Ejemplo estructura de trabajo en un proyecto Angular

A los ejemplos básicos que hemos visto anteriormente se les ha dado una cierta organización por directorios, módulos y componentes, pero para un proyecto medio o más complejo donde cada vez se hace más y más extenso en el número de elementos necesarios, es aconsejable seguir una estructura de trabajo y una serie de pautas de buenas prácticas :

- En general se debe crear una estructura de directorios/módulos por grupo de funciones concretas que vaya a realizar cada uno, intentando de encapsular con la menor dependencia posible de otros.

- Podemos diferenciar varios tipos de módulos : 1º Módulos de características o funcionalidades (**Features Modules**), son los módulos/componentes que agrupan funcionalidades comunes, se crearán en directorios que alojarán servicios, directivas personalizadas, componentes, etc… relacionados entre sí, éstos sólo se declararán en un único módulo con el nombre correspondiente. 2º Módulo de características compartidas (**Shared feature module**), son los módulos/componentes que se van a reutilizar en la aplicación, se alojan en un directorio nombrado como **shared** . 3º Módulo principal (**Core Module**), módulo alojado en su directorio nombrado como **core** que normalmente va a proveer de los **servicios**, **modelos de datos**, **guards** (*son middlewares que se ejecutan antes de cargar una ruta y determinan si se puede cargar dicha ruta o no, se usan para cargar rutas o no en función de los permisos de los distintos perfiles que existan*), o **interceptors** (*para interceptar solicitudes y respuestas http para manejar errores*) , necesarios y con alcance global en la aplicación. Estos elementos sólo se proveen desde este módulo, con el objetivo de crear una sóla instancia de cada uno (singleton). En el caso de que la aplicación se realice basada en módulos, sólo el **AppModule** debe importar el módulo **core**.

- Se puede crear un directorio específico para módulos/componentes que se van a cargar bajo demanda (**Lazy Loaded folders**), esto mejora el rendimiento de la aplicación ya que durante el inicio sólo se cargan los módulos necesarios.

- Otras buenas prácticas …, las clases de componentes o servicios no deben de superar las 400 líneas de código, al igual que las funciones/métodos no deben de superar las 75 líneas de código.

- Para nombrar los archivos de clases debemos de cumplir el siguiente patrón **característica.tipo.ts** , ej: **listado.component.ts**

- Delegar la lógica compleja de componentes a servicios. Sólo se dejará en el componente la lógica necesaria para la vista .

- Se debe usar la lógica para realizar operaciones de datos e interactuar con los datos en un servicio.

A continuación se muestra un ejemplo de una posible estructura para un proyecto Angular usando el IDE online **https://stackblitz.com/** , con el cual podemos probar proyectos de Angular sin necesidad de instalar nada, incluso nos da la posibilidad de descargar nuestro proyecto a local.

Ejemplo estructura proyecto

Más información en : https://angular.io/guide/styleguide