

Noms : - Berrissoul Saad  
- El Fallah Mohamed

DATA-INE2

Encadré par : Ziyati Houssaine

## Projet Scala

- SBT est un outil de génération open source pour les projets Scala. Ses principales caractéristiques sont : Prise en charge native de la compilation du code Scala et de l'intégration avec de nombreux frameworks de test Scala Compilation, test et déploiement continu.

On ajoute les dépendances relatives à SPARK nécessaires pour notre projet :

```
build.sbt x
1  ThisBuild / version := "0.1.0-SNAPSHOT"
2
3  ThisBuild / scalaVersion := "2.12.10"
4
5  lazy val root = (project in file("."))
6    .settings(
7      name := "simple"
8    )
9
10 libraryDependencies ++= Seq(
11   "org.apache.spark" %% "spark-core" % "2.4.6",
12   "org.apache.spark" %% "spark-sql" % "2.4.6",
13   "org.apache.spark" %% "spark-mllib" % "2.4.6",
14   "org.apache.spark" %% "spark-streaming" % "2.4.6" % "provided",
15   "org.scala-sbt" %% "util-logging" % "1.3.0-M2",
16   "org.elasticsearch" %% "elasticsearch-spark-20" % "7.16.2"
17 )
```

- On importe les librairies nécessaires au début de notre script SCALA :

```
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.types._
import org.apache.spark.streaming
import org.apache.spark.sql.functions._
import org.apache.log4j._
import org.elasticsearch.spark.sql._
```

- SparkSession est le point d'entrée de Spark SQL. On crée une SparkSession à l'aide de SparkSession. La méthode builder nous donne accès à l'API Builder qu'on utilise pour configurer la session, ainsi qu'on va utiliser tous les cœurs de notre machine à l'aide de la méthode config(master = local[\*]) :

```
object Main {

  def main(args: Array[String]): Unit = {

    val spark = SparkSession.builder().master("local[*]").appName("first")
      .config("spark.es.nodes", "localhost")
      .config("spark.es.port", "9200")
      .getOrCreate()

    Logger.getLogger("org").setLevel(Level.ERROR)
  }
}
```

- On simule la provenance des logs en temps réels à l'aide d'un script python, supposons qu'on se situe au sein d'une organisation et qu'on veut savoir les sites web les plus fréquentés par le personnel.

Ensuite on définit la structure de notre Dataframe, et on laisse spark écouter dans une repository ou les logs proviennent en temps réels :

```
//Data Structure
val schema = StructType(List(
  StructField("Protocol", StringType, true),
  StructField("HTTP status", StringType, true),
  StructField("URL", StringType, true),
  StructField("Path", StringType, true),
  StructField("IP Adress", StringType, true)))

//Create DataFrame
val StreamDF = spark.read.option("delimiter", "").schema(schema)
  .csv( path = "/home/hdsaad/Documents/simple/output")
StreamDF.createOrReplaceTempView( viewName = "SDF")
val outDF = spark.sql( sqlText = "select * from SDF")
```

- On transfère le Dataframe en mode streaming vers Elasticsearch qui tourne sur le port 9200 de notre machine sous l'index data :

```
//write DF to elasticSearch
outDF.write
  .format( source = "org.elasticsearch.spark.sql")
  .option("es.port", "9200")
  .option("es.nodes", "localhost")
  .mode( saveMode = "append")
  .save( path = "data/doc")
}
```

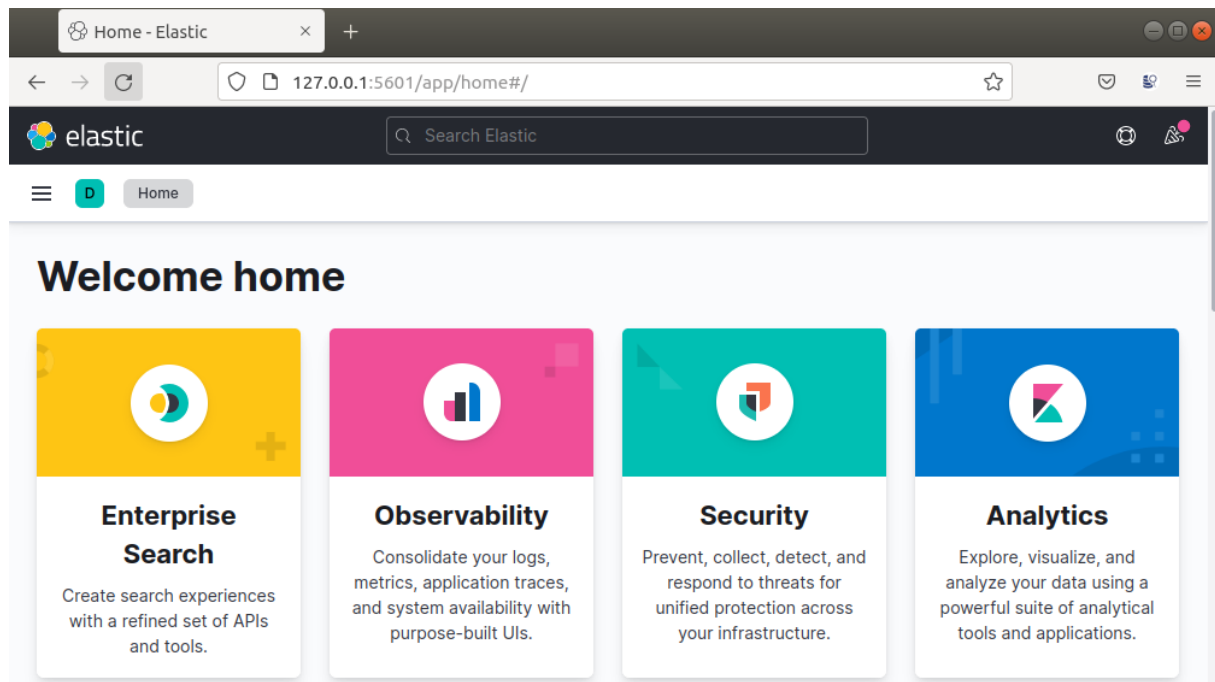
- On lance ElasticSearch, ce dernier est un moteur de recherche et d'analyse RESTful distribué, conçu pour répondre à une multitude de cas d'utilisation :

```
hdsaad@vm: ~/Downloads/elasticsearch-7.16.3
File Edit View Search Terminal Help
hdsaad@vm:~/Downloads$ cd elasticsearch-7.16.3/
hdsaad@vm:~/Downloads/elasticsearch-7.16.3$ ./bin/elasticsearch ←
warning: usage of JAVA_HOME is deprecated, use ES_JAVA_HOME
Future versions of Elasticsearch will require Java 11; your Java version fr
om [/opt/java/jdk1.8.0_71/jre] does not meet this requirement. Consider swi
tching to a distribution of Elasticsearch with a bundled JDK. If you are al
ready using a distribution with a bundled JDK, ensure the JAVA_HOME environ
ment variable is not set.
warning: usage of JAVA_HOME is deprecated, use ES_JAVA_HOME
Future versions of Elasticsearch will require Java 11; your Java version fr
om [/opt/java/jdk1.8.0_71/jre] does not meet this requirement. Consider swi
tching to a distribution of Elasticsearch with a bundled JDK. If you are al
ready using a distribution with a bundled JDK, ensure the JAVA_HOME environ
ment variable is not set.
```

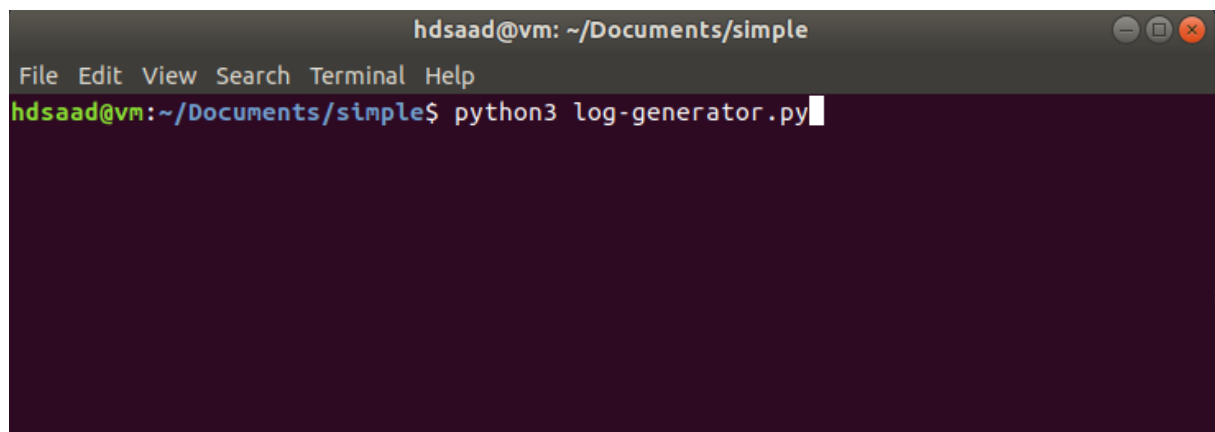
- Après on lance Kibana , l'outil qu'on va utiliser pour visualiser les données et restituer les résultats sous forme d' un Dashboard :

```
hdsaad@vm: ~/Downloads/kibana-7.16.3-linux-x86_64
File Edit View Search Terminal Help
hdsaad@vm:~/Downloads$ clear
hdsaad@vm:~/Downloads$ cd kibana-7.16.3-linux-x86_64/
hdsaad@vm:~/Downloads/kibana-7.16.3-linux-x86_64$ ./bin/kibana ←
log [20:03:40.500] [info][plugins-service] Plugin "metricsEntities" is disab
led.
log [20:03:40.618] [info][server][Preboot][http] http server running at http
://localhost:5601
log [20:03:40.717] [warning][config][deprecation] Starting in 8.0, the Kiban
a logging format will be changing. This may affect you if you are doing any spec
ial handling of your Kibana logs, such as ingesting logs into Elasticsearch for
further analysis. If you are using the new logging configuration, you are alread
y receiving logs in both old and new formats, and the old format will simply be
going away. If you are not yet using the new logging configuration, the log form
at will change upon upgrade to 8.0. Beginning in 8.0, the format of JSON logs wi
ll be ECS-compatible JSON, and the default pattern log format will be configurab
le with our new logging system. Please refer to the documentation for more infor
mation about the new logging format.
```

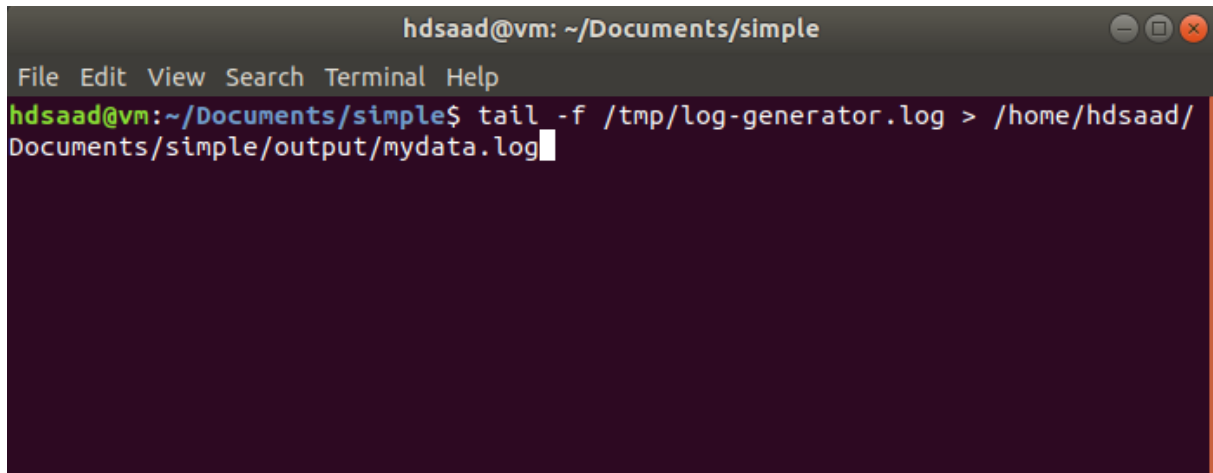
## - Vue général pour Elasticsearch :



## - On exécute le script Python nécessaire pour la génération aléatoire des logs :

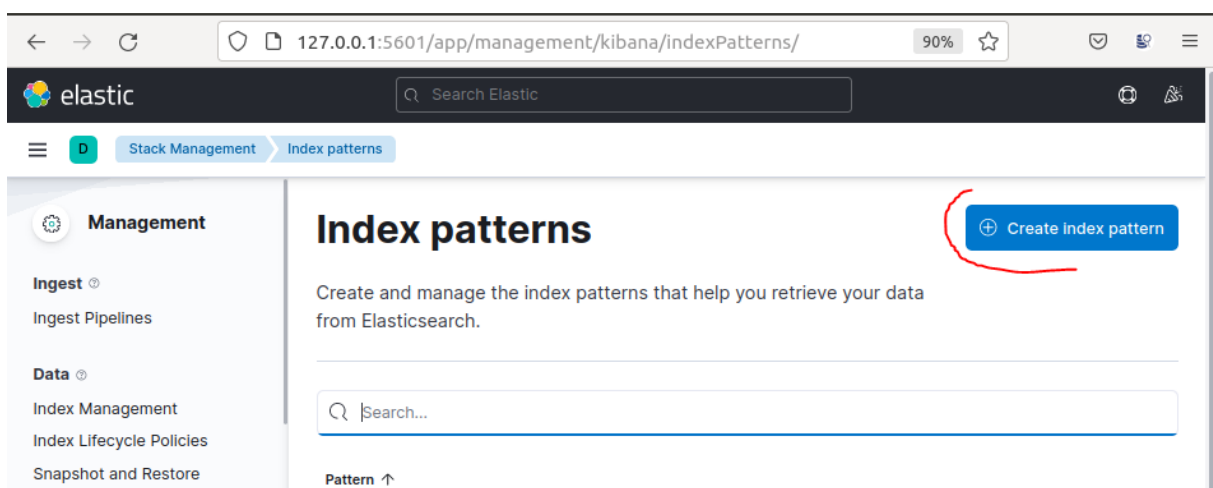


- On transfère les logs générés vers la repository ou spark est en train d'écouter :



```
hdsaad@vm: ~/Documents/simple
File Edit View Search Terminal Help
hdsaad@vm:~/Documents/simple$ tail -f /tmp/log-generator.log > /home/hdsaad/
Documents/simple/output/mydata.log
```

- Sous kibana, on crée un index pattern avec le nom prédéfini dans le script SCALA :



- On insère le nom de l'index : data

elastic

Stack Management Index patterns

### Create index pattern

**Name**

data\*

Use an asterisk (\*) to match multiple characters. Spaces and the characters , / ? \* < > | are not allowed.

**Timestamp field**

Select a timestamp field

No matching data stream, index, or index alias has a timestamp field.

[Show advanced settings](#)

[Close](#) **Create Index pattern**

✓ Your index pattern matches 1 source.

data Index

Rows per page: 10

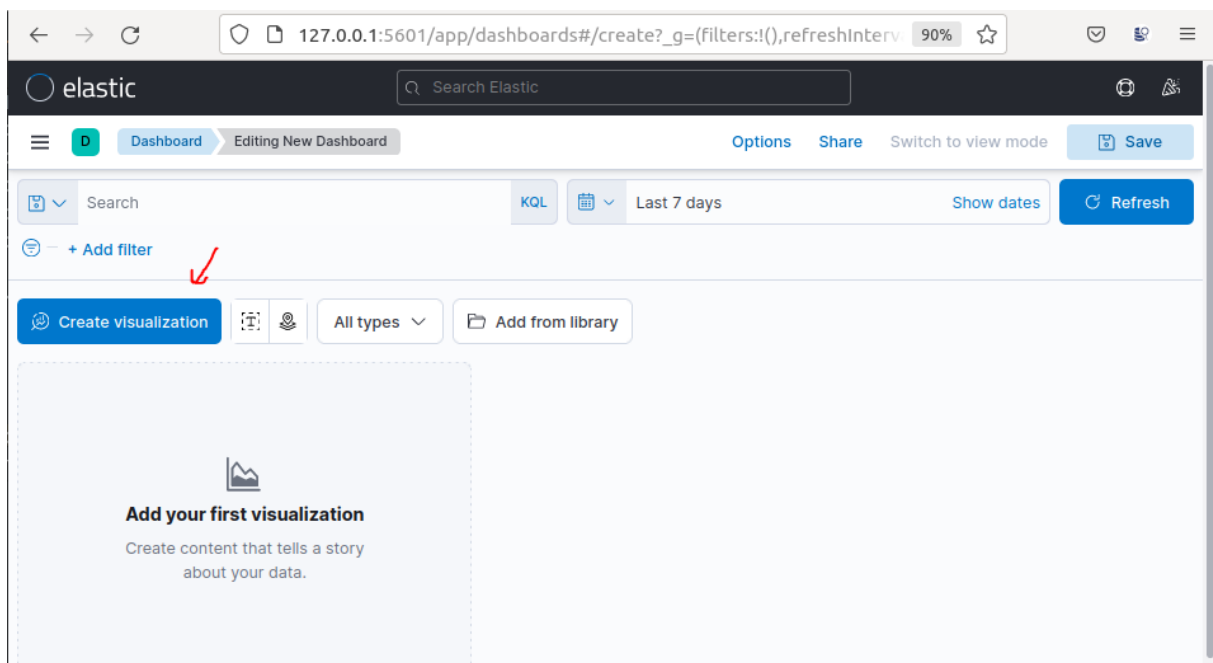
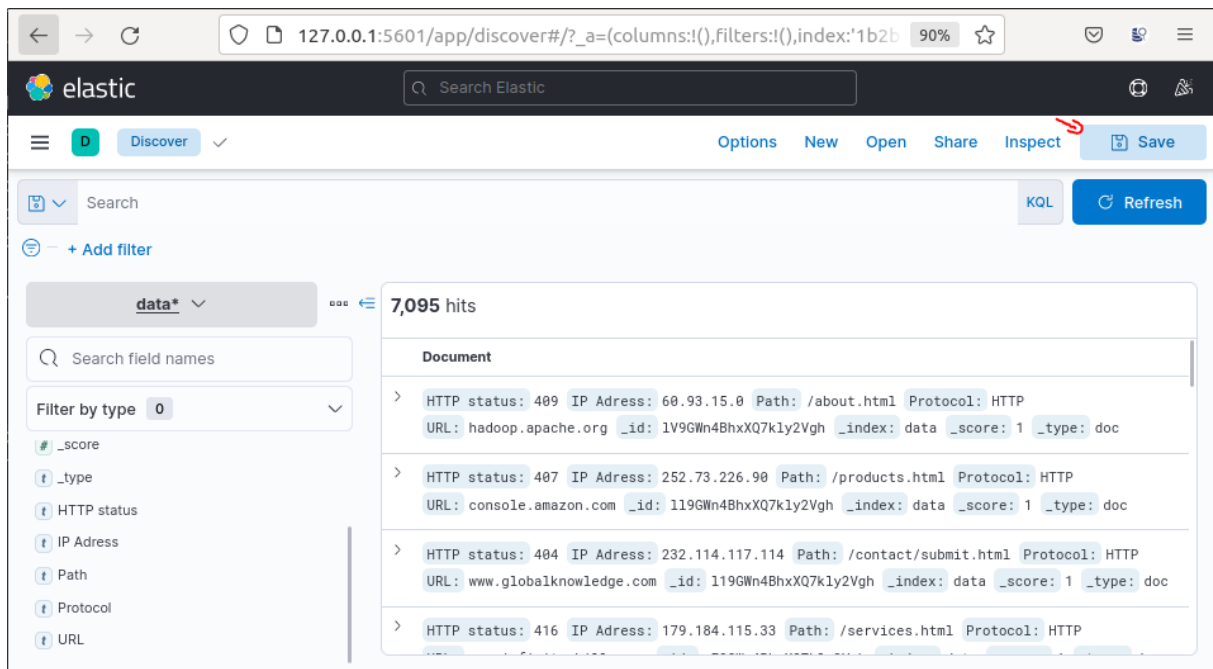
- On voit que les données ont été bien récupérées par kibana :

elastic

Stack Management Index patterns data\*

Name ↑	Type	Format	Searcha...	Aggrega...	Excluded
HTTP status	text		●		
HTTP status.keyword	keyword		●	●	
IP Address	text		●		
IP Address.keyword	keyword		●	●	
Path	text		●		
Path.keyword	keyword		●	●	
Protocol	text		●		
Protocol.keyword	keyword		●	●	
URL	text		●		
URL.keyword	keyword		●	●	

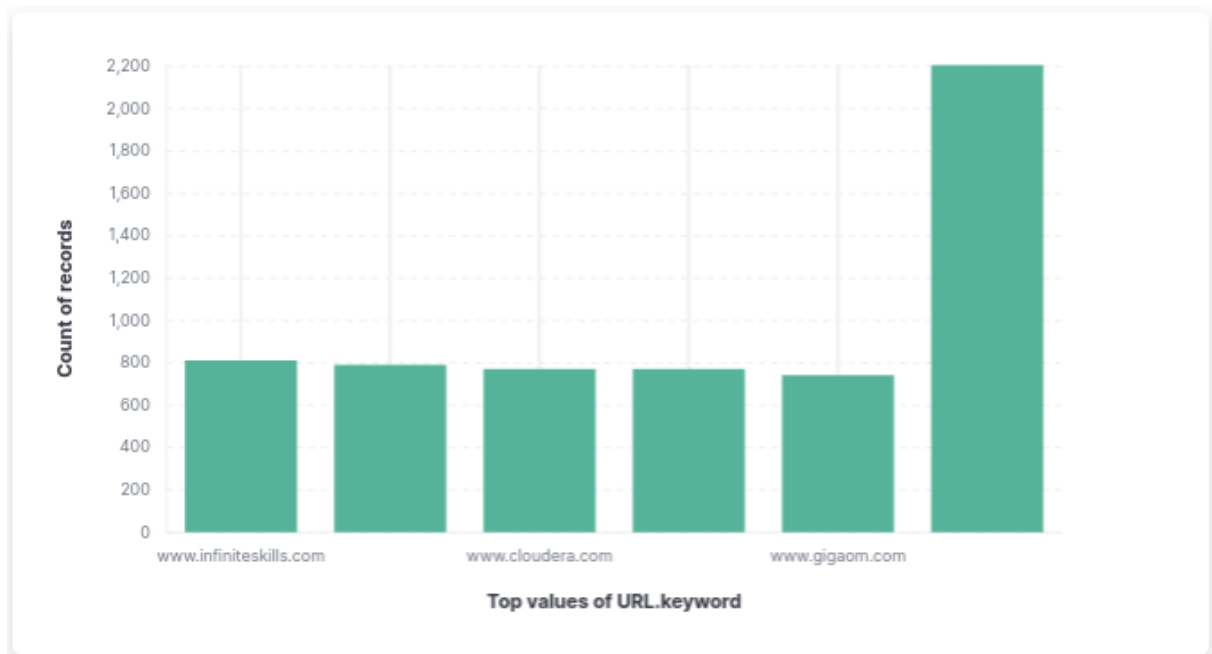
- Jusqu' à ce moment, on est arrivé à récupérer 7095 enregistrements, on va essayer de faire des visualisations sur cet échantillon :



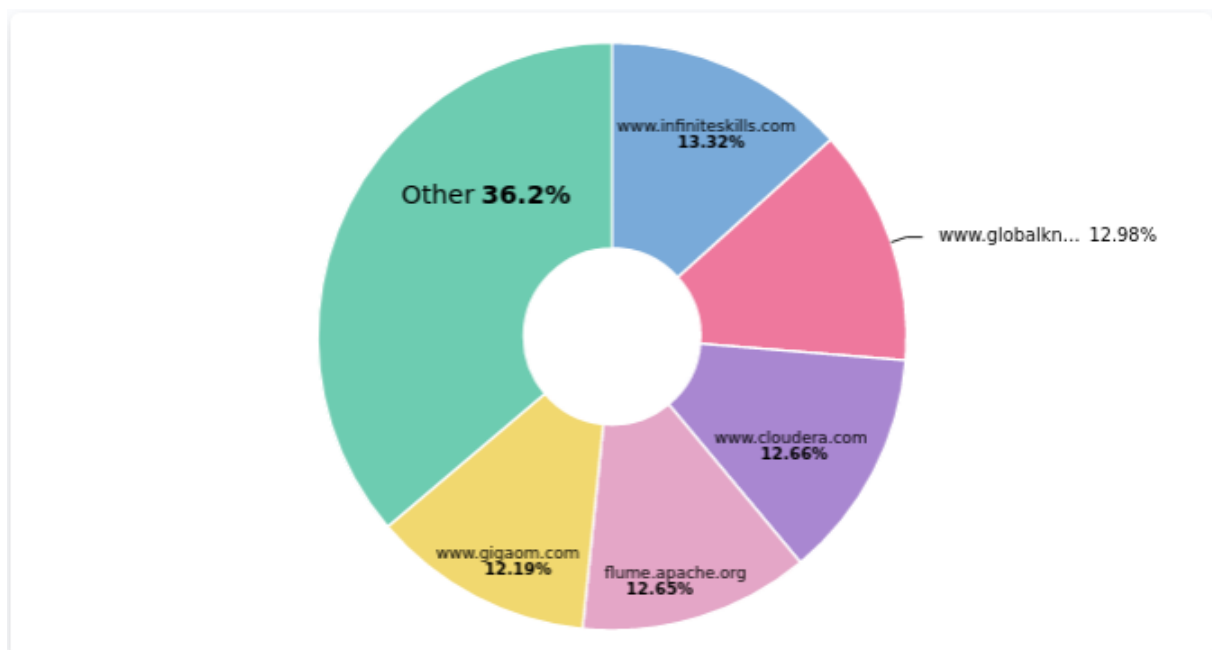


## - Visualisation des données :

### 1- Sites web les plus fréquentés sous forme de bars :



### Sous forme de donut :

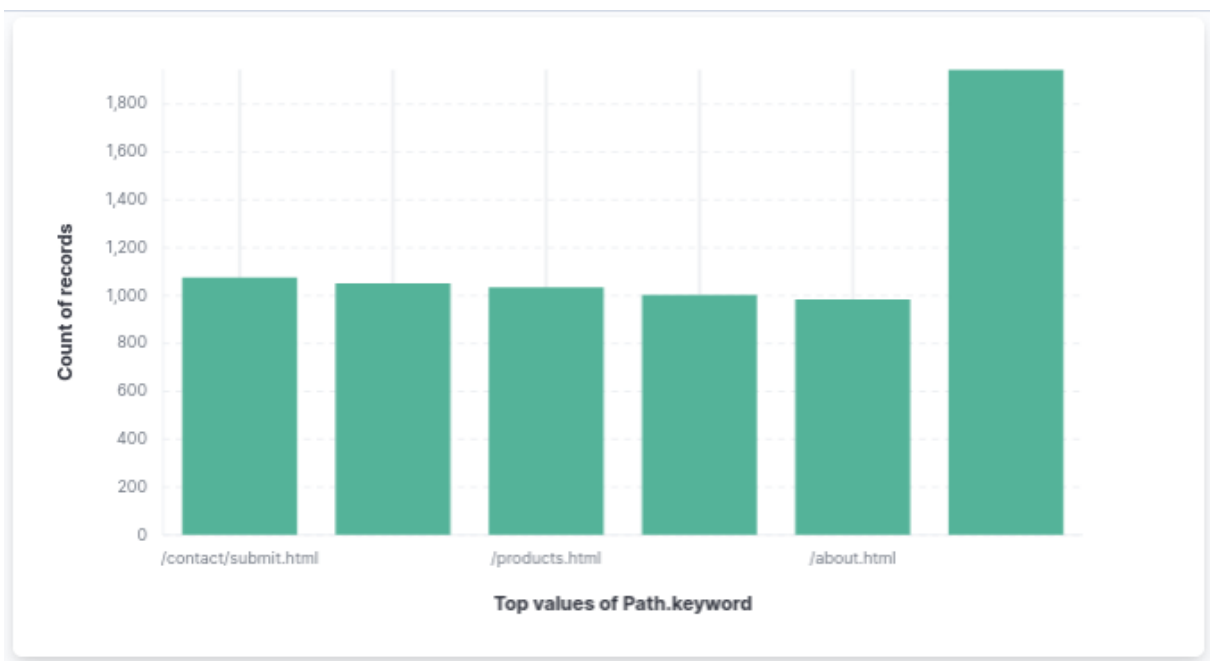


Alors [www.infineteskills.com](http://www.infineteskills.com) est le plus fréquenté par le personnel de l'organisation.

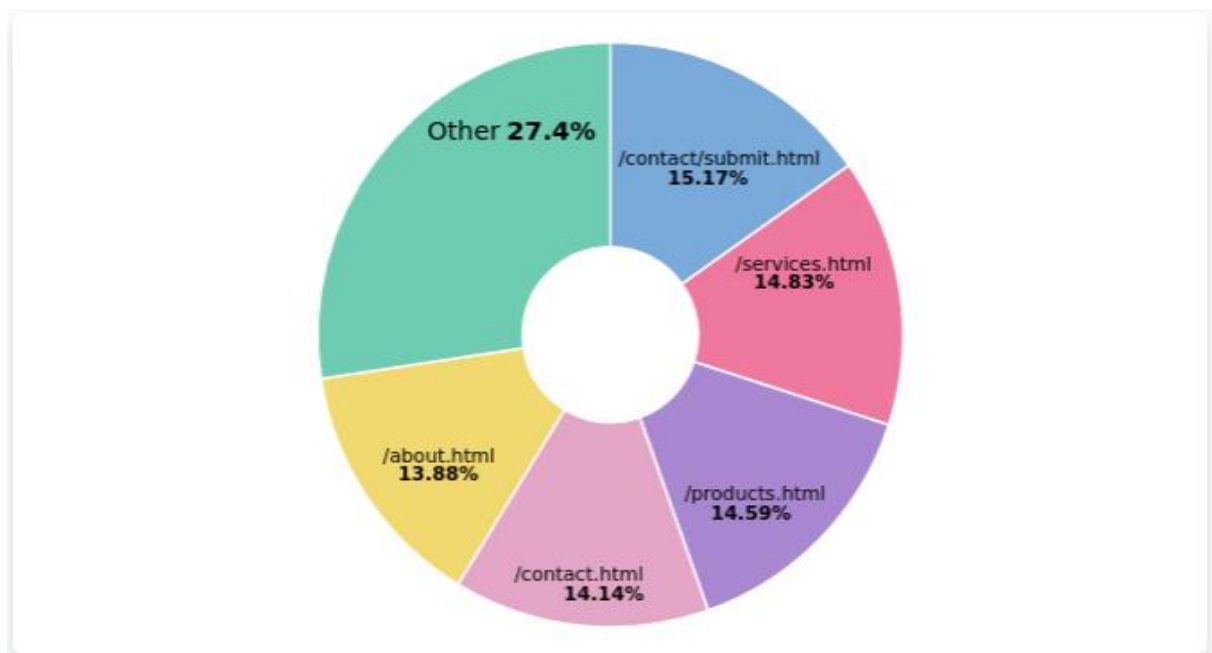
Une table de nombre de visites pour chaque site :

Top values of URL.keyword	Count of records
www.infiniteskills.com	812
www.globalknowledge.com	791
www.cloudera.com	772
flume.apache.org	771
www.gigaom.com	743
Other	2,207

2- Les index les plus utilisés sous forme de bars :



Sous forme de donut :



**/contact/submit.html** est le plus utilisés.

Table de comptage :

Table

Top values of Path.keyword	Count of records
/contact/submit.html	1,076
/services.html	1,052
/products.html	1,035
/contact.html	1,003
/about.html	985
Other	1,944

### 3- HTTP status reçus :

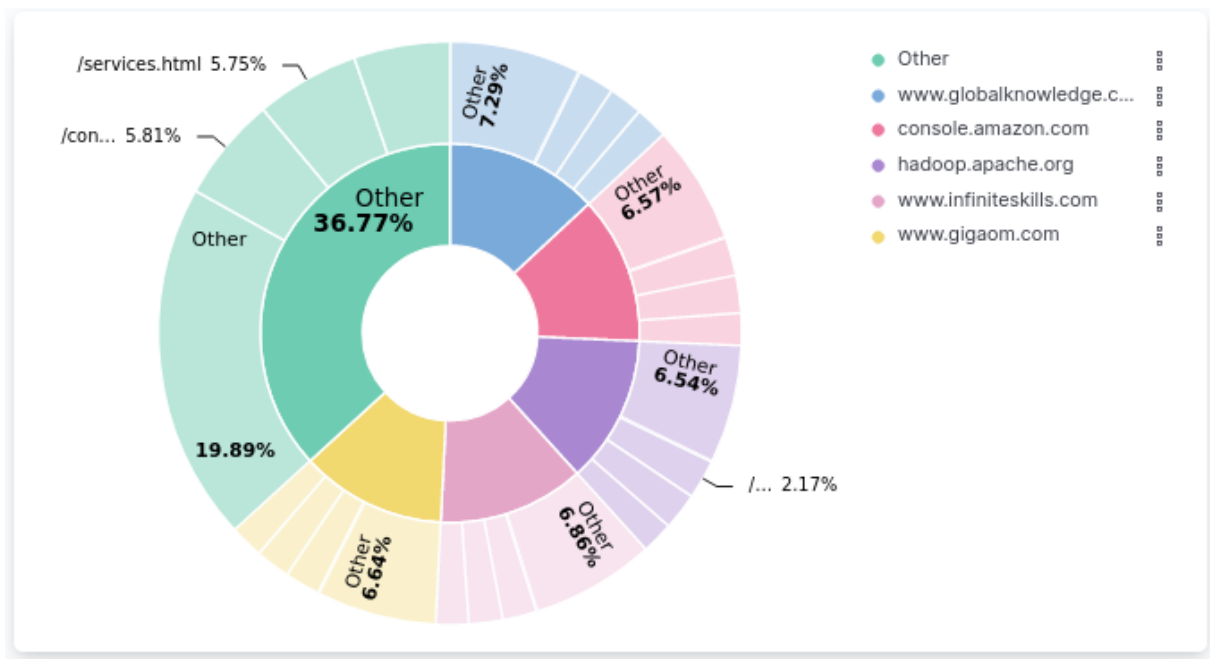
Table	
Top values of HTTP status.keyword	Count of records
505	176
415	167
305	164
401	163
404	161
Other	6,264

### 4- Adresses IP :

Table	
Top values of IP Adress.keyword	Count of records
0.102.105.143	1
0.116.22.98	1
0.121.28.101	1
0.122.194.176	1
0.123.173.67	1
Other	7,090

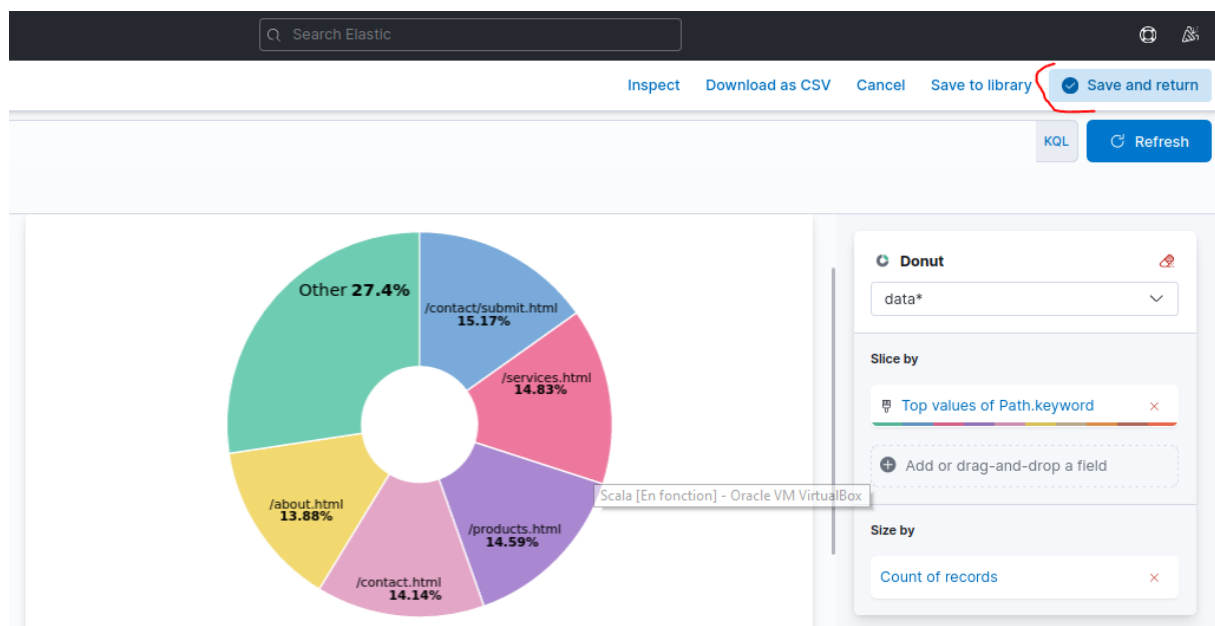
On constate que le max de records pour chaque adresse IP est 1, donc on conclue que chaque utilisateur a effectué une seule visite sur un seul site web, 7095 machines différentes dans notre organisation.

## 5- Vision globale pour les visites sur les sites web sous forme de donut :



## - Création du Dashboard :

Pour chaque visualisation, on sauvegarde le résultat en cliquant sur le bouton : Save and return



A fin qu' on puisse consulter notre Dashbord une autre fois :

