

A complex network diagram with numerous nodes of various sizes and colors (white, blue, orange, green, pink, yellow) connected by thin grey lines. Some nodes are highlighted with larger, colored circles. Three rectangular boxes are overlaid on the network: a blue one on the left, a pink one in the center, and a green one on the right. Various labels are scattered throughout the network, including 'data brokers', 'database agriculture', 'genomics', 'analytics', 'search', 'privacy', 'predictive policing', 'curation', 'global finance', 'urban informatics', 'surveillance', 'sharing economy', 'archiving', 'smart cities', 'regulation', 'lot', 'infrastructure', and 'technology'.

**UNIVERSITE DE LA NOUVELLE-CALEDONIE**

# **RAPPORT DE PROJET TUTORE**

**CREATION D'UNE INTERFACE D'ENVOI ET  
D'ACQUISITION AUTOMATIQUE DE DONNEES  
DANS DES RESEAUX SOCIAUX POUR UNE  
ANALYSE TEXTUELLE**

**MACCARINELLI Chloé et MARTIN Thomy**

# Table des matières

1) L'environnement de travail .....	3
2) Les fonctionnalités d'OPING.....	3
3) L'application OPING .....	4
1- Le fond (back-end) .....	4
L'API Facebook et la création de l'application OPING:.....	4
L'installation de L'API :.....	4
Les Jetons d'accès (Access Tokens) :.....	6
Les graphes :.....	6
Les permissions :.....	7
2- L'interface (Frontend).....	8
Fonctionnement de l'application :.....	8
4) L'implémentation d' OPING.....	12
5) L'API Tweeter .....	14
6) L'analyse des données récupérées .....	15
Conclusion .....	17
Remerciements .....	18
Références .....	19

De nos jours, les réseaux sociaux sont omniprésents. Ils sont utilisés aussi bien pour partager nos centres d'intérêts que pour demander à un groupe de personnes son avis sur des sujets variés. Leur finalité est multiple : certains permettent de mettre en évidence ses compétences dans un domaine, d'autres de promouvoir ses propres produits... Ils permettent de garder un œil sur l'actualité ou *a contrario* de diffuser de nombreuses «mauvaises informations (fake news)». Quand bien même ils peuvent s'avérer être néfastes par bien des égards (contenus inappropriés ...), ils restent le meilleur moyen de communication massive. Dès lors, comment restituer une information de manière cohérente quand le contexte ne s'y prête pas. Effectivement le sens premier d'une information véhiculée sur les réseaux peut être aisément altéré.

Ainsi, il nous a été proposé de travailler sur le sujet suivant :

« Création d'une interface d'envoi et d'acquisition automatique de données dans des réseaux sociaux pour une analyse textuelle »

Étant donné que nous souhaitons appliquer des algorithmes en analyse de sentiments, nous nous sommes fixés pour objectif d'obtenir des opinions de différents domaines.

Dans notre cas l'évènement le plus médiatisé était la coupe du monde de football 2018 sur lequel nous avons récupéré des commentaires à partir de l'application créée. Ces commentaires ont été analysés brièvement à partir d'un outil open source.

Par manque de temps l'interprétation des résultats (avec un expert en traitement du langage) n'est pas intégrée dans ce rapport.

Nous allons donc dans la suite de ce rapport détailler les étapes de réalisation de l'application «Oping» et de l'utilisation des données acquises par l'outil.

Les données ne sont pas prétraitées avant l'analyse. Cela demande un travail conséquent et une expertise en traitement automatique du langage. Nous avons néanmoins été guidés par un spécialiste dans l'utilisation d'outil d'analyse de données textuelles. Ainsi dans le cadre de ce projet, pour analyser ces jeux de données, nous utilisons CORTEXT, une plateforme développée principalement par l'Institut National de la Recherche Agronomique (INRA) et Le Laboratoire Interdisciplinaire Sciences Innovations Sociétés (LISIS). CORTEXT offre de nombreuses fonctionnalités dont nous parlerons dans la suite de ce rapport.

## 1) L'environnement de travail



Pour notre projet, nous travaillons sous PhpStorm, un IDE développé par JetBrains. Pour le « versionnage » de notre application mais également pour le suivi de notre code, nous utilisons Git.

Git est un logiciel de gestion de versions, très utilisé par les développeurs. En effet, Git est utilisé pour gérer des codes sources. L'interface de PhpStorm permet en quelques clics de mettre sa version d'un projet en ligne (push), et ainsi de tenir au courant les autres développeurs du projet des modifications qui ont été effectuées. Ils peuvent aussi récupérer ce code (pull) et le modifier à leur tour.

Pour la persistance des données nous avons opté pour le système de gestion de base de données relationnel MySQL. Pour faciliter l'exploitation des données nous utilisons phpMyAdmin qui est simple d'utilisation pour les requêtes au vu du nombre de données dont nous disposons pour le moment (des milliers de mots).

## 2) Les fonctionnalités d'OPING

La figure 1 : Cas d'utilisation présente les cas d'utilisations d'OPING :

- Publier simultanément dans différents groupes Facebook de l'utilisateur
- récupérer les commentaires sur des publications précises (par id\_publication)
- récupérer les commentaires sur toutes les publications en rapport avec un thème donné

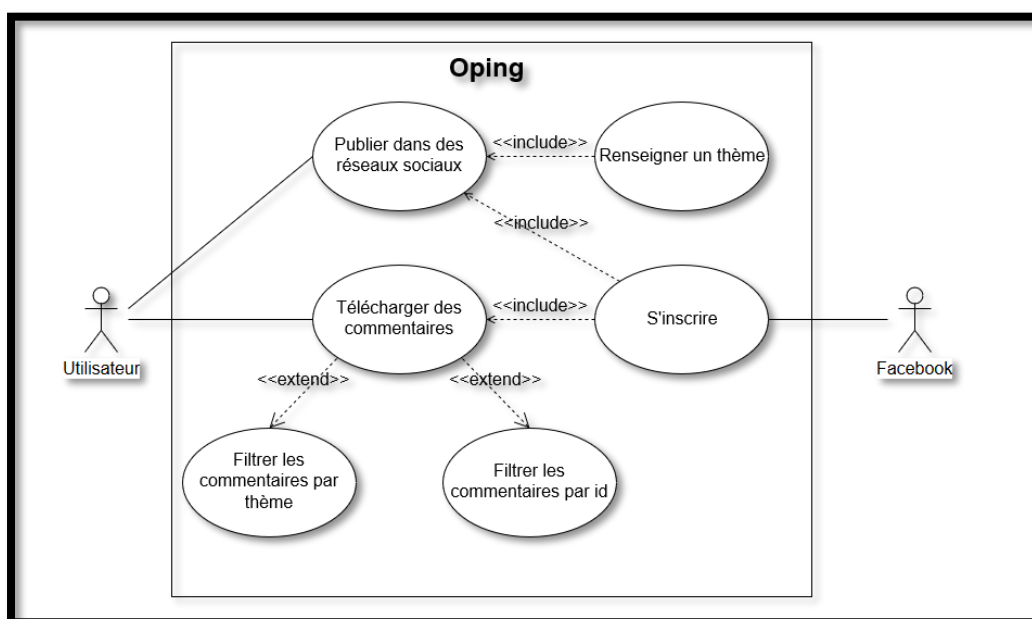


Figure 1 : Cas d'utilisation

### 3) L'application OPING

#### 1- Le fond (back-end)

L'application communique avec l'API Facebook pour l'acquisition de données (commentaires..). D'autres réseaux sociaux seront intégrés.


#### L'API Facebook et la création de l'application OPING:

Une API (Application Programming Interface), est une interface qui permet d'interagir (plus facilement) avec différents programmes (réception et transmission de données). Nous avons donc utilisé l'API Graph de Facebook.

Les requêtes aux bases de données de Facebook se font au travers d'une application «développeur» Facebook.

#### L'installation de L'API :

#### Téléchargement de la librairie Facebook et intégration dans notre projet :



Nom	Modifié le	Type	Taille
.idea	05/07/2018 19:26	Dossier de fichiers	
assets	05/07/2018 19:26	Dossier de fichiers	
bootstrap	05/07/2018 19:26	Dossier de fichiers	
controller	05/07/2018 19:26	Dossier de fichiers	
css	05/07/2018 19:26	Dossier de fichiers	
Facebook	19/07/2018 10:00	Dossier de fichiers	
images	05/07/2018 19:26	Dossier de fichiers	
model	05/07/2018 19:26	Dossier de fichiers	
view	05/07/2018 19:26	Dossier de fichiers	
index.php	05/07/2018 19:26	Fichier PHP	2 Ko
oping.sql	05/07/2018 19:26	PostgreSQL	6 Ko

Nom	Modifié le	Type	Taille
Authentication	05/07/2018 19:26	Dossier de fichiers	
Exceptions	05/07/2018 19:26	Dossier de fichiers	
FileUpload	05/07/2018 19:26	Dossier de fichiers	
GraphNode	05/07/2018 19:26	Dossier de fichiers	
Helpers	05/07/2018 19:26	Dossier de fichiers	
Http	05/07/2018 19:26	Dossier de fichiers	
HttpClient	05/07/2018 19:26	Dossier de fichiers	
PersistentData	05/07/2018 19:26	Dossier de fichiers	
PseudoRandomString	05/07/2018 19:26	Dossier de fichiers	
Url	05/07/2018 19:26	Dossier de fichiers	
autoload.php	05/07/2018 19:26	Fichier PHP	3 Ko
Facebook.php	05/07/2018 19:26	Fichier PHP	19 Ko
FacebookApp.php	05/07/2018 19:26	Fichier PHP	3 Ko
FacebookBatchRequest.php	05/07/2018 19:26	Fichier PHP	10 Ko
FacebookBatchResponse.php	05/07/2018 19:26	Fichier PHP	5 Ko
FacebookClient.php	05/07/2018 19:26	Fichier PHP	8 Ko
FacebookRequest.php	05/07/2018 19:26	Fichier PHP	13 Ko
FacebookResponse.php	05/07/2018 19:26	Fichier PHP	11 Ko
fb-callback.php	05/07/2018 19:26	Fichier PHP	3 Ko
login.php	05/07/2018 19:26	Fichier PHP	2 Ko
polyfills.php	05/07/2018 19:26	Fichier PHP	2 Ko
SignedRequest.php	05/07/2018 19:26	Fichier PHP	9 Ko

#### Création d'un compte développeur et création de l'app :

Après avoir créé un compte pour développeur, on crée notre application ; celle-ci reçoit alors un identifiant d'app et une clé secrète. La figure 2 : Tableau de bord Facebook présente le tableau de bord de l'interface pour la gestion de l'application «développeur».

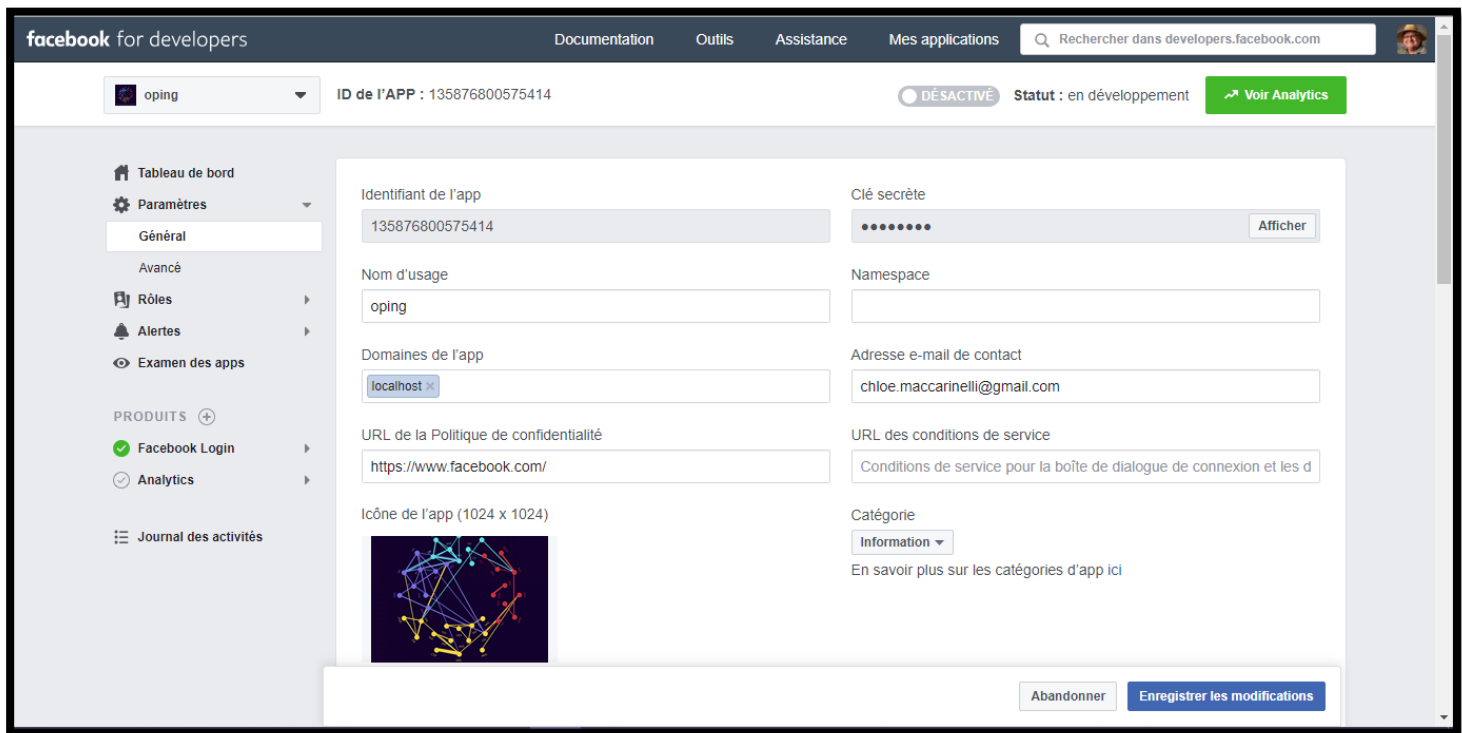


Figure 2: Tableau de bord Facebook

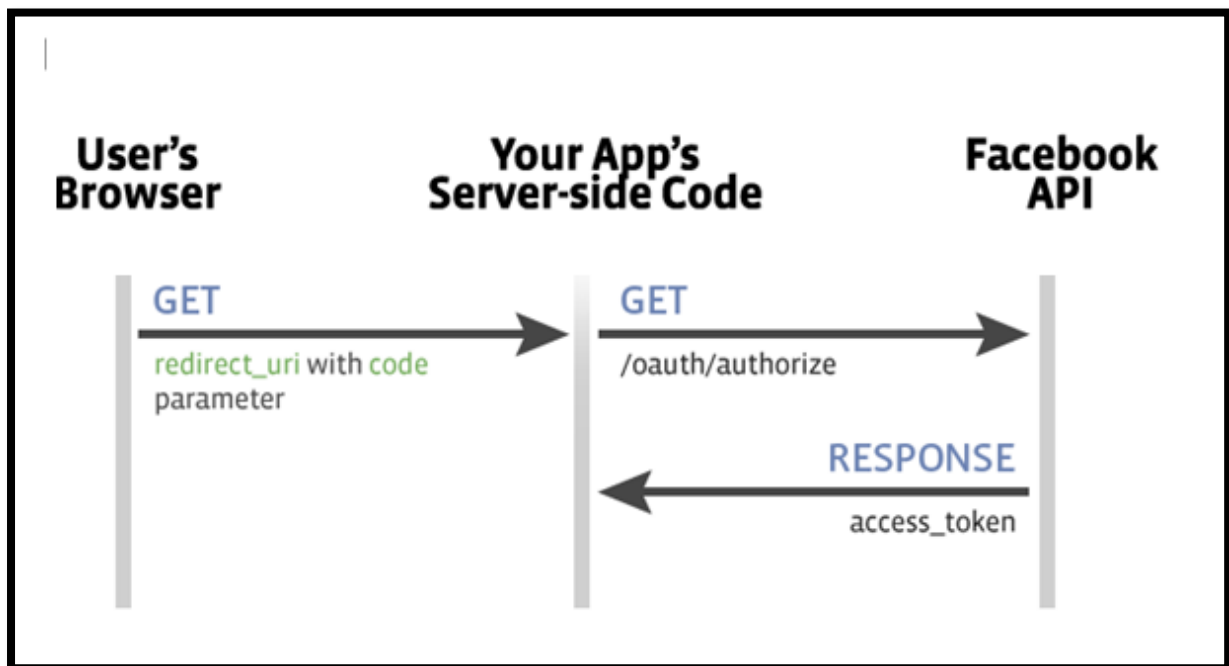


Figure 3: Communication vers l'API

L'API impose de générer un « Jeton d'accès (Access Token) » pour utiliser ses fonctionnalités via l'interface Oping.

La figure 3 : Communication vers API décrit la communication entre l'API Facebook et l'application.

## Les Jetons d'accès (Access Tokens) :

Les jetons d'accès permettent à notre application d'accéder à l'API Graph. Ils remplissent généralement deux fonctions et permettent :

- à notre application d'accéder aux informations d'un utilisateur sans avoir besoin du mot de passe de cet utilisateur.
- d'identifier les données restreintes par l'utilisateur (voir partie permissions).

Tous les points de terminaison de l'API Graph nécessitent un jeton d'accès quel qu'il soit. Chaque fois que nous accédons à un point de terminaison, notre demande doit donc en contenir un.

## Les graphes :

En réponse à chacune des requêtes valides qui lui sont envoyées, l'API Facebook retournera un graphe contenant les informations demandées, à condition bien sûr d'être en règle au niveau des jetons d'accès... Les Graphes nous permettent d'obtenir le contenu des commentaires des publications, les dates de commentaires, etc ...

D'après la figure 3.1 on définira un type de nœud du graphe comme étant une entité distincte dans Facebook, c'est-à-dire un utilisateur, un commentaire, une page, un groupe.... Chaque type de nœud a un identifiant unique.

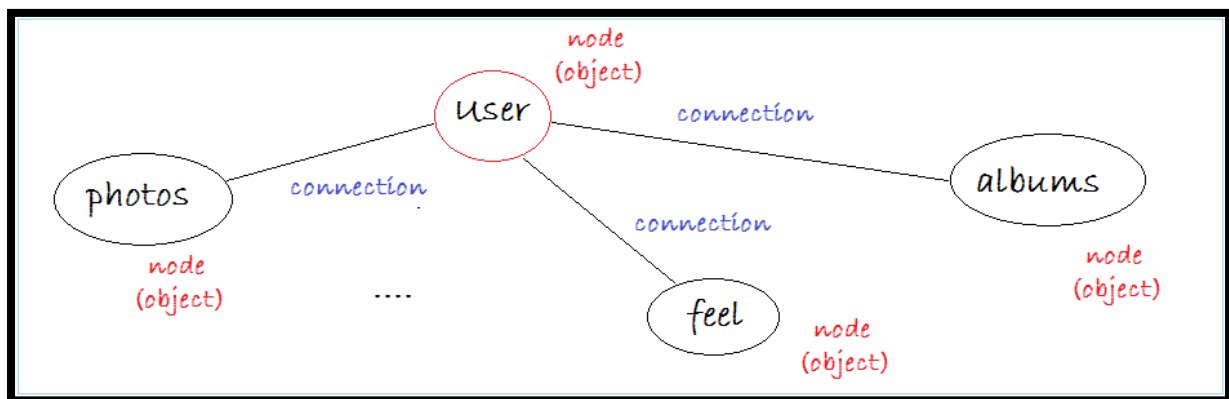


Figure 4.1: les nœuds (nodes)



```
{
  "name": "Coca-Cola",
  "id": "820882001277849"
}
```

D'après la figure 4.1 la réponse standard (format JSON) fournit par L'API Graph d'une requête sur les attributs « id » et « nom » lié à la page Coca-Cola.

## Les permissions :

L'API Facebook propose de nombreuses fonctionnalités pour accéder aux types de nœud et à leurs attributs. L'API Graph restreint l'accès aux entités via des permissions d'accès propres à chacune d'elles. Le développeur doit justifier de l'utilisation de l'accès à ses permissions dans son application avant de la rendre publique, c'est-à-dire accessible aux autres utilisateurs Facebook. Pour cela, via le tableau de bord de l'application « développeur » (Figure 2 : Tableau de bord Facebook) il envoie des demandes d'accès à ses permissions. Facebook validera ou non les demandes après avoir testé notre application.

Les permissions demandées pour l'utilisation d'Oping sont principalement des demandes d'autorisation d'accès aux données de Groupes (API Goups) et aux données de l'utilisateur (nom, identifiant,...).

La figure 4 : Permissions, présente un exemple de formulaire de demande de permissions à soumettre.

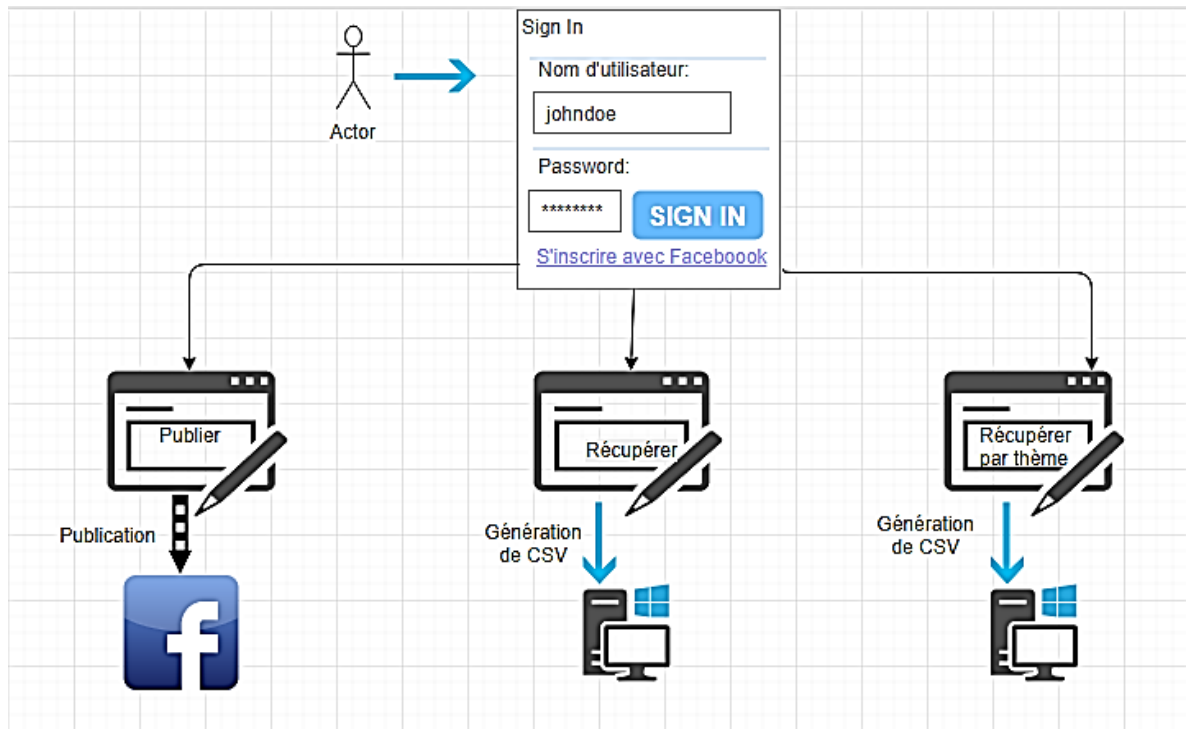
Figure 5: Permissions



## 2- L'interface (Frontend)

### Fonctionnement de l'application :

Comme évoqué dans le 2), la figure 5 : Utilisation, présente les 3 principales fonctionnalités d'OPING. Ces fonctionnalités sont détaillées plus bas.



### Inscription et connexion :

Figure 6: Utilisation

L'utilisateur de l'application doit être inscrit dans 'OPING'. D'après la figure 6 : Connexion et API, l'utilisateur d'OPING renseigne ses identifiants Facebook via le module sécurisé de connexion de l'API Facebook.

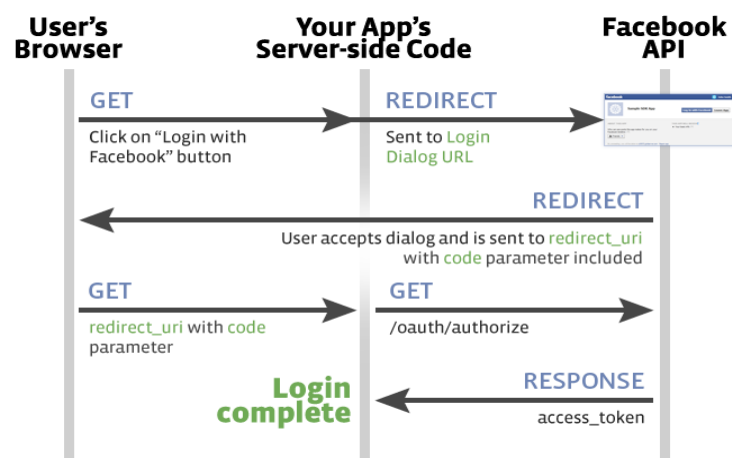
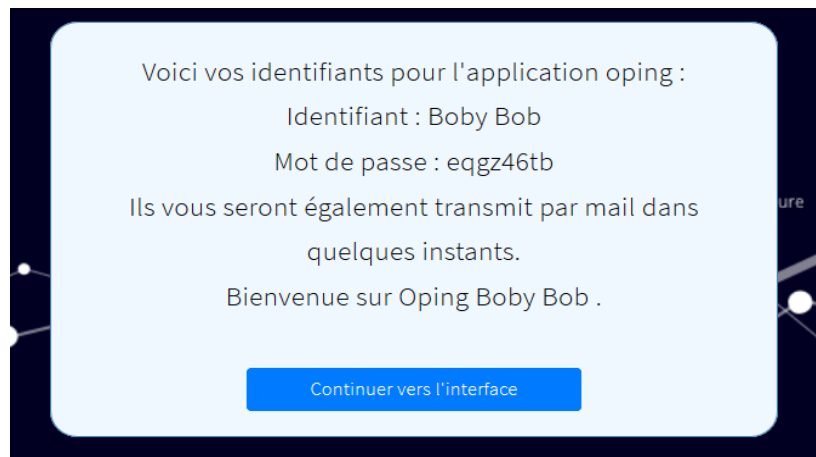


Figure 6: Connexion et API

Après validation de sa connexion à Facebook, Oping inscrit son nom d'utilisateur Facebook et lui génère un mot de passe d'après la figure 7 : génération mot de passe. Ces données sont enregistrées en base de données en tant qu'identifiant de connexion.

Pour obtenir le nom d'utilisateur, l'API Facebook permet une connexion au compte Facebook via le «bouton de connexion : fb.login button» (redirection vers login.php de l'API Graph Facebook). Ce module renvoie les informations de l'utilisateur selon les permissions obtenues.

(Voir code **Annexe 1**)



**Figure 7: génération mot de passe**

## **Les fonctionnalités :**

- ➔ publication sur un ou plusieurs groupes simultanément (voir code **Annexe 2**)
- ➔ la récupération de commentaires par id de publication (voir code **Annexe 3**)
- ➔ la récupération de commentaires par thème (voir code **Annexe 4**)

## **Publication sur un ou plusieurs groupes simultanément**

Oping permet de publier dans différents groupes simultanément. D'après la figure 8 : formulaire de publication, à chaque publication l'utilisateur renseigne un thème, un texte et choisit les groupes cibles (auxquels il appartient). Il peut également rajouter des liens à sa publication.

**Figure 8: Formulaire de Publication**

Dès que la publication est effective, le Graph renvoi les informations la concernant (thèmes, id de publication, id de groupe). Ces données sont sauvegardées dans la base de données.

Les identifiants permettront de récupérer, ultérieurement, les commentaires via les requêtes de l'API (voir partie 3) 1- Les graphes).

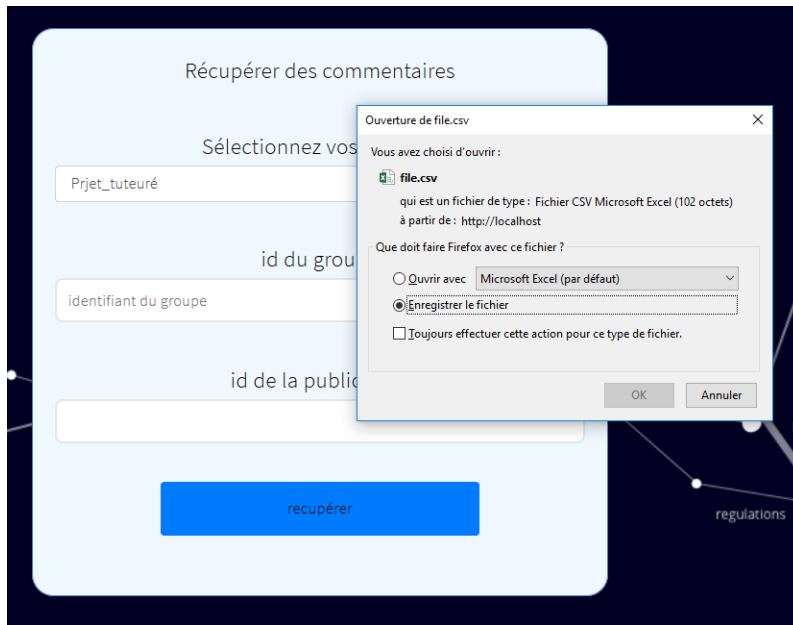
### Récupérer les commentaires des publications

D'après la figure 9 : Formulaire de récupération par id, l'utilisateur peut récupérer les commentaires d'une publication en renseignant l'Id de la publication et en renseignant le groupe correspondant (avec son id ou dans la liste déroulante).

**Figure 9: Formulaire de Récupération par id**

Pour chaque commentaire, les informations suivantes sont enregistrées dans un CSV accessible à l'utilisateur par téléchargement (figure 10 : téléchargement du CSV):

- La date
- L'heure
- L'identifiant
- Le contenu



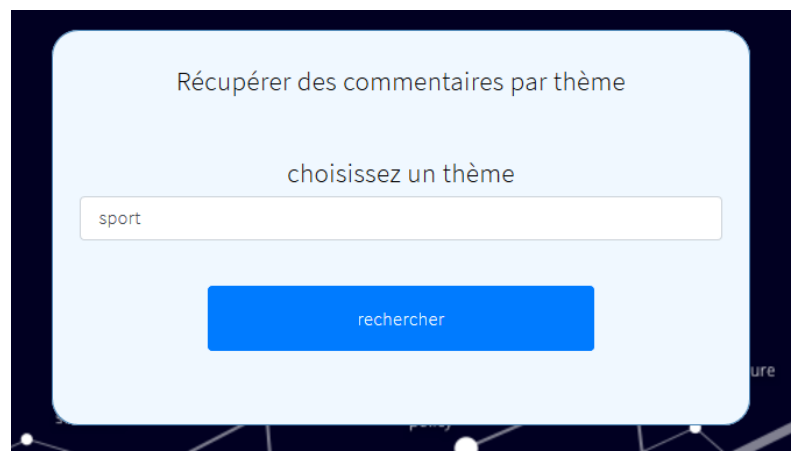
**Figure 10 : téléchargement du CSV**

## Récupérer des commentaires par thème :

Comme vu plus haut, l'utilisateur renseigne un thème lors de sa publication. Ce thème est enregistré en base de données avec l'identifiant de la publication. Ces données nous permettent de récupérer des commentaires en les triant par thème.

Exemple d'acquisition de données en rapport avec le football.

D'après figure 11 : formulaire de récupération par thème, il suffit de sélectionner le thème en rapport avec ses publications.



**Figure 11 : Formulaire de récupération par thème**

#### 4) L'implémentation d' OPING

Le design pattern Modèle-Vue-Contrôleur (MVC), présenté dans la figure 12 : modèle MVC, est un modèle d'architecture logiciel permettant la séparation des données (model), de l'interface (vue) et des traitements (contrôleur). Ce patron de conception simplifie les maintenances logicielles ainsi que l'évolution de ces derniers.

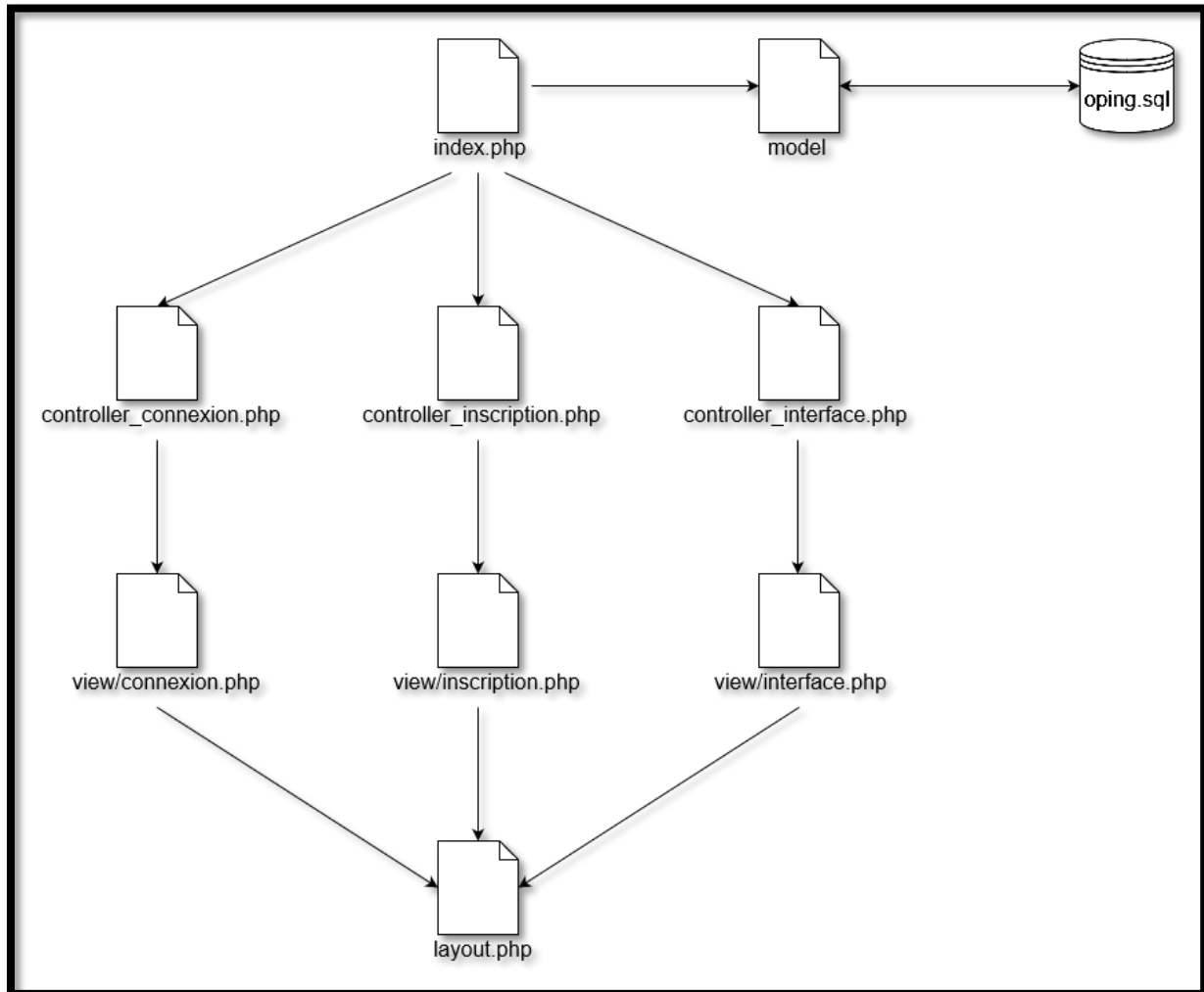


Figure 12 : modèle MVC

Dans la figure 13 : Diagramme de classe, on retrouve les liens entre nos classes principales (voir aussi schéma entité/association en annexe 0). Les figures 14 et 15 représentent les diagrammes de séquence pour les différentes fonctionnalités d'OPING, la publication et la récupération de commentaires. Elles mettent en évidence la communication entre notre interface, notre contrôleur et l'API.

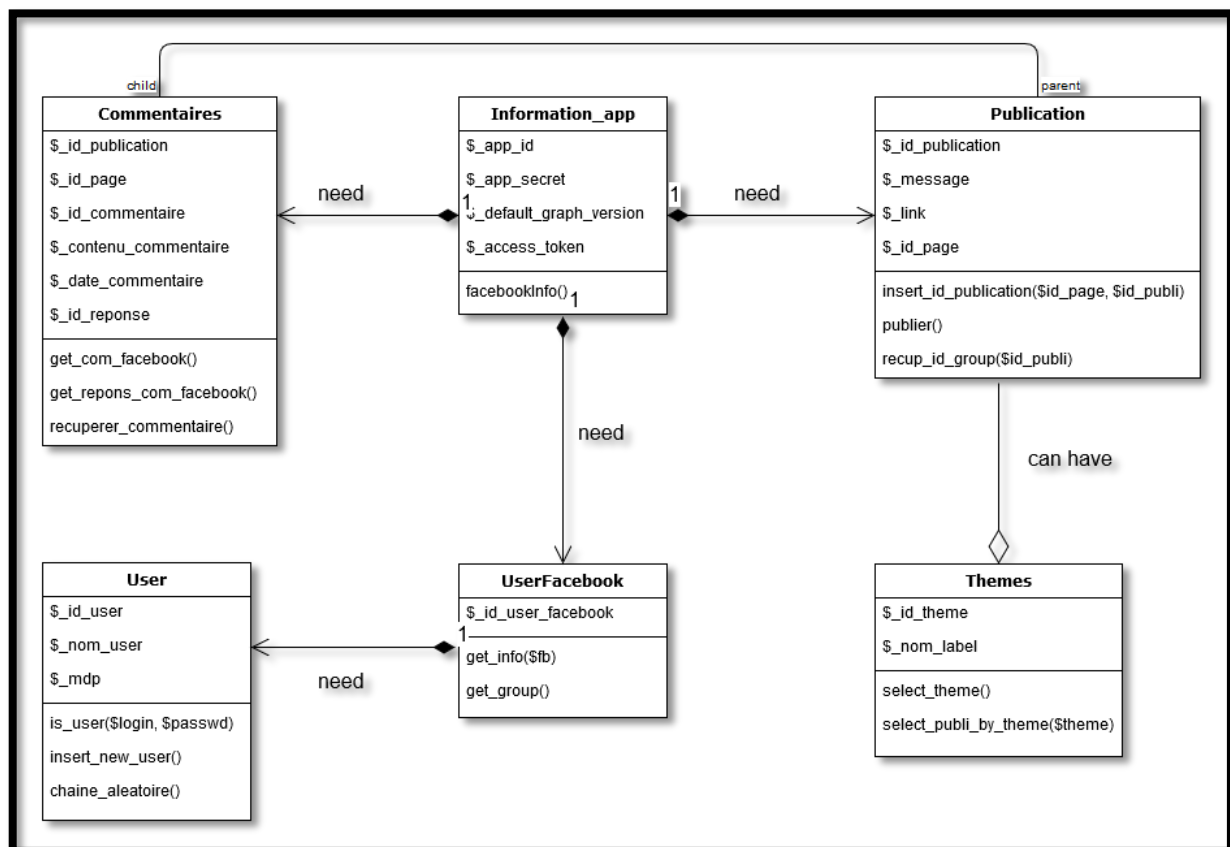


Figure 13 : Diagramme de Classe

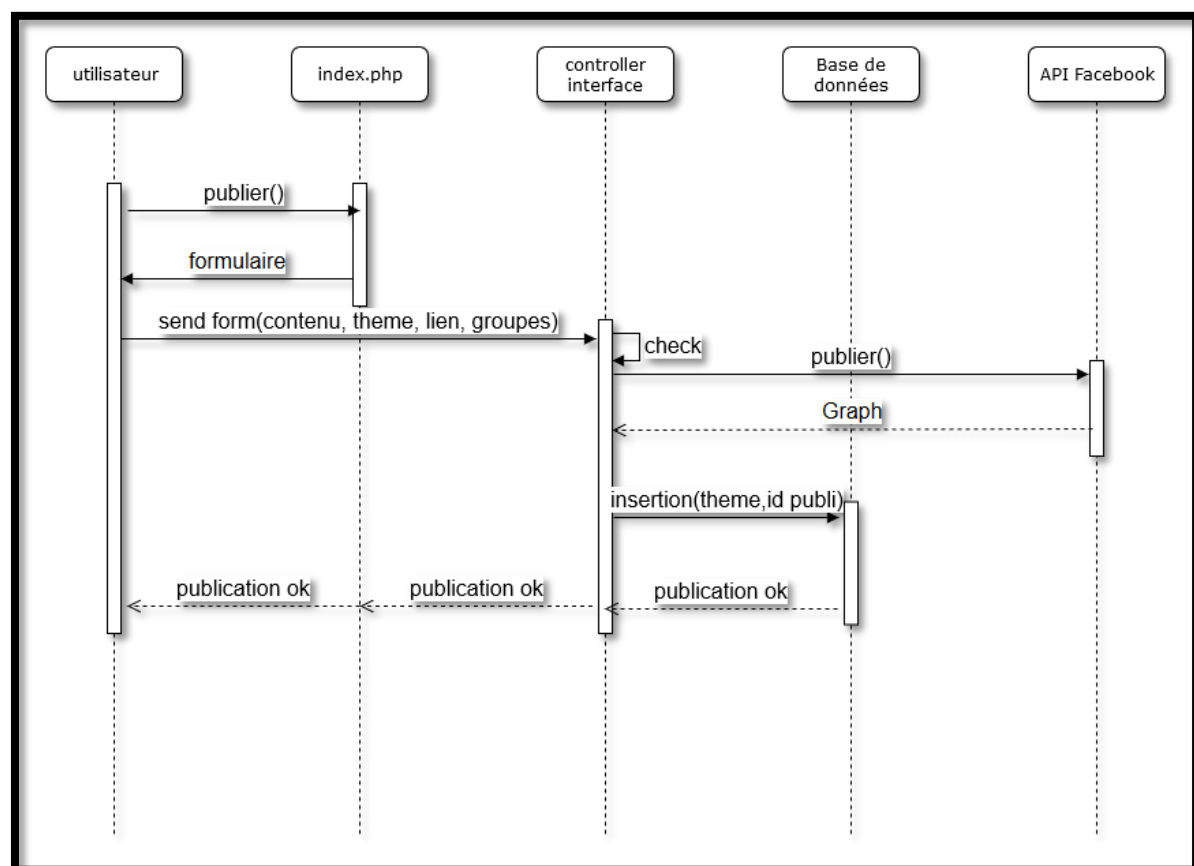


Figure 14 : Diagramme de séquence : Publication

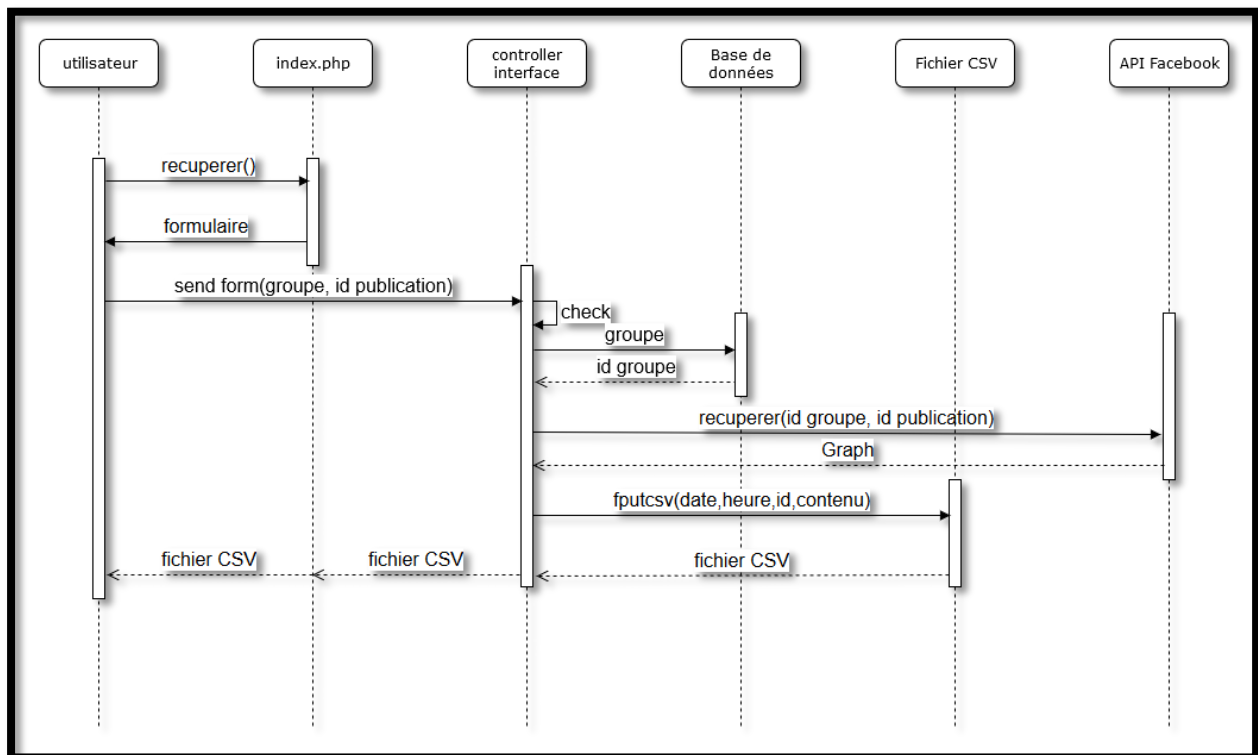


Figure 15 : Diagramme de séquence : Récupération de commentaires

## 5) L'API Tweeter

En parallèle de la création d'OPING avec Facebook, nous avons travaillé sur l'API Twitter.

L'utilisation de l'API twitter est beaucoup moins restrictive en termes de permissions sur le type de données récupérées mais nous restreint dans la quantité.

Pour récupérer des tweets nous avons comme pour Facebook créé un compte développeur et une app avec une clé, un identifiant app et un jeton d'accès. Nous avons également téléchargé une bibliothèque appelée cURL PHP qui permet d'effectuer des requêtes vers un site web donné (ici tweeter). Nous avons pu, grâce à cette bibliothèque, simplifier notre code pour la récupération des tweets et ainsi ne pas avoir à télécharger la librairie de l'API tweeter.

La récupération est très simple, après avoir réussi à accéder à Tweeter via notre code nous pouvons récupérer facilement les tweets comme dans l'exemple de **l'Annexe 5 : Tweeter**.

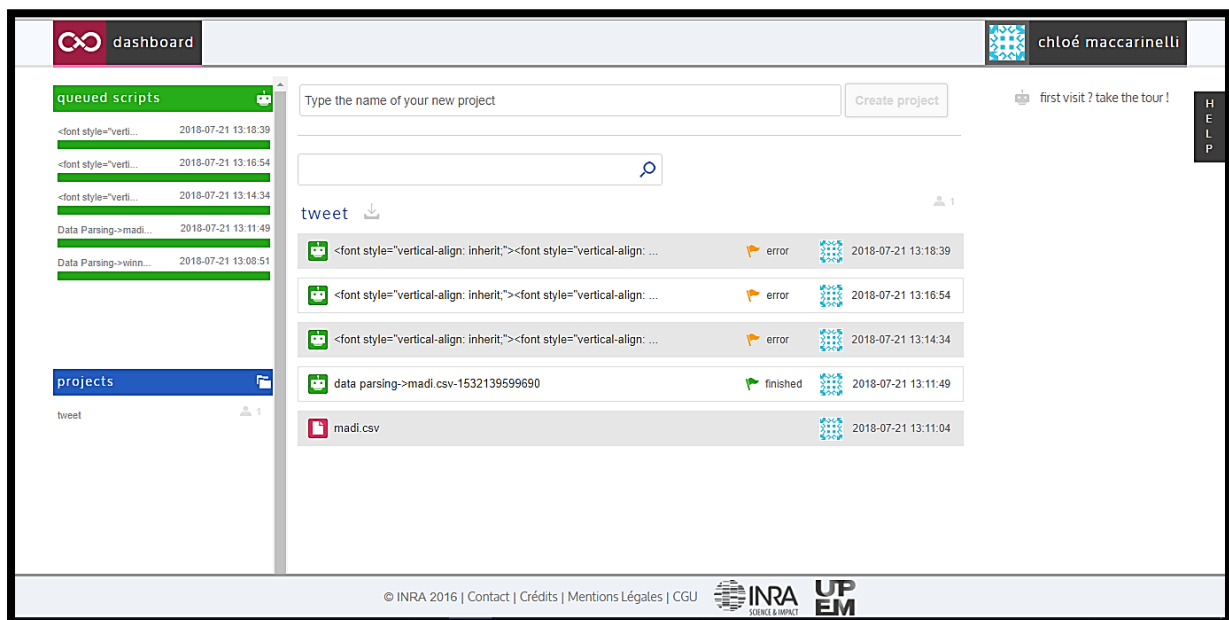


## 6) L'analyse des données récupérées

L'objectif de la récupération de données est, ici, de pouvoir faire du traitement automatique du langage naturel (TALN). Mais qu'est-ce donc ?

Le TALN est né sous une forme assez différente dans les années 50 avec le traitement automatique des conversations et le célèbre « test de Turing ». Aujourd'hui le TALN est notamment utilisé pour analyser le contenu et la signification de données textuelles, comme par exemple les sentiments qui ressortent d'un texte ou alors la fréquence d'apparition de certains mots.

Le TALN propose différents champs de recherche tels que l'analyse syntaxique, sémantique, l'extraction d'informations ou même le traitement des signaux (la parole par exemple).



**Figure 16 : CORTEXT Dashboard**

Ici, nous nous sommes concentrés sur l'extraction d'information avec l'outil CORTEXT présenté dans l'introduction.

Nous pouvons effectuer différentes analyses en ce qui concerne l'extraction d'information :

- ✚ Fouille de textes
- ✚ Recherche d'information
- ✚ Reconnaissance d'entités nommées
- ✚ Classification et catégorisation de documents
- ✚ Systèmes de tutorat intelligents
- ✚ Analyse de sentiment
- ✚ La recommandation automatique de documents

Les premières analyses réalisées sont sur le thème du football. Les commentaires récupérés proviennent de groupe Facebook assez actifs (en Français). Nous avons effectué une fouille de texte et le résultat est visible dans l'**Annexe 6**.

Les résultats obtenus ne nous permettent pas ici d'affirmer, en suivant cet exemple, que Messi est le meilleur joueur du monde. Nous remarquons simplement que dans le jeu de données fourni, les termes Messi et Meilleur joueur du monde apparaissent souvent ensemble.

L'un des principaux inconvénients des outils d'analyse du langage est que les textes étudiés doivent être à minima structurés et pour certains, en anglais.

Pourquoi en anglais ?

La plupart des intelligences artificielles qui font du traitement de données en français sont « formées » avec des textes tels que les journaux comme le Monde. Elles ne sont pas aussi bien « habituées » au langage courant retrouvé dans les réseaux sociaux que les intelligences « formées » en anglais.

Avec CORTEXT, si le jeu de données n'est pas assez important (des dizaines de milliers de commentaires), il est plus rapide d'obtenir des résultats cohérents avec des données en anglais et des phrases bien construites.

## Conclusion

Dans ce projet, nous avons codé une interface de publication et d'acquisition de données dans des réseaux sociaux. L'objectif étant d'effectuer des analyses textuelles sur les données récupérées. Pour ce faire, nous avons utilisé différentes technologies ainsi que différents langages. Pour l'analyse textuelle nous avons utilisé un logiciel open source.

La réalisation de ce projet a permis d'appréhender le travail en équipe auquel nous serons confrontés en Master tout en restant dans un environnement agile (point sur l'avancée du projet chaque semaine, adaptation des objectifs en conséquence, ..).

Les revues de code avec un professionnel nous ont apporté une méthodologie de "codage" non négligeable, plus particulièrement dans l'organisation du code ; comme cet entrepreneur le dirait : « il faut mettre chaque chose à la bonne place ». Même si cela paraît évident, il n'est pas si simple de l'appliquer dans tous les cas.

L'objectif final de la réalisation de ce projet nous a particulièrement intéressé. Effectivement, l'opinion est présente sur tous les réseaux sociaux or, personne ne prend le temps de faire la part entre l'opinion d'untel ou d'un autre. Et pourquoi ? Tout simplement parce qu'il y a trop d'informations à trier. Ainsi, l'analyse textuelle à laquelle nous soumettons les commentaires obtenus via notre application va permettre, pour des sujets d'actualités, de pouvoir enfin obtenir des statistiques d'opinions en phase avec la réalité, contrairement aux sondages qui sont destinés, la plupart du temps, à un faible pourcentage de la population. Ce travail demande un travail conséquent de prétraitement des masses de données textuelles acquises.

L'application n'est évidemment pas encore optimale, elle sera améliorée pour permettre son utilisation par des entreprises ou autre ; les données ayant de nos jours une très grande valeur.

## **Remerciements**

Ce travail a été effectué dans le cadre de la formation en licence informatique à l'Université de la Nouvelle-Calédonie.

Nous souhaitons remercier Monsieur Tokotoko, notre tuteur, pour la qualité de l'encadrement dont nous avons pu bénéficier tout au long de ce travail. Nous tenons à lui exprimer toute notre reconnaissance pour son soutien durant cette période.

Nous remercions également Monsieur Poircuitte pour l'aide qu'il nous a apporté durant toute la période de revue de code et finalisation du projet.

Nos remerciements s'adressent également à Monsieur Plancq qui nous a apporté de nombreux détails et conseils sur l'analyse et traitement du langage avec l'outil CORTEXT.

Nous souhaitons également remercier Monsieur Flouvat pour les cours de Techno Web dispensés en 2<sup>ème</sup> année de licence, et qui nous ont été très utiles dans la structuration de notre code.

## Références

- <https://www.supinfo.com/articles/single/1345-c-est-quoi-une-api>
- <https://managerv2.cortex.net/login>
- <https://docs.cortex.net/>
- <https://www.facebook.com/>
- <https://twitter.com/>
- Cours de Techno Web 2017 de F.Flouvat
- <https://developers.facebook.com>