Benemérita Universidad Autónoma de Puebla
Facultad de Ciencias de la Computación

D E S A R R O L L O D E
APLICACIONES WEB

Examen 3: Formulario con validación

Docente:
Prof. Roberto Elvira Enríquez

Jesús Huerta Aguilar
202041509

NRC: 12562
Sección: 003

SEXTO SEMESTRE

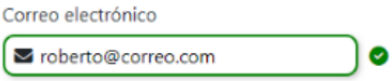
Puebla, Pue. 17/11/2023

ACTIVIDAD 1:
FORMULARIO

ELEMENTOS DETECTADOS A SIMPLE VISTA

Para llevar a cabo el desarrollo de este formulario, es fundamental analizar diversos componentes que servirán como base. Inicialmente, nos enfocamos en los elementos <input> de tipo texto. Estos crean campos de una sola línea para ingresar datos como nombre, correo electrónico y contraseña. El campo destinado al nombre no impone restricciones, mostrándose tal como el usuario lo introduce.

En el caso del campo de correo electrónico, las únicas dos condiciones son la ausencia de espacios y la inclusión única del carácter '@'.



Respecto al campo de la contraseña, es esencial que se oculte mediante puntos, algo que se logra fácilmente con una etiqueta HTML específica.

A continuación, consideramos los elementos <button>. Uno de ellos se destina a enviar la información ingresada en los campos anteriores, mientras que el otro serviría para limpiar dicha información. En este proyecto, me centraré exclusivamente en el desarrollo del botón de envío. Al activarse, debe verificar primero que los campos estén correctamente llenos y cumplan con las especificaciones requeridas. Posteriormente, un archivo .js procesará la información. En este archivo, una variable booleana determinará si todos los datos recibidos son correctos. Si hay algún error, esta variable cambiará a 'false'.

Si todo está correcto, la variable permanecerá en 'True', señalando al documento HTML que puede mostrar los datos ingresados en el formulario en el lado derecho de la pantalla, lo cual indicaría que todo funciona adecuadamente.

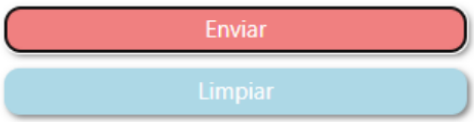
Lo que se mostrará del lado derecho será exactamente los datos que ingreso el usuario, en el mismo orden y con un diseño bastante similar al campo de la izquierda, las únicas diferencias que habrá es

que los campos de texto ahora serán textos dentro de la etiqueta <p> con un borde añadido con la biblioteca Bootstrap además de que estos no tendrán ningún color, solo un gris.

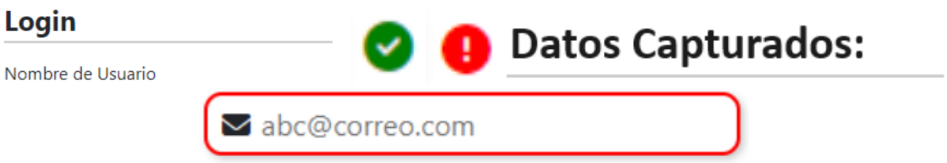
Otra diferencia es que la contraseña ya no aparecerá oculta.

En cuanto al diseño, el formulario de ejemplo establece que los campos incompletos deben resaltarse con un borde rojo, mientras que aquellos completos y correctos tendrán un borde verde.

Los botones mantendrán el color predeterminado proporcionado por Bootstrap, aunque en el documento muestran los botones con distintos colores.



Además, como elementos decorativos adicionales, se contempla el uso de distintos tamaños de fuentes y otros íconos que cambiarían de color al igual que el borde de los campos de texto. Estos detalles de diseño los desarrollaré en la medida que el tiempo me lo permita.



DESARROLLO DEL CÓDIGO (LO MAS IMPORTANTE)

index.html

Bloque de cabecera (<head>): Incluye la definición del tipo de documento (DOCTYPE), la configuración del idioma, la codificación de caracteres (UTF-8) y el título de la página. Se cargan dos hojas de estilo CSS: una de Bootstrap para estilizar la página con elementos predefinidos, y otra personalizada (**style.css**) para estilos específicos del sitio.

```
1. <!DOCTYPE html>
2. <html lang="es">
3. <head>
4.     <meta charset="UTF-8">
5.     <title>Formulario con Validación</title>
6.     <!-- Bootstrap CSS -->
7.     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/
   bootstrap/4.3.1/css/bootstrap.min.css">
8.     <!-- Estilos personalizados -->
9.     <link rel="stylesheet" href="style.css">
10. </head>
```

Cuerpo de la página (<body>):

- **Contenedor principal ( ):** Sirve como contenedor para todos los elementos de la página, utilizando la clase **container** de Bootstrap para un diseño responsivo.

```
13.     <div class="container">
```

- **Cabecera (<header>):** Presenta un encabezado con el título "Formulario FCC" en un formato centrado.

```
14.         <header class="text-center">
15.         <h1>Formulario FCC</h1>
16.     </header>
17.     <br><br>
```

- **Formulario de inicio de sesión (<form id="miFormulario">):** Contiene campos para el nombre de usuario, el correo electrónico y la contraseña, con botón de envío. Está diseñado para ser interactivo y está colocado dentro de una columna de la rejilla de Bootstrap (**col-md-6**).

```
18.           <div class="row">
19.         <!-- Sección del formulario -->
20.         <div class="col-md-6">
21.             <form id="miFormulario">
22.                 <h1>Login</h1>
23.                 <hr>
24.                 <div class="form-group">
25.                     <label for="usuario">Nombre de Usuario:</label>
26.                     <input type="text" class="form-
   control" id="usuario" name="usuario">
27.                 </div>
28.                 <div class="form-group">
29.                     <label for="email">Correo Electrónico:</label>
30.                     <input type="email" class="form-
   control" id="email" name="email">
31.                 </div>
32.                 <div class="form-group">
33.                     <label for="contrasena">Contraseña:</label>
34.                     <input type="password" class="form-
   control" id="contrasena" name="contrasena">
35.                 </div>
36.
37.                 <button type="submit" class="btn btn-
   primary">Enviar</button>
38.             </form>
39.         </div>
```

- **Sección de resultados ( ):** Esta sección, que también utiliza una columna de la rejilla de Bootstrap, está destinada a mostrar los resultados del formulario. Incluye párrafos con **id** para insertar los datos capturados.

```
41.             <div id="resultados" class="col-md-6 oculto">
42.             <h1>Datos Capturados</h1>
43.             <hr>
44.
45.             <label>Nombre de Usuario::</label>
46.             <p class="form-
   control" id="resUsuario"><span id="resUsuario"></span></p>
47.             <label>Nombre de Usuario::</label>
48.             <p class="form-
   control" id="resEmail"><span id="resEmail"></span></p>
49.             <label>Nombre de Usuario::</label>
50.             <p class="form-
   control" id="resContrasena"><span id="resContrasena"></span></p>
51.
52.         </div>
```

- **Pie de página (<footer>):** El pie de página contiene una imagen institucional y un enlace de contacto con la información de la universidad y el autor del código.

```
60.          <footer class="row">
61.         <div class="col-md-6">
62.             <img src="https://comunicacion.buap.mx/sites/all/themes
   /plantilla_1/images/logo_buap.png" class="img-fluid" alt="buap">
63.         </div>
64.         <address class="text-right col-md-6">
65.             Escrito por: <a href="mailto:jesus.huertakg@gmail.com">
   Jesus Huerta Aguilar ©</a><br>
66.             Benemérita Universidad Autónoma de Puebla<br>
67.             Facultad de Ciencias de la Computación<br>
68.         </address>
69.     </footer>
```

- **Integración de Bootstrap y JavaScript:** Se cargan las bibliotecas de JavaScript necesarias para las funcionalidades de Bootstrap, así como un script personalizado (**script.js**), probablemente para manejar la lógica del formulario y la visualización de los resultados.

```
73.      <script src="https://code.jquery.com/jquery-
   3.3.1.slim.min.js"></script>
74. <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.
   7/umd/popper.min.js"></script>
75. <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/
   bootstrap.min.js"></script>
76. <script src="script.js"></script>
77. </body>
78. </html>
```

style.css

Clases .valido y .invalido: Estos estilos se aplican a elementos para indicar su estado de validación. La clase **.valido** agrega un borde verde a un elemento, lo que generalmente se usa para indicar que la entrada del usuario es correcta o aceptable. Por otro lado, la clase **.invalido** añade un borde rojo, utilizado comúnmente para señalar errores o entradas inválidas en un formulario.

```
2. .valido {
3.     border: 2px solid green;
4. }
5.
6. .invalido {
7.     border: 2px solid red;
8. }
```

Clase .oculto: Esta clase establece el estilo **display: none** a los elementos, lo que los hace invisibles en la página. Este estilo es útil para ocultar elementos que solo deben mostrarse bajo ciertas condiciones, como mensajes de error, información adicional, o en este caso, la sección de resultados del formulario hasta que se envíen los datos.

```
1. .oculto {
2.     display: none;
3. }
```

Estilos para header y footer: Estos estilos definen la apariencia de las cabeceras y pies de página de la web. Para ambos, se establece un color de fondo (**#003b5c**, un tono oscuro de azul) y un relleno (**padding**) para dar espacio alrededor del contenido interno. La cabecera (**header**) tiene un color de texto blanco, mientras que el pie de página (**footer**) utiliza un color azul claro (**#00b5e2**) para el texto, creando una apariencia cohesiva y estéticamente agradable.

```
4. header {
5.     background-color: #003b5c;
6.     color: white;
7.     padding: 20px 0;
8. }
9. footer{
10.     background-color: #003b5c;
11.     color: #00b5e2;
12.     padding: 10px;
13.
14. }
```

script.js

Añadir un oyente de eventos al formulario: El script comienza por agregar un oyente de eventos al formulario identificado por **miFormulario**. Este evento se activa cuando el formulario intenta enviarse. La función **event.preventDefault()** se utiliza para evitar que el formulario se envíe de manera predeterminada, lo que permite que el script maneje la lógica de validación y presentación de datos.

```
1. document.getElementById('miFormulario').addEventListener('submit', function(
   event) {
2.     event.preventDefault();
```

Obtención de valores de los campos del formulario: Se recuperan los valores de los campos del formulario (**usuario**, **email**, **contrasena**) utilizando **document.getElementById()**. Estos valores se utilizarán para la validación y para mostrar los resultados si el formulario es válido.

```
5.     var usuario = document.getElementById('usuario');
6.     var email = document.getElementById('email');
7.     var contrasena = document.getElementById('contrasena');
```

Validación de los campos del formulario: Se realiza una validación simple para cada campo, comprobando si están vacíos (utilizando **trim()** para eliminar espacios en blanco). Si un campo está vacío, se le añade la clase **invalido** (definida en **style.css**) para indicar visualmente que hay un error. Si el campo tiene un valor válido, se le asigna la clase **valido**. La variable **esValido** se utiliza para rastrear si todos los campos del formulario son válidos.

```
10.     var esValido = true;
11.     if (usuario.value.trim() === '') {
12.         usuario.classList.add('invalido');
13.         esValido = false;
14.     } else {
15.         usuario.classList.remove('invalido');
16.         usuario.classList.add('valido');
17.     }
18.
19.     if (email.value.trim() === '') {
20.         email.classList.add('invalido');
21.         esValido = false;
22.     } else {
23.         email.classList.remove('invalido');
24.         email.classList.add('valido');
25.     }
26.     if (contrasena.value.trim() === '') {
27.         contrasena.classList.add('invalido');
28.         esValido = false;
29.     } else {
30.         contrasena.classList.remove('invalido');
31.         contrasena.classList.add('valido');
32.     }
```

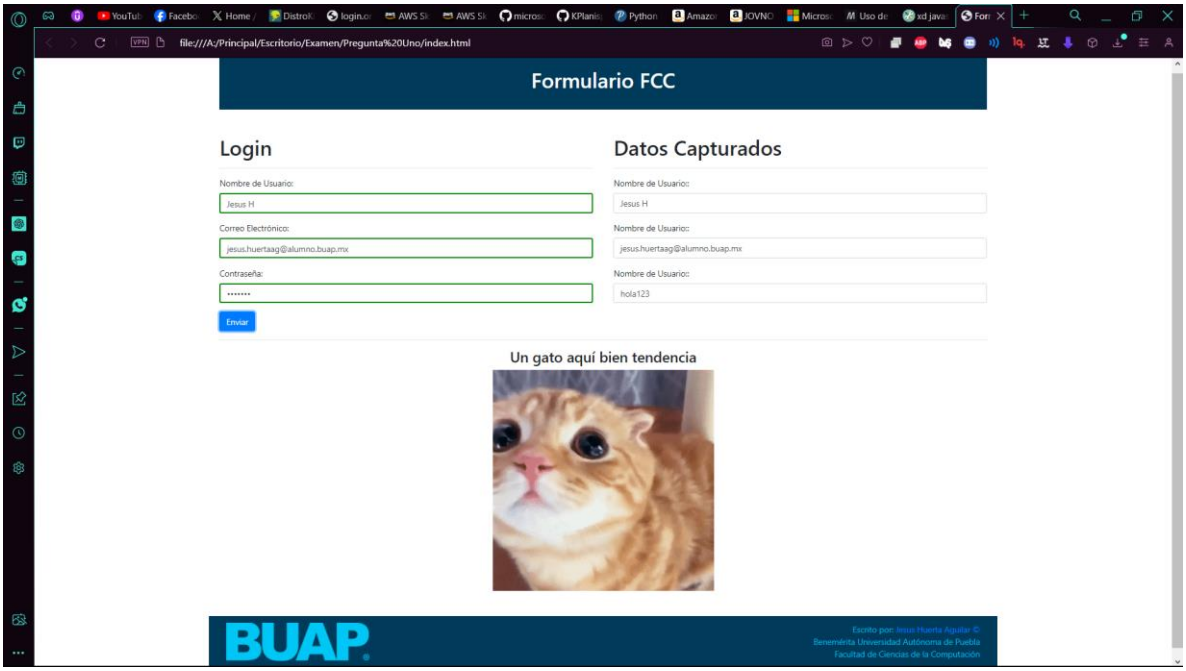
Mostrar resultados si el formulario es válido: Si todos los campos son válidos (**esValido** es **true**), el script muestra los valores ingresados en la sección de resultados de la página, asignando los valores de los campos a los elementos con **id resUsuario**, **resEmail**, y **resContrasena**. Además, se hace visible la sección de resultados retirando la clase **oculto**. Si el formulario no es válido, se mantiene oculta la sección de resultados.

```
36.     if (esValido) {
37.         document.getElementById('resUsuario').textContent = usuario.value;
38.         document.getElementById('resEmail').textContent = email.value;
39.         document.getElementById('resContrasena').textContent = contrasena.va
    lue;
40.         document.getElementById('resultados').classList.remove('oculto');
41.     } else {
42.         document.getElementById('resultados').classList.add('oculto');
43.     }
```

EJECUCIÓN

Al cargar el sitio web se muestra claramente que cumple con los requisitos solicitados.

Con todos los elementos correctos:



Con elementos faltantes: