I. Descripción y Propósito del Proyecto

El objetivo es simular y demostrar cómo un sistema de autenticación en dos pasos puede mejorar la seguridad de los procesos de inicio de sesión. Este enfoque garantiza que incluso si un atacante obtiene las credenciales de un usuario, no podrá acceder sin completar el segundo paso, que depende del control del correo electrónico del usuario.

El propósito de este proyecto es implementar una simulación del proceso de autenticación en dos pasos, un enfoque común en la seguridad moderna para verificar la identidad de un usuario. Este sistema busca garantizar que, además de validar las credenciales (correo y contraseña) del usuario, también se confirme su identidad mediante un proceso adicional de verificación por correo electrónico.

Flujo del sistema

1. **Creación de Usuario:**
    - Los nuevos usuarios deben poder registrarse, ingresando sus datos
2. **Ingreso de Credenciales:**
    - Los usuarios deben proporcionar su correo electrónico y contraseña. El sistema valida estas credenciales contra la base de datos.
3. **Envío de Correo Electrónico:**
    - Si las credenciales son correctas y el usuario existe, el sistema envía un correo electrónico a la dirección proporcionada. Este correo contiene un enlace con un token único que es necesario para proceder al siguiente paso de la autenticación.
4. **Confirmación de Registro:**
    - El usuario debe hacer clic en el enlace enviado para completar el registro. Este enlace activa un endpoint en el sistema que valida el token y, si es válido, completa el proceso de autenticación.

II. Descripción del Contenido del Archivo Program.cs

Este archivo define la configuración principal y el punto de entrada de la aplicación ASP.NET Core. Aquí se configuran los servicios, middleware y componentes necesarios para que la aplicación funcione correctamente.

1. **Configuración de Servicios:**
    - Se añaden servicios esenciales como controladores para la API, Swagger para la documentación automática de la API, y el contexto de base de datos para manejar la conexión a MySQL.
    - Se registra la lógica para mapeo de objetos entre entidades y modelos DTO.
    - Se inyectan dependencias para servicios personalizados que manejan funcionalidades específicas como productos, usuarios, tokens de sesión y el envío de correos electrónicos.
2. **Autenticación JWT:**
    - El sistema de autenticación se configura para usar JWT (JSON Web Tokens). Esto incluye la validación de tokens y la configuración de las reglas para determinar su autenticidad, vigencia y firma.
3. **Generación de Documentación con Swagger:**
    - Swagger se configura para generar documentación de la API y permitir la autenticación usando tokens JWT en el entorno de desarrollo, facilitando la prueba de endpoints protegidos.
4. **Configuración del Pipeline de Middleware:**
    - Se define el comportamiento de la aplicación al manejar solicitudes HTTP, incluyendo la habilitación de HTTPS, la gestión de autorizaciones y la inclusión de la interfaz de Swagger en el entorno de desarrollo.
5. **Ejecución de la Aplicación:**
    - Finalmente, se define cómo y cuándo se ejecutará la aplicación, iniciando todos los servicios y configuraciones previas.

III. Descripción del Contenido del Archivo UserController.cs

Este archivo define el controlador encargado de gestionar la creación, autenticación y confirmación de usuarios en la API. Las operaciones aquí definidas son fundamentales para la gestión de usuarios y la implementación del flujo de autenticación mediante JWT.

Este controlador centraliza toda la lógica relacionada con la gestión de usuarios y autenticación mediante JWT. Incluye funcionalidades para la creación de usuarios,

validación mediante tokens, envío de correos electrónicos y generación de tokens JWT seguros.

1. **Crear Usuario:**
    - Endpoint que recibe las credenciales de un nuevo usuario, los registra en el sistema y, si el registro es exitoso, envía un correo electrónico de confirmación. Este correo contiene un enlace para validar la cuenta, con un token de verificación incluido.
2. **Confirmar Registro:**
    - Este endpoint es llamado cuando un usuario hace clic en el enlace de confirmación enviado a su correo. El sistema valida el token recibido, genera un JWT para el usuario y retorna dicho token, confirmando el registro.
3. **Iniciar Sesión:**
    - Endpoint que permite a los usuarios iniciar sesión. Valida las credenciales proporcionadas y, si son correctas, genera y almacena un JWT en el sistema, el cual se usa para autenticar futuras solicitudes del usuario.
4. **Construcción de Tokens JWT:**
    - El controlador incluye dos métodos privados para la creación de tokens JWT. Estos métodos generan tokens que contienen información esencial del usuario (claims), junto con la clave de firma y las configuraciones necesarias para garantizar su validez y seguridad.
5. **Envío de Correos Electrónicos:**
    - En el proceso de registro, se utiliza un servicio de correos electrónicos para enviar el correo de confirmación. El contenido del correo incluye un enlace dinámico que lleva al usuario al proceso de confirmación.

IV. Descripción del Contenido del Archivo ProductoController.cs

Este archivo define el controlador para gestionar las operaciones relacionadas con productos en la API. El controlador expone varios endpoints que permiten realizar acciones como obtener productos, actualizar stock, y crear nuevos productos.

1. **Inyección de Dependencias:**

- Se inyectan los servicios necesarios para gestionar la lógica de negocio relacionada con productos, y se utiliza AutoMapper para realizar conversiones entre entidades y DTOs.

2. **Endpoints Definidos:**
   - **Obtener Productos:**
     - Un endpoint para obtener la lista de productos.
     - Un endpoint para obtener un producto específico por su ID.
   - **Gestionar Stock:**
     - Endpoints para aumentar o disminuir el stock de un producto usando un DTO que contiene el ID y la cantidad a cambiar.
   - **Crear Productos:**
     - Endpoint para crear un producto individual.
     - Endpoint para crear una lista de productos en una única solicitud.
3. **Validaciones y Manejo de Errores:**
   - Se incluyen validaciones para asegurarse de que los datos de entrada no sean nulos y se gestionan los errores cuando no se encuentra un producto o los datos enviados son incorrectos.
4. **Autorización:**
   - El controlador está protegido por JWT, lo que significa que solo los usuarios autenticados pueden acceder a los endpoints.