**UNIVERSIDAD MARIANO GÁLVEZ DE GUATEMALA**

Sede Portales

Facultad de Ingeniería en Sistemas

Curso: **ANÁLISIS DE SISTEMAS II**



**Estudiantes**

JACKELINE MARLENY AQUINO BRAVO

DIEGO ANDRES ESQUIVEL BELTETON

LUCERO ESMERALDA ARRIAGA LOPEZ

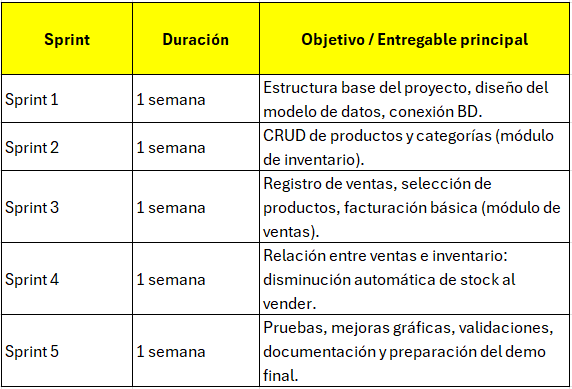
CRISTIAN YOVANY GONZALEZ RUIZ

ALAN DANIEL JOLON ZACARIAS

**Asignar nombre al equipo scrum**

CanchesTechnology

**Definir el sistema que se construirá por sprints**

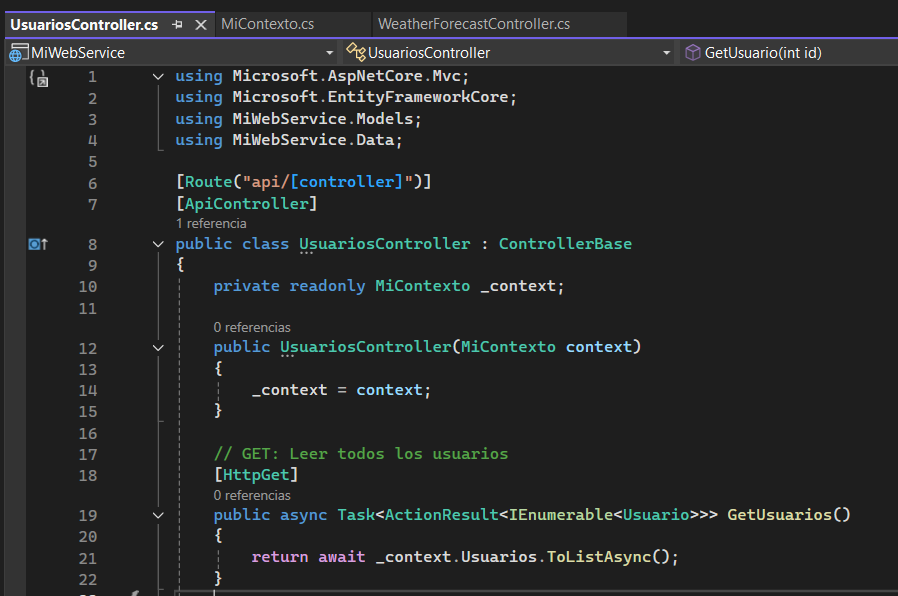


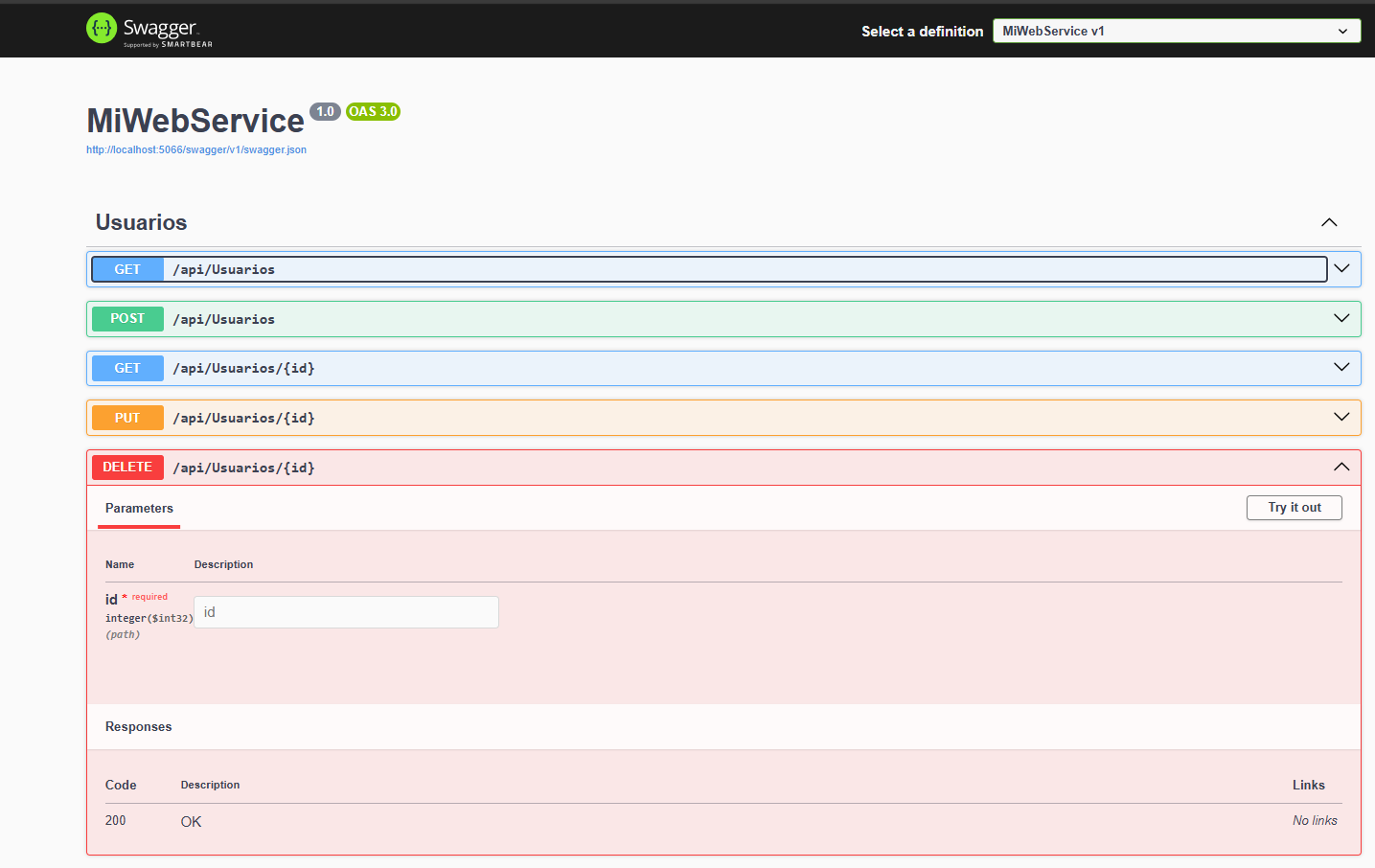
Asignar el rol a cada integrante

| **Integrante** | **Rol Scrum** | **Responsabilidad principal** |
| --- | --- | --- |
| **Diego Andrès Esquivel Beltetòn** | *Scrum Master* | Coordina reuniones diarias, facilita el trabajo del equipo, se asegura de seguir Scrum. |
| **Jackeline Marleny Aquino Bravo** | *Product Owner* | Define los requerimientos del ERP, prioriza el backlog, representa al “cliente”. |
| **Lucero Esmeralda Arriaga López** | *Developer (Frontend)* | Diseña e implementa las interfaces de usuario de ambos módulos. |
| **Cristian Yovany González Ruiz** | *Developer (Backend)* | Crea la lógica y API para productos, stock y operaciones de venta. |
| **Alan Daniel Jolón Zacarías** | *Developer / QA* | Apoya en frontend y backend, realiza pruebas y validaciones de funcionamiento. |

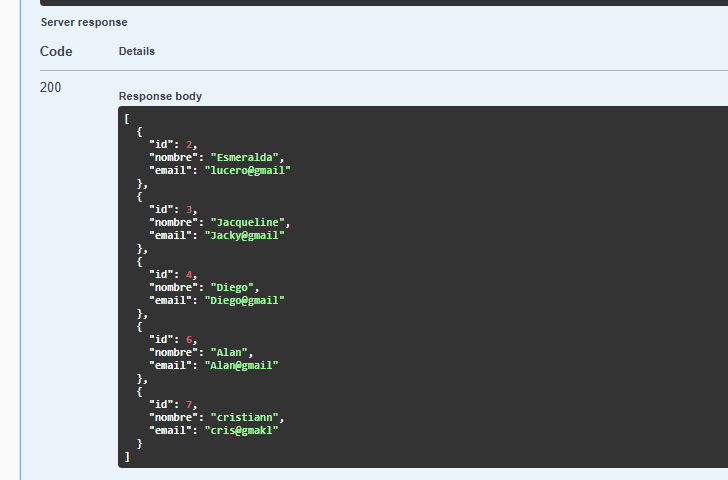
**Construcción de un Web Service:**

Diseño y construcción del Web Service  
Servicio RESTful usando ASP.NET Core con el controlador UsuariosController.

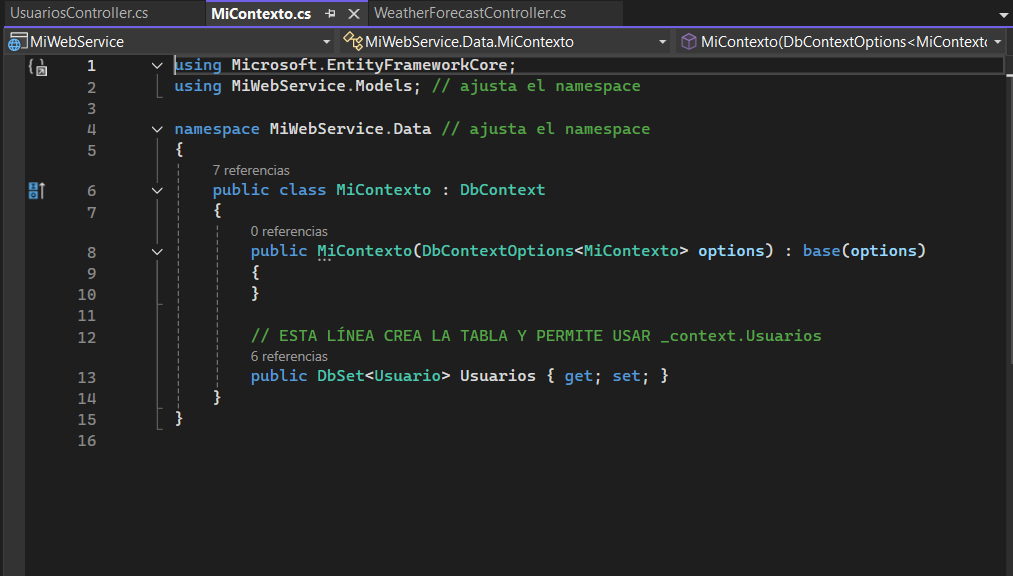


Desacoplamiento cumplido: el backend está separado y puede ser consumido por cualquier frontend o cliente externo.  


Endpoint que consulta información relevante  
Cumplido con el endpoint GET api/usuarios, que devuelve todos los usuarios.



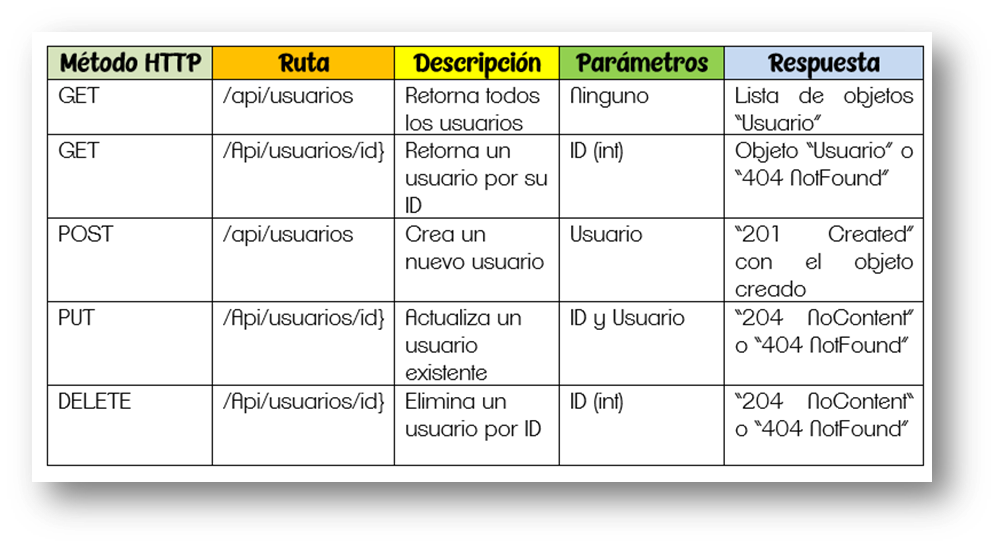
Implementación del endpoint, lógica y conexión con la base de datos  
Implementado con Entity Framework Core y DbContext.



**Objetivo del Web Service**

El objetivo de este Web Service es proporcionar una API RESTful que permita realizar operaciones CRUD sobre una tabla de usuarios. Está desarrollado en ASP.NET Core y utiliza Entity Framework Core para interactuar con una base de datos relacional. Este servicio facilita el desacoplamiento entre el backend y cualquier cliente frontend que necesite consumir información de usuarios.

**Documentar los endpoints y su funcionalidad**

****

**instrucciones de instalación y despliegue**

Se detallan los pasos necesarios para instalar y ejecutar el Web Service desarrollado con ASP.NET Core y Entity Framework Core:

#### Requisitos previos

* .NET SDK 7.0 o superior instalado.
* Visual Studio 2022 o Visual Studio Code.
* SQL Server Express, LocalDB o cualquier otra instancia de SQL Server.
* Herramienta de pruebas como Swagger

#### instalación y ejecución

1. Clonar o descargar el proyecto  
    Descargar el código fuente del Web Service y abrir la solución ( en Visual Studio 2022 o abrir la carpeta del proyecto en Visual Studio Code.
2. Configurar la base de datos  
    En el archivo appsettings.json, localizar la cadena de conexión dentro de la sección "ConnectionStrings" y ajustarla según la configuración local.
3. Aplicar las migraciones y crear la base de dato Abrir la Consola del Administrador de Paquetes (Package Manager Console) en Visual Studio y ejecutar los siguientes comandos: Add-Migration Inicial Update-Database
4. Ejecutar el Web Service  
    Presionar F5 o hacer clic en el botón “Iniciar depuración” en Visual Studio.  
    Al ejecutarse correctamente, se abrirá el navegador con la documentación generada por Swagger (/swagger) donde se pueden ver y probar los endpoints disponibles del Web Service.
5. Pruebas  
    Desde Swagger UI o Postman, se pueden enviar peticiones GET, POST, PUT, y DELETE al endpoint /api/usuarios para probar el funcionamiento del CRUD.

**Historias de Usuario** **de Productos en Bodega**

## Descripción

**Como** encargado de inventario,  
**Quiero** registrar el ingreso de nuevos productos al sistema,  
**Para** mantener el inventario actualizado y reflejar las entradas de productos de manera precisa.

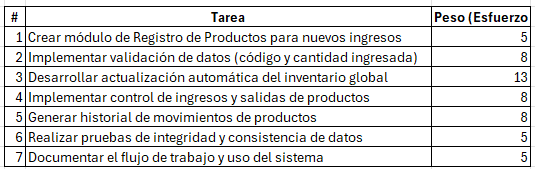
## Criterios de Aceptación:

1. **Registro de productos**:  
   El sistema debe permitir ingresar productos nuevos, proporcionando datos como nombre del producto, descripción, categoría, precio, cantidad, y proveedor.
2. **Validación de datos**:  
   El sistema debe validar que los datos ingresados sean correctos (por ejemplo, cantidad debe ser un número positivo, precio debe ser un valor numérico, etc.).
3. **Actualización automática de inventario**:  
   Al registrar un producto, el sistema debe actualizar el inventario automáticamente, sumando las cantidades ingresadas al stock disponible.
4. **Historial de cambios**:  
   El sistema debe registrar el historial de cada ingreso de producto, incluyendo la fecha, el usuario que lo ingresó y las cantidades registradas, para poder hacer auditorías y verificar la trazabilidad.

## Valor para el Negocio:

* **Precisión en el inventario**: Registrar de manera adecuada el ingreso de productos asegura que el inventario refleje de manera precisa las existencias disponibles, evitando errores en el stock.
* **Eficiencia operativa**: El registro de productos es un proceso clave en la gestión del inventario, y tenerlo bien integrado garantiza que el sistema esté siempre actualizado.
* **Facilidad para auditorías**: Un historial detallado de los ingresos de productos facilita las auditorías internas y la trazabilidad de cada movimiento en el inventario.

## Detalle de tareas



**Historia de Usuario:Registro de Clientes**

**Historia de Usuario: Registro de Clientes**

**Descripción:** Como administrador del sistema y/o vendedor  
Quiero registrar nuevos clientes proporcionando sus datos completos,  
Para gestionar su información y facilitar futuras interacciones o transacciones.

**Criterios de Aceptación:**

1. **Creación de clientes:**
   * El sistema debe permitir registrar clientes con los siguientes datos:
     + NIT
     + Nombre
     + teléfono
     + dirección
2. **Edición y eliminación de clientes:**
   * El administrador debe poder actualizar los datos de un cliente registrado.
   * El administrador debe poder eliminar registros de clientes cuando sea necesario.
3. **Validación de datos:**
   * El sistema debe validar la unicidad del documento de identificación y del correo electrónico.
   * Notificar al administrador si los datos ingresados son incorrectos o incompletos.

Detalle de Tareas

