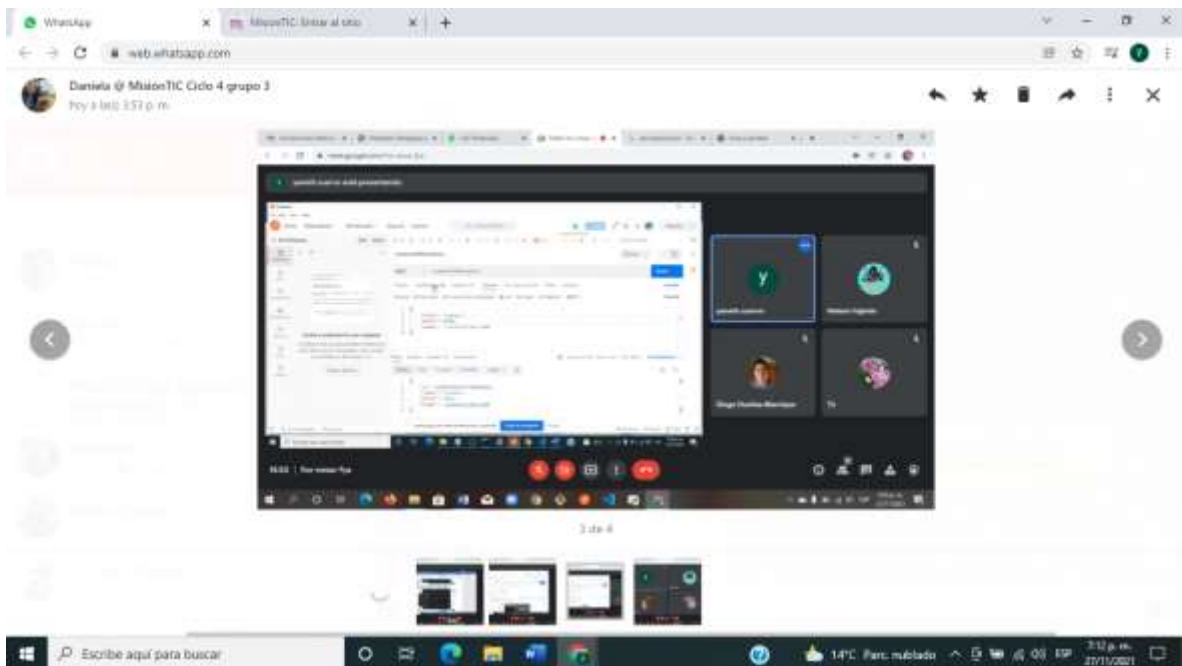


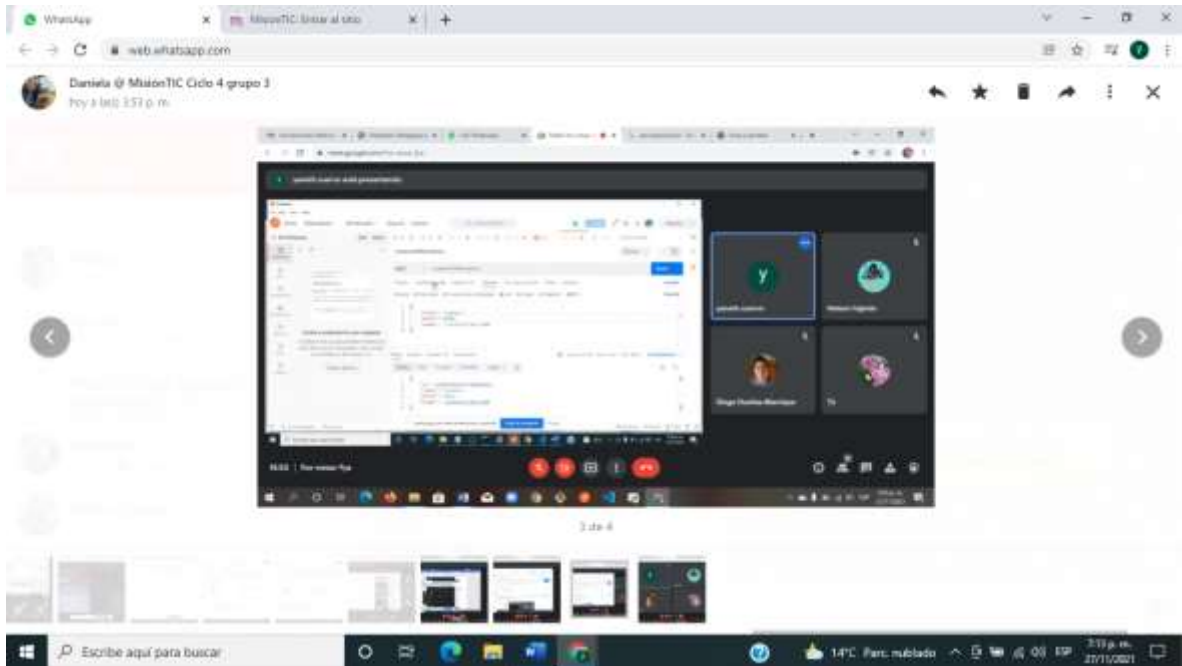
SPRINT 5. PROYECTO TIENDA VIRTUAL MASCOTA FRONTEND- ANGULAR

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

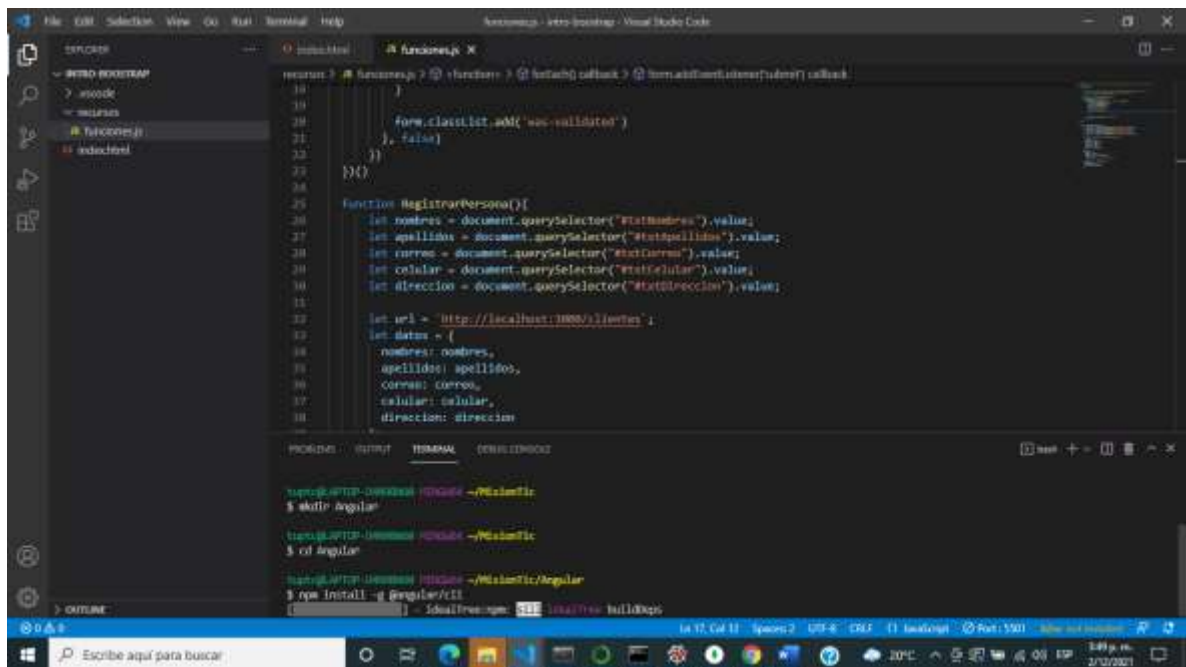
TU
LOGO



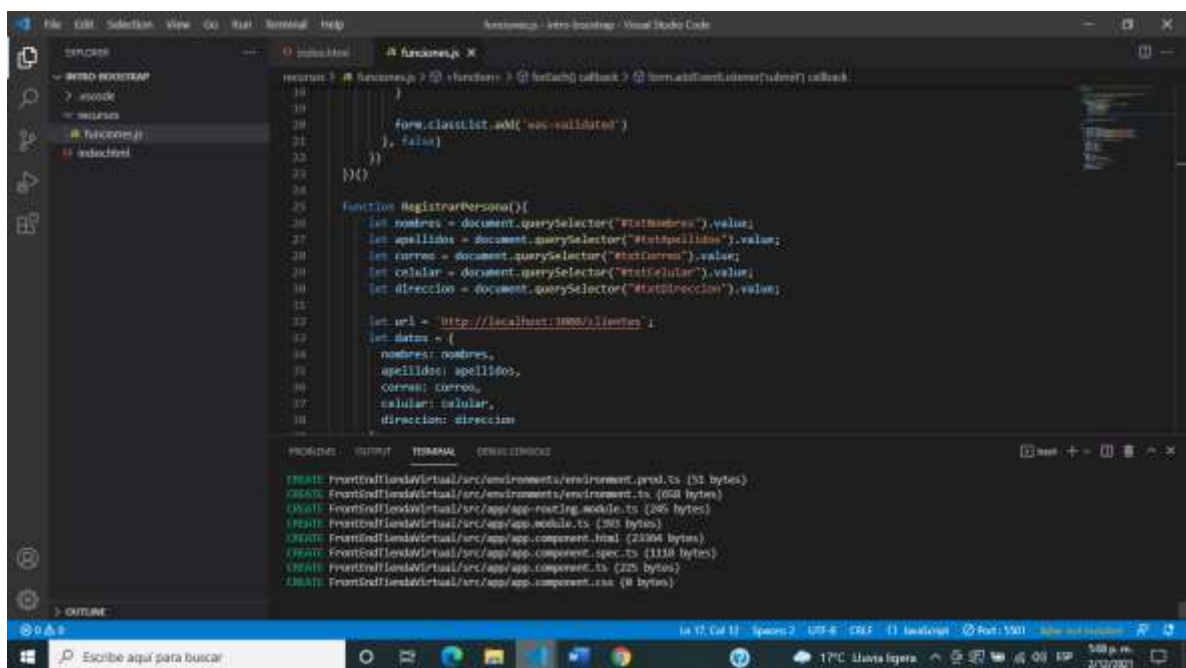
3. Ahora vamos a trabajar en frontend por medio de la herramienta framework Angular

ANGULAR

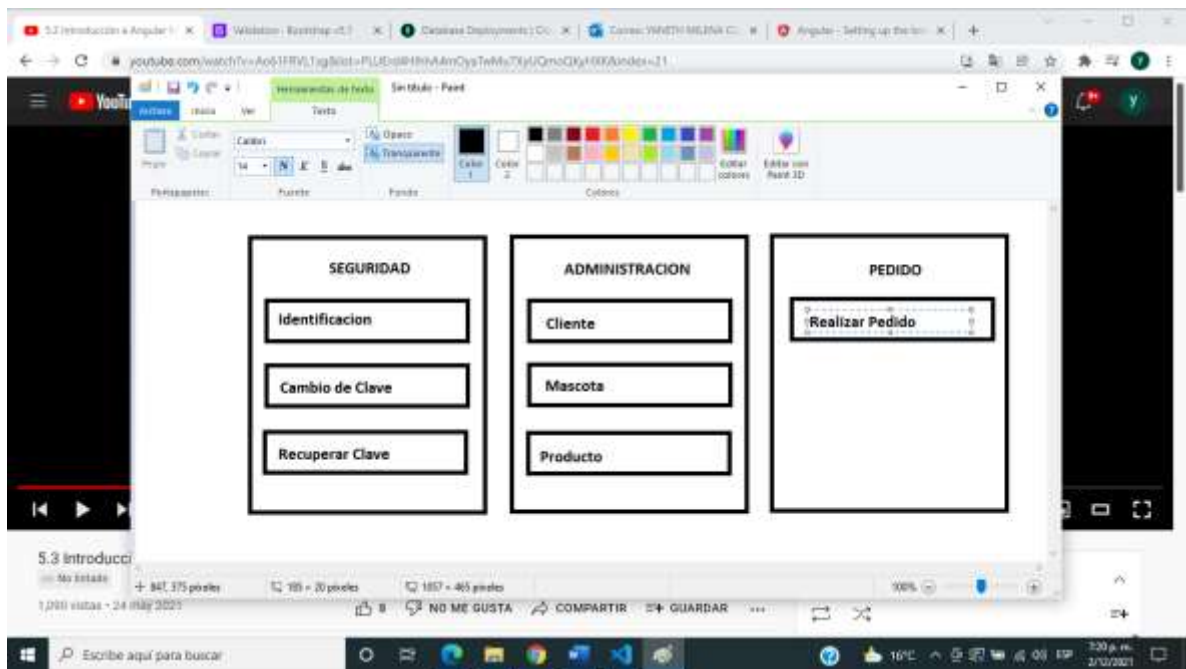
Instalamos angular desde visual studio code



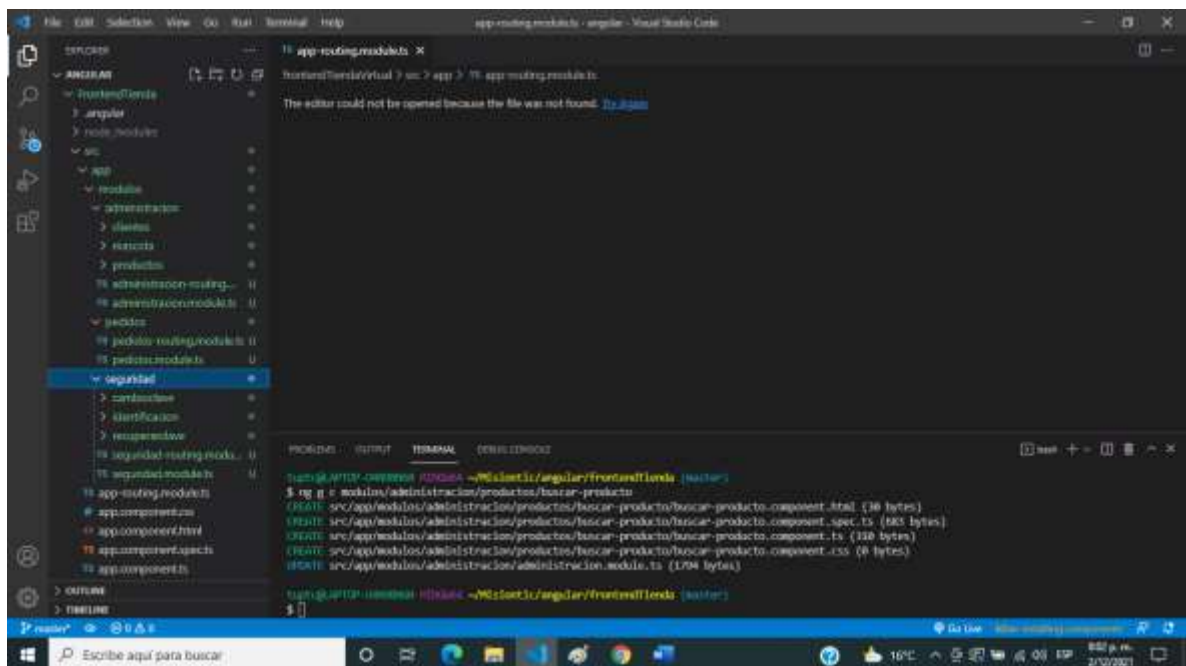
Creamos una carpeta donde vamos a ubicar todas nuestra interfaz grafica en angula y dentro de esta creamos nuestro proyecto del front end tienda virtual



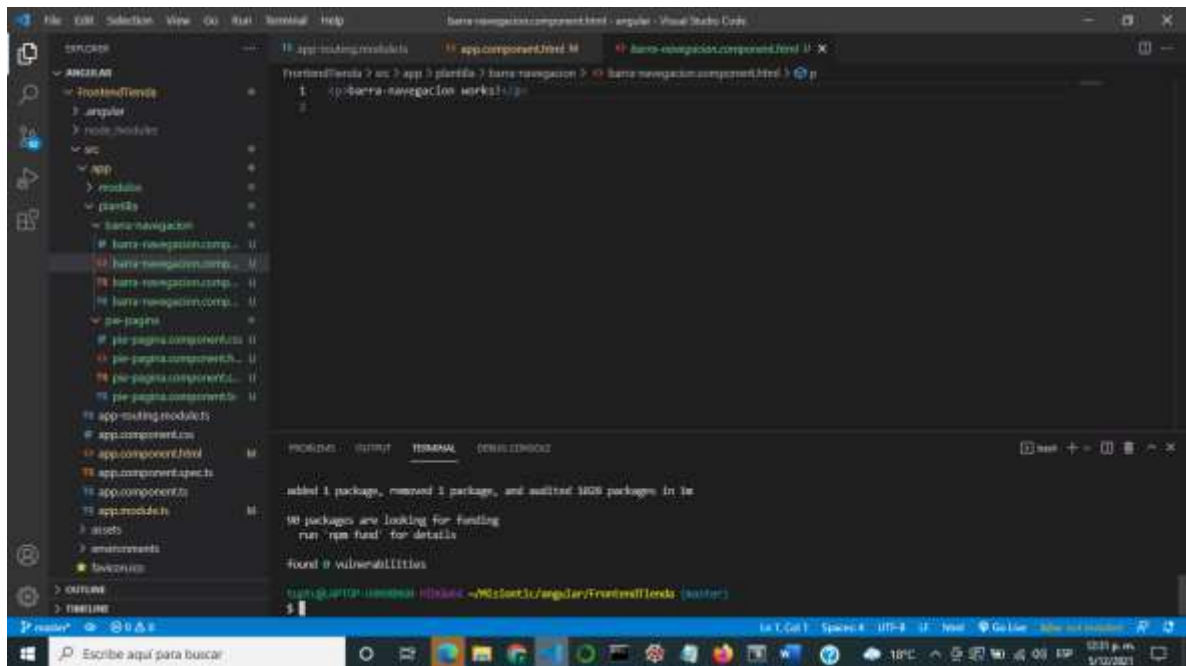
Determinamos los módulos que vamos a implementar



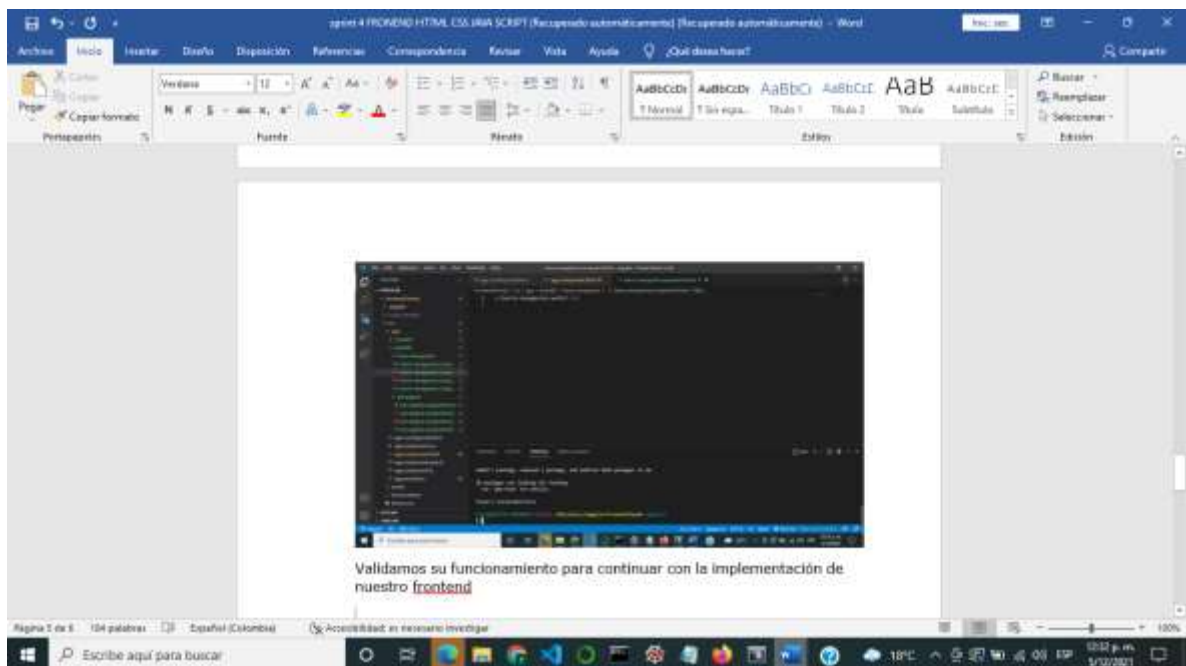
Creamos cada modulo con cada uno de sus componentes



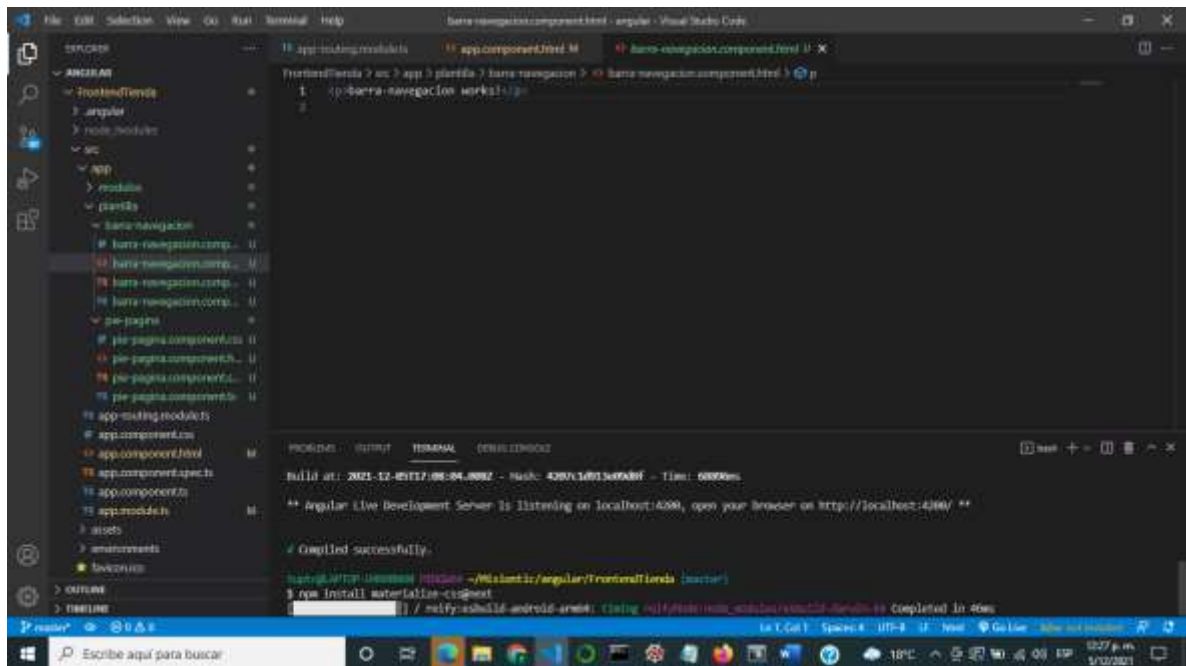
Creamos nuestra barra de navegación



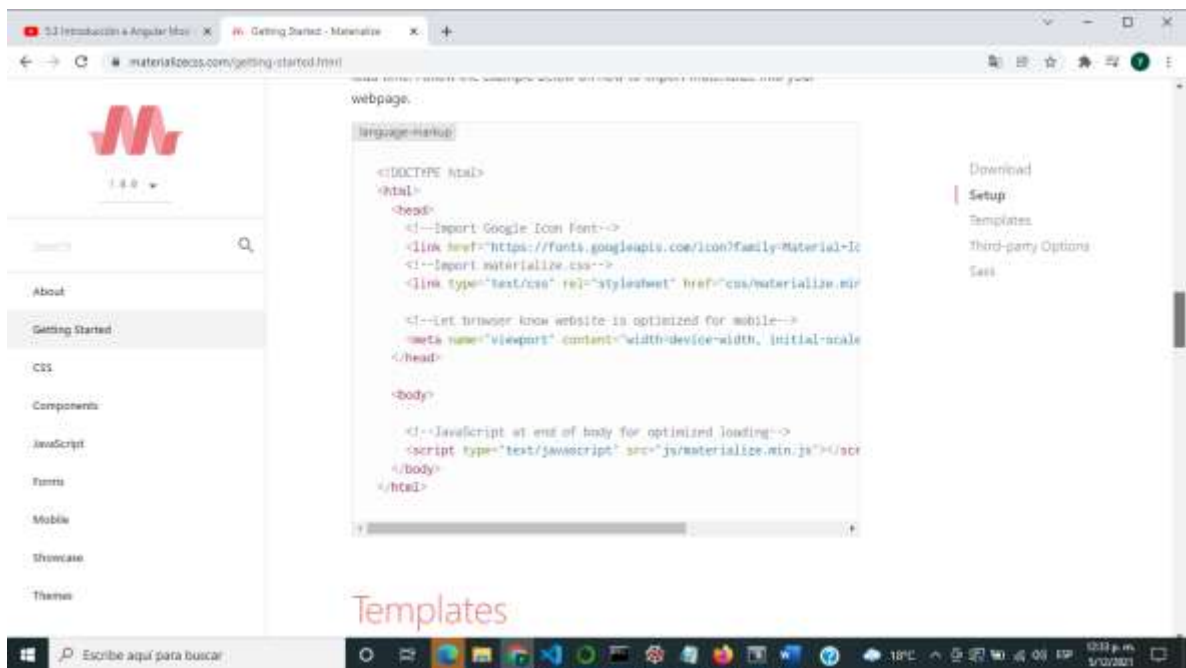
Validamos su funcionamiento para continuar con la implementación de nuestro frontend



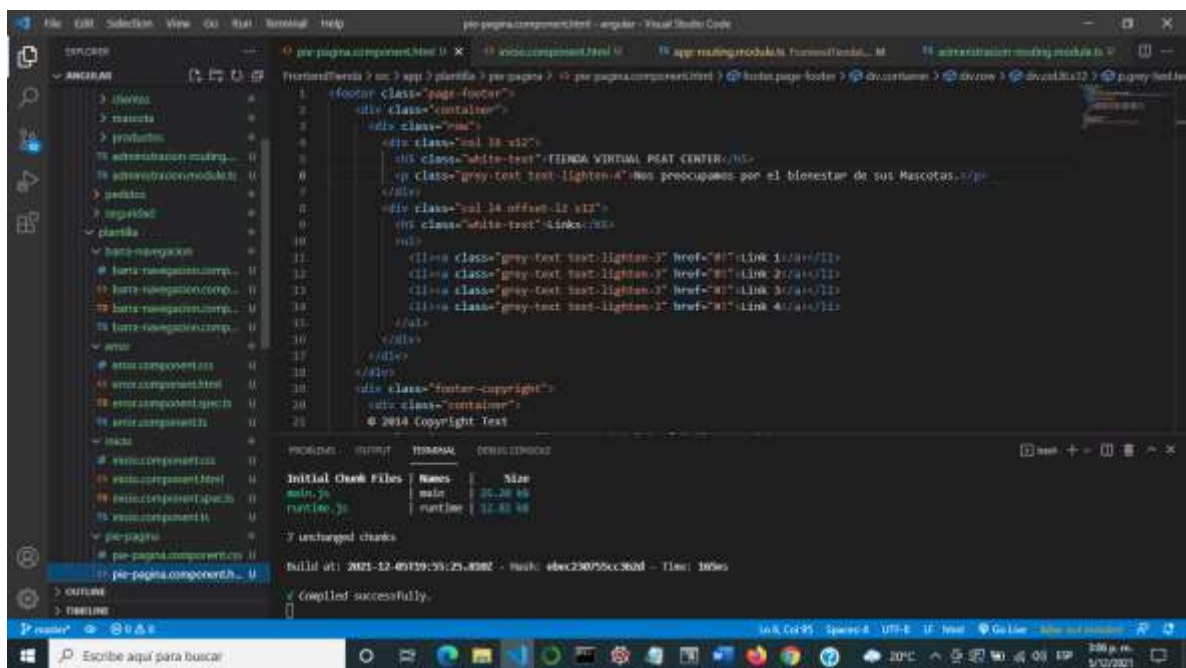
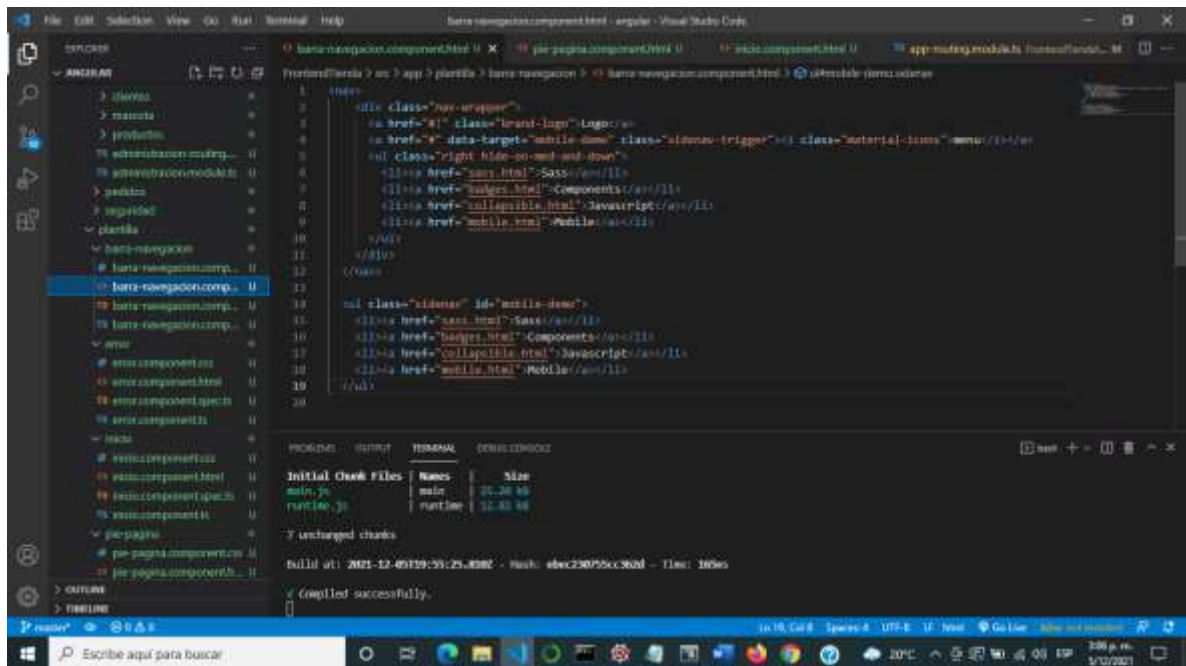
Instalamos nuestro framework CSS condiseño planos pero de gran utilidad



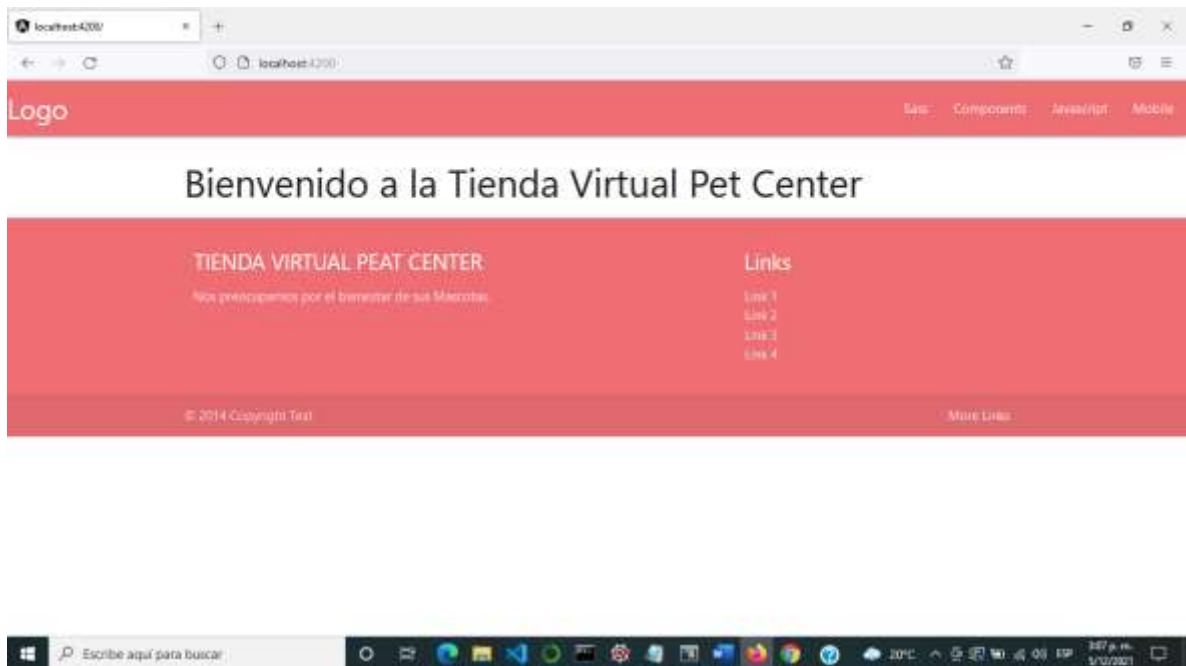
Copiamos la plantilla que Materilize CSS nos ofrece en su pagina



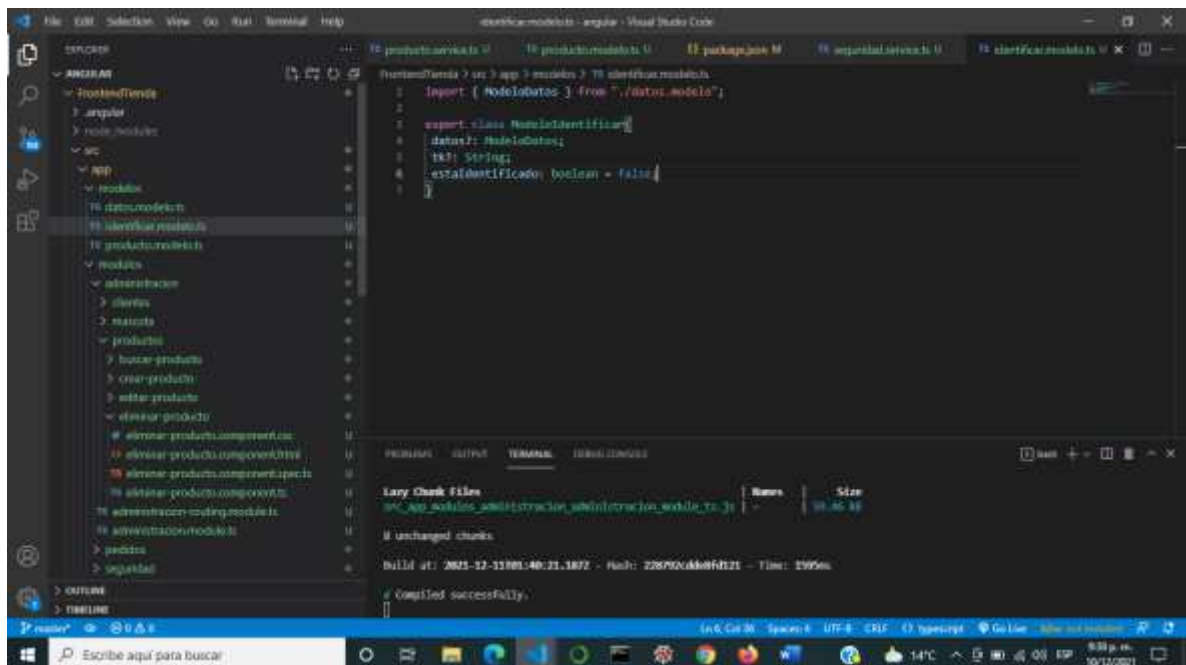
Implementamos en código que nos ofrece el framework Materilize por medio de davtar y Footer por lo que obtenemos



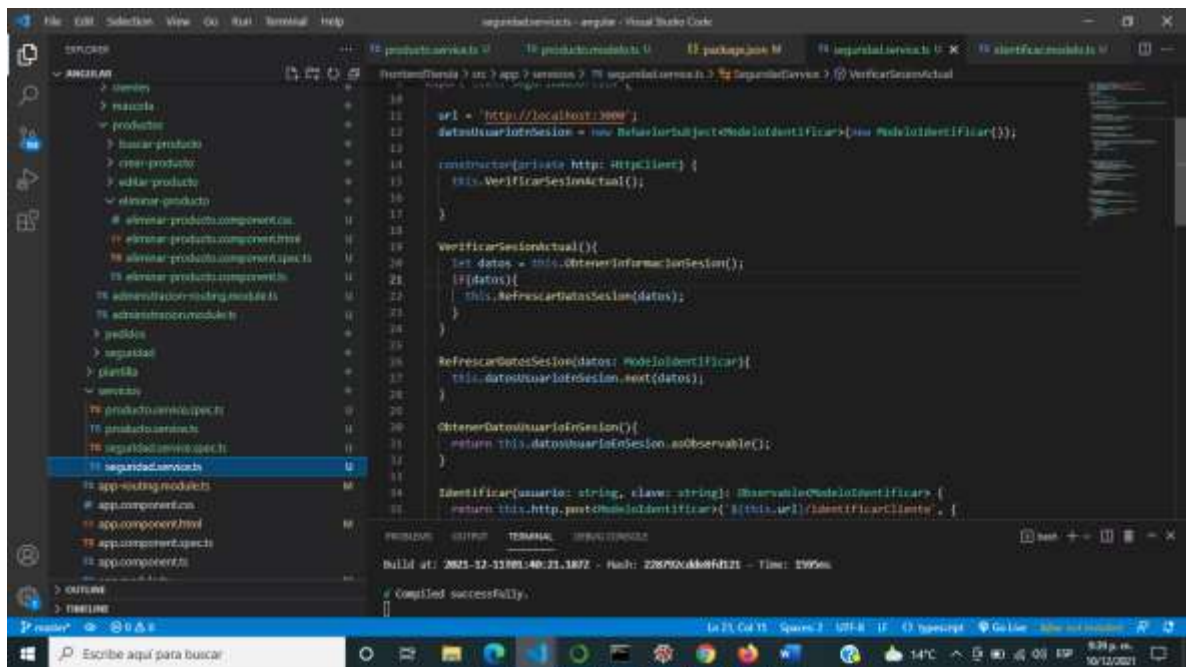
Validamos la aplicación en Mozilla y obtenemos



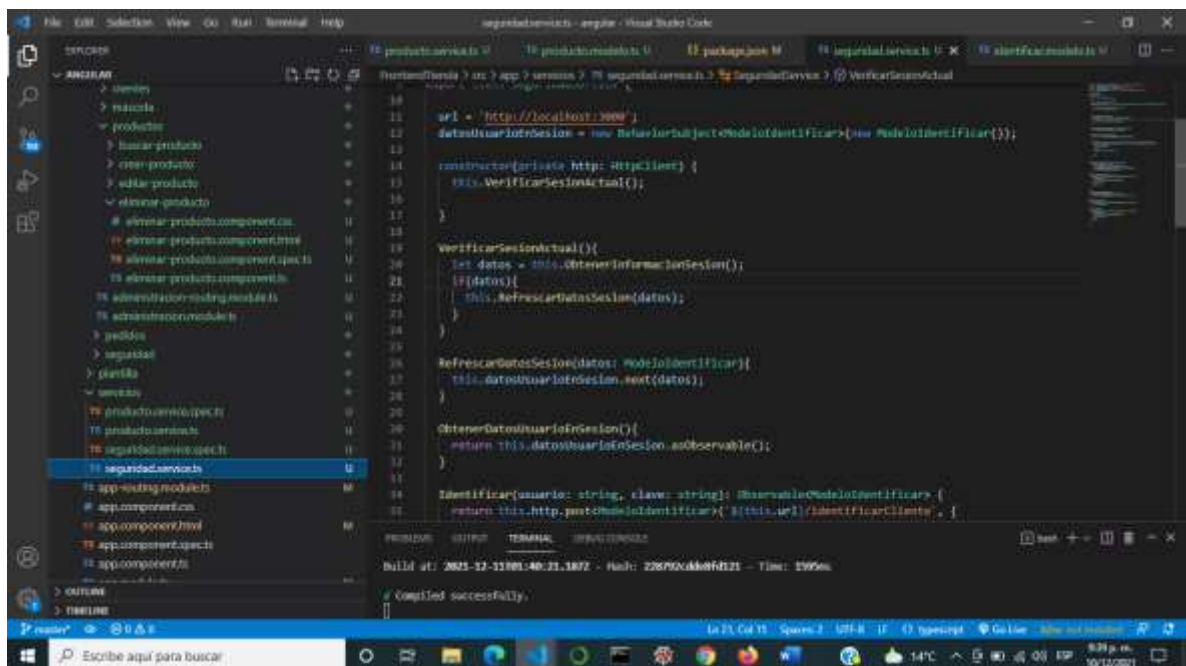
Ahora implementamos nuestra opción de identificación para el ingreso al sistema por lo que es necesario crear un modelo al cual llamaremos identificar



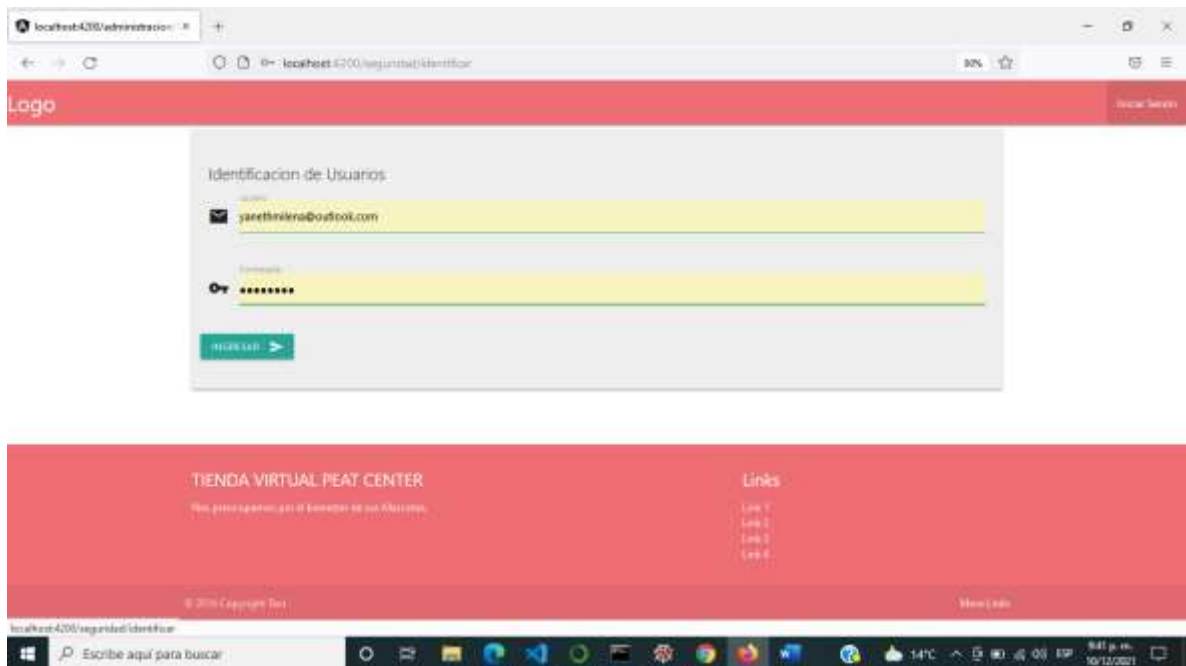
Igual creamos un servicio el cual nos permitirá realizar la conexión con el backend de nuestra tienda virtual mediante el enrutamiento a nuestro puerto localhost:3000



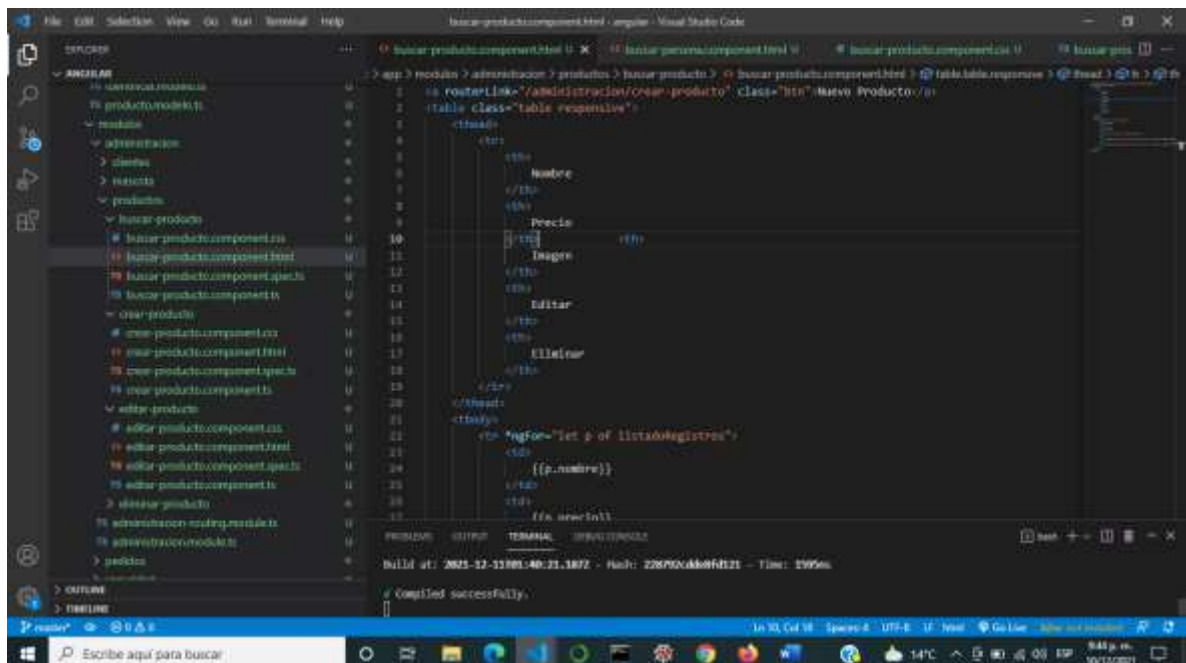
Y la configuración en nuestra plantilla de barra de navegación que es la puerta inicial de nuestro frontend al sistema de mascotas



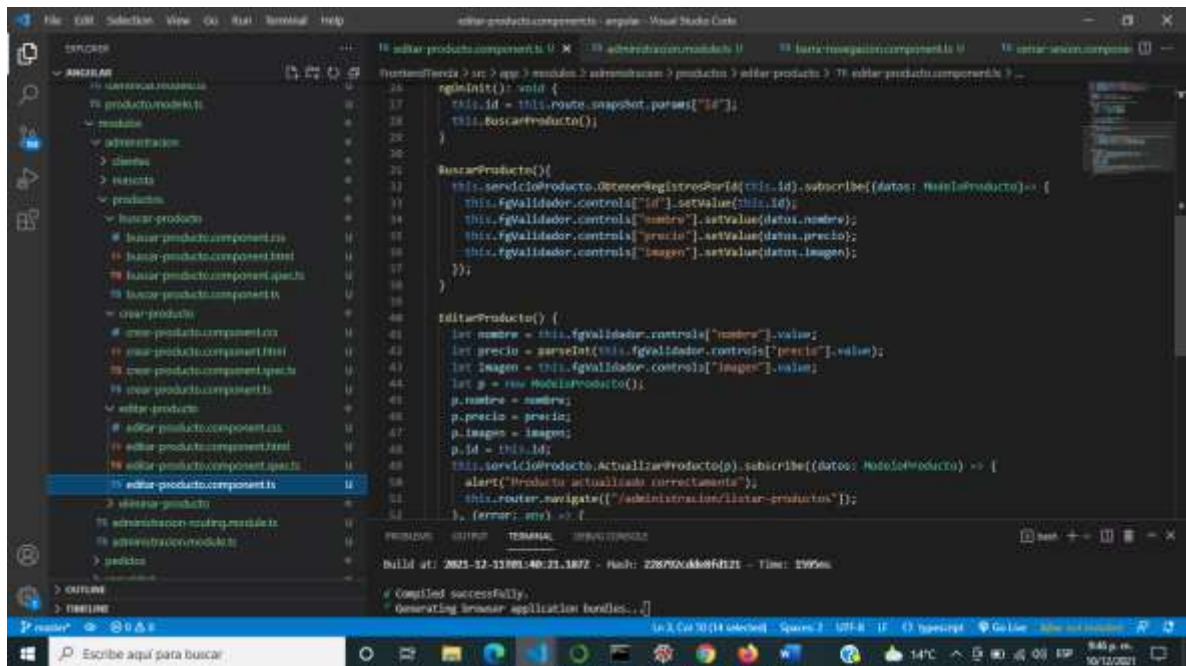
Por lo cual tenemos este resultado en nuestro localhost:4200



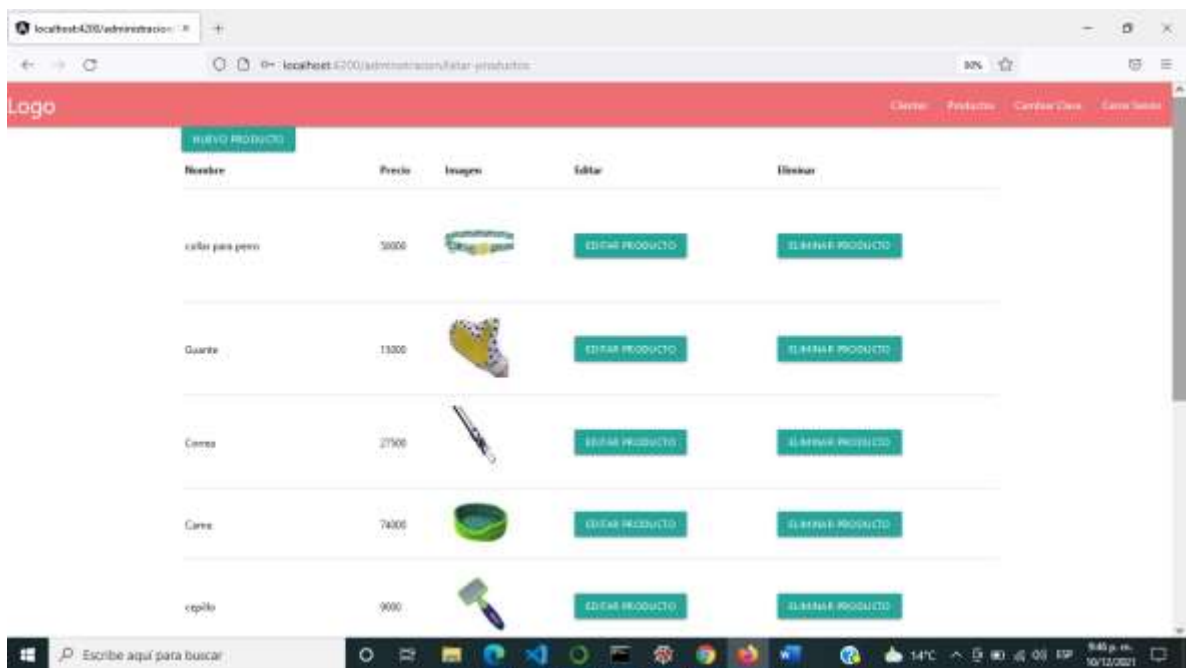
Luego de ingresar al sistema por medio del usuario y la clave generada y enviada mediante mensaje al correo electrónico configuramos el ingreso a nuestro producto en donde vamos a implementar las operaciones crud



Configuramos cada operación tanto en el html como en el controlador de las operaciones buscar, eliminar, crear y editar para este caso lo trabajamos sobre el modelo producto mediante la creación de una lista de productos.



Validamos en el localhost:4200 los resultados obtenidos

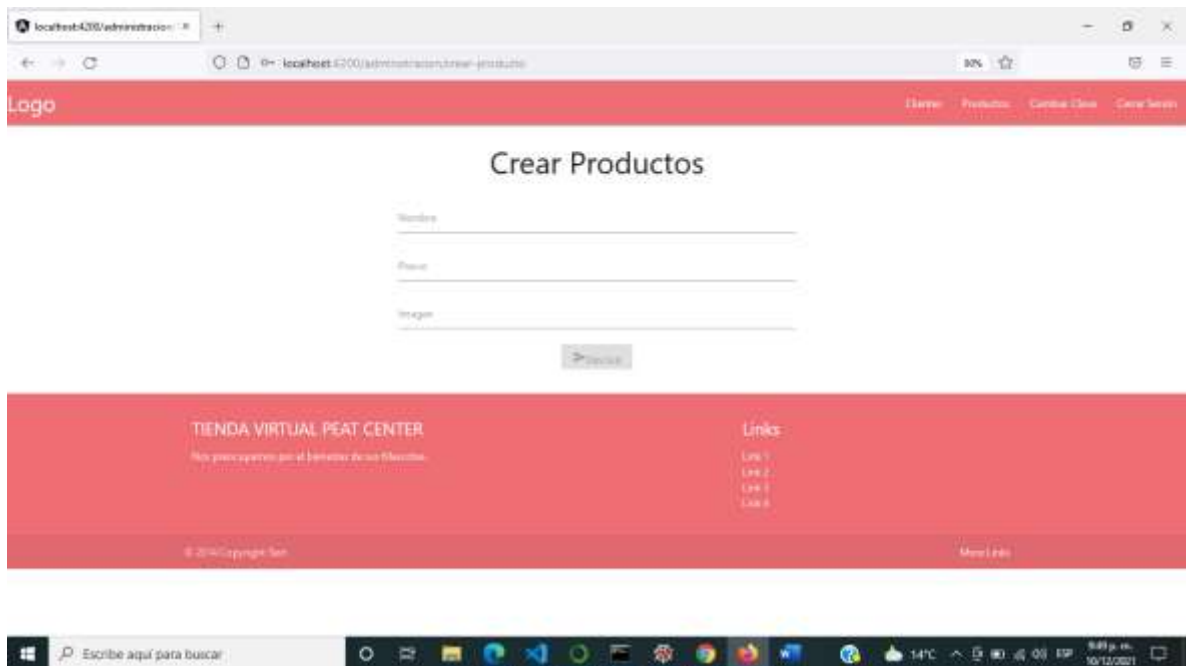


Para la creación de un nuevo producto se implemento un botón de Crear Producto en donde podemos incluir n cantidad de productos con todos sus atributos como son el nombre, precio, valor e imagen

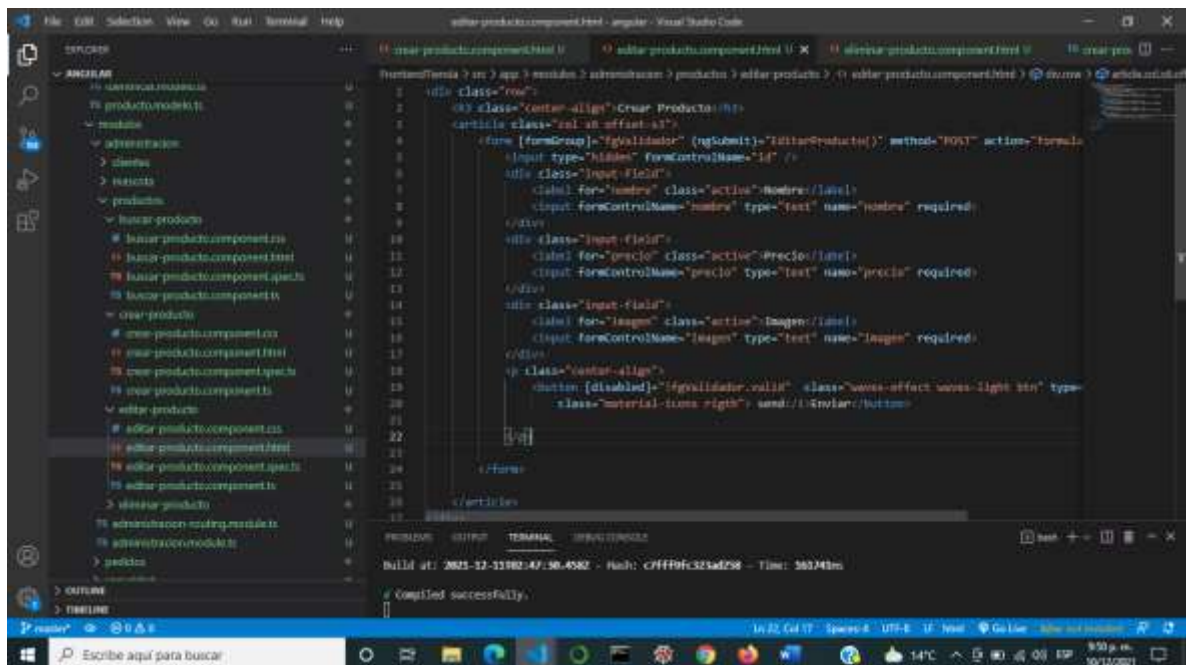
```
ngSubmit(): void {  
  this.validador.formGroup = this.fb.group({  
    nombre: ['', [Validators.required]],  
    precio: ['', [Validators.required]],  
    imagen: ['', [Validators.required]]  
  });  
  
  this._constructor.injector.get(ProductoService).  
    createProducto(this._router).subscribe(  
      (response) => {  
        alert('Producto almacenado correctamente');  
        this.router.navigate(['/administracion/listar-productos']);  
      }, (error) => {  
        alert('Error almacenando el producto');  
      })  
    );  
}
```

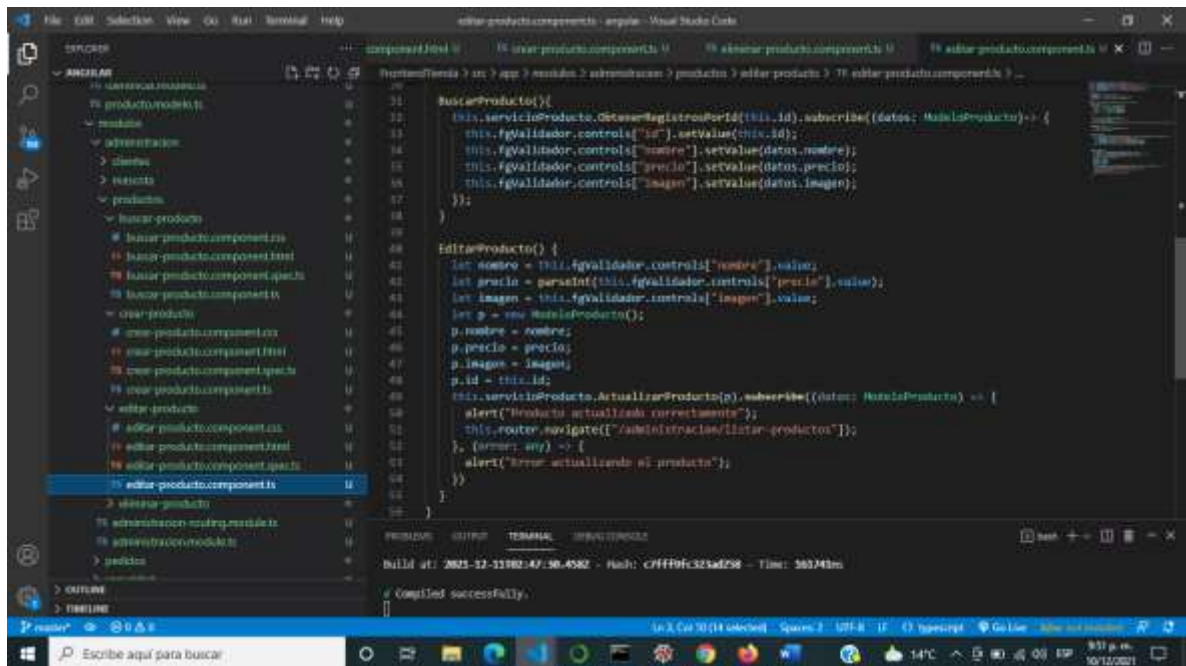
```
<div class="row">  
  <div class="center-align"> Crear Productos </div>  
  <div class="col s12 m12 l12">  
    <form [formGroup]="validador" (ngSubmit)="guardarProducto()" method="POST" action="form">  
      <div class="input-field">  
        <label for="nombre">Nombre </label>  
        <input type="text" name="nombre" required />  
      </div>  
      <div class="input-field">  
        <label for="precio">Precio </label>  
        <input type="text" name="precio" required />  
      </div>  
      <div class="input-field">  
        <label for="imagen">Imagen </label>  
        <input type="text" name="imagen" required />  
      </div>  
      <div class="center-align">  
        <button type="button" class="waves-effect waves-light btn">  
          Guardar </button>  
      </div>  
    </form>  
  </div>  
</div>
```

Los resultados obtenidos mediante la ejecución son

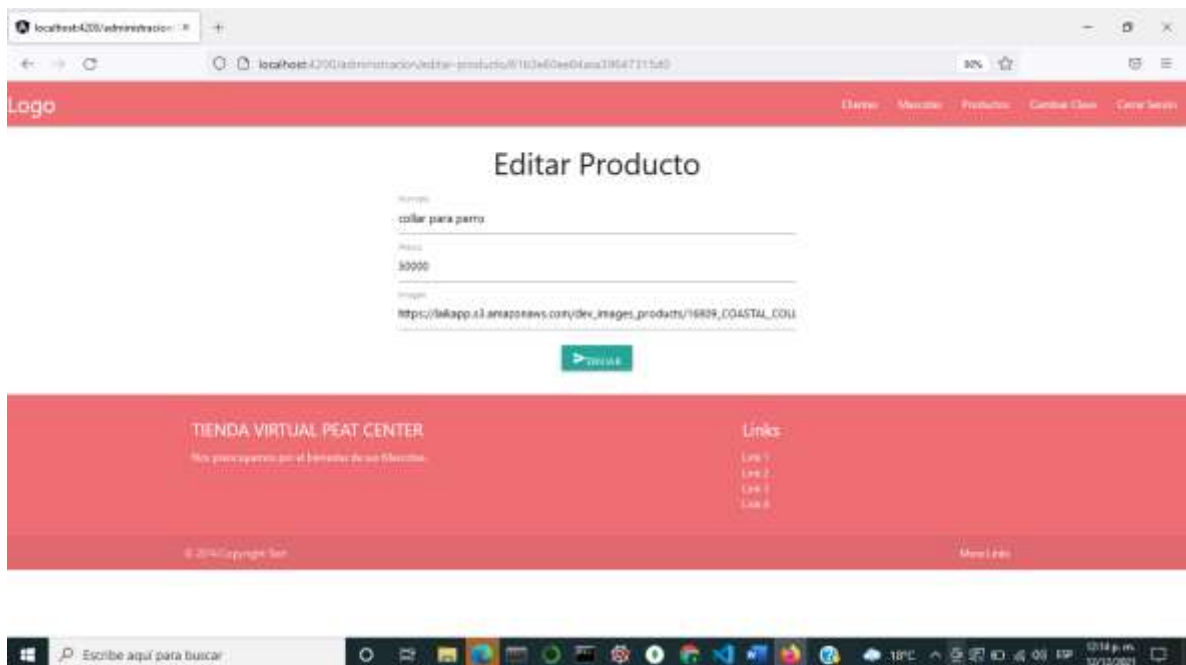


Igualmente se programo la opción de editar algún atributo del producto el cual nos permitirá modificar cualquier producto o cualquier campo que le pertenece como es nombre, precio e imagen.





En donde al ejecutar el programa se obtendrá los datos del producto actual a modificar y en cada campo se puede realizar o sobre escribir la modificación pertinente



```

// src/app/productos/componentes/filisnae-products.component.ts
import { Component, OnInit } from '@angular/core';
import { FilisnaeProductsComponent.html } from './filisnae-products.component.html';
import { FilisnaeProductsComponent.css } from './filisnae-products.component.css';
import { FilisnaeProductsComponent.specs } from './filisnae-products.component.specs';
import { Router } from '@angular/router';
import { FilisnaeProductsService } from '../../services/filisnae-products.service';

@Component({
  selector: 'filisnae-products',
  templateUrl: './filisnae-products.component.html',
  styleUrls: ['./filisnae-products.component.css'],
})
export class FilisnaeProductsComponent implements OnInit {
  id: string = '';

  constructor(
    private serviceFilisnaeProducts: FilisnaeProductsService,
    private router: Router,
    private route: ActivatedRoute
  ) {}

  ngOnInit(): void {
    this.id = this.route.snapshot.params['id'];
    this.filisnaeProducts();
  }

  filisnaeProducts() {
    this.serviceFilisnaeProducts.filisnaeProducts(this.id).subscribe((data: any) => {
      alert('Producto actualizado correctamente');
      this.router.navigate(['/administracion/filisnae-productos']);
    }, (error: any) => {
      alert('Error actualizando el producto');
    });
  }
}

```

The screenshot shows a web browser window with the address bar displaying 'localhost:4200/administracion/web/index.php/products/61034041e9b0a379647313d3'. The page content is partially obscured by a modal dialog box. The dialog box has a title 'localhost4200' and a message 'Producto actualizado correctamente' (Product updated successfully). There is a blue 'Aceptar' (Accept) button at the bottom right of the dialog. The background of the page shows a dark red header with a 'Logo' and a navigation menu. Below the header, there is a section titled 'TENDA VIRTUAL PEAT CENTER' and a 'Links' section with several links.

