Cómo usar el proyecto Base de Angular

Tabla de Contenido

1.- Introducción.

El proyecto base Angular se desarrolló para que cuando se les presente un nuevo proyecto a los técnicos del área de Ingeniería de Software éstos puedan iniciarlo en el framework Angular con solamente copiar, pegar y cambiar su nombre. Entre las ventajas del proyecto base se encuentran:

- La rapidez con el que el técnico puede empezar a trabajar debido a que ya no tiene que configurar un proyecto desde cero.
- El técnico tiene herramientas previamente desarrolladas por el mismo departamento de Ingeniería de Software, con la opción de aportar nuevas.
- Se logra homologar todos los proyectos Angular desarrollados en Ingeniería de Software.

El proyecto base está guardado y versionado mediante la herramienta SVN Tortoise de la siguiente manera:

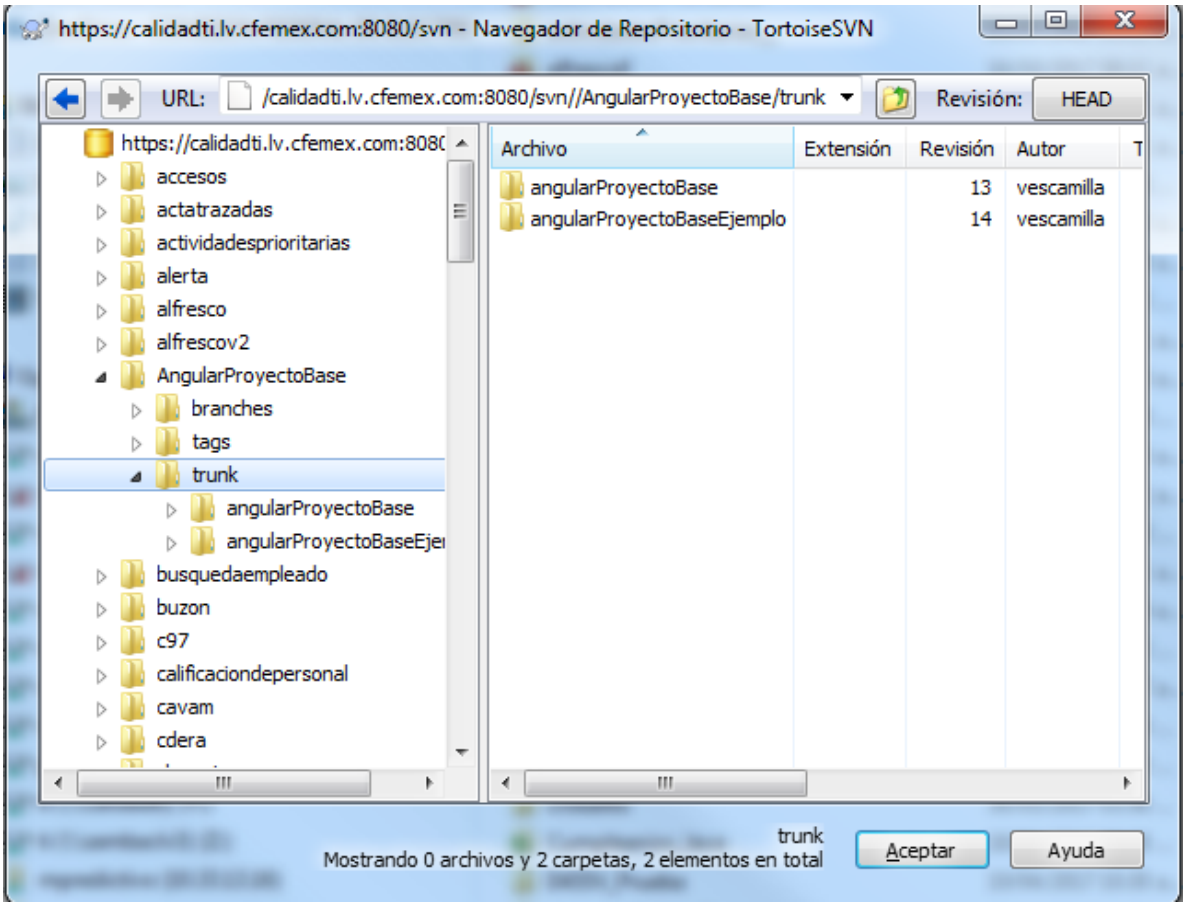

Ilustración 1. Proyecto base en SVN

Como se puede observar se dividió el proyecto base en dos carpetas, cada una es un proyecto de Angular independiente, funcional y fue desarrollado con el IDE Webstorm de JetBrains. A continuación, se presenta una pequeña explicación sobre cada subproyecto.

- angularProyectoBase. Este proyecto cumple al cien por ciento con la finalidad de ser descargado y renombrado para empezar un nuevo desarrollo, ya que no contiene ningún componente o vista, más que las cosas que se generan por defecto al hacer un nuevo proyecto Angular con la terminal de cli. A pesar de esto, se cuenta con **la carpeta "IS_modules"** que contiene todos los módulos (herramientas) desarrolladas por Ingeniería de Software, aunque no están implementados si están listos para ser usados.
- angularProyectoBaseEjemplo. Este proyecto tiene la finalidad de servir como ejemplo mostrando algunas funcionalidades básicas de Angular, de hecho, sólo contiene los archivos fuente, por lo que si se quiere correr este proyecto se tienen que copiar los archivos de configuración del otro subproyecto. También contiene **la carpeta "IS_modules"** para que se prueben las herramientas ya creadas.

2. Forma de uso

1. Descargar el proyecto angularProyectoBase de SVN con la opción de Obtener (Ilustraciones 2 y 3).
2. Copiar y pegar el proyecto descargado.
3. Renombrar la carpeta del proyecto copia por el nombre del nuevo proyecto a desarrollar (Ilustración 4).
4. Abrir el proyecto con el IDE de su elección, en este ejemplo se usó el ya mencionado Webstorm.
5. En este punto se puede renombrar el nombre del proyecto que el IDE guarda y es opcional, en el caso de Webstorm es con clic derecho en la carpeta más superior del diagrama de proyecto, eligiendo refractor y rename. A continuación, se siguen los pasos de las Ilustraciones 5, 6 y 7.
6. Se abre el archivo ".angular-CLI.json" y se debe de renombrar el proyecto como en la Ilustración 8.
7. De igual manera, se abre el archivo "package.json" y se renombra el proyecto como en la Ilustración 9.
8. En este punto ya se puede utilizar el proyecto, pero no está de más probarlo. Para esto, se abre la terminal de cli que incluye el IDE y se corre el proyecto con la sentencia "ng serve" como se muestra en la Ilustración 10.

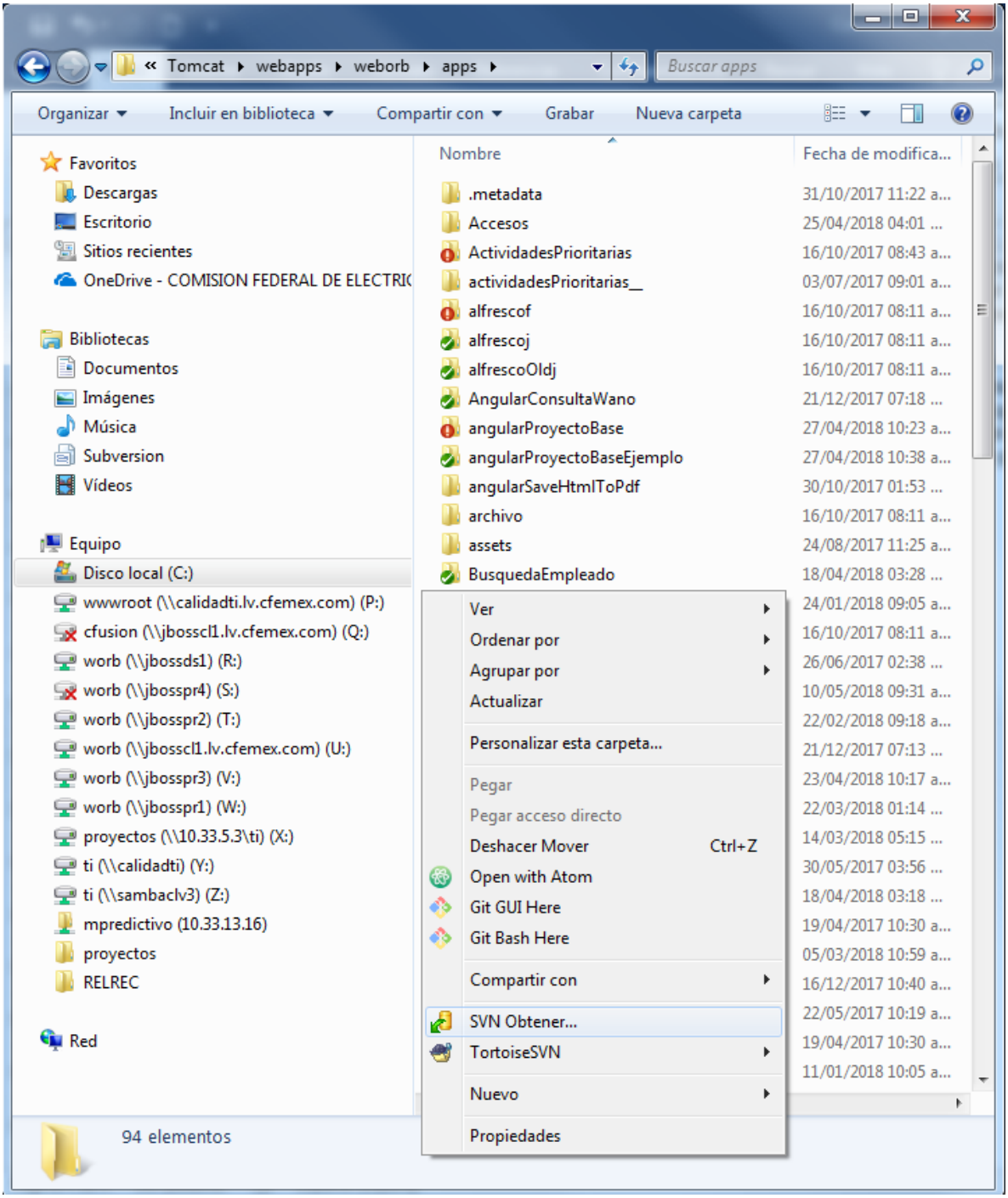

Ilustración 2. Descargar de SVN 1.

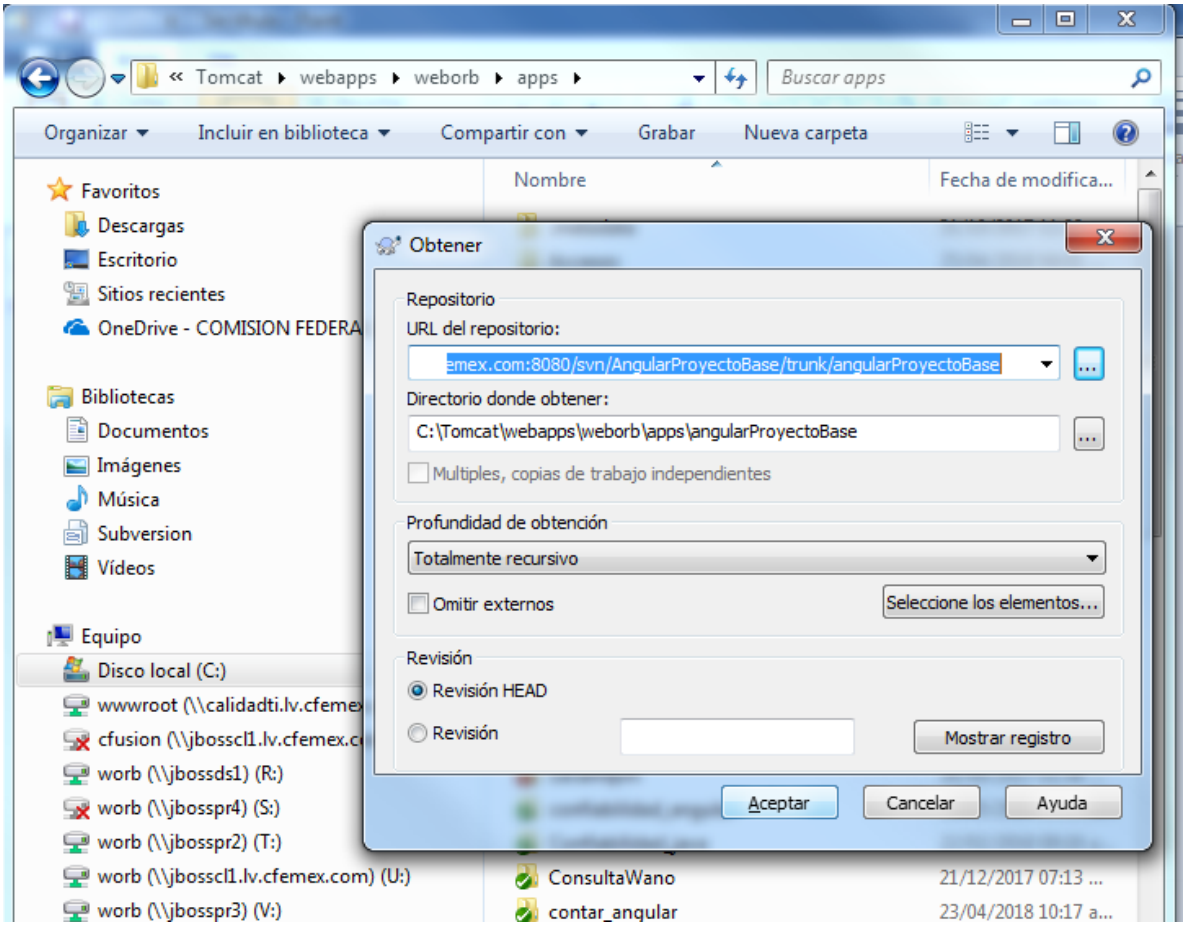

Ilustración 3. Descargar de SVN 2.

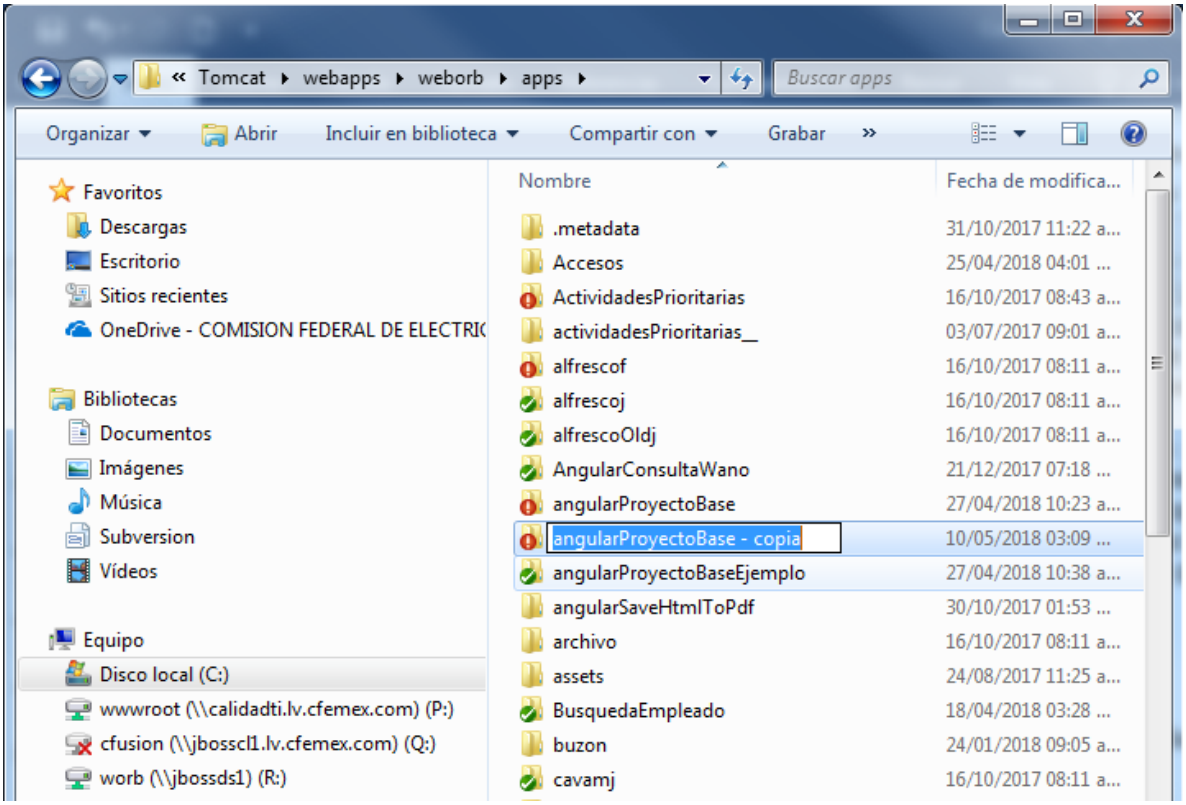

Ilustración 4. Renombrar la carpeta del proyecto.

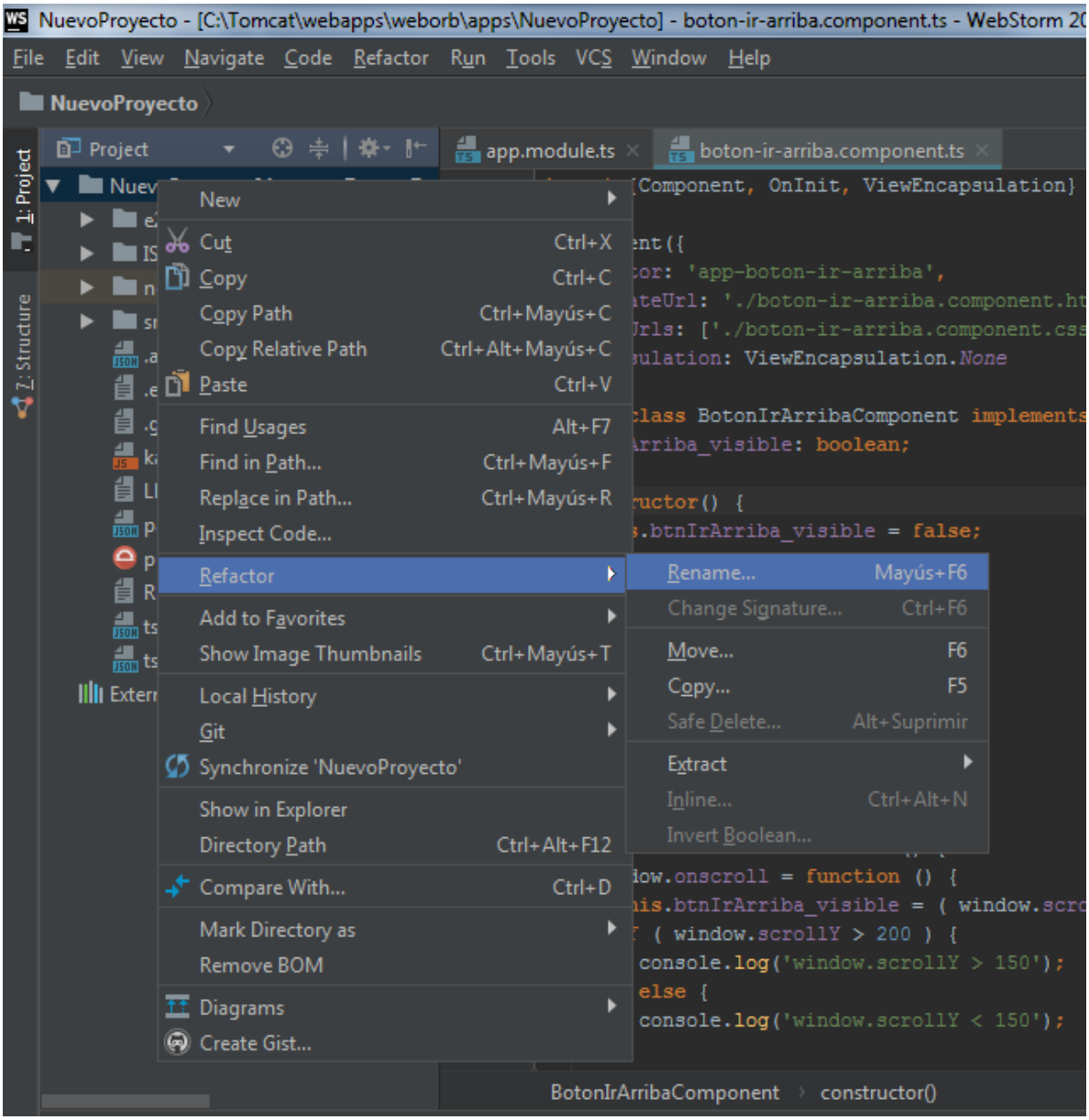

Ilustración 5. Renombrar el proyecto dentro del IDE.

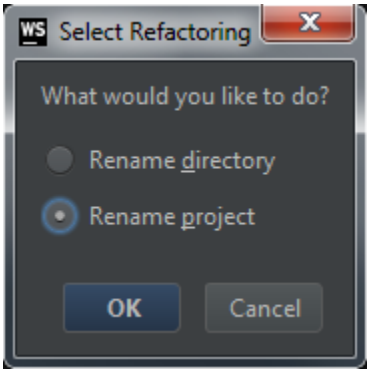

Ilustración 6. Renombrar el proyecto dentro del IDE 2.

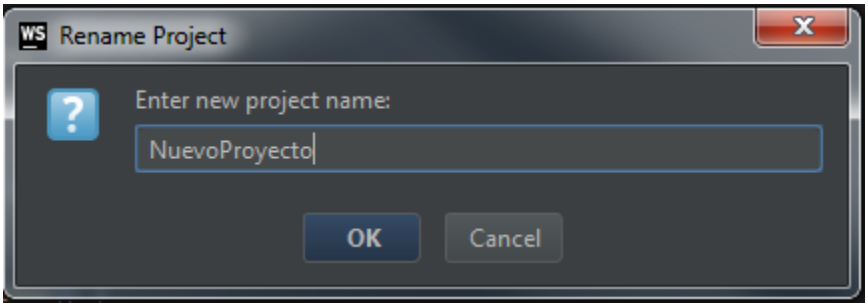

Ilustración 7. Renombrar el proyecto dentro del IDE 3.

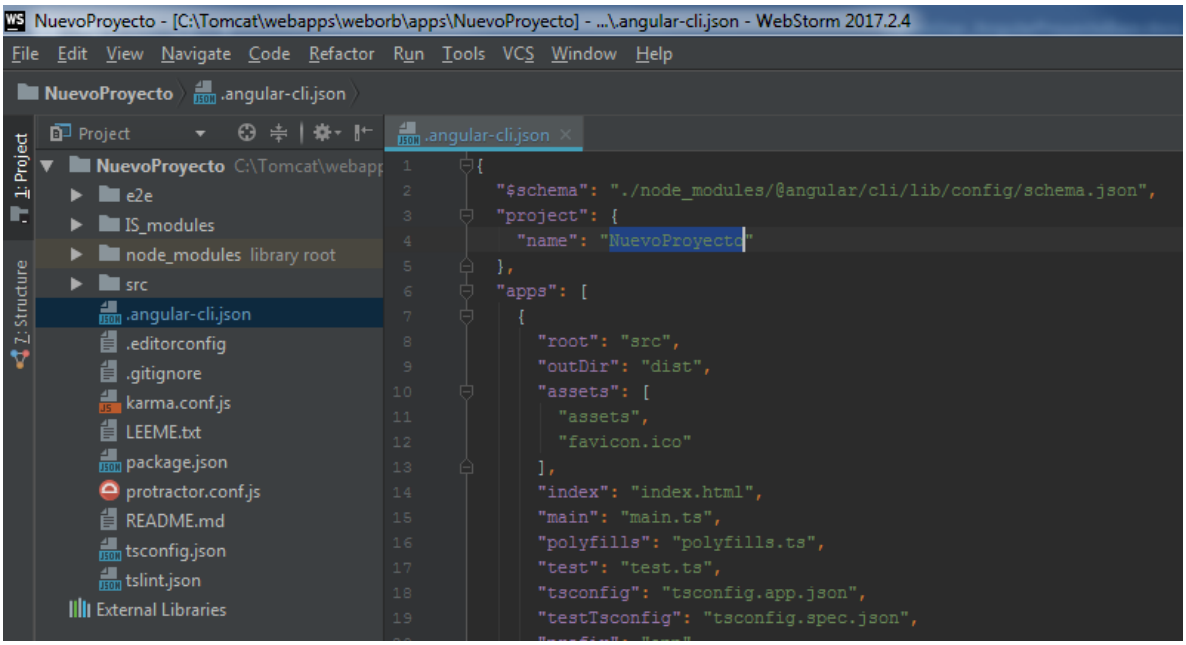

Ilustración 8. Renombrar el proyecto en el archivo angular-cli.json.

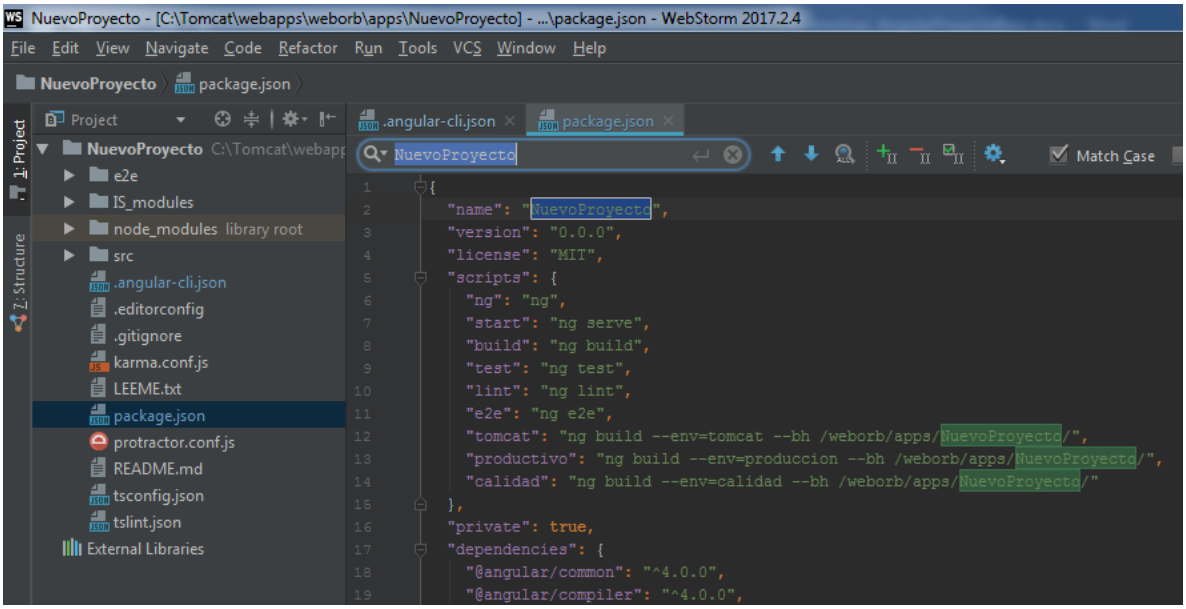

Ilustración 9. Renombrar el proyecto en el archivo package.json.

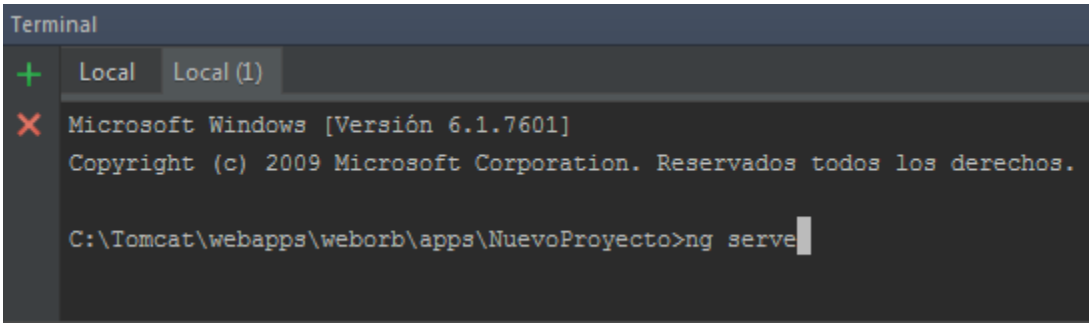

Ilustración 10. Probar el proyecto.

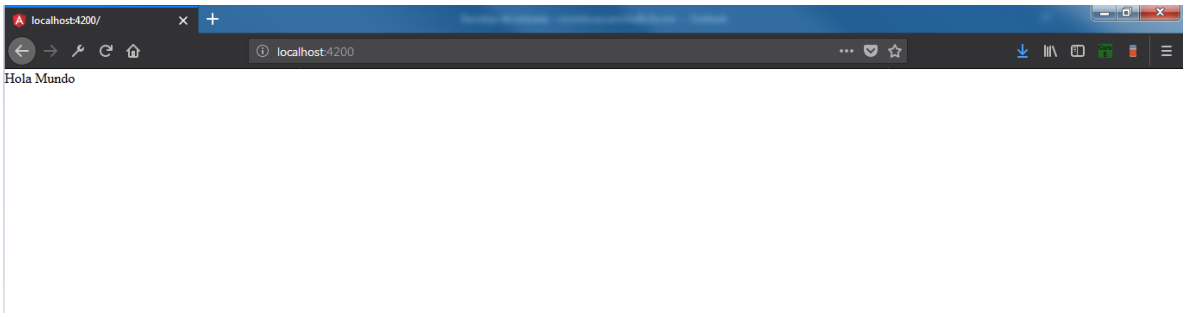

Ilustración 11. Proyecto corriendo en el navegador.

3. Estructura del proyecto base

En esta sección se revisará la estructura del proyecto base por el nivel de sus carpetas.

3.1 Nivel de configuración

Aquí se encuentran los archivos de configuración del proyecto, son los que el framework utiliza para indicarle su funcionamiento a un navegador, como por ejemplo Google Chrome.

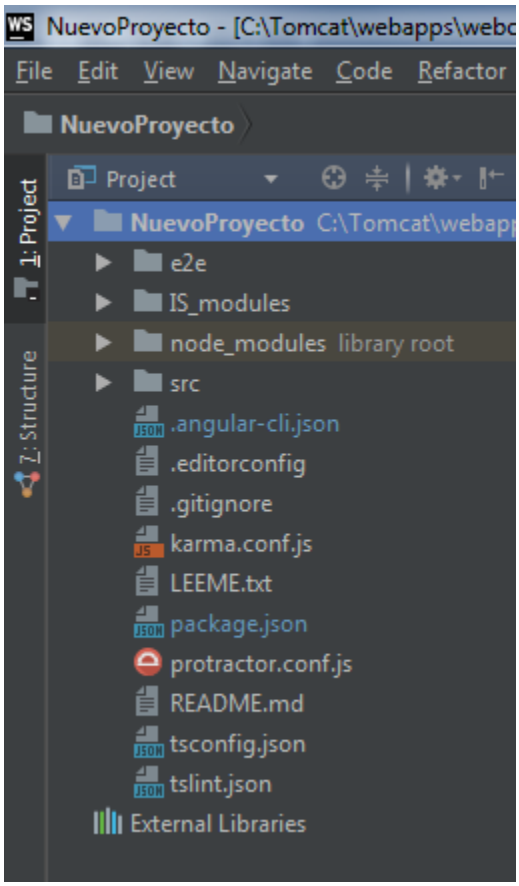

Ilustración 12. Estructura del proyecto - configuración.

La mayoría de los archivos no son de importancia para el desarrollador porque no hay necesidad de modificarlos, además se crearon de forma automática cuando se corrió el comando de cli “ng new nombre-proyecto” en la terminal.

A continuación, se definen de manera breve los archivos y carpetas que son importantes conocer:

- **Carpeta IS_modules**. Aquí los técnicos de la oficina de Ingeniería de Software pueden guardar sus módulos, los cuales son programas independientes a cualquier proyecto y que pueden ser implementados de manera sencilla y rápida.
- **Carpeta node_modules**. Este es el repositorio de Angular para guardar todas sus librerías propias y externas. Las librerías externas son las que cualquier desarrollador puede descargar mediante la terminal de cli, por ejemplo, si un proyecto necesita convertir archivos PDF a XLS entonces el desarrollador puede buscar librerías de uso gratuito en los sitios de

nodeJS que cumplan con los requisitos, y descargar una mediante los comandos de la terminal cli que indique su documentación.

- **Carpeta src**. En esta carpeta se guarda el código fuente del proyecto, se profundizará más adelante.
- **Archivo .angular-cli.json**. Es aquí donde el framework le indica al navegador las rutas de algunos archivos importantes para la visualización de la página:
  - ❖ Nombre del proyecto.
  - ❖ Ruta de la carpeta de imágenes (assets) y el nombre del archivo favicon.
  - ❖ Archivos de estilo CSS externos al código fuente (como los de primeng).
  - ❖ Archivos de JavaScript externos al código fuente (como el de Weborb).
  - ❖ Ruta de los archivos de environment, los cuales son los que usan los proyecto de la CLV para indicar las rutas de los servidor-es locales, de calidad y productivo.
- **Archivo package.json**. Principalmente indica el número de la versión del framework y las versiones de todas las librerías con las que está compilado el proyecto. También, indica el nombre del proyecto, su versión y los comandos internos del framework que utiliza para ubicar la ruta física del environment.

3.2 Nivel más exterior del código fuente

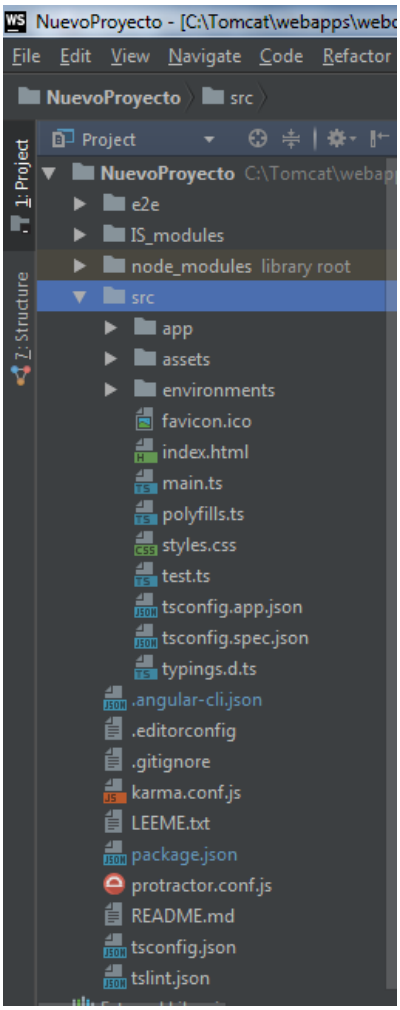

Ilustración 13. Nivel exterior del código fuente.

A pesar de que en este nivel ya se pueden encontrar los primeros archivos del código fuente del proyecto, para el navegador y para el framework siguen siendo archivos de configuración. Esto es debido a que lo primero que abre el navegador es el archivo **index.html**, el cual contiene la estructura de etiquetas de todo el sitio web: <html>, <head> y <body>, y que, además, con ayuda de los archivos que se encuentran el nivel de configuración del proyecto arman la página web. Cabe destacar que dentro de index.html está la primera etiqueta Typescript: **<app-root>**, que representa al componente raíz que se encuentra dentro de la carpeta de app.

De igual manera, los archivos **styles.css y main.ts** son los primeros en cargarse en el navegador; el primero contiene estilos para todo el proyecto y el segundo contiene la primera clase Typescript que es la primera en ejecutarse (viéndolo del lado de programación). Ambos archivos afectan al index.html, pero se recomienda que el desarrollador no los tome mucho en cuenta y que se concentre a trabajar desde el componente raíz ubicado en el siguiente nivel de estructura.

Los otros archivos siguen siendo puramente de configuración del framework, pero es importante mencionar que aquí también está el archivo **favicon.ico:** el ícono de la aplicación que aparece en la pestaña del navegador y que es visible a todos los usuarios.

Por último, se tienen las siguientes carpetas:

- **App.** Contiene el siguiente nivel de la aplicación, que es donde los desarrolladores van a ir construyendo el proyecto.
- **assets.** Aquí se pueden guardar las imágenes y otros archivos que utilice el proyecto.
- **environments.** Esta carpeta guarda los archivos que le indican al navegador la ruta que debe acceder para los diferentes webservices desarrollados en la CLV, cada servidor tiene su archivo y encima de todos ellos se encuentra el “environment.ts”, el cual es el que deben importar los desarrolladores en sus webservices: porqué dependiendo de dónde se esté corriendo la aplicación, Angular se encargará de tomar la ruta correcta. Por ejemplo, si el proyecto se corre en JbossCl1 entonces Angular selecciona de manera automática la ruta guardada en el archivo “environment.calidad.ts” con ayuda de este archivo y de los archivos de configuración ya mencionados anteriormente.

3.3 Nivel del componente raíz.

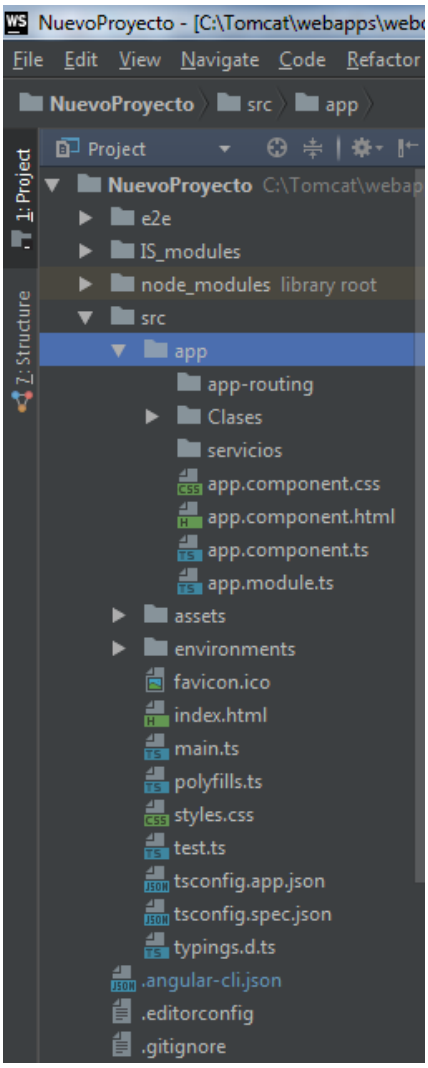

Ilustración 14. Nivel del componente raíz.

Desde este nivel se recomienda empezar a desarrollar el nuevo proyecto, ya que cuando se crean nuevos componentes, módulos, servicios, o cualquier otra cosa mediante la terminal de cli las cosas van apareciendo en dentro de app.

El componente raíz se encuentra aquí, y como todos los componentes se compone de su hoja de *estilos CSS*, su hoja de *etiquetas HTML* y su hoja de *programación Typescript*. Por otro lado, es el primero en cargarse y ejecutarse, por lo que si se requiere poner instrucciones que trabajen siempre que se visite el sitio se recomienda ponerlas aquí. De igual manera los estilos del componente raíz afectan a todo el proyecto, haciendo que cualquier componente puede tomar sus clases. En su HTML se pueden poner las etiquetas de los componentes que se quieran visualizar, o también poner la etiqueta <router-outlet> cuando se tenga más de una vista.

El archivo **module.ts** que se encuentra al mismo nivel del componente raíz tal vez sea aún más importante que éste último, porque es el que le indica a todo el proyecto cuáles librerías va a utilizar de las que se encuentran en IS_modules y en node_modules, así como también los componentes, módulos y servicios. Esto es para que cuando un usuario visite el sitio, no se vuelva lento y pesado al cargar todo lo que exista dentro de las carpetas del proyecto. A continuación, se detallan las partes que lo conforman con una Ilustración:

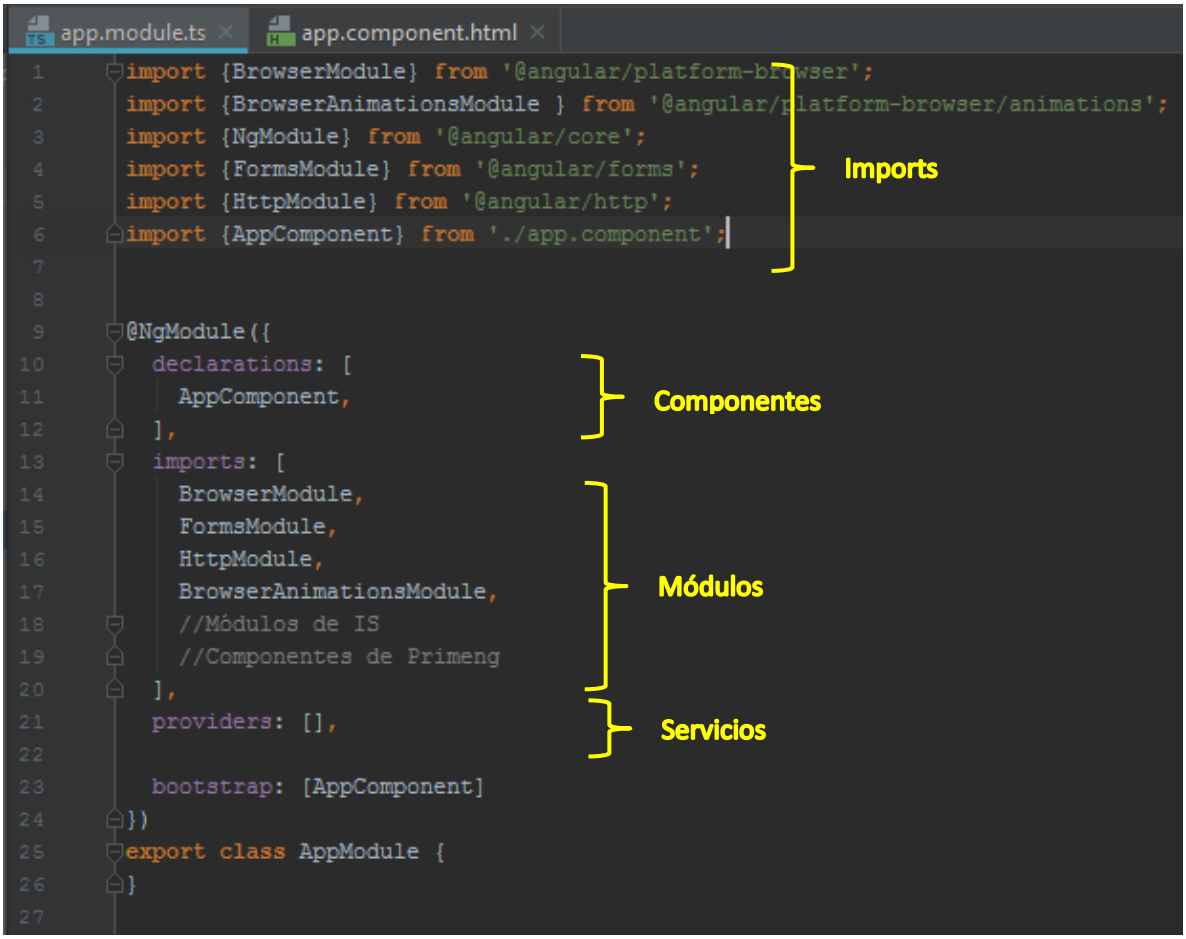

Ilustración 15. Archivo module.ts.

Por último, en este proyecto base se añadieron algunas **carpetas a modo de recomendación**, aquí se pueden guardar los archivos: el manejador de vistas app-routing, clases BEANS, clases de servicios, y un archivo para guardar todas las constantes que puedan ser utilizadas por uno o más componentes.

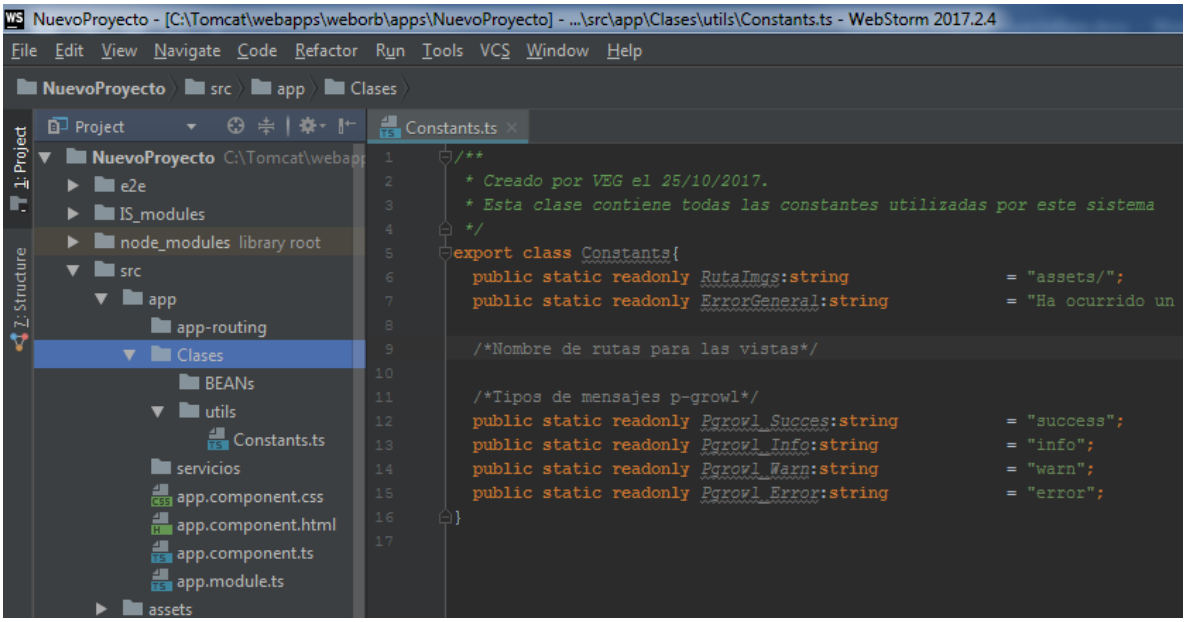

Ilustración 16. Sugerencias para un proyecto organizado.

4. Proyecto base de ejemplo

Como se mencionó al principio, este proyecto sirve como ejemplo de lo que se puede hacer en Angular, mediante implementaciones de funcionalidades de algunos desarrollos que se han tenido en Ingeniería de Software.

Para utilizar este proyecto se tienen que haber descargado ambos proyectos base, después se deben de copiar todos los archivos que no tenga el proyecto base de ejemplo del otro proyecto y pegarlos dentro de la carpeta angularProyectoBaseEjemplo.

Este proyecto contiene varias vistas, cada una tiene el propósito de mostrar un ejemplo, su estructura es la de la Ilustración 17. Hay que señalar que en este documento no se va a profundizar en cómo funciona cada vista y cada componente.

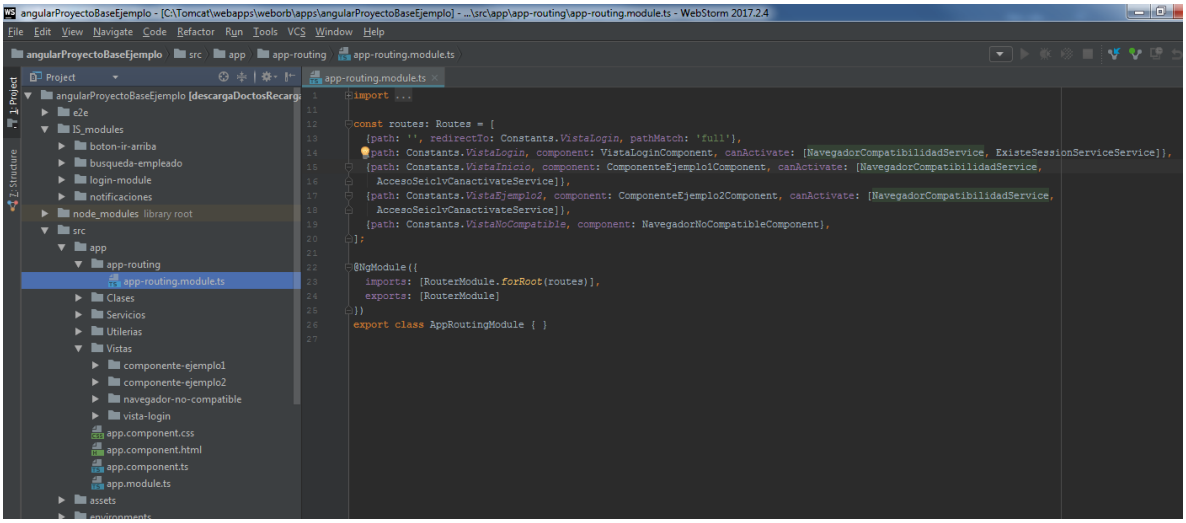

Ilustración 17. Proyecto de ejemplo: estructura.

En la Ilustración se puede apreciar el archivo de app-routing abierto, el cual es el manejador de las vistas y con él se pueden identificar todas las que están, por poner en términos simples, “registradas” dentro del proyecto. Sin ir más lejos, si se requiere una vista nueva, se tiene que dar de alta en este archivo para que posteriormente se pueda mandar a llamar en algún componente. También se puede observar que este manejador posee ejemplos de los archivos de servicio **canActivate**, con los cuales se puede ejecutar una instrucción que determina si se muestra o no cada vista antes que cargue.

4.1 Componente raíz

Aunque este no es una vista si tiene varias cosas implementadas que afectan o pueden ser utilizadas por todo el proyecto:

- El ícono animado de carga (Loading…) del proyecto.
- La estructura para mensajes temporales.
- La estructura para mensajes con pop-ups.
- El menú deslizable sidebar de Primeng.
- El encabezado de la aplicación que sólo aparece si el usuario ha iniciado sesión.
- La etiqueta <router-outlet> que manda a llamar al manejador de vistas.

4.2 Vista Login

Es la vista que aparece por defecto si no se ha iniciado una sesión. Esta vista implementa el módulo de login, y le indica que es un proyecto con acceso público mediante el Input *esLoginPublico*, es decir, que no es necesario que el proyecto y el usuario estén dados de alta en las tablas de la CLV que sirven para llevar el inicio de sesión y los módulos de acceso.

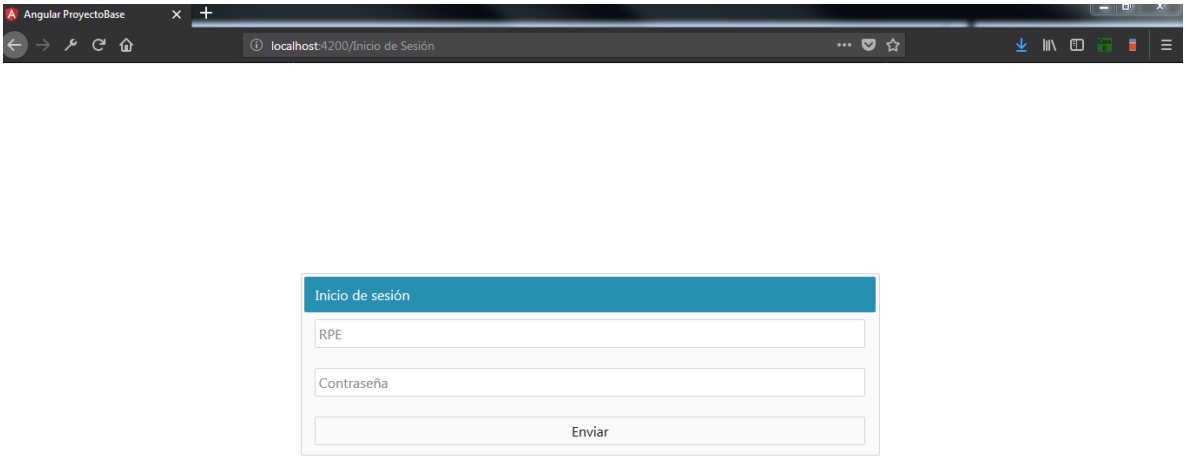

Ilustración 18. Vista Login.

4.3 Vista Inicio

Es la vista principal del proyecto. Aquí se pueden apreciar los siguientes ejemplos:

- Uso de ordenamiento responsivo mediante clases de Primeng (ui-grid).
- Uso del componente panel de Primeng.
- Uso del cambio de vista mediante un botón.
- En la hoja CSS tiene una sentencia para cambiar el color del panel de Primeng.

Ilustración 19. Vista Inicio.

4.4 Vista de Búsqueda de empleado

Tiene un buscador de empleados de la CLV y muestra el resultado. Además:

- Utiliza el módulo de busqueda-empleado, el cual devuelve un objeto de tipo Empleado.
- Ejemplifica la manera de utilizar GET y SET de una variable de clase para tener más control de ella y evitar que tenga el valor Null.
- En el HTML se actualiza la información del empleado que seleccione el usuario en tiempo real con ayuda de etiquetas <label> y las doble llaves.
- De igual manera que la vista inicio aquí se ejemplifica el cambio de vista mediante un botón.

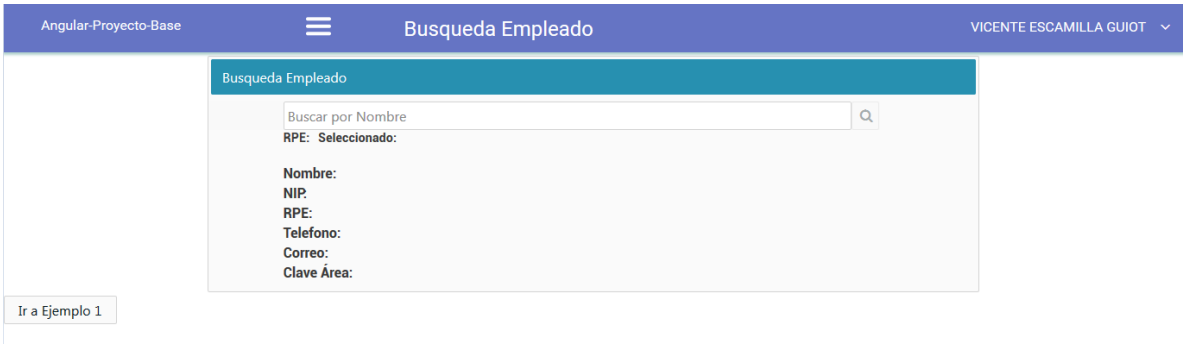

Ilustración 20. Vista de Búsqueda Empleado.

4.5 Menú deslizable

Se encuentra dentro de la carpeta Utilerias y es el contenido del componente sidebar de Primeng.

Utiliza el servicio del módulo de login para construir un menú de navegación dinámico, mediante los accesos que tenga el usuario a los diferentes módulos del sistema, sin embargo, como al menos se muestra la opción de “Inicio” entonces el menú nunca se dibujará vacío.

En la parte superior muestra la foto y algunos datos del usuario que inició la sesión.

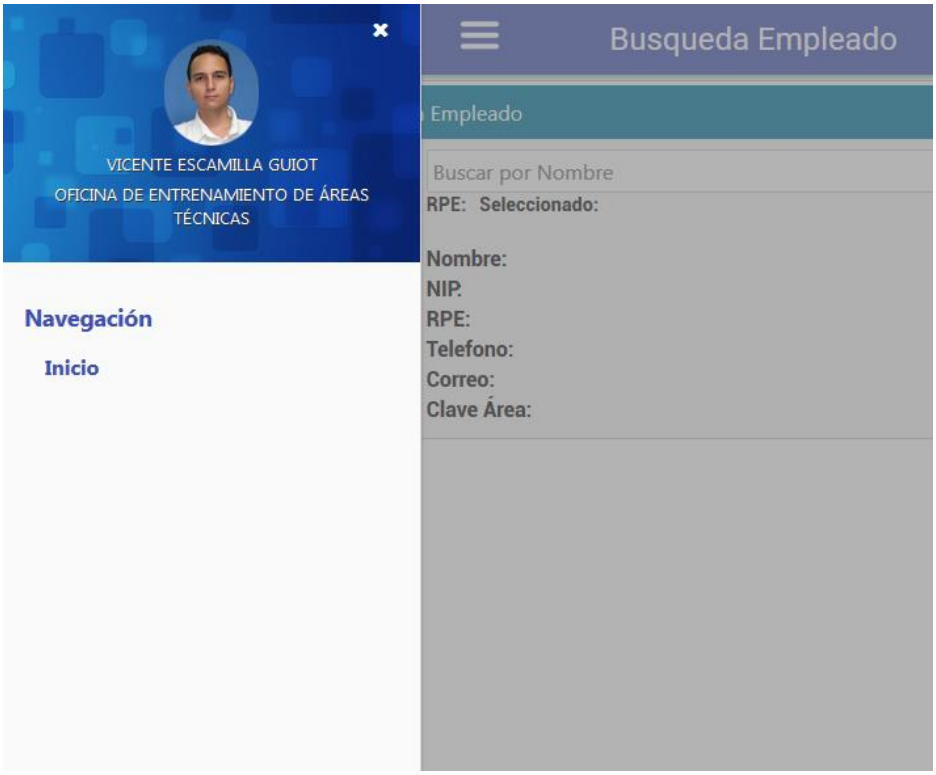

Ilustración 21. Menú deslizable.

4.6 Encabezado del proyecto

Se trata de un contenedor sencillo que estará visible siempre que el usuario tenga una sesión iniciada. Aquí se muestra el nombre del proyecto y el nombre del usuario: si se le da clic aparece un menú con la opción de cerrar sesión. Por último, en el encabezado es donde el usuario puede abrir el menú deslizable al darle clic al menú de hamburguesa que se encuentra situado junto al nombre del proyecto. Finalmente se muestra el nombre de la vista actual. El encabezado aparece en las Ilustraciones 19 y 20.