

**Universidad de Nariño.**

**Ingeniería de Sistemas.**

**Diplomado de actualización en nuevas tecnologías para el desarrollo de Software.**

### Taller Unidad 3 Frontend

**Presentado por: Dana Criollo.**

- Inicialicé mi espacio de trabajo con angular, el nombre que le coloqué a mi proyecto fue 'dcfruverFE':

The screenshot shows the Visual Studio Code interface. The terminal window at the bottom displays the command: `PS C:\Users\JAIRO\Downloads\Diplomado\Modulo2\dcfruver-frontend> ng new dcfruverFE`. A red arrow points to the command. The response from the terminal is: `Would you like to add Angular routing? Yes or No? Yes Which prebuilt theme would you like to use? Material`. Another red arrow points to the 'Material' option. The rest of the terminal output shows the creation of files like angular.json, package.json, README.md, tsconfig.json, editorconfig, .gitignore, etc. The status bar at the bottom shows the weather as 15°C and ventoso (windy).

- Instalé los paquetes necesarios para usar Bootstrap en mi proyecto:

The screenshot shows the VS Code interface with the terminal tab active. The command `npm install bootstrap jquery @popperjs/core` is being typed into the terminal. The output shows that 3 packages were added and 907 packages are looking for funding. A red arrow points to the command in the terminal.

```

src > app > app.component.html ...
Go to component [ You, hace 18 minutos ] | author (You)
1 <!-- This content below is only a placeholder -->
2 <!-- and will be replaced by your content -->
3 <!-- The content below is only a placeholder -->
4 <!-- and will be replaced by your content -->
5 <!-- The content below is only a placeholder -->
6 <!-- and will be replaced by your content -->
7 <!-- to get started with your project! -->
8 <!-- and will be replaced by your content -->
9
10 <style>
11   :host {
12     font-family: -apple-system, BlinkMacSystemFont, "Segoe UI", Roboto, Helvetica, Arial, sans-serif, "Apple Color Emoji", "Segoe UI E
13     font-size: 14px;
14     color: #333;
15     box-sizing: border-box;
16     -webkit-font-smoothing: antialiased;
17     -moz-osx-font-smoothing: grayscale;
18   }
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
  
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL GITLENS

PS C:\Users\JAIRO\Downloads\Diplomado\Modulo2\dcfruver-frontend\dcfruverFE> `npm install bootstrap jquery @popperjs/core`

added 3 packages, and audited 907 packages in 5s

107 packages are looking for funding  
run `npm fund` for details

found 0 vulnerabilities

PS C:\Users\JAIRO\Downloads\Diplomado\Modulo2\dcfruver-frontend\dcfruverFE>

You, hace 18 minutos Lin. 1, col. 1 Espacios: 2 UTF-8 LF HTML Go Live 02:06 p.m. 01/08/2023

- Agregué las referencias a los archivos .css y .js de Bootstrap necesarios para su funcionamiento en este proyecto desde el archivo 'angular.json':

The screenshot shows the VS Code interface with the editor tab active, displaying the `angular.json` file. The `scripts` section is highlighted with a red box. A red arrow points to the `scripts` section in the code.

```

{
  "projects": {
    "dcfruverFE": {
      "architect": {
        "build": {
          "options": {
            "scripts": [
              "node_modules/bootstrap/dist/js/bootstrap.min.js",
              "node_modules/@popperjs/core/dist/umd/popper.min.js",
              "node_modules/bootstrap/dist/js/bootstrap.min.js"
            ]
          }
        }
      }
    }
  },
  "options": {
    "polyfills": [
      "zone.js"
    ],
    "tsConfig": "tsconfig.app.json",
    "assets": [
      "src/favicon.ico",
      "src/assets"
    ],
    "styles": [
      "node_modules/bootstrap/dist/css/bootstrap.min.css",
      "src/styles.css"
    ],
    "scripts": [
      "node_modules/jquery/dist/jquery.min.js",
      "node_modules/@popperjs/core/dist/umd/popper.min.js",
      "node_modules/bootstrap/dist/js/bootstrap.min.js"
    ]
  },
  "configurations": {
    "production": {}
  }
}
  
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL GITLENS

PS C:\Users\JAIRO\Downloads\Diplomado\Modulo2\dcfruver-frontend\dcfruverFE> `npm install bootstrap jquery @popperjs/core`

added 3 packages, and audited 907 packages in 5s

107 packages are looking for funding  
run `npm fund` for details

found 0 vulnerabilities

PS C:\Users\JAIRO\Downloads\Diplomado\Modulo2\dcfruver-frontend\dcfruverFE>

You, hace 1 minuto Lin. 35, col. 14 Espacios: 2 UTF-8 LF JSON Go Live 02:21 p.m. 01/08/2023

Hice un commit con los cambios hasta este punto llamado "Proyecto con Bootstrap instalado".

- Creé un repositorio remoto con el nombre 'dcfruverFE' y lo sincronicé con mi repositorio local:

**Create a new repository**

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Required fields are marked with an asterisk (\*).

Owner \* DanaCriolloLopez Repository name \* dcfruverFE  dcfruverFE is available.

Great repository names are short and memorable. Need inspiration? How about solid-lamp ?

Description (optional) Taller Unidad 3 Frontend - Módulo 2 Diplomado

Public Anyone on the internet can see this repository. You choose who can commit.  Private You choose who can see and commit to this repository.

Initialize this repository with:  Add a README file This is where you can write a long description for your project. Learn more about READMEs.

Add .gitignore  Choose which files not to track from a list of templates. Learn more about ignoring files.

Choose a license  A license tells others what they can and can't do with your code. Learn more about licenses.

You are creating a public repository in your personal account.

DanaCriolloLopez/dcfruverFE · Public

1 main · 1 branch · 0 tags ↓

Go to file Add file

**About**

Taller Unidad 3 Frontend - Módulo 2 Diplomado

README  Activity  0 stars  1 watching  0 forks

**Releases**

No releases published Create a new release

**Packages**

No packages published Publish your first package

**Code**

main · 1 branch · 0 tags

Go to file Add file

**dcfruverFE**

This project was generated with Angular CLI version 16.1.6.

**Development server**

Run `ng serve` for a dev server. Navigate to `http://localhost:4200/`. The application will automatically reload if you change any of the source files.

- Creé mi primer componente llamado ‘pagina-principal’, la cual será mi página principal de la aplicación web en donde se desplegarán los productos a la venta del fruver:

```

PS C:\Users\JATIRO\Downloads\Diplomado\Modulo2\dcfruver-frontend\dcfruverFE> ng generate component pagina-principal
CREATE src/app/pagina-principal/pagina-principal.component.html (31 bytes)
CREATE src/app/pagina-principal/pagina-principal.component.spec.ts (623 bytes)
CREATE src/app/pagina-principal/pagina-principal.component.ts (241 bytes)
CREATE src/app/pagina-principal/pagina-principal.component.css (0 bytes)
● UPDATE src/app/app.module.ts (513 bytes)
○ PS C:\Users\JATIRO\Downloads\Diplomado\Modulo2\dcfruver-frontend\dcfruverFE> []

```

- Agregué en el archivo ‘app-routing.module.ts’ las rutas que me van a dirigir a mi página principal:

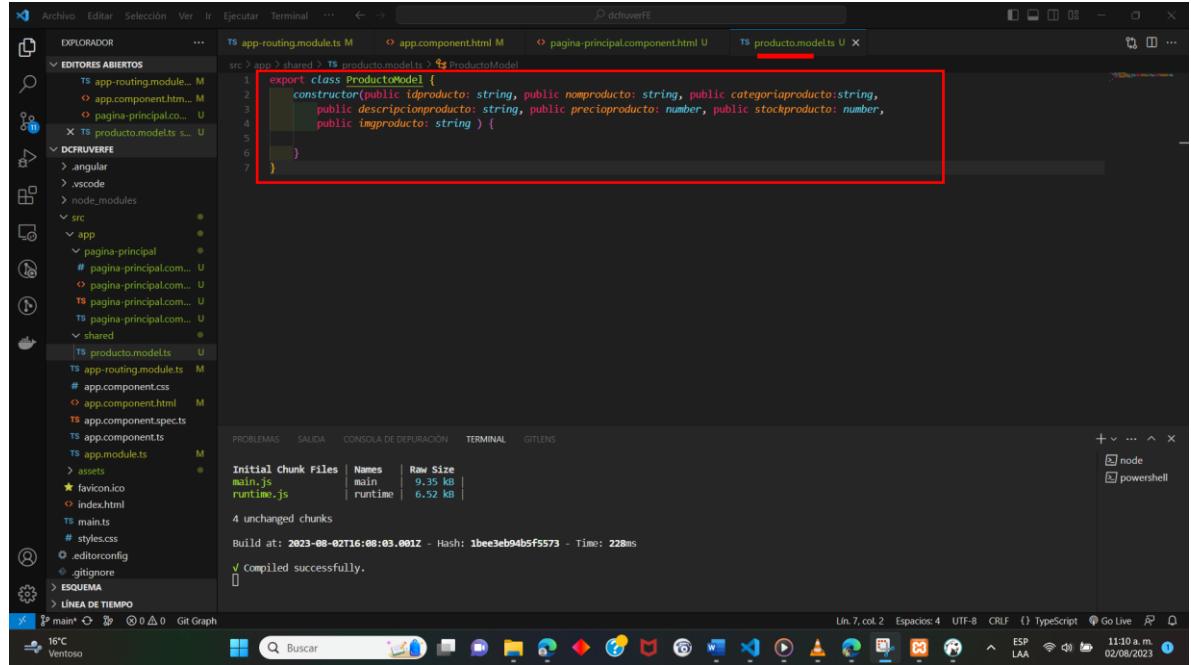
```

const routes: Routes = [
  //Rutas que redirigen a mi página principal
  { path: 'principal', component: PaginaPrincipalComponent },
  //Cuálquier ruta que pongan se va a redirigir a mi página principal
  { path: '**', redirectTo:'/principal', pathMatch: 'full' },
];

```

- Creé la carpeta shared y dentro de ella creé los siguientes archivos:

Archivo 'producto.model.ts' que contiene la clase ProductoModel:

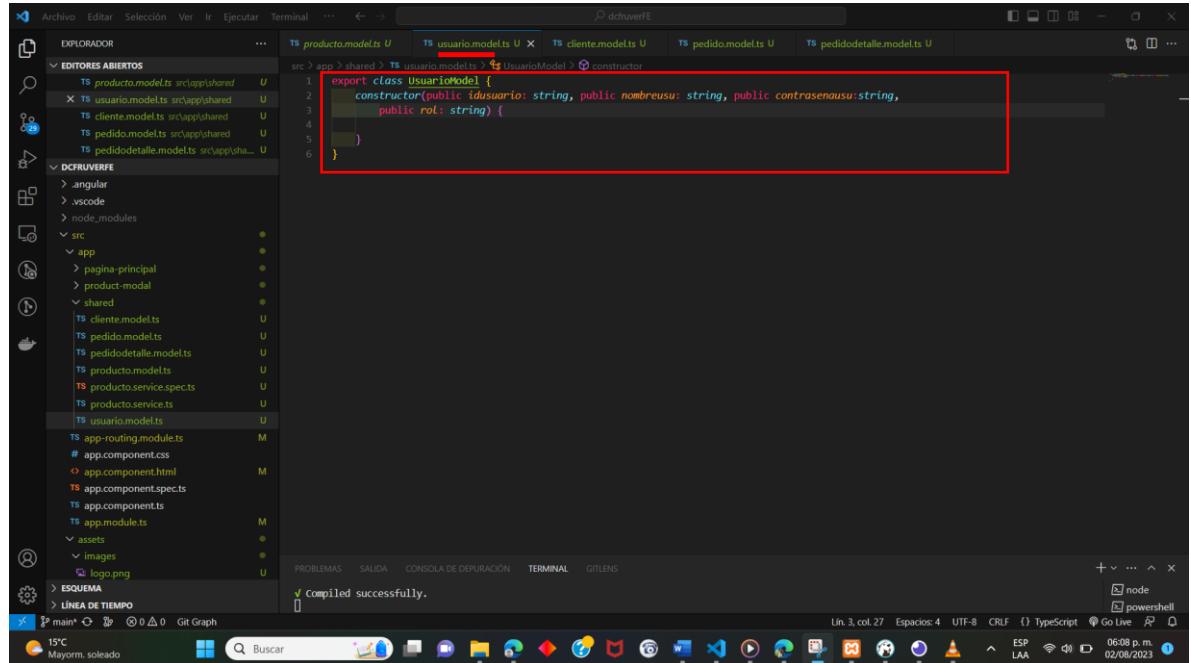


```

src > app > shared > TS producto.model.ts > ProductoModel
1 export class ProductoModel {
2     constructor(public idproducto: string, public nombreproducto: string, public categoriaproducto:string,
3         public descripcionproducto: string, public precioproducto: number, public stockproducto: number,
4         public imgproducto: string ) {
5
6     }
7 }

```

Archivo 'usuario.model.ts' que contiene la clase UsuarioModel:

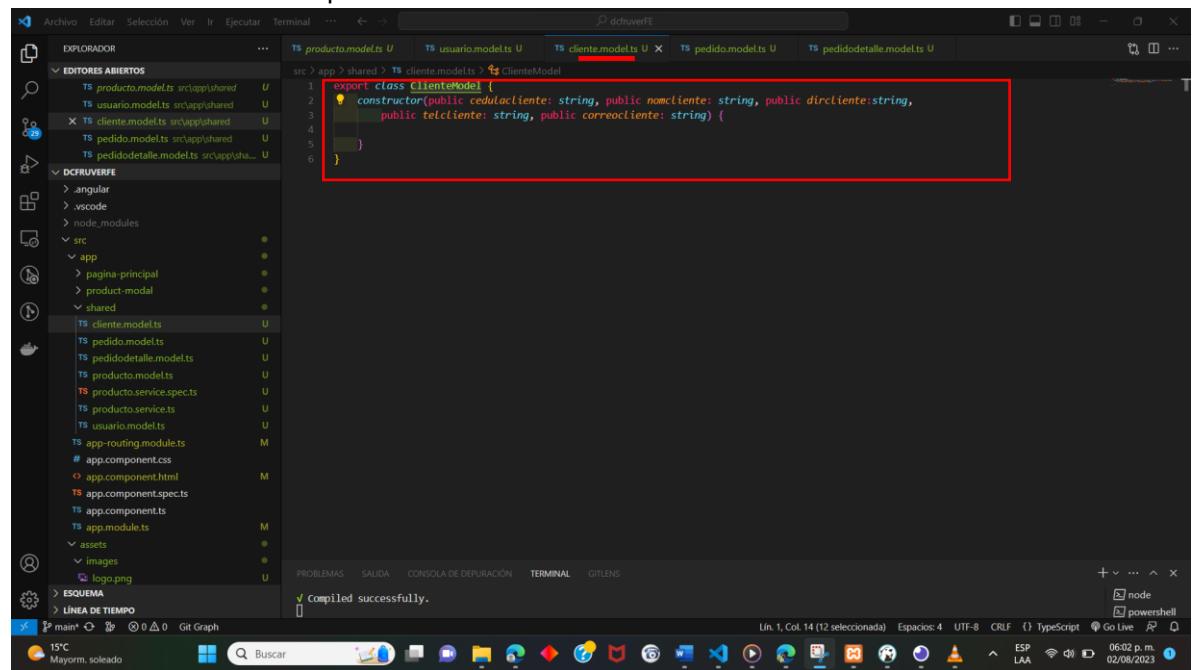


```

src > app > shared > TS usuario.model.ts > UsuarioModel > constructor
1 export class UsuarioModel {
2     constructor(public idusuario: string, public nombreusu: string, public contrasenausu:string,
3         public rol: string) {
4
5     }
6 }

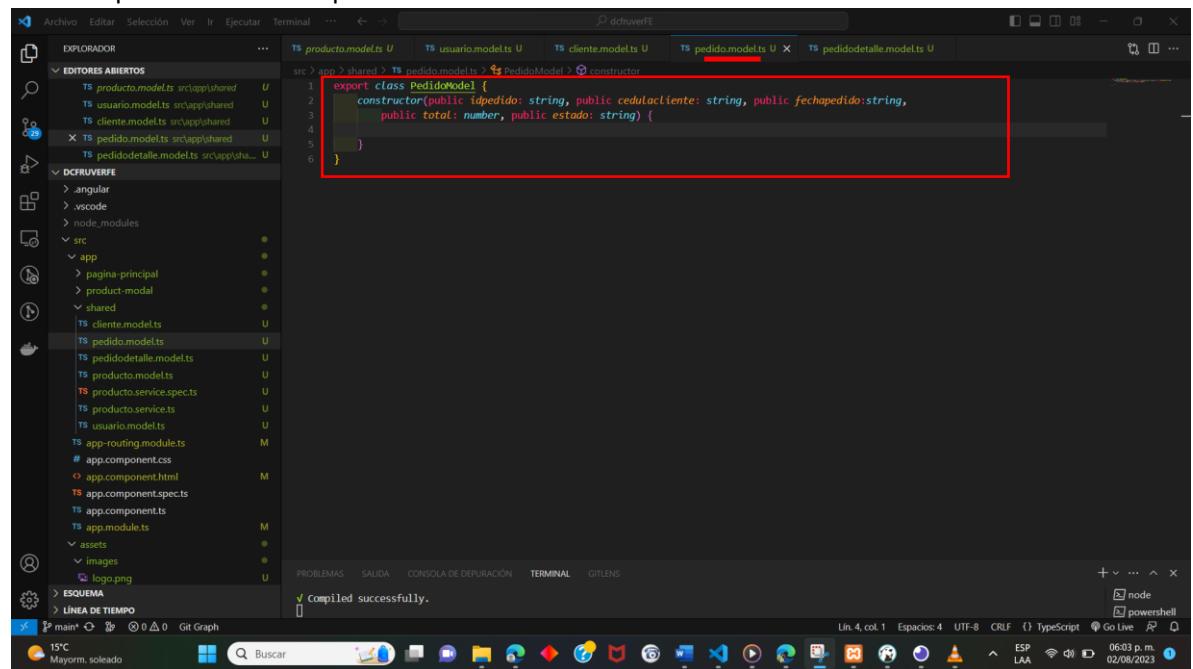
```

Archivo 'cliente.model.ts' que contiene la clase ClienteModel:



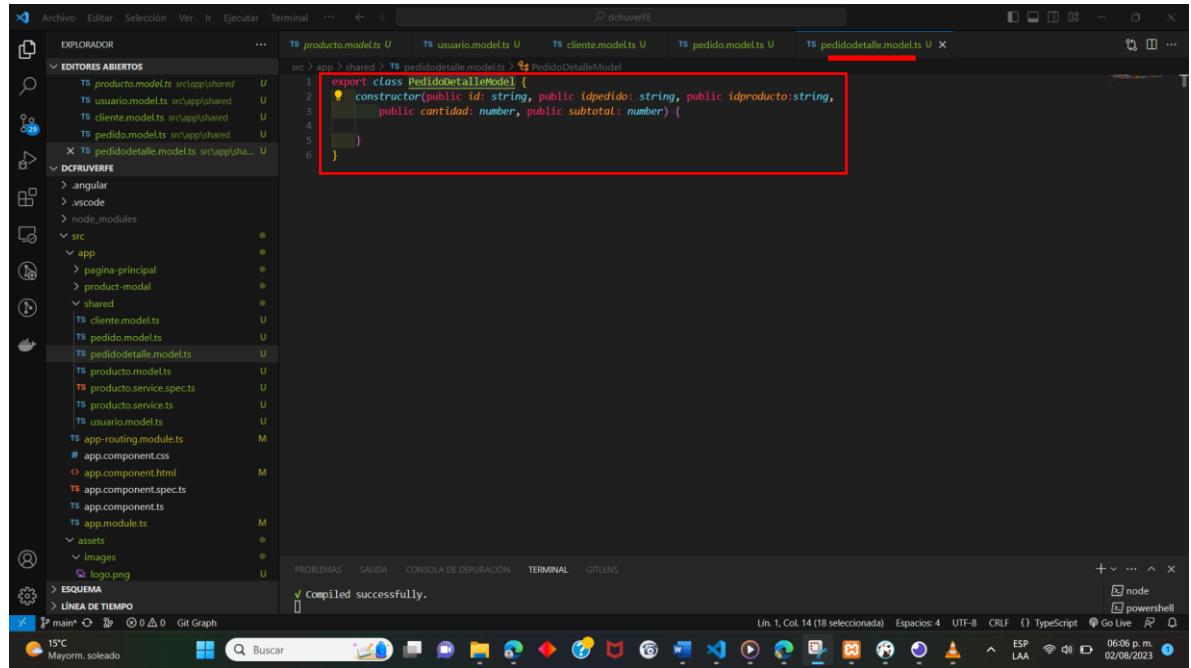
```
1 export class ClienteModel {
2     constructor(public cedulacliente: string, public nombrecliente: string, public dircliente:string,
3                 public telcliente: string, public correocliente: string) {
4     }
5 }
```

Archivo 'pedido.model.ts' que contiene el PedidoModel:



```
1 export class PedidoModel {
2     constructor(public idpedido: string, public cedulacliente: string, public fechapedido:string,
3                 public total: number, public estado: string) {
4     }
5 }
```

## Archivo 'pedidodetalle.model.ts' que contiene el PedidoDetalleModel:



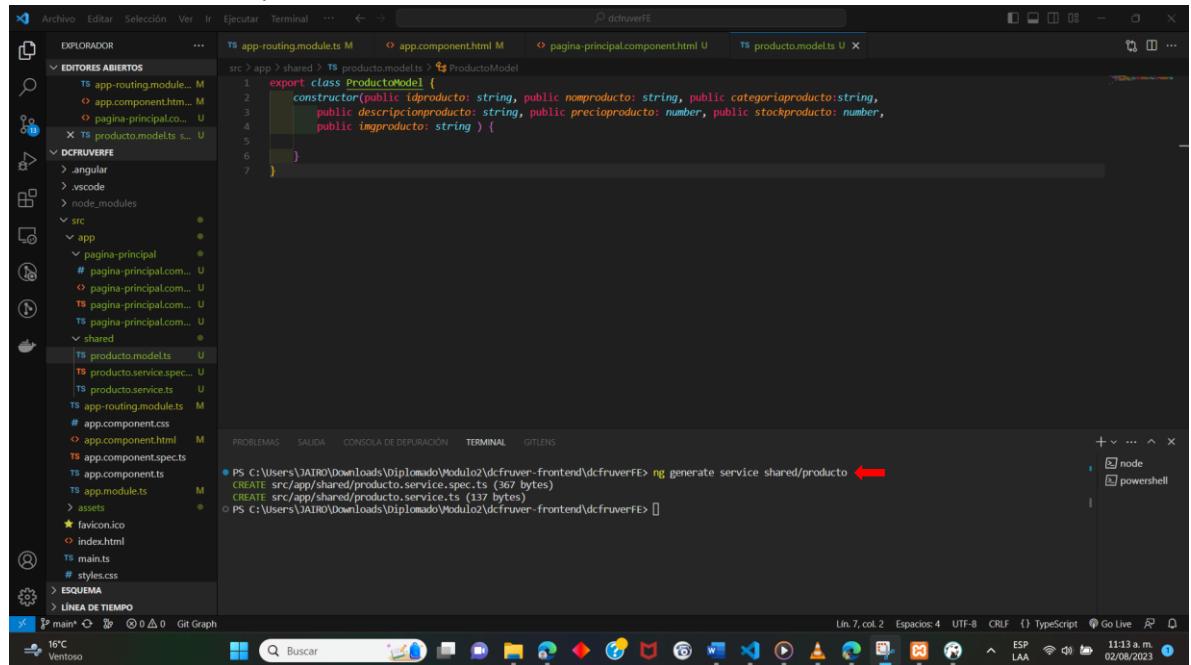
```

1 export class PedidoDetalleModel {
2     constructor(public id: string, public idpedido: string, public idproducto:string,
3                 public cantidad: number, public subtotal: number) {
4
5     }
6

```

- Creé el servicio para cada modelo creado anteriormente dentro de mi carpeta shared:

## Creación del servicio para el ProductoModel:



```

1 export class ProductoModel {
2     constructor(public idproducto: string, public nombreproducto: string, public categoriaproducto:string,
3                 public descripcionproducto: string, public precioproducto: number, public stockproducto: number,
4                 public imgproducto: string) {
5
6     }
7

```

TERMINAL: ng generate service shared/producto

Creación del servicio para UsuarioModel, ClienteModel, PedidoModel y PedidoDetalleModel:

```

src > app > shared > ts producto.model.ts > ProductService
1 //Modelo para Producto
2 export class ProductoModel {
3   constructor(public idproducto: string, public nombreproducto: string, public categoriaproducto: string,
4   public descripciónenproducto: string, public precioproducto: number, public stockproducto: number,
5   public imgproducto: string) {
6
7 }
8 }

src > app > shared > ts usuario.model.ts > UsuarioService
1 CREATE src/app/shared/usuario.service.spec.ts (362 bytes)
2 CREATE src/app/shared/usuario.service.ts (136 bytes)

src > app > shared > ts cliente.model.ts > ClienteService
1 CREATE src/app/shared/cliente.service.spec.ts (362 bytes)
2 CREATE src/app/shared/cliente.service.ts (136 bytes)

src > app > shared > ts pedido.model.ts > PedidoService
1 CREATE src/app/shared/pedido.service.spec.ts (357 bytes)
2 CREATE src/app/shared/pedido.service.ts (135 bytes)

src > app > shared > ts pedido_detalle.model.ts > PedidoDetalleService
1 CREATE src/app/shared/pedido_detalle.service.spec.ts (392 bytes)
2 CREATE src/app/shared/pedido_detalle.service.ts (142 bytes)

PS C:\Users\Jairo\Downloads\Diplomado\Modules\Modulo2\dcfruver-frontend> ng generate service shared/usuario
CREATE src/app/shared/usuario.service.spec.ts (362 bytes)
CREATE src/app/shared/usuario.service.ts (136 bytes)
PS C:\Users\Jairo\Downloads\Diplomado\Modules\Modulo2\dcfruver-frontend> ng generate service shared/cliente
CREATE src/app/shared/cliente.service.spec.ts (362 bytes)
CREATE src/app/shared/cliente.service.ts (136 bytes)
PS C:\Users\Jairo\Downloads\Diplomado\Modules\Modulo2\dcfruver-frontend> ng generate service shared/pedido
CREATE src/app/shared/pedido.service.spec.ts (357 bytes)
CREATE src/app/shared/pedido.service.ts (135 bytes)
PS C:\Users\Jairo\Downloads\Diplomado\Modules\Modulo2\dcfruver-frontend> ng generate service shared/pedidodetalle
CREATE src/app/shared/pedido_detalle.service.spec.ts (392 bytes)
CREATE src/app/shared/pedido_detalle.service.ts (142 bytes)
PS C:\Users\Jairo\Downloads\Diplomado\Modules\Modulo2\dcfruver-frontend>

```

- Agregué los anteriores servicios e importe los módulos ‘HttpClientModule’ y ‘FormsModule’ en el archivo ‘app.module.ts’:

```

src > app > ts app.module.ts > AppModule
1 import { NgModule } from '@angular/core';
2 import { BrowserModule } from '@angular/platform-browser';
3 import { HttpClientModule } from '@angular/common/http'; ←
4 import { FormsModule } from '@angular/forms'; ←
5 import { AppRoutingModule } from './app-routing.module';
6 import { AppComponent } from './app.component';
7 import { PaginaPrincipalComponent } from './pagina-principal/pagina-principal.component';
8 import { ProductoService } from './shared/producto.service';
9 import { NgbModule } from '@ng-bootstrap/ng-bootstrap';
10 import { ProductModalComponent } from './product-modal/product-modal.component';
11 import { UsuarioService } from './shared/usuario.service';
12 import { ClienteService } from './shared/cliente.service';
13 import { PedidoService } from './shared/pedido.service';
14 import { PedidoDetalleService } from './shared/pedidodetalle.service';

You have 2 minutos | 1 author (You)
15
16 @NgModule({
17   declarations: [
18     AppComponent,
19     PaginaPrincipalComponent,
20     ProductModalComponent
21   ],
22   imports: [
23     BrowserModule,
24     AppRoutingModule,
25     HttpClientModule, ←
26     FormsModule, ←
27     NgbModule ←
28   ],
29   providers: [
30     ProductoService,
31     UsuarioService,
32     ClienteService,
33     PedidoService,
34     PedidoDetalleService ←
35   ],
36 })
37 
```

- En los archivos ‘service.ts’ de cada servicio creado anteriormente creé las funciones que me van a permitir interactuar con mi backend creado en el taller 2:

#### Funciones del archivo ‘producto.service.ts’:

```

src > app > shared > TS producto.services > TS ProductoService > actualizarProducto
1 import { HttpClient } from '@angular/common/http';
2 import { Injectable } from '@angular/core';
3 import { ProductoModel } from './producto.model';
4
5 @Injectable({
6   providedIn: 'root'
7 })
8 export class ProductoService {
9
10   // Definimos la URL base del servidor donde se encuentra la base de datos
11   BASE_URL = 'http://localhost:3000';
12
13   // En el constructor inyectamos el servicio HttpClient
14   constructor(private http:HttpClient) {}
15
16   // Función que obtiene los productos registrados en la base de datos
17   obtenerProductos() {
18     // Realiza una solicitud HTTP GET al servidor y espera un array de objetos ProductoModel como respuesta
19     return this.http.get<ProductoModel[]>(`${this.BASE_URL}/productos`);
20   }
21
22   // Función que agrega un producto en la base de datos
23   // Recibe un objeto ProductoModel como parámetro
24   agregarProducto(producto: ProductoModel) {
25     return this.http.post<string>(`${this.BASE_URL}/productos`, producto);
26   }
27
28   // Función que actualiza un producto existente en la base de datos
29   // Recibe un objeto ProductoModel como parámetro
30   actualizarProducto(producto: ProductoModel) {
31     return this.http.put<string>(`${this.BASE_URL}/productos/${producto.idproducto}`, producto);
32   }
33
34   // Función que borra un producto de la base de datos según su ID
35   // Recibe el ID del producto como parámetro
36   borrarProducto(idproducto: string) {
37     return this.http.delete<string>(`${this.BASE_URL}/productos/${idproducto}`);
38   }
}

```

#### Funciones del archivo ‘usuario.service.ts’:

```

src > app > shared > TS usuario.services > TS UsuarioService > obtenerCredenciales
1 import { HttpClient } from '@angular/common/http';
2 import { Injectable } from '@angular/core';
3 import { UsuarioModel } from './usuario.model';
4
5 @Injectable({
6   providedIn: 'root'
7 })
8 export class UsuarioService {
9
10   // Definimos la URL base del servidor donde se encuentra la base de datos
11   BASE_URL = 'http://localhost:3000';
12
13   // En el constructor inyectamos el servicio HttpClient
14   constructor(private http:HttpClient) {}
15
16   // Función que obtiene las credenciales de los usuarios registrado en la base de datos
17   obtenerCredenciales() {
18     // Realiza una solicitud HTTP GET al servidor y espera un array de objetos UsuarioModel como respuesta
19     return this.http.get<UsuarioModel[]>(`${this.BASE_URL}/usuarios`);
20   }
21
22 }

```

## Funciones del archivo 'cliente.service.ts':

```
src > app > shared > cliente.service.ts > ClienteService > constructor
1 import { HttpClient } from '@angular/common/http';
2 import { Injectable } from '@angular/core';
3 import { ClienteModel } from './cliente.model';
4
5 @Injectable({
6   providedIn: 'root'
7 })
8 export class ClienteService {
9
10   // Definimos la URL base del servidor donde se encuentra la base de datos
11   BASE_URL = 'http://localhost:3000';
12   // En el constructor injectamos el servicio HttpClient
13   constructor(private http:HttpClient) {}
14
15   // Función que obtiene los clientes registrados en la base de datos
16   obtenerClientes() {
17     // Realiza una solicitud HTTP GET al servidor y espera un array de objetos ClienteModel como respuesta
18     return this.http.get<ClienteModel[]>(`${this.BASE_URL}/clientes`);
19   }
20 }
```

## Funciones del archivo 'pedido.service.ts':

```
src > app > shared > pedido.service.ts > PedidoService
1 import { HttpClient } from '@angular/common/http';
2 import { Injectable } from '@angular/core';
3 import { PedidoModel } from './pedido.model';
4
5 @Injectable({
6   providedIn: 'root'
7 })
8 export class PedidoService {
9
10   // Definimos la URL base del servidor donde se encuentra la base de datos
11   BASE_URL = 'http://localhost:3000';
12   // En el constructor injectamos el servicio HttpClient
13   constructor(private http:HttpClient) {}
14
15   // Función que obtiene los pedidos pendientes registrados en la base de datos
16   obtenerPedidosPendientes() {
17     // Realiza una solicitud HTTP GET al servidor y espera un array de objetos PedidoModel como respuesta
18     return this.http.get<PedidoModel[]>(`${this.BASE_URL}/pedidos`);
19   }
20
21   // Función que obtiene todos los pedidos registrados en la base de datos
22   obtenerPedidos() {
23     // Realiza una solicitud HTTP GET al servidor y espera un array de objetos PedidoModel como respuesta
24     return this.http.get<PedidoModel[]>(`${this.BASE_URL}/pedidos`);
25   }
26
27   // Función que confirma un pedido registrado en la base de datos
28   confirmarPedido(idpedido: string) {
29     return this.http.get<string>(`${this.BASE_URL}/pedidos/${idpedido}`);
30   }
31
32   // Función que rechaza un pedido registrado en la base de datos
33   rechazarPedido(idpedido: string) {
34     return this.http.get<string>(`${this.BASE_URL}/pedidosss/${idpedido}`);
35   }
36
37 }
```

## Funciones del archivo 'pedidodetalleservice.ts':

```

src > app > shared > TS pedidodetalleservice.ts > Pedidodetalleservice
1 import { HttpClient } from '@angular/common/http';
2 import { Injectable } from '@angular/core';
3 import { PedidoDetalleModel } from './pedidodetalleservice.model';
4
5 @Injectable({
6   providedIn: 'root'
7 })
8 export class Pedidodetalleservice {
9
10   // Definimos la URL base del servidor donde se encuentra la base de datos
11   BASE_URL = 'http://localhost:3000';
12
13   // En el constructor inyectamos el servicio HttpClient
14   constructor(private http:HttpClient) {}
15
16   // Función que obtiene los detalles de un pedido específico registrados en la base de datos
17   obtenerPedidoDetalle(idpedido: string) {
18     // Realiza una solicitud HTTP GET al servidor y espera un array de objetos PedidoDetalleModel como respuesta
19     return this.http.get<PedidoDetalleModel[]>(`${this.BASE_URL}/pedidodetalles/${idpedido}`);
20   }
21
22   // Función que agrega un detalle a un pedido en la base de datos
23   // Recibe un objeto PedidoDetalleModel como parámetro
24   agregarProducto(pedido: PedidoDetalleModel) {
25     return this.http.post<string>(`${this.BASE_URL}/pedidos/${pedido.id}`,pedido);
26   }
27
28   // Función que actualiza la información de los detalles de un pedido existente en la base de datos
29   // Recibe un objeto PedidoDetalleModel como parámetro
30   actualizarProducto(pedido: PedidoDetalleModel) {
31     return this.http.put<string>(`${this.BASE_URL}/pedidos/${pedido.id}`,pedido);
32   }
33
34   // Función que borra un detalle de un pedido de la base de datos según su ID
35   // Recibe el ID del producto como parámetro
36   borrarProducto(id: string) {
37     return this.http.delete<string>(`${this.BASE_URL}/productos/${id}`);
38   }
}

```

- Desde mi proyecto 'dcfruver-backend' instalo cors:

```

You hace 50 segundos | 1 author (You)
1 import express from 'express';
2 import router from './Routes/routes.js';
3 import { sequelize } from './Database/database.js';
4 import cors from 'cors'; ↴
5
6 //Creo una instancia de express
7 const app = express();
8
9 //Middleware
10 app.use(cors()); ↴
11
12 //Para aceptar solicitudes con formato json
13 app.use(express.json());
14
15 //Montar enrutador en app principal
16 app.use(router);
17
18 //Crear una variable port con el valor 3000
19 app.set('port',3000);
20
21 //Test a Base de datos
22 const testDB = async () => {
23   try {
    ...
  }
  catch (err) {
    console.log(err);
  }
}
24
25 //Conectar a la base de datos
26 testDB().catch((err) => {
27   console.log(`No se pudo conectar a la base de datos ${err}`);
28 });
29
30 //Run the application
31 app.listen(app.get('port'), () => {
32   console.log(`The app is running on port ${app.get('port')}`);
33 });

```

PS C:\Users\JAIRO\Downloads\Diplomado\Modulo2\dcfruver-backend> npm i cors ↴

added 2 packages, and audited 136 packages in 2s

12 packages are looking for funding  
run 'npm fund' for details

Found 0 vulnerabilities

- Desarrolle los archivos .htm , .ts y .css del componente que creé anteriormente llamado ‘pagina-principal’. En este componente, se encontrará la página principal de la tienda desde la vista del usuario, como los filtros y la lista de productos a la venta presentado en tarjetas y también su asociación con el carrito de compras. Los estilos fueron una mezcla de Bootstrap y estilos propios del archivo .css de este componente:

The screenshot shows a dual-monitor setup. The left monitor displays the 'src' folder of a project in VS Code, containing files such as 'pagina-principal.component.html', 'pagina-principal.component.ts', and 'pagina-principal.component.css'. The right monitor displays the 'app' folder, containing files like 'carrito', 'pagina-principal', and 'shared'. Both monitors show the terminal tab with the message 'Compiled successfully.' and various system icons at the bottom.

```

src > app > pagina-principal > #pagina-principal.component.css > #ovalo
1  /* Estilo para el título de la Tienda*/
2  .ovalo {
3    background: linear-gradient(to right, #318656, #3ba7be); /* Gama de color verde */
4    width: 500px; /* Ancho del óvalo */
5    height: 90px; /* Alto del óvalo */
6    border-radius: 50%; /* Hace que el contenido sea un óvalo */
7    display: flex;
8    justify-content: center;
9    align-items: center;
10   margin: 0 auto; /* Centra el óvalo horizontalmente */
11   margin-top: 50px; /* Ajusta el margen superior para centrar verticalmente */
12   box-shadow: 0 2px 4px rgba(0, 0, 0, 0.2); /* Sombra para el óvalo */
13 }

.titulo {
14   color: white; /* Color del texto en blanco */
15   font-size: 24px; /* Tamaño del texto */
16   text-align: center; /* Centra el texto dentro del óvalo */
17   text-shadow: 1px 1px 2px rgba(0, 0, 0, 0.2); /* Sombra para el texto */
18 }
19
20 /*Estilo para el color del div que contiene los filtros */
21 .filtro-verde-clarito {
22   background-color: #ac7ead; /* color verde claro */
23   padding: 10px; /* Añadimos un poco de espacio interno para mejorar el aspecto */
24   border-radius: 5px; /* Agregamos bordes redondeados para un mejor aspecto */
25 }
26
27 }
28
29

```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL GITLENS

✓ compiled successfully.

Lín. 11, col. 81 Espacios: 4 UTF-8 CRLF CSS Go Live PowerShell ESP LAA 11:09 p. m. 04/08/2023

- En el anterior componente también abre un modal que contiene la información detallada de los productos. Este modal lo programé en un componente diferente, el cual llame ProductModal, lo desarrollado en los archivos .html y .ts y la importación en el archivo "app.module.ts" de este componente se muestra a continuación:

```

src > app > product-modal > product-modal.component.html > product-modal.component.specs.ts > app.module.ts M
1  Go to component
2  <!-- Este componente despliega el modal que contiene la información detallada del producto -->
3  <div class="modal-header">
4    <h5 class="modal-title" id="productoModalLabel">{{ productoSeleccionado?.nombre }}</h5>
5    <button type="button" class="close" (click)="dismissModal()" aria-label="Close">
6      <span aria-hidden="true">&amptimes</span>
7    </button>
8  </div>
9  <div class="modal-body">
10   <img [src]="productoSeleccionado?.imgproducto" class="img-fluid mb-3" alt="{{ productoSeleccionado?.nombre }}"/>
11   <p>{{ productoSeleccionado?.descripcionproducto }}</p>
12 </div>

```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL GITLENS

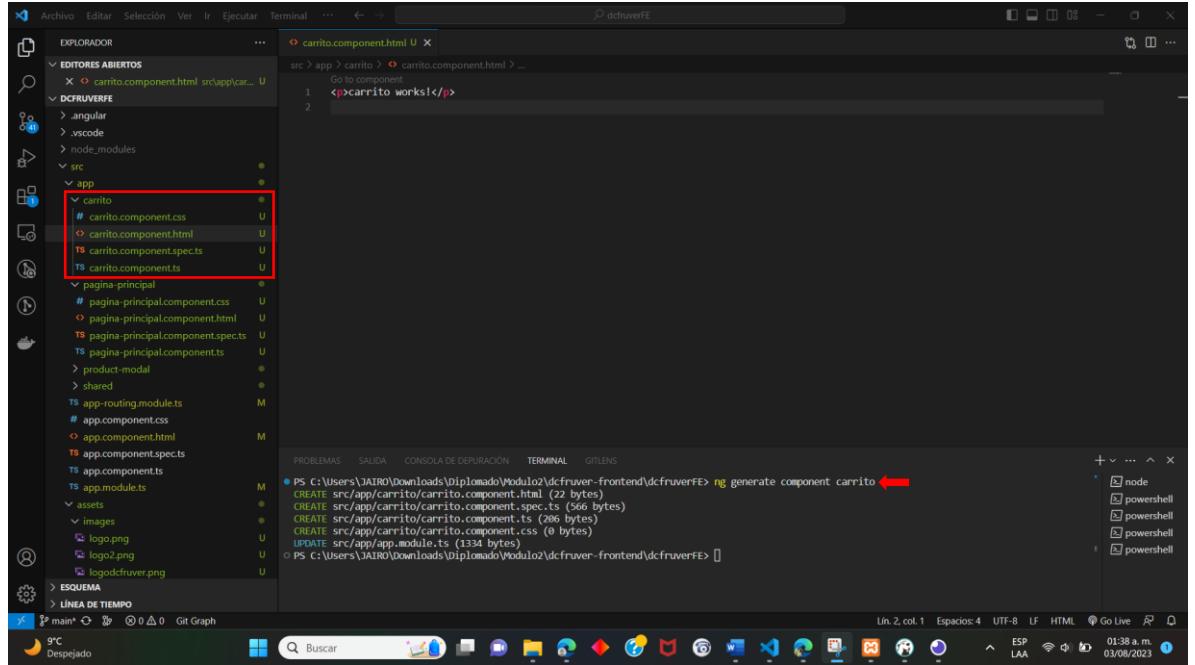
component.ts (229 bytes)  
CREATE src/app/product-modal/product-modal.component.css (0 bytes)  
UPDATE src/app/app.module.ts (917 bytes)  
PS C:\Users\Jairo\Downloads\Diplomado\Modules\dcfuruver-frontend\dcfuruverFE> ng generate component ProductModal[]

Lín. 1, col. 91 Espacios: 4 UTF-8 LF HTML Go Live PowerShell ESP LAA 05:16 p. m. 02/08/2023

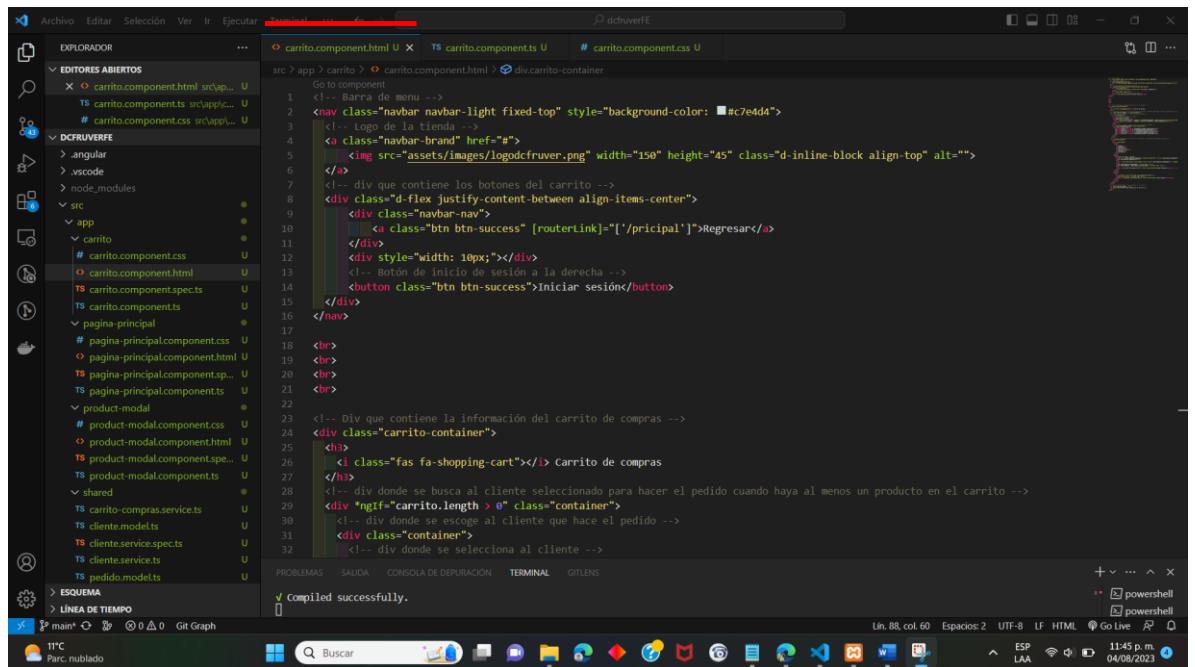
```
src > app > product-modal > product-modal.component.ts > ProductModalComponent > dismissModal  
1 import { Component, Input } from '@angular/core';  
2 import { NgbActiveModal } from '@ng-bootstrap/ng-bootstrap';  
3  
4 @Component({  
5   selector: 'app-product-modal',  
6   templateUrl: './product-modal.component.html',  
7   styleUrls: ['./product-modal.component.css']  
8 })  
9 export class ProductModalComponent {  
10   @Input() productoSeleccionado: any;  
11  
12   constructor(public activemodal: NgbActiveModal) {}  
13  
14   //Función que cierra el modal  
15   dismissModal() {  
16     this.activemodal.dismiss();  
17   }  
18 }  
  
component.ts (229 bytes)  
CREATE src/app/product-modal/product-modal.component.css (0 bytes)  
UPDATE src/app/app.module.ts (917 bytes)  
PS C:\Users\JAIRO\Downloads\Diplomado\Modulo2\dcfruver-frontend\dcfruverFE> ng generate component ProductModal
```

```
src > app > app.module.ts > AppModule  
1 import { NgModule } from '@angular/core';  
2 import { BrowserModule } from '@angular/platform-browser';  
3 import { FormsModule } from '@angular/forms';  
4 import { AppRoutingModule } from './app-routing.module';  
5 import { AppComponent } from './app.component';  
6 import { PaginaPrincipalComponent } from './pagina-principal/pagina-principal.component';  
7 import { ProductoService } from './shared/producto.service';  
8 import { NgbModule } from '@ng-bootstrap/ng-bootstrap';  
9 import { ProductModalComponent } from './product-modal/product-modal.component';  
10  
11 You have 56 minutos | 1 author (You)  
12 @NgModule({  
13   declarations: [  
14     AppComponent,  
15     PaginaPrincipalComponent,  
16     ProductModalComponent ←  
17   ],  
18   imports: [  
19     BrowserModule,  
20     AppRoutingModule,  
21     HttpClientModule,  
22     FormsModule,  
23     NgbModule  
24   ],  
25   providers: []  
})  
  
component.ts (229 bytes)  
CREATE src/app/product-modal/product-modal.component.css (0 bytes)  
UPDATE src/app/app.module.ts (917 bytes)  
PS C:\Users\JAIRO\Downloads\Diplomado\Modulo2\dcfruver-frontend\dcfruverFE> ng generate component ProductModal
```

- Creé el componente llamado CarritoComponent que mostrará el contenido del carrito que voy a llamar desde el componente página principal:



Desarrolle los archivos .html , .ts y .css. En este componente se concreta el pedido del cliente, siguiendo las sugerencias del docente el cliente se lo elige de los clientes que ya están creados en la bd y se lista los productos añadidos al carrito:



The screenshot shows a Windows desktop environment with a Microsoft Visual Studio Code (VS Code) window open. The title bar of the VS Code window reads "ddrdriverEE". The left sidebar (File Explorer) lists several project files and folders, including "EDTORES ABIERTOS", "SRC", "APP", and "CARRO". The main editor area displays a CSS file named "carrito.component.css" with the following content:

```
src/app/carrito/>#carrito.component.css>.button {
    background-color: #007B3E;
    color: white;
    border: none;
    padding: 5px 10px;
    cursor: pointer;
}

.button:hover {
    background-color: #005E2E;
}

/* Estilos para el container que contiene la información del carrito */
.carrito-container {
    margin: 20px;
    padding: 10px;
    border: 1px solid #cccccc;
    border-radius: 5px;
    background-color: #ebfcef;
}

.carrito-container h3 {
    margin-top: 0;
    margin-bottom: 10px;
    text-align: center;
}

/* Estilo para la etiqueta select-client */
.select-client {
    margin-bottom: 20px;
}

/* Estilos para la información del cliente seleccionado */

```

The status bar at the bottom of the VS Code window shows the message "Compiled successfully." The taskbar at the bottom of the screen includes icons for File, Edit, Selection, Ver, Ir, Ejecutar, Terminal, and Go Live, along with a powershell icon. The system tray shows the date and time as "04/08/2023 11:46 p.m.". The bottom right corner of the screen shows the system tray with icons for battery, signal, and network.

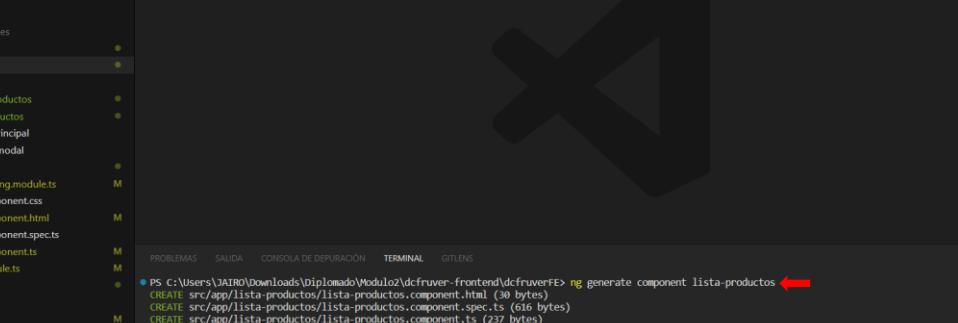
Este componente fue agregado en los archivos ‘app.module.ts’ y ‘app-routing.module.ts’, como estos archivos seguirán siendo modificados a lo largo del trabajo, pondré un pantallazo al final o se lo podrá observar completo desde el proyecto remoto de git.

- Para que sea más fácil el manejo del carrito cree el servicio ‘carrito-compras.service.ts’:

```
src > app > shared > carrito-compras.service.ts > CarritoComprasService > agregarProducto
1 import { Injectable } from '@angular/core';
2 import { ProductoModel } from './producto.model';
3
4 @Injectable({
5   providedIn: 'root'
6 })
7 export class CarritoComprasService {
8   // Lista privada de productos en el carrito con sus cantidades
9   private carrito: { producto: ProductoModel, cantidad: number }[] = [];
10
11   // Método para agregar un producto al carrito
12   agregarProducto(producto: any, cantidad: number) {
13
14     // Buscamos si el producto ya está en el carrito
15     const productoEnCarrito = this.carrito.find(item => item.producto.idproducto === producto.idproducto);
16
17     // Si el producto ya está en el carrito, aumentamos la cantidad
18     if (productoEnCarrito) {
19       productoEnCarrito.cantidad += cantidad;
20     } else {
21       // si el producto no está en el carrito, lo agregamos
22       this.carrito.push({ producto, cantidad });
23     }
24   }
25
26   // Método para obtener el carrito completo
27   obtenerCarrito() {
28     return this.carrito;
29   }
30
31   // Método para calcular el total del carrito
32   calcularTotal(): number {
33     let total = 0;
```

Nota: En este punto hice un commit con el nombre "Componentes donde se desarollo las vistas del Cliente de la tienda".

- Para el lado del Administrador de la tienda empecé creando los componentes ‘listar-productos’ y ‘editar-productos’ :



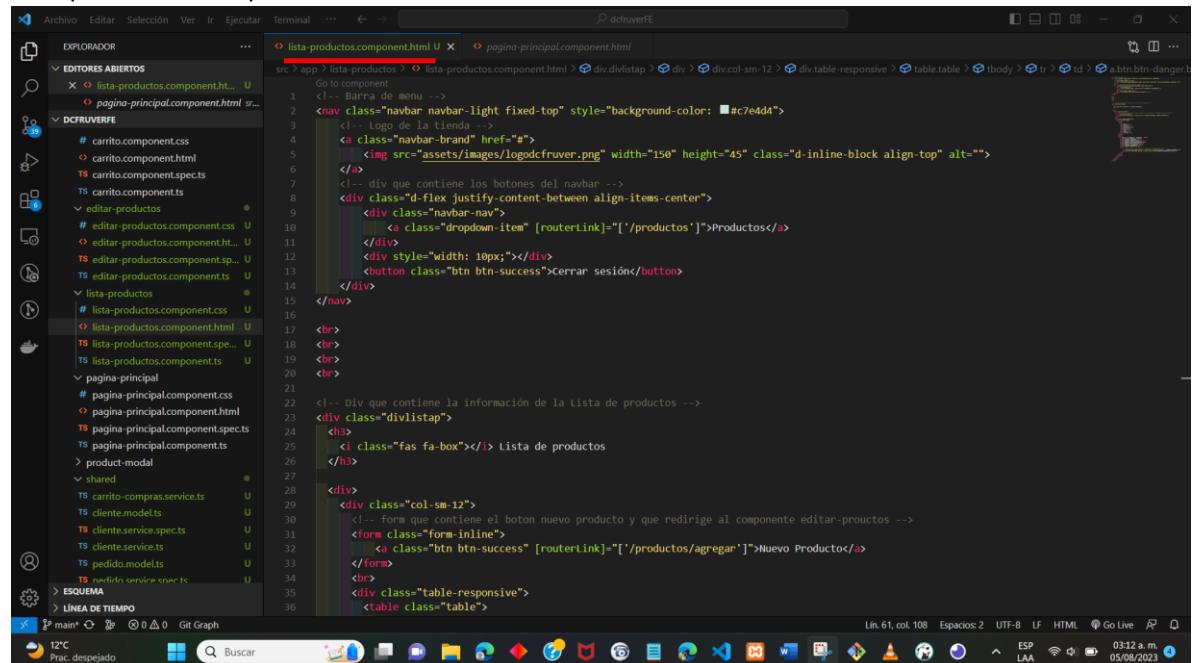
The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** On the left, it lists the project structure for "DCFRUVERFE". The "src/app" folder contains components like "carrito", "editar-productos", "lista-productos", "pagina-principal", "product-modal", and "shared".
- Terminal:** At the bottom, the terminal window displays the command "ng generate component lista-productos" being run twice. The first execution is shown with a red arrow pointing to the command line, and the second execution is shown with a red arrow pointing to the output of the command.

Estos componentes fueron agregados en los archivos 'app.module.ts' y 'app-routing.module.ts', como estos archivos seguirán siendo modificados a lo largo del trabajo, pondré un pantallazo al final o se lo podrá observar completo desde el proyecto remoto de git.

- Para desarrollar los anteriores componentes, hice uso de ‘producto.model.ts’ y el ‘producto.service.ts’ creados anteriormente y ubicados en la carpeta shared. En ‘listaproductos’ desarrolle el listado de productos existentes en la bd, en la cual se puede hacer las acciones de agregar, borrar y crear un nuevo producto. Y en el componente ‘editarproductos’, vamos a encontrar un formulario que nos va a servir para modificar un producto o para registrar uno nuevo. A continuación adjuntare los archivos .html .ts y .css de los 2 componentes:

### lista-productos.component.html:



```

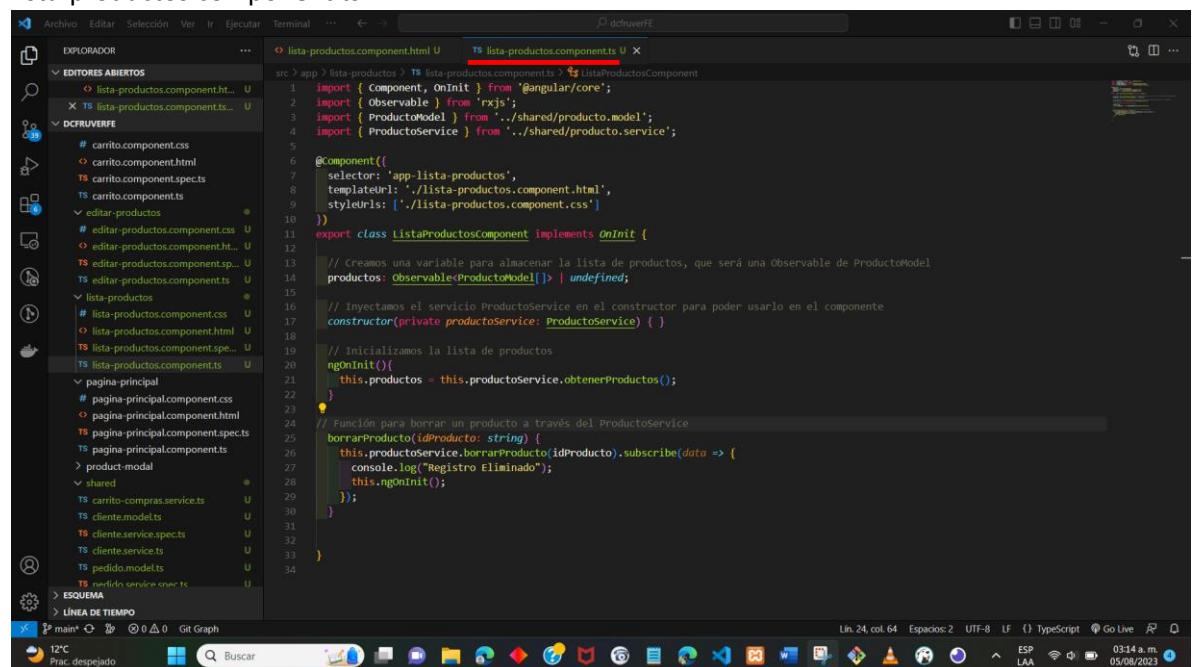
<!-- Barra de menu -->
<nav class="navbar navbar-light fixed-top" style="background-color: #c7e4d4">
  <!-- Logo de la tienda -->
  <a class="navbar-brand" href="#">
    
  </a>
  <!-- div que contiene los botones del navbar -->
  <div class="d-flex justify-content-between align-items-center">
    <a class="dropdown-item" [routerLink]="/productos">Productos</a>
  </div>
  <div style="width: 10px;"></div>
  <button class="btn btn-success">Cerrar sesión</button>
</nav>

<br>
<br>
<br>
<br>

<!-- Div que contiene la informacion de la Lista de productos -->
<div class="divlistap">
  <h3> <i class="fas fa-box"></i> Lista de productos </h3>
  <div class="col-sm-12">
    <!-- form que contiene el boton nuevo producto y que redirige al componente editar-products -->
    <form class="form-inline">
      <a class="btn btn-success" [routerLink]="/productos/agregar">Nuevo Producto</a>
    </form>
    <br>
    <div class="table-responsive">
      <table class="table">
        <thead>
          <tr>
            <th>Nombre del Producto</th>
            <th>Precio</th>
            <th>Stock</th>
            <th>Acciones</th>
          </tr>
        </thead>
        <tbody>
          <tr>
            <td>Camisa Negra DCFU</td>
            <td>10000</td>
            <td>10</td>
            <td>
              <a href="#">Ver</a>
              <a href="#">Actualizar</a>
              <a href="#">Borrar</a>
            </td>
          </tr>
        </tbody>
      </table>
    </div>
  </div>
</div>

```

### lista-productos.component.ts:



```

import { Component, OnInit } from '@angular/core';
import { Observable } from 'rxjs';
import { ProductoModel } from '../shared/producto.model';
import { ProductoService } from '../shared/producto.service';

@Component({
  selector: 'app-lista-productos',
  templateUrl: './lista-productos.component.html',
  styleUrls: ['./lista-productos.component.css']
})
export class ListaProductosComponent implements OnInit {

  // Creamos una variable para almacenar la lista de productos, que será una observable de ProductoModel
  productos: Observable<ProductoModel[]> | undefined;

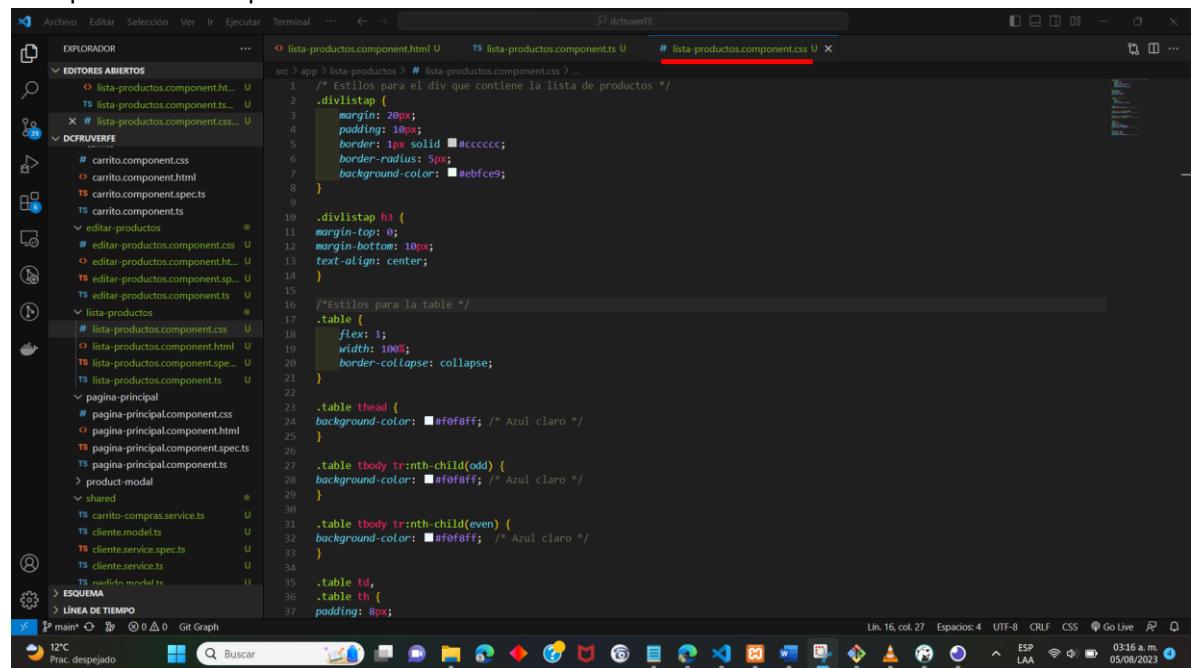
  // Inyectamos el servicio ProductoService en el constructor para poder usarlo en el componente
  constructor(private productoService: ProductoService) { }

  // Inicializamos la lista de productos
  ngOnInit(){
    this.productos = this.productoService.obtenerProductos();
  }

  // Función para borrar un producto a través del ProductoService
  borrarProducto(idProducto: string) {
    this.productoService.borrarProducto(idProducto).subscribe(data => {
      console.log("Registro Eliminado");
      this.ngOnInit();
    });
  }
}

```

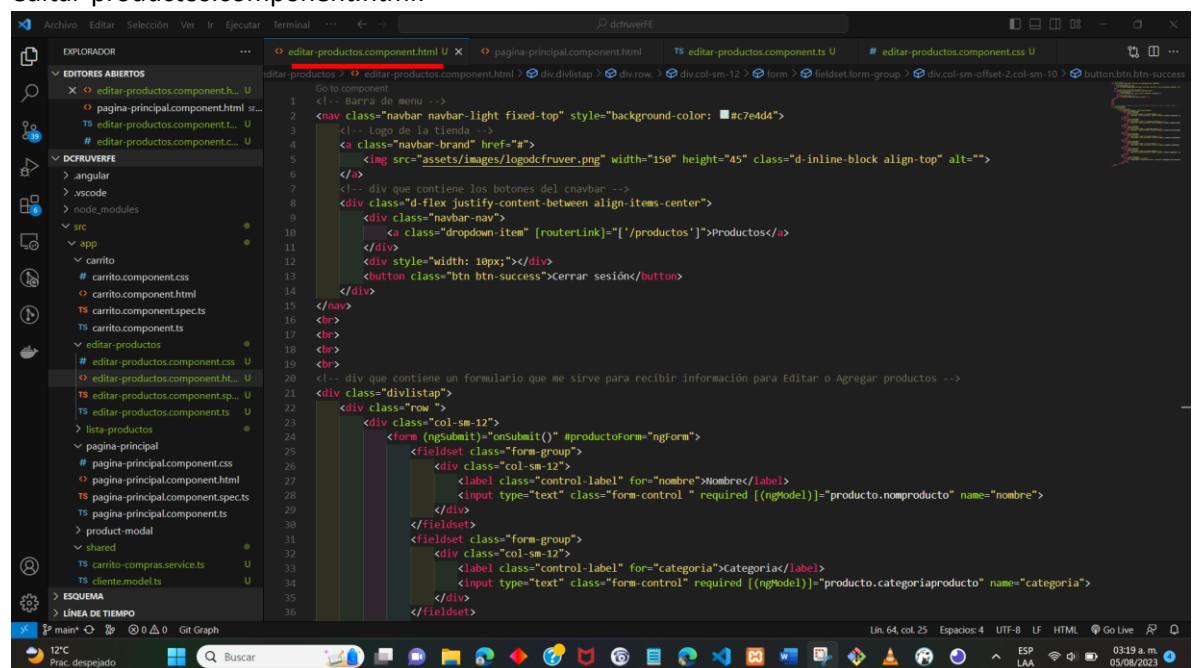
### lista-productos.component.css:



```
src > app > lista-productos > # lista-productos.component.css > ...
1  /* Estilos para el div que contiene la lista de productos */
2  .divlistap {
3    margin: 20px;
4    padding: 10px;
5    border: 1px solid #cccccc;
6    border-radius: 5px;
7    background-color: #nebfce9;
8  }
9
10 .divlistap h3 {
11   margin-top: 0;
12   margin-bottom: 10px;
13   text-align: center;
14 }
15
16 /*Estilos para la table */
17 .table {
18   flex: 1;
19   width: 100%;
20   border-collapse: collapse;
21 }
22
23 .table thead {
24   background-color: #f0faff; /* Azul claro */
25 }
26
27 .table tbody tr:nth-child(odd) {
28   background-color: #f0faff; /* Azul claro */
29 }
30
31 .table tbody tr:nth-child(even) {
32   background-color: #f0faff; /* Azul claro */
33 }
34
35 .table td,
36 .table th {
37   padding: 8px;

```

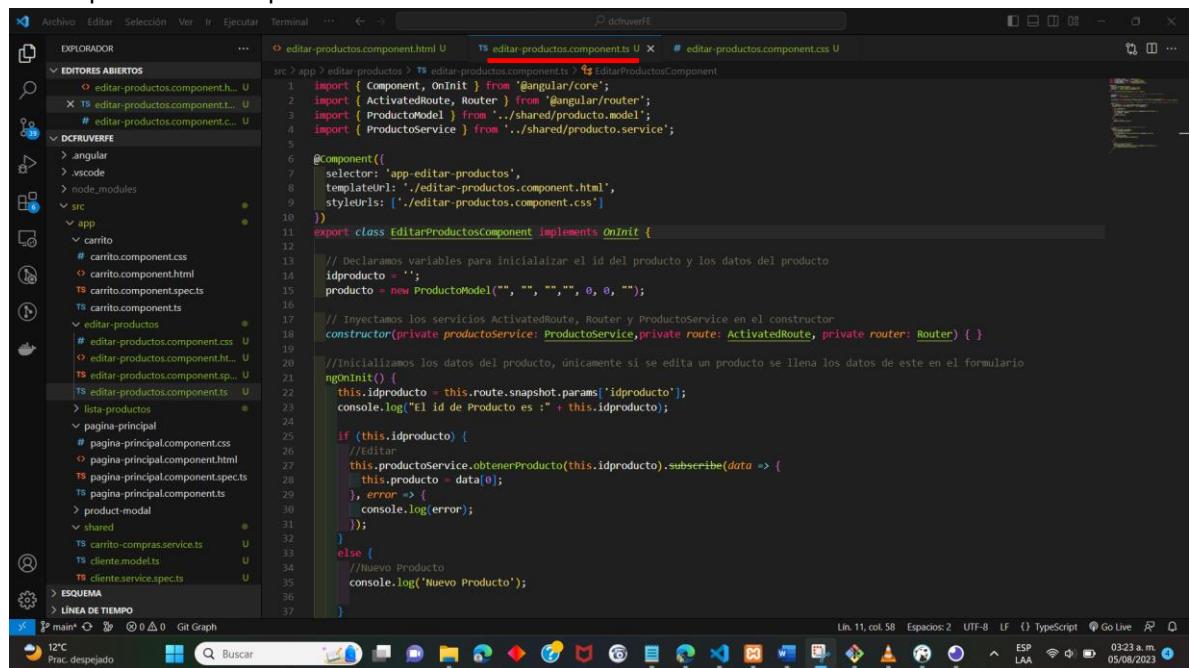
### editar-productos.component.html:



```
src > editar-productos > editar-productos.component.html > pagina-principal.component.html > editar-productos.component.ts > # editar-productos.component.css ...
1  <!-- Barra de menu -->
2  <nav class="navbar navbar-light fixed-top" style="background-color: #c7e4d4">
3    <!-- Logo de la tienda -->
4    <a class="navbar-brand" href="#">
5      
6    </a>
7    <!-- div que contiene los botones del crábar -->
8    <div class="d-flex justify-content-between align-items-center">
9      <div class="navabar-nav">
10        <a class="dropdown-item" [routerLink]="/productos">Productos</a>
11      </div>
12      <div style="width: 10px;"></div>
13      <button class="btn btn-success">cerrar sesión</button>
14    </div>
15  </nav>
16  <br>
17  <br>
18  <br>
19  <br>
20  <!-- div que contiene un formulario que me sirve para recibir información para Editar o Agregar productos -->
21  <div class="divlistap">
22    <div class="row">
23      <div class="col-sm-12">
24        <form (ngSubmit)="onSubmit()" #productoForm="ngForm">
25          <fieldset class="form-group">
26            <div class="col-sm-12">
27              <label class="control-label" for="nombre">Nombre</label>
28              <input type="text" class="form-control" required [(ngModel)]="producto.nombreproducto" name="nombre">
29            </div>
30          </fieldset>
31          <fieldset class="form-group">
32            <div class="col-sm-12">
33              <label class="control-label" for="categoria">Categoria</label>
34              <input type="text" class="form-control" required [(ngModel)]="producto.categoriaciaproducto" name="categoria">
35            </div>
36          </fieldset>

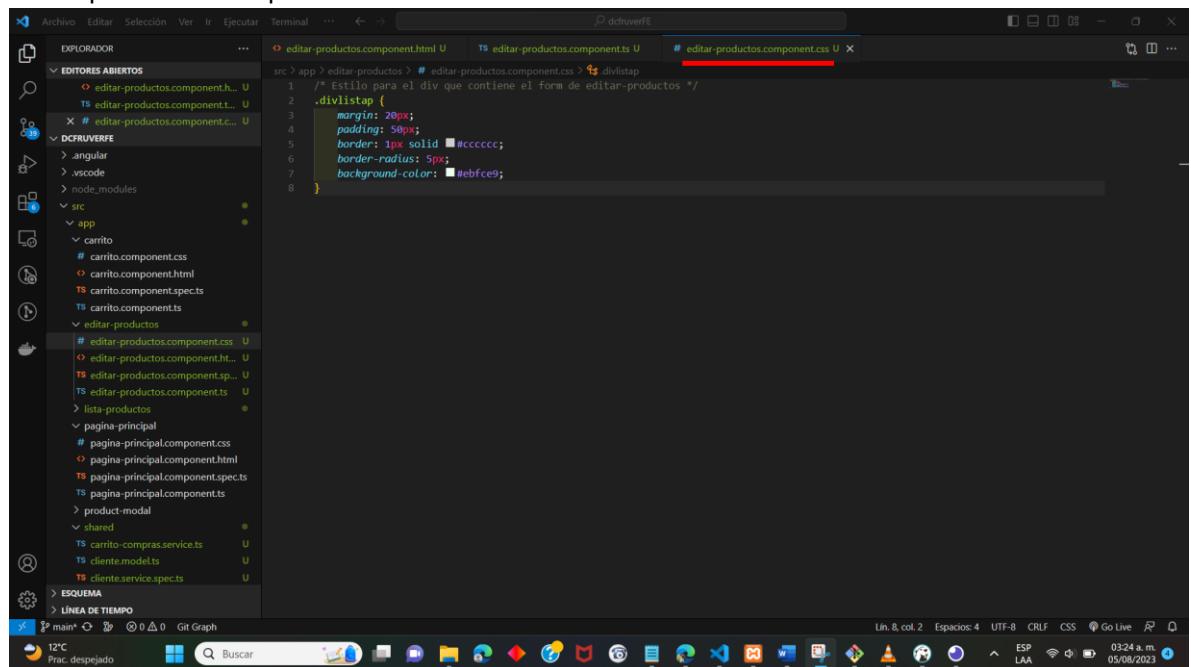
```

### editar-productos.component.ts:



```
src > app > editar-productos > TS editar-productos.component.ts U # editar-productos.component.ts U
1 import { Component, OnInit } from '@angular/core';
2 import { ActivatedRoute, Router } from '@angular/router';
3 import { ProductoModel } from './shared/producto.modelo';
4 import { ProductoService } from './shared/producto.service';
5
6 @Component({
7   selector: 'app-editar-productos',
8   templateUrl: './editar-productos.component.html',
9   styleUrls: ['./editar-productos.component.css']
10 })
11 export class EditarProductosComponent implements OnInit {
12
13   // Declaramos variables para inicializar el id del producto y los datos del producto
14   idproducto = '';
15   producto = new ProductoModel("", "", "", "", 0, 0, "");
16
17   // Inyectamos los servicios ActivatedRoute, Router y ProductoService en el constructor
18   constructor(private productService: ProductoService, private route: ActivatedRoute, private router: Router) { }
19
20   //Inicializamos los datos del producto, únicamente si se edita un producto se llenan los datos de este en el formulario
21   ngOnInit() {
22     this.idproducto = this.route.snapshot.params['idproducto'];
23     console.log("El id de Producto es :" + this.idproducto);
24
25     if (this.idproducto) {
26       //Editar
27       this.productService.obtenerProducto(this.idproducto).subscribe(data => {
28         this.producto = data[0];
29       }, error => {
30         console.log(error);
31       });
32     } else {
33       //Nuevo Producto
34       console.log('Nuevo Producto');
35     }
36   }
37 }
```

### editar-productos.component.css:



```
src > app > editar-productos > # editar-productos.component.css U # editar-productos.component.css U
1 /* Estilo para el div que contiene el form de editar-productos */
2 .divlistap {
3   margin: 20px;
4   padding: 50px;
5   border: 1px solid black;
6   border-radius: 5px;
7   background-color: #ebfcef;
8 }
```

- Creé el componente ‘lista-pedidos’, en el cuál el Administrador podrá ver la lista de pedidos y podrá confirmar o rechazar los pedidos:

```
PS C:\Users\Jairo\Downloads\diplomado\Modulo2\dcfruver-frontend> ng generate component lista-pedidos
CREATE src/app/lista-pedidos/lista-pedidos.component.html (28 bytes)
CREATE src/app/lista-pedidos/lista-pedidos.component.spec.ts (602 bytes)
CREATE src/app/lista-pedidos/lista-pedidos.component.ts (229 bytes)
CREATE src/app/lista-pedidos/lista-pedidos.component.css (0 bytes)
UPDATE src/app/app.module.ts (1770 bytes)
```

### lista-pedidos.component.html:

```
<!-- Barra de menu -->
<nav class="navbar navbar-light fixed-top" style="background-color: #c7e4d4">
  <!-- Logo de la tienda -->
  <a class="navbar-brand" href="#">
    
  </a>
  <!-- div que contiene los botones del navbar -->
  <div class="d-flex justify-content-between align-items-center">
    <div class="navbar-nav">
      <a class="dropdown-item" [routerLink]="/pedidos">Pedidos</a>
    </div>
    &nbsp;
    &nbsp;
    <div class="navbar-nav">
      <a class="dropdown-item" [routerLink]="/productos">Productos</a>
    </div>
    &nbsp;
    &nbsp;
    <div style="width: 10px;"></div>
    <button class="btn btn-success">Cerrar sesión</button>
  </div>
</nav>
```

### lista-pedidos.component.ts:

```
src > app > lista-pedidos > lista-pedidos.component.ts U
1 import { Component, OnInit } from '@angular/core';
2 import { PedidoModel } from './shared/pedido.model';
3 import { PedidoService } from './shared/pedido.service';
4 import { Observable } from 'rxjs';
5
6 @Component({
7   selector: 'app-lista-pedidos',
8   templateUrl: './lista-pedidos.component.html',
9   styleUrls: ['./lista-pedidos.component.css']
10 })
11 export class ListaPedidosComponent implements OnInit {
12
13   // Creamos una variable para almacenar la lista de pedidos, que será una Observable de PedidoModel
14   pedidos: Observable<PedidoModel[]> | undefined;
15
16   // Inyectamos el servicio PedidoService en el constructor para poder usarlo en el componente
17   constructor(private pedidoService: PedidoService) {}
18
19   // Inicializamos la lista de pedidos
20   ngOnInit() {
21     this.pedidos = this.pedidoService.obtenerPedidos();
22   }
23
24   // Función para obtener el texto del estado para que sea entendible para el usuario
25   getEstadoTexto(estado: string): string {
26     switch (estado) {
27       case 'C':
28         return 'Confirmado';
29       case 'P':
30         return 'Pendiente';
31       case 'R':
32         return 'Rechazado';
33     }
34   }
35 }
```

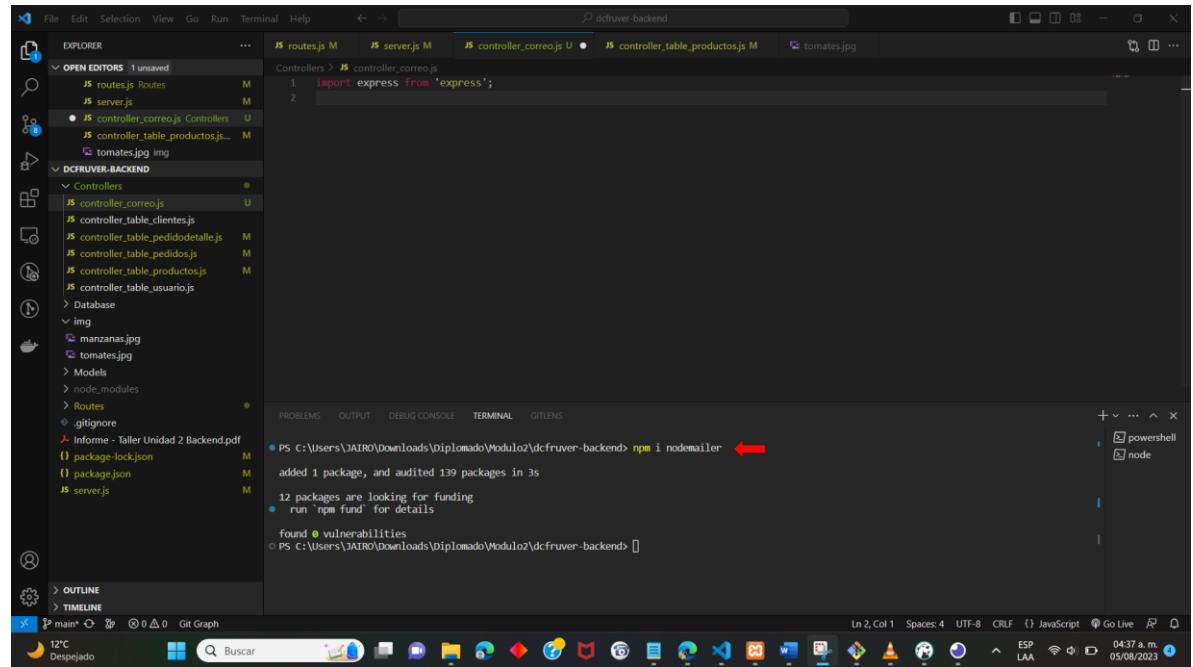
### lista-pedidos.component.css:

```
src > app > lista-pedidos > lista-pedidos.component.css U
1 /* Estilos para el div que contiene la lista de pedidos */
2 .divlistap {
3   margin: 20px;
4   padding: 10px;
5   border: 1px solid #cccccc;
6   border-radius: 5px;
7   background-color: #ebeff9;
8 }
9
10 .divlistap h3 {
11   margin-top: 0;
12   margin-bottom: 10px;
13   text-align: center;
14 }
15
16 /* Estilos para la table */
17 .table {
18   flex: 1;
19   width: 100%;
20   border-collapse: collapse;
21 }
22
23 .table thead {
24   background-color: #f0f0ff; /* Azul claro */
25 }
26
27 .table tbody tr:nth-child(odd) {
28   background-color: #f0f0ff; /* Azul claro */
29 }
30
31 .table tbody tr:nth-child(even) {
32   background-color: #f0f0ff; /* Azul claro */
33 }
```

Nota: En este punto hice un commit en mi repositorio remoto de los componentes de las vistas del Administrador de la tienda llamado "Componentes donde se desarrollo las vistas del Administrador de la tienda".

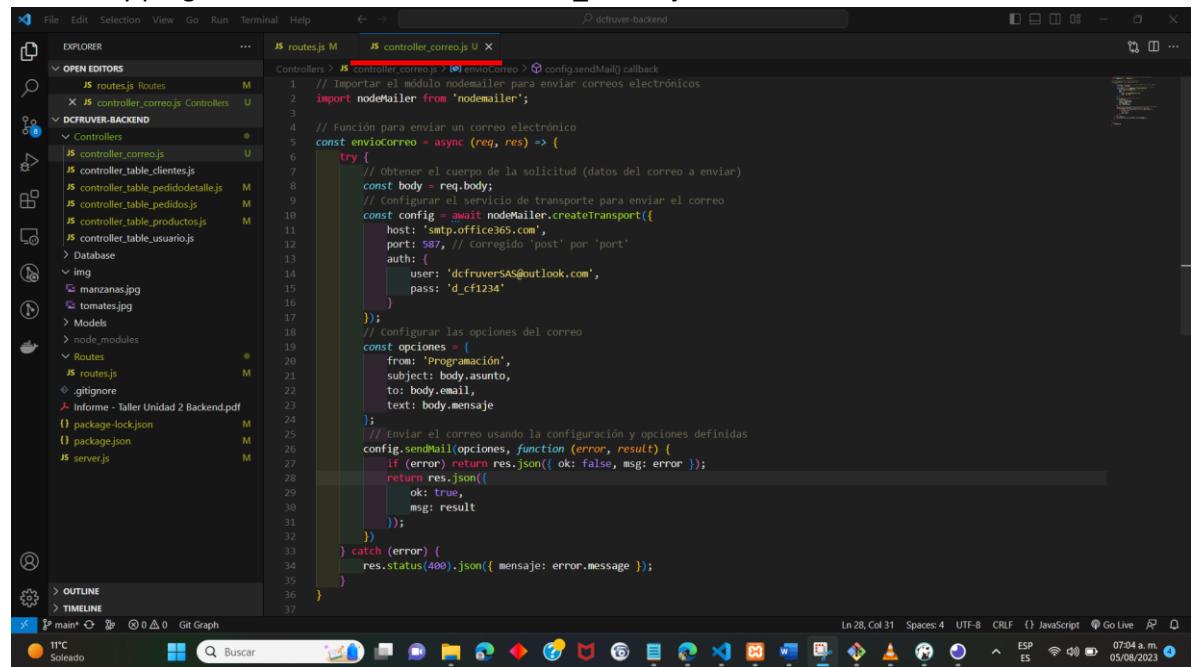
- Para implementar la función que se envíe un correo cuando se confirme o rechace un pedido, primero lo implemente desde el ‘dcfruver-backend’, usando nodemailer. Creé el archivo ‘controller\_correo.js’ y lo añadí a las rutas del archivo ‘routes.js’ :

Instalación de nodemailer en el ‘dcfruver-backend’:



```
PS C:\Users\JATIRO\Downloads\Diplomado\Modules\Modulo2\dcfruver-backend> npm i nodemailer
added 1 package, and audited 139 packages in 3s
12 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities
PS C:\Users\JATIRO\Downloads\Diplomado\Modules\Modulo2\dcfruver-backend>
```

Creación y programación del archivo ‘controller\_correo.js’:



```
const config = require('../config');
const nodemailer = require('nodemailer');

const enviarcorreo = async (req, res) => {
    try {
        // Obtener el cuerpo de la solicitud (datos del correo a enviar)
        const body = req.body;
        // Configurar el servicio de transporte para enviar el correo
        const config = await nodemailer.createTransport({
            host: 'smtp.office365.com',
            port: 587, // corregido 'post' por 'port'
            auth: {
                user: 'dcfruverSA@outlook.com',
                pass: 'd_cf1234'
            }
        });
        // Configurar las opciones del correo
        const opciones = {
            from: 'Programación',
            subject: body.asunto,
            to: body.email,
            text: body.mensaje
        };
        // Enviar el correo usando la configuración y opciones definidas
        config.sendMail(opciones, function (error, result) {
            if (error) return res.json({ ok: false, msg: error });
            return res.json({
                ok: true,
                msg: result
            });
        })
    } catch (error) {
        res.status(400).json({ mensaje: error.message });
    }
}
```

## Archivo 'routes.js' :

```

routes.js M JS controller_correo.js U
Routes > JS routes.js ...
33 //rutas para las consultas de la tabla clientes
34
35 router.get('/clientes', getClientes);
36 router.post('/clientes', postClientes);
37 router.put('/clientes/:cedulacliente', putClientes);
38 router.delete('/clientes/:cedulacliente', deleteClientes);
39 router.get('/clientes/:cedulacliente', getCliente);
40
41 //rutas para las consultas de la tabla pedidos
42
43 router.get('/pedidos', getPedidos);
44 router.post('/pedidos', postPedidos);
45 router.put('/pedidos/:idpedido', putPedidos);
46 router.delete('/pedidos/:idpedido', deletePedidos);
47 router.get('/pedidos', getPedidosPendientes);
48 router.get('/pedidos/:idpedido', getEstadoPedido);
49 router.get('/pedidos/:idpedido', putEstadoRechazar);
50
51 //rutas para las consultas de la tabla pediodetalles
52
53 router.get('/pediodetalles', getPedidoDetalle);
54 router.get('/pediodetalles/:id', getPedidoDetalleId);
55 router.post('/pediodetalles', postPedidoDetalle);
56 router.put('/pediodetalles/:id', putPedidoDetalle);
57 router.delete('/pediodetalles/:id', deletePedidoDetalle);
58
59 router.post('/envio', enviarCorreo);
60
61 export default router;
62

```

- Desde mi proyecto 'dcfruverFE', creé un modelo para enviar el correo llamado 'CorreoModel', que tiene todos los parámetros del correo que definí en el backend:

```

correo.model.ts M
export class CorreoModel {
  constructor(public email: string, public asunto: string, public mensaje:string) {
  }
}

```

- Después creé un servicio llamado ‘correo’, para implementar la comunicación con el backend y efectuar el envío del correo:

The screenshot shows the VS Code interface with the following details:

- Explorador (Explorer):** Shows the project structure with files like `lista-pedidos.component.html`, `lista-pedidos.component.ts`, `correo.model.ts`, and `app.module.ts`.
- Editor:** Displays the `app.module.ts` file with code related to the `CarritoComprasService`. A red arrow points to the command in the terminal.
- Terminal:** Shows the command `ng generate service shared/correo` being run in the PowerShell terminal.

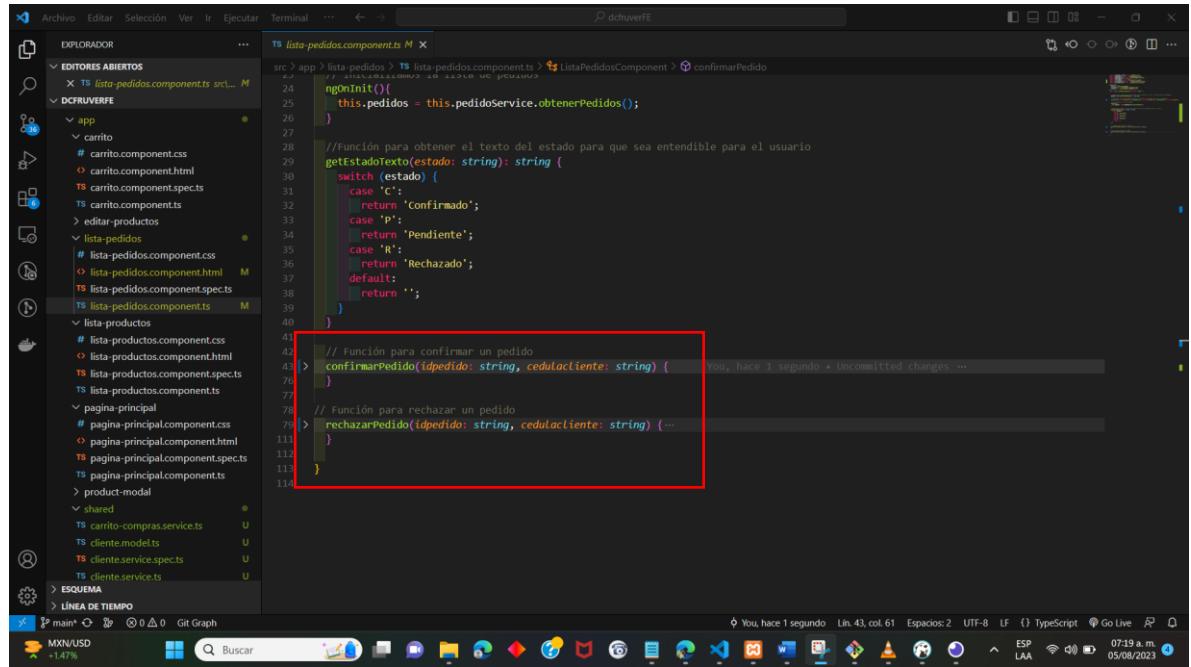
The screenshot shows the Visual Studio Code interface with the following details:

- Explorador (Explorer View):** Shows the project structure with files like 'correo.service.ts', 'pagina-principal.component.ts', 'pagina-principal.component.html', etc.
- Code Editor:** Displays the file 'correo.service.ts' containing the following code:

```
src > app > shared > TS correo.service.ts
1 import { HttpClient } from '@angular/common/http';
2 import { Injectable } from '@angular/core';
3 import { CorreoModel } from './correo.model';
4
5 @Injectable({
6   providedIn: 'root'
7 })
8 export class CorreoService {
9
10
11   // Definimos la URL base del servidor donde se encuentra el servicio de enviar el correo
12   BASE_URL = 'http://localhost:3000';
13
14   // En el constructor inyectamos el servicio HttpClient
15   constructor(private http:HttpClient) {}
16
17   //Función envia una solicitud HTTP POST al servidor para enviar un correo electrónico
18   enviarCorreo(correo: CorreoModel) {
19     return this.http.post<(req: String)>(`${this.BASE_URL}/envio`, correo);
20   }
21 }
22
```

The status bar at the bottom shows: Lin. 22, col. 1 Espacios: 2 UTF-8 LF {} TypeScript Go Live

- Implementé la acción de enviar correos desde el componente lista-pedidos, en la cuál va a activar la función de enviar el correo cada que se confirma o rechaza un pedido:



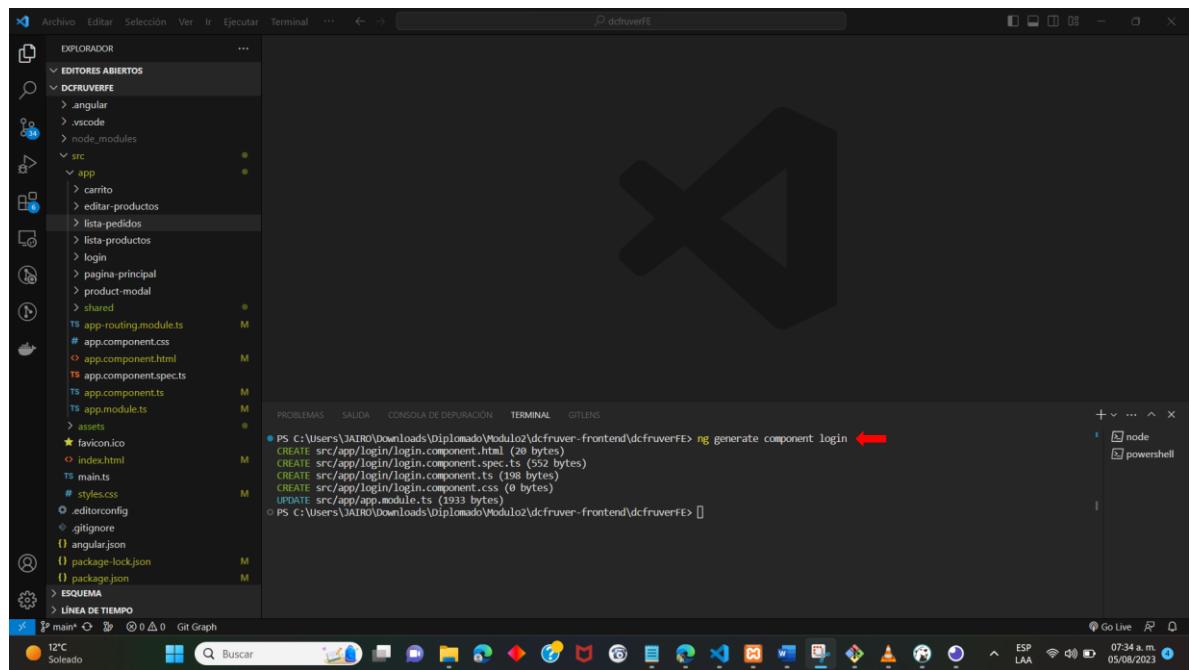
```

src > app > lista-pedidos > lista-pedidos.component.ts > ListaPedidosComponent > confirmarPedido
24   ngOnInit() {
25     this.pedidos = this.pedidoService.obtenerPedidos();
26   }
27
28   // Función para obtener el texto del estado para que sea entendible para el usuario
29   getEstadoTexto(estado: string): string {
30     switch (estado) {
31       case 'C':
32         return 'Confirmado';
33       case 'P':
34         return 'Pendiente';
35       case 'R':
36         return 'Rechazado';
37       default:
38         return '';
39     }
40   }
41
42   // Función para confirmar un pedido
43   confirmarPedido(idpedido: string, cedulacliente: string) {
44   }
45
46   // Función para rechazar un pedido
47   rechazarPedido(idpedido: string, cedulacliente: string) {
48   }
49
50 }

```

Nota: hice un coommit en mi repositorio remoo de la actualización del componente lista-pedido con el nombre "Cambios en el componente lista-pedidos para implementar el envio de correo al cliente".

- Para programar el inicio de sesión para el administrador, primero creé el componente 'login':



```

PS C:\Users\Jairo\Downloads\Diplomado\Modulo2\dcfruver-frontend\dcfruverFE> ng generate component login
CREATE src/app/login/login.component.html (28 bytes)
CREATE src/app/login/login.component.spec.ts (552 bytes)
CREATE src/app/login/login.component.ts (198 bytes)
CREATE src/app/login/login.component.css (0 bytes)
UPDATE src/app/app.module.ts (193 bytes)

```

- Programe los archivos .html y .ts del anterior componente:

## login.component.html:

## login.component.ts:

The screenshot shows a Windows desktop environment. In the foreground, a terminal window is open with the command `ddruler`. The terminal output shows the following:

```
src > app > login > login.component.html U TS autenticacion.service.ts U TS login.component.ts U
1 import { Component, OnInit } from '@angular/core';
2 import { FormBuilder, FormGroup, Validators } from '@angular/forms';
3 import { AutenticacionService } from '../shared/autenticacion.service';
4 import { Router } from '@angular/router';
5
6 @Component({
7   selector: 'app-login',
8   templateUrl: './login.component.html',
9   styleUrls: ['./login.component.css']
10 })
11 export class LoginComponent implements OnInit {
12
13   // Declaración de propiedad para el formulario
14   public myForm!: FormGroup;
15
16   // Declaración de propiedad para el formulario
17   constructor(private fb: FormBuilder, private autenticacionService: AutenticacionService, private router: Router) {}
18
19   // Método que se ejecuta al inicializar el componente
20   ngOnInit(): void {
21     this.myForm = this.createMyForm();
22   }
23
24   // Método para crear el formulario con validaciones
25   private createMyForm(): FormGroup {
26     return this.fb.group({
27       usuario: ['', [Validators.required]],
28       password: ['', [Validators.required]]
29     });
30   }
31
32   PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL GITLENS
33   4 unchanged chunks
34
35 Build at: 2023-08-05T19:05:47.517Z - Hash: e80f7854c486afad - Time: 519ms
36
37 Compiled successfully.
```

In the background, a file explorer window is visible, showing the project structure. It includes files like `login.component.html`, `autenticacion.service.ts`, `login.component.ts`, `pagina-principal.component.css`, `pagina-principal.component.html`, `pagina-principal.component.spec.ts`, `pagina-principal.component.ts`, `product-modal.component.css`, `product-modal.component.html`, `product-modal.component.spec.ts`, `product-modal.component.ts`, and several shared services and models. The `pagina-principal` folder is expanded, showing its sub-components and specifications.

Este componente fue agregado en los archivos ‘app.module.ts’ y ‘app-routing.module.ts’, como estos archivos seguirán siendo modificados a lo largo del trabajo, pondré un pantallazo al final o se lo podrá observar completo desde el proyecto remoto de git.

- Para manejar de forma más sencilla en inicio de sesión, cree el servicio 'autenticacion.service.ts':

Nota: En este punto hice un commit en mi repositorio remoto llamado "Inicio de sesion".

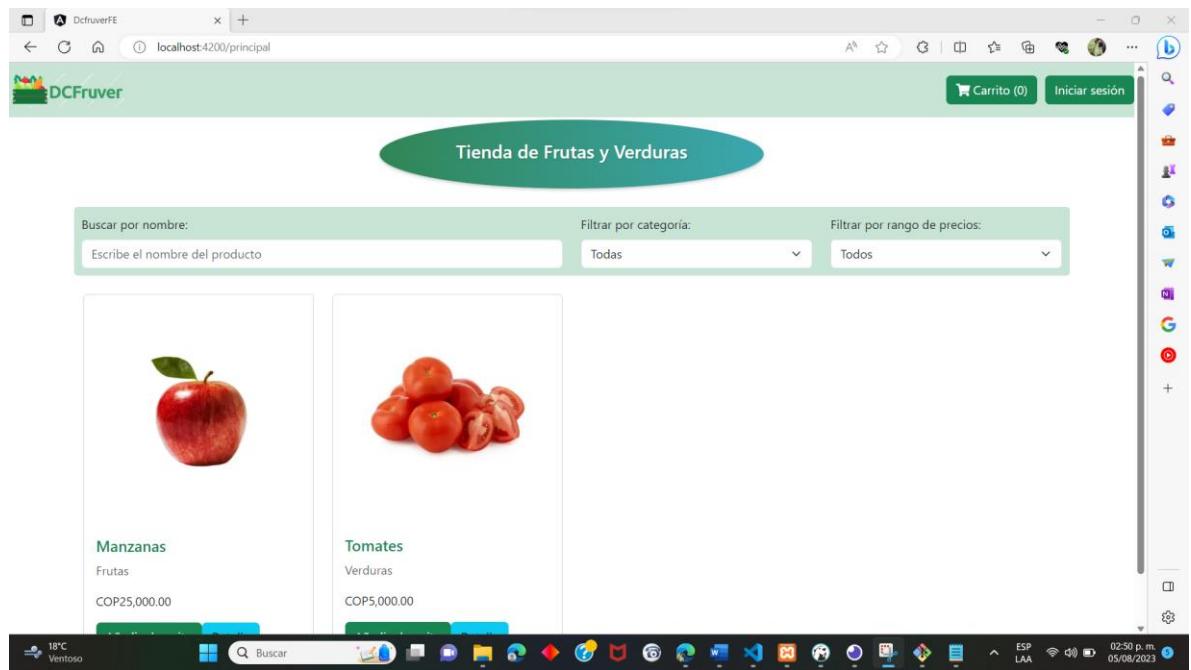
- Finalmente los archivos ‘app-routing.module.ts’ y ‘app.module.ts’ quedaron así:

```
1 import { NgModule } from '@angular/core';
2 import { BrowserModule } from '@angular/platform-browser';
3 import { HttpClientModule } from '@angular/common/http';
4 import { FormsModule, ReactiveFormsModule } from '@angular/forms';
5 import { AppRoutingModule } from './app-routing.module';
6 import { AppComponent } from './app.component';
7 import { PaginaPrincipalComponent } from './pagina-principal/pagina-principal.component';
8 import { ProductosService } from './shared/producto.service';
9 import { NgbModule } from '@ng-bootstrap/ng-bootstrap';
10 import { ProductModalComponent } from './product-modal/product-modal.component';
11 import { UsuarioService } from './shared/usuario.service';
12 import { ClienteService } from './shared/cliente.service';
13 import { PedidoService } from './shared/pedido.service';
14 import { PedidoDetalleService } from './shared/pedidodetalle.service';
15 import { CarritoComponent } from './carrito/carrito.component';
16 import { ListaProductosComponent } from './lista-productos/lista-productos.component';
17 import { EditarProductosComponent } from './editar-productos/editar-productos.component';
18 import { ListaPedidosComponent } from './lista-pedidos/lista-pedidos.component';
19 import { CorreoService } from './shared/correo.service';
20 import { LoginComponent } from './login/login.component';
21
22
23 @NgModule({
24   declarations: [
25     AppComponent,
26     PaginaPrincipalComponent,
27     ProductModalComponent,
28     CarritoComponent,
29     ListaProductosComponent,
30     EditarProductosComponent,
31     ListaPedidosComponent,
32     LoginComponent
33   ],
34   imports: [
35     BrowserModule,
36     AppRoutingModule,
37     HttpClientModule,
38     FormsModule,
39     NgbModule,
40     ReactiveFormsModule
41   ],
42   providers: [
43     ProductosService,
44     UsuarioService,
45     ClienteService,
46     PedidoService,
47     PedidoDetalleService,
48     CorreoService
49   ]
50 })
51 export class AppModule {
```

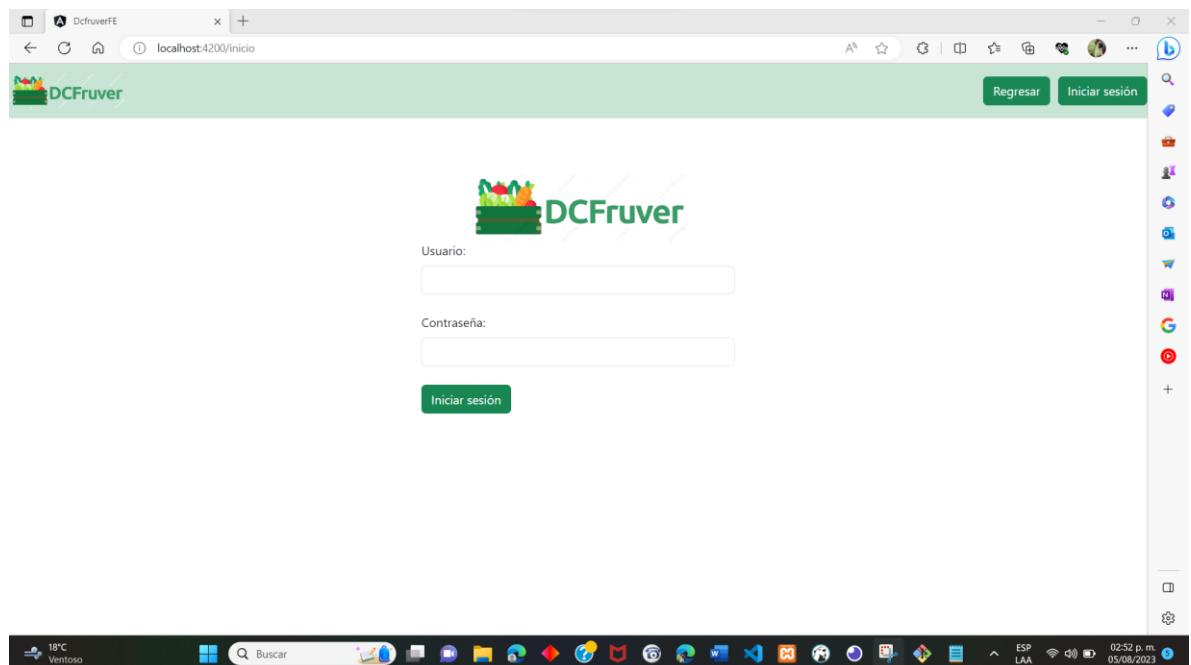
Nota: En este punto hice un commit en mi repositorio remoto llamado "app-routing y app-module".

## Prueba del funcionamiento del Frontend

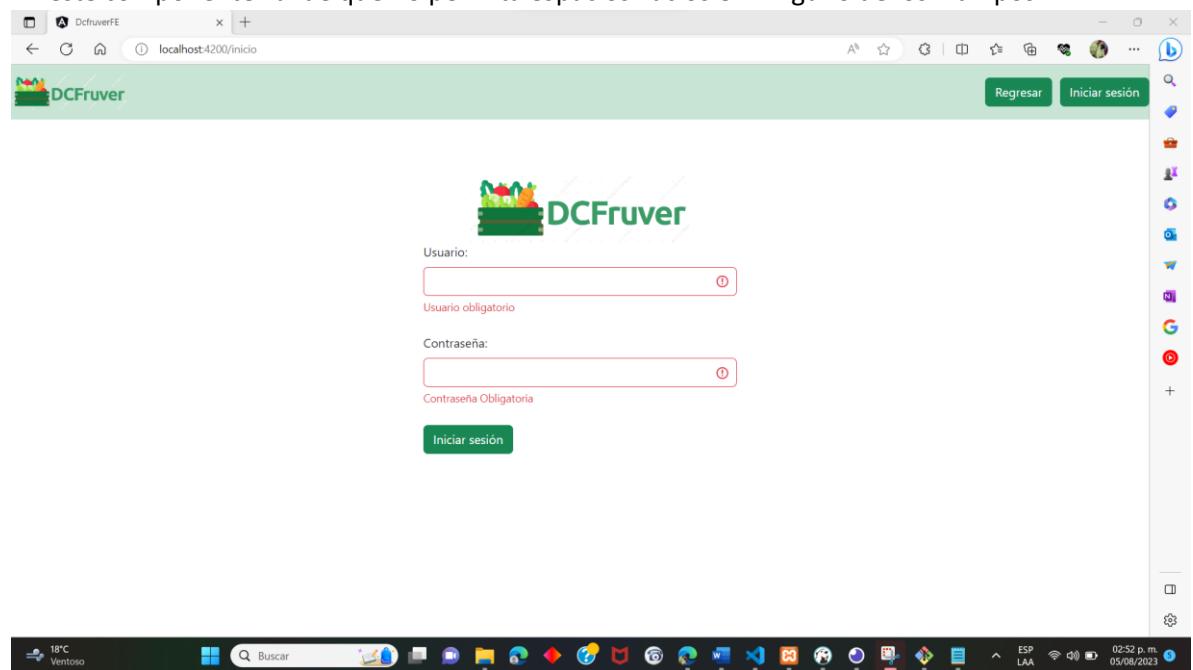
- La primera ruta que abre al ejecutar el proyecto es '/principal':



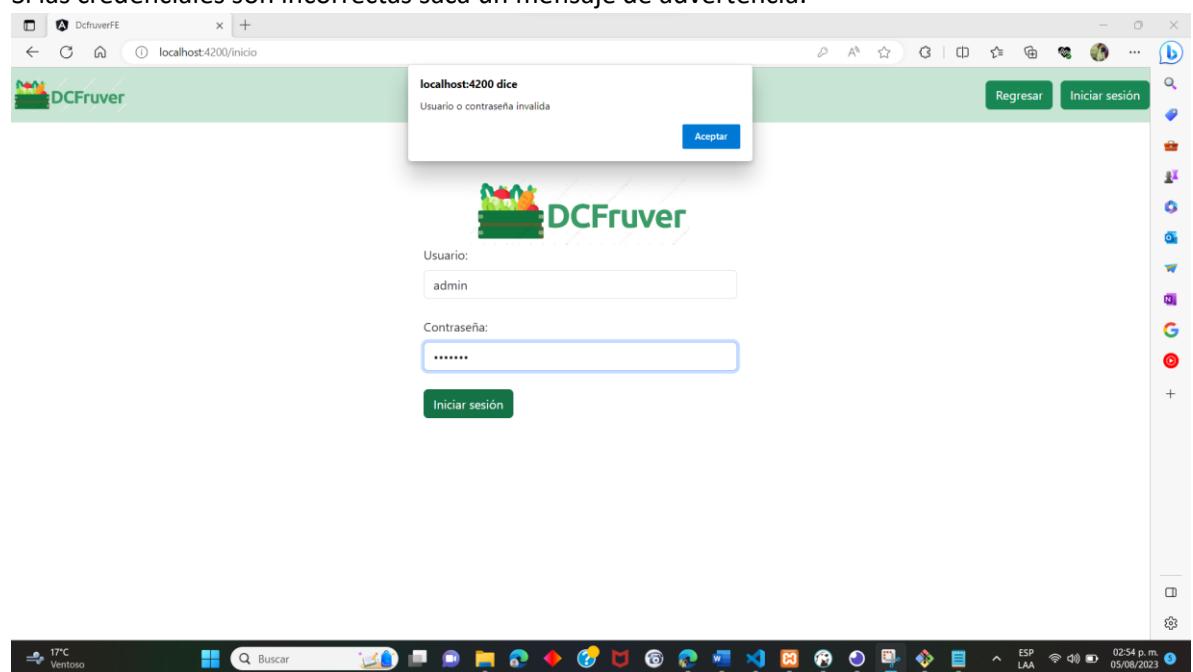
- Despues le di click en el botón de Iniciar sesión ubicado en el menú superior, el cual dirige a la Ruta '/inicio':



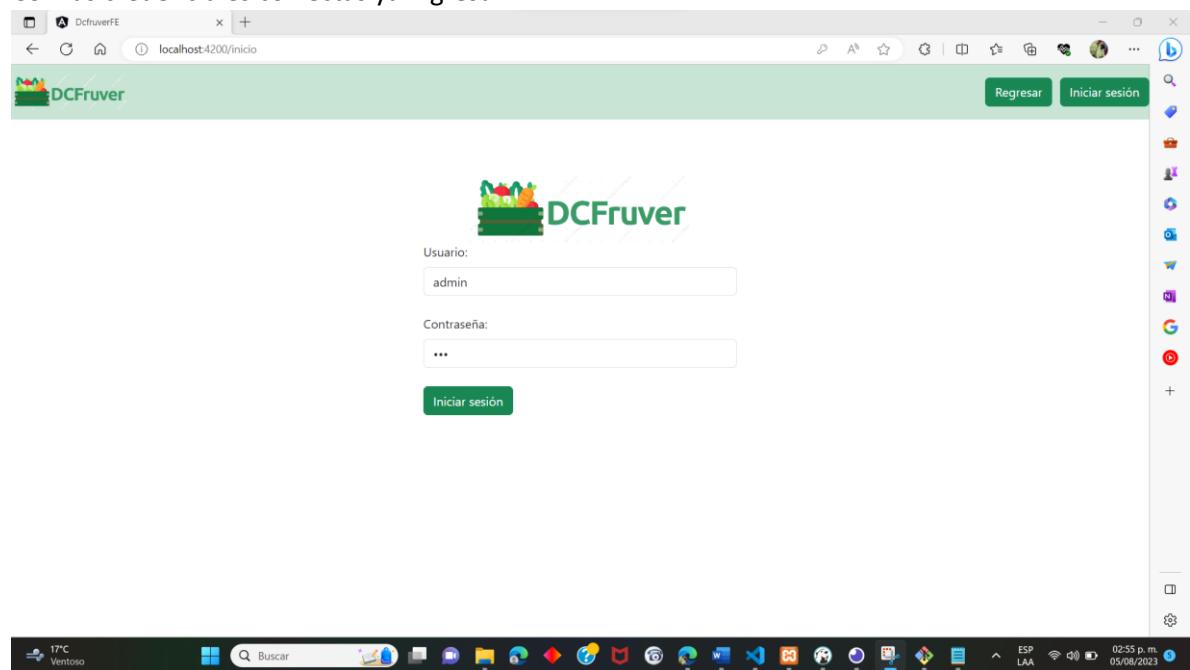
En este componente válido que no permita espacios vacíos en ninguno de los 2 campos:



Si las credenciales son incorrectas saca un mensaje de advertencia:



Con las credenciales correctas ya ingresa:



- Despues de iniciar sesión se redirige a la ruta '/pedidos', en la cual se puede confirmar o rechazar pedidos:

A screenshot of a web browser window titled 'DcfruverFE' with the URL 'localhost:4200/pedidos'. The page title is 'Listado de pedidos'. It displays a table with 12 rows of order information. The columns are: Id, Cedula Cliente, Fecha Pedido, Total, Estado, and Acciones. The 'Acciones' column contains two buttons for each row: 'Confirmar' (green) and 'Rechazar' (red). The table rows are:

Id	Cedula Cliente	Fecha Pedido	Total	Estado	Acciones
1	1111	2023-07-26	30000	Confirmado	<button>Confirmar</button> <button>Rechazar</button>
2	1111	2023-07-28	65000	Rechazado	<button>Confirmar</button> <button>Rechazar</button>
4	1111	2023-08-03	30000	Confirmado	<button>Confirmar</button> <button>Rechazar</button>
5	1111	2023-08-03	30000	Rechazado	<button>Confirmar</button> <button>Rechazar</button>
6	1111	2023-08-03	30000	Rechazado	<button>Confirmar</button> <button>Rechazar</button>
7	2222	2023-08-03	25000	Rechazado	<button>Confirmar</button> <button>Rechazar</button>
8	2222	2023-08-03	25000	Confirmado	<button>Confirmar</button> <button>Rechazar</button>
9	2222	2023-08-03	25000	Confirmado	<button>Confirmar</button> <button>Rechazar</button>
10	1111	2023-08-03	5000	Confirmado	<button>Confirmar</button> <button>Rechazar</button>
11	1111	2023-08-03	30000	Rechazado	<button>Confirmar</button> <button>Rechazar</button>
12	1111	2023-08-03	5000	Confirmado	<button>Confirmar</button> <button>Rechazar</button>

The browser's toolbar and taskbar are visible at the bottom.

- Dándole click en productos en el menú superior, nos redirige a la ruta '/productos' donde podemos ver la lista de los productos actuales, pero también podemos acceder a las funciones de editar, agregar o borrar algún producto:

Id	Nombre	Categoría	Descripción	Precio	Stock	Imagen	Acciones
1	Manzanas	Frutas	Manzanas frescas y jugosas. Caja de 10 kg.	25000	55	assets/images/manzana-roja.webp	<button>Editar</button> <button>Borrar</button>
2	Tomates	Verduras	Tomates maduros y de excelente calidad. Bolsa de 1 kg.	5000	20	assets/images/Tomate-Chonto.webp	<button>Editar</button> <button>Borrar</button>

- Desde el botón 'Edita'r voy a modificar la Descripción y el precio de los 2 productos registrados hasta el momento, esto se lleva a cabo en la vista '/editar/:idproducto' :

#### Modificación:

## Modificación:

A screenshot of a web browser window titled "DcfruverFE". The URL is "localhost:4200/productos/editar/2". The page displays a form for modifying a product. The form fields are as follows:

- Nombre: Tomates
- Categoría: Verduras
- Descripción: Los tomates son una valiosa fuente de nutrientes, incluyendo vitaminas como la vitamina C y vitamina A, así como minerales como el potasio. También contienen antioxidantes, como el licopeno, que se ha asociado con beneficios para la salud.
- Precio: 2000
- Stock: 20
- Imagen: assets/images/Tomate-Chonto.webp

At the bottom of the form is a green "Enviar" button.

## Resultado:

A screenshot of a web browser window titled "DcfruverFE". The URL is "localhost:4200/productos". The page displays a table titled "Lista de productos" (List of products) with the following data:

Lista de productos							
Nuevo Producto							
ID	Nombre	Categoría	Descripción	Precio	Stock	Imagen	Acciones
1	Manzanas	Frutas	Las manzanas son una excelente fuente de vitaminas y minerales, como la vitamina C, potasio y fibra. También contienen antioxidantes que ayudan a proteger el cuerpo contra los daños causados por los radicales libres. Además, las manzanas son bajas en calorías y grasas, lo que las convierte en una opción saludable para incluir en una dieta equilibrada.	2150	55	assets/images/manzana-roja.webp	<a href="#">Editar</a> <a href="#">Borrar</a>
2	Tomates	Verduras	Los tomates son una valiosa fuente de nutrientes, incluyendo vitaminas como la vitamina C y vitamina A, así como minerales como el potasio. También contienen antioxidantes, como el licopeno, que se ha asociado con beneficios para la salud.	2000	20	assets/images/Tomate-Chonto.webp	<a href="#">Editar</a> <a href="#">Borrar</a>

- Ahora desde en botón ‘Nuevo producto’ añadiré más productos en la tienda:

Nombre  
Pimenton Amarillo

Categoría  
Verduras

Descripción  
El pimiento amarillo es un vegetal muy versátil en la cocina, y se utiliza en una amplia variedad de preparaciones culinarias. Se puede consumir crudo en ensaladas.

Precio  
3450

Stock  
15

Imagen  
assets/images/pimenton-amarillo.jpg

Enviar

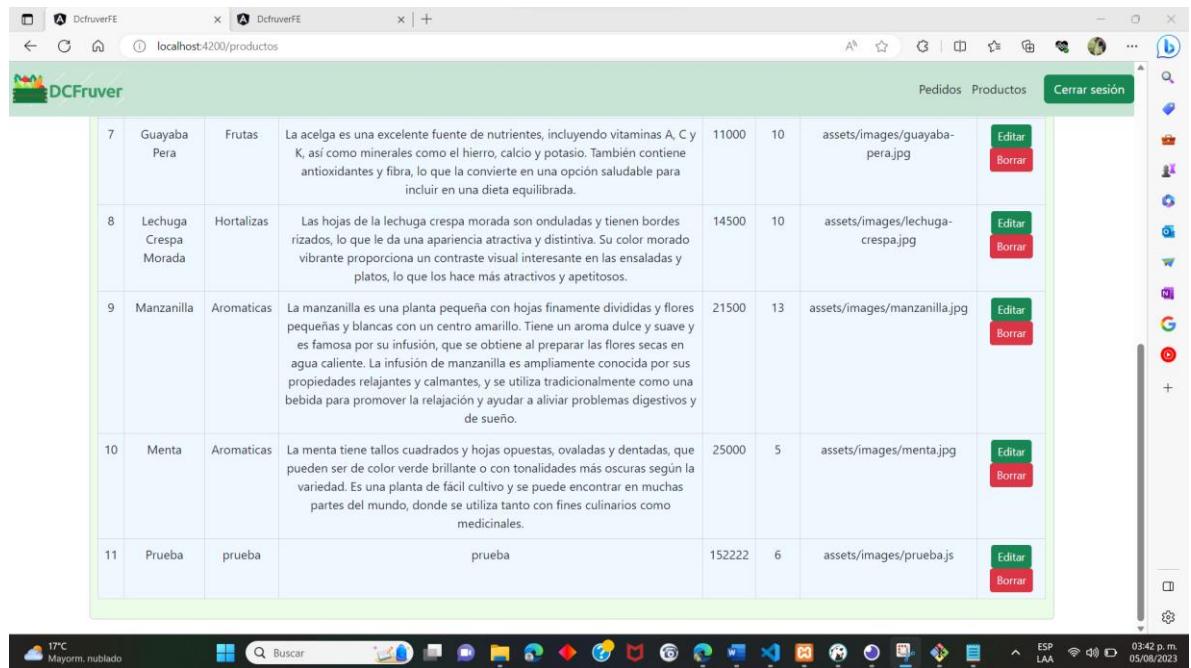


**Nuevo Producto**

**Lista de productos**

Lista de productos							
Nuevo Producto							
ID	Nombre	Categoría	Descripción	Precio	Stock	Imagen	Acciones
1	Manzanas	Frutas	Las manzanas son una excelente fuente de vitaminas y minerales, como la vitamina C, potasio y fibra. También contienen antioxidantes que ayudan a proteger el cuerpo contra los daños causados por los radicales libres. Además, las manzanas son bajas en calorías y grasas, lo que las convierte en una opción saludable para incluir en una dieta equilibrada.	2150	55	assets/images/manzana-roja.webp	<a href="#">Editar</a> <a href="#">Borrar</a>
2	Tomates	Verduras	Los tomates son una valiosa fuente de nutrientes, incluyendo vitaminas como la vitamina C y vitamina A, así como minerales como el potasio. También contienen antioxidantes, como el licopeno, que se ha asociado con beneficios para la salud.	2000	20	assets/images/Tomate-Chonto.webp	<a href="#">Editar</a> <a href="#">Borrar</a>
5	Pimenton Amarillo	Verduras	El pimiento amarillo es un vegetal muy versátil en la cocina, se utiliza en una amplia variedad de preparaciones culinarias. Se puede consumir crudo en ensaladas, relleno en platos horneados, asado a la parrilla o utilizado como ingrediente en guisos, sopas, salsas y salteados. Su color vibrante y su sabor dulce lo convierten en un ingrediente atractivo y delicioso para realzar platos y agregar un toque de frescura a las comidas.	3450	15	assets/images/pimenton-amarillo.jpg	<a href="#">Editar</a> <a href="#">Borrar</a>

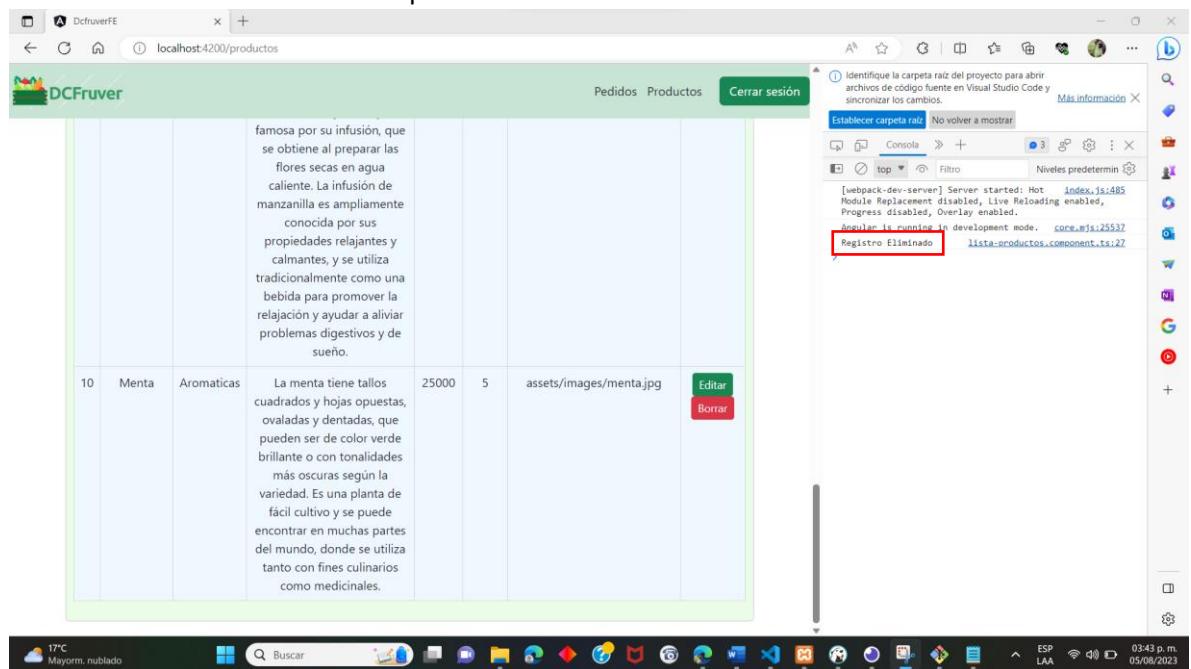




A screenshot of a Windows desktop environment. In the center is a Microsoft Edge browser window titled 'DcfruverFE' showing the URL 'localhost:4200/productos'. The page displays a table of products with columns for ID, Name, Category, Description, Price, Stock, Image, and Action buttons ('Editar' and 'Borrar'). The products listed are: 7. Guayaba Pera (Frutas), 8. Lechuga Crespa Morada (Hortalizas), 9. Manzanilla (Aromaticas), 10. Menta (Aromaticas), and 11. Prueba (prueba). The bottom right corner of the screen shows the system tray with the date and time (03:42 p.m., 05/08/2023) and network status.

	ID	Name	Category	Description	Price	Stock	Image	Action
	7	Guayaba Pera	Frutas	La acelga es una excelente fuente de nutrientes, incluyendo vitaminas A, C y K, así como minerales como el hierro, calcio y potasio. También contiene antioxidantes y fibra, lo que la convierte en una opción saludable para incluir en una dieta equilibrada.	11000	10	assets/images/guayaba-pera.jpg	<button>Editar</button> <button>Borrar</button>
	8	Lechuga Crespa Morada	Hortalizas	Las hojas de la lechuga crespa morada son onduladas y tienen bordes rizados, lo que le da una apariencia atractiva y distintiva. Su color morado vibrante proporciona un contraste visual interesante en las ensaladas y platos, lo que los hace más atractivos y apetitosos.	14500	10	assets/images/lechuga-crespa.jpg	<button>Editar</button> <button>Borrar</button>
	9	Manzanilla	Aromaticas	La manzanilla es una planta pequeña con hojas finamente divididas y flores pequeñas y blancas con un centro amarillo. Tiene un aroma dulce y suave y es famosa por su infusión, que se obtiene al preparar las flores secas en agua caliente. La infusión de manzanilla es ampliamente conocida por sus propiedades relajantes y calmantes, y se utiliza tradicionalmente como una bebida para promover la relajación y ayudar a aliviar problemas digestivos y de sueño.	21500	13	assets/images/manzanilla.jpg	<button>Editar</button> <button>Borrar</button>
	10	Menta	Aromaticas	La menta tiene tallos cuadrados y hojas opuestas, ovaladas y dentadas, que pueden ser de color verde brillante o con tonalidades más oscuras según la variedad. Es una planta de fácil cultivo y se puede encontrar en muchas partes del mundo, donde se utiliza tanto con fines culinarios como medicinales.	25000	5	assets/images/menta.jpg	<button>Editar</button> <button>Borrar</button>
	11	Prueba	prueba	prueba	152222	6	assets/images/prueba.jpg	<button>Editar</button> <button>Borrar</button>

- Eliminaré con el botón 'Borrar' el producto con Id= 11:



A screenshot of a Windows desktop environment. In the center is a Microsoft Edge browser window titled 'DcfruverFE' showing the URL 'localhost:4200/productos'. The page displays a table of products with columns for ID, Name, Category, Description, Price, Stock, Image, and Action buttons ('Editar' and 'Borrar'). The products listed are: 7. Guayaba Pera (Frutas), 8. Lechuga Crespa Morada (Hortalizas), 9. Manzanilla (Aromaticas), 10. Menta (Aromaticas), and 11. Prueba (prueba). The bottom right corner of the screen shows the system tray with the date and time (03:43 p.m., 05/08/2023) and network status. On the right side of the screen, a Visual Studio Code window is open, showing the 'Console' tab with the message '[webpack-dev-server] Server started: Hot Index.js:485 Module Replacement disabled, Live Reloading enabled, Progress disabled, Overlay enabled. Angular is running in development mode. core.mjs:25537 Registro Eliminado lista-productos.component.ts:27'. A red box highlights this message.

	ID	Name	Category	Description	Price	Stock	Image	Action
	7	Guayaba Pera	Frutas	La acelga es una excelente fuente de nutrientes, incluyendo vitaminas A, C y K, así como minerales como el hierro, calcio y potasio. También contiene antioxidantes y fibra, lo que la convierte en una opción saludable para incluir en una dieta equilibrada.	11000	10	assets/images/guayaba-pera.jpg	<button>Editar</button> <button>Borrar</button>
	8	Lechuga Crespa Morada	Hortalizas	Las hojas de la lechuga crespa morada son onduladas y tienen bordes rizados, lo que le da una apariencia atractiva y distintiva. Su color morado vibrante proporciona un contraste visual interesante en las ensaladas y platos, lo que los hace más atractivos y apetitosos.	14500	10	assets/images/lechuga-crespa.jpg	<button>Editar</button> <button>Borrar</button>
	9	Manzanilla	Aromaticas	La manzanilla es una planta pequeña con hojas finamente divididas y flores pequeñas y blancas con un centro amarillo. Tiene un aroma dulce y suave y es famosa por su infusión, que se obtiene al preparar las flores secas en agua caliente. La infusión de manzanilla es ampliamente conocida por sus propiedades relajantes y calmantes, y se utiliza tradicionalmente como una bebida para promover la relajación y ayudar a aliviar problemas digestivos y de sueño.	21500	13	assets/images/manzanilla.jpg	<button>Editar</button> <button>Borrar</button>
	10	Menta	Aromaticas	La menta tiene tallos cuadrados y hojas opuestas, ovaladas y dentadas, que pueden ser de color verde brillante o con tonalidades más oscuras según la variedad. Es una planta de fácil cultivo y se puede encontrar en muchas partes del mundo, donde se utiliza tanto con fines culinarios como medicinales.	25000	5	assets/images/menta.jpg	<button>Editar</button> <button>Borrar</button>
	11	Prueba	prueba	prueba	152222	6	assets/images/prueba.jpg	<button>Editar</button> <button>Borrar</button>

- Ahora le di click en ‘Cerrar sesión’ y volví a la página principal:

Cómo se puede observar ya se añadieron los productos que añadí anteriormente:

The screenshot shows the homepage of the DCFruver website. At the top, there is a navigation bar with a logo, a search icon, and links for 'Carrito (0)' and 'Iniciar sesión'. Below the header, a green oval contains the text 'Tienda de Frutas y Verduras'. There are three filter sections: 'Buscar por nombre:' with a search input field containing 'Escribe el nombre del producto', 'Filtrar por categoría:' with a dropdown set to 'Todas', and 'Filtrar por rango de precios:' with a dropdown set to 'Todos'. Below these filters, there are four product cards arranged in a row. Each card has an image, the product name, its category, and its price. The products are: Manzanas (Frutas, COP2,150.00), Tomates (Verduras, COP2,000.00), Pimenton Amarillo (Verduras, COP3,450.00), and Acelga (Hortalizas, COP2,990.00). The bottom of the screen shows a Windows taskbar with various icons and system status.

- Ejemplo del uso y funcionamiento del filtro por nombre:

This screenshot shows the same DCFruver website as the previous one, but with a search term 'Guayaba' entered into the 'Buscar por nombre:' input field. A red arrow points to the search input field. Another red arrow points to the product card for 'Guayaba Pera', which is highlighted with a red border. The product card includes an image, the name 'Guayaba Pera', the category 'Frutas', and the price 'COP11,000.00'. Below the card are two buttons: 'Añadir al carrito' and 'Detalle'. The rest of the page and interface are identical to the first screenshot.

- Ejemplo del uso y funcionamiento del filtro por categoría:

Buscar por nombre: Escribe el nombre del producto

Filtrar por categoría: Frutas

Filtrar por rango de precios: Todos

<b>Manzanas</b> Frutas COP2,150.00 <a href="#">Añadir al carrito</a> <a href="#">Detalle</a>	<b>Guayaba Pera</b> Frutas COP11,000.00 <a href="#">Añadir al carrito</a> <a href="#">Detalle</a>
---	--

17°C Mayorm. nublado 03:50 p.m. 05/08/2023

Buscar por nombre: Escribe el nombre del producto

Filtrar por categoría: Verduras

Filtrar por rango de precios: Todos

<b>Tomates</b> Verduras COP2,000.00 <a href="#">Añadir al carrito</a> <a href="#">Detalle</a>	<b>Pimenton Amarillo</b> Verduras COP3,450.00 <a href="#">Añadir al carrito</a> <a href="#">Detalle</a>
--	--

17°C Mayorm. nublado 03:51 p.m. 05/08/2023

Screenshot of a web browser showing a product listing for vegetables. The URL is [localhost:4200/principal](http://localhost:4200/principal). The search bar contains "Escribe el nombre del producto". The filter dropdown shows "Hortalizas" with a red arrow pointing to it. The results show two items: "Acelga" and "Lechuga Crespa Morada".

**Acelga**  
Hortalizas  
COP2,990.00  
[Añadir al carrito](#) [Detalle](#)

**Lechuga Crespa Morada**  
Hortalizas  
COP14,500.00  
[Añadir al carrito](#) [Detalle](#)

Below the browser window is a Windows taskbar with various pinned icons and system status.

Screenshot of a web browser showing a product listing for aromatic plants. The URL is [localhost:4200/principal](http://localhost:4200/principal). The search bar contains "Escribe el nombre del producto". The filter dropdown shows "Aromaticas" with a red arrow pointing to it. The results show two items: "Manzanilla" and "Menta".

**Manzanilla**  
Aromaticas  
COP21,500.00  
[Añadir al carrito](#) [Detalle](#)

**Menta**  
Aromaticas  
COP25,000.00  
[Añadir al carrito](#) [Detalle](#)

Below the browser window is a Windows taskbar with various pinned icons and system status.

- Ejemplo del uso y funcionamiento del filtro por precio:

Screenshot of the DCFruver website showing a search interface with price filtering.

**Search Bar:** Buscar por nombre: Escribe el nombre del producto

**Category Filter:** Filtrar por categoría: Todas

**Price Filter:** Filtrar por rango de precios: Menos de \$10,000 (highlighted with a red arrow)

Imagen	Nombre	Categoría	Precio	Opciones
	Manzanas	Frutas	COP2,150.00	Añadir al carrito   Detalle
	Tomates	Verduras	COP2,000.00	Añadir al carrito   Detalle
	Pimenton Amarillo	Verduras	COP3,450.00	Añadir al carrito   Detalle
	Acelga	Hortalizas	COP2,990.00	Añadir al carrito   Detalle

Bottom navigation bar: 17°C Mayorm. nublado, Buscar, various icons, 03:53 p.m., 05/08/2023

Screenshot of the DCFruver website showing a search interface with price filtering.

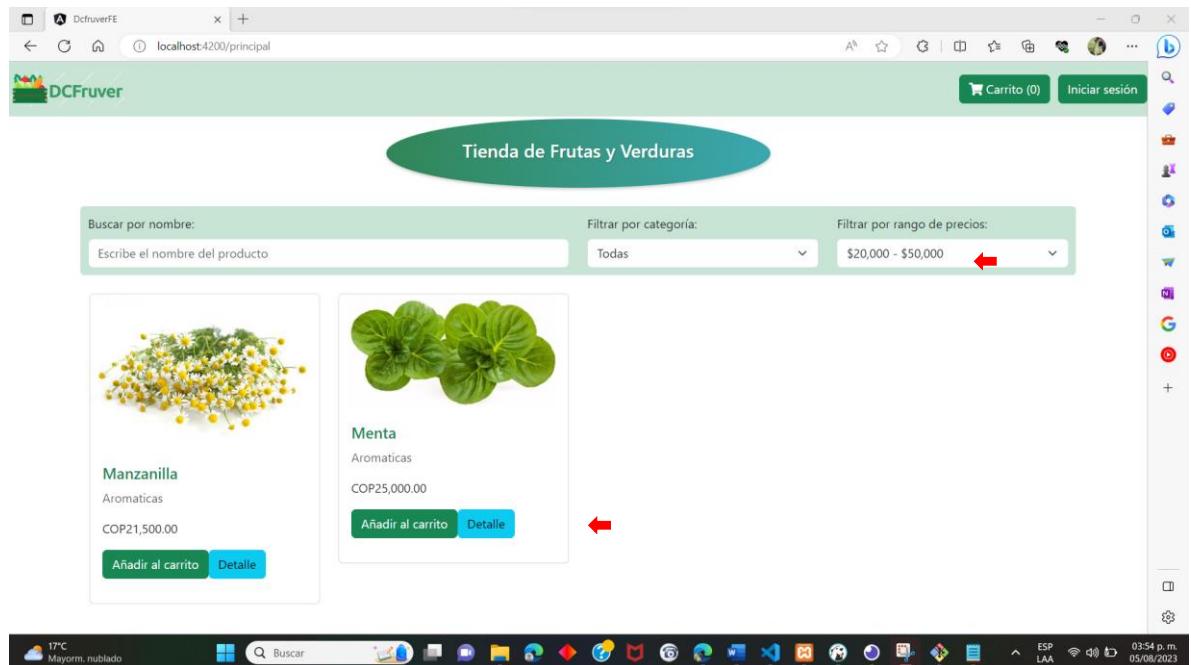
**Search Bar:** Buscar por nombre: Escribe el nombre del producto

**Category Filter:** Filtrar por categoría: Todas

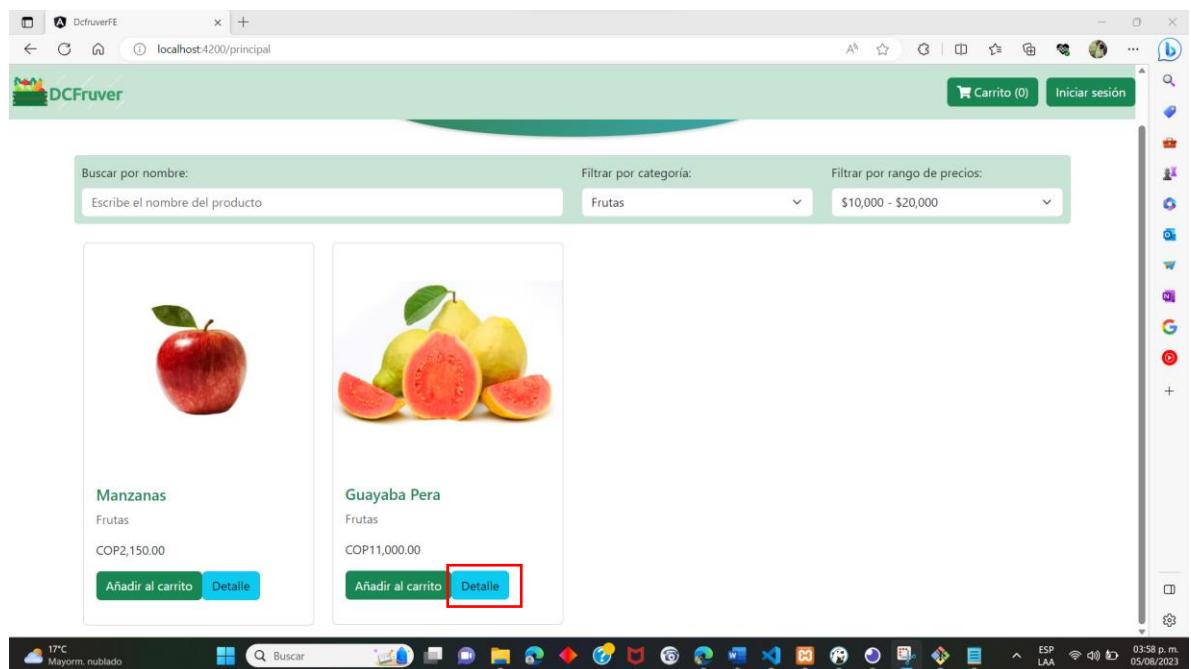
**Price Filter:** Filtrar por rango de precios: \$10,000 - \$20,000 (highlighted with a red arrow)

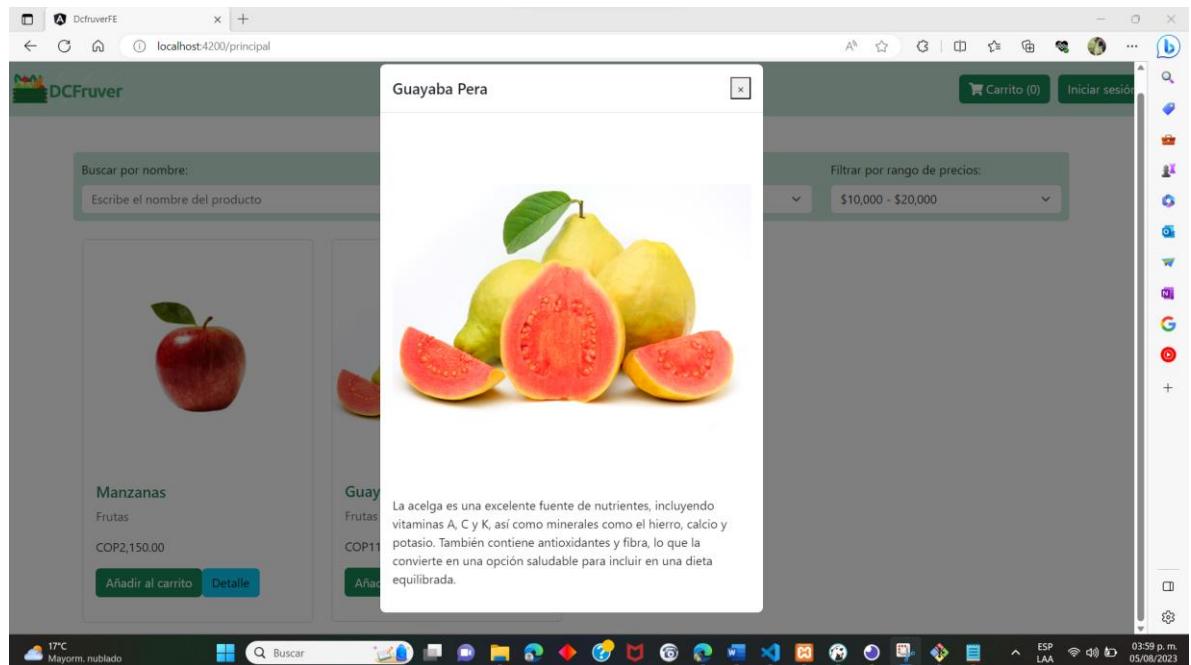
Imagen	Nombre	Categoría	Precio	Opciones
	Guayaba Pera	Frutas	COP11,000.00	Añadir al carrito   Detalle
	Lechuga Crespa Morada	Hortalizas	COP14,500.00	Añadir al carrito   Detalle

Bottom navigation bar: 17°C Mayorm. nublado, Buscar, various icons, 03:53 p.m., 05/08/2023

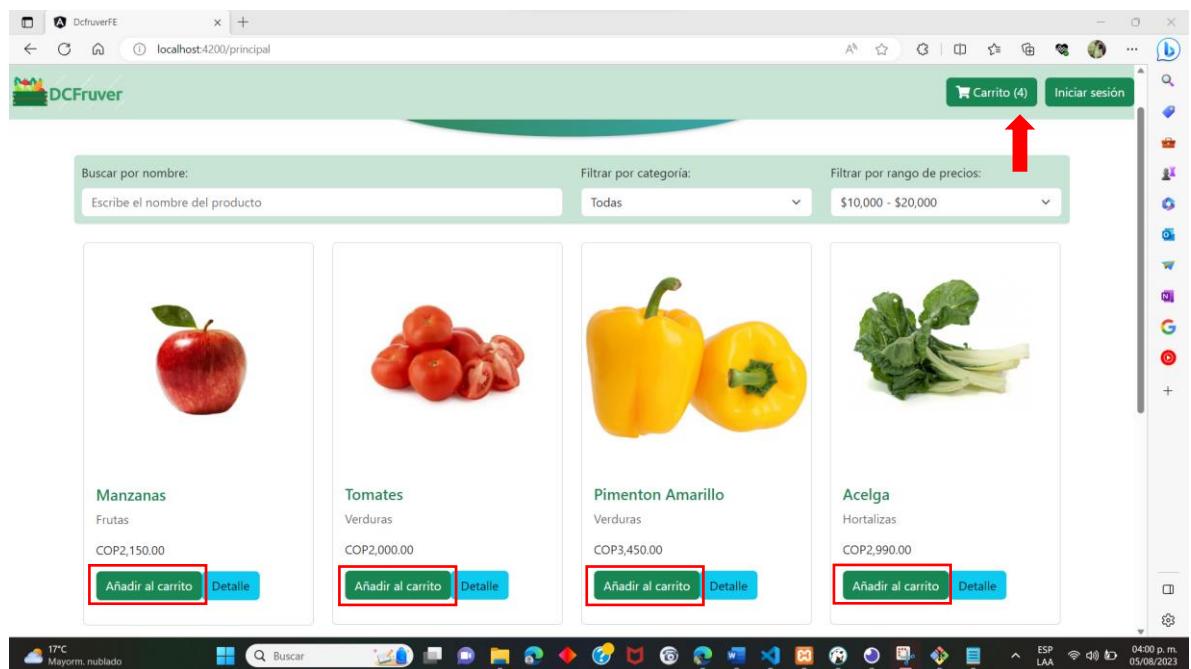


- Si le doy click en el botón 'Detalle' se abre la descripción del producto:



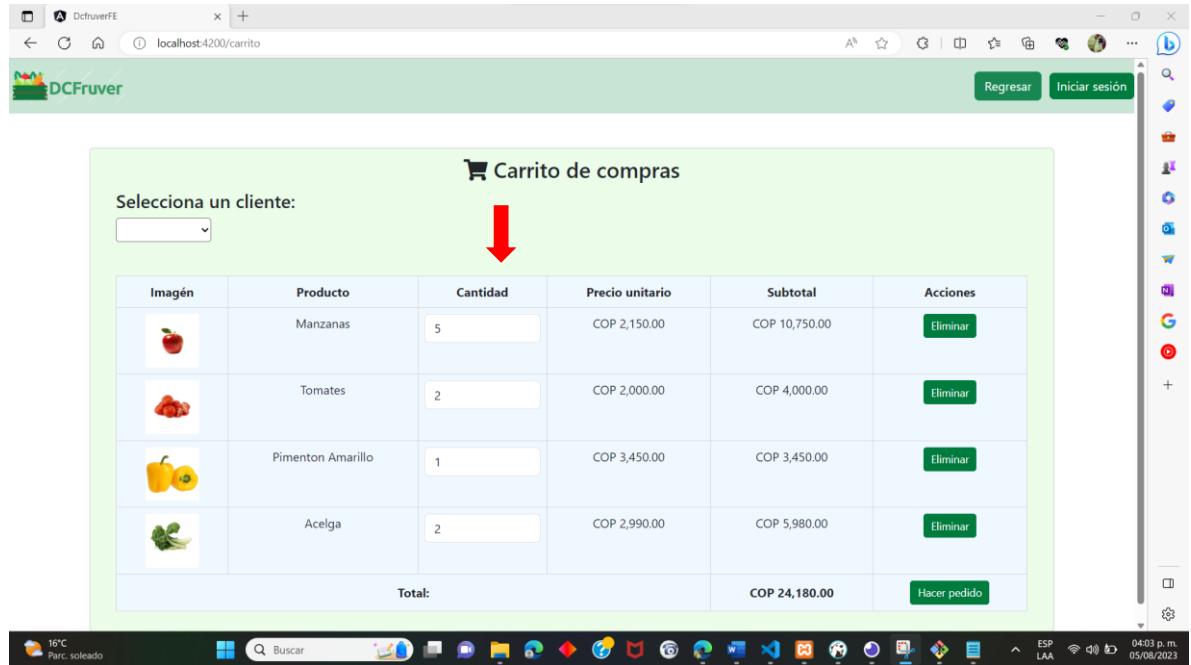


- Ahora añadiré los siguientes productos al carrito:



- Le doy click en el botón ‘Carrito’ del menú superior y hago las siguientes acciones:

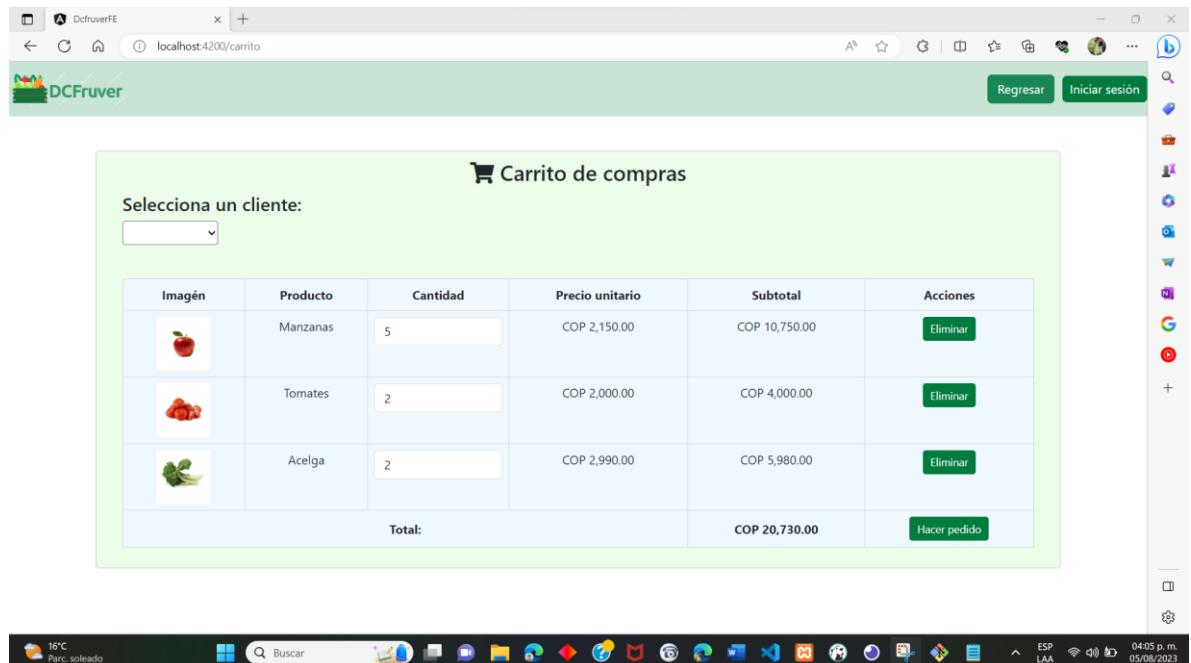
Selecciono la cantidad que quiero de cada producto:



The screenshot shows a web browser window titled "Dcfruver" with the URL "localhost:4200/carrito". The main content is a "Carrito de compras" (Shopping Cart) table. The table has columns: Imagen (Image), Producto (Product), Cantidad (Quantity), Precio unitario (Unit Price), Subtotal (Subtotal), and Acciones (Actions). The table contains four items: Manzanas (Apples) at 5 units, Tomates (Tomatoes) at 2 units, Pimenton Amarillo (Yellow Pepper) at 1 unit, and Acelga (Broccoli) at 2 units. The total subtotal is COP 24,180.00. A green "Hacer pedido" (Place Order) button is at the bottom right. A red arrow points to the "Eliminar" (Delete) button for the Pimenton Amarillo row.

Imagen	Producto	Cantidad	Precio unitario	Subtotal	Acciones
	Manzanas	5	COP 2,150.00	COP 10,750.00	<button>Eliminar</button>
	Tomates	2	COP 2,000.00	COP 4,000.00	<button>Eliminar</button>
	Pimenton Amarillo	1	COP 3,450.00	COP 3,450.00	<button>Eliminar</button>
	Acelga	2	COP 2,990.00	COP 5,980.00	<button>Eliminar</button>
Total:				COP 24,180.00	<button>Hacer pedido</button>

Resultado de darle click en el botón ‘Eliminar’ del Pimentón Amarillo:



The screenshot shows the same web browser window and shopping cart table. The Pimenton Amarillo row has been removed, leaving only three items: Manzanas (5 units), Tomates (2 units), and Acelga (2 units). The total subtotal is now COP 20,730.00. The red arrow from the previous screenshot is no longer present.

Imagen	Producto	Cantidad	Precio unitario	Subtotal	Acciones
	Manzanas	5	COP 2,150.00	COP 10,750.00	<button>Eliminar</button>
	Tomates	2	COP 2,000.00	COP 4,000.00	<button>Eliminar</button>
	Acelga	2	COP 2,990.00	COP 5,980.00	<button>Eliminar</button>
Total:				COP 20,730.00	<button>Hacer pedido</button>

### Selecciono un cliente:

The screenshot shows a web browser window titled "Carrito de compras" (Shopping Cart). At the top left, there is a dropdown menu labeled "Selecciona un cliente:" with "Luis Torres" selected. A red box highlights this dropdown. Below it, a box contains customer information: Cédula: 1111, Nombre: Luis Torres, Dirección: Pasto - Nar, Teléfono: 7202020, and Correo: pruebadiplo10@gmail.com. The main area displays a table of items in the cart:

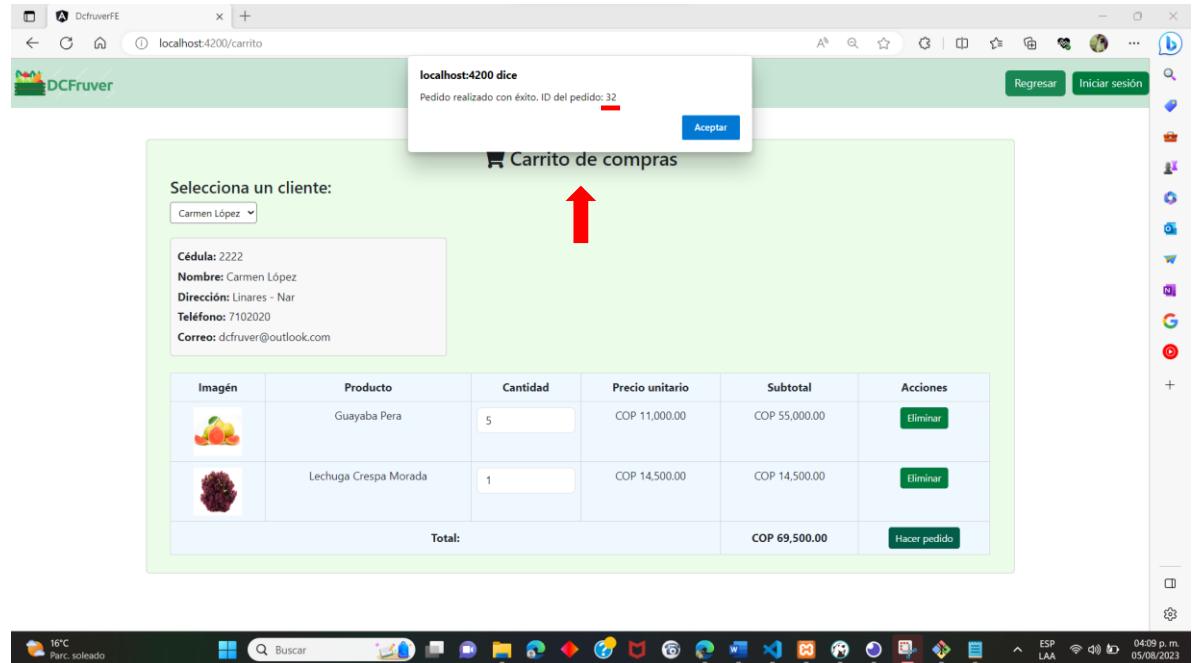
Imagen	Producto	Cantidad	Precio unitario	Subtotal	Acciones
	Manzanas	5	COP 2,150.00	COP 10,750.00	<button>Eliminar</button>
	Tomates	2	COP 2,000.00	COP 4,000.00	<button>Eliminar</button>
	Acelga	2	COP 2,990.00	COP 5,980.00	<button>Eliminar</button>
Total:			COP 20,730.00	<button>Hacer pedido</button>	

At the bottom of the screen, the Windows taskbar shows various pinned icons and the system tray indicates the date and time as 05/08/2023.

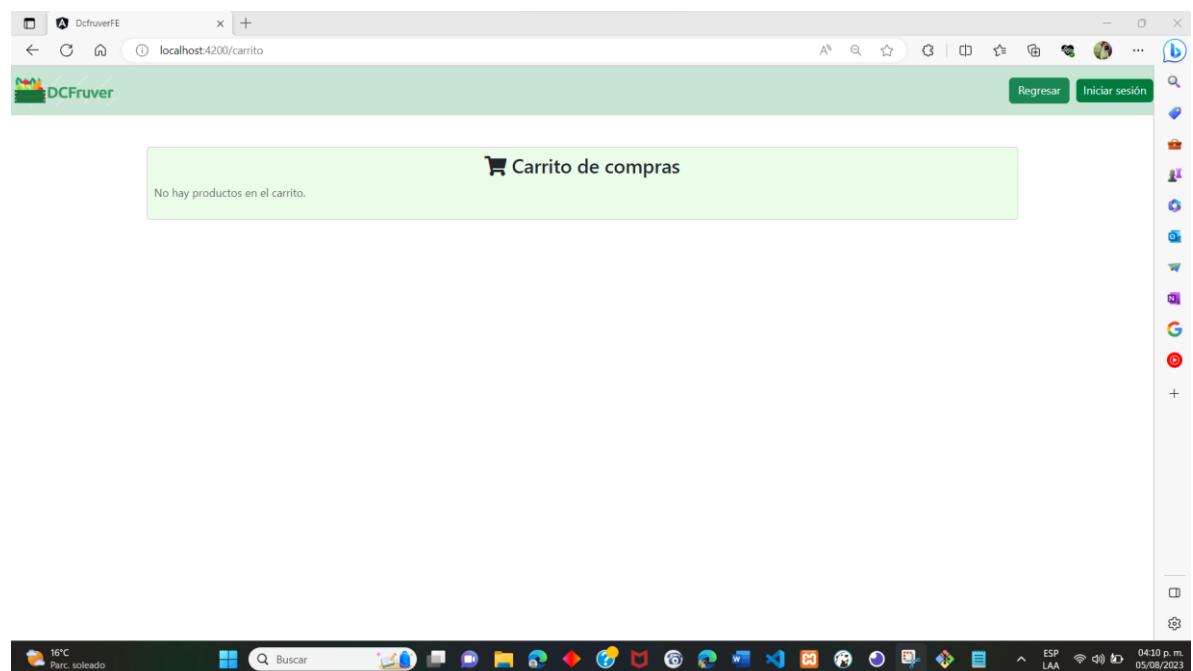
Al darle click en ‘Hacer pedido’, sale un mensaje informando que el pedido se realizo con éxito y me da el id del pedido:

The screenshot shows the same web application interface as the previous one, but now with a modal dialog box centered on the screen. The dialog box has a white background and a thin black border. It contains the text "localhost:4200 dice" (localhost:4200 says) and "Pedido realizado con éxito. ID del pedido: 31". At the bottom right of the dialog is a blue "Aceptar" (Accept) button. A large red arrow points upwards from the bottom of the dialog towards the "Hacer pedido" button in the shopping cart table. The rest of the interface remains the same, showing the shopping cart items and their details.

- Hago otro pedido:

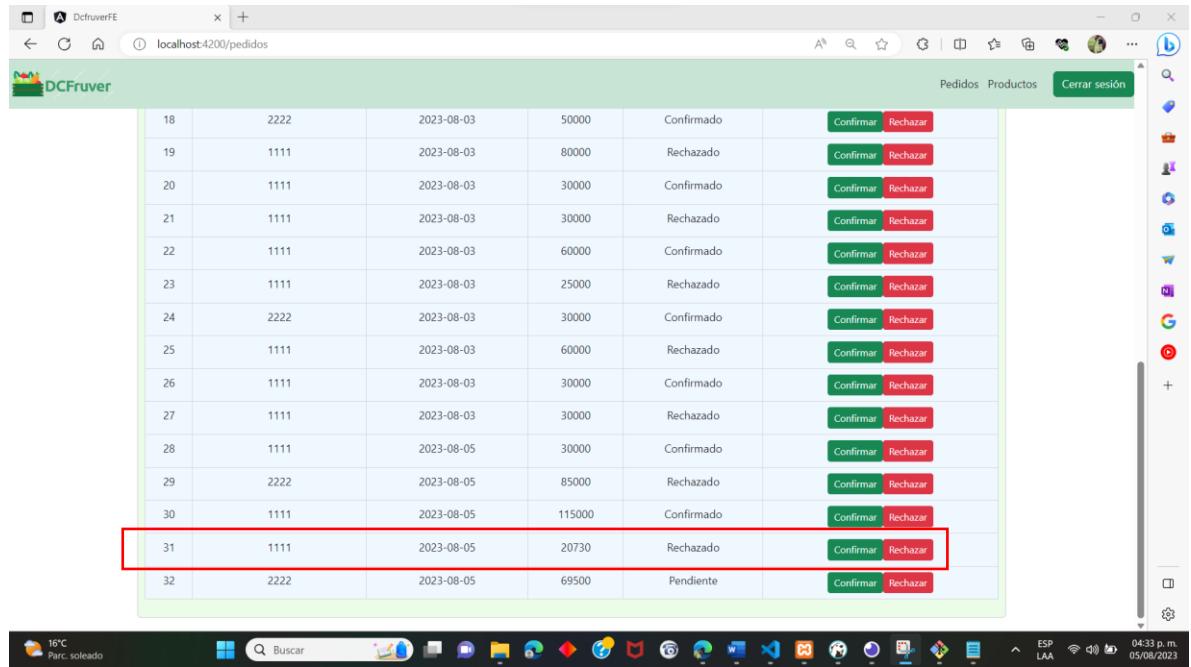


- En caso de no tener nada añadido al carrito de compras, se muestra lo siguiente:



- Vuelvo a entrar a la sesión del Admin y podemos ver que al final de la lista de pedidos se añadieron los que acabamos de registrar con Id=31 y Id=32:

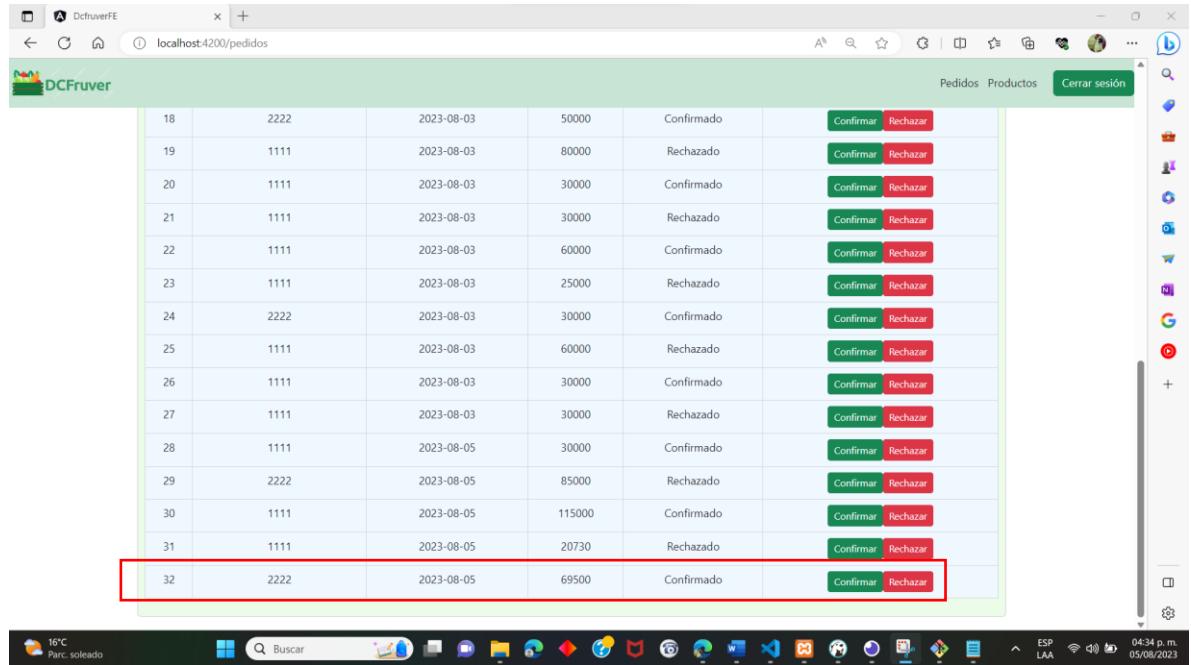
Al pedido con Id = 31 le di click en rechazar:



Detailed description: A screenshot of a web browser window titled 'DcfruverFE'. The URL is 'localhost:4200/pedidos'. The page displays a table of orders. Order ID 31 is highlighted with a red box. The table has columns: ID, Cliente, Fecha, Monto, Estado, and two buttons: 'Confirmar' and 'Rechazar'. Order 31 has a status of 'Rechazado'.

						Pedidos	Productos	Cerrar sesión
18	2222	2023-08-03	50000	Confirmado		<button>Confirmar</button>	<button>Rechazar</button>	
19	1111	2023-08-03	80000	Rechazado		<button>Confirmar</button>	<button>Rechazar</button>	
20	1111	2023-08-03	30000	Confirmado		<button>Confirmar</button>	<button>Rechazar</button>	
21	1111	2023-08-03	30000	Rechazado		<button>Confirmar</button>	<button>Rechazar</button>	
22	1111	2023-08-03	60000	Confirmado		<button>Confirmar</button>	<button>Rechazar</button>	
23	1111	2023-08-03	25000	Rechazado		<button>Confirmar</button>	<button>Rechazar</button>	
24	2222	2023-08-03	30000	Confirmado		<button>Confirmar</button>	<button>Rechazar</button>	
25	1111	2023-08-03	60000	Rechazado		<button>Confirmar</button>	<button>Rechazar</button>	
26	1111	2023-08-03	30000	Confirmado		<button>Confirmar</button>	<button>Rechazar</button>	
27	1111	2023-08-03	30000	Rechazado		<button>Confirmar</button>	<button>Rechazar</button>	
28	1111	2023-08-05	30000	Confirmado		<button>Confirmar</button>	<button>Rechazar</button>	
29	2222	2023-08-05	85000	Rechazado		<button>Confirmar</button>	<button>Rechazar</button>	
30	1111	2023-08-05	115000	Confirmado		<button>Confirmar</button>	<button>Rechazar</button>	
31	1111	2023-08-05	20730	Rechazado		<button>Confirmar</button>	<button>Rechazar</button>	
32	2222	2023-08-05	69500	Pendiente		<button>Confirmar</button>	<button>Rechazar</button>	

Al pedido con Id = 32 le di click en confirmar:



Detailed description: A screenshot of a web browser window titled 'DcfruverFE'. The URL is 'localhost:4200/pedidos'. The page displays a table of orders. Order ID 32 is highlighted with a red box. The table has columns: ID, Cliente, Fecha, Monto, Estado, and two buttons: 'Confirmar' and 'Rechazar'. Order 32 has a status of 'Confirmado'.

						Pedidos	Productos	Cerrar sesión
18	2222	2023-08-03	50000	Confirmado		<button>Confirmar</button>	<button>Rechazar</button>	
19	1111	2023-08-03	80000	Rechazado		<button>Confirmar</button>	<button>Rechazar</button>	
20	1111	2023-08-03	30000	Confirmado		<button>Confirmar</button>	<button>Rechazar</button>	
21	1111	2023-08-03	30000	Rechazado		<button>Confirmar</button>	<button>Rechazar</button>	
22	1111	2023-08-03	60000	Confirmado		<button>Confirmar</button>	<button>Rechazar</button>	
23	1111	2023-08-03	25000	Rechazado		<button>Confirmar</button>	<button>Rechazar</button>	
24	2222	2023-08-03	30000	Confirmado		<button>Confirmar</button>	<button>Rechazar</button>	
25	1111	2023-08-03	60000	Rechazado		<button>Confirmar</button>	<button>Rechazar</button>	
26	1111	2023-08-03	30000	Confirmado		<button>Confirmar</button>	<button>Rechazar</button>	
27	1111	2023-08-03	30000	Rechazado		<button>Confirmar</button>	<button>Rechazar</button>	
28	1111	2023-08-05	30000	Confirmado		<button>Confirmar</button>	<button>Rechazar</button>	
29	2222	2023-08-05	85000	Rechazado		<button>Confirmar</button>	<button>Rechazar</button>	
30	1111	2023-08-05	115000	Confirmado		<button>Confirmar</button>	<button>Rechazar</button>	
31	1111	2023-08-05	20730	Rechazado		<button>Confirmar</button>	<button>Rechazar</button>	
32	2222	2023-08-05	69500	Confirmado		<button>Confirmar</button>	<button>Rechazar</button>	

Al hacer las anteriores acciones se envió un correo de rechazo y confirmación del pedido a los correos registrados:

	cedulacliente	nombrelcliente	dircliente	telcliente	correocliente
1	1111	Luis Torres	Pasto - Nar	7202020	pruebadis10@gmail.com
2	2222	Carmen López	Linares - Nar	7102020	dcfruver@outlook.com
3	3333	Pablo Torres	Medellín	7302020	prueba@ej.com

La información del correo del cuál se envían los correos se encuentra registrado en el archivo 'controller\_correo.js' del proyecto 'dcfruver-backend' desarrollado en el taller 2:

```

// Importar el módulo nodemailer para enviar correos electrónicos
import nodemailer from 'nodemailer';

// Función para enviar un correo electrónico
const enviarCorreo = async (req, res) => {
  try {
    // Obtener el cuerpo de la solicitud (datos del correo a enviar)
    const body = req.body;
    // Configurar el servicio de transporte para enviar el correo
    const config = await nodemailer.createTransport({
      host: 'smtp.office365.com',
      port: 587, // corregido 'post' por 'port'
      auth: {
        user: 'dcfruverAS@hotmail.com',
        pass: 'd_cf1234'
      }
    });
    // Configurar las opciones del correo
    const opciones = {
      from: 'Programación',
      subject: body.asunto,
      to: body.email,
      text: body.mensaje
    };
    // Enviar el correo usando la configuración y opciones definidas
    config.sendMail(opciones, function (error, result) {
      if (error) return res.json({ ok: false, msg: error });
      return res.json({ ok: true, msg: result });
    });
  } catch (error) {
    res.status(400).json({ mensaje: error.message });
  }
}
  
```

Prueba de que sí se enviaron los correos, aunque como son marcados como spam no llegan a su destinatario:

#### Correo de rechazo del pedido con Id = 31:

The screenshot shows an Outlook inbox with a red box highlighting a specific email. The subject of the email is "DCFruver - Pedido Rechazado". The message body contains the following text:

DCFruver SAS  
Para: rebadiplo10@gmail.com  
Su pedido, con número 31, ha sido rechazado

Traducir mensaje a: Español | Nunca traduzca de: Inglés  
Para volver a enviar este mensaje, haga clic aquí.

postmaster@outlook.com  
Para: postmaster@outlook.com  
DCFruver - Pedido Rechazado

**Delivery has failed to these recipients or groups:**  
pruebadiplo10@gmail.com  
Your message wasn't delivered because the recipient's email provider rejected it.

#### Correo de confirmación del pedido con Id = 32:

The screenshot shows an Outlook inbox with a red box highlighting a specific email. The subject of the email is "DCFruver - Confirmación de pedido". The message body contains the following text:

DCFruver SAS  
Para: dcfruver@outlook.com  
Su pedido, con número 32, ha sido confirmado

Traducir mensaje a: Español | Nunca traduzca de: Inglés  
Para volver a enviar este mensaje, haga clic aquí.

postmaster@outlook.com  
Para: postmaster@outlook.com  
DCFruver - Confirmación de ...

**Delivery has failed to these recipients or groups:**  
dcfruver@outlook.com  
Your message wasn't delivered because the recipient's email provider rejected it.

Nota: Por último, hice 2 commits. El primero de la carpeta shared que contiene todos los modelos y servicios usados en este proyecto llamado "Archivos service y model configurados

para este proyecto". Y el segundo commit, contiene las imágenes usadas en este proyecto como otros archivos, a este commit lo llame "Imagenes y otros archivos".

Y en el proyecto remoto creado para el Taller 2 de este módulo, le añadí algunas cosas que necesite para completar este Taller 3, todo quedo registrado en un commit llamado "Modificaciones Taller 3".