
“SafeTweet” RAPPORT

MAY Madeth

AUGER Nathan
DURAND Pierre
LOPEZ Julio
NOUVELIÈRE Benjamin

Table des matières

Introduction	3
Contexte	3
Domaine d'application des fake news	3
Dans un contexte de R&D	3
Réseau social utilisé	4
Choix du langage et du modèle	4
Conception et fonctionnalités	6
Erreur de conception	6
Conception de l'application	7
Fonctionnalités de l'application	8
Organisation/Gestion de projet	11
Répartition des tâches au sein du groupe	12
Temps du travail au sein du groupe	13
Échange avec le client	14
Résultats	14
État du projet à la fin du module	14
Bilan	16
Conclusion	16
Pistes d'amélioration	16
7. Bibliographie	17
8. Annexes	18

1. Introduction

Le projet qui nous a été assigné par M. May est un projet de R&D nommé “Fake News” qui a pour but de lutter contre les fausses informations diffusées notamment sur internet.

a. Contexte

L'accès aux informations sur internet devient facile et instantané grâce aux moyens techniques dont nous disposons (internet illimité, 4G) et à la démocratisation de l'usage des objets connectés et des réseaux sociaux. Cependant, un phénomène mondialement connu est en train de bouleverser notre comportement : les fake news. L'impact de fake news a été étudié sur plusieurs aspects de notre vie quotidienne sur les réseaux sociaux : l'apprentissage, le raisonnement, la prise de décision, etc.

En tant qu'étudiants en Master première année d'informatique à Le Mans Université et dans un cadre R&D, notre travail consiste à étudier une nouvelle approche permettant de détecter les fake news.

b. Domaine d'application des fake news

Lors de ce projet, nous avons d'abord suivi une démarche où nous allions tenter de savoir si n'importe quel tweet serait vrai ou faux et cela pour n'importe quel domaine d'application : politique, sport, virus mondial, harcèlement, ...

Cependant cette démarche était trop ambitieuse car des messages faux de politique ne sont pas les mêmes messages que des messages faux de sport par exemple, ainsi nous avons cherché un domaine par lequel nous sommes touchés, ce qui nous a poussé à choisir le domaine du harcèlement.

Notre application a donc pour but de rassurer ou/et de prévenir les personnes victimes de harcèlement ou de diffamation.

2. Dans un contexte de R&D

Ce projet étant un projet de recherche et développement, il a fallu comparer les différents supports ou technologies afin de rendre notre étude la plus optimale possible.

a. Réseau social utilisé

Les fakes news sont partout que ce soit dans les journaux, les forums, les sites web ou bien encore les réseaux sociaux.

Nous avons choisi de travailler avec les réseaux sociaux car c'est là où l'activité est la plus dense et où les fake news ont le plus de chances d'être vues et relayées par les utilisateurs.

Travailler avec plusieurs réseaux sociaux différents n'était pas utile nous avons donc choisi de nous concentrer sur un en particulier, Twitter, car les messages sont limités en nombre de caractères et c'est le réseau social à la mode.

L'API Twitter a été utilisé pour pouvoir recevoir un flux de messages d'un compte connecté à Twitter.

b. Choix du langage et du modèle

Tout d'abord nous avons dû nous renseigner sur la manière la plus adaptée afin de travailler sur ce projet, nous avons donc réalisé plusieurs benchmarks afin de déterminer dans un premier temps quel langage et technologie nous utiliserons.

Tab 1. Benchmark langage modèle

Langages du modèle	Spécificités
Python	<ul style="list-style-type: none">- Large panel de bibliothèque (Numpy, Panda, NLTK, SpaCy, SciKit-learn)- Intègre facilement des APIs- Performance en temps- Communauté développée et active
JVM	<ul style="list-style-type: none">- Bon panel de bibliothèque (CoreNLP, ND4J, DL4J)- Calcule sur CPU- Machine virtuelle- Communauté développée
C/C++	<ul style="list-style-type: none">- Quelques bibliothèque (CUDA, TensorFlow, Caffe)- Utilisation de pointeurs- Calcule sur GPU- Accède à des APIs de haut niveau et flexibles
JavaScript	<ul style="list-style-type: none">- Très peu de bibliothèque- Intègre mal les APIs

Nous avons choisi Python, comme on peut le voir, il possède un large panel de bibliothèques, des APIs faciles à intégrer et une communauté développée et active.

Ayant choisi python comme langage pour le modèle, nous avons dû choisir la bibliothèque que nous utiliserons.

Tab 2. Benchmark bibliothèque python d'apprentissage automatique

Bibliothèque	Spécificités
TensorFlow	<ul style="list-style-type: none"> - <i>Bien documenté</i> - <i>Grande communauté</i> - <i>Graph statique</i> - <i>Debugging non intuitif</i> - <i>Bonne visualisation de graph</i> - <i>Déploiement très facile</i> - <i>Date de 2011 développé par Google</i>
Pytorch	<ul style="list-style-type: none"> - <i>Documenté</i> - <i>Tutoriels officiels</i> - <i>Graph dynamique</i> - <i>Debugging intuitif</i> - <i>Bonne visualisation de graph</i> - <i>Facile à utiliser</i> - <i>Date de 2016 développé par Facebook</i>

Quand ce benchmark a été réalisé nous n'avions pas d'idée précise de quoi utiliser, au final nous avons utilisé Pytorch car nous avons trouvé un code déjà existant en Pytorch.

Enfin, il a fallu choisir entre une régression linéaire et un réseau de neurones pour notre modèle.

Tab 3. Benchmark technologie modèle

Technologies du modèle	Spécificités
Régression linéaire	<ul style="list-style-type: none"> - <i>Facile et rapide à calculer</i> - <i>Réajustement manuel de la courbe</i>
Réseau de neurones	<ul style="list-style-type: none"> - <i>Réajustement automatiquement de la courbe</i>

Les technologies étant encore obscures pour nous au début du projet nous avons décidé de voir de nous-même quelle technologie serait la mieux à utiliser. Et après comparaison avec notre base de données, le réseau de neurones nous donne des prévisions plus justes que la régression linéaire.

3. Conception et fonctionnalités

a. Erreur de conception

Tout d'abord, nous ne nous étions pas posé la question suivante : "Dans quelles circonstances les utilisateurs utiliseront notre application ?", ainsi notre première idée a été de réaliser une Single Page Application (SPA) accessible de partout, c'est à dire PC/téléphone/tablette.

Afin de réaliser cela nous avons effectué un nouveau benchmark nous permettant de savoir quel framework web serait le mieux pour ce projet.

Tab 4. Benchmark framework SPA

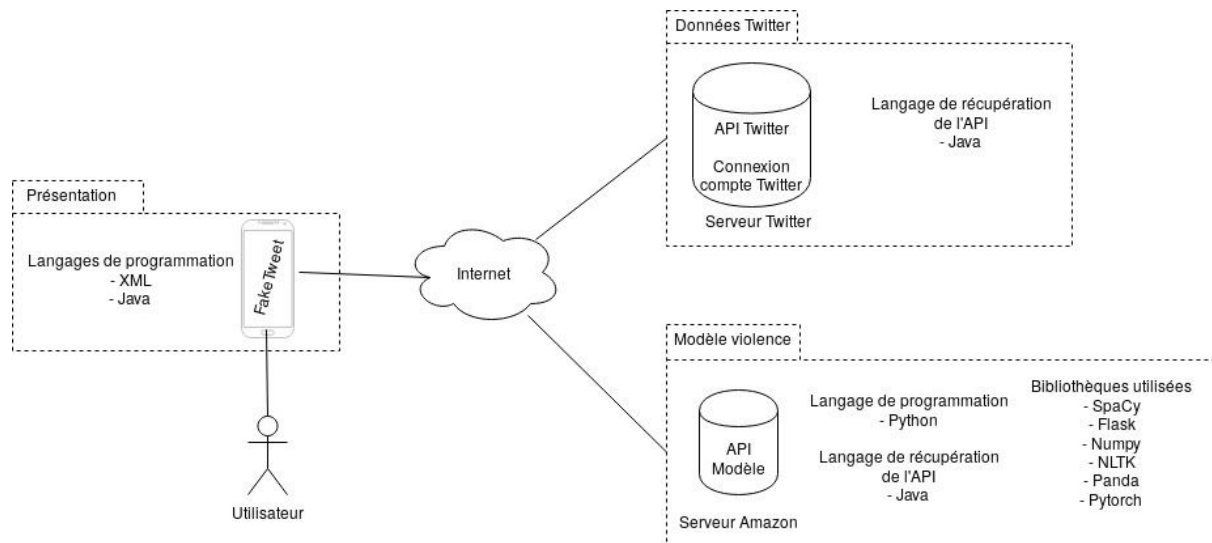
	Angular	React	Vue
Taille	65.5 KB	36.3 KB	30.8 KB
Temps de performance	1.5 s	1 s	1 s
Temps de rechargement	1486.6 ms	796.5 ms	806.5 ms
Apprentissage	Facile	Facile	Très facile

Grâce à ce comparatif nous avons choisi Vue car une partie de l'équipe l'avait déjà utilisé et qu'il était plus léger que ses concurrents.

Nous avons travaillé pendant neuf heures sur cette voie qui n'était pas la bonne, en effet, nous nous sommes finalement posé la question : "Dans quelles circonstances les utilisateurs utiliseront notre application ?" et la réponse était évidente. De nos jours les réseaux sociaux sont utilisés en très grande partie sur les téléphones portables, le choix d'une application mobile nous à sembler être une évidence, l'application ne serait pas une SPA mais une application mobile Twitter-like.

b. Conception de l'application

Figure 1. DAT de l'application "SafeTweet"



La présentation de l'application "SafeTweet" à l'utilisateur se fait grâce à un smartphone via l'application mobile.

Tout d'abord l'utilisateur va se connecter à son compte Twitter afin qu'une timeline des tweets relatifs à son compte soit visible, cette connexion est réalisée grâce à une requête au serveur Twitter.

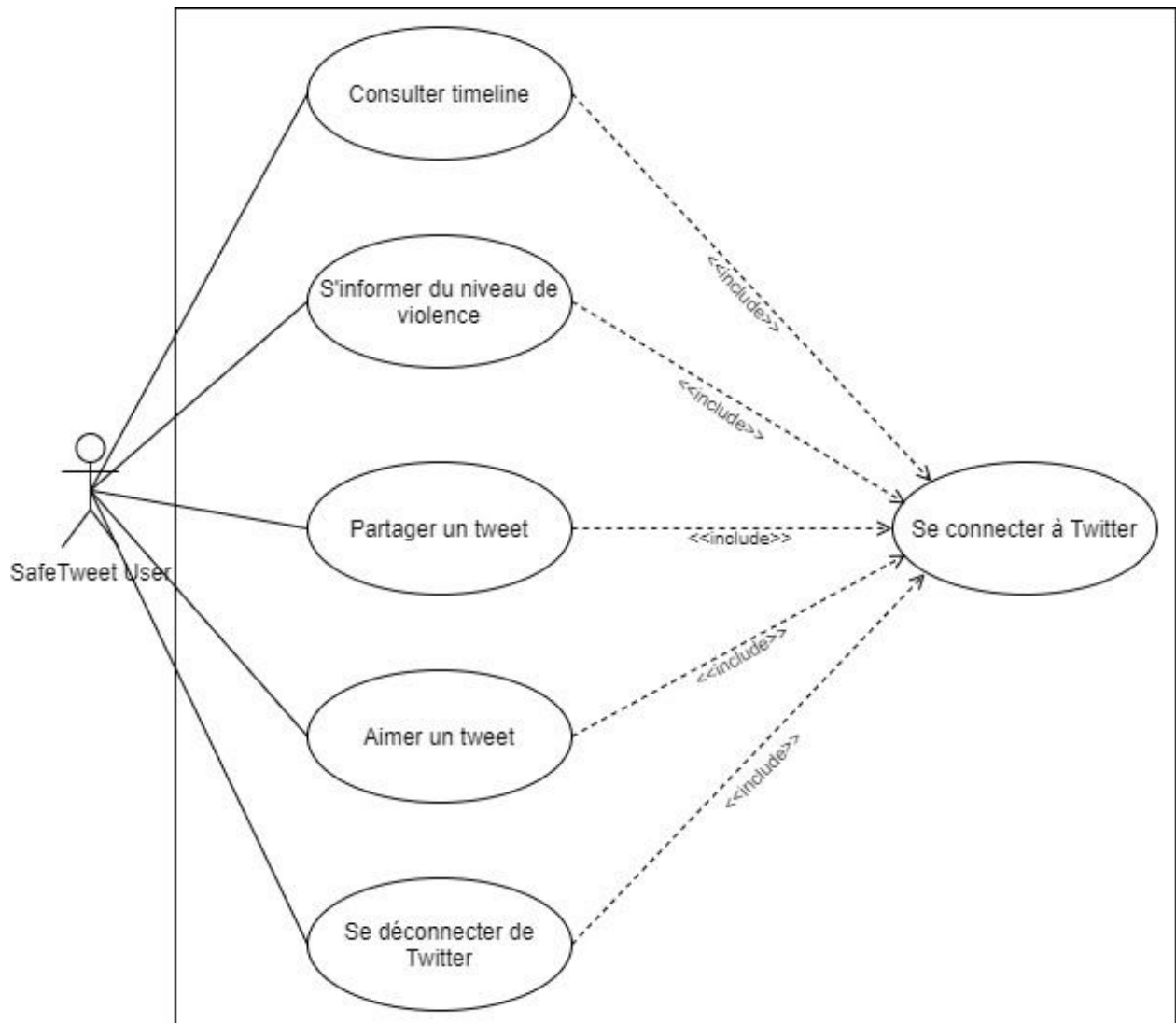
Lorsque l'utilisateur est connecté, l'écran d'accueil s'ouvre, une requête va être envoyée à l'API Twitter qui va renvoyer les tweets à l'application. Une fois que l'application a reçu les tweets, elle va envoyer une requête au modèle afin d'analyser les tweets pour afin les afficher avec leur niveau de menace.

Afin de la différencier avec l'application de Twitter, il a été défini les fonctionnalités supplémentaires suivantes :

- afficher le degré de violence d'un tweet (trois pictogrammes)
- pouvoir filtrer les tweets suivant leur degré de violence (sans ou avec messages violents)

c. Fonctionnalités de l'application

Figure 2. Diagramme de cas d'utilisation de l'application "SafeTweet"



L'utilisateur de l'application doit se connecter à Twitter afin d'utiliser les fonctionnalités de "SafeTweet", il pourra ainsi :








- consulter sa timeline obtenue grâce à l'API Twitter
- s'informer du niveau de violence du tweet grâce à l'API du modèle de violence
- partager un tweet sur les réseaux sociaux, de discussion souhaités
- aimer un tweet
- se connecter à Twitter
- se déconnecter de Twitter

Parmi ces fonctionnalités, afin que notre application fonctionne nous avons besoin de consulter la timeline, s'informer du niveau de violence d'un tweet et se connecter à Twitter, ce soit donc nos fonctionnalités principales.

Les autres sont des fonctionnalités optionnelles.

Afin de notifier l'utilisateur et lui permettre de filtrer les tweets, des informateurs ont été créés.




Tab 5. Informateurs usager v1

Pictogramme	Signification
	<i>Garantis l'utilisateur que le message ne contienne pas de harcèlement qu'il soit vrai ou faux.</i>
	<i>Le message est vrai mais le harcèlement est plutôt faible.</i>
	<i>Le message est faux mais le harcèlement est plutôt faible.</i>
	<i>Le message est vrai et le harcèlement est fort.</i>
	<i>Le message est faux et le harcèlement est fort.</i>
	<i>Permet de choisir le filtrage des messages suivant leur degré de harcèlement ou de diffamation.</i>
	<i>Exemple : Applique un filtre sur tous les messages que récupère l'utilisateur afin que seuls les messages dits « SAFE » soient affichés.</i>

Nous avons réalisé des maquettes (cf Annexe 1. Annexe 2.) afin de les présenter à Monsieur May et avoir des retours.

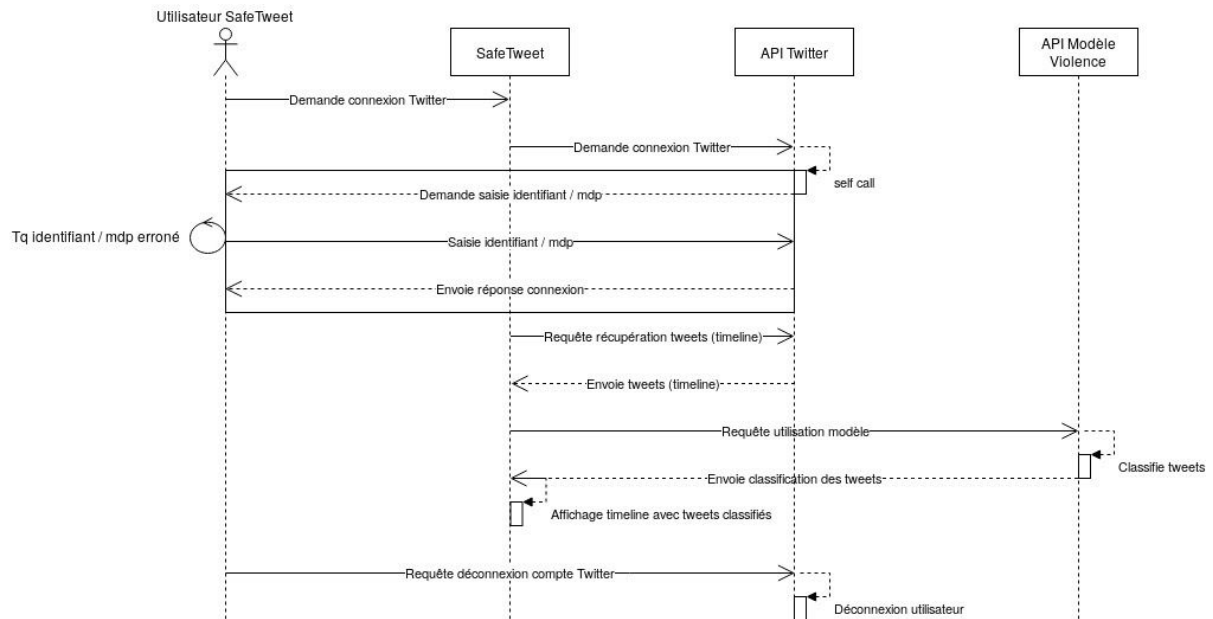
Après avoir discuté avec Monsieur May il a été décidé de placer le pictogramme de degré de harcèlement ou de diffamation du tweet avant le message, afin que l'utilisateur puisse savoir si le message est menaçant ou non avant de le lire. Il a aussi été décidé de changer la forme les pictogrammes (cf Annexe 3. Annexe 4. Annexe 5. Annexe 6.).

Tab 6. Informateurs usager v2

Pictogramme	Signification
	<i>Signifie qu'un tweet ne présente aucune offense à l'égard de l'utilisateur.</i>
	<i>Signifie qu'un tweet présente une offense à l'égard de l'utilisateur.</i>
	<i>Signifie qu'un tweet présente une forte offense à l'égard de l'utilisateur.</i>

Ainsi lors d'une première connexion l'application se déroule de cette manière :

Figure 3. Diagramme de séquence de l'application "SafeTweet" à la première connexion



L'utilisateur de l'application SafeTweet veut se connecter pour la première fois, il va alors faire une demande de connexion Twitter via SafeTweet qui va le mettre en relation avec le service de connexion de Twitter.

Il va pouvoir saisir un identifiant et un mot de passe afin de se connecter à son compte Twitter, une fois la saisie terminée il appuie sur connexion, l'API Twitter va lui demander de resaisir ses informations si elles sont fausses ou bien le connecter si elles sont vraies.

Une fois l'utilisateur connecté, SafeTweet va automatiquement envoyer une requête afin de récupérer les tweets de la timeline de l'utilisateur.

Une fois les tweets reçus par SafeTweet, l'application va envoyer une requête afin d'utiliser le modèle pour classer les tweets et enfin afficher la timeline à l'utilisateur avec les tweets classifiés.

Enfin si l'utilisateur veut se déconnecter il va le faire via l'API Twitter en envoyant une requête.

Nous pouvons donc imaginer le scénario suivant.

Romain a 15 ans, il est en seconde au lycée, partout autour de lui et sur YouTube il entend des histoires de harcèlement et de cyber-harcèlement. Un de ses camarades de classe en est d'ailleurs victime, alors il partage du temps avec afin de lui faire oublier ces messages incessants et menaçants.

Le soir en rentrant chez lui Romain reçoit une tonne de notification de Twitter, c'était une photo de son ami et lui en train de crouler sous les insultes. Il bloque tous les comptes qui l'insultent mais des nouveaux viennent à chaque fois. Il doit agir ! Il cherche sur Google une solution et tombe sur l'application "SafeTweet" qui permet de masquer les tweets menaçants, il en fait part à son ami et lui envoie le lien de téléchargement via github.

Il met alors l'application sur son téléphone grâce à Android Studio, et peut ainsi l'utiliser, une page s'affiche (*cf Screenshot 2.*), comme il ne connaît pas l'application, Romain va appuyer sur le bouton "Pourquoi SafeTweet ?" et comprendre que l'application permet de déterminer les tweets menaçants et de les cacher, exactement ce dont il a besoin.

Il va donc appuyer sur le bouton retour de son téléphone afin de cliquer sur "Connexion via Twitter", de là une page Twitter s'ouvre, il rentre son identifiant et son mot de passe Twitter afin d'accéder à son compte, la page se ferme et il est de retour sur l'application ouverte au menu "Accueil" (*cf Screenshot 3.*), et quel malheur ! Tous les tweets sont notifiés de rouge, des tweets vraiment menaçants, mais Romain se rappelle de ce qu'il a lu dans la page "Pourquoi SafeTweet ?" et appuie sur l'oiseau en haut à droite de son écran de téléphone, un menu déroulant s'ouvre, il choisit l'option "Afficher que verts" et là ... miracle ! Tous les tweets menaçants ont disparu, il voit les messages de ses amis de lycée qui le défendent ou qui parlent de tout et de rien. Il peut même se permettre de charger des nouveaux tweets sans risquer un message menaçant.

Romain étant de nouveau soulagé, utilise désormais que cette application et passe donc au travers de tous les tweets méchants envoyés par les gens de son lycée.

4. Organisation/Gestion de projet

Durant ce projet, nous avons travaillé par sprint (méthode SCRUM) d'une semaine (séance) où nous avons des objectifs définis en début de séance à remplir durant la séance.

Cette méthode nous a permis de répondre au mieux aux besoins du client et de façon extrêmement rapide, ce qui nous a permis de toujours aller dans une bonne direction concernant les choix de conception de l'application.

Lorsque l'application a commencé, l'ouverture d'un trello (lien : <https://trello.com/b/36cFQVrh/safetweet>) a été nécessaire afin de voir l'avancée de l'application en temps réel.

Notre code et toute notre documentation sont accessibles à ce lien : https://github.com/nouveliere-benjamin/Fake-News-2019_2020

a. Répartition des tâches au sein du groupe

En ce qui concerne la répartition des tâches au sein du groupe, voici un tableau récapitulatif :

Tab 7. Répartition du travail par membre du groupe

	Nouvelière. B	Durand. P	Lopez. J	Auger. N	Total
Gestion de Projet	60	25	5	10	100
Analyse des besoins	25	25	25	25	100
Analyse & conception	25	25	25	25	100
Codage	25	40	10	25	100
Réalisation BD, requêtes	20	30	0	50	100
Réalisation IHM, CSS, images	45	40	10	5	100
Tests	60	10	0	30	100
Rédaction documentation technique	45	35	0	20	100
Rédaction rapport	70	30	0	0	100
Travail global réalisé	41,667	28.889	8.333	21.111	100

b. Temps du travail au sein du groupe

En ce qui concerne le temps de travail au sein du groupe :

Tout le groupe travaillait sur les horaires de son planning (cf CR1) sauf aux dates du 13/12/2019, du 08/01/2020, du 16/01/2020 et du 10/03/2020 (précision à la fin de tous les comptes rendu).

Nous avons eu trois entretiens avec Monsieur May en dehors de nos séances de projet et un autre pendant (45 minutes chacun).

Durand Pierre a travaillé de son côté pour implémenter le maximum de pages application pour la présentation de l'oral blanc du 21/02/2020.

Nouvelière Benjamin a réalisé les comptes rendus/rapports, envoyé les mails et a fait de la gestion de projet en dehors des heures du planning.

A partir de la première semaine de confinement; il a été décidé de travailler un maximum sur le projet. Néanmoins cela était compliqué car Lopez Julio n'avait pas de PC pouvant faire tourner l'application, et Durand Pierre n'a pas avancé de son côté.

Ainsi les 23/03/2020 toute la journée, 24/03/2020 l'après-midi et 26/03/2020 toute la journée Auger Nathan et Nouvelière Benjamin ont travaillé en vocal (discord) sur les derniers détails de l'application à savoir la vue de l'accueil et de la page de connexion, la liaison avec l'API, les diapositives pour l'oral et la documentation.

Suite à cet oral, de nouvelles documentations ont vu le jour, le manuel d'utilisation (readme) par Auger Nathan afin de permettre une reprise de l'application par d'autres étudiants de M1 l'année prochaine et le manuel utilisateur par Durand Pierre afin de permettre aux utilisateurs de comprendre l'application et de pouvoir s'en servir.

Voici un tableau récapitulatif :

Tab 8. Temps de travail par membre du groupe

	Nouvelière. B	Durand P.	Lopez. J	Auger. N
Travail en séance	46h15	46h15	46h15	46h15
Travail personnel	59h	14h	0h	28h
Réunion	3h	3h	3h	3h
Total	108h15	63h15	49h15	77h15

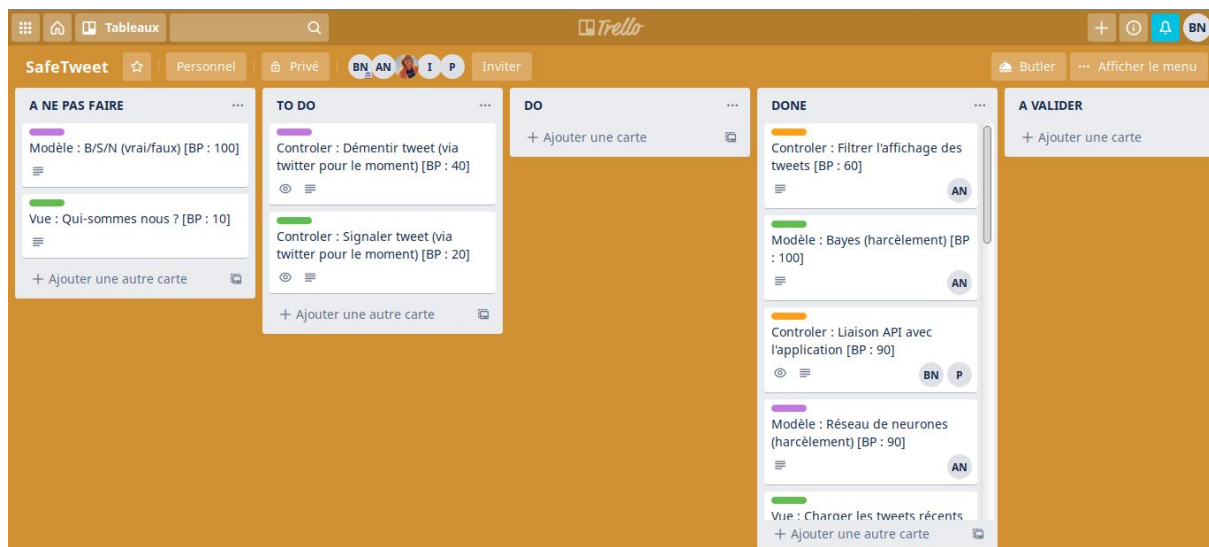
c. Échange avec le client

Lors de notre projet nous avons sollicité le client à plusieurs reprises afin de lui montrer l'avancée du projet et qu'il nous donne son ressenti, ou alors des pistes de développement de l'application ou bien du modèle permettant de définir la menace d'un tweet. Ces sollicitations ont été faites via des mails ou des réunions (cf CR Réunions).

5. Résultats

a. État du projet à la fin du module

Screenshot 1. Trello à la fin du projet



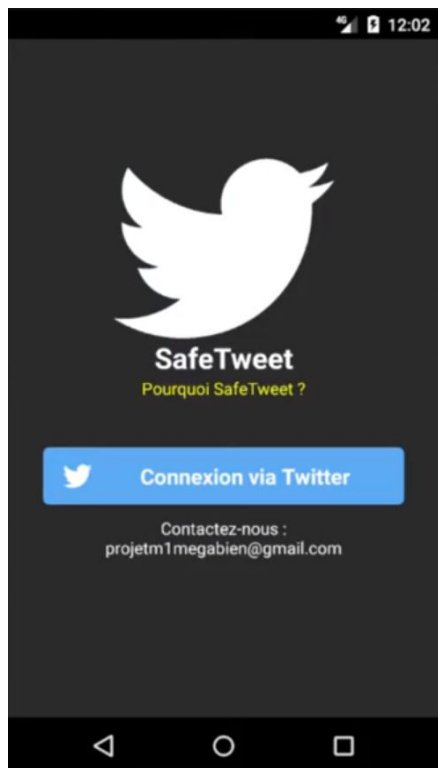
Le trello à la fin du projet nous montre bien la réflexion qu'il y a eu au niveau de l'application mobile, comme on peut le voir, la base du projet à savoir si un tweet est vrai ou faux n'a pas été implémenté et a été remplacé par le fait de savoir si un tweet est menaçant ou non, en rapport avec le thème de notre application.

Comme l'application est vouée à être reprise par d'autres personnes, nous avons décidé également de ne pas faire la partie "Qui sommes-nous ?".

En ce qui concerne le reste à faire ce sont des fonctionnalités déjà possibles grâce à Twitter (qui est ouvert quand l'on appuie sur un tweet), ce n'était donc pas nécessaire de les importer sur notre application.

Ainsi toutes nos fonctionnalités principales décrites en 3.c. sont fonctionnelles et les fonctionnalités optionnelles les plus importantes le sont également.

Screenshot 2. Menu de connexion application SafeTweet



Screenshot 3. Menu d'accueil application SafeTweet



Comme nous pouvons le voir sur ces screenshots, l'application a subi quelques changements par rapport aux maquettes (*Annexe 0.* & *Annexe 4.*) réalisées au début du projet mais conserve néanmoins en général l'application garde les mêmes structures et ses fonctionnalités principales

6. Bilan

a. Conclusion

Travailler sur ce projet de R&D a été quelque chose de très enrichissant pour nous, nous étions libres du domaine de recherche, des technologies à utiliser et du support pour distribuer notre application.

De plus, travailler avec la méthode agile SCRUM nous permettait de ne pas nous perdre dans la direction à prendre, c'était bénéfique autant à nous qu'au client. Le projet avance également beaucoup plus vite, nous nous fixons des choses à faire dans la semaine pour les réaliser, ainsi il y a un challenge à relever.

Notre application nous permet de discerner les tweets menaçants venant de Twitter grâce à l'API du modèle de réseau de neurones discernant le niveau de menace, et de cacher les tweets en direct à l'utilisateur, il peut donc charger des nouveaux tweets ou des anciens tweets sans risque. Les fonctionnalités importantes que nous avons définies sont donc fonctionnelles.

b. Pistes d'amélioration

Le cyber-harcèlement étant un fléau qui prend de plus en plus d'ampleur chaque année, une version anglaise de l'application pourrait être une très bonne idée, un nouveau modèle serait nécessaire comme les mots menaçants en anglais et en français ne sont pas les mêmes, des datasets anglaises sont facilement trouvables sur internet.

Néanmoins même le modèle français n'est pas infallible, il reste des tweets menaçants qui lui échappent ou bien des tweets non menaçants qui sont catégorisés comme étant menaçants. Une possibilité afin d'améliorer le modèle serait de permettre aux utilisateurs de démentir le jugement du modèle et lorsqu'un certain nombre de démentis pour le même tweet sont envoyés, celui-ci rejoint le modèle dans la catégorie menace (ou sans menace).

7. Bibliographie

Références :

[1] Medium. : <https://medium.com/@timoreilly/how-i-detect-fake-news-ebe455d9d4a7>

[2] Twitter Kit Android :

<https://github.com/twitter-archive/twitter-kit-android/wiki/Show-Tweets>

8. Annexes

Annexe 0. Maquette page de connexion



Ici l'utilisateur peut se connecter à son compte Twitter, appuyer sur "Mot de passe oublié ?", "Qui-sommes nous ?" ou encore "Nous contacter".

Annexe 1. Maquette v1 page d'accueil sans filtre



Ici l'utilisateur peut voir l'information concernant la menace dessus.
Annexe 2. Maquette v1 page d'accueil avec le filtre "SAFE"



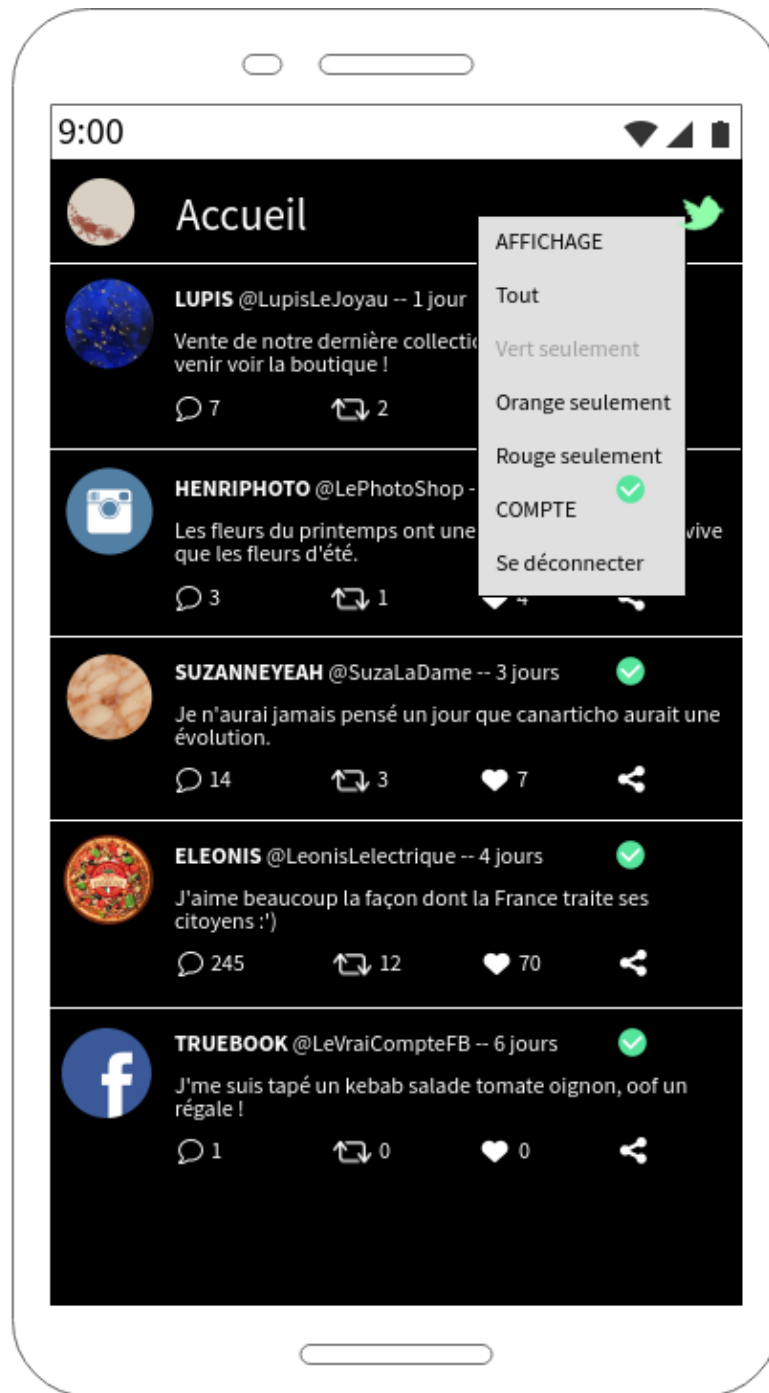
Ici l'utilisateur ne peut voir que les messages dits "SAFE" c'est à dire sans menace.

Annexe 3. Maquette v2 page d'accueil sans filtre



Ici l'utilisateur peut voir l'information concernant la menace dessus.

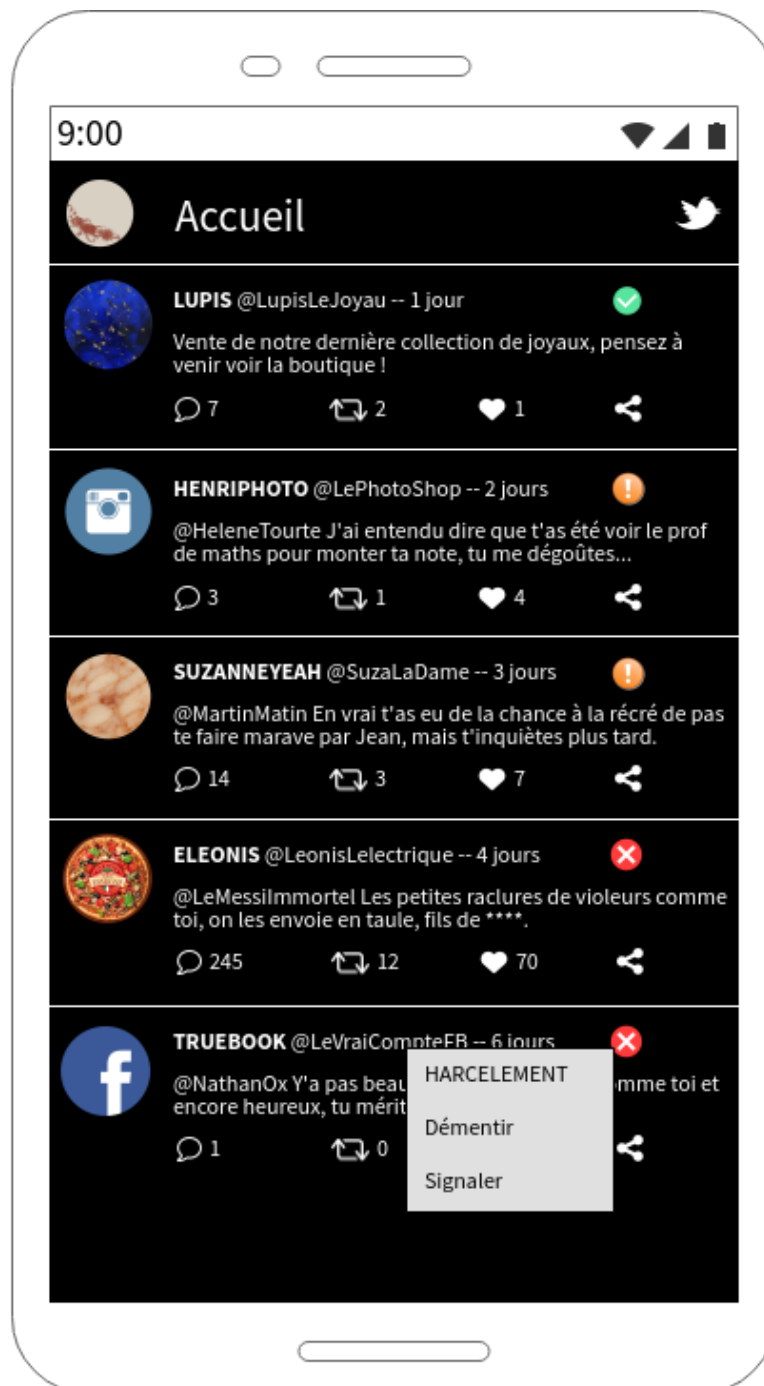
Annexe 4. Maquette v2 page d'accueil avec le filtre "SAFE" (menu déroulant)



Option permettant de placer un filtre sur les tweets visibles, selon leur niveau d'offense (pas de harcèlement vert, harcelant/diffamant orange et très harcelant/diffamant rouge).

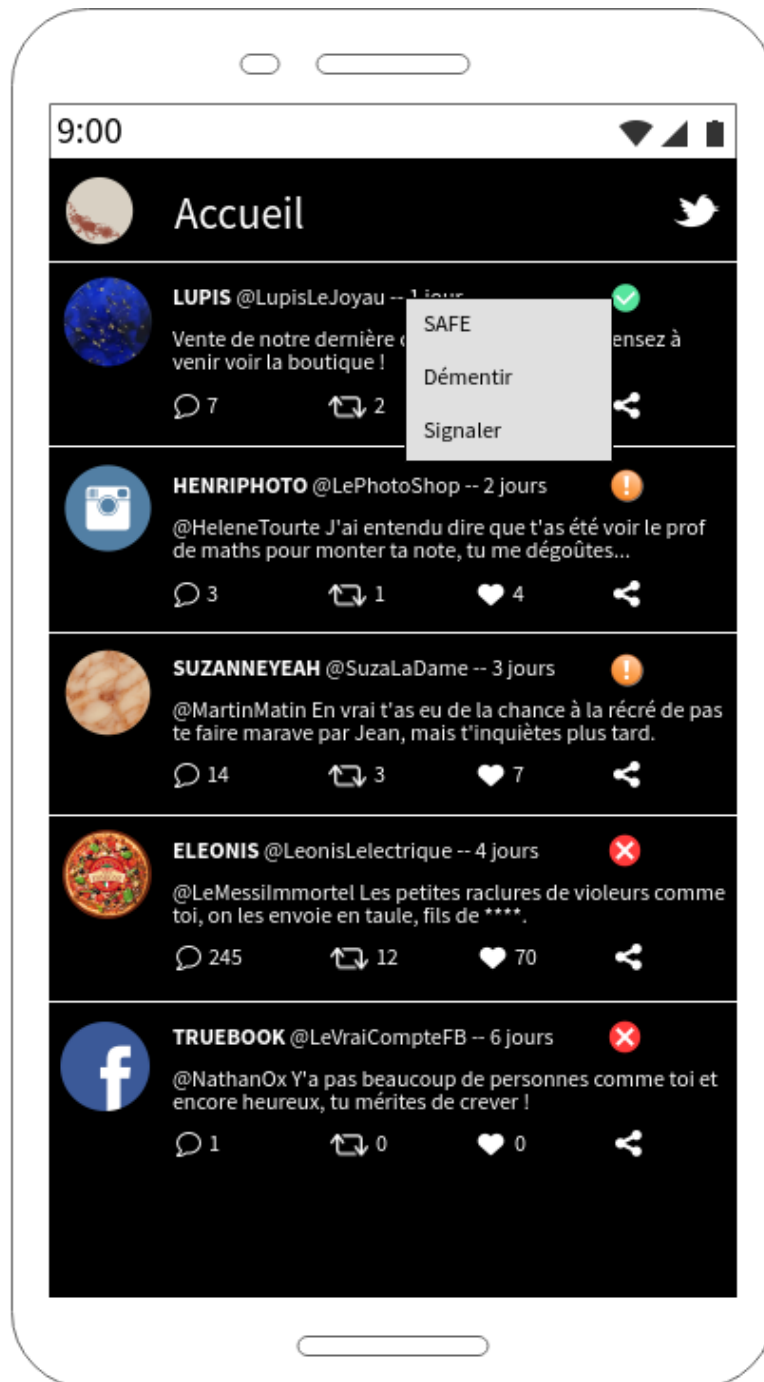
Permet également de déconnecter le compte connecté.

Annexe 5. Maquette v2 page d'accueil sans filtre (menu déroulant sur un pictogramme d'offense élevé)



Option permettant à l'utilisateur de démentir l'offense du tweet afin qu'il soit revérifié.
Il permet également de signaler la personne qui a écrit ce tweet.

Annexe 6. Maquette v2 page d'accueil sans filtre (menu déroulant sur un pictogramme d'offense nulle)



Présente les mêmes options que la maquette précédente.