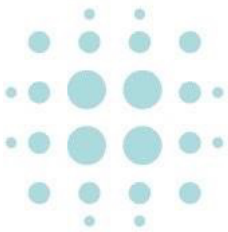
Hes·so

MSE | MASTER OF SCIENCE IN ENGINEERING

Projet Web Mining : CyberCraft

Cahier des charges



CYBERCRAFT

Antony Carrard – Killian Vervelle

Table des matières

1 Contexte

Il existe à ce jour plusieurs sites de configuration sur mesure d'ordinateurs et de benchmark de leurs composants. Les données sur lesquelles reposent ces services ne sont pas toujours mises à jour en temps réel ce qui va à l'encontre même de la pertinence de leurs recommandations aux utilisateurs. Notre microservice permettra de répondre aux mêmes besoins utilisateur de manière centralisée et au travers d'une interface utilisateur simplifiée, tout en garantissant l'exactitude et l'exhaustivité des données renvoyées à l'utilisateur (compatibilité technique, prix actuels...).

2 Objectifs

L'objectif du projet est de développer un microservice capable, en toute autonomie, de configurer un ordinateur :

- Répondant aux contraintes utilisateurs ;
- Reposant sur les dernières technologies de composants disponibles sur le marché et au meilleur prix ;
- Proposant les meilleurs prix des composants du jour en Suisse ;
- Respectant les normes de comptabilité technique.

Le but est de proposer un configurateur d'ordinateur rapide et paramétrable, sans dépasser le budget de l'utilisateur.

3 Périmètre

Le microservice portera principalement sur la configuration d'ordinateurs prévus à des utilisations gourmandes en ressources telles que le gaming, la modélisation et rendu 3d, le minage de cryptos...

4 Données

4.1 Source de données

Nos données proviendront de 3 sites :

- www.userbenchmark.ch (référence pour les essais de performance de composants d'ordinateur)
- www.ldlc.ch (site leader dans la vente de composants d'ordinateurs)
- www.toppreise.ch (site leader dans le benchmarking des meilleures offres fournisseurs pour des composants électroniques)

4.2 Droits d'utilisation

Les conditions d'utilisation des 3 sites, à partir desquels nous scrapperons nos données, semblent contenir une clause interdisant toute diffusion de leurs données à des fins commerciales. Notre projet sera conduit dans un cadre purement universitaire et ne violera en aucun cas cette condition.

4.3 Description (attributs, quantité)

Notre microservice fonctionnera à partir de 3 csv :

- Données composants principaux (GPU, CPU, mémoire et SSD) : csv fournit par le site www.userbenchmark.ch. Il contiendra les données de performance (score de benchmark) de ces 4 composants.
  Attributs: Type, Part Number, Brand, Model, Rank, Benchmark, Samples, URL
- Données composants principaux et secondaires : csv qui résultera du processus de scrapping du site www.ldlc.ch. Il contiendra diverses données sur les dernières technologies de composants disponibles sur le marché.
  Attributs: name, model, compatiblity_list, price
- Données fournisseurs de composants : csv qui résultera du processus de scrapping sur le site www.toppreise.ch. Il contiendra les offres ainsi que les URL des fournisseurs de chaque composant
  Attributs: Type, Part Number, Brand, Model, Rank, Benchmark, Samples, URL

4.4 Extraction (méthodes)

Une grande partie de nos données sera extraite de pages HTML à l'aide d'un web crawler (Scrappy). Afin de garantir un accès sécurisé et continu aux données de ces sites, en évitant de se faire « black-lister », nous paramètrerons un délai de requête de 4 secondes afin de simuler un comportement humain. De plus, nous masquerons notre adresse IP à l'aide d'un proxy fournit par le site https://scrapeops.io .

```
process = CrawlerProcess(settings={
        "FEEDS": {
            #"mb.csv": {"format": "csv"},
            #"rad.csv": {"format": "csv"},
            #"power.csv": {"format": "csv"},
            #"case.csv": {"format": "csv"},
            #"gpu2.csv": {"format": "csv"},
            #"cpu2.csv": {"format": "csv"},
            "ram2.csv": {"format": "csv"},
            #"ssd2.csv": {"format": "csv"}
        },
        "DOWNLOAD_DELAY" : 4
    })
```

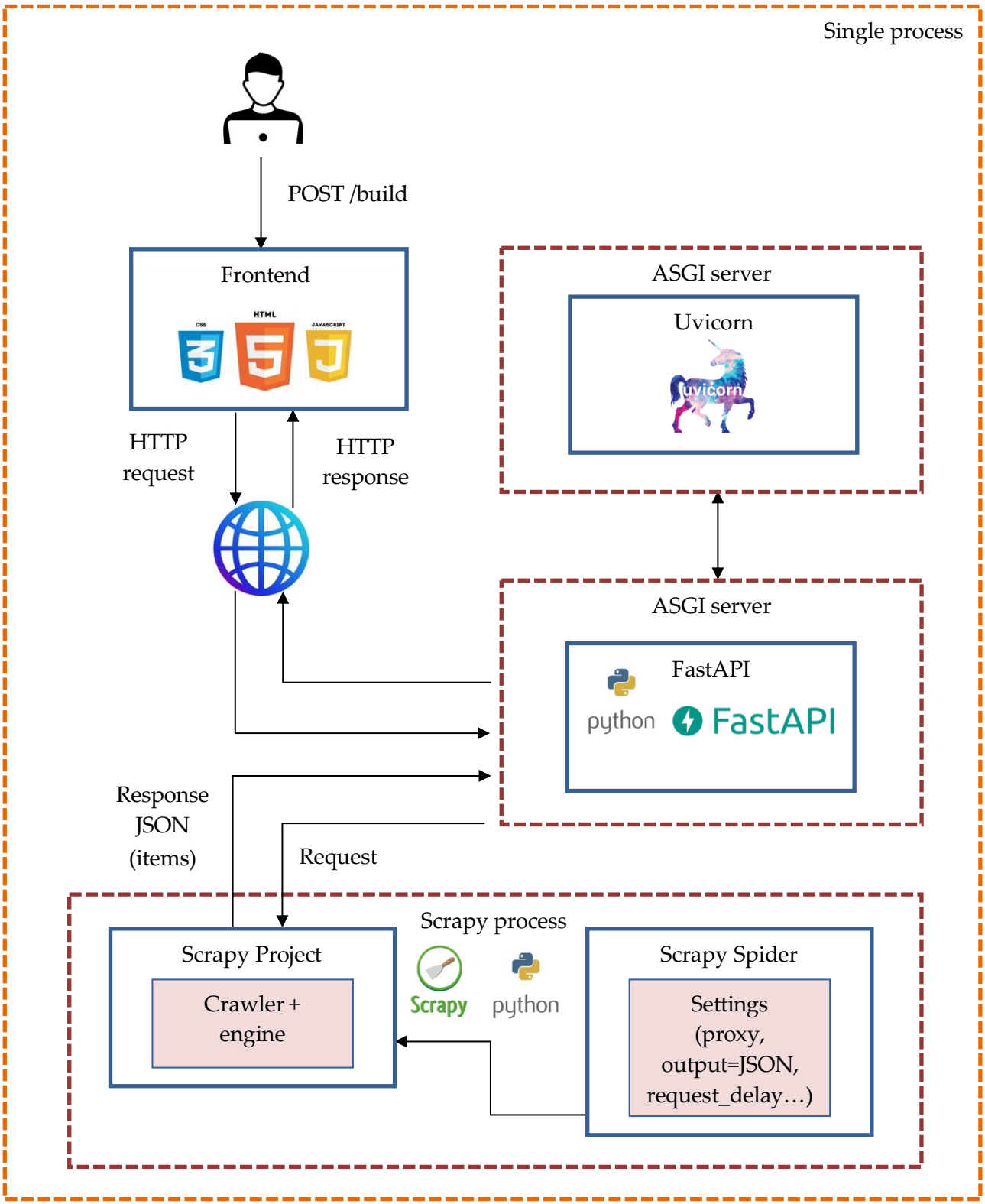
```
# Crawler Scrapy
class QuotesSpider(scrapy.Spider):
    name = "spider"
    API_KEY = "4f4f5166-b931-4ffd-bb24-4bdf93ced0a5"
    SCRAPEOPS_PROXY_ENABLED = True
    download_delay = random.randint(4, 10)
    LOG_LEVEL='DEBUG'

    def get_scrapeops_url(self, url):
        payload = {'api_key': self.API_KEY, 'url': url}
        proxy_url = 'https://proxy.scrapeops.io/v1/?' + urlencode(payload)
        return proxy_url
```

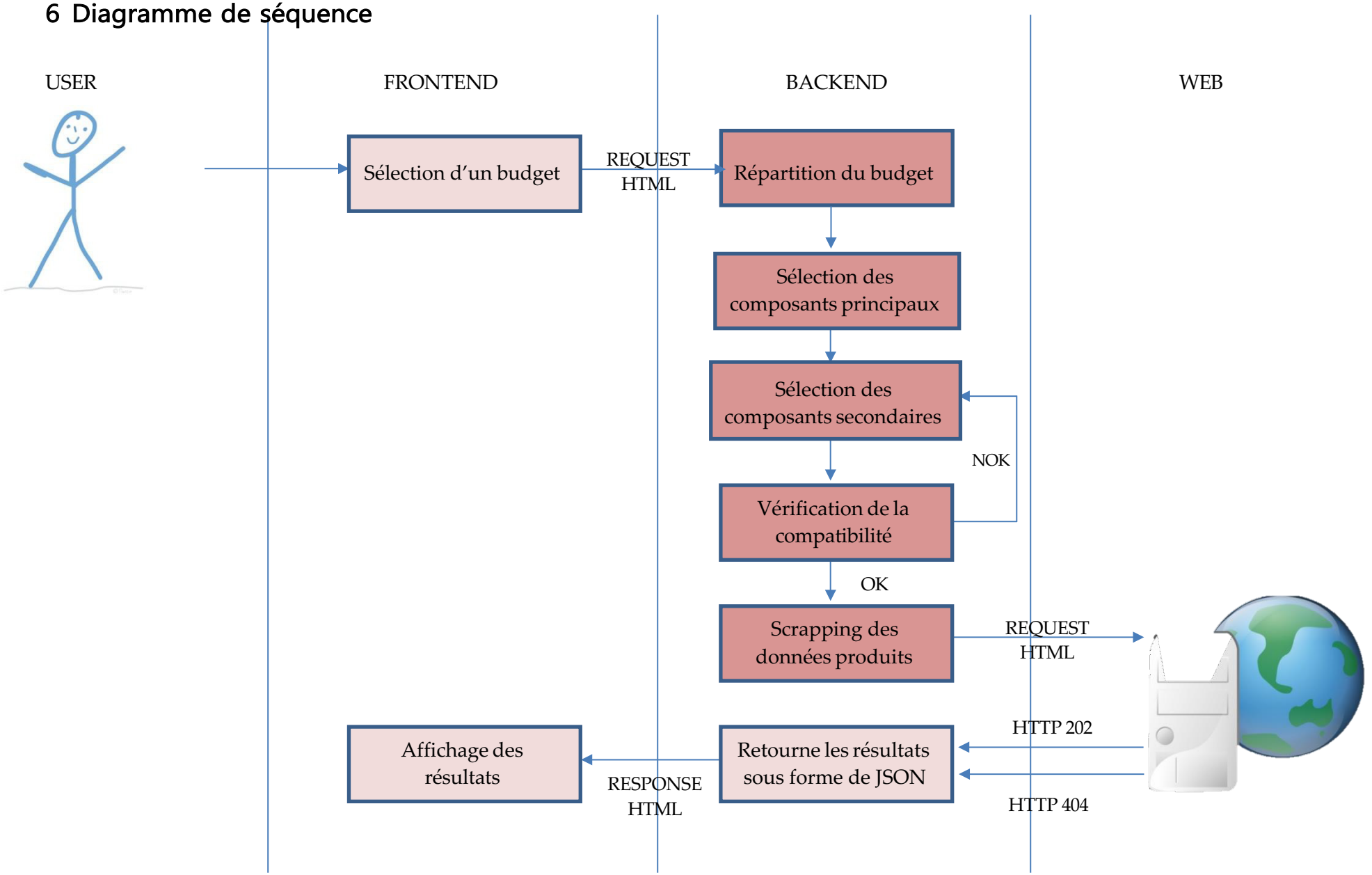
Le processus Scrappy nous retournera plusieurs csv qui seront ensuite utilisés dans le processus de configuration de l'ordinateur.

Pour l'extraction de données issus de sites comportant des composants dynamiques, nous utiliserons la librairie Selenium, qui permet d'obtenir des données chargées dynamiquement sur le site.

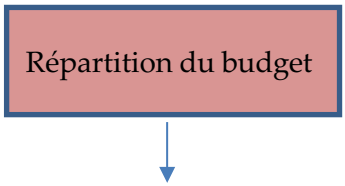
5 Architecture



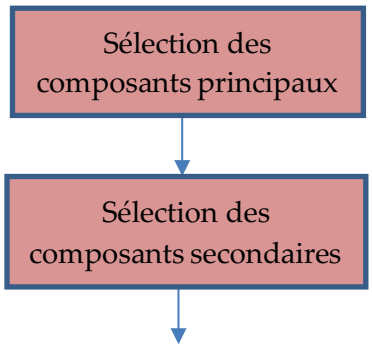
6 Diagramme de séquence



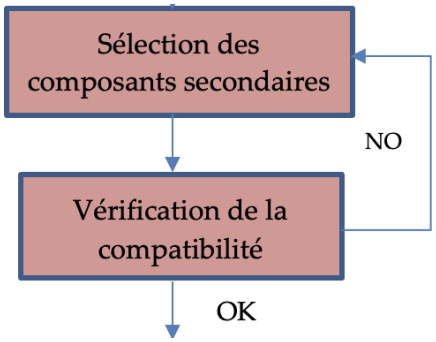
Sous-processus de composition de la configuration :



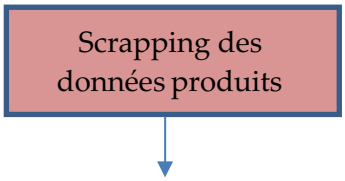
La répartition est basée sur une moyenne de budget calculée pour chaque composant et à partir de plusieurs dizaines de configurations : 30% pour le GPU, 20% pour le CPU, 10% pour la RAM, 10% pour le SSD et le pourcentage restant pour les composants secondaires.



La sélection des composants se fait en cascade, en débutant par les composants les plus influents sur la performance du système, les composants « principaux ». Ces composants ne répondent à aucune exigence de compatibilité et seront donc sélectionnés indépendamment. Lors de la sélection, nous utiliserons les données extraites de www.userbenchmark.com pour trouver, selon un budget donné, les composants possédant le meilleur score benchmark (score de simulation attestant de la performance d'un composant).



Une fois la sélection des composants principaux terminée, nous passerons à la sélection des composants secondaires, qui quant à eux, sont concernés par les normes de compatibilité. Nous avons extrait ces normes du site www.ldlc.ch et les avons traduit sous la forme de conditions au sein de notre code afin de tester pour un composant donné, sa compatibilité avec l'ensemble des composants déjà sélectionnés. Dans le cas où un composant s'avérerait incompatible, nous prendrons tout simplement le composant le précédent, et ce jusqu'à remplir les conditions de compatibilité. A savoir que la sélection de ces composants se base sur leur prix respectif, et non leur score benchmark, cette donnée n'étant pas toujours disponible.



Et enfin, une fois la liste des composants établie, nous scaperons sur www.toppreise.ch les dernières offres fournisseurs disponibles sur le marché et composerons la réponse HTML qui sera retournée dans le frontend à l'utilisateur (produits et fournisseurs sélectionnés).

7 Technique d'analyse

Afin de donner une bonne estimation de la valeur de chaque composant, nous allons fusionner les données de Benchmark de www.userbenchmark.com, et les prix du jour venant de www.ldlc.ch. C'est à partir de ces données que nous allons ensuite pouvoir estimer quels sont les meilleurs composants en fonction du budget du client.

La compatibilité des composants sera analysée à l'aide d'un système de règles. Par exemple, un certain processeur va imposer un socket particulier pour la sélection de la carte mère. Ou encore, l'alimentation devra avoir une puissance supérieure à tous les composants contenus dans l'ordinateur.

8 Contraintes et risques

Les contraintes que nous avons identifiées sont :

- Disponibilité continue des données du marché
- Extraction des données en temps réel (à chaque requête utilisateur)

Quant au risque majeur de notre projet, il porte sur la disponibilité des données. Nous devrons minimiser le risque d'une perte d'accès à ces données et procéder consciencieusement et avec précaution lors du scrapping de data.

9 Décisions et résultats attendus

Dans une optique de minimisation des risques techniques de notre application :

- Seuls les composants des 3 dernières générations seront considérés ;
- La résolution et la performance ne seront pas des paramètres de requête utilisateur car ces données ne sont pas toujours accessibles et difficile à caractériser ;
- Nous n'utiliserons pas la loi d'Amdahl car elle ne permet pas trouver la meilleure combinaison CPU-GPU mais uniquement la meilleure GPU pour un CPU donné ;
- Nous regrouperons les GPU par modèle, afin de réduire la dimensionnalité de notre dataset, car les métriques de performance ne diffèrent que très peu entre les variantes d'un même modèle

En termes de résultat, nous nous attendons à observer un nombre important de combinaisons possibles de composants. Ainsi, nos règles/conditions de compatibilité technique devront être rigoureuses et efficaces afin de garantir un résultat cohérant et fonctionnel à l'utilisateur.

10 Planning

START DATE 04.03

END DATE 06.23

PROJECT DURATION in days 82

WBS NO.	TASK NAME	STATUS	ASSIGNED TO	START DATE	END DATE	DURATION in days
1	**Gestion de projet**	**Complete**		**4.3**	**5.5**	**13**
1.1	Cahier des charges	Complete	TEAM	4.3	4.14	12
1.2	Planning projet	Complete	TEAM	4.3	4.14	12
1.3	Validation projet	Not Started	**TEAM**	5.5	5.5	1
2	**Data processing**	**Complete**		**1.28**	**2.14**	**18**
2.1	Collection des données	Complete	TEAM	4.17	4.21	5
2.2	Nettoyage des données	Complete	TEAM	4.17	4.21	5
2.3	Analyse exploratoire	Complete	TEAM	4.17	4.21	5
2.4	Feature selection	Complete	TEAM	4.17	4.21	5
3	**Développement du microservice**	**In Progress**				**1**
3.1	Conception	In Progress	TEAM	4.24	5.19	26
3.1.1	Frontend	Complete	TEAM	4.24	5.19	26
3.1.2	Backend	In Progress	TEAM	4.24	5.19	26
3.1.3	Scrapper	Complete	TEAM	4.24	5.19	26
3.2	Optimisation	In Progress	TEAM	4.24	5.19	26
3.3	Tests de fonctionnement	In Progress	TEAM	5.19	5.22	4
3.4	Validation	Not Started	TEAM	5.19	5.22	4
3.5	Deploiement	Not Started	TEAM	5.19	5.22	4
4	**Suivi de projet**	**In Progress**				**1**
4.1	Documentation	In Progress	TEAM	4.3	6.23	82
4.4	Présentation finale	Not Started	TEAM	6.23	6.23	1