Resumen del proceso de análisis:

“Implementar una estrategia de autenticación que permita un mínimo de seguridad en el uso del software, puede ser alguna de las siguientes: validación de token (estático) y vencimiento del mismo, validación de acceso por IPs, entre otras. Puedes usar manejo de sesiones, cookies, headers o cualquier otra que consideres. Bono extra: Implementar endpoints de autenticación: Login, Logout.”

En el siguiente documento se explica el desarrollo y la implementación de la capa de seguridad que se usó para la api de Rick and Morty.

En la implementación para la capa de seguridad se utilizó jsonWebToken que es una herramienta de creación de tokens que se almacenarán en el header y se validarán con los generados al loguearse, esta herramienta permite que solo las personas ingresadas que tengan el token sean autorizadas de ver la información que se ha restringido en las rutas.

Lo primero que se hizo fue instalar la dependencia jwt (jasonWebToken) en node para poder acceder a sus servicios.

Se creó un sistema de Log in y Log out con los cuales se generan los token a los usuarios registrados, para esto en la capa de servicios se generó el archivo auth.js el cual tendrá las funciones necesarias para este proceso (también se instaló la dependencia bcrypt la cual ayuda a encriptar y desencriptar las contraseñas para compararlas con las almacenadas en la base de datos )

En la capa de las rutas se instauró los servicios de log in y log out y en el archivo index.js y se declaró las rutas para verificar los end points.

Posteriormente en la capa de servicios hay un archivo llamado authetication.js que es el que valida los tokens generados, recordando que estos tokens estarán en el header y son otorgados al loguearse con un usuario y una contraseña validas.

Y se finaliza protegiendo las rutas y solo se le concederá permiso a los usuarios que tengan un token valido (ósea que se hayan registrado exitosamente y generado su token sin problemas, el código de protección de las rutas se ubica en la capa de rutas en el archivo index.js en la línea 31 .

Facilidades al momento de realizar el ejercicio:

Tengo conocimiento en manejo de django lo cual me permitió conocer la implementación de jwt y en manejo de un proyecto por capas, lo cual me facilitó entender el código y poder llegar a una posible solución.

En algún punto hice un pequeño proyecto con node.js y trabaje con end points en postman lo cual me facilito la interpretación de los archivos anexados.

Dificultades encontradas:

Realmente no pude probar los end points ya que una de mis falencias es el trabajo con bases de datos, siendo realistas y reconociendo mis impedimentos siempre me he encontrado con dificultades cuando no se trabaja con entornos que conozco o manejo, aunque con tiempo aprendo y me adapto con facilidad. Se me hizo complicado y me tomo mucho tiempo trabajar con sqlite. Y realmente no pude conectarla base de datos con los archivos que implementé.

Otra falencia es que aunque entendía gran parte del código, reconocía los middlewares implementados en la capa de servicios y pude reconocer las funciones de post, get y demás las cuales nos permitían acceder a los end points. La forma como plantearon la conexión y el acceso a los datos realmente nunca la había visto lo cual me produjo algo de frustración pero aun así continué con el desarrollo del ejercicio.

Aunque intenté creando otra base de datos, intente creando una nueva tabla llamada Users en la bd administrada con el proyecto (rickandmorty\_v1.db) se me hizo imposible poder acceder

Como puedo mejorar si me permiten trabajar con ustedes:

* Creo que la parte mas importante al momento de aprender es trabajar y al estar rodeado de más programadores de los cuales pueda aprender seria genial, ya que podré convertir estas falencias mencionadas anteriormente en grandes fortalezas y poder aportar cada vez mas a mis equipos de trabajo.
* Sé que no tengo el conocimiento que tal vez ustedes desean, pero parte de mejorar siempre es reconocer los errores que tenemos y espero poder tener una retroalimentación de su parte.

Bonus track:

Aunque no me funcionaron como deseaba los end points, hice un sistema de Registro de usuarios para acceder a la base de datos.

En los servicios en el archivo creado auth.js esta implementada la función register, y se declaró con un método post

Para la creación se consumen los servicios de bcrypt, usando la función hash para encriptar la contraseña y dar mayor seguridad al momento de acceder y proteger la información del usuario.