Laboratorio: 3
Asignatura: ICI 4240
Tema: Validación de formularios (HTML5, SASS, JS, Bootstrap, y jQuery)
Caso de estudio: Formulario de registro

1- Crear un directorio llamado laboratorio2, en el directorio vamos a crear cuatro sub-directorios: images, sass, js y css.
2- Abrir el IDE o editor de código llamado Visual Studio Code (VSC). Luego, creamos un archivo llamado "index.html".
3- Instalar en VSC las siguientes extensiones: Javascript Snippets, jQuery Code Snippets y SASS Snippets.
4- Editar el archivo index.html dando el siguiente comando "html5-boilerplate", debe generar la estructura que debe contener una página HTML5.

```
<!DOCTYPE html>
<!--[if lt IE 7]>      <html class="no-js lt-ie9 lt-ie8 lt-ie7"> <![endif]-->
<!--[if IE 7]>         <html class="no-js lt-ie9 lt-ie8"> <![endif]-->
<!--[if IE 8]>         <html class="no-js lt-ie9"> <![endif]-->
<!--[if gt IE 8]>      <html class="no-js"> <!--<![endif]-->
<html>
    <head>
        <meta charset="utf-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <title></title>
        <meta name="description" content="">
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <link rel="stylesheet" href="">
    </head>
    <body>
        <!--[if lt IE 7]>
            <p class="browsehappy">You are using an <strong>outdated</strong> browser. Please <a href="#">upgrade your browser</a> t
        <![endif]-->

        <script src="" async defer></script>
    </body>
</html>
```

Figura 1. Usando html5-boilerplate para generar estructura HTML.

5- En el archivo index.html adicionar los archivos que se requieren para usar el Framework Bootstrap. Se trabajará con la versión 5.3 de Bootstrap. Se incluirá las fuentes JS y CSS.

CSS	https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/css/bootstrap.min.css
JS	https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/js/bootstrap.bundle.min.js

Deben adicionar dos atributos en JS como defer y crossorigin="anonymous"
Responda las siguientes preguntas:
- ¿Qué es un CDN?
- ¿Qué significa la extensión min.js y min.css?
- ¿ Cuál es la diferencia entre los atributos async y defer?

6- Vamos a adicionar las etiquetas para crear un formulario. Haciendo uso de los componentes de Bootstrap tipo Form en la subcategoría Validation:

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

```
        <form class="row g-3">
          <div class="col-md-4">
            <label for="validationServer01" class="form-label">First name</label>
            <input type="text" class="form-control is-valid" id="validationServer01"
value="Mark" required>
            <div class="valid-feedback">
              Looks good!
            </div>
          </div>
          <div class="col-md-4">
            <label for="validationServer02" class="form-label">Last name</label>
            <input type="text" class="form-control is-valid" id="validationServer02"
value="Otto" required>
            <div class="valid-feedback">
              Looks good!
            </div>
          </div>
          <div class="col-md-4">
            <label for="validationServerUsername" class="form-label">Username</label>
            <div class="input-group has-validation">
              <span class="input-group-text" id="inputGroupPrepend3">@</span>
              <input type="text" class="form-control is-invalid"
id="validationServerUsername" aria-describedby="inputGroupPrepend3
validationServerUsernameFeedback" required>
              <div id="validationServerUsernameFeedback" class="invalid-feedback">
                Please choose a username.
              </div>
            </div>
          </div>
          <div class="col-md-6">
            <label for="validationServer03" class="form-label">City</label>
```

```
            <input type="text" class="form-control is-invalid" id="validationServer03" aria-
describedby="validationServer03Feedback" required>
            <div id="validationServer03Feedback" class="invalid-feedback">
              Please provide a valid city.
            </div>
          </div>
          <div class="col-md-3">
            <label for="validationServer04" class="form-label">State</label>
            <select class="form-select is-invalid" id="validationServer04" aria-
describedby="validationServer04Feedback" required>
              <option selected disabled value="">Choose...</option>
              <option>...</option>
            </select>
            <div id="validationServer04Feedback" class="invalid-feedback">
              Please select a valid state.
            </div>
          </div>
          <div class="col-md-3">
            <label for="validationServer05" class="form-label">Zip</label>
            <input type="text" class="form-control is-invalid" id="validationServer05" aria-
describedby="validationServer05Feedback" required>
            <div id="validationServer05Feedback" class="invalid-feedback">
              Please provide a valid zip.
            </div>
          </div>
          <div class="col-12">
            <div class="form-check">
              <input class="form-check-input is-invalid" type="checkbox" value=""
id="invalidCheck3" aria-describedby="invalidCheck3Feedback" required>
              <label class="form-check-label" for="invalidCheck3">
                Agree to terms and conditions
              </label>
```

```
            <div id="invalidCheck3Feedback" class="invalid-feedback">
              You must agree before submitting.
            </div>
          </div>
        </div>
        <div class="col-12">
          <button class="btn btn-primary" type="submit">Submit form</button>
        </div>
    </form>
```

Probar y analizar cada una de las etiquetas y class que contiene.

Por ejemplo:
Cada etiqueta input tiene un class llamada “form-control” o “form-select” o “form-check-input”, pero a su vez viene acompañada de otra clase llamada “is-valid” o “is-invalid”, esa clase hace que los input aparezca con un borde de color rojo (is-invalid) o verde (is-valid). Esa clases hacen activar o visualizar los mensajes que se encuentran en una etiqueta div con la class=”valid-feedback” o class=”invalid-feedback”.

Ahora, vamos ajustar las etiquetas de tal manera que queden con los siguiente campos en idioma español: nombres, apellidos, correo electrónico, Comuna, Región, , contraseña y el validar que debe aceptar términos y condiciones.

Usaremos solo los mensajes tipo “invalid-feedback”. A su vez se cambiaran los atributos id y for. Por lo tanto las etiquetas quedan:

```
<div class="container">
        <form class="row g-3 needs-validation" novalidate>
          <ul id="campos">
           <li>
          <div class="col-md-4">
            <label for="nombre" class="form-label">Nombres</label>
            <input type="text" class="form-control" id="nombre" required>
```

```
          <div class="invalid-feedback">
            El campo nombre es requerido
          </div>
        </div>
        </li>
        <li>
        <div class="col-md-4">
          <label for="apellidos" class="form-label">Apellidos</label>
          <input type="text" class="form-control" id="apellidos" required>
          <div class="invalid-feedback">
            El campo apellidos es requerido
          </div>
        </div>
      </li>
      <li>
        <div class="col-md-4">
          <label for="email" class="form-label">Correo Electrónico</label>
          <div class="input-group has-validation">
            <input type="email" class="form-control" id="email"  required>
            <div class="invalid-feedback">
              El correo electrónico es requerido
            </div>
          </div>
        </div>
      </li>
      <li>
        <div class="col-md-4">
          <label for="contrasenya" class="form-label">Contraseña</label>
          <div class="input-group has-validation">
            <span class="input-group-text" id="inputGroupPrepend3"></span>
            <input type="password" class="form-control" id="contrasenya" required>
            <div  class="invalid-feedback">
```

```
            La contraseña es requerida
          </div>
        </div>
      </div>
    </li>
    <li>
      <div class="col-md-3">
        <label for="region" class="form-label">Región</label>
        <select class="form-select" id="region" required>
          <option selected disabled value="">Seleccione una opción</option>
          <option>Santiago</option>
          <option>Atacama</option>
          <option>Coquimbo</option>
          <option>Valparaíso</option>
        </select>
        <div class="invalid-feedback">
         El campo región es requerido
        </div>
      </div>
    </li>
    <li>
      <div class="col-md-6">
        <label for="comuna" class="form-label">Comuna</label>
        <input type="text" class="form-control" id="comuna" required>
        <div class="invalid-feedback">
          La comuna es requerida
        </div>
      </div>
    </li>
     <li>
      <div class="col-12">
        <div class="form-check">
```

```
                <input class="form-check-input" type="checkbox" value=""
id="invalidCheck3" required>
                <label class="form-check-label" for="invalidCheck3">
                  Acepta términos y condiciones
                </label>
                <div id="invalidCheck3Feedback" class="invalid-feedback">
                  Debe validar las condiciones
                </div>
              </div>
            </div>
          </li>
          <li>
            <div class="col-12">
              <button class="btn btn-primary" type="submit">Registrarse</button>
            </div>
          </li>
          </ul>
        </form>

    </div>
```

7- Vamos adicionar el atributo novalidate en la etiqueta form. A su vez adicionar una clase adicionar en la etiqueta form llamada “needs-validation”

8- Vamos a crear un archivo js llamado app.js con el siguiente código:

```
(() => {
    'use strict'

    // Fetch all the forms we want to apply custom Bootstrap validation styles to
    const forms = document.querySelectorAll('.needs-validation')
```

```
  // Loop over them and prevent submission
  Array.from(forms).forEach(form => {
    form.addEventListener('submit', event => {
      if (!form.checkValidity()) {
        event.preventDefault()
        event.stopPropagation()
      }

      form.classList.add('was-validated')
    }, false)
  })
})()
```

Y probamos, al dar clic al botón registrarse debe mostrarnos la siguiente imagen:

Responda las siguientes preguntas
¿Qué significa “use strict”?

9- Ahora vamos a validar el formato de la contraseña. Para esto es necesario considerar un patrón del texto, el cual se puede tomar de la siguiente url: https://www.html5pattern.com/Passwords

Agregamos el patrón adicionando el atributo pattern=”” y valor del patrón.

10- Modificamos el script en JS para validar el formato del patrón. Por lo que se debe crear una función llamada checkPattern, con el siguiente código:

```
function checkPattern(){
    var elem = document.getElementById("contrasenya");
    var pattern = elem.getAttribute("^(?=.*\d)(?=.*[a-z])(?=.*[A-Z])(?!.*\s).*$");
    var re = new RegExp(pattern);
    if (re.test(elem.value)) {
       return true;
    } else {
       return false;
    }
  }
```

Responda la siguiente pregunta

¿ Para qué sirve RegExp? ¿Qué es, un método o clase?

11- Modificamos el siguiente código en la validación :

```
(() => {
    'use strict'
    // Fetch all the forms we want to apply custom Bootstrap
validation styles to
    const forms = document.querySelectorAll('.needs-
validation');
    let password = document.getElementById("contrasenya");

    const campos= document.getElementById("campos");

    // Loop over them and prevent submission
    Array.from(forms).forEach(form => {
      form.addEventListener('submit', event => {
        if (!form.checkValidity()) {
```

```
            if(!checkPattern() && password.value!==""){
              campos.children[3].getElementsByClassName("invalid-
    feedback")[0].innerHTML = "Contraseña no valida";
            }

            event.preventDefault()
            event.stopPropagation()
          }

      form.classList.add('was-validated')
        }, false)
      })
    })()
```

Y probamos ¡!!

Responda la siguiente pregunta
¿Cuál sería la instrucción para imprimir las variables en consola?

12- Ahora vamos a trabajar con pre-procesador SASS. Vamos a crear un archivo llamado styles con extensión .scss en el directorio sass. Adicionamos el siguiente estilo:

```
ul{
    list-style: none;
}
```

```
$font-stack: Helvetica, sans-serif;
$primary-color: #333;

body {
  font: 100% $font-stack;
  color: $primary-color;
}
```

Ahora, vamos a generar los css:
Ya instalado la extensión sass, en la parte inferior buscamos el icono que dice "Watch Sass" y damos click, se originarán archivos en el directorio css.

No olvidar incluir el css en el archivo index.html.

Ahora vamos a trabajar jQuery, el cual es una librería de Javascript:

Debemos ir la página oficial de jQuery: jquery.com y vamos a crear otro JS que tenga el nombre appJquery.js

Para validar formularios se hace uso de una librería llamada jqueryvalidation. (https://jqueryvalidation.org/documentation/)

El link de la librería es: https://cdnjs.cloudflare.com/ajax/libs/jquery-validate/1.19.0/jquery.validate.min.js, esta librería requerida de jQuery, ya que es una extensión de la librería jQuery.

Por lo tanto queda los cuatro JS de la siguiente manera:

```
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/js/bootstrap.bundle.min.js" defer crossorigin="anonymous"></script>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.4/jquery.min.js" crossorigin="anonymous"></script>
<script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/jquery-validate/1.19.0/jquery.validate.min.js" defer></script>
<script src="js/appJquery.js" defer></script>
```

También es necesario adicionar el atributo **name** para cada etiqueta <input>.

Por ejemplo:

```
<div class="col-md-4">
    <label for="nombre" class="form-label">Nombres</label>
    <input type="text" class="form-control" id="nombre" name="nombre" required>
    <div class="invalid-feedback"> El campo nombre es requerido </div>
</div>
```

A su vez se debe eliminar las etiquetas

```
<div class="invalid-feedback"> </div>
```

Adicionamos el siguiente código en el js llamado appJquery.js

```
$(document).ready(function() {
    $( ".needs-validation" ).validate( {
        errorClass: "is-invalid",
        validClass: "is-valid",
        rules: {
                nombre:{
                    required: true,
                },
                apellidos:{
                    required: true,
                },
                email: {
                    required: true,
                },
                contrasenya:{
                    required: true,
                    pwcheck: true,
                    minlength: 8
```

```
        },
        comuna:{
            required:true
        },
        region:{
            required: true
        }
    },
    messages:{
      nombre: {
        required: "el nombre es requerido"
      },
      apellidos:{
        required: "apellidos es requerido"
      },
      email:{
        required: "el correo es requerido",
        email: "el formato no es el correcto"
      },
      contrasenya:{
        required: "la contraseña es requerida",
        pwcheck: "la contraseña no tiene un formato válido",
        minlength: "debe contener 8 caracteres"
      },
      region:{
        required:"la región es requerida"
      },
      comuna:{
        required: "la comuna es requerida"
      },
      terminos:{
        required: "es requerido aceptar términos y condiciones"
```

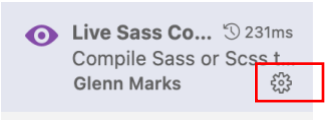
```
        }


      }
    });
    $.validator.addMethod("pwcheck",
                    function(value, element) {
                       return /^(?=.*\d)(?=.*[a-z])(?=.*[A-Z])(?!.*\s).*$/.test(value);
    });
});
```

Adicionar estilos necesarios para que los mensajes de error aparezcan en color ROJO.

Si desea cambiar la ruta de donde origina los .css el SASS compiler:

1 – ir a la extensión Live Sass Compiler y dar clic sobre el icono de configuración:



2- Clic sobre el icono de configuración e ir a la opción “Configuración de la extensión”.

3- Buscar y seleccionar la opción que dice Live Sass Compile > Settings: Formats, dar clic en la opción que dice “editar en settings.json”

4- Cambiar la opción que dice savepath

```
"liveSassCompile.settings.formats": [
    {
      "format": "expanded",
      "extensionName": ".css",
      "savePath": "/css"
    }
  ]
```