TP5 - Suivi des logs de production

Infrastructure de production

SUIVI DES LOGS

L'objectif de ce TP est de mettre en place une infrastructure pour suivre en temps réel les logs des applications. Ce TP doit être réalisé uniquement sous Docker .

Les éléments de notation

Répondez au questionnaire

Installation de la plateform

Le générateur de log et les données de géolocalisation sont issus d'un workshop donné par xebia : https://github.com/xebia-france/workshop-kibana

1. télécharger Logstash

```
docker pull docker.elastic.co/logstash/logstash:7.4.0
```

2. installer Elasticsearch

```
docker pull docker.elastic.co/elasticsearch/elasticsearch:7.4.0
```

3. télécharger Kibana

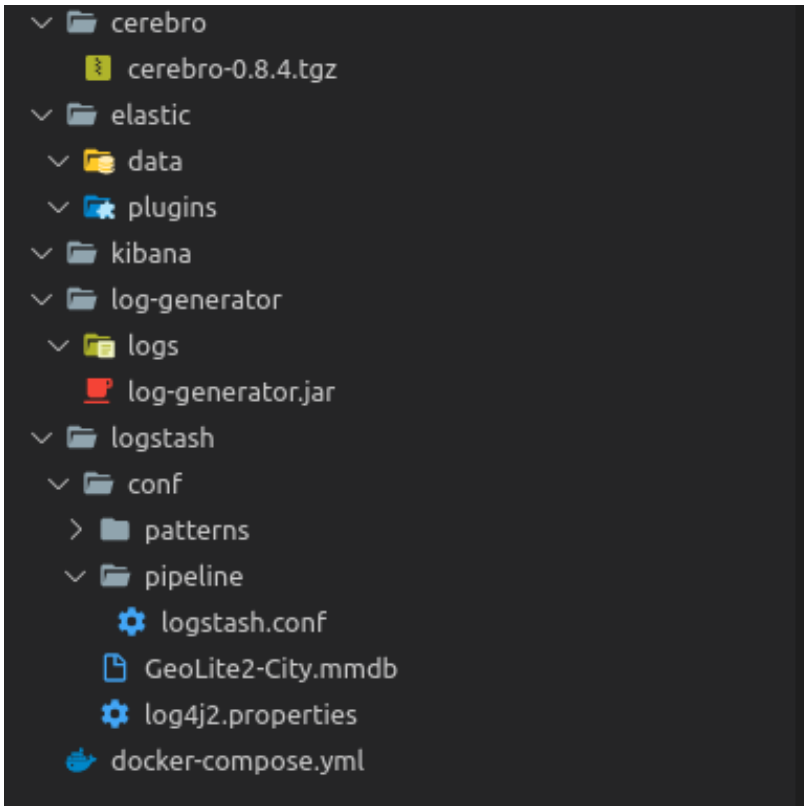
```
docker pull docker.elastic.co/kibana/kibana:7.4.0
```

4. télécharger le générateur de log, log-generator.jar
5. télécharger les données de géolocalisation, GeoLite2-City.mmdb.gz
6. télécharger la dernière release de cerebro

Configuration des composants

démarrer les composants

L'arborescence à mettre en œuvre pour ce TP :



Pour tous vos travaux vous utiliserez les liens pour autoriser les communications entre les différents containers.

La partie du docker compose pour lancer elasticsearch est :

```
elasticsearch:
    image: docker.elastic.co/elasticsearch/elasticsearch:7.4.0
    environment:
        - xpack.security.enabled=false
        - xpack.monitoring.enabled=false
        - xpack.ml.enabled=false
        - xpack.graph.enabled=false
        - xpack.watcher.enabled=false
        - bootstrap.memory_lock=true
        - "ES_JAVA_OPTS=-Xms512m -Xmx512m"
        - discovery.zen.minimum_master_nodes=1
        - discovery.type=single-node
        - http.cors.enabled:true
        - http.cors.allow-origin:*
    ulimits:
      memlock:
        soft: -1
        hard: -1
      nofile:
        soft: 65536
        hard: 65536
    volumes:
      - /home/etud/diogene.moulron/TP/05/elastic/data:/usr/share/elasticsearch/data:rw
      - /home/etud/diogene.moulron/TP/05/elastic/plugins:/usr/share/elasticsearch/plugins:rw
    ports:
      - "9200:9200"
      - "9300:9300
```

Démarrer logstash :

```
logstash:
    image: docker.elastic.co/logstash/logstash:7.4.0
    environment:
        - xpack.monitoring.enabled=false
        - config.reload.automatic=true
        - verbose=true
    expose:
        - "6540"
    ports:
        - "6540:5000"
    links:
        - elasticsearch:elasticsearch
    volumes:
        - /home/etud/diogene.moulron/elastic/logstash/conf/pipeline:/usr/share/logstash/pipeline
        - /home/etud/diogene.moulron/elastic/logstash/conf/log4j2.properties:/usr/share/logstash/config/log4j2.
properties
```

Le fichier log4j2 et ajouter un fichier logstash.conf avec la configuration par defaut.

```
input {
     tcp {
                port => 5000
        }
}
filter {}
output {
        stdout { codec => rubydebug{metadata =>  true } }
}
```

Démarrage de kibana :

```
kibana:
    image: docker.elastic.co/kibana/kibana:7.4.0
    environment:
       - ELASTICSEARCH_HOSTS=http://elasticsearch:9200
    ports:
        - "5601:5601"
```

les port correspondent à :

- 5000: Logstash TCP input
- 9200: Elasticsearch HTTP
- 9300: Elasticsearch TCP transport
- 5601: Kibana

1. Ecrire les lignes de commande docker run de chaque composant
2. Ecrire le docker-compose de lancement des composants (elasticsearch, logstash, kibana)

1. Attention aux droits de vos répertoires
2. Attention a l'utilisation des ports sur votre machine

Pour toutes les lignes de commande et pour toutes les références entre serveurs, il faut utiliser les alias à la place des ip et/ou host

pour stopper tous les containers

```
docker rm -f $(docker ps -a -q)
```

si vous souhaitez avoir des informations sur la santé et les index de elasticsearch :

Démarrage de cerebro :

```
cerebro:
    image: openjdk:8
    volumes:
      - /home/etud/diogene.moulron/elastic/cerebro:/usr/src/cerebro:ro
    ports:
      - "9000:9000"
    command: "bash -c 'cp -r /usr/src/cerebro/cerebro-0.8.4.tgz /tmp/cerebro-0.8.4.tgz; tar -xvf /tmp/cerebro-
0.8.4.tgz; /cerebro-0.8.4/bin/cerebro '"
```

Il est aussi possible de démarrer cerebro directement depuis votre poste.

Configuration et lancer logstash

- Créer un fichier de configuration (logstash.conf) dans le répertoire conf
- Copier le fichier d'information de géolocalisation dans le répertoire conf
- Créer un répertoire patterns dans la conf

- Ajouter une entrée

logstash.conf

```
input {
        tcp {
                port => 5000
        }
}
```

- ajouter la zone de filtre

logstash.conf

```
filter {

}
```

- configurer la sortie vers ES

logstash.conf

```
output {
        elasticsearch {
                hosts => "elasticsearch:9200"
        }
}
```

- Lancer le générateur de log pour valoriser ES

```
Usage: <main class> [options]
  Options:
        --help
       Default: false
  * -logs, -n
       Number of logs to generate
    -repeat, -r
       Repeat every N milliseconds
       Default: 0
    -threads, -t
       Number of threads to use
       Default: 1

java -jar log-generator.jar -n 5000 -r 1500 | nc 192.168.0.12 5000
```

sous Windows il est possible de lancer netcat-win32

```
java -jar log-generator.jar -n 500 -r 1500 | netcat-1.11\nc64 192.168.0.12 5000
```

Le log généré est

```
11-11-2016 16:28:24.508 [pool-1-thread-1] INFO com.github.vspiewak.loggenerator.SearchRequest - id=27,
ip=90.84.144.93,category=Portable,brand=Apple
11-11-2016 16:28:24.525 [pool-1-thread-1] INFO com.github.vspiewak.loggenerator.SellRequest - id=28,
ip=92.90.16.190,email=client29@gmail.com,sex=M,brand=Apple,name=iPhone 5C,model=iPhone 5C - Jaune -
Disque 32Go,category=Mobile,color=Jaune,options=Disque 32Go,price=699.0
11-11-2016 16:28:24.540 [pool-1-thread-1] INFO com.github.vspiewak.loggenerator.SearchRequest - id=29,
ip=93.31.186.100,category=Portable
11-11-2016 16:28:24.549 [pool-1-thread-1] INFO com.github.vspiewak.loggenerator.SearchRequest - id=30,
ip=109.211.12.248,category=Baladeur,brand=Apple,color=Argent,options=Disque 16Go
11-11-2016 16:28:24.557 [pool-1-thread-1] INFO com.github.vspiewak.loggenerator.SellRequest - id=31,
ip=86.73.160.167,email=client32@gmail.com,sex=M,brand=Apple,name=iPad mini,model=iPad mini - Blanc,
category=Tablette,color
```

- Consulter les logs générés sous kibana (http://192.168.0.12:5601/)

```
{
  "_index": "logstash-2016.11.11",
  "_type": "logs",
  "_id": "AVhUJpEVMJugJY_jEVP5",
  "_score": null,
  "_source": {
    "@timestamp": "2016-11-11T16:09:22.450Z",
    "geoip": {},
    "port": 38968,
    "@version": "1",
    "host": "172.17.0.1",
    "message": "11-11-2016 17:08:51.558 [pool-1-thread-2] INFO com.github.vspiewak.loggenerator.
SearchRequest - id=3770,ip=81.251.86.65,category=Baladeur,color=Jaune,options=Disque 32Go\r",
    "tags": [
      "_geoip_lookup_failure"
    ]
  },
  "fields": {
    "@timestamp": [
      1478880562450
    ]
  },
  "sort": [
    1478880562450
  ]
}
```

Les logs sous cette forme ne sont pas exploitables. Il faut effectuer des traitements dans le filtre de logstash pour les mettre en forme.

- Parse de la log pour extraire les données

logstash.conf

```
   grok {
               patterns_dir => ["/usr/share/logstash/patterns"]
               match => ["message","%{LOG_DATE:log_date} \[%{NOTSPACE:thread}\] %{LOGLEVEL:log_level} %
{NOTSPACE:classname} - %{GREEDYDATA:msg}"]
   }
```

LOG_DATE est un pattern (téléchargeable ici) spécifique qui permet d'extraite la date de la log, avec la mise a jour du lancement de logstash

```
    volumes:
        - /home/etud/diogene.moulron/elastic/logstash/conf/pipeline:/usr/share/logstash/pipeline
        - /home/etud/diogene.moulron/elastic/logstash/conf/patterns:/usr/share/logstash/patterns
        - /home/etud/diogene.moulron/elastic/logstash/conf/log4j2.properties:/usr/share/logstash/config/log4j2.
properties
```

- La nouvelle forme de la log est

```
{
  "_index": "logstash-2014.11.03",
  "_type": "logs",
  "_id": "i8UgsVDbR6-ZZ4LP4Y3TWw",
  "_score": null,
  "_source": {
    "message": "03-11-2014 14:12:59.999 [pool-34-thread-2] INFO com.github.vspiewak.loggenerator.SellRequest -
id=166099&ua=Mozilla/5.0 (Windows NT 5.1; rv:2.0.1) Gecko/20100101 Firefox/4.0.1&ip=94.228.34.210
&email=client100@gmail.com&sex=M&brand=Apple&name=iPod Touch&model=iPod Touch - Violet - Disque
64Go&category=Baladeur&color=Violet&options=Disque 64Go&price=449.0\r",
    "@version": "1",
    "@timestamp": "2014-11-03T13:13:00.245Z",
    "host": "FRO9272454D",
    "path": "D:\\etudes\\Cours\\ELK\\logstash-1.4.2\\bin\\app.log",
    "log_date": "03-11-2014 14:12:59.999",
    "thread": "pool-34-thread-2",
    "log_level": "INFO",
    "classname": "com.github.vspiewak.loggenerator.SellRequest",
    "msg": "id=166099&ua=Mozilla/5.0 (Windows NT 5.1; rv:2.0.1) Gecko/20100101 Firefox/4.0.1&ip=94.228.34.210
&email=client100@gmail.com&sex=M&brand=Apple&name=iPod Touch&model=iPod Touch - Violet - Disque
64Go&category=Baladeur&color=Violet&options=Disque 64Go&price=449.0\r"
  }
}
```

Entre chaque modification logstash la recharge.

Pour visualiser les erreurs :

```
docker logs -f  infraprod_logstash_1
```

Visualisation des composants

Visualiser sous kibana

Ajout de filtre pour une meilleure exploitabilité

- **Extraire les données de la requête (id=60,ip=80.78.9.34,brand=Apple,name=iPhone 5C,model=iPhone 5C - Vert - Disque 32Go, category=Mobile,color=Vert,options=Disque 32Go,price=699.0)**
  Utilisation du filtre kv
- **Tagger les requêtes en sell ou search (com.github.vspiewak.loggenerator.SearchRequest ou com.github.vspiewak.loggenerator.SellRequest)**
  Utilisation d'une combinaison de mutate et de add_tag en conjonction avec du conditionnel.
- **Ajouter les informations de géolocalisation** dans le fichier de conf logstash

```
geoip {
        source => "ip"
        database => "/usr/share/logstash/config/GeoLite2-City.mmdb"
}
```

- **Conversion des informations de prix en number**
  utilisation du filtre mutate et convert
- **Extraction des options**
  utilisation du filtre mutate et split

Logs exploitable

Le nouveau format des logs est plus exploitable maintenant

```
{
  "_index": "logstash-2014.11.03",
  "_type": "logs",
  "_id": "4tqUHrG6TsOkDFC5AV3Nrg",
  "_score": null,
  "_source": {
    "message": "03-11-2014 14:35:59.916 [pool-21-thread-2] INFO com.github.vspiewak.loggenerator.SellRequest -
id=20411&ua=Mozilla/5.0 (X11; Linux x86_64; rv:2.0) Gecko/20100101 Firefox/4.0&ip=80.215.33.111
&email=client412@gmail.com&sex=M&brand=Google&name=Nexus 4&model=Nexus 4 - Noir - Disque
16Go&category=Mobile&color=Noir&options=Disque 16Go&price=199.0\r",
    "@version": "1",
    "@timestamp": "2014-11-03T13:35:59.916Z",
    "host": "FRO9272454D",
    "path": "D:\\etudes\\Cours\\ELK\\logstash-1.4.2\\bin\\app.log",
    "log_date": "03-11-2014 14:35:59.916",
    "thread": "pool-21-thread-2",
    "log_level": "INFO",
    "classname": "com.github.vspiewak.loggenerator.SellRequest",
    "msg": "id=20411&ua=Mozilla/5.0 (X11; Linux x86_64; rv:2.0) Gecko/20100101 Firefox/4.0&ip=80.215.33.111
&email=client412@gmail.com&sex=M&brand=Google&name=Nexus 4&model=Nexus 4 - Noir - Disque
16Go&category=Mobile&color=Noir&options=Disque 16Go&price=199.0\r",
    "tags": [
      "sell"
    ],
    "id": 20411,
    "ua": "Mozilla/5.0 (X11; Linux x86_64; rv:2.0) Gecko/20100101 Firefox/4.0",
    "ip": "80.215.33.111",
    "email": "client412@gmail.com",
    "sex": "M",
    "brand": "Google",
    "name": "Nexus 4",
    "model": "Nexus 4 - Noir - Disque 16Go",
    "category": "Mobile",
    "color": "Noir",
    "options": [
      "Disque 16Go"
    ],
    "price": 199,
    "geoip": {
      "ip": "80.215.33.111",
      "country_code2": "FR",
      "country_code3": "FRA",
      "country_name": "France",
      "continent_code": "EU",
      "latitude": 48.860000000000014,
      "longitude": 2.3499999999999943,
      "timezone": "Europe/Paris",
      "location": [
        2.3499999999999943,
        48.860000000000014
      ]
    }
  }
}
```


Création d'un dashboard métier

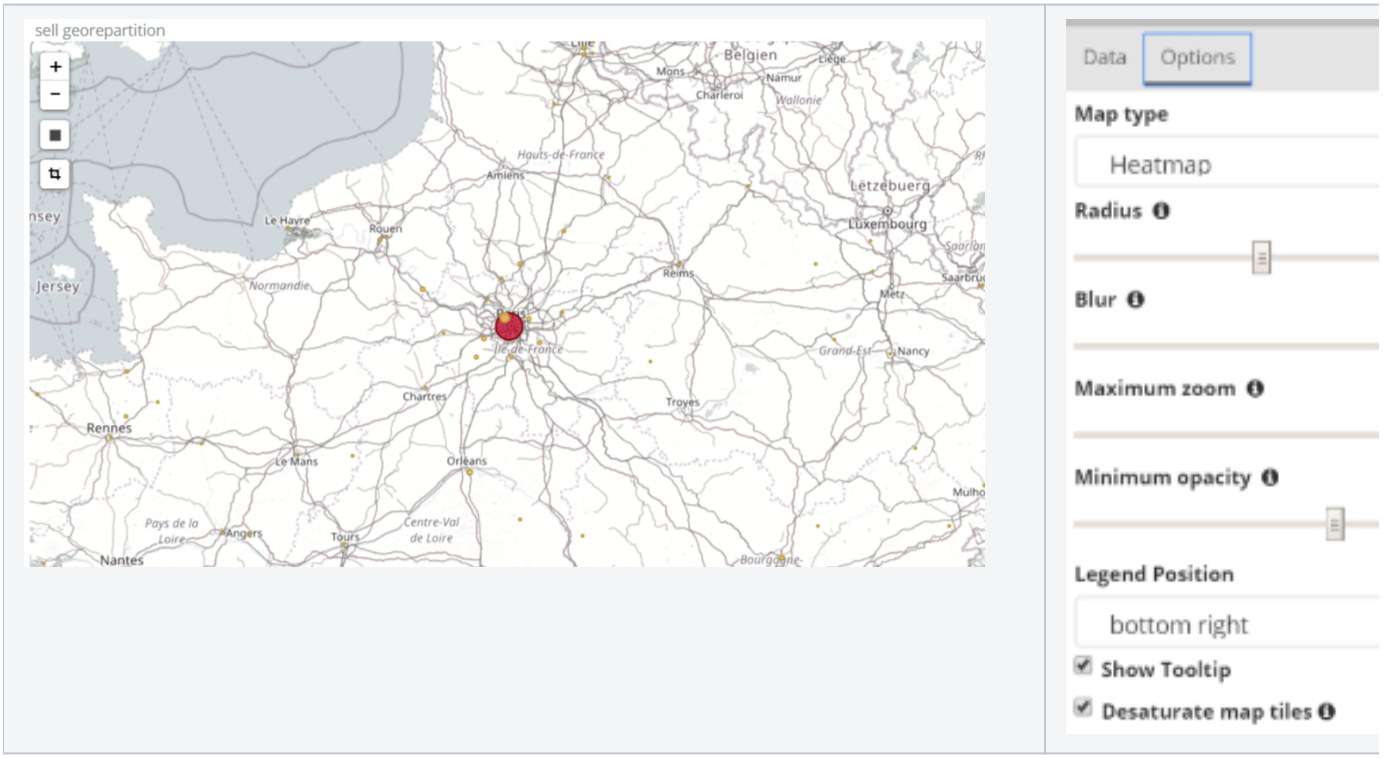
Préparation

La préparation consiste en la création de query préparées :

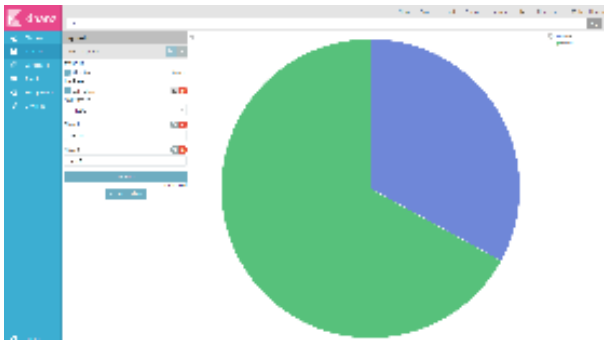
1. dans la zone de recherche, ajoutez :
    a. tags:search
    b. tags:sell
    c. sex:M
    d. sex:F

Ajout d'un widget de localisation des ventes

Le widget s'appelle Tile map.

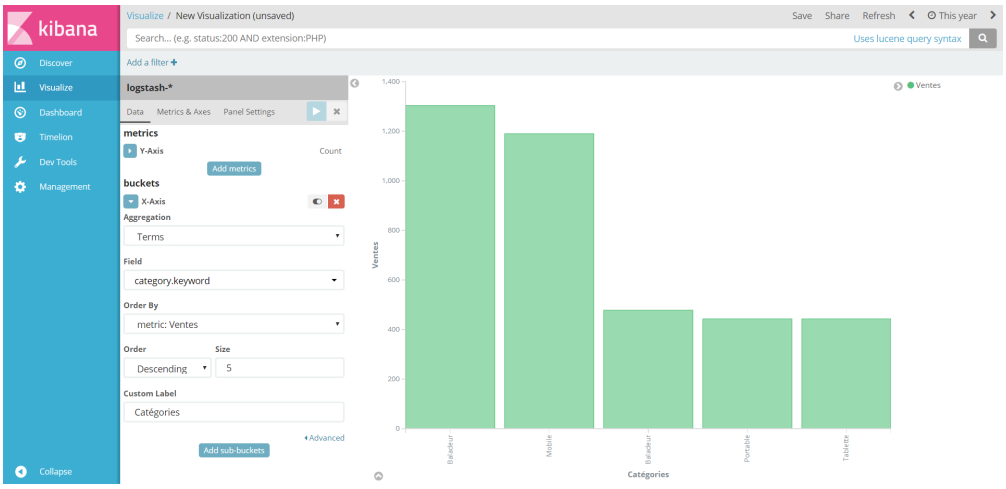


Répartition des ventes selon le sexe



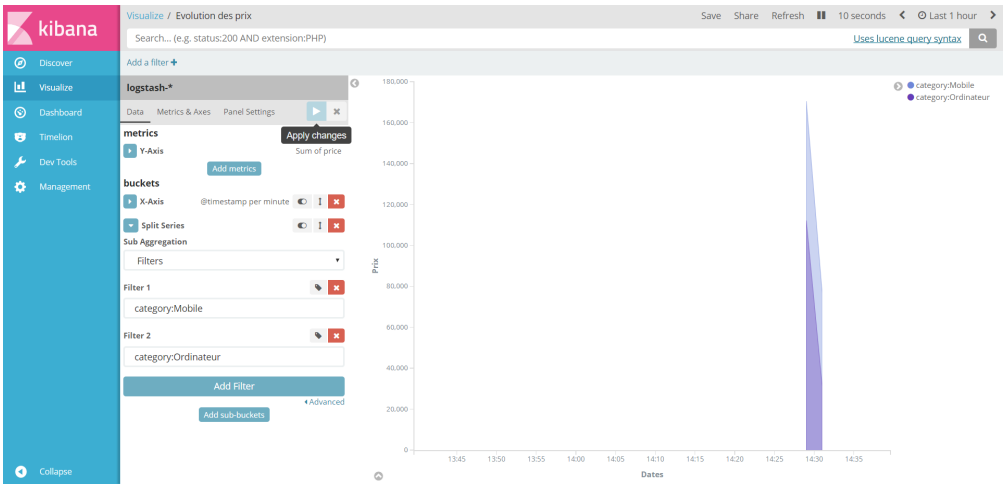
Ajout d'un widget Vertical Bar

Ajouter un widget vertical bar pour mettre les types de ventes (Catégories) sur la période

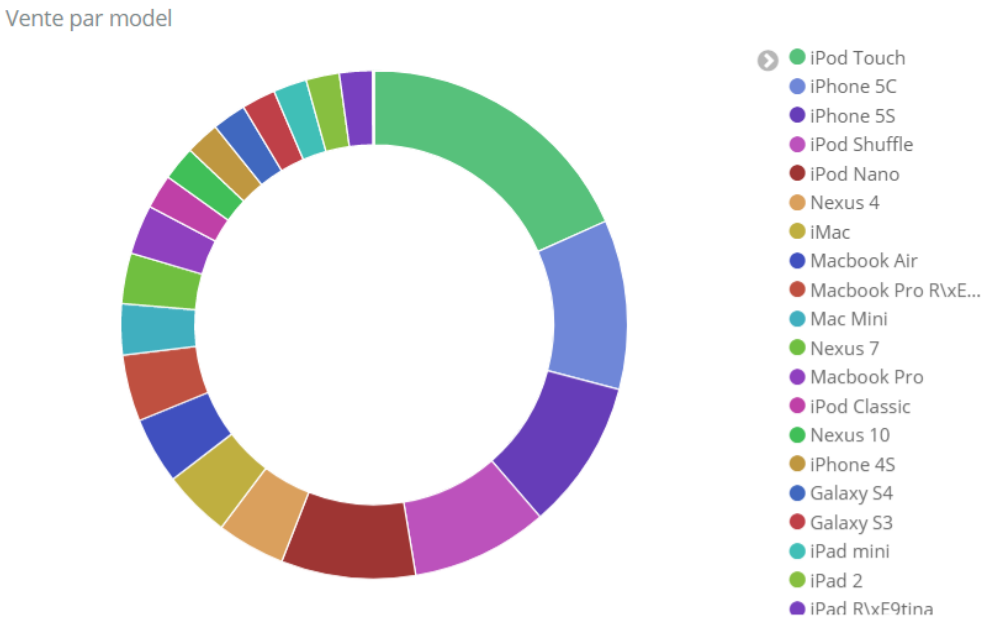


Ajout d'un widget sur le montant total des ventes

Ajouter un widget Line pour sommer les ventes sur la période



Ajouter un camembert donnant le nombre de vente par modèle



Pour ceux qui ont le plus avancé, vous pouvez faire un tableau de bord