5 –

5.1Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

5.3 -

Prueba CanBeCancelledBy\_AdminCancelling\_ReturnsTrue:

Esta prueba verifica que el método CanBeCancelledBy en la clase Reservation funcione correctamente cuando un usuario con privilegios de administrador intenta cancelar una reserva. Se asegura de que el método retorne true cuando un administrador realiza la acción de cancelación, lo que es esencial para permitir que los administradores tengan la capacidad de cancelar reservas.

Prueba CanBeCancelledBy\_SameUserCancelling\_ReturnsTrue:

En esta prueba, se evalúa si el método CanBeCancelledBy en la clase Reservation se comporta adecuadamente cuando el mismo usuario que creó la reserva intenta cancelarla. La prueba comprueba que el método devuelva true, lo que confirma que un usuario puede cancelar su propia reserva sin problemas.

Prueba CanBeCancelledBy\_AnotherUserCancelling\_ReturnsFalse:

Esta prueba se centra en verificar el comportamiento del método CanBeCancelledBy cuando un usuario diferente al que hizo la reserva intenta cancelarla. La prueba garantiza que el método devuelva false en esta situación, lo que es importante para prevenir que usuarios no autorizados cancelen reservas que no les pertenecen, fortaleciendo así la seguridad y control del sistema.

Texto

Descripción generada automáticamente

Captura de pantalla de un teléfono celular

Descripción generada automáticamente

Pantalla de computadora con letras

Descripción generada automáticamente

Error en la prueba



Refactorización de codigo

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Sitio web

Descripción generada automáticamente

6.3 –

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

6.2 –

La prueba "Get\_ReturnsWeatherForecasts" evalúa si el método Get en el controlador WeatherForecastController de una aplicación web es capaz de devolver una lista de pronósticos del tiempo y si esta lista contiene exactamente 5 elementos.

6.3 –

El test unitario escrito en C# utiliza el framework de pruebas NUnit. Lo que hace es verificar el comportamiento del método Get en la clase WeatherForecastController en la aplicación web API simple. Basicamente, verifica que el método Get del controlador WeatherForecastController devuelve una lista de pronósticos del tiempo que no es nula y que tiene exactamente 5 elementos. Este es un ejemplo para asegurarse que la aplicacion siga funcionando correctamente a medida que se realizan cambios en el código.

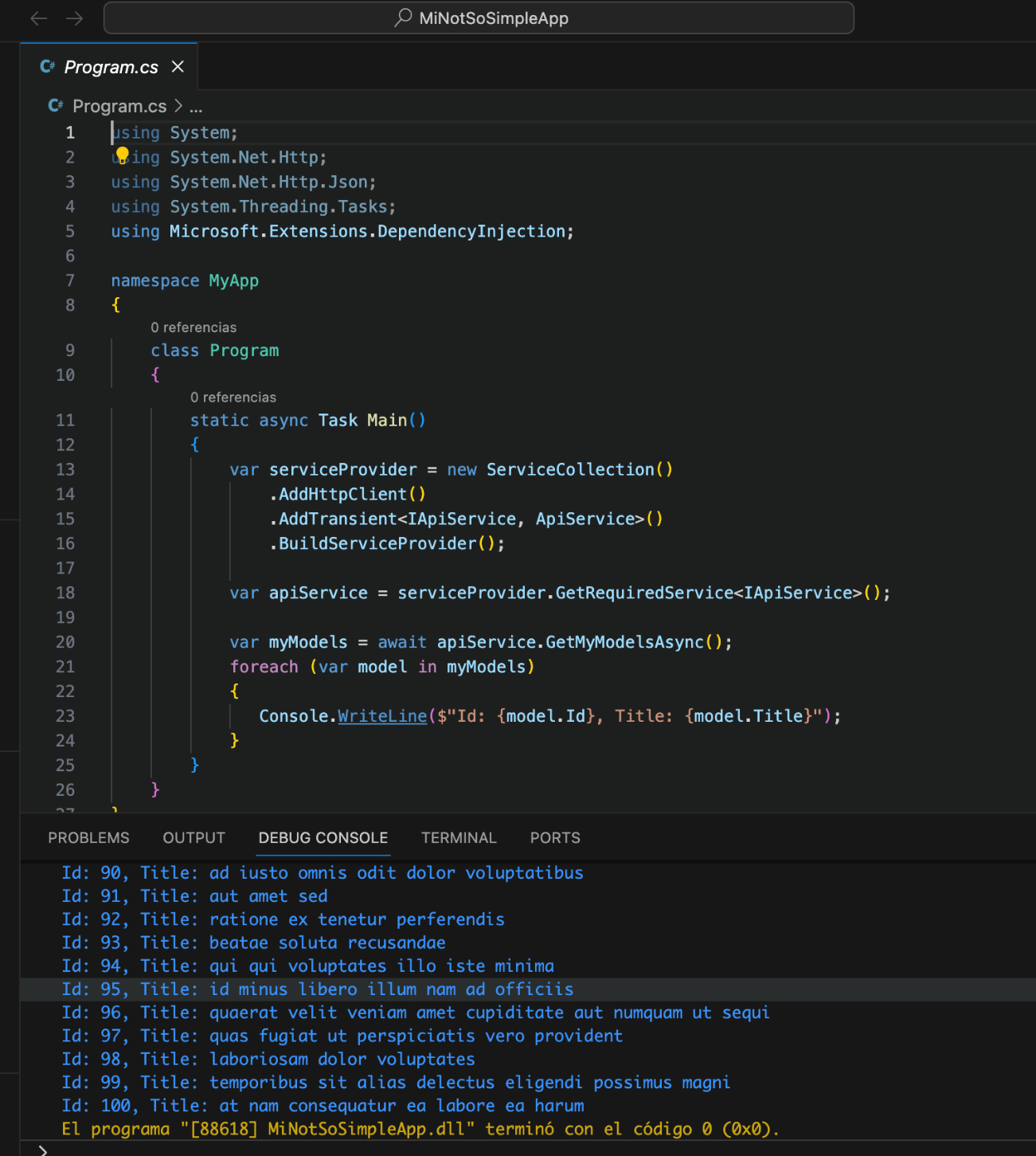
6.4 –

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente con confianza media

8 –

8.1 –



El código en Program.cs constituye una aplicación desarrollada en C# que establece conexión con un servicio externo mediante HTTP para obtener datos. Emplea la técnica de inyección de dependencias para configurar y acceder a un servicio denominado IApiService.

Para poner a prueba este código, es necesario crear una variante simulada de IApiService que proporcione datos ficticios en lugar de efectuar peticiones reales a un servicio externo. El servicio externo que se pretende "simular" en el código se refiere a la interfaz IApiService, la cual se emplea para realizar solicitudes HTTP y adquirir información externa.

Texto

Descripción generada automáticamenteTexto

Descripción generada automáticamente

8.3 –

En este ejemplo de prueba unitaria, se emplea Moq para emular el comportamiento de un servicio HTTP genuino al evaluar el método GetMyModelsAsync de la clase ApiService. En primer lugar, se crea un objeto llamado mockHttpMessageHandler que simula las respuestas HTTP. Este objeto se configura para devolver respuestas simuladas en lugar de efectuar solicitudes reales.

Luego, se realiza la configuración del contenedor de servicios para que utilice este objeto emulado en lugar del servicio HTTP auténtico al instanciar IApiService. De este modo, cuando se ejecute el caso de prueba, el método GetMyModelsAsync hará uso del servicio simulado, permitiendo así la verificación del comportamiento de ApiService sin depender de solicitudes HTTP reales. Esta estrategia se utiliza para confirmar que la lógica de ApiService funciona correctamente con las respuestas simuladas.

8.4 –

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

8.5 –

Pantalla de computadora con letras

Descripción generada automáticamente con confianza media

Captura de pantalla de computadora

Descripción generada automáticamente

< se modifico model.title para que develva asdasd222 en vez de Test Title y esto lo hace fallar, al volverlo a la normalidad se soluciona el problema>

8.6 –

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Básicamente se elimina la ultima línea del assert, ya que no verificamos contenido, solo cantidad, y modificamos que la cantidad sea 2, ya que previamente en la colección estábamos pidiendo 2.

9 –

Captura de pantalla de un celular

Descripción generada automáticamenteInterfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

<https://github.com/OctavioGarcia1337/unit-test-nsa/>