

# Application Web



*CANON Natchley (chef de groupe), CLOCHARD Lucas,  
FRAYSSE Vincent, ROY Thomas, SINEL-BOUCHER Paul,  
GOURMELON Florence, BUHR Killian, DIAW Serigne,  
DIALLO Mamadou Yaya, VILLENAVE Maëva*

GROUPE 9

Projet inter-promotion

Année universitaire : 2017/2018

# SOMMAIRE

<b>1. Objectif du groupe</b>	<b>3</b>
<b>2. Organisation</b>	<b>4</b>
2.1. Planning	4
2.1.1. Planning prévisionnel	4
2.1.2. Planning effectif	5
2.2. Contrôles qualité mis en place au sein du groupe	6
2.2.1. Gestion des fichiers	6
2.2.2. Vérification du code	6
2.3. Répartition des tâches	6
<b>3. Développement de l'application Web</b>	<b>7</b>
3.1. Outils à dispositions	7
3.1.1. HTML5	7
3.1.2. CSS	7
3.1.3. BOOTSTRAP	7
3.1.4. JavaScript	8
3.1.5. JQuery	8
3.1.6. AJAX	8
3.2. Découpage des tâches	9
3.3. Structure et contenu du site Web	10
3.3.1. Header et footer	10
3.3.2. Page d'accueil	11
3.3.3. Page "Thème"	14
3.3.4. Page "Recherche"	17
Header	17
Graphiques	18
<b>4. Missions complémentaires</b>	<b>21</b>
4.1. Définition de la structure du site web en collaboration avec le groupe Qualité	21
4.2. Assurer la communication avec les groupes Analyse Statistique, BD-Requêtes et Qualité	22
<b>5. Difficultés rencontrées et solutions apportées</b>	<b>24</b>
5.1. Header	25
5.2. Problème des requêtes AJAX page index :	25
<b>Problème d'intégration</b>	<b>26</b>

## 1. Objectif du groupe

L'objectif de notre groupe est de mettre en place l'application web du projet WatchNews. Le site web est la seule interface visible par les utilisateurs. Il permet donc de faire le lien entre le travail des différents groupes du projet et l'utilisateur final. Le site web doit afficher:

- des statistiques générales sur les sept derniers jours,
- des analyses par thème sur cette même période,
- des analyses par mot clé en fonction de la période renseignée par l'utilisateur.

Notre groupe s'occupe de la structure du site, de son contenu ainsi que de la récupération des données à y afficher. Le design du site, c'est-à-dire les fichiers CSS, sont produits en partie par le groupe Qualité/Communication. Nous devons définir les CSS positionnels, permettant de garantir la bonne disposition des différents éléments sur le site web. Le site doit être web responsive, c'est-à-dire qu'il doit s'afficher correctement sur l'ensemble des supports (ordinateur, tablette et portable).

Le site web doit contenir des analyses statiques (simple renvoi de données) et dynamiques (renvoi de données en fonction d'informations fournies par l'utilisateur). Les différentes fonctionnalités du site doivent prendre en compte le travail des autres équipes du projet. Le bon déroulement de notre mission dépend donc également du bon fonctionnement de l'ensemble des groupes.

Les données à afficher sur le site sont demandées au groupe Groupe BD Requêtes sous forme de requêtes AJAX. Les données nous sont renvoyées au format JSON. Le Groupe Qualité/Communication s'occupe du design du site web. Un fichier CSS s'appliquant aux pages HTML doit donc être fourni par ce groupe. Ces deux groupes sont donc en lien direct avec notre groupe. La bonne communication avec ces deux groupes doit donc être assurée.



## 2. Organisation

### 2.1. Planning

#### 2.1.1. Planning prévisionnel

Un planning prévisionnel a été réalisé avant le commencement du projet. Ce planning a été conçu à titre indicatif, nous permettant principalement de nous guider les premiers jours. Ce planning n'est pas le planning effectif, il a été ajusté au cours du projet.

Planning prévisionnel	Première semaine					Deuxième semaine				
	08-janv	09-janv	10-janv	11-janv	12-janv	15-janv	16-janv	17-janv	18-janv	19-janv
Gestion de la configuration des outils										
<b>Première version du site web</b>										
Modification et finalisation du plan du site										
Définition des différentes fonctionnalités du site										
Prévision des tâches, page par page										
Maquette du site web										
Se mettre d'accord sur le design du site web										
Codage de la maquette du site web										
Récupération des données (sortie json)										
Codage des fonctionnalités du site web										
<b>Deuxième version du site web</b>										
Définition des nouvelles fonctionnalités à intégrer										
Définition des modifications à apporter à la structure du site web										
Récupération des données (sortie json)										
Développement de la seconde version du site web										
Oral										
<b>Tout au long du projet</b>										
Échange avec le groupe BD – Requêtes										
Échange avec le groupe Qualité										
Rédaction du rapport										

La première semaine aurait été consacrée à la conception et à l'implémentation d'une première version du site web. La seconde semaine correspondait à une consolidation du travail réalisé lors de la première semaine ainsi qu'à l'ajout de nouvelles fonctionnalités définies en fonction de nos idées ainsi que du travail réalisé par les autres groupes. Des

modifications de la structure du site web auraient été apportées en conséquence. Les échanges entre les groupes Qualité/Communication et BD-Requête aurait eu lieu tout au long des deux semaines et la rédaction du rapport aurait eu lieu au fil des deux semaines.

### 2.1.2. Planning effectif

Le planning effectif correspond au planning réel. Comme dans tout projet, des évolutions ou imprévus impactent le temps de réalisation des tâches et donc le planning. Le planning effectif est donc très différent du planning prévisionnel.

Planning effectif	Première semaine					Deuxième semaine				
	08/01	09/01	10/01	11/01	12/01	15/01	16/01	17/01	18/01	19/01
Gestion de la configuration des outils										
<b>Première version du site web</b>										
Modification et finalisation du plan du site										
Définition des différentes fonctionnalités du site										
Prévision des tâches pour la V1, page par page										
Maquette du site web										
Implémentation de la structure du site web										
Définition d'une liste de requêtes par page en fonction des use cases										
Création des jeux de données test au format JSON										
Mise en place d'un serveur test en local										
Développement de la première version du site web										
<b>Deuxième version du site web</b>										
Définition des nouvelles fonctionnalités à intégrer										
Définition des modifications à apporter à la structure du site web										
Définition des modifications à apporter aux fonctionnalités du site web										
Modification de la structure du site web										
Redéfinition de la liste de requêtes										
Modification des fonctionnalités du site web										
Création des jeux de données test au format JSON										
Développement de la seconde version du site web										
<b>Tout au long du projet</b>										
Communication avec le groupe BD – Requetes										
Communication avec le groupe Analyse Statistique										
Communication avec le groupe Qualité										
Rédaction du rapport										
Test du site web sur le serveur local										
Test du site web sur le serveur test										
Oral										

Une première version du site web a été rendue à la fin de la première semaine. Elle était constituée d'une page d'accueil fonctionnelle, c'est-à-dire affichant des données, ainsi qu'une première ébauche des pages "Thème" et "Recherche". La connexion avec le Groupe BD Requetes était bien établie mais la base de données n'était pas remplie. Les données qui ont été présentées correspondaient donc à des données fictives.

La seconde semaine correspond au développement des pages "Thème" et "Recherche" ainsi qu'à la prise en compte de nouvelles fonctionnalités à intégrer au site web en fonction du travail des différents groupes. Ces nouvelles fonctionnalités impliquent de nouvelles requêtes à demander au groupe BD Requetes ainsi que la modification de la structure du site web.

La rédaction du rapport a commencé plus tardivement et l'on a dû inclure une communication avec le groupe Analyse Statistique afin de garantir la compréhension des requêtes ainsi que de déterminer la faisabilité de celles-ci.

## **2.2. Contrôles qualité mis en place au sein du groupe**

### **2.2.1. Gestion des fichiers**

Pour assurer la traçabilité des modifications et des versions, l'utilisation de GitHub a été choisi pour assurer la gestion des fichiers. GitHub permet à notre groupe et à l'ensemble des groupes d'avoir une vision sur l'avancement des travaux. Pour le groupe 10 (Qualité et communication), il était important qu'ils aient accès au dernier fichier car toutes modifications de notre part pouvaient avoir un impact sur le bon fonctionnement de leur fichier CSS.

### **2.2.2. Vérification du code**

Les codes HTML doivent vérifier les règles W3C. Concernant les fichiers HTML ainsi que les morceaux de codes HTML se trouvant dans les codes JavaScript, une vérification a été réalisée grâce à un validateur. Des corrections ont été apportées jusqu'à la bonne validation en W3C. L'ensemble des personnes du groupe ont veillé au respect des règles de la charte de codage établie par le groupe 10.

## **2.3. Répartition des tâches**

Le site Web est composé de trois pages. Jusqu'au milieu de la deuxième semaine, l'équipe était divisée en trois groupes, un groupe s'occupant d'une page. L'équipe faisant la page d'accueil s'occupait dans un premier temps également du header et du footer. Ce groupe était constitué de Mamadou, Vincent et Maeva. Le deuxième groupe s'occupait de la page thème. Ce groupe était constitué de Lucas, Serigne et Thomas. Le troisième s'occupait de la page recherche. Il était composé de Paul et Killian.

Florence était chargée de la communication avec le groupe BD-Requêtes. Elle s'est également occupée de la connexion au serveur. Elle a aidé l'ensemble des groupes concernant la récupération des données renvoyées (JavaScript et requêtes Ajax).

Tout au long du projet, des modifications sur le header ont dû être apportées pour s'adapter à toutes les pages. Ces modifications ont été réalisées par plusieurs membres du groupe.



## 3. Développement de l'application Web

### 3.1. Outils à dispositions

#### 3.1.1. HTML5

HTML5 est la dernière révision majeure du HTML qui spécifie deux syntaxes d'un modèle abstrait défini en termes de DOM : HTML5 et XHTML5. Le langage comprend également une couche application avec de nombreuses API, ainsi qu'un algorithme afin de pouvoir traiter les documents à la syntaxe non conforme. Notre page HTML5 est composée de ces éléments:

- un en-tête de page : **HEADER**
- une barre de navigation : **NAV**
- une zone principale pour le contenu : **ARTICLE**
- différentes sections : **SECTION**
- un pied de page : **FOOTER**

#### 3.1.2. CSS

Le CSS est un langage informatique utilisé sur l'internet pour mettre en forme les fichiers HTML. Ainsi, les feuilles de style, aussi appelées les fichiers CSS, comprennent du code qui permet de gérer le design d'une page en HTML. Une grande partie css a été réalisée par le groupe qualité. Nous avons utilisé CSS seulement pour gérer le positionnement de certains éléments. Le calendrier de la page recherche a son propre fichier css.

#### 3.1.3. BOOTSTRAP

Bootstrap fournit plusieurs composants sous forme de plugins utilisant la bibliothèque jQuery. Ces composants permettent l'addition de nouvelles fonctionnalités au niveau de l'interface (exemple : carrousels) mais aussi d'améliorer le fonctionnement de composants existants (exemple : auto-complétion). Il a été au coeur de la structure du site web pour que ce dernier puisse être responsif.

### 3.1.4. JavaScript

Le JavaScript est un langage de programmation de scripts orienté objet. Il est majoritairement utilisé sur Internet, conjointement avec des pages Web HTML. Il permet d'apporter des améliorations au langage HTML. Il nous a permis d'écrire des fonctions pour traiter les données reçues grâce aux requêtes AJAX. Pouvant modifier notre code HTML directement, il nous sert à ajouter des éléments dans nos pages automatiquement ou sur demande de l'utilisateur. Tous les graphiques ont été écrits en javascript dans un script par page.

Nous avons utilisé plusieurs plug in et techniques libres de droits trouvées sur internet pour nos graphiques. Google chart propose différents modèles de graphiques et leurs scripts. Il faut pour les utiliser rentrer cette ligne de code qui récupère toutes les méthodes google et leurs scripts.

```
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
```

Nous avons aussi utilisé le plugin jqCloud pour nos différents nuage de mots:

```
<script src="bower_components/jqcloud2/dist/jqcloud.min.js"></script>
```

Et le plug in just gage pour créer nos 3 jauges sur la page d'accueil:

```
<script src="raphael.2.1.0.min.js"></script>
```

```
<script src="justgage.1.0.1.min.js"></script>
```

### 3.1.5. JQuery

JQUERY est une librairie JavaScript. C'est une surcouche JavaScript permettant de simplifier la syntaxe de ce langage. Elle nous a également permis d'appeler des fonctions facilitant la gestion du dynamisme du site web (apparition, suppression d'éléments).

### 3.1.6. AJAX

AJAX est une méthode JQUERY permettant de réaliser des requêtes asynchrones, c'est-à-dire sans rechargement de la page. Ainsi, il permet de faciliter l'échange de données entre le client et le serveur. Dans le cadre du projet, les requêtes AJAX sont utilisées afin d'envoyer et de récupérer les données de la base de données afin de créer chacun des graphiques.



Exemple de requête AJAX utilisée dans le site :

```
function request_top_10() {  
    $('#top10_sources').hide(); // On cache la div ayant pour identifiant top10_sources  
    $.ajax({ // Début de la requête AJAX  
        url: 'http://localhost:5000/test', // Appel à l'API locale renvoyant un fichier  
        JSON  
        type: 'GET', // L'utilisateur peut envoyer et recevoir des données  
        dataType: 'json', // Type de données reçues  
        success: draw_basic, // En cas de succès, lancer la fonction draw_basic  
        error: ajax_failed, // En cas d'échec, lancer la fonction ajax_failed  
    });  
}
```

L'URL peut prendre en compte un ou plusieurs paramètres renseignés par l'utilisateur. Ce paramètre peut par la suite être pris en compte dans les requêtes réalisées par le groupe BD-Requêtes. Chaque requête et donc chaque graphique du site web correspond à une URL que nous avons définie avec le groupe BD-Requêtes. Ainsi, le résultat de la requête est personnalisé en fonction des demandes de l'utilisateur.

La communication avec ce groupe était ainsi primordiale car il a fallu se mettre d'accord sur la forme des JSON renvoyés par l'appel de la fonction AJAX ainsi que sur la forme des paramètres envoyés par l'utilisateur à leur API.

### 3.2. Découpage des tâches

Une liste de tâches communes s'appliquant à toutes les pages peut être identifiée. Ces tâches sont plus spécifiques en fonction des fonctionnalités définies pour chacune des pages : prise en compte de paramètres et donc renvoi de données, différence de graphiques.

Ainsi, chacun des groupes devait :

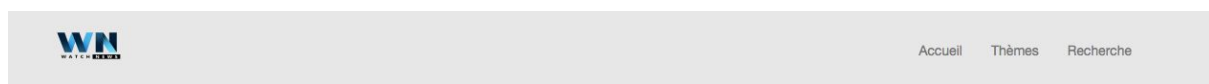
- **Coder la structure de la page** : la majorité des éléments de la page contiennent des identifiants afin de faciliter les appels JavaScript et le travail du groupe Qualité. Certains éléments contiennent des éléments Bootstrap permettant de gérer la contrainte de web-responsivité du site web.
- **Définir et générer les graphiques permettant de représenter au mieux l'information** : différentes bibliothèques JavaScript ont été utilisées afin de générer les graphiques (JustGage pour les jauges, JQCloud pour les nuages de mots, Google Charts pour les autres graphiques).

- **Produire des codes JavaScript** contenant des requêtes AJAX permettant d'afficher des graphiques en fonction ou non des données fournies par l'utilisateur et donc, de gérer le dynamisme de chacune des pages en fonction des événements.
- **Générer des données test au format JSON** : les données test ont été générées selon le format reçu lors de l'appel de l'API du groupe BD Requetes. Elles ont été insérées dans un serveur local afin de vérifier le bon fonctionnement des requêtes AJAX.
- **Garantir le fait que la page soit web responsive** : tous les éléments du site web doivent correctement s'afficher sur l'ensemble des supports.

### 3.3. Structure et contenu du site Web

#### 3.3.1. Header et footer

Nous avons voulu un seul header et footer pour l'ensemble des pages. Le header contient le logo réalisé par le groupe qualité et communication ainsi que 3 liens permettant de naviguer facilement entre les pages.

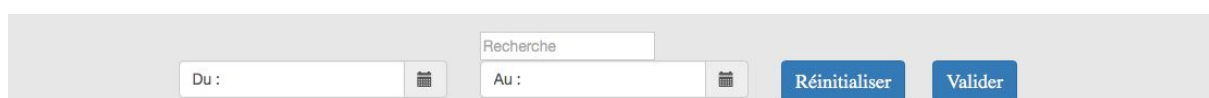


En fonction de la page sur laquelle l'utilisateur est positionné, le header gère l'affichage des barres de sélection :

- barre de sélection du thème pour la page thème permettant à l'utilisateur de choisir le thème qui l'intéresse,



- barre de recherche de mots clés pour le page Recherche permettant à l'utilisateur d'entrer le mot qui l'intéresse ainsi que la période souhaitée.



Le header est fixé sur l'ensemble des pages ce qui signifie que dans le cas où l'utilisateur navigue vers le bas de la page, le header restera affiché en haut permettant de faciliter sa navigation au sein du site. Nous avons eu des conflits de superposition entre le header et les barres de sélection présentes dans la page Thème et Recherche.

Le footer du site contient le logo de la formation, ainsi que les liens vers les différents comptes WatchNews créés sur les différents réseaux sociaux.



Le Header et le footer sont positionnés dans un fichier JavaScript indépendant permettant de les intégrer dans toutes les pages. Pour ce faire, nous avons utilisé la fonction `insertAdjacentHTML("position","html")`. Cette fonction permet notamment de définir la position exacte du code html à intégrer. L'inconvénient de cette méthode est la lisibilité. Quand on veut insérer un gros morceau de code il faut le mettre en page et l'indenter avec des ' ' et +.

### 3.3.2. Page d'accueil

La page d'accueil est la première page que l'utilisateur voit quand il arrive sur le site. Elle contient :

- une brève présentation du site réalisée en collaboration avec le groupe qualité/communication avec un lien vers la page thème et la page recherche



Le site web qui analyse quotidiennement la presse en ligne.  
Obtenez des analyses statistiques générales, par thème ou par mot-clé.

Analyse par thème

Analyse par mot clé

- des statistiques générales sur l'actualité des sept derniers jours présentée sous forme de graphique de plusieurs types,
- la liste des sources utilisées avec un lien vers le site web de la source.



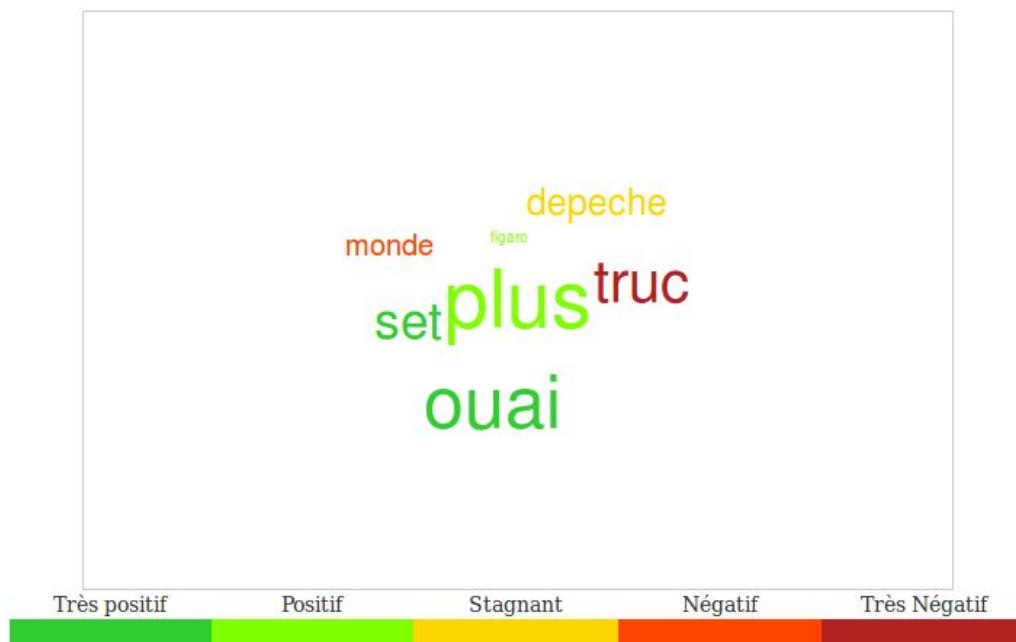
Les statistiques présentes sur la page d'accueil sont considérées comme des analyses statiques, l'utilisateur n'a aucune influence sur celles-ci. Les analyses sont effectuées sur les 7 jours glissants et concernent l'ensemble des sources et l'ensemble des thèmes. On retrouve :

- Le top 3 des sentiments les plus représentés dans les articles publiés sous forme de jauge :



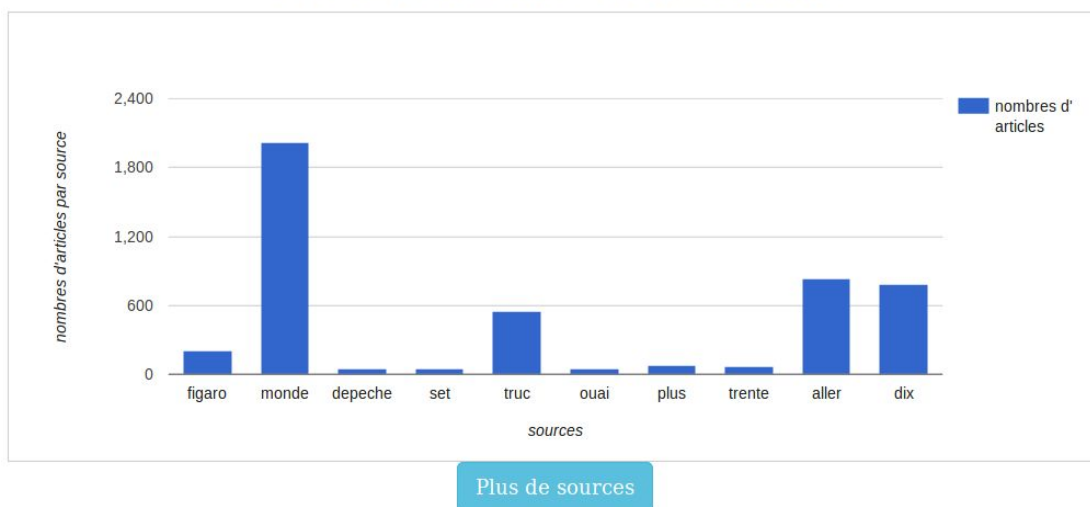
- Le thème le plus traité de la semaine avec le pourcentage d'articles géré dynamiquement par des fonctions javascripts qui sont regroupés :
- Les 10 mots les plus pertinents sous forme de nuage de mots où la taille varie en fonction de la fréquence d'apparition et la couleur en fonction de la tendance du mot (à la hausse, stable, à la baisse) :

## Top 10 des mots clés de la semaine



- Le top 10 des sources ayant publié le plus d'article sous forme de graphique avec la possibilité d'afficher l'ensemble des sources avec le nombre d'articles sous forme de tableau dans une pop-up à l'aide d'un bouton :

## TOP 10 des sources de la semaine



Pour tracer les graphiques, nous avons regroupé toutes les fonctions dans un seul fichier Javascript (graph.js). Pour la page d'accueil, l'ensemble des graphiques se charge dès son ouverture. C'est pour celle-ci que nous avons fait les premières requêtes AJAX en continu.

Toutes les fonctions pour tracer les graphiques et les requêtes sont intégrées dans un seul fichier javascript pour plus de facilité de manipulation et modification.

### 3.3.3. Page “Thème”

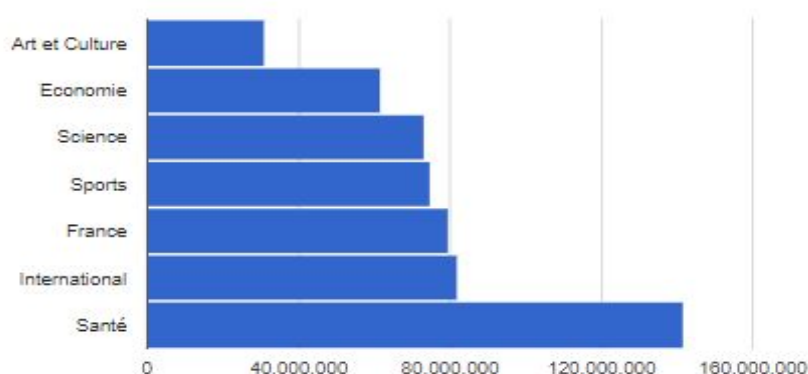
La page thème est accessible en cliquant sur le lien “Analyse par thème” présent sur la page d'accueil. Le but de cette page est d'observer les sujets les plus évoqués selon un thème précis.

L'en-tête de la page contiendra la liste des thèmes existants avec un bouton radio à activer par l'utilisateur pour chacun. Celui-ci n'aura plus qu'à sélectionner le thème désiré pour observer les résultats correspondants et pourra modifier le thème sélectionné pour observer d'autres résultats.



Lorsque l'on arrive sur cette page, il n'y a aucun thème qui est sélectionné. Dans ce cas, il est affiché un graphique correspondant aux thèmes les plus traités pour la semaine courante (autrement dit les thèmes ayant le plus d'articles lors des 7 derniers jours). Ce graphique est représenté sous la forme d'un diagramme à barre horizontale réalisé avec Google Charts.

Diagramme en barre des thèmes les plus traités de la semaine



Une fois que l'utilisateur a cliqué sur un thème, le premier graphique général disparaît pour faire place à deux autres graphiques relatifs au thème choisi avec les fonctions "hide()" et "show()" de jquery.

Le premier graphique relatif au thème indique les 10 mots les plus traités concernant lors de la semaine écoulée représenté sous la forme d'un nuage de mots. La taille des mots est discriminée selon le nombre de fois où ils sont traités. La tendance est illustrée par la coloration des mots. Il y a 5 tendances différentes : "en très forte hausse", "en hausse", "stagnante", "en baisse", "en très forte baisse" et chacune a une couleur correspondante. Le nuage de mots est réalisé avec le plug-in jQuery jQCloud. Ce plug-in nous permet de donner corps au nuage de mots et de le mettre à jour lorsque l'utilisateur change le thème sélectionné.

Sous ce graphique, il y a une légende qui indique à quel thème correspond chaque couleur car il n'est pas toujours évident de deviner quelle couleur représenterait la plus forte positivité ou négativité. Cette légende a été construite via une table HTML.

## Nuage des mots les plus traités par semaine





La deuxième visualisation de données énonce le verbe, l'entité nommée et l'adjectif qui ont été les plus associés au thème (toujours au cours de la dernière semaine), ainsi que leurs tendances respectives. Ces informations sont écrites dans un tableau Google Charts.

### Verbe, nom et adjectif les plus associés au thème et leur tendance

Mots liés au thème	Tendance
Boire	En très forte hausse
Johnny	En très forte baisse
Savoureux	Stagnante

### 3.3.4. Page “Recherche”

La page “Recherche” est accessible en cliquant sur l’onglet “Recherche” dans le header de la page d’accueil. La version “par défaut” de cette page n’affiche rien hormis le header et le footer. Néanmoins, le header contient un moteur de recherche composé de trois nouvelles zones de saisies; une zone pour le mot, une pour renseigner la date de début et pour celle de fin de période. On trouve ensuite deux boutons; l’un pour valider la saisie et l’autre pour réinitialiser nos trois champs.

#### Header

Le but du header était de fixer le menu ainsi que la barre de recherche, tant pour la page des “Thème” que pour la page “Recherche”. Pour la page de recherche, nous avons donc fixé les options grâce au `fixed_header.js`. Puis, nous avons remarqué que lorsque la page rétrécissait, la barre de recherche cachait tout l’écran. Nous avons donc opté pour la réduction de la barre à un bouton, comme le fait déjà le menu, de façon à être web responsive.

Dans cette barre, plusieurs boutons ont été positionnés. La zone de saisie, le choix du thème, le choix de la source, le choix de la période, les dates de début et de fin d’analyse, ainsi que les boutons “Valider” et “Réinitialiser”. Les boutons pour le choix de thème et de source ont été écartés.

Toute la page devait être responsive, c’est pourquoi, pour l’affichage des graphiques, des titres, et du lien wikipédia, nous avons utilisé des classes bootstrap. Ainsi, les divisions concernées sont correctement affichées, tout type d’écran confondu.

Après des comparatifs entre certaines librairies Bootstrap et des forums internet, nous avons décidé de prendre un calendrier en libre accès avec un `.css` et un `.js` associés. Notre choix s’est porté sur ce programme car il est très complet, plus facilement implémentable et permet de naviguer plus rapidement entre les mois et années que les calendrier Bootstrap que nous avons regardé.

Avant de pouvoir afficher les différentes données, il faut vérifier que les valeurs entrées par l’utilisateur soient plausibles. Ainsi, nous avons créé une fonction de contrôle s’exécutant lorsque l’utilisateur appuie sur le bouton “valider”. La fonction vérifie que l’on rentre un mot et les deux dates. Il vérifie aussi que les dates soient dans l’ordre et qu’elles soient possibles (exemple d’erreur: 40/13/2000) car, pour des problèmes de codage, nous n’avons pas pu mettre les dates en format date. (cf. partie 5 : “difficultés et problèmes rencontrés”). Une autre contrainte concerne le mot. En effet, si les requêtes retournées par le serveur sont

vides, cela veut dire que le mot n'existe pas dans la base de données et donc cela bloque l'affichage des différents éléments de résultats.

## Graphiques

La partie complexe des graphiques fut le fait d'intégrer les différents paramètres optionnels; sources, thèmes et période. Néanmoins cette idée a été abandonnée au milieu de la deuxième semaine voyant que nous ne réussissions pas à obtenir des résultats au niveau des attentes. De plus, cela rajoutait énormément de travail à d'autres groupes surtout sur le paramètre de la période analysée. Ainsi, tous nos graphiques représentant les sources et les thèmes dans leur intégralité.

Une fois les vérifications effectuées, le bouton "valider" utilise des fonctions Ajax pour récupérer le résultat de la requête au format JSON.

Pour écrire le lien de la requête Ajax permettant d'accéder aux requêtes, il faut récupérer les données entrées en recherche, puis les ajouter à l'adresse du serveur dans un ordre précis avec des "/" entre chaque valeur. Le choix de l'ordre fut pris par concertation avec le groupe requête. Voici un exemple de lien de requête :

```
url:'http://130.120.8.250:5000/article_per_label' + '/' + value_search_bar + '/' + start_date_ajax + '/' + end_date_ajax,
```

Un problème s'est posé avec le format des dates. En effet, les dates sont rentrées au format "jj/mm/aaaa". Ici, l'ajax, lira trois valeurs au lieu d'une, il a donc fallu modifier le format des dates au suivant "jj-mm-aaaa" (cf. partie 5 : "difficultés et problèmes rencontrés").

Les graphiques affichés sur la page "Recherche" proviennent tous du site consacré aux graphiques de la librairie google chart. Nous utilisons trois barplot et trois graphiques en courbes. Un graphique général représentant l'évolution de parution du mot dans tous les articles traités. Deux graphiques en courbes représentant l'évolution des parutions du mot par thème et par source. Et enfin, deux barplot représentant le nombre d'apparition du mot par source et par thème.

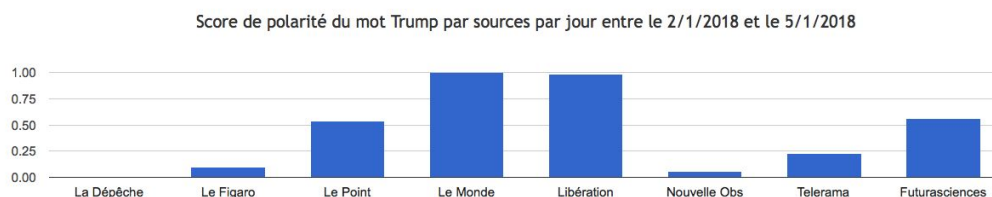
Chaque graphique est traité à quelques exceptions près de la même façon. D'abord, on charge les packages relatifs au graphique puis on transforme les fichiers JSON en tableau et enfin on insère les données dans le graphique.

Les sources et plus particulièrement les JSON traitant des sources doivent néanmoins être traité différemment des autres fichiers car leur nombre peut changer. En effet, ce nombre variable pose un certain problème car il rend variable le nombre de colonnes et cela est embêtant en JavaScript. La méthode que nous appliquons pour transformer un fichier JSON

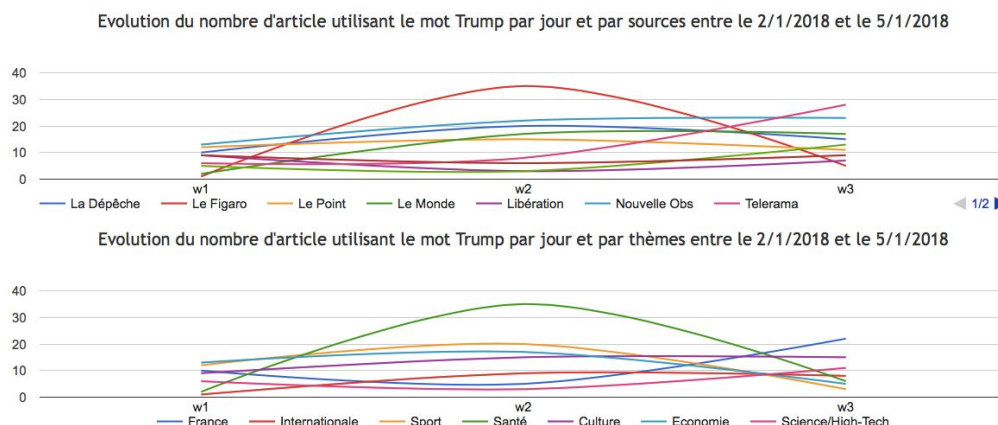
en tableau insère ligne par ligne les données dans ce dernier. Ainsi, rendre le nombre de colonnes variable rend le travail plus compliqué. Pour palier à ce problème on utilise deux boucles imbriquées qui permettent d'insérer les données par date en fonction du nombre de sources.

```
data.addColumn('string', 'theme');
for (var g = 1; g <= Object.keys(json_graph2).length; g++) {
  if (json_graph2[g].date==week){
    data.addColumn('number', json_graph2[g].label); //add every distinct sources present in the Json into column
    col=col+1;
  }
}
for (var i = 1; i <= Object.keys(json_graph2).length; i+=col) {
  var tab = [json_graph2[i].date];
  for (var j = 0; j < col; j++) { //create a table proportional to the number of sources selected
    tab.splice(j+1, 0, json_graph2[j+i].number_article);
  }
  data.addRow(tab); //add the table to generate the lines
}
```

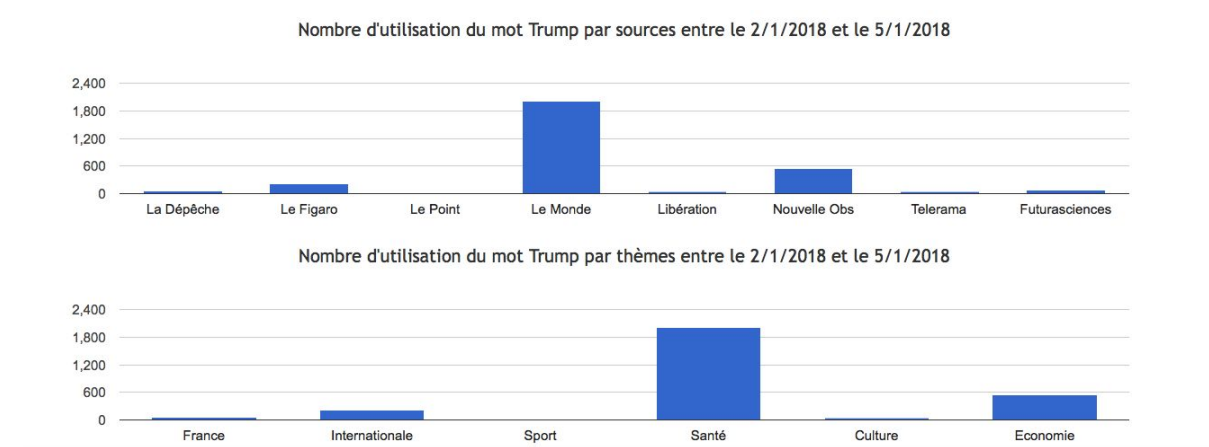
Le premier graphique représenté est celui montrant la polarité du mot en fonction de la source l'utilisant. En effet, on sait que Le Figaro est un journal plutôt de mouvance de droite alors que Libération est plus à gauche. On veut donc montrer qu'ils ne parlent pas de la même manière d'un sujet.



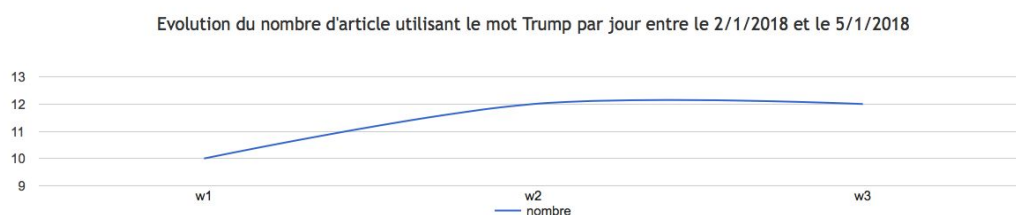
Les deux graphiques suivants sont des séries chronologiques montrant le nombre d'articles dans lequel le mot apparaît le mot choisi en fonction du temps, puis par source pour le premier graphique, et par thème pour le second.



Les deux graphiques suivants sont des diagrammes en barres montrant le nombre d'articles où le mot choisi apparaît en fonction de la source pour le premier graphique, puis en fonction du thème.



Le dernier graphique est une série chronologique montrant le nombre d'articles dans lequel le mot apparaît le mot choisi en fonction du temps.



Concernant les titres, ces derniers sont paramétrés et évolués grâce aux trois zones de saisie du header.

Une autre tâche demandée était l'affichage du lien wikipedia si jamais le mot est une entité nommée possédant un lien wikipedia. Ainsi, tout comme pour l'affichage des graphiques de la page, le lien wikipedia s'applique uniquement si la requête reçue par le serveur n'est pas nulle.

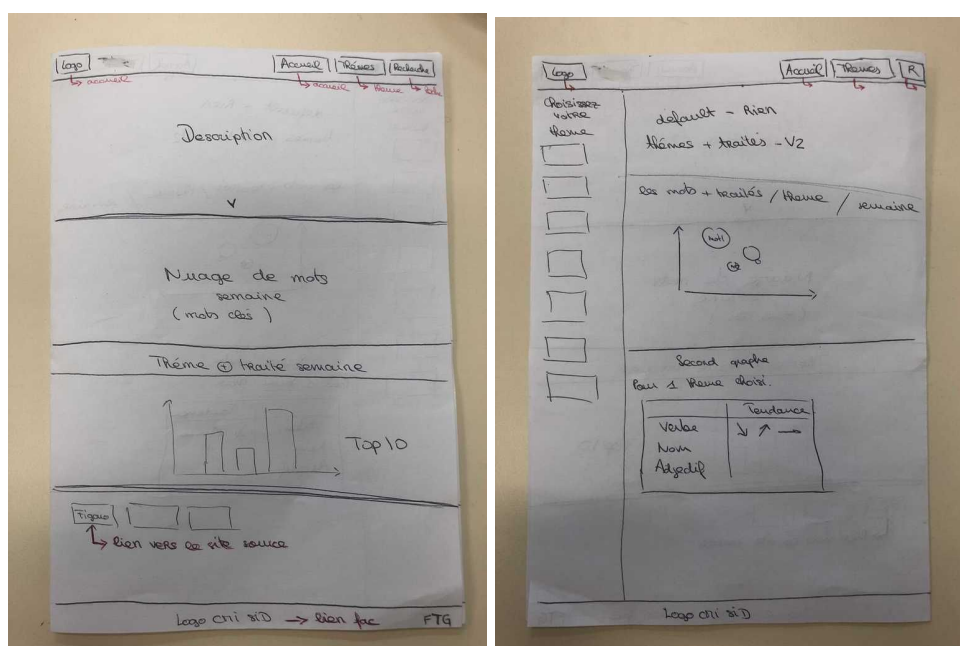
 [Donald\\_Trump](#)

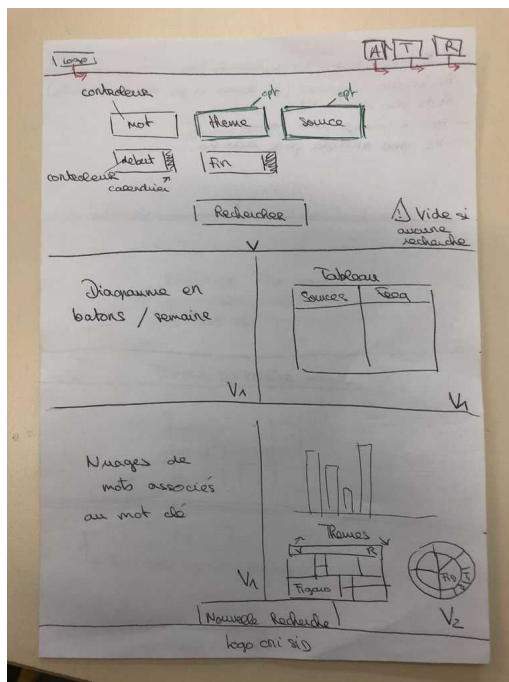
## 4. Missions complémentaires

### 4.1. Définition de la structure du site web en collaboration avec le groupe Qualité

L'ensemble des groupes Qualité/Communication et Web ont participé à la définition de la structure globale du site web a été défini en collaboration avec le groupe.

Les trois photos correspondent à la maquette du site web. Le design a évolué au fil des jours.





## 4.2. Assurer la communication avec les groupes Analyse Statistique, BD-Requêtes et Qualité

Le site, ne communique pas directement avec le groupe d'analyse statistiques. En effet, nous envoyons des paramètres tels qu'un thème, un mot et/ou une période de temps au groupe de BD requête, et celui ci envoie les paramètres nécessaires au groupe de statistiques afin qu'ils effectuent leurs analyses. C'est ensuite le groupe de BD requête qui nous renvoie les données sous format json afin que nous puissions les visualiser sous forme de graphique et donc les valoriser. Nous avons réussi à récupérer les données que nous envoient le groupe BD grâce aux routes Flask, cependant la base de données n'est pas assez riche pour que nous puissions afficher les graphiques correctement. C'est pourquoi certaines requêtes nous renvoient des fichiers json stockés en dur du côté du groupe BD.

Nous avons travaillé sur un serveur local identique au serveur de test. Le serveur de test doit communiquer grâce à la librairie Flask de python avec le site, ainsi nous avons téléchargé Flask sur nos ordinateurs et nous avons créé un fichier python afin d'envoyer des json de "tests" sur des urls précises. Ceci avait pour but de nous permettre de faire nos requêtes ajax et de n'avoir plus qu'à modifier l'url de la route lors de la mise en production (passer notamment de localhost à 130.120.8.250). Nous pouvions ainsi tester l'envoi des paramètres.

La communication la plus complexe a été avec le groupe de qualité puisque nous avons deux styles d'interactions avec eux. La première concernant la partie visuelle du site (le style css)



et la deuxième concernant la qualité du code implémenté. Pour cette dernière, nous devions respecter des normes imposées par le groupe qualité. Nous avons eu quelques retours suite à l'envoi de nos codes mais dans l'ensemble la qualité semblait satisfaire aux exigences. Pour la partie visuelle du site, la communication a été plus complexe. La partie css et la partie html dans la création d'un site, sont deux parties très liées. Surtout lors de l'utilisation d'un framework tel que Bootstrap. En effet, nous avons utilisé certaines classes pour gérer le responsive du site, et le groupe qualité en utilisaient d'autres pour le visuel. Il y a donc eu des conflits entre les classes qui ont ralenti la phase d'intégration. De plus, la groupe qualité nous a parfois demandé de modifier notre html afin d'éviter ce genre de conflits, ceci nous a donc obligé à recoder certaines parties ce qui en soit a été une perte de temps. Mais dans l'ensemble, le résultat est plutôt ressemblant à l'idée de départ.

## 5. Difficultés rencontrées et solutions apportées

Problème d’affichage dynamique du nuage de mot sur la page Thème : nous avons initialement importé la version 1.0.4 du plug-in jqCloud qui permet d’afficher le nuage de mots. Le problème était la mise à jour dynamique du graphique lorsque l’utilisateur souhaite changer le thème sélectionné. Ce plug-in comporte une fonction de mise à jour, mais celle-ci n’était pas encore implémentée dans cette version et il a fallu chercher une version plus récente (version 2) qui fonctionnait.



Les différents mots du nuage ne s’affichaient pas correctement. Au lieu d’afficher le mot avec le poids le plus fort au milieu et les autres autour, les mots été affichés tous les uns au dessus des autres (voir image ci-dessus) ou bien d’une autre manière singulière, même après la mise à jour du changement de thème. Il s’est avéré que c’était la fonction “hide” de jQuery qui était à l’origine du problème, sans que l’on sache vraiment pourquoi. Pour résoudre ce problème, nous avons remplacé les fonctions jQuery “hide” et “show”, qui concernaient la partie du nuage de mots, par une classe “hide” appliquée à cette partie que l’on supprime dans la partie javascript liée. Une fois ce changement de méthode appliquée, les mots s’affichent normalement.

### Header

Il ne fallait pas que le header prenne toute la place lorsque que la page rétrécissait, nous avons donc décidé de réduire le header à un bouton (sous forme de petite loupe) qui

permet d'afficher ou de cacher le menu, plus un autre pour afficher les paramètres de choix du thème dans la page thème ou les paramètres de la barre de recherche dans la page recherche. Nous avons utilisé la class="navbar-toggle collapsed" de bootstrap pour ce faire.

## 5.2. Problème des requêtes AJAX page index :

Nous avons eu des difficultés pour trouver une forme correcte de fichier JSON. Nous avons donc eu à adapter les fonctions javascript servant à tracer les graphiques en fonction du modèle de JSON que nous avons choisi. Notamment au niveau des boucles et du traitement des données. Par exemple nous avons eu un problème de compatibilité entre le type de tableau retourné par le traitement du JSON et le tableau de string demandé par la fonction :

```
$(".word_cloud_row_graph").jQCloud(some_words);
```

On a donc utilisé une variable de type array et une boucle pour le remplir.

Toutes les fonctions ont des paramètres en entrée ce qui nous a obligé à revoir les méthodes google chart pour bien en comprendre le fonctionnement :

```
google.charts.load('visualization', '1', {packages: ['corechart', 'bar']});  
google.charts.setOnLoadCallback(function(){
```

Le type des requêtes AJAX 'GET' ou 'POST' en test sur le serveur local n'était pas évident pour récupérer nos données fictives.

### Date format américain

Un des gros problème de la page recherche à été le format des dates. En effet, nous devons jongler entre 3 formats de date:

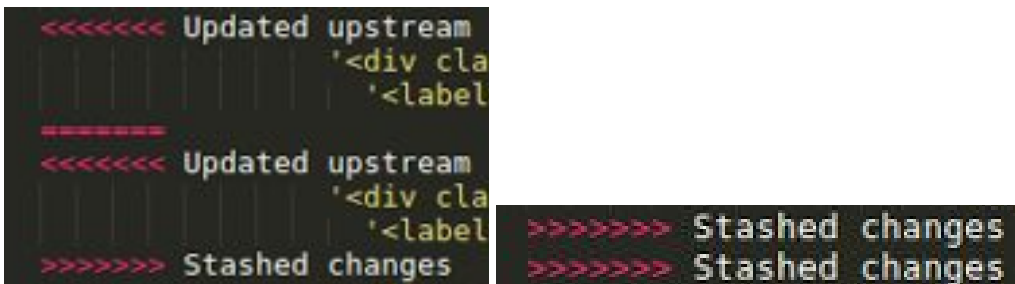
- dd/mm/yyyy (format français)
- mm/dd/yyyy (format américain)
- yyyy-mm-dd (format pour le groupe requête)

Le format date de javascript est codé en américain, ainsi, mettre une date avec un jour supérieur à 12 affichera une erreur car il le traduira en tant que mois. La fonction formattedDate (cf. image ci-dessous) pallie à ce problème. Elle permet de ressortir une date selon le paramètre 'type' de la fonction. Ainsi, en rentrant n'importe quelle date en format français la fonction ressort le format de date souhaité. Elle traite aussi les erreurs de saisie tel que des dates impossibles (ex : '40/13/2000') ou bien du texte (ex : 'abcd') et ressort un 'false' si c'est le cas.

```
function formattedDate(date,type) {
  var day="";var month="";var year="";
  if (date.substring(2,3)=='/' && date.substring(5,6)=='/'){
    day=date.substring(0,2); month=date.substring(3,5); year=date.substring(6,10)}
  if (date.substring(2,3)=='/' && date.substring(4,5)=='/'){
    day=date.substring(0,2); month=date.substring(3,4); year=date.substring(5,9)}
  if (date.substring(1,2)=='/' && date.substring(3,4)=='/'){
    day=date.substring(0,1); month=date.substring(2,3); year=date.substring(4,8)}
  if (date.substring(1,2)=='/' && date.substring(4,5)=='/'){
    day=date.substring(0,1); month=date.substring(2,4); year=date.substring(5,9)}
  if (day!="" && month!="" && year!=""){
    if (month.length < 2) month = '0' + month;
    if (day.length < 2) day = '0' + day;
    if (day<=31 && month<=12 && type=='mmdyyy'){
      return `${month}/${day}/${year}`;
    }else if (day<=31 && month<=12 && type=='yyyymmdd'){
      return `${year}-${month}-${day}`;
    }else{
      return false;
    }
  }else{
    return false
  }
}
```

### Ajax graphique recherche

Un problème est revenu à plusieurs reprises sur la fin du projet. Ce problème été lié au logiciel GitKraken qui nous permettait de rassembler nos données. Sur certaines mise à jour de fichier les lignes des images suivantes apparaissaient et rendaient les fichiers illisibles. Nous supposons que le problème vient de la suppression de bout de code d'une version à une autre.



## Problème d'intégration

### Modification de l'architecture du serveur :

Le fichier index.html, correspondant à la page d'accueil, été placé dans la racine du serveur alors que les pages "Recherche" et "Thème" était dans un dossier à part. Cela posait problème lors de la navigation entre les différentes pages. Afin de faciliter la navigation entre les pages dans le site web et clarifier les liens HTTP, tout les fichiers HTML ont été placé dans le dossier racine (avec l'accord du groupe Supervision).

## **Abandon de plusieurs fonctionnalités du site web : notamment au niveau de la page “Recherche”**

### Page d’accueil

#### Affichage des sources

Nous avons pensé afficher les différentes sources dans une section de la page avec leurs logos et un lien vers le site de la source. Mais à cause des modifications de la base de données et d’une mauvaise communication avec le groupe qualité nous avons pris du temps à savoir qui devrait le faire et comment, en dur ou avec une requête AJAX.

### Page “Thème”

- Sélectionner plusieurs thèmes

L’utilisateur aurait normalement dû avoir le choix de sélectionner plusieurs thèmes et de pouvoir comparer les deux thèmes entre eux. Cette idée a été abandonnée car, vu la durée du projet, elle aurait été trop compliquée à implémenter.

- Sélectionner une ou toutes les sources et un ou tout les thèmes, ainsi que la période : pour les mêmes raisons que l’ “Abandon de la sélection multi-thème”, la sélection de plusieurs thèmes et plusieurs sources a été abandonnées.

### Page “Recherche”

- Malgré nos efforts, plusieurs projets d’amélioration de la pages n’ont pas aboutis par fautes de temps ou parce qu’ils n’étaient pas prioritaires. On trouve dans ces axes d’améliorations certains graphiques, des listes déroulantes, certaines modifications css/html et l’autocomplétion de la zone de saisie du mot.
- Nous avons du supprimer 2 types de graphiques. Tout d’abord des nuages de mots ont été supprimé à la demande de notre chef de groupe. Ensuite, nous avons supprimer un pie chart représentant les sentiments associés au mot car les sentiments sont calculés par article et non pas par mot. Ainsi, représenter ces données n’avait pas de réel sens même si l’information n’était pas totalement fausse. Les différentes listes (thème, sources et périodes) ont été supprimées car elles causaient des problèmes au groupe requête car l’utilisateur pouvait aussi bien choisir, un thèmes que tous, une source que toutes. La période a elle été enlevé car les analyses étaient faites par jour et non par semaine ni par mois. Certaines modifications faites par le groupe qualité en HTML/css n’ont pas pu être ajoutées par manque de temps et par peur de conflit entre les ajouts HTML de dernières minutes.

- Nous avons réussi à faire une autocomplétion en locale sur la barre de recherche, mais avec un petit jeu de données (500 mots). Nous n'avons pas trouvé de solution pour éviter de récupérer tous les mots potentiellement requêtable sur le site. En effet, cela ralentirait le chargement de la page du fait qu'il a 80 000 mots potentiellement requêtable dans la base de données.

Manque de données :

En raison de l'incertitude de données dans la base de données, plusieurs graphiques n'ont pas été réalisés :

- Nuage de mots synonymes au mot entrée par l'utilisateur dans la page "Recherche"
- Nuage de mot des mots associés aux mots clés de l'utilisateur dans la page "Recherche" lorsque l'utilisateur réalisait une recherche sur une entité nommée.

### **Intégration de notre travail dans le serveur test**

Pour parer aux problèmes d'insertion de données dans la base, des fichiers JSON ont dû être créés afin de démontrer le bon fonctionnement de la connexion entre notre groupe et le groupe BD-Requêtes.

## 6. Conclusion

Durant ce projet, nous avons cherché avant tout à remplir les cas d'utilisation définies par le groupe Supervision et de mettre en valeur le travail des autres groupes tout en respectant les différentes contraintes. En effet, le site devait être web responsive et dynamique.

Nous avons eu des problèmes de communication avec les autres groupes, notamment Qualité/Communication mais aussi BD-Index pour savoir quels types de données nous allions recevoir.

Le projet a été une expérience enrichissante pour chaque membre du groupe, que ce soit en terme de gestion de projet mais aussi en terme d'apprentissage de langages de programmation web.