## 1-Phase de production du code

Le code a été écrit en utilisant la bibliothèque Serilog pour générer des fichiers journaux dans différents formats. Voici le code utilisé :

```
using System;
using Serilog;
using Serilog.Formatting.Compact;
using Serilog.Formatting.Json;

class Program
{
    static void Main(string[] args)
    {

        Log.Logger = new LoggerConfiguration()
            .MinimumLevel.Debug()
            .WriteTo.File("app.csv", outputTemplate: "{Timestamp:yyyy-MM-ddTHH:mm:ss.fffZ},{Level},{Message}{NewLine}")
            .WriteTo.File("app.tsv", outputTemplate: "{Timestamp:yyyy-MM-ddTHH:mm:ss.fffZ}\t{Level}\t{Message}{NewLine}")
            .WriteTo.File(new CompactJsonFormatter(), "app.ndjson")
            .WriteTo.File("app.log", outputTemplate: "{Timestamp:yyyy-MM-ddTHH:mm:ss.fffZ} {Level} {Message}{NewLine}")
            .CreateLogger();


        for (int i = 0; i < 10; i++)
        {
            Log.Information("This is info  ");
            Log.Error("This is error  " );
            Log.Debug("This is debug  " ); //
        }


        Log.CloseAndFlush();
    }
}

```

Dans ce code, j'ai configuré Serilog pour enregistrer les journaux dans différents formats : texte simple, JSON structuré, JSON compact, CSV, TSV et NDJSON. Le code inclut également la génération de journaux d'informations, d'avertissements et d'erreurs simulées.

# Phase de fichier app.log

J'ai créé plusieurs fichiers journaux dans différents formats et je les ai placés sur un serveur Ubuntu en utilisant la commande suivante :

-scp net8.0.zip root@138.68.68.147:/root/hydra

J'ai utilisé cette commande pour transférer les fichiers vers le serveur distant où les journaux seront stockés et traités ultérieurement.

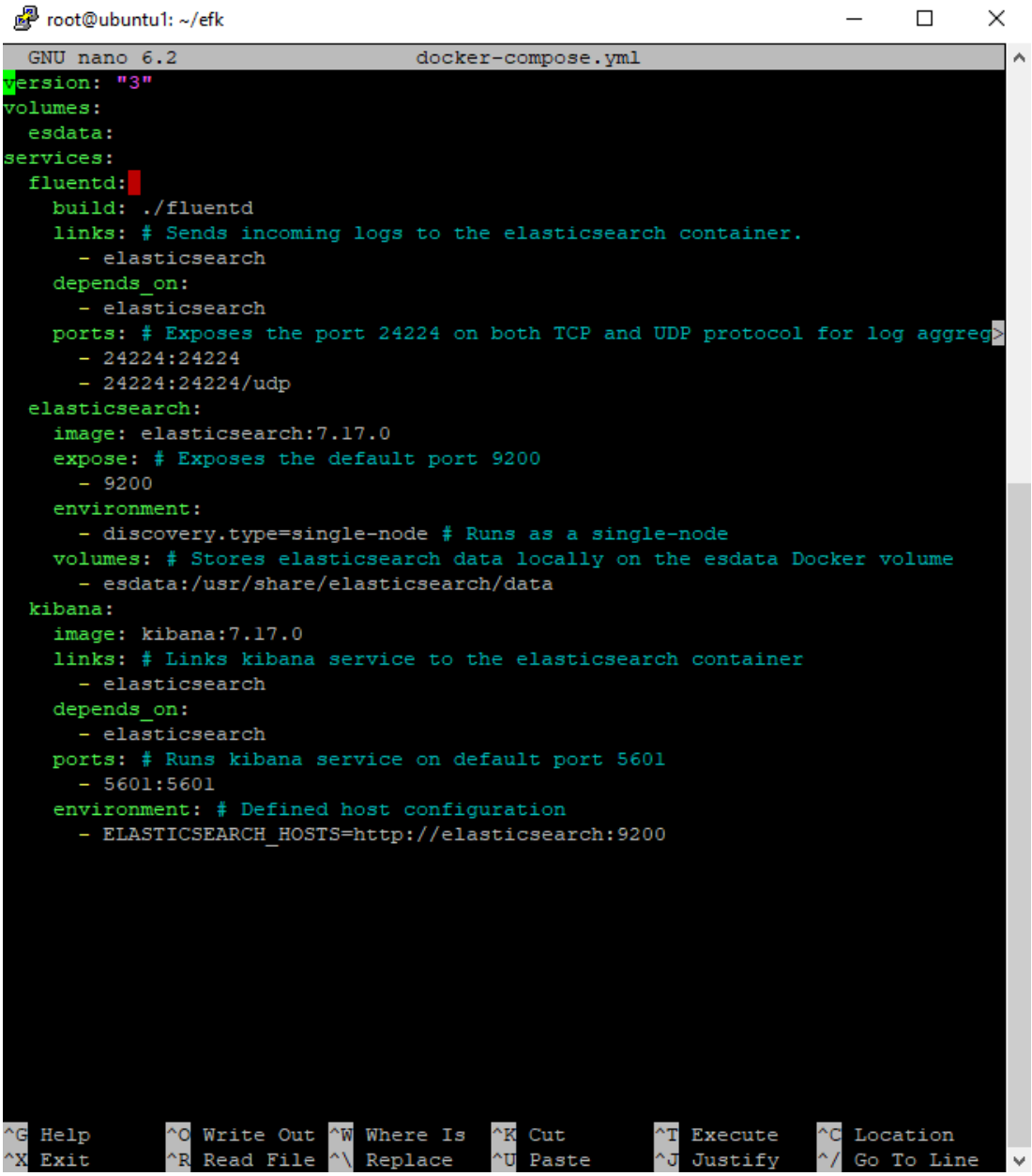
Phase de traitement Fluentd

J'ai installé la pile EFK avec Docker.

Création d'un dossier et changement de répertoire

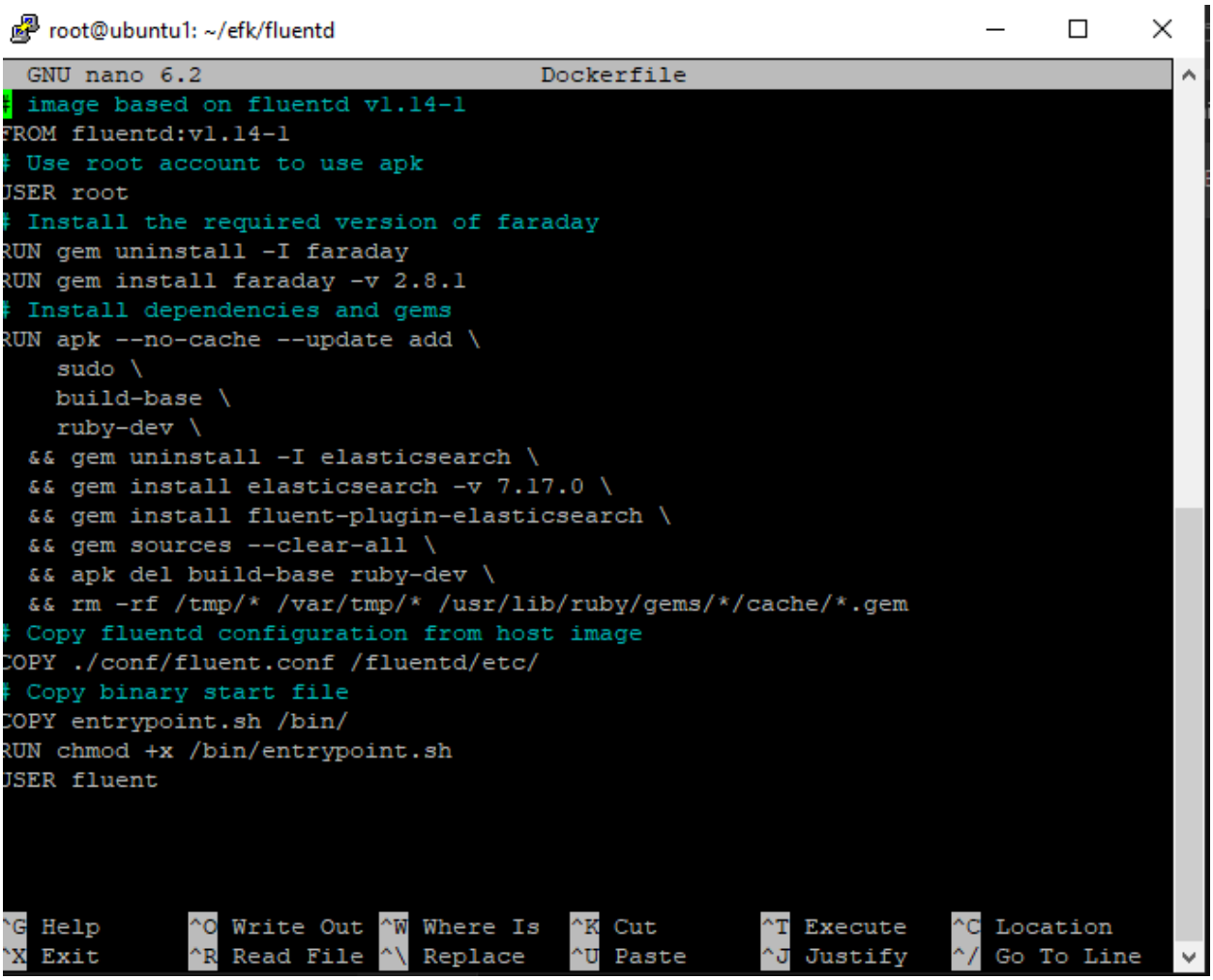
mkdir -p ~/efk; cd ~/efk

Création du fichier `docker-compose.yml`




Création des fichiers Fluentd

Dockerfile




Entrypoint.sh

root@ubuntu1: ~/efk/fluentd

```
  GNU nano 6.2                          entrypoint.sh
#!/bin/sh
# Source vars if file exists
DEFAULT=/etc/default/fluentd
if [ -r $DEFAULT ]; then
    set -o allexport
    . $DEFAULT
    set +o allexport
fi
# If the user has supplied only arguments, append them to `fluentd` command
if [ "${1#-}" != "$1" ]; then
    set -- fluentd "$@"
fi
# If the user does not supply a config file or plugins, use the default
if [ "$1" = "fluentd" ]; then
    if ! echo $@ | grep -e ' \-c' -e ' \-\-config' ; then
      set -- "$@" --config /fluentd/etc/${FLUENTD_CONF}
    fi
    if ! echo $@ | grep -e ' \-p' -e ' \-\-plugin' ; then
      set -- "$@" --plugin /fluentd/plugins
    fi
fi
exec "$@"

                          [ Read 22 lines ]
^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^/ Go To Line
```

Fluent.conf

root@ubuntu1: ~/efk/fluentd/conf

```
  GNU nano 6.2                         fluent.conf *
<source>
  @type forward
  port 24224
  bind 0.0.0.0
</source>
<match *.**>
  @type copy
  <store>
    @type elasticsearch
    host elasticsearch
    port 9200
    logstash_format true
    logstash_prefix fluentd
    logstash_dateformat %Y%m%d
    include_tag_key true
    type_name access_log
    tag_key @log_name
    flush_interval 20s
  </store>
  <store>
    @type stdout
  </store>
</match>
```

Lancer les conteneurs Docker

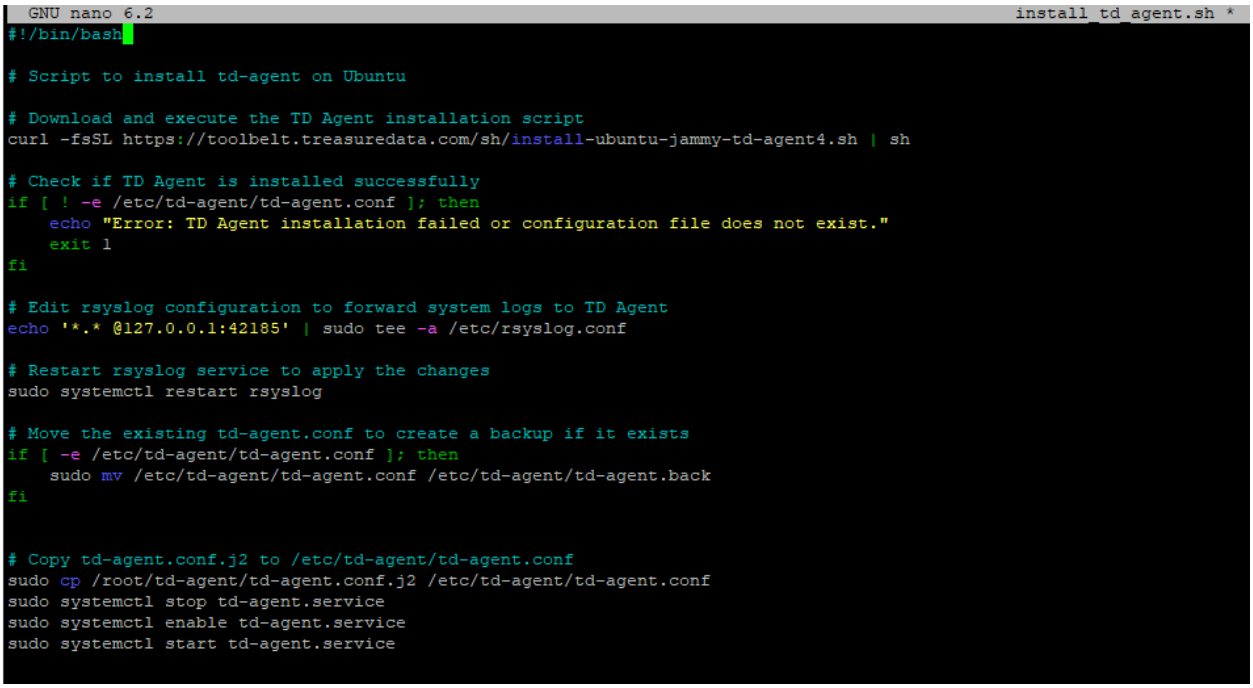
cd ~/efk/

docker-compose up -d

# configuration de TD Agent (Fluentd) sur Ubuntu 22.04 pour collecter des logs

Création d'un script pour installer TD Agent

td-agent/install_td_agent.sh




td-agent/td-agent.conf.j2

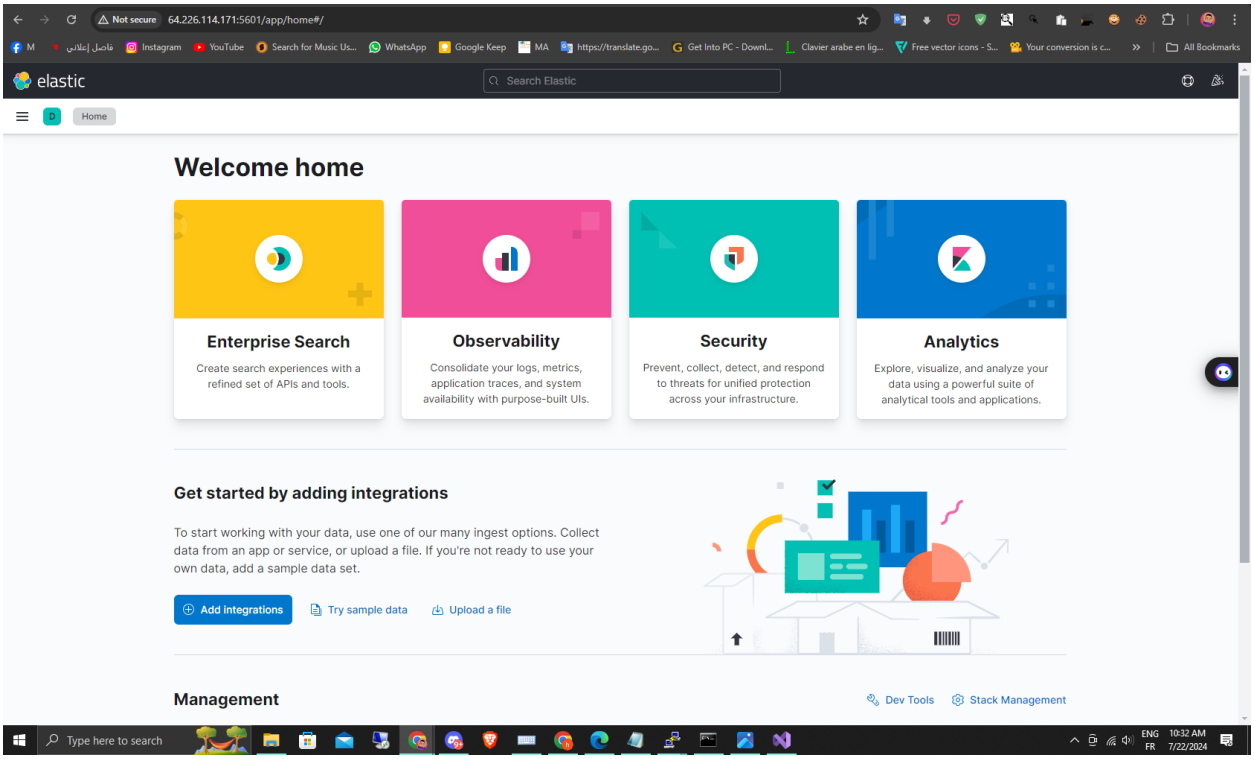
```
  GNU nano 6.2                                                    td-ager
<source>
  @type tail
  @id input_tail
  path /root/hydra/app.csv, /root/hydra/app.log, /root/hydra/app.ndjson, /root/hydra/app.tsv
  pos_file /var/log/td-agent/app.log.pos
  tag app.logs
  format none
</source>

<match app.logs>
  @type forward
  @id forward_app_logs
  <server>
    host 64.226.114.171
    port 24224
  </server>
</match>
```

**chmod +x install_td_agent.sh**
**./install_td_agent.sh**

## Phase du serveur Kibana

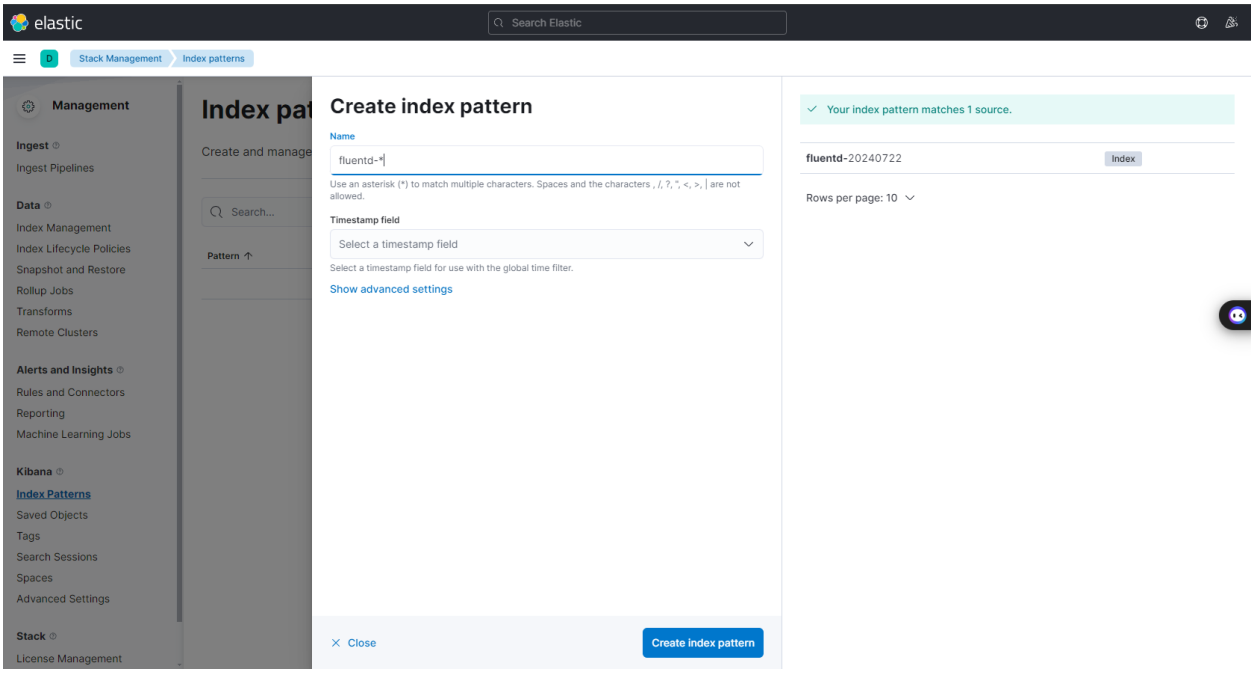
## Configuration du tableau de bord dans Kibana

Ouvrez le navigateur et allez à [http://64.226.114.171:5601/](http://64.226.114.171:5601/)

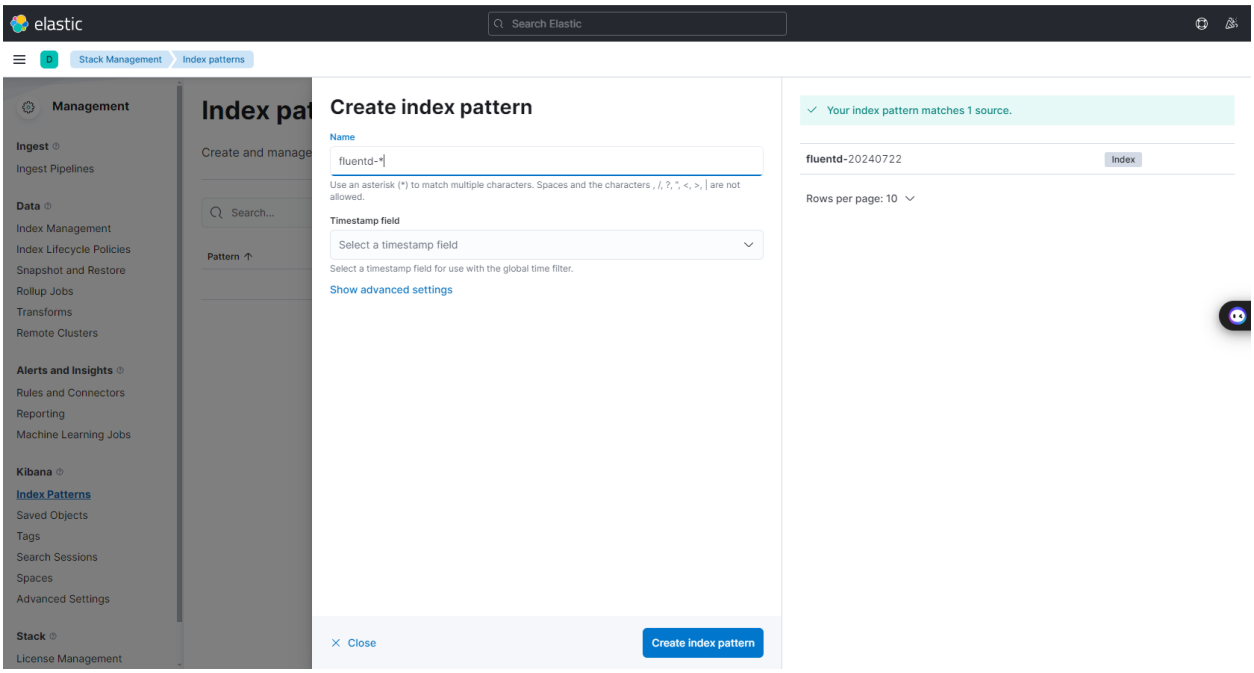


2- "Explore on My Own" sur la page d'accueil.

3-"Stack Management" pour configurer le modèle d'index dans la section de gestion.
4 Dans le menu de gauche de Kibana,  "Index Patterns" puis "Create Index Pattern".




5-Entrez le nom du modèle d'index comme `fluentd-*`, et définissez le champ de l'horodatage sur `@timestamp`, puis cliquez sur "Create index pattern".




6- Voici la capture d'écran du tableau de bord de surveillance et d'analyse des logs de Kibana. Tous les logs répertoriés sont extraits d'Elasticsearch et envoyés par l'agrégateur de logs Fluentd.