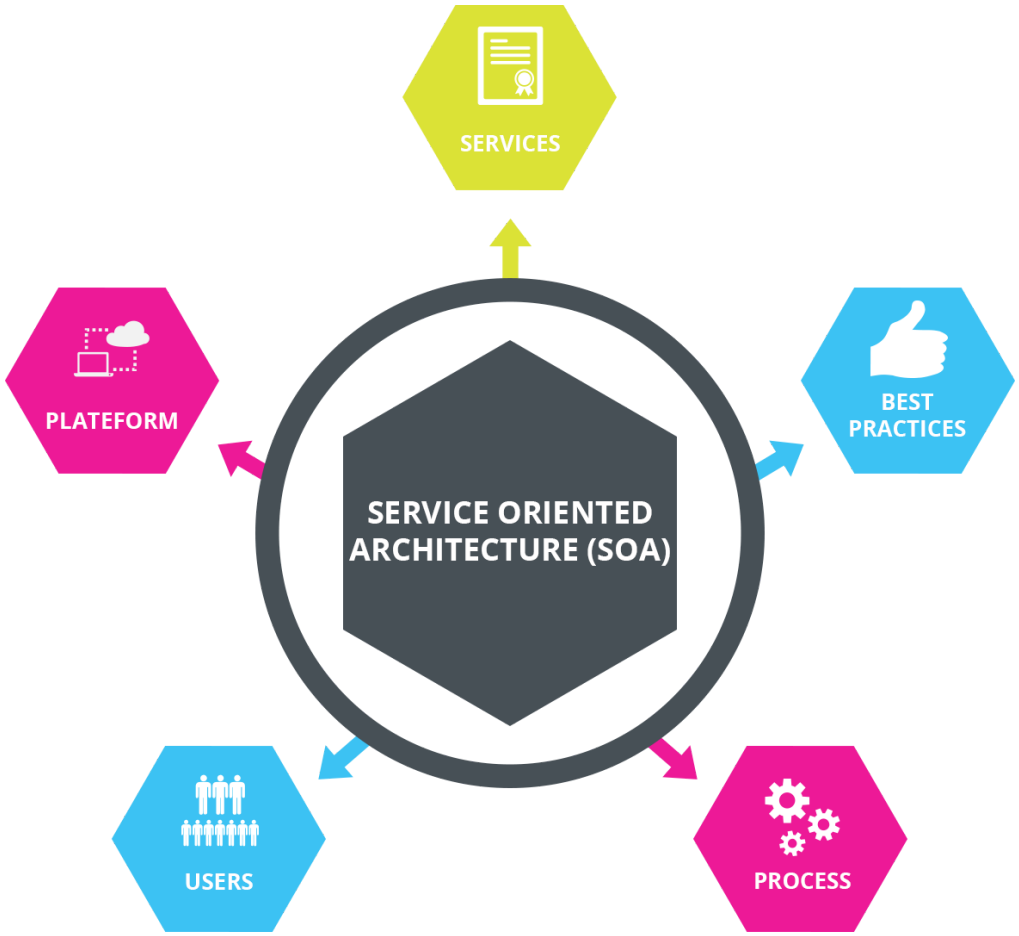
Architectures Orientées Services

Rapport de Projet



Abdou Yaya SADIAKHOU

29-11-2022

M2 DATASCALE

INTRODUCTION

Le but de ce projet est de mettre en place une plateforme pour l'analyse de données de réseaux sociaux à base de services web: InPoDa.

DESCRIPTION

Un utilisateur fait un tweet sur le réseau social. Le tweet est publié à travers un service web SOAP qui va stocker la publication dans une table (Tweet) de la base de données.

La plateforme InPoDa que nous avons développé fournit un ensemble de fonctionnalités que nous avons par ailleurs considéré comme des services à part entière afin de découpler le code le maximum possible.

Entre autre pour un tweet donné on peut:

- Identifier l'auteur du tweet (ServiceIdentificationAuteur)
- Extraire les différents hashtag (ServiceHashtagExtraction)
- Prédire le sentiment dégagé par le tweet (ServiceSentimentAnalysis)
- Identifier les topics du tweet (ServiceIdentificationTopic)

A cela s'ajoutent les services d'analyse de données sur la base de données entière.

- Afficher le top K des hashtags (ServiceTopKHashtag), des utilisateurs (ServiceTopKUsers) et des topics (ServiceTopKTopics)
- Le nombre de publications par utilisateur (ServiceUserPublish), par hashtag (ServiceHashtagPublish) et par topic (ServiceTopicPublish)

Identification des services et modélisation fonctionnement interne

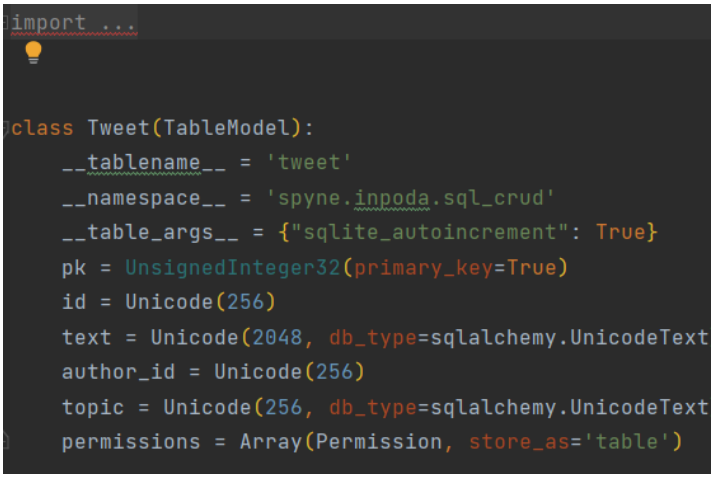
SERVICES DE PRÉTRAITEMENT

- **serviceTweet:** C'est le service d'ingestion de tweet. A chaque publication ce service récupère le tweet et le stocke dans la base de données ainsi les autres services pourrons requêter sur les données.

En entrée: les données d'un tweet

→ format JSON

A l'intérieur: désérialisation et stockage dans la base de données



En sortie: message de succès

- **ServiceIdentificationAuteur**: Ce service permet d'identifier l'auteur d'un tweet à partir d'un tweet donné.

  Fonctionnement: Le tweet est récupéré dans la base de données et ensuite on renvoi l'identifiant de l'auteur

```python
import ...


class ServiceAuthorIdentification(ServiceBase):

    @rpc(Unicode, _returns=String)
    def identifyAuthor(ctx,tweet_id):
        # get tweet from database
        tweet = ctx.udc.session.query(Tweet).filter_by(id=tweet_id).one()
        return f"Author is {tweet.author_id}"
```

- **ServiceIdentificationTopic**: Ce service permet d'identifier le topic d'un tweet (politique, culturel, musical…)

  Fonctionnement: Le tweet est récupéré dans la base de données et ensuite on extrait les topics pour ensuite les stocker dans une table Topic avec le tweet comme clé secondaire.

  → Table métier Topic

```python
from spyne import UnsignedInteger32, Unicode, Array

from db.dbInstance import TableModel, Permission


class Topic(TableModel):
    __tablename__ = 'topic'
    __namespace__ = 'spyne.inpoda.sql_crud'
    __table_args__ = {"sqlite_autoincrement": True}

    id = UnsignedInteger32(primary_key=True)
    topic = Unicode(256)
    tweet_id = Unicode(256)
    permissions = Array(Permission, store_as='table')
```

Service Topic Identification:

```python
class ServiceTopicIdentification(ServiceBase):

    @rpc(Unicode, _returns=String)
    def identifyTopic(ctx, tweet_id):
        # get tweet from database
        tweet = ctx.udc.session.query(Tweet).filter_by(id=tweet_id).one()
        topics = tweet.topic.split("<->")
        # save topics in Topic table
        for topic in topics:
            if ctx.udc.session.query(Topic).filter_by(topic=topic, tweet_id=tweet_id).count() == 0:
                obj = Topic(topic=topic, tweet_id=tweet_id)
                ctx.udc.session.add(obj)
                ctx.udc.session.flush()
        return f"Topics of this tweets: {tweet.topic}"
```

- **ServiceExtractionHashtag**: Ce service permet d'extraire les hashtags d'un tweet donné.

  Fonctionnement: Le tweet est récupéré de la base de données et ensuite on extrait les différents hashtags à l'aide d'une expression régulière pour ensuite stocker chaque hashtag dans la table métier Hashtag.

  → regex: `re.findall(r"#(\w+)", text)`

  → Classe métier

```python
class HashTag(TableModel):
    __tablename__ = 'hashtag'
    __namespace__ = 'spyne.inpoda.sql_crud'
    __table_args__ = {"sqlite_autoincrement": True}

    id = UnsignedInteger32(primary_key=True)
    hashtag = Unicode(256)
    tweet_id = Unicode(256)
    permissions = Array(Permission, store_as='table')
```

  → Service extraction de hashtag

```
class ServiceExtractionHashtag(ServiceBase):
    @rpc(Unicode, _returns=Array(String))
    def extractTweetHashtag(ctx, tweet_id):
        extracted_hashtags = extractHashtagFromText(ctx.udc.session.query(Tweet)
                                                    .filter_by(id=tweet_id)
                                                    .one().text)
        # save extracted hashtags in database
        for hashtag in extracted_hashtags:
            obj = HashTag(hashtag=hashtag.lower(), tweet_id=tweet_id)
            # check if hashtag already exists
            if ctx.udc.session.query(HashTag).\
                    filter_by(hashtag=hashtag.lower(), tweet_id=tweet_id)\
                    .count() == 0:
                ctx.udc.session.add(obj)
                ctx.udc.session.flush()

        return extracted_hashtags
```

- **ServiceSentimentAnalysis:** Ce service permet d'analyser le sentiment dégagé par un tweet donné.

  Fonctionnement: Le service extrait le tweet de la base de données et ensuite applique un algorithme d'analyse de sentiment qui prédit si le tweet est: NÉGATIVE, POSITIVE ou NEUTRE. Ensuite le résultat est stocké dans la table Sentiment avec le tweet comme clé secondaire.

  → Table métier

```
class Sentiment(TableModel):
    __tablename__ = 'sentiment'
    __namespace__ = 'spyne.inpoda.sql_crud'
    __table_args__ = {"sqlite_autoincrement": True}

    id = UnsignedInteger32(primary_key=True)
    sentiment = Unicode(256)
    polarity = Float()
    subjectivity = Float()
    tweet_id = Unicode(256)
    permissions = Array(Permission, store_as='table')
```

→ Analyse de sentiment:

```python
def predictTweetSentiment(text):
    import textblob
    from translate import Translator
    # translate text to english
    translator = Translator(to_lang="en", from_lang="fr")

    def getSubjectivity(text):
        return textblob.TextBlob(text).sentiment.subjectivity

        # Create a function to get the polarity

    def getPolarity(text):
        return textblob.TextBlob(text).sentiment.polarity

        # Create two new columns "Subjectivity" & "Polarity"
    text = translator.translate(text)
    print("TEXT --- ", text)
    subj = getSubjectivity(text)
    polarity = getPolarity(text)

    def getAnalysis(score):
        if score < 0:
            return "Negative"
        elif score == 0:
            return "Neutral"
        else:
            return "Positive"

    sentiment = getAnalysis(polarity)

    return sentiment, polarity, subj
```

→ Service analyse de sentiment

```python
class ServiceSentimentAnalysis(ServiceBase):
    @rpc(Unicode, _returns=String)
    def predictSentiment(ctx, tweet_id):
        sentiment, polarity, subjectivity = predictTweetSentiment(
            ctx.udc.session.query(Tweet)
            .filter_by(id=tweet_id)
            .one().text)
        # save extracted hashtags in database
        obj = Sentiment(sentiment=sentiment.lower(),
                        tweet_id=tweet_id,
                        polarity=polarity,
                        subjectivity=subjectivity)
        # check if hashtag already exists
        if ctx.udc.session.query(Sentiment) \
                .filter_by(sentiment=sentiment,
                           tweet_id=tweet_id) \
                .count() == 0:
            ctx.udc.session.add(obj)
            ctx.udc.session.flush()

        return sentiment
```

SERVICES D'ANALYSE

- **ServiceTopKhashtag:** Ce service permet de renvoyer les K hashtag les plus utilisés dans les tweets

  Fonctionnement: Charge tous les hashtags de la table Hashtag en faisant un **groupBy** sur le nom du hash et un **count** sur les occurrences de ce tag.

```python
class ServiceTopKHashTag(ServiceBase):
    @rpc(Unicode, _returns=Iterable(Unicode))
    def getTopKHashTag(ctx, K):
        # get all hashtags
        hashtags = ctx.udc.session.query(HashTag).all()
        # group by hashtag
        hashtags = {k: list(v) for k, v in itertools.groupby(hashtags,
                                                             key=lambda x: x.hashtag)}
        # sort hashtags by count
        hashtags = sorted(hashtags.items(), key=lambda x: len(x[1]), reverse=True)
        # get top K hashtags
        hashtags = hashtags[:int(K)]
        # yield results
        for hashtag in hashtags:
            yield f"{hashtags.index(hashtag)+1} -> " \
                  f"{hashtag[0]}: {len(hashtag[1])} tweets with this hashtag"
```

- **ServiceTopKTopic:** Ce service permet de renvoyer les K topics les plus récurrents dans les tweets.

  Fonctionnement: Le service récupère tous les topics de la table Topic en faisant un ***group by*** suivi d'un ***count*** sur le nombre d'occurrence ensuite renvoi les K premiers topics.

```python
class ServiceTopKTopic(ServiceBase):
    @rpc(Unicode, _returns=Iterable(Unicode))
    def getTopKTopic(ctx, K):
        topics = ctx.udc.session.query(Topic).all()
        # group by topic
        topics = {k: list(v)
                  for k, v in
                  itertools.groupby(topics, key=lambda x: x.topic)}
        # sort topics by count
        topics = sorted(topics.items(), key=lambda x: len(x[1]), reverse=True)
        # get top K topics
        topics = topics[:int(K)]
        # yield results
        for item in topics:
            yield f"{item[0]}: {len(item[1])} tweets about this topic"
```

- **ServiceTopKUser:** Ce service permet de renvoyer les utilisateurs les plus actifs.

  Fonctionnement: Le service récupère les tweets de la table Tweet en faisant un ***group by*** suivi d'un ***count*** sur l'attribut **AUTHOR_ID** ensuite renvoi les K

premiers éléments.

```python
class ServiceTopKUser(ServiceBase):
    @rpc(Unicode, _returns=Iterable(Unicode))
    def getTopKUser(ctx, K):
        tweets = ctx.udc.session.query(Tweet).all()
        # group by user
        tweets = {k: list(v) for k, v in itertools.groupby(tweets, key=lambda x: x.author_id)}
        # sort tweets by count
        tweets = sorted(tweets.items(), key=lambda x: len(x[1]), reverse=True)
        # get top K users
        tweets = tweets[:int(K)]
        # yield results
        for item in tweets:
            yield f"{item[0]}: {len(item[1])} tweets by this user"
```

- **ServiceUserPublish:** Ce service permet de renvoyer les publications d'un utilisateurs données

  Fonctionnement: Le service récupère tous les tweets de l'utilisateur en entrée et renvoie cette liste récupérée.

```python
class ServiceUserPublish(ServiceBase):
    @rpc(Unicode, _returns=Iterable(Unicode))
    def publishByUser(ctx, user_id):
        tweets = ctx.udc.session.query(Tweet).filter_by(author_id=user_id).all()
        yield f"Found {len(tweets)} tweets by user {user_id}"
        for t in tweets:
            yield "--> " + t.text
```

- **ServiceTopicPublish:** Ce service permet de renvoyer les tweets sur un topic précis

  Fonctionnement: Le service récupère tous les topics en filtrant sur le topic en entrée et renvoie cette liste récupérée en associant aux tweets concernés.

```python
class ServiceTopicPublish(ServiceBase):
    @rpc(Unicode, _returns=Iterable(Unicode))
    def publishByTopic(ctx, topic):
        topic = topic.lower()
        topics = ctx.udc.session.query(Topic).filter_by(topic=topic).all()
        yield f"Found {len(topics)} tweets with topic {topic}"
        for t in topics:
            tweet = ctx.udc.session.query(Tweet).filter_by(id=t.tweet_id).one()
            yield "---> "+tweet.text
```

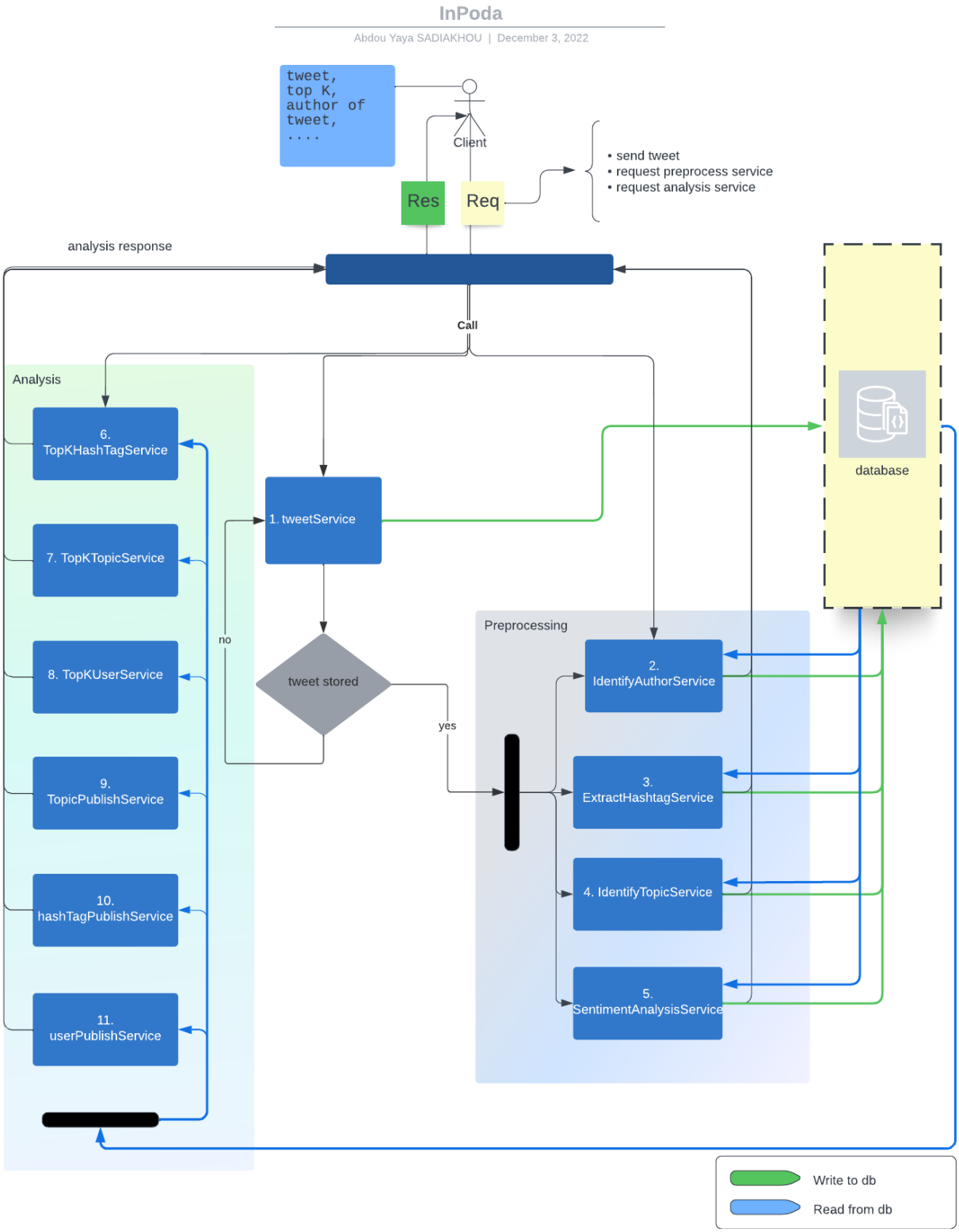
- **ServiceHashtagPublish:** Ce service permet de renvoyer les tweets contenant un hashtag donné.

  Fonctionnement: Le service récupère tous les tweets ayant le hashtag en entrée et renvoie cette liste récupérée en associant aux tweets concernés.

```python
class ServiceHashTagPublish(ServiceBase):
    @rpc(Unicode, _returns=Iterable(Unicode))
    def publishByHashTag(ctx, hashtag):
        hashtag = hashtag.lower()
        hashtags = ctx.udc.session.query(HashTag).filter_by(hashtag=hashtag).all()
        yield f"Found {len(hashtags)} tweets with hashtag {hashtag}"
        for h in hashtags:
            # get tweet from database
            print(h)
            tweet = ctx.udc.session.query(Tweet).filter_by(id=h.tweet_id).one()
            yield "--->" + tweet.text
```

Architecture

Nous pouvons à partir du descriptif déduire l'architecture suivante pour notre application globale.


InPoda
Abdou Yaya SADIAKHOU | December 3, 2022
tweet,
top K,
author of
tweet,
....
Client
Res
Req
• send tweet
• request preprocess service
• request analysis service
analysis response
Call
database
Analysis
6.
TopKHashTagService
7. TopKTopicService
8. TopKUserService
9.
TopicPublishService
10.
hashTagPublishService
11.
userPublishService
1. tweetService
no
tweet stored
yes
Preprocessing
2.
IdentifyAuthorService
3.
ExtractHashtagService
4. IdentifyTopicService
5.
SentimentAnalysisService
Write to db
Read from db

RÉSULTATS

Nous avons développé le coté serveur avec spyne et le client avec zeep

Nous avons déployé sur deux serveurs différents.

Dans le premier serveur on a déployé les services de prétraitement sur le port 5000

```
DEBUG:spyne.util.appreg:Registering <spyne.application.Application object at 0x0000020E788D2BB0> as ('preprocessing.inpoda.services', 'Application')
INFO:root:listening app1 to http://127.0.0.1:5000
INFO:root:wsdl app1 is at: http://localhost:5000/?wsdl
```

Dans le second serveur on a déployé les services d'Analyse de données sur le port 5001

```
DEBUG:spyne.util.appreg:Registering <spyne.application.Application object at 0x0000017A2E41A640> as ('analysis.inpoda.services', 'Application')
INFO:root:listening app2 to http://127.0.0.1:5001
INFO:root:wsdl app2 is at: http://localhost:5001/?wsdl
```

Ensuite un client zeep sous forme console pour faire les requêtes

Menus:

```
------------- BIENVENUE DANS LA CONSOLE DE CONSOMMATION DE SERVICES  --------------
-----------------------------------------


Les différents services:
1  . Service de preprocessing de données
2  . Service d'analyse de données
-1 . Quitter
```

```
 ------------- BIENVENUE DANS LA CONSOLE DE PREPROCESSING DE DONNÉES  -------------
 ----------------------------------------


 Les différentes fonctionnalités:
 0  . Publier un ensemble de tweets
 1  . Identification auteur de la publication
 2  . Extraction des hashtags
 3  . Analyse de sentiment
 4  . Identification du/des topic
 -1 . Quitter
```

```
 ------------- BIENVENUE DANS LA CONSOLE D'ANALYSE DE DONNEES  -------------
----------------------------------------

Les différentes fonctionnalités:
1  . Top K hashtags
2  . Top K utilisateurs
3  . Top K topics
4  . Nombre de publications par utilisateur
5  . Nombre de publications par hashtag
6  . Nombre de publications par topic
-1 . Quitter
```

Préprocessing

→ Publications d'un ensemble de tweets

```
Enter your choice: 0
Publication d'un ensemble de tweets...

This may take a while...
Tweet 1421616335700824064 successfully saved
Tweet 1421599703116943360 successfully saved
Tweet 1421599163561742339 successfully saved
Tweet 1421591889095057416 successfully saved
Tweet 1421582795294617605 successfully saved
Tweet 1421581383454052359 successfully saved
Tweet 1421575939700445184 successfully saved
Tweet 1421569996858269697 successfully saved
```

→ Identification auteur

```
IDENTIFYING AUTHOR...
AUTHOR of tweet **Goumin des éléphants joueurs la même fatigue même 😩 #twitter225** is 1339914264522461187
AUTHOR of tweet **@ericbailly24 @maxigr04del  mes tontons vous avez fait votre part , JO prochain on ira en demi final au moins. BRAVO à v
AUTHOR of tweet **Ah oui le sommeil là sera compliqué. #CIV  est éliminé des JO , Ahi on peut faire ça ?** is 1339914264522461187
AUTHOR of tweet **31 juillet , journée internationale de la femme africaine ❤️  #jifa** is 1339914264522461187
AUTHOR of tweet **Le pedigree 😂😂😂😂 https://t.co/D3Rv7A2BOF** is 717025418
AUTHOR of tweet **@isabelle170516 @leonna_julie @Steiner2502 Vous avez tt à fait raison! le silence incompréhensible du gouver-noument et
 leur implication à ce plan  diabolique maquillé!** is 992904738516717570
AUTHOR of tweet **@LynLyna12 @leonna_julie La grande muette continue et continuera de le rester! À part quelques irréductibles à la retrai
AUTHOR of tweet **Under wsh 😳😳😳** is 736523371
AUTHOR of tweet **Les bains d'apollon #versailles #nocturne #appollon à Château de Versailles https://t.co/SFGXP06r61** is 1471684208
```

→ Extraction hashtag

```
Enter your choice: 2

EXTRACTING HASHTAGS...
TWEET:  Goumin des éléphants joueurs la même fatigue même 😩 #twitter225
HASHTAG of this tweet:#twitter225
-------------------------
TWEET:  @ericbailly24 @maxigr04del  mes tontons vous avez fait votre part , JO prochain on ira en demi final au moins. BRAVO à vous . #SupportriceMazo #domie #CIV
HASHTAG of this tweet:#SupportriceMazo #domie #CIV
-------------------------
TWEET:  Ah oui le sommeil là sera compliqué. #CIV  est éliminé des JO , Ahi on peut faire ça ?
HASHTAG of this tweet:#CIV
-------------------------
TWEET:  31 juillet , journée internationale de la femme africaine ❤️  #jifa
HASHTAG of this tweet:#jifa
-------------------------
TWEET:  Le pedigree 😂😂😂😂 https://t.co/D3Rv7A2BOF
HASHTAG of this tweet:No hashtags found
```

→ Analyse de sentiment

```
TWEET:  31 juillet , journée internationale de la femme africaine ❤️  #jifa
Sentiment of this tweet is NEUTRAL
-------------------------
TWEET:  Le pedigree 😂😂😂😂 https://t.co/D3Rv7A2BOF
Sentiment of this tweet is NEUTRAL
-------------------------
TWEET:  @isabelle170516 @leonna_julie @Steiner2502 Vous avez tt à fait raison! le silence incompréhensible du gouver-noument et des merdias
 leur implication à ce plan  diabolique maquillé!
Sentiment of this tweet is NEGATIVE
-------------------------
TWEET:  @LynLyna12 @leonna_julie La grande muette continue et continuera de le rester! À part quelques irréductibles à la retraite?
Sentiment of this tweet is POSITIVE
-------------------------
TWEET:  Under wsh 😳😳😳
Sentiment of this tweet is NEUTRAL
```

→ Identification topic

```
IDENTIFYING TOPIC...
TWEET:  Goumin des éléphants joueurs la même fatigue même 😩 #twitter225
TOPICS of this tweet: General
-------------------------
TWEET:  @ericbailly24 @maxigr04del  mes tontons vous avez fait votre part , JO prochain on ira en demi final au moins. BRAVO à vous . #SupportriceMazo #dom
TOPICS of this tweet: Sports Event<->Person<->Person<->Athlete<->Athlete
-------------------------
TWEET:  Ah oui le sommeil là sera compliqué. #CIV  est éliminé des JO , Ahi on peut faire ça ?
TOPICS of this tweet: Sports Event
-------------------------
TWEET:  31 juillet , journée internationale de la femme africaine ❤️  #jifa
TOPICS of this tweet: General
-------------------------
TWEET:  Le pedigree 😂😂😂😂 https://t.co/D3Rv7A2BOF
TOPICS of this tweet: General
```

Analysis

→ Top K hashtag

```
Enter your choice: 1
Enter the number of top hashtags: 3
1 -> civ: 2 tweets with this hashtag
2 -> twitter225: 1 tweets with this hashtag
3 -> supportricemazo: 1 tweets with this hashtag
```

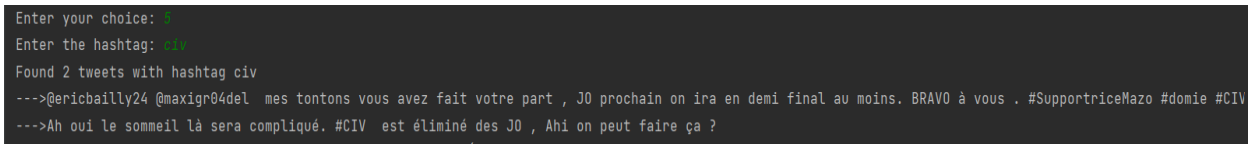
→ Top K topic

```
Enter your choice: 3
Enter the number of top topics: 5
General: 3 tweets about this topic
Sports Event: 1 tweets about this topic
Person: 1 tweets about this topic
Athlete: 1 tweets about this topic
Movie: 1 tweets about this topic
```

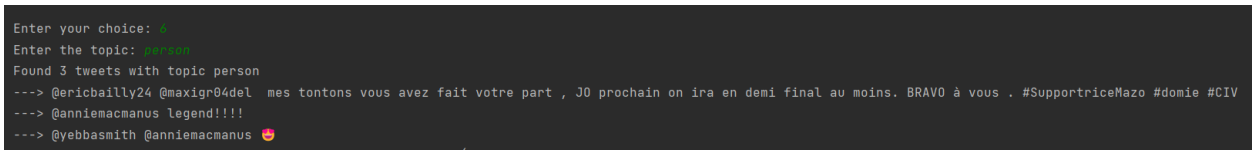
→ Top K users:

```
Enter your choice: 2
Enter the number of top users: 5
1339914264522461187: 4 tweets by this user
372993152: 2 tweets by this user
717025418: 1 tweets by this user
992904738516717570: 1 tweets by this user
736523371: 1 tweets by this user
```

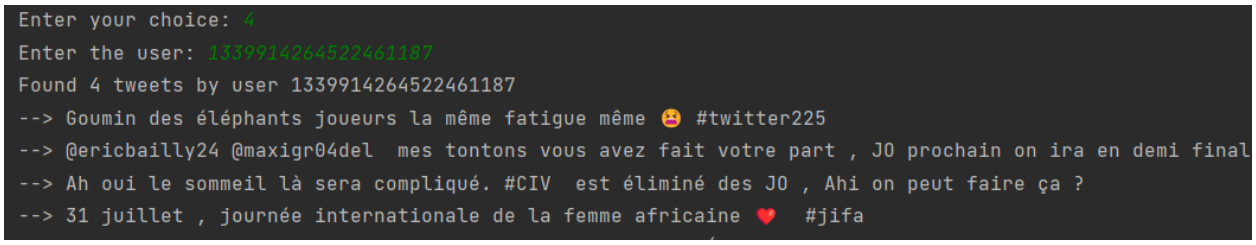
→ nombre de publications par hashtag



→ Nombre de publications par topic:



→ Nombre de publications par utilisateur:



Lien repository github: https://github.com/abdoufermat5/inpoda_web_services