
INSTITUTO TECNOLÓGICO DE CUAUTLA
INGENIERÍA EN SISTEMAS COMPUTACIONALES

Diseño y Desarrollo de Aplicaciones Móviles.

Docente: I.S.C. ARMANDO RODRÍGUEZ JÍMENEZ

PRÁCTICA FINAL.
Menú de un Restaurante con integración
de sección de comentarios y chat.

Entrega 2: Sección de Chat.

Presentan:

Onofre Ramírez Blanca Haidée	13680224
Pérez Bolaños Lizbeth	13680233

GRUPO 1
TURNO: Matutino
Octavo semestre

Introducción:

La opinión de los clientes es importante para todas las empresas, ya que los ayuda a mejorar el servicio que ofrecen. Por lo tanto, se debe implementar páginas web dinámicas, que permitan una fácil interacción con el usuario.

Para lograr esta interacción diversos sitios web implementan técnicas como son los comentarios y chats. La utilización de un chat permite que el usuario pueda resolver sus dudas, enviándole un mensaje al administrador.

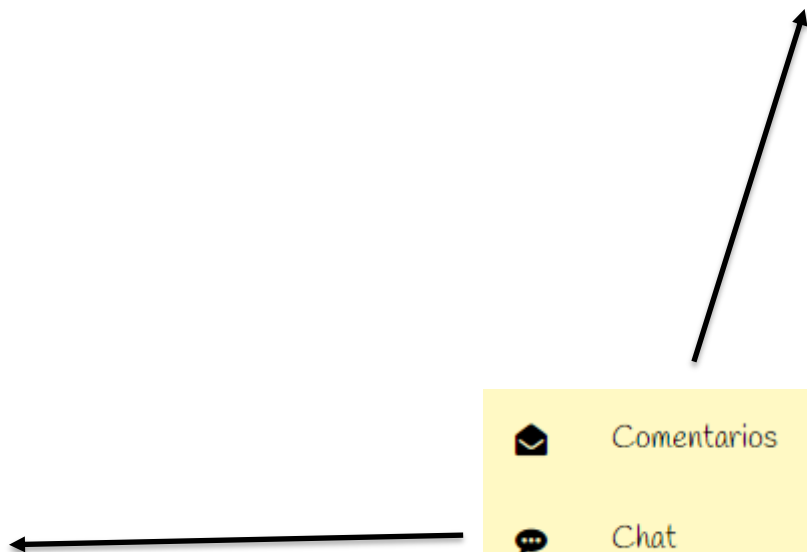
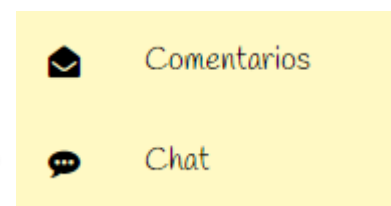
Sección de Chat

La sección de chat se puede acceder mediante la barra de navegación, donde se encuentra un menú desplegable con el nombre Danos tu opinión, mostrando dos opciones comentarios y chat.

Vista de escritorio



Vista móvil



Maquetado HTML del elemento Danos tu opinión en el menú de navegación.

La creación del menú desplegable se realiza con una lista no ordenada `` mediante la clase `dropdown-content` de materialize.

Menú desplegable escritorio

```
<ul id="menu3" class="dropdown-content">
  <li>
    <a class="black-text" href="opinion.html"><i class="fa fa-envelope-open black-text" aria-hidden="true"></i>Comentarios</a>
  </li>
  <li><a class="black-text" href="chat.html"><i class="fa fa-commenting black-text" aria-hidden="true"></i>Chat</a>
  </li>
</ul>
```

Menú desplegable móvil

```
<ul id="menu4" class="dropdown-content">
  <li>
    <a class="black-text" href="opinion.html"><i class="fa fa-envelope-open black-text" aria-hidden="true"></i> Comentarios</a>
  </li>
  <li><a class="black-text" href="chat.html"><i class="fa fa-commenting black-text" aria-hidden="true"></i> Chat</a>
  </li>
</ul>
```

Para el despliegue del menú se utiliza la clase `dropdown-button` de materialize y su activación se realiza por medio del Id señalado en su diseño.

Vista de escritorio

```
<li>
  <a class="dropdown-button black-text" href="#" data-activates="menu3">
    <i class="fa fa-comments fa-2x black-text prefix" aria-hidden="true"></i>Danos tu opinión
    <i class="fa fa-chevron-circle-down right" aria-hidden="true"></i>
  </a>
</li>
```

Vista móvil

```
<li>
  <a class="dropdown-button black-text" href="#" data-activates="menu4">
    <i class="fa fa-comments black-text prefix" aria-hidden="true"></i> Opinión
    <i class="fa fa-chevron-circle-down black-text right" aria-hidden="true"></i>
  </a>
</li>
```

Página de Chat


La página de chat se divide en dos vistas: administrador y cliente. En la parte superior de ambas vistas, se encuentra un enlace que permite al usuario iniciar sesión con su cuenta de Google.

Vista administrador




En la vista del administrador se hace uso de la etiqueta table para mostrar los usuarios registrados. Por otro lado, el div con Id contenedorMensajes se utiliza para mostrar los mensajes enviados, el textarea con Id mensaje se emplea para enviar el mensaje escrito y la etiqueta a cuenta con un evento llamado guardarMensaje.


```
<div class="col l6 m4 s3 scroll card-panel lime lighten-4" id="formChat">
  <h5 id="mensajes"><i class="material-icons">person_pin</i> Usuarios:</h5>
  <table id="listaUsuarios"></table>
</div>
<!--chat-->
<div class="col l6 m8 s9 row card-panel" id="formChat">
  <!--Lugar en donde se mostraran los mensajes-->
  <div class="col l12 m12 s12 scroll" id="contenedorMensajes"></div>
  <!--Aquí se escribe el mensaje a enviar-->
  <form>
    <div class="col l10 m10 s10">
      <textarea id="mensaje" rows="5" cols="10"></textarea>
    </div>
    <!--Envío del mensaje-->
    <div class="col l2 m2 s2">
      <a onClick="guardarMensaje()" href="#"><i class="small material-icons red-text">send</i></a>
    </div>
  </form>
</div>
```

Tu opinión nos importa

 Iniciar sesión con Google

Usuarios:

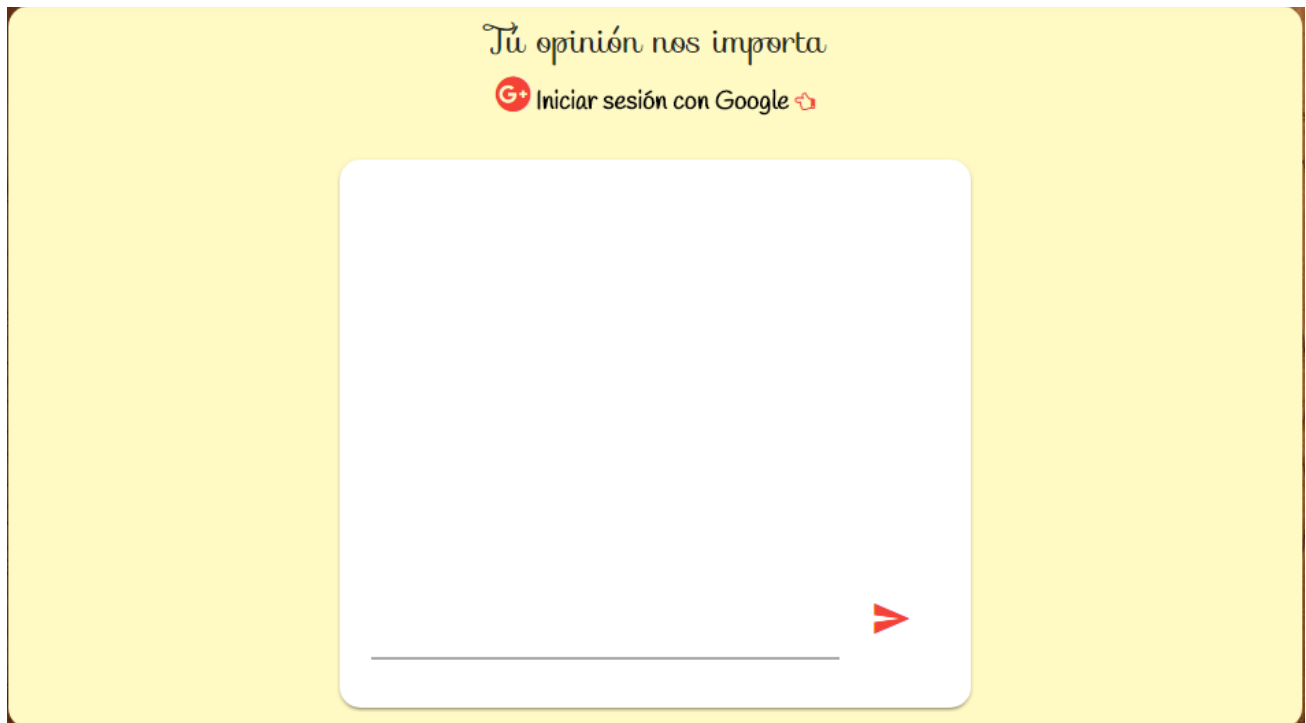
- ☐ Administrador Mamma Mia 
- ☐ Blanca Haidee Onofre Ramirez 
- ☐ Lizbeth Pérez Bolaños 



Vista cliente

En la vista del cliente se utiliza del div con Id contenedorMensajes para mostrar los mensajes enviados, el textarea con Id mensaje se emplea para enviar el mensaje escrito y la etiqueta a cuenta con un evento llamado guardarMensaje.

```
<div class="col l3"></div>
<!--chat-->
<div class="col l6 m12 s12 row card-panel" id="formChat">
  <!--Lugar en donde se mostraran los mensajes-->
  <div class="col l12 m12 s12 scroll" id="contenedorMensajes"></div>
  <!--Aquí se escribe el mensaje a enviar-->
  <form>
    <div class="col l10 m10 s10">
      <textarea id="mensaje" rows="5" cols="10"></textarea>
    </div>
    <!--Envio del mensaje-->
    <div class="col l2 m2 s2">
      <a onClick="guardarMensaje()" href="#"><i class="small material-icons red-text">send</i></a>
    </div>
  </form>
</div>
<div class="col l3"></div>
```



Script de funcionalidad de la página chat

En el archivo [chatAdmin.html](#) y [chat.html](#) se agrega el script llamado chat.js en el cual se integra la configuración de la BD, se realizan las operaciones necesarias para el almacenamiento y manejo de datos.

Para el correcto funcionamiento del chat se agregan los elementos del DOM que serán manipulados.

```
/*Agregando los elementos del DOM para chat*/
const mensaje = document.getElementById('mensaje');
const enviar = document.getElementById('enviar');
const contenedor = document.getElementById('contenedorMensajes');
const textSesion = document.getElementById("textoSesion");
const nameInput = document.getElementById("name");
const listUsers = document.getElementById("listaUsuarios");
```

Función inicio sesión

Cuando el usuario inicia sesión sus datos son guardados en el directorio [users](#), al mismo tiempo se actualiza los datos dentro del directorio [conexion](#), con el fin de cambiar el estado de conexión del usuario en la vista del administrador.

```
function iniciarSesion() {
  const provider = new firebase.auth.GoogleAuthProvider();
  firebase.auth().signInWithPopup(provider).then((result) =>{
    var token = result.credential.accessToken;
    var user = result.user;
    let userName = result.user.displayName;
    let email = result.user.email;
    var photo = result.user.photoURL;
    // The Google credential, this contain the Google access token:
    let credential = result.credential;

    /*Guarda el nombre del usuario conectado*/
    usuarioConectado = userName;

    /*Guardar Usuario*/
    firebase.database().ref('users/' + usuarioConectado).update({
      username: userName,
      email: email,
      foto: photo
    });
    /*verificar usuarios en línea */
    firebase.database().ref('conexion/' + usuarioConectado).update({
      username: userName,
      status: 'conectado'
    });
  }).catch(error => console.error(`Error : ${error.code}: ${error.message}`));

  if(usuarioConectado !== admon){
    vistaCliente();
  }else if(usuarioConectado === admon){
    vistaAdmon();
  }
}
```

Funcion cerrar sesión

Cuando el usuario cierra la sesión sus datos son guardados en el directorio users, al mismo tiempo se actualiza los datos dentro del directorio conexion, con el fin de cambiar el estado de conexión del usuario en la vista del administrador.

```
function cerrarSesion() {
  firebase.auth().signOut()
    .then(() =>{
      console.log('te has deslogeado')
      nameInput.value = '';
    }).catch(error => console.error(`Error : ${error.code}: ${error.message}`));

  /*verificar usuarios en linea */
  firebase.database().ref('conexion/'+ usuarioConectado).update({
    name: usuarioConectado,
    status: 'desconectado'
  });

  borrarMensajes();
}
```

Mostrar usuarios en la vista del administrador

Se hace referencia a la B.D. conexion para tomar todos los datos almacenados, posteriormente se toma el valor status y se realiza una comparación.

- Si el valor es igual a conectado, se envía un icono de color verde.
también se añaden los controles necesarios para su manipulación.
- Si el valor es igual a desconectado, se envía un icono de color gris.
Asimismo, se desactiva el checkbox.

Finalmente, para tener una visualización estructurada se emplea una tabla, enviandola al HTML mediante la variable listUsers que hace alusión al Id listaUsuarios.

```
/*Mostrar usuarios en la vista del chat Administrador*/
firebase.database().ref('conexion').on('value', function (snapshot) {
  let html = '';
  snapshot.forEach(function(e) {
    let elemento = e.val();
    let nombre = elemento.name;
    let estado = elemento.status;
    if(estado == "conectado"){
      html += `
      <tr>
        <td>
          <input type="checkbox" name="checkbox" id="${nombre}" value="${nombre}"><label for="${nombre}">${nombre}</label>
        </td>
        <td>
          <i class="material-icons light-green-text accent-3-text">check_circle</i>
        </td>
      </tr> `;
    }else{
      html += `
      <tr>
        <td>
          <input type="checkbox" id="usuario" disabled="disabled"><label for="usuario">${nombre}</label>
        </td>
        <td>
          <i class="material-icons grey-text darken-3-text">cancel</i>
        </td>
      </tr> `;
    }
  });
  listUsers.innerHTML = html;
});
```

Función para guardar los mensajes

En primer lugar verificamos si el mensaje proviene del cliente o del administrador.

- Si se trata del cliente, se procede a generar un nuevo documento dentro del directorio chat/nombre del cliente conectado, en donde se almacenará el nombre del cliente, el mensaje y la hora en que se envía el mensaje.
- Si es el administrador, se verifica el nombre del cliente seleccionado mediante los controles indicados en su creación, en seguida se guarda el mensaje en el directorio chat/nombre del cliente seleccionado guardando el nombre del administrador, el mensaje y la hora de enviado.

```
/*Evento para enviar y guardar mensajes */
function guardarMensaje(){

    /*Recupera el mensaje ingresado */
    var mensajeEnviado = mensaje.value;
    var usuarioConectado = nameInput.value;

    /*Envia los datos a la BD*/
    if( usuarioConectado != admon){
        firebase.database().ref('chat/' + usuarioConectado).push({
            name : usuarioConectado,
            message: mensajeEnviado,
            time: timeStamp()
        });
        mensaje.value = "";
        vistaCliente();
    }
    else{
        /*Usuario seleccionado*/
        var cliente = document.getElementsByName("checkbox");
        for (var x=0; x < cliente.length; x++) {
            if (cliente[x].checked) {
                usuarioConectado = cliente[x].id;
            }
        }
        /*Envia los datos a la BD*/
        firebase.database().ref('chat/' + usuarioConectado).push({
            name : admon,
            message: mensajeEnviado,
            time: timeStamp()
        });
        mensaje.value = "";
        vistaAdmon();
    }
}
```


Función vista cliente

Se hace referencia al directorio chat/nombre del cliente, se recuperan todos los datos y se almacenan en variables, con el fin de crear la estructura que mostrará el nombre del usuario, el mensaje y la fecha en la que fue enviado.

```
/*Mostrar los mensajes enviado en el contenedor */
/*Cliente */
function vistaCliente(){
  firebase.database().ref('chat/' + nameInput.value).on('value', function(snapshot){
    var html = '';
    snapshot.forEach(function(e) {
      var elemento = e.val();
      var usuario = elemento.name;
      var mensajeEnviado = elemento.message;
      var fecha = elemento.time;
      html += `<hr>
      <h5 id="nombreUsuario">${usuario} </h5>
      <p id="mensajes"> ${mensajeEnviado}</p>
      <span id="fecha">${fecha}</span>`;
    });

    contenedor.innerHTML = html;
    /*Posicionar el scroll*/
    contenedor.scrollTop = contenedor.scrollHeight;
  });
}
```

Función vista administrador

Para la vista administrador, primero se verifica el cliente seleccionado por el administrador, para luego hacer referencia al directorio chat/nombre del cliente selecionado, posteriormente se recuperan los datos y se guardan en variables, por último se crea la estructura que presentará el nombre del usuario, el mensaje y la fecha de envío.

```
/*Administrador */
function vistaAdmon() {
  /*Usuario seleccionado*/
  var cliente = document.getElementsByName("checkbox");
  for (var x=0; x < cliente.length; x++) {
    if (cliente[x].checked) {
      usuarioConectado = cliente[x].id;
    }
  }
  firebase.database().ref('chat/' + usuarioConectado).on('value', function(snapshot){
    var html = '';
    snapshot.forEach(function(e) {
      var elemento = e.val();
      var usuario = elemento.name;
      var mensajeEnviado = elemento.message;
      var fecha = elemento.time;

      html += `<hr>
      <h5 id="nombreUsuario">${usuario} </h5>
      <p id="mensajes"> ${mensajeEnviado}</p>
      <span id="fecha">${fecha}</span>`;
    });
    contenedor.innerHTML = html;
    /*Posicionar el scroll*/
    contenedor.scrollTop = contenedor.scrollHeight;
  });
}
```

Funcion borrar mensajes

Finalmente, cuando se cierra la sesión se borran los datos.

```
/*Borrar los mensajes al cerrar sesión*/  
function borrarMensajes(){  
    var html = '';  
    html += `<hr>  
        <h5 id="nombreUsuario"></h5>  
        <p id="mensajes"></p>  
        <span id="fecha"></span>`;   
    contenedor.innerHTML = html;  
}
```

Conclusiones

Anteriormente, las páginas web eran únicamente informativas, sin embargo, con el paso del tiempo surgió la necesidad de implementar técnicas y herramientas que permitieran tener una mayor interacción con el usuario.

La integración de un chat en un sitio web beneficia la comunicación entre el cliente y la empresa, permitiendo resolver las dudas que tenga el usuario en tiempo real. Es importante destacar que este tipo de herramienta es aplicable en diversas áreas, aunque su objetivo pueda ser diferente.

El diseñar y desarrollar un chat no es tarea fácil, a pesar de ello, con la implementación de tecnologías como CSS, HTML, Javascript y Firebase, se facilita su realización. HTML en conjunto con CSS cuentan con la capacidad de crear vistas agradables para el usuario, mientras que Javascript permite el manejo de los datos ingresados, sin embargo, se requiere incluir bases de datos para almacenar la información, por esa razón se empleo firebase que cuenta con diversas funciones como la autenticación, además de facilitar el ingreso, modificación y consulta de los datos.