Setup

Antes de empezar a realizar los problemas que aparecen a continuación, siga los siguientes pasos en orden usando la consola:

1. Entre en la carpeta `setup` y ejecute el comando `sh setup.sh`. Este script se encarga de la instalación de Mongodb en la máquina, incluyendo todos los comandos necesarios durante el examen (`mongosh`, `mongoimport`...).

2. Entre en la carpeta `setup` y ejecute el comando `sh db.sh`. Este script se encarga de cargar unos datasets en MongoDB que necesitarás durante el examen.

3. Comprueba que todo se ha ejecutado correctamente ejecutando `mongosh` y combrobando las colecciones cargadas.

Problema 1

La carpeta `api` contiene un servidor web que arranca una API REST que está incompleta. La especificación OpenAPI está en `schema/library.schema.yaml`.

Tenga en cuenta las siguientes consideraciones:

- Vamos a dejar que la base de datos gestione las id, con lo que usaremos _id como nuestras id tratándola como un string.

- Por simplicidad no se permite editar la información de los libros.

Complete los apartados que aparecen a continuación.

Apartado 1.

Actualmente la API no se está ejecutando en la ruta que está especificada en el documento OpenAPI. Modifique el servidor para que coincidan.

Apartado 2.

Actualmente la ruta GET /book está devolviendo la información completa de cada libro, pero eso no debería ser así. Modifique el servidor para que de cada libro se devuelva sólo la información especificada en el documento OpenAPI.

Apartado 3.

Queremos hacer nuestra API restful y para eso nos falta una parte muy importante, HATEOAS. Vamos a empezar a implementarlo en alguna de las rutas, pero no queremos modificar los datos que tenemos en la base de datos.

En `GET /book` añade a cada libro del array `results` un atributo `link` que enlace a la ruta completa de ese libro: `/book/{id}`

De forma que por ejemplo se devuelva lo siguiente (por simplicidad sólo se muestra un libro en los resultados y puede ser que la ruta no sea correcta del todo):

```
{
  "results": [
    {
```

```
      "_id": "646332b5b3767c0bcb5d4b3b",
      "title": "Speaking JavaScript",
      "author": "Axel Rauschmayer",
      "link": "localhost:3000/api/book/646332b5b3767c0bcb5d4b3b"
    }
  ],
  "next": null
}
```

Modifica el archivo OpenAPI para tener en cuenta esta modificación.

Apartado 4.

En la ruta DELETE /book/{id} no se están aplicando todas las respuestas definidas en la especificación OpenAPI. Modifique el servidor para que se tengan en cuenta todos los casos definidos.

Problema 2

Haciendo uso de mongosh complete el archivo problema2.txt con las operaciones que se piden en los siguientes apartados. Para cada una indique únicamente la instrucción realizada, no hace falta indicar la solución. De forma que la respuesta del apartado 1 quede por ejemplo como se muestra a continuación:

```
## Apartado 1
Instrucción
```

Tenga en cuenta que cada apartado se responde con una única consulta y que esta tiene que seguir siendo válida si se añaden o eliminan documentos de la colección.

Apartado 1.

En la colección `listingAndReviews` indique el/los nombre(s) del alojamiento con más reviews.

Apartado 2.

En la colección `listingAndReviews` indique el/los nombre(s) del alojamiento con más amenities.

Apartado 3.

En la colección `listingAndReviews` indique para cada tipo de `property_type` el número de alojamientos de ese tipo.

Apartado 4.

En la colección `listingAndReviews` indique el número de alojamientos que tienen 2, 3, 4 o 5 `beds`.

Tenga en cuenta las siguientes recomendaciones durante el examen:

- Asegúrese de hacer push de forma periódica y no lo deje para el final.
- Gestione los errores y las excepciones de forma adecuada.
- Asegúrese de que la versión de la aplicación que entrega compila y ejecuta.
- Antes de entregar el examen, asegúrese de que ha hecho al menos un push.