**AUTENTICACION EN ADONISJS CON JWT**

**Qué es JWT**

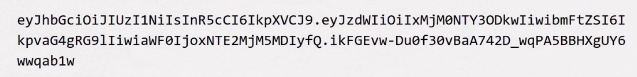
JWT (JSON Web Token) es un estándar qué está dentro del documento RFC 7519.

En el mismo se define un mecanismo para poder propagar entre dos partes, y de forma segura, la identidad de un determinado usuario, además con una serie de claims o privilegios.

Estos privilegios están codificados en objetos de tipo JSON, que se incrustan dentro del payload o cuerpo de un mensaje que va firmado digitalmente.

**Token JWT**

En la práctica, se trata de una cadena de texto que tiene tres partes codificadas en Base64, cada una de ellas separadas por un punto, como la que vemos en la imagen siguiente:



**Header:** encabezado dónde se indica, al menos, el algoritmo y el tipo de token, que en el caso del ejemplo anterior era el algoritmo HS256 y un token JWT.

**Payload:** donde aparecen los datos de usuario y privilegios.

**Signature:** una firma que nos permite verificar si el token es válido.

**Ciclo de vida de un Token JWT**



**PASOS:**

**1 CREAR PROYECTO**

npm init adonis-ts-app@latest 03-auth

**2 EJECUCION**

node ace serve –watch o

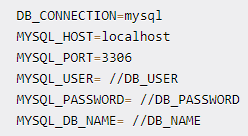
npm run dev

**3 INSTALACION DE ORM**

npm i @adonisjs/lucid@alpha

node ace invoke @adonisjs/lucid

**4 ACTUALIZAR EL ARCHIVO .env**



Si encuentra este error, **Client does not support authentication protocol requested by server**; mientras prueba su API, siga estos pasos para resolverlo:

npm install mysql2

Abra config/database.ts y actualice el campo client a mysql2. Eso debería corregir el error y estará listo para el desarrollo y las pruebas.

**5 CONFIGURAR LA AUTORIZACION Y AUTENTICACION**

Para configurar la autenticación y la autorización, instale el paquete Auth de AdonisJS y configure su proceso de autenticación y autorización.

Puede instalar el paquete Auth usando el siguiente comando:

npm install @adonisjs/auth@alpha

A continuación, llame al comando invoke para configurar el paquete Auth recién instalado con el siguiente comando:

node ace invoke @adonisjs/auth

Cuando llame al comando de invocación, lo llevará a través de un proceso de configuración para el paquete Auth. Siga los pasos a continuación para configurar el proceso de autenticación del token API.

1 Elija Lucid y API Token ya que estamos construyendo API.

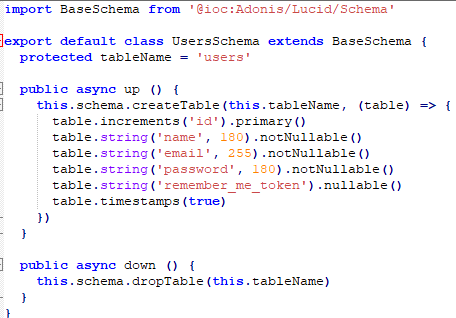
2 Escriba el modelo User para su autenticación.

3 Presione Y para crear una nueva migración de usuario.

4 Elige Database como tu proveedor.

5 Presiona Y para crear una migración api\_token.

Después de seguir los pasos con éxito, debería tener 2 migraciones generadas en su carpeta database/migrations. Abres la migración xxxx\_users.ts para actualizar la información según las necesidades de tu aplicación.



Por último, agregaremos el middleware auth al archivo start/kernel.ts. Esto siempre verificará cada solicitud para asegurarse de que esté correctamente autenticada.

Server.middleware.registerNamed({

auth: 'App/Middleware/Auth',

})

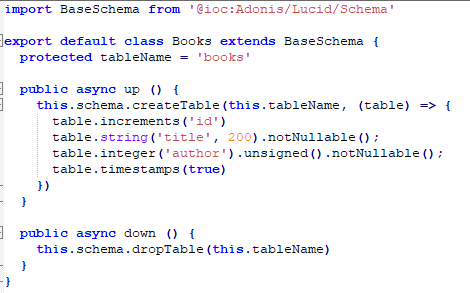
Después de configurar la autenticación y la autorización, necesitamos crear más migraciones para la aplicación books.

**6 CREAMOS MIGRACIONES**

A continuación, vamos a crear las migraciones restantes para el modelo books siguiendo los pasos que se describen a continuación.

Para crear una nueva migración, escriba el siguiente comando:

node ace make:migration books



Antes de ejecutar la migración debe tener la base de datos creada y activa, además de estar corriendo el servidor de Adonis.

Ahora ejecutamos la migración con el siguiente comando:

node ace migration:run

En caso de ser necesario reversar las migraciones:

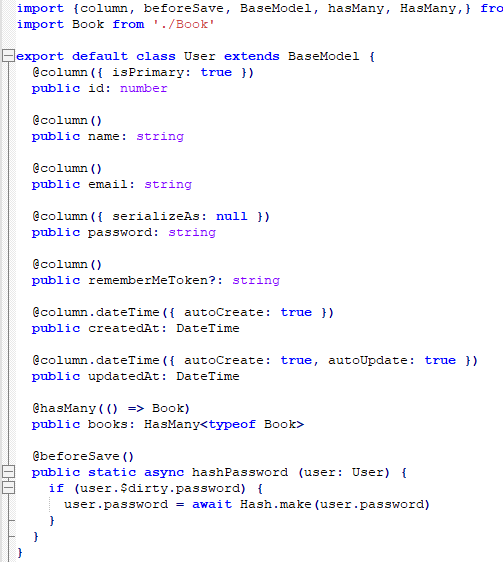
node ace migration:rollback

**7 CREAMOS LOS MODELOS**

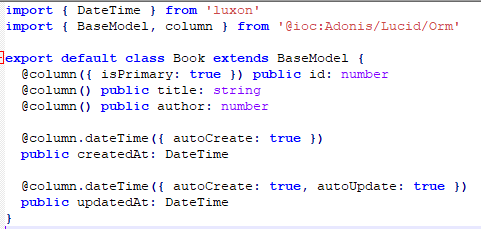
node ace make:model User

node ace make:model Book

**Modificamos el modelo User:**



**Modificamos el modelo Book**

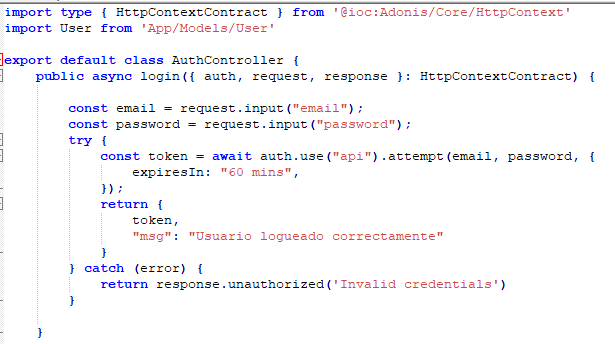


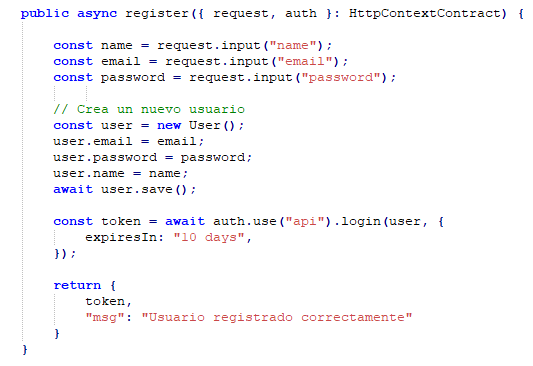
**8 CREAMOS CONTROLADORES**

node ace make:controller Auth

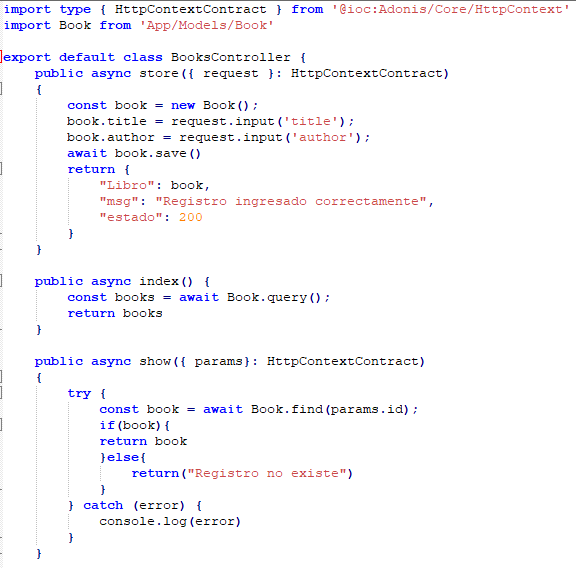
node ace make:controller Book

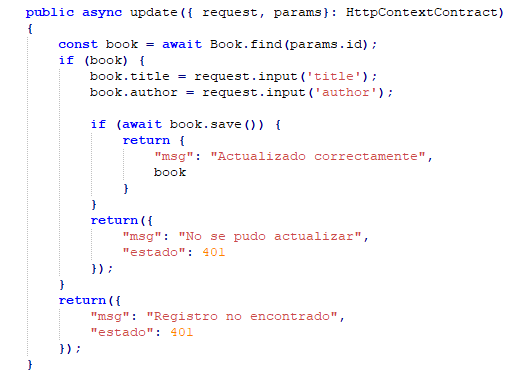
**Modificamos el controlador Auth**



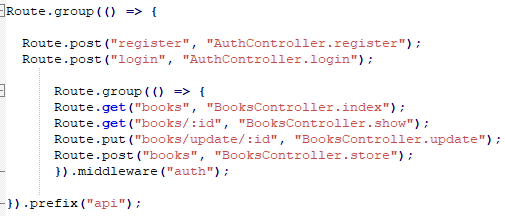


**Modificamos el controlador Book**





**9 Creamos las rutas**



**10 Pruebas:**

1. **Registrar un Usuario.**

* Seleccionar el método POST y escribir la ruta.
* En el body se ingresa la información en formato Json, seleccionando las opciones raw y Json.
* Clic en Send

1. **Loguear usuario con sus credenciales y token**

* Seleccionar el método POST y escribir la ruta.
* En Authorization / Type seleccionar Bearer Token, y en el campo Token pegar el token generado al registrar el usuario.
* En Headers, escribir **Accept** en la columna KEY, y **application/json** en la columna VALUE.
* En Body escribir las credenciales del usuario en formato Json, seleccionando las opciones raw y JSON.
* Finalmente clic en Send.

1. **Ingresar registros a la tabla books**

* Seleccionar el método POST y escribir la ruta.
* En Authorization / Type seleccionar Bearer Token, y en el campo Token pegar el token generado del usuario logueado.
* En Headers, escribir Accept en la columna KEY, y application/json en la columna VALUE.
* En Body escribir los datos del book en formato Json, seleccionando las opciones raw y JSON.
* Finalmente clic en Send.

1. **Consultar listado de libros que tiene la base de datos.**

* Seleccionar el método GET y escribir la ruta.
* En Authorization / Type seleccionar Bearer Token, y en el campo Token pegar el token del usuario logueado.
* En Headers, escribir Accept en la columna KEY, y application/json en la columna VALUE.
* Finalmente clic en Send.