Indicaciones Generales

- Los grupos deberán tener entre 3 y 4 integrantes.
- El trabajo obligatorio consistirá en realizar :

  Un Portal en Angular que invoque a la ASP.NET Web API.

  Una .NET Solution en el lenguaje de programación C# utilizando las tecnologías Winforms, ASP.NET Web API, ADO.NET y ASP.NET Webservices, existiendo la posibilidad de negociar con el profesor el uso de otras tecnologías.

  Opcionalmente se puede añadir también un cliente con Cordova y Angular que genere una App para Android.

  Realizar una demo del obligatorio por Zoom de 15 minutos, en horario de clase.
- La fecha de entrega será el 25 de noviembre de 2021, en horario de clase.
- El trabajo obligatorio vale 100 puntos y aporta el 25% de la calificación del curso.
- Se debe obtener un mínimo de 60 puntos para aspirar a la aprobación del curso.

Planteo del Problema

Opción a)

Se desea realizar un sistema que permita almacenar y revisar listados de Albums, Canciones, Bandas, Integrantes, siguiendo la siguiente realidad de negocio que se describe a continuación.

Album: Nombre, Año Creación, Banda, Canciones (lista), GéneroMusical

Banda: Nombre, GéneroMusical, AñoCreación, AñoSeparación, Integrantes (lista)

Canción: Nombre, Albums (pueden ser varios), Duración, Año, GéneroMusical, Data (contenido mp4), Cantante (de tipo Integrante)

Integrante: Nombre, Apellido, FechaNacimiento, Foto.

Requerimientos en Aplicación Winform que invoca un WebService SOAP:

- ABM de Album
- ABM de Banda
- Añadir y quitar Integrante a Banda
- Añadir y quitar Canción a Album
- ABM de Canción (sin incluir data)
- ABM de Integrante (incluye foto)

Requerimientos en Sitio Angular que invoca la Web API REST:

- Login y Log-out
- Página simple de Registrarme (nombre usuario y password)
- Distintas vistas con filtros y listados de
    - Albums
    - Bandas
    - Canciones
    - Integrantes
    - (En todas las anteriores se requiere estar logueado al sitio)
- Votar del 1 al 5 y hacer una breve reseña en texto sobre los ítems
    - Bandas
    - Canciones
- Opcionales que usted desee agregar.

Opción b) un sistema que en lugar de tener como entidades a Album, Banda, Cancion, Integrante tenga a otras cuatro entidades de un tipo de línea de negocio distinta.

Ejemplo: Veterinaria, Dueño, Mascota, CarnetInscripcion

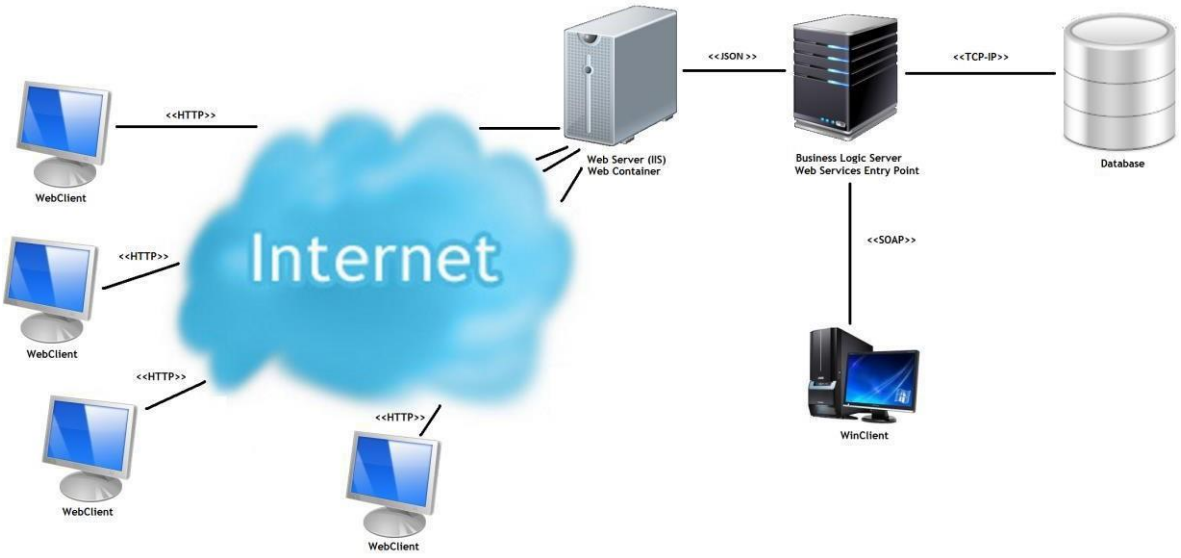
Otro ejemplo: Producto, CarritoCompras, ListaPrecios, etc.

Arquitectura

El sistema debe tener una arquitectura lógica en 3 capas y una arquitectura física N-Tier, encontrándose entre los Tiers como mínimo:

- Un componente servidor, el cual tendrá la lógica y persistencia del sistema.
- Una base de datos en el motor SQL Server 2019 Express, la cual será accedida por el componente servidor, ambos componentes deben estar en la misma LAN.
- Un componente interfaz de usuario en modo ventanas que servirá para mantener las codigueras de la aplicación. Se conectará via ASP.NET Web Services con el componente servidor. Podrá haber N instancias de este componente.
- Un componente servidor web (Internet Information Services) el cual se usará como web-server para hostear la aplicación web Angular del siguiente componente.
- Un componente Angular, el cual contendrá un proyecto web, el cual luego se desplegará en un directorio virtual a correr en la máquina con el IIS. Dicho componente se comunicará via Web API con el componente servidor.
- Un componente web-service y un componente Web API que no formarán parte de un nuevo tier pues tendrán que estar físicamente en la misma máquina que el componente servidor.
- (Opcionalmente) un componente mobile con Cordova y Angular que permitirá embeber en una App Android un sitio Angular pensado para generarse como una App que simula capacidades nativas.

Diagrama Arquitectura Física de la aplicación (propuesta genérica)



Orden sugerido de resolución

Fase de Análisis Previo

1. Realice un Diagrama de Clases Conceptual en UML que indique como se relacionan las entidades de la realidad planteada.
2. Realice un Diagrama de Arquitectura Física N-Tier, en donde muestre como se podrán distribuir los componentes del sistema.
3. Documente cuáles serán los protocolos de intercambio de información entre componentes, en donde habrá web-services o webapi, cuál será el componente servidor, cuál o cuáles componentes tendrán acceso a la base de datos, etc.

Fase de Diseño Previo

4. Documente la Elección del Diseño basándose en el Diagrama de Clases Conceptual realizado y los requerimientos del sistema.
5. Realice el Diagrama de Clases de Implementación en UML basándose en partes anteriores, pensando en que sigue diseñando un sistema en donde tendría colecciones con objetos en memoria. Hágalo solo para las clases de la capa lógica del componente servidor.
6. Suponiendo que utilizará el patrón Facade como punto de entrada a la capa lógica en el componente servidor, diseñe los Value Objects que se utilizarán para enviar información a los otros componentes. También defina las interfaces de la clase Facade (IFacadeWeb, IFacadeWin) y de los WebServices.

Fase de Implementación Business Server - WinClient

7. Programe los WebServices ASP.NET y Web API tontamente, exponiendo solo su interfaz y genere los WSDL correspondientes.
8. Programe la llamada desde los WebServices y Web API a la fachada singleton.
9. Implemente el componente business server en cuanto a su capa lógica y en lugar de programar los DAO y la capa de persistencia, utilice por ahora, colecciones de objetos en memoria.
10. Implemente un proyecto en consola TestModel, el cual permita invocar los requerimientos de Alta-Baja-Modificación del business server y realice un main que pueble el sistema con datos de prueba.
11. Valide con el tutor e implemente las pantallas del componente WinClient, debe utilizar controladores e implementar lo mejor posible Model-View-Controller.
12. Establezca un Proxy con el WebService de punto de entrada al servidor desde el componente WinClient y pruebe las funcionalidades del componente.

Fase de Rediseño, pasaje a DAOs

13. Identifique cuales serían los DAOs que surgirían en este sistema si se quisiera aplicar el patrón de diseño DAO. Documente cuáles son los DAOs y qué atributos tienen.
14. Documente cuales serían los cambios a nivel de encabezado de los métodos que realizará para sustituir el concepto de colecciones de objetos en memoria por DAOs en todas las clases involucradas (colecciones, fachada, clases de negocio).
15. Realice un Diagrama de Clases de Implementación en donde distinga qué clases estarán dentro de la capa lógica y qué clases habrá en la capa de persistencia. Para este diagrama considere en que se aplicará el patrón DAO sin AbstractFactory.

Fase de Creación de Esquema BD y Mapeo Objetos-Relación

16. Realice un Modelo Entidad Relación (MER) que se corresponda con el Diagrama de Clases Conceptual, la Elección del Diseño de puntos anteriores y los DAO encontrados en el punto anterior, considerando los mapeos Objeto-Relación que surgirían.
17. Haga un script de creación de la base de datos, el cual se pueda volver a ejecutar y autoregenere el esquema por entero.
18. Implemente las modificaciones a la capa lógica y de persistencia para tener clases DAO en lugar de objetos en memoria. Para las consultas y sentencias SQL que sean de alta-baja-modificación, tiene permitido utilizar DataSets, pero ningún DataSet puede salir del componente servidor hacia fuera via WebServices o WebAPI, debe ser convertido a una colección de Value Objects. Los accesos a persistencia via DAOs para resolver requerimientos de negocio, deben ser implementados usando Transacciones y SQL estándar. El acceso a BD debe hacerse mediante ADO.NET y se debe dejar configurado en el web.config del proyecto WebServices, el ConnectionString y demás datos de configuración.

Fase de Implementación del componente Angular

19. Diseñe y valide la estética, esquema de navegación del sitio y funcionalidad de las páginas del sitio Angular.
20. Diseñe e implemente la página de login del sitio al que accederán los clientes, luego el botón de logout y la página de Registro.
21. Implemente las páginas web, controladores, servicios y directivas del sitio Angular.