**Dpto. Informática - I.E.S. La Marisma**

**MÓDULO PROYECTO**

**C.F.G.S.   
DESARROLLO DE APLICACIONES MULTIPLATAFORMA**

**FichAPP**

**Autor/es: EMILIO RAMÍREZ JOSÉ**

**Fecha: 24/042025**

**Tutor: AGUEDA MARIA LOPEZ MORENO**

| **FICHA DEL TRABAJO FINAL** | | |
| --- | --- | --- |
| **TÍTULO DEL TRABAJO** | **FichAPP** | |
| **AUTOR: EMILIO RAMÍREZ JOSÉ** | | **FECHA: 10/06/2025** |
| **TUTOR: AGUEDA MARIA LOPEZ MORENO** | | |
| **TITULACIÓN** | **CFGS DESARROLLO DE APLICACIONES MULTIPLATAFORMAS** | |
| **PALABRAS CLAVE** | **Fichaje, Control horario, Registro de jornada, Asistencia, Tiempo de trabajo, Marcaje, Entrada y salida, Puntualidad** | |
| **RESUMEN DEL PROYECTO.**   * **(Español) Esta aplicación sirve como gestor de fichas de los trabajadores de una empresa. Se encarga de determinar cuando entra un trabajador y cuando termina, si entra más tarde re calcula el tiempo que debe, también calcula el número de pausas que hace el trabajador.**   **(English) This application serves as an employee record manager. It keeps track of when a worker starts and finishes their shift. If they arrive late, it recalculates the time they owe. It also tracks the number of breaks the employee takes.** | | |

**ÍNDICE**

**1. Introducción**

1.1 Introducción a la memoria

1.2 Descripción

1.3 Objetivos generales

1.4 Beneficios

1.5 Motivaciones personales

1.6 Estructura de la memoria

**2. Estudio de viabilidad**

2.1 Introducción

2.1.1 Tipología y palabras clave

2.1.2 Descripción

2.1.3 Objetivos del proyecto

2.1.4 Clasificación de los objetivos

2.1.5 Definiciones, acrónimos y abreviaciones

2.1.6 Partes interesadas

2.1.7 Referencias

2.1.8 Documentación del proyecto

2.2 Estudio de la situación actual

2.2.1 Contexto

2.2.2 Lógica del sistema

2.2.3 Descripción física

2.2.4 Diagnóstico del sistema actual

2.2.5 Normativa y legislación

2.3 Requisitos del sistema

2.3.1 Requisitos

2.3.2 Restricciones del sistema

2.3.3 Catalogación y priorización de los requisitos

2.4 Alternativas y selección de la solución

2.4.1 Alternativa 1:

2.4.2 Alternativa 2:

2.4.x Conclusiones

2.5 Planificación del proyecto

2.5.1 Recursos del proyecto

2.5.2 Tareas del proyecto

2.5.3 Planificación temporal

2.6 Evaluación de riesgos

2.6.1 Lista de riesgos

2.6.2 Catalogación de riesgos

2.6.3 Plan de contingencia

2.7 Presupuesto

2.7.1 Estimación de coste material

2.7.2 Estimación de coste personal

2.7.3 Resumen y análisis coste beneficio

2.8 Conclusiones

2.8.1 Beneficios

2.8.2 Inconvenientes

**3. Análisis**

3.1 Introducción

3.2 Requisitos funcionales de usuarios

3.3 Requisitos no funcionales

3.4 Diagramas de casos de uso / Casos de uso

3.5 Diagramas lógigo de datos

3.6 Diagramas de clases

3.7 Diagramas de interacción

3.8 Menús de navegación

3.9 Conclusión del análisis

**4. Diseño**

4.1 Introducción

4.1.1. Selección del entorno de desarrollo

4.1.2. Selección de base de datos

4.2 Configuración de la plataforma

4.3 Capas de la aplicación

4.4 Estructura de la base de datos

4.5 Arquitectura de la aplicación

**5. Implementación**

5.1 Introducción

5.2 Codificación de las diferentes capas

5.3 Integración de las herramientas de apoyo

**6. Pruebas**

6.1 Introducción

6.2 Pruebas

6.3 Resultados Obtenidos

6.4 Conclusiones

**7. Conclusiones**

7.1 Conclusiones finales

7.2 Desviaciones temporales

7.3 Posibles ampliaciones y modificaciones

7.4 Valoración personal

**8. Bibliografía**

**9. Glosario**

**10. Anexo**

1. **INTRODUCCIÓN**

**1.1 INTRODUCCIÓN A LA MEMORIA.**

Este documento describe el desarrollo del proyecto Sistema de Gestión de Asistencia Laboral con Geolocalización, una aplicación web diseñada para automatizar el registro de entradas, salidas y pausas de los trabajadores, garantizando precisión y transparencia mediante el uso de tecnologías modernas como React.js, Firebase y APIs de geolocalización. La memoria detalla el proceso de análisis, diseño, implementación y pruebas, siguiendo las directrices del módulo de Proyecto del ciclo formativo DAM.

**1.2 DESCRIPCIÓN.**

El sistema permite:

* + Fichaje inteligente: Registro de horarios con validación por geolocalización (solo dentro del área laboral).
  + Cálculo automático: Compensación de retrasos y control de pausas activas.
  + Panel administrativo: Visualización de datos y generación de informes.Está dirigido a pequeñas y medianas empresas que buscan eliminar errores manuales y optimizar la gestión de asistencia.

**1.3 OBJETIVOS GENERALES.**

* 1. Desarrollar una aplicación web funcional que registre fichajes con geolocalización.
  2. Implementar autenticación segura por roles (empleado vs. administrador).
  3. Garantizar precisión en el cálculo de horas trabajadas y pausas.
  4. Cumplir con los requisitos del módulo Proyecto (DAM), demostrando competencias en desarrollo multiplataforma, bases de datos y seguridad.

**1.4 BENEFICIOS.**

* Para empresas:
  + Reducción de fraudes en fichajes.
  + Ahorro de tiempo en gestión manual.
  + Informes automatizados para nóminas.
* Para trabajadores:
  + - Transparencia en el registro de su jornada.
    - Accesibilidad desde cualquier dispositivo.

**1.5 MOTIVACIONES PERSONALES.**

Este proyecto nace de:

* + Interés por la tecnología aplicada a problemas reales: Solucionar un proceso tedioso (control de asistencia) con herramientas modernas.
  + Deseo de aprender: Profundizar en desarrollo web full-stack (React, Firebase) y APIs como geolocalización.
  + Alineación con DAM: Demostrar las competencias adquiridas en el ciclo (bases de datos, interfaces, seguridad).

**1.6 ESTRUCTURA DE LA MEMORIA.**

El documento se organiza en:

* Introducción (contexto y objetivos).
* Estudio de viabilidad (análisis técnico y económico).
* Análisis (requisitos y diagramas UML).
* Diseño (arquitectura y tecnologías).
* Implementación (código y funcionalidades clave).
* Pruebas (casos y resultados).
* Conclusiones (logros y mejoras futuras).
* Anexos (manuales, código relevante).

1. **ESTUDIO DE VIABILIDAD**
   1. **INTRODUCCIÓN**
      1. **TIPOLOGÍA Y PALABRAS CLAVE**

* Tipología: Aplicación de escritorio para gestión de fichajes laborales.
* Palabras clave:
  + **Control de asistencia**
  + **C#**
  + **SQL Server**
  + **Visual Studio**
  + **Aplicación Windows**
    1. **DESCRIPCIÓN**

Sistema de escritorio desarrollado en C# con .NET Framework que permite registrar la entrada, salida y pausas de los trabajadores. Utiliza SQL Server como base de datos para almacenar los registros de manera segura y eficiente.

* + 1. **OBJETIVOS DEL PROYECTO**
* Automatizar el fichaje laboral mediante una interfaz intuitiva.
* Validar la identidad del trabajador (usuario/contraseña).
* Generar informes de horas trabajadas, retrasos y pausas.
* Garantizar la persistencia de datos con SQL Server.
  + 1. **CLASIFICACIÓN DE LOS OBJETIVOS**

| Tipo | Objetivo |
| --- | --- |
| Funcional | Registro de fichajes y generación de informes |
| Técnico | Desarrollo en C# con SQL Server |
| Económico | Coste mínimo (licencias gratuitas para estudiantes) |

* + 1. **DEFINICIONES, ACRÓNIMOS Y ABREVIACIONES**
* SQL Server: Sistema de gestión de bases de datos relacionales de Microsoft.
* .NET Framework: Entorno de desarrollo para aplicaciones Windows.
* C#: Lenguaje de programación orientado a objetos.
  + 1. **PARTES INTERESADAS**
* Empresas: Para controlar la asistencia de empleados.
* Trabajadores: Usuarios finales que registran su jornada.
* Departamento de RRHH: Gestión de nóminas y horarios.
  + 1. **REFERENCIAS**
* Documentación de [Microsoft SQL Server](https://docs.microsoft.com/en-us/sql/).
* Guía oficial de [C# y .NET](https://docs.microsoft.com/en-us/dotnet/csharp/).
  + 1. **DOCUMENTACIÓN DEL PROYECTO**

Incluirá:

* Manual de usuario (instrucciones de uso).
* Manual técnico (instalación y configuración de SQL Server).
* Memoria del proyecto.
  1. **ESTUDIO DE LA SITUACIÓN ACTUAL**
     1. **CONTEXTO**

Muchas empresas aún utilizan sistemas de fichaje manuales (hojas de cálculo o relojes físicos), lo que genera errores y pérdida de tiempo. Este proyecto ofrece una solución digital integrada en el entorno Windows.

* + 1. **LÓGICA DEL SISTEMA**
* Login del trabajador (autenticación con SQL Server).
* Registro de entrada/salida (timestamp en la base de datos).
* Cálculo automático de horas trabajadas y pausas.
* Generación de informes para RRHH.
  + 1. **DESCRIPCIÓN FÍSICA**
* Front-end: Aplicación Windows Forms o WPF en C#.
* Back-end: SQL Server para almacenamiento de datos.
* Entorno de desarrollo: Visual Studio Community (gratuito).
  + 1. **DIAGNÓSTICO DEL SISTEMA ACTUAL**
* Problemas detectados:
  + Fichajes manuales inexactos.
  + Falta de integración con sistemas de nóminas.
* Oportunidades:
  + Mayor precisión y trazabilidad con SQL Server.
    1. **NORMATIVA Y LEGISLACIÓN**
* Cumple con el Reglamento General de Protección de Datos (RGPD).
* Los datos se almacenan de forma segura en SQL Server (encriptación opcional).
  1. **REQUISITOS DEL SISTEMA**
     1. **REQUISITOS**
     + Funcionales:
  + RF1: Login con autenticación en base de datos.
  + RF2: Registro de fichajes con marca de tiempo.
  + RF3: Generación de informes en PDF/Excel.
    - No funcionales:
  + RNF1: Compatibilidad con Windows 10/11.
  + RNF2: Tiempo de respuesta < 1 segundo en operaciones locales.
    1. **RESTRICCIONES DEL SISTEMA**
* Requiere SQL Server instalado en el equipo o servidor.
* Solo funciona en entornos Windows.
  + 1. **CATALOGACIÓN Y PRIORIZACIÓN DE LOS REQUISITOS**

| **Prioridad** | **Requisito** |
| --- | --- |
| Alta | RF1, RF2 |
| Media | RD3 |
| Baja | RNF1 (compatibilidad) |

* 1. **ALTERNATIVAS Y SELECCIÓN DE LA SOLUCIÓN**
     1. **ALTERNATIVA 1: C# + SQL Server**
* Ventajas:
  + Alto rendimiento y seguridad.
  + Ideal para entornos empresariales Windows.
* Inconvenientes:
  + Requiere licencia de SQL Server para producción (no para desarrollo).
    1. **ALTERNATIVA 2: C# + SQLite**
* Ventajas:
  + Base de datos ligera y portable.
* Inconvenientes:
  + Limitada para múltiples usuarios concurrentes.
    1. **CONCLUSIONES**

Se elige Alternativa 1 (SQL Server) por su escalabilidad y compatibilidad con entornos profesionales.

* 1. **PLANIFICACIÓN DEL PROYECTO**
     1. **RECURSOS DEL PROYECTO**
* Humanos: 1 desarrollador (el alumno).
* Tecnológicos: Visual Studio Community, SQL Server Express (gratis).
  + 1. **TAREAS DEL PROYECTO**
* Diseño de la base de datos (Semana 1).
* Desarrollo de la interfaz en C# (Semana 2-3).
* Conexión a SQL Server (Semana 4).
* Pruebas y documentación (Semana 5).
  + 1. **PLANIFICACIÓN TEMPORAL**

| **Tarea** | **Duración** |
| --- | --- |
| Análisis y diseño de BD | 1 semana |
| Desarrollo Front-End | 2 semanas |
| Integración con SQL Server | 1 semanas |
| Pruebas y depuración | 1 semana |

* 1. **EVALUACIÓN DE RIESGOS**
     1. **LISTA DE RIESGOS**
* Errores en la conexión a SQL Server.
* Problemas de rendimiento con grandes volúmenes de datos.
  + 1. **CATALOGACIÓN DE RIESGOS**

| **Riesgo** | **Probabilidad** | **Impacto** |
| --- | --- | --- |
| Conexión a SQL Server | Media | Alto |
| Rendimiento | Baja | Medio |

* + 1. **PLAN DE CONTINGENCIA**
* Para conexiones: Usar try-catch y mensajes de error descriptivos.
* Para rendimiento: Optimizar consultas SQL con índices.
  1. **PRESUPUESTO**
     1. **ESTIMACIÓN DE COSTE MATERIAL**
* Software:
  + Visual Studio Community: 0€.
  + SQL Server Express: 0€ (para desarrollo).
* Hardware: Ordenador estándar (ya disponible).
  + 1. **ESTIMACION DE COSTE PERSONAL**
* Horas estimadas: 50 horas (valoradas a 15€/hora como becario) = 750€.
  + 1. **RESUMEN Y ANÁLISIS COSTE BENEFICIO**
* Coste total: 750€ (solo mano de obra).
* Beneficios:
  + Elimina costes de sistemas de fichaje físico.
  + Reducción de errores en nóminas.
  1. **CONCLUSIONES**
     1. **BENEFICIOS**
* Sistema robusto y profesional con SQL Server.
* Fácil integración en entornos Windows empresariales.
  + 1. **INCONVENIENTES**
* Dependencia de Windows y SQL Server.
* Curva de aprendizaje para administrar la base de datos.

1. **ANÁLISIS**
   1. **INTRODUCCIÓN**

Este apartado detalla el análisis del sistema de gestión de fichajes, incluyendo requisitos funcionales y no funcionales, diagramas UML y el modelo de datos. Se ha realizado un estudio exhaustivo de las entidades y sus relaciones para garantizar que el sistema cumpla con las necesidades de la empresa y los trabajadores.

* 1. **REQUISITOS FUNCIONALES DE USUARIOS**

Actor: Empleado

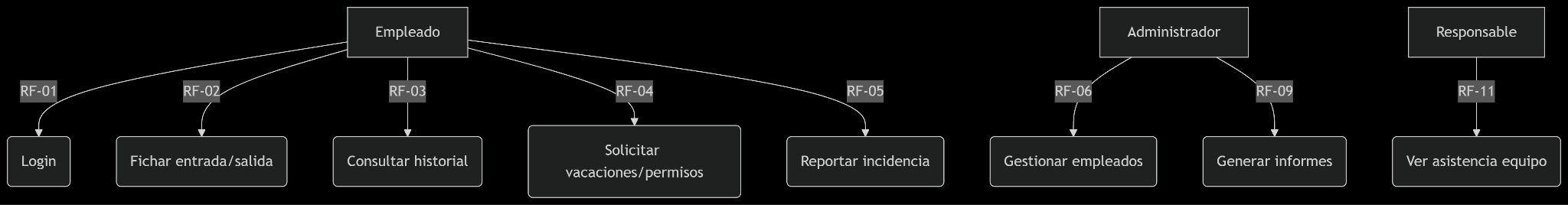
* RF-01: Iniciar sesión con credenciales (usuario/contraseña).
* RF-02: Registrar entrada/salida con marca de tiempo y método de fichaje (tarjeta, huella, etc.).
* RF-03: Consultar su historial de fichajes y horarios asignados.
* RF-04: Solicitar vacaciones o permisos.
* RF-05: Reportar incidencias (retrasos, ausencias).

Actor: Administrador (RRHH)

* RF-06: Gestionar altas, bajas y modificaciones de empleados.
* RF-07: Asignar horarios y departamentos.
* RF-08: Aprobar/rechazar solicitudes de vacaciones o permisos.
* RF-09: Generar informes de asistencia, incidencias y horas extras.
* RF-10: Configurar métodos de fichaje (ej: habilitar huella digital).

Actor: Responsable de Departamento

* RF-11: Visualizar asistencia de su equipo.
* RF-12: Validar incidencias reportadas por empleados.
  1. **REQUISITOS NO FUNCIONALES**
* RNF-01: Compatibilidad con Windows 10/11 y SQL Server 2019+.
* RNF-02: Tiempo de respuesta < 2 segundos en operaciones CRUD.
* RNF-03: Encriptación de datos sensibles (contraseñas, DNI).
* RNF-04: Interfaz intuitiva (máximo 3 clics para fichar).
* RNF-05: Backup automático diario de la base de datos.
  1. **DIAGRAMAS DE CASOS DE USO/ CASOS DE USO**

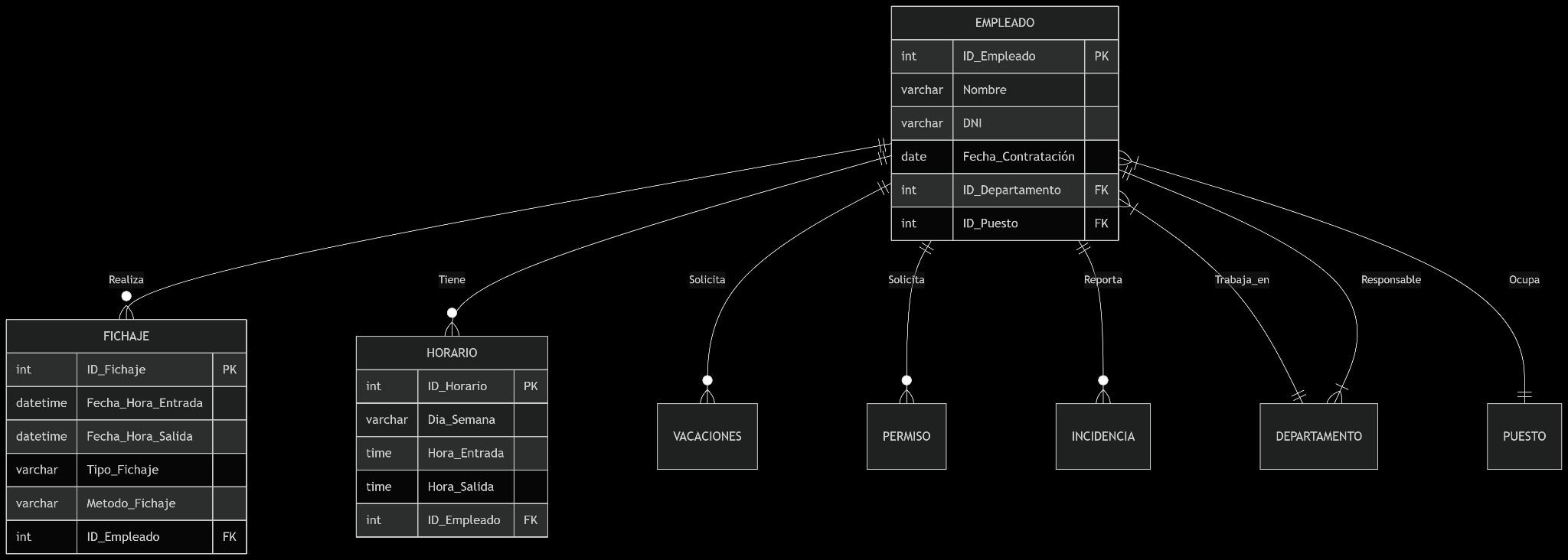
****

**Caso de uso destacado: "Registrar fichaje"**

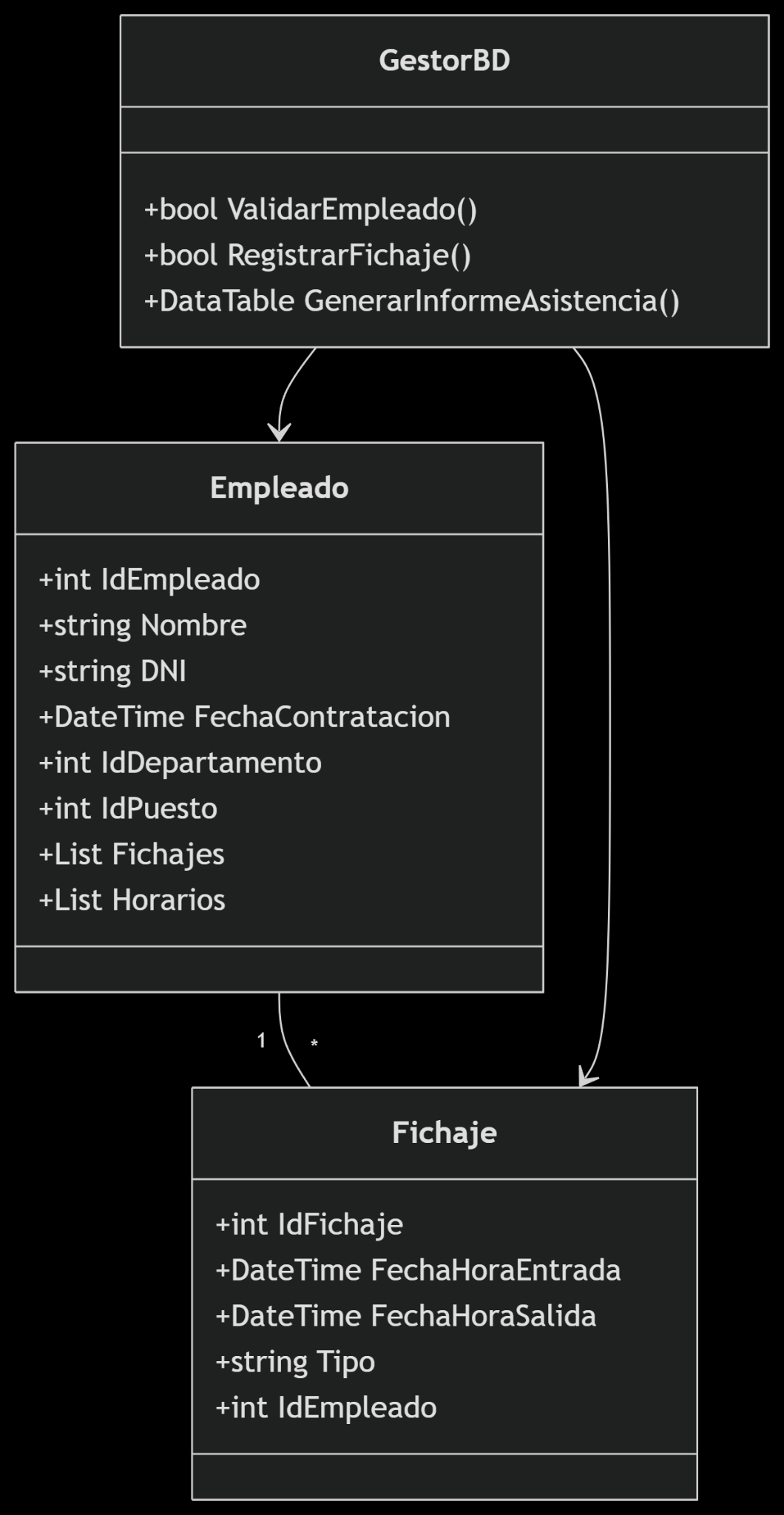
* Precondición: Empleado autenticado.
* Flujo básico:

1. El Empleado selecciona "Fichar entrada".
2. El sistema valida el método de fichaje (ej: huella).
3. Se registra Fecha\_Hora\_Entrada y Ubicación Fichaje en la tabla FICHAJES.

* Flujo alternativo:
  + Si el empleado está fuera del horario asignado (HORARIO), se marca como "Retraso".
  1. **DIAGRAMAS LÓGICO DE DATOS**

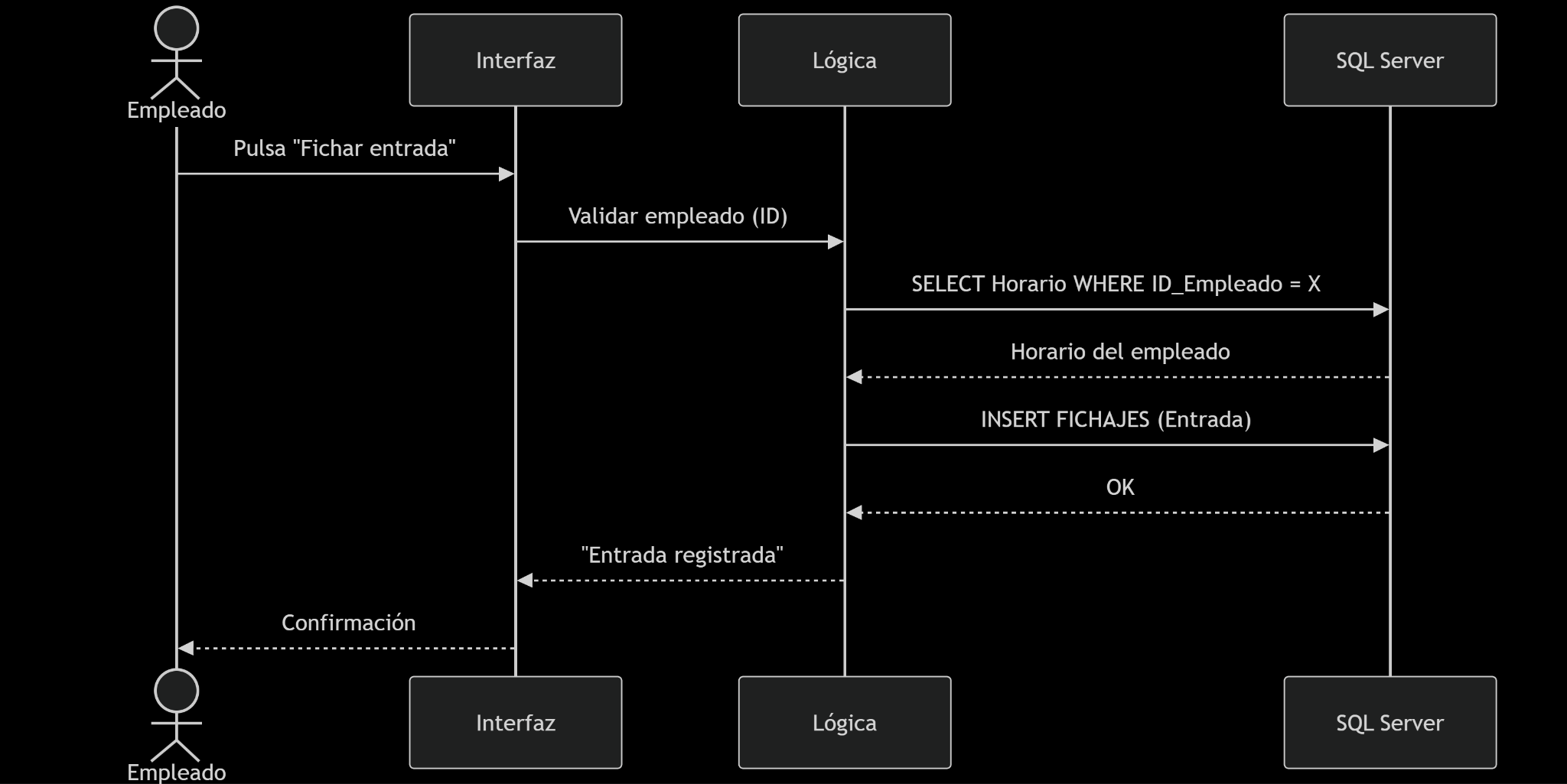
****

* 1. **DIAGRAMAS DE CLASES**

****

* 1. **DIAGRAMAS DE INTERACCIÓN**

Secuencia: "Registrar fichaje"

****

* 1. **CONCLUSIÓN DE ANÁLISIS**
* Cobertura de requisitos: El sistema satisface las necesidades de empleados, administradores y responsables.
* Modelo de datos robusto: Las entidades y relaciones cubren todos los escenarios (fichajes, horarios, incidencias).
* Tecnologías adecuadas: C# y SQL Server garantizan escalabilidad y seguridad.

1. **DISEÑO**
   1. **INTRODUCCIÓN**

En esta sección se define la arquitectura técnica del sistema, las tecnologías seleccionadas y la estructura de la base de datos. El diseño se alinea con los requisitos analizados previamente y garantiza escalabilidad, seguridad y facilidad de mantenimiento.

* + 1. **SELECCIÓN DEL ENTORNO DE DESARROLLO**
* Lenguaje de programación: C# (por su integración con Windows y SQL Server).
* Entorno de desarrollo: Visual Studio 2022 (Community Edition, gratuita).
  + Ventajas:
    - Soporte para .NET Framework y .NET Core.
    - Herramientas integradas para diseño de interfaces (Windows Forms/WPF).
    - Depurador avanzado y conectividad con SQL Server.
* Control de versiones: Git + GitHub (para gestión del código fuente).
  + 1. **SELECCIÓN DE BASE DE DATOS**
* Motor de base de datos: Microsoft SQL Server 2019 (Express Edition).
  + Ventajas:
    - Compatibilidad nativa con C# (Entity Framework).
    - Soporte para transacciones ACID y encriptación de datos.
    - Licencia gratuita para desarrollo (SQL Server Express).
* Alternativas consideradas y descartadas:
  + MySQL: Menor integración con el ecosistema Microsoft.
  + SQLite: No soporta múltiples usuarios concurrentes eficientemente.
  1. **CONFIGURACIÓN DE LA PLATAFORMA**
* Requisitos mínimos:
  + Hardware:
    - Procesador x64 de 2 GHz.
    - 4 GB de RAM.
    - 10 GB de almacenamiento (para SQL Server).
  + Software:
    - Windows 10/11.
    - .NET Framework 4.8 o .NET 6.0.
* Configuración inicial:
  + Instalar Visual Studio con los workloads:
    - ".NET desktop development".
    - "Data storage and processing" (para SQL Server).
  + Instalar SQL Server Management Studio (SSMS) para administrar la base de datos.
  1. **CAPAS DE LA APLICACIÓN**

| **Capa** | **Tecnologías** | **Responsabilidad** |
| --- | --- | --- |
| **Presentación** | **Windows Forms/WPF** | **Interfaz de usuario (formularios de fichaje, login, informes).** |
| **Lógica** | **C# + Entity Framework** | **Validación de reglas de negocio (ej: cálculo de horas extras).** |
| **Datos** | **SQL Server + Stored Procedures** | **Almacenamiento y recuperación de datos.** |

* 1. **ESTRUCTURA DE LA BASE DE DATOS**

Tablas principales (basadas en el modelo E-R):

* **EMPLEADO:**

| **CREATE TABLE EMPLEADO (**  **ID\_Empleado INT IDENTITY(1,1) PRIMARY KEY,**  **Nombre NVARCHAR(50) NOT NULL,**  **Apellido1 NVARCHAR(100) NOT NULL,**  **Apellido2 NVARCHAR(100) NOT NULL,**  **DNI\_NIF NVARCHAR(20) NOT NULL UNIQUE,**  **Fecha\_Nacimiento DATE,**  **Direccion NVARCHAR(200),**  **Telefono NVARCHAR(20),**  **Email NVARCHAR(100),**  **Contrasenia NVARCHAR(8),**  **Fecha\_Contratacion DATE NOT NULL,**  **ID\_Departamento INT NULL,**  **ID\_Puesto INT NOT NULL**  **);** |
| --- |

* **FICHAJE:**

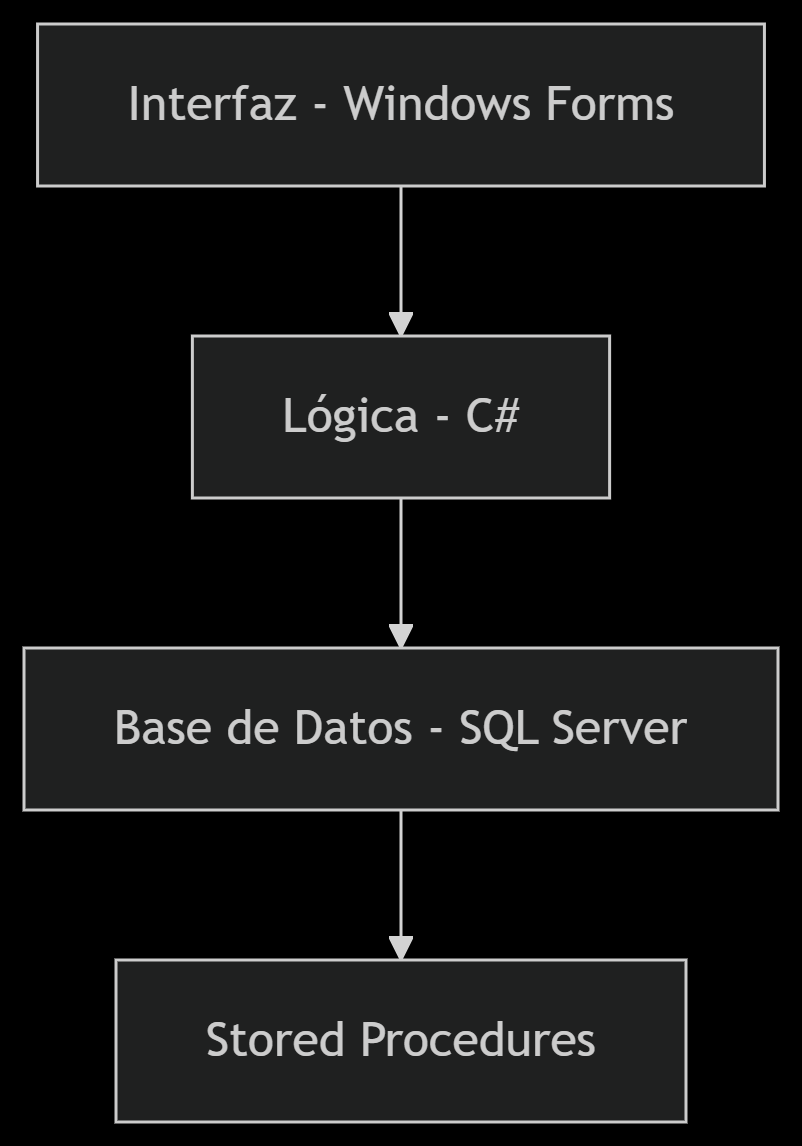
| **CREATE TABLE FICHAJE (**  **ID\_Fichaje INT IDENTITY(1,1) PRIMARY KEY,**  **ID\_Empleado INT NOT NULL,**  **Fecha\_Hora\_Entrada DATETIME NOT NULL,**  **Fecha\_Hora\_Salida DATETIME NULL,**  **Tipo\_Fichaje NVARCHAR(10) NOT NULL CHECK (Tipo\_Fichaje IN ('Entrada', 'Salida')),**  **Metodo\_Fichaje NVARCHAR(10) NOT NULL CHECK (Metodo\_Fichaje IN ('Tarjeta', 'Huella', 'Facial', 'Web', 'App')),**  **Ubicacion\_Fichaje NVARCHAR(100),**  **Estado NVARCHAR(20) NOT NULL CHECK (Estado IN ('Normal', 'Retraso', 'Ausencia', 'Horas Extra', 'Conflicto')),**  **CONSTRAINT FK\_Fichaje\_Empleado FOREIGN KEY (ID\_Empleado) REFERENCES EMPLEADO(ID\_Empleado)**  **);** |
| --- |

* **HORARIOS:**

| **CREATE TABLE HORARIO (**  **ID\_Horario INT IDENTITY(1,1) PRIMARY KEY,**  **ID\_Empleado INT NOT NULL,**  **Dia\_Semana NVARCHAR(10) NOT NULL CHECK (Dia\_Semana IN ('Lunes', 'Martes', 'Mi�rcoles', 'Jueves', 'Viernes', 'S�bado', 'Domingo')),**  **Hora\_Entrada TIME NOT NULL,**  **Hora\_Salida TIME NOT NULL,**  **Flexible BIT DEFAULT 0,**  **CONSTRAINT UQ\_Horario\_Empleado\_Dia UNIQUE (ID\_Empleado, Dia\_Semana),**  **CONSTRAINT FK\_Horario\_Empleado FOREIGN KEY (ID\_Empleado) REFERENCES EMPLEADO(ID\_Empleado)**  **);** |
| --- |

Relaciones:

* Claves foráneas según el modelo E-R proporcionado.
* Índices en campos de consulta frecuente (ej: ID\_Empleado en FICHAJE)
  1. **ARQUITECTURA DE LA APLICACIÓN**

****

Flujo:

* El usuario interactúa con un formulario de Windows Forms.
* La capa de lógica (C#) valida los datos y ejecuta consultas mediante Entity Framework.
* SQL Server procesa las consultas y devuelve resultados

1. **IMPLEMENTACIÓN**
   1. **PATRÓN MODELO VISTA CONTROLADOR.**
   2. **CODIFICACIÓN DE LAS DIFERENTES CAPAS**
   3. **INTEGRACIÓN DE LAS HERRAMIENTAS DE APOYO**
2. **PRUEBAS**
   1. **INTRODUCCIÓN**
   2. **PRUEBAS**
   3. **RESULTADOS OBTENIDOS**
   4. **CONCLUSIONES**
3. **CONCLUSIONES**
   1. **CONCLUSIONES FINALES**
   2. **DESVIACIONES TEMPORALES**
   3. **POSIBLES AMPLIACIONES Y MODIFICACIONES**
   4. **VALORACION PERSONAL**
4. **BIBLIOGRAFÍA**
5. **GLOSARIO**
6. **ANEXO**