Introducción

Programación Web

La Web

- **Plataforma** en la cual los **recursos** están distribuidos en la **red** y están siendo extendidos en todo momento con posibilidades ilimitadas.
- Se hizo popular con aplicaciones como clientes de correo, buscadores, chats, redes sociales, blogs, aplicaciones de comercio electrónico, entre otras.
- Además de estas aplicaciones de propósito general, existe una gran diversidad de soluciones que adoptan el ambiente web, como son: Administradores de contenido (CMS), suites para trabajo colaborativo, almacenamiento en la nube, procesamiento en la nube con suites de aplicaciones como CRM, ERP, Office, entre otras.

La Web

- Se reinventa día a día buscando alternativas que le permitan ofrecer todas sus ventajas, pero con la posibilidad de ofrecer controles visuales más amigables para el trabajo del usuario.

Las aplicaciones Web

- Desde el punto de vista de la arquitectura se distinguen dos lados; uno es el cliente, donde se encuentra el usuario final utilizando la aplicación por medio de un navegador (Chrome, Firefox, Edge, Safari, Opera, etc.).
- A través de este cliente web, el usuario interactúa con la aplicación localizada al otro lado, en el servidor, que es donde residen realmente los datos, reglas y lógica de la aplicación.
- Comúnmente utilizan lo que se conoce como clientes ligeros los cuales no ejecutan demasiadas labores de procesamiento para la ejecución de la aplicación misma, aunque actualmente se está asignando parte del procesamiento de los datos cliente.

Relevancia de las aplicaciones Web

- Ofrecen la oportunidad de acceso a diversas aplicaciones a través de equipos de cómputo personales y otros dispositivos móviles.
- Permiten acceder desde cualquier sitio donde se tenga acceso a Internet o acceso a la red en la que se encuentra disponible la aplicación.

Tarea

Integrarse en equipos y definir los beneficios de:

Aplicaciones de escritorio

VS

Aplicaciones web

VS

Aplicaciones móviles

Inconvenientes de las aplicaciones de escritorio

- Falta de portabilidad de la aplicación a diferentes sistemas operativos.
- Dificultades a la hora de realizar actualizaciones o correcciones al programa ya que las instalaciones están diseminadas.

A pesar de estos inconvenientes, no se puede despreciar el enorme beneficio del aprovechamiento de los recursos de procesamiento disponibles en cada equipo.
Además, se debe considerar que hay herramientas que facilitan la portabilidad, además de herramientas que apoyan para la distribución de actualizaciones.

Beneficios de las aplicaciones Web

No requiere instalar software especial (en los clientes)

- En esencia, solo necesitamos disponer de un navegador de páginas web (Edge, Firefox, Opera, Chrome, safari, etc.), los cuales suelen venir con el propio sistema operativo.

Bajo costo en actualizar los equipos con una nueva versión

- Los navegadores web visualizan las páginas web que son proporcionadas por el servidor web dinámicamente. Si la actualización se realiza en el servidor, automáticamente la ven todos los usuarios.

Beneficios de las aplicaciones Web

Acceso a la última y mejor versión

- Como consecuencia del punto anterior, se evita que pueda existir algún equipo que ejecute una versión diferente y desactualizada. Si existen ordenadores con distintas versiones del programa se pueden originar problemas de consistencia en la información o pérdida de funcionalidad.

Seguridad (Safety)

- Debido a que la información y la lógica de negocios están centralizadas en el servidor, en caso de robo o incendio, la empresa no ha perdido información y puede desplegar rápidamente un nuevo puesto de trabajo con cualquier dispositivo.

Beneficios de las aplicaciones Web

Movilidad

- Si el software está ubicado en un servidor web en Internet, cualquier usuario con un equipo personal o dispositivo móvil y una conexión a Internet podría acceder a la aplicación.

Reducción de costes en los puestos cliente (mayor longevidad)

- Debido a que las páginas se ofrecen desde el servidor web (donde se suelen ejecutar la mayoría de los procesos y la lógica de negocio), el equipo cliente queda relegado a mostrar los resultados y formularios, para lo cual no es necesario un hardware potente en los puestos de trabajo, lo que se traduce en reducción de costes y una mayor longevidad en el uso de los mismos (no hay que cambiar el hardware de los puestos porque ahora se requieran operaciones más complejas).

Reducción de costos en el desarrollo de una aplicación única

- Con una aplicación web no es requerido desarrollar aplicaciones nativas para dispositivos móviles o específicas para escritorio.

Dominio

- Un dominio es un **nombre** o **dirección de internet** que puede ser alfanumérica y se vincula a una dirección física (IP) que generalmente es una computadora, por ejemplo **google.com**.

- Se utilizan para representar las direcciones de los sitios web y evitar la complejidad al usuario de recordar la dirección IP de cada sitio al que requiere acceder.

- Su adquisición generalmente genera un costo.

Elementos que conforman un dominio

El nombre

- Generalmente asociado la razón social, marca o nombre de la página. Por ejemplo en google.com.mx, el nombre del dominio es “google”

La extensión

- Identifica el tipo de dominio que es. En el ejemplo anterior la extensión es “.com.mx”

En general hay dos tipos de dominios de Internet

Internacionales

- Este tipo de dominios son los que no delimitan a una página como perteneciente a una región en particular. En los últimos años en un intento por vender más dominios han surgido nuevas terminaciones para dominios Internacionales, sin embargo los principales son: .com, .net y .org

Territoriales

- Los dominios regionales fueron otorgados para cada país y su terminación es la abreviación del país. Por ejemplo: .com.mx (México), .com.ar (Argentina), .com.br (Brasil), etc.

Ejemplos de dominios

- Sitios de Organizaciones de Negocios

.biz

- Sitios Comerciales

.com

- Sitios Educativos

.edu

- Sitios de Organismos Gubernamentales

.gov o gob

- Sitios de Apertura Libre

.info

- Sitios de Sistemas y Redes

.net

- Sitios de Organizaciones

.org

DNS (Servidor de nombres de dominio)

- Son servidores en internet dedicados a mantener una base de datos de mapeo de los nombres de dominio registrados y su correspondiente IP.
- De tal forma que se facilite la traducción del nombre de dominio a una IP que se pueda ubicar en la red.

Ejercicio

- Analizar el proceso que se realiza desde que el cliente coloca la URL de una aplicación en el explorador web, hasta que el cliente puede obtener una respuesta del servidor

¿Cómo funciona?

El usuario ingresa la dirección www.itsur.edu.mx

Se debe transformar la dirección de internet en una dirección IP por medio de los DNS.

La dirección IP permite contactar al servidor web que aloja el sitio web y una vez obtenida, se envía una solicitud al servidor a fin de lograr acceder al recurso.

Una vez con el texto HTML, el navegador realiza un análisis de este y si lo requiere realiza otras peticiones de gráficos y al resto de los archivos que formen parte de la página.

Qué recurso se solicita? Si no solicita ninguno en específico se manda el default, por ejemplo: index.php

4

Index

www.itsur.edu.mx
146.78.23.5

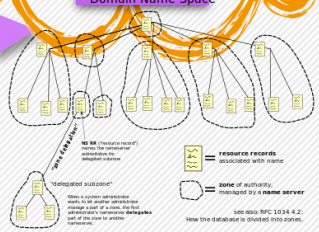
Mostrar 146.78.23.5

3

2

www.itsur.edu.mx
Está en la IP 146.78.23.5

DNS
Domain Name Space



¿Dónde está www.itsur.edu.mx?

1

www.itsur.edu.mx

Cliente a través del navegador

Tecnologías para el desarrollo de aplicaciones web

¿Qué necesito para trabajar con aplicaciones web?

Tecnologías de desarrollo

- IDE o editores de texto (Block de notas o uno más fortalecido)
- Gestor de base de datos (páginas dinámicas)
- Frameworks (Visual, Codificación, Desarrollo-Visual-Código-Arquitectura)
- Seleccionar lenguaje de programación (leguaje del lado del servidor)
- Si se utiliza lenguaje del lado del servidor es necesario un servidor web.

Requisitos

- Navegador
- Conexión local o a internet (dependiendo el enfoque)
- Hospedaje-Publicación
  - Hardware
    - Local (compartido en la intranet)
      - IP
    - Internet
      - IP pública (80 o corriendo en otro puerto)
      - Dominio (Opcional)
      - Host (Espacio, Procesadores, Ram)
      - Servidor FTP (Opcional)
  - Software
    - Servidor web (Apache, Glass fish, Internet Information Services [IIS])

Planificación de aplicaciones web

Planificación de aplicaciones

El trabajo en las diversas fases para una aplicación web se verá ampliado, puesto que debemos realizar otras actividades que comúnmente no consideramos con aplicaciones de escritorio.

Este impacto se debe principalmente a que la audiencia de las aplicaciones web es más amplia por el alcance que tienen a través de internet, además de que las plataformas de consumo no son solamente equipos de escritorio, sino que hay una gran gama de dispositivos en las que se pueden consumir (Computadoras personales, Dispositivos móviles, SmartTVs, etc).

Análisis

- Identificar y analizar requerimientos (funcionalidades y contenidos necesarios).
- Identificar mercado objetivo
  - Identificar los perfiles de audiencia o mercado: principal y secundaria
  - Identificar las capacidades y restricciones (habilidades) de la audiencia
  - Identificar necesidades típicas, requerimientos del navegador (al que irá dirigido, plataforma de consumo, etc.)

Diseño (1)

Definir la estrategia de comunicación: colores y definición del estilo visual

Prototipado de requerimientos funcionales

- Plantear wireframes o prototipos incorporando los diversos tamaños de pantalla a los que irá dirigido.
- Definir las interacciones y la disponibilidad de las diferentes funcionalidades de la aplicación de acuerdo a los dispositivos objetivo

Definir los requerimientos tecnológicos que la aplicación requiere tanto en recursos de servidor, como disponibilidad de dominio y hosting, así como las plataformas y lenguajes de desarrollo.

Diseño (2)

Identificar la integración con otras aplicaciones (tanto propias como de terceros), es muy común que las aplicaciones a desarrollar requieran establecer mecanismos de comunicación con aplicaciones existentes o futuras.

Mapa del sitio (navegación)

- Establecimiento de los vínculos, menús y diversos caminos que se establecerán para navegar por las diferentes páginas

Establecer los mecanismos que provean a la aplicación de la seguridad requerida.

Diseño (3)

Recopilación y Creación de Contenidos

- Identificar el contenido existente: como textos, imágenes, videos, fotografías, logotipos, entre otros.
- Identificar y crear o editar contenido textos, traducciones, videos, imágenes editadas, creación de la marca, entre otros.

Optimización de los recursos a publicar, es decir, compresión de imágenes, videos u otros recursos, común mente se generar contenidos de baja resolución (que no afecten la calidad visual del diseño).

Codificación

- Desarrollo de la aplicación respetando la arquitectura establecida y usando diversos lenguajes para el desarrollo

Lenguaje de marcas	Lenguaje de presentación	Lenguaje del lado del cliente	Lenguaje del lado del servidor
• HyperText Markup Language – HTML • Permite definir la estructura de los elementos en una página.	• Cascading Style Sheet – CSS • Permite definir el aspecto visual de los elementos definidos con HTML	• Javascript – JS • Permite dar interacción y funcionalidad a la aplicación sin requerir del todo interacción constante con el servidor.	• PHP, JSP, ASP.NET, Python u otro • Permite trabajar del lado del servidor con operaciones que por seguridad o por conveniencia deban ser ejecutadas por el servidor, tales como el tratamiento y almacenamiento de datos.

Pruebas

- Prueba de los contenidos y revisión de los mismos: comprobación de enlaces, revisión imágenes y diversos recursos.
- Pruebas funcionales: pruebas sobre los requisitos funcionales de la aplicación y sobre elementos interactivos
- Pruebas de navegación
- Pruebas de estrés(consiste en probar los límites que un sistema puede soportar) y de carga (la cantidad de peticiones que un sistema puede soportar). Esto se realiza siempre que la audiencia sea muy amplia.
- Pruebas en los distintos navegadores.
- Pruebas en distintos dispositivos.

Implantación

- Gestionar la compra del hosting (espacio y recursos requeridos).
- Gestionar la compra del dominio (nombre).
- Instalación y configuración del Servidor y del Hosting.
- Publicación de la aplicación web y el contenido requerido.

Compilador e intérprete

- Para que el procesador puedan comprender las instrucciones que contiene un programa desarrollado previamente, el código fuente escrito en los lenguajes de programación actuales debe convertirse a un formato legible por la máquina.
- Dependiendo del lenguaje de programación, este proceso puede quedar a cargo de un compilador o un intérprete.

Intérprete

- Procesa el código fuente de un proyecto durante su tiempo de ejecución, es decir, mientras el software se está ejecutando.
- Un intérprete siempre procesa el código línea por línea, de modo que lee, analiza y prepara cada secuencia de forma consecutiva para el procesador.
- En cuanto una línea de código fuente se ha traducido a los correspondientes comandos legibles por la máquina, esta se envía directamente al procesador.
- Se interrumpe prematuramente la ejecución si se produce un fallo durante el procesamiento, la línea de código problemática se detecta inmediatamente después de ocurrir el fallo.

Compilador

- Traduce todo el código fuente de un proyecto a código máquina antes de ejecutarlo. Solo entonces el procesador ejecuta el software.
- El procesador cuenta con todos los componentes necesarios para ejecutar el software, procesar las entradas y generar los resultados.
- No obstante, en muchos casos, durante el proceso de compilación tiene lugar un paso intermedio fundamental: antes de generar la traducción final en código máquina, la mayoría de los compiladores suelen convertir el código fuente en un código intermedio (también llamado código objeto) que, a menudo, es compatible con diversas plataformas y que, además, también puede ser utilizado por un intérprete.

Interprete vs Compilador

	Intérprete	Compilador
Momento en que se traduce el código fuente	Durante el tiempo de ejecución del software	Antes de ejecutar el software
Procedimiento de traducción	Línea por línea	Siempre todo el código
Presentación de errores de código	Después de cada línea	En conjunto, después de toda la compilación
Velocidad de traducción	Alta	Baja
Eficiencia de traducción	Baja	Alta
Coste de desarrollo	Bajo	Alto
Lenguajes típicos	PHP, Perl, Python, Ruby	C, C++

Solución híbrida

- Algunos plataformas tratan de aprovechar las ventajas de ambos enfoques haciendo uso de JITCompiler (Just In Time Compiler).
- El cual trata de identificar el código que más se repite para mantenerlo en caché.