Nestjs, Tu primer aplicacion Backend desde cero - YouTube

https://www.youtube.com/watch?v=7fOjuCGE\_jk

Vamos a empezar lo primero que voy a hacer es crear un proyecto desde cero utilizando nestjs ahora como yo sé que este es su primer proyecto lo que voy a hacer es también mostrarles cómo instalarlo

Entonces lo primero es que debemos tener instalado Node js para poder instalar nes js ya que nes es un framework de Node Entonces lo primero es confirmar si tenemos la versión de node adecuada por ejemplo en

(05:57) mi caso voy a aquí limpiar la consola y el primer Comando que voy a ejecutar es el comando

**node -- version**

y bueno pueden ver que en mi caso tengo la versión 18 pero con cualquier versión superior la 16 que ya es como digamos el estándar actual pues pueden hacer este tipo de proyectos sin ningún problema ahora una vez tenemos nuestra versión de Note instalada vamos a instalar nestjs con el siguiente Comando simplemente vamos a decir

**npm install @nestjs/cli -g**

este es el generador de proyectos o programa de consola de njs gracias a este Comando vamos a poder crear proyectos crear código y ejecutar el código entonces adicionalmente a esto vamos a Añadir la opción gu G que es para instalarlo globalmente ahora este proceso de instalación del módulo como Quizás es la primera vez que lo vamos a instalar o quizás esto eh puede que nuestra conexión no sea tan buena va a tomar algunos minutos Entonces vamos a esperar que este proceso termine y luego continuamos Okay aquí ya se terminó de instalar este clip ahora utilizándolo vamos a crear nuestro proyecto de neso Entonces vamos a movernos primero hacia el escritorio para eso simplemente voy a

(06:58) limpiar aquí la terminal o consola y vamos a escribir El Comando CD desktop y a partir de allí voy a crear un proyecto utilizando El Comando:

**nest new “nombre\_del\_proyecto”**

el nombre del proyecto que queramos en mi caso voy a llamar el proyecto nest app y de esta forma si doy un enter lo primero que va a hacer es crear una carpeta vamos a verlo ahí está y luego también me va a preguntar Con qué administrador de paquetes

Quiero instalar paquetes o digamos hacer disponible módulos extras en nes y bueno en mi caso yo utilizo mucho npm

descripción pero eh si no conocen nada de otros administradores utilicen npn y con eso estaría entonces voy a darle enter y con Esto va a empezar a instalarse ahora este proceso de instalación también va a tomar algunos minutos porque Bueno uno en pian es un poco lento y otro que también vamos a tener que instalar muchos módulos porque nes ya trae muchos módulos integrados Entonces vamos a esperar que instale y luego continuamos Okay aquí ya se terminado de crear el proyecto así que ya podemos cerrar la terminal y vamos a arrastrar nuestra carpeta dentro de nuestro editor de código ahora a partir

(07:58) de aquí lo primero que vamos a hacer es echarle un ojo a la estructura de proyectos que nos ha creado Lo principal es entender estos archivos que se llaman nes Y esta carpeta src Pero bueno para darles un comentario rápido todo lo que está fuera desde src o fuera de la carpeta src son archivos de configuración aquí hay múltiples herramientas por ejemplo sl slink es para poder tener como una forma de escribir código como una serie de recomendaciones para tener un código ordenado luego también tenemos git ignore que es para que podamos ignorar

(08:27) archivos y no subirlos a github pri que es para poder tener algunas reglas de formateado de código eh nesc que esto es para poder configurar el framework en este caso nesc no es muy importante para este ejemplo porque no vamos a estar modificándolo pero a futuro si ustedes quieren alterar algo del framework estarían utilizando este archivo aparte de esto También tenemos un archivo package lock y package json esto está relacionado con el administrador de paquetes si utilizan jarn o pnpm por ejemplo esto no lo van a ver pero al final lo que tenemos con estos archivos de aquí es una lista de las dependencias

(08:58) que utilizan es ahora si prestamos un poco de atención a lo que tenemos aquí para empezar tenemos una enorme cantidad de scripts no es importante de momento que los conozcan todos los más importantes son tan solo dos eh uno es el comando St Death y el otro es el comando build El Comando Start Death lo que va a hacer es que va a crear un servidor de desarrollo es decir va a ejecutar el proyecto y cada vez que nosotros escribimos código va a reiniciar el código o va a mostrar los cambios inmediatamente como notem en Note luego también tenemos build que es

(09:28) para poder convertir el código Recuerden que aquí estamos utilizando typescript y otras herramientas Entonces cuando hagamos build él va a convertir todo el código a javascript tan solamente y bueno También tenemos otros comandos relacionados con link testing y demás Pero bueno eso no es necesario para elaborar Este ejemplo el día de hoy y luego también tenemos dependencias que es para poder tener nuestros módulos principales y dep dependencies que es para poder tener módulos relacionados en desarrollo tan solamente y bueno Esto es así de simple ahora aparte de esto

(09:58) También tenemos la carpeta o perdón con el archivo ts config el archivo ts config es para que nosotros podamos alterar e la versión de typescript bueno en mi caso pueden ver que ya tien la versión es 2017 esto significa que puedo estar utilizando singa wade y todas esas características nuevas Pero si ustedes quieren utilizar una característica más moderna aún pueden modificarlo simplemente aquí ahora todo esto es asunto de typescript y no de njs ahora en cuanto al src que es la carpeta del código fuente Aquí es donde nosotros vamos a colocar nuestro código eso lo

(10:27) vamos a modificar luego pero luego también tenemos esta carpeta test que es para poder tener ent test y bueno si queremos ejecutar esto vamos a abrir la consola aquí se abre la terminal de en este caso visual Studio code y Aquí vamos a escribir simplemente El Comando mpm Run Start 2.

(10:45) Def si doy un enter esto lo que va a hacer es ejecutar nuestro proyecto de backend y lo va a ejecutar en el puerto 3000 De hecho aquí no lo menciona pero pueden ver que hay como una salida de ejecución entonces si yo abro mi navegador y escribo por ejemplo el local Host 3000 pueden ver que allí ve un Hello world listo si yo busco otra cosa por ejemplo escribo Slash cualquier cosa enter pueden ver que me dice status code 404 o no se encuentra etcétera eso significa que también tiene una especie de manejador de errores o manejador de páginas no encontradas Pero bueno con esto ya

(11:16) tenemos una idea básica que ya nuestro proyecto está funcionando ahora muy aparte de esto también vamos a ejecutar vamos a cancelar aquí la consola También tenemos el comando para poder convertir el código que es el que les mencioné voy a limpiarlo aquí y el otro Comando es npn room build si doy un enter esto lo que va a hacer es convertir nuestro proyecto para producción es decir minificacion en producción ejecutarías El Comando mpn Start si doy un enter él va a ejecutar el proyecto en este caso ya no con un watcher sino que simplemente va a mantener el proyecto en ejecución y

(11:52) bueno Esto es lo mismo si yo vengo aquí en en la carpeta dis y escribo node dis Slash y escribo o ejecuto main.js si un enter pueden ver que es la misma salida pero bueno por lo general El Comando en N Star es el que van a ejecutar en producción ahora aparte de esto también otra cosa muy importante es cómo funciona todo esto y bueno en la carpeta src nosotros tenemos archivos como main.

(12:21) ts y pueden ver que hay tres archivos que empiezan con app Bueno de hecho cuatro que empiezan con app y bueno Esto significa que en ns nosotros tenemos una forma sencilla de poder dividir nuestras múltiples rutas nest tiene un concepto que se llama módulos los módulos lo que hacen es agrupar distintas rutas con sus funcionalidades relacionadas Por ejemplo si yo tengo eh usuari que quiero registrar productos que quiero listar o categorías que quiero crear o conceptos similares lo que podemos hacer es que vamos a poder dividir en módulos cada una de estas rutas por ejemplo en los usuarios van a necesitar que se creen Listen eliminen actualicen Entonces eso puede ser un

(12:53) módulo luego también con los productos tambén quiero quizás que se actualicen creen eliminen y demás también va a ser otro módulo y así entonces para unir todos esos módulos él utiliza un archivo llamado main.ts Ese es el archivo que arranca todo el proyecto Entonces lo más importante entender aquí es que cuando ejecutamos main.

(13:10) ts él ejecuta una función dentro de esta función él llama un módulo llamado nest Factory que es como una forma de decir voy a cargar un módulo principal y luego él lo que hace es llamar otro archivo que se llama app module este app module lo que es es básicamente digamos nuestro módulo inicial y bueno en una aplicación de nes nosotros podríamos tener múltiples módulos Pero al final siempre hay un módulo que carga el resto eh si han digamos utilizado algún framework de frontend esto lo pueden comparar con los componentes por ejemplo en el frontend siempre hay un componente que contiene

(13:40) al resto aquí en el backend también Es algo similar con nes de hecho nes justamente se inspira en el framework de frontend angular es por eso que pueden ver esa similitud y bueno lo único que hace Es como decir carga todos los módulos o carga un módulo y este módulo adentro puede tener más módulos y al final inícialo en el puerto 3,000 ahora si nosotros por ejemplo queremos eh ver qué de qué se trata ese módulo Vamos a entrar aquí vamos a verir en modul que es el que está llamando pueden ver que para empezar tenemos una clase encima de la clase hay algo que se llama un

(14:10) decorador que es solamente digamos funciones que extienden esta clase y luego tenemos importaciones ahora la parte importante es este módulo o ar modul en nosotros tenemos este decorador para poder convertir una clase y decir Okay este va a agrupar distintas funcionalidades en uno solo Este es un módulo y Bueno aquí lo que tiene es justamente tres una llado imports controllers y providers los imports es para poder Añadir funcionalidad extra a este módulo Por ejemplo si Quiero conectar una base de datos eh los controles es para poder tener por aparte

(14:40) en un archivo todas las rutas que puede tener este módulo por ejemplo eh rutas get post delete etcétera pueden estar aquí en un archivo por aparte y luego providers que es para poder decirle Okay también puedo tener funciones que se encargan de comunicarse con la base de datos y que puede ser reutilizada en todas partes Entonces yo puedo tener una clase que el resto de digamos partes de mi módulo puede utilizar y ya está ahora aparte de esto vamos a entrar en controller y aquí nosotros tenemos una clase que lo que hace es justamente

(15:10) poder crear rutas los controladores en N se crean de esta forma primero creamos una clase eh simplemente creamos una clase con un nombre adentro llevamos un constructor Si queremos esto es opcional pero es muy importante que cuando nosotros vamos a definir rutas vamos a colocarle un nombre a una función luego vamos a colocarle un decorador y luego lo que queremos que retorne eso al cliente de momento quizás va a sonar muy confuso es por eso que lo que vamos a hacer como primer paso es Eliminar todo

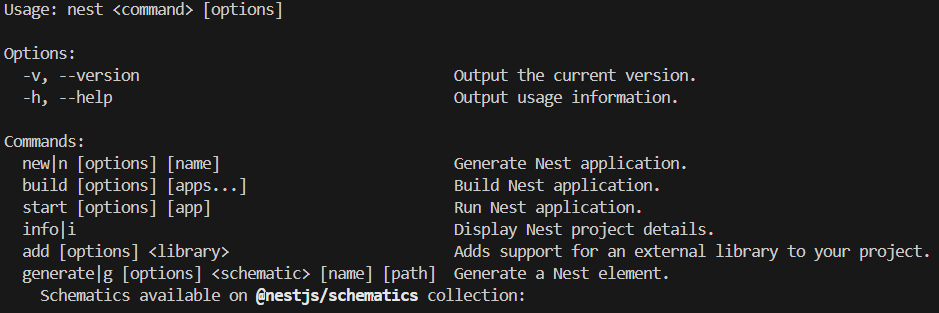
(15:36) esto y vamos a empezar desde cero Entonces como no no sabemos cómo funciona el app controller Spec vamos a quitarlo este es para testing no es importante para este ejemplo el app controller dias también lo vamos a eliminar el app modu sí lo vamos a dejar y solamente el app service lo vamos a quitar Entonces nos vamos a quedar con el main.

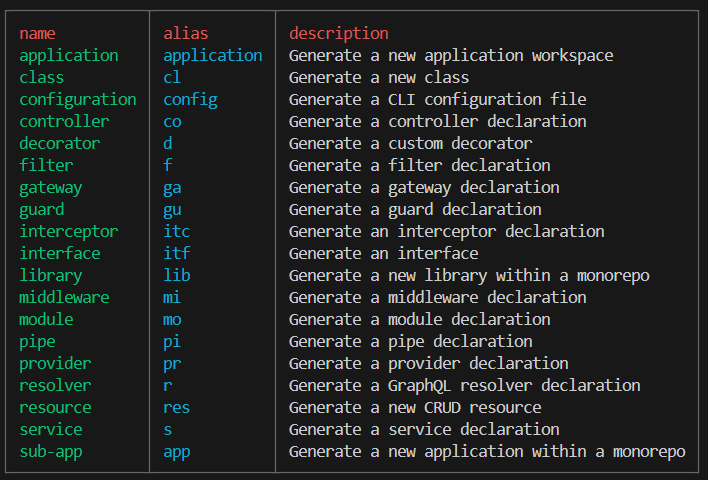
(15:59) ts y el ad modu ahora este ad module estaba importando otros archivos que acabo de eliminar entonces también lo vamos a quitar estos dos líneas lo vamos a quitar y aquí también vamos a quitarlo de sus importaciones entonces de momento yo tengo tan solo un main.ts que llama este ad modle pero está vacío no tiene no no tiene ninguna ruta no tiene clases no tiene nada más y bueno es lo que vamos a hacer en la siguiente sección vamos a empezar a crear nuestras primeras eh rutas a partir de crear nuestro primer módulo con nuestro proyecto creado ahora lo que vamos a hacer es crear nuestro primer módulo para poder hacerlo nosotros podemos crear aquí de nuevo nuestros archivos que se llamen por ejemplo app pun modle

(16:30) app controller app service y demás ahora nosotros por lo general podemos hacerlo pero en realidad en la práctica muchos de esos radores utilizan lo que se conoce como un comando para generar todos estos archivos Entonces vamos a abrir la terminal nuevamente y la terminal de visual Studio code ya nos ubica en nuestro proyecto Así que vamos a limpiar y nes El Comando que habíamos utilizado para crear el proyecto también nos da comandos para poder crear archivos de esa forma No tenemos que crearlos a mano

Entonces por ejemplo si nosotros escribimos simplemente

nest y damos un enter lo que vamos a ver aquí como salida es una especie de tabla y en esa tabla justamente están listados todos los comandos que yo puedo ejecutar en es por ejemplo en nuestro caso si queremos crear un módulo aquí tenemos el El Comando module y Bueno aquí también te da una descripción de cómo utilizarlos





y Bueno aquí por ejemplo hemos utilizado New también hemos visto El Comando build St y demás Pero justamente aquí me dice puedes utilizar el comando generate y luego las opciones que ves aquí listadas aquí debajo entonces en mi caso Yo quiero generar un módulo Entonces vamos a utilizar n gener mod vamos a limpiar entonces voy a

escribir:

**nest generate module “nombre\_modulo”**

y con esto luego vamos a tener que colocarle un nombre entonces en mi caso voy a crear un módulo que se va a llamar tareas damos un enter y aquí lo que va a hacer es que uno va a crear una carpeta llamada tareas y adentro justamente va a crear una clase llamada task module.

(17:48) ts si yo lo abro pueden ver que justamente es tan solo una clase y tiene un decorador module que viene desde n no hay más entonces es un módulo vacío de hecho esto ya lo habíamos visto en mod si recuerdan ad modle es exactamente lo mismo solamente que este ad modu es el módulo raíz digamos Ese es el módulo que nuestra aplicación Realmente está cargando Pero dentro de admil nosotros podemos cargar más módulos entonces si nosotros lo viéramos aquí por ejemplo este admil no es para que nosotros lo modifiquemos sino que vamos a estar utilizando este task module para poder

hacerlo y Bueno aquí Qué vamos a hacer en el task module bueno nosotros vamos a crear Rutas para que una aplicación frontend luego pueda crear tareas eliminar tareas listar tareas entonces aquí lo que nosotros vamos a necesitar también es un controlador los controladores son para poder crear rutas entonces quizás dicen okay tenemos que crear entonces aquí en module o como vieron en aquí en App modu vamos a tener que crear una sección llamada controllers y así como estaba antes también vamos a tener que crear un arreglo e importarlo Y sí vamos a tener que hacer eso pero no lo tenemos que

hacer manualmente de nuevo podemos abrir la consola y así como teníamos ad modle Ahora puedo generar

**nest generate controller “nombre\_controller”**

nest generate y también puedo crear una un controlador utilizando la palabra controller escríbanlo bien con dos l Y luego el nombre que quieran escribo Tas por ejemplo ahora como es muy común que en producción nosotros vamos a crear controladores y luego vamos a querer testear losos eh se van a crear dos archivos pero en nuestro caso como no estamos viendo testing vamos a Añadir una opción más que se llama gu gu no gu gu Spec este Comando lo que va a hacer

es como decirle no quiero el archivo de testing solo el controlador entonces voy a dar un enter y esto lo que va a hacer es que si esperamos un segundo pueden ver que ha aparecido un nuevo archivo de typescript que se llama task controller ts De hecho si abro este archivo de aquí van a ver que es exactamente lo mismo es una clase con un decorador controller y con un nombre que dice task que es como decir cuando tú solicites la ruta Slash task desde el frontend nosotros vamos a procesarlo con los métodos de esta clase

(19:45) es así de simple y listo ahora aparte de esto también luego Quizás vamos a querer comunicarnos con una base de datos o Quizás vamos a querer comunicarnos con un backen externo y reutilizar esas funciones entonces lo que nos dicen es Es que podemos crear un servicio de igual manera no tenemos que hacerlo manualmente Entonces vamos a limpiar aquí le digo nest generate y service De hecho no tengo que escribir generate completamente también podemos escribir simplemente G y luego también podríamos escribir service que es el nombre y aquí el nombre del del servicio que queramos

(20:16) que se llama task ahora si tienen mucho problema entendiendo Qué es un servicio simplemente piénsenlo como un archivo que va contener por ejemplo todas nuestras en nuestros métodos que podemos llamar desde cualquier parte de la aplicación es como una carpeta utils Por así decirlo y bueno si venimos aquí en Tas service.

(20:39) ts pueden ver que ahí está la clase y también aquí tiene un decorador que se llama injectable y justamente se llama injectable porque nosotros podemos inyectar esta clase en otras partes de la aplicación podemos reutilizarla y bueno como no le coloqué el gu gu no Spec pueden ver que también ha creado un archivo task service Spec Entonces esto no es necesario lo podemos quitar podemos darle un clic derecho eliminar y ya está ahora si nosotros analizamos un poco el ad module pueden ver que aquí está el task module Y si analizamos el task mod.

(21:02) ts pueden ver que ya le colocó por mí el controllers y el providers y aquí está justamente task controller que hemos creado y el Tas service que también hemos creado Entonces no tenidos que hacerlo manualmente y Listo ya tenemos creado entonces nuestros módulos si nosotros ejecutamos esto por ejemplo yo le digo ahora nuevamente en mpm room St nuevamente va a arrancar la aplicación no debería tener ningún error Ahí está y luego si entramos en el local Host 3000 pueden ver que me dice 404 porque no hay nada si entro por ejemplo en el Slash Tas y doy un enter me dice 404 pero lo que podemos hacer de momento

(21:36) es que si yo vengo aquí en el task controller y aquí por ejemplo en esta clase como les digo Aquí pueden crear rutas lo que pueden hacer es hacer esto de aquí pueden crear una función o en este caso como estamos en una clase sería un método y Aquí vamos a llamarlo por ejemplo Hello World ahora este Hello World como es un método vamos a tener que escribirlo de esta forma y para decirle que cuando llamen a través de un método get a Slash task quiero que ejecute lo de Aquí vamos a colocar un decorador Entonces desde nes nosotros podemos importar algo que se llama un

(22:04) decorador llamado get entonces aquí simplemente le digo arriba vas a colocar un arget y ya está esto es como decir cuando hag una petición get a sl task vamos a ejecutar lo que esté aquí y lo que voy a ejecutar Es simplemente retornar un texto que diga Hello World entonces si yo guardo esto aquí el servidor ya se ha reiniciado porque hemos ejecutado con el comando stard Death Y si yo vengo aquí en el frontend y refresco con el lash Tas pueden ver que me dice Hello World nuevamente y claro hemos hecho hecho mucho asunto para poder ver lo mismo pero realmente estamos creando nuestro primer módulo y

(22:33) es por eso que aquí se ve muy similar al inicio pero lo hemos creado desde cero y bueno continuamos en la siguiente sección con otra parte muy importante en nes que se le conoce como las entidades en la sección anterior hemos aprendido Cómo crear un proyecto y un módulo ahora vamos a aprender Cómo crear algo que se llama una entidad en nes nosotros podemos crear lo que se conocen como entidades que nos sirven para poder detallar Cómo lucen por ejemplo un dato por ejemplo en este en este caso nosotros tenemos tareas las tareas pueden tener ids títulos y demás entonces todo eso nosotros podemos

(23:04) detallarlo en una clase de esa forma cuando lo utilicemos en typescript podemos obtener autocompletado y corrección de errores gracias a esos tipos de datos Entonces por ejemplo aquí en Tas yo puedo crear un archivo que se llame task entity Tas ahora este archivo no es realmente requerido de hecho muchos archivos de aquí no son enteramente requeridos pero sí son recomendados Porque muchos proyectos de ns lo utilizan ahora en task entity lo que nosotros podemos hacer es crear una clase que se llame por ejemplo tarea y aquí nosotros podemos detallar por

(23:32) ejemplo qué datos quiero supongamos que una tarea va a tener un ID un título una descripción también va a tener un estatus y aquí nosotros podemos decirle Qué tipo de datos van a hacer por ejemplo le digo este ID va a ser un Stream Este título también va a ser un Stream la descripción también e incluso el estatus solamente que el asunto con el estatus es que probablemente nosotros no vamos a querer que sea e cualquier estado sino algunos que nosotros hemos definido entonces para eso en lugar de estar colocando un Stream yo puedo utilizar un enum entonces aquí puedo

(24:01) decirle Por ejemplo voy a crear un enum que se llame Tas status y aquí puedo detallar los distintos métodos por ejemplo puedo decir Quiero un en un para declarar cuando una tarea esté abierta Es como decir una tarea pendiente De hecho también podría haberla llamado pendiente en progress y Don't hecho Aquí voy a cambiarlo voy a decir pending por ejemplo Okay entonces tengo tres estados tengo una tarea llamada Bueno puede tener un estado pendiente puede tener un estado en progress o Don entonces en lugar de decirle Okay el estado va a ser por ejemplo manualmente le digo hacer un

(24:30) pending o impr progress y demás le digo esto está basado en este task en un y bueno Esto es opcional también no es requerido bueno estás status y bueno es tan solo para modo de tener digamos las funcionalidades de typescript o utilizar estas funcionalidades en nuestro código y bueno como esto de Aquí vamos a utilizarlo luego también lo vamos a exportar y ya estaría ahora para darle simplemente una idea de cómo se utiliza eso como en este caso tenemos una clase luego tenemos que instanciar entonces puede decir New tas Esto me va a

(24:59) devolver una tarea Okay vamos a guardar una constante esta tarea y esta tarea si yo coloco un punto pueden ver que tiene aquí las cuatro propiedades tiene un ID un status un título y una descripción entonces a través de esta forma es como vamos a utilizar esta entidad ahora a modo de comentario una entidad en la práctica en digamos en proyectos reales se utiliza para que nosotros podemos eh claro definir En typescript qué datos tiene pero adicionalmente también a través de esta clase nosotros podríamos generar la tabla de nuestra base de datos eso no está relacionado con es

(25:29) sino con módulos que se conocen como orms los orms lo que hacen es que leen estas clases y los transforman en Tablas de sql y bueno se puede hacer esto porque si se dan cuenta como aquí tengo un ID que es un Stream esto también perfectamente podría ser un ID de una tabla que sea de tipo dato barsar lo mismo también si tengo un estatus por ejemplo y nosotros lo guardamos en una base de datos no sql esto podría ser también una especie de eh digamos objeto o podría ser también unas serie de arreglo de strings y demás Entonces al final nosotros podemos convertir estas clases en en este caso modelos de

(26:04) nuestra base de datos ahora eso de nuevo no está relacionado con nes enteramente sino con orms y nes tiene toda una documentación ahora muy aparte de generar la base de datos También sirve para poder generar otro tipo de esquemas Por ejemplo si utilizan graphql también pueden utilizarlo para poder generarlo o si utilizan por ejemplo algún módulo que necesite de leer una clase y transformarla a otro en otra herramienta gracias a que lo hemos definido aquí es fácil hacerlo pero bueno para darles un ejemplo en mi sitio que pueden encontrarlo en fastweb.com pueden buscar

(26:35) simplemente en nills y hace ya algunos unas semanas he creado dos ejemplos uno con graphql y otro con type urm ambos eh utilizan las entidades solamente que por ejemplo aquí se utiliz las entidades para generar esquemas de graphql y aquí se utilizan las entidades para generar las tablas yo Les recomiendo que continúen con este luego que es que utiliza type RM para que puedan ver más a fondo Cómo se utiliza estas entidades y bueno de esta forma forma es como ya tenemos definido Esto entonces la siguiente sección Vamos a continuar con

(27:02) los servicios con nuestra entidad definida ahora vamos a crear un servicio para poder hacerlo vamos a venir ahora en Tas service.ts y Aquí vamos a crear algunos métodos y bueno los servicios sirven para que nosotros podamos crear métodos que podemos reutilizar Por ejemplo si nosotros queremos crear una ipi de tareas vamos a necesitar listar tareas crear actualizar y eliminar entonces voy a crear métodos Como por ejemplo voy a llamar uno que se llame get all task que es como decir Quiero obtener todas las tareas y bueno no puede tener Cualquier nombre Entonces vamos a copiar esto de aquí Unas cuatro

(27:32) veces más o bueno tres veces más y también voy a crear unos métodos para poder crear una tarea también para actualizar una tarea update Tas y finalmente también voy a crear un método para poder eliminar una tarea entonces de esta forma tengo ahora cuatro métodos y a través de estos cuatro métodos yo puedo empezar a utilizarlo en otras partes y Bueno aquí viene el detalle para poder utilizar estos métodos yo tengo que inyectar esta clase en otro lado donde lo quieran utilizar Entonces vamos a venir en el task controller y aquí Y en lugar de mostrar simplemente un Hello World vamos a hacer lo

(28:01) siguiente cuando visiten a través del método get el Slash Tas nosotros no vamos a devolver un Hello World sino vamos a devolver una lista de tareas entonces para poder hacer eso primero tengo que comunicarme con este Tas service o poder llamar este método Entonces vamos a Añadir esta clase dentro del controlador y bueno para poder hacer eso aquí tenemos un constructor que esto es un método que tienen todas las clases y si se les hace difícil entender métodos e en este caso clases propiedades y demás yo les recomendaría que vayan al curso de

(28:31) javascript orientado objetos de hecho eso también lo tengo en mi sitio Bueno también está en YouTube pueden buscarlo como javascript orientado objetos eh Fast y va a encontrar el curso en donde se menciona todos esos conceptos básicos ahora en el constructor lo que vamos a hacer es justamente Añadir o lo lo que se le conoce como inyectar una clase entonces aquí para poder hacer eso primero tenemos que traer la clase que queremos inyectar Entonces le digo voy a traer y Aquí vamos a colocarle que desde

(28:55) la desde la misma carpeta vamos a llamar el T Service y desde aquí tan solo podemos importar una sola cosa que sería el Tas service y bueno una vez tenemos esa clase tenemos que instanciar la así como hacemos un New en este caso podemos hacerlo desde aquí utilizando simplemente eh un nombre de esta forma por ejemplo llamo Tas en plural task service y pueden ver que está en minúscula 2s pun Tas service y esto de aquí lo que va a hacer es que va a crear ahora una eh una especie de propiedad dentro de esta clase pero para poder hacer eso vamos a añadirle un private o

(29:26) un public ahora con esto si yo vengo aquí en lugar de colocarle un Hell World yo puedo decirle retorna y puedo decirle dis punto y pueden ver que aquí está el Tas service Okay entonces aquí está Obviamente el mismo método que es Hell World pero si escribo Tas service Y nuevamente escribo un punto un control espacio pueden ver que están los cuatro métodos que estaban definidos en la otra clase en este caso el que me importa es el get all task y si lo ejecuto como este método de aquí Me debería retornar

(29:51) algo de hecho vamos a verlo eh lo que yo voy a hacer Es que cuando me retorne algo esto yo voy a retornável que hace un segundo esto estaba de esta forma yo le estaba retornando al cliente un Stream ahora lo que yo hago con esta línea de código es decirle cuando eh este método te retorne algo Yo le voy a retornar eso al cliente ahora este get all task si vamos allí vamos a ver que no tiene nada Entonces de momento vamos a retornar un simple arreglo con un valor que diga id1 con un título que diga eh First task primer

(30:22) tarea y finalmente también con una descripción que diga algo Como por ejemplo eh sonas alguna tarea listo Esto es lo que voy a retornar Entonces el cliente guardo esto de aquí entonces cuando ejecuten get all task yo voy a retornar este arreglo de aquí listo vamos a guardarlo y también se puede formatear para que se vea mejor y en controllers también vamos a formatearlo Entonces cuando pidan el arac controllers o mejor dicho cuando pidan el get sl task voy a devolverles esto ahora esta función no es muy correcto que le llamemos Hell World vamos a llamarle eh get all task también y Bueno

(30:57) aquí puede tener el mismo nombre porque este método está relacionado con esta clase y este método le pertenece a la clase Tas service y bueno con esto Ahora si yo quiero ejecutarlo simplemente vamos a ir en nuestro consola pueden ver que no tengo errores y si vengo ahora en en local Host 3 Tas pueden ver que allí está en este caso la arreglo con un solo objeto y bueno además de esto Si nosotros queremos luego crear Pues también vamos a tener que hacer funcionar ahora también el create task entonces en el create task nosotros vamos a tener que también colocar aquí lógica pero para el creas nosotros vamos

(31:29) a tener que recibir datos Entonces cuando yo voy a crear una tarea yo voy a querer Añadir algo Dentro de este arreglo entonces como quiero añadirlo en lugar de tenerlo aquí lo que voy a hacer es seleccionar esto voy a pegarlo afuera es decir Al mismo nivel de los métodos y esta de aquí va a ser una propiedad de la clase es decir va a ser algo que la clase va a poseer Entonces en este caso voy a decirle Okay Esto va a ser un una constante que se va a llamar tareas y como igual va a ser un arreglo ahora para que esto sea una propiedad privada voy a colocarle esto tan solo se va a hacer desde la clase o a través de sus

(32:00) métodos y también puedo colocarle la entidad o puedo decir Okay esta digamos este arreglo de aquí si yo le coloco un cursor pueden ver que me dice okay esto es un arreglo y tiene ID título y descripción Pero mis tareas quizás pueden tener más datos entonces voy a utilizar la entidad que habíamos creado antes entonces voy a importar la entidad aquí le digo voy a importar eh De hecho aquí voy a importar task desde task entity y aquí lo que voy a hacer es simplemente colocarle dos puntos Tas y aquí como esto es un arreglo voy decir

(32:31) es un arreglo de tareas Entonces si nosotros vemos aquí pueden ver que me está dando un Sor rayador rojo porque aquí me dice el tipo number no es asignable un tipo stram y la razón es que justamente la tarea nosotros si lo vemos el ID Debería ser un stram no un número entonces lo que puedo hacer aquí es decir Okay este va a ser un número de esta forma ahora eso está bien para empezar pero si se dan cuenta aún me sigue dando error si yo coloco aquí me dice okay la propiedad status es requerida o falta en este objeto que estás añadiendo Y es porque aquí también

(33:02) yo tengo que añadirle un estatus y este estatus yo tengo que utilizar también los múltiples valores por ejemplo teníamos un valor Open bueno en mi caso era pending y con esto funcionaría Ahora el asunto con esto es que Eh Al momento de definir este status eh si venimos aquí en Tas entity nosotros teníamos justamente esta este nun para poder declararlo Entonces nosotros podemos utilizar justamente ese n para poder eh definirle ese valor también entonces voy a exportarlo también okay y aquí voy a hacer lo mismo le digo que voy a importar desde la misma entidad el task status y aquí voy a decirle Okay voy a

(33:37) utilizar ese Tas status punto y voy a decirle que estas tareas inicialmente va a estar en pending o pendiente listo Entonces esto es lo mismo voy a retornar esta propiedad de tareas cuando pidan get Tas de esta forma simplemente le digo vas a retornar Tas solamente que aquí para que pueda acceder esto como es una propiedad vamos a colocarle dis punto Tas y bueno cuando yo quiero crear algo por lo contrario lo que voy a hacer es recibir parámetros dentro de est de este método Entonces yo puedo decirle Okay voy a recibir un título voy a recibir una descripción Solo que aquí

(34:09) voy a tener que decirle Qué tipo de datos Entonces le digo el título va a ser el tipo dato Stream la descripción va a ser el tipo dato stram y aquí nosotros vamos a decirle que vamos a añadirlo a este arreglo que habíamos creado Entonces eso es bastante fácil simplemente le digo dis task Push y de esta forma yo puedo Añadir algo dentro del arreglo ahora lo que que voy Añadir vamos a crearlo aquí afuera primero voy a decir voy a crear una una tarea nueva que se llame task y esa tarea va a tener como propiedades el título que me están pasando la descripción que me están

(34:37) pasando pero adicionalmente también tengo que colocarle un estado inicial entonces también le voy a decir el estatus que va a tener la nueva tarea por defecto siempre va a ser Tas status pending ya que es una tarea nueva entonces eh va a estar en estado pendiente y como es una tarea también que necesita un ID voy a colocar aquí un ID y como debe ser un Stream el que le necesito pasar puedo colocar aquí también valor dos o puedo colocar cualquier valor incluso puedo utilizar módulos de tercero eh Como ID por ejemplo pero en este caso simplemente voy a utilizar un objeto date Esto va a

(35:08) generarme un objeto de de fecha de javascript y luego voy a utilizar el método tu ISO String que va a convertirlo simplemente a una cadena e esta fecha Entonces como la fecha siempre es única o el en este caso la fecha con la hora siempre es única esto se va a convertir a un Stream que va a ser como una especie de dato aleatorio de hecho esto lo pueden probar también en consola del fronte si copian esto de aquí y viene una en un navegador en la consola y lo pegan y dan un enter pueden ver que esto es lo que nosotros vamos a generar como un ID y

(35:38) bueno Esta es una forma en realidad de cómo utilizarlo pero si no quieren ese tipo de Stream nosotros vamos a instalar también un módulo más vamos a limpiar aquí la consola vamos a cancelarlo con control c y puede decir MP install uid Y si doy un enter esto lo que va a hacer es que va a instalar un módulo extra entonces este módulo yo lo puedo importar aquí arriba le digo voy a import ortar algo desde y Aquí vamos a colocar el módulo uid y este módulo sirve para generar ids únicos Entonces desde aquí le digo voy a importar el

(36:07) módulo que bueno su función aquí tiene varias ID en este caso creo que me están srado porque no encuent los tipos de datos me dicen Necesito los tipos de datos de uid cuando estamos utilizando módulos que están desarrollados en javascript necesitamos Instalar los tipos de datos por aparte así que tenemos que ejecutar esto de aquí Entonces vamos a venir aquí en la consola y le digo env install @types uid y como esto tan solo para desarrollo vamos a colocarle la opción gu d mayúscula listo Ahora sí con esto yo voy a poder decirle Okay ya se va es

(36:38) esos tres puntos y le digo Quiero el módulo v4 que es para generar un Stream Entonces le digo que ya no vas a generarlo de esta forma sino vamos ejecutar v4 y ya está entonces de esta forma estoy creando una nueva tarea con un ID generado con un título una descripción con un estatus pendiente y esto es lo que voy a Añadir al arreglo de tareas y ya está ahora como yo quiero retornar justamente en la nueva tarea creada simplemente voy a decirle Okay después de que lo añadas vas a retornar al a que utilice el método la tarea nueva y ya está entonces si yo quiero

(37:08) ver esto ahora simplemente vamos a venir en el controlador y aquí así como tenemos un método get voy a crear otro método que se llama por ejemplo create task este create task lo que va a hacer Es que para empezar vamos a tener que llamar el servicio entonces le digo voy a llamar el dis task service y quiero llamar su método create task ejecutarlo Él espera dos parámetros Él espera el parámetro eh title aquí pueden verlo espera el parámetro title y también espera el parámetro description entonces para yo pasárselo primero tengo que comprobar De qué forma llegan los datos

(37:40) aquí al backend Entonces vamos a comentarlo de momento y vamos a comprobar Cómo el el backen recibe los datos del cliente entonces aquí para poder hacerlo primero vamos a definir que este ya no va a ser un método get sino va a ser un método post y ya está de esta forma ahora nosotros podemos tener métodos post Al momento de crear datos y Bueno aquí pueden ver que también lo he importado por mí pero si no se dan cuenta pues también pueden hacerlo manualmente pueden escribirlo aquí post y luego venir aquí escribir post también pero bueno es una eh es lo mismo al final es la misma forma y Bueno

(38:11) aquí adentro lo que voy a hacer es que primero Cómo puedo recibir los datos que el cliente envíe y bueno para poder hacer eso nosotros tenemos ahora lo que se llama un Body eh aquí por ejemplo tenemos un ar Body que es una forma de decir el request body el request body es tan solo una forma en la que el cliente nos puede enviar un dato en formato json o textos etcétera En nuestro caso cuando decimos @ Body nos está enviando un json con múltiples valores ahora este Body yo lo tengo que guardar en algún lugar

(38:36) entonces voy a decirle Okay lo que me está enviando el cliente yo lo puedo guardar en un valor que se llama por ejemplo New task o nueva tarea y demás al final esto depende eh más del eh del desarrollador como quieran llamarlo es bastante libre en ese aspecto entonces aquí pueden llamarlo New task y vamos a decirle que las tareas inicialmente van a tener un valor n y ya está entonces si yo quiero ver esto simplemente le digo conso log New task vamos a verlo escribir bien okay Y de momento voy a retornar un texto que diga por ejemplo

(39:05) guardando Okay listo vamos a guardar esto de aquí y si nosotros queremos probarlo tenemos que ejecutar nuevamente el backen y le digo mpn room Def Esto bueno mpn room stard Def listo y de esta forma va a arrancar el servidor listo entonces para poder probarlo en mi caso voy a estar utilizando un programa que se llama insomnia que no sé si lo tengo aquí en son Mia no lo tengo vamos a descargarlo Este programa de aquí eh es bastante práctico Porque e me permite poder verlo en otro en otra ventana y pesa bastante

(39:37) poco así que simplemente lo descargamos esto lo que va a permitirnos es poder tener una interfaz fácil para poder hacer peticiones como pueden ver también la descarga no demora mucho pesa alrededor de 120 megas Pero bueno si no quieren utilizar esto también otras opciones es postman o utilizar el módulo Thunder cine aquí en visel Studio code solo que en mi caso se ve muy mal el módulo cuando hago zoom aquí en la pantalla por eso que no lo utilizo para tutoriales o bueno ya no lo estoy utilizando para tutoriales listo est es insomnia Entonces cuando carga insomnia

(40:07) lo que vamos a ver es justamente una interfaz en donde vamos a poder hacer peticiones por ejemplo en nuestro caso hacíamos una petición Slash Tas a local Host 3000 Aquí también podemos hacer lo mismo le digo puede hacer una eh una petición de hecho voy a extenderlo digo puedo hacer una petición a http localhost 3000 el hashtag Y si envío pueden ver que me da un arreglo de tareas que aquí también puedo acercarlo un poco Ahí está el arreglo de tareas ahora lo mismo también puedo hacer una petición post sl Tas Si yo envío me dice guardando Y si yo vengo aquí en la

(40:41) consola del backend y abro aquí la terminal pueden ver que hay un objeto vacío pero esto significa que ya le he enviado un dato Solo que como no he escrito nada aquí pues es como diciéndole el cliente no ha enviado nada pero si quiero enviar algo puedo venir aquí en body en esta sección llamada Jason y aquí yo puedo escribir un objeto de esta forma y puedo decir okay Voy a enviarte un título de una nueva tarea que se va a llamar por ejemplo tengo que limpiar mi casa luego vamos a colocarle una descripción aquí un description que

(41:09) es el valor que espera y aquí le puedo decir por ejemplo eh tengo que hacerlo a las 3 pm supongamos Entonces yo envío esto y me hice guardando y ahora si yo veo la consola del backend Allí están los datos entonces pueden ver que es el objeto justamente que el cliente ha escrito entonces utilizando eso como yo sé que New Tas tiene dos valores yo voy a pasarle esos dos valores ahora un asunto muy importante con esto es que el New Tas es un valor Any eso significa que si yo escribo New Tas aquí debajo y coloco un punto pueden ver que typ screen no me da autocompletado de de

(41:40) valores porque é no sabe qué es lo que hay adentro justamente porque es en pero en en este caso lo que me dice njs es que cuando yo tengo valores que están siendo transportados desde el cliente al servidor O mejor dicho el cliente me está enviando esos valores nosotros podemos utilizar lo que se conoce como un dto un un dto o digamos Data transfer object es justamente como su nombre indica un objeto que está siendo transferido a través del cliente al servidor y nosotros podemos detallar Qué es lo que está llegando de esa forma podríamos validarlo podríamos tener autocompletado y demás entonces La idea

(42:12) es que si nosotros vamos a crear eh una tarea debemos especificar qué es lo que está viniendo Entonces vamos a venir aquí en Tas voy a crear una carpeta Por lo general se hace esto okay Y en esta carpeta vamos a crear una carpeta llamada dto De hecho no es enteramente necesario pero es algo Bastante dentro de dto vamos a crear un archivo que se llama por ejemplo eh tasd to.

(42:53) sn seotronix [Música] dato stram y la descripción también tiene que ser de tipo dato stram esto lo voy a exportar y listo de esto simplemente se trata el dto de esta forma ya podemos reutilizarlo entonces aquí en el controlador que es el que está recibiendo el body le digo Okay vamos a importar ese dto le digo voy a importar algo desde vamos a colocar un punto y un Slash de la carpeta dto el archivo Tas dto y desde aquí voy a traer la clase llamada create task dto y ahora simplemente le digo este New task tiene los siguientes valores que es create task dto esto Ahora si yo por ejemplo escribo New task y doy un punto pueden ver que me

(43:32) dice okay puede venir un título y una descripción y ya está ahora el asunto es que no es tan solo para poder ver el autocompletado sino que ahora yo voy a descomentar este servicio y justamente este servicio esperaba un título y una descripción Entonces le digo el primer valor o el primer parámetro es New task title y el segundo parámetro es New task eh pun description listo si nosotros lo vemos extendido pueden ver que quedaría de esta forma la consulta ahora esto de aquí cuando se ejecuta me retorna un objeto con los valores ID title

(44:02) description y estatus Entonces eso es lo que voy a retornar al cliente Entonces le digo Aquí cuando terminas vas a retornar al cliente estos valores listo entonces si yo formateo esto de aquí pueden ver que í estaría entonces de esta forma también tengo el crear vamos a probarlo eh Recuerden que aquí se está ejecutando el servidor Entonces vamos a venir ahora en Thunder client y primero vamos a hacer la petición get de Slash task pueden ver que tengo la única tarea que he creado el backend Recuerden que

(44:26) este archivo de aquí las tareas que tenemos cread en el servicio esto es tan solo en memoria es decir esto se reinicia el servidor Y nuevamente se vuelve a crear Esto no es una forma de guardar datos Allí se tendría que utilizar una base de datos pero de forma resumida esto simula eh nuestra base de datos Okay y bueno vamos a hacer una petición get Slash Tas ahí está la única tarea vamos a hacer ahora una petición post y le voy a enviar esta tarea envío y pueden ver que se ha creado una tarea con un ID ese es el ID de la tarea el

(44:55) título la descripción y el estatus está pendiente ahora si yo hago una petición get de todas las tareas pueden ver que tengo dos ahora okay Lo mismo puedo crear otra tarea por ejemplo voy a crear otra petición post y ahora va a ser por ejemplo eh tengo que estudiar ns y por ejemplo tengo que hacerlo a las 2 pm envío y pueden ver que se ha creado Entonces si hago una petición get sl Tas ahí están todas las tareas Claro si nosotros venimos en el código y presionamos control S esto lo que va a hacer es reiniciar el código del servidor Entonces si nosotros volvemos a ejecutar la petición como se ha reiniciado otra vez se ha limpiado el

(45:29) arreglo Pero bueno La idea es que ya estamos guardando datos y muy bien entonces como han visto nosotros tenemos que crear primero un servicio en donde nosotros vamos a poder guardar datos aquí podrían ser consultas perfectas de sql o de cualquier otro otra base de datos relacionada a por ejemplo guardar datos a a futuro y al final cuando tenemos el servicio ya podemos llamar losos a través de controladores es decir crear métodos que utilicen el servicio y es por eso que justamente aquí estamos estamos enlazando el controlador con el

(45:58) servicio y aparte de esto también hemos creado un dto que es simplemente para detallar los datos que están llegando al backend o los datos que el cliente ha enviado para poder tener autocompletado o saber qué es lo que está viniendo de hecho el dto aquí nosotros también podemos detallar por ejemplo eh Qué valores esperamos validar luego es decir podemos decirle luego que yo espero que el título eh tenga determinada cantidad de caracteres y demás Pero eso lo vamos a ver en una sección adicional Entonces vamos a cerrar esto de Aquí vamos a cerrar todo y continuamos en la

(46:27) siguiente sección con las dos operaciones que nos faltan en esta sección Vamos a aprender Cómo eliminar y Cómo actualizar para poder hacerlo vamos a venir ahora en el Tas service y así como teníamos métodos para poder obtener todas las tareas o crear todas las tareas o bueno crear una tarea vamos a hacer funcionar también estos dos de momento vamos a empezar con el delete que creo que es más fácil entonces en el delete para poder eliminar simplemente necesitamos un ID para poder buscarlo Entonces como ya sabemos el ID es un stram Así que voy a decir el ID es de tipo dato stram esto

(46:56) de aquí al elinar me va a devolver por ejemplo la tarea que ha quitado entonces voy a decirle cuando elimines me devuelves la tarea que has eliminado Ahora aquí me está dando un error porque esto no está devolviendo eso pero lo que voy a hacer Es que aquí nosotros vamos a tener que utilizar el arreglo y le digo vas a utilizar el método find Y utilizando el método F él puede perdón el método filter él puede eh quitar un elemento o mejor dicho puede hacer una condición crear un nuevo arreglo sin la tarea que ha coincidido de esta forma es como un eliminar entonces de esta forma

(47:27) si yo lo ejecuto aquí le puedo decir Okay Al momento de filtrar vamos a estar utilizando por cada tarea que recorras una condición si la tarea por ejemplo en su propiedad ID es distinta de el ID que me están pasando Entonces esto de aquí te va a devolver un arreglo sin esa tarea entonces de esta forma si yo aquí por ejemplo le coloco un return Esto va a estar devolviendo en este caso un arreglo de tareas y bueno Esto de aquí ya no está coincidiendo digamos con con este parámetro de aquí porque va a estar retornando un arreglo entonces Aquí voy

(47:56) a quitarlo y de hecho no tiene que retornarlo porque al momento de eliminar lo único que espero es que lo actualice entonces también puedo decir okay esto no va a retornar nada simplemente va a actualizar las tareas que ya tenemos con este nuevo arreglo ahora para que puedan entender esto mucho mejor pueden venir en el navegador y en un un digamos una consola del navegador pueden hacer esto si ustedes tienen valores como 1 2 y tres por ejemplo y lo tienen guardado en un una constante que se llame array por ejemplo o ar si ustedes quieren quitar el valor dos ustedes pueden utilizar el

(48:26) método filter entonces pueden hacer algo como esto le dicen arreglo quiero filtrar y por ejemplo vamos a llamar a cada elemento x Okay x x y demás okay entonces aquí le digo por cada elemento x quiero que lo recorras Y si x es distinto de 2s pueden ver que de esta forma tan solo me devuelve un arreglo con uno y tres es lo que estoy haciendo aquí justamente si la tarea en ID es igual al ID que me están pasando entonces esa tarea no la añadas en el nuevo arreglo y ese nuevo arreglo que estás creando asín aselo al digamos a la propiedad que Ya teníamos antes entonces de esta forma es como un eliminar Pero

(49:00) bueno es lo que vamos a estar utilizando aquí en el cliente y ya está entonces voy a guardar esto y para utilizarlo vamos a venir en el controlador y vamos a crear otro otro método más que se va a llamar igual delete task y este delete Tas vamos a colocarle @ delete y cuando llamo este delete aquí arriba Recuerden que se ha importado okay Y para hacer funcionar esto si yo quiero eliminar algo obviamente no puedo solicitarlo simplemente con un Slash d sino que tengo que colocarle algún valor entonces hay varias formas de pasarle un valor

(49:29) uno es utilizando un query por ejemplo le puedo decir de todas las tareas quiero eliminar que tenga el id1 o el ID 2 etcétera Pero en mi caso voy a decirle cuando te pasen de Slash Tas Slash eh algún ID por ejemplo x y z 1 2 3 que es un ID pues digo cuando eh te pasen un valor así vamos a estar extrayendo esta parte de la URL que vendría a ser justamente algo que puede cambiar porque pueden enviarte un 23 un 24 1 25 etcétera entonces este valor de aquí y se le conoce como un parámetro y bueno en ns para poder definir ese valor como

(50:02) ustedes saben aquí habíamos utilizado en get task que esto es como decir cuando hago una petición a Slash task Okay lo mismo también en Post cuando hago una petición Slash task y aquí si yo escribo delete de esta forma es como decirle cuando hag una petición delete sl task pero aquí le digo cuando eh hagan una petición Slash task sl un ID entonces para poder hacer eso aquí le digo dos punos ID esto es como decir de Tas luego de Esto va a venir un número que Yo no sé cuál será o un texto que no sé cuál

(50:29) será entonces de esta forma le digo que cuando sea un ID voy a procesarlo con esta función ahora este delete simplemente va a llamar el el método de task service que se llama delete task y aquí le voy a decir Okay cuando vas a eliminar algo Necesito el ID que están pasándote entonces justamente para extraer esa parte del de la URL es decir lo que te están pasando aquí nes tiene aquí un decorador que se llama @param okay Y si yo lo importo de esta forma aquí arriba pueden ver que lo ha importado Aquí está eh simplemente voy a

(51:00) decirle qué nombre va a tener entonces voy a decirle va a tener el nombre que se va a llamar ID lo voy a guardar en una propiedad ID que va a ser de tipo dato stram Entonces ese es el valor que voy a pasarle al delitas y ya está de esta forma va a eliminar Entonces yo guardo esto y vengo aquí en la consola pueden ver que no tengo errores entonces si yo por ejemplo hago una petición get de todas las tareas envío pueden ver que ya ahí tengo una tarea si yo quiero crear algo nuevo le digo po envío se ha

(51:23) crear una nueva tarea se una petición get de todas las tareas ahí ya están las dos Y si quiero eliminar una Por ejemplo yo quiero eliminar esta tarea de aquí que tiene este ID copio esto le digo de tareas sl pego todo ese ID voy a dejarlo un poco para que puedan verlo mejor ahí está voy a acercar Ahora sí Y una vez he pegado eso lo que podemos hacer ahora es decirle Okay vamos a enviar una petición delete y le digo de local Host 3000 de todas las tareas quiero eliminar esta tarea envío y me dice no esy respuesta pero si hago una petición ahora eh get

(51:55) de El http localhost 3000 sl task y envío pueden ver que en este caso ya no está la otra tarea que habíamos creado Entonces si quiero eliminar esto Esto es más fácil porque tiene un ID un Entonces le digo elimina también la tarea uno delete envío se va Y si hago una petición get de todas las tareas pueden ver que ya está vacío el arreglo claro esto si reinicio nuevamente va a estar iniciando con la tarea que tenemos aquí en el servicio esta tarea de prueba pero esto significa que ya las operaciones están

(52:26) funcionando ahora en cuanto al eliminar o Perdón Al actualizar es muy similar en cuanto a recibir un Stream como parámetro Solamente que también tengo que recibir los campos que voy a actualizar Entonces el update Tas va a recibir tres parámetros uno va a ser un ID que es un Stream y luego también podría recibir un título y una descripción como parámetros extra pero una forma más sencilla es agrupándolos Entonces yo puedo decirle Aquí voy a recibir Campos para actualizar updated fields o Campos actualizados y bueno de momento voy a decirle Any y Bueno aquí

(52:55) lo que voy a hacer es tratar de hacer una búsqueda primero y a partir de esa búsqueda voy a estar eh digamos actualizando porque no puedo actualizar tan solamente tengo que buscar primero si existe o no Entonces como es un método eso de buscar una sola tarea yo puedo crear aquí otro Método en el servicio que se llame algo como esto get eh task incluso puedo colocarle algo como esto le digo get task by ID Quiero buscar una tarea por ID entonces aquí lo que voy a hacer es que justamente vamos a utilizar el método find de javascript

(53:26) entonces puedo hacero así le digo de estas tareas utilizo el método fine y por cada tarea si la tarea en el ID coincide con el ID que te van a pasar que aquí voy a decirle cuando buscan una tarea te tienen que pasar un ID si coincide esto te va a retornar un objeto que va a ser una tarea entonces aquí le digo vas a retornar ese objeto entonces este método también yo puedo decirle aquí vas a estar retornando esa tarea o vas a estar retornando un objeto de tipo tarea y ya está ahora esto de aquí si nosotros lo encontramos retornarlo si no

(53:57) no va a Añadir nada Es decir no va a retornar ningún valor Entonces esto de aquí nosotros vamos a utilizarlo en el update Tas le digo OK Sí de hecho vamos a crear primer ejecución le digo de dis find task by ID de hecho por aquí debería estar getas by ID se llama obtener una tarea por ID le paso el ID esto te debería retornar una tarea nueva Y utilizando esa tarea le digo si existe o no ahora de momento voy a considerar que existe entonces simplemente voy a Eh modificarlo entonces para modificarlo nosotros tenemos que decir okay de esta tarea que

(54:34) has encontrado su título voy a colocar en Nuevo título su descripción voy a colocar una descripción y demás y puedo hacerlo manualmente sin embargo cuando me envían un dato para actualizar yo no sé si me van a enviar un título y una descripción o tan solo un título o tan solo una descripción entonces lo que voy a hacer Es que voy a combinar o voy a estar utilizando una propiedad de los objetos de javascript que se llama assigned es decir si por ejemplo de nuevo en el ejemplo de javascript si nosotros aquí voy a limpiar esto si tenemos un objeto que se llama por

(55:04) ejemplo a y tiene como valor x1 y y2 o supongamos que es y z3 vamos a colocarle Okay y guardo si yo tengo este objeto a pueden ver que tiene dos valores Pero me falta Y entonces yo quiero actualizarlo de tal forma de que y se añada Esto entonces lo que puedo hacer es crear otro objeto que se llame b y aquí le creo el valor y entonces le digo y va a ser igual a 2 claro entonces tengo a ahora tengo el objeto a Ahí está tengo el objeto B objeto B tan solamente Ahí está pero quiero combinarlos entonces para combinarlo existe una propiedad de

(55:38) los objetos de javas que se llama object assign y de esta forma si yo le digo que quiero que combines a y b Al momento de ejecutarlo aquí doy un enter pueden ver que me crea un objeto con los tres valores porque sean combinado ahora si nosotros tenemos un objeto que tiene propiedades repetidas se van a reemplazar también se los muestro Por ejemplo si tengo una constante a Bueno vamos a colocar ahora una constante x que tiene el valor por ejemplo name y por ejemplo es Ryan y tiene un Last name vamos a escribirlo bien Last name que es

(56:09) ra por ejemplo tengo x que vale Ryan reay o nombre Rey bueno perdón nombre Ryan apellido rate pero si yo quiero actualizar el apellido por ejemplo yo lo que puedo hacer es claro puedo actualizarlo manualmente o puedo crear otro objeto que se llame y y este objeto y puede tener Last name tiene que llamarse igual la propiedad y por ejemplo supongamos que quiero que se llame mcmillan okay doy un enter entonces tengo a que vale x bueno perdón tengo x que vale nombre y Last name tengo y que vale tan solo Last name pero yo quiero que este Last name se reemplace aquí entonces lo mismo object

(56:45) assign le digo object pun assign y combino los dos valores X y Y si doy un enter pueden ver que el valor que me está devolviendo es rey en mcmillan es decir actualizado el valor entonces entonces aquí debajo de esta tarea vamos a tratar de actualizarlo utilizando object assign Entonces sería esta forma utilizamos object assign que es un objeto global de javascript es por eso que no tenemos que importarlo y aquí voy a pasar el primer objeto y luego lo que quiero actualizar que sería el segundo campo que me está pasando la función es

(57:14) decir el updated fields Esto me va a retornar un nuevo objeto que voy a llamar New task por ejemplo y justamente con este New tendría que añadirlo O mejor dicho reemplazarlo en el arreglo ya que este sería el objeto actualizado ahora para poder actualizarlo Entonces vamos a Acceder al arreglo le digo dis Tas map y al recorrer con la función map cada uno de los objetos si este coincide con una propiedad voy a hacer que se reemplace Entonces sería de esta forma primero le digo por cada tarea que estás recurriendo le digo si la tarea en su propiedad ID es igual al en este caso al

(57:49) ID que me han pasado como primer parámetro de la función es decir este ID de aquí y luego simplemente le digo si es igual entonces vamos a tratar de colocar la nueva tarea entonces le digo Aquí Como operador ternario New task caso contrario le digo utiliza la tarea que ya estaba entonces en otras palabras si la tarea coincide pues va a colocar el nuevo el nuevo dato actualizado y si no lo va a dejar como está esto me va a dar un nuevo arreglo porque esto eh actualiza pero devuelve algo nuevo entonces voy a decirle que las tareas va

(58:19) a ser igual a este nuevo arreglo que ha creado y finalmente voy a retornar esto le digo retornan nuevas eh el arreglo que sería bueno en este caso podría retornarlo le digo New task aunque también podríamos simplemente enviar un mensaje o algo pero con esto creo que sería de esta forma Entonces lo primero que haríamos es Buscar la tarea Luego lo que voy a hacer es combinarlas esto sería una actualización si no hay nada nuevo lo va a dejar como está Y luego va a recorrerlo y va a actualizar claro aquí no estoy colocando ninguna condicional para ver si primero existe

(58:50) la tarea tampoco estoy comprobando si realmente cambió algo o si están los campos eh que realmente necesit y aquí también No importa si actualizó o no es decir si ocurrió un cambio siempre voy a estar actualizando Pero bueno Esto lo pueden ir mejorando luego con condicionales pero de momento está Eh bueno nos va a servir para funcionar o para actualizar esta esto de aquí Entonces vamos a probarlo ahora en el servicio o mejor dicho en el controlador y así como teníamos una ruta post y demás vamos a crear una ruta put Ah ahora un asunto con esto para poder

(59:22) actualizar nosotros tenemos métodos por ejemplo como put y Patch quizás si ustedes utilizan métodos como post get y delete y demás es bastante fácil de entender pero cuando actualizamos hay dos métodos o dos verbos hdtp el primero es el método más popular que es put este significaría que queremos actualizar todo el objeto pero también hay otro método que también sirve para actualizar que se llama Patch solo que en este caso es para indicar que queremos actualizar parcialmente algo es decir no queremos

(59:51) actualizar en su totalidad solo una pequeña parte de un dato entonces en nuestro caso como como nosotros sí queremos actualizar parcialmente es decir probablemente tan solo Quiero actualizar el título o tan solo la descripción pero no quiero actualizar ambos entonces eh o mejoro no es necesario que actualice ambos Entonces en este caso queda mucho mejor Patch Entonces vamos a utilizar en lugar de put Patch y aquí también necesito el ID Así que también lo mismo le digo voy a colocar aquí entre dobles comillas dos puntos ID vamos a colocarle un dato en

(1:00:19) mi caso voy a llamarle updat as y con este updat nosotros vamos a empezar a utilizar el servicio que ya habíamos creado entonces eh Ya tenemos el servicio instanciado simplemente le digo dis punto vamos a escribirlo aquí dis pun task service y aquí le digo update Tas ahora nosotros para recibir los datos vamos a hacer lo mismo que habíamos hecho con el eliminar vamos a recibir el ID por parámetro Entonces esto de aquí es Exactamente igual le digo @param voy a recibir un ID después de la URL y esto lo voy a guardar en una propiedad ID ahora en cuanto al dato que Quiero actualizar voy a recibirlo a

(1:00:54) través del request bu Entonces vamos a hacer algo similar a Esto entonces Aquí voy a colocar una coma le digo @body y vamos a decirle que aquí están los campos que se van a llamar Campos actualizados ahora para actualizar nosotros vamos a querer en lugar de colocar n vamos a querer colocar qué Campos vamos a estar esperando entonces así como teníamos un dto para crear tareas también vamos a crear otro dto pero para actualizar no tenemos que crear un archivo podemos crearlo aquí mismo por ejemplo puedo hacer esto le digo Sport cl update tas

(1:01:25) dto y aquí adentro vamos a colocar las propiedades que yo espero ahora el asunto con esto es que probablemente para actualizar Yo puedo esperar un título y una descripción pero de nuevo probablemente tan solo Quiero actualizar uno de los dos entonces puedo decirle esto le puedo decir cuando actualicen quiero que envíen un título y una descripción y Recuerden que en nuestro caso también si vemos el task entity También tenemos otra propiedad que se llamaba Don Okay entonces aquí lo que vamos a hacer es que esa propiedad también vamos a tener que establecerla entonces le digo okay también vas a

(1:01:55) recibir un status De hecho aquí le hemos colocado eh la propiedad de status Okay y estatus puede tener los valores de Tas status entonces aquí también lo mismo le digo puede tener los valores de Tas status de hecho lo tengo que importar aquí arriba Entonces le digo voy a importar eh Tas status De hecho aquí no me lo está dando creo que es porque es en camel Case okay Bueno ahí está ahora este de hecho esto viene desde la misma carpeta que se llamaba Tas entity y ya está listo entonces este Tas status lo que vamos a hacer Es asignárseles okay entonces de

(1:02:32) esta forma Simplemente estoy detallando qué es lo que puede llegar ahora en este caso eh De hecho algo que me faltó aquí es que al momento de crear también me pueden mandar el estado si por ejemplo quieren crear una tarea hecha no sería lo ideal Pero bueno también podría colocarlo entonces en nuestro caso no tiene mucho sentido que cree una tarea nueva y que ya esté hecha entonces es por eso que no no lo voy a hacer en esta Api pero justamente para actualizar no es necesario que me envíen todos los

(1:02:55) datos Así que aquí voy colocarle opcional con este signo de interrogación y esto es para que typ es sepa que hay veces en las que algunos datos van a ser indefinidos Entonces vamos a venir aquí en task controller y le voy a decir Okay no va a ser de tipo Any sino va a ser el tipo update vamos a decirlo bien Tas dto Simplemente escribiéndolo ya le importa pueden ver que aquí está entonces debajo lo que vamos a hacer es utilizarlo simplemente Entonces el primer parámetro es el ID y el segundo es el updated fields ahora el asunto con este update

(1:03:25) de aquí es que una ve es esto actualiza devuelve una tarea nueva Porque justamente Si vemos el dato el momento de update De hecho aquí no me da mucha información me dice dos puntos en vamos a ir allí vamos a ir al servicio nuevamente en el Tas service y vamos a añadirle una propiedad extra por ejemplo aquí le digo que esto también va a ser tipo update Tas dto okay Y aquí también vamos a colocarle un dos puntos y le voy a decir esto va a retornar una tarea Entonces ahora sí en el controlador si posiciona el cursor pueden ver que me hice esto retorno una tarea listo Entonces esto significa que cuando vamos

(1:03:55) a ejecutar este servicio él me va a retornar algo y justamente esto yo lo puedo retornar al cliente entonces también puedo decirle return formateo y así quedaría listo entonces con esto ya tenemos la forma de poder actualizar un dato o una tarea vamos a probarlo vamos a guardar aquí se está ejecutando no tengo errores y entonces vamos a venir en insomnia Okay y vamos a hacer una petición ahora el asunto con nuestro en nuestro endpoint que tenemos aquí es que tenemos que enviarlo a través del método

(1:04:24) Patch no funciona a través del método entonces voy a enviar el método Patch le digo que voy a hacer una petición de todas las tareas primero get para ver el ID tengo el ID un pero vamos a crear una tarea nueva le digo post le envío esta tarea se crea algo nuevo voy a actualizar ahora con Patch le digo Quiero actualizar esta tarea con este voy a copiar esto coloco un Slash y ahora si quiero actualizarlo tengo que colocar algo entonces aquí en el título voy a decir Okay vamos a colocarle eh tarea actualizada de momento y Si yo envío esto pueden ver

(1:04:56) que lo único que ha cambiado es el título el resto se mantiene esto es justamente lo que hace o lo que utilizaríamos cuando utilizamos Patch ahora en este caso aparte de esto también puedo colocar un description y este description también le digo okay esto es descripción actualizada Okay vamos a enviarlo y esto también actualiza tan solo la descripción y lo último con el estatus le digo el estatus por ejemplo vamos a colocarle Don okay Bueno en este caso no sé los estados que tenía vamos a verlo había tres pending in progressing Entonces vamos a colocarle Don envío me dice done o puedo cambiarlo nuevamente a

(1:05:31) pending Okay vamos a escribir bien envío se actualiza y Si yo envío algo por ejemplo lo envío cualquiera Y ejecuto en es caso también lo actualiza Por qué Porque al momento de nosotros enviar esto no hay ninguna forma en la que se esté validando Entonces eso lo vamos a modificar en la última sección en donde voy a enseñarles cómo validar a través de justamente esta entidad que acabamos de crear Pero eso lo vamos a hacer en la siguiente sección en esta última sección vamos a validar los datos para poder hacerlo vamos a

(1:06:00) estar utilizando una biblioteca que ya tiene n o que ya la tiene integrada que se llama validator o Class validator de hecho pueden buscarlo de esta forma n validations y esto lo va a llevar a la documentación de validaciones en es bastante fácil de hecho utilizar esta biblioteca porque como ya estamos creando clases es tan fácil como Añadir más decoradores entonces es tan fácil primero como instalar este Comando vamos a copiarlo vamos a venir en la consola vamos a aquí con control c y vamos a pegar esto de aquí Entonces estos son dos clases uno es Class Transformer y Class validator Class validator es el

(1:06:33) que realmente nos permite hacer las validaciones entonces para poder empezar a utilizarlo primero vamos a venir en la entidad y aquí nosotros vamos a e digamos colocar unos decoradores extras ahora lo estoy colocando o lo puedo colocar aquí si en ese caso Yo quisiera estar eh digamos validando el momento de utilizar las entidades pero recuerden que en nuestro caso cuando yo quiero validar eh si bien estos son los datos que yo espero nosotros tenemos una clase dedicada a ese tipo de de datos que están llegando desde el cliente al backend recuerdan los dto bueno los dto justamente significan eso eh datos que

(1:07:10) están transfiriéndole al servidor Entonces eso es lo que vamos a querer validar entonces tiene mucho más sentido en lugar de colocar en la entidad colocarlo aquí en este titoo Entonces vamos a importar aquí Class validator les digo voy a importar algo d y Aquí vamos a decirle Class validator ahora esto de aquí lo que trae son algunos decoradores por ejemplo supongamos que yo quiero asegurarme de que el título que me están enviando sea realmente un Stream porque de momento si yo vengo aquí y le digo por ejemplo vamos a crear un post y le digo voy a

(1:07:38) crear un título okay Y coloco por ejemplo 20 que es válido en Jason Recuerden que en Jason me permite enviar ese tipo de datos y una descripción que diga por ejemplo un Stream y yo envío esto esto carga y bueno en este caso ha fallado vamos a ver qué ha pasado aquí Ah bueno no ha ejecutado Vamos a darle emp room stard Def porque esto debe funcionar igual no hay ningún problema vamos a enviarlo y pueden ver que bueno en este caso lo he enviado mal porque es tan solo Slash Tas a través del método post envío y pueden ver que lo crean y claro esto es válido pero no Debería ser

(1:08:11) un número el título es un texto entonces para validar eso yo le puedo decir Okay cuando creen algo el título va a ser un arroba y Aquí vamos a colocarle por ejemplo String ahora esto nosotros vamos a tener que importarlo le digo is aquí pueden ver que cuando escribo is es para colocar distintos tipos de decoradores en este caso vamos a buscar el e String Okay y el is String es para decir Okay este este en este caso Este título Debería ser un Stream es tan simple como eso voy a guardarlo Y si ahora envío lo mismo envío esto de aquí pueden ver que sigue funcionando Es decir me sigue

(1:08:43) permitiendo Añadir números y la razón es que si bien estamos añadiendo decoradores para decir qué es lo que esperamos tenemos que Añadir algo más que también nos dice la documentación y es que debemos ejecutar estas validaciones entonces para ejecutarlo nosotros nosotros podemos hacerlo en cada ruta o hay una forma mucho más simple que es utilizando esta palabra que se llama use global pipes vamos a hacerlo vamos a venir en main.ts y debajo de este app vamos a escribir app.us global pipes que es esto de aquí

(1:09:09) y al ejecutarlo nosotros podemos pasarle una validación la validación justamente viene desde nes el módulo entonces aquí desde ncore De hecho aquí la documentación también la específica que es en es caso lo coloca New validation Pipe Ahora hay una forma más rápida de importarlo que simplement escribiendo New validation Ah Pipe y se da un enter pueden ver que le importa De hecho aquí le importa es la biblioteca common yo creía que lo importaba desde Core pero es des de command ahora con esta simple línea de código es como decir cuando yo cree validaciones en múltiples entidades o mejor dicho en múltiples dto siempre

(1:09:43) van a validarse Entonces si existe una clase que está utilizando los validadores de class validator se van a validar automáticamente Entonces esto es una forma de decir valida todo listo entonces una vez tenemos eso si ahora sí vuelvo a ver la consola y ejecuto esto pueden ver que me envié un error me dice status 400 ya le colocó el código 400 y el mensaje del error me dice el título Debería ser un Stream Okay entonces aquí le digo vamos a enviarle un Stream envío y ahí ya está ahora el asunto es que

(1:10:09) claro me está enviando un Stream pero está vacío y bueno Esto también es válido porque cumple con la validación pero adicionalmente también le puedo decir que no este vacío entonces voy a colocarle otro decorador aquí debajo y esto se va a llamar is not empty también se importa desde Class validator Y si yo guardo y ahora envío esto me dice el título no debería estar vacío y no es que funcione uno otro funciona ambos Por ejemplo si yo envío 20 me dice el título Debería ser un stram si envío el Stream vacío me dice el título no debería estar vacío y si envío por ejemplo eh a por

(1:10:39) ejemplo y envío esto es válido porque ya tiene algo ahora el asunto es que probablemente también quisiera que el título tenga una cantidad mínima de caracteres entonces también tenemos otro decorador que se llama min por ejemplo y este min es para nosotros colocarle la cantidad de caracteres Entonces le digo mínimo que sea de tres caracteres entonces aquí le coloco por ejemplo a envío eh Debería ser mínimo tres Entonces le digo a b c envío Debería ser mínimo tres y bueno en este caso es min porque dice el título no Debería ser menos que tres Okay entonces eh en es caso para poder validarlo supongamos que

(1:11:10) yo lo coloco d claro yo esto pasa ahora la razón de que esto me esté dando el mismo problema a pesar de que son tres es que claro este mínimo si nosotros vemos aquí eh En este caso aquí me da un comentario dice esto revisa si el primer número es menor que el segundo es decir en este caso está haciendo una comparación de números es decir está comparando un título con el buen en este caso un String que sea menor a tres y claro esta comparación no tiene mucho sentido Entonces nosotros lo que esperamos es una longitud un min lent realmente entonces Aquí le voy a decir

(1:11:40) mínimo va a ser una longitud de tres y claro el min ya no lo estaría utilizando Pero bueno es algo también a tomar muy en cuenta cuando utilizamos este tipo de cadores sabemos que estamos validando distintos tipos de datos entonces voy a enviar esto nuevamente y pueden ver que ahora sí pasa okay y listo de esta forma estamos validando ahora en el caso de los otros parámetros por ejemplo la descripción también puedo hacer lo mismo le digo okay esto Debería ser un stram el asunto con esto es que quizás eh obviamente ustedes van a tener que colocar eh Otras consideraciones si

(1:12:09) quieren luego tener otros campos de números y demás Pero bueno Eso ya depende de de los decoradores que tengan a su disposición Pero bueno muy aparte de esto Si queremos también validar el actualizar también lo mismo por ejemplo en este caso voy a decirle Okay el título para actualizar Debería ser un Stream la descripción también y el estatus también solo que primero el título ya no es requerido Es decir el título no tiene que enviarme sino que lo único que voy a hacer Es que voy a decirle Okay si el título me lo envías al menos yo espero que sea un Stream

(1:12:41) Pero puedes que también no me lo envíes porque quizás tan solo quieres actualizar la descripción Entonces para decirle que es opcional también hay un decorador especial que se llama is optional que es justamente como decir okay puedes enviar este este dato Pero si no puedes enviar este otro De hecho aquí también voy a colocarle es opcional y también entonces los tres son opcionales porque bueno Estoy actualizando entonces aquí también le puedo decir import el decorador y optional listo ahora eh obviamente

(1:13:07) también podemos validar que sea longitud máxima y demás Pero bueno ya entendieron ahora Lo importante es este estatos de aquí si yo guardo esto y ahora Quiero actualizar algo por ejemplo le digo voy a hacer una petición get de todas las tareas y Quiero actualizar la tarea uno Okay vamos a utilizar método Patch Aquí está entonces le digo okay Quiero actualizar el título y la descripción envío se actualiza y bueno también le digo Quiero actualizar el estatus Entonces le digo Aquí Quiero actualizar también el estatus y vamos a colocarle cualquier cosa Si yo envío

(1:13:33) esto Esto es válido porque esto es un dato Stream y es opcional ahora el asunto es que probablemente yo quiero que coincida con esos tres valores que del enum es decir los tres valores de aquí pending y progress it done entonces para poder hacer que coincida con ese valor lo único que voy a hacer Es que voy a utilizar otro decorador que justamente va a decir is in este is in tiene como parámetro un arreglo en donde yo puedo pasarle por ejemplo los múltiples strings que van a coincidir en nuestro caso Ustedes deben pensar Okay

(1:14:01) Quizás lo vas a pasar manualmente es decir vas a pasarle d y demás Pero bueno no es necesario porque nosotros ya tenemos el enum aquí arriba este status entonces puedo acceder los uno por uno le digo de Tas status eh vamos a colocarle también quiero que sea por ejemplo o bien Open bueno en este caso se llamaba impr progress eh Tas status también quiero que sea por ejemplo o bien eh pending y finalmente Tas status que sea d Entonces yo formateo este is in sería básicamente los múltiples valores Entonces vamos a importarlo también le digo también voy a importar el is in listo Entonces ahora cada vez

(1:14:37) que me envíen un dato él va a comprobar que sea uno de estos tres valores y no me va a dar un error Entonces vamos a verlo envío esto que es cualquier cosa el estatus y pueden ver que mis estatus Debería ser uno de los siguientes valores pending Bueno he colocado veces pending eh aquí está uno Debería ser impr progress listo Entonces vamos a vamos a guardar vamos a enviarlo y bueno me dice uno Debería ser estos tres valores pending y Don entonces voy a colocarle por ejemplo Don Okay envío y en ese caso sí funciona ahora el asunto es cuando yo envío en minúscula obviamente esto no va a coincidir porque esto está haciendo una eh una

(1:15:11) comparación estricta Entonces en este caso obviamente no es igual Don en minúscula que mayúscula Pero si ustedes quieren evitarse eso también pueden convertirlo Al momento de recibirlo Pero bueno de momento creo que así funciona Entonces lo vamos a dejar allí y listo entonces de esta forma hemos terminado nuestra aplicación ya tenemos todas las operaciones ya estamos validando ahora finalmente en lo siguiente es empezar a Quizás desplegarlo eso ya se lo dejo en otros videos porque Bueno ya es un proceso que he explicado en muchos ejemplos eh digamos Cómo desplegar en

(1:15:40) múltiples servicios pero para darles una idea sencilla hace poco he creado un video en donde les muestro algunos servicios donde pueden desplegarlo gratuitamente eso les puede servir para aprender a desplegar Entonces por ejemplo el vide es este de aquí alternativa sag heroku eso lo pueden buscar en YouTube o se le dejo también el enlace en la descripción y bueno Allí va a encontrar alternativas como estas cuatro de aquí que es rail W bersell y render todas estas ofrecen crearte una cuenta gratuita no piden ningún tipo de

(1:16:04) tarjeta ni nada de eso y puedes desplegar tu proyecto en es pero bueno de momento lo voy a dejar aquí y voy a cerrar esto de aquí y finalmente lo que sería es hacer el build Entonces vamos a abrir la terminal vamos a cancelar aquí y vamos a decir mpn room build y damos un enter entonces finalmente Cuando hacemos el build lo que va a hacer n es que va a leer todo el código y va a generar la carpeta de producción obviamente Aquí está nuestro código entonces con esto nosotros tendríamos que pensar que si vamos a desplegar algo lo que tendríamos que desplegar la carpeta de distribución Pero bueno en

(1:16:36) este caso ya Eso depende más del de la nube que utilicen Y bueno ya estaría Entonces si van a desplegarlo lo único que tienen que saber es que tienen que ejecutar El Comando Start y ya está ejecutaría nuestra aplicación de producción y bueno eso ha sido todo por el video del día de hoy si tienen alguna duda pueden dejar en los comentarios y Bueno voy a estar publicando algunos videos más de nes y ya estoy preparando un curso un poco más extenso para el canal principal Así que van a poder verlo en algunos días y bueno allí voy a

(1:17:01) explicar mucho más a detalle Qué son los controladores los servicios ya tengo un video de nes que es algo viejo entonces voy a estar actualizándolo con más contenido y colocando algunas secciones extras además de también enseñar el despliegue en ese video porque Bueno es un curso para iniciantes entonces si quieren estar atentos les dejo aquí al final del video este canal principal en donde allí publico los cursos listo eso ha sido todo por el video del día de hoy si tienen alguna duda dejan los comentarios o una sugerencia de qué tema

(1:17:27) les gustaría ver y nos vemos en un próximo video