Desarrollo Backend para aplicación de gestión empresarial en proyectos de construcción industrial en ALAMO GLOBAL

Luis Carlos Villan Jurado

Trabajo de grado para optar el título de ingeniero de telecomunicaciones

Director

Yuli Andrea Álvarez

Magister en Ingeniería Eléctrica

Docente de la facultad de ingeniería de telecomunicaciones

Universidad Santo Tomás, Bucaramanga

División de ingenierías y arquitectura

Ingeniería de telecomunicaciones

2024

Dedicatoria

Dedico este logro a mi madre, quien ha sido partícipe y apoyo fundamental en este proceso desde el primer día. A mis hermanos, que han sido guías esenciales en mi desarrollo académico y profesional. También a mi familia y amigos, quienes han sido clave en mi vida al impulsarme a creer en lo que soy capaz y ayudarme a alcanzar este logro

Agradecimientos

Deseo expresar mi sincero agradecimiento a la empresa ALAMO GLOBAL por integrarme a su equipo y permitirme realizar este proyecto. Me llevo las enseñanzas y la experiencia compartida que fueron fundamentales durante el desarrollo de mis prácticas profesionales.

Quiero también agradecer y reconocer a la profesora Yuli Andrea Álvarez por su apoyo, dedicación, tolerancia y compromiso durante este proceso. Agradezco igualmente a los demás profesores de la facultad que fueron parte de mi crecimiento académico y profesional.

Contenido

Introducción……...12

1.**Desarrollo backend para aplicación de gestión empresarial en proyectos de construcción industrial**…….. 14

1.1 Planteamiento del problema ……14

1.2 Justificación……. 15

1.3 Objetivos…….. 16

1.3.1 Objetivo general…… 16

1.3.2 Objetivos específicos……16

2. Marco referencial……... 17

2.1 Marco teórico……... 17

2.1.1 Concepto de marco teórico……17

2.2 Marco conceptual……21

2.3 Marco legal…….. 28

3. Método…….. 30

4. Resultados…….. 37

5. Discusión…….....47

6. Conclusiones…….. 48

Referencias…….50

Lista de figuras

Figura 1: *Representación del diagrama de flujo que permite ver la dinamica de creación y finalización de un proyecto en la aplicacion*……..……...37

Figura 2: *Draft del funcionamiento de la UI con el software FIGMA*……..38

Figura 3. *Esquema del diagrama entidad relación realizado en Draw.io*……..39

Figura 4. *Representacion de las 4 capas en visual studio*……..40

Figura 5: *Creación del metodo de extensión que permite implementar JWT en el API*…….40

Figura 6. *Codigo SQL que representa los SPs*……41

Figura 7. *Configuración del JOB en SQL Server Agent*……..41

Figura 8. *Respuesta del endpoint correspondiente a los proyectos en swagger*…….42

Figura 9. *Respuesta del endpoint de login usando ThunderClient en Visual Studio Code*…….42

Figura 10. *Creacion de la vista login*…….……43

Figura 11. *Creacion de la vista de sign up*……..43

Figura 12. *Creacion de la vista home con rol de Admin*……44

Figura 13. *Creacion de offcanvas dinámic*……44

Figura 14. *Creacion del overall de Cwas por proyecto*……..45

Figura 15. *Creacion modal para editar un Cwa*……45

Figura 16. *Creacion de la vista de reglas de credito por proyecto divididas por disciplina*…..46

Figura 17. *Modal para agregar una regla de credito por proyecto por disciplina*…….46

Resumen

La empresa ALAMO GLOBAL realiza el seguimiento de los proyectos utilizando metodologías de Advanced Work Packaging (AWP) y Earned Value (EV). Sin embargo, enfrenta desafíos de eficiencia y organización al agregar y mostrar la información, lo que complica la gestión efectiva de los proyectos. El objetivo de mis prácticas fue desarrollar un sistema backend en .NET con EF CORE y SQL SERVER que mejorara estos procesos al calcular de manera dinámica y precisa el valor del trabajo realizado. Implementé un JOB en SQL Server que analiza los datos desde los niveles más detallados del proyecto, permitiendo un cálculo exacto del valor del trabajo según los paquetes de trabajo. Este sistema demostró ser preciso en los cálculos y contribuyó significativamente a la optimización del seguimiento del proyecto. La implementación de este sistema suscitó discusiones importantes sobre la adaptabilidad de estas prácticas en el contexto de la construcción, especialmente en proyectos de gran escala, y resaltó la necesidad de herramientas tecnológicas específicas para el sector.

Palabras clave: backend, desarrollo de software, .NET Core, Advanced Work Packaging, Earned Value.

Abstract

ALAMO GLOBAL tracks projects using Advanced Work Packaging (AWP) and Earned Value (EV) methodologies. However, it faces efficiency and organization challenges when adding and displaying information, complicating effective project management. The objective of my internship was to develop a backend system in .NET with EF CORE and SQL SERVER that would improve these processes by dynamically and accurately calculating the earned value. I implemented a process in SQL Server that analyzes data from the most detailed levels of the project, allowing for an exact calculation of the earned value according to the work packages. This system proved to be precise in its calculations and significantly contributed to the optimization of project monitoring. The implementation of this system sparked important discussions about the adaptability of these practices in the construction context, especially in large-scale projects, and highlighted the need for specific technological tools for the sector..

Keywords: backend, software development, .NET Core, Advanced Work Packaging, Earned Value.

Glosario

.NET: marco de trabajo para el desarrollo de aplicaciones en varios lenguajes de programación, mantenido por Microsoft.

Abstracción: proceso de ocultar los detalles complejos detrás de una operación simple, facilitando el uso de software.

API (Application Programming Interface): conjunto de reglas y especificaciones que las aplicaciones pueden seguir para comunicarse entre ellas.

AutoMapper: herramienta en .NET que automatiza el mapeo entre tipos de objetos, simplificando las asignaciones entre ellos.

AWP (Advanced Work Packaging): metodología de gestión que organiza la ejecución de proyectos en paquetes de trabajo manejables para mejorar la eficiencia y la efectividad.

Backend: parte de la aplicación de software que no interactúa directamente con el usuario, pero maneja la lógica y la base de datos.

Clase de Contexto: en EF Core, representa una sesión con la base de datos, permitiendo consultar y guardar instancias de las entidades.

Códigos HTTP: respuestas estándar de tres dígitos que indican el estado de la respuesta a una solicitud hecha a un servidor.

Controladores: en el desarrollo de software, especialmente en MVC, manejan la lógica de entrada y salida de las aplicaciones.

Data Seeding: proceso de población inicial de una base de datos con datos necesarios para su funcionamiento.

EF CORE (Entity Framework Core): framework de mapeo objeto-relacional para .NET, que permite a los desarrolladores trabajar con datos utilizando objetos específicos del dominio.

Encapsulación: concepto de POO donde se oculta el estado interno de los objetos y solo se expone una interfaz para interactuar con esos datos.

EV (Earned Value): técnica de gestión de proyectos que mide el progreso del trabajo en términos de alcance, tiempo y costos.

Función: bloque de código que realiza una tarea específica, puede ser llamado cuando sea necesario.

Herencia: mecanismo de la POO que permite a una clase derivar de otra, heredando sus métodos y propiedades.

IdentityDbContext: contexto especializado en EF Core que se utiliza para configurar la autenticación y la autorización de usuarios.

Interfaces: contratos en programación que especifican qué métodos debe implementar una clase sin definir cómo.

JOB en SQL Server: tarea programada en SQL Server para ejecutar operaciones como scripts o procedimientos almacenados en un momento específico o de manera recurrente.

JWT (JSON Web Token): método estándar de seguridad para crear tokens de acceso que permiten la propagación de identidad y privilegios.

Metodología: sistema de métodos usados en un área particular, como en la gestión de proyectos, para mejorar la organización y los resultados.

Patrones de Diseño: soluciones estándar para problemas comunes en el diseño de software.

Polimorfismo: capacidad de una función para procesar objetos de diferentes formas dependiendo de la clase a la que pertenecen.

POO (Programación Orientada a Objetos): paradigma de programación basado en la conceptuación de los elementos de software como objetos.

Protocolo HTTP (Hypertext Transfer Protocol): protocolo de comunicación utilizado para sistemas de información hipermedia como la web.

Refresh Token: un refresh token es un tipo de token de autenticación utilizado en el proceso de autenticación y autorización en aplicaciones y sistemas informáticos.

Repositorios: en programación, utilizados para encapsular la lógica necesaria para acceder a las fuentes de datos.

Responsabilidad Única: principio de diseño de software que establece que una clase o módulo debe tener una y solo una razón para cambiar.

SQL (Structured Query Language): lenguaje de programación utilizado para gestionar y manipular bases de datos relacional.

SQL Server: sistema de gestión de bases de datos relacional desarrollado por Microsoft, utilizado para almacenar y recuperar datos según sea necesario por otras aplicaciones.

Tipos de Datos: clasificaciones de datos que indican al compilador o al intérprete cómo el programador pretende usar los datos.

Unidad de Trabajo: patrón de diseño que mantiene un seguimiento de los cambios en una transacción y los resuelve de manera coherente.

User Interface (UI): parte de un sistema o aplicación con la cual los usuarios interactúan directamente.

Introducción

En la industria de la construcción, la integración eficiente de tecnologías avanzadas es fundamental para manejar la complejidad y magnitud de los proyectos actuales. ALAMO GLOBAL, una empresa líder en proyectos de construcción industrial, enfrenta desafíos persistentes en la gestión eficaz y segura de datos operativos y financieros. Estos desafíos comprometen la capacidad de la empresa para mantenerse competitiva en un mercado en rápida evolución. La empresa ha identificado la necesidad crítica de poder automatizar los procesos de integración y lógica de los datos, como primera fase a través de un backend como base sólida para la creación de un software completo que permita este objetivo.

Este informe se centra en el proyecto emprendido para desarrollar un backend robusto utilizando .NET Core y EF Core, destinado a resolver los problemas de gestión de datos de la empresa. El documento comienza con un análisis detallado de los antecedentes y las deficiencias del sistema actual, estableciendo la justificación para una revisión significativa del backend. Las actividades planificadas abarcan el diseño de la base de datos y una recopilación exhaustiva de información sobre cómo se aplican Advanced Work Packaging (AWP) y Earned Value (EV) en proyectos reales. Además, se incluirá la presentación de un diagrama de flujo que detalle el ciclo de vida del proyecto y un borrador de la interfaz de usuario, realizado en Figma, para visualizar la aplicación final.

La estrategia adoptada combina técnicas de ingeniería de software con un enfoque de seguridad integrado,incluyendo la implementación de JWT con refresh token para proteger el acceso a la información. Mediante la adopción de estas tecnologías y métodos, ALAMO

GLOBAL busca no solo mejorar su eficiencia operativa, sino también asegurar la integridad y confidencialidad de los datos críticos.

El desarrollo de este proyecto representa un paso crucial para ALAMO GLOBAL en su esfuerzo por adaptarse a las demandas de un entorno de construcción cada vez más digitalizado y seguro. Se espera que las mejoras resultantes en la gestión del proyecto proporcionen un marco más robusto y escalable, crucial para el éxito a largo plazo de la empresa en la industria.

1. Desarrollo Backend para aplicación de gestión empresarial en proyectos de construcción industrial en ALAMO GLOBAL

1.1 Planteamiento del problema

En el competitivo sector de la construcción industrial, ALAMO GLOBAL se enfrenta a desafíos significativos en la gestión de datos operativos y financieros. Aunque la empresa ha hecho esfuerzos para integrar tecnologías avanzadas, persisten problemas de eficiencia y seguridad en el manejo de información crítica. Estos problemas afectan la capacidad de la empresa para tomar decisiones informadas y gestionar sus proyectos de manera eficiente y segura.

La falta de un sistema robusto y automatizado que integre y procese datos utilizando Advanced Work Packaging (AWP) y Earned Value (EV) limita la visibilidad del progreso real de los proyectos y la precisión del seguimiento financiero. Esto se traduce en retrasos, sobrecostos y oportunidades perdidas en un mercado donde la rapidez y la precisión son cruciales para el éxito.

La necesidad crítica de ALAMO GLOBAL es desarrollar e implementar un backend eficiente y seguro que pueda automatizar la integración y lógica de los datos, estableciendo una base sólida para un sistema de gestión empresarial completo. Esta solución debe permitir la gestión en tiempo real de gastos e ingresos, mejorar la eficiencia operativa y asegurar la protección de la información confidencial, lo cual es esencial para mantener la competitividad y garantizar la entrega exitosa de proyectos.

1.2 Justificación

la naturaleza compleja y a gran escala de los proyectos de construcción industrial exige una gestión precisa y eficiente de enormes volúmenes de datos. La implementación de un sistema que pueda automatizar la integración y procesamiento de estos datos es esencial para mejorar la precisión del seguimiento del progreso y la gestión financiera de los proyectos. Esto, a su vez, reducirá significativamente los riesgos de retrasos y sobrecostos, optimizando los recursos y mejorando los márgenes de beneficio.

Además, el sector de la construcción está experimentando una transformación digital, donde la adopción de tecnologías avanzadas se ha convertido en un requisito para mantener la competitividad. Un backend que utilice .NET Core y EF Core no solo proporcionará la escalabilidad y seguridad necesarias, sino que también facilitará la integración con otras plataformas y tecnologías emergentes, asegurando que ALAMO GLOBAL permanezca a la vanguardia en innovación tecnológica.

La seguridad es otra consideración crítica. En la era de la información, la protección de datos sensibles y confidenciales es primordial. El desarrollo de un sistema que incorpore medidas de seguridad avanzadas, como JWT con refresh token para la gestión de sesiones y autorizaciones, es fundamental para prevenir accesos no autorizados y violaciones de datos.

Finalmente, la mejora en la eficiencia operativa y la toma de decisiones informadas que un sistema de backend adecuadamente diseñado y implementado puede ofrecer, posicionará a ALAMO GLOBAL como líder en la gestión eficiente de proyectos de construcción industrial. Este proyecto no solo es una inversión en tecnología sino también en el futuro estratégico de la empresa, asegurando su crecimiento y sustentabilidad a largo plazo en un mercado global competitivo

1.3 Objetivos

1.3.1 Objetivo general

Desarrollar e implementar un backend eficiente y seguro para la aplicación de gestión empresarial, utilizando .NET

1.3.2 Objetivos específicos

1. Identificar y analizar la documentación existente relevante para la gestión financiera y la eficiencia operativa en proyectos de construcción industrial.
2. Extraer datos clave necesarios para el seguimiento en tiempo real de gastos, ingresos y métricas de eficiencia.
3. Crear un diagrama de flujo detallado que represente el ciclo de vida completo de un proyecto de construcción industrial, desde la planificación hasta la finalización.
4. Definir la metodología para la integración y representación de los datos recopilados en la aplicación, asegurando que la información sea precisa y fácilmente comprensible.
5. Diseñar y desarrollar una base de datos utilizando principios de normalización y optimización de consultas para manejar eficientemente los datos.
6. Implementar un patrón de diseño de software que garantice la estructura y modularidad del sistema, facilitando futuras expansiones o modificaciones.
7. Desarrollar el backend de la aplicación utilizando .NET Core y EF Core para maximizar el rendimiento y la escalabilidad.
8. Seguir las mejores prácticas de codificación, como Clean Code, para garantizar que el código sea mantenible y legible.

9. Utilizar JWT para la gestión segura de sesiones y autorizaciones, protegiendo así la confidencialidad e integridad de los datos.

2. Marco referencial

2.1 Marco teórico

La industria de la construcción, así como los sectores petrolero y de energías, son pilares fundamentales del desarrollo económico global. Estos sectores manejan proyectos de gran escala que requieren inversiones significativas en términos de tiempo y dinero. La complejidad inherente a estos proyectos demanda una gestión precisa y eficiente para asegurar su éxito.

El seguimiento claro y detallado de los costos y tiempos en estos sectores es crucial para el éxito de los proyectos. La falta de comunicación y coordinación puede llevar a retrasos, sobrecostos y una ejecución ineficiente. Contar con sistemas robustos de gestión que permitan una total transparencia y comunicación entre todas las partes involucradas es esencial para asegurar el cumplimiento de los plazos y presupuestos. Esto facilita la identificación y resolución temprana de problemas, contribuyendo a una gestión de proyectos más efectiva y eficiente(Malsam, 2023)

Por tal razon el framework Advanced Work Packaging (AWP) es un enfoque metodológico integrado que tiene como objetivo mejorar el rendimiento y la previsibilidad de los proyectos a lo largo de su ciclo de vida. Este proceso abarca desde la planificación inicial del proyecto hasta su finalización, y se centra en identificar y ejecutar actividades de manera que se

eliminen las restricciones y se optimice la secuencia de construcción. El AWP se fundamenta en la creación de paquetes de trabajo específicos y bien definidos que faciliten la ejecución eficiente y libre de impedimentos de los proyectos de construcción

El proceso de AWP incluye varios componentes esenciales, entre ellos:

1. Engineering Work Packages (EWPs): Son entregables de ingeniería utilizados para desarrollar paquetes de trabajo de construcción (CWPs). Estos incluyen dibujos, especificaciones y datos de proveedores necesarios para la construcción
2. Construction Work Packages (CWPs): Definen un alcance específico del trabajo en una zona de construcción determinada. Incluyen detalles como presupuesto, cronograma y requisitos de rendimiento
3. Installation Work Packages (IWPs): Planes detallados que aseguran que todos los elementos necesarios para completar la instalación de una parte del trabajo en el campo estén organizados y disponibles antes de que comience el trabajo. Estos paquetes generalmente cubren un alcance de trabajo que puede ser completado por una cuadrilla en una o dos semanas
4. Construction Work Areas (CWAs): Áreas específicas dentro de un proyecto donde se agrupan múltiples CWPs para facilitar la planificación y ejecución del trabajo. Los CWAs ayudan a organizar el trabajo de manera lógica y secuencial, optimizando el uso de recursos y minimizando interferencias.

La efectividad del AWP depende en gran medida de la alineación y coordinación entre los propietarios del proyecto, contratistas y los líderes de diversas disciplinas involucradas en la

planificación, ingeniería, control del proyecto y ejecución del trabajo. Esta coordinación asegura que los paquetes de trabajo se desarrollen y se ejecuten de manera secuencial y libre de restricciones, optimizando el flujo de trabajo y mejorando la previsibilidad del proyecto (Long International, n.d.)

El Earned Value Management (EVM) es una metodología de gestión de proyectos que integra el alcance, el cronograma y los recursos para medir objetivamente el rendimiento y el progreso del proyecto. EVM permite a los gerentes de proyecto evaluar el trabajo realizado en relación con el plan y el presupuesto aprobados, proporcionando una visión clara del desempeño del proyecto (PMI, n.d.).

Componentes Clave

1. Planned Value (PV): Representa el presupuesto autorizado asignado al trabajo programado para completarse en un momento específico.
2. Actual Cost (AC): Es el costo real incurrido por el trabajo realizado durante un período específico.
3. Earned Value (EV): Mide el valor del trabajo realmente completado hasta la fecha en comparación con el presupuesto autorizado para ese trabajo.

El EVM es crucial para la gestión efectiva de proyectos porque:

- Proporciona Indicadores de Desempeño: Permite a los gerentes de proyecto medir y monitorear el progreso del proyecto con precisión.

- Facilita la Identificación Temprana de Problemas: Al comparar el PV, AC y EV, los gerentes pueden identificar desviaciones y tomar medidas correctivas de manera oportuna.
- Mejora la Planificación y el Control: Ayuda a los gerentes de proyecto a mantener el control sobre el presupuesto y el cronograma, asegurando que el proyecto se mantenga en el camino correcto.

Tanto el Advanced Work Packaging como el Earned Value Management son metodologías esenciales para mejorar la eficiencia y la efectividad en la gestión de proyectos de construcción industrial. La integración de estas metodologías en la infraestructura de gestión de ALAMO GLOBAL puede proporcionar una base sólida para el éxito de los proyectos, asegurando una ejecución eficiente, segura y predecible.

Además, la automatización de estos frameworks a través de la creación de un software especializado trae numerosos beneficios. Entre estos beneficios se incluyen la reducción de errores humanos, la mejora en la precisión y velocidad de los cálculos, y la capacidad de realizar un seguimiento en tiempo real del progreso del proyecto. La automatización también facilita la integración de datos y la generación de informes detallados, lo cual mejora la toma de decisiones y permite una mejor asignación de recursos. En conjunto, estos avances no solo optimizan la gestión del proyecto, sino que también aumentan la transparencia y la comunicación entre todos los stakeholders, contribuyendo significativamente al éxito global de los proyectos.

2.2 Marco conceptual

Advanced Work Packaging (AWP): Es una metodología estructurada que organiza el trabajo de construcción en paquetes manejables desde la planificación hasta la ejecución. Su objetivo es optimizar la secuencia de actividades y eliminar restricciones para mejorar la eficiencia y previsibilidad del proyecto (Long International, n.d.).

Earned Value Management (EVM): Es una técnica de gestión de proyectos que integra el alcance, el cronograma y los recursos para medir el desempeño del proyecto. Proporciona una visión objetiva del progreso del proyecto y facilita la toma de decisiones informadas (PMI, n.d.).

*API REST:*Una API REST (Representational State Transfer) es un estándar de arquitectura para la comunicación entre sistemas. Utiliza métodos HTTP y permite la interacción entre aplicaciones a través de un conjunto de reglas bien definidas. REST es conocido por su simplicidad y escalabilidad (Fielding & Taylor, 2020).

Backend: Es la parte del software que no interactúa directamente con el usuario final, pero maneja la lógica de la aplicación, la base de datos, y la integración con otras aplicaciones o

servicios. Es fundamental para el procesamiento de datos y la ejecución de operaciones internas (Bass et al., 2021).

Integridad de Datos: se refiere a la exactitud y consistencia de los datos almacenados en una base de datos. Es crucial para garantizar que los datos sean fiables y se mantengan sin alteraciones no autorizadas a lo largo de su ciclo de vida (Silberschatz et al., 2020).

Persistencia de Datos: Es la característica de los sistemas de software que asegura que los datos se almacenen de manera permanente y puedan ser recuperados incluso después de que la aplicación que los creó haya terminado (Fowler, 2019).

Entidades Relacionales: Las entidades relacionales son objetos del mundo real o conceptos que se representan en una base de datos relacional mediante tablas. Cada entidad tiene atributos y se relaciona con otras entidades a través de claves primarias y foráneas (Date, 2019).

Usuario: En el contexto de software, un usuario es una entidad que interactúa con un sistema o aplicación. Los usuarios pueden tener diferentes niveles de acceso y permisos según sus roles asignados (Benyon, 2020).

Esquema Entidad Relacional: El esquema entidad-relacional es un modelo utilizado para definir la estructura lógica de una base de datos. Representa las entidades, sus atributos y las relaciones entre ellas, facilitando el diseño y la implementación de bases de datos relacionales (Elmasri & Navathe, 2020).

*Normalización de Bases de Datos:*es el proceso de organizar los datos en una base de datos para reducir la redundancia y mejorar la integridad de los datos. Este proceso se lleva a cabo mediante la aplicación de un conjunto de reglas conocidas como formas normales. Cada forma normal establece criterios específicos para la estructura de las tablas y las relaciones entre ellas(Date, 2019).

- Primera Forma Normal (1NF)**:** Todos los atributos deben contener valores atómicos (indivisibles). Cada valor en una columna debe ser único, y no deben existir conjuntos repetidos de columnas.
- Segunda Forma Normal (2NF)**:** Debe estar en 1NF y todos los atributos no clave deben depender completamente de la clave primaria. Es decir, no debe haber dependencias parciales de la clave primaria.
- Tercera Forma Normal (3NF): Debe estar en 2NF y todos los atributos no clave deben depender únicamente de la clave primaria. No deben existir dependencias transitivas.

- Cuarta Forma Normal (4NF):Debe estar en BCNF y no debe tener dependencias multivaluadas no triviales. Esto significa que una entidad no debe tener conjuntos independientes de valores multivaluados.

Stored Procedure (Procedimiento Almacenado): es un conjunto de instrucciones SQL que se almacenan en una base de datos y pueden ser ejecutadas de manera repetida. Los procedimientos almacenados son utilizados para realizar operaciones repetitivas, encapsular lógica de negocio compleja, y mejorar la eficiencia y seguridad de las transacciones en la base de datos(Silberschatz, 2020).

Características Clave:

- Reutilización del Código: Almacenar la lógica de negocio en la base de datos permite su reutilización sin necesidad de escribir las mismas consultas repetidamente.
- Mejora del Rendimiento: Los procedimientos almacenados pueden mejorar el rendimiento porque las consultas precompiladas y optimizadas se almacenan directamente en la base de datos.
- Seguridad: Permiten un control más fino sobre los permisos, dado que los usuarios pueden ejecutar procedimientos almacenados específicos sin tener acceso directo a las tablas subyacentes.
- Mantenimiento: Facilitan el mantenimiento del código, ya que cualquier cambio en la lógica puede realizarse directamente en el procedimiento almacenado sin necesidad de modificar el código de la aplicación.

API FLUENT: La API Fluent en Entity Framework Core (EF Core) es una forma de configurar modelos de datos utilizando métodos en lugar de anotaciones. Esta API permite configurar propiedades y relaciones entre entidades de manera detallada y precisa, proporcionando una mayor flexibilidad y control sobre el modelo de datos(Microsoft Docs. n.d.).

Características Clave:

- Permite definir configuraciones complejas que no son posibles con las anotaciones de datos.
- Mantiene las clases de entidad limpias y separadas de la lógica de configuración.
- Facilita la creación y mantenimiento de configuraciones coherentes a lo largo del tiempo.

Program: El archivo Program.cs en una aplicación .NET Core es el punto de entrada principal de la aplicación. Este archivo configura y lanza el host de la aplicación, que maneja la inicialización, configuración y el ciclo de vida de la aplicación(Microsoft Docs, n.d.).

Características Clave:

- Define el host de la aplicación, que puede ser un host web o genérico.
- Configura los servicios y middleware que se utilizarán en la aplicación.
- Inicia la ejecución de la aplicación.

*CORS (Cross-Origin Resource Sharing)***:** Es un mecanismo de seguridad que permite a un servidor controlar si un navegador debe permitir solicitudes de recursos desde un dominio diferente (origen cruzado) al del servidor. Este mecanismo es crucial para proteger aplicaciones web de ataques de origen cruzado, como el Cross-Site Request Forgery (CSRF) (Mozilla Developer Network (MDN), n.d.).

Características Clave:

- Permite especificar qué orígenes tienen permiso para acceder a los recursos.
- Puede definir métodos HTTP permitidos, encabezados, y otros parámetros de seguridad.
- Mejora la seguridad de las aplicaciones web al controlar el acceso entre dominios.

HTTP Rate Limiting: Es una técnica utilizada para controlar el número de solicitudes que un usuario puede hacer a un servidor en un periodo de tiempo específico. Su objetivo es prevenir abusos, ataques DDoS, y garantizar la disponibilidad de recursos (Microsoft Docs. n.d.).

Características Clave:

- Define límites de solicitudes por usuario, IP, o clave de API.
- Puede implementar políticas de limitación de tasa por hora, minuto, segundo, etc.
- Ayuda a proteger la infraestructura del servidor contra sobrecargas y abusos.

*Dependencias***:** En el contexto de la inyección de dependencias, una dependencia es un objeto que una clase necesita para realizar su función. La inyección de dependencias es un patrón de diseño que facilita la creación y gestión de dependencias, promoviendo un código más limpio y modular(Microsoft Docs, n.d.).

Características Clave:

- Facilita la gestión de dependencias y su configuración.
- Promueve la separación de preocupaciones y el diseño modular.
- Mejora la testabilidad del código al permitir la inyección de objetos simulados.

Middleware: El middleware en ASP.NET Core es un componente de software que se encadena en la tubería de solicitudes HTTP. Cada componente de middleware tiene la oportunidad de manejar solicitudes y respuestas, y de pasar el control al siguiente componente en la cadena (Microsoft Docs. n.d.).

Características Clave:

- Procesa las solicitudes entrantes antes de que lleguen a un controlador.
- Puede modificar la solicitud y la respuesta.
- Permite añadir funcionalidades transversales como autenticación, logging, y manejo de errores.

*Lazy Loading:*Es un patrón de diseño que retrasa la inicialización de un objeto hasta el momento en que se necesita. En el contexto de Entity Framework, permite cargar entidades relacionadas de la base de datos sólo cuando se accede a ellas por primera vez (Microsoft Docs. n.d.).

Características Clave:

- Mejora el rendimiento inicial de la aplicación al cargar sólo los datos necesarios.
- Puede reducir el uso de memoria y recursos al evitar la carga de datos innecesarios.
- Puede introducir latencia adicional al acceder a los datos si las entidades relacionadas no están ya cargadas.

2.3 Marco legal

Leyes Nacionales

Ley de Protección de Datos Personales (Privacy Act of 1974):La Ley de Protección de Datos Personales de 1974 tiene como objetivo principal proteger la privacidad de los datos personales manejados por las agencias del gobierno federal de los Estados Unidos. Establece normas rigurosas sobre la recolección, almacenamiento, uso y divulgación de información personal, garantizando que los individuos tengan ciertos derechos sobre sus datos, como el derecho a acceder y corregir la información que se tiene sobre ellos (Bureau of Justice Assistance, n.d.).

En el contexto del proyecto de implementación de un backend para ALAMO GLOBAL, esta ley implica la necesidad de implementar medidas de seguridad robustas para proteger los datos personales manejados por la aplicación. Esto incluye:

- Encriptación de Datos: Uso de técnicas de encriptación para asegurar que los datos sensibles estén protegidos tanto en tránsito como en reposo.
- Control de Acceso: Establecimiento de controles estrictos sobre quién puede acceder a la información personal y bajo qué condiciones.
- Auditorías y Monitoreo: Implementación de sistemas de auditoría y monitoreo para detectar y responder rápidamente a cualquier acceso no autorizado o violación de datos.

Leyes Estatales

*Leyes de Privacidad de Datos de Texas:*Las leyes de privacidad de datos de Texas están diseñadas para proteger la información personal y datos confidenciales de los residentes del

estado. Estas leyes establecen normas sobre cómo se debe manejar, almacenar y proteger la información personal, y otorgan a los individuos derechos específicos sobre sus datos (Casetext, n.d.).

Aplicación en el Proyecto: En el desarrollo del backend para ALAMO GLOBAL, es crucial implementar prácticas de seguridad de datos que cumplan con estas leyes estatales, incluyendo:

- Protección de Datos Sensibles: Asegurar que cualquier dato personal recogido, almacenado o procesado por la aplicación esté adecuadamente protegido contra accesos no autorizados y violaciones de datos.
- Notificación de Brechas de Seguridad: Establecer procedimientos para notificar a los individuos y a las autoridades competentes en caso de una violación de datos, conforme a las normativas estatales.
- Conformidad con la Texas Business and Commerce Code: Seguir las directrices establecidas en el Código de Comercio y Negocios de Texas para la gestión de datos personales y confidenciales.

3. Método

3.1 Investigación y análisis documental.

Investigación de Advanced Work Packaging: Se comienza con un análisis exhaustivo sobre el framework AWP, su metodología y la forma en cómo deben ser aplicados en los proyectos. Para esto, se realiza una búsqueda del framework en sus repositorios oficiales, como el PMI (Project Management Institute).

*Investigación de Earned Value:*Se lleva a cabo una búsqueda profunda sobre esta técnica para la gestión de proyectos, observando de manera detallada cómo se calculan sus índices de desempeño para evaluar el rendimiento del proyecto en función de sus objetivos..

Análisis de implementación de AWP y EV: Se solicita documentación sobre proyectos realizados por ALAMO GLOBAL donde se apliquen AWP y EV, con el objetivo de observar su implementación y comportamiento con los datos. Estos proyectos están en ejecución, por lo cual se debe hacer un seguimiento del comportamiento de cada proyecto para identificar cualquier factor que pueda ser importante al momento de crear el backend.

3.2 Generación de entregables conceptuales.

*Creación del diagrama de flujo:*Se elabora un diagrama de flujo que representa el ciclo de vida de un proyecto desde su inicio hasta su finalización en la aplicación, con el fin de comprender cómo será el flujo de los datos y los posibles comportamientos frente a eventualidades.

Identificación de entidades y sus propiedades: Teniendo en cuenta la interacción de los datos y las partes implicadas en el desarrollo de un proyecto, se realiza la abstracción necesaria sobre las entidades involucradas, buscando normalizarlas hasta la Cuarta Forma Normal (4FN)..

*Creación de un borrador de UI:*Como siguiente paso, se crea un borrador de la Interfaz de Usuario (UI) para visualizar el ciclo de vida de un proyecto de principio a fin utilizando EV y AWP. En esta interfaz se diseñan las posibles vistas, formularios y funcionalidades, dependiendo de los roles y el avance del proyecto.

3.3 Mejoramiento y Aplicación de las Recomendaciones del Feedback por parte de ALAMO GLOBAL

Recoleccion del Feedback: Durante el desarrollo del proyecto, se recolectarán comentarios y sugerencias de los stakeholders clave a través de reuniones periódicas sobre los entregables generados.

*Análisis del Feedback***:** Se realizará un análisis detallado del feedback recibido para identificar áreas de mejora y aspectos críticos que necesitan ajustes.

Implementación de Mejoras: Las recomendaciones serán implementadas siguiendo un plan de acción estructurado, asegurando que los cambios se integren de manera eficiente en el proyecto.

Evaluación de Resultados: Se monitorizarán los cambios implementados para evaluar su efectividad y realizar ajustes adicionales si es necesario.

3.4 Establecimiento de la Arquitectura del Backend

En esta sección se detalla el proceso de diseño y configuración de la arquitectura del backend, organizada en cuatro capas principales: Persistence, Domain, Application y API. Cada capa tiene un propósito específico y contribuye a la separación de responsabilidades y la modularidad del sistema.

Detalles por Capa:

- *Capa de Persistencia (Persistence Layer)***:**Esta capa se encarga de la interacción directa con la base de datos. Contiene la clase de contexto de la base de datos y las configuraciones de las entidades utilizando la API Fluent.
    - Componentes:
        - Clase de Contexto: Definición del contexto de la base de datos, que maneja las conexiones y las operaciones CRUD.
        - Configuración de Entidades: Configuración detallada de las entidades utilizando la API Fluent para definir las relaciones y restricciones de base de datos.
- *Capa de Dominio (Domain Layer):* Contiene las entidades del dominio y las interfaces que definen los contratos que deben seguir las implementaciones.
    - Componentes:
        - Entidades: Definición de las clases de entidad que representan los objetos del dominio.
        - Interfaces: Definición de interfaces para los repositorios y otras dependencias del dominio.
- *Capa de Aplicación (Application Layer):*Gestiona la lógica de negocio y la coordinación de las operaciones entre las distintas capas.

    - Componentes:
        - Unidad de Trabajo (Unit of Work): Implementación del patrón Unit of Work para manejar las transacciones y la coherencia de datos.
        - Repositorios: Implementación de los repositorios que interactúan con la capa de persistencia para realizar las operaciones de datos.
    - Elección del Patrón de Diseño Unit of Work: La elección de este patrón se basa en la necesidad de gestionar las transacciones de manera eficiente, garantizando que todas las operaciones de una transacción se realicen correctamente antes de confirmarse. Esto minimiza las inconsistencias y facilita el seguimiento de los cambios durante el ciclo de vida de la transacción.
- *Capa de API (API Layer):*Expone los servicios y puntos de entrada de la aplicación al mundo exterior, gestionando las solicitudes HTTP.
    - Componentes:
        - Configuración Principal del Programa: Configuración inicial y middleware necesarios para la aplicación.
        - Controladores: Definición de los controladores que gestionan las solicitudes HTTP y las respuestas.
        - Servicios: Implementación de la lógica de negocio que se expone a través de la API.

3.5 Creación del Backend

Configuración inicial: Para este caso, se configura el proyecto con las librerías necesarias para los requerimientos del API y se establecen las capas para la estructura del backend.

El proceso de creación se llevó a cabo de la siguiente manera:

- Se configura la clase de contexto con *IdentityDbContext* para conectar con la base de datos y permitir la administración de usuarios.
- Se crea la cadena de conexión con el administrador de base de datos SQL Server.
- Se añaden al programa todas las configuraciones necesarias para el correcto funcionamiento del API: CORS, ciclo de vida de las dependencias, HTTP Rate Limit, middleware de JWT.
- Se crean las entidades en la capa de *Domain* y se definen sus relaciones por convención.
- Se crean las configuraciones de entidades por API Fluent en la capa de *Persistence*.
- Inserción de las entidades en la clase de contexto.
- Generación de la migración que permitirá crear la base de datos.
- Generación de las interfaces (contratos).
- Generación de los repositorios.
- Generación de Unit of Work que permitirá el Lazy Loading de las instancias de cada repositorio.
- Creación de los servicios de usuario para los procesos de administración de usuarios y roles.
- Creación del servicio CallOut que permitirá invitar a los usuarios a la aplicación asignándoles un rol.
- Creación de los controladores y sus respectivos endpoints.

3.6 Creación del JOB en SQL Server para Calcular el Earned Value (EV)

A continuación se describe el proceso de creación del JOB en SQL Server que servirá para recorrer el proyecto desde su parte más granular hasta obtener el EV del proyecto.

Creación de Procedimientos Almacenados (Stored Procedures) Involucrados:Para esto se tiene en cuenta la estructura de la base de datos y cómo funciona de manera atómica cada propiedad y relación de las entidades. Posteriormente, se construyen los procedimientos almacenados que cumplan con funciones específicas y permitan la salida de datos necesaria para el análisis en el siguiente Stored Procedure (SP):

Definición de Procedimientos Almacenados:Se desarrollan SPs específicos para calcular diversos componentes del EV, tales como los costos planificados, costos reales y valores ganados.Cada SP se encarga de una parte específica del cálculo, asegurando modularidad y facilidad de mantenimiento.

- Estructura y Relaciones:Se aseguran las relaciones y dependencias entre las entidades en la base de datos para que los SPs puedan extraer y manipular los datos necesarios de manera efectiva.

*Acoplamiento de los Procedimientos Almacenados:*Se toma cada Stored Procedure y se unen haciendo uso de tablas temporales para almacenar los datos que estos arrojan y así permitir el análisis completo del proyecto.

- Uso de Tablas Temporales:Los datos intermedios generados por cada SP se almacenan en tablas temporales para un análisis consolidado.
- Consolidación y Análisis:Los SPs individuales se ejecutan secuencialmente, y los datos se consolidan para obtener un análisis integral del EV.

*Pruebas del Funcionamiento:*Se realizan distintas pruebas del conjunto de procedimientos en diferentes contextos, simulando diversas posibilidades en cada proyecto.

- Pruebas de integracion: Estas pruebas aseguran que los SPs funcionan correctamente cuando se combinan.
- Simulacion de escenarios: Se simulan diferentes escenarios del proyecto para verificar la precisión y eficiencia de los SPs en condiciones reales.

*Creación del JOB:*Se utilizan las herramientas que ofrece SQL Server para crear el JOB sobre la base de datos en la que se está trabajando y se configura la periodicidad de análisis del JOB.

- Definición del JOB: Se configura el JOB en SQL Server Agent para que ejecute los SPs a intervalos regulares.

Configuración de la Periodicidad:Se define la periodicidad del JOB para asegurar que el cálculo del EV se realiza en los intervalos necesarios para el monitoreo continuo del proyecto.

4. Resultados

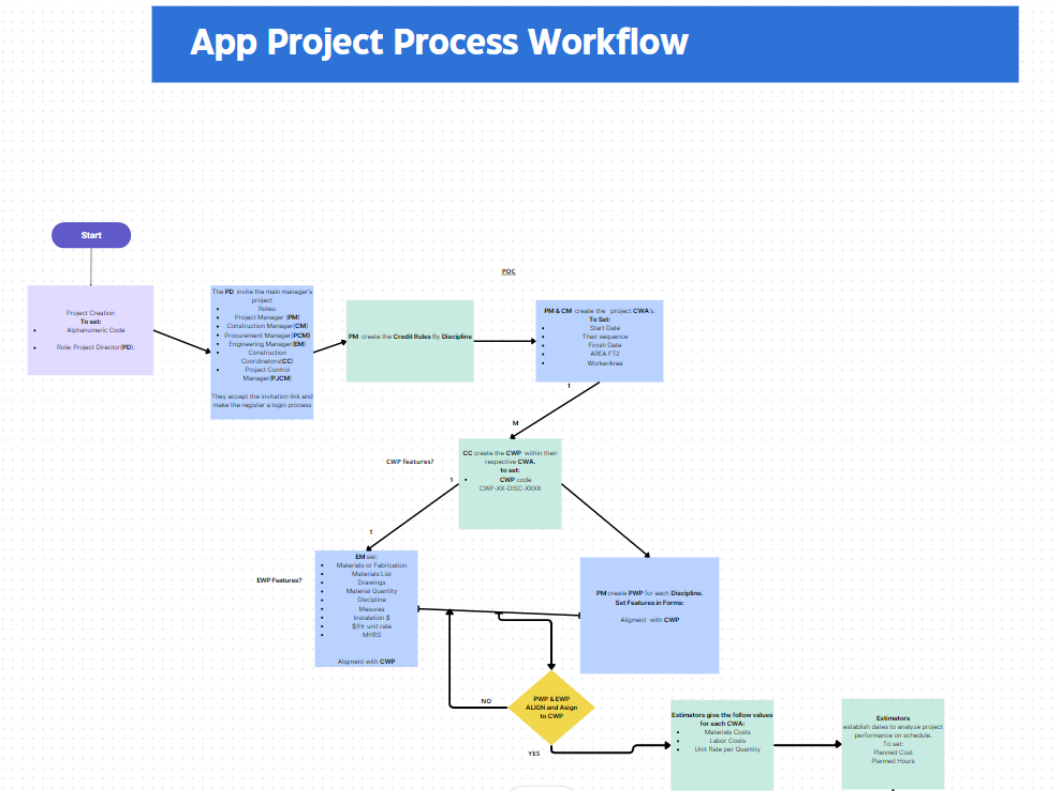
Durante el período de prácticas, se han alcanzado numerosos resultados que reflejan un avance significativo en el desarrollo del proyecto. A continuación, se presentan algunos de los logros más destacados,

4.1 Generación de entregables conceptuales:

Cada uno de los siguientes entregables son la base conceptual que permitió comprender la lógica de negocio y estructurar bien el sistema de Backend.

Diagrama de Flujo: Creación de un diagrama de flujo que representa el ciclo de vida del proyecto desde su inicio hasta su finalización.

Figura 1: *Representación del diagrama de flujo que permite ver la dinamica de creación y finalización de un proyecto en la aplicacion*



Borrador de UI en Figma: Desarrollo de un borrador de la interfaz de usuario (UI) en Figma, detallando las posibles vistas, formularios y funcionalidades según los roles y el avance del proyecto.

Figura 2: *Draft del funcionamiento de la UI con el software FIGMA*

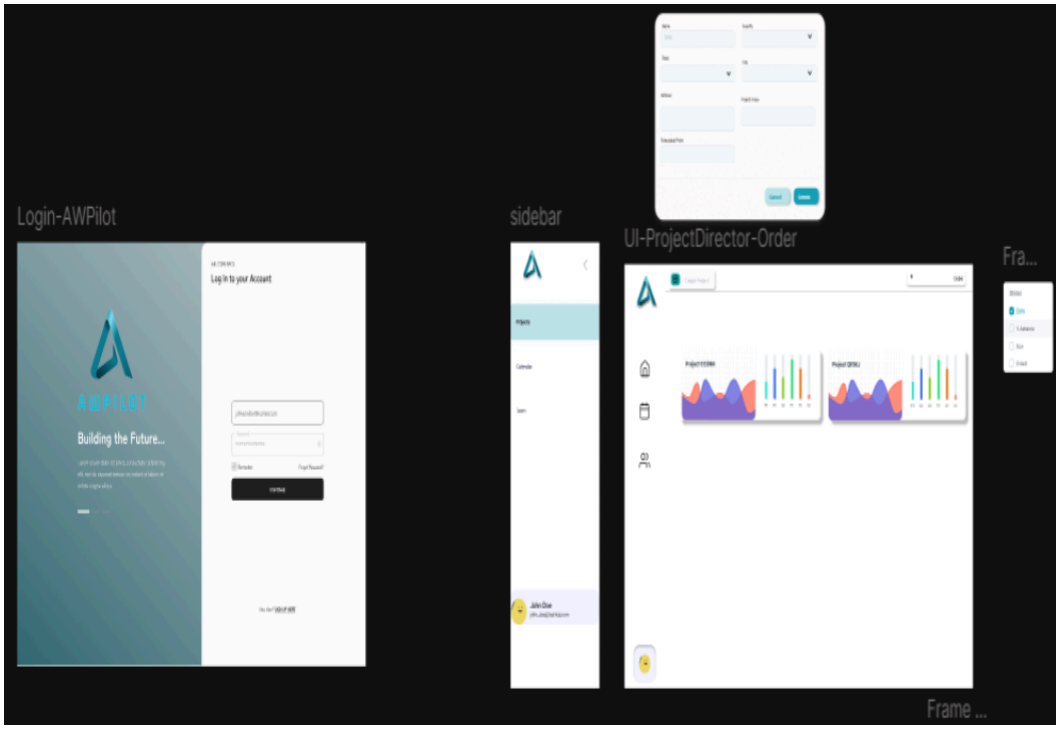
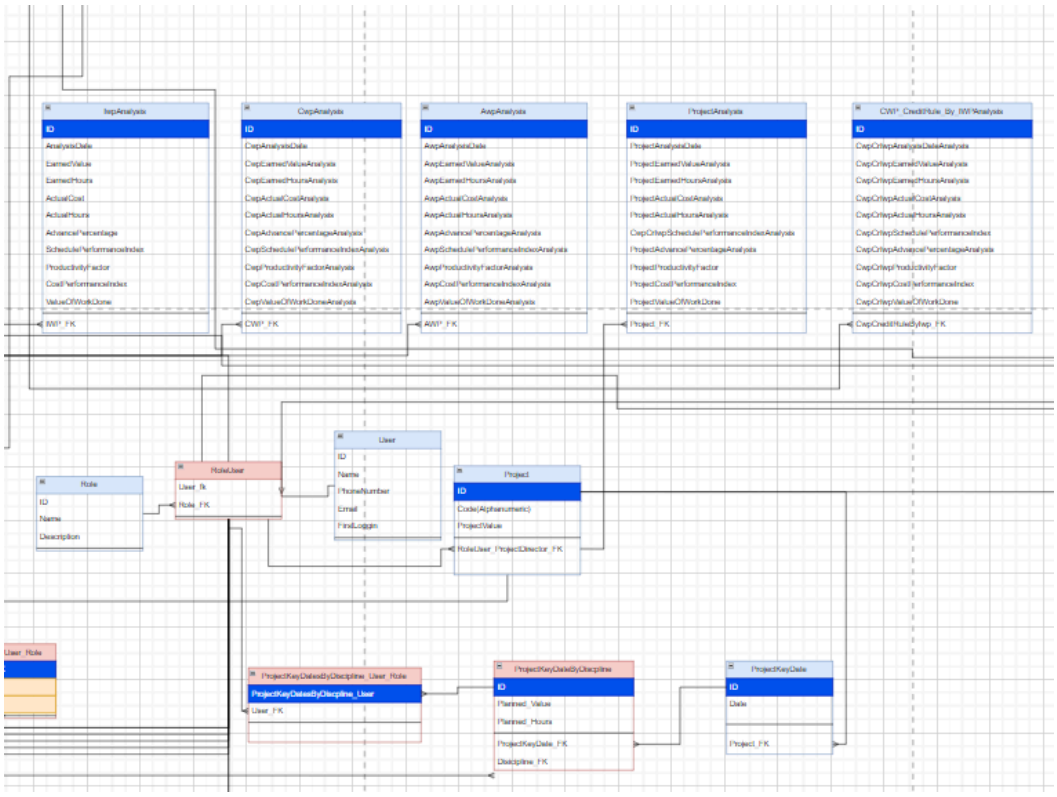


Diagrama Entidad-Relación: Creación del diagrama entidad-relación con la abstracción necesaria sobre las entidades involucradas para lograr normalizarlas hasta la 4FN.

Figura 3. *Esquema del diagrama entidad relación realizado en Draw.io*



4.2 Creación del backend en Visual Studio:

Despliegue de una arquitectura de backend bien definida y estructurada en cuatro capas: Persistence, Domain, Application y API.Ademas se observa la implementacion de JWT.

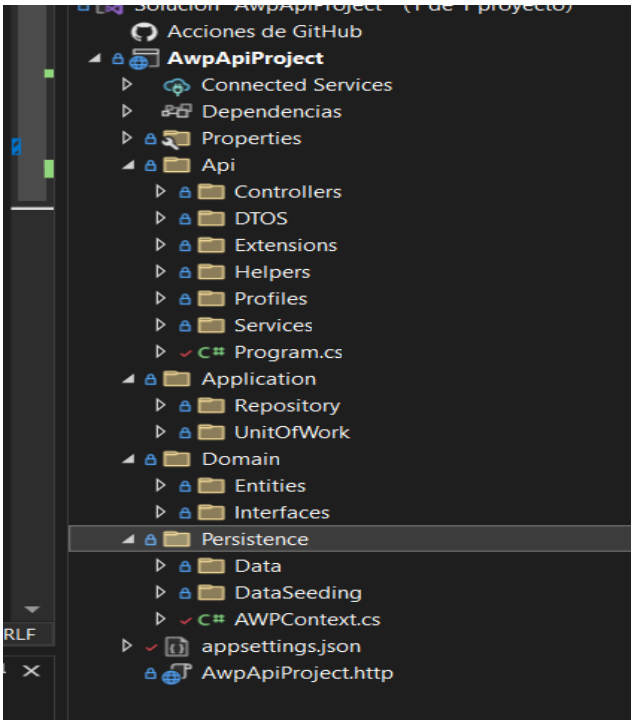
Persistence Layer: Clase de contexto y configuraciones de entidades mediante API Fluent.

*Domain Layer***:** Entidades y interfaces definidas.

*Application Layer***:** Implementación de Unit of Work y repositorios.

API Layer: Configuración del programa, controladores y servicios.

Figura 4. *Representacion de las 4 capas en visual studio*



Creacion del middleware de JWT.

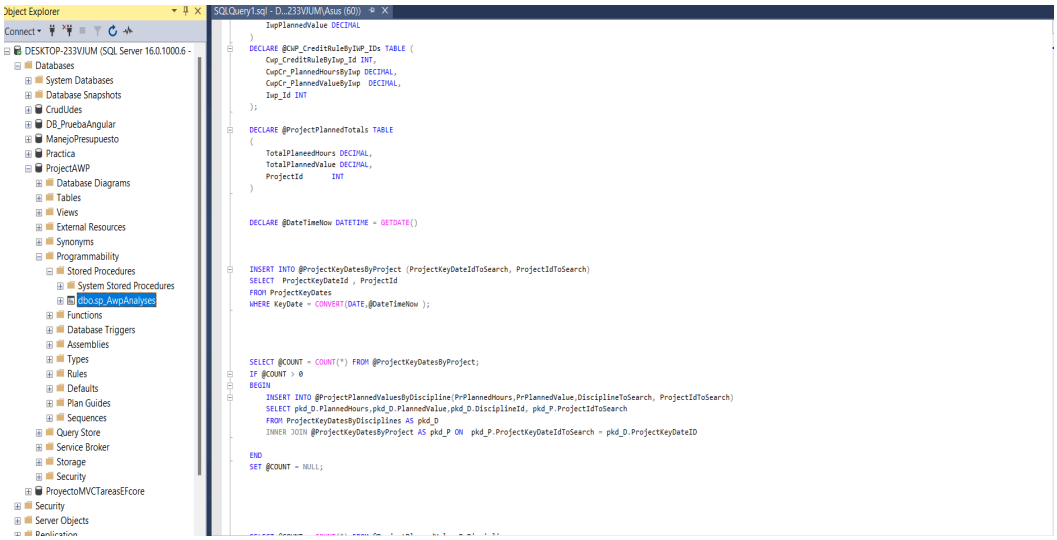
Figura 5: *Creación del metodo de extensión que permite implementar JWT en el API.*

```
1 referencia
public static void AddJwt(this IServiceCollection services, IConfiguration configuration)
{
    //Configuring the JWT and get the JWT settings from appsettings
    services.Configure<JWT>(configuration.GetSection("JWT"));

    services.AddAuthentication(options =>
    {
        options.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;
        options.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;
        //sets the default JWT authentication system for the API

    }).AddJwtBearer(options =>
    {
        options.RequireHttpsMetadata = false;
        options.SaveToken = true;
        //Here we use IdentityModelToken to set the configuration for JWT

        options.TokenValidationParameters = new TokenValidationParameters
        {
            ValidateIssuerSigningKey = true,
            ValidateIssuer = true,
            ValidateAudience = true,
            ValidateLifetime = true,
            ClockSkew = TimeSpan.Zero,
            ValidIssuer = configuration["JWT:Issuer"],
            ValidAudience = configuration["JWT:Audience"],
            IssuerSigningKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(configuration["JWT:key"]))
        };
    });
```

4.3 Creacion del JOB en sql sever.

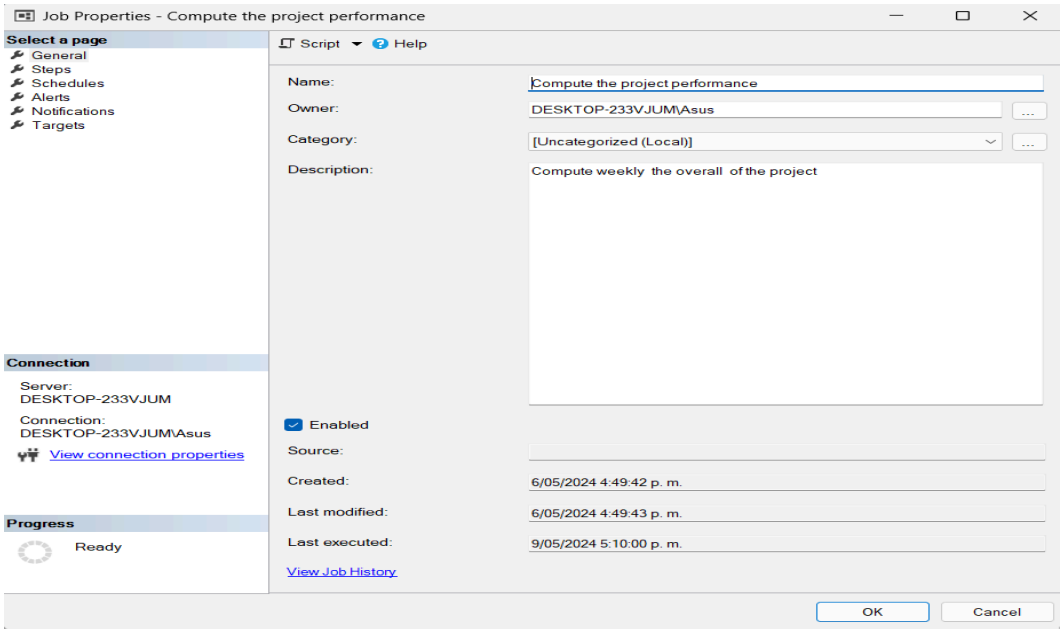
Procedimientos almacenados necesarios para la creación del JOB.

Figura 6. *Codigo SQL que representa los SPs*



Implementación de un JOB en SQL Server Agent para la ejecución periódica de los procedimientos almacenados y la obtención del Earned Value.

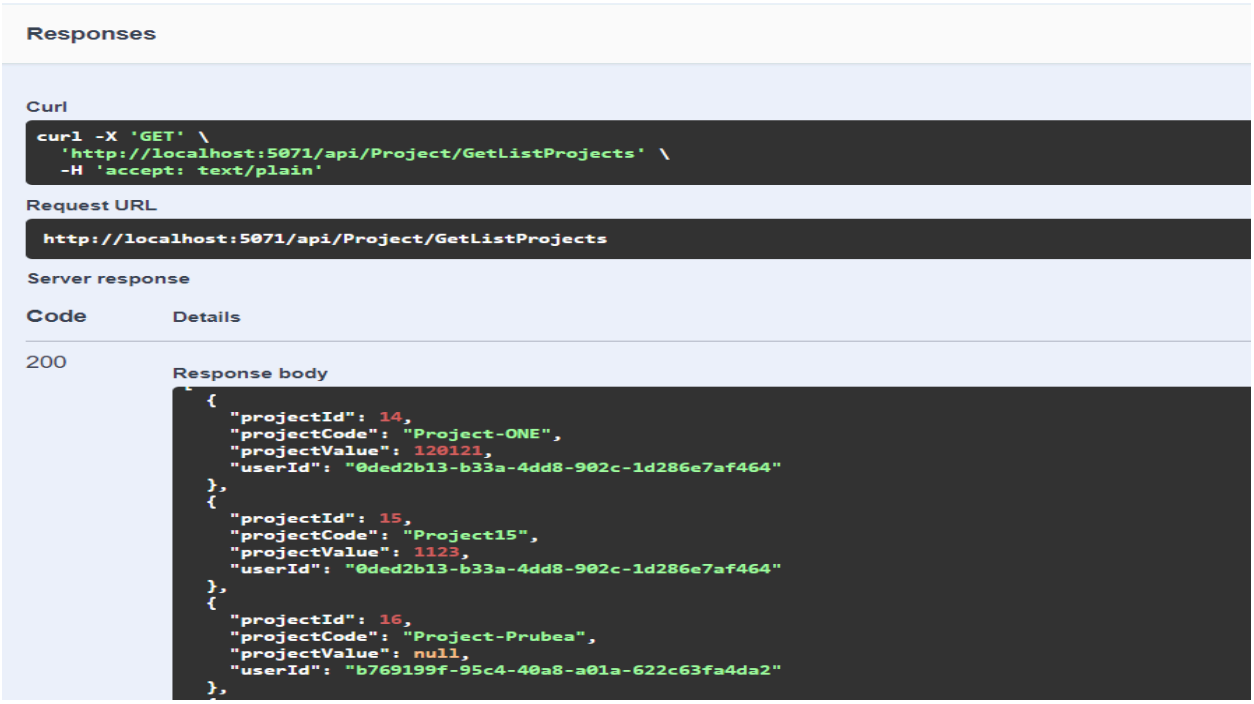
Figura 7. *Configuración del JOB en SQL Server Agent*



4.4 Pruebas del backend

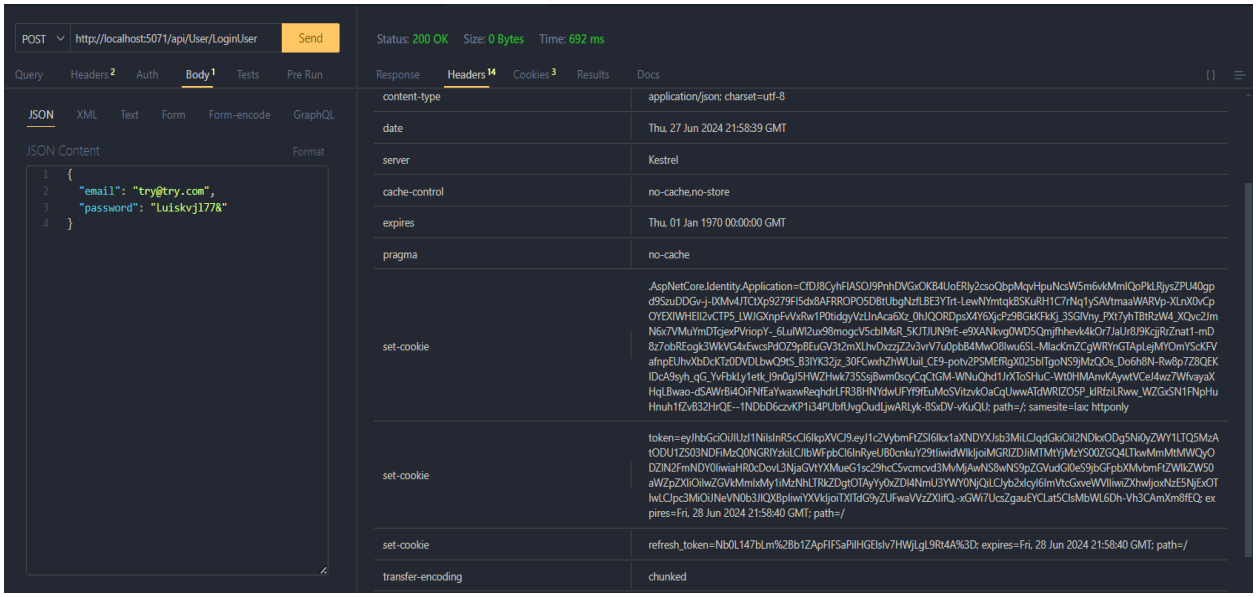
Pruebas de endpoints y salida de datos como respuesta.

Figura 8. *Respuesta del endpoint correspondiente a los proyectos en swagger*



Prueba del funcionamiento de JWT en el endpoint de login

Figura 9. *Respuesta del endpoint de login usando ThunderClient en Visual Studio Code*



4.5 Resultados adicionales

Durante los últimos meses de las prácticas profesionales, logré avanzar en aspectos adicionales para la entrega de lo que sería la segunda fase de la aplicación. En este caso, comencé a desarrollar la interfaz de usuario utilizando el framework Angular, junto con otras librerías como Angular Material y Bootstrap 5. Durante este proceso, he logrado realizar las siguientes vistas, que están totalmente conectadas y funcionan de manera eficaz con el backend.

Figura 10. *Creacion de la vista login*

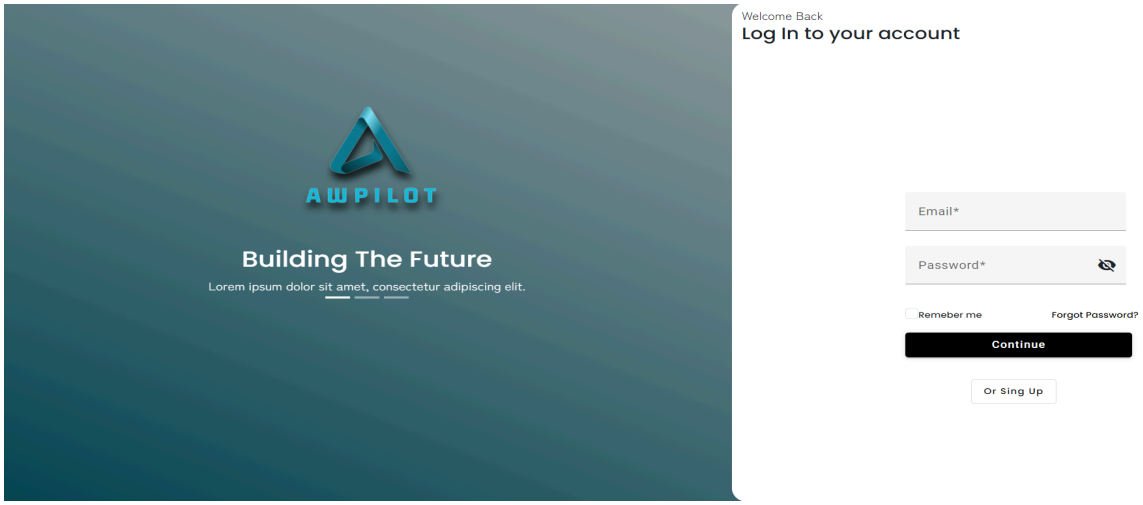


Figura 11. *Creacion de la vista de sign up*

Figura 12. *Creacion de la vista home con rol de Admin*

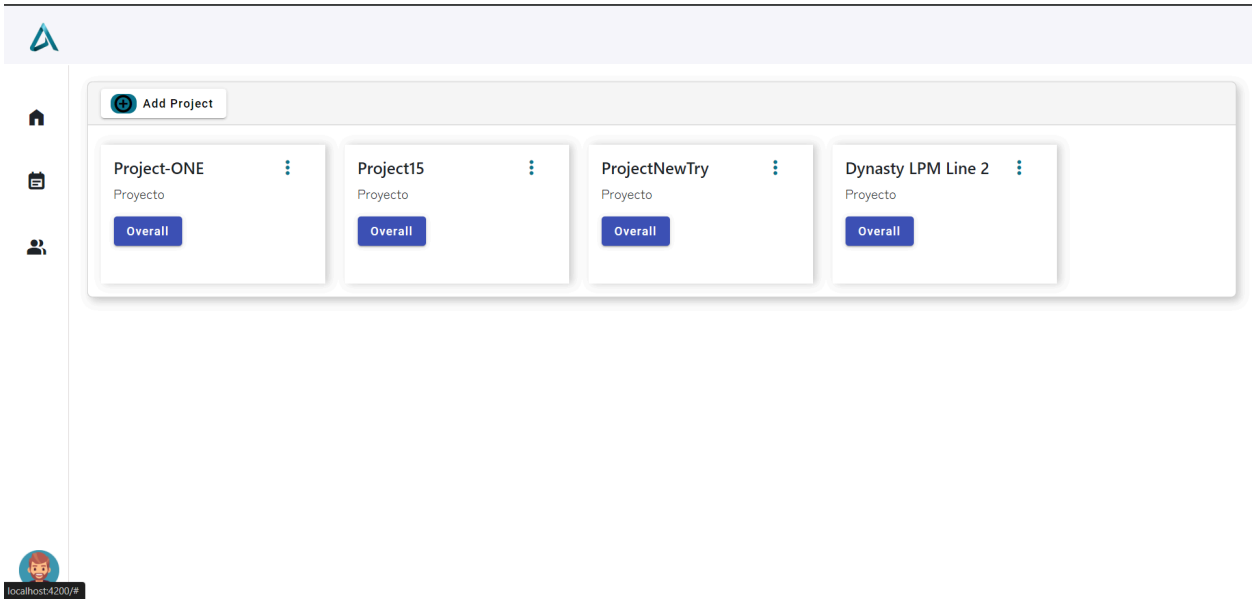


Figura 13. *Creacion de offcanvas dinámico*

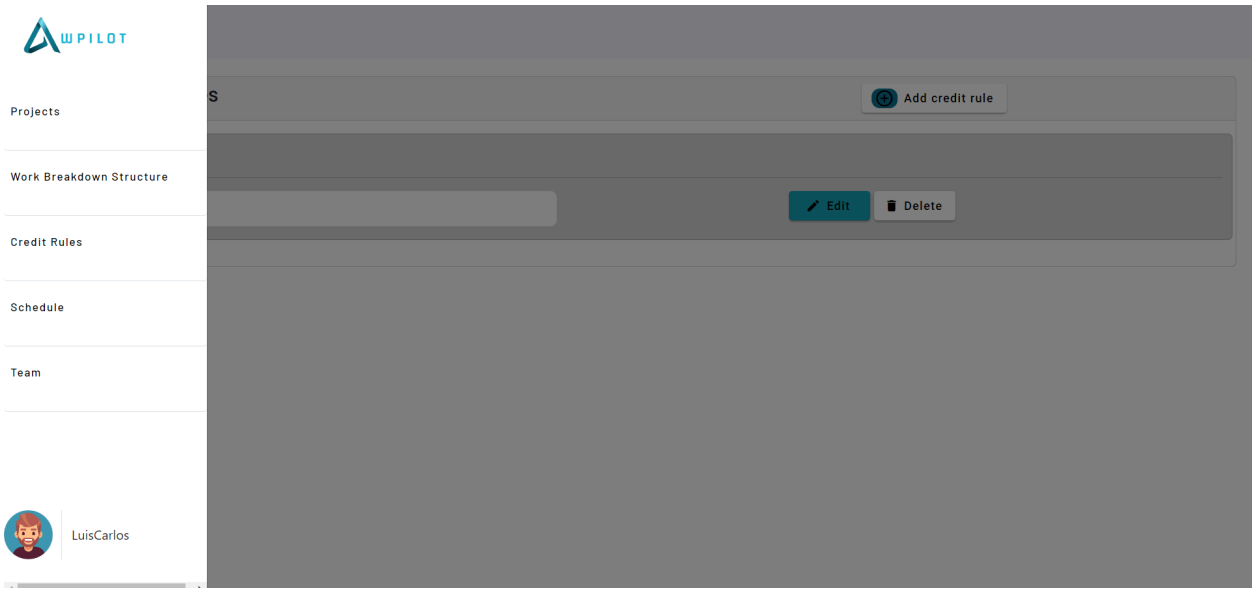


Figura 14. *Creacion del overall de Cwas por proyecto*

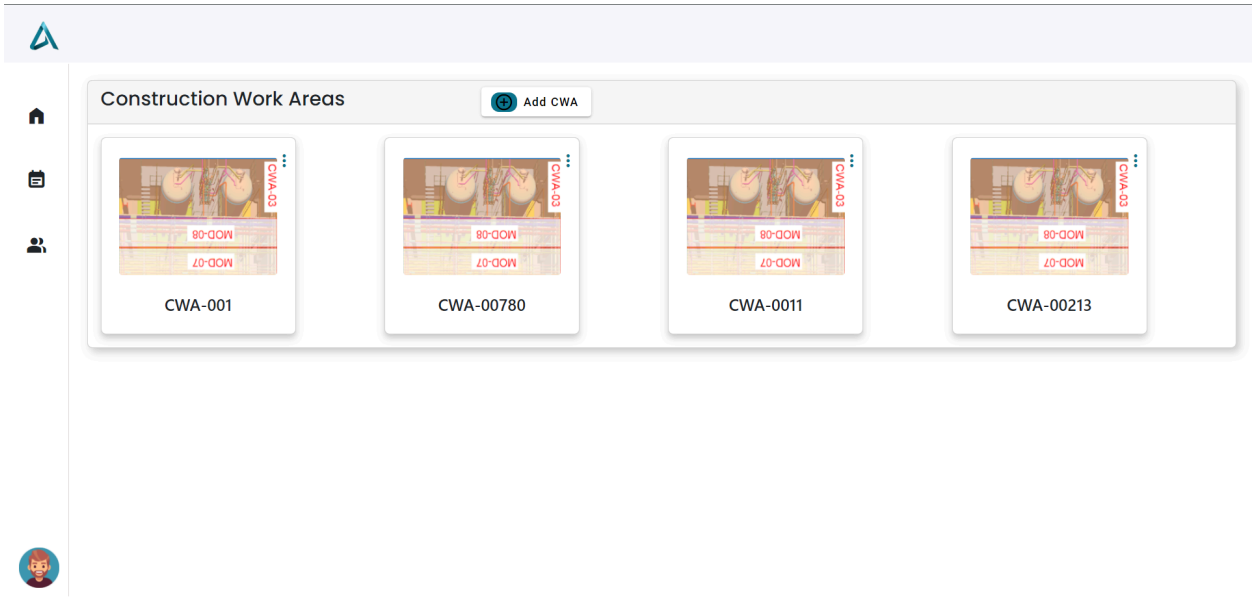


Figura 15. *Creacion modal para editar un Cwa*

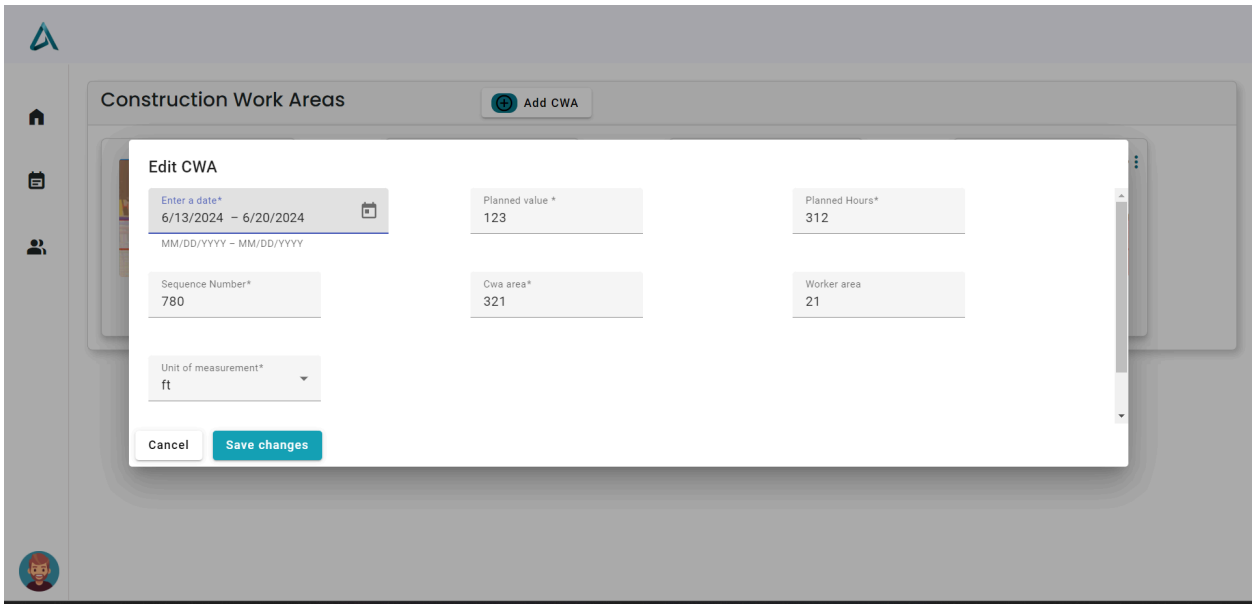


Figura 16. *Creacion de la vista de reglas de credito por proyecto divididas por disciplina*

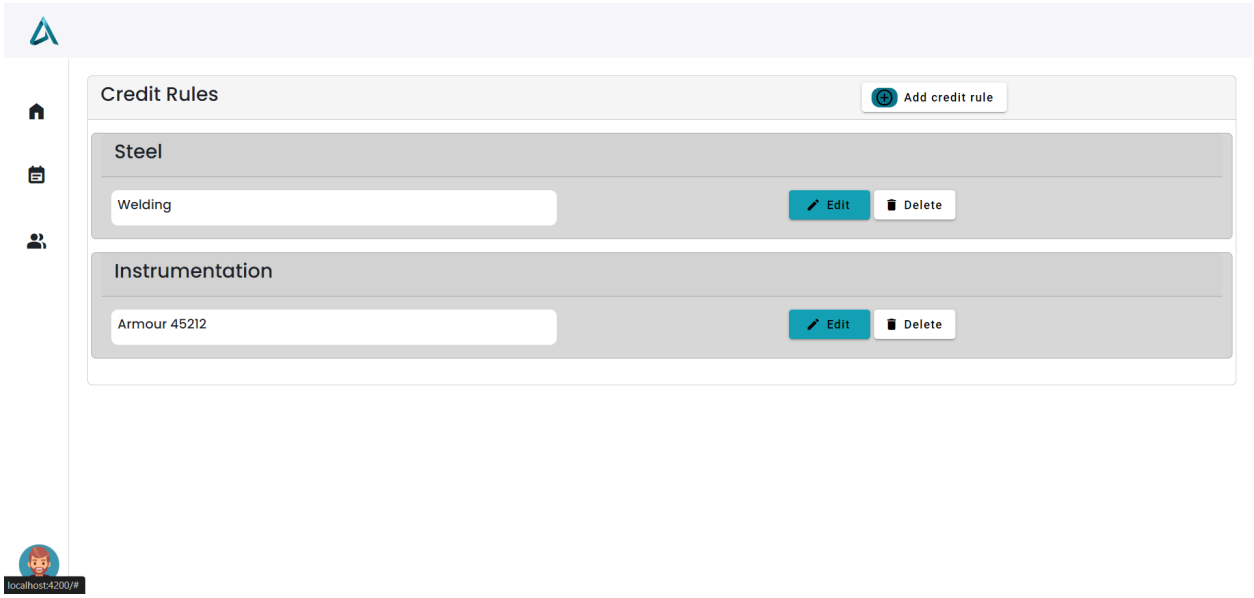
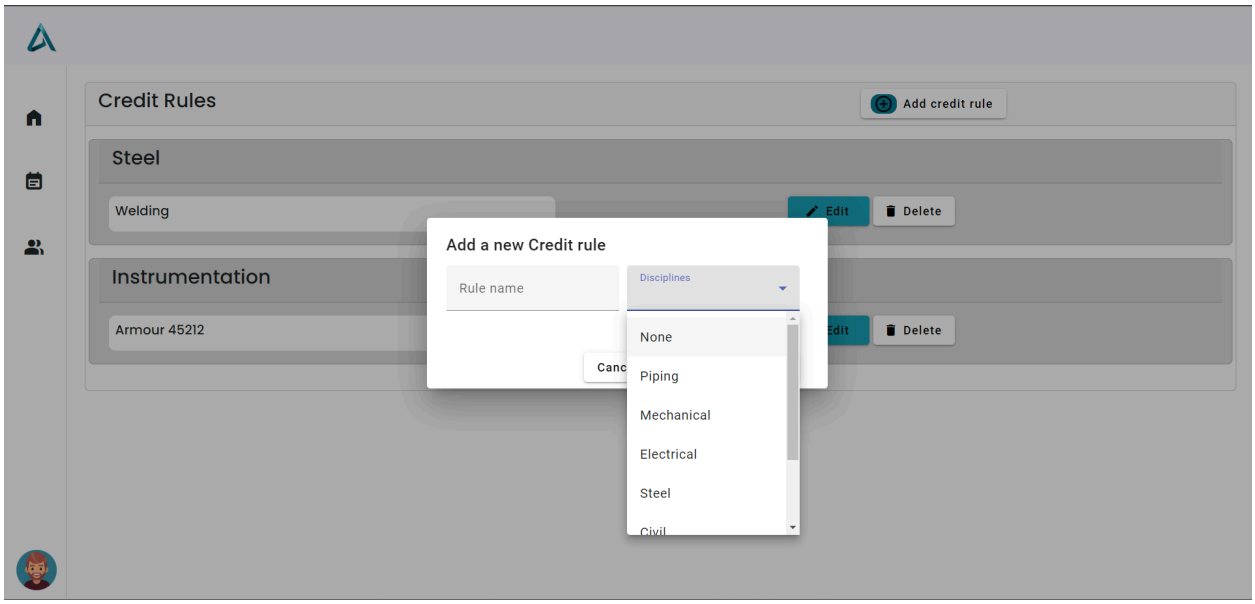


Figura 17. *Modal para agregar una regla de credito por proyecto por disciplina*



5. Discusión

Durante la fase de desarrollo y evaluación del backend para el proyecto, he observado que este componente ha cumplido con las expectativas planteadas inicialmente, demostrando ser una herramienta eficaz en la visualización y optimización del cálculo del Earned Value (EV) mediante la aplicación de Advanced Work Packaging (AWP). La implementación de AWP ha permitido una estructuración más clara y una ejecución más eficiente de los procesos, facilitando una mejor gestión y seguimiento del progreso del proyecto.

Seguridad del Backend

En términos de seguridad, el backend ha alcanzado los niveles esperados, proporcionando un entorno robusto y protegido. Sin embargo, se ha identificado la necesidad de implementar medidas de seguridad adicionales al momento de su despliegue. Estas medidas incluyen configuraciones avanzadas de firewall y la instalación de certificados SSL para habilitar el uso de HTTPS. Estas precauciones son cruciales para proteger los datos transmitidos entre el cliente y el servidor, garantizando así la integridad y la confidencialidad de la información.

Mejoras en el Frontend

Respecto al frontend, aunque cumple con las funciones básicas, se reconoce la necesidad de realizar mejoras sustanciales para optimizar la usabilidad y la experiencia del usuario. Será esencial enfocarse en mejorar la interfaz para que sea más intuitiva y amigable, asegurando que los usuarios puedan navegar y operar el sistema con facilidad y eficiencia. Esto incluirá revisar el

diseño actual, mejorar los flujos de usuario y realizar pruebas exhaustivas para identificar y corregir cualquier deficiencia en la interfaz de usuario.

Próximos Pasos

Las etapas futuras del proyecto deberán centrarse en implementar las mejoras de seguridad sugeridas, continuar con el desarrollo y refinamiento del frontend, y realizar pruebas continuas para asegurar que tanto la funcionalidad como la seguridad del sistema cumplan con todos los requisitos y estándares de la industria.

Este proceso no solo ayudará a fortalecer la estructura técnica del proyecto, sino que también mejorará significativamente la experiencia general del usuario, facilitando una mayor adopción y satisfacción del sistema final.

6. Conclusiones

Cumplimiento de Objetivos del Backend: El desarrollo del backend ha cumplido con éxito los objetivos establecidos, demostrando una capacidad eficaz para gestionar y optimizar el cálculo del Earned Value (EV) a través del uso del Advanced Work Packaging (AWP). Esto ha facilitado una supervisión más precisa y estructurada de los proyectos, permitiendo una evaluación clara del progreso y la eficiencia operativa.

Importancia de la Seguridad: A lo largo del proyecto, se ha reafirmado la importancia de la seguridad en el desarrollo de aplicaciones. Aunque el sistema ha mostrado un buen nivel de seguridad, el proceso ha destacado la necesidad de implementar medidas adicionales, como

configuraciones de firewall avanzadas y certificados SSL, para garantizar una protección óptima durante el despliegue en entornos de producción.

*Desarrollo del Frontend***:** El frontend, aunque funcional, requiere mejoras significativas para alcanzar el nivel de eficiencia, intuición y amigabilidad que los usuarios finales esperan. Este aspecto del proyecto necesita una atención especial para mejorar la interfaz de usuario y la experiencia general, lo cual es crucial para la adopción y satisfacción del usuario.

Estrategia de Implementación y Pruebas: Las estrategias de implementación y pruebas han sido esenciales para identificar áreas de mejora y asegurar que todas las funcionalidades cumplan con los requisitos establecidos. Las pruebas continuas serán fundamentales para mantener la robustez del sistema a medida que se integran nuevas funcionalidades y se realizan ajustes basados en el feedback de los usuarios.

Impacto Organizacional: El proyecto ha demostrado el potencial para impactar positivamente en la gestión de proyectos de la empresa, ofreciendo herramientas que facilitan la toma de decisiones basadas en datos y mejoran la planificación y ejecución de proyectos. La adopción de este sistema puede conducir a una mejora significativa en la gestión de recursos y en la eficacia operativa de la organización.

Referencias

Malsam, W. (2023). Construction Cost Management: Managing Construction Costs. *ProjectManager*.Recuperado el 4 de Julio de 2024: https://www.projectmanager.com

Long International. (n.d.). *CIIs Advanced Work Packaging*. Recuperado el 4 de Julio de 2024 https://www.long-intl.com/articles/ciis-advanced-work-packaging/

Bass, L., Clements, P., & Kazman, R. (2021). *Software Architecture in Practice*. Addison-Wesley Professional.

Benyon, D. (2020). *Designing User Experience: A Guide to HCI, UX and Interaction Design*. Pearson.

Date, C. J. (2019). *An Introduction to Database Systems*. Pearson.

Elmasri, R., & Navathe, S. B. (2020). *Fundamentals of Database Systems*. Pearson.

Silberschatz, A., Korth, H. F., & Sudarshan, S. (2020). *Database System Concepts*. McGraw-Hill Education.

Microsoft Docs. (n.d.). Fluent API. Recuperado el 5 de Julio de 2024 https://learn.microsoft.com/en-us/ef/core/modeling/#use-fluent-api-to-configure-a-model

Microsoft Docs. (n.d.). Program.cs.Recuperado el 5 de Julio de 2024 https://docs.microsoft.com/en-us/aspnet/core/fundamentals/host/web-host?view=aspnetcore-5.0

Mozilla Developer Network (MDN). (n.d.). CORS. Recuperado el 5 de Julio de 2024 https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS

Microsoft Docs. (n.d.). Rate Limiting. Recuperado el 5 de Julio de 2024 https://learn.microsoft.com/en-us/aspnet/core/performance/rate-limit?view=aspnetcore-8.0

Microsoft Docs. (n.d.). Dependency Injection. Recuperado el 5 de Julio de 2024 https://docs.microsoft.com/en-us/dotnet/core/extensions/dependency-injection

Microsoft Docs. (n.d.). Middleware. Recuperado el 5 de Julio de 2024 https://docs.microsoft.com/en-us/aspnet/core/fundamentals/middleware?view=aspnetcore-5.0

Microsoft Docs. (n.d.). Lazy Loading. Recuperado el 5 de Julio de 2024 https://docs.microsoft.com/en-us/ef/core/querying/related-data/lazy

Date, C. J. (2019). *An Introduction to Database Systems*. Pearson.

Fielding, R. T., & Taylor, R. N. (2020). *Principled Design of the Modern Web Architecture*. ACM.

Fowler, M. (2019). *Patterns of Enterprise Application Architecture*. Addison-Wesley.

Long International. (n.d.). *CIIs Advanced Work Packaging*. Recuperado el 4 de Julio de 2024 https://www.long-intl.com/articles/ciis-advanced-work-packaging/

Microsoft Learn. (2024). What is Azure role-based access control (Azure RBAC).Recuperado el 4 de Julio de 2024 https://learn.microsoft.com

Project Management Institute (PMI). (n.d.). Earned Value Management (EVM). Recuperado el 4 de Julio de 2024 https://www.pmi.org/learning/library/earned-value-management-systems-analysis-8026

Silberschatz, A., Korth, H. F., & Sudarshan, S. (2020). *Database System Concepts*. McGraw-Hill Education.

Bureau of Justice Assistance. (n.d.). *Privacy Act of 1974*. Retrieved from https://bja.ojp.gov/program/it/privacy-civil-liberties/authorities/statutes/1279

Casetext. (n.d.). *Section 541.101 Effective 7/1/2024: Controller Duties; Transparency*. Recuperado el 4 de Julio de 2024 https://casetext.com/statute/texas-codes/business-and-commerce-code/title-11-personal-identity-information/subtitle-c-consumer-data-protection/chapter-541-consumer-data-protection/subchapter-c-controller-and-processor-data-related-duties-and-prohibitions/section-541101-effective-712024-controller-duties-transparency?searchWithin=true&listingIndexId=texas-codes.business-and-commerce-code&q=protection%20of%20sensitive%20data&type=statute&sort=relevance&p=1&tab=keyword&jxs=

Apéndices

Apéndice A. *Nombre del apéndice*

Apéndice como tal o nota de véase archivo en fuente externa

Apéndice B. *Nombre del apéndice*

Apéndice como tal o nota de véase archivo en fuente externa

Apéndice C. *Nombre del apéndice*

Apéndice como tal o nota de véase archivo en fuente externa