



TECNOLÓGICO
NACIONAL DE MÉXICO



TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE MILPA ALTA II

ASIGNATURA:

MEAN STACK FOR FRONT – END

TEMA:

PRACTICA 2 ANGULAR

ELABORA:

POZOS RIVERA ALEJANDRO

NÚMERO DE CONTROL:

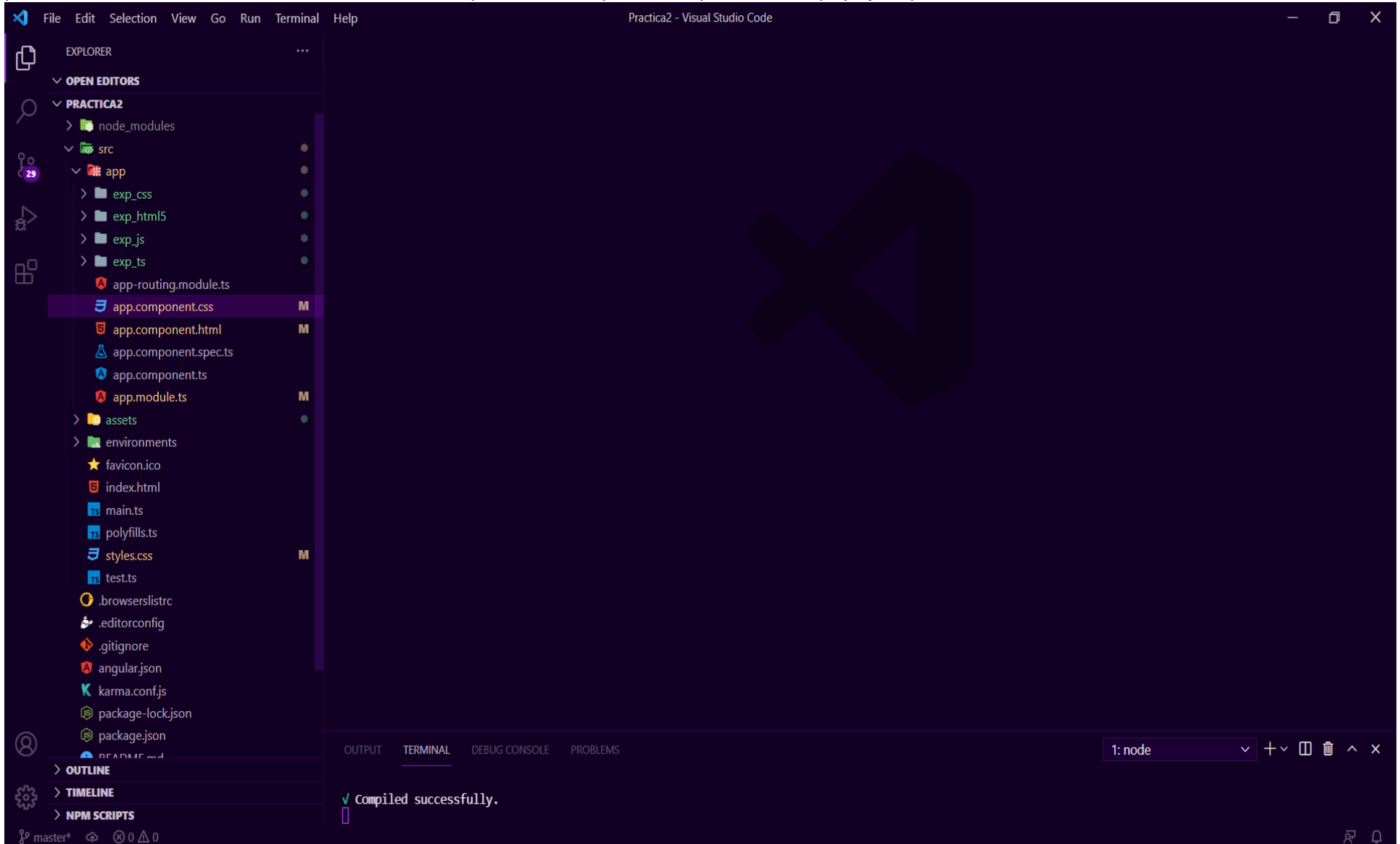
171190025

PROFESOR:

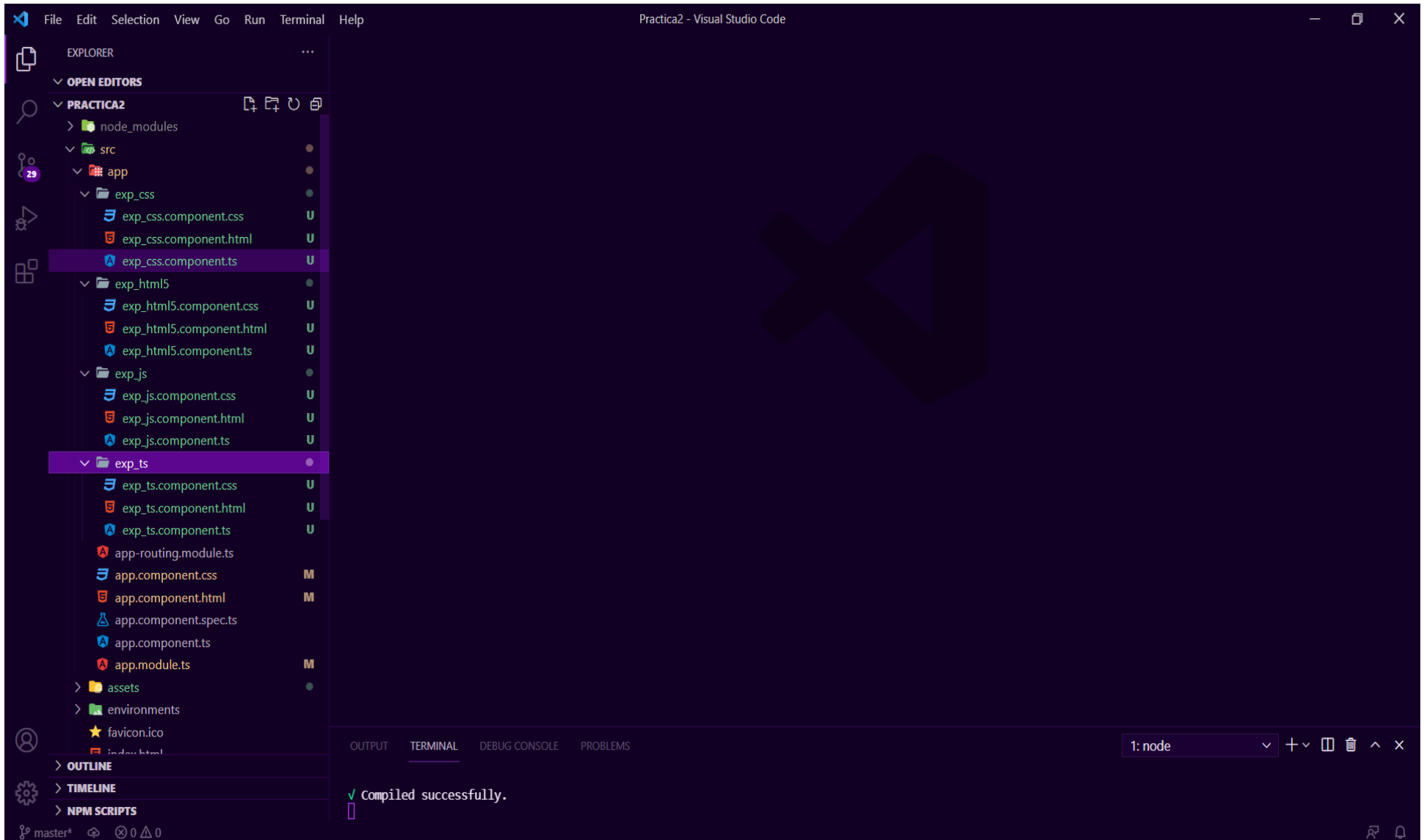
ROLDAN AQUINO SEGURA

19 DE MAYO DE 2021

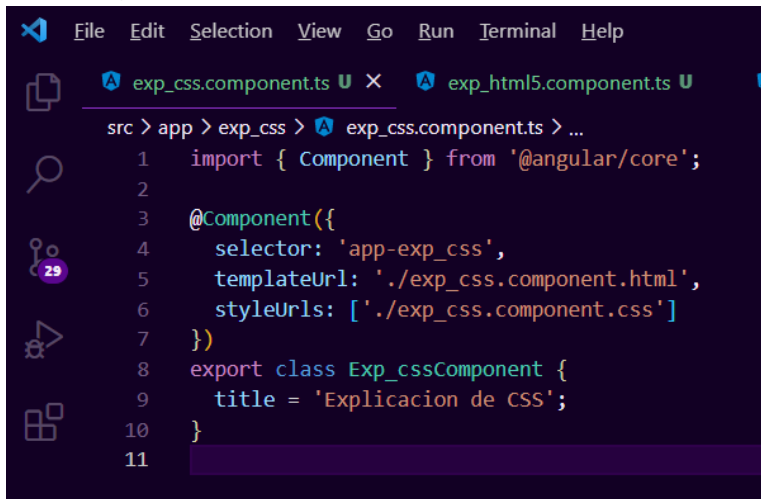
Una vez que ya tengamos nuestro nuevo proyecto. Crearemos 4 carpetas correspondientes para los 4 componentes que contendrá la práctica. En este caso los nombres de las carpetas son exp_css, exp_html5, exp_js y exp_ts.



Dentro de cada carpeta tiene que contener los 3 componentes esenciales para crear nuevos componentes los cuales son, component.css, component.html y por último component.ts.

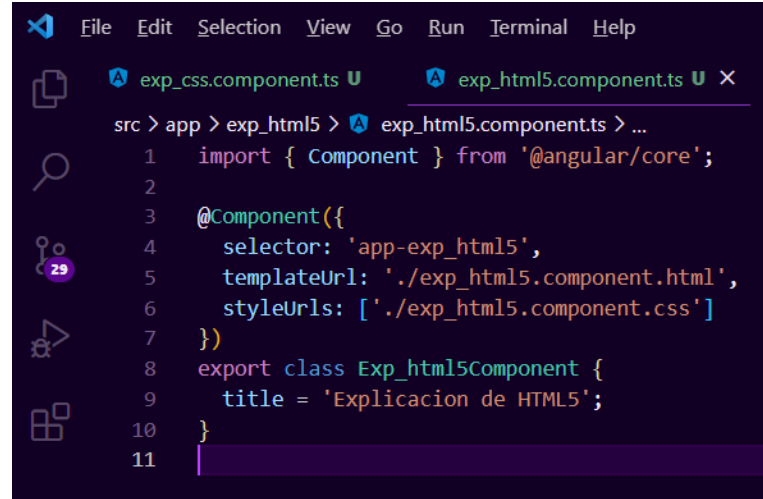


Una vez que ya tengamos los componentes de las carpetas tenemos que cambiar los datos que están dentro de los component.ts para que tengan el nombre de la referencia que tiene la carpeta.



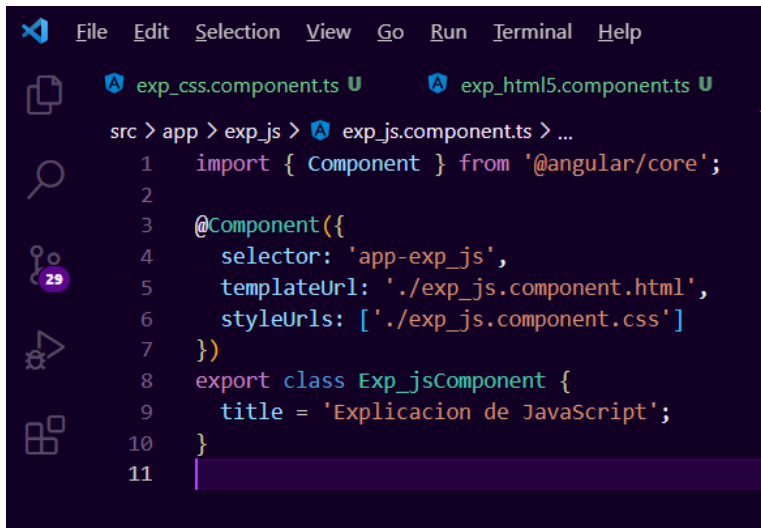
```
File Edit Selection View Go Run Terminal Help
exp_css.component.ts x exp_html5.component.ts U
src > app > exp_css > exp_css.component.ts > ...
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-exp_css',
5   templateUrl: './exp_css.component.html',
6   styleUrls: ['./exp_css.component.css']
7 })
8 export class Exp_cssComponent {
9   title = 'Explicacion de CSS';
10 }
11
```

Imagen de exp_css



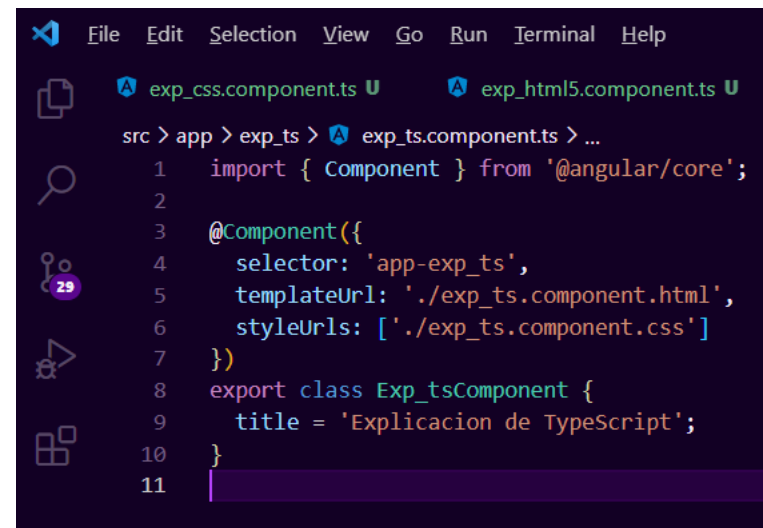
```
File Edit Selection View Go Run Terminal Help
exp_css.component.ts U exp_html5.component.ts U x
src > app > exp_html5 > exp_html5.component.ts > ...
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-exp_html5',
5   templateUrl: './exp_html5.component.html',
6   styleUrls: ['./exp_html5.component.css']
7 })
8 export class Exp_html5Component {
9   title = 'Explicacion de HTML5';
10 }
11
```

Imagen de exp_html5



```
File Edit Selection View Go Run Terminal Help
exp_css.component.ts U exp_html5.component.ts U
src > app > exp_js > exp_js.component.ts > ...
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-exp_js',
5   templateUrl: './exp_js.component.html',
6   styleUrls: ['./exp_js.component.css']
7 })
8 export class Exp_jsComponent {
9   title = 'Explicacion de JavaScript';
10 }
11
```

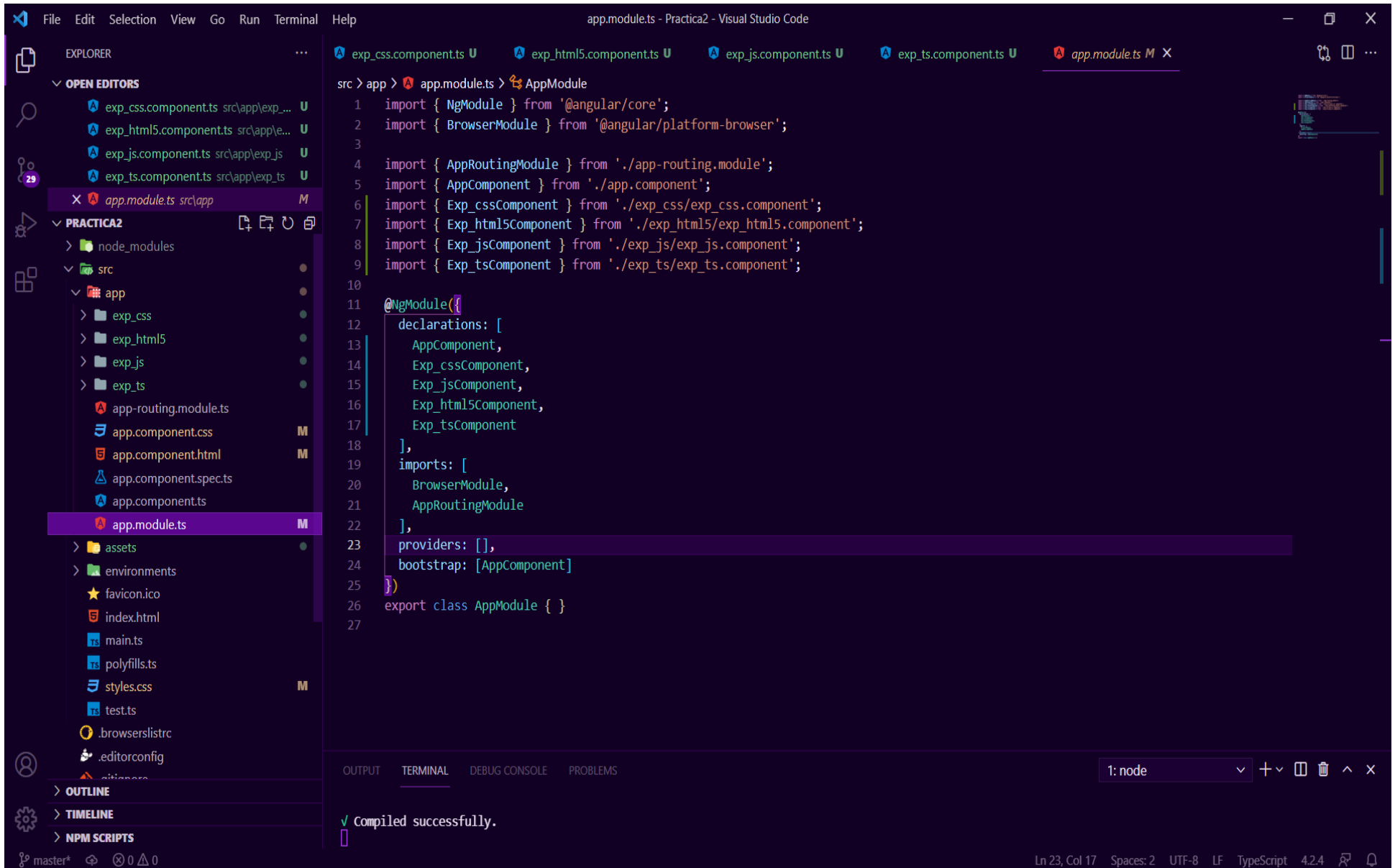
Imagen de exp_js



```
File Edit Selection View Go Run Terminal Help
exp_css.component.ts U exp_html5.component.ts U
src > app > exp_ts > exp_ts.component.ts > ...
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-exp_ts',
5   templateUrl: './exp_ts.component.html',
6   styleUrls: ['./exp_ts.component.css']
7 })
8 export class Exp_tsComponent {
9   title = 'Explicacion de TypeScript';
10 }
11
```

Imagen de exp_ts

Ahora daremos de alta los componentes dentro del apartado de app.module.ts.



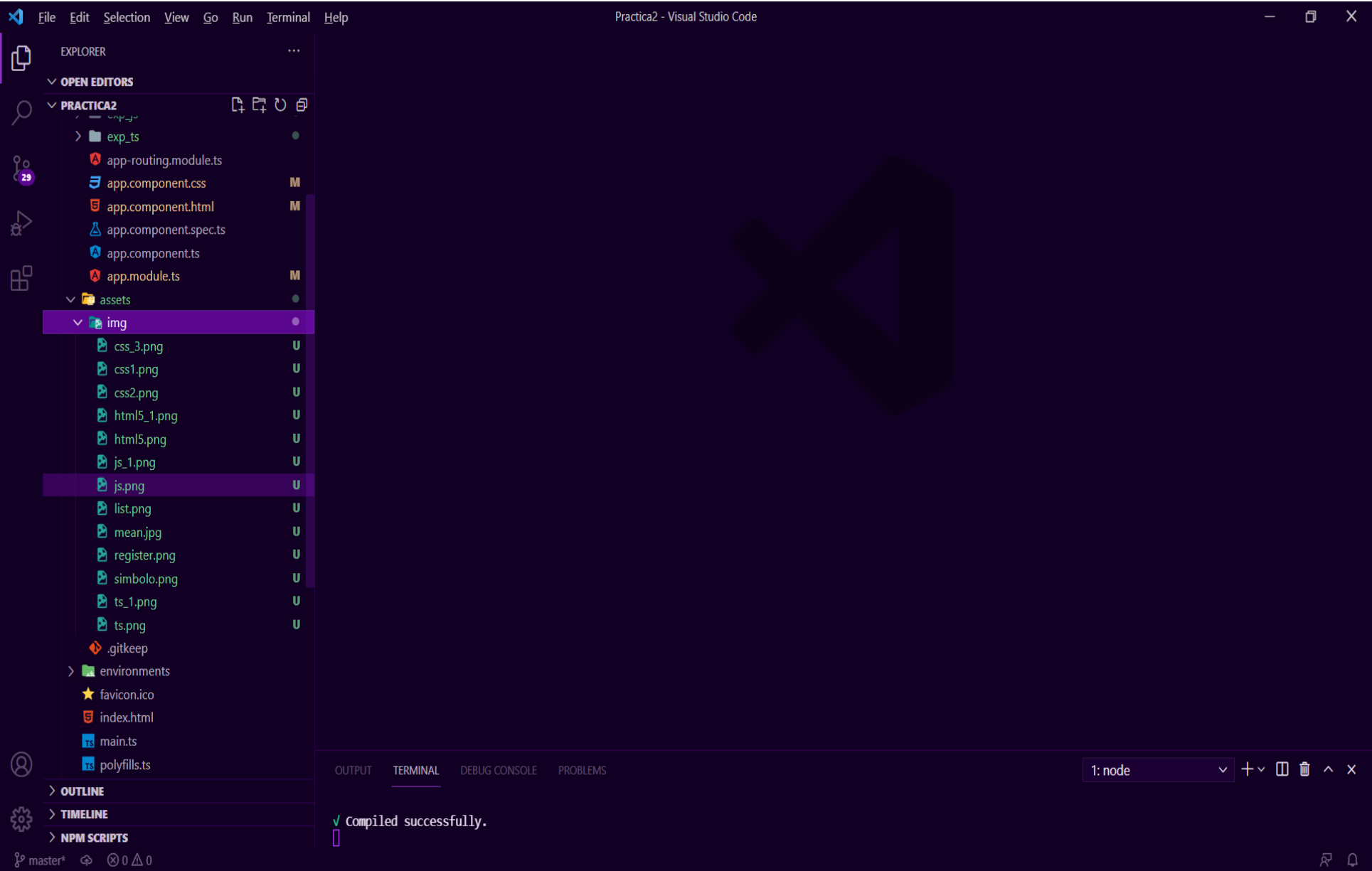
```
File Edit Selection View Go Run Terminal Help
app.module.ts - Practica2 - Visual Studio Code

EXPLORER
OPEN EDITORS
  exp_css.component.ts src\app\exp_... U
  exp_html5.component.ts src\app\exp... U
  exp_js.component.ts src\app\exp_js U
  exp_ts.component.ts src\app\exp_ts U
  X app.module.ts src\app M
PRACTICA2
  node_modules
  src
    app
      exp_css
      exp_html5
      exp_js
      exp_ts
      app-routing.module.ts
      app.component.css M
      app.component.html M
      app.component.spec.ts
      app.component.ts
      app.module.ts M
      assets
      environments
        favicon.ico
        index.html
        main.ts
        polyfills.ts
        styles.css M
        test.ts
        .browserslistrc
        .editorconfig
      tsconfig.json

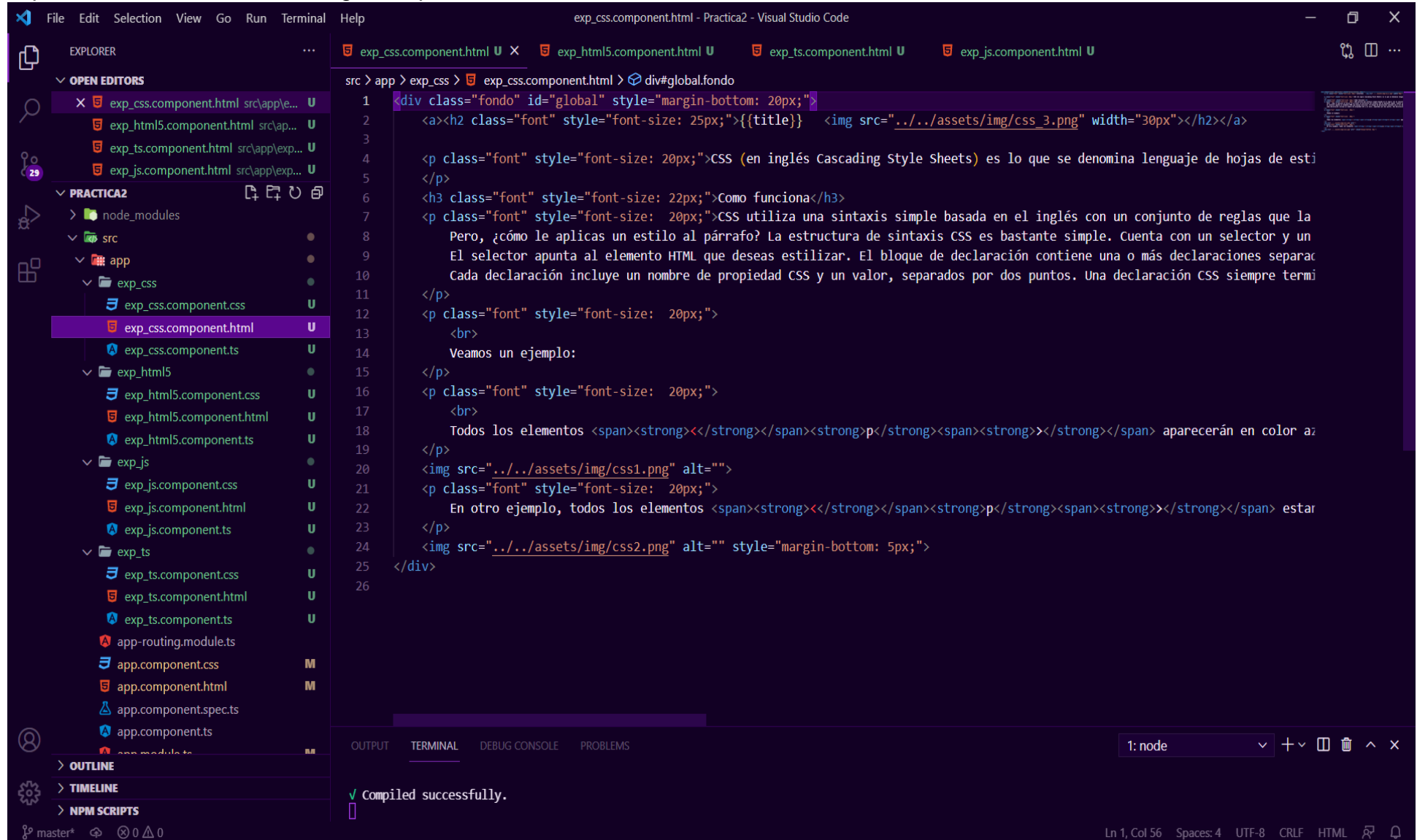
src > app > app.module.ts > AppModule
1  import { NgModule } from '@angular/core';
2  import { BrowserModule } from '@angular/platform-browser';
3
4  import { AppRoutingModule } from './app-routing.module';
5  import { AppComponent } from './app.component';
6  import { Exp_cssComponent } from './exp_css/exp_css.component';
7  import { Exp_html5Component } from './exp_html5/exp_html5.component';
8  import { Exp_jsComponent } from './exp_js/exp_js.component';
9  import { Exp_tsComponent } from './exp_ts/exp_ts.component';
10
11  @NgModule({
12    declarations: [
13      AppComponent,
14      Exp_cssComponent,
15      Exp_jsComponent,
16      Exp_html5Component,
17      Exp_tsComponent
18    ],
19    imports: [
20      BrowserModule,
21      AppRoutingModule
22    ],
23    providers: [],
24    bootstrap: [AppComponent]
25  })
26  export class AppModule { }
27

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS
1: node
✓ Compiled successfully.
```

Agregamos una carpeta dentro del apartado de assets llamada img para introducir las imágenes que vayamos a ocupar.



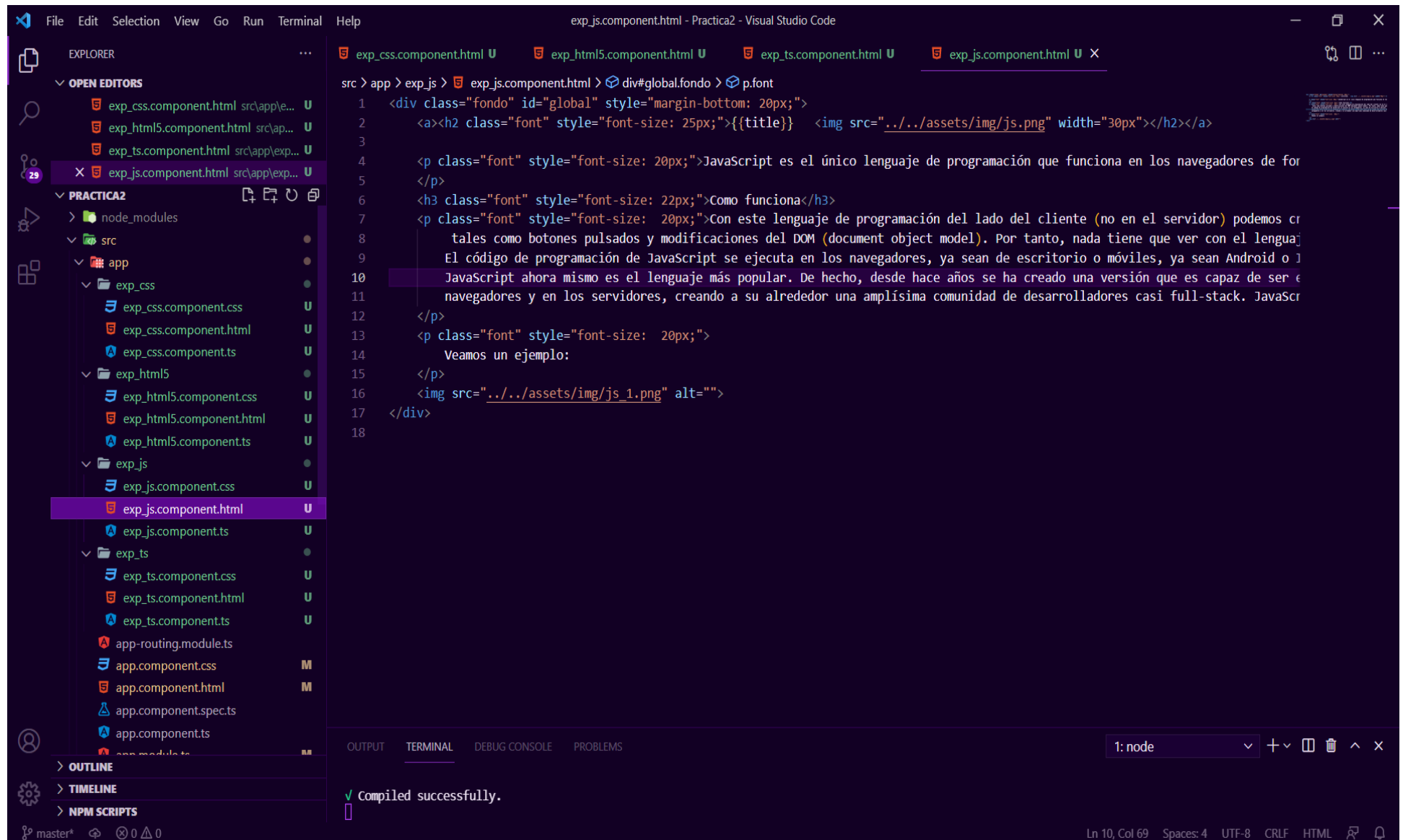
Crearemos dentro del apartado de exp_css.component.html crearemos el contenido que tendrá el componente, referente a la explicación de CSS. Y unas imágenes que son sirven de referencia.



```
exp_css.component.html
1 <div class="fondo" id="global" style="margin-bottom: 20px;">
2   <a><h2 class="font" style="font-size: 25px;">{{title}} </h2></a>
3
4   <p class="font" style="font-size: 20px;">CSS (en inglés Cascading Style Sheets) es lo que se denomina lenguaje de hojas de esti
5   </p>
6   <h3 class="font" style="font-size: 22px;">Como funciona</h3>
7   <p class="font" style="font-size: 20px;">CSS utiliza una sintaxis simple basada en el inglés con un conjunto de reglas que la
8     Pero, ¿cómo le aplicas un estilo al párrafo? La estructura de sintaxis CSS es bastante simple. Cuenta con un selector y un
9     El selector apunta al elemento HTML que deseas estilizar. El bloque de declaración contiene una o más declaraciones separac
10    Cada declaración incluye un nombre de propiedad CSS y un valor, separados por dos puntos. Una declaración CSS siempre termi
11  </p>
12  <p class="font" style="font-size: 20px;">
13    <br>
14    Veamos un ejemplo:
15  </p>
16  <p class="font" style="font-size: 20px;">
17    <br>
18    Todos los elementos <span><strong></strong></span><strong>p</strong><span><strong></strong></span> aparecerán en color az
19  </p>
20  
21  <p class="font" style="font-size: 20px;">
22    En otro ejemplo, todos los elementos <span><strong></strong></span><strong>p</strong><span><strong></strong></span> estar
23  </p>
24  
25 </div>
26
```

```
OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS
1: node
✓ Compiled successfully.
```

Crearemos dentro del apartado de exp_js.component.html crearemos el contenido que tendrá el componente, referente a la explicación de JS. Y unas imágenes que son sirven de referencia.



```
exp_js.component.html - Practica2 - Visual Studio Code

src > app > exp_js > exp_js.component.html > div#global.fondo > p.font
1 <div class="fondo" id="global" style="margin-bottom: 20px;">
2 <a><h2 class="font" style="font-size: 25px;">{{title}} </h2></a>
3
4 <p class="font" style="font-size: 20px;">JavaScript es el único lenguaje de programación que funciona en los navegadores de for
5 </p>
6 <h3 class="font" style="font-size: 22px;">Como funciona</h3>
7 <p class="font" style="font-size: 20px;">Con este lenguaje de programación del lado del cliente (no en el servidor) podemos cr
8 tales como botones pulsados y modificaciones del DOM (document object model). Por tanto, nada tiene que ver con el lenguaj
9 El código de programación de JavaScript se ejecuta en los navegadores, ya sean de escritorio o móviles, ya sean Android o I
10 JavaScript ahora mismo es el lenguaje más popular. De hecho, desde hace años se ha creado una versión que es capaz de ser e
11 navegadores y en los servidores, creando a su alrededor una amplísima comunidad de desarrolladores casi full-stack. JavaScr
12 </p>
13 <p class="font" style="font-size: 20px;">
14 Veamos un ejemplo:
15 </p>
16 
17 </div>
18
```

```
1: node
✓ Compiled successfully.
```


Crearemos dentro del apartado de exp_js.component.html crearemos el contenido que tendrá el componente, referente a la explicación de JS. Y unas imágenes que son sirven de referencia.

```
exp_html5.component.html - Practica2 - Visual Studio Code

src > app > exp_html5 > exp_html5.component.html > div#global.fondo > p.font
1 <div class="fondo" id="global" style="margin-bottom: 20px;">
2 <a><h2 class="font" style="font-size: 25px;">{{title}} </h2></a>
3
4 <p class="font" style="font-size: 20px;">
5     HTML5 es un estándar que sirve como referencia del software que conecta con la elaboración de páginas web en sus diferentes
6     página web, como texto, imágenes, vídeos, juegos, entre otros.
7 </p>
8 <h3 class="font" style="font-size: 22px;">Como funciona</h3>
9 <p class="font" style="font-size: 20px;">
10     Vamos a ver cómo es proceso cuando se solicita una página HTML a través del navegador. El proceso es el siguiente:
11 </p>
12 
13 <p class="font" style="font-size: 20px;">
14     Esquema de solicitud de una Web
15     Desde el navegador se realiza una petición a un servidor, lo que se hace a través de una dirección del tipo http://.../inc
16 </p>
17 </div>
18
```

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

1: node

✓ Compiled successfully.

Ln 13, Col 23 Spaces: 4 UTF-8 CRLF HTML

Crearemos dentro del apartado de exp_js.component.html crearemos el contenido que tendrá el componente, referente a la explicación de JS. Y unas imágenes que son sirven de referencia.

```
exp_ts.component.html - Practica2 - Visual Studio Code

src > app > exp_ts > exp_ts.component.html > div#global.fondo > img

1 <div class="fondo" id="global">
2   <a><h2 class="font" style="font-size: 25px;">{{title}}   </h2></a>
3
4   <p class="font" style="font-size: 20px;">TypeScript es un superconjunto de JavaScript con un tipado que se compila para JavaScr
5   </p>
6   <h3 class="font" style="font-size: 22px;">Como funciona</h3>
7   <p class="font" style="font-size: 20px;">El sistema de tipos de Typescript realiza una formalización de los tipos de Javascript
8   durante el tiempo de diseño nos ayuda a evitar errores en tiempo de ejecución, como podría ser pasar el tipo de variable i
9   Además de los tipos String y Number admite los siguientes tipos básicos:
10  </p>
11  <p class="font">- Boolean: tipo de dato lógico que representa verdadero o falso.</p>
12  <p class="font">- Array: tipo de dato estructurado que permite almacenar una colección de elementos.</p>
13  <p class="font">- Tuple: similar al array, pero con un número fijo de elementos escritos.</p>
14  <p class="font">- Enum: representa al tipo enumeración.</p>
15  <p class="font">- Any: indica que la variable puede ser de cualquier tipo. Es muy útil a la hora de trabajar con librerías exte
16  <p class="font">- Void: indica que una función no devolverá ningún valor.</p>
17  <p class="font">- Never: este tipo representa el tipo de valores que nunca se producen.</p>
18  <p class="font" style="font-size: 20px;">
19    Veamos un ejemplo:
20  </p>
21  
22 </div>
23
```

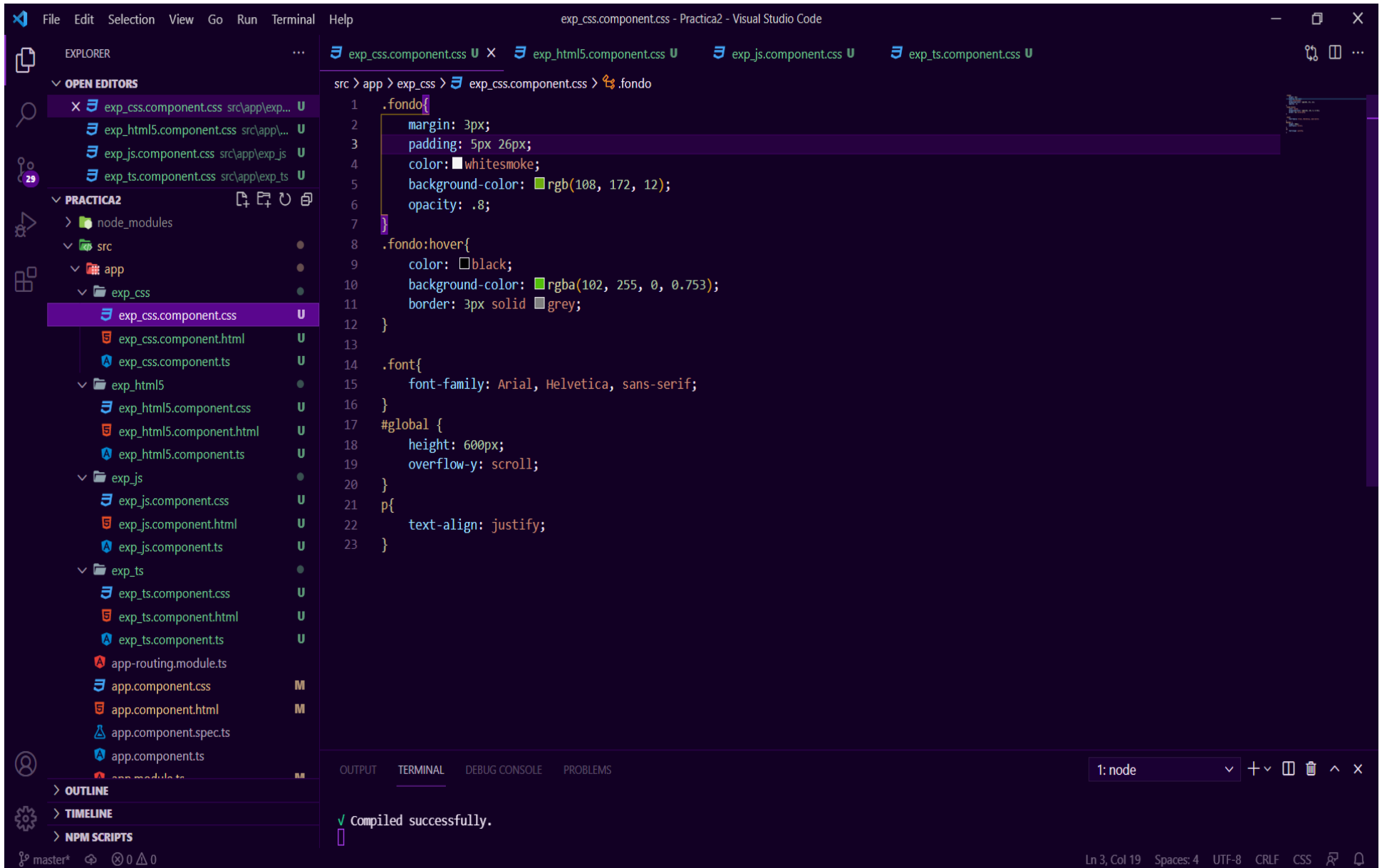
OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

1: node

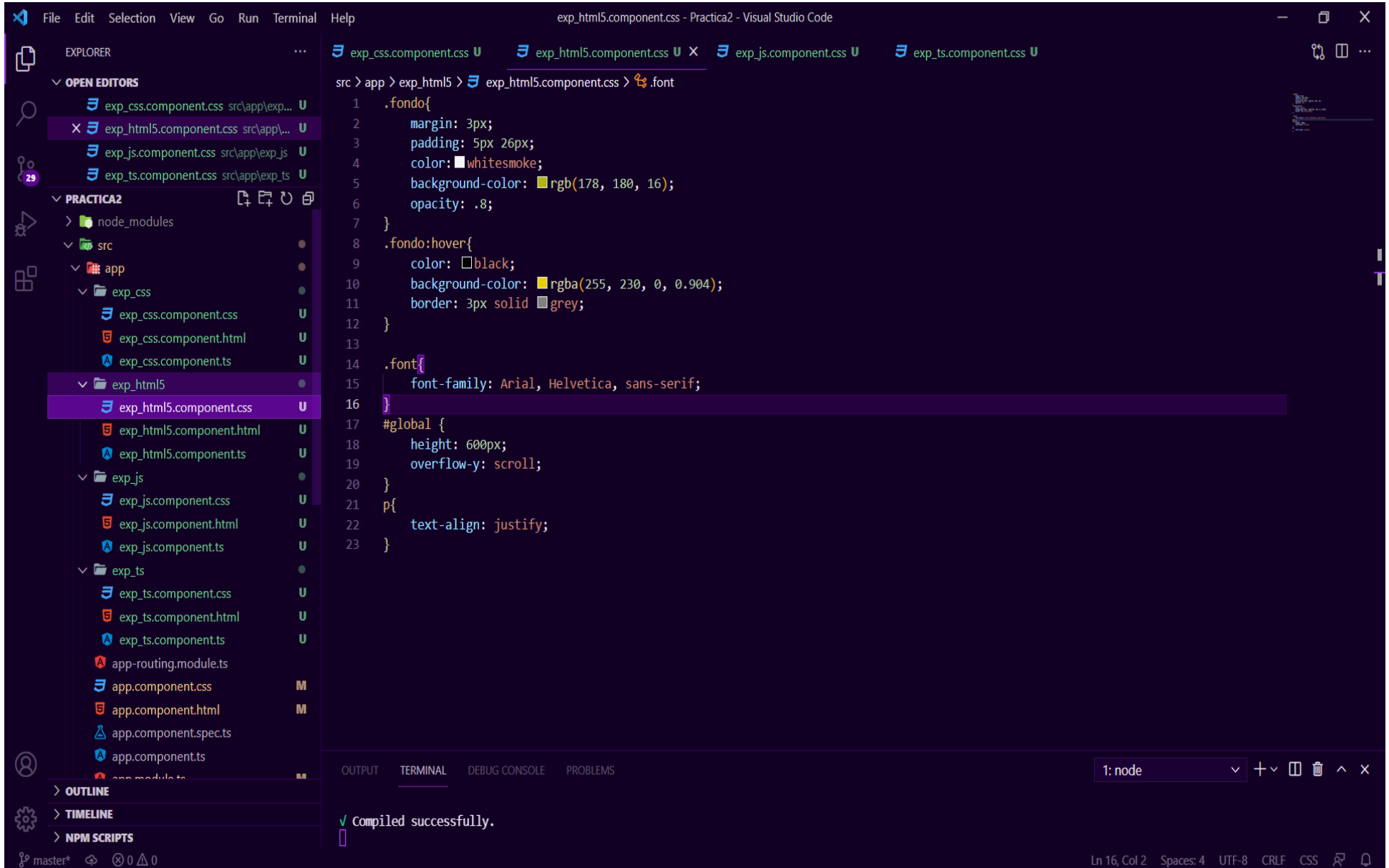
✓ Compiled successfully.

Ln 21, Col 73 Spaces: 4 UTF-8 CRLF HTML

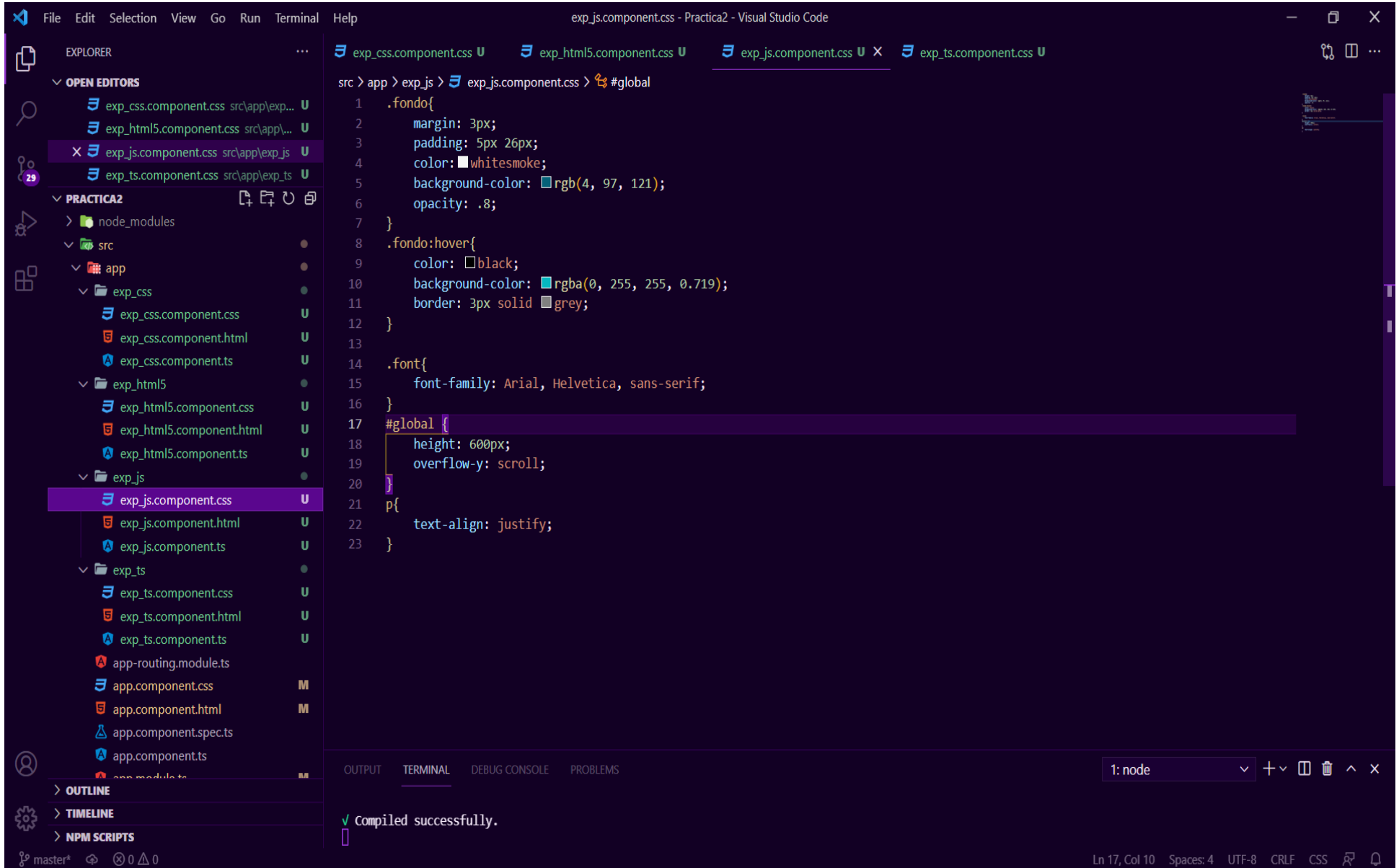
Agregamos los estilos dentro del archivo exp_css.component.css para que se vea más llamativo.



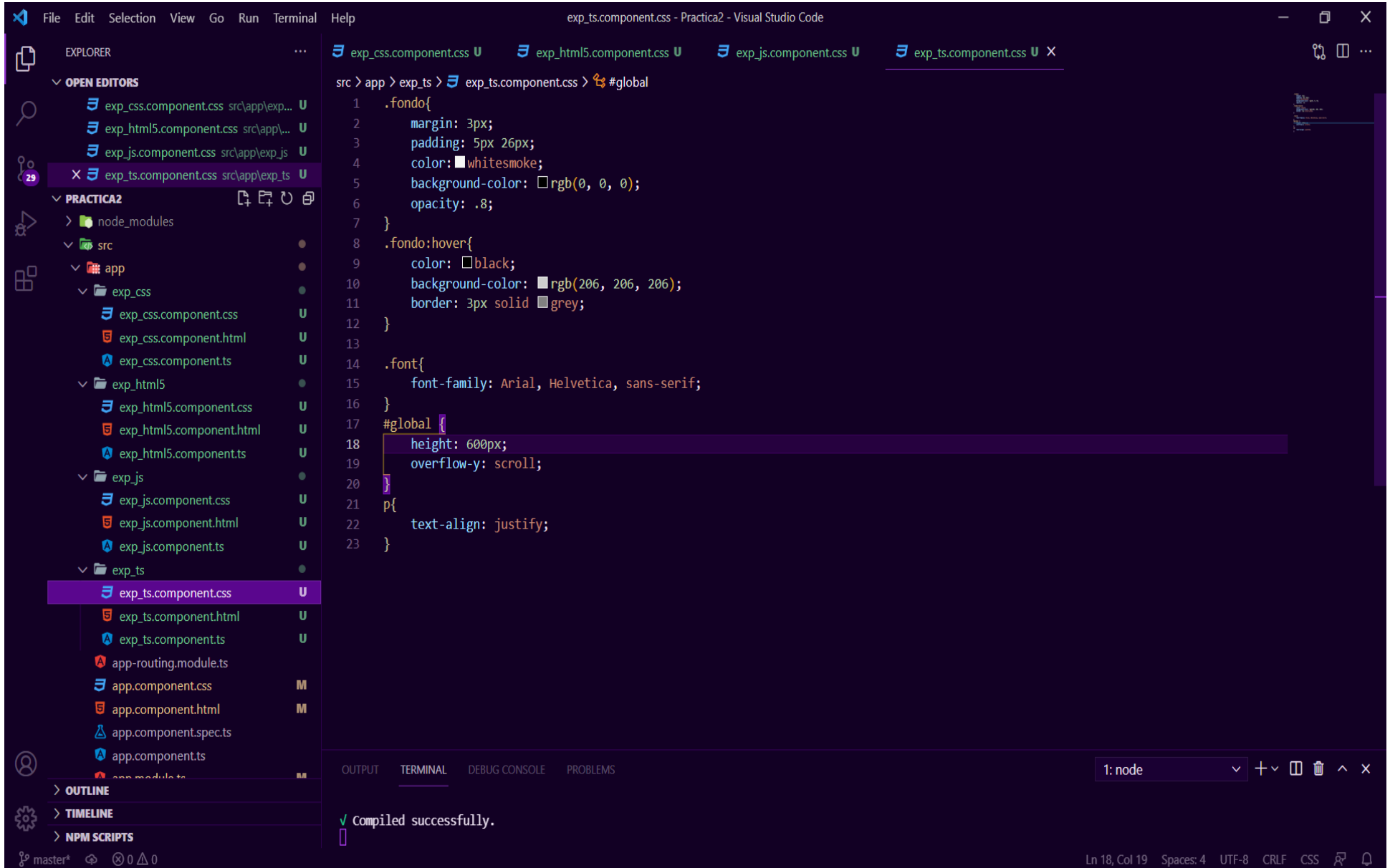
Agregamos los estilos dentro del archivo exp_html5.component.css para que se vea más llamativo.



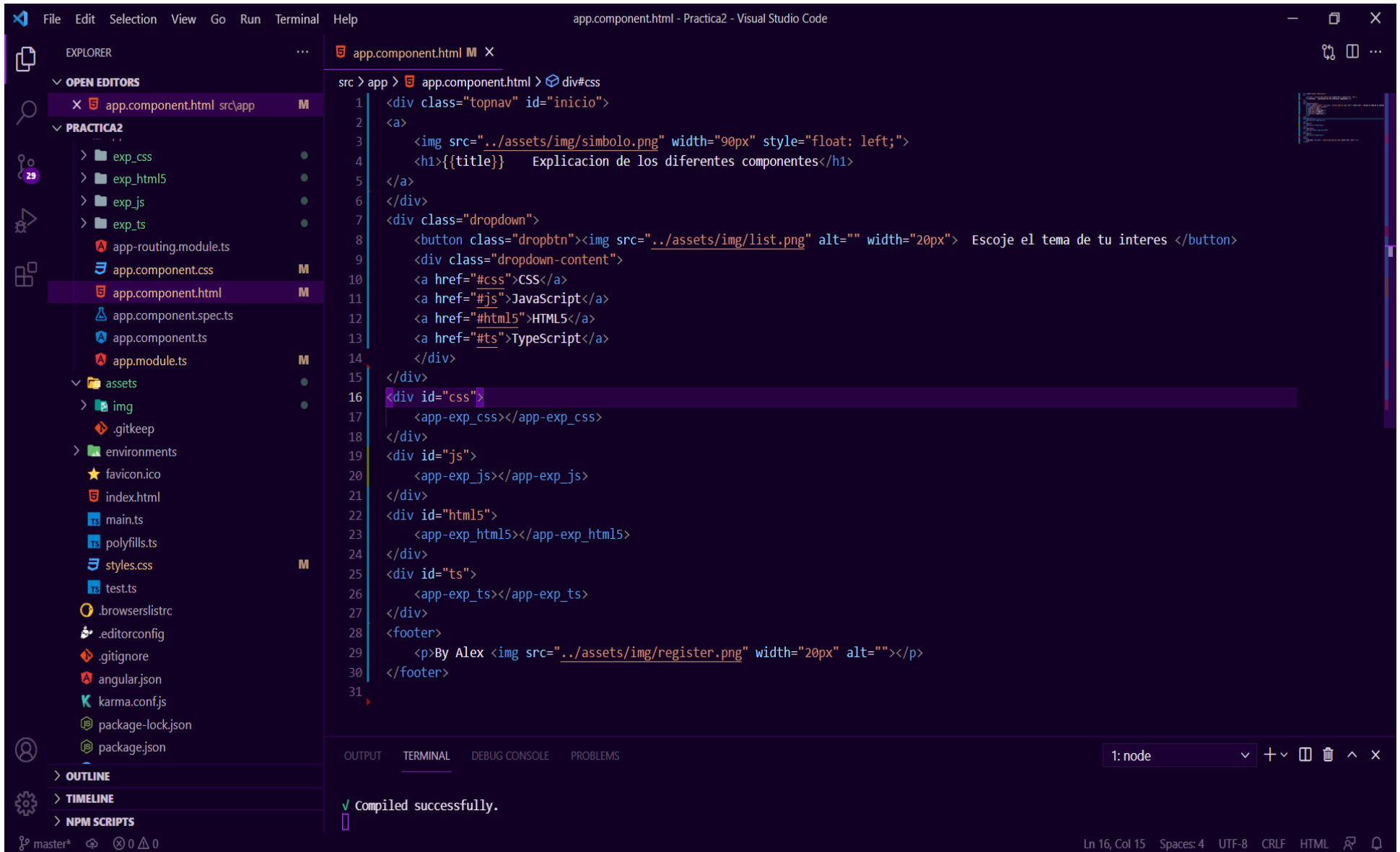
Agregamos los estilos dentro del archivo exp_js.component.css para que se vea más llamativo.



Agregamos los estilos dentro del archivo exp_ts.component.css para que se vea más llamativo.



Ahora vamos a agregar los componentes dentro del apartado de app.component.html para poder visualizarlos dentro del proyecto.



```
src > app > app.component.html > div#css
1 <div class="topnav" id="inicio">
2 <a>
3   
4   <h1>{{title}} Explicacion de los diferentes componentes</h1>
5 </a>
6 </div>
7 <div class="dropdown">
8   <button class="dropbtn"> Escoje el tema de tu interes </button>
9   <div class="dropdown-content">
10    <a href="#css">CSS</a>
11    <a href="#js">JavaScript</a>
12    <a href="#html5">HTML5</a>
13    <a href="#ts">TypeScript</a>
14   </div>
15 </div>
16 <div id="css">
17   <app-exp_css></app-exp_css>
18 </div>
19 <div id="js">
20   <app-exp_js></app-exp_js>
21 </div>
22 <div id="html5">
23   <app-exp_html5></app-exp_html5>
24 </div>
25 <div id="ts">
26   <app-exp_ts></app-exp_ts>
27 </div>
28 <footer>
29   <p>By Alex </p>
30 </footer>
31
```

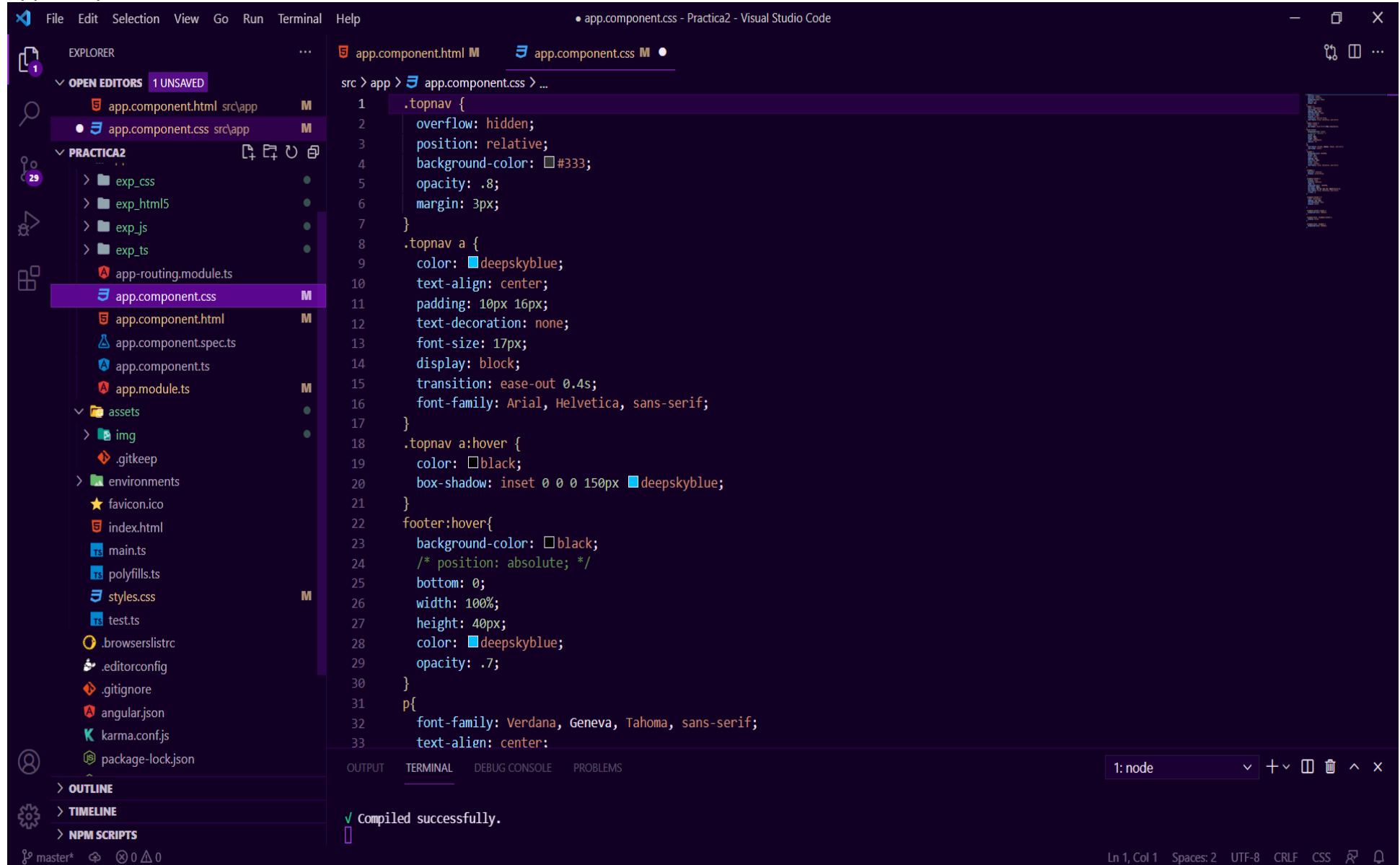
OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

1: node

✓ Compiled successfully.

Ln 16, Col 15 Spaces: 4 UTF-8 CRLF HTML

Por último, agregamos los estilos de la pagina principal para que tenga un mayor aprecio visual. Esto lo hacemos dentro de app.component.css.



Visual Studio Code interface showing a CSS file being edited.

EXPLORER

- OPEN EDITORS (1 UNSAVED)
 - app.component.html src\app
 - app.component.css src\app
- PRACTICA2
 - exp_css
 - exp_html5
 - exp_js
 - exp_ts
 - app-routing.module.ts
 - app.component.css
 - app.component.html
 - app.component.spec.ts
 - app.component.ts
 - app.module.ts
 - assets
 - img
 - .gitkeep
 - environments
 - favicon.ico
 - index.html
 - main.ts
 - polyfills.ts
 - styles.css
 - test.ts
 - .browserslistrc
 - .editorconfig
 - .gitignore
 - angular.json
 - karma.conf.js
 - package-lock.json
- OUTLINE
- TIMELINE
- NPM SCRIPTS

app.component.css

```
src > app > app.component.css > footer: hover
34 }
35 .dropbtn {
36   background-color: #4CAF50;
37   margin: 3px;
38   color: white;
39   padding: 16px;
40   font-size: 16px;
41   border: none;
42   cursor: pointer;
43   font-family: Arial, Helvetica, sans-serif;
44 }
45
46 .dropdown {
47   position: relative;
48   display: inline-block;
49 }
50
51 .dropdown-content {
52   display: none;
53   position: absolute;
54   right: 0;
55   background-color: #4CAF50;
56   min-width: 160px;
57   box-shadow: 0px 8px 16px 0px rgba(0,0,0,0.2);
58   font-family: Arial, Helvetica, sans-serif;
59   z-index: 1;
60 }
61
62 .dropdown-content a {
63   color: whitesmoke;
64   padding: 12px 16px;
65   text-decoration: none;
```

OUTPUT

```
1: node
✓ Compiled successfully.
```

Ln 23, Col 29 Spaces: 2 UTF-8 CRLF CSS

Ya solo falta visualizar lo cambios en el navegador.



The screenshot shows a web browser window with two tabs: 'Practica2' and 'Iconos de Sitio web - 46,361'. The address bar shows 'localhost:4200/#ts'. The page has a dark background with a large, stylized 'EWAIS' logo in the background. The main heading is 'Practica2 Explicacion de los diferentes componentes' in blue. Below the heading, there is a green button that says 'Escoje el tema de tu interes'. The content area has a green background and features a section titled 'Explicacion de CSS' with a small CSS icon. The text explains that CSS (Cascading Style Sheets) is a language for styling HTML elements and separates content from visual representation. It also includes a section titled 'Como funciona' which describes the syntax of CSS, mentioning selectors, declarations, and the use of property names and values separated by colons and enclosed in curly braces.

Practica2 Explicacion de los diferentes componentes

Escoje el tema de tu interes

Explicacion de CSS

CSS (en inglés Cascading Style Sheets) es lo que se denomina lenguaje de hojas de estilo en cascada y se usa para estilizar elementos escritos en un lenguaje de marcado como HTML. CSS separa el contenido de la representación visual del sitio.

Como funciona

CSS utiliza una sintaxis simple basada en el inglés con un conjunto de reglas que la gobiernan. Como mencionamos anteriormente, HTML no fue hecho con la intención de utilizar elementos de estilo, sino solo para el marcado de la página. Fue creado simplemente para describir el contenido. Pero, ¿cómo le aplicas un estilo al párrafo? La estructura de sintaxis CSS es bastante simple. Cuenta con un selector y un bloque de declaración. Primero seleccionas un elemento y luego declaras lo que quieres hacer con él. Bastante sencillo, ¿verdad? Sin embargo, hay reglas que debes recordar. Las reglas de la estructura son bastante simples, así que no te preocupes. El selector apunta al elemento HTML que deseas estilizar. El bloque de declaración contiene una o más declaraciones separadas por punto y coma. Cada declaración incluye un nombre de propiedad CSS y un valor, separados por dos puntos. Una declaración CSS siempre termina con un punto y coma, y los bloques de declaración están rodeados por llaves.

Practica2

Iconos de Sitio web - 46,361

+

← → ↻ 🏠

localhost:4200/#ts

... 🛡️ ☆

⬇️ 📄 📖 👤 🔒 ☰



Practica2 Explicacion de los diferentes componentes

Escoje el tema de tu interes

Explicacion

CSS

JavaScript

HTML5

TypeScript

CSS (en Cascading Style Sheets) es lo que se denomina lenguaje de hojas de estilo en cascada y se usa para estilizar elementos de un lenguaje de marcado como HTML. CSS separa el contenido de la representación visual del sitio.

Como funciona

CSS utiliza una sintaxis simple basada en el inglés con un conjunto de reglas que la gobiernan. Como mencionamos anteriormente, HTML no fue hecho con la intención de utilizar elementos de estilo, sino solo para el marcado de la página. Fue creado simplemente para describir el contenido. Pero, ¿cómo le aplicas un estilo al párrafo? La estructura de sintaxis CSS es bastante simple. Cuenta con un selector y un bloque de declaración. Primero seleccionas un elemento y luego declaras lo que quieres hacer con él. Bastante sencillo, ¿verdad? Sin embargo, hay reglas que debes recordar. Las reglas de la estructura son bastante simples, así que no te preocupes. El selector apunta al elemento HTML que deseas estilizar. El bloque de declaración contiene una o más declaraciones separadas por punto y coma. Cada declaración incluye un nombre de propiedad CSS y un valor, separados por dos puntos. Una declaración CSS siempre termina con un punto y coma, y los bloques de declaración están rodeados por llaves.

localhost:4200/#js

Practica2

Iconos de Sitio web - 46,361

localhost:4200/#js

...

Explicacion de JavaScript

JavaScript es el único lenguaje de programación que funciona en los navegadores de forma nativa (lenguaje interpretado sin necesidad de compilación). Por tanto se utiliza como complemento de HTML y CSS para crear páginas webs.

Como funciona

Con este lenguaje de programación del lado del cliente (no en el servidor) podemos crear efectos y animaciones sin ninguna interacción, o respondiendo a eventos causados por el propio usuario tales como botones pulsados y modificaciones del DOM (document object model). Por tanto, nada tiene que ver con el lenguaje de programación Java, ya que su principal función es ayudar a crear páginas webs dinámicas. El código de programación de JavaScript se ejecuta en los navegadores, ya sean de escritorio o móviles, ya sean Android o Iphone. Sirve para exactamente lo mismo, da igual en el tipo de dispositivo que se ejecute el navegador. JavaScript ahora mismo es el lenguaje más popular. De hecho, desde hace años se ha creado una versión que es capaz de ser ejecutada también en el lado del servidor (Node JS). Por tanto, ahora mismo se ejecuta JavaScript en los navegadores y en los servidores, creando a su alrededor una amplísima comunidad de desarrolladores casi full-stack. JavaScript del lado del servidor compete en igualdad de condiciones con PHP, por ejemplo.

Veamos un ejemplo:

```
<html>
<head>
<script type="text/javascript">
function getAction()
{
    var x=document.forms.myForm;
    alert(x.action);
}
```

Practica2

Iconos de Sitio web - 46,361

localhost:4200/#html5

HTML5

Explicacion de HTML5

HTML5 es un estándar que sirve como referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado HTML) para la definición de contenido de una página web, como texto, imágenes, vídeos, juegos, entre otros.

Como funciona

Vamos a ver cómo es proceso cuando se solicita una página HTML a través del navegador. El proceso es el siguiente:

```
graph LR; Browser[Browser] -- "http://...../index.html" --> Server[Server]; Server -- "HTML Content" --> Browser;
```

Explicacion de TypeScript

TypeScript es un superconjunto de JavaScript con un tipado que se compila para JavaScript puro. Sigue el patrón javascript y también es un lenguaje multiparadigma, sin embargo, agrega funcionalidad y sintaxis orientada a objetos, clases y escritura vistos en lenguajes como C- y Java.

Como funciona

El sistema de tipos de Typescript realiza una formalización de los tipos de Javascript, mediante una representación estática de sus tipos dinámicos. Esto permite a los desarrolladores definir variables y funciones tipadas sin perder la esencia de Javascript. Poder definir los tipos durante el tiempo de diseño nos ayuda a evitar errores en tiempo de ejecución, como podría ser pasar el tipo de variable incorrecto a una función. Además de los tipos String y Number admite los siguientes tipos básicos:

- Boolean: tipo de dato lógico que representa verdadero o falso.
- Array: tipo de dato estructurado que permite almacenar una colección de elementos.
- Tuple: similar al array, pero con un número fijo de elementos escritos.
- Enum: representa al tipo enumeración.
- Any: indica que la variable puede ser de cualquier tipo. Es muy útil a la hora de trabajar con librerías externas.
- Void: indica que una función no devolverá ningún valor.
- Never: este tipo representa el tipo de valores que nunca se producen.

Veamos un ejemplo: