**EJERCICIO 2**

Las siguientes preguntas están orientadas a la comprensión del protocolo HTTP. Son agnósticas al lenguaje de programación, la idea es comprender los conceptos del estándar:

1. **¿Qué es un servidor HTTP?**

Un servidor Web es un programa que utiliza HTTP (Hypertext Transfer Protocol) para servir los archivos que forman páginas Web a los usuarios, en respuesta a sus solicitudes, que son reenviados por los clientes HTTP de sus computadoras. Las computadoras y los dispositivos dedicados también pueden denominarse servidores Web.

1. **¿Qué son los verbos HTTP? Mencionar los más conocidos**

Estos verbos indican qué acción queremos realizar sobre el servidor y son GET, POST, PUT, PATCH, DELETE, HEAD, CONNECT, OPTIONS y TRACE. Cada uno indica una acción diferente a la que el servidor debe responder.

HEAD, por ejemplo, indica que únicamente queremos que se responda con los encabezados de la respuesta, y se ignore el cuerpo de datos. DELETE significa que queremos eliminar un recurso, etc

1. **¿Qué es un request y un response en una comunicación HTTP? ¿Qué son los headers?**

En una comunicación HTTP un request y un reponse son 2 tipos de mensajes. Casi todo lo que aparece en un navegador ha sido transmitido a través de HTTP mediante requests y responses entre navegador y servidor. El navegador abre una conexión a un servidor y realiza una solicitud. El servidor procesa la solicitud del cliente y devuelve una respuesta.

* **Un HTTP request se compone de**:
* Método: GET, POST, PUT, etc. Indica que tipo de request es.
* Path: la URL que se solicita, donde se encuentra el resource.
* Protocolo: contiene HTTP y su versión, actualmente 1.1.
* Headers. Son esquemas de key: value que contienen información sobre el HTTP request y el navegador. Aquí también se encuentran los datos de las cookies. La mayoría de los headers son opcionales.
* Body. Si se envía información al servidor a través de POST o PUT, ésta va en el body.
* **HTTP response esta compuesto por:**
* Protocolo. Contiene HTTP y su versión, actualmente 1.1.
* Status code. El código de respuesta, por ejemplo: 200 OK, que significa que el GET request ha sido satisfactorio y el servidor devolverá los contenidos del documento solicitado. Otro ejemplo es 404 Not Found, el servidor no ha encontrado el resource solicitado.
* Headers. Contienen información sobre el software del servidor, cuando se modificó por última vez el resource solicitado, el mime type, etc. De nuevo la mayoría son opcionales.
* Body. Si el servidor devuelve información que no sean headers ésta va en el body.

**b)** Los HTTP headers son la parte central de los HTTP requests y responses, y transmiten información acerca del navegador del cliente, de la página solicitada, del servidor, etc.

1. **¿Qué es un queryString? (En el contexto de una url)**

Es un término informático que se utiliza para hacer referencia a una interacción con una base de datos. Es la parte de una URL que contiene los datos que deben pasar a aplicaciones web como los programas CGI.

1. **¿Qué es el responseCode? ¿Qué significado tiene los posibles valores devueltos?**

Los códigos de estado de respuesta HTTP indican si se ha completado satisfactoriamente una solicitud HTTP específica. Las respuestas se agrupan en cinco clases según sus valores:

* Respuestas informativas (100–199)
* Respuestas satisfactorias (200–299),
* Redirecciones (300–399),
* Errores de los clientes (400–499),
* y errores de los servidores (500–599).

1. **¿Cómo se envía data en un Get y cómo en un POST?**

El método GET envía los datos usando la URL, en cambio el método POST los envía de forma que no podemos verlos (en un segundo plano u "ocultos" al usuario).

**7. ¿Qué verbo http utiliza el navegador cuando accedemos a una página?**

GET es el tipo más simple de método de solicitud HTTP; La que usan los navegadores cada vez que hace clic en un enlace o escribe una URL en la barra de direcciones. Indica al servidor que transmita los datos identificados por la URL al cliente. Los datos nunca deben ser modificados en el lado del servidor como resultado de una solicitud GET. En este sentido, una petición GET es de sólo lectura, pero por supuesto, una vez que el cliente recibe los datos, es libre de hacer cualquier operación con ella por su cuenta, por ejemplo, formatearla para su visualización.

**8. Explicar brevemente qué son las estructuras de datos JSON y XML dando ejemplo de estructuras posibles.**

* JavaScript Object Notation (JSON) es un formato basado en texto estándar para representar datos estructurados en la sintaxis de objetos de JavaScript. Es comúnmente utilizado para transmitir datos en aplicaciones web (por ejemplo: enviar algunos datos desde el servidor al cliente, así estos datos pueden ser mostrados en páginas web, o vice versa)

Es posible incluir los mismos tipos de datos básicos dentro de un JSON que en un objeto estándar de JavaScript - cadenas, números, arreglos, booleanos, y otros literales de objeto. Esto permite construir una jerarquía de datos, como por ejemplo:

{  
    "arrayColores":[{  
            "nombreColor":"rojo",  
            "valorHexadec":"#f00"  
        },  
        {  
            "nombreColor":"verde",  
            "valorHexadec":"#0f0"  
        },  
        {  
            "nombreColor":"azul",  
            "valorHexadec":"#00f"  
        },  
        {  
            "nombreColor":"cyan",  
            "valorHexadec":"#0ff"  
        },  
        {  
            "nombreColor":"magenta",  
            "valorHexadec":"#f0f"  
        },  
        {  
            "nombreColor":"amarillo",  
            "valorHexadec":"#ff0"  
        },  
        {  
            "nombreColor":"negro",  
            "valorHexadec":"#000"  
        }  
    ]  
}

* Es un lenguaje abierto, que sigue el estándar (W3 Consortium) derivado de SGML y optimizado para su uso en la WWW. Permite describir el sentido o la semántica de los datos. A diferencia de HTML, separa el contenido de la presentación. Es un Meta-Lenguaje, que permite la definición de lenguajes concretos de representación de documentos.

Los elementos de un documento XML deben seguir una estructura de “árbol” (estrictamente jerárquica), deben estar correctamente anidados, no se pueden superponer entre ellos.Sólo puede haber un elemento raiz, en el que estén contenidos todos los demás. Por ejemplo:

<libreria>

<libro>

<autores>

<autor>Elizabeth Castro</autor>

<autores>

<titulo>XML Guía de Aprendizaje</titulo>

<precio moneda="euros">30</precio>

<descriptores>

<descriptor>lenguajes<descriptor>

<descriptor>XML<descriptor>

<descriptores>

</libro>

<libro>

<autores>

<autor>Benoit Marchal</autor>

<autores

<titulo>XML con ejemplos</titulo>

<precio moneda="euros">45</precio>

<descriptores>

<descriptor>lenguajes<descriptor>

<descriptor>XML<descriptor>

<descriptores>

</libro>

</libreria>

1. **Explicar brevemente el estándar SOAP**

SOAP es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML.

Es un paradigma de mensajería de una dirección sin estado, que puede ser utilizado para formar protocolos más completos y complejos según las necesidades de las aplicaciones que lo implementan. Puede formar y construir la capa base de una "pila de protocolos de web service", ofreciendo un framework de mensajería básica en el cual los web services se pueden construir. Este protocolo está basado en XML y se conforma de tres partes:

* Sobre (envelope): el cual define qué hay en el mensaje y cómo procesarlo.
* Conjunto de reglas de codificación para expresar instancias de tipos de datos.
* La Convención para representar llamadas a procedimientos y respuestas.
* El protocolo SOAP tiene tres características principales:
* Extensibilidad (seguridad y WS-routing son extensiones aplicadas en el desarrollo).
* Neutralidad (bajo protocolo de transporte TCP puede ser utilizado sobre cualquier protocolo de aplicación como HTTP, SMTP o JMS).
* Independencia (permite cualquier modelo de programación).

1. **Explicar brevemente el estándar REST Full**

Es un servicio que funciona como un estándar para compartir información, en un sistema de doble vía: Consulta y Respuesta (Request => Response).

Al consultar una API se deben especificar parámetros de consulta para que el servicio sepa lo que queremos consultar, basado en una estructura previamente definida provista por el API por medio de una documentación.

La arquitectura REST al trabajar sobre el protocolo HTTP, los procedimientos o métodos de comunicación son los mismos que HTTP, siendo los principales: GET, POST, PUT, DELETE. Otros métodos que se utilizan en RESTful API son OPTIONS y HEAD. Este último se emplea para pasar parámetros de validación, autorización y tipo de procesamiento, entre otras funciones.

Otro componente de un RESTful API es el “HTTP Status Code”, que le informa al cliente o consumidor del API que debe hacer con la respuesta recibida. Estos son una referencia universal de resultado, es decir, al momento de diseñar un RESTful API toma en cuenta utilizar el “Status Code” de forma correcta.

Por ejemplo, el código 200 significa OK, que la consulta ha recibido respuesta desde el servidor o proveedor del API. Los códigos de estado más utilizado son:

* § 200 OK
* § 201 Created (Creado)
* § 304 Not Modified (No modificado)
* § 400 Bad Request (Error de consulta)
* § 401 Unauthorized (No autorizado)
* § 403 Forbidden (Prohibido)
* § 404 Not Found (No encontrado)
* § 422 (Unprocessable Entity (Entidad no procesable)
* § 500 Internal Server Error (Error Interno de Servidor)

**11.** **Qué son los headers en un request? ¿Para qué se utiliza el key Content-type en un header?**

Son esquemas de key: value que contienen información sobre el HTTP request y el navegador. Aquí también se encuentran los datos de las cookies. La mayoría de los headers son opcionales.

b) Content-Type es la propiedad de cabecera (header) usada para indicar el media type (en-US) del recurso, le dice al cliente que tipo de contenido será retornado.

**EJERCICIO 3**

Recomendamos previamente entender los conceptos de la sintaxis “json” antes de arrancar con los ejercicios.

Descargar el POSTMAN (aplicación para realizar request como cliente), adjuntando un screen de resolución para cada ítem:

1. Realizar un request GET a la URL:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente<https://vdfactory-234311.firebaseio.com/contacts.json>

1. Realizar un request POST a la URL anterior, y con body:

{

"firstname":"Pablo",

"email":"pablo.perez@virtualdreams.io"

}Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamenteTip: (Marcar la opción “raw” como body)

1. Realizar nuevamente un request GET a la URL:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente[https://vdfactory-234311.firebaseio.com](https://vdfactory-234311.firebaseio.com/contacts.json)[/contacts.json](https://procontacto-reclutamiento-default-rtdb.firebaseio.com/contacts.json)

**b) ¿Qué diferencias se observan entre las llamadas el punto 1 y 3?**

La diferencia entre el primero y el tercero es que ahora se ven reflejados los registros agregados mediante las request POST realizadas.

**EJERCICIO 4**

Solicitar usuario de Trailhead enviando un email a [ariel.tarsitano@virtualdreams.io](mailto:ariel.tarsitano@virtualdreams.io)

Cambiar el idioma de Trailhead a inglés.

Realizar los siguientes módulos de Trailhead:

* [Fundamento de la plataforma Salesforce](https://trailhead.salesforce.com/content/learn/modules/platform_dev_basics?trailmix_creator_id=strailhead&trailmix_slug=prepare-for-your-salesforce-platform-developer-i-credential)
* [Fundamentos de Apex y .NET](https://trailhead.salesforce.com/content/learn/modules/apex_basics_dotnet?trailmix_creator_id=strailhead&trailmix_slug=prepare-for-your-salesforce-platform-developer-i-credential)
* [Modelado de datos](https://trailhead.salesforce.com/content/learn/modules/data_modeling?trailmix_creator_id=strailhead&trailmix_slug=prepare-for-your-salesforce-platform-developer-i-credential)
* [Fundamentos y base de datos de Apex](https://trailhead.salesforce.com/content/learn/modules/apex_database?trailmix_creator_id=strailhead&trailmix_slug=prepare-for-your-salesforce-platform-developer-i-credential)
* [Desencadenadores de Apex](https://trailhead.salesforce.com/content/learn/modules/apex_triggers?trailmix_creator_id=strailhead&trailmix_slug=prepare-for-your-salesforce-platform-developer-i-credential)
* [Apex Integration Services](https://trailhead.salesforce.com/en/content/learn/modules/apex_integration_services)

Se recomienda usar el mismo Playground para todos módulos solicitados. Excepto que se solicite crear uno nuevo en el enunciado del Módulo.

Para revisar la resolución de los módulos, compartir la URL del perfil público de Trailhead en una url dentro del Readme.

* **URL**: **https://trailblazer.me/id/midelgado**

**EJERCICIO 5**

Explicar que son conceptualmente, qué datos almacenan en forma estándar y cómo se relacionan el resto (algunos no se relacionan entre sí) cada uno de los siguientes objetos de Salesforce:

1. Lead
2. Account
3. Contact
4. Opportunity
5. Product
6. PriceBook
7. Quote
8. Asset
9. Case
10. Article

Los campos enumerarlos a través de una lista de texto en el Readme y las relaciones a través de un diagrama UML simple realizado con Drawio.

## Ejemplo de diagrama

Drawio: <https://app.diagrams.net/>

Que una cuenta tiene muchos contactos se representa de la siguiente manera:



Completar el resto de las relaciones agregando el resto de los objetos enumerados.

Exportar el diagrama de Drawio, subirlo al repositorio y agregarlo dentro del readme con la etiqueta para [linkear imágenes](https://stackoverflow.com/questions/14494747/how-to-add-images-to-readme-md-on-github).

Se puede usar el schema builder de salesforce para entender las relaciones.

<https://trailhead.salesforce.com/content/learn/modules/data_modeling/schema_builder>

1. **Lead:** Representa un prospecto o un cliente potencial
2. **Account** : Representa una cuenta individual, que es una organización o persona involucrada con su negocio (como clientes, competidores y socios). Este objeto se puede utilizar para asociar un artículo con categorías de datos de un grupo de categorías de datos o para consultar las selecciones de categorías para un artículo.
3. **Contact:** Representa un contacto, que es una persona asociada a una cuenta.
4. **Opportunity:** Representa una oportunidad, que es una venta o un trato pendiente.
5. **Product:** Representa un producto que vende su organización.
6. **PriceBook:** Representa una lista de precios que contiene la lista de productos que vende su organización.
7. **Quote**: El objeto Cotización representa una cotización, que es un registro que muestra los precios propuestos para productos y servicios.
8. **Asset:** Representa un artículo de valor comercial, como un producto vendido por su empresa o un competidor, que un cliente ha comprado e instalado.
9. **Case:** Representa un caso, que es un problema o problema del cliente.
10. **Article:** Una selección de categoría de datos representa una categoría de datos que clasifica un artículo.

**Diagrama

Descripción generada automáticamente**

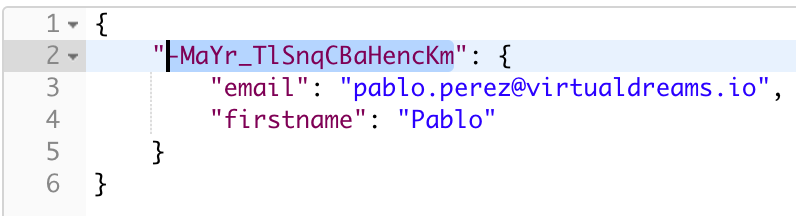
# EJERCICIO 6

Realizar las siguientes actividades sobre el Playground 1 del ejercicio 4:

1. Consultar tu ID haciendo un GET con POSTMAN a este WS:

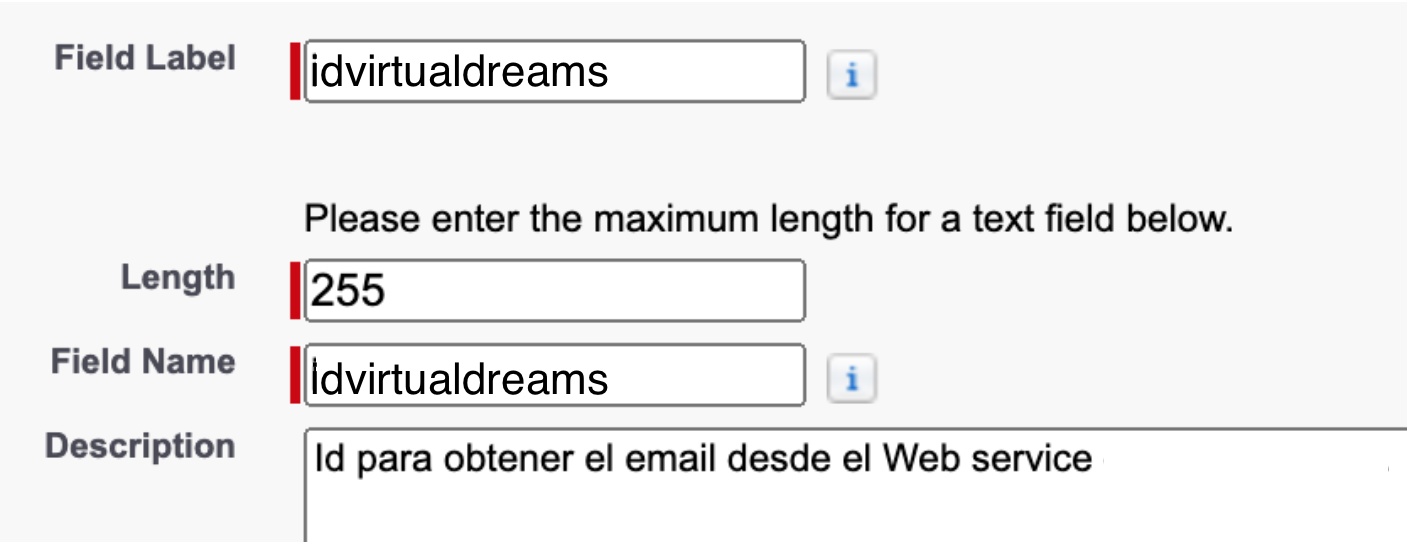
<https://vdfactory-234311.firebaseio.com/contacts.json>

Por ejemplo este es mi ID:



Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

1. Agregar un campo al objeto Contact llamado **idvirtualdreams** de tipo texto de 255 caracteres**. De la siguiente manera:**

No omitir poner el siguiente check cuando se crea el campo.****

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

1. Desarrollar un trigger para que cuando un usuario Modifica o Crea un contacto de Salesforce **completando el campo generado el punto B con el ID del punto A**, se invoque al Web Service con el **idvirtualdreams** obtenga los datos del nombre y el email de la respuesta y actualice el campo **email** del contacto. Usar Playground 1.

# **EJERCICIO 7**

Responder las siguientes preguntas brevemente sobre:

## Soluciones de Salesforce

1. **¿Qué es Salesforce?**

Salesforce es una solución de gestión de relaciones con clientes que une empresas y clientes. Es una plataforma CRM integrada que brinda a todos sus departamentos, incluidos marketing, ventas, comercio y servicios, una vista única y compartida de cada cliente.

1. **¿Qué es Sales Cloud?**

Sales Cloud es una aplicación de gestión de relaciones con clientes (CRM) de Salesforce basada en la nube. Incluye herramientas de gestión de contactos, automatización de las ventas, previsión de ventas y productividad.

1. **¿Qué es Service Cloud?**

El software de servicio al cliente Service Cloud proporciona gestión de casos, acceso al cliente en todos los canales, integración a los sistemas de datos preexistentes, aplicaciones de integración preincorporadas, tickets de asistencia, base de conocimientos, enrutamiento y escalamiento, y gestión de colas.

1. **¿Qué es Health Cloud?**

Health Cloud es una plataforma especialmente diseñada para la gestión

clínica de pacientes por medio de tecnologías on-cloud, la cual ofrece: una

comunicación más personalizado entre pacientes, miembros, proveedores y

prestadores de servicios de salud, y un mejor ajuste a los procesos, servicios y

datos médicos según el perfil de cuidado de cada paciente.

1. **¿Qué es Marketing Cloud?**

Salesforce Marketing Cloud es un proveedor de software y servicios de análisis y automatización de marketing digital. Desarrolla software de análisis y automatización de marketing para correo electrónico, móvil, social y marketing en línea. También ofrece servicios de consultoría e implementación.

## Funcionalidades de Salesforce

1. **¿Qué es un RecordType?**

Se utiliza para crear y mantener tipos de registro en su organización. Se pueden mostrar varios formatos de página y valores de lista de selección según los tipos de registro.

1. **¿Qué es un ReportType?**

Los ReportType le permiten crear un marco en el asistente para informes, desde el que los usuarios pueden crear y personalizar informes. Puede crear los tipos de informes personalizados a partir de las relaciones (principal-detalle y búsqueda) entre objetos de forma que puede:

* Seleccionar qué objetos mostrar a los usuarios que creen y personalicen informes
* Definir las relaciones entre objetos que se muestran a los usuarios que creen y personalicen informes
* Seleccionar qué campos de objetos podrán utilizarse como columnas en los informes

1. **¿Qué es un Page Layout?**

Un Page Layout nos permite personalizar el diseño y organización de detalles y editar páginas de registros en Salesforce. Los diseños de página se pueden usar para controlar la apariencia de campos, listas relacionadas y enlaces personalizados en la página de detalles y edición de objetos estándar y personalizados.

1. **¿Qué es un Compact Layout?**

Los Compact Layout controlan qué campos aparecen en el encabezado. Para cada objeto, puede asignar hasta 10 campos, incluyendo el campo Nombre, para mostrar en ese área.

1. **¿Qué es un Perfil?**

Los perfiles definen cómo acceden los usuarios a objetos y datos y qué pueden hacer en la aplicación.

1. **¿Qué es un Rol?**

Los roles controlan el nivel de visibilidad que un usuario tiene sobre los datos de su organización. Usuarios en cualquier función dada pueden ver, editar, e informar sobre todos los datos para funciones por debajo de ellos en la jerarquía de roles.

1. **¿Qué es un Validation Rule?**

Las reglas de validación verifican que los datos ingresados por usuarios en registros cumplen los estándares que especifica antes de poder guardarlos. Una regla de validación puede contener una fórmula o expresión que evalúa los datos en uno o más campos y ofrece un valor “Verdadero” o “Falso”.

1. **¿Qué diferencia hay entre una relación Master Detail y Lookup?**

Las relaciones de búsqueda se usan cuando los objetos están relacionados solo en ciertos casos. A veces, los contactos se asocian a cuentas específicas, aunque a veces es solo un contacto. Los objetos de las relaciones de búsqueda suelen funcionar como objetos independientes que tienen sus propias fichas en la interfaz de usuario.

En las relaciones del tipo principal-detalle, el objeto de detalle no funciona de manera independiente. En realidad, este objeto depende del objeto principal. Por lo tanto, si se elimina un registro del objeto principal, también se eliminarán todos los registros relacionados del objeto de detalle. A la hora de crear relaciones principal-detalle, siempre se crea el campo de relación en el objeto de detalle.

Por último, es posible crear un tercer tipo de relación denominada relación jerárquica. Las relaciones jerárquicas son un tipo de relación de búsqueda especial. La principal diferencia entre estos dos tipos es que las relaciones jerárquicas solo están disponibles en el objeto Usuario. Puede usarlas, por ejemplo, para crear cadenas de gestión entre los usuarios.

1. **¿Qué es un Sandbox?**

Sandbox es una copia de Producción, en donde podemos realizar todos los cambios que queramos a la configuración y hacer testeos, sin molestar a los usuarios.

1. **¿Que es un ChangeSet?**

Se utiliza para enviar personalizaciones de una organización de Salesforce a otra. Por ejemplo, puede crear y probar un nuevo objeto en una organización de espacio aislado y luego enviarlo a su organización de producción mediante un conjunto de cambios. Los conjuntos de cambios solo pueden contener modificaciones que puede realizar a través del menú Configuración. Por ejemplo, no puede utilizar un conjunto de cambios para cargar una lista de registros de contactos. Los conjuntos de cambios contienen información sobre la organización. No contienen datos, como registros.

1. **¿Para qué sirve el import Wizard de Salesforce?**

Permite importar hasta 50.000 registros a la vez. Es una interfaz sencilla para especificar parámetros de configuración, fuentes de datos y asignaciones de campos (en el archivo de importación se asignan los mismos nombres de campos que en Salesforce).

1. **¿Para qué sirve la funcionalidad Web to Lead?**

Web-to-Lead permite diseñar un formulario incluyendo las preguntas más adecuadas para cada tipo de negocio con el objetivo de insertarlo en un blog o una web corporativa, automatizando así la captación de leads y la integración de los datos de los usuarios en el CRM.

1. **¿Para qué sirve la funcionalidad Web to Case?**

Puede recopilar las solicitudes de servicio de atención al cliente directamente del sitio web de su empresa y generar automáticamente casos nuevos. La configuración de Web to case implica la habilitación de la función, la selección de ajustes y la adición del formulario Web to case a su sitio Web.

1. **¿Para qué sirve la funcionalidad Omnichannel?**

Permite a sus clientes conectar de forma transparente con su equipo de asistencia a través de múltiples canales. Al mismo tiempo, sus agentes de asistencia tienen acceso inmediato a una imagen holística de la persona a la que están a punto de ayudar.

1. **¿Para qué sirve la funcionalidad Chatter?**

Chatter es una aplicación de colaboración en tiempo real de Salesforce que permite a sus usuarios trabajar juntos, comunicarse y compartir información. Chatter conecta, compromete y motiva los usuarios para trabajar de forma eficiente en la organización independientemente de la función o ubicación. Chatter permite a los usuarios colaborar en oportunidades de ventas, casos de servicio, campañas y proyectos con aplicaciones integradas y acciones personalizadas.

## Conceptos generales

1. **¿Qué significa SaaS? ¿Salesforce es Saas?**

Significa  “Software as a Service” es un modelo de distribución de software donde el soporte lógico y los datos que maneja se alojan en servidores de una compañía de tecnologías de información y comunicación (TIC), a los que se accede vía Internet desde un cliente.

Salesforce es una compañía de PaaS (Plataforma como Servicio), un concepto que nace como resultado de la aplicación al desarrollo de Software del modelo SaaS (Software como Servicio). Este modelo abarca el ciclo completo para desarrollar e implantar aplicaciones desde Internet.

1. **¿Que significa que una solución sea Cloud?**

Significa que permite el acceso remoto a softwares, almacenamiento de archivos y procesamiento de datos por medio de Internet, siendo así, una alternativa a la ejecución en una computadora personal o servidor local. En el modelo de nube, no hay necesidad de instalar aplicaciones localmente en computadoras.

La computación en la nube ofrece a los individuos y a las empresas la capacidad de un pool de recursos de computación con buen mantenimiento, seguro, de fácil acceso y bajo demanda.

1. **¿Que significa que una solución sea On-Premise?**

significa decir que es mantenido en un servidor de la empresa, que puede estar en un salón específico o, hasta mismo, en un armario. En general, la empresa necesita comprar un servidor o una computadora que pueda ser utilizada como tal para que el software de CRM sea instalado y contar con un equipo de TI para realizar el mantenimiento.

1. **¿Que es un pipeline de ventas?**

Para controlar la evolución de las oportunidades, es necesario establecer diferentes etapas. No en vano, cuando la negociación avanza, aumenta la confianza en que la oportunidad se convertirá en una venta. Por otro lado, el número de oportunidades disminuye a medida que se llega a las etapas finales de la negociación, de ahí que se denomine al proceso pipeline de ventas o también embudo de ventas.

Salesforce es uno de los CRM que permite incluir las etapas como uno de los campos cuando se crean nuevas oportunidades. Generalmente, las etapas de ese proceso son: prospección (un cliente se muestra interesado en nuestro producto), desarrollo, negociación o revisión y cierre (ya sea porque se ha conseguido la venta o porque se ha perdido al cliente).

1. **¿Que es un funnel de ventas?**

El Funnel de Ventas es una representación de las etapas por las que un potencial cliente pasa, desde el primer contacto con la empresa hasta el cierre de la venta.

1. **¿Qué significa Customer Experience?**

El Customer Experience, también llamada experiencia del cliente o CX, es la experiencia que formará tu consumidor en función de sus interacciones con tu marca, que pueden ser positivas o negativas

1. **¿Qué significa omnicanalidad?**

La omnicanalidad es la interconexión y simultaneidad de todos los canales existentes en un mercado y la capacidad de una compañía para comunicar, vender y fidelizar a sus clientes en ellos.

1. **¿Qué significa que un negocio sea B2B?**

Cuando hablamos de B2B, nos referimos al entorno de negocio a negocio: el espacio comercial en el que las empresas intercambian productos, servicios o información con otras empresas.

Existen tres contextos principales en los que se producen transacciones comerciales B2B:

* Un negocio que adquiere materiales para la cadena de suministro (por ejemplo, una empresa de ropa que compra el tejido y los materiales para fabricar sus prendas).
* Una empresa que requiere los servicios de otra para facilitar su actividad empresarial (por ejemplo, una empresa de software que recurre a una empresa de contabilidad para presentar la declaración fiscal).
* Una empresa que revende los bienes y servicios que adquiere (por ejemplo, una agencia que aplica una marca blanca a una herramienta en línea reconfigurada y la vende a un cliente).

1. **¿Qué significa que un negocio sea B2C?**

B2C permite a los minoristas crear y coordinar experiencias y transacciones online de compradores entre canales y dispositivos digitales. Estas experiencias normalmente ocurren en Internet o en dispositivos móviles. Pero los compradores no se limitan a los sitios web y las plataformas móviles.

1. **¿Qué es un KPI?**

Es un sistema de medición que, expresado normalmente en un porcentaje, nos dice el grado de progreso o cumplimiento de un objetivo de la empresa. Los objetivos a medir con esto son de todo tipo, como: Satisfacción de los clientes.

1. **¿Qué es una API y en qué se diferencia de una Rest API?**

Una API es un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones. API significa interfaz de programación de aplicaciones.

Las API permiten que sus productos y servicios se comuniquen con otros, sin necesidad de saber cómo están implementados. Esto simplifica el desarrollo de las aplicaciones y permite ahorrar tiempo y dinero. Las API le otorgan flexibilidad; simplifican el diseño, la administración y el uso de las aplicaciones, y proporcionan oportunidades de innovación, lo cual es ideal al momento de diseñar herramientas y productos nuevos (o de gestionar los actuales).

En contraste, una API típica especifica cómo los componentes de software deben interactuar entre sí utilizando el protocolo web (HTTP) como intermediario.Las API's de REST suelen utilizar el protocolo de comunicación de la web (nuevamente, HTTP), pero no están limitadas de la misma manera que lo es un servicio web

1. **¿Qué es un Proceso Batch?**

Se conoce como sistema por lotes o modo batch, a la ejecución de un programa sin el control o supervisión directa del usuario que se denomina . Este tipo de programas se caracterizan porque su ejecución no precisa ningún tipo de interacción con el usuario.

Generalmente, este tipo de ejecución se utiliza en tareas repetitivas sobre grandes conjuntos de información, ya que sería tedioso y propenso a errores realizarlo manualmente

1. **¿Qué es Kanban?**

La vista Kanban ofrece un resumen visual de los registros de una vista de lista. Esta vista muestra una imagen de todo el trabajo y permite ordenar, resumir, filtrar y mover fácilmente las oportunidades en las distintas etapas.

1. **¿Qué es un ERP? ¿Salesforce es un ERP?**

El término ERP se refiere a Enterprise Resource Planning, que significa “sistema de planificación de recursos empresariales”. Estos programas se hacen cargo de distintas operaciones internas de una empresa, desde producción a distribución o incluso recursos humanos.

Muchos son los productos que podemos encontrar actualmente en el mercado de los sistemas de planificación de recursos empresariales. No obstante, algunos sobresalen debido a cualidades como su experiencia en el mercado, su reducido coste o su servicio de calidad. Un ejemplo de ello es Sales Force como ERP. Este producto creado a finales de los noventa, se especializó en la comercialización única y exclusivamente de un CRM destinado a mejorar la comunicación de los negocios con sus clientes. Consistía en un modelo denominado SaaS, también conocido como software como servicio. A día de hoy, ha evolucionado notablemente y Sales Force como ERP se presenta como una opción a tener en cuenta a la hora de implantar un sistema.