Rapport de stage

BTS SIO SLAM 1^{ère} année Lycée Turgot



Liberté Égalité Fraternité



Remerciements

Je tiens à remercier les personnes qui ont participé au succès de mon stage.

Dans un premier temps, je tiens à adresser mes remerciements à mon maître de stage, M. Laurent Hadjadj, architecte technique du service informatique au sein de l'établissement FranceAgriMer, pour son accueil et son aide lorsqu'il s'agissait de me guider dans les tâches concernant les missions qui m'étaient confiées.

Table des matières

Remerciements	2
Présentation du stage	4
Nature du stage	4
Présentation de l'entreprise	4
Contexte du programme Expadon 2	6
Exigences fonctionnelles	9
Architecture technique	10
Réalisation	11
Première étape	11
Asynchronisme	12
Le logo de l'application	15
L'accessibilité	17
Compréhensibilité de l'application	17
La présentation	18
Retour sur expérience	20
Compétences acquises/développées	20
Pistes d'amélioration de l'application	20
Conclusion	21

Présentation du stage

Nature du stage

Le stage de première année en BTS SIO dure 6 semaines. Mon stage s'est déroulé au sein du service informatique de l'organisme FranceAgriMer, plus précisément au sein du module « Agrément » du projet Expadon 2. Ce module permet aux entreprises de demander l'agrément de leurs produits pour l'exportation depuis la France ou l'importation vers la France. J'étais alors encadré par mon tuteur, M. Laurent Hadjadj, architecte technique du service informatique.

Présentation de l'entreprise

L'établissement FranceAgriMer est un organisme de l'état reconnu par les collectivités territoriales et les professionnels des secteurs agricoles, agroalimentaires et de la pêche, tant pour la gestion des aides européennes que nationales.

C'est un établissement public administratif créé en 2009 et placé sous la tutelle du ministère de l'Agriculture et de l'alimentation.

Le siège de l'organisme est situé au 12 rue Henri Rol-Tanguy à Montreuil.



Contexte du programme Expadon 2

- ► Le programme Expadon 2 vise à offrir aux exportateurs français des services plus efficaces de gestion des procédures sanitaires et phytosanitaires (SPS) nécessaires aux exportations de produits agricoles et agroalimentaires.
- ▶ Les accords internationaux prévoient que chaque pays importateur (hors UE) fixe, après négociation avec le pays exportateur, les conditions sanitaires, notamment la nécessité (ou non) d'un agrément de l'entreprise de fabrication et le modèle du certificat sanitaire et phytosanitaire (SPS) à présenter de manière obligatoire en accompagnement de chaque catégorie de produit.
- ▶ Le programme « Expadon 2 » a pour objectif de mettre en œuvre une plateforme globale dématérialisée de bout en bout permettant une gestion de l'ensemble des obligations sanitaires et phytosanitaires nécessaires à l'exportation des produits agricoles et agroalimentaires françaises : conception de modèles de formulaires de certificats, formulaire de saisie en ligne, diffusion d'informations, plateforme d'échange numérique (EDI), signature électronique, plateforme gouvernementale dématérialisée, transmission de données vers les pays tiers. La plateforme est destinée à l'ensemble des opérateurs exportateurs en France.
- ► La volumétrie annuelle attendue est d'environ 600 à 800 nouveaux agréments délivrés par an et d'environ 400.000 certificats délivrés par an (sanitaire + phytosanitaire).

<u>Quelques chiffres sur le marché des produits agricoles et alimentaires en France en 2020</u>

- De juillet 2019 à juin 2020, les exportations françaises de produits agricoles et alimentaires ont baissé légèrement (-0,5%) pour s'établir à 62,8 Md€.
- La valeur des exportations vers les pays tiers est de 24,1 Md€, en hausse de 1,2 %.

► La France s'est maintenue au 6ème rang des exportateurs au niveau mondial, mais avec une part de marché de 4,6 %, en baisse de 0,2 point.

Les chiffres du programme Expadon 2

- ► 400 000 certificats délivrés par an
- ► 600 à 800 instructions de dossiers d'agréments
- ▶ 1380 modèles de certificats sanitaires
- ► 16 000 utilisateurs prévus

Bénéficiaires du programme

- ► Les utilisateurs finaux
- ► Les opérateurs demandeurs
- Les responsables de filières
- ► Les agents des services déconcentrés DD(ec)PP, DRAAF/DAAF, les services économiques des ambassades de France à l'étranger
- ► Les services centraux de la DGAL
- L'UAEXP au sein de la MAEI à de FranceAgriMer

L'équipe programme interne

- ► DGAL (3 ETP)
- ► FranceAgriMer (3 ETP + 1 apprenti)

Principales filières agricoles

- Les principales filières sanitaires regroupent les familles :
 - ► Animaux vivants,
 - ▶ Denrées animales d'origine animale (DAOA) : Lait et produits laitiers/ Viandes et produits carnés/ Autres denrées alimentaires,
 - Génétique animale,
 - Produits animaux et alimentation animale.
- ► La téléprocédure Certificat est ouverte aux « Lait et produits laitiers », puis progressivement aux « Viandes et produits carnés » (T3 2021) et à l'ensemble des filières sanitaires.

► Initialement prévues dans Expadon 2, les demandes de certificats de la filière phytosanitaire sont gérées dans TRACES, idem pour tous les sujets liés au Brexit.

Présentation du sujet de stage

Dans le contexte du programme Expadon2, est apparu le besoin d'une analyse des logs de navigation sur l'application Agrément. Cette analyse se voulait donc automatisée et compréhensible pour le service informatique. L'enjeu de cette analyse est de mesurer la « santé » de l'application, identifier les évolutions possibles et prévenir les incidents applicatifs.

Cette demande fait notamment suite au retour d'utilisateurs étrangers tels que les ambassades françaises à l'étranger, qui mentionnent des temps de réponses insatisfaisants lors de leur navigation sur l'application.

La mission qui m'a été attribuée, dans le cadre de ce stage, est donc la réalisation d'un tableau de bord automatisé et dynamique permettant l'analyse des logs du module « agrément » de l'application Expadon2.

Par la suite, ce tableau de bord automatisé pourra être utilisé pour analyser les logs des autres modules de l'application Expadon2 (infocom et certificat).

Exigences fonctionnelles

À la suite de divers échanges, j'ai pu définir un certain nombre d'exigences fonctionnelles pour répondre au besoin du service informatique, par le biais d'un tableau de bord.

Ces exigences fonctionnelles sont donc des indicateurs dont la liste se trouve ci-dessous :

EXF1	Le nombre de requêtes qui correspond à des actions des
	utilisateurs sur l'application.
EXF2	Le statut des requêtes (200 (requête nominale), 304
	(redirection), 404 (incident),)
EXF3	Le pourcentage de requêtes provenant du RIE (Réseau
	Interministériel)
EXF4	La provenance des requêtes (pays et villes)
EXF5	Un nombre de requêtes par intervalle de temps de réponse
	avec un top 10 des requêtes les plus longues

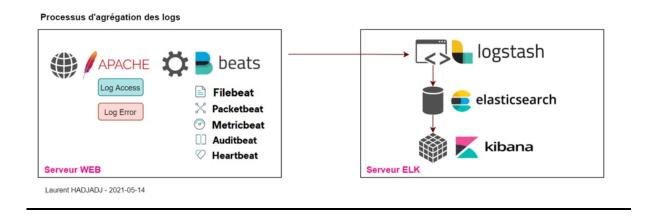
Le tableau de bord automatisé doit également offrir la possibilité de saisir un intervalle de temps afin d'étudier les indicateurs de manière plus précise.

Architecture technique

Comme énoncé précédemment, le tableau de bord automatisé doit s'alimenter d'un outil d'indexation et de recherche de données dénommé Elasticsearch.

Les logs sont chargés à partir de différents composants (Beats et Logstach). Kibana est une surcouche d'Elasticsearch qui permet l'affichage de templates de tableaux de bord.

Voici un schéma de la structure des surcouches ELK (Elastic Logstach Kibana):

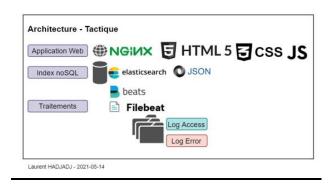


J'ai réalisé le développement de l'application et les appels dans des fichiers Javascript appelés par des fichiers HTML/CSS.

J'ai également développé des requêtes noSql au sein d'objets JSON.

Le code de l'application web est stocké sur un serveur Nginx.

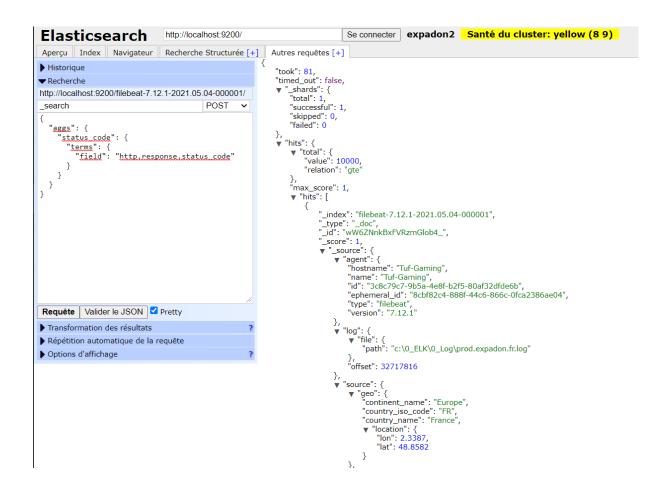
Voici le schéma de l'architecture fournit au démarrage de mon stage :



Réalisation

Première étape

Dans une première étape de mon stage, j'ai voulu me familiariser avec les outils définis dans l'architecture. J'ai notamment commencé à tester du code javascript et des requêtes Elasticsearch dans la console Elasticsearch head (extension web). En local le serveur Elasticsearch écoute sur le port 9200. Les requêtes Elasticsearch (noSql) sont au format JSON et les résultats sont également retournés en format JSON.



Ensuite, il s'agit d'effectuer ces requêtes via le code Javascript. Pour cela, j'ai utilisé les méthodes Ajax prévues à cet effet. Ce type de méthode prend en paramètre la requête JSON à envoyer à Elasticsearch. L'écriture d'une fonction Ajax se fait de cette manière dans le code :

```
function ajax(object){
28
         return $.ajax(
29
30
           url: searchHost+'/_search',
31
           timeout:3000, //3 second timeout
32
           type: 'POST',
33
           dataType: 'json',
34
           contentType: "application/json; charset=utf-8",
35
           crossDomain: true,
           data: JSON.stringify(object)
37
         });
38
```

L'appel de cette fonction se fait de cette manière :

```
//fonction de test (elastic search down ou up)

function requestTest(){

var query = {}; //requête simple juste pour tester ES

ajax(query).done(function(result) {
 | sessionStorage.setItem('request-test',JSON.stringify(result));
}).fail(function(x){
 | console.log(x);
});

var request_test=JSON.parse(sessionStorage.getItem('request-test'));

var request_test=JSON.parse(sessionStorage.getItem('request-test'));
```

La variable request_test stocke les données retournées par l'appel Ajax.

Asynchronisme

J'ai alors rencontré une difficulté que je n'avais pas anticipé avant la réalisation de ces appels. L'affichage de la page HTML se fait avant la récupération des données par l'appel Ajax. En effet, l'appel Ajax a une durée de quelques

secondes pour traiter la requête dans Elasticsearch.

N'ayant que rarement manipulé le langage de programmation Javascript avant mon stage, j'ai alors sollicité mon tuteur de stage qui a identifié la cause de ce « bug » comme un problème d'asynchronisme : le chargement de la page n'est pas synchronisé avec le résultat des appels Ajax.

Il suffit alors d'ajouter l'attribut « async : false » à l'appel Ajax

```
function ajax(object){
28
         return $.ajax(
29
30
          {
            async: false,
31
           url: searchHost+'/ search',
32
33
           timeout:3000, //3 second timeout
           type: 'POST',
34
           dataType: 'json',
           contentType: "application/json; charset=utf-8",
           crossDomain: true,
37
           data: JSON.stringify(object)
38
         });
40
```

L'étape suivante est la manipulation des graphiques proposés par le framework chart.js (Javascript). Ces graphiques prennent en paramètre des noms (exemple : France) avec le nombre de requêtes associé. On peut alors réaliser des graphiques de différents types (en bâton, courbes, camemberts, ...). L'écriture d'un nouveau graphique se fait de cette manière :

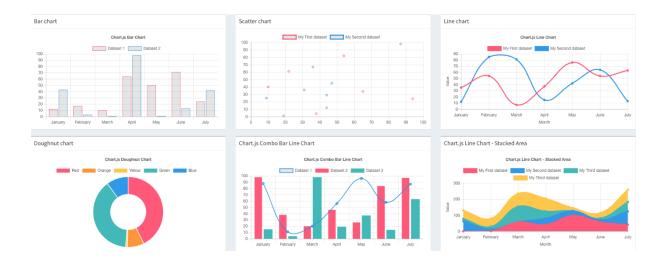
Dans cet exemple, on a indiqué le type : « line » qui est en fait une courbe. Le canvas (élément HTML) paramètre de la fonction getElementById est l'id de l'élément dans lequel on va injecter le graphique dans le fichier HTML. Il s'agit du fonctionnement du framework Chart.js.

Cet élément se présente tout simplement de cette manière :

```
<canvas id="canvas_day_agr" height="auto" aria-label="Graphique : requête par jour pour agrément" role="img"></canvas>
```

Chart.js propose une documentation très complète sur son site ce qui m'a permis de comprendre assez vite son fonctionnement au cours de la réalisation.

Voici des exemples de graphiques pouvant être réalisés à partir du framework Chart.js. :

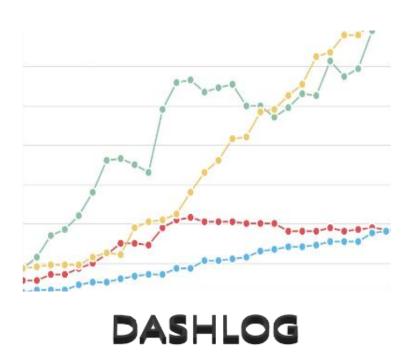


Le logo de l'application

En parallèle, j'ai dû choisir un logo pour le tableau de bord afin de le dissocier des autres applications et comprendre son intérêt.

J'ai donc nommé le tableau de bord automatisé DashLog. J'ai réalisé un logo simple et sobre car l'application est destinée aux membres du service informatique de l'entreprise.

Voici le logo que j'ai choisi :



L'étape suivante de mon stage a consisté dans l'implémentation d'un intervalle de temps dans le tableau de bord automatisé. L'utilisateur doit avoir la possibilité de saisir un intervalle de temps dans un formulaire.

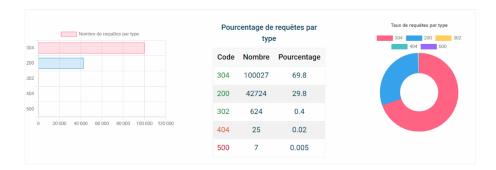
La récupération des données du formulaire se fait par la méthode get. Ces données sont stockées dans l'url et récupérées grâce à un code javascript. Mon idée au départ était l'utilisation d'une méthode post (classique pour un formulaire), mais javascript ne prend pas en charge la récupération de ces données car elles nécessitent un serveur étant capable de les stocker (un serveur PHP, par exemple).

J'ai alors organisé mes graphiques par catégories en utilisant le framework Zurb Foundation 3.0 (HTML/CSS). La raison pour laquelle on a utilisé ce type de framework est que les classes appelées permettent de rendre le tableau de bord « responsive » : sa mise en forme s'adapte en fonction du support : pc, tablette ou encore smartphone.

On indique alors au HTML la taille en colonnes à prendre en fonction de la définition du support: small pour les mobiles, medium pour les tablettes et large pour les ordinateurs. L'écran est alors divisé en 12 colonnes.

Voici un exemple d'utilisation que j'ai pu faire :

La première carte comprenant les types de requêtes s'affiche donc de cette manière avec un support large (type ordinateur) :



Et de cette manière avec un écran réduit (type téléphone) :



L'accessibilité

Il se pose alors également la question de l'accessibilité. Par exemple, une personne aveugle doit pouvoir glisser son curseur sur un élément et savoir de quoi il s'agit. La diction de son ordinateur lui dira alors de quel élément il s'agit (et l'url vers lequel il l'envoie lorsque l'élément est un lien).

Afin d'assurer cette accessibilité et permettre la diction des éléments, il faut indiquer un titre à un élément avec l'attribut « title ».

L'utilisation se fait de cette manière :

<img class="fr-logo" id="dashlog-logo" title="Accueil"</pre>

Dans cet exemple, une personne aveugle pourra glisser son curseur sur l'élément et la diction automatique lui lira le fait qu'il s'agit d'un lien vers l'accueil.

Compréhensibilité de l'application

En prévision de la réunion de présentation de l'application, j'implémente des variables globales qui permettent au service informatique d'adapter le programme facilement à différents modules (l'application étant développée au départ pour le module « agrément » du projet Expadon2).

```
//définition des champs
var fieldDate = "@timestamp";
var fieldStatut = "http.response.status_code";
var fieldPays = "source.geo.country_name";
var fieldVille = "source.geo.city_name";
var fieldTime = "http.response.times";
```

Le code a déjà été commenté pour partie tout au long du développement de l'application. Je prends donc du temps pour me consacrer à la compréhensibilité de l'application en commentant de manière plus détaillée le code de l'application.

La présentation

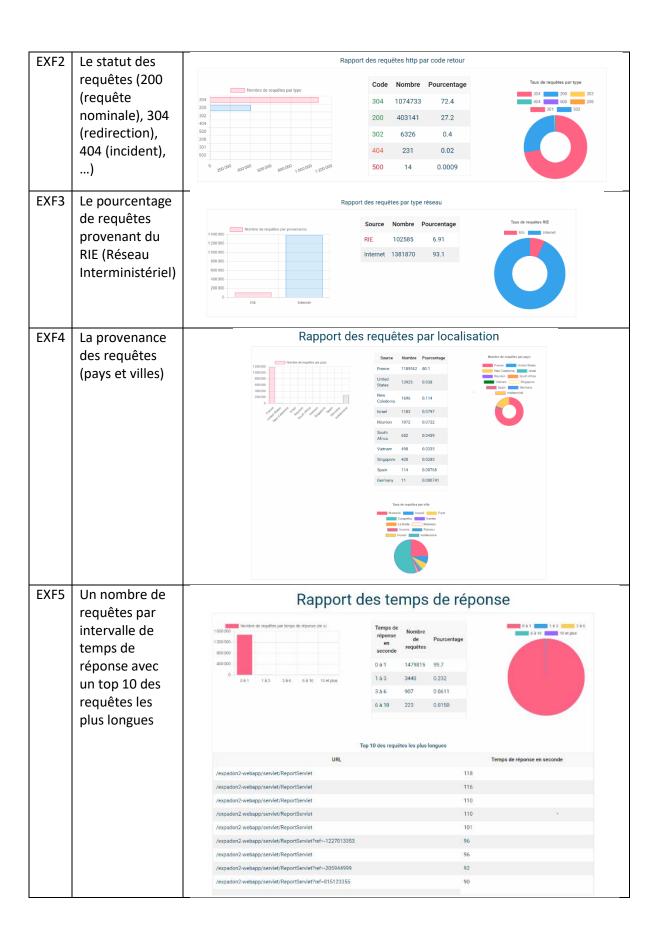
Le stage touchant à sa fin, une réunion de présentation du projet est organisée. Il y a au programme de cette réunion une présentation du contexte et une démonstration de l'utilisation des tableaux de bord. Certaines pistes d'amélioration sont également évoquées lors de cette réunion comme la mise en place de Kibana (surcouche d'Elastic automatique et complète de tableaux de bord) en parallèle des tableaux de bord spécifiques développés dans l'application Dashlog.

Une fois la réunion terminée, le répertoire contenant l'application a été partagé aux membres du service informatique concernés.

Présentation des tableaux des bords finaux

Voici la liste des tableaux de bords avec les exigences fonctionnelles auxquelles ils répondent :





Retour sur expérience

Compétences acquises/développées

Lors de ce stage, j'ai appris à manipuler des outils qui étaient nouveaux pour moi tels que le langage de programmation Javascript avec l'utilisation de différents framework tels que Charts.js (pour les graphiques) ou encore Jquery (bibliothèque de fonction Javascript destinée à synthétiser du code). Je me suis également familiarisé avec l'utilisation de Framework CSS tel que Zurb Foundation. Cela m'a permis de développer de nouvelles connaissances dans le domaine de l'informatique et plus particulièrement dans le domaine du développement et des solutions logicielles.

L'utilisation de ces nouveaux outils m'a donc permis de développer ma capacité à m'adapter dans un nouvel environnement avec lequel je ne suis pas familiarisé au départ.

Pistes d'amélioration de l'application

Selon moi, l'aspect du site à développer est sa charte graphique. Lors des 6 semaines de ce stage, je me suis consacré à la réponse apportée aux exigences fonctionnelles qui m'étaient confiées en délaissant légèrement le développement de la direction graphique de l'application Dashlog même si sa structure reste tout à fait correcte.

Conclusion

Ce stage représente pour moi une toute première expérience dans le monde professionnel. Je suis très heureux de l'avoir effectué au sein d'un établissement public français tel que FranceAgriMer. Je suis également très satisfait de constater que j'ai abouti à une réalisation qui a été partagée au sein du service informatique et qui sera utilisée au quotidien.

Il est à noter que, dans le contexte sanitaire actuel, le stage s'est déroulé la plupart du temps en télétravail.