Evaluación Técnica de Conocimientos "Confiamed"

Fecha: 10/02/2024

Nombres: Anthony Larrea

Parte Teórica (10 minutos)

1. Explique el uso de las palabras claves **ASYNC y AWAIT**, ¿En qué afectan al programa?

El método async permite que una función sea ejecutada de manera asíncrona, es decir, sin bloquear el flujo del programa. await se usa dentro de una función async para esperar a que una promesa se resuelva, sin bloquear el hilo principal, pero sí detiene la ejecución dentro de esa función hasta obtener una respuesta.

2. Qué ventajas tiene el uso de realizar clases que reciban parámetros genéricos, e.g. `class Documento <T>`, y cuándo debería ser usado:

Las clases genéricas permiten reutilizar código para diferentes tipos de datos, sin tener que definir una clase para cada tipo. Esto se usa cuando queremos que una clase funcione con diferentes tipos de datos sin importar su tipo específico.

3. Explique lo que es una comunicación **SÍNCRONA y ASÍNCRONA**:

La comunicación **síncrona** es cuando el programa debe esperar a que se complete una operación antes de continuar. La **asíncrona** permite que otras tareas sigan ejecutándose mientras se espera la respuesta de una operación.

4. Explique cuál es la diferencia entre **OBSERVER y PROMISE**:

Un **Observer** es una entidad que observa cambios en un conjunto de datos y reacciona a esos cambios en tiempo real. Una **Promise** representa un valor que puede estar disponible ahora, en el futuro o nunca, y se resuelve una sola vez.

5. Detalle **VENTAJAS y DESVENTAJAS** de usar microservicios:

   Ventajas: Los microservicios permiten que las diferentes partes del sistema trabajen de manera independiente; si una falla, las demás siguen funcionando.

   Desventajas: Puede ser complicado gestionarlos si hay muchos y su comunicación puede volverse compleja.

6. Describa la idea general de una **ARQUITECTURA DE MICROSERVICIOS**:

La arquitectura de microservicios divide una aplicación en servicios pequeños e independientes que se pueden desarrollar, desplegar y escalar de manera individual sin afectar al sistema completo.

7. Explique lo que es el formato **JSON** y contrástelo con **XML**:

   JSON es un formato de intercambio de datos más ligero y fácil de leer/escribir para los humanos y las máquinas.

   XML es más detallado y suele usarse para estructurar datos con etiquetas. JSON es más utilizado hoy en día por su simplicidad y menor peso.

8. Explique lo que es una lectura sucia de una base datos; sus **VENTAJAS** y **DESVENTAJAS**:

   Una **lectura sucia** ocurre cuando se leen datos no confirmados o no permanentes de una transacción, lo que puede llevar a inconsistencias si esa transacción falla. Ventajas: Puede ayudar en pruebas. Desventajas: Puede causar errores si esos datos se usan como definitivos.

9. Explique cómo funciona una **CONSULTA** a una base de datos:

   Una consulta extrae información de la base de datos usando lenguaje SQL. Se pueden combinar varias tablas con joins y activar eventos automáticos con triggers para facilitar las operaciones.

10. Cuáles son los tipos de **ELIMINACIÓN** de registros dentro de una base de datos:

    DELETE: Elimina registros específicos, pero deja la estructura de la tabla.

    TRUNCATE: Elimina todos los registros de una tabla, pero la tabla sigue existiendo.

    DROP: Elimina completamente una tabla de la base de datos.

11. Que **DESVENTAJAS** tiene realizar programas que ejecuten código de forma paralela y en qué escenarios se puede considerar este tipo de programación:

    Las desventajas incluyen la dificultad para depurar y manejar datos compartidos correctamente. Es útil cuando se necesita procesar muchas tareas al mismo tiempo, como en aplicaciones que manejan gran cantidad de usuarios simultáneamente.

12. Indique cuál es la mejor forma de manejar datos sensibles en un aplicativo, como por ejemplo claves o cadenas de conexión.

    Se pueden usar técnicas como el cifrado y servicios de autenticación como **JWT**. Además, es recomendable almacenar claves de acceso en archivos seguros, fuera del código fuente, como archivos de entorno.

13. ¿Es recomendable enviar credenciales (login) en una petición **GET** como parámetros de consulta?
    No es recomendable usar **GET** para enviar credenciales, ya que los parámetros de consulta se almacenan en la URL, lo que puede comprometer la seguridad. Es mejor usar **POST** para ese tipo de datos.
14. ¿Qué haría para prevenir un ataque de inyección de SQL?
    Para prevenir inyecciones de SQL, es importante usar **sentencias preparadas** o **ORMs** (Objetos Relacionales de Mapeo) en lugar de concatenar consultas SQL directamente en el código.

15. Explique lo que es el **CORS** y cómo se debe usar para mejorar la seguridad de los aplicativos

    Es una política de seguridad que controla qué dominios pueden hacer peticiones a un servidor. Se usa para evitar que sitios no autorizados accedan a recursos de una aplicación.

16. Explique cuál es la diferencia entre una prueba de integración y una prueba unitaria
    Las **pruebas unitarias** verifican una pequeña parte del código (una función o método) de manera aislada.
    Las **pruebas de integración** verifican que varios componentes del sistema trabajen juntos correctamente.

17. ¿Cuál es la diferencia entre una prueba de carga, estrés y concurrencia?

    Pruebas de carga: Miden el rendimiento del sistema bajo condiciones normales de uso.

    Pruebas de estrés: Verifican cómo se comporta el sistema bajo un uso extremo.

    Pruebas de concurrencia: Miden cómo el sistema maneja múltiples usuarios simultáneamente.

Parte Práctica (2 horas)

Instrucciones iniciales

1. Cree una carpeta con su "NOMBRE_APELLIDO" en la siguiente ubicación: C:\repos 2.
Cree una base de datos con su NOMBRE_APELLIDO Utilice la instancia local SQLSERVER "MSSQLLocalDB".
3. **NO se permite el uso de generadores de código ni AI generativa.**

Desarrollo

Realizar un aplicativo web que cargue formularios de manera dinámica, cada formulario tendrá asociado uno o varios inputs de diferentes tipos.

Iniciar el ejercicio tomando en cuenta dos formularios principales, cada formulario será representado por un botón en la página web:

- El primer botón cargará un formulario para ingresar datos básicos de una persona, como nombres, fecha de nacimiento, estatura, etc.
- El segundo botón cargará un formulario para ingresar datos de su mascota, como especie, raza, color, nombre, etc.

Los formularios pueden ser cambiados de manera dinámica, lo que implica que se pueden agregar, retirar o modificar el tipo de dato de los campos según lo necesite el administrador de la página.

Tomar en cuenta un buen diseño de APIS, frontend y base de datos que permitan ESCALABILIDAD y MANTENIBILIDAD.

CRITERIOS DE ACEPTACIÓN

- Los botones cargados para mostrar los formularios también deben ser obtenidos desde la base de datos.
- Contemplar APIs para listar, crear, modificar y eliminar formularios e inputs.
- El front deberá ser realizado en Angular y el backend en net core.
- NO incorporar validaciones de los inputs, el estilo visual tampoco es una prioridad.

BONUS

- Incorporar botones en los formularios para mostrar la información ingresada.
- Incorporar validaciones básicas.
- Colocar el back en un contenedor.

Instrucciones finales

1. En el explorador, ubíquese en la carpeta del proyecto y abra el GIT BASH
2. Inicie el repositorio GIT mediante comandos
3. Agregue los cambios al stage local y confirme los cambios
4. Cree una rama con el comando `git branch NOMBRE_APELLIDO`
5. Use el comando para cambiar de rama `git checkout NOMBRE_APELLIDO`
6. Agregue el repositorio remoto con el siguiente comando `git remote add origin https://logicstudiosa.visualstudio.com/Evaluacion%20Web%20Confiamed/_git/Evaluacion %20Web%20Confiamed`
7. Versione el codigo fuente hacia el repositorio remoto usando el comando `git push origin NOMBRE_APELLIDO`