Desafío



Aclaraciones:

El siguiente desafío fue diseñado por el Staff de G&L Group con el fin de evaluar los conocimientos de los postulantes al curso de PHP.

Puede encontrar distintas formas de resolver el desafío, le recomendamos priorizar los métodos más óptimos que encuentre, priorice siempre la performance.

Este desafío no representa un examen de ingreso a la empresa, pero es obligatoria su aprobación para la obtención del certificado del curso.

Datos

### Generar una base de datos de nombre “desafio2” con las siguientes tablas:

|  |  |  |
| --- | --- | --- |
| Tabla: usuarios | | |
| Nombre Campo | **Tipo dato** | **Atributos** |
| id | Int | Obligatorio, Clave primaria,  Autoincremental |
| usuario | longitud máxima de 50 caracteres | Obligatorio, Clave primaria |
| nombre | longitud máxima de 50 caracteres | - |
| apellido | longitud máxima de 50 caracteres | - |
| contrasena | longitud máxima de 255 caracteres | Obligatorio |
| tipo | Int | Obligatorio, Si es 0 es administrador.  Si es 1 es usuario. |
| estado | Int | Obligatorio Si es 0 esta inactivo  Si es 1 esta activo |
| fechaAlta | Date | Current timestamp |

|  |  |  |
| --- | --- | --- |
| Tabla: materias | | |
| Nombre Campo | **Tipo dato** | **Atributos** |
| id | Int | obligatorio, clave primaria, autogenerado |
| carrera\_id | Int | Obligatorio, clave foranea de la columna id de la tabla Carreras |
| nombre | longitud máxima de 50 caracteres | obligatorio |
| descripcion | Longitud maxima de 255 caracteres | - |
| carga\_horaria | Int | Obligatorio, opciones entre 2,4,8 o 10 |

|  |  |  |
| --- | --- | --- |
| Tabla: carreras | | |
| Nombre Campo | **Tipo dato** | **Atributos** |
| id | Int | obligatorio, clave primaria, autogenerado |
| nombre | longitud máxima de 50 caracteres | Obligatorio |
| descripcion | Longitud maxima de 255 caracteres | - |

|  |  |  |
| --- | --- | --- |
| Tabla: Cursadas | | |
| Nombre Campo | **Tipo dato** | **Atributos** |
| id | Int | Obligatorio, clave primaria, autogenerado |
| usuario\_id | Int | Obligatorio, clave foranea de la columna id de la tabla Usuarios |
| materia\_id | Int | Obligatorio, clave foranea de la columna id de la tabla Materias |
| nota | Int | Opcional |
| fecha | DATE | Opcional |

Vamos a darle 2 script para esto:   
1) “estructuras.sql” donde se crearan las tablas.   
2) “datos.sql” donde tendrán las carreras y los usuarios.

Funcionalidades

### Se pide: desarrollar un sistema web que se conecte con la base de datos creada anteriormente y posea los siguientes controles y funcionalidades:

**Pantalla LOGIN**

**Obligatorio:**

* No se debe mostrar la contraseña que se ingresa.
* No pueden ingresar usuarios y/o administradores que no estén activos.
* Se deben validar los datos ingresados (es decir, no se puede llamar al servidor, si los campos ingresados, están vacíos).

**Pantalla Principal (depende del tipo de usuario)**  
Para usuarios tipo administradores: (es decir con tipo = 0 )   
  
**Obligatorio:**

* Debe validarse que ingresó un administrador activo (validación con el servidor).
* Debe tener un menú propio, que sea distinto al menú de usuarios.
* El menú debe tener acceso a ABM de Materias (“MATERIAS”) y acceso a ABM de Cursadas (“CURSADAS”).

**Opcional:**

* Menú con acceso a ABM de Usuarios (“USUARIOS”)
* Menú con acceso a ABM de Correlatividades (“Correlatividades”)

Para usuarios tipo usuarios: (es decir con tipo = 1 )   
  
**Obligatorio:**

* Debe validarse que ingresó un usuario activo (validación con el servidor).
* Debe tener un menú propio, que sea distinto al menú de administradores.
* El menú debe tener acceso al historial del usuario (“HISTORIAL”).

**Opcional:**

* Menú con acceso a datos del usuario.
* Se debe poder tener acceso a una opción o sección, donde pueda desconectarse (y re direccionar al login).

**Pantalla MATERIAS (para administradores)**  
**Obligatorio:**

* Debe poder accederse desde la opción del menú de administradores (no puede ser accedida por un usuario).
* Debe listar todas las materias que ya fueron registradas (como se hizo en el desafío 1).
* Debe poder Crear una materia para una carrera previamente registrada (como se hizo en el desafío 1).
* Debe poder Modificar una materia (como se hizo en el desafío 1)

**Opcional:**

* Contar con un filtro de las materias en el listado (se aconseja utilizar el comparador LIKE para implementar, también se aconseja usar el evento keyup de jQuery).

**Pantalla CURSADAS (para administradores)**  
**Obligatorio:**

* Debe poder accederse desde la opción del menú de administradores (no puede ser accedida por un usuario).
* Debe listar todas las cursadas que ya fueron registradas.
* Debe poder dar de alta una cursada, una cursada involucra, un usuario registrado (que sea de tipo usuario y este activo) y una materia que este registrada.
* Se debe poder Modificar una cursada registrada.
* Se debe poder Eliminar una cursada registrada.
* La materia y el usuario a seleccionar, deben ser selects que se generen en base a los datos que hay en la base de datos, funcionamiento similar a la selección de carrera, cuando se da de alta una materia, no se deben ver los usuarios que no estén activos (y obviamente no deben aparecer usuarios administradores).

**Opcional:**

* Contar con un filtro de las cursadas en el listado que me permita filtrar por algo, ya sea usuario o materia.

**Pantalla HISTORIAL (para usuarios)**  
**Obligatorio:**

* Debe poder accederse desde la opción del menú de usuarios (no puede ser accedida por un administrador).
* Debe listar todas las cursadas que fueron registradas para ese usuario.

**Opcionales:**

* La contraseña guardada en la base de datos, se encuentre encriptada con la función SHA1().

**Aclaraciones**

* El código debe estar comentado e indentado (tabulado).
* El sistema será probado en Google Chrome.
* Se recomienda poner separados los archivos de JavaScript, es decir, un archivo JS para cada vista.
* En caso de utilizar más campos en la base de datos, informarlo de alguna forma clara, para tener en cuenta en el momento de la corrección.

**Formato de entrega:**

Un zip con el nombre y apellido de los integrantes, ejemplo si son Ramiro Cerrato y Sergio Raggio, se debe llamar “**RamiroCerrato-SergioRaggio.zip**”, el mismo debe contener:

* Código fuente
* Scripts de la base de datos ( 2 archivos):
  + “estructura.sql” que contenga la estructura de la base de datos
  + “datos.sql” que contenga los datos brindados, y datos de ejemplo
* Documentación (un archivo en Word que contenga explicaciones que crean necesarias, posibles mejoras, extras que hicieron, en caso de hacerlos)

# 