Universidad Internacional San Isidro Labrador

Facultad de Ingeniería de Sistemas

Curso:

Programación Avanzada

Título del Proyecto:

TaskFlow

(Sistema de Gestión de Tareas y Proyectos)

Estudiante:

Diego Abarca Mejía

Profesora:

Estefania Boza Villalobos

Cuatrimestre II del 2025

Contenido

Índice de Figuras 3

Objetivo General 4

Objetivos Específicos 4

Justificación del Proyecto 4

Alcances Esperados 4

Requerimientos del Proyecto 5

Tecnológicos 5

Funcionales 5

No Funcionales 5

RESUMEN TÉCNICO 6

Funcionalidades Clave: 6

Público Objetivo 6

Problema Resuelto 6

Descripción de Vistas 7

Explicación Detallada del Uso de HTML y CSS en el programa: 7

Tecnologías del backend y base de datos 8

Tecnologías elegidas y ¿Por qué? 8

Funcionalidades completadas 9

Código de Conexión a la Base de Datos 9

Conclusiones 11

Enlaces 11

Índice de Figuras

Figura no. 1: Diagrama entidad relación 9

Objetivo General

Desarrollar una aplicación web funcional para la gestión eficiente de tareas y proyectos, permitiendo a los usuarios organizar, priorizar y realizar seguimiento de sus actividades mediante una interfaz intuitiva, con autenticación de usuarios, almacenamiento en base de datos y generación de reportes.

Objetivos Específicos

1. Diseñar y desarrollar el Frontend utilizando HTML, CSS y tecnologías complementarias para garantizar una interfaz amigable y responsiva.
2. Implementar el Backend en C# con funcionalidades como registro de usuarios, CRUD de tareas y generación de reportes.
3. Configurar una base de datos SQL para almacenar la información de usuarios, tareas y proyectos, asegurando integridad y seguridad de los datos.
4. Establecer la conexión entre Frontend, Backend y Base de Datos para garantizar un flujo de datos eficiente.

Justificación del Proyecto

En el ámbito académico y profesional, la organización de tareas es fundamental para mejorar la productividad. Una herramienta web que permita gestionar actividades, asignar prioridades y recibir recordatorios ayudará a los usuarios a optimizar su tiempo. Este proyecto demostrará habilidades en desarrollo full-stack, integrando tecnologías como HTML, CSS, C# y SQL, fundamentales en el desarrollo de aplicaciones web modernas.

Alcances Esperados

Registro y Autenticación de Usuarios

- Los usuarios podrán crear cuentas e iniciar sesión de manera segura.
- Se almacenarán credenciales encriptadas en la base de datos.

Gestión de Tareas (CRUD)

- Crear, editar, eliminar y marcar tareas como completadas.
- Asignar fechas límite, prioridades y categorías.

Visualización y Filtrado de Tareas

- Mostrar tareas en listas o tableros interactivos.

- Filtrar por estado (pendiente, en progreso, completado).

Requerimientos del Proyecto

Tecnológicos

- Frontend: HTML, CSS, Bootstrap.
- Backend: C# ASP.NET Core o MVC.
- Base de Datos: SQL Server o MySQL.
- Entorno de Desarrollo: Visual Studio / Visual Studio Code.

Funcionales

- Sistema de login y registro.
- CRUD completo de tareas.
- Interfaz intuitiva y responsiva.

No Funcionales

- Seguridad básica (protección contra SQL Injection).
- Documentación del código y manual de usuario.

RESUMEN TÉCNICO

TaskFlow es una aplicación web desarrollada con ASP.NET Core MVC que permite a los usuarios organizar tareas, priorizarlas y realizar seguimiento de su progreso. Incluye autenticación básica, CRUD de tareas y un formulario de contacto.

Funcionalidades Clave:

- Login/Registro: Autenticación de usuarios con validación.
- Dashboard: Visualización y gestión de tareas (crear, editar, eliminar).
- Contacto: Formulario para enviar mensajes al soporte técnico.

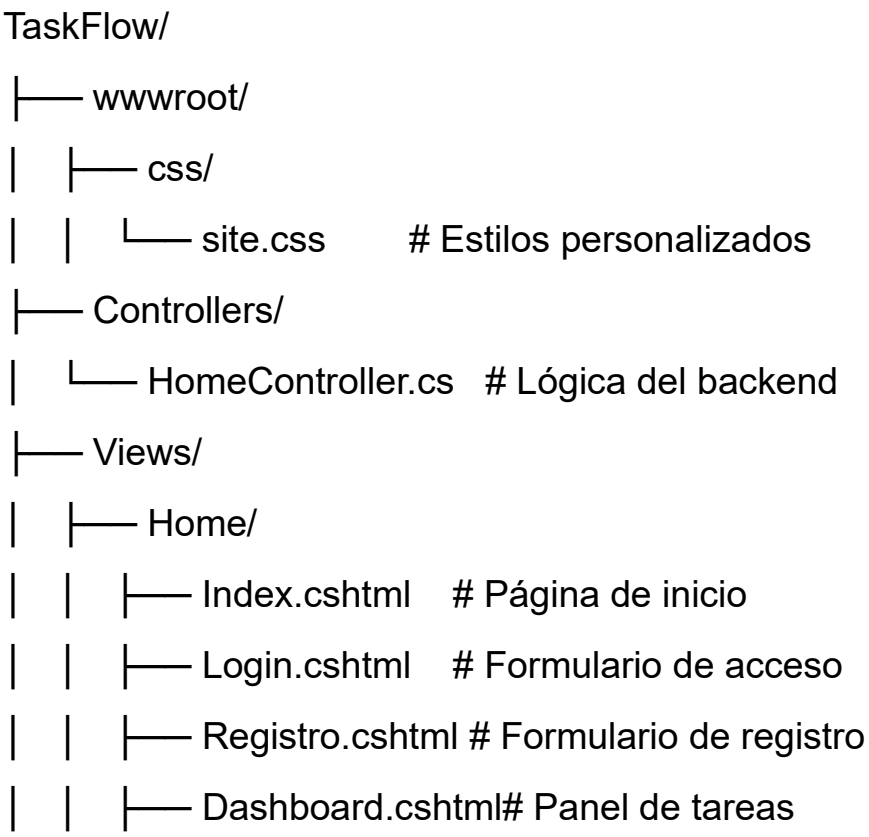
Público Objetivo

- Estudiantes
- Profesionales o cualquier persona que necesite organizar sus actividades diarias.

Problema Resuelto

Centraliza la gestión de tareas en una plataforma accesible desde cualquier dispositivo, mejorando la productividad personal.

Diagrama no. 1: Estructura del sistema



```
│   │   └── Contacto.cshtml # Página de contacto
│   └── Shared/
│       └── _Layout.cshtml  # Plantilla común
└── Program.cs              # Configuración inicial
```

Descripción de Vistas

- Página de Inicio (Index.cshtml): Presenta el sistema con un botón de acceso al login.
- Diseño minimalista con menú de navegación.
- Login (Login.cshtml): Formulario con campos: email y contraseña.
- Validación básica en el backend.
- Dashboard (Dashboard.cshtml): Tabla interactiva con lista de tareas.
- Botones para editar/eliminar tareas.
- Contacto (Contacto.cshtml): Formulario para enviar mensajes al equipo de soporte.
- Muestra confirmación de envío.

Explicación Detallada del Uso de HTML y CSS en el programa:

Estructura Semántica Avanzada: El proyecto utiliza etiquetas HTML semánticas para mejorar accesibilidad

Técnicas aplicadas:

- aria-label para accesibilidad
- role para definir zonas interactivas
- Preconexión a recursos externos

Formularios con Validación Nativa: Los formularios utilizan características modernas de HTML

1. **Características destacables:**
    - Validación con pattern usando expresiones regulares
    - Atributo required para campos obligatorios
    - Placeholders descriptivos
2. **CSS: Técnicas Modernas de Estilizado**
    - Sistema de Diseño Modular con Variables CSS (Archivo site.css con diseño sistemático).
    - Layouts Flexibles con Grid y Flexbox:
        - Sistema de grid para el dashboard

        - Técnicas responsive:
        - Unidades relativas (rem, %, vw/vh)
        - Media queries con breakpoints variables
        - Grid areas para reorganización en móviles
    - Animaciones y Microinteracciones
3. **Integración Avanzada HTML + CSS**
    - Componente de Tareas con Estado
    - Dark Mode con Variables CSS

Tecnologías del backend y base de datos

- Backend: ASP.NET Core 6.0
- ORM: Entity Framework Core
- Base de datos: SQL Server Express

Tecnologías elegidas y ¿Por qué?

Backend: ASP.NET Core con C#

Razón:

- Alto rendimiento y escalabilidad para aplicaciones web.
- Integración nativa con Entity Framework Core.
- Soporte oficial de Microsoft (actualizaciones frecuentes).

ORM: Entity Framework Core

Razón:

- Simplifica el acceso a datos sin escribir SQL manual.
- Migraciones automáticas (sincroniza BD con modelos C#).
- Soporte para múltiples motores (SQL Server, MySQL, PostgreSQL).

Base de Datos: SQL Server Express

Razón:

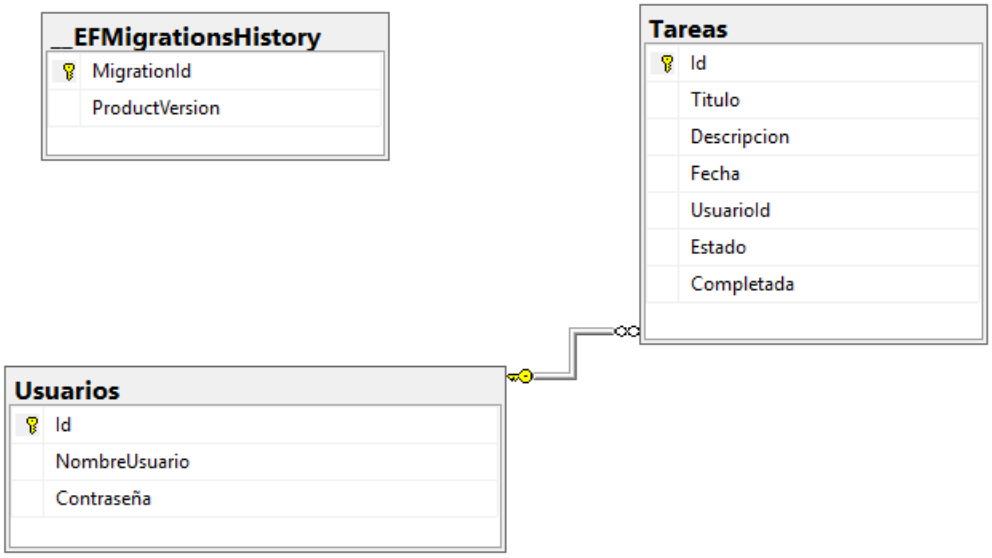
- Gratis para desarrollo (localdb incluido en Visual Studio).
- Compatibilidad total con Entity Framework.
- Herramientas de gestión (SSMS, Azure Data Studio).

Frontend: HTML + CSS+ JavaScript

Razón:

- Ligero y rápido (sin frameworks pesados).
- Fácil mantenimiento para proyectos académicos.

Figura no. 1: Diagrama entidad relación



Relaciones:

- 1:N (Un usuario tiene muchas tareas).
- Claves:
  - PK (Primary Key): Id en ambas tablas.
  - FK (Foreign Key): UsuarioId en TAREA.

Funcionalidades completadas

- Conexión a BD: Configuración de AppDbContext y cadena de conexión.
- Modelado de datos: Clases Usuario y Tarea con relaciones.
- Migraciones: Creación inicial de tablas en SQL Server.
- CRUD básico: Crear/leer tareas (desde código, aún no en UI).

Código de Conexión a la Base de Datos

Ubicación del Código

- Data/AppDbContext.cs → Define la estructura de la base de datos.
- appsettings.json → Contiene la cadena de conexión.
- Program.cs → Configura la inyección de dependencias.

Configuración del DbContext

```
using Microsoft.EntityFrameworkCore;
using TaskFlow.Models;


namespace TaskFlow.Data
{
    public class AppDbContext : DbContext
    {
        // Constructor que recibe opciones de configuración
        public AppDbContext(DbContextOptions<AppDbContext> options) :
base(options)
        {
        }


        // Tablas en la base de datos
        public DbSet<Usuario> Usuarios { get; set; }
        public DbSet<Tarea> Tareas { get; set; }
    }
}
```

¿Qué hace?

- Hereda de DbContext (clase base de Entity Framework Core).
- DbSet<T> representa una tabla en la base de datos.
- El constructor permite inyectar la configuración (cadena de conexión).

Cadena de Conexión:

```
{
```

```
  "ConnectionStrings": {
    "DefaultConnection":
"Server=(localdb)\\mssqllocaldb;Database=TaskFlowDB;Trusted_Connection=True;"
  }
}
```

Explicación:

- Server=(localdb)\\mssqllocaldb → Usa SQL Server LocalDB (incluido en Visual Studio).
- Database=TaskFlowDB → Nombre de la base de datos (se crea automáticamente).
- Trusted_Connection=True → Autenticación con credenciales de Windows (sin usuario/contraseña explícitos).

Conclusiones

Estado Actual:

- Base de datos funcional con tablas relacionadas.
- Backend conectado a SQL Server.
- Estructura limpia (Models, Data, Migraciones).

Falta por Desarrollar:

1. UI para CRUD:
    - Forms para crear/editar tareas.
    - Vista de listado con filtros.
2. Autenticación:
    - Login con Identity.
3. Validaciones:
    - Campos requeridos y formatos (email, fechas).

Enlaces

Vídeo del proyecto:

Grabación de
pantalla 2025-08-10