

DOCUMENTACIÓN MIIB

Daw 1 Reto Segunda Evaluación



Beñat Ezquerro, Ivan Boiev, Markel Irastorza e Imanol Pérez

Página Web HTML, CSS, JS

Al entrar a página principal, aparece texto personalizado y fotos de nuestro complejo. Más abajo hay botones, que te llevan a registrarse. Allí puedes poner tus datos personales para registrarse.

A continuación se muestran botones pulsando sobre los cuales podrá conocer más información sobre el restaurante o gimnasio. Allí también encontrará un botón de menú. Al hacer clic en él, accederá a una página web con el menú de nuestro restaurante. También en el apartado de actividades encontrarás un botón que te llevará a una página con todas las actividades de nuestro complejo. Allí podrás fotos de nuestros deportes. También hay un botón en nuestro sitio web que lo lleva a una galería de fotos. Allí podrás hacerte una mejor idea de cómo son nuestros lugares. Casi al final hay un mapa. Podrás utilizarlo para estar seguro de la dirección de nuestro establecimiento. En la parte inferior hay un enlace a nuestra página de políticas. Allí podrá leer más sobre todas las partes de nuestra política de privacidad.

En cuanto a la parte de HTML y CSS, intentamos que todo fuera lo más bonito y correcto posible. Tomamos información de Internet para mejorar la apariencia de nuestro sitio. Puedes notar esto en el encabezado cuando accedes a la página principal, o en el pie de página en forma de ondas en la parte inferior. También utilizamos JavaScript para mejorar la funcionalidad de nuestro sitio.

¡Regístrate ya!

RESTAURANTE	GIMNASIO
-------------	----------



```
function ocultarBarra() { // Funcion para ocultar el menu al hacer click en la X
var barra_lateral = document.querySelector(".barra_lateral");
var gris = document.getElementById("gris");

if (barra_lateral.style.display === "block") {
    barra_lateral.style.display = "none";
    gris.style.display = "none";
} else {
    barra_lateral.style.display = "block";
    gris.style.display = "block";
}
}
```

✕

RESTAURANTE

ZONA DE DEPORTE

MAPA

POLÍTICA

INSCRÍBETE

RESTAURANTE

ZONA DE DEPORTE

```
<button class="menu_button"></button>
</div>
</head>
<body>
<div id="gris"></div>
<div class="barra_lateral">
<button id="ocultarBarra"></button>
<h3><a href="#restaurante" class="barra_link">Restaurante</a></h3>
<h3><a href="#gym" class="barra_link">Zona de deporte</a></h3>
<h3><a href="#mapa" class="barra_link">Mapa</a></h3>
<h3><a href="politica.html" class="barra_link">Política</a></h3>
<h1 id="inscripcion_menu">Inscríbete </h1>
<h3><a href="formularios/restaurante.php" class="barra_link">Restaurante</a></h3>
<h3><a href="formularios/gimnasio.php" class="barra_link">Zona de deporte</a></h3>
</div>
```

XML,XSD: JERARQUIZACION DE GUARDADO DE DATOS

Hemos hecho un esquema de guardado de datos con respecto a las tablas "Deportistas", "Comensales" y "Empleados" en las cuales hemos puesto patrones y restricciones para que el guardado de datos sea correcto y fiel a el orden y contenido de las tablas.

```
<?xml version="1.0" encoding="UTF-8"?>
<Susccripcion xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="SubsGym.xsd">
  <Susccripcion>
    <NombreD>Benat</NombreD>
    <ApellidoD>Ezquerro</ApellidoD>
    <Email>bezquerro@gmail.com</Email>
    <Telefono>+34666555222</Telefono>
    <FechaInicio>2024-02-18T11:30:00</FechaInicio>
    <tipoSubs>Semanal</tipoSubs>
    <Zona>kayak</Zona>
  </Susccripcion>
</Susccripcion>

<xsd:element name="ApellidoD">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="[A-Z]{1}[a-z]*" />
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

FORMULARIOS DE RESERVA A GIMNASIO Y RESTAURANTE Y PHP, INSERCIONES A LA BASE DE DATOS

Cuando comenzamos a crear los formularios, íbamos a crear un solo formulario con un desplegable y según lo que se elija en el desplegable, se Reserva el Restaurante o el Gimnasio. Al final se optó por crear un formulario individual para cada tipo de reserva



Nos quedamos con tres formularios los cuales son el formulario de Reservas a Restaurante, el de Suscripciones al Gimnasio y uno para Aplicar a un trabajo en el restaurante o el gimnasio.

Estos formularios tienen el código PHP separado, pero inicialmente el código PHP iba a estar integrado en el formulario, ya que era más sencillo de implementar. Finalmente se decidió crear un PHP para los distintos formularios porque cuando estaban integrados, al enviar el formulario, ocurrían errores con el código de Java. Al meter el HTML del formulario en el archivo PHP, el código de PHP se ejecutaba dentro del formulario, así logrando evitar redirecciones.

```
$nombre = $_POST["nombre"];
$apellidos = $_POST["apellido"];
$email = $_POST["email"];
$telefono = $_POST["telefono"];
$fecha = $_POST["fecha"];
$fecha_format = substr($fecha, 0, 10) . ' ' . substr($fecha, 11, 5) . ':00';
$menu = $_POST["menu"];
$mesa = rand(1, 40);
$cod = "A";
$num_mesa = $mesa . ' ' . $cod;
```

Dentro del código de PHP se empieza extrayendo los valores de los campos del formulario, dependiendo del método usado en el formulario (POST y GET).

Luego se crea la conexión a la base de datos, creando variables que guardan los parámetros para la conexión y usándolos en otra variable para la conexión:

\$conn = new mysqli(\$servidor, \$usuario, \$password, \$basedatos); y luego se verifica la conexión por si hay errores al conectarse. Tras verificar la conexión, se crean variables con la query a ejecutar en la base de datos y esas variable se ejecutan llamando a la variable de la conexión: **\$conn->query(\$consulta)**. Tras haber hecho todas las query necesarias, se cierra la conexión: **\$conn->close();**

```
$servidor = "dbrds.cicqmqa0ite.us-east-1.rds.amazonaws.com";
$usuario = "admin";
$password = "ASdiiqw--ad45";
$basedatos = "BBDDProyectoGym1";

// Crear conexión usando las credenciales
$conn = new mysqli($servidor, $usuario, $password, $basedatos);

// Verificar conexión
if ($conn->connect_error) {
    die("Conexión fallida: " . $conn->connect_error);
}

// Variables en las que guardo consultas a la base de datos ("$sql_comensales" y "$sql_servir")
$sql_comensales = "INSERT INTO Comensales (NombreC, ApellidoC, Email, Telefono, DiaYHora, idMesa, idMenu)
('$nombre', '$apellidos', '$email', '$telefono', '$fecha_format', '$mesa', '$menu')";
// Función para ejecutar la consulta
$conn->query($sql_comensales);
```

Tras eso se hizo un código JavaScript para que al enviar el formulario, saliera un Pop-Up en la parte superior de la pantalla indicando si se ha hecho la reserva correctamente o no. Si la reserva se guarda sin problemas en la base de datos, se le redirigirá a la pagina PHP diciendo que la reserva/suscripcion se ha hecho con éxito, mostrando los datos del solicitante y opciones para volver al menu o a la pagina de rellenar el formulario. En caso de haber cualquier error en la solicitud por campos dejados en blanco, se notificará mediante un Pop-Up con un mensaje de error que dice "Todos los campos deben de ser rellenados, inténtelo de nuevo" y dejará al usuario corregir los datos.

Eso se hizo con una función de JavaScript llamada "Mensaje()" que se ejecuta tras enviar el formulario. Esta función extrae los datos del formulario y los guarda en variables, para luego usarlas en la condición. Si la condición se cumple, se crea un window.alert() que saca un Pop-up notificándole al usuario del error y devuelve un return false para que el formulario no se envíe.

```

funcion Mensaje(){
    window.alert("nombre");
    // variables en las que guardo los datos de los campos
    var nombre = document.forms["formGym"]["nombre"].value;
    var apellido = document.forms["formGym"]["apellidos"].value;
    var email = document.forms["formGym"]["email"].value;
    var telefono = document.forms["formGym"]["telefono"].value;
    var suscripcion = document.forms["formGym"]["suscripcion"].value;
    var fecha = document.forms["formGym"]["fecha"].value;
    var zona = document.forms["formGym"]["zona"].value;
    var nombre_apellido = nombre + " " + apellido;
    window.alert("Error al insertar los datos, inténtelo de nuevo");
    // muestra un pop up, si todos los campos estan completos, te da la bienvenida. Si no, te da error al insertar los datos y
    if (nombre == "" || apellido == "" || fecha == "" || email == "" || telefono == "" || suscripcion == "" || zona == "") {
        window.alert("Error al insertar los datos, inténtelo de nuevo");
        return false; // Evitar el envío del formulario si hay un error
    }
    else {
        return true; // Enviar el formulario en caso de que todos los datos sean correctos
    }
}

```

JAVA, CONSULTAS A LA BASE DE DATOS(SWING)

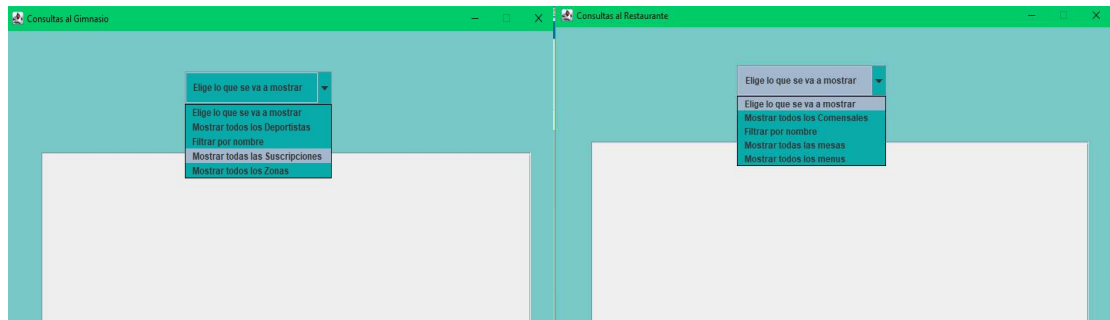
Al empezar, pensábamos hacer todas las peticiones(las de ambos sectores) en una misma ventana y solo usar una ventana pero al final pensamos que sería mejor dividirlo en sectores para optimizar el código y facilitar su lectura al estar dividido.

Programa Construido en Java SWING, este programa está hecho para que los empleados de MIIB Restauración & Deporte puedan consultar registros de la base de datos sobre el restaurante y la zona de deportes. Cuenta con tres ventanas, una en la que eliges el sector de el cual quieres extraer datos, usando un JComboBox para elegir el sector y un JButton para confirmar la elección. Además cuenta con un fondo azul y el logo que representa nuestra marca. Al confirmar la elección de sector, se abrirá la respectiva ventana por encima.



Por otro lado tenemos las ventanas de los sectores, estas están conformadas de un JComboBox que te da cuatro opciones. En el caso del restaurante son: Mostrar todos los comensales, Filtrar por nombre, Mostrar todas las mesas, Mostrar todos los menús. En la zona de deporte son: Mostrar todos los Deportistas, Filtrar por nombre, Mostrar todas las Suscripciones, Mostrar todos las Zonas. Al elegir la opción que comparten ambos ("Filtrar por nombre"), se mostrará un cuadro de texto para filtrar los resultados por el nombre escrito y un botón para confirmar esos cambios. Al elegir cualquiera de las opciones y confirmar la elección, el programa te mostrará una tabla (hecha con JTable) con la información pedida. Estas ventanas tienen un fondo azul y botones de color turquesa

oscuro, además de el logo de MIIB como icono de aplicación



BASE DE DATOS

Inicialmente iba a haber un hotel, con lo cual iba a haber tres tablas más (ZonaDescanso, Suites, Descansa[Tabla N:M] y Huéspedes) pero tras haber hecho la base de datos completa, se decidió que sería muy complicado hacerlo así, con lo que descartamos la idea del hotel y seguimos con el resto de tablas.

Tras rehacer toda la base de datos y ponerse a hacer los formularios, nos dimos cuenta de que la base de datos no estaba bien diseñada así que la volvimos a rehacer y al final nos quedamos con ese diseño.

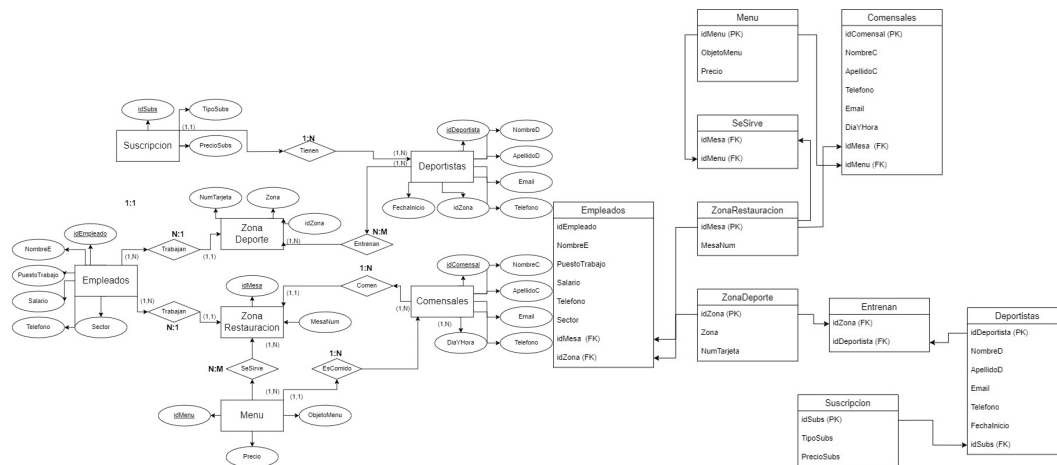
Tablas de la base de datos:

- ZonaRestauracion(idMesa, MesaNum)
- ZonaDeporte(idZona, NumTarjeta, Zona)
- Menu(idMenu, Precio, ObjetoMenu)
- Comensales(idComensal, NombreC, ApellidoC, Email, Telefono, DiaYHora)
- Deportistas(idDeportista, NombreD, ApellidoD, Email, Telefono, idZona)
- Suscripcion(idSubs, TipoSubs, PrecioSubs)
- Empleados(idEmpleado, NombreE, PuestoTrabajo, Salario, Telefono, Sector)

Nuestra base de datos está principalmente compuesta por dos tablas centrales, ZonaDeporte y ZonaRestauracion. Ambas tablas están conectadas a la tabla de Empleados con relación 1:N[Trabajan] (de Zona a Empleado) en ambos casos, pues los empleados trabajan en ambas zonas. La tabla ZonaRestauracion tiene conectada la tabla Menu mediante una relación N:M[SeSirve] y tiene conectada la tabla Comensales mediante una relación N:1 (Hacia ZonaRestauracion). A su vez, la tabla Menú está conectada a la tabla Comensales con una relación de 1:N[EsComido] (de menú a Comensal).

La tabla ZonaDeporte está conectada a la tabla Deportistas con una relación N:M[Entrenan] y la tabla Deportistas está conectada a la tabla

Suscripcion usando una relación de 1:N[Tienen] (de Suscripción a Deportistas).




SERVIDOR AWS(BASE DE DATOS) y SERVIDOR DE NAZARET(SERVIDOR WEB)

Al inicio pensábamos alojar tanto la base de datos como la página web dentro del servidor EC2 de AWS Academy, pero tuvimos problemas con Java así que terminamos creando un servidor RDS para la base de datos y usando el servidor de Nazaret como servidor web.

El servidor RDS de Amazon Web Services lo usamos para alojar la base de datos. Es un tipo de instancia específica para montar bases de datos y en el que se pueden crear varias bases de datos. Por otro lado está el servidor de Nazaret,

El cual lo usamos para alojar la página web y que pueda ser visible tanto desde la misma red como desde una red externa. Se puede acceder desde internet poniendo la ip del router al que está conectado el servidor (85.50.79.98) y el puerto ssh junto con el puerto HTTP (23 y 80).

La URL para acceder desde fuera de Nazaret sería la siguiente: <http://85.50.79.98:2380>

 <http://85.50.79.98:2380>

La URL para acceder desde Nazaret sería la siguiente: <http://192.168.1.223>

 <http://192.168.1.223>