Prueba técnica para desarrollo back-end utilizando el Framework Spring

Diseñar e implementar una API Rest para cargar archivos y generar un registro del mismo en una Base de Datos (BD). Los archivos **no se guardarán**, sólo se guardará el registro de la subida del mismo con la utilización de una función hash y la salida de aplicar la función hash se debe almacenar en la BD.

Para esto se debe implementar un endpoint que permita cargar archivos a través de FORM-DATA y por parámetro en la URL poder determinar el algoritmo de hash a aplicar sobre los archivos.
Las funciones hash que se deben aplicar sobre los archivos pueden ser SHA-256 o SHA-512 del conjunto SHA-2. Además, si el archivo ya se subió previamente, es decir, ese hash ya está presente en la BD, se debe guardar la fecha de última vez en la que se cargó el archivo en una variable llamada “lastUpload”.

En la BD se debe guardar:

- el nombre del archivo en una columna “fileName”
- el hash con el algoritmo SHA-256 en una columna “hash-sha-256”
- el hash con el algoritmo SHA-512 en una columna “hash-sha-512”
- La fecha de la última vez que se cargó el archivo “lastUpload”, solo si el archivo fue cargado previamente.

Especificaciones del endpoint:

URL	localhost:8080/api/documents/hash
Method	POST
Query String	**hashType** → Valores posibles: “SHA-256” o “SHA-512”
Body (multipart/form-data)	**documents** → Lista de documentos
Response Status	**201** → Se guardó correctamente el registro de archivos **400** → Se produjo un error (No se subieron archivos o el se pasó mal el parámetro)

Respuesta del endpoint para subir documentos:

```
{
  "algorithm": "SHA-256",
  "documents": [
    {
      "fileName": "test.txt",
      "hash": "8cc6618520b943d7a32a69899f5c68ad2d43eae6e211d2c646b89cef4e7137f2"
    },
    {
      "fileName": "test2.txt",
      "hash": "8cc6618520b943d7a32a69899f5c68ad2d43eae6e211d2c646b89cef4e7137f3",
      "lastUpload": "2022-04-27T15:21:53.678Z"
    }
  ]
}
```

Además del endpoint para subir documentos, también será necesario implementar un endpoint que permita listar todos los documentos subidos con sus hash, que siga las siguientes especificaciones:

Especificaciones del endpoint:

URL	localhost:8080/api/documents
Method	GET
Response Body	Lista de documentos
Response Status	**200** → Se devolvieron correctamente todos los documentos

Respuesta del endpoint para listar todos los documentos:


```
[
    {
      "fileName": "test.txt",
      "hash-sha-256": "8cc6618520b943d7a32a69899f5c68ad2d43eae6e211d2c646b89cef4e7137f2",
      "hash-sha-512": "h74s33e3b22b159ef0d93e5947a7e28045b1a8f1452287b3c25c07312527519eef871
       9333f86080fc5e969e49aaa6faaa911cb7a64e1f938fb7b058ac4c6fyh6"
    },
    {
      "fileName": "test2.txt",
      "hash-sha-256": "dfg6618520b943d7a32a69899f5c68ad2d43eae6e211d2c646b89cef4e7137f3",
      "hash-sha-512": "g35233e3b22b159ef0d93e5947a7e28045b1a8f1452287b3c25c07312527519eef871
       9333f86080fc5e969e49aaa6faaa911cb7a64e1f938fb7b058ac4c6g74d",
      "lastUpload": "2022-04-27T15:21:53.678Z"
    },
    {
      "fileName": "test3.txt",
      "hash-sha-256": "abc6618520b943d7a32a69899f5c68ad2d43eae6e211d2c646b89cef4e7137f4",
      "hash-sha-512": "7a0133e3b22b159ef0d93e5947a7e28045b1a8f1452287b3c25c07312527519eef871
       9333f86080fc5e969e49aaa6faaa911cb7a64e1f938fb7b058ac4c62567",
      "lastUpload": "2022-04-28T15:21:53.678Z"
    }
]
```


Finalmente, se deberá hacer un endpoint para encontrar un archivo por su hash y devolver su hash y última actualización, estas son sus especificaciones:

Especificaciones del endpoint:

URL	localhost:8080/api/documents
Method	GET
Query String	**hashType** → Valores posibles: “SHA-256” o “SHA-512” **hash** → Valor del hash
Response Body	Documento
Response Status	**200** → Se devolvió el documento **404** → No hay ningún documento con ese nombre

Respuesta del endpoint para buscar un documento por su hash:

```
{
      "fileName": "test2.txt",
      "hash": "dfg6618520b943d7a32a69899f5c68ad2d43eae6e211d2c646b89cef4e7137f3",
      "lastUpload": "2022-04-27T15:21:53.678Z"
}
```

Consideraciones a tener en cuenta

- Se debe utilizar PostgreSQL como BD
- Se debe utilizar separación de lógica por capas (Controller, Service, Repository,
- Model, Utils, etc.)
- Se pueden utilizar librerías externas
- Se debe manejar los errores que se puedan generar devolviendo una respuesta
- JSON con el siguiente formato:

```
{
   error: {
      "timestamp": 1651081923915,
      "status": 400,
      "message": "El parámetro ‘hash’ solo puede ser ‘SHA-256’ o ‘SHA-512’",
      "path": "/api/hash"
   }
}
```