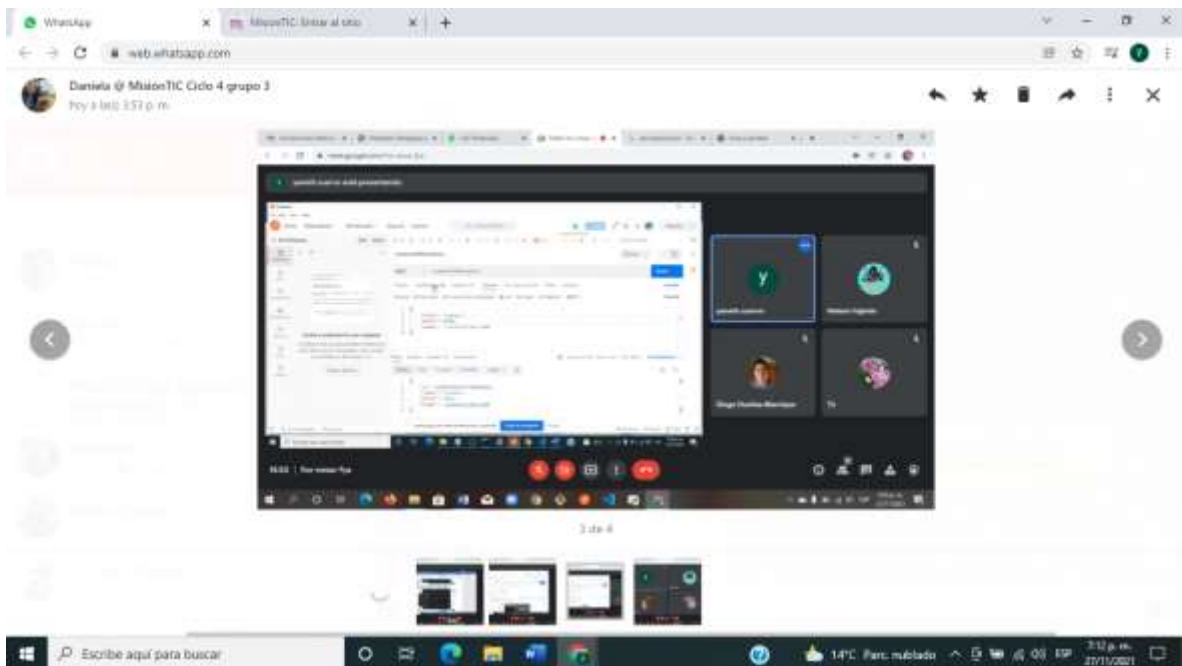


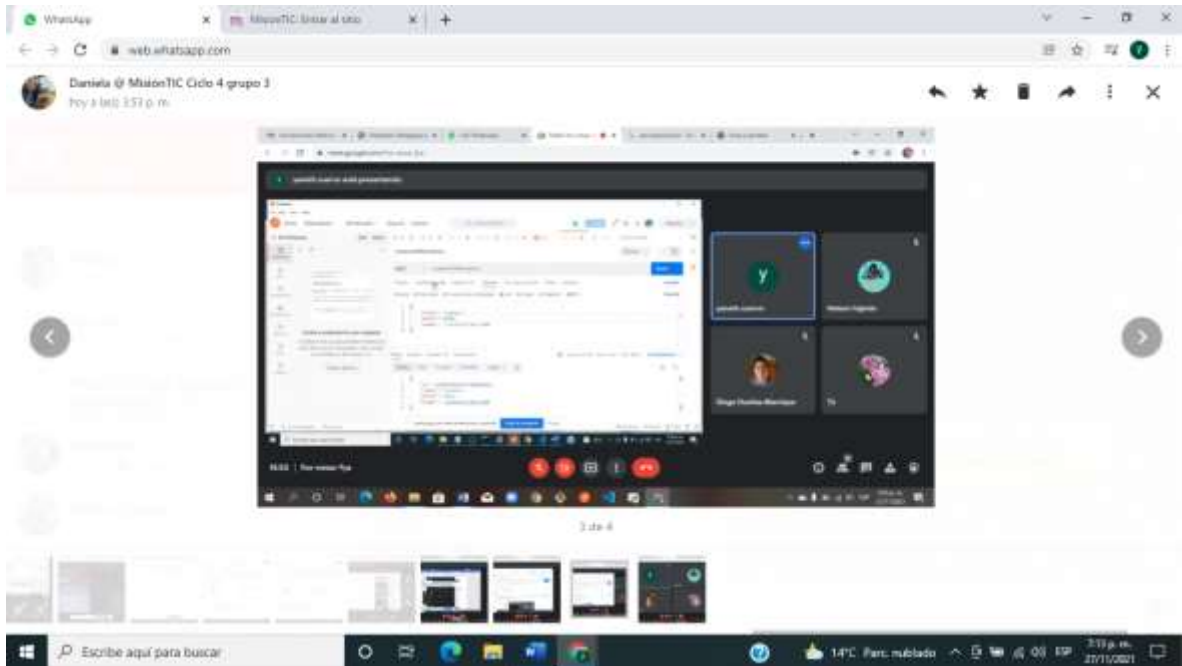
## SPRINT 4. PROYECTO TIENDA VIRTUAL MASCOTA FRONTEND- HTML CSS JAVA SCRIPT

### Integrantes Grupo 12. Equipo 3 LOS TIC DE LOS NODEJS

| Nombre                           | Cedula     | Rol                            | Nivel de Participación |
|----------------------------------|------------|--------------------------------|------------------------|
| CONTRERAS NICOLAS CAMILO         | 1193112644 | Administrador de Configuración | Alto                   |
| DUEÑAS MANRIQUE DIEGO FERNANDO   | 1026283999 | Diseñador UI                   | Alto                   |
| FAJARDO ENRIQUEZ ZULY DANIELA    | 1086135083 | Diseñador de Software          | Alto                   |
| NELSON GERARDO FAJARDO PATARROYO | 80155325   | Lider del Equipo               | Alto                   |
| YANETH MILENA CUERVO BARAHONA    | 40047928   | Tester                         | Alto                   |
|                                  |            |                                |                        |

1. Subir las evidencias de las reuniones diarias (pantallazo de las reuniones)





2. Pantallazo del tablero Kanban con las tareas asignadas y realizadas.

#### TABLERO KANBAN GRUPO 3 CICLO 4 SPRINT 4

### TABLERO KANBAN

Calle Esperanza, 33 San Sebastián  
12557 62278919  
www.tuwebaqui.com

TU  
LOGO

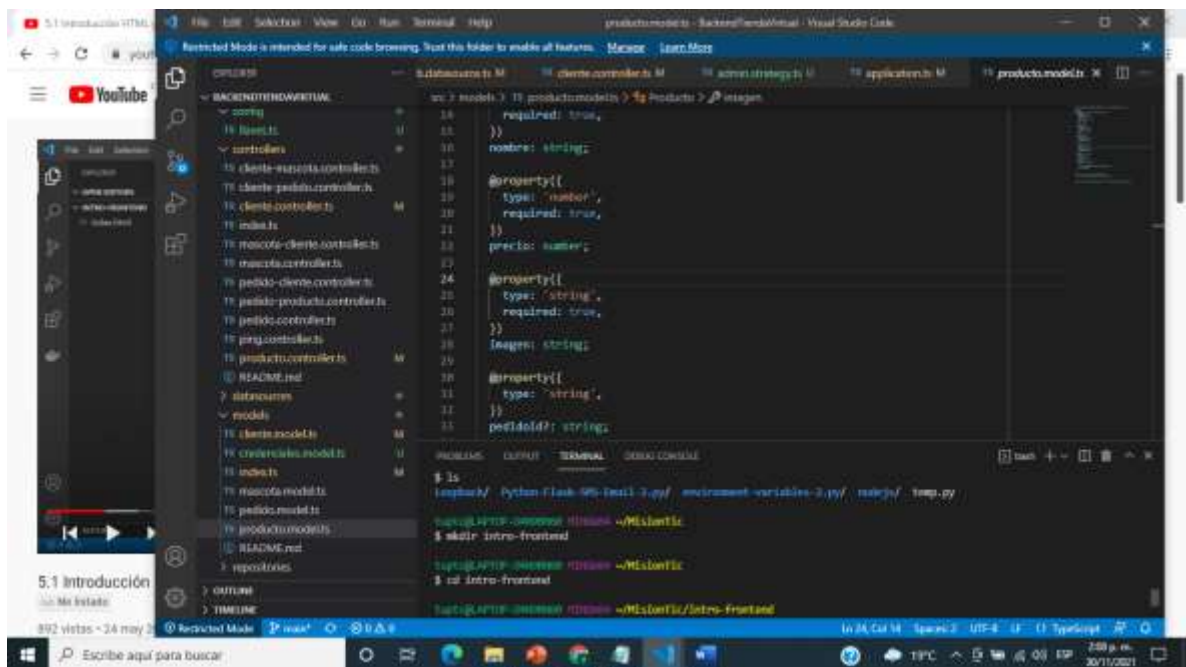


3. Ahora vamos a trabajar en frontend es decir la interfaz grafica de usuario mediante lenguajes como Html Css y Java Script. Con la unión de los tres lograremos algo usable y bonito

Con el Html definimos la diagramación y los elementos con el que el usuario puede interactuar como botones texto entre otros.

Creamos una carpeta donde vamos a colocar los archivos relacionados con html

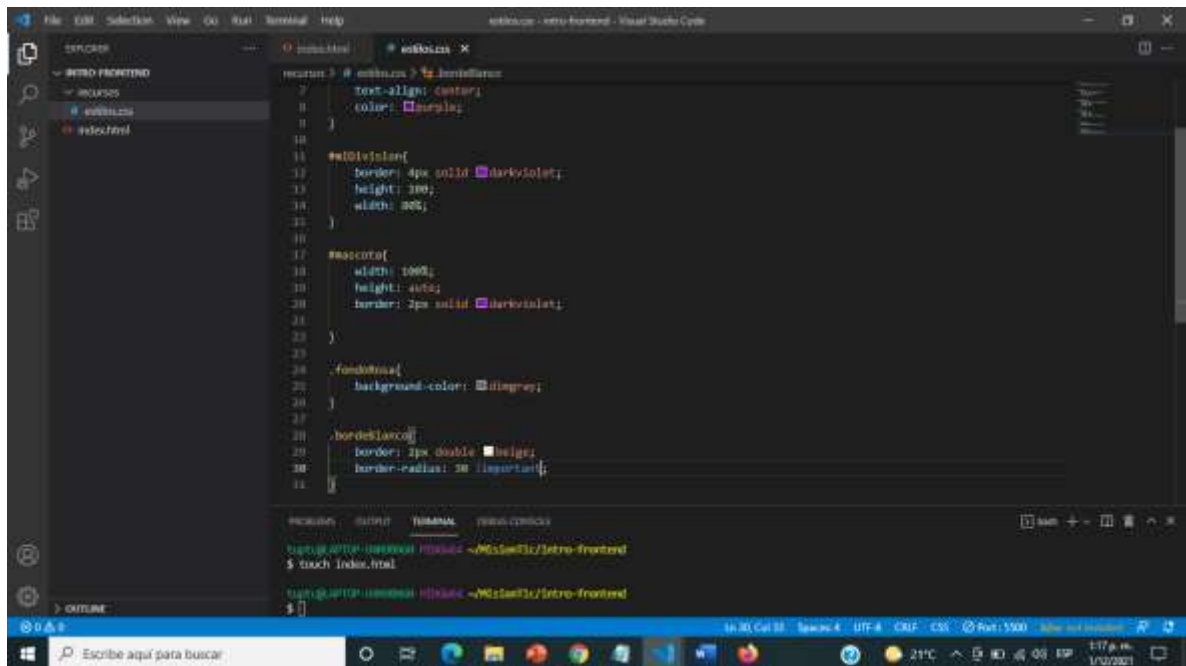
## HTML



Ahora ingresamos a la carpeta y le pedimos generar un archivo llamado **touch index.html** en el cual crearemos las etiquetas

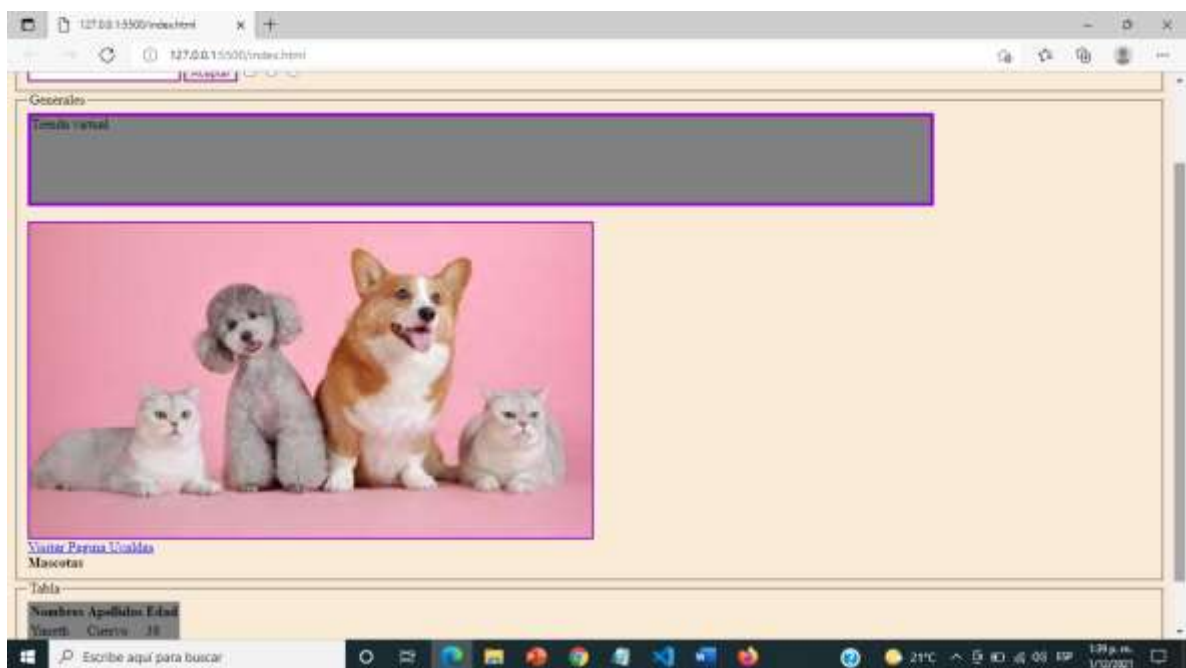






```
1 header {
2   text-align: center;
3   color: white;
4 }
5
6 #div1 {
7   border: 4px solid darkviolet;
8   height: 300px;
9   width: 800px;
10 }
11
12 #mascotas {
13   width: 1000px;
14   height: 400px;
15   border: 2px solid darkviolet;
16 }
17
18 .fondo {
19   background-color: #d3d3d3;
20 }
21
22 .border {
23   border: 2px double #f0e68c;
24   border-radius: 30px;
25 }
```

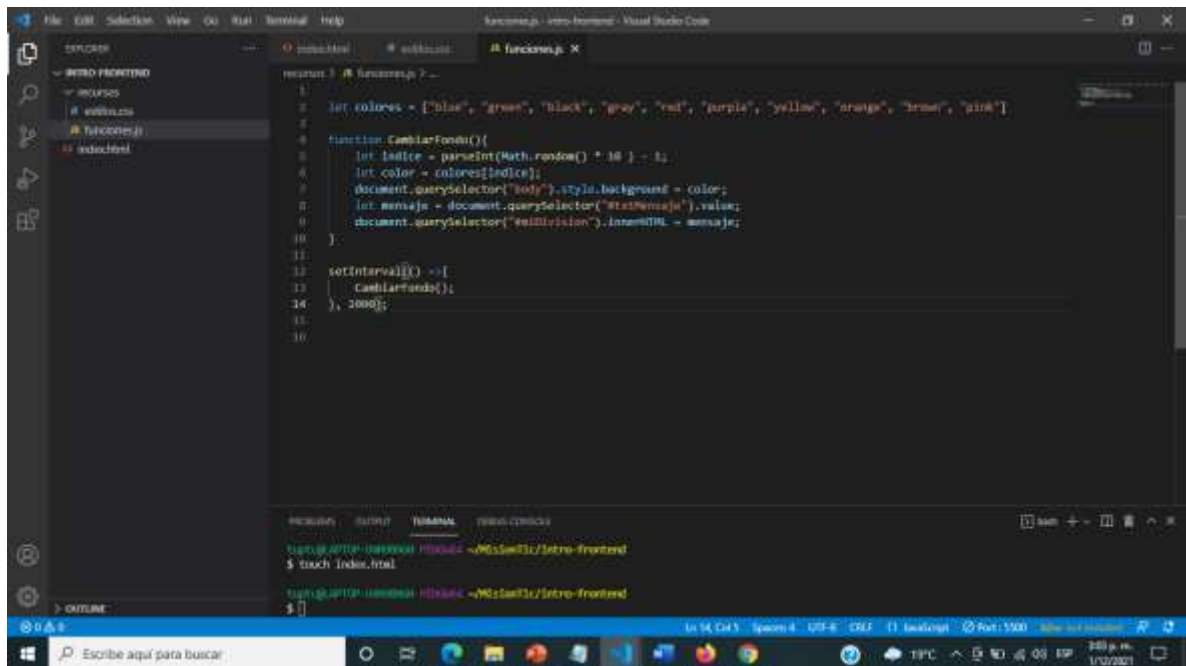
Y como resultado en la aplicación de los estilos en CSS obtenemos



## JAVA SCRIPT

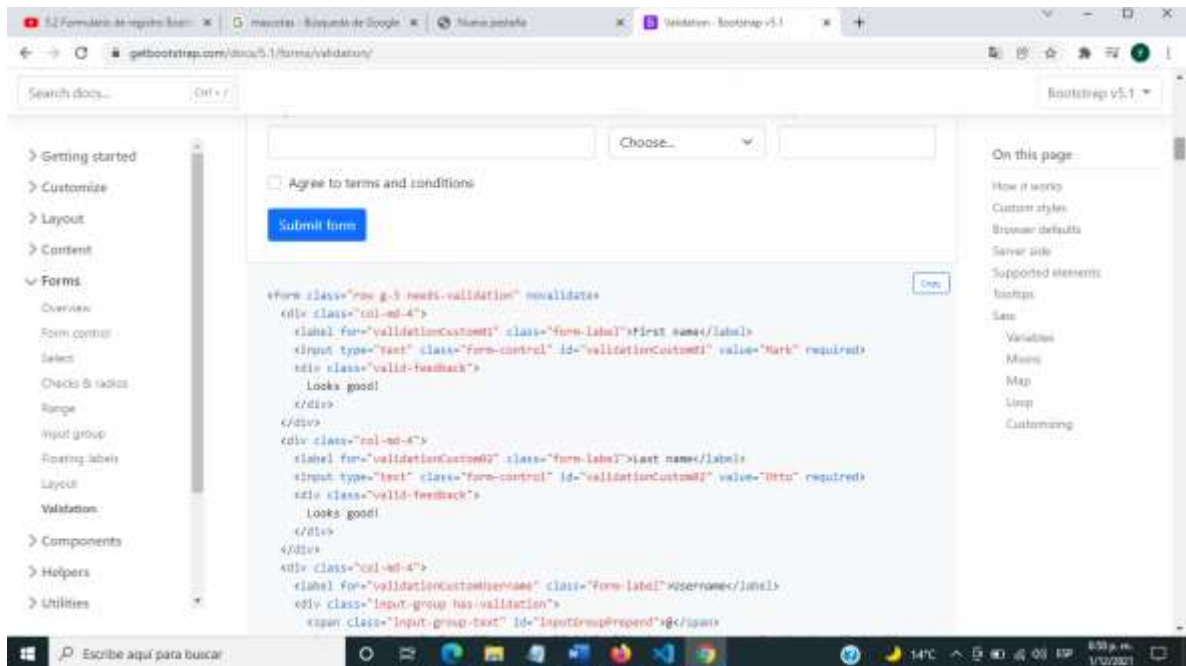
Ahora por medio de JavaScript vamos a ver como obtenemos la información como podemos modificar los diferentes estilos o acciones frente a los objetos ya implementados en la interfaz grafica



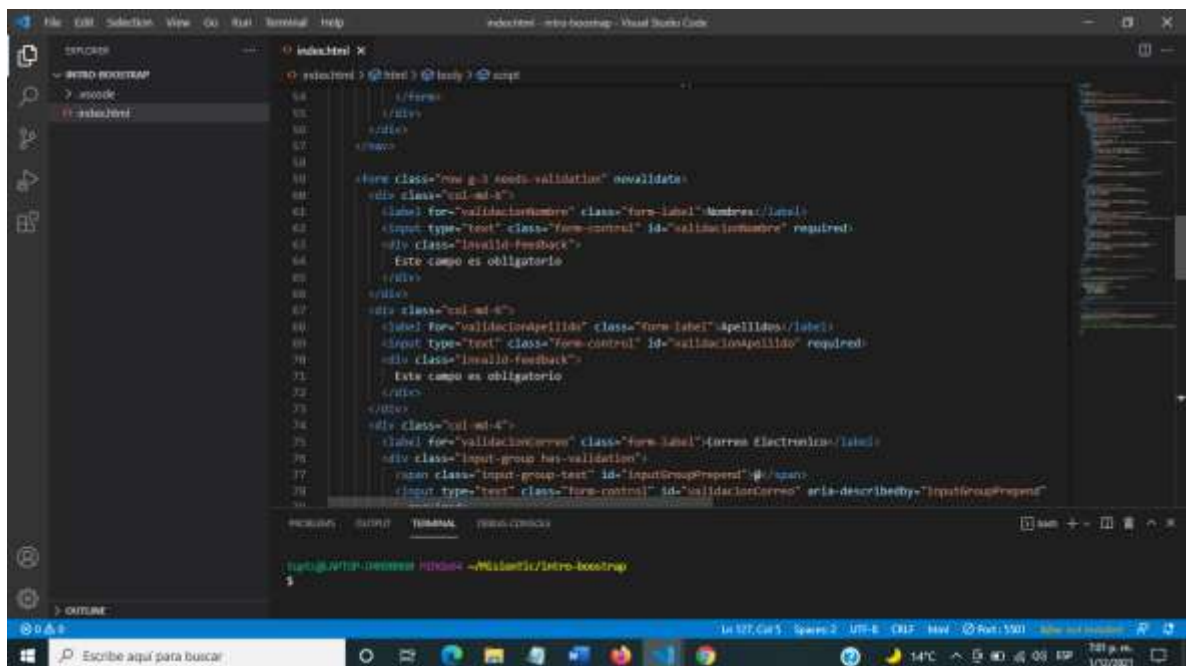


## BOOTSTRAP

Mediante el uso de la herramienta Bootstrap la cual trae una implementación de framework es decir las plantillas que nos servirán para el montaje de una forma fácil de la interfaz grafica. La cual trae consigo el código para la aplicación de diferentes plantillas copiamos ese código y lo pegamos en una capeta que creamos para dicha funxion **intro-bootstrap**

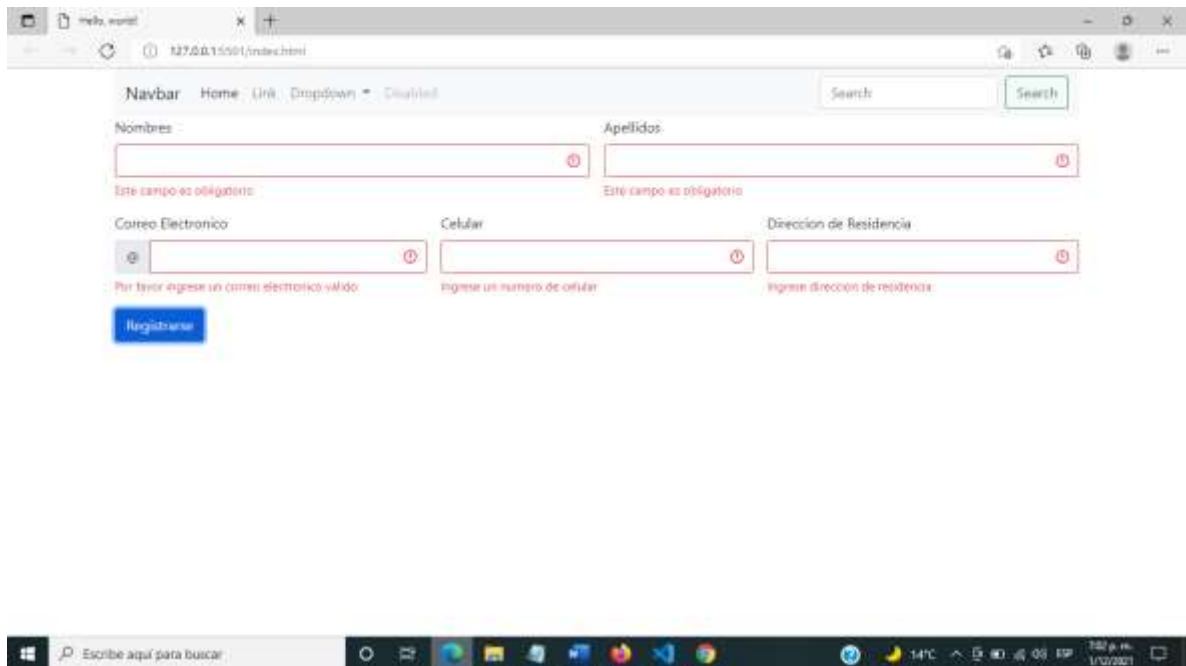


Modificamos el código ejemplo de acuerdo a nuestras necesidades para el caso de nuestra entidad Persona



Validamos el correcto funcionamiento de los cambio realizados

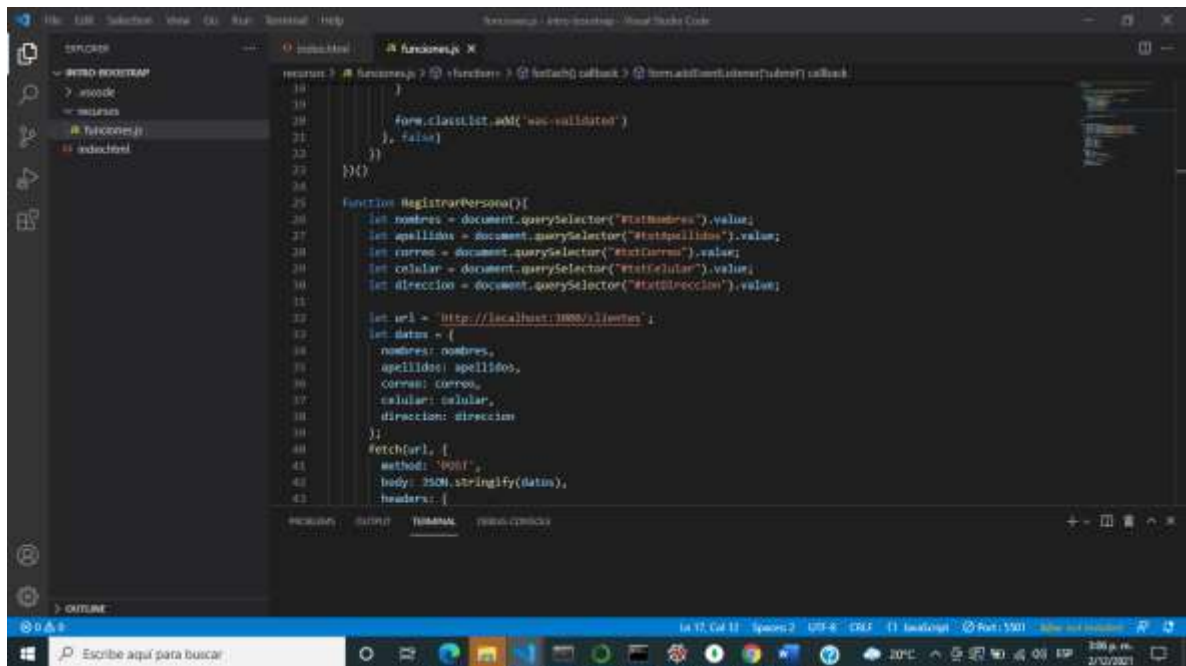




Para validar el funcionamiento de la plataforma mediante el Bootstrap activamos la conexión por medio de Python e igualmente ejecutamos nuestra aplicación en Visual basic para validar su correcto funcionamiento mediante la función RegistrarPersona validamos que se este enviando el mensaje de registro. Donde podemos observar el registro realizado en la base de datos y validamos el envío del mensaje.

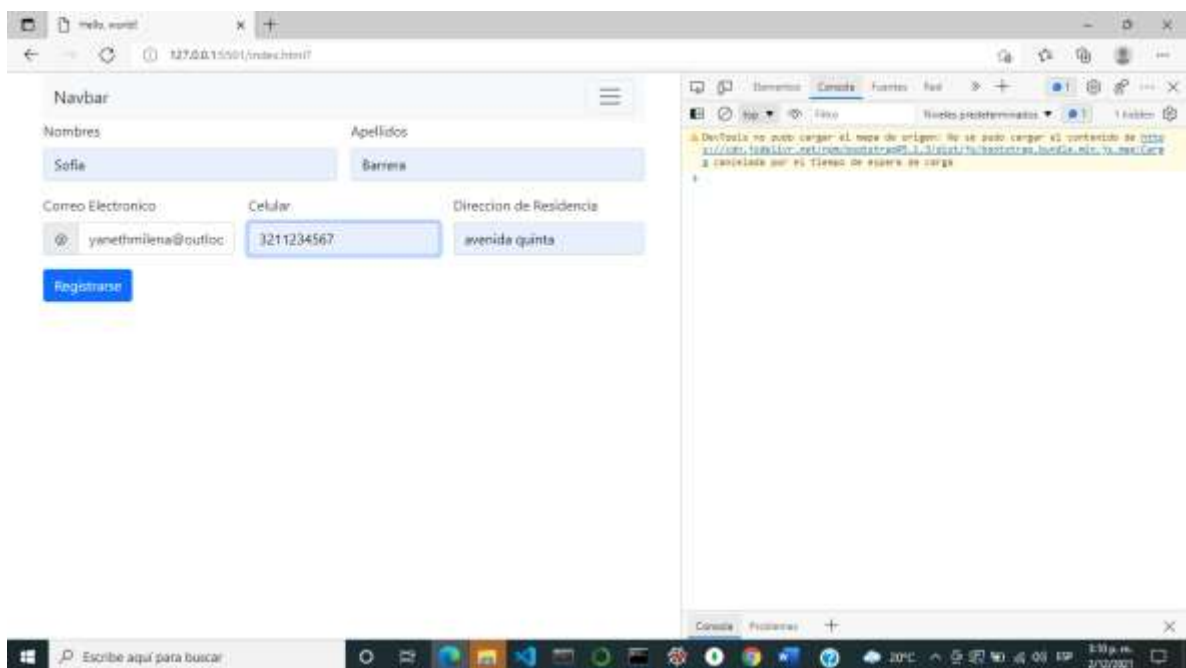
Es la forma de crear un framework CSS para construccion de la interfaz grafica que soporte toda la interacción que tenga las personas con nuestro sistema de informa y como utilizar los desarrollos propios

Imagen de la función.js en donde validamos el registro de la persona tanto en la base de datos como en el envío del correo

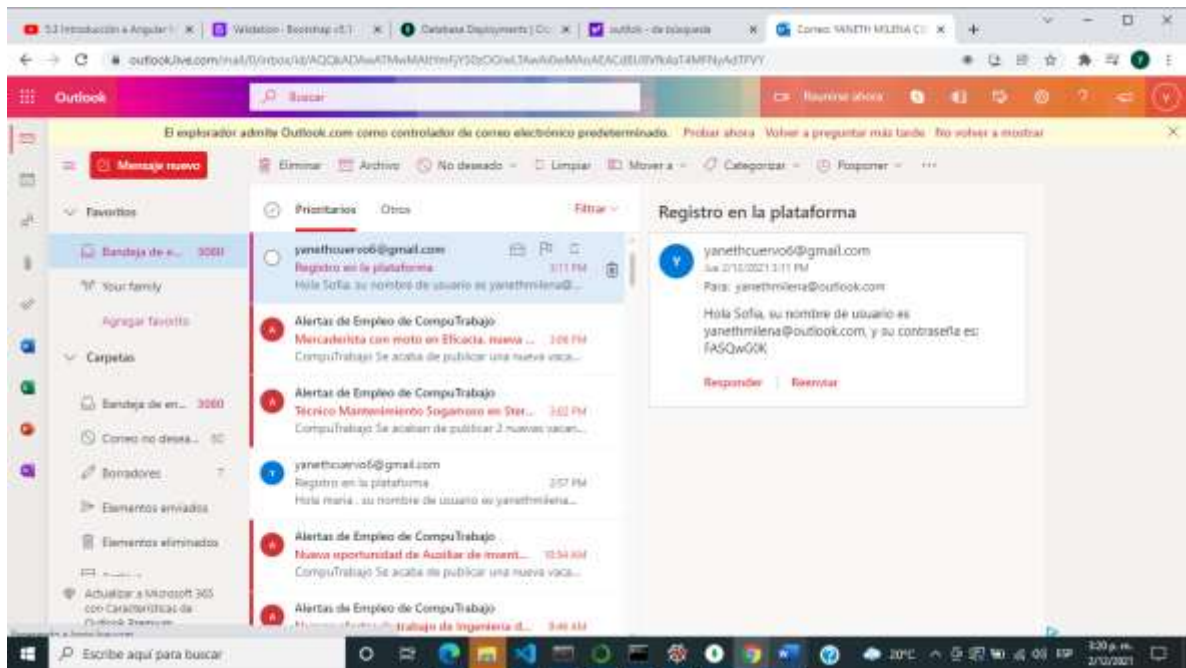


```
18 }
19 }
20
21 form.classList.add('was-validated')
22 }, false)
23 })
24 })
25
26 function RegistrarPersona(){
27   let nombres = document.querySelector("#txtNombres").value;
28   let apellidos = document.querySelector("#txtApellidos").value;
29   let correo = document.querySelector("#txtCorreo").value;
30   let celular = document.querySelector("#txtCelular").value;
31   let direccion = document.querySelector("#txtDireccion").value;
32
33   let url = "http://localhost:3000/api/usuarios";
34   let datos = {
35     nombres: nombres,
36     apellidos: apellidos,
37     correo: correo,
38     celular: celular,
39     direccion: direccion
40   };
41   fetch(url, {
42     method: 'POST',
43     body: JSON.stringify(datos),
44     headers: {
45       'Content-Type': 'application/json'
46     }
47   })
48 }
```

Resultados de la interfaz grafica en el registro de la persona

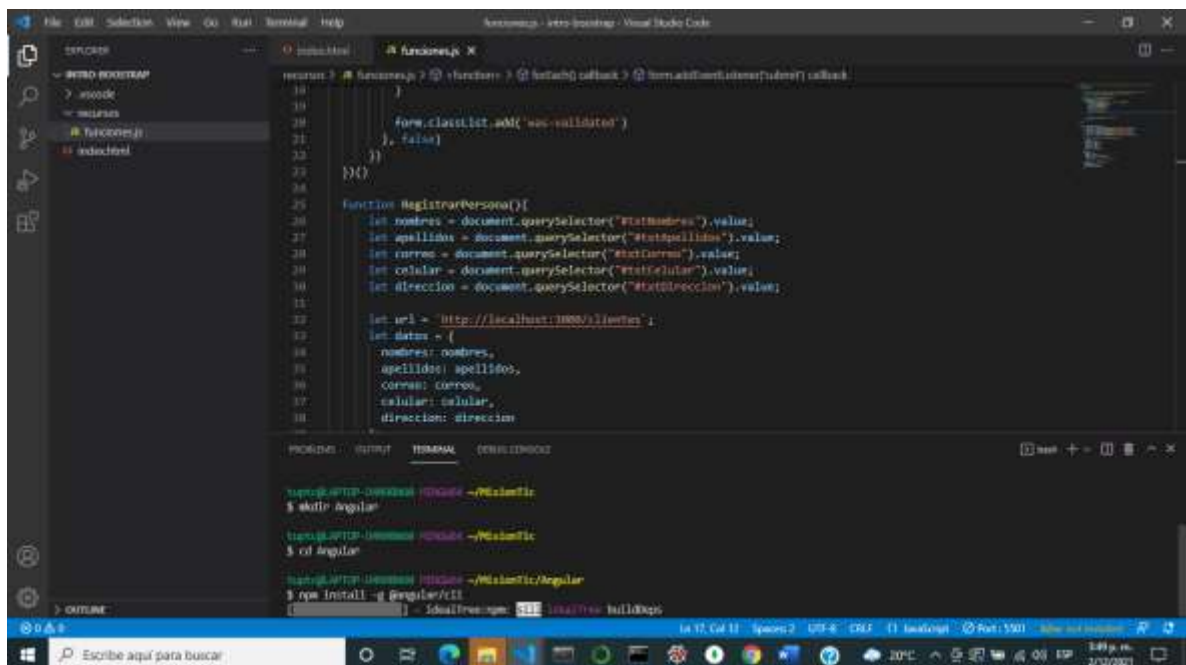


Soporte del mensaje enviado al correo sobre el registro de la persona



## ANGULAR

Instalamos angular desde visual studio code



The image shows a VS Code editor window with a file named 'funciones.js' open. The code is as follows:

```

function RegistrarPersona([
    let nombrs = document.querySelector("#txtNombrs").value;
    let apellidos = document.querySelector("#txtApellidos").value;
    let correo = document.querySelector("#txtCorreo").value;
    let celular = document.querySelector("#txtCelular").value;
    let direccion = document.querySelector("#txtDireccion").value;

    let url = "http://localhost:3000/usuarios";
    let datos = {
        nombrs: nombrs,
        apellidos: apellidos,
        correo: correo,
        celular: celular,
        direccion: direccion
    }
    fetch(url, {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json'
        },
        body: JSON.stringify(datos)
    })
    .then(response => response.json())
    .then(data => console.log(data))
    .catch(error => console.log(error))
    .finally(() => {
        document.querySelector("#btnRegistrar").disabled = true;
    })
}

```

The bottom panel shows the 'TERMINAL' with a list of file sizes for various modules and components:

```

CREATE FrontEnd\src\assets\environment.prod.ts (51 bytes)
CREATE FrontEnd\src\assets\environment.ts (608 bytes)
CREATE FrontEnd\src\assets\app\app-routing.module.ts (285 bytes)
CREATE FrontEnd\src\assets\app\app.module.ts (303 bytes)
CREATE FrontEnd\src\assets\app\app.component.html (23904 bytes)
CREATE FrontEnd\src\assets\app\app.component.spec.ts (1138 bytes)
CREATE FrontEnd\src\assets\app\app.component.ts (225 bytes)
CREATE FrontEnd\src\assets\app\app.component.css (8 bytes)

```

Creamos cada modulo con cada uno de sus componentes

