1. **Defina qué es análisis y diseño de sistemas, y describa los pasos de una de las**

**metodologías que conozca.**

Es un proceso por medio del cual recabamos información a cerca de un problema dado, esta información nos sirve para profundizar el porque del problema e identificar las necesidades para luego plantear una o varias soluciones por medio del desarrollo de un sistema de información.

Pasos de SCRUM.

Una de las metodologías que he tenido la oportunidad de trabajar es SCRUM por medio de Azure.

**Inicio:** Analizar e Identificar las necesidades del sprint.

**Planificación y estimación**: Realizar la planificación, estimación y compromiso con historias de usuario para los diferentes integrantes del equipo, identificar las tareas asignadas.

**Implementación:** En esta etapa se crean los entregables y se lleva a cabo todo lo planificado.

**Revisión y retrospectiva:** En esta fase de demuestra y valida lo implementado por cada integrante del equipo, se discute los atrasos, que salió bien y que se puede mejorar para el siguiente sprint.

**Lanzamiento:** Esta fase es la entrega del producto final, puesta en producción, se envían todos los entregables y posibles mejoras.

Cabe resaltar que en cada una de las etapas se realizan reuniones diarias para avances y/o atrasos de algunos de los integrantes del equipo, esto es muy importante para un trabajo efectivo.

1. **Explique la importancia del diseño de pantallas, y que haría para que sean fáciles de aprender a utilizar, eficientes, confiables y den satisfacción a los usuarios.**

Para una experiencia agradable para el usuario es necesario el diseño de pantallas, el desarrollo de prototipos iniciales para que el usuario pueda interactuar con el producto y dar el visto bueno, es importante que las pantallas sean amigables para el usuario fácil de utilizar y que el usuario se sienta cómodo con ello para esto usaría lo que se conoce como UX (user xperience) que su objetivo es entregar al usuario soluciones intuitivas, eficientes y bonitas para que la experiencia del mismo sea satisfactoria.

1. **¿Qué es una transacción de base de datos?**

Es una unidad de trabajo en la que se realizan varias operaciones de base de datos, su objetivo principal es asegurar la consistencia, ya que al ser varias operaciones y en el momento de alguna falla se pierde la consistencia, una transacción asegura que se realicen todas las operaciones o ninguna, únicamente se asegura el cambio si la transacción tiene éxito.

1. **Escriba un documento XML que permita intercambiar información de tarjetas de crédito entre dos sistemas distintos: nombre del sistema, localización, número de tarjeta, ID único de cliente, etc.**

<?xml version="1.0" encoding="UTF-8"?>

<credit-card-info xmlns="http://www.prueba-promerica.gt/credit-card-info">

<info>

<sistema>PRUEBA</Sistema>

<client\_id>1234567</client\_id>

<card\_no>1020</card\_no>

<card\_brand>1020</card\_brand>

<card\_class>NI</card\_class>

<emission\_date>12/12/2023</emission\_date>

<due\_date>202612</due\_date>

<credit\_type>ON</credit\_type>

<status>A</status>

<limit>10000</limit>

<limit\_daily>1000</limit\_daily>

<limit\_month>20000</limit\_month>

<dir>Guatemala, Guatemala</dir>

<payment\_date>15/01/2024</payment\_date>

<cutoff\_date>24/12/2023</cutoff\_date>

<currency>978</currency>

<currency\_code>GTQ</currency\_code>

</info>

</ credit-card-info >

1. **Defina y describa el concepto de Web Service.**

Es una tecnología que nos permite el intercambio de información entre 2 aplicaciones a través de internet por medio del protocolo HTTP, el intercambio de información se realiza a través del protocolo SOAP (xml detallado) o a través de servicios rest (json), la gran ventaja de utilizar web service es que la información se puede intercambiar sin importar la tecnología en la que fue desarrollado por lo que es independiente de estos.

1. **¿Qué es AJAX?**

Es un conjunto de tecnologías javascript que se utiliza para realizar peticiones a servicios web, se obtiene información del servidor en segundo plano, es decir es una petición asíncrona por lo que se puede actualizar la información sin necesidad de recargar la página.

1. **Defina el concepto de Variable de Sesión**

Es una variable global que se define en el inicio de sesión y que es persistente es decir se puede acceder a ella desde cualquier pagina dentro del proyecto.

1. **Defina el concepto de Template (o MasterPage hablando de .NET)**

Es la pagina principal del proyecto, toda la estructura base para todas las demás paginas que pueden heredar esta estructura, por ejemplo en ella se puede definir un header y footer general y que las demás paginas que pertenecen al proyecto puedan heredarlo.

1. **¿Qué es un Postback? ¿Qué es un Callback?**

Un postback ocurre cuando los datos (la página completa) en la página se publican desde el cliente al servidor, es decir, los datos se devuelven al servidor y, por lo tanto, la página se actualiza, es como enviar al servidor la pagina asp completa llena de datos.

Un callback es similar con la diferencia que no se actualiza la pagina, solo se llaman y se reciben datos del servidor.

1. **Describa el concepto de SSL**

Es un protocolo de seguridad, proporciona un medio seguro para el intercambio de información entre 2 dispositivos a través de internet, cifrando y descifrando la información enviada.

1. **Escriba una página Web en lenguaje HTML y el respectivo código Javascript (si fuera necesario) que solicite el nombre del usuario, contraseña, y que valide que los campos no vayan vacíos antes de enviar la solicitud al servidor. Además, ¿Qué es lo primero que haría del lado del servidor al recibir esta información y por qué?**

Archivo html adjunto login-geko.html.

Lo primero que haría del lado del servidor es volver a validar la información si bien ya existe una validación del lado del cliente está es fácil de manipular y pasar, es necesario volver a validar para tener mejor seguridad, luego de validar verificar que las credenciales sean correctas y en caso de que no coincidan enviar un mensaje al usuario especificando que las credenciales son incorrectas.

--A Escriba la consulta en SQL que devuelva el nombre del proyecto y sus productos correspondientes del proyecto Premia cuyo código es 1

SELECT P.NOMBRE, PPP.DESCRIPCION AS 'NOMBRE PRODUCTO'

FROM PROYECTO P

INNER JOIN PRODUCTO\_PROYECTO PP ON P.COD\_PROYECTO = PP.COD\_PROYECTO

INNER JOIN PRODUCTO PPP ON PP.COD\_PRODUCTO = PPP.COD\_PRODUCTO

WHERE P.COD\_PROYECTO = 1;

--B Escriba una consulta SQL que devuelva los distintos mensajes que hay, indicando a qué proyecto y producto pertenecen.

SELECT M.COD\_FORMATO, F.NOMBRE AS 'FORMATO MENSAJE', M.COD\_MENSAJE, PPP.NOMBRE AS 'NOMBRE PROYECTO', P.DESCRIPCION AS 'NOMBRE PRODUCTO'FROM MENSAJE M

INNER JOIN PRODUCTO\_PROYECTO PP ON M.COD\_PRODUCTO\_PROYECTO = PP.COD\_PRODUCTO\_PROYECTO

INNER JOIN PRODUCTO P ON PP.COD\_PRODUCTO = P.COD\_PRODUCTO

INNER JOIN PROYECTO PPP ON PP.COD\_PROYECTO = PPP.COD\_PROYECTO

INNER JOIN FORMATO\_MENSAJE F ON M.COD\_FORMATO = F.COD\_FORMATO

/\*

Escriba una consulta SQL que devuelva los distintos mensajes que hay, indicando a qué proyecto y producto pertenecen. Pero si el mensaje está en todos los productos de un

proyecto, en lugar de mostrar cada producto, debe mostrar el nombre del proyecto y un solo producto que diga “TODOS”.

\*/

SELECT DISTINCT PROY.NOMBRE,

CASE

WHEN SUBP.TOTAL\_PROD >= (SELECT COUNT(\*) FROM PRODUCTO) THEN 'TODOS'

ELSE PRO.DESCRIPCION

END AS PRODUCTO

FROM (

SELECT pro.COD\_PROYECTO, me.COD\_FORMATO, count(DISTINCT pro.COD\_PRODUCTO) as TOTAL\_PROD from PRODUCTO\_PROYECTO pro

inner join MENSAJE me on me.COD\_PRODUCTO\_PROYECTO = pro.COD\_PRODUCTO\_PROYECTO

group by pro.COD\_PROYECTO, me.COD\_FORMATO) AS SUBP

INNER JOIN PRODUCTO\_PROYECTO PROP ON SUBP.COD\_PROYECTO = PROP.COD\_PROYECTO

INNER JOIN PRODUCTO PRO ON PROP.COD\_PRODUCTO = PRO.COD\_PRODUCTO

INNER JOIN PROYECTO PROY ON PROY.COD\_PROYECTO = PROP.COD\_PROYECTO

1. Código C# adjunto
2. Diagrama Er adjunto en carpeta.
3. **Tomando de referencia el siguiente caso:**

**Una entidad bancaria desea mejorar las opciones de pago de seguros, actualmente la**

**aseguradora le envía al banco en un archivo los saldos de las cuentas aseguradas, el banco**

**al final del día le envía a la aseguradora otro archivo con los pagos realizados. Para esto**

**desea conectarse directamente con la aseguradora para poder obtener los saldos**

**actualizados, y que la aseguradora también tenga actualizados los pagos en el mismo**

**instante en que se realizan en el banco. La conexión entre los sistemas se hará a través de**

**Internet, debe utilizar tecnologías .NET para el desarrollo en ambos extremos de la**

**comunicación (banco y aseguradora).**

**Nombrar las tecnologías, protocolos, medidas de seguridad, consideraciones que se deben**

**de tomaren cuenta, y cualquier otro punto que considere necesario para su**

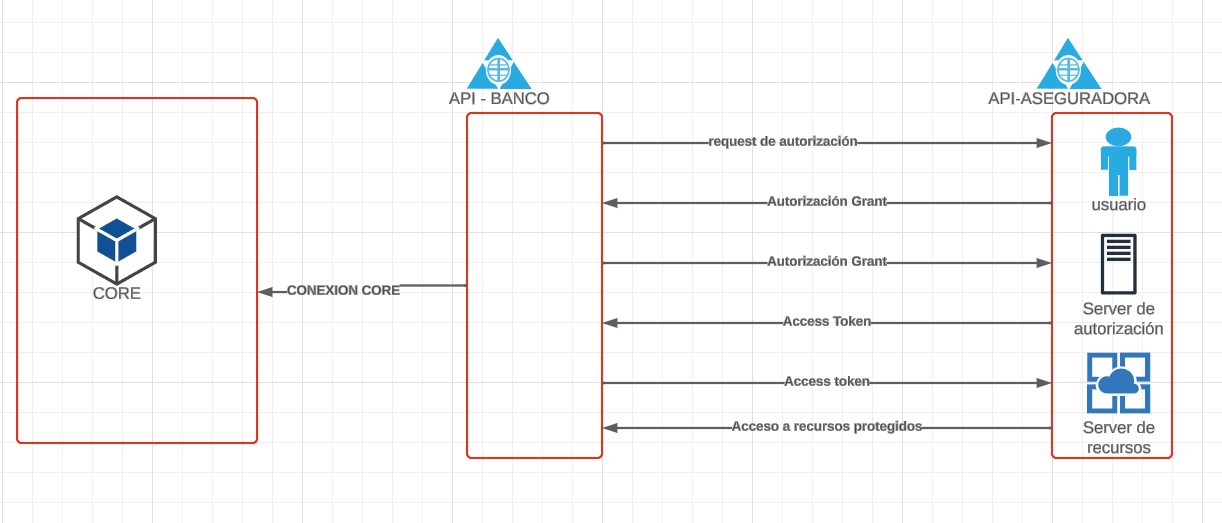
**implementación, además describa brevemente la función que tendría cada una de ellas.**

**Debe de verse desde el punto de vista de que usted trabaja para el banco, y que la**

**aseguradora solicita que usted sea la persona que defina todo el sistema. Si lo considera**

**necesario puede agregar un diagrama sencillo.**

**Diagrama.**



**EXPLICACIÓN.**

Se propone construir dos poderosas API desarrollada en C# .NET CORE tanto para el banco como para la asegurador, que funcionen como interfaz para no acceder directamente al Core bancario, esta interfaz contará con todas la medidas de seguridad autenticando y protegiendo los recursos por medio de IDENTITY SERVER que funciona bajo el estándar oauth 2.0

**ASP.NET Core Identity** SERVER proporciona API que controlan la autenticación, la autorización y la administración de identidades. Las API permiten proteger los puntos de conexión de un back-end de API web con autenticación basada en cookie. Existe una opción basada en tokens para los clientes que no pueden usar cookie.

Tal como se muestra en el diagrama.

**¿Cómo funciona OAuth 2.0?**

En el nivel más básico, antes de poder utilizar OAuth 2.0, el cliente debe adquirir sus propias credenciales, un id de cliente y un client secret, del servidor de autorización para identificarse y autenticarse al solicitar un token de acceso.

Con OAuth 2.0, las solicitudes de acceso son iniciadas por el cliente, por ejemplo, una aplicación móvil, un sitio web, una aplicación de televisión inteligente, una aplicación de escritorio, etc. La solicitud, el intercambio y la respuesta de los tokens siguen el siguiente flujo general:

1. El cliente solicita autorización (solicitud de autorización) al servidor de autorización, proporcionando el id y el client secret como identificación; también proporciona los ámbitos y un URI de extremo (URI de redireccionamiento) al que enviar el token de acceso o el código de autorización.
2. El servidor de autorización autentica al cliente y verifica que los ámbitos solicitados están permitidos.
3. El propietario del recurso interactúa con el servidor de autorización para conceder el acceso.
4. El servidor de autorización redirige de vuelta al cliente con un código de autorización o un token de acceso, según el tipo de concesión, También puede devolverse un token de actualización.
5. Con el token de acceso, el cliente solicita acceso al recurso desde el servidor de recursos.

En el diagrama se muestra la comunicación entre el banco y la aseguradora, pero la idea es realizar lo mismo cuando la aseguradora solicite recursos del banco.

Lo servicios necesarios para que exista una comunicación efectiva.

**Autenticación de usuario.**

Este servicio debe existir tanto en nuestra api-banco como en nuestra api-aseguradora.

Por medio de credenciales de acceso el cliente (ya sea el banco y/o aseguradora) solicitara acceso a nuestro servidor de autenticación, si es un usuario valido se le da acceso al cliente a los recursos.

**Lista de cuentas aseguradas y saldos.**

Nuestra api-aseguradora, debe exponer un servicio al api-banco para obtener la lista de cuentas aseguradas con su respectivos saldos para que el banco pueda proceder al pago, al momento de realizar el pago, api-banco enviará una notificación en tiempo real al api-aseguradora por medio de la mensajeria en la nube de firebase.

**Mensajería en la nube de Firebase**

es una solución de mensajería multiplataforma que le permite enviar mensajes de manera confiable y sin costo.

Con FCM, puede notificar a una aplicación cliente en este caso api-banco que hay nuevos pagos efectuados u otros datos disponibles para sincronizar la información. Puede enviar mensajes de notificación para impulsar el consumo del servicio lista de cuentas y pagos que proporcionará el api-banco.

**Lista de cuentas y pagos.**

Nuestra api-banco debe exponer un servicio al api-aseguradora para obtener la lista de pagos con la respectiva información (este se puede activar en el momento que se reciba la notificación del pago realizado).

**CONCLUSION.**

De esta forma cubrimos el requerimiento garantizando la seguridad a través de identity server y oauth 2, sin comprometer el acceso al CORE bancario y al CORE de aseguradora que cuentan con sus propias medidas de seguridad, únicamente nuestra solución contará con nuestras apis que funcionan como interfaz de comunicación entre ambas.