Universidad Fidélitas

Curso

Programación Avanzada

Estudiantes

Karina Picado Aguilar

Steve Barboza Picado

Sergio cabezas Martínez

Profesor

Bryan Cerdas Salas

Avance I

viernes 18 Julio, 2025.

Base de Datos

Diagrama Entidad-Relación

Script Base de Datos

```
-- Creación de la base de datos
CREATE DATABASE ProyectoFinalTareaDB;
GO
USE ProyectoFinalTareaDB;
GO

-- Creacion Tabla de usuarios
CREATE TABLE Usuario (
    IdUsuario INT IDENTITY(1,1) PRIMARY KEY,
    Nombre NVARCHAR(100) NOT NULL,
    Email NVARCHAR(100) NOT NULL UNIQUE
);
```

```
-- Creacion Tabla de tareas
CREATE TABLE Tarea (
    IdTarea INT IDENTITY(1,1) PRIMARY KEY,
    Titulo NVARCHAR(200) NOT NULL,
    Descripcion NVARCHAR(1000),
    Prioridad NVARCHAR(10) NOT NULL CHECK (Prioridad IN ('Alta', 'Media', 'Baja')),
    Estado NVARCHAR(20) NOT NULL CHECK (Estado IN ('Pendiente', 'En Proceso', 'Finalizada', 'Fallida')),
    FechaCreacion DATETIME NOT NULL DEFAULT GETDATE(),
    FechaEjecucion DATETIME,
    IdUsuario INT NOT NULL,
    CONSTRAINT FK_Tarea_Usuario FOREIGN KEY (IdUsuario) REFERENCES Usuario(IdUsuario)
);

-- Creacion Tabla de logs de ejecución de tareas
CREATE TABLE LogTarea (
    IdLog INT IDENTITY(1,1) PRIMARY KEY,
    IdTarea INT NOT NULL,
    FechaInicio DATETIME NOT NULL DEFAULT GETDATE(),
    FechaFin DATETIME,
    Estado NVARCHAR(20) NOT NULL CHECK (Estado IN ('En Proceso', 'Finalizada', 'Fallida')),
    MensajeError NVARCHAR(1000),
    CONSTRAINT FK_LogTarea_Tarea FOREIGN KEY (IdTarea) REFERENCES Tarea(IdTarea)
);
```

```
-- Datos de ejemplo
INSERT INTO Usuario (Nombre, Email) VALUES ('Admin', 'admin@gmail.com');
INSERT INTO Usuario (Nombre, Email) VALUES ('Soporte', 'soporte@gmail.com');

INSERT INTO Tarea (Titulo, Descripcion, Prioridad, Estado, FechaEjecucion, IdUsuario)
VALUES
('Enviar correo', 'Envía un correo, 'Alta', 'Pendiente', '2025-07-20', 1),
('Generar reporte', 'Reporte de ventas', 'Media', 'Pendiente', '2025-07-21', 2);

INSERT INTO LogTarea (IdTarea, Estado, MensajeError)
VALUES (1, 'En Proceso', NULL);
```

Enlace al repositorio de GitHub

https://github.com/steveebpe27/ProyectoFinalGrupo2.git

Arquitectura de la Solución

La solución propuesta para el sistema de ejecución de un queue de tareas estará construida bajo una arquitectura en capas, en este caso el diseño Modelo-Vista-Controlador (MVC), este modelo es muy adoptado en aplicaciones web modernas de hoy en día, por su capacidad de separar responsabilidades y facilitar el mantenimiento y la escalabilidad de los sistemas.

Esta arquitectura se implementará sobre la plataforma ASP.NET con modelo MVC antes mencionado, integrando Entity Framework para la gestión de datos y SQL Server como motor de base de datos relacional.

Las tecnologías que serán utilizadas en el proyecto serán las siguientes:

- ASP.NET MVC: Framework para el desarrollo web bajo el patrón Modelo-Vista-Controlador.
- Entity Framework: ORM para mapear objetos C# a la base de datos relacional.
- SQL Server: Motor de base de datos relacional para el almacenamiento persistente de datos.

- Razor: Motor de vistas para construir interfaces de usuario dinámicas.
- C#: Lenguaje principal para la lógica de negocio y controladores.

Los componentes y capas de arquitectura que se estarán utilizando en el proyecto serían las siguientes:

- Capa de Presentación (Vistas - Razor Views)

  Esta capa será la encargada de la interacción directa con el usuario final. Estará compuesta por páginas web que permitirán a los usuarios crear, editar y monitorear tareas, así como visualizar el estado actual de la cola y el historial de ejecución. En este caso el proyecto estará utilizando Razor como motor de vistas para HTML y C#.

- Capa de Controladores (Controllers)

  Los controladores estarán actuando como intermediarios entre la capa de presentación y la lógica de negocio. Recibiendo las solicitudes HTTP del usuario, ejecutando la lógica correspondiente como validaciones, procesamiento de datos y control de flujos, determinando qué vista debe ser mostrada o qué respuesta debe devolverse.

  En esta solución, se tendrán controladores como TaskController para la gestión de tareas y QueueController para la administración y monitoreo de la cola de ejecución.

- Capa de Modelo (Models y Entity Framework)

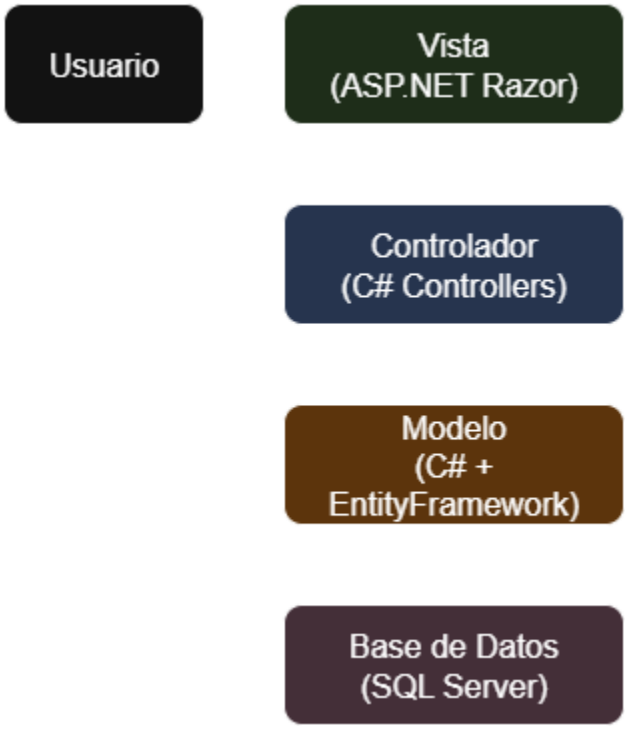
  En esta capa es donde se definirá cómo lucen y se comportaran los datos principales del sistema, como los usuarios, las tareas y los logs de tareas. Para trabajar con estos datos se usará Entity Framework, la cual es una herramienta que nos permitirá escribir y leer información de la base de datos usando objetos de C#, en lugar de tener que escribir instrucciones complicadas en SQL.

- Capa de Datos (SQL Server)

La base de datos SQL Server almacenará de manera persistente todas las entidades relevantes del sistema y gracias a las relaciones entre las tablas y claves foráneas se asegurará la consistencia y fiabilidad de los datos.

La base de datos incluirá las tablas necesarias para usuarios, tareas y logs de ejecución, permitiéndonos así una trazabilidad bastante completa de las operaciones del sistema.

Diagrama de la Arquitectura



Justificación de la Arquitectura

La elección de ASP.NET MVC junto con Entity Framework para el proyecto se da con la necesidad de construir un sistema web modular, organizado y a su vez seguro. El uso de capas con el modelo MVC permitirá:

1. Facilitar el mantenimiento y pruebas, ya que cada capa puede ser modificada o probada de manera independiente.

2. Escalar el sistema de manera sencilla, permitiendo la inclusión de nuevas funcionalidades como autenticación de usuarios y generación de reportes, sin tener que afectar la estructura central del proyecto.

3. Reutilizar código y componentes, maximizando la eficiencia del desarrollo y disminuyendo el riesgo de cometer errores.

4. Separar responsabilidades, asegurando que la lógica de negocio, el acceso a datos y la presentación se mantengan independientes el uno con el otro.