

10/12/2019

RAPPORT DE STAGE

Import via une API et traitements de fichiers XML.

Implémentation d'une base de données relationnelle et insertions des datas.

Visualisation des datas sous Flask

Titre professionnel

« Concepteur Développeur d'Application mention Data Analyst »

CONTENU

1	PRESENTATION DU CANDIDAT	3
2	PRESENTATION DE L'ENTREPRISE.....	4
3	PROJECT SUMMARY	5
4	DESCRIPTION DU PROJET	6
4.1	Les acteurs du projet	6
4.2	Genèse du projet	6
4.3	Schéma descriptif	7
4.4	Planning de réalisation	8
5	DECOUVERTE DE L'API E-BIHR.....	9
5.1	Présentation de l'API TEST mise à disposition par BIHR	9
5.2	Protocole de connexion à l'API TEST	9
5.3	Sélection des catalogues.....	9
5.4	Premiers tests de requêtage.	11
5.5	Passage de l'API de TEST à l'API de PROD.	12
6	IMPORT DES FICHIERS XML ISSUS DE L'API.....	13
6.1	Fonctionnalités attendues	13
6.2	Outils de développement	13
6.3	Analyse des principaux éléments de code : API_BIHR_CoDoUn.ipynb	14
7	PREMIERES ANALYSES DE LA STRUCTURE DES FICHIERS XML.....	17
7.1	Outils utilisés pour analyser les .XML.....	17
7.2	Arborescence des fichiers XML	18
7.3	Analyse du fichier cat-ref-FR3787ED	19
7.4	Analyse du fichier cat-prices	19
7.5	Analyse du fichier cat-images	19

7.6	Analyse du fichier cat-attributes.....	20
8	IMPLEMENTATION DE LA BASE DE DONNEES	21
8.1	Installation du serveur MYSQL.....	21
8.2	Détermination des tables et dictionnaires	22
8.3	Création des tables : CONNECT DB & CREATION DES TABLES .ipynb	24
8.4	Modèle conceptuel de données réalisé avec phpMyAdmin	25
8.5	Le MCD fait apparaitre une erreur de conception	26
9	TRAITEMENTS DES FICHIERS ET INSERTION DES DATAS DANS LA BASE.....	27
9.1	Import des bibliothèques Python.....	27
9.2	Parsage du fichier avec Lxml	28
9.3	Génération du dataframe Pandas	28
9.4	Les traitements effectués sous Pandas	30
9.5	L'insertion des données du dataframe dans la table	32
10	LIVRABLE : IMPLEMENTATION DU DUMP DE LA BASE BIHR_DB	34
11	VISUALISATION DES DONNEES AVEC FLASK.....	35
11.1	Fonctionnalités attendues	35
11.2	Maquettage du site	36
11.3	Implémentation de site sous FLASK.	38
11.4	TODO.....	41
12	CONCLUSIONS.....	42
13	REMERCIEMENTS	43
14	RESSOURCES	44
15	ANNEXES	45

1 PRESENTATION DU CANDIDAT

Titulaire d'un BTS professions immobilières, j'ai une expérience professionnelle de dix années en gestion immobilière pour le compte d'investisseurs institutionnels. A partir de 1996, j'ai rejoint le Crédit Agricole à Issy-les-Moulineaux où nous gérons la couverture immobilière de la branche assurance, UNIFICA PACIFICA. Par la suite, j'ai intégré le service ADB de la Société Générale où nous étions mandatés pour gérer les actifs des SCPI conçues par le service d'Asset Management.

Revenu sur Tours en 2005, j'ai occupé un poste de développeur commercial au sein d'une PME spécialisée dans les sports mécaniques. Outre la promotion commerciale auprès des utilisateurs finaux, j'ai mené de nombreuses actions de lobbying auprès de la FFSA (Fédération Française du Sport Automobile), conclus des accords commerciaux avec les principaux distributeurs répartis sur tout le territoire et contribué à la promotion des actions de l'entreprise auprès de la presse spécialisée.

Fin 2015, m'étant formé à l'utilisation de SolidWorks, j'ai intégré le bureau d'étude d'un constructeur de grues forestières pour effectuer la modélisation 3D et la mise au jour des liasses de plan durant 18 mois.

Au terme de ce contrat, étant toujours curieux et souhaitant me former aux technologies de l'information, l'opportunité d'intégrer la première promotion Data Analyst du CEFIM s'est offerte à moi.

J'apprécie particulièrement les solutions de Data Management et souhaite approfondir mes connaissances sur ces sujets.

Le travail que je vous présente aujourd'hui a été réalisé sur la période de stage en entreprise allant du 29 octobre au 27 novembre 2019.

2 PRESENTATION DE L'ENTREPRISE



S-BIKE 37 est un concessionnaire multi-marques moto et quad installé à Saint Cyr sur Loire (37). Une équipe de quatre personnes anime cette sympathique structure reprise début 2017 par Olivier FALCON et son épouse.

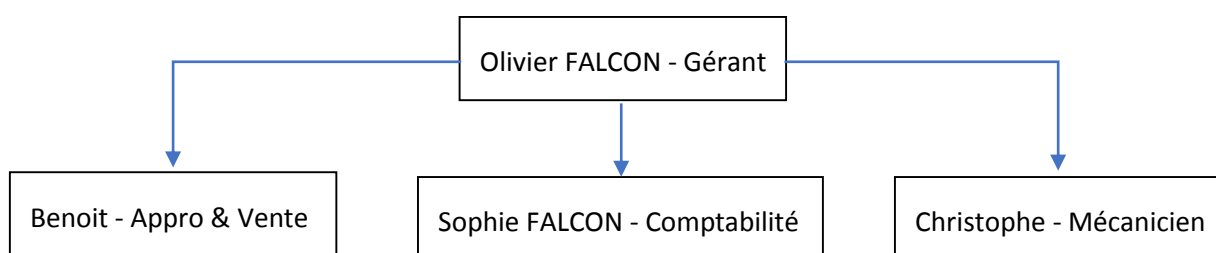


Figure 1 : Organigramme S-BIKE 37

Les principales activités de l'entreprise sont :

- Vente de motocyclettes et quads neufs ou d'occasions.
- Réparations et entretiens des véhicules.
- Vente d'équipements et d'accessoires pour la pratique de la moto.

La clientèle est constituée essentiellement de particuliers et de quelques professionnels pour la partie quads agricoles.

En septembre 2019, l'entreprise a pris livraison d'un site internet réalisé sous WordPress par W3P, une agence web basée à Veigné (37).

3 PROJECT SUMMARY

Good afternoon,

I am François Bastier, 46 years old. I worked for 10 years in the property management departments of two major French banks.

I came back to Tours in 2005 and worked in a small company specialized in the manufacture of karts.

In 2019, I joined and followed the "DATA ANALYST" training course provided by CEFIM.

The project I am presenting to you today was carried out in November 2019 at the request of S-BIKE 37, a motorcycle dealer based in Saint Cyr sur loire.

The purpose of this project is to recover data from BIHR, a wholesaler specialized in the distribution of motorcycle parts, equipment, consumables and accessories. The recovered data will be processed and inserted into a MySQL database. A copy of this database will then be sent to W3P, a web agency to feed the S-BIKE 37 online store.

Moreover, in order to complete the project, a data access and visualization solution will be implemented using a site created with the FLASK framework.

I started the project by asking what were the expectations and needs of the different actors.

Then I documented the solutions available to me to carry out this work.

After proposing a plan and obtaining the agreement of the various protagonists, I started the development using my best knowed tools (or I should say my least unknown...).

Despite its simplicity, this project represents a large number of hours spent to documenting myself and looking for solutions to the problems that have arisen over time.

I hope you will be as interested in this project as I enjoyed working on it.

4 DESCRIPTION DU PROJET

4.1 Les acteurs du projet



Concessionnaire motos et quads, S-BIKE 37 est le maître d'ouvrage de ce projet. Ayant pris livraison d'un site internet réalisé par la société W3P, Olivier FALCON souhaite voir les produits distribués par la société BIHR sur ce site. A cette fin, il s'est adressé à W3P pour implémenter l'API mise à disposition par le grossiste.



W3P est une agence WEB spécialisée dans la réalisation de sites web sous Wordpress. Elle a réalisé le site www.sbike37.com sous WordPress et implémenté le module WooCommerce pour la création de la boutique en ligne. Face à la demande complémentaire de S-BIKE 37 et ne pouvant gérer la mise en place de l'API avant mars 2020, Willy Bonneau, le gérant de W3P, a proposé que nous lui fournissions un « dump » SQL des données fournies par l'API pour pouvoir insérer les produits rapidement sur la boutique en ligne.



Acteur européen incontournable sur le marché des pièces et accessoires motocyclistes, la société BIHR, basée à Bartenheim (68), possède un catalogue de plus de 160 000 références. Afin de faciliter l'accès à ses catalogues, à ses stocks et faciliter les prises de commandes, BIHR a développé une API, nommée e-Bihr, destinée initialement à être implémentés sur les CRM de ses clients.

4.2 Genèse du projet

« Hé François, maintenant que le site web existe, je veux pouvoir vendre les produits BIHR sur la boutique en ligne avant Noël. J'ai vu qu'ils nous mettent à disposition une API. Va voir et tiens-moi au courant. »

-- Olivier Falcon – octobre 2019 –

C'est ainsi que débute ce projet.

Ma première action est de contacter la société BIHR pour avoir de plus amples informations sur les fonctionnalités de l'API. Mon correspondant me demande de me connecter à l'adresse suivante : <https://apitest.bihr.net:8047/Help/QuickStart/> (Annexe) où sont décrites toutes les fonctionnalités de l'API de façon exhaustive.

Ma seconde action est de contacter la société W3P pour l'informer de la demande d'Olivier. Après prise de connaissance du document cité ci-dessus, Willy me répond qu'il ne peut pas s'en occuper avant Noël. Au mieux, si je lui fournis des données propres, il peut utiliser l'API REST du module Woocommerce pour alimenter la boutique en ligne.

Après avoir informé Olivier de mes échanges avec BIHR et W3P, je propose de fournir à Willy une base de données MySQL contenant les informations nécessaires pour alimenter les fiches produites et décrivant l'arborescence du catalogue.

4.3 Schéma descriptif

L'architecture du projet apparait distinctement sur le schéma descriptif ci-dessous : L'API, le backend, le SGBDR, le livrable et le frontend.

Le langage Python et du micro-Framework FLASK ont été retenus parce qu'ils ont fait l'objet d'enseignements durant ces neuf mois passés au CEFIM.

L'utilisation de MySQL comme SGBDR répond au souhait émis par W3P.

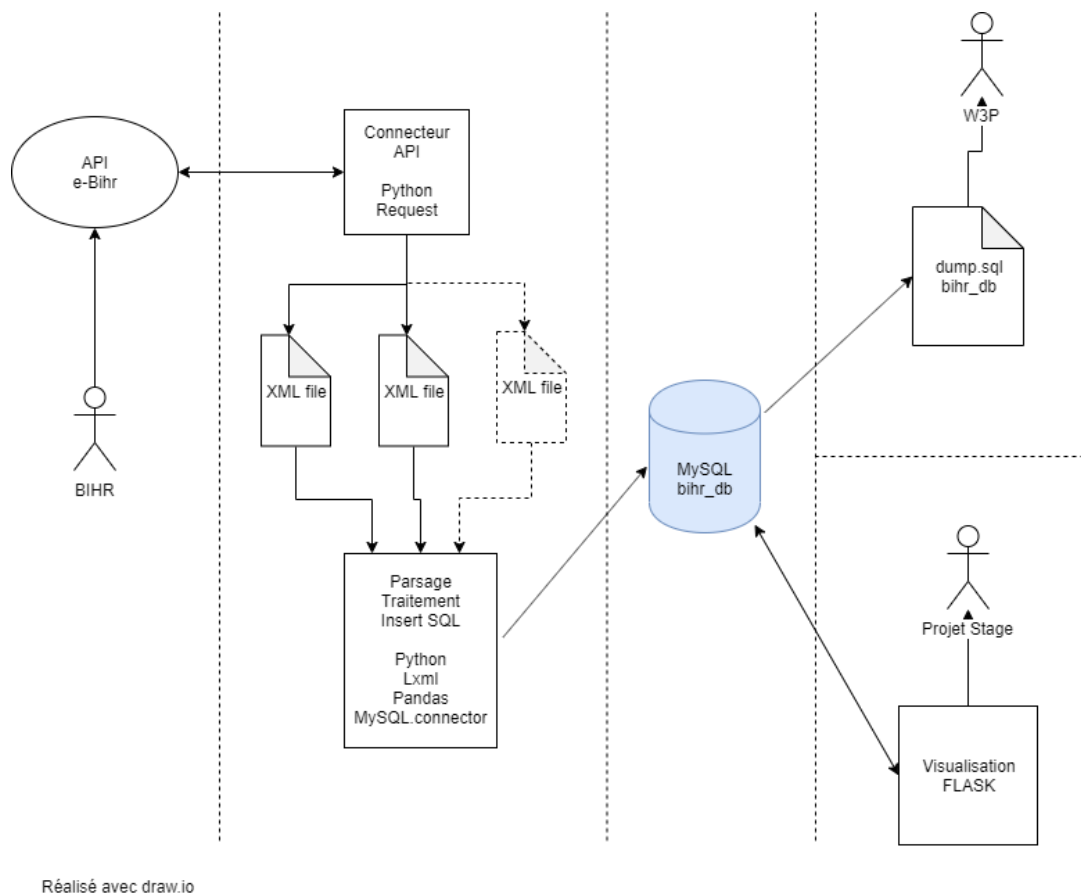


Figure 2 : Schéma descriptif du projet

4.4 Planning de réalisation

Le temps imparti pour l'étude et la réalisation de ce projet était raisonnable pour un professionnel expérimenté. En tant que « junior » très peu expérimenté, les délais furent très courts. Travaillant seul, j'ai essayé de m'astreindre au planning dont vous trouverez l'illustration ci-dessous.



Figure 3 : Diagramme de Gantt

<input type="radio"/>	Découverte de l'API BIHR	T	28/10/2019	01/11/2019	0.2 months	100
<input type="radio"/>	Tests de fonctionnement	T	28/10/2019	05/11/2019	0.3 months	100
<input type="radio"/>	Création du module d'interrogation de l'API	T	05/11/2019	08/11/2019	0.1 months	80
<input type="radio"/>	Analyse des XML - réalisation du MCD pour la base SQL	T	08/11/2019	11/11/2019	0.1 months	100
<input type="radio"/>	Création de la base Mysql et de ses tables suivant MCD	T	12/11/2019	12/11/2019	0 months	100
<input type="radio"/>	Création des modules de traitement des fichiers XML (parsage, nettoyage et insert...	T	13/11/2019	23/11/2019	0.4 months	80
<input type="radio"/>	Livraison du dump de la base Mysql à W3P	M	27/11/2019	27/11/2019	-	100
<input type="radio"/>	Réalisation du Front-end sous Flask	T	25/11/2019	05/12/2019	0.4 months	50

Figure 4 : Planning

Les délais impartis pour la réalisation du projet ne m'ont pas permis de procéder à une analyse fonctionnelle poussée. Cela aura pour conséquence une erreur dans la conception du modèle conceptuel de données (MCD) qui sera mise en évidence dans le chapitre consacré à l'implémentation de la base de données.

5 DECOUVERTE DE L'API E-BIHR

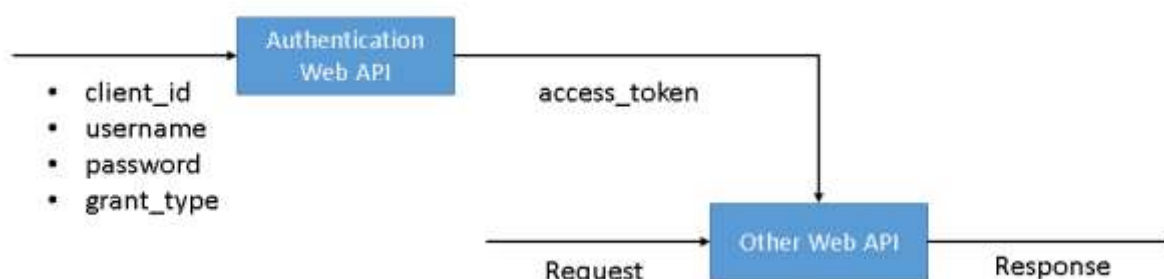
5.1 Présentation de l'API TEST mise à disposition par BIHR

L'API mise à disposition par la société BIHR permet d'obtenir des catalogues (CATALOG), de vérifier la disponibilité des produits (INVENTORY), de créer et de suivre des commandes (ORDER).

Elle est destinée principalement aux clients professionnels de la société pour alimenter leurs CRM.

La documentation se trouve à l'adresse suivante : <https://apitest.bihr.net:8047/Help>

5.2 Protocole de connexion à l'API TEST



La documentation nous informe des étapes à suivre pour appeler n'importe quelle API dont nous aurions besoin.

La première étape consiste à obtenir un access token auprès de l'API d'authentification. Ce jeton permettra ensuite d'interroger les autres web API disponibles.

Pour télécharger un catalogue, la procédure est la suivante :

- 1^{ère} étape : Obtenir l'access_token auprès de l'API d'authentification
- 2^{ème} étape : Demande de récupération d'un catalogue : il faut préalablement obtenir un ticket
- 3^{ème} étape : L'obtention du ticket permet de demander une génération de catalogue
- 4^{ème} étape : Vérifier l'avancement de la génération du lien de téléchargement renvoyé au terme de la génération du catalogue. L'API nous fournit alors un download_Id
- 5^{ème} étape : Télécharger le catalogue avec le download_Id.

5.3 Sélection des catalogues

La documentation décrit la totalité des Web API disponibles. Nous avons retenu les WEB API suivantes :

API	Description
POST https://apitest.bihr.net:8046/oauth2/token	Gets a temporary access token.
GET api/v2/Catalog/fullref/	Requests the Full References Catalog Generation. This catalog is one xml file containing all the products and their properties. This xml file is zipped in a *.7z file.
GET api/v2/Catalog/fullprices/	Requests the Full Prices Catalog Generation. This catalog is one xml file containing all the products, their prices and discount rates. This xml file is zipped in a *.7z file.
GET api/v2/Catalog/fullimg/	Requests the Full Images Catalog Generation. This catalog is one xml file containing all the products and their main picture name. These image names must be concatenated to the Product Root URL (provided by the Catalog/productsrooturl API) to get the full image URL. This xml file is zipped in a *.7z file.
GET api/v2/Catalog/fullattr/	Requests the Full Attributes Catalog Generation. This catalog is one xml file containing all the products and their attributes. This xml file is zipped in a *.7z file.
GET api/v2/Catalog/status/?ticketId={ticketId}	Returns the Catalog Generation Status for a requested Catalog.
GET api/v2/Catalog/productsrooturl/	Returns the product pictures root URL. This root URL has to be concatenated with the product images URL provided in the Catalogs.
GET api/v2/Catalog/brandimage/?brandId={brandId}	Returns the image name of the requested brand. This name has to be concatenated with the brand root URL provided by the /Catalog/brandsrooturl API.
GET api/v2/Catalog/brandsrooturl/	Returns the brands root URL. This root URL has to be concatenated with the brand images provided by the Catalog/brandimage API.

Afin de bien comprendre comment fonctionne l'API de test, j'ai procédé à quelques requêtes avec Curl en ligne de commande. Le lecteur trouvera un exemple de requêtage dans l'illustration ci-après.

```
C:\Administrateur : C:\Windows\system32\cmd.exe
```

```
Microsoft Windows [version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Tous droits réservés.
```

```
C:\Windows\system32>curl -H "Content-Type: application/x-www-form-urlencoded" -X POST -d "client_id=8274d6301e274557b4d13ad14f54ff3f&grant_type=password&username=FR3787ED&password=238E41E4269640B8CA95BA182D31D7DD" https://apitest.bihr.net:8046/oauth2/token
```

```
<"access_token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1bmRldWUfbmFtZSI6IkZSMzc4N0UElwiic3UiljoiriIzNzg3RUQilCjYy2xlljpbik1hhbmFnZXIiLCJTaXB1cnZpc29yIj0sImZlcyl6ImlhOdBHBzOj8vYXBPdGVzdC5iaWhYLm5ldDo4MDQ2LglsImF1ZCI6IjgyNzRkNmMwMTAgNzQ1NTdiNGQxM2FkMTRmNTRmZjNmIiwiaWF0IjE1NzA1OTQ2MDU9Lij-WESXjlIEGxp3g-ksBgU-uUrEKb-M-YK7s_G7QJ8g","token_type":"bearer","expires_in":1799>
```

```
C:\Windows\system32>curl -H "Authorization: bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1bmRldWUfbmFtZSI6IkZSMzc4N0UElwiic3UiljoiriIzNzg3RUQilCjYy2xlljpbik1hhbmFnZXIiLCJTaXB1cnZpc29yIj0sImZlcyl6ImlhOdBHBzOj8vYXBPdGVzdC5iaWhYLm5ldDo4MDQ2LglsImF1ZCI6IjgyNzRkNmMwMTAgNzQ1NTdiNGQxM2FkMTRmNTRmZjNmIiwiaWF0IjE1NzA1OTQ2MDU9Lij-WESXjlIEGxp3g-ksBgU-uUrEKb-M-YK7s_G7QJ8g" -X GET https://apitest.bihr.net:8047/api/v2/Catalog/fullref
```

```
<"TicketId":"0bf1f5e56f9b40e3b34a2c570bbef825","ResultCode":"OK">
```

```
C:\Windows\system32>
```

```
C:\Windows\system32>curl -H "Authorization: bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1bmRldWUfbmFtZSI6IkZSMzc4N0UElwiic3UiljoiriIzNzg3RUQilCjYy2xlljpbik1hhbmFnZXIiLCJTaXB1cnZpc29yIj0sImZlcyl6ImlhOdBHBzOj8vYXBPdGVzdC5iaWhYLm5ldDo4MDQ2LglsImF1ZCI6IjgyNzRkNmMwMTAgNzQ1NTdiNGQxM2FkMTRmNTRmZjNmIiwiaWF0IjE1NzA1OTQ2MDU9Lij-WESXjlIEGxp3g-ksBgU-uUrEKb-M-YK7s_G7QJ8g" -X GET https://apitest.bihr.net:8047/api/v2/Catalog/status/0bf1f5e56f9b40e3b34a2c570bbef825
```

```
<"PositionInQueue":0,"RequestStatus":"PROCESSING","DownloadId":null>
```

```
C:\Windows\system32>
```

```
C:\Windows\system32>curl -H "Authorization: bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1bmRldWUfbmFtZSI6IkZSMzc4N0UElwiic3UiljoiriIzNzg3RUQilCjYy2xlljpbik1hhbmFnZXIiLCJTaXB1cnZpc29yIj0sImZlcyl6ImlhOdBHBzOj8vYXBPdGVzdC5iaWhYLm5ldDo4MDQ2LglsImF1ZCI6IjgyNzRkNmMwMTAgNzQ1NTdiNGQxM2FkMTRmNTRmZjNmIiwiaWF0IjE1NzA1OTQ2MDU9Lij-WESXjlIEGxp3g-ksBgU-uUrEKb-M-YK7s_G7QJ8g" -X GET https://apitest.bihr.net:8047/api/v2/Catalog/status/0bf1f5e56f9b40e3b34a2c570bbef825
```

```
<"PositionInQueue":0,"RequestStatus":"PROCESSING","DownloadId":null>
```

```
C:\Windows\system32>
```

```
C:\Windows\system32>curl -H "Authorization: bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1bmRldWUfbmFtZSI6IkZSMzc4N0UElwiic3UiljoiriIzNzg3RUQilCjYy2xlljpbik1hhbmFnZXIiLCJTaXB1cnZpc29yIj0sImZlcyl6ImlhOdBHBzOj8vYXBPdGVzdC5iaWhYLm5ldDo4MDQ2LglsImF1ZCI6IjgyNzRkNmMwMTAgNzQ1NTdiNGQxM2FkMTRmNTRmZjNmIiwiaWF0IjE1NzA1OTQ2MDU9Lij-WESXjlIEGxp3g-ksBgU-uUrEKb-M-YK7s_G7QJ8g" -X GET https://apitest.bihr.net:8047/api/v2/Catalog/status/0bf1f5e56f9b40e3b34a2c570bbef825
```

```
<"Message":"Authorization has been denied for this request.">
```

```
C:\Windows\system32>
```

Comme on peut le constater ci-dessus, ce procédé peu pratique a échoué suite au dépassement du délai de validité du token.

5.5 Passage de l'API de TEST à l'API de PROD.

Au terme du développement de l'application de requêtages des Web API (sujet décrit dans le chap. 6), j'ai contacté BIHR pour obtenir les identifiants de production.

Je les ai obtenus sous 24 heures par mail ainsi que l'adresse de la documentation afférente à l'API de prod.

<https://api.bihr.net/api-docs/index.html>

Je constate quelques différences entre l'API de TEST et l'API de Prod :

- Ma requête auprès de l'API d'authentification ne fonctionne plus. (Sujet abordé dans le chap. 6)
- Certains services sont dépréciés :

GET /api/v2/Catalog/status/{ticketId}

Obsolete: Use /api/v2/Catalog/status instead. Returns the Catalog Generation Status for a requested Catalog.

GET /api/v2/Catalog/brandimage/{brandId}

Obsolete: Use /api/v2/Catalog/brandimage instead. Returns the image name of the requested brand.

Cette nouvelle documentation offre par contre la possibilité de tester chaque Web API en ligne et informe mieux sur les modèles de réponses.



GET /api/v2/Catalog/fullref Requests the Full References Catalog Generation (Type 0).

This catalog is one file containing all the products and their properties.

Parameters

No parameters

Responses

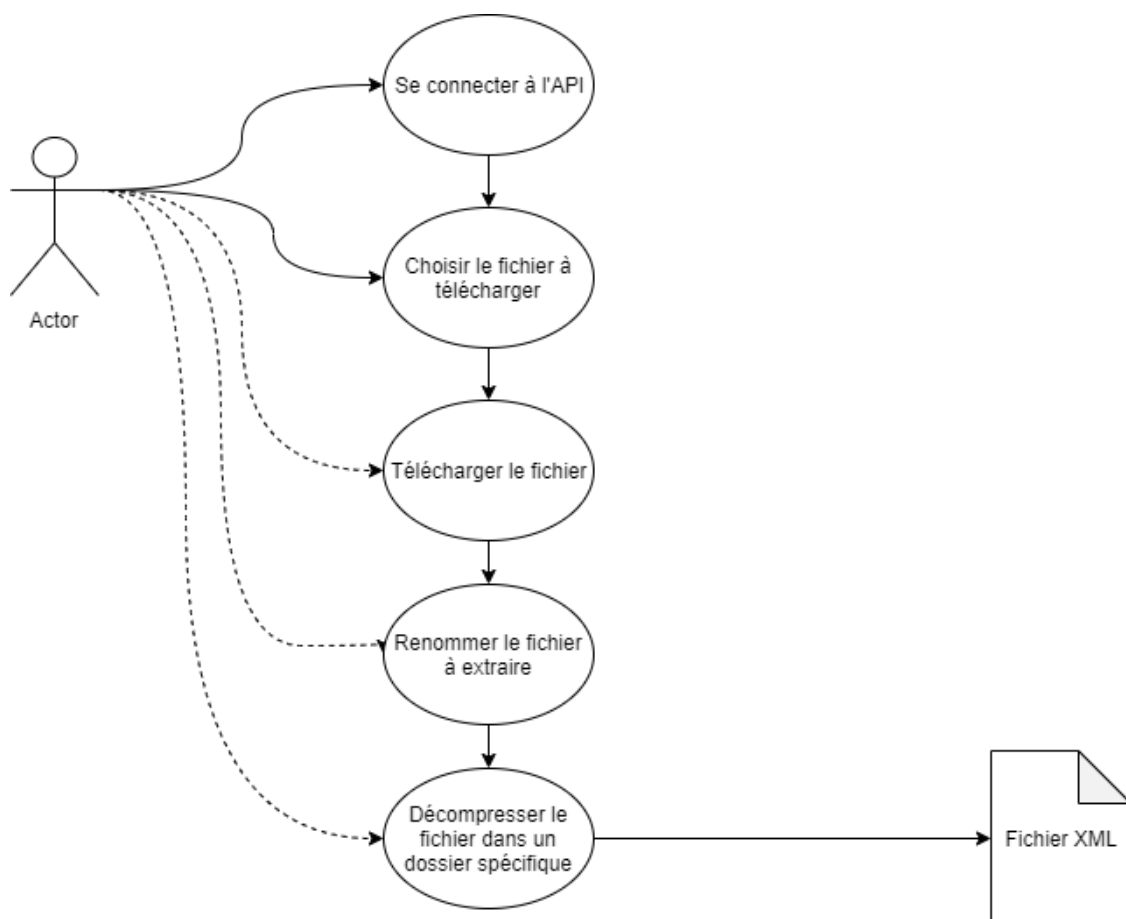
Response content type: text/plain

Code	Description
200	An object containing the corresponding Request Ticket id. This Ticket id can be used as input of the Catalog/status API to get the Request Status. Example Value : User <pre>{ "TicketId": "7982c9ed13e4176e891ed179e533d", "ResultCode": "OK"}</pre>
401	When the API call is unauthorized
403	When the API call is forbidden
429	When the API call quota exceeded

```
CatalogGenerationResponseDataContract {  
    PositionInQueue    Integer(32x32)  
    // example: 0  
    // The generation position in queue.  
    RequestStatus      string  
    // example: OK  
    // The request status.  
    DownloadId         string  
    // example: 8e782cb7f8134cb54c58a9f08f0c0b  
    // The download id.  
}
```

6 IMPORT DES FICHIERS XML ISSUS DE L'API

6.1 Fonctionnalités attendues



6.2 Outils de développement

Le développement a été effectué sous JUPYTER NOTEBOOK.

Librairies utilisées :

- Urllib (module natif)
- Json (module natif)
- Requests
- Time (module natif)
- PYunpack

6.3 Analyse des principaux éléments de code : API_BIHR_CoDoUn.ipynb

Les éléments d'authentification fournis par BIHR ont été rangés dans un fichier json de façon à ne pas les faire apparaître directement dans le code. On les charge en lecture avec la méthode load de la librairie Json.

```
1 authn_settings = json.load(open("private/authn_prod.json", "r"))
2
3 data = {
4     'Username': authn_settings['customerId'],
5     'Password': authn_settings['securityKey']
6 }
7
8 post_data = urlencode(data).encode('utf8')
```

Les lignes de code ci-dessous permettent à l'utilisateur de choisir le fichier à télécharger.

```
1 choice = {'1': 'fullref',
2           '2': 'fullprices',
3           '3': 'fulling',
4           '4': 'fullattr'
5           }
6 for x, y in choice.items():
7     print(x, "-", y)
8
9 yourChoice = input("Saisissez votre choix : ")
10 catalog = choice[yourChoice]
11 print(f'Téléchargement du fichier {catalog}')
```

J'avais utilisé initialement la librairie Requests pour faire la demande d'identification. Sur APITEST, Requests fonctionnait parfaitement. Lors du passage à l'API de production, il me fut impossible d'obtenir le token d'accès. Le problème fut résolu en utilisant la librairie urllib.

[illegible]

Comparaison des requêtes CURL fournies dans la documentation :

API TEST :

```
Curl -H "Content-Type: application/x-www-form-urlencoded" -X POST -d  
"client_id={clientID}&grant_type=password&username={CustomerID}&password={securityKey}"  
https://apitest.bihr.net:8046/oauth2/token
```

API PROD:

```
Curl -X POST "https://api.bihar.net/api/v2/Authentication/token" -H "accept: text/plain" -H "Content-Type: multipart/form-data" -F "UserName={CustomerID}" -F "PassWord= {securityKey}"
```

API TEST : Les variables sont transmises via POST avec le paramètre -d <data> et doivent être préalablement url-encodées.

API PROD : Les variables sont transmises avec le paramètre -F « fichier »

Infos disponibles : <https://curl.haxx.se/docs/manual.html>

Je n'ai pas réussi à reproduire ce changement avec la librairie Requests.

Demande d'authentification pour récupérer un token d'accès.

```
1 with urlopen("https://api.bihr.net/api/v2/Authentication/token", post_data) as response:
2     resp = json.load(response)
3
4     access_token = resp['access_token']
5
6     access_token
```

Demande d'un Ticket_ID précisant le catalogue choisi.

```
1 headers = {'Authorization': f'bearer {access_token}',}
2
3 Ticket = requests.get(f'https://api.bihr.net/api/v2/Catalog/{catalog}', headers=headers)
4 resp = Ticket.json()
5
6 Ticket_ID = resp['TicketId']
7
8 Ticket_ID
```

A l'aide du Ticket_ID, on effectue une demande de génération de catalogue.

```
1 Download = requests.get(f'https://api.bihr.net/api/v2/Catalog/status?ticketId={Ticket_ID}', headers=headers)
2 resp = Download.json()
3
4 Download_ID = resp['DownloadId']
5
6 status = resp['RequestStatus']
7 PositionInQueue = resp['PositionInQueue']
8
9 print(status)
10 print(f'PositionInQueue = {PositionInQueue}')
```

La durée de génération du catalogue est variable. Il est nécessaire de relancer la requête régulièrement jusqu'à ce que le catalogue soit généré et que le Download_ID soit communiqué en réponse.

A cette fin, j'ai utilisé une boucle While pour relancer le processus et la méthode sleep de la librairie time pour éviter de surcharger le serveur. (time.sleep est réglé à 10 secondes d'attente dans l'exemple suivant)


```
WAITING  
115  
WAITING  
115  
WAITING  
115  
WAITING  
115  
WAITING  
115  
WAITING  
115  
WAITING  
115  
WAITING  
115  
DONE  
0  
Download_Id = 1402cb04a1a946269e34ddd74e7846f2
```

Téléchargement du catalogue.

```
1 url = f'https://api.bihir.net/api/v2/Catalog/download?downloadId={Download_ID}'
2
3 r = requests.get(url, headers=headers)
```

Méthode utilisée pour renommer le fichier téléchargé.

```
1 r.headers
```

```
{'Server': 'nginx/1.15.10', 'Date': 'Sat, 07 Dec 2019 10:19:05 GMT', 'Content-Type': 'application/octet-stream', 'Content-Length': '4505551', 'Connection': 'keep-alive', 'Content-Disposition': 'attachment; filename="cat-ref-FR3787ED-2019-12-07 11-04.01-07-12-2019-11-13-03.7z"; filename=UTF-8\\\'cat-ref-FR3787ED-2019-12-072011.04.01-07-12-2019-11-13-03.7z"', 'X-Rate-Limit-Limit': '7d', 'X-Rate-Limit-Remaining': '29999', 'X-Rate-Limit-Reset': '2019-12-14T10:19:04.9519764Z'}
```

```
1 dfile = r.headers['Content-Disposition']
2 dfile = dfile.split("; ")
3 dfile = dfile[1]
4 dfile = dfile.replace("filename=", "")
5 dfile = dfile.replace("\'", "")
6
7 dfile
```

'cat-ref-FR3787EO 2019-12-07 11.04.01-07-12-2019 11-13-03.72'

Rangement du fichier téléchargé dans son dossier de destination. (download_Files/)

```
1 with open(f"download_Files/{dfile}", "wb") as file:
2     file.write(r.content)
```

Décompression du fichier à l'aide de PYunpack et rangement dans dossier (unzipped_files/).
PYunpack est la seule librairie que j'ai trouvée qui me permette de décompresser les fichiers .7z.

```
1 # On dézippe le fichier reçu et on le range dans dossier unzipped
2 new = Archive('download_files/{dfile}').extractall('unzipped_files/.')
3
```

7 PREMIERES ANALYSES DE LA STRUCTURE DES FICHIERS XML

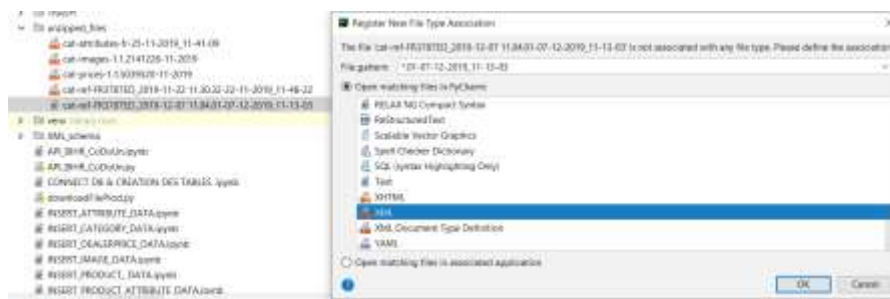
Je dispose à présent de quatre fichiers différents qui ne sont pas encore associés au format XML.

- cat-attributes
- cat-images
- cat-prices
- cat-ref-FR3787ED

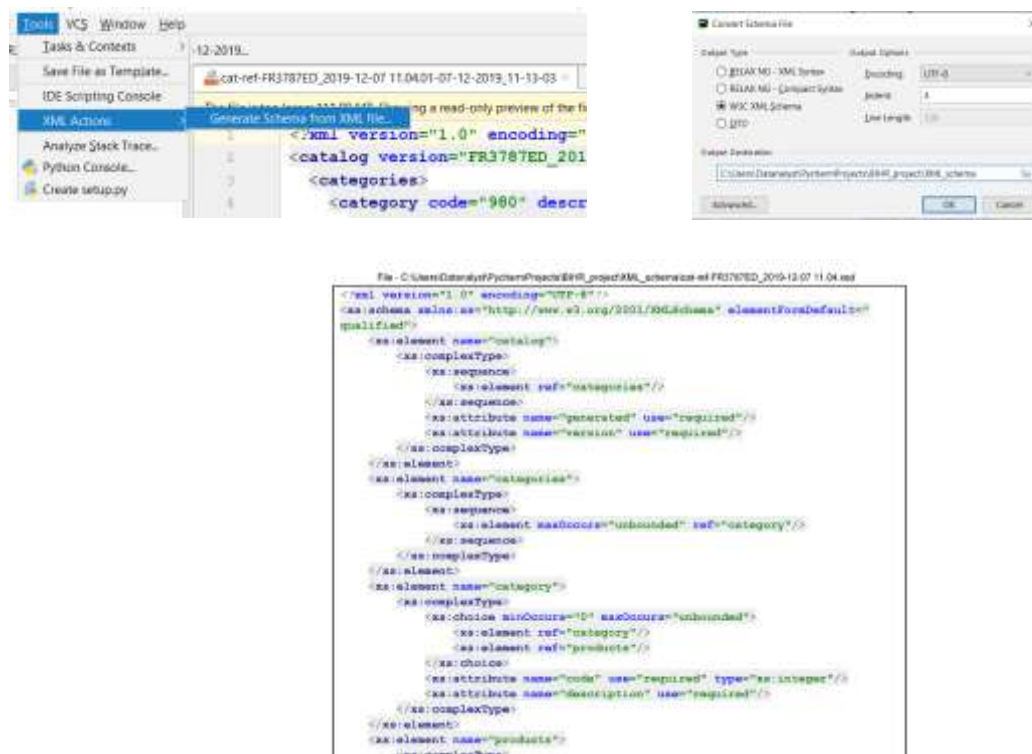


7.1 Outils utilisés pour analyser les .XML

PyCharm permet d'associer un fichier sans suffixe à un type donné.



Pycharm permet aussi de générer un schéma .xsd à partir d'un fichier XML

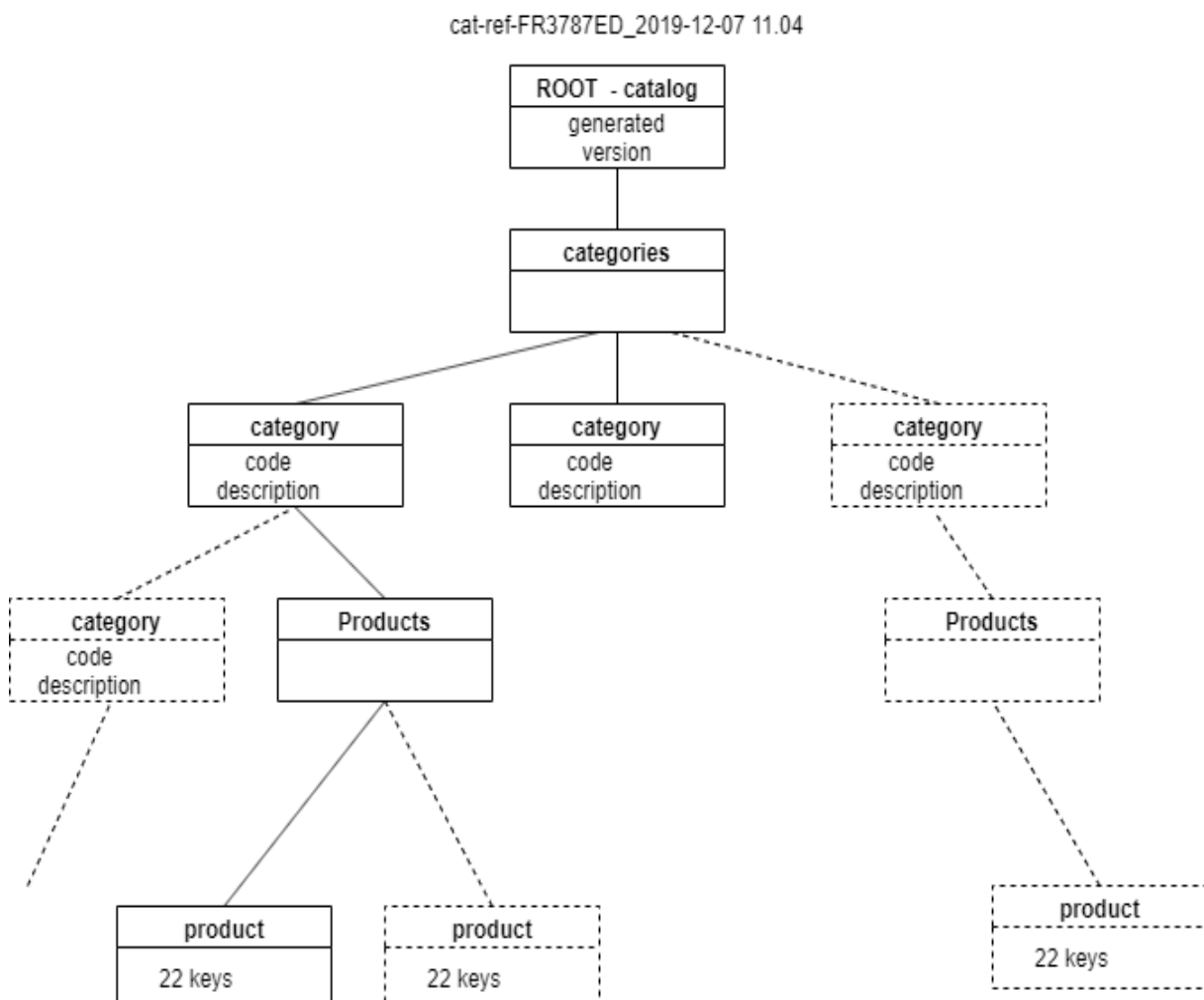


N'étant pas un spécialiste du XML et de la lecture des fichiers .xsd, j'ai utilisé le logiciel XML Viewer édité par Mindfusion pour faciliter l'analyse des quatre fichiers. Ce petit logiciel permet d'ouvrir, de parcourir et de faire une recherche très facilement dans des fichiers XML.



7.2 Arborescence des fichiers XML

Pour chacun des quatre fichiers XML et à l'aide du .xsd et de par XML Viewer, j'ai réalisé un schéma de l'arborescence à la manière de l'exemple ci-dessous. Ces schémas me fourniront une aide précieuse pour définir la méthode à utiliser pour parcourir les fichiers et définir les informations à extraire.



7.3 Analyse du fichier cat-ref-FR3787ED

Notre intérêt se porte sur les balises <category> et <product>.

<category> comporte deux attributs permettant de créer les catégories de produits et d'y affecter les produits. Une balise <category> peut être parent d'une autre balise <category>.

<product> comporte vingt-deux attributs décrivant le produit (prix, taille du colis, référence...)

Hiérarchiquement, ces deux balises sont toujours séparées par la balise <products>.

7.4 Analyse du fichier cat-prices

Notre intérêt se porte sur la balise <price>.

<price> comporte deux attributs décrivant le prix professionnel (productID, dealerPriceHT)

Il est constaté, que contrairement ce qui mentionné dans la documentation, l'attribut DiscountRate n'apparaît pas dans le fichier que nous avons reçu.

Price Node

Attribute	Type	Description	Example
ProductId	String	Product Code	0040038
DealerPriceHT	Decimal	Dealer Price excluding VAT	96,583333
DiscountRate	Decimal	Discount Rate (can be null)	0.00

```
cat-prices-1.1.5039520-11-20...
1 <?xml version="1.0" encoding="utf-8"?>
2 <catalog version="1.1.50395" generated="2019-11-20 14:01:25">
3   <prices>
4     <price productId="4430024202" dealerPriceHT="27,84" />
5     <price productId="4430024102" dealerPriceHT="35,75" />
```

7.5 Analyse du fichier cat-images

Notre intérêt se porte sur la balise <product>.

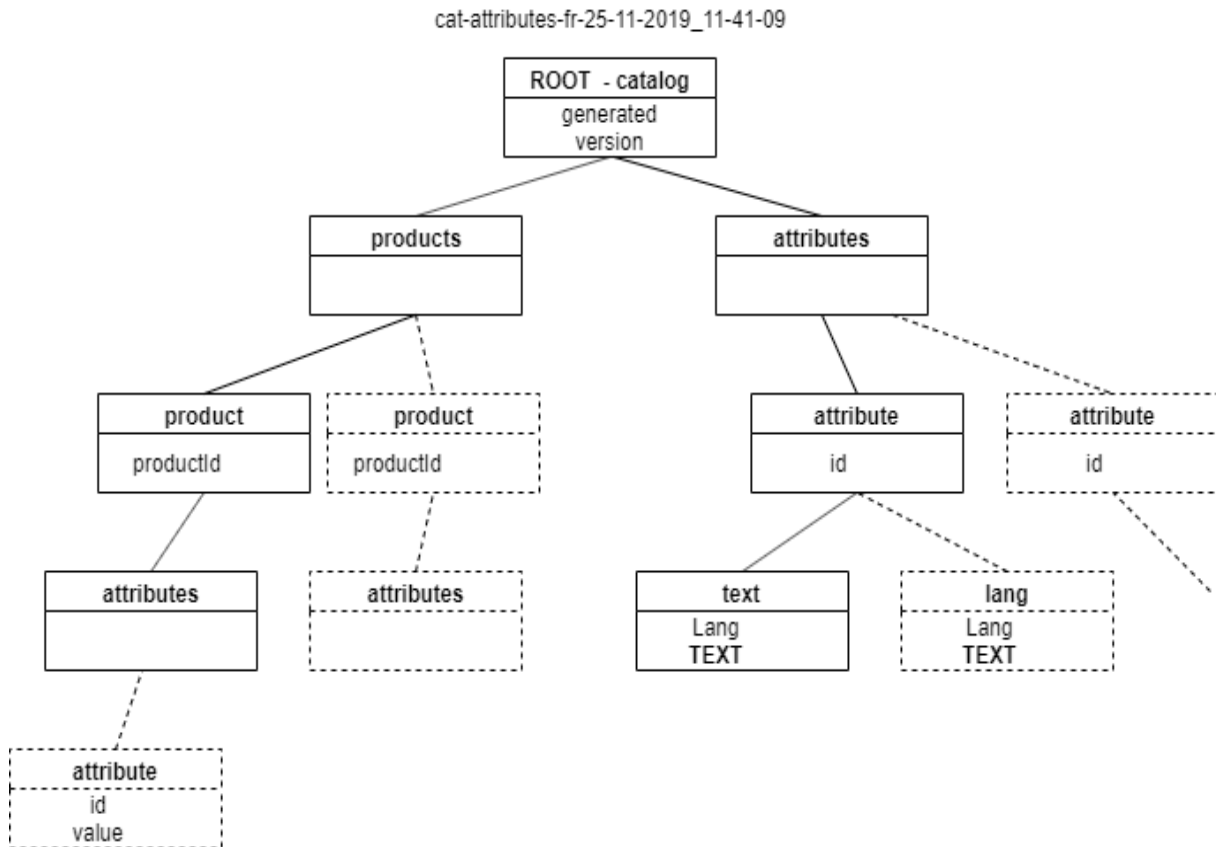
<price> comporte trois attributs permettant d'obtenir la ou les images du produit (productID, defaultdocumentID, isDefault)

La documentation précise que defaultDocumentID doit être préfixé de l'adresse fourni par l'API api/v2/Catalog/productsrooturl/ et suffixé par « .jpeg »

Par la suite, nous constaterons que le suffixe à utiliser est « .jpg »

7.6 Analyse du fichier cat-attributes

Une difficulté supplémentaire apparaît : <ROOT> comportent deux enfants <products> et <attributes>.



Dans la branche <products>, il nous faut récupérer productId, id et value.

Dans la branche <attributes>, il nous faut récupérer id et TEXT pour lang = « fr ».

8 IMPLEMENTATION DE LA BASE DE DONNEES

8.1 Installation du serveur MYSQL

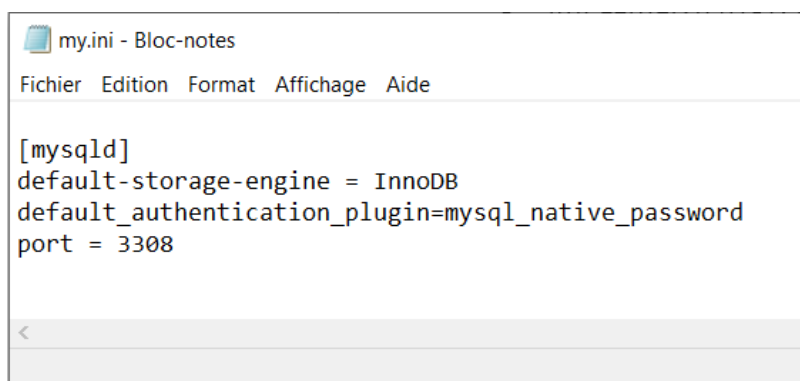
J'ai choisi d'utiliser l'environnement WAMP de WampServer pour installer MySQL sur mon poste. Je disposais déjà de MySQL Community Server sur mon poste de travail et j'aurais pu travailler directement dessus.

Les raisons qui m'ont poussées à faire ce choix sont :

- La possibilité d'utiliser phpMyAdmin qui m'est plus familier que WorkBench.
- La possibilité d'installer une copie du site SBIKE37.com si j'avais eu recours à des tests pour utiliser l'API REST de WooCommerce.
- La rapidité d'installation.
- Un environnement identique à celui proposé par OVH, l'hébergeur du site.

J'ai procédé à deux modifications sur le MySQL installé avec WampServer :

- Changement de port de 3306 vers 3308 de façon à ne pas générer de conflit avec la version de MySQL déjà installée.
- Modification dans le fichier my.ini du moteur de stockage par défaut. (Passage de mylsam vers InnoDB)



```
my.ini - Bloc-notes
Fichier Edition Format Affichage Aide

[mysqld]
default-storage-engine = InnoDB
default_authentication_plugin=mysql_native_password
port = 3308
```

La base bihr_db est créée à partir de phpMyAdmin avec un jeu de caractères UTF8 insensible à la casse (Espérant ainsi faciliter le moteur de recherche que je souhaite implémenter sous Flask).

Bases de données



8.2 Détermination des tables et dictionnaires

L'analyse des quatre fichiers XML vus dans le chapitre précédent et le document BIHR « CatalogContent » (en Annexe) m'ont permis de déterminer les six tables suivantes :

- cat-ref-FR3787ED

Table Category 1


#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires
1	categoryId 🔑	char(6)	utf8_general_ci		Non	Aucun(e)	
2	DESCRIPTION	varchar(250)	utf8_general_ci		Non	Aucun(e)	
3	PARENT	char(6)	utf8_general_ci		Oui	NULL	None ou category parent

Table Product

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires
1	barCode	varchar(255)	utf8_general_ci		Oui	NULL	
2	brandId	varchar(255)	utf8_general_ci		Oui	NULL	
3	discountClass	varchar(255)	utf8_general_ci		Oui	NULL	
4	endOfLifeProduct	tinyint(1)			Oui	NULL	
5	furtherDescription	varchar(255)	utf8_general_ci		Oui	NULL	
6	height	mediumint(9)			Oui	NULL	
7	ispartialshippingallowed	tinyint(1)			Oui	NULL	
8	isremainingonbackorderallowed	tinyint(1)			Oui	NULL	
9	length	mediumint(9)			Oui	NULL	
10	longDescription_1	varchar(255)	utf8_general_ci		Oui	NULL	
11	longDescription_2	varchar(255)	utf8_general_ci		Oui	NULL	
12	longDescription_3	varchar(255)	utf8_general_ci		Oui	NULL	
13	productId 🔑	varchar(255)	utf8_general_ci		Non	Aucun(e)	
14	publicPriceHT	decimal(10,2)			Oui	NULL	
15	publicPriceTTC	decimal(10,2)			Oui	NULL	
16	salesMultiple	tinyint(3)		UNSIGNED	Oui	NULL	
17	shortDescription_1	varchar(255)	utf8_general_ci		Oui	NULL	
18	shortDescription_2	varchar(255)	utf8_general_ci		Oui	NULL	
19	shortDescription_3	varchar(255)	utf8_general_ci		Oui	NULL	
20	volume	mediumint(9)			Oui	NULL	
21	weight	smallint(6)			Oui	NULL	
22	width	mediumint(9)			Oui	NULL	
23	parentProduct 🔑	char(6)	utf8_general_ci		Non	Aucun(e)	issu de l'élément category parent


- cat-images

Table Image

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires
1	defaultDocumentId	varchar(255)	utf8_general_ci		Oui	NULL	
2	urlImage	varchar(255)	utf8_general_ci		Oui	NULL	préfixe via api bihr
3	isDefault	tinyint(1)			Oui	NULL	
4	productId 	varchar(255)	utf8_general_ci		Non	Aucun(e)	

- cat-prices

Table Dealerprice

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires
1	dealerPriceHT	decimal(10,2)			Oui	NULL	
2	productId 	varchar(255)	utf8_general_ci		Oui	NULL	
3	discountRate	decimal(5,2)			Oui	NULL	

- cat-attributes

Table ProductAttribute : fork <products>

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires
1	productId 	varchar(255)	utf8_general_ci		Non	Aucun(e)	
2	attributes	json			Oui	NULL	Contient les attributs de chaque productId

Le json a le format suivant [{'id' : '...', 'value' : '...'}, {'id' : '...', 'value' : '...'}.....]

Table Attribute : fork <attributes>

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires
1	attributeID 	smallint(6)			Non	Aucun(e)	
2	textFr	varchar(255)	utf8_general_ci		Oui	NULL	

Remarque : Les types BOOLEAN sont remplacés en type tinyint(1) par MySQL , true prenant la valeur '1' et toute autre valeur prenant la valeur '0'. Il en sera tenu compte lors de la préparation des données sous Python.

8.3 Création des tables : CONNECT DB & CREATION DES TABLES .ipynb

Le lecteur trouvera en exemple ci-dessous, un extrait du code permettant la création de la table Image dans la base bihr_db.

La totalité du code est disponible en annexe.

- a. Import du connecteur mysql.connector et de la méthode Error

```
1 import mysql.connector
2 from mysql.connector import Error
```

- b. Création de la requête SQL que l'on range dans la variable ImageQuery

```
1 ImageQuery = """ CREATE TABLE Image (
2     defaultDocumentId VARCHAR(255),
3     urlImage VARCHAR(255),
4     isDefault BOOLEAN,
5     productId VARCHAR(255) NOT NULL,
6     FOREIGN KEY (productId) REFERENCES Product(productId)
7 ) """
```

- c. On procède à la connexion et on envoie notre requête qui est exécutée. Si une erreur survient, Error l'intercepte et l'affiche. Au terme de l'exécution, le curseur et la connexion sont fermés.

```
1 try:
2     connection_config = {
3         'host': "localhost",
4         'port': 3308,
5         'database': 'bihr_db',
6         'user': 'BASTIER',
7         'passwd': "DA2019",
8         'autocommit': True
9     }
10
11     connection = mysql.connector.connect(**connection_config)
12
13     cursor = connection.cursor()
14     result = cursor.execute(ImageQuery)
15     print("Image table created successfully ")
16
17 except mysql.connector.Error as error:
18     print("Failed to create table in MySQL: {}".format(error))
19 finally:
20     if (connection.is_connected()):
21         cursor.close()
22         connection.close()
23         print("MySQL connection is closed")
```

Image table created successfully
MySQL connection is closed

8.4 Modèle conceptuel de données réalisé avec phpMyAdmin

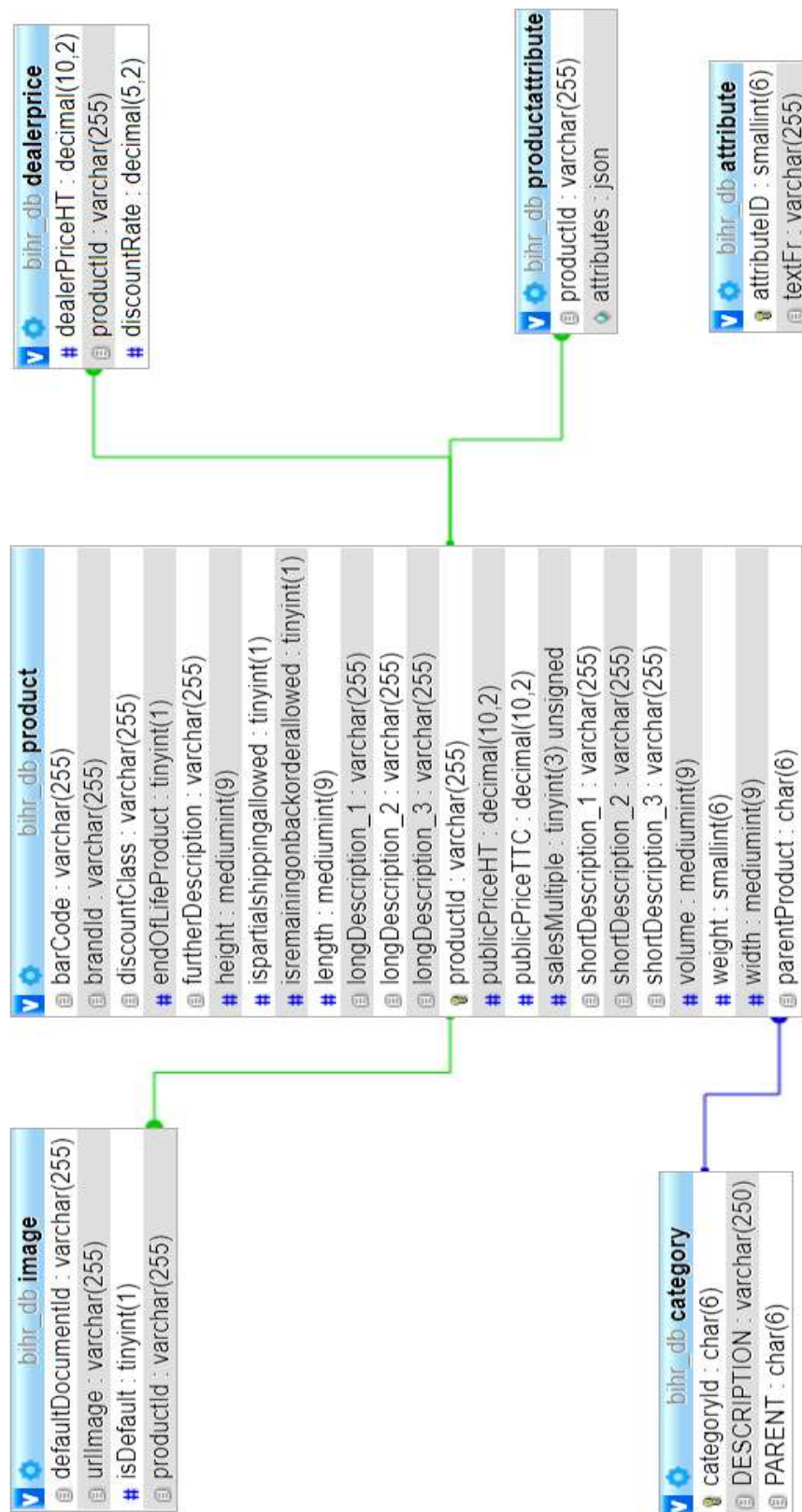
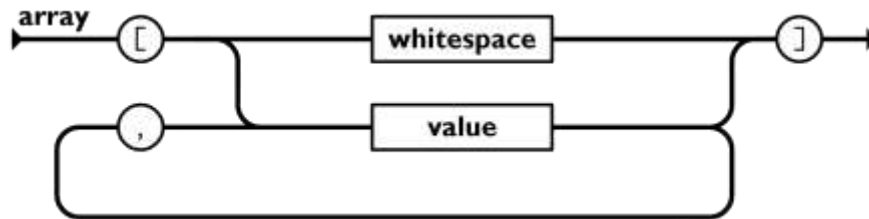


Figure 5 Modèle conceptuel de données - bihr_db

8.5 Le MCD fait apparaître une erreur de conception

A la lecture du MCD, je m'aperçois qu'il m'est impossible de relier la table Attribute à la table ProductAttribute avec une FOREIGN KEY. La Table Attribute est donc orpheline.

La colonne attributes de la table ProductAttribute est au format JSON. Les données y sont rangées avec le format json suivant : une liste de dictionnaires.



La clé id figurant dans ce dictionnaire ne peut pas être liée avec la colonne attributeID de la table attribute

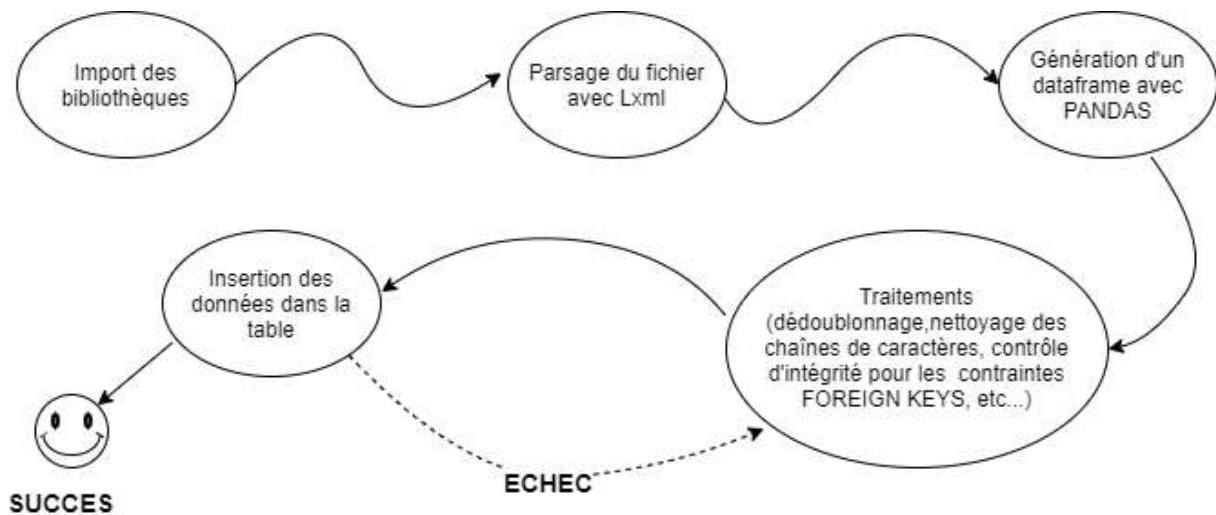
Après réflexion, je pense qu'il eut été mieux de créer une seule table en utilisant une clé primaire composite constituée du productID et l'id de l'attribut suivant le tableau suivant :

NOM	TYPE	Clé	
ProductID	VARCHAR(255)	FOREIGN KEY vers table Product COMPOSITE PRIMARY KEY	Issu de la branche <products>
Value	VARCHAR(255)	COMPOSITE PRIMARY KEY	Issu de la branche <products>
Textfr	VARCHAR(255)		Issu de la branche <attributes>

Préalablement à l'insertion dans la base, lors du traitement des données sous Python, nous génèrons deux dataframes avec PANDAS que l'on joint avec l'attribut id.

9 TRAITEMENTS DES FICHIERS ET INSERTION DES DATAS DANS LA BASE

Les six fichiers Python créés pour le passage des XML, la récupération des données, les traitements avec PANDAS et l'insertion des données dans les tables ont tous la même structure.



Liste des six fichiers python réalisés avec Jupyter Notebook :

- INSERT_ATTRIBUTE_DATA.ipynb
- INSERT_CATEGORY_DATA.ipynb
- INSERT_DEALERPRICE_DATA.ipynb
- INSERT_IMAGE_DATA.ipynb
- INSERT_PRODUCT_DATA.ipynb
- INSERT_PRODUCT_ATTRIBUTE_DATA.ipynb

Plutôt que d'imposer au lecteur une litanie de code pour chaque fichier (disponibles dans les annexes), je propose d'observer le cas général pour chaque élément de la structure. Les variations dues aux spécificités des fichiers XML récupérés seront ajoutées en commentaire.

9.1 Import des bibliothèques Python

```
1 from lxml import etree
2 import pandas as pd
3 import mysql.connector
4 from mysql.connector import Error
5 import re
```

LXML nous permettra de parser et de parcourir les fichiers grâce à ces méthodes `findall()` et `iter()`. Je n'ai pas retenu la librairie native de Python car Lxml possède une méthode `getparent()` qui sera utilisée dans le fichier `INSERT_CATEGORY_DATA.ipynb`. De plus, le parsing avec lxml est très simple.

La librairie PANDAS est le “super-excel ” qui permettra le nettoyage des données.

La librairie mysql.connecteur est le connecteur fourni par Oracle pour se connecter à une base MySQL.

La librairie RE me permet de nettoyer rapidement des chaînes de caractères lorsqu’une erreur apparaît lors de la tentative d’insertion des données dans la table concernée.

Dans le fichier INSERT_IMAGE_DATA, j’ai utilisé les librairies REQUESTS et JSON car il était nécessaire de générer le préfixe de l’URL des images avec la Web API [GET api/v2/Catalog/productsrooturl/](#)

9.2 Parsage du fichier avec Lxml

Identique dans chaque fichier, seul le fichier à parser est modifié.

```
1 # à modifier avant chaque traitement d'un nouveau fichier XML
2 refPath = 'unzipped_files/cat-images-1.1.2141226-11-2019'
3
4 xtree = etree.parse(refPath)
5 xroot = xtree.getroot()
```

9.3 Génération du dataframe Pandas

A la différence du parsage, la génération du dataframe diffère d’un fichier à l’autre.

Voyons d’abord le cas le plus simple :

```
1 df_cols = ["productId", "defaultDocumentId", "isDefault"]
2 rows = []
3
4 for i,ec in enumerate(xroot.iter('product')):
5     # ajout à la liste du dict décrivant l'élément category
6     rows.append({"productId": ec.attrib['productId'],
7                 "defaultDocumentId": ec.attrib['defaultDocumentId'],
8                 "isDefault":ec.attrib['isDefault']})
9 # transformation de la liste en dataframe
10 dfImage = pd.DataFrame(rows, columns=df_cols)
11 dfImage
```

	productId	defaultDocumentId	isDefault
0	0040038	00/00000000000000000000000000000000	false
1	006MS036	00/00000000000000000000000000000000	false
2	00A-B8120-92E	00/00000000000000000000000000000000	false
3	00A-B8240-12B	00/00000000000000000000000000000000	false
4	00A-B8240-12K	00/00000000000000000000000000000000	false

Nous parcourons le chemin amenant à la balise <product> avec xroot.iter, nous rangeons ses attributs dans un dictionnaire que nous ajoutons à la liste rows. Nous réitérons cette méthode pour chaque balise <product> grâce à une boucle for sur enumerate(xroot.iter(product))

Nous créons ensuite le dataframe avec les éléments de la liste rows et les noms de colonnes rangés dans la liste df_cols.

Un peu moins simple :

```
1 for products in xroot.iter('products'):
2     for a,ec in enumerate(products.getchildren()):
3         products.getchildren()[a].attrib['cat_parent'] = str(products.getparent().get('code'))
4
```

extrait de INSERT_PRODUCT_DATA

Pour chaque balise <products>, nous allons chercher tous ses enfants <product> pour récupérer leurs attributs et nous attrapons l'attribut 'code' de la balise parent <category> (possible grâce à lxml).

Avec xroot.findall et la création du fichier json:

```
1 for att in xroot.findall('products'):
2     #print(att.tag)
3     for i,att1 in enumerate(att.getchildren()):
4         #print(att1.attrib['productId'])
5         b = att1.getchildren()
6         for i,att2 in enumerate(b):
7             c = att2.getchildren()
8             row = []
9             for i,y in enumerate(c):
10                # row = []
11                row.append({"id": y.attrib['id'], "value": y.attrib['value']})
12            row_json = json.dumps(row, ensure_ascii=False)
13            rows.append({"productId": str(att1.attrib['productId']),
14                        "attributes": (row_json)})
15
```

extrait de INSERT_PRODUCT_ATTRIBUTE_DATA 1

Xroot.findall('products') permet de trouver tous les enfants de la branche <products>. Nous récupérons l'attribut productId de <product> et nous rassemblons les différents attributs de chaque balise <attribut> dans un dictionnaire. (Cf. schéma du paragraphe 7.6)

9.4 Les traitements effectués sous Pandas

Les traitements effectués avec Pandas sont principalement :

- a. La recherche de doublons sur les futures clés primaires : (exemple issu de INSERT_PRODUCT_DATA)

Nous constatons la présence de 73 doublons au plus

```
1 # dfProduct comporte 185240 lignes
2 resultat = dfProduct.groupby('productId').nunique()
3 resultat.shape
```

(185167, 23)

Recherche des doublons

```
1 dfDoublon = dfProduct.groupby(['productId'])['cat_parent'] \
2             .count() \
3             .reset_index(name='count') \
4             .sort_values(['count'], ascending=False)
5
```

Nous les isolons

```
dfDoublon = dfDoublon.set_index("count")
dfDoublon = dfDoublon.drop(1, axis=0)
```

Nous les supprimons de dfProduct

```
1 dfProduct = dfProduct.set_index("productId")
2
3 for i in enumerate(dfDoublon['productId']):
4     dfProduct = dfProduct.drop(i[1], axis=0)
5
6 dfProduct.shape
```

(185095, 22)

- b. Le remplacement de la valeur 'true' par '1' pour les types BOOLEAN (exemple issu de INSERT_IMAGE_DATA)

```

1 dfImage.isDefault.replace("true","1", regex=True, inplace=True)
2 dfImage

```

	productId	defaultDocumentId	isDefault
0	0040038	00/00000000000000000000000000000000	false
1	006MS036	00/00000000000000000000000000000000	false
2	00A-B8120-92E	00/00000000000000000000000000000000	false
3	00A-B8240-12B	00/00000000000000000000000000000000	false
4	00A-B8240-12K	00/00000000000000000000000000000000	false

- c. La vérification de la présence d'éléments pour une FOREIGN Key (exemple issu de INSERT_IMAGE_DATA)

Rapprochement des productID pour la FOREIGN KEY

```

1 # à modifier avant chaque traitement d'un nouveau fichier XML
2 refPath1 = 'unzipped_files/cat-ref-FR3787ED_2019-11-22 11.30.32-22-11-2019_11-48-22'
3
4 xtree1 = etree.parse(refPath1)
5 xroot1 = xtree1.getroot()

```

```

1 df_cols1 = ["productId"]
2 rows1 = []
3 for products in xroot1.iter('products'):
4     for a,ec in enumerate(products.getchildren()):
5         rows1.append(ec.attrib['productId'])
6 print(len(rows1))

```

185240

Suppression des 72 productId qui étaient doubles sur le fichier REF

#####(TODO : Gérer les doublons REF-Product pour supprimer le traitement ci-dessous)#####

```

1 rows2 = []
2 doublons = []
3 for i in rows1:
4     if i not in rows2:
5         rows2.append(i)
6     else:
7         doublons.append(i)
8 print(len(rows2))
9 print(len(doublons))

```

185167

73

```

1 # solution 2 :
2 for i in doublons:
3     if i in rows2:
4         rows2.remove(i)
5 print(len(rows2))

```

185095


```

1 integrite = pd.Series(dfImage['productId'].isin(rows2))
2 integrite

```

```

0      True
1      True
2      True
3     False
4     False

```

```

1 #Insertion de la série integrite dans dfImage
2 dfImage['Integrité'] = integrite
3 dfImage

```

	productId	defaultDocumentId	IsDefault	Integrité
0	0040038	00/00000000000000000000000000000000	false	True
1	006MS038	00/00000000000000000000000000000000	false	True
2	00A-B8120-92E	00/00000000000000000000000000000000	false	True
3	00A-B8240-12B	00/00000000000000000000000000000000	false	False
4	00A-B8240-12K	00/00000000000000000000000000000000	false	False
...
264349	ZV9550	db/dbea38050d8f49fd928ecc859dcfcc66	1	True
264350	ZV9600	36/36412b4c920d487ea3f596ad3b7a8b2b	1	True

La colonne Integrité que l'on rajoute au dataframe dfImage permettra avec une simple boucle if de filtrer les lignes à insérer lors de la requête d'insertion SQL.

9.5 L'insertion des données du dataframe dans la table

L'insertion des données dans la table ne varie pas d'un fichier à l'autre : seule une boucle if pourra être faite si l'on doit insérer les données dans une table comportant une FOREIGN KEY.

```

1 connection_config = {
2     'host': 'localhost',
3     'port': 3308,
4     'database': 'bihr_db',
5     'user': 'BASTIER',
6     'passwd': 'DA2019',
7     '#autocommit': True
8 }

```

```

1  try:
2      connection = mysql.connector.connect(**connection_config)
3
4      for i in range(dfImage.shape[0]):
5          img = dfImage.iloc[i]
6          if img['Integrité'] == True:
7              URL_image = rootImage + "/" + str(img['defaultDocumentId']) + ".jpg"
8              print(i)
9              print(URL_image)
10             imageInsertQuery = """INSERT INTO Image (productId, defaultDocumentId, urlImage, isDefault)
11                                 VALUES
12                                 (""" + str(img['productId']) + """, """ + str(img['defaultDocumentId']) + """,
13                                 """ + URL_image + """, """ + str(img['isDefault']) + """)"""
14             #print(imageInsertQuery)
15             cursor = connection.cursor()
16             result = cursor.execute(imageInsertQuery)
17         else:
18             continue
19
20     connection.commit()
21     print("Insertion datas in Image table successfully ")
22     cursor.close()
23
24 except mysql.connector.Error as error:
25     print("Failed to insert datas in Image table : {}".format(error))
26
27 finally:
28     if (connection.is_connected()):
29         cursor.close()
30         connection.close()
31     print("MySQL connection is closed")

```

<http://static.bihir.pro/v2/xlarge/48/4898j78cbe47ad8ca5b8e7ca20beabde.jpg>
264344
<http://static.bihir.pro/v2/xlarge/8a/8af8a97cbd994ad782678a495289c098.jpg>
264345
<http://static.bihir.pro/v2/xlarge/84/84f6be78532faa90a20ca5ac4e7e9a3c.jpg>
264347
<http://static.bihir.pro/v2/xlarge/1a/1aab87b992b04fe3bfe9c914b09905q7.jpg>
.....

10 LIVRABLE : IMPLEMENTATION DU DUMP DE LA BASE BIHR_DB

Je me place dans la console MySQL et je rentre ma commande pour créer le dump. J'obtiens une erreur.

```
d:\wamp64\bin\mysql\mysql5.7.26\bin\mysql.exe
mysql> mysqldump -u BASTIER -p DA2019 bihr_db > bihr_db_dump1.sql;
ERROR 1064 (42000): Erreur de syntaxe près de 'mysqldump -u BASTIER -p DA2019 bihr_db > bihr_db_dump1.sql' à la ligne 1
mysql>
```

En effet, mysqldump n'est pas une commande SQL mais un exécutable devant être lancé à partir de l'invite de commande.

```
Invite de commandes
D:\wamp64\bin\mysql\mysql5.7.26\bin>mysqldump -u BASTIER -pDA2019 bihr_db > bihr_dump.sql
mysqldump: [Warning] Using a password on the command line interface can be insecure.
D:\wamp64\bin\mysql\mysql5.7.26\bin>
```

Un warning de sécurité apparaît mais le dump est créé. Il ne me reste plus qu'à le transmettre à W3P avec le MCD. W3P pourra l'importer dans la base de son choix.

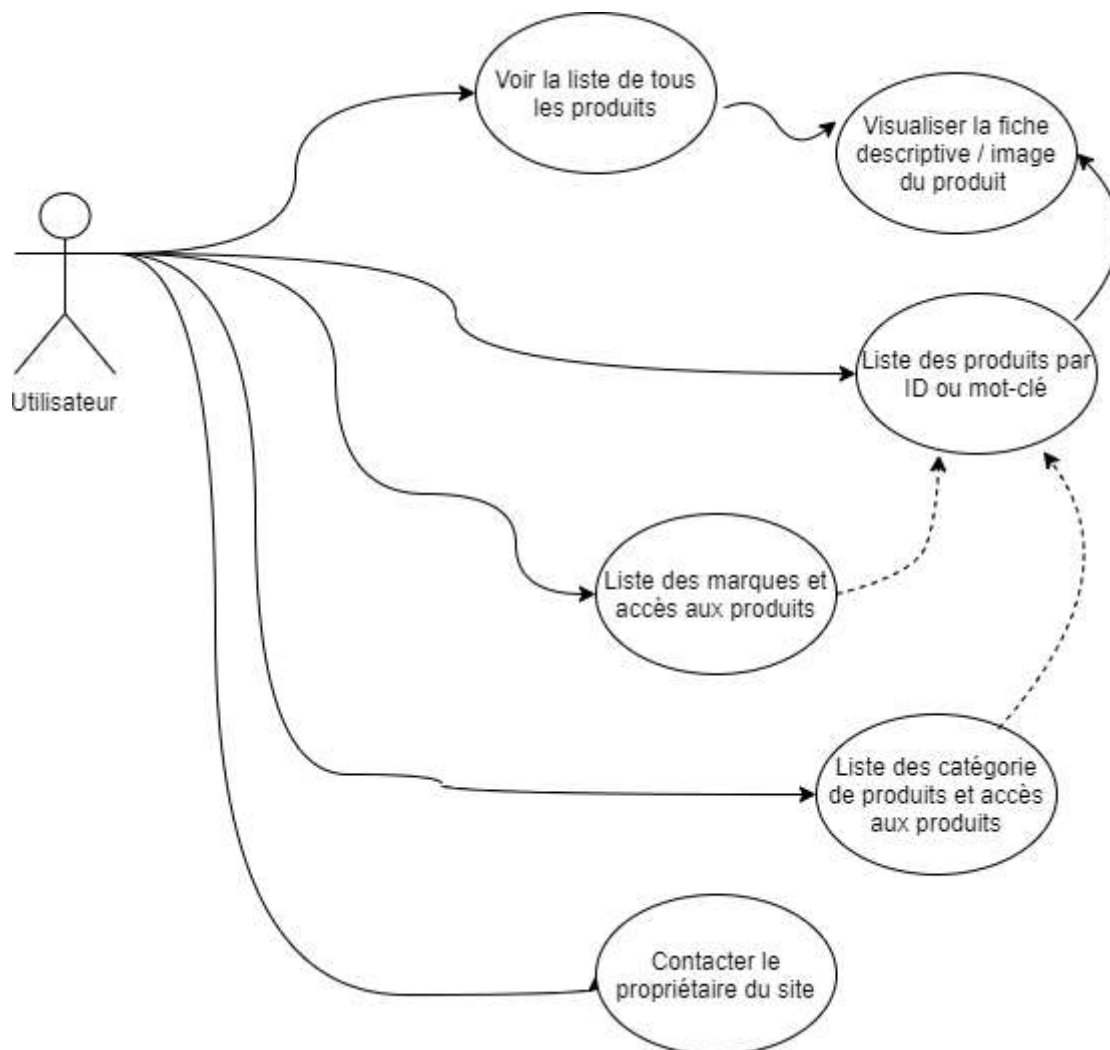
 bihr_dump.sql 08/12/2019 11:15 SQL Text File 95 991 Ko

```
D:\wamp64\bin\mysql\mysql5.7.26\bin\bihr_dump.sql - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window 2
bihr_dump.sql
1  -- MySQL-dump 10.13 Distrib 5.7.26, for Win64 (x86_64)
2  --
3  -- Host: localhost    Database: bihr_db
4  --
5  -- Server version: 5.7.26
6  --
7  /*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
8  /*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
9  /*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
10 /*!40101 SET NAMES utf8 */;
11 /*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
12 /*!40103 SET TIME_ZONE='+00:00' */;
13 /*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
14 /*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0 */;
15 /*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
16 /*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;
17
18 --
19 -- Table structure for table `attribute`
20 --
21
22 DROP TABLE IF EXISTS `attribute`;
23 /*!40101 SET @saved_cs_client = @@character_set_client */;
24 /*!40101 SET character_set_client = utf8 */;
25 CREATE TABLE `attribute` (
26   `attributeID` smallint(4) NOT NULL,
27   `textFr` varchar(255) DEFAULT NULL,
28   PRIMARY KEY (`attributeID`)
29 ) ENGINE=InnoDB DEFAULT CHARSET=utf8;
30 /*!40101 SET character_set_client = @saved_cs_client */;
31
32 --
33 -- Dumping data for table `attribute`
34 --
35
36 LOCK TABLES `attribute` WRITE;
37 /*!40000 ALTER TABLE `attribute` DISABLE KEYS */;
38 INSERT INTO `attribute` VALUES (1,'Nombre de dents pignon (kit)');
39 /*!40000 ALTER TABLE `attribute` ENABLE KEYS */;
40 UNLOCK TABLES;
```

11 VISUALISATION DES DONNEES AVEC FLASK

Ce projet de visualisation des données sous Flask vient en marge du projet principal visant à livrer une base de données complète à un sous-traitant.

11.1 Fonctionnalités attendues



11.2 Maquettage du site

Les maquettes ont été réalisées sur le service en ligne : <http://framebox.org/>

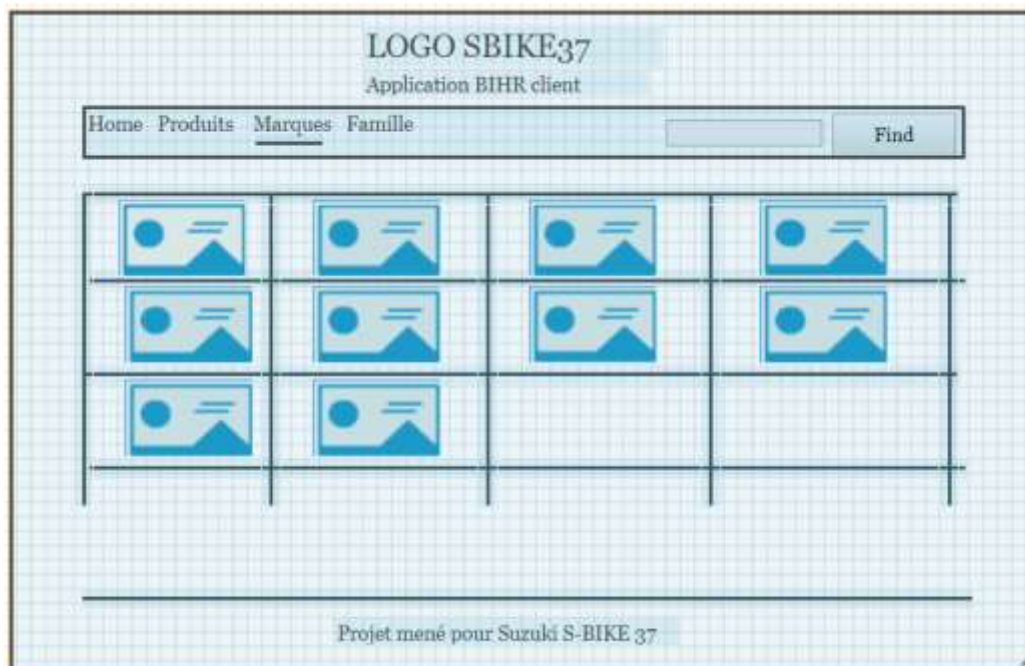
Page d'accueil



Page liste produits



Page liste marques

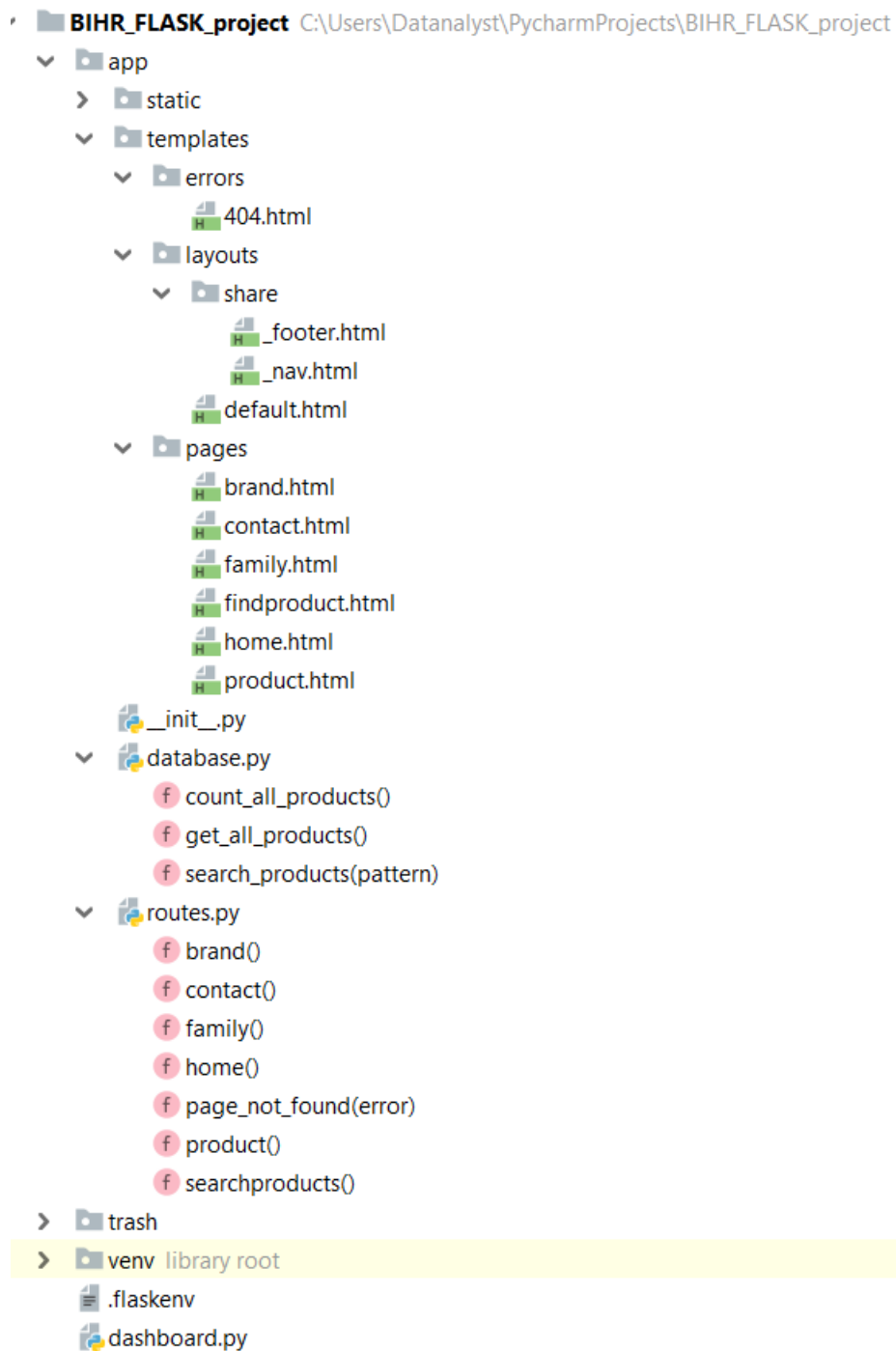


Page fiche produit



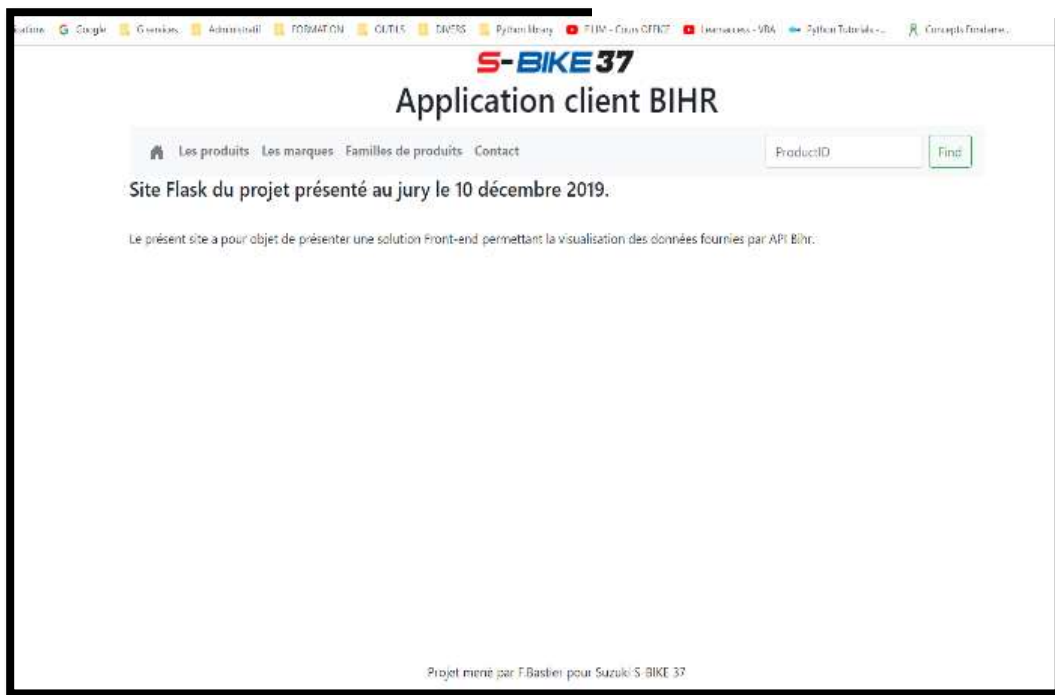
11.3 Implémentation de site sous FLASK.

Travaillant avec PyCharm sur ce projet, j'ai réalisé le travail dont vous pouvez voir l'arborescence ci-dessous.

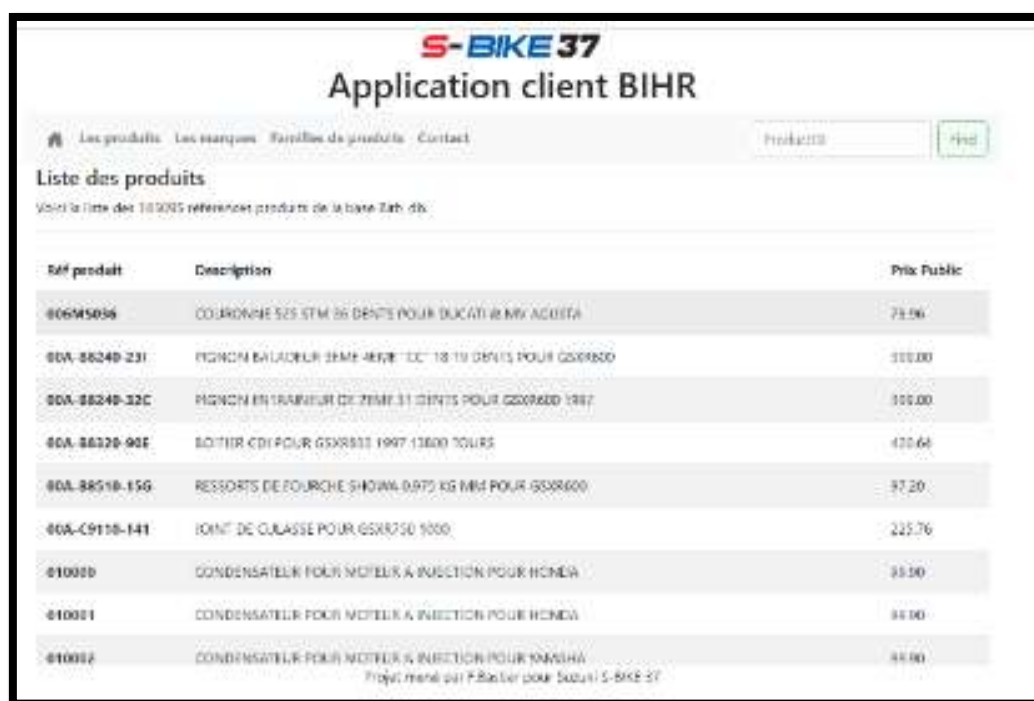


Le lecteur trouvera sur la page suivante l'illustration des pages réalisées et mises en forme avec Bootstrap.

Home



Product



contact

S-BIKE37
Application client BIHR

[Les produits](#) [Les marques](#) [Familles de produits](#) [Contact](#)

Pour nous contacter :

S-BIKE 37
1, rue Pierre de Coubertin
37540 Saint Cyr sur Loire
Tel: (+33) 02 47 54 64 66

Magasin
magasin@sbike37.com

Projet mené par F.Bastier pour Suzuki S-BIKE 37

Home

S-BIKE37
Application client BIHR

[Les produits](#) [Les marques](#) [Familles de produits](#) [Contact](#)

Recherche produits
Résultat de la recherche : 1 résultat(s)

Réf produit	Description	Prix Public
00A-B8320-90E	BOITIER CDI POUR GSXR600 1997 13800 TOURS	420.64

Projet mené par F.Bastier pour Suzuki S-BIKE 37

Le petit moteur de recherche a été implémenté : (ci-dessous les extraits de code de la fonction dans database.py et la route avec le requêtage dans routes.py)

```
database.py routes.py
37
38
39 def search_products(pattern):
40     db = sql.Connect(
41         host="localhost",
42         user="BASTIER",
43         passwd="DA2019",
44         database="bihr_db",
45         port=3308
46     )
47     cursor = db.cursor(dictionary=True)
48     pattern = f"%{pattern}%"
49     cursor.execute(
50         "SELECT productId, longDescription_1, publicPriceTTC FROM product "
51         "WHERE productId LIKE %s OR longDescription_1 LIKE %s:", (pattern, pattern)
52     )
53     listofproducts = cursor.fetchall()
54     db.close()
55     return listofproducts
56
```

```
database.py routes.py
37
38 @app.route('/findproduct', methods=['GET', 'POST'])
39 def searchproducts():
40     s = ''
41     if request.method == 'POST':
42         # return "Vous avez envoyé : {name1}".format(name1=request.form['recherche'])
43         s = request.form['recherche']
44     pattern = s
45     #print(pattern)
46     findproduct_list = search_products(pattern)
47     count_result = len(findproduct_list)
48     print(count_result)
49     return render_template("pages/findproduct.html", findproducts=findproduct_list, countresult=count_result)
50
```

11.4 TODO

Le développement du site n'est malheureusement arrivé à son terme pour l'échéance prévue.

La page Brand soulève le problème de récupérer les images des marques avec la Web API qui a été modifiée récemment

La page Fiche produit pourra se faire avec l'utilisation des flexbox BootStrap pour la mise en page et nécessite une requête SQL un peu plus complexe que ce qui vous a été présenté dans ce rapport.

12 CONCLUSIONS

Sur le respect du cahier des charges

Je considère avoir réussi à fournir le livrable demandé dans les délais impartis. Je me suis trouvé dépassé pour la réalisation du site sous FLASK et je n'ai pas réussi à la mener à son terme.

Sur la gestion de projet

J'aurais souhaité avoir plus de temps pour finir le projet en entier, j'ai beaucoup apprécié le fait de pouvoir le suivre de la demande initiale jusqu'à la fourniture du livrable.

Dans l'ensemble, j'ai perdu du temps sur :

- L'apprentissage des technologies : Chacun des outils utilisés furent à apprendre ou à approfondir.
- L'analyse des fichiers XML : La compréhension des schémas m'a pris beaucoup de temps
- La découverte de l'API : De nombreux tests ont été nécessaires pour saisir le fonctionnement malgré une documentation accessible et bien construite.

Pour conclure

Le travail qui a été mené durant ces cinq semaines m'a réconcilié avec Python et ses librairies. A défaut d'avoir pu mener à bout la totalité du projet, l'intérêt que je porte à la préparation des jeux de données et les solutions de Data Management est renforcé.

13 REMERCIEMENTS

Tout d'abord, je souhaite remercier le CEFIM et la Région Centre qui m'ont offert l'opportunité de suivre cette formation de neuf mois dans d'excellentes conditions. La prise en charge des contraintes administratives et financières fut une réelle aide pour suivre sereinement le programme.

Je remercie Jean Lou LEBARS et les différents intervenants pour leurs compétences, leur patience, leur gentillesse et leur disponibilité.

Je remercie Olivier FALCON et son équipe pour leur excellent accueil durant les dix semaines de stage et la confiance qu'ils m'ont accordé.

Enfin, je veux remercier mes onze co-apprenants qui m'ont toujours insufflé l'énergie et la motivation lorsque cela était nécessaire, ils m'ont beaucoup appris durant cette année passée ensemble.

14 RESSOURCES

Les principales librairies utilisées :

Lxml : <https://lxml.de/index.html>

Request : <https://requests-fr.readthedocs.io/>

Pandas : <https://pandas.pydata.org/>

Pyunpack : <https://pyunpack.readthedocs.io/>

Python Driver for MySQL : <https://www.mysql.com/fr/products/connector/>

Les IDEs :

Pycharm : <https://www.jetbrains.com/fr-fr/pycharm/>

Jupyter Notebook : <https://jupyter.org/>

Framework et librairie CSS :

FLASK : <https://flask.palletsprojects.com/en/1.1.x/>

Bootstrap : <https://getbootstrap.com/>

Plateforme de développement et SGBDR :

MySQL sous WAMP : <http://www.wampserver.com/>

Outils divers :

XML viewer : <https://www.mindfusion.eu/xml-viewer.html>

Veille et documentation :

- <https://docs.python.org/fr/3.7/index.html>
- <https://www.json.org/json-fr.html>
- <https://www.w3schools.com/python/>
- <https://datatofish.com/python-tutorials/>
- <https://blog.miguelgrinberg.com/post/the-flask-mega-tutorial-part-i-hello-world>
- <https://curl.trillworks.com/>
- <https://developer.mozilla.org/fr/docs/Web/XPath>
- <https://sql.sh/>
- <https://pynative.com/python-mysql-execute-stored-procedure/>
- Tutoriel Flask : <https://youtu.be/ajrfDEi8F7Y>
- <https://stackoverflow.com/>

15 ANNEXES

- I. CatalogContent BIHR
- II. Schéma XSD exemple
- III. API_BIHR_CoDoUn
- IV. CONNECT DB & CREATION DES TABLES
- V. INSERT_CATEGORY_DATA
- VI. INSERT_PRODUCT_DATA
- VII. INSERT_DEALERPRICE_DATA
- VIII. INSERT_IMAGE_DATA
- IX. INSERT_ATTRIBUTE_DATA
- X. INSERT_PRODUCT_ATTRIBUTE_DATA