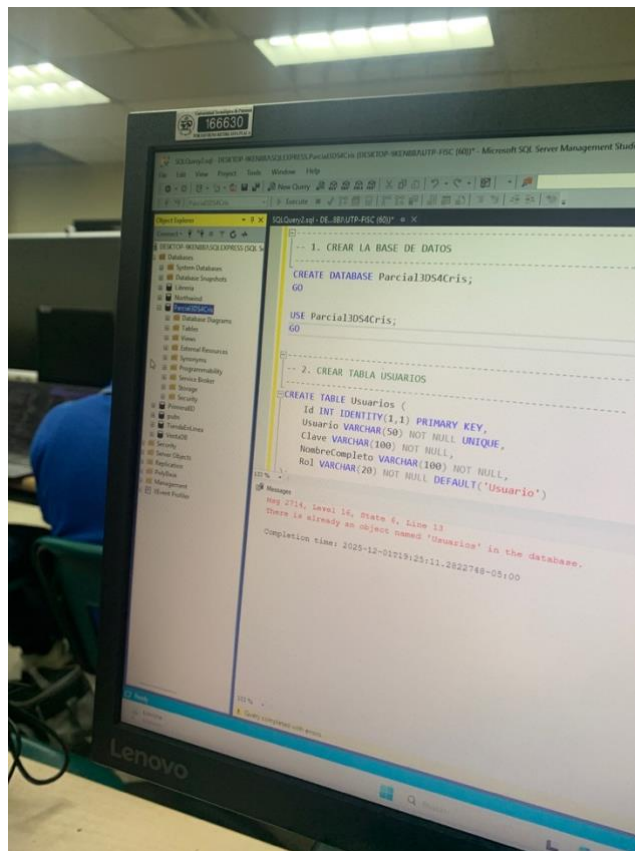
Un abogado puede llevar varios casos.

Un caso tiene solo un abogado asignado.

CG Casos 1 — N CG Eventos

Un caso puede tener muchas audiencias o reuniones.

Cada evento pertenece a un único caso.



4. ¿Por qué elegí esta estructura?

Ventajas

1. Muy organizada: separa clientes, casos, documentos y eventos.
2. Escalable: puedes añadir módulos sin romper nada.
3. Evita duplicación: cada dato se guarda una sola vez.
4. Rápida de consultar: tablas pequeñas y optimizadas.
5. Compatible con ASP.NET y EF Core directamente.

Desventajas

1. Requiere conocimiento básico de SQL.
2. Más tablas significa más mantenimiento.
3. Relaciones pueden fallar si no se controla bien.
4. Necesita un servidor SQL instalado.
5. Backup y restauración ocupan más espacio

5. ¿Qué interfaz gráfica utilizaría?

Para este proyecto seleccioné ASP.NET Core MVC (Model–View–Controller) como la interfaz gráfica principal.

Razones de la elección: ASP.NET MVC ofrece una separación clara entre la capa visual (Views), la lógica (Controllers) y los datos (Models).

Esta estructura es la más adecuada para aplicaciones profesionales que requieren organización, escalabilidad y mantenimiento a largo plazo.