Tecnológico de Monterrey

Campus Querétaro

Cuaderno de apuntes para evaluación de competencias

Marco Iván Flores Villanueva
A01276586

Materia: Construcción de software y toma de decisiones
Grupo: 501

HTML y arquitectura

La arquitectura que se ha usado durante el curso es la de *Cliente-servidor*. El cliente consiste en el mismo usuario y su computadora, mientras que el servidor es aquella computadora capaz de comprender lo que el usuario esté pidiendo.

La infraestructura va a estar limitada por sus capacidades, lo más importante será cuanto disco duro tiene y cuanta capacidad para procesar tendrá. Si a una computadora se le acaba la RAM, se tiene que reiniciar. Si el disco duro se acaba, la computadora sigue funcionando pero ya no puede guardar más información. La estructura de datos que nos incumbe para este curso es la del desarrollo web, la cual identificamos como un *árbol*.

Frontend

La parte de cliente se conforma por el frontend. En HTML (HyperText Markup Language), las etiquetas se abren y cierran:

- <html> ... </html>
- <p src=> ... </p>

CSS (Cascading Style Sheets) sirve para:

- Colores
- Fuente
- Renderización
- Imágenes
- Animaciones
- Formato (márgenes)

JavaScript programa la funcionalidad de los elementos, utiliza el paradigma de la programación orientada a eventos. Se eligió JS porque los primeros navegadores vieron que el rendimiento era más óptimo para modificar el *árbol*.

Versionamiento

Un repositorio es un sistema de control de versiones. Lo que se intenta hacer es tener un lugar donde esté todo el código del proyecto y cada miembro del proyecto pueda estar añadiendo cambios al mismo, detectando cuando hay un nuevo cambio y el proyecto crezca de manera organizada. Es esencial saber eliminar y crear branches, pues de esta manera, cada miembro del proyecto puede contribuir trabajando por separado en una *feature* independiente

para después fusionarla con el resto del proyecto sin tener que poner en riesgo la funcionalidad del mismo. Algunas de las herramientas de versionamiento más conocidas son:

- Git
- Hg
- SVN
- HelixCore
- CVS
- Bazaar, etc.

Tenemos el repositorio local y remoto. Vamos a tener un repositorio en la nube donde estarán los cambios globales, y cada computadora del equipo hará cambios locales en el repositorio local. Todo lo que subamos debe ser una versión estable para no generar conflictos.

Áreas de versionamiento

En el repositorio local, el *working tree* es cuando abrimos la carpeta del proyecto y observamos los archivos (por ejemplo, un proyecto de página web). El repositorio local sigue siendo el *working tree*, pero el repositorio local es aquello que ya está guardado dentro y sincronizado con el remoto. Cuando hago un cambio en algún archivo del *working tree*, tengo que poner un comando para pasarlo al *staging area*. Este paso entre el working tree y el local repo será el *staging area*.

```
1   index.html --add/restore-> staging area <-commit/reset-> local repo
```

Si alguien quisiera empujar desde su local repo al remote repo, necesitaría haber trabajado en la última versión del documento, o no se podrá hacer. Es importante ponerse de acuerdo y comunicarse con los demás miembros del equipo para que no ocurran problemas, y también es importante hacer commit cada que se pueda.

En resumen, el working tree es un pedazo del proyecto en cualquier momento (usualmente es el momento actual). Cuando agrego código, modifico el working tree. El staging area es donde se colocan los cambios del working tree antes de hacerlos permanentes, y el repositorio es la colección de commits (cambios permanentes realizados a través de la historia del proyecto).

Selectores de CSS

Existen varios tipos de selectores en CSS para lograr darle estilo a los diferentes componentes de una aplicación web:

1. Elemento
2. ID
3. Clase
4. Atributo
5. Combinados
6. Pseudo-clases

Selector por elemento: a todos los párrafos p se les aplicará un padding de 20px

```
p {
padding: 20px
}
```

Selector por ID: a cada elemento del HTML que quiero que tenga la asignación de destacado, debo marcarlo así:

```
#destacado {
margin: 10px
}
```

Y en el HTML:

```
<img id="destacado">
```

Selector por clase: los id se utilizan para los estilos y demás. Las clases también funcionan para JS. Para usar las clases en CSS se usan así:

```
.clase{
font-weight: bold;
}
```

Y en el HTML:

```
<img class="clase">
```

Selector por atributo: quiero que todos los elementos del HTML que tengan tal atributo se modifique su comportamiento.

```
input [type="text"] {
background-color: red;
}
```

Y en el HTML:

```
<input type="text">
```

Selectores combinados: este ejemplo afecta a los párrafos contenidos en los div que tengan la clase container.

```
div.container p {

}
```

Y en el HTML

```
<div class="container">
<p>...</p>
</div>
```

Selectores por pseudo-clases: la pseudo-clase hover corresponde a cuando el cursor pasa por encima de un elemento.

Propiedades básicas de CSS

Colores

```
p {

/*Por nombre*/
color: red;

/*RGB hexadecimal*/
color: #ff0000;

/*RGB decimal*/
color: rgb(255,0,0);

/*RGBA (con transparencia)*/
color: rgb(255,0,0, 0.005);
}
```

Tipografías

```css
p {

/*Familia de fuentes*/
font-family: Arial, Helvetica, sans-serif;

/*Tamaño de fuente*/
font-size: 16px;

/*Peso de fuente*/
font-weight: bold; /*o normal, 100,900*/

/*Estilo de fuente*/
font-style: italic; /*o normal, oblique*/

/*Decoración de texto*/
text-decoration: underline;

/*Transformación de texto*/
text-transform: uppercase;

/*Alineación de texto*/
text-align: center;

/*Altura de línea*/
line-height: 1.5;

/*Espacio entre letras y palabras*/
letter-spacing: 1px;
word-spacing: 2px;
}
```

Clase 26 marzo: sobre datos e información

El dato es una representación simbólica de un atributo o característica de una entidad. No tiene valor semántico en sí mismo, pero convenientemente tratado (procesado) se puede utilizar en la realización de cálculos o toma de decisiones.

datos -> procesamiento -> información

La información es un conjunto organizado de datos que constituyen un mensaje sobre un determinado ente o fenómeno. Por otra parte, en la interacción de la humano-computadora, la entrada es la información producida por el usuario. La entrada viene también de dispositivos de redes, etc.

El DSS (Decision Support Systems) es un sistema de soporte a las decisiones es un conjunto de personas, procedimientos, hardware y software, bases de datos y dispositivos que apoyan a tomar decisiones a problemas específicos para problemas semi-estructurados o no estructurados. Puede realizar análisis "Qué pasa si?".

Frameworks

Los *Frameworks* son bibliotecas prediseñadas de estilos y componentes que facilitan el desarrollo web. Contienen varios componentes, entre ellos:

- Sistema de grid responsivo
- Componentes UI prediseñados
- Utilidades para espaciado, tipografía y colores
- Normalización entre navegadores

Algunas ventajas de los frameworks son las siguientes:

- Permite un desarrollo más rápido
- Consistencia
- Responsividad
- Comunidad y documentación
- Accesibilidad
- Compatibilidad entre navegadores

Requisitos funcionales para el sistema diseñado (VentaDB)

Automatización de importación y gestión de comentarios

- Eliminar la dependencia de un programa externo para hacer "scraping" de los comentarios mediante el uso de una librería como selenium library (en Python).
- Importación de scraped data a bases de datos.

Visualización y edición de datos

- Visualización: Visualizar datos (por medio de usuarios con permiso)en base de datos para posteriormente realizar modificaciones en ellos.
- Edición interna: Modificar datos directamente en la página, validando los datos con JavaScript desde el lado del cliente y después desde el servidor.
- Edición externa: Exportar un archivo editable, modificarlo y resubirlo (posible automatización con Python para realizar cambios en bulk, así como para filtrar y ordenar datos una vez descargados).

- Datos editables de productos:
  Cantidad.
  Descripción.
- Datos NO editables:
  Número de producto.
  Nombre del producto.
  Costo por pieza.

Gestión de cuentas y clientes

- Generación de usuarios con permisos diferentes para administradores y socios de la empresa.
- Registro de clientes en una base de datos con:
  Nombre del perfil de Facebook (es posible que cambie).
  Enlace del perfil de Facebook (único y no editable)
  ID de cliente.
- Script para verificar si un cliente ya se encuentra en la base de datos para evitar la duplicación del mismo.
- Registro de vendedores con:
  Nombre.
  Contacto (teléfono).
  Correo.
  ID de vendedor.

Base de datos para ventas

- Creación de bases de datos para almacenar:
  Ventas asignadas:
  ID de cliente
  Enlace del perfil de Facebook
  Número de producto
  ID de vendedor
  ID de seguimiento

Registro de actividades y cambios

- Límite de tiempo para sesiones individuales inactivas de 4 min.
- Guardar historial de ediciones para rastrear quién editó qué.

Generación de reportes

- Exportar reporte en PDF cada N número de minutos, horas o días.

- Sólo usuarios con permisos pueden pedir la descarga de los documentos.

Bases de datos

Una base de datos es una colección de datos relacionados/no relacionados que representa aspectos del mundo real. Es diseñada, construida y poblada de datos con un significado específico. Existen bases de datos tradicionales y no tradicionales, y son usadas por una gran cantidad de empresas y negocios con distintas áreas de acción. En nuestro caso, usaremos bases de datos (específicamente MariaDB) para realizar nuestro proyecto de sistema para el socio formador.

Algunos conceptos a recordar son los siguientes:

Usuario: persona que por medio de una aplicación usa la base de datos para actualizar, borrar, insertar o consultar información.

Aplicación de base de datos: programa computacional que sirve como intermediario entre manejador de BD y usuario.

Manejador de base de datos (DBMS): programa computacional para crear, procesar y administrar la base de datos.

Introducción al backend

En toda aplicación web existen dos lados: el *frontend* y el *backend*. De la parte del *frontend* (cliente), tenemos al HTML, CSS y JS, como ya se ha explicado previamente. Pero del lado del *backend*, tenemos a NodeJS. Es de gran relevancia nunca conectar el cliente directamente a la base de datos, pues esto conlleva un riesgo al brindar a cualquier usuario, malicioso o no, la oportunidad de acceder a datos que podrían ser privados. Se espera que nuestro proyecto esté en un servidor de arquitectura monolítica.

Patrón de arquitectura MVC

Un patrón de arquitectura puede ser descrito como la forma en la que se acomodan los directorios de un proyecto. Como buena práctica de desarrollo, se busca simplificar el proyecto lo más posible y mantenerlo ordenado usando uno de estos patrones. Un patrón común para el desarrollo web es el MVC (Modelo-Vista-Controlador). Cabe mencionar que otra buena práctica para el desarrollo de estas aplicaciones es crear diagramas de despliegue (o también llamados diagramas de implementación) que me permitan observar cómo se estructura un proyecto y cuáles son sus componentes.

Específicamente, para el patrón MVC que usaremos durante el desarrollo del proyecto, notaremos que en el servidor que use NodeJS y Express, se tendrá también una estructura de carpetas: rutas, vistas, controladores, modelos, public, y *utils* (para cosas genéricas que los demás folders utilicen). Un proyecto hoy en día puede conectarse a *n* número de BD.

Modelo - Conexión con datos (BD)
Vista - Manejo de interfaces (ojo: no sólo UI)
Controlador - Puente entre modelo y vistas, reciben y manejan los requests.

Para inicializar el proyecto:

```
npm init -y
sudo npm install express
sudo npm install ejs
sudo npm install body-parser
```

Para correr el servidor:

```
pm2 start index.js --watch
```

Para *matar* el servidor:

```
pm2 kill
```

Repositorio de proyecto

https://github.com/tuzobus/veinte-25