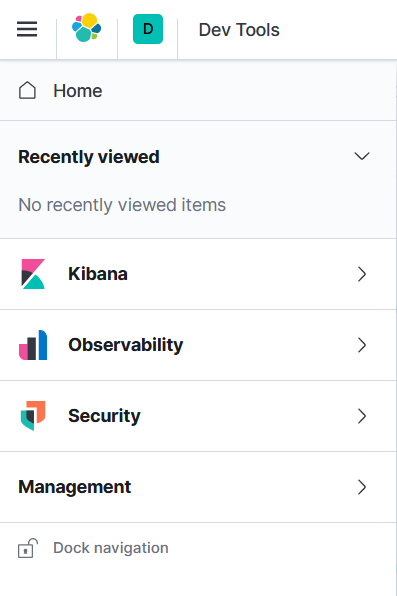
**Creación de Índices en Kibana**

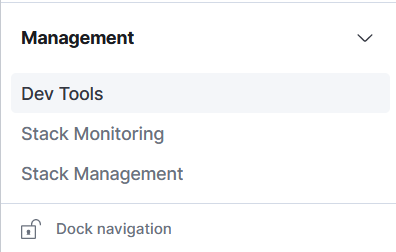
1. Primero nos dirigimos a la url <https://cloud.elastic.co/home> en donde se creo el deployment sps\_practica.
2. Damos clic sobre el link de **Kibana**



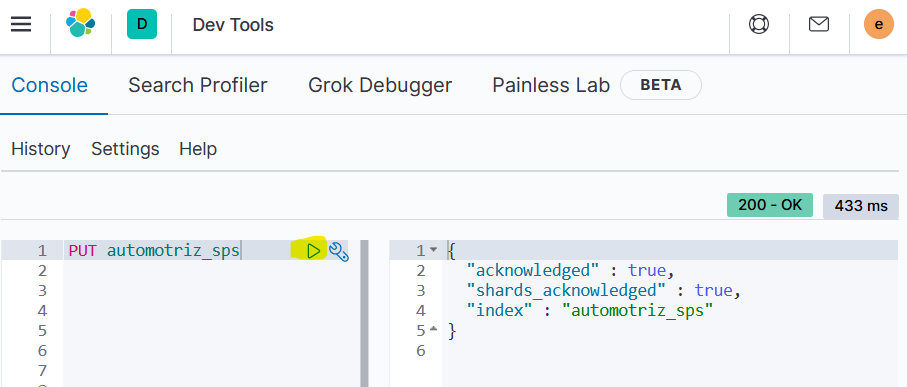
1. En la siguiente ventana nos dirigimos al menú desplegable de la esquina superior izquierda y da clic sobre la opción **Management**.



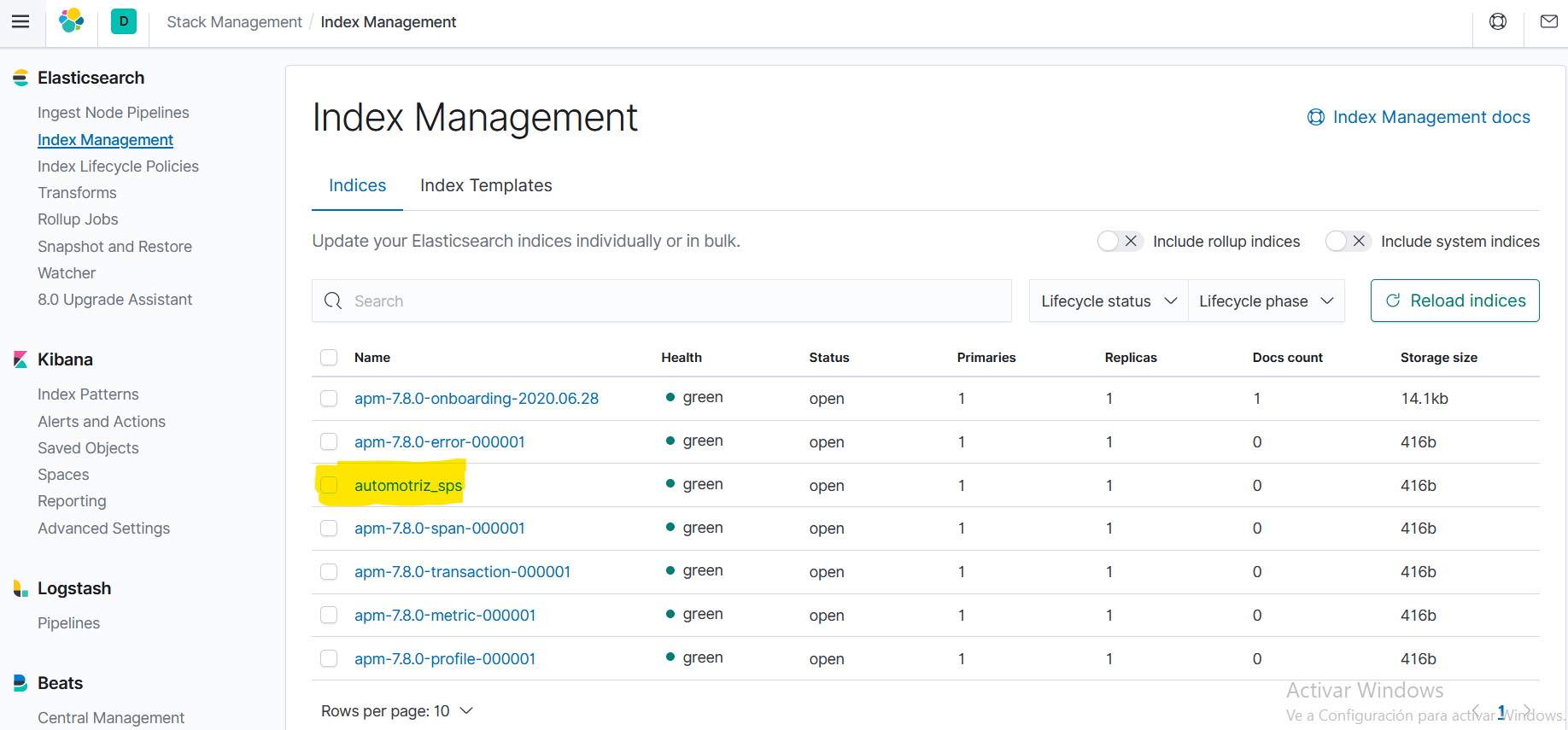
1. Posteriormente se da clic sobre la opción DevTools



1. En la nueva ventana podremos crear los índices que deseamos.
2. Para crear el índice, ingresamos en la pantalla las siguientes líneas
   1. PUT automotriz\_sps , y damos clic en el botón de enviar request(botón en forma de triángulo marcado en amarillo en la imagen) esto nos retornara un mensaje en estructura JSON donde nos indicara que se creó el índice de forma exitosa.

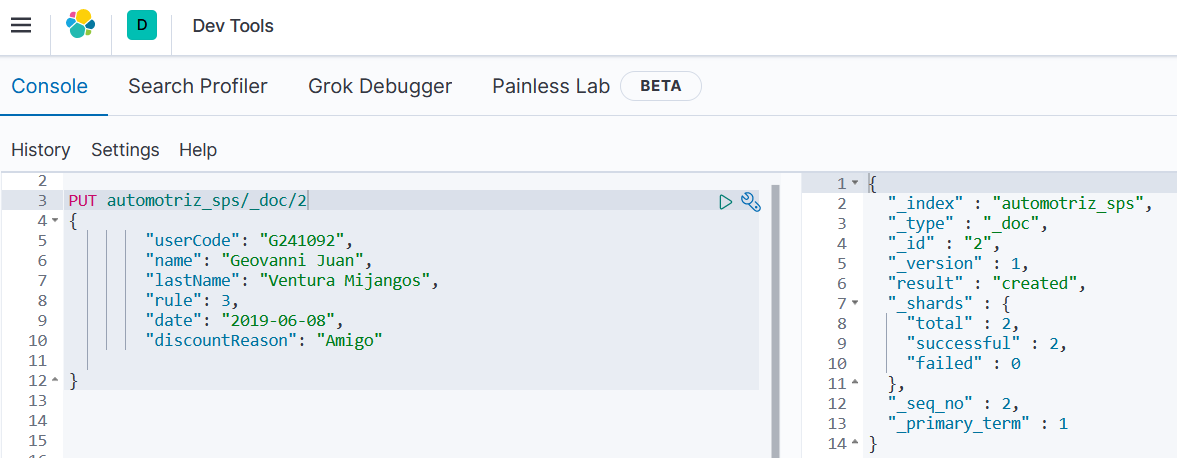


1. Puedes conformar la creación del índice dirigiéndote al menú y dando clic en Management> Stack Management > Index Management. En esta ventana podrás verificar que existe el índice creado y posteriormente verificar que su estado este en verde para poder ser utilizado de forma correcta.

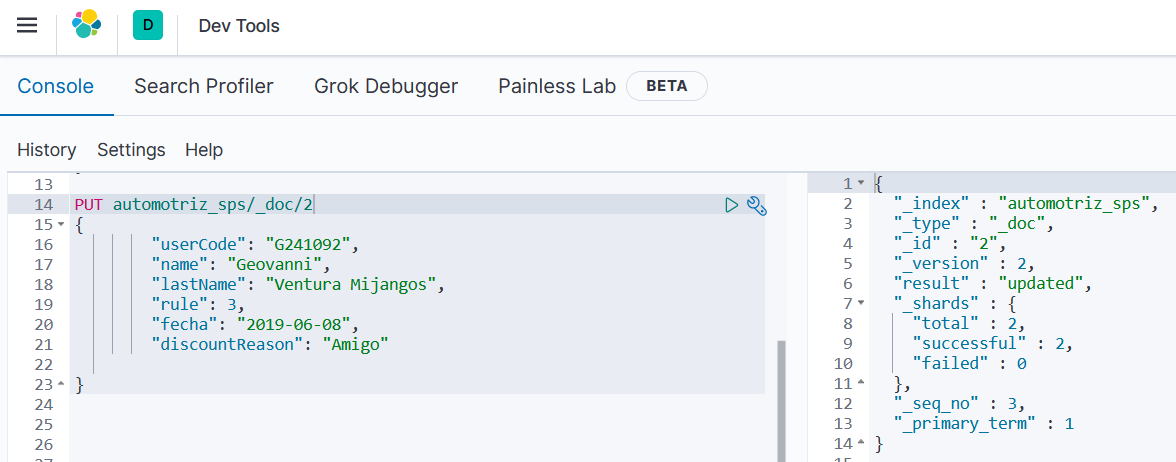


1. Para agregar registros o documentos al índice ingresamos la siguiente línea
   1. PUT /<nombre del indice>/\_doc/<identificador del documento>

A estas líneas se agrega la estructura que se requiera para el índice, para el caso de esta practica se agrega como se muestra en la imagen. También nos retornara un mensaje de respuesta en el cual nos indicara si se agrego correctamente el registro.

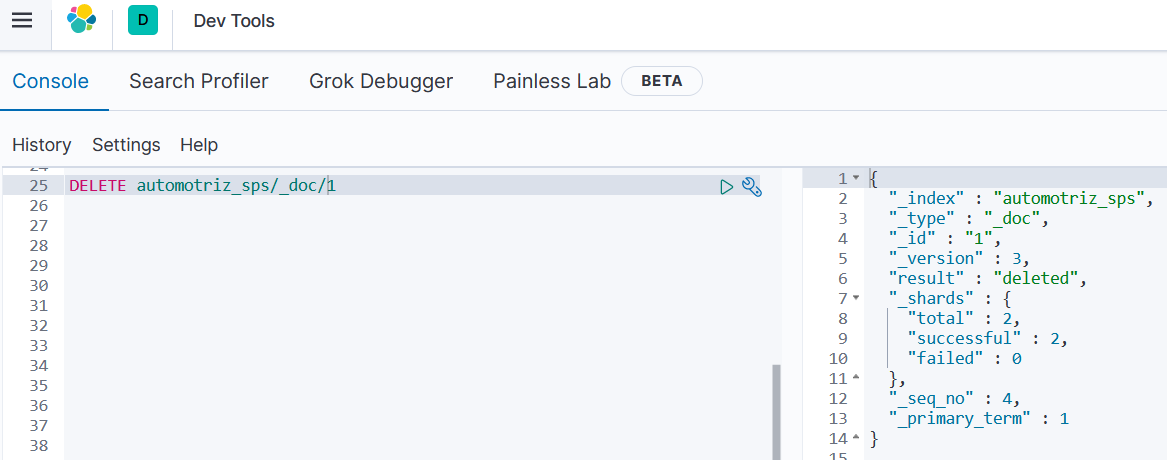


1. Para la actualización se sigue el mismo procedimiento que para agregar solo indicando el id del registro a modificar ya que de lo contrario al no encontrar el registro lo creara.

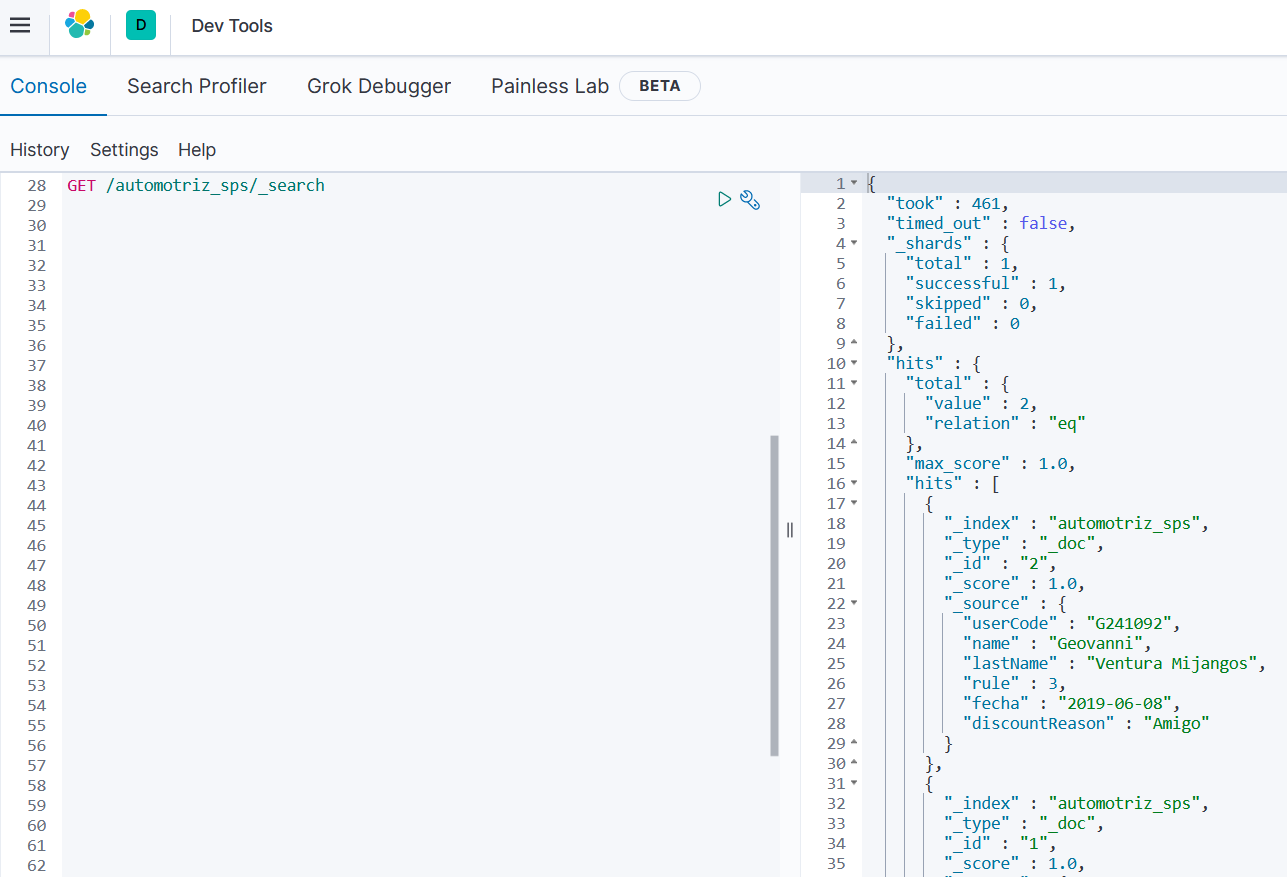


1. En el caso de la eliminación se agregan las siguientes líneas
   1. DELETE /<nombre del indice>/\_doc/<identificador del documento>

En este caso de no encontrar el documento nos indicara que el registro no fue encontrado.



1. Por último para obtener todos los registros en el índice usamos las siguientes líneas
   1. GET /<nombre del indice>/\_search



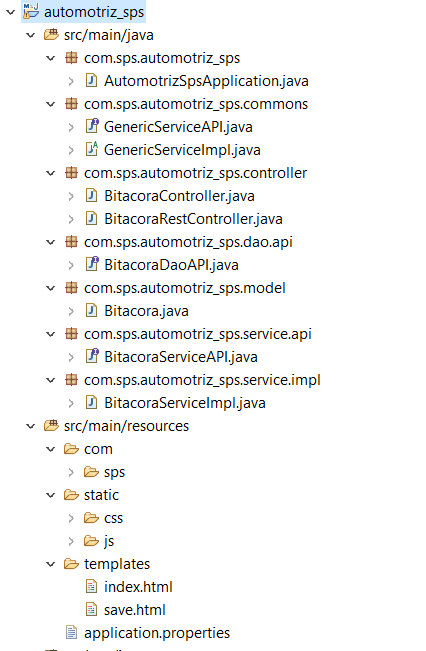
API REST

Para la creación del API Rest se decidió realizarlo con el lenguaje Java y la infraestructura de SpringBoot. Se planteo la necesidad de un registro de los usuarios que realizaban cambios en las reglas de descuento de una automotriz, específicamente que se pudiera realizar la creación de registros, actualización, eliminación y obtención de la bitácora de movimientos.

Para este caso se realizo estas operaciones simulando dicha actividad, es decir que si en algún momento se requiere terminar esta funcionalidad es funcional, tendrá que adaptarse a la funcionalidad completa, pero de momento únicamente cumple con el CRUD básico solicitado.

El proyecto con spring cuenta con la siguiente estructura:

* una clase main que es la que inicializa el API
* cuenta con un paquete de archivos commons que son clases genéricas de un crud básico, esto se realizo de esta manera ya que si quisiéramos hacer un CRUD básico de otra actividad únicamente tendríamos que hacer referencia a estas clases.
* Un paquete de controladores en donde tenemos la lógica de las diferentes peticiones.
* Un paquete de archivos dao que nos dan la interfaz común en la API.
* Un paquete de modelos donde se define la representación de nuestros campos, es decir las variables y tipos de datos que se enviaran o retornaran del mismo.
* Para la parte de la vista se implementa dos archivos que son dos vistas una que muestra dos datos existentes y otra que genera o altera los datos. Estas vistas si realizan con las librerías de Bootstrap.
* Finalmente tenemos el application.properties, este archivo nos apoya para realizar la conexión ya sea a la Base de datos o en este caso a los índices que creamos anteriormente.



Para la creación del contenedor se crea el archivo Dockerfile donde se agregan las siguientes sentencias:

FROM openjdk:8-jdk-alpine

VOLUME /tmp

EXPOSE 8080

ARG JAR\_FILE=/target/automotriz\_sps-0.0.1-SNAPSHOT.jar

ADD ${JAR\_FILE} app.jar

ENTRYPOINT ["java","-Djava.security.egd=file:/dev/./urandom","-jar","/app.jar"]

Ya que para el contenedor se necesita el archivo .jar del api se crea mediante la herramienta dando clic derecho sobre el proyecto y seleccionando Maven install.

Para construir el contenedor en la consola se ingresan los siguientes comandos:

**docker build -t automotriz\_sps .**

Posteriormente para comprobar la creacion se teclea el siguiente comando:

**docker images**

finalmente, para ejecutar el contenedor es con el siguiente comando:

**docker run -it -p 8080:8090 automotriz\_sps.**