Hasta el momento lo único que hemos hecho es puro maquillaje y ordenar un poco la información de nuestra

aplicación.

Pero ahora viene lo bueno vamos a empezar a trabajar con la lógica para el manejo de todas estas listas

de tareas o listas de deseos de cosas que tengo que hacer no voy a pedirles que regresemos a la aplicación

y es momento de centralizar toda la lógica o todos los por decirlo así los tubos o las listas de deseos

hay que manejarlas en algún lugar y ese lugar por lo general es un servicio.

Los servicios en angular son muy utilizados para manejar toda la lógica de mi aplicación o de mi proceso

en este caso les voy a pedir que creamos un nuevo servicio mediante el la web que es exactamente el

mismo proceso para crearlo como se hace con el angular Ciela en angular haríamos un de cenar generarme

un servicio pero no vamos a usar ya tiene su propio sí y sería ahí ONIC espacio generarme un servicio

lo quiero dentro de ser vices Slash y seguido del nombre del servicio que le quiero dar.

En este caso se va a llamar deseos deseos servis no hace falta poner la palabra porque ya sabe que como

estoy generando un servicio le va a poner la palabra servicio.

Presionamos Enter y si la carpeta no existe la va a crear por nosotros.

No se preocupen por esa actualización si quieren ustedes la hacen.

Yo la voy a dejar así por ahora cierro la terminal noten que ahora aquí tengo la carpeta de servicios

y tengo este archivo de pruebas.

Este archivo en los puntos Speck ya lo había mencionado antes pero para hacer hincapié este tema de

las pruebas unitarias y de integración Nakuru en el curso avanzado de angulas.

Por un momento yo no voy a hacer nada con este archivo así que lo puedo borrar.

También hay un comando que pueden añadir a la instrucción del avión Excel o no lo sé para no crear esas

pruebas.

Ok entonces trabajamos en esta parte de los deseos.

Aquí yo voy a manejar las listas de las tareas.

Pensemos que las listas no voy a crear de una vez.

Es una propiedad que yo voy a tener pública aquí en mi clase.

Recuerden que si yo no le pongo nada y no lo pongo públic explícitamente.

Siempre queda pública es decir si yo la quiero privada tengo que especificar que sea privada.

Si yo no pongo nada eso lo deja pública.

Esto lo voy a manejar de tipo.

De esta manera y lo inicializar con un arreglo vacío.

Pero no es muy buena práctica que nosotros trabajemos estos modelos de tipo n porque no sacamos provecho

del poder de scrip.

Entonces lo que vamos a hacer es crearnos nuestras clases en este caso serían los modelos de la información

como yo la quiero manejar así que comencemos a crear esos modelos hagan click derecho en la carpeta

p.p. nuevo folder.

También se puede hacer mediante el iPhone que sea la mejor no especificamos mucho en esa área y voy

a crearme el modelo en una carpeta llamada modders dentro de Models click derecho nuevo archivo y es

un simple archivo de Tappsi común y corriente.

Vamos a ponerle nombre.

Vista yo hoy te apunto Model .3 y presionan enter esta va a ser una simple clase común y corriente.

Así que comencemos a crearla.

Voy a poner o empezar con la palabra Export porque necesito utilizarla en otros archivos fuera de este

archivo de lista hay un punto modo el punto 13.

Por eso la palabra Export class para decir que son una clase y se va a llamar lista Haytham abren y

cierran llaves.

Esa es la definición de mi clase.

Cuál es la idea que yo quiero hacer en esta lista.

Bueno esta va a ser una tarea en particular.

Por ejemplo comprar leche como yo salir a bailar no pasear al perro recoger el Puppo del perro o lo

que sea.

Entonces qué es eso.

Eso en particular es un ítem de la lista para tener una descripción de tipo string y también va a tener

un estado de completado que es de tipo bullían.

De esta manera creemos el constructor parentesis abren y cierran llaves el constructor debería de pedir

únicamente la descripción como un tipo porque es sólo la descripción porque obviamente las tareas se

van a crear con su estado completado en falso no tendría sentido crear una tarea que ya terminé.

Entonces voy a decir que disputó descripción va a ser igual a la descripción que yo recibo como argumento

quien el constructor y también voy inicializar el disco autocompletado igual a falso.

Eso va a ser todo lo que yo necesito en mi lista.

Heute ahora vamos a crearnos otro modelo Mayfield que se va a llamar lista puntu Model punto 13 esta

lista tiene información ya sobre todo la agrupación de las tareas que yo quiero asignar.

Ya vamos a ver Ok empecemos con la palabra Export porque quiere usarla fuera de este archivo Class y

el nombre va a ser lista.

Noten que la L es mayúscula porque la definición de las clases siempre va capitalizada no es que sea

obligatorio yo perfectamente lo puedo poner como una minúscula pero las clases como son objetos que

me van a servir a mi para crear instancias de este elemento se acostumbra que la l siempre sea mayúscula

o estén capitalizadas correctamente.

De esa forma estas listas yo las voy a manejar con las siguientes propiedades voy a tener un number

esteril yo lo voy a generar.

El título es decir cuál es el título que tiene esta lista como por ejemplo recolectar piedras del infinito

y las tareas tendría que ser recolectar la piedra del alma del poder y todas las cosas.

Todas estas cinco piedras.

El título va a ser de tipo Stroheim coloquemos un creada en el tipo Deux Ex con mayúscula porque esto

es un objeto.

También esto es un objeto pero bueno esos son primitivos y este es un objeto ya propiamente pero bueno

los de siempre van con la de mayúscula.

Luego voy a poner un terminada en qué va a ser otra fecha que hace referencia cuando yo terminé la tarea.

Eso también se va a hacer y se va a manejar de forma automática.

Luego voy a poner una bandera Julianna terminada o completada que se va a cambiar va a cambiar su valor

cuando yo tenga todos los items que va a tener esta lista ya completados.

Y por último voy a tener los items que es un arreglo de lista item tap lo declaramos de esta manera

asegunes asegúrense va a importar el lista al ítem el modelo que acabamos de crear.

Ok definamos ahora el constructor recuerda que el constructor es la instrucción que se llama cuando

nosotros creamos una instancia de la lista ok aquí yo debería de recibir el título por fuerza o sea

el título tendría que recibirlo porque siempre voy a ocupar el título es decir antes de crear una lista

yo tengo que saber esa lista qué nombre va a tener por ejemplo compras de supermercado qué sé yo.

Superhéroes Amencer cosas por el estilo.

Entonces digamos que disputo título va a ser igual al título que recibo en el constructor.

Las demás propiedades yo las puedo deducir.

Por ejemplo el disputo creada en va a ser igual a mi ciudad.

Es decir cuando se crea una nueva instancia ese es el momento en el que yo lo voy a grabar enter la

propiedad terminada en la A inicializar la voy a dejar así vacía porque todavía no la necesito y sólo

se va a poner cuando se completen todas las tareas.

Voy a hablar sobre el terminada y lo voy a poner aquí como falso o sea la lista no está terminada cuando

se crean que están sin terminar.

Luego voy a decir que los disputo items para igual a un arreglo vacío.

Por qué.

Porque cuando estoy creando una lista obviamente no tengo ninguna tarea por hacer.

Simplemente estoy definiendo la lista ahora para el Heydi de la misma lo voy a definir de la siguiente

manera va a ser igual a New Day con la T mayúscula siempre.

Punto G esto me va a obtener un número un número entero el cual va a ser único va a ser imposible en

mi aplicación de que dos listas tengan exactamente el mismo instante o que se creen en el mismo instante

eso no va a suceder pero lo ideal sería que Udes obviamente se vayan a una base de datos lo graven ahí

y que la base de datos sea quien les asigne Leidi pero para nuestro ejercicio eso es más que suficiente.

Ok voy a grabar los cambios en el modelo tengo mis dos objetos aka mis dos modelos cerramos el lista

Model y cerramos el lista Heute ahora qué tipo sería.

Estas listas ok estas listas de qué tipo serían sería de tipo lista Top asegúrense que no importe de

la lista model pero en qué momento se va a utilizar este servicio prácticamente en toda mi aplicación.

Y ese es el objetivo del servicio.

Manejar una única instancia a lo largo de toda mi aplicación es decir yo no quiero crear una instancia

cuando estoy trabajando en la página de las listas terminadas o en la página de listas pendientes o

sea yo quiero manejar un único arreglo de listas.

Ahí estarán todas.

Puedo poner acá un mensaje de consola voy a poner aquí un Consol lo que diga servicio inicializar voy

a grabar los cambios en el deseo servis este constructor se va a inicializar o se va a llamar una única

vez aunque yo ponga este esta información o esta referencia al servicio en varios componentes.

Por ejemplo voy a la carpeta pellet 1 y el 1 Ph.D.

Voy a definirme mi constructor.

Aquí es necesario el constructor porque si no yo no puedo inyectar nada para inyectar el servicio.

Voy a ponerlo como públic deseos servis de tipo deseos servis Tapp asegúrese de que no importe.

Obviamente la ruta que ustedes tienen cuando yo hago esto y grabo los cambios cuando estoy aquí está

bueno mi constructor ustedes van a notar que aquí ahora va a decir que el servicio se inicia liso perfecto.

Ahora hagamos esto mismo a copiar esto y voy a ir mal Taub.

Altas 2 PACh y voy a hacerlo exactamente igual.

Lo voy a pegar pero acuérdense que como esto es otra clase tengo que volverlo a importar acá del deseo

servis voy a grabar los cambios.

Regreso a la aplicación y aquí quiero que ustedes noten algo lo que yo les mencioné de los servicios

me permite manejar una única instancia del mismo es decir si me voy a quedar terminados ven que no vuelve

a aparecer el servicio inicializar.

A pesar de que aquí está inyectado el constructor Sí regreso pendientes.

No es que se vuelve a lanzar.

Por qué.

Porque el servicio está de forma que es exactamente lo que yo necesito para que toda mi aplicación comparta

la misma información.

Lo vamos a dejar así.

Yo sé que tal vez no hicimos mucho pero esto es sentar las bases de la lógica de toda mi explicación.