**Segundo Parcial PW3**

1. Un artículo en internet comenta que: Usar un ORM tiene las siguientes desventajas:
2. Las aplicaciones pueden ser mas lentas.
3. Se puede necesitar una inversión de tiempo de aprendizaje del ORM (como operarlo, como hacer consultas, etc.)

Por lo tanto, en general se desaconseja su uso”.

Indique si esta de acuerdo justificando claramente con sus palabras.

1. **Web Services: Indique Verdadero o Falso: Justifique si es falso.**
2. **En ASP .NET las propiedades se declaran con el atributo [WebMethod] sobre el nombre de cada propiedad.FALSO**

La afirmación es **incorrecta**. En **ASP.NET**, las propiedades **no** se declaran con el atributo [WebMethod].

El atributo [WebMethod] se utiliza en **ASP.NET Web Services (ASMX)** para exponer métodos de una clase como servicios web que pueden ser invocados a través de protocolos como **SOAP**. Este atributo se coloca **sobre los métodos**, no sobre las propiedades.

Las **propiedades** en C# o en el marco de trabajo **ASP.NET** (ya sea **Web Forms**, **MVC**, o **Core**) se definen de manera habitual sin el uso del atributo [WebMethod].

1. **No pueden utilizarse mediante el protocolo HTTP, ya que sería muy peligroso. FALSO**

La afirmación es **incorrecta**. En **ASP.NET Web Services (ASMX)**, los métodos marcados con [WebMethod] **sí pueden utilizarse mediante el protocolo HTTP**. De hecho, los **servicios web ASMX** fueron diseñados específicamente para ser accesibles a través de HTTP, utilizando el protocolo **SOAP** (Simple Object Access Protocol) para la comunicación.

Sin embargo, es cierto que utilizar **HTTP sin seguridad** puede ser peligroso, ya que la información se transmite en **texto plano**. Esto no es una limitación de la tecnología, sino una cuestión de seguridad en cómo se utiliza HTTP. Para proteger los datos sensibles, se recomienda utilizar **HTTPS** (HTTP Secure), que encripta las comunicaciones, evitando que los datos puedan ser interceptados o manipulados durante su transmisión.

**Seguridad con HTTP y servicios web**

* **HTTP (sin cifrado)**: Los datos se envían en texto plano, lo que hace posible que sean interceptados y leídos si se capturan las solicitudes o respuestas. Esto es inseguro para información sensible (como contraseñas o datos personales).
* **HTTPS (con cifrado)**: Utiliza **SSL/TLS** para cifrar la comunicación entre el cliente y el servidor. Esto garantiza que los datos no puedan ser interceptados o modificados por terceros, haciéndolo adecuado para servicios web que manejan información sensible.

1. **En ASP .NET se crean dentro de un archivo con extensión .asmx y debe estar si o si dentro del mismo proyecto MVC. FALSO**

La afirmación es **incorrecta**. Aunque es cierto que los **servicios web ASMX** en **ASP.NET** se crean dentro de archivos con la extensión **.asmx**, **no es obligatorio que estén dentro del mismo proyecto MVC**. Los servicios ASMX son una tecnología más antigua, anterior a **ASP.NET MVC**, y pueden residir en proyectos independientes.

**Detalles:**

1. **Archivos .asmx**: Los servicios web basados en **ASMX** se definen en archivos con la extensión **.asmx**. Estos archivos actúan como el punto de entrada del servicio web, y los métodos marcados con [WebMethod] dentro de ellos son los que se exponen como servicios.
2. **Independencia de proyectos MVC**: Los servicios web ASMX pueden estar en **cualquier tipo de proyecto web en ASP.NET**, y no es necesario que estén **dentro del mismo proyecto MVC**. De hecho, ASP.NET MVC y los servicios web ASMX son tecnologías separadas.
   * En proyectos modernos de **ASP.NET MVC**, o incluso en **ASP.NET Core**, lo más común es utilizar **Web APIs** (basadas en REST) para exponer servicios web, en lugar de utilizar ASMX.
   * Si se desea, un proyecto MVC puede consumir un servicio web ASMX que se encuentre en otro proyecto o incluso en otro servidor, ya que se accede a ellos mediante HTTP o HTTPS, igual que a cualquier otro servicio web.
3. **Evolución de tecnologías**: La tendencia actual es **migrar** o desarrollar servicios utilizando **ASP.NET Web API** o **gRPC** en lugar de ASMX, ya que estas tecnologías son más modernas, eficientes, y están mejor adaptadas para trabajar con el enfoque **RESTful** o con **protocolos más ligeros** que SOAP.
4. **Desde .NET puedo consumir un Web Service construido en php. VERDADERO**

Desde **.NET** puedes **consumir un Web Service construido en PHP** sin ningún problema, siempre y cuando el servicio expuesto siga los estándares de comunicación que .NET puede interpretar, como **SOAP** o **REST**.

**Detalles:**

1. **Servicios SOAP**:
   * Si el servicio PHP es un **Web Service basado en SOAP**, .NET puede consumirlo fácilmente utilizando herramientas como **Add Service Reference** en Visual Studio, que genera los proxies necesarios para interactuar con el servicio.
   * PHP puede crear servicios SOAP usando la extensión **SOAP de PHP**. Un servicio PHP expuesto mediante SOAP sigue un protocolo estándar que .NET comprende, ya que ambos soportan las mismas especificaciones de SOAP.
2. **Servicios REST**:
   * Si el servicio PHP es un **API RESTful**, .NET también puede consumirlo utilizando clases como HttpClient para realizar solicitudes HTTP (GET, POST, PUT, DELETE, etc.). Aquí no se utiliza SOAP, sino que el servicio devuelve respuestas generalmente en formato **JSON** o **XML**, que luego pueden ser procesadas en la aplicación .NET.
   * En este caso, el servicio PHP puede estar usando frameworks como **Laravel**, **Symfony**, o incluso ser un servicio PHP básico que responde a solicitudes HTTP.
3. **Explique con sus palabras que significa el término “Lazy loading” o “Carga Perezosa” en relación a LINQ ¿Para que sirve?**

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. **Entity Framework: Indique Verdadero o Falso: Justifique si es falso.**
2. **LINQ solo se puede aplicar sobre el origen de datos Entity Framework (contexto). FALSO**

**Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja** **Captura de pantalla de un celular con texto

Descripción generada automáticamente**

La afirmación es incorrecta. **LINQ (Language Integrated Query)** no se limita a **Entity Framework** ni a un contexto de base de datos en particular. LINQ es una característica del lenguaje .NET que permite realizar consultas de manera declarativa sobre una amplia variedad de **orígenes de datos**. Entity Framework es solo uno de ellos.

LINQ se puede aplicar a diferentes tipos de orígenes de datos, como:

1. **Colecciones en memoria**: Puedes usar LINQ para consultar colecciones como **listas** (List<T>), **arrays** o cualquier colección que implemente la interfaz IEnumerable<T> o IQueryable<T>. Este es conocido como **LINQ to Objects**.
2. **Bases de datos**: LINQ puede utilizarse en combinación con **Entity Framework** para consultar bases de datos. A esto se le llama **LINQ to Entities**, que permite realizar consultas sobre el contexto de la base de datos que EF maneja.
3. **XML**: LINQ ofrece la capacidad de consultar y modificar datos XML usando **LINQ to XML**, que permite manipular archivos o estructuras XML de forma sencilla.
4. **SQL (LINQ to SQL)**: Aunque LINQ to SQL es otra tecnología diferente a Entity Framework, también permite realizar consultas SQL a través de LINQ, pero está más orientado a trabajar con bases de datos SQL Server de manera directa.
5. **Otros proveedores**: Existen extensiones y bibliotecas que permiten utilizar LINQ para acceder a otras fuentes de datos, como **LINQ to JSON** (para manipular datos JSON), **LINQ to DataSet**, y más.

En resumen, LINQ es una herramienta **versátil y generalizada** que se puede aplicar a una gran variedad de fuentes de datos, no solo a **Entity Framework**.

1. **Una de las desventajas de EF es que no soporta herencia.**

La afirmación es incorrecta. Una de las características **clave** de **Entity Framework (EF)** es que **sí soporta herencia** en modelos de datos. De hecho, Entity Framework ofrece varias formas de trabajar con la herencia en bases de datos, permitiendo mapear clases en C# o VB.NET que utilizan herencia a tablas en una base de datos relacional.

Existen tres estrategias principales que se pueden usar con EF para mapear herencia:

1. **Table per Hierarchy (TPH)**: En esta estrategia, todas las clases de una jerarquía de herencia se almacenan en una sola tabla, y se distingue entre las diferentes clases mediante una columna discriminadora. Es la estrategia predeterminada de EF y es eficiente en términos de número de tablas, aunque puede generar tablas con muchas columnas (al combinar los atributos de todas las clases).
2. **Table per Type (TPT)**: En este enfoque, cada clase de la jerarquía de herencia se mapea a una tabla distinta. Cada tabla contiene solo los atributos específicos de su clase y una clave externa (foreign key) para relacionarla con la tabla base. Este método puede ser más claro, pero introduce un overhead en las consultas, ya que implica varias uniones (joins).
3. **Table per Concrete Type (TPC)**: En esta estrategia, cada clase concreta de la jerarquía tiene su propia tabla, y no se utiliza una tabla para la clase base. Las tablas no comparten columnas comunes, lo que significa que los datos comunes se duplican en cada tabla.
4. **En LINQ to entities existen dos tipos de consulta: 1) Sintaxis de expresiones de consulta (LinQ) y 2) Sintaxis de consulta basada en eventos (con expresiones gamma) FALSO**

Basada en métodos y con expresiones lambdas.

Interfaz de usuario gráfica, Texto, Aplicación

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

Descripción generada automáticamente

1. **Los enfoques de EF son 2, model first y database first, en este ultimo se debe partir de un modelo relacional correcto. FALSO**

Los enfoques de EF son 3, model first, database first y code first.

1. **a) Defina ORM**

Una captura de pantalla de un celular

Descripción generada automáticamente con confianza media

1. **Indique y explique cuales son los enfoques.**

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente