11086 - Programación en Ambiente Web - UNLu

Segundo Parcial 2020

Luciano Palmieri (148478)

luciano.palmieri.97@gmail.com

1. ¿Qué diferencia existe entre una petición HTTP generada por un agente de usuario de forma asincrónica respecto a una sincrónica? ¿Cómo puede distinguir una aplicación web entre ambas?

La diferencia básica se basa en que, al ser una petición asincrónica, luego de realizar la petición continúa la ejecución del sistema al mismo tiempo que se espera la respuesta del servidor, mientras que al ser síncrona al realizar la petición se bloquea el sistema hasta recibir la respuesta, brindando una mala experiencia de usuario a quien utilice la aplicación. A diferencia de las peticiones síncronas una petición asíncrona es transparente para el usuario ya que se realiza en segundo plano sin interrumpir la ejecución de la aplicación.

Una aplicación web puede distinguir entre ambos tipos de peticiones gracias a que en las peticiones asíncronas se realiza una instancia de la función XmlHttpRequest, luego utilizando el método open especificando en el tercer argumento el tipo de petición con el parámetro async: bool en true, como por ej: xhr.open("GET", "http://example.com", true). De este modo se añade el campo X-Requested-With: XMLHttpRequest dentro de la información de agente de usuario del header de HTTP enviado al servidor, pudiendo este determinar qué tipo de petición es.

2. ¿Qué diferencias existen entre el diseño responsivo y el universal? ¿En qué conceptos hay que hacer hincapié al momento de definir las media querys en cada caso?

Las diferencias que existen entre ambos diseños son que con el diseño universal se realiza para una o varias resoluciones específicas para un corto abanico de dispositivos que a la vez no son un mismo sistema, sino que al reconocer desde que dispositivo se realizaba la conexión se lo redirigía a otra vista con dimensiones distintas a la original, obligando al usuario a tenerse que adaptar a las resoluciones configuradas. Mientras que con el paso del tiempo y con la aparición de diversos dispositivos de diferentes tamaños empezó a tomar protagonismo la idea del diseño responsivo que no es realizar una gran cantidad de diseños para cada tipo de dispositivo, sino ser más flexible, hacer un único diseño que sepa adaptarse a las necesidades del dispositivo en cuestión, siendo transparente para el usuario esta transformación y haciendo que la usabilidad del sistema sea realmente cómoda.

Los conceptos a los que hay que hacer hincapié al momento de realizar diseño universal es al tipo de dispositivo que esta realizando la petición del recurso. Mientras que los conceptos a los que hay que hacer hincapié al momento de realizar un diseño responsivo son el tamaño de pantalla, el tipo de dispositivo y la orientación y de esta manera tener la capacidad de adaptarse a cada dispositivo, creando una solución única.

A nivel de implementación el diseño responsivo tiene tres conceptos claves.

- El primero de ellos es el uso de los Media Queries permitiendo aplicar estilos condicionalmente teniendo en cuenta parámetros de la pantalla.
- El segundo se trata del diseño web fluido, el cual se trata de layouts definidos en porcentajes que se ajustan a los anchos de la pantalla.
- el tercero se trata de los elementos fluidos dentro de estos layouts, como son las fuentes, las imágenes o elementos multimedia.

3. ¿Por qué decimos que no son directamente comparables REST y SOAP en el contexto de los Web Services?

Si bien existen diferencias notables entre REST y SOAP y ambos están basados sobre el principio SOA se dice que no son directamente comparables ya que ambos están orientados en conceptos distintos, SOAP está orientado a los servicios y se encarga de montar servicios web basado en el transporte de mensajes, mientras que REST está orientado a recursos y se concentra en interactuar con recursos con estado, en lugar de mensajes u operaciones. Además, REST es un estilo arquitectónico, mientras que SOAP es un formato de intercambio de mensaje.

El principio SOA ofrece:

- Contratos de servicio estandarizados.
- Poco acoplamiento.
- Abstracción de servicio.
- Reusabilidad del servicio.
- Servicio autónomo.
- Statelessness.
- Servicio de descubrimiento.
- Componibilidad.

4. Explique brevemente tres principios de desarrollo seguro y de un ejemplo para cada uno.

Algunos de los principios de desarrollo seguro son:

- Modularidad: dividir el programa en múltiples módulos y a su vez que esos módulos tengan funciones semindependientes que realizan tareas específicas, pero no demasiado complejas, haciendo de esta manera que pueda ser un módulo de programación mantenible. Un posible ej. para este principio puede ser un módulo de autenticación y de esta manera todo lo que tiene que ver con autenticación está completamente separado del resto del sistema.
- Aislación: consiste en aislar los distintos módulos que realizan llamadas a otros módulos de código y de todos modos debería poder funcionar correctamente incluso si otros ocasionan fallas. Este principio sirve para que una falla no interrumpa el uso del sistema en su totalidad y a la vez facilita a la identificación del fallo. Un posible ej. es si llega a haber un robo de información de algún modulo especifico como puede ser el de autenticación, solo seria robado la información que corresponde a este modulo y no todos los datos del sistema.
- Defensa en profundidad: consiste en implementar más de una medida de seguridad por si el sistema es atacado deba superar más de una medida de seguridad y no sea fácilmente vulnerado. Si bien esta metodología parezca demasiado para la mayor parte de los casos se obtiene una menor probabilidad de que el sistema sea vulnerado. Un posible ej. seria que solo se acepten tokens que sean recibidos a través de una red interna que se sepa que ya es segura y, además, los tokens enviados deberían estar protegidos por una clave que encima es temporal, por lo que si la llegan a robar no les sirva por demasiado tiempo.

5. ¿Cómo se relaciona el header HTTP Content-Security-Policy con la seguridad de un sistema web y por qué es fundamental su uso hoy en día? ¿Se puede implementar esto mismo de otra forma que no sea vía header HTTP (a nivel del server web)?

El Content Security Policy (CSP) es una capa de seguridad adicional que su principal objetivo es mitigar y reportar ataques, incluyendo Cross Site Scripting (XSS) y ataques de inyección de datos. Content Security Policy es el nombre de un encabezado de HTTP que los navegadores modernos usan para mejorar la seguridad de la página web. El encabezado Content-Security-Policy le permite restringir los recursos como JavaScript, CSS o cualquier elemento que se cargue en el navegador. Estos ataques son usados con diversos propósitos, desde robar información hasta desfiguración de sitios o distribución de malware.

Los ataques Cross Site Scripting consisten en inyectar scripts maliciosos del lado del cliente en un sitio web y usar el sitio web como método de propagación. Se aprovechan de la confianza del navegador en el contenido que recibe del servidor. El navegador del usuario ejecutará los scripts maliciosos porque confía en la fuente del contenido, aun cuando dicho contenido no provenga de donde se supone.

Content Security Policy hace posible que los administradores de servidores reduzcan o eliminen las posibilidades de ocurrencia de Cross Site Scripting mediante la especificación de dominios que el navegador considerará como fuentes válidas de scripts ejecutables. Un navegador compatible con Content Security Policy solo ejecutará scripts de los archivos fuentes especificados en esa lista blanca de dominios, ignorando completamente cualquier otro script.

Aunque se usa principalmente como un encabezado de respuesta HTTP, también puede aplicarlo a través de una etiqueta meta de HTML. Se debe ubicar el “Content-Security-Policy” dentro del http-equiv junto a su política dentro del atributo content que se ubicaran dentro de la etiqueta meta incluida en la etiqueta head. Un ejemplo puede ser el siguiente:

<meta http-equiv = "Content-Security-Policy" content = "default-src 'self'">

6. ¿Por qué es útil un buen análisis de riesgos a la hora de priorizar las mejoras de seguridad que podamos aplicar a nuestro sistema web?

Es necesario para la empresa hacer una adecuada evaluación de riesgos que le permita saber cuáles son las principales vulnerabilidades del sistema y cuáles son las amenazas que podrían explotar las mismas. En cuanto la empresa identifique los riesgos podrá establecer medidas para mitigar los mismos.

Hay muchas metodologías para la evaluación de riesgos, pero todas parten de la identificación de las posibles vulnerabilidades y las amenazas que acechan las vulnerabilidades, las cuales pueden estar relacionadas por recursos humanos, eventos naturales o fallas técnicas.

Ahora, a la hora de priorizar las mejoras de seguridad que podemos aplicar a nuestro sistema web es útil una buena evaluación de riesgos para poder saber cuales son los factores de mayor impacto para realizar mejoras sobre ellos para evitar esas vulnerabilidades, para las cuales es necesario hacer una valoración adecuada para determinar cuáles son los factores más críticos para la empresa. Esta valoración suele hacerse en términos de la posibilidad de ocurrencia del riesgo y del impacto que tenga la materialización del mismo, pero esta misma no debe ser solo una valoración calculada matemática o estadísticamente, sino que además debe ser valorada lógicamente para por ej priorizar entre dos posibles riesgos del mismo nivel.

7. Describa cómo generar una buena estrategia de SEO a partir del uso de herramientas semánticas.

En general podemos decir que los motores de búsqueda utilizan índices generados por web spiders, las cuales son un programa informático que inspecciona las páginas del World Wide Web de forma metódica y automatizada. Uno de los usos más frecuentes que se les da consiste en crear una copia de todas las páginas web visitadas para su procesado posterior por un motor de búsqueda que indexa las páginas proporcionando un sistema de búsquedas rápido.

SEO son un conjunto de acciones orientadas a mejorar el posicionamiento de un sitio web en la lista de resultados de los motores de búsqueda de internet. Trabaja aspectos técnicos como la optimización de la estructura y los metadatos de una web, pero también se aplica a nivel de contenidos, con el objetivo de volverlos más útiles y relevantes para los usuarios.

La manera de generar una buena estrategia de SEO a partir del uso de herramientas semánticas es con el uso de los siguientes factores:

- Contenido de la etiqueta body <body>

- Frecuencia y densidad de las palabras clave.
- Contenido de la etiqueta title <title>
- Utilización del atributo ALT en las etiquetas img <img>
- Contenido de la etiqueta de encabezado <h1>
- Utilización de textos de hiperlink.

8. ¿Cuáles son las ventajas y desventajas del modelo serverless en el cloud respecto al modelo tradicional basado en infraestructura (servers físicos / VMs).

Las ventajas del modelo serveless en el cloud son:

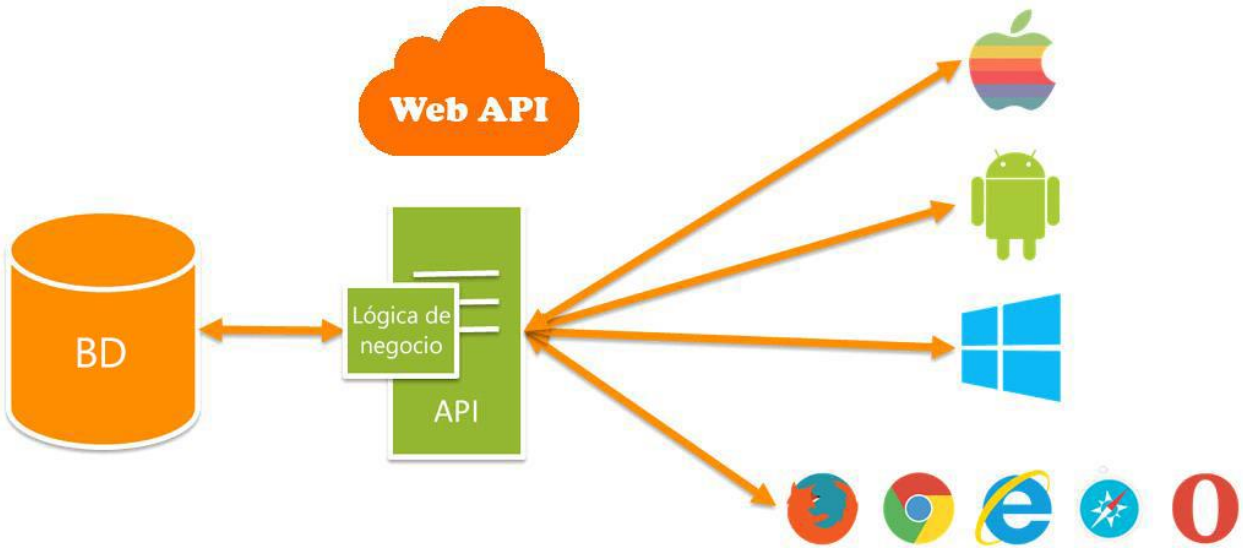
- Menos inversión inicial al poder utilizar software sin tener que realizar una inversión inicial en recursos.
- Actualizaciones automáticas de seguridad y además la no necesidad de personal dedicado a realizar las mismas.
- Poder centrarse en la construcción y mantención del sistema al poder desligarse en gran medida de la seguridad.
- Fácil escalado vertical tanto como para aumentar como para disminuir la capacidad con la asignación dinámica de recursos.
- Reducción de costos no solo por pagar solo por el tiempo informático que se consume gracias a la asignación dinámica de recursos, sino también obteniendo un ahorro de los costos de mantenimiento de los mismos.

Mientras que las desventajas son:

- El nivel de confianza en la seguridad de los datos ya que los datos propietarios de la empresa los posee un tercero en el cual hay que confiar en la seguridad que implementen.
- Que se desee tener mayor control sobre la infraestructura conllevando a ser esto una desventaja para empresas con recursos humanos suficientes para poder realizar parte de las que son clasificadas como ventajas.
- La dependencia hacia un determinado proveedor.

9. Imagine tiene que implementar un sistema de firma digital: dado un pdf de entrada debe devolverlo firmado digitalmente. Para ello, y dado que debe integrarse a sistemas web existentes, debe diseñar una arquitectura que facilite dicha integración. Comente sobre los componentes de la misma y qué cuestiones contempla, dificultades, etc.

Para la implementación del sistema de firma digital sobre PDF y poder ser integrada a diversos sistemas web se ideó la creación de una API, la cual se encargará de realizar la firma de la documentación como lo es un archivo con formato PDF, autenticado a través de la utilización de un token y una contraseña. La arquitectura ideada para la implementación de la misma es la siguiente:



Los principales componentes que se idearon para esta implementación son dos módulos principales los cuales uno se encarga de realizar la autenticación en sí del token y contraseña ingresados por el usuario y un segundo módulo, el cual se encargará de la funcionalidad principal del sistema de la creación de una firma digital a partir de la respuesta emitida por el módulo de autenticación, la inserción de la misma sobre la documentación y para finalizar el circuito el envío de la misma al usuario que la solicitó.

A grandes rasgos la mecánica de la API es la siguiente:

1. El usuario ingresa la documentación a la cual desea insertarle una firma digital.
2. El sistema le solicita al cliente que inserte en el puerto USB el token físico previamente obtenido por un organismo certificado.
3. Luego de reconocer el token físico el sistema le solicita al cliente una contraseña para constatar que el poseedor del token sea el dueño del mismo.
4. El sistema realiza la mecánica descripta por los módulos mencionados.
5. El usuario recibe una nueva versión de la documentación con la firma solicitada ya insertada en la misma.

Algunas de las dificultades que se pueden ver para el funcionamiento de la API son:

- La previa obtención del token y su respectiva contraseña, la cual implica un cierto tramite fuera de este sistema, siendo así un impedimento para el usuario del sistema.
- La validación en sí de los métodos de autenticación ya que deben ser constatados con los organismos de emisión.
- La obtención y mantenimiento de las firmas de los certificados realizados por las autoridades de certificación utilizados para las firmas de las documentaciones emitidas.

10. Suponga que está desarrollando una API que puede ser consumida utilizando diferentes formatos de intercambio de datos ¿De qué forma puede determinar el backend el formato a utilizar para atender un cliente determinado? ¿Cómo debería comportarse el mismo en caso de no conocer el formato solicitado?

Desde el lado del cliente a través del protocolo HTTP permite que se especifique en qué formato se quiere recibir el recurso solicitado, pudiendo indicar varios en orden de preferencia, para ello se utiliza el header Accept pudiéndose completar de la siguiente manera:

GET: http://tech.tribalyte.eu/category/apps?pageSize=10&page=1

Accept: application/pdf, application/json

Desde la API se verificará que sea un formato de archivo valido y devolverá el recurso en el primer formato disponible y, de no poder mostrar el recurso en ninguno de los formatos indicados por el cliente o simplemente el cliente no especificó un formato, el servidor devolverá el código de estado HTTP 406 Not Acceptable, el cual representa que el servidor no es capaz de devolver los recursos en ninguno de los formatos aceptados por el cliente.

En caso de una respuesta satisfactoria enviada desde el servidor, este se devolverá el recurso con el header Content-Type completado, para que el usuario sepa qué formato se devuelve para poder manipularlo.