Bienvenidos al curso de nestjs en este curso voy a enseñarles a utilizar uno de los frameworks más populares del ecosistema de javascript typescript y nos para crear aplicaciones web backend nexs es un framework muy versátil utilizado en muchos proyectos en producción y permite tener un flujo de desarrollo rápido para crear aplicaciones backend que te permiten hacer desde conectar una simple base de datos hasta tener aplicaciones en tiempo real con websockets Api de graphql hasta la posibilidad de crear microservicios es decir puedes utilizar njs para crear proyectos muy pequeños o proyectos muy grandes en este curso te enseñaré todas sus bases desde

Qué es nestjs

Cómo instalarlo

Cómo crear tu primer proyecto

Cómo conocer sus conceptos básicos

Cómo desplegarlo en la nube

Así que si no conoces nada del framework Pero quieres entender todas sus partes de una forma muy práctica en este curso voy a tomar ese enfoque

Antes de empezar a escribir código es muy importante saber de qué se trata el framework nestjs.

Nestjs es un framework Open source para crear aplicaciones backend escalables de Node es decir es un framework que nos permite poder crear aplicaciones encima de nodejs por lo que si ya conocen este entorno de ejecución van a poder crear aplicaciones profesionales en poco tiempo este framework está desarrollado utilizando typescript y usa muchos módulos relacionados a este ecosistema para poder crear aplicaciones web backen mucho más grandes en nestjs nosotros podemos utilizar módulos como typeORM Class validator type sql y muchos otros módulos y no tan solo utiliza type Script sino que también tiene muchos paradigmas de programación dentro del framework por ejemplo combina elementos de programación orientados objetos programación funcional y programación reactiva está desarrollado, encima del framework Express por lo que si ya conocen muchos conceptos de Express o han creado aplicaciones básicas con este framework nestjs es como un siguiente paso o digamos que es un framework que utiliza mucho de los conceptos de Express y los mejora o le añade algunos comandos extras para que podamos tener más facilidades y crear aplicaciones web más escalables

Aunque no es requerido que trabajemos con Express dentro de nes porque también nes nos permite poder incluso utilizar por debajo fastify en lugar de Express este framework trae sus propios módulos pero también permite integrar paquetes de la comunidad a través del administrador de paquetes de node llamado mpm.

La estructura de carpetas del proyecto y la forma de organizar los archivos está inspirado en un framework muy popular de javascript que se llama Angular en donde si ustedes han utilizado este framework quizás conocen algunos conceptos como módulos controladores servicios inyección de dependencias y así bueno nest toma muchas de estas ideas y los trae al mundo del backend pero no es necesario saber angular para poder utilizarlo

Así que si nunca han utilizado angular en el frontend no hay ningún problema en la práctica nestjs

No inventa muchas cosas nuevas sino que hace uso de muchos módulos populares en el ecosistema de no Mods y typescript y los junta bajo una arquitectura simple y ordenada para que nosotros podamos implementar desde cero un backend para que este también pueda escalar muy fácilmente a miles de usuarios en un mismo servidor no es necesario confundir nestjs con el framework nextjs que es un framework de frontend basado en react, nestjs de por sí se escribe distinto y es un framework mucho más enfocado en el backend es decir también pueden utilizar nextjs junto con nestjs en un proyecto mucho más grande.

Primero voy a comentarles un poco acerca de las herramientas que vamos a utilizar y también las extensiones que yo espero que ustedes tengan instalados para que puedan tener un entorno mucho más cómodo de desarrollo lo primero es saber si tienen node 10 instalado

Abren una terminal o consola Y a partir de allí simplemente escriben El comando:

**node --version**

Y si dan un enter van a ver la versión de Node en mi caso tengo la versión 21 pero con cualquier versión superior a la 18 les va a bastar para este curso ahora si no saben nada de Node les reitero que pueden ir al curso básico de Nodejs que está en la descripción para que de esa forma puedan estudiar primero node puedan saber qué es y ya luego puedan continuar con nestjs

No tiene ningún sentido empezar con nest sin saber Node porque nest es un framework de Node

Entonces por debajo siempre van a estar utilizando los paquetes de Noe por lo tanto es necesario ya tener node Y al menos saber de qué se trata luego los siguientes programas que también voy a estar utilizando son extensiones de visual Studio code

Las principales son las siguientes la primera que vamos a necesitar de tener instalado es **material icons** que es en realidad un paquete de iconos para visual Studio code

Les recomiendo que lo instalen sería este de aquí que se llama material icons icon Team que viene por parte de Philip kf y

Bueno aquí simplemente le dan instalar y ya estaría este paquete de material icons no solamente va a colocar carpetas sino que cuando creemos algunos archivos de nes algunos iconitos van a cambiar relacionados a nes Así que espero que lo instalen Y luego el siguiente paquete que también espero que instalen es un programa llamado

**Thunder cin o una extensión llamada Thunder cine** esta extensión es requerida también o también Espero que lo instalen porque les va a dar en su barra de visual Studio code un iconito nuevo que dice

**Thunder client** el cual sirve para poder hacer peticiones Http es decir para poder hacer peticiones get post put y delete y como vamos a estar trabajando con es va a ser muy cómodo có a utilizarlo desde aquí así que ya Espero que lo tengan instalado y finalmente una extensión que yo espero que instalen pero ya no tan solamente para visual Studio code sino también para su navegador se llama de esta forma **Jason viewer** esta extensión es una extensión para navegadores basados en Chrome Es decir para navegadores como Google Chrome brave opera Microsoft Edge y bueno lo que van a hacer es simplemente ir en esta extensión en su página de Google Chrome en su página de la tienda de Chrome y simplemente van a darle en Añadir a Chrome Okay le van a dar Añadir extensión y listo qué es lo que es esta extensión esta extensión cuando nosotros recibamos un json lo va a acomodar con colores De hecho voy a darles un ejemplo sencillo hay una página que se llama json placeholder que nos da datos de ejemplo de una red Api y supongamos que yo quiero obtener algunos datos pues lo que hace la extensión es colocarlo de esta forma ordenados es decir No Solamente vamos a ver un Stream allí sino que lo coloca como una especie de editor y les coloca como una especie también de colores entonces pueden personalizarlo incluso con temas si vienen aquí le dan en options pueden escoger algún tema en mi caso yo por lo general escojo un tema que se llama Midnight por ejemplo o Midnight o monokai cualquiera de los dos y si vienen y refrescan bueno guarden aquí le den guardar y refrescan pues ya tendrían un cambio de estilo y de esta forma también pueden leer fácilmente des el navegador las peticiones get que se hagan entonces también espero que tengan esta extensión para Google Chrome instalado les va a venir bastante bien para poder analizarlo y listo de esta forma pues ya podemos iniciar con el curso de nes Vamos a continuar ahora con lo que vendría a ser la instalación de nest lo primero que vamos a hacer en esta sección es instalar njs para poder hacerlo vamos a abrir una terminal y como ya tenemos nodejs instalado vamos a ejecutar El Comando

npm install gg@nextjs slc

esto lo que va a hacer es instalar el programa de nextjs o el click de nextjs en todo nuestro computador

Este es un click o Comando de terminal que nos va a permitir generar proyectos de ns ahora una vez instalado lo que nos va a dar es algunos comandos que no tan solamente van a crear un proyecto sino también nos van a permitir poder ejecutar código o también poder generar código es decir que a través de los parámetros del Comando vamos a poder ayudarnos a crear código un poco más rápido una vez ya tenemos El Comando vamos a comprobar si está para poder hacer eso escribimos n-v y nos va a dar la versión de nes que tenemos instalado en mi caso tengo la versión 9.5 como pueden ver otra forma es también escribiendo nes gu gu version y nos va a dar la versión de nes que tenemos instalado que es la misma o también podemos escribir simplemente El Comando nes sin nada y esto lo que va a hacer es listar todos los comandos que posiblemente

Utilizar en este framework lo primero es el comando **New** que es para poder generar un nuevo proyecto luego tenemos El Comando **build** para construir estar para iniciar en producción la aplicación info para poder ver los detalles de un proyecto y también tenemos la forma de Añadir otras bibliotecas utilizando El Comando ad de n luego también está un comando generate que es para poder generar código y aquí tenemos una tabla extensa que vamos a ir viendo a lo largo del curso para ver cómo generar otras

porciones en este caso de decoradores controladores clases y demás todo esto lo vamos a ir viendo paso a paso pero bueno este es el comando típico de nes con el que vamos a inicial ahora lo primero que vamos a hacer es crear un proyecto a partir de este Comando vamos

a limpiar la terminal cuando yo abro una terminal me Ubica mi usuario ustedes probablemente también estén así Simplemente vamos a navegar al escritorio con el comando CD desktop escríbanlo bien y una vez allí vamos a crear un proyecto de next Yes en mi caso

voy a decir nest New My First app aquí me dice qué paquete Qué administrador de paquetes quieres crear en este caso voy a darle impan doy un enter y se empieza a crear el proyecto como pueden ver se está creando una carpeta allí y bueno vamos a abrir esta carpeta con nuestro

editor de código en mi caso voy a estar utilizando visual Studio code ahora a partir de aquí lo que vamos a tener Es simplemente una configuración de src también vamos a tener una configuración de prer de typescript y demás todo esto

lo vamos a indagar un poco en la segunda sección o la sección que viene después de esta Pero bueno lo que vamos a venir aquí es eh recorrer un poco los archivos principales y cuando nosotros tenemos este proyecto eh Por lo general vamos a

ver que hay unos típicos archivos que empiezan con app punto algo está app controller controller module y service y Main ts de momento no vamos a modificar nada simplemente vamos a ejecutarlo para poder ejecutar un proyecto de nes

primero tenemos que asegurarnos que se termine de instalar como pueden ver este proceso de instalación toma algo de tiempo la razón es que eh Para poder entender Por qué eh toma tanto es que si yo vengo aquí en el package Jason vamos

a ver que aquí tengo una lista de paquetes bastante extenso que instalan eso entre ellos están eh plugins Para slin hay otros paquetes de typescript hay otros paquetes para prettier el framework de testing Yes hay otros

paquetes para poder hacer también super test es decir testing end to end y otros paquetes adicionales que vien de por sí dentro de la propio framework como podrían ser los arns de momento no tenemos que indagar tanto en esto simplemente tenemos que esperar un

segundo a que se termine de generar estos paquetes y luego vamos a continuar aquí ya tenemos nuestro proyecto terminado y me dice puede ejecutar C de myf App y luego mpn Run Start de hecho vamos a cerrar esto vamos a abrir la

terminal de visual Studio code y el primer Comando que vamos a ejecutar es el comando mpm Run o mpm Start nada más y damos un enter y este Comando en Start lo que va a hacer es que va a ejecutar

el proyecto en un puerto que en este caso sería el puerto 3000 lo que significa que si yo abro un navegador y escribo el local Host 3000 y doy un enter vamos a ver un Hello World Esta es la bienvenida típica que nos da n ahora Este no es el único Comando que vamos a

ejecutar de hecho este Comando Start lo ejecutarías solamente cuando estamos en producción Pero hay otro Comando más vamos a cancelarlo con control c dos veces Okay limpiamos y vamos a ejecutar El Comando npn room Def ese Comando npn

room Def lo que va a hacer es ejecutarlo en modo desarrollo pero aquí de hecho no se llama Def como tal sino se llama mpn room Start do. Def Y bueno ya con esto lo que va a hacer es ejecutarlo en modo watch Qué significa que va a estar

vigilando por los cambios que nosotros hagamos en código así que todo cambio que hagamos lo va a ver o lo va a reflejar inmediato en el navegador si yo refresco esto pues pueden ver que dice Hello World pero si voy de momento en el

app controller un segundo aquí y aquí tenemos una función que dice tienes que ir a app service vamos a ir allí y aquí hay un texto vamos a cambiarlo por un hola mundo digo Hola mundo y guardo pueden ver que la consola se reinicia y

Esto indica que ya el código está actualizado lo que significa que si refresco la página en el local Host 3000 vamos a ver un hola mundo listo Este es el comando típico que vamos a estar ejecutando cuando estemos desarrollando El Comando mpen room Start 2. Def El

Comando mpen Start pues lo vamos a ejecutar en producción ahora no es lo único que vamos a ejecutar porque esto es typescript eso significa que necesitamos compilar el código por lo que también vamos a necesitar de otro

Comando que es el comando mpn Run build para poder hacerlo vamos a cancelar aquí la consola y voy a decir mpn r build y esto lo que va a hacer es tratar de convertir todo el código que nosotros ya hemos desarrollado y lo va a convertir

dentro una carpeta dis Okay aquí está todo el código convertido que este es el que ejecutarías en producción de nuevo eh Para ejecutar en producción simplemente hacemos en p Start y esto ya ejecutaría el código en producción de

hecho ya incluye el proceso allí del de la ejecución de node Entonces esto es lo mismo pero bueno aquí no vemos mucha diferencia Ya lo vamos ir viendo cuando despleguemos nuestro proyecto en una nube vamos a cancelar con control c y ahora lo que vamos a ver también son

otros dos comandos que también Vienen con nes el primero es el npn room lint que es para poder eh ejecutar un linter o lo que es el programa slint y lo que hace esto es vigilar en todo el código y

comprobar si por ahí tenemos algunas reglas que se están rompiendo por ejemplo quizás estamos dejando un conso por allí una variable sin usar un import sin usar y así pero bueno de momento no tengo nada porque son proyecto en blanco

entonces aquí me dice no hay nada o no hemos encontrado nada que rompa en este caso con las reglas de slink lo siguiente algo a tomar en cuenta que aquí también tenemos unos cuantos imports de hecho si yo vengo en App

Service en module nosotros aquí estamos utilizando estas comillas simples Así que luego cuando emos código debemos tener en cuenta que si nosotros importamos algo Es probable que vamos a tener que configurar esto pero no se

preocupen cuando ya lancemos estos desarrollando o lancemos algunas características más pues allí vamos a ir reparando vamos a ir acomodando este framework para que se adap aca nuestras configuraciones de visual Studio code

listo entonces con esto pues ya tenemos nuestro H mundo en la siguiente sección vamos a analizar un poco la estructura de proyectos que nos dan nes Yes en esta sección vamos a ahondar un poco en la estructura de proyecto que

nos dan n yeses para poder entender mucho mejor cómo es que vamos a estar escribiendo nuevo código Lo importante es entender en es que cuando nosotros creamos un proyecto se genera esta estructura típica la primera es la carpeta dis esta carpeta 10 es para que

nosotros podamos colocar todo el código convertido de typescript a jav es decir aquí va a ir todo el código de javascript que al final se va a ejecutar en un proyecto de verdad O mejor dicho en un servidor lo siguiente que tenemos es la carpeta node modules esta en

realidad no es para nosotros es para funcionamiento interno de noes si no entienden esta carpeta hay un curso básico de noes donde se explica mucho más a detalle Qué significa la carpeta node modules Y cómo funcionan los módulos de node lo siguiente que tenemos

es una carpeta src aquí es donde realmente vamos a colocar todo nuestro código esta carpeta es para nosotros poder Añadir más lógica como podrían ser rutas urls o funciones que se van a ejecutar cuando una URL sea visitada es

decir aquí realmente va a estar el código de backend luego tenemos la carpeta test que la carpeta test Es simplemente para que nosotros podamos escribir test de end to end es decir test que van a testear toda la

aplicación para este curso no vamos a andar tanto en tanto detalle para poder ver cómo funciona esto Pero al menos le voy a dejar listo un tipo de recurso ahí en la descripción para que luego ustedes puedan Investigar un poco más luego

finalmente por fuera tenemos los archivos de configuración el primero es slink que es para poder tener reglas de linting es decir reglas en donde va a poder subrayar con un código o con un tipo de color algún tipo de error que

nosotros tengamos o algún tipo de regla que se esté rompiendo una variable que no estemos utilizando un import que se ha llamado y nadie está utilizando o quizás por ahí le deamos un punto y coma o un espacio de más o un mensaje de consola de más luego tenemos un archivo

punto git inore que es para poder ignorar archivos Esto es para poder evitar que se suban archivos esenciales como pueden ser las variables de entorno que por lo general traen primary kys o Secret kys que son eh útiles para

conectarnos a base de datos Y eso que son datos sensibles entonces con el git ignor ignoramos ese tipo de archivos luego tenemos el priti rc el priti rc es para que nosotros podamos configurar la herramienta prettier que al final nos ayuda a ordenar el código o formatear el

código luego también tenemos nes click json que es para configuración interna de nes eh No muy comúnmente se utiliza ese archivo de configuración más que cuando queramos alterar algo interno del framework el package lock es un archivo

de configuración que está relacionado con el package json Y esto es para poder administrar dependencias y están conectados también con la carpeta node modules luego el ritmic que es un archivo simplemente para describir el

proyecto un tsconfig build.js que es para poder colocar archivos de configuración de typescript en este caso hay dos De hecho hay un tsconfig json y un tsconfig build.js la razón de que

haya dos es que el primero el ts config json es para que nosotros podamos configurar la base de nuestro typescript es decir las reglas base y luego hay un ts confit build que lo que hace es Añadir unas reglas adicionales en

producción cuando hace el build de la aplicación la construcción en producción ahora muy Aparte de todo esto algo que también vamos a querer personalizar luego van a ser los iconos de nuestro editor en este caso como ya les mostré

en la sección de requerimientos o instalaciones pues lo que voy a hacer ahora es personalizar un poco porque pueden ver que sale con el icono de angular ahora por qué se con el icono de angular bueno en material icons el el paquete de de iconos que hemos instalado

en visual Studio code aquí por lo general se utilizan estos iconos porque como termin en. module service esos son nombres típicos del framework de angular de frontend Pero como estamos trabajando en N lo que podemos hacer es esto vamos

a presionar f1 en nuestro editor y vamos a escribir workspace json y vamos a esta opción que dice Open Pro workspace settings Jason si yo ahora Allí se crea una carpeta bscope dentro de nuestro

proyecto y aquí lo que vamos a hacer es colocar una propiedad que nos va a ayudar a poder indicarle a visual Studio code que queremos utilizar iconos de de material pero para nestjs en este caso vamos a colocarle material icon Team

active icon pack y Aquí vamos a escoger ns si yo guardo esto lo que va a pasar en src es que estos iconos van a cambiar a iconos de ns porque al final estoy trabajando con este framework de backend okay Y ya está entonces cada vez que

abramos el proyecto en visual Studio code pues va a tomar esos iconos ahora muy aparte de todo esto Qué significan estas porciones de código que tengo aquí de App controller ts module.ts app service.ts y demás bueno Estos son

archivos que de una u otra forma indican que el paquete de nes está requiriendo de algo que se conoce como un módulo eh para entenderlo mejor voy a colocar un gráfico cuando nosotros tenemos una

aplicación de njs Por lo general podríamos pensarlo como un recuadro en grande Okay en una aplicación de backend nosotros no solamente vamos a tener rutas básicas sino que vamos a tener muchas funcionalidades por ejemplo puedo

tener tener usuarios productos categorías supongamos que estoy creando un backend de e-commerce y aquí por ejemplo yo puedo crear un módulo que solamente tenga toda la lógica de la autenticación vamos a llamarle out de

hecho puedo llamarle out module Okay vamos a extender un poco ahora este módulo solo se va a encargar de la autenticación pero luego puedo tener otros módulos adicionales entonces en una aplicación puedo tener varios puedo tener un módulo tan solamente de

productos puedo llamarle product modu también puedo tener un módulo de usuarios puedo llamarle user o users module y luego puedo tener un módulo por ejemplo para tomar tareas vamos a llamarle task modu entonces un módulo al

final es t solamente una parte de toda nuestra aplicación y esta parte puede estar conformada a su vez por otras subpartes entre esas subpartes que conforman un módulo puede estar por ejemplo un controller que tan solamente

son funciones que se van a ejecutar cuando una URL es visitada luego también podríamos tener algo llamado un service que son funciones que se pueden reutilizar y que típicamente se utilizan para poder colocar consultas de base de

datos es decir las consultas de base de datos típicamente se reusan Entonces el service lo que hace es poder utilizarse en para poder hacer consultas a un a una base de datos ya sea sql o no sql y aquí

adentro también puedo tener otras partes mucho más importantes como puede ser por ejemplo algo llamado dto que es para poder eh colocar la descripción de cómo vamos a recibir los datos en el backen esto es una esto es un abreviado de

hecho de Data transfer object luego Aquí también puedo tener funciones que se van a ejecutar de forma intermedia algo que se llama como interceptors y demás en realidad hay muchas otras lógicas que pueden ir dentro de un módulo pero lo que voy es que en un módulo nosotros

podemos tener varios archivos Entonces cuando hemos creado nuestro proyecto de nes ya tenemos un módulo allí De hecho si lo notan aquí en mi proyecto Yo tengo un app module y arriba tengo un app controller y un app controller Spec y un

app service todo esto al final está relacionado con este app modle okay Entonces digamos que todo este pack o de archivos que tienen casi el mismo nombre están relacionados o representan lo mismo ahora el que los une es este app

module que es el inicio digamos de esos archivos dentro del app module ustedes van a encontrar el app controller el app service y ya el app controller de por sí llama también o está muy relacionado con su app controller es Spec que es para

poder hacer testing en este curso no vamos a andar en el testing Pero al menos Les comento para que puedan entenderlo ahora este módulo en realidad está dentro de una aplicación recuerdan que nosotros teníamos un recuadro grande que contenía otros módulos Bueno aquí en

main.ts es donde se llama entonces este es el archivo inicial que arranca todo el backend y como pueden ver siempre va a llamar un módulo que es el app module Okay esta es la estructura de proyecto de nes yj ahora en realidad necesitamos

todo esto realmente no Entonces vamos a limpiar solamente nos vamos a quedar con el main.ts y aloud module en cuanto el app service el app controller y el app controller Spec vamos a eliminarlos Okay solo me quedo con el app module y aquí

ya no necesito importar estos archivos porque ya no existen y tampoco necesito importar o Añadir estos dos archivos entonces Este es un proyecto básico de njs y de hecho si yo ejecuto esto vamos a abrir la terminal de visual Studio

code y ejecuto npn room St Def y doy un enter y venimos ahora en el navegador vamos a venir aquí y refresco pues ya no hay un hola mundo simplemente me devuelve un texto que dice status 404

Canon get sl y not Found esto es justamente porque al momento de hacer nuestra petición pues básicamente me está diciendo que no hay ningún tipo de eh página dentro de nich Yes es por eso que me está dando este error ahora no es

un error como tal es básicamente una respuesta del backend que me indica que no existe ninguna URL o no hay ninguna función que retornar en la sección anterior hemos hablado un poco acerca de los módulos en

njs que son una pequeña parte dentro de un proyecto ahora Aquí vamos a empezar a crear Algunos porque de momento solo tengo el ad module que es digamos el módulo principal que puede contener a su vez otros módulos supongamos que en este

proyecto de backend Yo quiero crear una Api un backend que acepte tareas es decir Yo puedo crear tareas listar tareas eh guardar tareas editarlas y demás para poder hacer eso como todos están relacionados con el la entidad de

tareas pues vamos a llamar ese módulo módulo de tareas o task module para poder ordenarlo dentro de src vamos a crear por ejemplo una carpeta que se llame task y aquí nosotros podríamos colocar toda nuestra lógica de backend

todas las URL las funciones las métodos de base de datos todo relacionado con tareas tan solamente Entonces ese es un módulo ahora para poder crearlo también vamos a crear un archivo llamado pun module.ts aquí se llamaría task en

singular task o en plural también pueden llamarlo task module.ts y este task modu lo que va a contener aquí es una clase las clases podemos llamarlo de esta forma task module Okay en plural también

para que coincida con el archivo esto se exporta y para indicar que esto es una clase que viene de nestjs vamos a importar desde el paquete @ nescon vamos a importar una función que se llama

module en realidad no es una función como tal sio es un decorador esto quiere decir que yo voy a ejecutar arriba de la clase este decorador y con esto tendría este eh objeto para configurar ahora si

formateo un poco si presionan f1 y escriben formma document o también atajo de teclado dependiendo del sistema operativo en el que estén pueden ver que allí tengo algunos errores se ordena pero tengo algunos errores la razón no

es que í hemos hecho algo mal sino que si nosotros colocamos el cursor encima de estos errores me dicen que se espera un argumento o bueno aquí espera un argumento vamos a colocar un objeto vacío pero aquí por ejemplo vamos a

colocar el cursor encima de estos sub guilles rojos y me dice delete C r y es un tema de slink ahora como les comenté en nest ya viene con algunas configuraciones como de slink y prettier aquí en slink nosotros vamos a quitar

este configuración de rules y vamos a colocar otra cosa por ejemplo aquí nosotros ya viene con la configuración básica pero nosotros podemos extenderlo en este caso la que nos hace falta es poder decirles link que vamos a estar

desarrollando desde Windows la razón es que por lo general Windows acaban con otro tipo de caracter al final entonces para poder decirle que ignore eso vamos a colocar aquí un una propiedad que se llama prettier de hecho se escribe de

esta forma prettier sl prer dos puntos un arreglo y Aquí vamos a colocar el error y voy a colocarle que el n Line vamos a colocarlo en auto si yo guardo esto y vengo ahora en el editor esto

debería irse Pero algunas veces no se va inmediatamente entonces lo que pueden hacer es presionar f1 en visual Studio code y escriben sl link server y van a llegar en esta opción que dice slink restart slink server si dan un clic va a

reiniciar slink y ya no les va a dar el sobrado rojo porque esta regla se ha añadido Okay simplemente escríbanlo bien como Ven aquí para que no les dé el error ahora esta esta este módulo que tenemos aquí es solamente para poder

indicar qué es lo que yo voy a estar guardando pero este proceso de crear módulos es tan repetitivo porque en un proyecto no tan solo vamos a tener una sola ruta una sola entidad sino que podemos tener también rutas dedicadas a

productos categorías compras ventas órdenes de compra y demás entonces para no estar recreando esto desde cero nosotros vamos a tener comandos Pero antes de aprender esos comandos debemos tener claro que aquí Nosotros hemos

creado un módulo Pero en ningún momento le hemos dicho a otra parte de la aplicación que utilice este módulo entonces en el ad module aquí en la sección de imports vamos a conectarlo este ad module con este task Okay entonces tengo que importar este archivo

desde ad module entonces vengo aquí voy a importar desde ara Perdón desde task o de pun sl task voy a importar task module Y desde allí voy a traer la clase que recuerden se está exportando que se llama Tas module Y esa clase lo voy a

Añadir aquí en import ya con esto básicamente al importar este archivo y añadirlo en ad module significa que ya están conectados entonces si yo ejecuto el proyecto aquí no tengo errores y bueno de momento no tengo nada que Mostrar aquí en la terminal Así que esto

va a seguir funcionando okay Bueno lo siguiente es que como les menciono no hay necesidad de crear esto manualmente podemos crearlo con un comando Entonces vamos a ir en la terminal vamos a minimizar un poco aquí puede cancelar con control c y una forma rápida de

generarlo es utilizando El Comando de nestjs llamado nest generate pero de hecho si ustedes escriben nest generate gu gu help y dan un enter van a ver comandos de ayuda que pueden obtener

para saber qué pueden ejecutar cuando ejecuten El Comando nest generate é les dice que para empezar pueden tener una abreviación puede ser nest generate o la letra g nest g y seguido pueden

colocarle estas eh palabras que indican que ustedes van a generar código ya sea de una nueva aplicación de una clase de una configuración de un controlador y demás sé que aquí hay muchos términos nuevos pero el que nos importa de

momento es el módulo Entonces él me dice que puedo generar un módulo escribiendo la palabra modle o la palabra mo puede generar un módulo una declaración de módulo vamos a ver qué es eso voy a venir aquí en terminal voy a minimizar

un poco Voy a limpiar con control l Okay o como limpi en su terminal y aquí voy a decir nest generate module y vamos a llamar por ejemplo un mod para crear proyectos vamos a llamarle projects doy

un enter y esto lo que va a hacer es que así como tengo Tas se va a crear una carpeta llamada project en project vamos a ver un project modle igual lo mismo que si nosotros lo hubiéramos escrito Okay y dentro de App module también ya

se añadió project module es decir ya lo importó y ya lo añadió por mí bastante Genial porque aquí yo no tengo que escribirlo manualmente ahora no es la única forma de de escribir este mismo Comando porque hay abreviaciones para ahorrarnos a escribir también algunas

palabras largas Entonces yo puedo escribir nest generate o puedo escribir simplemente nest de esta forma nest generate en lugar de escribir modul puedo escribir mo que es un abreviado de modle Y luego el nombre por ejemplo también quiero crear un eh un módulo de

autenticación vamos a llamarle a vamos a dar un enter y se va a crear una carpeta llamada a o también podemos incluso más brevemente podemos colocarle nest gmo y el nombre del módulo en este caso

supongamos que quiero crear un módulo de usuarios doy un enter y se va a crear también una carpeta de users listo entonces de esta forma pues ya tengo distintas carpetas que al final me van a permitir poder enlazar distintos tipos

de funcionalidades dentro de App module de hecho en App modu pueden ver que están listados todos los módulos que utiliza esta aplicación Y de nuevo si yo entro en cada uno de ellos es como si yo hubiera escrito manualmente pero me está ahorrando varios segundos de desarrollo

Porque pueden ver que aquí ya escribió la típica plantilla para generar módulos Entonces el primer task mod lo hemos creado manualmente el resto lo hemos generado con comandos y pueden ver que es bastante cómodo hacerlo de esa forma de momento vamos a cerrar todo esto con

un clic derecho Close y continuamos en la siguiente sección aprendiendo acerca de los controladores en nest cuando hablamos de controladores no solamente estamos hablando de funciones que se van a llamar cuando una ruta sea

visitada por ejemplo aquí en nuestro módulo de tareas nosotros tenemos un task modle pero vamos a querer Añadir lógica algo que se tiene que ejecutar Cuando alguien visite determinada URL dentro de nuestro proyecto entonces para

poder hacer eso vamos a venir aquí en task y vamos a crear un archivo que se llame por ejemplo task.on controller ts y aquí nosotros vamos a crear una clase también entonces vamos a crear aquí una

clase que se llame task controller esto lo vamos a exportar Y también voy a añadirle un decorador que viene desde ns common @ njs common y este decorador se va a llamar controller este controller

vamos a colocar aquí arriba controller de hecho en este caso sería con el @ controller en mayúscula y aquí también vamos a colocarle un objeto vacío esto de aquí es similar a como habíamos hecho el modle pero este controller le

pertenece al módulo al módulo de Tas entonces vengo aquí y le digo en este caso este modul va a tener un controller o controllers y aquí en este arreglo vamos a importar este archivo task controller dentro de task module

Entonces le digo importa desde aquí task controller vamos vamos a importar la clase task controller y vamos a añadirlo dentro de este módulo task controller Okay entonces un módulo puede contener varios controladores Y en este caso está

conteniendo el task controller ahora dentro de ese task controller qué es lo que vamos a crear Bueno aquí nosotros podemos definir rutas de URL del backend que pueden ser visitadas por ejemplo supongamos que yo quiero crear una ruta

llamada tareas pues puedo aquí decirle que voy a crear una función llamada get all task por ejemplo y aquí lo que vamos a hacer es que vamos a retornar un texto que diga por ejemplo obteniendo todas

las tareas de una aplicación ahora para que esto sea visitado tengo que especificar a través de qué método http va a ser visitado por lo general son los métodos get el método get es un tipo de método que indica que se va a obtener

datos Y esto viene desde ns common como pueden ver si yo doy un click él lo importa por mí aquí arriba ahora en estos paréntesis que es un decorador yo tengo que indicar Cómo se va a llamar la URL Y en este caso voy a colocar el has

Tas esta forma de crear una URL es bastante típica en la hs y indica que aquí se va a pedir a través del método get un a la url Hash Tas se va a ejecutar esta función cuando se visite

esta URL y se va a devolver este valor si yo guardo esto y ejecuto todo esto nuevamente con mpen room Start Def y ahora visito la URL sl task es decir si entro en el local Host 3000 sl task y

doy un enter pueden ver que dice obteniendo todas las tareas ahora de esta forma es como nosotros vamos a ir creando lo que serían nuestros controladores Porque si yo quiero crear más rutas pues simplemente me empiezo a Añadir más URL es decir Yo puedo copiar

esto de aquí y puedo pegarlo aquí debajo y ya tendría que cambiar dos cosas primero el el nombre de la función vamos a decirle que voy a crear una ruta al Index vamos a llamarle Index y cuando visiten el Slash el Slash básico Aquí

vamos a devolverle algo que diga por ejemplo homepage o página inicial guardo y ya con esto si vengo en nuestra aplicación en el local Host 3000 tan solamente pueden ver que dice página inicial en realidad nosotros no vamos a

estar creando páginas realmente pero es para darnos una idea de que estas URL son al final el la forma típica de poder ir a una determinada función en sgs De hecho voy a quitar esto de aquí okay Y

nos vamos a quedar simplemente con este único controlador esto es básicamente una función que nos va a permitir visitar algo o ejecutar algo y ahora aquí estoy retornando algo pero no es necesario que retorne algo directamente

antes yo puedo buscar quizás en una base de datos en una base de datos o puedo hacer una petición a otro backend o otra Api y así entonces antes de devolver algo primero puedo ejecutar lógica

Entonces esto es lo que más ustedes van a estar modificando cuando estén creando Api en sgs así como nosotros podemos generar los módulos de forma automática recuerdan módulos aquí también puedo generar controladores de forma

automática es decir este proceso en el que yo he creado un task controller luego añadido en module el task controller y luego he generado una función o una clase allí dentro de de un archivo de typescript todo eso se puede

hacer de forma automática claro esta parte lo voy a tener que modificar pero por el resto se puede generar de forma automática vamos a verlo abro la terminal Aquí voy a limpiar la terminal y aquí voy a ejecutar por ejemplo El Comando nest generate y ya no voy a

generar el comando módulo sino ahora voy a generar El Comando controller y luego el nombre del controlador por ejemplo en mi caso Quiero crear un controlador llamado Hello por ejemplo doy un enter y esto aquí va a crear otra carpeta que se

llama Hello y este Hello al final va a ser un controller como pueden ver un Hello controller no se confundan este no es un módulo Este es un controlador Incluso se creado un un Spec es decir un archivo de testing pero solamente tiene

un Hello controller ahora algo que ha hecho es que lo ha anidado o lo ha añadido dentro de App module pero no lo ha añadido dentro de imports porque no es un módulo sino lo he añadido dentro de controllers Qué significa que estos

son funciones que se van a ejecutar cuando en la ruta principal de la aplicación sea llamada en otras palabras si vengo en Hello y abro Hello controller aquí yo puedo crear otras funciones por ejemplo puedo decirle @get Por ejemplo y aquí le digo cuando llamen

a la función Index o la función principal es decir la página principal a través del método get vamos a colocarle aquí un Slash por ejemplo cuando se visiten la aplicación principal vamos a retornar un texto que diga homepage y ya

está si yo guardo Y ejecuto nuevamente esto con mpn room st. Def y vengo aquí en el proyecto y escribo local Host 3000 vamos a refrescar y bueno en este caso no se está ejecutando vamos a ver Por qué bueno la razón es que aquí yo tengo

el Slash el homepage esto debería retornar un homepage pero esta ruta es probablemente que no se está visitando y la razón es que aquí arriba le han colocado por defecto el valor Hello es decir Para yo poder visitar esta página

tengo que colocar antes el valor Hello entonces tendría que escribir al como esto Slash Hello enter y allí recién lo veo pero en este caso no quiero eso simplemente voy a quitárselo voy a dejarlo vacío al dejarlo vacío pues ya ahora sí vengo en la página principal en

local Host 3000 nada más y voy a verr el homepage listo Entonces ese es un controlador es una función que se va a ejecutar cuando nosotros llamemos una URL así de simple hay veces en las que nosotros no vamos a querer generar un

controlador para al app module sino para cada uno de nuestros módulos que tenemos aquí dentro cómo lo hacemos Bueno vamos a hacerlo aquí desde la terminal le digo nest nest generate y vamos a generar por

ejemplo un módulo que sea para Perdón un controlador que sea para project ahora no lo voy a generar manualmente sino que voy a decirle dentro de la carpeta project Ahí es donde vas a generarlo Entonces él va a buscar la carpeta project y va a crear un archivo project

controller vamos a verlo enter y ahí lo creo entonces de esta forma es como él está generando ahora el controlador para projects esto quiere decir que ha generado un archivo llamado

project controller con la clase típica y aquí si yo vengo en project module ya lo añadió project module y nada más ya no necesita añadirlo en otro lugar porque como project module a su vez está siendo importado en ad modle no necesita añadirlo Aquí en controllers vamos de

vuelta supongamos que también quiero generar Lo mismo para users le digo nest generate o g co que es un abrigado de controller y luego le digo en users doy un enter y vamos a ver que aquí en users también se va a generar lo mismo un user

controller y y esto al final se ha añadido en out modu es decir user controller y esto se está añadiendo en user module es bastante práctico también supongamos que quiero generarlo para out pero algo que quizás se han dado cuenta

es que cuando yo he generado controladores para users y para projects se ha generado dos archivos estos archivos de Aquí estos archivos user controller Spec Y esto es porque cada archivo de controlador viene con su archivo de testing Pero quizás yo no

quiero eso de momento Entonces lo que podemos hacer es generar también un controlador para out sin ese archivo de testing para poder hacerlo simplemente le digo nest generate controller out por ejemplo para que genere un out.to

controller y le añado el gu gu no gu Spec eso lo que va a hacer es que va a generar el controller pero sin el archivo de testing vamos a verlo ahí está Solamente se genera el out controller y ya lo añadió en el out module listo de esta forma pues ya he

generado controladores en todos lados y no he tenido que hacerlo manualmente entonces este sería el paso típico para poder crear controladores en nest ys hasta este momento hemos generado

módulos y también hemos generado controladores en nuestra aplicación de hecho en tasc controller entonces hemos generado una función y esta retorna algo ahora como yo les comenté la lógica se puede escribir dentro de esta función

pero por lo general cuando se utiliza lógica reutilizable es decir consultas que quizás vas a querer llamar desde otros lados de la aplicación probablemente vas a generar una función que también se pueda reutilizar y cuando

hablamos de código en donde vamos a poder reusar lógica Ahí estamos hablando de algo llamado servicio entonces para poder crear un servicio vamos a venir dentro de task por ejemplo y voy a crear así como cre un controller y un module

un task pun Service ts esa también es una clase y como ya quizás pueden adivinar es un export Class task service es una clase normal de typescript estos

también utiliza un decorador pero en este caso el decorador no se va a llamar service sino se va a llamar desde nest common se va a llamar injectable okay Y vamos a colocarlo aquí arriba como un decorador este injectable indica que

nosotros vamos a poder inyectar código en otras partes de la aplicación Así que la lógica que escribamos aquí adentro vamos a poder reutilizarla o las funciones que creamos aquí dentro vamos a poder utilizarla ahora si se pregunta qué tipo de lógica podría reutilizar

bueno podrían ser funciones que consultan una base de datos en este caso con este controller sería esta lógica de aquí que retorna algo iría desde el Tas service es decir esto de aquí lo voy a cortar Y aquí adentro voy a colocar una

función que se llame por ejemplo get task por ejemplo y este getas retornaría este String que antes lo colocamos en otro lado pues lo colocamos aquí ahora No necesariamente tiene que ser un

String nosotros podemos retornar algo por ejemplo yo puedo decir aquí voy a retornar un arreglo y va a decir Tas un Tas dos y Tas TR un arreglo de String entonces quién van a utilizar Esto bueno Esto lo va a utilizar otras partes de la aplicación aún no sé quién lo va a

llamar pero justamente en Tas controller aquí le digo ya no vas a estar ejecutando la lógica aquí sino que vas a importar este Tas service dentro de task controller y vas a llamarlo como pueden ver todo pertenece dentro la carpeta

task entonces estoy hablando de una sola parte de la aplicación Cómo yo puedo llamar este Tas service de task controller muy fácil primero voy a tener que inyectar esta clase en el task controller voy a venir en el task

controller y voy a decir constructor voy a generar un constructor de una clase para aquel que no sepa un constructor en una clase simplemente una función que se ejecuta apenas la clase es instanciada y lo que vamos a hacer allí es colocarle

como eh Como parámetro un task service una variable llamada task service y este va a ser de tipo task service es decir va a ser de tipo la clase task service Okay y pueden ver que esto lo ha importado des el archivo que acabamos de

crear ahora esto de aquí esta esta propiedad lo tengo que guardar en unn lado o esta este parámetro lo tengo que guardar entonces lo que voy a hacer es crear una variable que se llame task service también y vamos a decirle que también es de tipo task service y aquí

le digo esta propiedad de la clase este prop de la clase dis Tas service va a ser igual al Tas service que estoy recibiendo aquí una sintaxis un poco verbosa pero básicamente lo que hace esto es que ahora yo puedo utilizar los

métodos que tiene este Tas service Dentro de este task controller lo que significa que yo tengo un controlador que no está retornando nada pero le puedes decir vas a llamar dis Tas service y vas a llamar el método get task esto como ustedes saben esta

función está retornando un arreglo de strings pero aquí yo quiero retornarlo a su vez cuando se llame este método Slash task En otras palabras para entenderlo más fácil Cuando el task controller se

llame con el Slash task Slash task va a ejecutar esta función esta función va a entrar en el task service y va a llamar el get task va a llamar esta función esta función va a retornar un arreglo y este arreglo que está retornando esta

función lo voy a retornar a su vez al cliente ya que el cliente o la aplicación cliente el navegador es el que está llamando esto vamos a guardarlo y vamos a probarlo vengo aquí vamos a abrir una terminal vamos a ejecutar todo

esto con el comando mpn room St Def vamos a venir aquí digo localhost 3000 sl task doy un enter y vamos a esperar que cargue y Bueno aquí debió cargar Slash task Bueno í tengo un error vamos

a ver cuál es el error aquí tengo un error que es un error muy típico en nes y vale la pena verlo el error que me dice aquí es que nes no puede resolver dependencias del task controller asegúrate de añadirlo en el task service

Qué es lo que está pasando aquí que yo he creado un task service lo he importado en task controller está bien pero el task module no sabe de ese task service nuevo o es decir solo conoce el Tas controller entonces aquí también

tengo que decirle import o bueno en este caso cuando nos nosotros estamos hablando digamos del módulo que estamos importando tengo que colocarlo ya no en un imports porque imports es para modules controllers es para

controladores De hecho aquí de abajo está entonces ahora tengo algo llamado providers y providers también es aotro arreglo y aquí es donde va el Tas service listo formateo y así quedaría Entonces ahora sí si vengo en la aplicación en local Host 3000 Tas pueden

ver que í está mi Tas 1 Tas 2 y Tas 3 entonces de esta forma pues ya puedo continuar y si más tarde Quiero crear otra lógica reutilizable supongamos que quiero crear otra función que se llame por ejemplo eh No lo sé testing y aquí

retorno algo retorno un texto que diga test y guardo simplemente veo en el Tas controller y le digo yo no quiero utilizar get Tas quiero utilizar eh test por ejemplo testing guardo y refresco y

ahora me dice test Okay entonces de esta forma puedo utilizar funciones que vienen desde otro lado de hecho Aquí voy a decir get Tas otra vez para importar Y este test lo voy a eliminar Ahora sí voy a guardar esto voy a guardar Aquí también y si refresco nuevamente Me está

retornando el arreglo entonces de esta forma yo puedo generar servicios y como ya pueden haber adivinado así como ya he generado manualmente este servicio también se puede de generar con Comando entonces voy a cerrar esto con click derecho Close all voy a abrir la

terminal vamos a cancelar con control c así como generamos un nest generate un module y un nest generate controller también puedo hacerlo para el service se genera de esta forma nest generate

service y por ejemplo quiero generar un servicio para users entonces le digo n generate users o gener service users Okay de esta forma generate enter y se va a generar dentro de users un user

service.ts y también un user service Spec es decir también su archivo de testing para este Service otra forma de generarlo es por ejemplo abreviando nest GS para service y luego por ejemplo yo

le digo dentro de projects vas a generar también un servicio para projects es decir se va a crear un archivo project service y project service Spec De hecho si Yo abro el service pueden ver que lo único que hace es crear un inecto o el

vacío el service Spec no es necesario que lo cree de hecho vamos a volver a crearlo también para otro para el out pero sin el el testing entonces voy a decir nest generate service y vamos a

colocarle aquí out gu gu no gu Spec esto para que no cree la archivo de testing Okay aquí en out entonces voy a dar un enter y esto lo que va a hacer es crear también el archivo de out service Y si abro ahí está el injectable que nosotros

creamos manualmente el out module ya lo añadió dentro de Provider Así que no tenemos que añadirlo manualmente tampoco y ya estaría entonces también el service se puede crear rápidamente sin necesidad que estemos haciéndolo manualmente ahora

otro ejemplo simplemente para que lo tengan más rápido en user Por ejemplo yo tengo el user service entonces aquí yo también podría crear otra función que se llame por ejemplo get users y este get

user va a retornar a un arreglo entonces el arreglo De dónde viene Bueno voy a crear por aquí una propiedad que sea private es decir solo se va a accer desde la clase va a ser users y aquí voy a colocarle un ID que sea luego un

nombre y luego también un font por ejemplo okay Que sea un número cualquiera esto voy a crear otro más y este arreglo Yo quiero retornarlo cada vez que llamen esta función entonces aquí le digo return this users y ya

estaría entonces de esta forma puedo retornar ese arreglo Quién va a utilizar esto el user controller entonces en el user controller les voy a decir cuando visiten el ar @ get de hecho de nes @get

de nescon de esta forma con g mayúscula Aquí le voy a quitar este texto ya luego le voy a explicar todo es texto de Qué significa y aquí voy a decirle get users y lo que va a hacer aquí es que va a llamar desde constructor de nuevo vamos

a hacer desde constructor vamos a llamar el private el user service bueno de aquí private no se recuerdan que habíamos hecho aquí en task solo para que lo recuerden en task Service en task controller Perdón habíamos hecho todo

esto habíamos creado un constructor Tab service la propiedad habíamos creado una variable que iba almacenar esa propiedad y es un proceso bastante largo Esta forma de escribir código se puede resumir un poco más la forma de resumirlo es de esta forma pueden

escribir private al inicio quitan este dis quitan esto de aquí y ya estaría esto es equivalente a lo que tenían hace unos segundos y es una forma más abreviada es por eso que es de esa forma es como vamos a estar importándole le

digo yo tengo un user service quiero importarlo aquí en App controller creo un constructor un private user service o users service y aquí le digo users service y ahí lo importa por mí un

objeto Bueno mejor dicho unas llaves ya que esto es una función y aquí ahora sí ya lo tengo Entonces le digo desde this.to user service puede llamar el get users y pueden ver que es un poco más abreviado guardo y ahora sí si vengo en

la aplicación Slash users enter debería ver el valor Ah bueno aquí me ha faltado colocar el el Slash users Ahora sí voy a guardarlo Allí se debería reiniciar bueno no se está ejecutando vamos a ejecutar en pn room Start 2. Def para

que inicie el servidor Y a partir de aquí si yo vengo en Slash users vamos a ver recargue ahí un poco bueno debería aparecer vamos a ver refrescamos y no se está viendo nada vamos a ver qué nos ha faltado Ah me ha faltado el ret returno

aquí es decir esta función está retornando algo pero yo no lo estaba retornando al cliente Entonces ahora sí refresco y ahí están los datos del usuario entonces de nuevo para repasar yo importo el user service lo inyecto

aquí yo ejecuto un Slash user cuando yo visito esta URL se va a ejecutar esta función esta función va a llamar el user serv get users Esto me va a dar un arreglo y yo voy a retornar el arreglo cuando se visite esta URL entonces por

eso que yo veo este arreglo aquí en pantalla listo Entonces esto es otra práctica que pueden utilizar para saber que allí estamos utilizando también otro service de otro módulo Entonces esto es una forma bast bastante ordenada de poder trabajar en el backend sin

necesidad de estar eh combinando recreando código manualmente hasta el momento hemos estado creando urls que son visitadas a través del método get si entramos en task controller vamos a ver que cuando

visitamos un Slash task estamos viendo el retorno de lo que es una función que de momento solamente nos retorna a un arreglo de tareas Esto se está funcionando a través del método get pero en una aplicación backend no siempre

vamos a tener tan solamente métodos get también podemos tener métodos post métodos put métodos delit y demás Entonces vamos a ver cómo funcionan otro tipo de métodos en njs para poder hacerlo de hecho es muy fácil porque así

como tengo una función get o un decorador get que se va a ejecutar también yo puedo generar métodos post y demás entonces puedo copiar esto y puedo pegarlo aquí debajo y puedo decirle voy a generar un método post esto viene de

nes common y aquí arriba lo ha importado por mí como pueden ver Okay en este post de aquí Ahí está entonces aquí me está dando error porque se está llamando igual pero aquí vamos a colocar otro nombre va a ser create task por ejemplo

y el create tas de momento va a retornar tan solamente un texto por ejemplo va a retornar un texto que diga creando tareas luego aquí en este post también vamos a copiarlo y así como tengo un

post también voy a tener un método put que es para actualizar que también viene de ngs comon como pueden ver aquí arriba ya lo importó y ya no va a ser un create Tas sino va a ser un update task lo mismo voy a copiar este valor de aquí Bueno aquí ya no va a ser creando tareas

va a ser actualizando tareas vamos a copiar esto vamos a pegarlo una vez más así como tengo un put tengo un delete que es un método http arriba también ya lo importo por mí como pueden ver y aquí ya no se va a llamar update Tas sino se

va a llamar delete task y aquí no va a decir actualizando tarea va a decir eliminando tarea y también hay otro método que es el método Patch que es un método también típico para actualizar

parcialmente entonces también voy a importarlo Aquí vamos a importarlo para que puedan verlo es el método Patch Okay y el método Patch lo llaman aquí debajo le dice tengo un método Patch de esta forma y se llama Slash Tas y el método

se llama pat Tas o por ejemplo update algo update Tas status por ejemplo supongamos que queremos actualizar un status Por qué el Patch se llama así porque el Patch actualiza tan solamente una porción un put actualiza una tarea

supongamos que nosotros tenemos un objeto como dice title en algo por ejemplo title con un Stream que diga primer tarea supongamos y luego tenemos un estatus que diga false Si yo envío un

put en teoría lo que quisiera actualizar es todo el objeto es decir quiero cambiar a un objeto que diga title supongamos que quiero que diga segunda tarea o tarea actualizada vamos a decirle y en también Quiero actualizar

el status que el status va a llegar y va a decir decir por ejemplo true entonces si yo actualizo todo el objeto todas las propiedades del objeto estaría enviando un putut pero en el Patch pasa algo distinto porque en el Patch no necesito

actualizar todo en el Patch yo diría yo tengo un objeto que dice title con algún String que dice cualquier cosa y un status que diga false y yo con el Patch le puedo decir tan solamente Quiero

actualizar el status a true Entonces no necesito actualizar el title Espero que con eso lo entiendan un Patch En otras palabras es una actualización parcial y el put se utilizaría para una actualización de todo el objeto entonces

aquí podría decirle actualizar el estado por ejemplo actualizando el estado de una tarea ahora cómo probamos estas peticiones bueno estas son peticiones http Entonces yo tengo que utilizar un cliente red para poder probarlo Entonces

vamos a hacerlo voy a venir aquí en las extensiones de visual Studio code y como les mostré Al momento de configurar el entorno de desarrollo eh vamos a estar utilizando una extensión que nos ayuda a

poder hacer peticiones que se llama thunderc entonces simplemente van a buscar en sus extensiones de visual Studio code viene aquí en extensiones y buscan Thunder cline en en caso ya lo tengo instalado una vez lo instalen

ustedes van a ver este icono de aquí entre aquí yo tengo varios es por eso que aparece tanto pero aquí ustedes van a ver este icono entonces lo que van a hacer es venir aquí en New request y van a decir Okay voy a hacer una petición

get a http 2 localhost 3000 o localhost 2.3000 task envían con la petición get y dice Tas 1 2 3 Esto es lo que ya teníamos en el navegador Ahora puedo enviar una petición post me dice creando

tarea una petición put actualizando tarea una petición delete me dice eliminando tarea una petición Patch me dice actualizando el estado de una tarea tarea Okay ya estamos haciendo todas las peticiones que tenemos aquí ahora algo más que también nosotros vamos a poder

hacer luego es ir creando esto mismo desde otros controladores desde otros módulos Entonces yo Les recomiendo que lo repitan Pero bueno de momento ya con esto nos bastaría entender los métodos http ahora algo adicional Es que aquí en

los Tas controller nosotros estamos retornando un Stream pero de nuevo nosotros podemos llamar desde el task service podemos llamar otras funciones es decir aquí yo tengo en Tas service vamos a ir allí con un control click pueden ir directo ahí Okay al Tas

service Okay y tengo Solamente una función que Tas pero también puedo tener una función create Tas que lo que va a hacer este create Tas es retornar el texto creando tareas puedo tener una función update Tas que diga actualizando tareas una función delit Tas que diga

eliminado tarea y luego puedo tener una función que se llama update Tas status que va a significar actualizando el estado de una sola tarea si yo guardo esto y quiero reutilizarlo en el otro lado pues simplemente le digo de task controller voy a llamar o mejor dicho de

task service el get task Aquí también le diría voy a llamar de dis task service el create task recuerden retornarlo de El update thistas service update tas

Okay del delete lo mismo dis Tas service delete task y lo mismo aquí en al Actualizar estado sería lo mismo dis Tas service update Tas o a status en este caso guardo y lo mismo si vengo ahora en

el Thunder client le digo petición get ahí están task 1 2 3 post cre tarea put actualizando tarea delete vamos delete eliminando tarea y el Patch actualizando el estado de una tarea listo sigue

funcionando igual y ahora también estoy utilizando el Tas service y mi controlador quedaría de esta forma muy limpio solamente el nombre la URL que se va a ejecutar y ya está ahora algo también que quiero que noten y que les va a ayudar a entender un poco mejor

cómo funcionan es es que si se dan cuenta yo estoy repitiendo el Slash Tas en todos lados Slash Tas Slash Tas y demás bueno Esto se puede evitar De hecho si yo dejo limpio en este caso vamos a colocar Si yo selecciono cada

uno de estos nombres en cada uno de estas eh rutas o controladores y lo dejo limpio sin nada como pueden ver ahora tanto el get el post put delete y demás están vacíos lo que yo puedo decir es que como todo esto está dentro de una

clase que la clase tenga ese prefijo porque ese prefijo o texto que se llamaba Slash Tas le precede a todas estas funciones todos se llaman con esa ruta entonces para poder colocarlo lo que yo puedo hacer es colocarlo arriba

en el controlador es decir aquí yo puedo decir controller y le coloco Slash task si yo guardo esto y vengo aquí en el Thunder client en el local Host 3000 le digo una petición get Slash task envío

sigue funcionando post sigue funcionando creando tareas actualizando tarea eliminando eh haciendo un Patch y demás ahora la razón de que funciona es porque justamente este Slash tag les precede a todas las funciones que se están

llamando y bueno este sería básicamente el resumen que vamos a estar usando el día de hoy para entender Cómo funciona el controlador y cómo se une con el servicio para que al final a través de una petición http podamos hacer

determinada lógica de momento vamos a cerrar esto de aquí y continuamos en la siguiente sección hasta estee momento hemos creado un controlador y este controlador hace uso de un service y ese service nos

puede retornar varios valores si yo vengo aquí en getas pueden ver que tengo varios Stream que puedo retornar ahora el tema es Cómo yo puedo retornar otro tipo de datos bueno para poder hacerlo es muy fácil porque ne ya soporta otros

tipos de datos que puede retornar directamente entonces así como retorno un arreglo de streams también puedo retornar un Stream que diga Hello World como ya lo hemos visto anteriormente esto quiere decir que si yo vengo en el local Host 3000 y refresco pueden ver

que dice Hello World es decir local Host 3000 task que es esta función la que se ejecuta cuando visito este get task me dice Hello Word pero supongamos que yo quiero cambiarlo por otro valor que diga 20 por ejemplo guardo refresco retorno

el valor 20 o un número 20.3 que es un número decimal también funciona y me retorna un 20.3 o también puedo retornar un arreglo de números por ejemplo un arrey de 10 33 y un 20 por ejemplo

guardo refresco y allí me retorna un arreglo de números o también puedo retornar un objeto también puedo decirle aquí qu a retornar un object ent aquí le digo voy a retornar unas llaves con un name que diga por ejemplo y de nombre y

una edad que diga de 30 por ejemplo guardo Y si refresco me retorna un objeto ahora si yo quiero validar este object pues lo que puedo hacer o digamos quiero ayudarme a tener tipado estático

puedo generar aquí un una interfaz porque aquí en este getas yo puedo especificar qué es lo que va a retornar Por ejemplo yo le puedo decir este getas va a retornar un String y ya aquí fallaría porque esto no es un String tendría que retornar un texto como este

Okay guardo Y si refresco allí estaría cumpliendo con ese valor porque esto retornando un String Pero si yo quiero decirle que voy a retornar un objeto voy a crear aquí una interface y aquí voy a llamarle user y este interface voy a

utilizarlo aquí ahora este interface va a retornar a algunos Campos Entonces le digo un user está conformado por un name que va a ser String y un ag que va a ser un number Y esto es justamente lo que voy a esperar que retorne entonces aquí

y le digo voy a retornar un objeto donde espera un name y y un H por ejemplo que sea 40 supongamos guardo Y si yo vengo aquí en la aplicación y refresco pueden ver que no Se está actualizando la razón

de que no se esté actualizando es porque no está compilando el código y es porque justamente aquí el get task espera eh o está siendo utilizado por este gle task Pero él al ejecutar esta función no sabe

qué es este tipo de dato y es porque este interface no está siendo exportado entonces lo que voy a hacer para solucionar el error es simplemente colocar un export Y si yo guardo pueden ver que el error en el otro lado ya se fue entonces vengo aquí en la aplicación

refresco y ya me permite ver el valor ahora la razón de que en este caso no me haya permitido ver el nuevo valor y se haya quedado con el anterior es porque no compila hasta que yo corrija el error Okay esto es un tema bastante típico en

N ahora muy aparte de todo esto así como retorno algo eh También yo puedo retornar una respuesta típica de Express como yao les comenté en nes es un framework basado en Express por lo que

también se puede utilizar la Api de Express si así lo necesitamos vamos a ir aquí ya no en Tas sino en controller en Hello controller Recuerden que es una ruta que se llama el inicio pues yo voy a decir cuando se llame la ruta inicial

quiero responder pero no con la típica sintaxis de nes sino con la sintaxis que viene desde Express en Express que es un framework eh tiene tres objetos request response e incluso un next bueno todo eso se puede importar es decir aquí ya

no diría de esta forma ya no voy a recibir solamente un Slash sino que la función Index que es la función que va a arrancar este Slash Pues voy a colocarle que va a esperar un rec de esta forma

que es un request que viene de next common y al ejecutarlo voy a guardar el valor en un request que va a ser de tipo request luego también puedo tener un Rest que es un abado de response voy a guardar una variable llamada response

que va a ser de tipo response este homage de aquí lo que va a hacer es que ya no va a retornarlo de esta forma porque ya no es un texto de nes sino que voy a poder acceder a este response y le puedo decir response.to status 200 y le

puedo decir un sent o un Jason el tema es que este respon de aquí para que pueda funcionar tengo que importar Estos tipos de datos des Express Le digo voy a importar desde Express un request y también un response listo Entonces ahora

sí le puedo decir pun Send o punto Jason entonces aquí le digo voy a enviar un json típico de Express y aquí voy a colocar un message Hello World si yo guardo esto aquí también puedo ver por consola el request.url que es típico de

Express guardo y cuando yo visito el Slash la ruta inicial me dice message Hello World y por consola también veo el Slash que es la ruta que Estoy visitando De hecho si ustedes quieren ver otra cosa por ejemplo digo cuando visiten el

Hello Entonces yo le digo Slash Hello enter y se me muestra el Hello World pero también por consola veo el request URL listo entonces de esta forma es como nosotros vamos a estar utilizando estas petición Entonces esta forma es la que

ustedes llamarían si necesitan utilizar la sintaxis de Express rara vez necesitarían esto pero aún así si hay un paquete un módulo o algo que necesiten extraer de esta forma o escribir la

sintaxis también es disponible los valores o estas funciones que nos da nes en esta sección Vamos a aprender Cómo podemos recibir un dato que viene desde lado cliente a nuestra aplicación

backend típicamente esto se hace desde una petición post una petición put o algún tipo de petición que intenta darle datos al en este caso yo voy a ir en Thunder cin Y si recuerdan cuando hago

una petición post al http local Host 3000 Tas es porque yo quiero crear algo de hecho esto me responde creando tarea pero yo desde el lado cliente desde la aplicación frontend no le estoy enviando nada aún para poder hacerlo voy a venir

aquí en Body que es justamente el cuerpo de la petición y aquí yo puedo enviar algo puedo enviar un objeto que tenga por ejemplo una propiedad que diga title por ejemplo como título de una tarea y dígalo como mi primer tarea Luego

también Puedo enviarle un estatus por ejemplo que diga eh true y listo este valor que yo estoy enviando des el cliente en Post yo lo puedo enviar al backend Pero cómo el backend puede verlo Bueno vamos a venir aquí vamos a venir

en task controller ts y en el create post aquí es donde yo quiero extraer ese valor ahora por qué lo tiene que hacer el controller porque el controller justamente maneja la petición http él recibe el post Entonces él es el

encargado de recibir esa información entonces aquí en el create Tas para poder recibirlo nosotros tenemos un decorador especial que se llama Body este Body nos permite poder extraer esa información entonces yo aquí le digo

@body lo llamo esto viene desde nes es decir aquí arriba se tiene que importar el body Aquí está lo he importado aquí al final okay Y este Body lo que va a hacer es que va a necesitar que yo cree una variable Entonces yo le digo voy a

crear una variable que se llame vamos a colocarle Tas que va a ser de tipo en de momento y este valor voy a querer verlo por consola le digo Quiero ver el valor Tas guardo y ya con esto Si yo envío algo desde el Thunder client de nuevo

envío esto envío voy a cerrar aquí un segundo abro la consola y pueden ver que ya está el valor que enviado desde el cliente entonces cada vez que yo envío algo si le digo mi segunda tarea envío y abro la terminal pueden ver que ya está

el valor mi segunda tarea entonces ya se está recibiendo los datos en este Body o en esta variable que ya lo tiene almacenado ese Body Ahora yo puedo pasárselo al create Tas yo puedo decirle vas a recibir esa tarea Solo que aquí me

está dando un error porque el create Tas realmente no espera nada el create Tas no está esperando nada en su parámetro Entonces yo le digo vas a esperar un task que va a ser de tipo en y aquí vas a ver por consola esa tarea guardo y ya no voy a ver el conso log del task aquí

sino voy a verlo aquí dentro de creas Okay vamos a probarlo nuevamente le digo envía abro la consola y allí siguen mostrándose por la consola la de neas ahora en este Tas service lo que voy a hacer es que cuando se reciba la tarea

voy a guardarlo en algún lugar entonces aquí yo voy a crear dentro de Tas service voy a crear un private Tas por ejemplo que es un arreglo y este arreglo Allí es donde va a ser almacenado esta tarea que yo estoy recibiendo Entonces

le digo Aquí cuando recibas algo nuevo en este Tas vas a guardar esta tarea y luego vas a retornar el arreglo de tareas o vas a retornar la nueva tarea en este caso yo voy a decirle vas a retornar la misma tarea que estamos

recibiendo y ya está Entonces qué es lo que va a hacer esto si yo vengo en Tas en Post lo que termin Tas y envío ahora me devuelve lo mismo Okay pero si yo hago una petición get de todas las

tareas debería ver la lista de tareas Okay debería ver todas las tareas que yo tengo almacenadas de momento Entonces cómo hago eso Bueno vengo en task controller y aquí en get task le digo voy a venir aquí en get task Y en lugar

de retornar este objeto voy a retornar el arreglo de tareas que tengo aquí Entonces le digo Esto va a retornar un arreglo le digo va a retornar un dis pun task entonces en otras palabras el create Tas va a recibir un dato lo vo a almacenar dentro del arreglo y luego con

el getas voy a poder retornarlo otra vez Entonces le digo voy a enviar un post con local Host 3000 que vo a decir mi primer tarea envío dice mi primer tarea esto es el dato que se ha guardado si lleo un get a Slash task me devuelve un

arreglo con un objeto yo digo que quiero guardar otro hago otra petición post mi segunda tarea estatus false envío se guardó porque me está devolviendo lo mismo vengo en get le digo Quiero todas las tareas y me devuelve dos tareas

tengo un arreglo el primer objeto dice mi primer tarea y el segundo dice mi segunda tarea en false Okay pueden ver que í están los valores esto significa que con esto yo ya puedo empezar a Añadir o obtener datos ahora a partir de

esto nosotros ya podríamos ir completando el resto de funciones porque justamente otro dato que también espera un Body es el update pero también tenemos que conocer otro valor que se llama parance para poder terminar estas

otras rutas ahora antes de continuar así como nosotros recibimos un Body también podemos recibir otros valores que nos van a ayudar a poder controlar o ejecutar lógica dentro de estos controller por ejemplo así como yo tengo

un Body también puedo recibir un query Qué es un query en una petición supongamos que yo vengo el Last Tas Supongo que yo al obtener quiero pasarle algún tipo de información adicional como quiero decir por ejemplo t solo quiero

un límite de dos es decir quiero dos tareas nada más si yo le coloco tres Quiero tres tareas o 10 10 tareas en este caso le digo solamente quiero dos tareas entonces para poder recibir este valor o poder procesar qué dato me ha

colocado el cliente en esa URL Yo puedo hacerlo a través de un query Entonces yo puedo venir aquí en task controller y así como tengo un Body Recuerden que esto es aquí en la petición get Okay eso sería aquí Entonces yo le digo así como

tengo un get también puedo recibir un query Desde nest common allí lo he importado por mí el query como pueden ver okay Y esto también lo guarda en una variable Entonces le digo voy a guardarle en query va a ser el tipo en de momento y puedo ver ese query le digo

Quiero ver ese query por consola entonces Esto va a ser que si yo envío la misma petición Tas pero con el signo de interrogación que es para indicar un query el nombre que queramos Cualquier nombre Okay límite incluso podemos enviar dos envío y esto me devuelve un

arreglo vacío Por qué me devuelve un arreglo vacío porque como el servidor se ha reiniciado ya el arreglo se ha reiniciado también entonces ya no existe Pero bueno abro aquí la terminal y pueden ver que í dice límite dos esto es todo lo que yo estoy escribiendo acá si

yo escribo por ejemplo a b c y aquí le escribo un 2 3 y envío Okay le doy en enviar vengo aquí en la consola y dice abc1 2 3 entonces de esta forma también yo puedo extraer algo que se llama query que es ese parámetro adicional para

poder extraer información con esto vamos a dejarlo aquí en la siguiente sección Vamos a continuar aprendiendo algo que se llama patterns en esta sección Vamos a aprender de otro parámetro que también está en ns que así como nos permite

extraer el query y el body también podemos extraer algo llamado patterns que de hecho es muy simple pero para poder entenderlo primero debemos colocar un ejemplo si vengo por ejemplo aquí en Slash task nosotros lo que vamos a

obtener es un arreglo de momento está vacío pero yo puedo crear algo yo puedo decirle voy a venir en en en tander client voy a hacer una petición post sl task Okay y le digo voy a crear un objeto que se diga mi primer tarea le

doy en enviar Y si yo refresco ahí está el valor ahora supongamos que yo tengo otro más le digo mi segunda tarea vamos a enviar refresco y ahí está pero yo tan solamente quiero una de estas tareas cómo lo puedo hacer bueno supongamos que

yo eh le añado un ID Al momento de crearse vamos a hacer eso Si yo quiero quiero Añadir un ID vengo en t service y le digo al momento de crear vas a ingresar una tarea vamos a decir vas a ingresar una tarea nueva pero vas a

añadirle algo nuevo Cómo hago eso Bueno yo puedo crear aquí le digo vas a copiar vamos a hacerlo así vamos a copiar todos los valores de tarea y también le vas a colocar un ID y este ID va a ser la longitud del arreglo de tareas de task

lent dis task lent + 1 es decir inicialmente la primer tarea va a tener el valor uno luego va a tener valor dos es decir este va a tener ID 2 luego ID 3 y así a medida que se vayan creando vamos a verlo refresco no hay nada nada

nuevamente Recuerden que eso est solamente guardado en la memoria del servidor Entonces le digo mi primer tarea envío si refresco le añado un id1 le digo mi segunda tarea envío refresco y está mi segunda tarea y listo de esta

forma yo puedo obtener todas las tareas pero supongamos que hay 10 y yo tan solamente quiero obtener la tarea dos Cómo puedo hacerlo bueno puedo decirle de todas las tareas Quiero la tarea con Slash 2 y si doy un enter Quiero ver eso

pero esa ruta no existe entonces claro yo puedo hacer esto ustedes dirán Okay te creas un task controller así como tienes un get all task vamos a crearnos otro más vamos a crear una copia voy a pegarlo aquí debajo y ya no se va a

llamar get Tas sino se va a llamar get task no necesita un query Así que lo vamos a quitar y en este caso vamos a retornar una única tarea entonces así como tengo una función que se llama get task para obtener todas las tareas también en service voy a crear otra

función que se llame get task en singular sin la s adicional y que va a buscar una tarea ahora cómo lo busco Bueno él va a tener un ID él va a esperar un ID que va a ser un tipo

number y aquí le digo si vamos a buscarlo de hecho le digo de This task vamos a buscar con el método find le digo por cada una de las tareas si la tarea en su propiedad ID coincide al ID

que estoy recibiendo Entonces lo vas a retornar guardo y ya con Esto va a estar buscando la tarea ahora esto lo va a recibir el task controller Okay aquí va a utilizar la función get task pero le espera un ID Bueno le paso el id2 este

sería Buscar la tarea dos guardo Y si yo vengo aquí en la consola digo Quiero la tarea dos refresco vamos a ver esto no está funcionando vamos a ver por qué vamos a venir en qué Tas y bueno en este caso no existe la tarea dos vamos a ver

refresco y no existe Por qué no está llegando a la ruta a pesar de que estoy escribiendo tas sl2 porque la ruta en realidad no tiene ese nombre aquí si nosotros venimos en Tas service Perdón en Tas controller la ruta no tiene ese

nombre Entonces yo le digo esta ruta se va a llamar Tas sl2 Recuerden que aquí todas las rutas les precede este Slash Tas es un prefijo entonces para yo llegar Slash Tas sl2 tengo que Añadir aquí Slash 2 porque esto se va a

concatenar con este valor entonces va a ser arriba para darles una idea empieza con el Slash das y luego se va a concatenar el Slash 2 vamos a guardarlo Y si ahora si escribo refresco ahí está el valor pero pueden ver que no retorna nada Por qué no retorna nada porque no

existe esa tarea cuando yo esco si yo vengo el Hash Tas no hay nada Okay es un arreglo vacío lo que voy a hacer es crear algo nuevamente le digo de post voy a crear mi primer tarea vamos a escribir aquí en lugar de que digan mi

segunda Bueno aquí sería mi primer tarea envío luego digo mi segunda tarea envío y si refresco ahí hay dos tareas Entonces ahora sí le puedo decir de todas las tareas voy a pedir la tarea número dos envío y allí funciona pero yo

le digo Quiero la tarea uno envío y ya me dice que no lo encuentra por qué Porque al momento de que vengo en task controller él esperaba el Slash 2 entonces yo puedo decir okay también puedo crear Slash 1 Slash 3 Slash cu pero siempre voy a tener que estar

recreando Esto entonces como esteo es un valor que va a ir cambiando en lugar de escribir Slash 4 voye colocarle dos puntos y un nombre ID por ejemplo Entonces esto es un parámetro porque nosotros no sabemos Qué valor va a

llegar aquí pero lo único que sé es que va a llegar después de Slash task entonces va a tener que ser Slash task sl sl Tas sl algo por ejemplo sl 1 2 3 y demás Entonces ese es el valor ID ent es

la forma genérica de llamar a todo ese esa posición Entonces cómo recibo ese valor bueno con un ar param un ar param que viene de nest common ahí arriba lo importado para esto lo espero con el nombre ID porque aquí lo he llamado ID

también y lo voy a guardar en una variable que se llama ID que va a ser de tipo stram y este ID lo puedo ver por consola le digo Quiero ver el ID Okay guardo Y si yo refresco esto Tas un pueden ver que no me devuelve nada pero

por consola ahí está el valor uno como pueden ver le coloco valor 100 que no existe enter y ahí está el valor 100 entonces aquí simplemente este ID se lo voy a pasar al get Tas le digo te voy a pasar el ID este ID este getas lo tiene

que recibir vamos a ver V en getas y él dice Ah okay Yo espero un number Okay entonces tengo que convertir este ID a un number bueno para hacerlo es muy fácil yo puedo decir pars int y le digo voy a convertir ese ID a un number

guardo y ahora sí si refresco yí está Pero bueno vamos a probar algo voy a crear un par de tareas post mi primer tarea envío otro post mi segunda tarea envío y si lleo una petición get de tareas ahí está mi primera y segunda

tarea le digo Quiero la tarea uno Tas sl1 Perdón sl1 Ahí va Tas sl2 ahí está mi segunda tarea y si digo Slash 3 que no existe me devuelve un estado vacío es

decir una página sin nada Ahora si supongamos ustedes no quieren devolver eso también pueden validarlo en el controller le pueden decir si el ID y en el Bueno aquí no No necesariamente pueden hacerlo en el en el service

porque es es el que busca realmente le dicen okay No voy a retornar inmediatamente sino que voy a guardar esto como un result vamos a llamarle task Found tarea encontrada Entonces yo le digo vas a retornar la tarea

encontrada pero si no la encuentras le digo si no hay una tarea encontrada lo que vas a retornar Aquí vamos a colocarle que va a retar un texto que diga no se encontró la tarea guardo Y si refresco Tas 3 me dice no se encontró la

tarea ahora en en realidad nest tiene una mejor forma de retornar esto porque ya no sería Simplemente no se encontró la tarea sino que yo puedo retornar un throw que diga por ejemplo New error t not Found Y si yo refresco ahí me dice

status 500 internal server error pero esto es un throw porque básicamente acamba con la ejecución del programa Aquí por ejemplo pueden ver que me dice t not Found Entonces si bien Esto es una forma de acabar con la ejecución del programa No es algo común entonces lo

que nosotros tenemos desde nes es una forma de responder con un mensaje es decir Yo puedo responder un New puedo retornar un return de hecho un New y aquí nosotros vamos vamos a obtener un objeto que viene desde nest common que

se llama http exception o http code de hecho entonces aquí lo que yo puedo retornar aquí en este Tas Found es por ejemplo un error que diga Bad request o Bad request error Por ejemplo en este

caso sería not Found error Okay que viene desde op bueno de ns common not Found ex section y aquí incluso al ejecutarlo pues ya estaría yo le puedo colocar un texto que diga la tarea con es ID no fue encontrada aquí se sería de

hecho un New falto esto guardo y yo vengo aquí y le digo okay Busco la tarea tres me dice la tarea con id3 no ha sido encontrada 404 entonces de esta forma es como yo lo controlaría ahora esta no es la única forma de escribirlo Pero bueno

es algo Bastante típico Entonces ustedes ya pueden luego manejar un poco mejor ahora si busco una tarea uno pues Esto va a ser igual para todos porque esto está manejándose para todas las funciones entonces me hice la tarea con id1 id2 no existen Okay a excepción de

que yo cree algo le digo mi primer tarea envío í se creó Entonces si busco la tarea uno tarea uno dan solamente ahí está la tarea uno pero si busc la tarea 100 supongamos pues no existe Me dice

Tas eh 404 okay o tarea con ID 100 no encontrada listo entonces de esta forma también estoy recibiendo este parans En otras palabras el parans es una forma en la que yo puedo Añadir dinámicamente un valor a la URL y luego extraer ese valor

para poder devolver una respuesta u otra dependiendo de lo que me están enviando ahora vamos a aprender un concepto de njs llamado dto o Data transfer object que lo único que hace es

especificar los datos que vamos recibiendo por ejemplo aquí en este Body cuando recibimos un task le hemos colocado de tipo en pero nosotros podemos especificar qué es lo que va va llegar desde el frontend para poder hacer eso como estoy en Tas voy a crear

dentro de Tas una carpeta llamada dto porque puedo tener varios y dentro de dto voy a crear un archivo que se llame por ejemplo create t.d. ts este archivo de aquí simplemente va a ser una

interfaz así que nosotros podemos crearlo de esta forma puedo decirle interface o también puede ser una clase también puedo llamarle Class Okay entonces puedo decirle Class create task dto vamos a llamarle de esta forma y con

esto aquí adentro voy a especificar qué es lo que espero recibir en este caso le digo Espero recibir un name o un title Recuerden que cu cuando yo envío una petición post aquí le estoy enviando un title y un estatus entonces voy a enviarle lo mismo un title de tipo

Stream y un estatus también de tipo stram ahora de nuevo esto puede ser tanto una clase como una interfaz depende de ustedes pero vamos a exportarlo vamos a cambiar una interfaz de momento ya luego vamos a cambiar una clase y en este interface vamos a

colocarlo aquí en el body Entonces yo le voy a decir voy a utilizar el create task dto y listo Ahora task yo coloco la propiedad si yo digo task punto puedo usarle tanto el title como el

description ahora este task que estoy recibiendo en el service también lo mismo en el también le digo esto es de tipo create task dto y ahora lo que va a hacer Es que este task que estoy recibiendo aquí Él sabe que estoy

recibiendo tanto un title como un description y bueno incluso a partir de allí ya puedo ir colocando otros datos eh especificando un poco más pero bueno eso es básicamente la idea de un dto ahora nosotros utilizando Este dto

nosotros podemos colocar también cuando estamos actualizando o estamos eh Por ejemplo obteniendo una búsqueda o especificando cualquier dato que realmente llega desde el frontend entonces este update Tas también podría

recibirse desde el desde el Front entonces también puedo crear un dto ya no se llamaría create Tas dto también podría llamarle update Tas dto por ejemplo entonces también puedo copiar esto también puedo pegarlo y le llamaría

update tasd to. ts ahora en el update task lo voy a actualizar no se va a llamar create sino se va a llamar update task dto y aquí va a ser un title pero la diferencia es que ese title va a ser

en este caso un dato que puede ser opcional como el description Porque tan solamente puede actualizar uno de los dos entonces aquí el momento del task controller ya no voy a colocarle que el body va a ser bueno aquí en el put ya no

va a ser un Body con un create Tas sino voy a decirle un @ Body que va a ser un task Pero va a estar basado en el update task dto Entonces le voy a pasar ese task allí el update task del service También necesita recibirlo entonces aquí

le digo voy a recibir un task y este va a ser un update task dto y aquí lo puedo recibir lo puedo mostrar por consola Tas pun algo Tas pun title o description de momento aú no lo voy a completar Pero bueno eso es básicamente un dto ahora si

yo quiero ver cómo funciona todo esto pues lo único que vamos a hacer es que vamos a enviar algo por ejemplo yo voy a venir aquí en th client en Post y Si yo envío un post a Slash task esto dice mi primer tarea pero si Igual yo quito este

estatus y ejecuto esto igual sigue añadiendo cosas lo que indica es que en realidad el dto es para especificar una interfaz en typescript no es requerido y esto solamente una facilidad para poder obtener autocompletado y análisis de

errores pero bajo Este dto nosotros podemos extenderlo para poder hacer validaciones y es lo que vamos a hacer en la siguiente sección en la sección anterior hemos aprendido Cómo poder utilizar dtos para

poder obtener autocompletado y análisis de errores en typescript Pero estas interfaces También nosotros podemos utilizarlo para obtener validaciones De hecho si nosotros buscamos en la documentación de njs como njs

validations vamos a llegar aquí en su documentación y nos va a indicar que necesitamos crear esos dto pero también vamos a instalar algún par de bibliotecas una llamada Class validator y Class Transformer vamos a copiarlo y vamos a instalarlo vengo aquí en nes en

la terminal y voy a decir m install Class validator y Class Transformer ahora lo que va a utilizar esto para poder eh validar son los dto que ya tenemos creado Entonces vamos a escribir nuevamente en room Start 2. Def mientras

arranca todo y por ejemplo en el create task voy a cambiar este interface por un Class nuevamente y aquí ya no voy a dejarlo como está sino que voy a importar desde y aquí es donde vamos a llamar los paquetes que es Class

validator y voy a traer desde aquí algunas decoradores por ejemplo supongamos que este title yo quiero que sea un Stream pues le digo Aquí que va a ser is stram Okay y aquí le coloco un

@is stram También nosotros podríamos colocar eh un description Aquí también lo mismo dice Stream y ya con esto estaría ahora el tema es que esto Val que sea un Stream nada más es decir Si yo envío un post en Thunder client y le

envío un title con un número envío pues esto debería validarlo el tema es que el create dto solamente está definiendo el valor pero nadie ejecuta esta validación el que debería ejecutar En todo caso es

el controller es decir en el post cuando yo recibo este create Tas este create Tas dto ya tiene las validaciones antes de que se ejecute yo voy a utilizar un decorador que viene de njs que se llama Us pipes de hecho vamos a importarlo por

aquí arriba este viene des de Us pipes lo voy a importar Y aquí le voy a decir post voy a utilizar este @use pipes y aquí voy a ejecutarlo y adentro le voy a pasar este New y vamos a llamar un una

clase que se llama validation Pipe Okay esto lo que va a hacer es que va a validar ese Pipe y al guardarlo cuando se va a ejecutar va a utilizar este cre para validar de hecho por ejemplo yo envío esto y me dice status 400 el

título Debería ser un Stream la descripción Debería ser un Stream Entonces por ejemplo digo que título un Stream ejecuto y bueno vamos a enviar y me dice la descripción Debería ser un Stream Okay description vamos a

colocarle allí un un String envío y ya pasa pero aquí también voy a decirle que mínimo sea un caracter Entonces vamos a colocarle aquí min vamos a colocarle uno por ejemplo Aquí también vamos a colocarle mint min lent 1 y aquí sería

mint lent 1 Vamos a probarle í ahora para que esto nos dé la propiedad vamos a ver qué es lo que nos falta y de hecho aquí la propiedad está en minúscula pero en realidad es mayúscula min len de esta

forma Okay es como una m mayúscula guardamos y ahora sí Si yo envío y ejecuto entonces el título Debería ser más o mayor o igual a un carácter la descripción igual mayor un carácter entonces escribo algo aquí algo aquí

ejecuto y ya pasa esa validación ahora en realidad esto es para cualquier tipo de modelo es decir yo aquí estoy validando lo que es un task un task tiene un title y un description pero también quizás quiero validar un usuario entonces también lo mismo vengo un user

vamos a venir en users en user service por ejemplo que sería nuestro eh nuestro service para poder crear algo entonces voy a decirle create user este create user va a ejecutar algo va a retornar de

momento un objeto vacío pero él va a esperar un usuario nuevo es decir va a esper un dato de usuario vamos a ir atrás un dato de usuario y este dato de usuario vamos a ir atrás este dato de usuario que yo pasaría voy a tener que

recibirlo de un interface entonces en user voy a crear un dto aquí un create user.dat vamos a exportar una clase que se llame create user dto y Aquí vamos a

colocarle lo mismo le voy a colocar algunas propiedades por ejemplo para un usuario voy a guardar un email un password voy a guardar un name y un ag supongamos entonces aquí el name el email voy a decirle va a ser tipo dat

Stream el password ser de tipo dato Stream el name va a ser de tipo de dato String y el va a ser de tipo dato number y aquí empezamos a validarlo Entonces le digo el email va a ser @ isemail que también tiene otro eh otro decorador

allí para comprobar que sea un correo luego también le digo esto es un Stream Okay y luego vamos a colocarle también que no sea vacío es decir también yo puedo colocarle un eh min lend pero también hay otro decorador que se llama

is not empty Okay Esto es para asegurarnos de que no nos de un dato vacío al momento de registrarnos lo mismo sería con el el password Stream is not empty Okay sería de esta forma Ah

Bueno De hecho voy a formatearlo para que quede mejor el name también lo mismo sería un String sería también un no vacío un campo no vacío y le da también sería un Stream Bueno aquí sería un is

number de hecho un is number y aquí podríamos colocarle un máximo de D un ar le digo Max y este ar Max podría ser 100 por ejemplo tengo que importar este Max también y ahí estaría Entonces esta es

mi validación para el create user dto entonces si yo aquí le digo Esto va a ser tipo create user dto Entonces él va a esperar ese campo ahora Quién va a utilizar esto Bueno voy a decirle Esto va a retornar al mismo usuario que está

recibiendo Okay simplemente va a retornar un usuario pero o también podría retornarle un objeto que diga aquí eh el usuario con un ID le digo el arreglo de usuario más un lent y aquí le

puedo colocar que va insertarse dentro del arreglo de usuarios este usuario que estoy recibiendo okay Y ya estaría ahora Aquí estos usuarios no van a coincidir porque no tienen el mismo eh la misma cantidad de props pero aquí voy a

decirle user de tipo en vamos a colocarle un arreglo de en vamos a guardarlo y ya estaría Entonces vamos vamos a venir ahora en el controller en user controller y al momento de hacer una petición post en ese caso que viene

n comon a user voy a crear un usuario voy a llamar el create user y este create user va a esperar un Body este Body va a esperar aquí un tipo de dato user que va a ser de tipo create user

dto y aquí le voy a pasar el usuario entonces de esta forma es como funciona en N Entonces ahora si yo vengo en este caso en el Thunder cin y una petición post pero ya no de Tas sino de users y

aquí envío como tal envío esto ya me debería dar un error pero bueno me falta el Pipe Okay me falta colocar el Pipe entonces también aquí le digo use pipes y aquí le coloco New validation Pipe vamos a importar eso también ya con eso

estaría con esto estaríamos validando el en este caso a través del dto vamos a verlo envío y ahí tengo un montón de errores me dice necesita el email no debería estar vacío el email Debería ser un Stream el email Debería ser un email

el password no debería estar vacío Debería ser un Stream el nombre no debería estar vacío Debería ser un String y la edad Debería ser no mayor a 100 y Debería ser un número conformado por una un Espe aplicación allí entonces

vamos a verlo vamos a ir colocándolo entonces le digo Okay vamos a ir eliminando cada uno e email como estoy creando un usuario voy a crear un email vamos a colocarle eh

usergioarboleda No debería estar vacía Okay age 30 envío y ahí Ya está ya está validado listo entonces de esta forma es como funcionan las validaciones en nest pero algo más que también pueden hacer es que en lugar de estar añadiendo el

Pipe en cada en cada controller aquí algo que se puede hacer es que si ya existe un dto en cada uno de los controllers que lo valide sin la necesidad que yo añada el use pipes en todos lados Entonces yo aquí lo puedo quitar el validation p también Okay ya

no lo uso aquí y si recuerdan también tenía otro en Tas en Tas también cuando utilizaba el create task también utilizar el p también lo quito de aquí porque ya no lo voy a estar utilizando Entonces no tiene sentido que lo importe pero una forma de import arlo sería

yendo en el Main en el Main en la aplicación principal y aquí voy a decirle que toda la aplicación va a utilizar lo que se llama use global pipes que es como decir un Pipe para toda la aplicación y nuevamente le digo New validation eh validation Pipe y lo

ejecuto aquí coloco unas llaves otra vez guardo y ahora sí todas las rutas van a tener validaciones siempre y cuando tengan un dto esto quiere decir que si yo vengo aquí en perón si yo vengo aquí en las rutas de Thunder client digo voy

hacer una petición a Tas a user Perdón envío y se ejecuta pero quito algo nada más name y Ed por ejemplo ejecuto y ya me do el error y esto recuerden que no tien validation Pipe Si yo envío tarea envío tarea también me dice el título

Debería ser más largo la descripción y demás Entonces ya está validando a pesar de que no está utilizando el validation Pipe otra cosa adicional también es que si yo le paso un dato correcto a las users Recuerden que user espera todos estos Campos y miren si yo le paso un

campo extra por ejemplo le paso no sé un test que diga 1 2 3 o ABC algo así envío y ya lo recibe igual pero esto no es correcto no debería recibir un campo é está validando el resto de Campos Porque

si yo cambio aquí por un un true envío y me da un error okay está validando ese campo solamente aquí este campo como no lo espera no lo valida y lo recibe De igual forma pero si yo no quiero eso también puedo decirle que eh utilice una

propiedad extra que se llama wh list para que limpie los datos que no corresponden es decir yo lo puedo decir aquí en validation Pipe le digo wh list en true y ya con esto Si yo le envío un campo que no corresponde como un test

por ejemplo envío y ya no lo acepta es decir lo acepta probablemente pero no no lo toma Entonces lo ignora entonces aquí no importa cuántos datos Yo añada igual aquí envío y no lo va a tomar Okay es

así de sencillo listo Entonces esto lo que vamos a estar utilizando el día de hoy para poder validar en este caso dentro de de nes ahora si ustedes se preguntan si se puede utilizar otra biblioteca de validación también se puede sin embargo Esta es la que

recomienda la propia documentación Así que es buena idea seguirlo vamos a dar un click derecho Close all para Cerrar todas las pestañas y continuamos en la siguiente sesión hasta el momento cuando hemos

creado rutas por ejemplo en controller supongamos que es una un controlador mucho más sencillo de momento siempre va a retar un 200 de hecho nosotros podemos revisar eso Si venimos en tander client y enviamos una petición a local Host

3000 y al inicial es decir la ruta básica localhost 3000 nada más con una petición get y bueno en este caso lo habíamos llamado Hello vamos a enviarlo y me envía un código de estado 200 Ahora qué pasa si yo quiero personalizar eso qué pasa si yo quiero retornar a otro

código de estado bueno nest ya tiene en este caso decoradores como ya lo podemos adivinar que nos permiten poder especificar otro código de estado por ejemplo supongamos que yo tengo aquí una función que se llama not Found page y

esta función cuando se busca siempre va a retornar un recurso no encontrado Entonces ya le digo return 404 not Found supongamos vamos entonces como quiero que esto retorne un un estado 400 Porque si yo llamo esto con un get por ejemplo

le digo not Found Esto va a retornar un 200 de hecho podemos verlo si yo vengo aquí en Thunder client le digo la petición get a not Found envío y me dice 404 not Found pero Miren el código est 200 entonces para poder indicarle que es

un código estado 400 voy a colocarle aquí http code que es un decorador que viene de common de n comon Aquí está lo ha importado y aquí le coloco el código de estado que quiero por ejemplo va a ser el 404 supongamos guardo y ahora si

hago la petición y envío m 404 not Found Okay lo mismo si yo hago aquí otra petición vamos a copiar esto y quiero por ejemplo cuando un ocurre un error le digo error page si yo no le coloco el

código de estado es decir si yo lo dejo tal cual Y eso se va a llamar por ejemplo error y yo pido la página local Host 3000 sl error me dice Bueno aquí me dice notfound vamos a cambiarlo porque diga error page o error Route vamos a

colocarle error rout guardo Y si envío me dice error Route Y si en este caso hago una petición por ejemplo hago una petición get y quiero que me devuelva 500 Aquí le voy a colocar lo mismo @http

code y le coloco 500 Okay guardo Y si veo aquí y envío pues ya estaría y interno al server er error 500 Entonces en realidad de esta forma con este http code nosotros podemos especificar el código de estado que esperamos en el

Front y bueno Esto es también para cualquier otra ruta incluso rutas correctas No necesariamente 400 Por ejemplo si yo creo aquí otra función que se llame something New alguien creó un dato nuevo entonces Aquí le voy a decir que va a retornar eh con una petición

get something va a ser una petición a New o algo así a New vamos a decirle y aquí va a retornar un http code que sea 2011 y como texto va a retornar lo que diga something New guardo Y si vengo

aquí en Tas le digo vas a pedir sl New con un get envío y me dice 201 que significa que un recurso nuevo se ha creado y ya estaría entonces de esta forma nosotros podríamos empezar a considerar también otros códigos de

estado dependiendo de lo que necesitamos devolver en esta sección Vamos a aprender ahora Cómo funciona un concepto de njs que se llama Pipe en este caso es simplemente una función que procesa algo

y retorna algo alterado para entenderlo mucho mejor vamos a verlo en este caso con algún tipo de ejemplo en nuestro caso ten estamos en el Hello controller supongamos que yo quiero crear otra función que se llame algo como get

number supongamos que creo otra otra función esta función se va a llamar cuando visit el a través de get el Slash ticket sl do pun noom Recuerden el param Bueno aquí vamos a utilizarlo Entonces cuando yo pido ticket y algún número

aquí yo guardar una variable nun Entonces vamos a decirle voy a esperar ese param ese param se va a llamar nun vamos a colocarlo aquí y voy a decirle que se va a guardar en una variable llamada n de tipo number listo este p lo

tengo que importar Entonces le digo importal lo de n es comon allí ya lo importó por mí como pueden ver aquí está ah en esta parte y bueno lo que voy a hacer ahora es que voy a empezar a retornar ese mismo número Y bueno ya con

esto lo que nosotros vamos a hacer es que vamos a retornarlo con una suma vamos a decirle más 14 supongamos Entonces yo envío un número y lo va a tener que retornar igual Entonces qué es lo que pasa cuando yo le coloco este valor vamos a verlo si yo llamo en

ticket en local Host 3000 sl ticket sl algún número por ejemplo 10 pueden ver que lo que está retornando no es una suma sino una concatenación De hecho también pueden verlo en Thunder client Porque da lo mismo en realidad de nuevo

esto digo ticket elash 10 envío y me devuelve 10 14 o coloco cualquier valor eh 30 envío y no me está devolviendo 44 si no me está devolviendo 30 14 porque en realidad esto no es un number A pesar

de que lo he colocado aquí Esa es la variable que lo almacena pero el valor viene como un String entonces lo que tengo que hacer es convertirlo para poder hacerlo ya existe un una especie de función que se le añade a este

parámetro para poder convertirlo y de hecho viene dentro del param porque este es el que lo almacena entonces Aquí voy a colocar una coma y eso se llama pars en este caso in Pipe Okay este pars in Pipe viene desde nest common también

desde la biblioteca nest common y lo que hace es que en este caso va a convertir este Stream a un entero si yo guardo y vengo ahora en el navegador y le digo ticket vamos a colocarle 44 envío y me

dice 58 porque 44 + 14 son 58 Okay lo mismo también aquí en Thunder clan le digo voy a enviar un número que sea 22 22 + 14 sería en este caso 3 se Entonces ahora sí lo está procesando como un

número e incluso ustedes pueden verlo también por consola le dicen Quiero ver type of the n guardan Y si hacen una petición le va a decir que es de tipo number pero si en el pars in Pipe es

decir si ese Pipe y envían le va a decir 22 144 porque es de tipo String Entonces en este caso ya con esto pues ya tendríamos eh la suma de ambos ahora el tema es el siguiente nosotros así como

estamos recibiendo un Pipe También nosotros podríamos utilizar otro tipo de datos por ejemplo supongamos que yo he creado por aquí otra función que se llame user active este is user active accede a

leer un valor un Pam también este param se va a llamar status lo voy a recibir aquí en un get que diga por ejemplo user Eh bueno no user vamos a colocarle active y luego un status una variable

que se llame status y bueno Aquí yo le digo ese va a ser un tipo bullan entonces voy a colocarle va a ser status puntos bullan pero si yo quiero verlo le digo vas a retornar ese estatus y quiero ver de qué tipo de dato es ese estatus

que está recibiendo si yo guardo vengo aquí en la en la barra de direcciones le digo estatus le digo true envío y bueno en este caso no lo está recibiendo vamos a ver Ah Se llama active status Perdón se llama active true envío y me dice

true Okay veo y no es un tipo bullan como pueden ver es un tipo dato stram de hecho lo mismo ustedes pueden escribir aquí active sl false Recuerden que este es el valor Dinámico Este es el param envío me dice false Y ustedes podrían

pensar que es un bullan pero miren es un String entonces lo que tengo que hacer es convertirlo también también lo mismo le digo en el param cuando lo recibes lo vas a convertir a un pars bullan o par Bull Pipe Okay guardo y ahora sí si hago

la misma petición envío me dice false y ese tipo Bulan entonces de esta forma ustedes pueden luego utilizar los típicos operadores de buanos para poder quizás continuar con algún tipo de lógica ahora estos son los pipes que

vienen ya incluidos en la documentación digamos en el propio framework pero también nosotros podríamos crear nuestros propios pipes en caso queramos hacer una conversión más personalizada algo un poco más manual vamos a crear

aquí una función vamos a llamarle grid saludar y esta función va a esperar un query Okay que va llegar a través de la URL este query lo vamos a almacenar y de momento este query va a esperar dos cosas de adentro un name String una coma

y también va a aper un age que va a ser de tipo number lo que va a hacer este grid es que va a funcionar con el método get se va a llamar Aquí vamos a decirle la ruta grid el query lo tengo que

importar en es common recuerden aquí arriba voy a importar el el query Ahí está y lo que voy a hacer es retornar aquí una concatenación que diga Hello el query name y le voy a decir tu edades y vamos a concatenar lo con el query age

ahora si yo veo estos valores le digo Quiero ver el type of the name de query Age y quiero ver el type of the query name y guardo vamos a verlo Yo vengo aquí en la barra de direcciones o

también puedo venir en Thunder cine le digo Aquí voy a buscar un grid voy a pasarle un query y aquí le coloco por ejemplo name Fast y voy a colocarle aquí un amper Stand para colocar la segunda el segundo query le digo el vamos a

colocarle 100 envío y me dice Hello Fast y Eh bueno me dice que tengo 100 años de edad entonces esto de aquí lo que hace es que está concatenando los valores pero el tema es que si nosotros vemos el tipo de dato ustedes podrían de nuevo

pensar que esto es un String esto es un number pero en realidad Cuando lo reciben lo ambos son String porque son una URL entonces ambos llegan de esa forma entonces lo que podemos hacer es que podemos crear nuestro propio Pipe

que procese ya no solo un campo es decir ya no solamente un campo único sino ambos objetos es decir un objeto ambas propiedades del objeto y desde í validarlo entonces para poder crear un Pipe Nosotros también podemos generarlo

rápidamente utilizando El Comando de nest de hecho vamos a eh venir aquí en la terminal cancelo un segundo y le digo nest generate Pipe y Aquí vamos a colocarle que lo va a generar para en este caso sería para el el controller

Hello entonces voy a decirle para Hello vamos a voy a llamarle pipes y aquí voy a llamarle validate vamos a colocarle user por ejemplo validate user vamos a colocarle todo junto en este caso Okay

también pueden colocar cel cas pero bueno voy a llamarle así enter y eso lo que va a hacer es que dentro de Hello va a crear una carpeta pipes y dentro de pipes va a crear un validate user de hecho un validate user Pipe también con su Spec Y qué es lo que hace esto bueno

está creando una clase tiene una función transform espera un valor que es el valor que vamos a pasarle que es el objeto y Bueno aquí recibe una propiedad de metadata que es simplemente para obtener más información ahora lo que vamos a hacer es verlo por consola vamos

a ver qué es lo que tiene esto value vamos a llamarle aquí comillas value y como value para que veamos por consola guardo Y quién va a usar Esto bueno vamos a venir en lugar de estar

llamándolo por ejemplo Al momento de utilizar este query este object de Aquí vamos a pasarlo en el query Aquí vamos a colocarle que estos valores antes de recibirlo vamos a colocarle que va a

utilizar un validate user Pipe Okay esto ya vendría desde la carpeta pip validation bueno valid user valid user Pipe y lo que va a hacer es que ahora va a recibir este valor de hecho vamos a

ejecutarlo nuevamente le digo mpm room St Def vamos a venir allí voy a colocarle vamos a colocarle Bueno de hecho en Thunder client sería el valor para que sea el mismo envío se ejecuta y

por consola pueden ver que el palio que estoy recibiendo es el objeto entero entonces a partir de allí lo que podría hacer es validar o convertir el dato a partir de que recibamos ese value cómo lo haríamos Bueno le podría decir en

validate user en el Pipe le puedo colocar aquí le digo eh vas a convertir con un par int el valor que viene de value punto ag Pero el value punto ag puede ser que sea otro valor entonces

puedo colocarle un punto to String por si acaso y voy a recibir aquí un age number un número al final que representa la edad le digo si ese número es no es numérico vamos a colocarle is not a

number es decir es como que me pasaron un dato como un texto eh Y ese texto no se puede convertir en entero va a dar Nan Entonces yo le digo ese texto es Nan si es Nan entonces significa que no se

pudo convertir ese número es como si intenta convertir el valor Hello Esto no se puede convertir a un entero va a dar un valor n si Y ese es un n significa que me pasar un dato incorrecto entonces voy a decirle vas a lanzar un throw New

error o New http exception sería en este caso y este http section viene también desde common y aquí le voy a decir la edad Debería ser un número y le coloco un código de estado en este caso puedo

decirle hay un una clase que viene desde nes que se llama hdtp status Okay una interfaz y de allí le digo va a ser un Bad request es un bat request que es justamente para indicar que es una un

dato erróneo voy a formatearlo y en caso contrario si todo va bien entonces vo decir vas a retornar todos los valores que ya recibiste pero en la edad vas a cambiar la edad por el age number que ya viene convertido entero entonces con

esto lo único que hace el p es recibir la información eh validarlo convertirlo Y retornarlo de tal forma que cuando yo recibo el age eh la edad aquí ya no sería un Stream sería un number porque ya pasó por el validate user Pipe vamos

a verlo eh yo hago la misma petición envío aquí me retorna lo mismo pero por consola pueden ver que ahora el ag el query ag es un number y ya no es un String antes era un String ahora es un number de hecho eso lo compruebo si el

query age yo yo le sumo algo le digo + 30 entonces yo envío esto y me dice 130 no me dice 1030 sino 130 porque lo ha sumado al ser un número y bueno eso es básicamente los pipes obviamente esto lo van a necesitar mucho cuando quieran

procesar más datos pero de momento al menos esto es una forma rápida de entenderlo hasta este momento hemos aprendido Cómo utilizar controller Services y demás ahora vamos a aprender otro concepto que tiene nes que se llama

guards que es básicamente como indica su nombre es una función que protege a otra función siempre y cuando cumpla una lógica es decir yo voy a V aquí en Hello y supongamos aquí yo tengo una función que se llama eh grid Y esta función grid

retorna a un usuario a partir del query de hecho en Thunder client esto ya lo teníamos ejecutando si recuerdan cuando hago un grid aquí Slash grid con un nombre envío Esto me retorna un Hello con algo ahora yo quiero retornarlo solo

si cumple determinada lógica Entonces yo puedo colocar una condicional aquí pero también puedo utilizar un guard que puedo luego reutilizar entonces para poder crearlo yo puedo venir aquí en la consola y le digo nest generate guard y puedo decirle En dónde quiero crearlo

que sería la carpeta sl Hello sl guards es Slash out vamos a llamarlo porque va a utilizar una especie de autenticación supongamos y esto lo va a crear dentro de Hello así como tengo pipes también va a crear un WS y dentro de WS Crea una

clase y bueno esta clase de momento retorna true de hecho qué es lo que hace que nos permite Ver información de hecho si yo quiero probarlo yo le digo conso Aquí vamos a ver y le digo por ejemplo

Quiero ver el información del request bueno para poder extraerlo é nos da una especie de sintaxis un poco complicada o larga de escribir pero bueno este context es básicamente la información de toda nuestra aplicación o toda la

petición le di context voy a cambiar a http eh Quiero obtener request Esto va a ser de tipo request esto lo vamos a guardar en request y aquí lo que vamos a hacer es que vamos a ver por consola el request eh vamos a decer URL por ejemplo

vamos a guardarlo Y si ejecuto esto eh vamos a decirle con nes nuevamente r Start Pero quién va a utilizar Este adward voy a venir aquí en Hello controller y aquí voy a eh utilizar aquí

en este grid así como tengo este get voy a decir voy a utilizar un usws aquí un @ usws y aquí voy a decirle el el nombre del Word que tenemos que se llama outward si mal no recuerdo okay Y al

guardarlo Pues debería Al momento de hacer la petición grid pasar por ese ese decorador ese usw de nuevo viene desde nestjs y el outward viene desde nuestra carpeta O Nuestro archivo que hemos

creado listo vamos a probarlo vengo aquí en Thunder client y le digo vo hac la petición del grid con el nombre y demás envío y ahí está y si abro la consola eh debería ver el valor de hecho aquí está la URL request URL ahora algo más que

también puedo hacer a partir de allí puedo decirle en lugar de return true puedo decir return F guardo Y si ejecuto aquí envío y me dice forbiden es decir ya no puedo acceder al recurso ahora lo

que pasa es que dependiendo del dato que yo reciba pues puedo continuar o no entonces yo voy a hacer una condicional le digo Aquí vamos a decirle si request pun URL vamos a decirles Igual igual a

gr grid pues vas a continuar o vamos a decirlo algo así return true Okay caso contrario si tiene un un valor distinto un query o algo distinto pues va a retornar false Entonces yo le digo Aquí

vamos a colocarle vamos a enviar esto y me dice forbiden pero si coloco solamente el grid sin nada de este valor envío bueno debería retornarlo pero claro est no es false porque esto esper el valor de hecho aquí sería lo contrario aquí diría si es un grid nada

más retorna false y si vas a retornar esto vas a retornar true Entonces qué significa que si solamente es un grid solo no va a continuar pero si sí tiene los datos correctos es decir es distinto a grid pues va a poder continuar

normalmente vamos a probarlo le digo grid envío esto funciona bueno debería funcionar bien Ah bueno está bien no retorna pero voy a retroceder para colocar el resto de Campos name y ag envío y allí funciona solamente que

cuando quito este valor ya cambia me coloca el forbiden entonces básicamente una condicional que interactúa o hace un intermediario pero justamente esto les puede ayudar al momento de que ustedes quieran comprobar roles Por ejemplo si

tienen un usuario con un rol ustedes pueden decir si el eh la sesión del usuario tiene un rol entonces continúa y ya ese adw lo pueden reutilizar porque este Hello controller este usw lo pueden copiar y lo pueden también hacer en

active Eso quiere decir que si van en active vamos a ver Slash active en true envío Esto me devuelve true pero claro aquí ya no no sería con esta lógica de la URL podrían hacer otra cosa le pueden

decir si request vamos a colocarle si request headers eh tiene una una propiedad llamada autorisation Por ejemplo Y esta es igual igual a true

Bueno no sé igual a ABC Entonces vas a retornar un true allí nada más vas a retornar un true bueno en este caso como es false de diría si es distinto de ABC pues entonces retornas un true De hecho aquí me está dando un error Ah bueno en

este caso es porque el Bull en String no va va a compararse vamos a decirle si Y en este caso Bueno aquí en este caso por la condicional vamos a colocar aquí si no hay un autorisation nada más retor

false entonces por ejemplo el momento de hacer la petición active through envío me dice forbiden pero si le añado un header un autoriz tal cual estoy enviando y le digo cualquier cosa envío y ya me deja entrar Okay ahora esta

misma lógica funciona también para el otra ruta grid es decir si yo pido grid y envío con el nombre y demás envío Ahí está eh pero si le quito el autoriz y envío me dice forbid entonces como pueden ver para eso se trata los Word

para poder colocar una condición y gracias a ese Este usws lo puedo importar en varias urls y ya estaría también hay otra opción de importar en el controller Pero bueno en este caso es la forma más típica que ustedes

utilizarían para ir validol de ruta en ruta en esta sección Vamos a aprender acerca de los middlewares los middlewares es un concepto que muchos frameworks de backen lo tienen y básicamente hace referencia a una

función que hace de intermediario es decir es una función que nos deja continuar con determinado URL siempre y cuando nosotros cumplamos una cierta lógica o también nos permite para hacer otra tarea por ejemplo registrar la

petición que va llegando o ir haciendo un tipo de consulta adicional para poder entenderlo vamos a venir a nuestro user controller dentro del del módulo de users y aquí nosotros teníamos algunas

peticiones algunos controles ahí básicos pero lo que vamos a hacer ahora es crear un middleware para todos estos es decir que cada vez que yo llego en get users o en Post users va a tener que pasar por una función y de esa forma voy a poder

registrar las peticiones que van llegando entonces para poder hacerlo ya existe un comando Entonces vamos a cancelar aquí la consola le voy a decir nes generate middleware y este middleware aquí yo puedo generarlo pero

en este caso Vamos a darle un nombre por ejemplo digo generate middleware Bueno aquí debí colocarle un nombre aquí me dice cómo vas a llamar El middleware Bueno vamos a colocarle y aquí decimos En dónde queremos crearlo en este caso

voy a llamarle eh dentro de users y vamos a colocarle algo como simplemente logger o algo así logger okay Y dentro de users que se crea una carpeta logger y ahí está logger middleware ahora esto

de aquí en ningún momento en el module si nosotros entramos en el user module se está llamando porque es una función que simplemente lo vamos a necesitar cuando realmente haga falta es decir cuando queramos pasarlo entonces no es

necesario que esté aquí también lo pueden quitar pueden eliminar y pueden volver a generarlo sin ningún problema y en este caso pueden decirle dentro de user pueden moverse dentro de user le

dicen CD src CD users de hecho CD users ahora listo ahora estoy dentro de este users y le puedo decir nest generate middleware o mi también es otra forma general y el nombre logger directamente

y dentro de Us se va a crear aquí un logger y logger middleware Okay entonces también es otra forma de generarlo Okay simplemente para que puedan ver otro enfoque vamos a decirle nuevamente npn room stard y bueno este logger

middleware lo que va a hacer es que simplemente tiene una un request un response y un next que es muy similar a Express Si alguna vez han llegado a utilizarlo Pero bueno este next es para que continúe con el resto de la función

Es decir para que continúe habitualmente Pero antes nosotros podemos hacer algo es decir Yo puedo decirle Muéstrame por ejemplo de request la información del original URL que es como la información de la URL que llega ahora est request

vamos a cambiar el en voy a decirle desde Express voy a importar su request y también el response también puedo importar le digo Este es de tipo request y est es de tipo response de Express

listo entonces de esta forma si yo guardo ahora ya me puede dar autocompletado le digo Quiero ver el original URL el original URL es la url Entonces esto Quién lo va a utilizar bueno todos los controllers que tenga

aquí si yo pido Slash users o post users ambos van a pasar por ese middleware para poder utilizarlo entonces tengo que llamarlo pero en este caso lo haría desde es decir voy en user module y aquí

dentro de user module voy a decirle vamos a utilizar un use pero en este user module voy a tener que importar entonces voy a decir implementa desde nest module Okay y des Aquí voy a tener

que utilizar una función configure y este configure aquí va a utilizar un consumer y este middleware consumer va a ser justamente el que me va a permiti utilizar el middleware que Acabo de crear entonces Aquí le voy a decir consumer apply y aquí le coloco mi

logger middleware que Acabo de crear que viene la carpeta logger logger middleware y aquí le digo for y le coloco en este caso el nombre de las rutas que sería el nombre de users guardo y ahora todos los user controll

todos los users que est allí pues van a ser o van a pasar por ese middleware cómo lo compré pues vamos a venir aquí en nuestras peticiones vamos a venir aquí en Thunder client y le digo voy a

hacer una petición a Slash users Okay get sl user envío Esto me devuelve lo mismo pero por consola voy a ver Slash user que es el original URL De hecho si yo vengo en loger middleware y le digo por ejemplo no sé aquí voy a colocar una

coma y le digo middleware y guardo pues ya con esto yo puedo volver a ejecutarlo y pueden ver que dice middleware users ahora no es la única forma También nosotros podemos colocarlo a nivel de función tan solamente es decir

supongamos que yo aquí en el momento de utilizar el user mod Yo aquí estoy usando a nivel de módulo es decir lo lo estoy configurando para todo el módulo de usuarios Pero si ustedes quieren también pueden llamarlo solo en unas

cuantas rutas pueden venir por ejemplo aquí en for rout Yo no le digo no es para todas las rutas users sino es para y vamos a empezar a nombrarlo Aquí voy a colocarle un de Debería ser un objeto de

hecho aquí okay un objeto un pad y le digo va a ser para la ruta users pero con el método vamos a colocarle el método que vendría desde request method request method va a ser punto get por

ejemplo entonces para la ruta get nada más va a funcionar vamos a guardarlo Y si yo vengo aquí y hago la petición get a users Ahí está middleware users pero si hago una petición post a users aquí

pueden ver que no se ejecutó O sea yo puedo enviar y no se ejecuta Okay pero si yo le coloco aquí en user mod le digo también vas a colocar aquí una coma y le digo también voy a colocar el pad user

pero con el método post y ejecuto allí Ahora cuando yo llamo también se va ejecutando ese middleware Okay en método post también entonces es una forma de ir cogiendo rutas en donde nosotros queremos ejecutar ese middleware ahora supongamos que nosotros no tan solamente

Tenemos uno sino que tenemos una serie de middlewares que queremos ir ejecutando Bueno también podemos ir creando varios entonces La idea es que aquí con este archivo nosotros estemos validando uno solo pero también

podríamos crear otro middleware que ya no solamente sea de logger sino sea de autenticación ación lo mismo vamos a eh abrir la terminal y le digo nes generate middleware se va a llamar

dentro de users vamos a crear otro que se llame out por ejemplo eh coloco allí otro y se va a crear otra carpeta allí que se llame users out out middleware y en out middleware nosotros validamos otra cosa por ejemplo lo mismo le digo

este va a ser de tipo request y algo aquí le digo est va a ser tipo request de Express Okay lo importó este va a ser response de Express vamos a importarlo y Aquí vamos a colocar una validación le

digo eh desde vamos a colocarle un cons eh vamos a importar autoriz de autorisation lo importo de requ header y allí le digo si no hay un autoriz si no

hay autorisation voy a lanzar aquí un throw que diga no autorize por ejemplo y ya estaría o también puedo en realidad para no acabar con la ejecución del programa Le puedes decir New http exception que es una forma de decir le

voy a enviar una sección al frontend le voy a decir no autorizado y aquí voy a colocarle un http status que es un una interfaz y le voy a decir no autorizado pero si sí lo tiene eh pues vamos a

decir que continúe o también puedo colocar otra condicional le puedo decir aquí si el autoriz es distinto vamos a colocarle a x yz1 2 3 Entonces si es

distinto a esto Pues también voy a colocarle aquí un throw New action forbiden ya está quién vo a utilizar Esto bueno lo mismo yo lo puedo venir luego en el en el user module y como ya

tengo un middleware ahora ya voy a Añadir este en este caso yo tengo aquí el configure con el consumer aquí en el apply yo tengo el loger middleware simplemente voy voy a volver a llamar el apply aquí le digo cuando termine este

primer apply voy a llamar otro segundo y aquí sería el out middleware y le digo lo mismo for routs y aquí puedo colocarle todas las rutas users por ejemplo y ya está entonces de esta forma

el primero solo sirve para registrar las peticiones que van llegando y el segundo m sirve para realmente autenticar o para poder ver que si el usuario puede entrar o no en la aplicación vamos a probarlo

ejecuto aquí con bueno mpn room stf y vengo en local Host 3000 users con el método post envío y bueno vamos a ver que cargue vamos a ver ahí Okay no autorizado genial no autorizado hasta

que yo añada un en este caso si yo vengo en el out midw tengo que Añadir un autoriz vamos a ir ahí en out Bueno sería header Perdón en autoriz lo marco envío no autorizado forbidden Okay ya no

me dice no autorizado me dice forbiden pero tengo que enviarle un x yz 1 2 3 envío y allí recién se ejecuta Entonces ya pasó por el middleware pero a su vez también iba pasando Si vemos la terminal eh también iba pasando por el middleware

normal Entonces no solamente pasa por el middleware de autenticación sino también por el middleware de login l entonces de esta forma ya estamos conociendo los middlewares eh es muy similar a los War de cierta forma solamente que esto está a nivel de http Entonces cuando nosotros

hacemos una petición Esto va a recibirse como una petición va a procesarse y Va a continuar Entonces nosotros podemos parar aquí la ejecución o continuarlo en este caso lo que hacíamos es que si el

usuario no no tenía la autorización o el header de autorización pues acabamos allí no continuaba pero si ya pasaba las las condicionales y no lo paraba pues continuaba con la ruta normalmente básicamente de eso se trata los

middlewares en nes en esta sección Vamos a aprender un comando que en mi opinión es uno de los más útiles de nes y que nos va a ayudar a poder generar apis de una forma mucho más rápida hasta el momento hemos

generado muchas del Código de nes que típicamente se crea desde cero hemos creado controladores servicios hemos creado WS hemos creado archivos incluso hasta de testing que se van creando de forma automática y también hemos estado

generando middlewares y WS ahora muchos de estos recursos que les estoy mostrando que digamos nos favorece los com nes en crearse Rápido también se pueden incluso crear mucho más rápido si

conocemos un comando de nes que se llama resource que básicamente como indica permite crear recursos Qué es un recurso por ejemplo en task yo tengo service module controller todo eso es un recurso de Tas con su dto incluso entonces en

lugar de estar creando primero el service luego el module luego el controler Y luego el dto fácilmente yo puedo crear todo eso con un solo Comando de hecho vamos a verlo vengo aquí en la consola y le digo vamos a venir aquí voy

a cerrar Esto bueno se la consola le digo n generate y supongamos que quiero generar un curso Entonces le digo generate resource y Aquí vamos a colocarle payments Entonces quiero generar un nuevo recurso una nueva carpeta que se lar payments y va a tener

también todos los recursos entonces aquí me dice Qué quieres generar r Api grql o microservicios o websocket esto es muy importante porque n realmente permite ser generado o permite crear Rutas para

Api pero también tiene la forma de poder generar código de graphql o tener una arquitectura de microservicios que funcionen bajo por ejemplo algún tipo de conexión tanto en Red o en este caso no http que significa no utilizando el

protocolo htt tp sino quizás a un un ue o algún tipo de software de ese estilo también tiene los websockets que es para poder tener conexión en tiempo real en este caso Vamos a darle rap nada más porque es lo más común y me dice qué quiere generar un crud Vamos a darle que

sí y y ya está entonces aquí mira lo que va a crear va a crear un controlador con su Spec un module un service un service también con su Spec y un dto más encima un entity que es otro archivo que indica

solamente el digamos el la interfaz que necesita ahora si yo vengo en payments ya está creado todo ese bloque de código que antes nosotros habíamos creado en manual ente si yo ven p modle Ahí está está ya enlazado con su controller y su

servicio entonces controller Incluso el controler ya tiene todo la ruta crud creada tiene el post El get el get único el Patch el delete e incluso si yo vengo aquí en este service los service claro

de momento vamos a ver los serv de momento solamente están retornando un Stream como se los mostré en un cierto punto en el en el curso Pero bueno con esto yo ya puedo empezar a reemplazarlo con mis consultas de backend obviamente no te va a crear las consultas de

backend porque es ya mucho más personalizado Pero al menos ya tenemos un gran Avance en cuanto al todos los la estructura de archivos que vamos a necesitar Entonces el entender paso a paso qué es lo que se necesita crear

también ayuda a que este Comando se vuelva mucho más útil porque ya saben todo lo que conlleva crear esto y también no es que vean algo raro todo lo que ven aquí es lo que ya hemos visto hasta el momento Quizás lo único que no

han visto es este entities que no es algo distinto simplemente una clase que indica el tipo de dato que es casi lo mismo que el dto Okay solamente que esto es para poder utilizarse en Al momento de tener autocompletado o especificar

otro tipo de datos pero de nuevo no es algo enteramente distinto de lo que es el dto listo entonces con esto ya hemos generado un recurso ahora si luego quieren generar otros más pues ya podrían simplemente reutilizarlo por

ejemplo lo mismo n generate resource y cambian el nombre y de esa forma continuarían hasta el momento hemos aprendido Cómo crear muchas rutas

controladores servicios y demás en nest pero otra cosa que nos falta y es muy importante es también Cómo conectar una base de datos si ustedes buscan njs database o algo así van a llegar en la documentación en donde ustedes van a

poder encontrar más de detalles de cómo poder Añadir una conexión ahora pasa algo cuando hablamos de conexiones de base de datos es algo tan complejo o puede ser algo muy sencillo en este caso cuando hablamos de cosas muy simples

nosotros podemos conectar directamente un paquete de npm para conectarnos a una base de datos o algo que hace n y que incentiva mucho es el uso de orms Porque si nosotros vemos aquí me dice Cómo puedo conectar un orm llamado type orm

que es un orm escrito en typescript y es para proyectos de nes y combina muy bien con la idea de nes de utilizar clases decoradores y demás pero no es el único si ustedes siguen bajando también van a

ver otras opciones por ejemplo también tiene la opción de conectarse con db y él ya tiene un un módulo especial para db de hecho todos estos ejemplos de utilizar mongodb o t urm ya lo he creado en el pasado así que les

voy a dejar los videos en la descripción de este curso para que ustedes puedan practicar con la base de datos que más se le es familiar ahora en mi caso en el caso de la base de datos o el orm que

voy a enseñarles para este curso al menos de forma muy básica voy a estar utilizando prisma Entonces vamos a buscarlo de esta forma n es prisma Okay y vamos a llegar en la documentación por hecho Aquí vamos a llegar en recip Creo

que lo tiene y Bueno aquí lo que me indica es Cómo utilizar este orm para aquel que no conoce qu es prisma prisma es un paquete de node que les permite comunicarse con la base de datos pero también les va a permitir generar la

base de datos o generar las tablas perdón de la base de datos esto es muy útil porque de esta forma utilizando la sintaxis de prisma ya se va a generar todas las tablas y no vamos a tener que hacerlo manualmente de hecho eso lo mismo que hace typ RM Solo que prisma

también genera tipos de datos Entonces no tenemos que escribirlo manualmente ahora para poder u inst arlo vamos a tener que instalarlo aquí me da un comando de cómo generar un nuevo proyecto esto no lo necesito solamente Necesito instalar prisma Así que voy a

copiar esto vamos a venir aquí voy a instalarlo np install prisma lo siguiente es inicializar lo con npx prisma o npx prisma init vamos a copiar esto ahí se instaló luego el siguiente es copiar el comando en px prism damos

un enter y por defecto esto trabaja con post sql Aunque en realidad no es necesario que sea post sql puede ser mysql también sin ningún problema y bueno lo que hace prisma es que crea una carpeta llamada prisma Okay donde Aquí

es donde va a estar la base de datos o las tablas de la base de datos luego también hay un archivo llamado punto inv que se ha creado y aquí estaría la dirección de postre sql y ya estaría de esta forma yo puedo conectarme ahora la

base de datos tenemos que ejecutarla de algún modo si ustedes no tienen post sql Al momento de entrar en su proyecto yo les voy a dejar un enlace donde les voy a mostrar Cómo instalar post sql en Windows y en Linux ahora en mi caso yo

voy a ejecutarlo con docker que es algo muy común que yo hago pero no necesario que lo usen con docker pueden usarlo de nuevo con una instalación normal de pod sql para aquel que no conoce docker no es neces necesario Okay en mi caso voy a

generar un docker compos y ML y lo que yo hago es que estos días utilizo el copilot para generar la documentación entonces puedo presionar control I y le digo eh generate a toer compost eh for

post sql Y eso lo que va a hacer es que va a generar una configuración básica de post sql en docker y bueno lo que significa esto es aquí que el usuario de postre va a ser Fast por ejemplo y le digo mi contraseña va a ser Fast

password e la base de datos vamos a colocarle nest db y no necesito los volumes el puerto Aquí voy a colocarle el puerto de mi base de datos va a ser 5499 por ejemplo y ya listo con esto

tengo una configuración para iniciar un podest entonces aquí le digo docker compost app d y Esto va a lanzar una base de datos de podre sql Entonces ahora sí en el archivo in ya puedo conectarme utilizando las credenciales

aquí va a decir podre sql cuál es tu usuario mi usuario es Fast mi contraseña es Fast password mi local Host va a ser 5499 que es el puerto donde está ejecutándose por sql y bueno todo esto

de aquí lo único que cambiaría es la base de datos también porque aquí diría nesdb y bueno a lo que voy es que si ustedes no utilizan docker en realidad lo único que necesitarían cambiar aquí es la url de su base de datos que en su caso sería casi lo mismo sería postes el

usuario quizás la contraseña que utilizaron cuando instalaron postres el local H sería 5432 y la base de datos la base de datos que ustedes vayan a crear creen una base de datos Okay en mi caso

simplemente lo voy a dejar así con la base de datos que voy a estar utilizando el día de hoy que se llama nesdb el Listo ya con esto él va a conectarse y va a generar la base de datos de hecho lo interesante es que yo vengo aquí en

prisma y y le puedo decir voy a generar un modelo de usuario y este modelo va a tener por ejemplo ID Bueno vamos a colocar algunos Campos únicos de hecho voy a colocar estos Campos que me está

sugiriendo sería ID email name password y fecha de creación estos Campos que ustedes Ven aquí de prisma es una sintaxis especial Que tiene primma para indicar qué es lo que va a generar al final dentro de la base de datos voy a

guardarlo y para poder utilizarlo simplemente voy a abrir la terminal y aquí voy a decir mpx y ya no vamos a decir prisma in sino voy a decir prisma migrate Def y guion name init vamos a

dar un enter y listo Esto va a conectarse a nuestra base de datos como pueden ver y allí lo que va a hacer es que va a generar todos nuestros nuestras tablas Bueno en realidad solamente es una pero lo que va a hacer en realidad

prisma es utilizar esta sintaxis y va a generar una tabla de sql o la sintaxis sql de una tabla de usuario y esto lo va a insertar dentro de la tabla de postes y listo con esto Ahora si yo vengo en mi proyecto vengo aquí en src y supongamos

que en user Yo quiero guardarlo en una tabla de verdad pues vengo aquí y tengo que conectarlo ahora para conectarnos a pris desde nes es muy fácil porque él ya nos da por ejemplo en este caso vamos a buscar vamos a seguir bajando ya nos da

una porción de código que puedo utilizar y que puedo crearlo en el Root que se llama src le entro le digo prisms.org pues vamos a empezar a llamar la función

Pero esto se tiene que importar con conor al ser un servicio entonces voy a quitar aquí ya le digo ya no voy tener un arreglo de usuarios ya eso se va Aquí voy a decir constructor y aquí voy a colocarle que va a esperar aquí por

ejemplo un private e prisma y Esto va a venir desde prisma la clase pero esto viene en este caso del módulo Recuerden que nosotros tenemos un prisma service de la clase y listo con esto ya puedo empezar a utilizar los datos Entonces le

digo desde dis punto y aquí é lo que va a obtener dis prisma le digo prisma el usuario Quiero buscar todos los datos que vengan del post sql y ya está est es una te base de datos bastante sencillo lo mismo también aquí le digo cuando

recibas un nuevo usuario eh Ya no tienes que añadirle un arreglo si no lo que vas a hacer Vas a retar de dis prisma el usuario vas a crearlo y los datos que vas a insertar vienen de user guardo y

ya estaría Entonces estos datos son la forma de crear en un en prisma y obtener datos de prisma vamos a verlo npn room stf y esto ya sería la consulta directa de la base de datos Tan sencillo como

eso Entonces yo vengo en localhost 3000 users vamos a ver si me responde algo bueno vamos a ver creo que acá ido algo okay el tema es que claro es llamado El service pero aquí en el user module no le he dicho que que utilice ese service

entonces aquí le providers también vas a llamar el prisma service guardo Ahora sí ejecuta ejecuto users y me dicen no autorizado Bueno aquí me dice no autorizado porque está pasando por los midw vamos a comentar estos midw de

momento m este consumer estos dos consumer vamos a comentarlos e vamos a refrescarlo y me da un arreglo vacío correcto pero ahora yo voy a crear algo entonces voy a venir en Thunder cine le digo voy a generar un usuario en Post

envío esto y bueno me dice no puedo enviar a users bueno se llama users envío interno al server er Por qué Porque quizás prisma No espera ese valor aquí é Espera yo le estoy enviando un age pero me dice no encuentro el argumento age te refieres a name Y es

que los datos que yo le estoy enviando no coinciden con lo que él espera Él espera que le envíe un email un password un name un ag Nova Okay El Noa Y qué más y nada más envío esto y bueno ahora me

dice la edad Ah bueno eh ahora está pasando por la validación Recuerden que el user module cuando nosotros lo validamos en el user controller esto al final cuando se hace una petición post está utilizando el dto y el dto espera

que yo le pase un age esto ya no va entonces le quito el Ed le digo no es necesario que le pases un age acuerdo y nuevamente vengo en el post de Thunder cine y ahora sí ya lo creó y miren le he colocado un ID con un Unique ID un ID

dentro de postes un email un name un password y una fecha de creación y actualización lo que significa que si lleo la petición get the users ahí está el dato guardado dentro de una base de datos ahora con estos datos guardados pues ya lo que haríamos es es seguir

haciendo consultas desde otras rutas Entonces no hay mucha diferencia a partir de aquí ya sería conocer el módulo de prisma esto ya lo he creado en un video por aparte Así que también les voy a dejar el video en la descripción en caso quieran conocer Cómo crear una

ap utilizando prisma ines Yes en esta sección Vamos a aprender ahora cómo documentar nuestra ipi de forma rápida para poder hacerlo no es necesario como configurar módulos

complicados de hecho nes ya tiene pensado eso al ser un problema bastante común en Api cuando hablamos de documentación hay un paquete en nes que pueden buscar pueden buscar como nes ns

docs o documentation documentation generator y van a llegar en varias documentaciones de nes hay unao que se llama compodoc o también pueden buscarlo de esta forma nests vamos a ver nes

swager qué swager bueno es también otro generador de de documentación que se tiene que configurar de hecho este es el más popular Así que yo les recomendaría empezar con este y luego pueden cambiarlo a otro pero bueno La idea es que nosotros vamos a Añadir esta porción

de código de aquí esto viene desde un módulo de n llamado swager vamos a copiarlo vamos a venir aquí en main.ts vamos a pegarlo por aquí y luego tenemos que Añadir esta porción de código Bueno de hecho tenemos que instalar también el

mod mó n swager vamos a abrir la terminal y vamos a pegarlo allí Okay Allí se instala de hecho este módulo es bastante ligero no es como un módulo muy pesado Entonces lo siguiente sería colocar esta configuración de aquí de

config y demás hasta la parte donde dice swager module vamos a copiar esto y vamos a añadirlo dentro de en este caso bootstrap después de global pip vamos a pegarlo por aquí y este swager mod

tenemos que añadirlo entonces en App Listen así como tenemos el app 3000 y demás Con esta configuración de swager mod ya estaría listo dentro de nuestra aplicación ya que lo está ampliando el

app Okay vamos a guardarlo y aquí ya está instalado Entonces le digo mpn room St y Listo ya podemos empezar a documentar nuestro proyecto ahora para poder hacer esto en realidad ya existen también decoradores entonces no tenemos

que crear un yml ni nada de eso y simplemente venimos en la ruta que queremos documentar por ejemplo vengo en users en controller y aquí tengo dos Rutas para documentar entonces lo que puedo hacer es esto puedo venir y le

digo okay todas estas si yo vengo en la documentación en localhost 3000 slapi esto ya generó la lista de todas las rutas que tengo en mi aplicación pero pueden ver que están desordenadas

Todas están conjuntas puedo empezar a separarlos para poder agruparlas yo puedo crearlas gracias a una etiqueta que se llama tax es decir Yo puedo agrupar todas estas bajo controller todas esta bajo Tas y demás vamos con

primero con users que son estas dos y le digo estas users vamos a colocarle aquí @api tax que viene de net swager y aquí le digo esto pertenece a la etiqueta users y esto es lo mismo para el

siguiente post guardo Y si refresco vamos a ver se crea una segunda ruta o una sección llamada users luego lo mismo si yo le digo okay Voy a venir en task le digo Aquí en task controller vamos a

decirle que todas estas rutas vamos a colocarle aquí @api tag también lo puedo colocar aquí y le digo se va a llamar task guardo Y si refresco aquí y veo ahora todas las rutas dentro del

controller o dentro de la clase controller pertenecen al grupo task lo mismo también con payments en payments Yo vengo el payment Se generó como un resource recuerdan payments sería

controller le digo todo esto pertenece a ipi tax como payments guardo y refresco y ahí se separan Tas user payments entonces aquí están las documentaciones ahora esta es la forma más básica pero

luego también ustedes pueden ir detallando más textos por ejemplo supongamos que en tareas vamos a verir ahí en Tas en controller le digo por ejemplo que aquí voy a colocar una

descripción de estas tareas aquí le digo al momento de obtener estas get all task vamos a colocarle Api operation y aquí describen el texto sumary le dicen get all task esto tiene todas las tareas si

refrescan dice que task aquí en Post también le dicen Bueno aquí no en Post le colocan Api operation y le dicen summary esto crea una tarea refrescan y esto crea una tarea entonces ahí pueden

ir generando esa documentación también tiene otras secciones si ustedes abren Aquí también tiene una sección llamada response que es para saber qué es lo que va a retornar cuando es un código 200 entonces aquí pueden decirle cuando se hace una petición get lo que van a

retornar va a ser un Api response con el código 200 cuando el código sea 200 va a ser retornar todas las tareas entonces refrescamos y van a ver que dice 200 retornar todas las tareas pero también

puede que retornemos un estado 403 que diga forbiden refresco y también puede deir 403 forbid entonces de esta forma nosotros vamos documentando también simplemente con código typescript y es muy fácil hacerlo Al tener ya

decoradores que van a transformar todo eso por nosotros los dto también algo muy interesante es que también se separan por aquí Entonces nosotros podemos ver el contenido de cada uno de estos claro de momento no existen están vacíos pero ya luego Ustedes pueden ir

creándolo y eso también se va a ver reflejado aquí listo entonces Esta es una forma bastante simple de empar con swager dentro de nes y Yes hasta el momento ya hemos generado varias porciones de código dentro de nes

Pero algo muy común que ustedes van a encontrarse al desarrollar aplicaciones backend y luego intentar comunicarlas con frontend es algo como esto voy a venir en el navegador y voy a abrir una pestaña nueva van a escribir en la barra

de direcciones about 2. Blank escríbanlo bien de nuevo about 2. Blank y aquí van a llegar una página en blanco y al ser un dominio que no es el local Host 3000 es decir est es como otra aplicación si yo hago una petición fetch es decir yo

escribo aquí fetch y y le digo por ejemplo vamos a colocar entre signos de paréntesis le digo voy a hacer una petición http local Host 3000 slapi sl

users o users tan Solamente le digo envío me aparece un error en el navegador me dice no Access control la origin heer Express resource básicamente me dice tengo ha sido bloqueado gracias a corse corse es un tema bastante común

en aplicaciones frontend y básicamente indica que el servidor en el cual estoy que por ejemplo es un servidor en blanco es decir no es el mismo dominio local Host 3000 pues no tiene he permitido pedir datos al local hoster meil porque

no son en la misma dirección de hecho esta porción de código que que escrito acá lo puedo copiar Y si yo vengo en mi sitio por ejemplo en fastweb.com como mi sitio tiene El dominio fastweb.com y yo pido desde aquí lanzo un error de course

Por qué Porque el error lo está lanzando el navegador el navegador no te deja pedir desde una página distinta al servidor Entonces el servidor tiene que autorizar esta petición entonces para poder hacerlo eh vamos a venir aquí en

en nes y para solucionar error de Core simplemente venimos en main.ts y ya tiene una document ación también para eso Entonces eh la forma de llamarlo es con un método que se le añade app entonces simplemente se le coloca por

aquí app enable course y aquí ya está con este simple función pues cualquier página va a poder pedirlo de nuevo Yo estoy aquí en en Fast web si yo llamo el fetch pueden ver que y me retornó algo

ya no me dio el error de corse Okay incluso hasta podemos eh convertirlo luego a un dato le digo conviértelo a un esta respuesta vamos a escribir res lo convierto en r. json vamos a enviarlo eh

de nuevo esto sería d Data consol log Data y bueno allí veo la respuesta del backing y pueden ver que ahí está la respuesta del backend Okay entonces este

es código de frontend que nosotros podemos llamar pero bueno esto ya no está siendo bloqueado ya que el backend lo tenemos ahí entonces de esta forma nosotros ya podríamos empezar a también solucionar el error de cor Bueno vamos a ver entonces voy a venir aquí en la

consola y vamos a hacer lo siguiente con esto si ustedes quieren validar por ejemplo el El dominio del cual quieren acceder pueden colocar también un origin el orig es una un Stream d ustedes pueden decirle tan solo El dominio tal

tiene permitido en este caso yo le digo Tan solo la web http vamos a colocarle https fastweb.com puede acceder por ejemplo Entonces yo ejecuto esto y si yo vengo aquí en la aplicación y yo vengo

en en el la página en blanco Recuerden que yo tenía una página en blanco llamada about Blank Y si yo abro aquí la consola y pego esa porción de código si recuerdan esa porción de código de fets me da el error de cor porque no es Fast

web me dice no no esta página no es Fast web Entonces yo vengo en Fast web y aquí sí le digo ha esa petición Y ahí sí me revuelve algo no me da el error de cor Okay entonces de esta forma También nosotros podemos limitar qué dominio tiene permitido poder pedir cosas a

nuestro backend cuando hablamos de aplicaciones frontend entonces de momento voy a dejarlo que todo el mundo pueda pedirlo y ya estaría en esta última sección les voy a enseñar ahora cómo desplegar esta

aplicación de nes que en realidad no tiene nada nuevo pero ustedes van a querer aprender también cómo desplegarlo y para poder hacerlo vamos a utilizar un servicio de la nube que se llama rway es un servicio bastante sencillo de

utilizar y como como Tenemos aquí una base de datos de postest también el servicio nos permite desplegar postest eh si ustedes nunca han utilizado redway van a poder obtener un enlace gratuito que les dejo en la descripción donde van a poder obtener $ gratis pero si ya lo

han utilizado Probablemente van a tener que pagar el consumo en realidad son centavos de dólar Así que tampoco no consume mucho y bueno Apenas termine el ejemplo pues pueden darle de baja o

eliminar el proyecto para que no consuma nada más okay Así que solamente son centavos de dólar los que van a gastar vamos a cerrar aquí con Close y el primer paso para desplegar un proyecto de next es poder saber si y se compila

todo correctamente con el comando mpn r build ustedes pueden comprobar esto eso lo que va a hacer es construir toda la aplicación convertir todo el código de typescript a producción y listo con esto ya está Entonces lo siguiente sería

subirlo un repositorio en mi caso yo tengo aquí mi repositorio que es Fast web voy a crear un nuevo repo le digo New repository o aquí con el botón New y le voy a llamar a todo este repo nes

vamos a colocarle course o nes course Okay voy a colocarlo en público voy a darle en crear repositorio y aquí es donde voy a subir todo el código que vamos a el día de hoy vamos a darle git Remote that origin y ahora sí le digo

git status y aquí está todas las modificaciones que hemos hecho Okay el inb también lo voy a dejar allí aunque no debería el inb por lo general es un una variable que debería ir privada Okay pero allí lo dejo por si acaso ustedes quieran modificarlo Entonces vamos a

colocarle git ad punto para subirlo todos los cambios git commit y voy a colocarle First commit to deploy por ejemplo Ya está entonces ahí son todos los cambios le digo git Push origin

Master en mi caso tengo la rama Master aquí ustedes pueden que lo tengan como rama Main si lo tienen en Main allí escriben Main en mi caso dice Master como pueden ver que dice Master listo y ya está entonces yo refresco aquí la

página y veo el código Ahí está todo el código en ns ya subido a github Entonces lo siguiente sería entrar en railway railway vamos a venir en railway.app y aquí nosotros vamos a entrar en nuestra cuenta si ustedes no tienen una cuenta

les dejo enlace en la descripción si ya tienen simplemente logé cuando entren en su cuenta van a ver una pantalla como esta al entrar en esta página de su cuenta van a venir en este botón donde dice New project y aquí ustedes van a escoger esta opción que dice desplegar

desde github repo lo que van a hacer aquí es escoger en esta sección que dice configurar github app y aquí les va a pedir acceso en mi caso yo tengo el proyecto desplegado en mi cuenta Fast web aquí una vez dentro de mi cuenta voy a venir y voy a buscar el repositorio

que se llama nestjs yion cse Ahí está y le voy a dar en autorizar en guardar listo Esto me va a redireccionar la misma plantilla le doy deploy from github repo y le digo quiero desplegar n scores Ahí está y le voy a dar en

desplegar ahora Esto va a dar error pero igual no hay ningún problema nosotros podemos ir Arreglándolo entonces lo que necesitamos para desplegar no tan solamente es nes porque Recuerden que nosotros necesitamos una eh una base de datos entonces el tema es que aquí para

poder leer esa base de datos necesitamos llamarlo y de hecho ahora sí a lo que me he dado cuenta es que él va a leer el in que tengo aquí entonces voy a ignorarlo le digo Ignora el archivo inb vamos a decirle Ignora el archivo yb Okay Vamos

a darle en git pun git comit le digo vas a ignorar el archivo yv inv ignorado y voy a hacer git Push origin Master la razón decer esto es porque el el repositorio de la nube o mejor dicho el

la plataforma de la nube va a leer la variable de entorno Entonces yo no quiero que haga eso porque en este caso quiero que lo lea desde su sección de variables pero Pero bueno muy aparte del backend ns también necesito una base de datos entonces venimos aquí donde dice

eh New y dentro de New vamos a buscar database vamos aer post sql Esto va a generar una base de datos de postres rápidamente y lo genial es que como postr por lo general tiene la misma eh

sintaxis En cuanto a la conexión si yo vengo dentro del recuadro de postes en variables vamos a ver esta variable que dice database URL Y si yo lo veo pueden ver que dice po sql el usuario la contraseña es decir Aquí acaba la

contraseña Aquí está El dominio el puerto y el nombre de la base de datos entonces simplemente tengo que Añadir esta variable hacerlo es muy simple dentro de n yo puedo decir en variables como lo único que necesita es una nueva variable llamada database URL le digo

necesita databas URL esto viene de postre le doy en guardar le doy en desplegar y listo con esto ya podríamos desplegar nes y es o todo el proyecto de nes porque no necesitamos nada más Entonces vamos a ver si realmente

tenemos un error o algo allí aquí se completó correctamente el deploy inicial vamos a esperar que aquí también se despliegue el in Okay aquí se terminó de desplegar

Entonces si nosotros queremos ver un dominio pues vamos a venir en settings y vamos a venir en esta sección que dice generate domain esta opción tan solamente lo van a ver si ustedes han colocado una tarjeta allí eh de crédito

o débito dentro de su cuenta Okay Al momento de que se han registrado voy a darle en generate domain okay Y ya genera un dominio este dominio Por lo general al momento que lo abrimos tiende a fallar un poco porque recién es un dns nuevo Pero bueno si esperamos unos

cuantos segundos Esto va a actualizarse y sé que se tiene que actualizar porque aquí cuando hemos visto ns pueden ver que no nos ha dado un recuadro en rojo sino nos ha dado un recuadro en verde significa que todo ha ido bien De hecho

si doy un click aquí vamos a ver los típicos mensajes que dan njs cuando ya se está ejecutando en producción Entonces vamos a refrescar esto de aquí y ya debería funcionar vamos a esperar un segundo Okay aquí el proyecto se ha ejecutado pero una configuración que me

faltó es que al momento de ejecutar nes en src main.ts si analizamos un poco él está leyendo el puerto 3000 y aquí en realidad nosotros debemos darle el puerto que le de rway entonces aquí le

digo va a usarle el process n. por que es una variable de entorno que da rway o cualquier otro entorno de la nube o en caso contrario no te den nada utiliza el 3000 Ahora sí voy a hacer otro guardado

git status ahí está el archivo que modifiqué git punto para seleccionarlo git comit le digo Puerto actualizado y un git Push origin Master Okay y esto lo que va a hacer nuevamente subir el

código y Al subir en github Pues si venimos aquí nuevamente se va a desplegar otro otro Push y hay que esperar unos segundos a que se despliegue nuevamente aquí ya se ha desplegado vamos a dar un click un control click y bueno esto ya debería

desplegar el proyecto en este caso en El dominio adecuado vamos a ver si aquí ya lo leo Okay vamos a ver dice nestar vamos a ver si lo sirve Okay si leemos un poco allí tenemos el mensaje típico

que da nes cuando está en producción que es este status code pero vamos a pedir por ejemplo el sl users que en este caso viene la base de datos Okay y me dio un error vamos a ver por qué y bueno si bien está pidiendo la base de datos la

base de datos en este caso está vacía no tiene las tablas es por eso que nosotros cuando desplegamos tenemos que generarlos entonces una forma de generarlo en de momento Recuerden que aquí tenemos la variable de entorno de la base de datos vamos a venir en

variable Aquí está en vr voy a copiarlo y una forma sencilla al menos de generar la base de datos sería colocando aquí en lugar de la base de datos de desarrollo voy a copiar esto voy a decir database

URL igual a la dirección que me está dando railway Okay voy a comentar esto y ahora sí desde la consola voy a decir mpx prisma y vamos a colocarle aquí un Push db qué va a ser el Push db que va a

leer bueno db Push Perdón db Push el db Push lo que va a hacer es que va a leer nuestra configuración de prisma y va a generarlo pero lo va a generar en la URL de de redway es decir la base de datos de producción eso es lo que va a haer es

que si yo entro aquí en postgress en Data vamos a ver que aquí va a cargar la tabla bueno la única tabla que tengo que es usuario y ahora sí si refresco aquí los datos pueden ver que me vuelve un arreglo vacío ahora si yo quiero crear

algo pues vamos a tener que utilizar esta URL para enviar una petición post Entonces le digo voy a enviar en Thunder client una petición post a esta dirección y aquí le envío los mismos datos de siempre envío allí cargo un poquito y esos son los datos que ya creo

en la base de datos de redway de hecho si yo vengo aquí en la aplicación y refresco Ahí están los datos que me está devolviendo la base de datos Y si yo voy a enr en usuario Ahí están los datos guardados obviamente esta misma

dirección que obtenemos aquí en vibs en database Ur ustedes pueden usarlo de su cliente postre sql favorito pg admin eh div viiv eh jd sql o lo que utilicen y va a obtener acceso a lo mismo datos ya

que la base de datos Está alojada aquí pero estas están solamente una interfaz y listo esa es básicamente la forma en la que podemos desplegar un proyecto de nes como pueden ver es un framework bastante sencillo no requiere tantas complicaciones tiene algunos términos

que sí pueden confundir pero pueden ver que ha sido bastante fácil llegar hasta esta parte eh siempre y cuando ustedes ya hayan practicado código de javascript código de typescript que son las bases para entender cualquier framework En fin espero que el video les haya servido Si

tienen una duda adicional o una petición de qué tipo de aplicación quiere que creen utilizando este framework pueden dejar los comentarios y nos vemos en un siguiente curso eso ha sido todo por el video del día de hoy si tienes dudas

puedes dejarla en los comentarios o en la descripción dejo un enlace para que te puedas unir a la comunidad de discord en donde encontrarás a otros desarrolladores o si en caso el enlace está caído puedes ir a fastd discord

para acceder más rápidamente dejo mi Twitter donde típicamente comparto algunos recursos interesant de desarrollo y programación en general mi Instagram donde comparto algunas noticias cortas todos los días el tiktok donde comparto videos cortos e

informativos y mi canal principal en donde comparto opiniones y noticias de tendencias nuevas además también dejo mi web en donde puedes reservar asesorías personalizadas Gracias por ver y nos