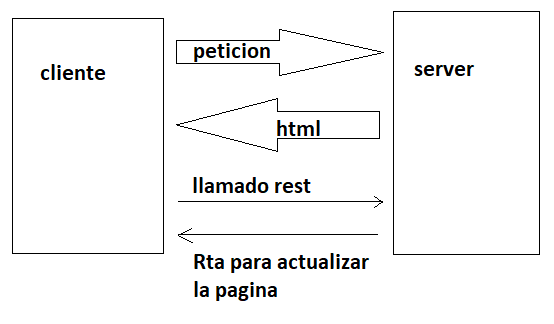
1. Todo esta basado en Basado en módulos (conjunto de componentes), componentes (html, css y js) y servicios (llamada a servicio)



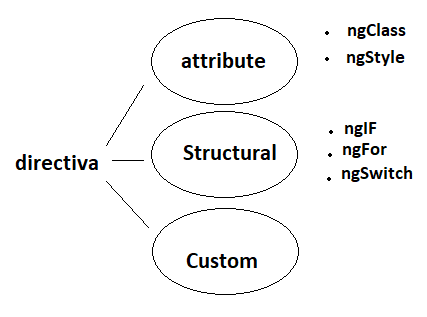
2.

Componente: un elemento que está compuesto por 3 partes,

* HTML, que es el que se va a visualizar en la interfaz de usuario
* Un archivo de lógica, alli estará todo el comportamiento
* CSS, allí estará el estilado del html

Servicio: es la forma de trabajar de manera síncrona con datos y llamadas, generalmente usado para comunicar componente y consumir servicios

Directivas: son una serie de elementos que aplicaremos a nuestro código con el fin de añadir funcionalidad



***Etiquetas:***

“ng-template”: es un plantilla para ser usado en la vista de dicho componente tantas veces sea necesaria, pero que no es visible; a menos que se defina específicamente en el html dentro de una contenedor

“ng-container”: es un contenedor para envolver un elemento, pero que no es renderizado en el html

“ng-content”: este elemento especifica dónde proyectar el contenido dentro de una plantilla de componente.

3.

Optimización de memoria frente a observable: cuando un objeto se destruye, siempre se debe eliminar la suscripción. Por ejemplo con el método OnDestroy()

Hooking: los ciclos de vidas sirven para ejecutar código en el evento de ciclo de vida específico de un componente en el momento que más nos convengan.

Lazy Loading: Es un proceso de cargar Módulos solo cuando navegas a la ruta, por lo que, la aplicación se puede cargar mucho más rápido.

Web Workers: permite que la aplicación ejecute la CPU en segundo plano, para que se concentre en el buen funcionamiento de la interfaz de usuario.

Web Components: son bloques visuales que agrupan todo el HTML, CSS y la lógica mediante JavaScript para la correcta renderización y funcionamiento del elemento. Los webs components al estar encapsulados se pueden usar en cualquier proyecto.

4.La herramienta que conozco y he utilizado es Jenkins. Esta herramienta verifica el repo git donde se deja el código y con cada commit en el código, la app lo toma y lo compila. Una vez compilado de forma correcta lo este lo despliega en el ambiente de test o producción depende sea el caso y configuración para ser usado.

Algo importante y deseable de tener es el concepto de merge request. Para que el desarrollador antes de subir sus cambios tengo una revisión y chequeo de parte de sus pares o lideres para evitar subir errores.

5. Minificar recursos (HTML, CSS y JavaScript)

Aprovechar el almacenamiento en caché de los navegadores

Codificación con baja complejidad computacional

Para la comunicación a varios componentes veo dos variantes:

-Padre a múltiples hijos con emitter (@input y @output)

-Con servicios y componentes suscriptos

Para este ultimo los mejor es el patrón de diseño “ Observer” , ya que , es un patrón de diseño que permite a un objeto notificar a otros objetos sobre cambios en su estado