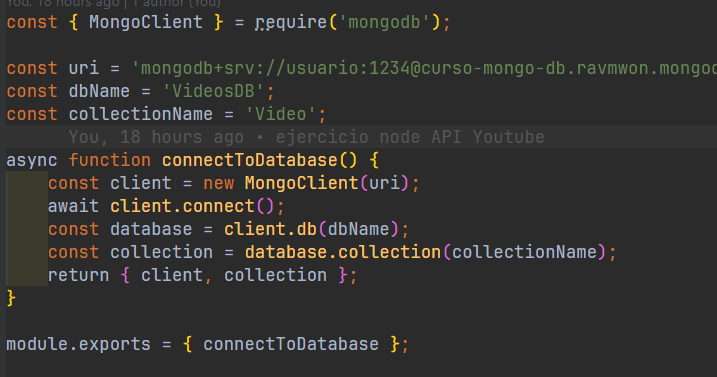
**PRÁCTICA ENTREGABLE API CON NODE Y EXPRESS**

**Resumen:**

El ejercicio pedía una Rest API de temática libre utilizando Express, mongoDb, Node y para hacer las peticiones yo he usado Postman.

La API que escogí ha sido la de Youtube para poder introducir el id de un canal y sacar su top 10 de vídeos más vistos.

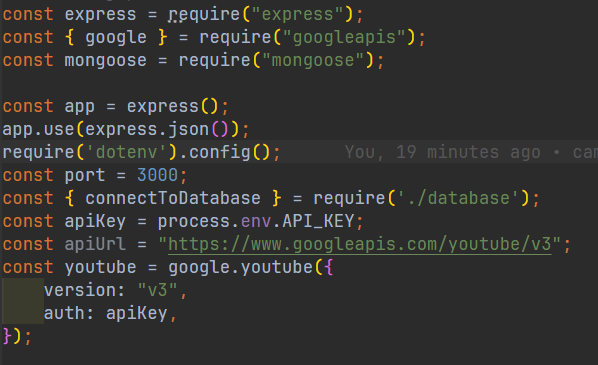
Primero he creado la conexión con mi base de datos de MongoDb.



He usado como parámetros la URI que me indicaba mongo en la web, el nombre de la base de datos que voy a usar y la colección que he creado para guardar la información de los vídeos.

Después he creado la función para conectar con la base de datos llamada ‘connectToDatabase()’ y he devuelto en ese método un objeto con los parámetros cliente y colección para poder usarlo en el fichero donde tenga todo el código con las peticiones.

Una vez creado el fichero con la conexión a la base de datos, he importado todos los módulos que necesitaba al principio del siguiente fichero de código.



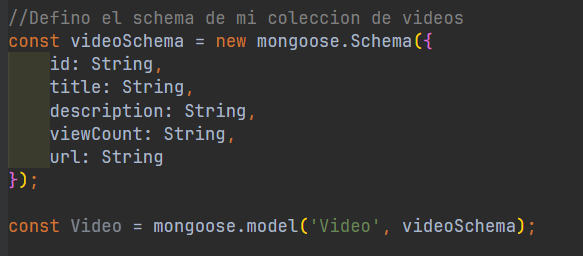
He instalado express, googleapis para poder utilizar la api de Youtube, mongoose para usar mongodb y dotenv para poder utilizar la api key de forma segura sin escribirla directamente en el código.

Para empezar, después de importar todos los módulos que necesitaba he llamado al archivo ‘.env’ donde tengo la variable de mi api key guardada, y me he traído el método connect to database del archivo anterior para poder hacer la conexión con la base de datos.

Luego he definido dentro de la variable de Youtube la versión de la api y la forma de autentificación que voy a usar.

Después he definido el esquema que voy a usar en mi base de datos con los datos que quiero guardar sobre los vídeos de un canal.

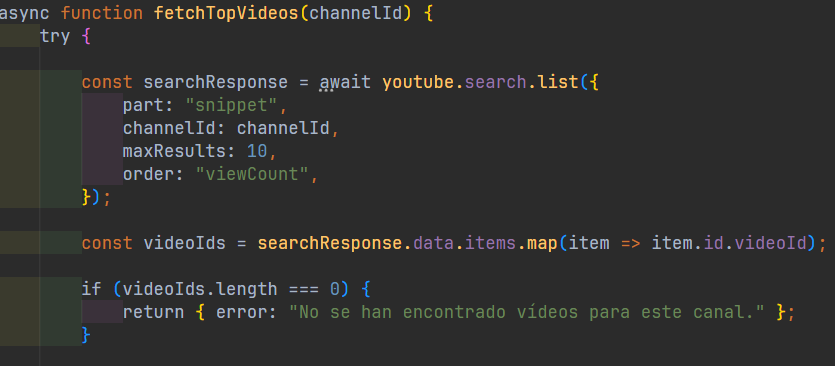
Y he indicado que voy a usar ese esquema como modelo.



En la siguiente captura he definido una función llamada ‘fetchTopVideos’ que obtiene por parámetro el id de un canal. He puesto un try and catch para capturar los errores y sacar un mensaje que me indique el tipo de error que devuelve el código.

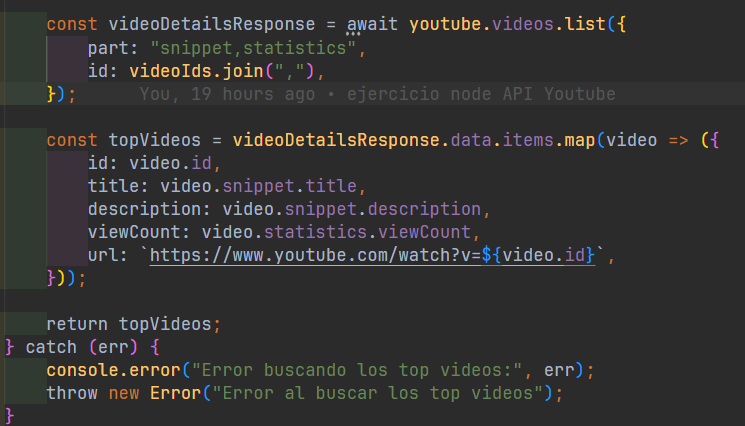
Para empezar he creado una variable llamada searchResponse donde mando los parámetros que necesita la api de youtube para poder usarse, en este caso el part, el channelId, maxResult donde especifico que solo quiero que saque 10 resultados y el order para que salgan según el número de visitas que tienen los vídeos.

He creado la variable videosId que va a ser la que obtenga por cada ítem el id de los vídeos que recupere. Para comprobar si me devuelve algún resultado he creado el if con su mensaje correspondiente.



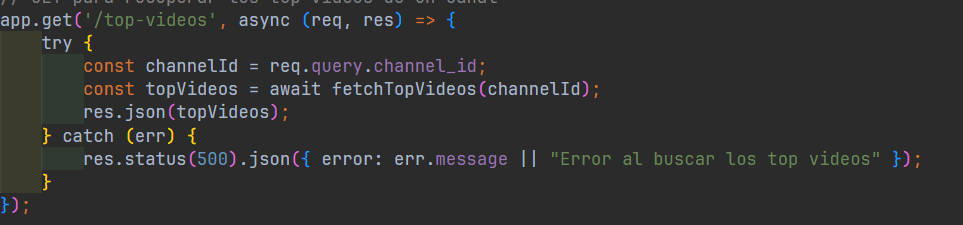
Para almacenar todos los datos estadísticos de los vídeos he creado la variable videoDetailResponse donde defino el part y el id del vídeo recuperado antes y separados por comas.

Con la variable topVideos he creado un mapeo de los detalles de los vídeos.

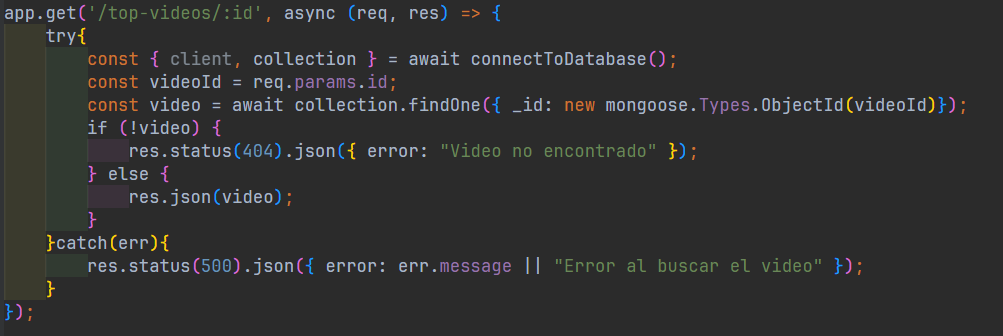
De cada vídeo se obtiene el id, el título, la descripción, la cantidad de visitas y su url.Después en la parte del catch he puesto un mensaje en caso de que no consiga sacar ningún vídeo.

A continuación, he hecho dos GET, el primero que me saque directamente los 10 vídeos más vistos del canal, y un segundo en el cuál si introduzco el id del vídeo me devuelva los datos exclusivamente de ese vídeo.

En este primero he utilizado la variable del canal para pasársela a la función de buscar los top vídeos y guardar la información en un json. Y en caso de que haya un error he puesto el código dentro de un try y un catch.

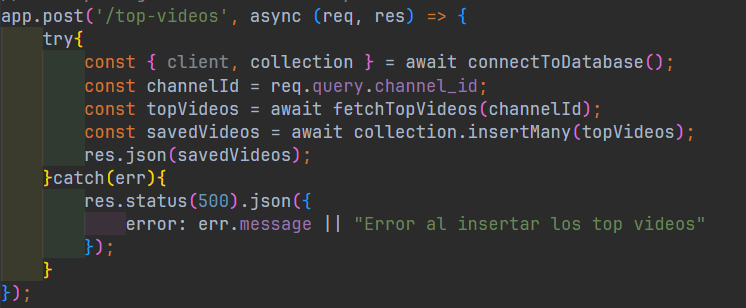


En el segundo GET he utilizado la conexión con la base de datos y he creado la variable videoId que recoja el id que introducimos en la request y una variable video que busque en mi base de datos ese id.



Para el post he hecho que se puedan introducir directamente los vídeos que encontremos en el top 10 a la base de datos. He vuelto a utilizar la conexión a la base de datos he utilizado el método de fetchtopvideos con la variable del canalId y he utilizado un insertMany a los topVideos que recupere el método.

Y al igual que en los demás he manejado cualquier error con el try y el catch.



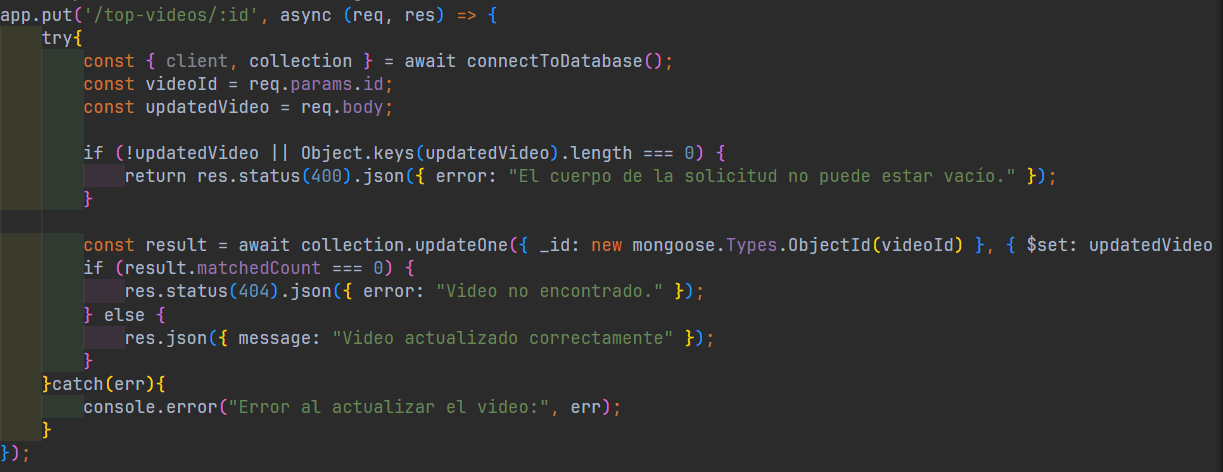
Lo siguiente es el PUT, que lo he definido de tal forma que según el id que le pasemos a la request nos actualice los datos con el contenido que escribamos en el body y pasemos por json. Para esto, he creado otra variable que recoja de la request el id, un updatevideo que guarde lo que mandemos por el body con formato json.

En caso de que esté vacío el body mandamos un mensaje avisando de esto.

Si hay contenido se utiliza el updateOne para actualizar la información de ese vídeo.

Si el vídeo no se encuentra por id salta un aviso, y si se actualiza también.

Por último, para controlar los errores he utilizado el try y catch.



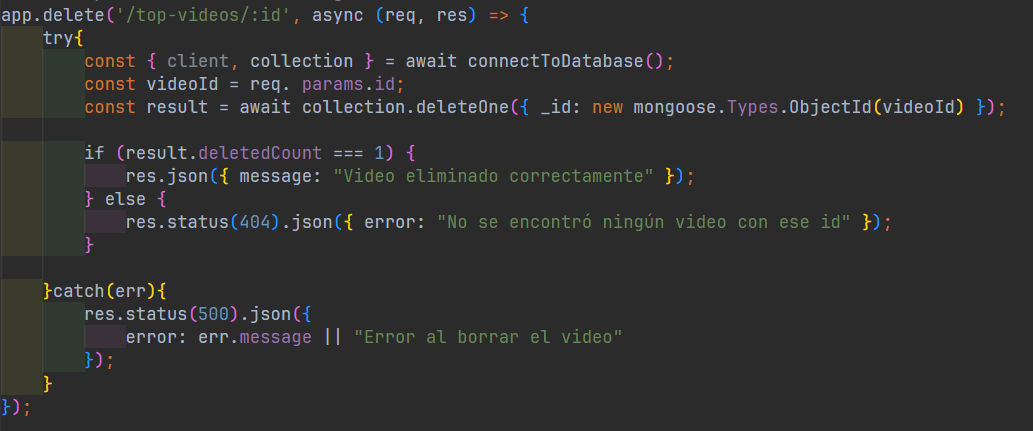
En cuanto a los DELETE he hecho dos de tal forma que el primero sea capaz de borrar un registro según el id, y el segundo que borre todo lo que haya en la colección.

El último aunque suena peligro lo he hecho para no tener que ir borrando los registros uno a uno para volver a hacer las pruebas del POST.

El primer delete tiene lo mismo que los demás, una variable que recoja la id del vídeo que se quiera borrar, y un result donde se utiliza el deleteOne para borrar ese registro según la id que recoja la variable anterior.

Para controlar que se hace correctamente he puesto un if para que si se haya borrado lance un mensaje y que si no encuentra nada con ese id nos avise.

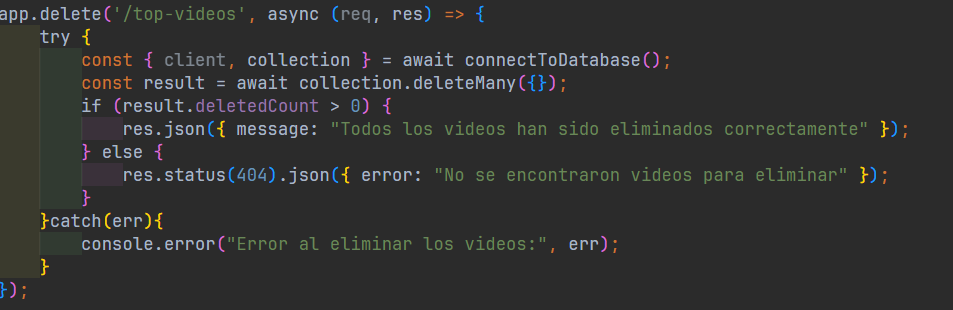
Y para terminar también he puesto un try y catch.



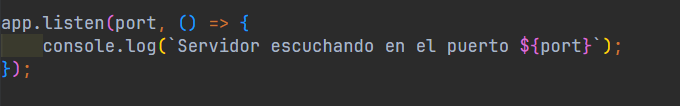
En el último DELETE la única variable que he utilizado ha sido la de result donde directamente utilizo el método de deleteMany para borrar todos los registros que haya.

Y defino un if que me avisa tanto de si se ha borrado todo correctamente como si no se ha encontrado nada.

Y por supuesto todo manejado con un try y catch.



Y ya para acabar he puesto a escuchar al puerto 3000 y he sacado un mensaje que nos avise de que está levantado y listo para usar.



**CONCLUSIONES / OBSERVACIONES**

El ejercicio en general me ha gustado bastante ya que nunca había utilizado ninguna API externa, y aunque me ha costado al principio manejarla y entender los parámetros que necesitaba para poder funcionar me ha gustado mucho poder sacar datos de la propia aplicación y guardarlas en una base de datos.

Lo que me ha costado más ha sido el que recogiera bien la API KEY que le estaba mandando y la parte de mandar los parámetros correctos que necesita youtube.list para devolverme la información.

Además, cuando estaba haciendo el put he tenido muchos problemas porque estaba mandando el body con los datos que quería actualizar pero no me di cuenta de que en el código me faltaba esta línea ‘app.use(express.json());’.

Pero por lo demás, el manejo con mongodb se me ha dado bastante bien.

Como observación creo que debería haber incluido más condiciones que se pueden dar como el que no haya dos registros con el mismo vídeo y sus mismos datos.

Te dejo el enlace del repositorio:  
**https://github.com/moreno025/NodeJS-UE**