52711

Bachelier en informatique de gestion

Rapport de stage effectué dans la firme

Fabricom-Equans

Ezeagwula Emmanuel

1 .Introduction

1.1 Présentation de l'environnement

1.2 Introduction de l'entreprise

Je fais actuellement mon stage chez Equans. Il s'agit d'une entreprise qui permet d'apporter des services multi-techniques. Equans fournit des solutions diverses et variées afin de pouvoir améliorer les équipements et les processus techniques de leurs clients.

Equans apporte un service complet allant du début de la conception jusqu'à l'installation en passant par tout ce qui est maintenance et le financement des équipements des clients.

Equans se concentre essentiellement sur trois segments de clientèle tels que les villes, les industries et également les bâtiments afin de rendre ces écosystèmes plus verts et efficaces sur le plan environnemental et énergétique.

1.3 Remerciements

Je tiens tout particulièrement à remercier mon Maître de Stage, Geoffrey Anthoon pour son accueil, le temps passé ensemble et le partage de son expertise et de ses conseils au quotidien. Je remercie également Nicolas Ruhland car grâce à son aide, j'ai pu accomplir mes tâches au sein de l'entreprise. Il fut d'une aide précieuse dans les moments les plus délicats.

Je tiens également à remercier toute l'équipe Equans de m'avoir accueilli si chaleureusement dans leurs rangs, mais également de répondre à toute question de ma part si gentiment.

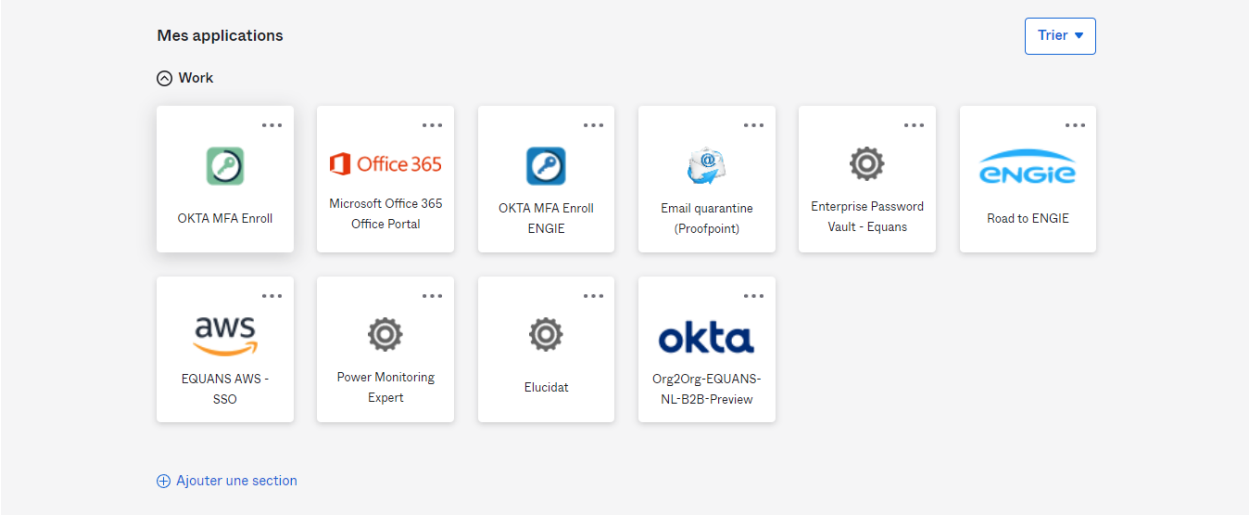
1.4 Collaboration avec Finemeca

La société finemeca est importante car c'est pour elle que je fais le projet. Finemeca est une société qui essentiellement accompagne les PME dans la construction de machines sur mesure et aussi l'automatisation car avec l'émergence de l'IOT(Internet of things), ils ont décidé de prendre les devants afin de proposer des solutions plus avancées.

Grâce à l'IOT, ça leur permet de monitorer les installations afin de pouvoir prévenir les problèmes techniques et également d'analyser les performances sur le long terme.

1.5 Infrastructure/Système informatique de chez Equans/Engie

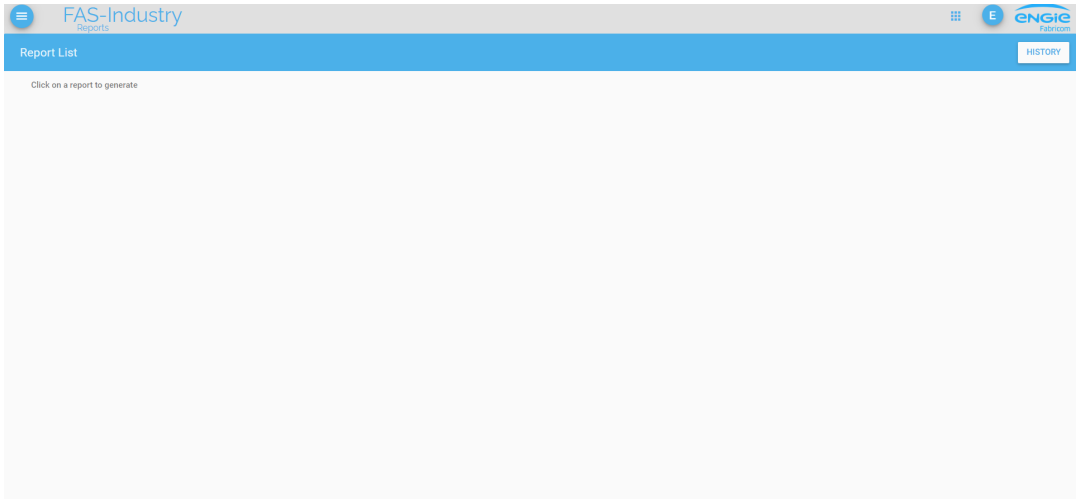
L'environnement informatique de chez Equans est assez rigoureux lors de la première utilisation parce que ce n'est pas centralisé à 100% dans un même site . Dans le tableau de bord d'Equans, on se retrouve après être authentifier avec une page web qui nous propose une suite applicative au développeur mais aussi d'autres applications pour les employés de l'entreprise.



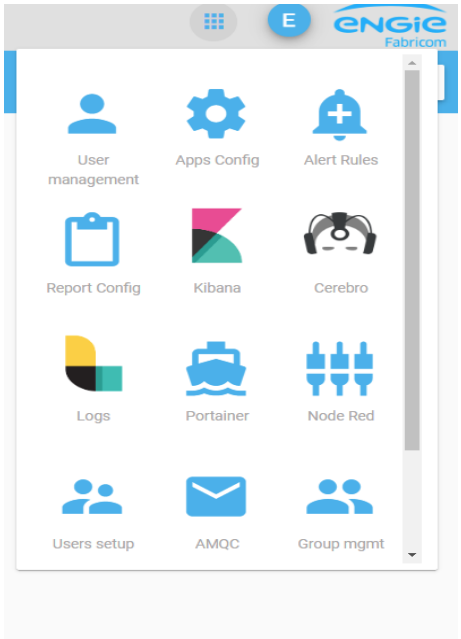
La 2e partie du système informatique qui nous sera utile pour tout ce qui est

réalisation de projet et aussi la suite applicative de développement se trouve sur le portail fas-industry.

Pour information, FAS (Fabricom Airport System) est l'équipe qui a initialement développé l'interface



La page est relativement vide au premier abord, mais elle comporte 2 volets de navigation qui montrent d'un côté les projets de dashboarding et aussi de l'autre coté l'écosystème d'application indispensable au bon fonctionnement et à la réalisation de projet.



2. Projet Finemeca (Projet principal)

2.1 Introduction

Le projet sur lequel je travaille actuellement est d'aider Finemeca avec pour objectif d'apporter un système de monitoring de données et de l'analyse de données également.

En effet, Finemeca est une entreprise qui propose des machines IOT sur mesure contenant des données importantes. Du coup, Pierre, le fondateur de la société Finemeca nous envoie les données captées par leur devices/machines par des protocoles que je devrais traiter par la suite.

Le but est de pouvoir offrir une solution de dashboarding qui apportera une solution simple et intelligente. De sorte qu'on puisse voir les données en temps réel pour qu'on puisse faire des constats sur le bon fonctionnement des machines.

Aussi, s'il y a le temps également de proposer un système d'alerte qui permettra de notifier à Finemeca que potentiellement leurs clients font face à une défaillance dans leur système.

Le but est que Finemeca puisse économiser de l'argent car il travaille avec la pression d'air, cette énergie coûte assez cher sur le marché, il faut donc trouver une solution efficace qui permet d'être plus économe.

2.2 Présentation

Au cours de mon stage je dois réaliser 3 dashboards différents qui représentent la visualisation des données de 3 différents capteurs.

1. air (pour la détection de fuite d'air)
2. centrale (la centrale de mesure)
3. devicev01 (pour la visualisation de la température)

Mon projet est donc de réaliser une visualisation graphique des données provenants des machines sur mesure de chez Finemeca.

Ainsi, lorsque les données seront capturés par la **Gateway** (l'intermédiaire qui permet de faire communiquer les capteurs sur le réseau) et envoyés par **l'Ingestor** (celui qui permettra de faire stocker ces données dans la base de données) ainsi à partir de ces données on peut commencer à créer nos dashboards comme affiché ci-dessous.

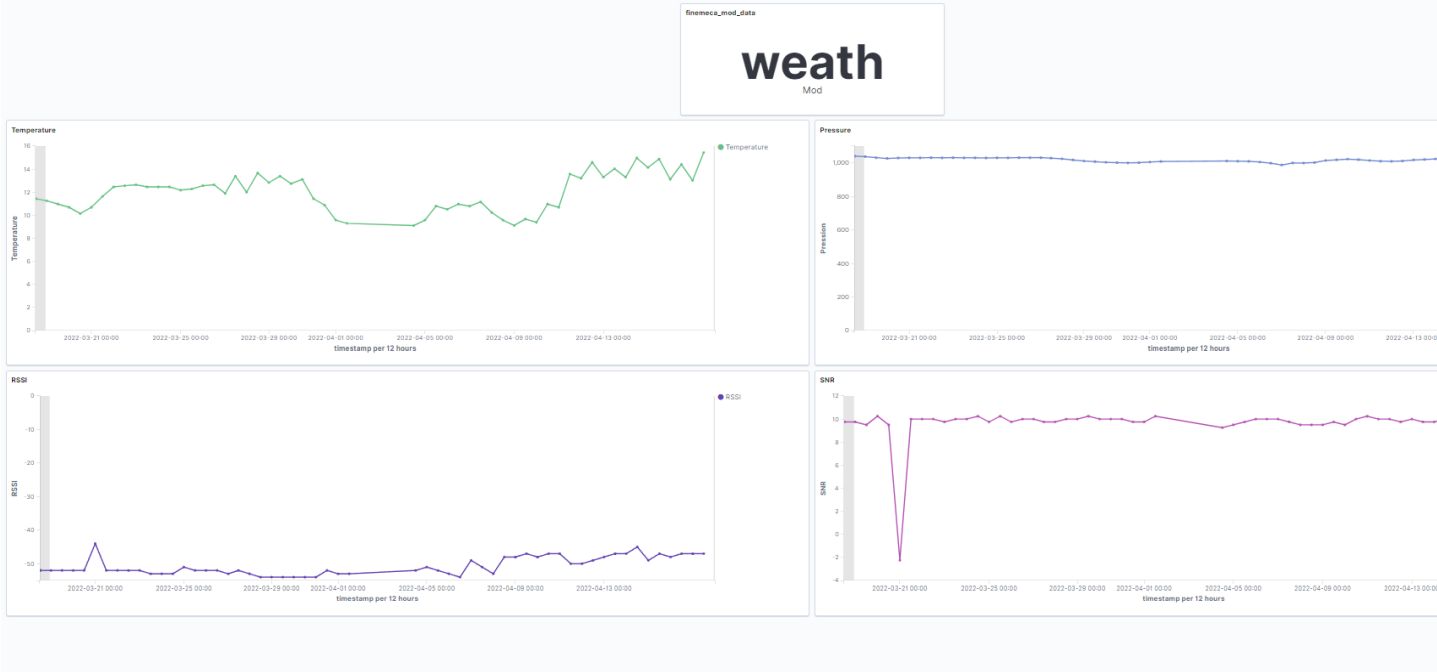
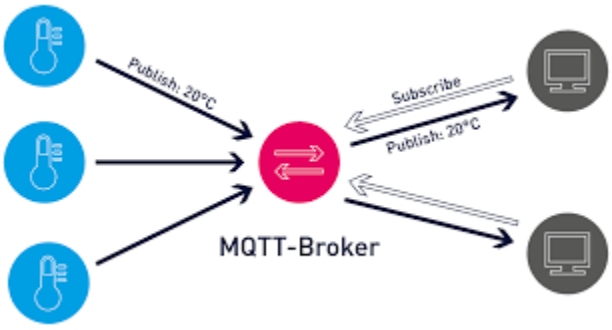


figure 2 : Dashboarding

2.3 Processus de réalisation de service

2.3.1 Creation de la Gateway

Qu'est-ce qu'une Gateway ? Lors de la réalisation du projet des 3 capteurs (air, centrale, devicev01) afin de pouvoir récupérer les données



provenant d'un device IOT, on doit passer par une gateway qui sert de pont pour communiquer avec le device via le protocol MQTT (Message Queuing Telemetry Transport).

Lorsque, Finemeca fait tourner sa machine, des données sont envoyés par le protocol MQTT et mon programme va intercepter ces données qui sont en format JSON et ces données seront envoyé sur ActiveMQ(AQMC) ce qui permettra de stocker ces données sous forme de file.


```
{
  "Customer": "finemeca",
  "idGateway": 170,
  "idDevice": 100,
  "Device": "testBroker",
  "Time": "2022-04-09 15:05:06",
  "msgCount": 202,
  "Length": 94,
  "RSSI": -47,
  "SNR": 10,
  "Mod": "weath",
  "Pres": 1015,
```


figure 4 : Données du capteur en JSON


```
Retrieved data from device: {'Customer': 'finemeca', 'idGateway': 170, 'idDevice': 100, 'Device': 'testBroker', 'Time': '2022-04-17 23:52:11', 'msgCount': 232, 'Length': 94, 'RSSI
1020, 'Pres-U': 'hPa', 'Temp': 14.51, 'Temp-U': 'C', 'Hum': 'NULL', 'Hum-U': 'NULL'}
Sent to AMQC!
Retrieved data from device: {'Customer': 'finemeca', 'idGateway': 170, 'idDevice': 100, 'Device': 'testBroker', 'Time': '2022-04-17 23:52:41', 'msgCount': 233, 'Length': 94, 'RSSI
1020, 'Pres-U': 'hPa', 'Temp': 14.51, 'Temp-U': 'C', 'Hum': 'NULL', 'Hum-U': 'NULL'}
Sent to AMQC!
```


figure 5 : La gateway dans le conteneur Docker

2.3.1.1 Utilité de la Gateway

La passerelle MQTT agit comme un pont pour les capteurs et les appareils IoT / MQTT. Lorsque les appareils MQTT sont répartis sur différents sites, nous avons besoin d'un serveur partagé pour centraliser les données et analyser ces données.

2.3.2 Creation du Ingestor

Qu'est-ce qu'un Ingestor ? un ingestor est essentiellement un programme qui va ingérer les données et les transférer dans la base de données.

Tout d'abord, on doit se poser la question : quelle donnée est plus importante qu'une autre ? Car ça ne sert à rien de tout garder. Je m'adresse directement à Finemeca et mes collègues afin de savoir quels sont les données à conserver dans la base de données.

Parfois, ça sera nécessaire d'avoir une transformation de données c'est à dire, si le client aimerait avoir une donnée mais qu'elle n'a pas été envoyée via le capteur (un capteur a ses limitations), alors je dois pouvoir ajouter une donnée en plus dans le développement par l'intermédiaire de formules. Par exemple, on m'avait demandé d'avoir une consommation journalière de l'électricité, ce qui veut dire que je dois réaliser une différence sur 2 de données afin d'en avoir une nouvelle.

Les données seront envoyés sur la base de données ElasticSearch.

```
response: {'_index': 'finemeca-devicev01-data-2022-4-17', '_type': '_doc', '_id': 'PbWFOYABZZpXiE0EVxg8', '_version': 1, 'result': '
_term': 1}
Sent deviceV01 metric: {'Customer': 'finemeca', 'idGateway': 170, 'idDevice': 100, 'Device': 'testBroker', 'msgCount': 237, 'Length'
'Temp-U': 'C', 'timestamp': datetime.datetime(2022, 4, 17, 22, 54, 41)}
```

figure 7 : Données envoyés à la base de données par l'Ingestor

2.3.2.1 Utilité du Ingestor

L'ingestion de données a pour processus d'obtention et d'importation de données en vue de leur utilisation immédiate ou de leur stockage dans une base de données. L'ingestion de données consiste à prendre quelque chose en charge ou à l'absorber. Les données peuvent être transmises en temps réel ou ingérées par lots.

2.3.2.2 Exemple

Transférer les données de divers systèmes internes vers une plateforme de reporting ou d'analyse à l'échelle de l'entreprise - un lac de données, un entrepôt de données ou un autre format de dépôt normalisé.

2.3.3 Dockeriser/Conteneuriser le programme

Chez Equans, la plupart des projets doivent pouvoir tourner dans un conteneur Docker car ça permet de faciliter la gestion de dépendances au sein d'un projet.

Lorsque le projet peut enfin tourner sans erreur et que la Gateway et l'Ingestor marchent parfaitement en harmonie, on peut enfin déployer nos images Docker fourni par le département informatique et pousser nos images Docker dans le portainer (gestionnaire de conteneurs) dans le serveur d'Equans.

Attention : Il faut savoir qu'on crée des conteneurs Docker distinct pour la Gateway et l'Ingestor donc 2 conteneurs Docker par capteur. Donc au total 6 conteneurs Docker sont crées au final.

Name	State	Quick actions	Stack	Image	Created	IP Address	Published Ports
sftp_server	running		docker-compose	atmoz/sftp	2022-04-11 15:46:17	172.18.0.31	2222:22
finemeca_gateway-air	running		docker-compose	equansbeacr01.azurecr.io/finemeca-gateway-air	2022-04-04 12:19:04	172.18.0.42	-
finemeca_ingestor-air	running		docker-compose	equansbeacr01.azurecr.io/finemeca-ingestor-air	2022-04-04 12:19:04	172.18.0.41	-
h2cs-front	running		docker-compose	equansbeacr01.azurecr.io/h2cs-front:latest	2022-03-17 14:40:06	172.18.0.40	5001:5000
h2cs-back	running		docker-compose	equansbeacr01.azurecr.io/h2cs-back:latest	2022-03-17 12:21:51	172.18.0.34	4000:4000
finemeca_ingestor-centrale	running		docker-compose	equansbeacr01.azurecr.io/finemeca-ingestor-centrale	2022-03-16 22:04:06	172.18.0.39	-

figure 8: Portainer avec les différents services

2.3.4 Prise de décision sur quels type de graphiques utilisés

Cette partie est importante car certains graphiques permettent de mieux mettre en avant la donnée et la problématique qu'on essaie d'analyser donc généralement on en parle lors du meeting du lundi sur quelles données devraient être afficher.

Lorsque le résultat est satisfaisant on peut conclure avec cette partie et plus tard si le client (Finemeca) voudrait d'autres types de visualisation on pourra le mettre à jour rapidement car les données qui sont dans les conteneurs docker tourne tous les jours sur le portainer.

3. Outils et Architecture

3.1 Outils utilisés

Les technologies utilisées afin de réaliser le projet de Finemeca sont :

- Python (Langage de programmation)
- Pip (Gestionnaire de librairie python)
- MQTT (Protocole de Messagerie)
- ActiveMQ (AQMC) (=Active Message Queue Client)
- Visual Studio Code (L'éditeur de code)
- Docker (Conteneur de service/programme)
- Portainer (Gestionnaire de conteneur)
- ElasticSearch (Base de Données NoSQL)
- Kibana (Visualisation de données pour ElasticSearch)
- Git (Gestionnaire de service)

3.1.1 Python

Python est le langage de programmation utilisé et jugé le plus pratique/efficace pour notre cas de figure. Grâce à l'installateur de paquets de Python, PIP, nous permet d'avoir accès à un catalogue de librairies qui nous facilite la vie pour nos usages.

Par example, la libraires Paho-mqtt, qui nous permet de communiquer avec les appareils connectés à travers le réseau avec le protocol MQTT.

Ainsi de même, pour la librairie Elasticsearch qui nous permet de communiquer avec l'API ElasticSearch. Du coup on peut effectuer des opérations sur la base de données tel qu'enregistrer de nouvelles données, supprimer des données, etc…

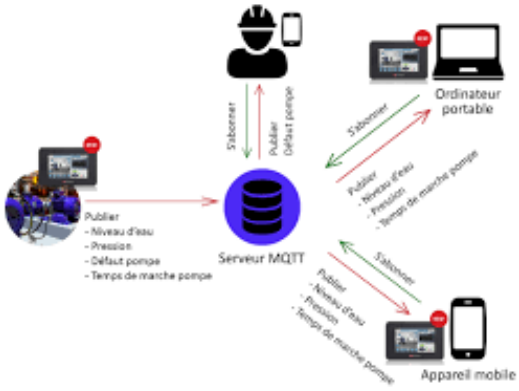
3.1.1.1 Pourquoi Python et pas un autre langage de programmation ?

On utilise python car c'est le langage qui nous permet d'interagir facilement et d'établir une connexion avec le protocol MQTT.

Python vient embarqué avec des petites libraires qui facilitent grandement le travail du développeur. Du coup, on doit pas se tracasser à recréer des fonctionnalités que des librairies viennent avec par défaut.

Python est également un langage de programmation facile d'utilisation lorsqu'on a les bases nécessaires sachant qu'à l'ESI, le langage de programmation par défaut est Java, Python n'est réellement pas compliqué d'apprendre car la syntaxe est assez libre.

3.1.2 MQTT



Le protocol MQTT (Message Queuing Telemetry Transport) est un protocole de communication qui nous servira essentiellement dans notre cas de figure de pouvoir retranscrire les messages émis par

les capteurs depuis Finemeca afin qu'on puisse prendre ces données sous format JSON qui permettra de traiter facilement celle-ci. On doit passer par ce protocol afin de pouvoir récolter les données des capteurs. Donc elle représente l'une des composante clé au niveau du développement car sans ce protocol on ne pourra pas avancer plus loin.

3.1.2.1 Pourquoi MQTT ?

Se situant au-dessus de la pile réseau TCP/IP, MQTT est un protocole de messagerie léger de type **publish/subscribe** conçu pour les réseaux à faible bande passante, à latence élevée et peu fiables. Les caractéristiques de MQTT en font une excellente option pour l'envoi **de gros volumes de messages de capteurs** vers des plateformes d'analyse et des solutions en cloud.

Ce protocole nous permet d'emmagasiner un gros volume de données et de faire du traitement de données dans du code.

3.1.3 ActiveMQ

Le service ActiveMQ est un service de messagerie open source populaire construit avec le langage Java. Il fonctionne comme un intergiciel(Middleware) orienté message (MoM).

ActiveMQ permet notamment d'envoyer des messages entre 2 à plusieurs applications. Dans notre cas de figure, ça nous permet de pouvoir stocker les messages de nos différents capteurs à travers une gateway. Ensuite activeMQ va servir de points relais pour que notre Ingestor puisse consommer les données se trouvant dans la pile de messages.

3.1.4 Docker

Docker est un outil qui permet de conteneuriser des applications dans des conteneurs logiciels afin de faciliter la gestion de dépendances de celle-ci.

Ainsi que, de proposer une certaine uniformisation des données et qu'on évite des problèmes du genre "J'ai une donné différente d'un système d'exploitation à un autre".

Dernièrement, ça permet de facilement résoudre d'éventuel bugs sur un conteneur cible que de devoir tout débugger.

3.1.5 Portainer

Portainer est un puissant gestionnaire de conteneurs Docker, Kubernetes, etc….

Cela apporte une solution efficace qui permet de gérer les états de nos machines Docker car il faut savoir qu'on travaille essentiellement avec des machines Docker donc il faut bien qu'on ait un outil qui puisse facilement voir l'état de santé, les logs ou encore les erreurs que les machines Docker nous renvoient.

3.1.6 ElasticSearch

ElasticSearch est une base de données orientée document, c'est essentiellement là dessus qu'on stocke les messages/données sous format JSON après qu'on ait bien effectué les traitements demandés en fonction du besoin du client/particulier.

Elasticsearch est également un moteur de recherche et d'analyse distribué et open-source basé sur Apache Lucene et développé en Java.

Au départ, il s'agissait d'une version évolutive du cadre de recherche open-source Lucene, puis la possibilité d'étendre horizontalement les indices Lucene a été ajoutée. Elasticsearch permet de stocker, de rechercher et d'analyser d'énormes volumes de données rapidement et en temps quasi réel, et de fournir des réponses en quelques millisecondes.

Il est capable d'obtenir des réponses de recherche rapides parce qu'au lieu de rechercher le texte directement, il recherche dans un index. Il utilise une structure basée sur des documents plutôt que sur des tables et des schémas.

3.1.7 Kibana

Premièrement, il faut savoir que ElasticSearch et Kibana font partie de la même suite de logiciel donc les données qui sont stockées dans la base de données d'ElasticSearch peuvent être utilisées afin de créer des visualisations.

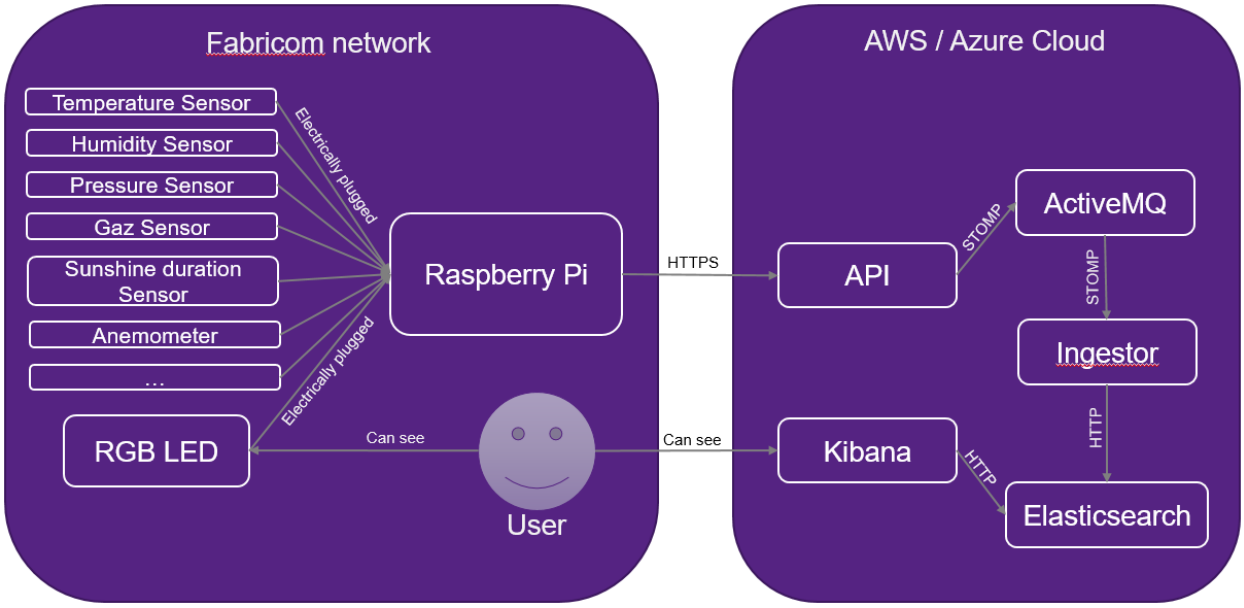
Kibana est donc un outil de gestionnaire de données d'Elasticsearch et également un outil pour créer des jolies visualisations tel que des histogrammes, des diagrammes linéaires, des graphiques linéaires, des camemberts, etc…

3.2 Architecture

Au cours de notre projet, ce qui va permettre le bon déroulement du projet principal est de respecter ces 3 points.

- Le développement **backend**
- Conteneurisation docker de nos programmes
- Visualisation graphique (Kibana - ElasticSearch)

3.2.1 Diagramme



3.2.1.1 Explication du diagramme

Ce diagramme permet de montrer le trajet d'une donnée provenant d'un capteur vers la base de données.

Les capteurs comme les capteurs d'humidité, de gaz, de pression, etc… sont électriquement branché sur un Raspberry Pi connecté sur un serveur du côté de chez Finemeca. Ensuite, on établit des requêtes via le protocol MQTT sur leur serveur afin de pouvoir récolter les messages qui seront ensuite envoyé sur **ActiveMQ**. Dès lors que nos messages se retrouvent sur ActiveMQ on pourra les envoyés sur notre base de données (**Elasticsearch**).

Enfin, comme Elasticsearch et Kibana appartiennent à la même branche, on peut créer nos dashboards que Finemeca pourra voir par la suite.

3.2.2 Backend

Avec l'architecture backend, on va pouvoir accéder et envoyer des données dans la base de données afin de pouvoir retransmettre les données jugées importantes dans une vue du côté client qui dans ce cas-ci est Kibana

Dans notre cas, notre base de données est ElasticSearch mais la vue Kibana fait partie du même écosystème qu'ElasticSearch.

3.2.3 Docker

La partie de conteneurisation est une partie très importante lors du développement de la solution surtout au niveau de l'architecture de chez Equans.

On doit passer par des conteneurs Docker afin qu'on puisse uniformiser les projets de sorte qu'on respecte le standard du département IT de chez Equans.

3.2.4 Visualisation graphique

On utilisera notamment Kibana, pour réaliser nos vues graphiques pour qu'on puisse montrer et mieux représenter les données et également voir s'il y a une amélioration à travers le temps.

4. Problème(s) rencontré(s)

4.1 Accès aux données via le MQTT

Le problème que j'ai du faire face au début du développement c'était que lors du codage, j'avais besoin de communiquer avec l'un des devices via une connection MQTT. ça me renvoyait une erreur car je ne pouvais pas établir la connection avec le MQTT de chez Equans.

Car pour établir une connection au sein du serveur MQTT de chez Equans, il fallait que je me fasse whitelisté mon adresse IP par le département IT. Du coup

je ne pouvais pas avoir accès aux données des devices. Sans celà, je ne pouvais pas déployer mon service car comme dit précédemment, on travaille avec des machines docker donc mon conteneur docker doit communiquer avec le docker qui établit la connection MQTT sur le serveur.

4.2 Délai pour recevoir les images Docker

Le second problème est le temps qu'on doit attendre afin qu'on puisse avoir accès aux images Docker dans lesquelles on pousse notre code dessus. Sans les images Docker, je ne peux pas déployer le service et par la même occasion ne pas pouvoir effectuer ma tâche.

5. Solution(s) trouvée(s)

5.1 1ère solution

La première solution qu'on m'a conféré au début du développement, c'est qu'au lieu de directement travaillé sur le MQTT du serveur de chez Equans, de directement travailler avec celle de chez Finemeca en attendant que je me fasse whitelisté.

Certes, il y a des modifications à apporter dans le code tel que l'adresse dans laquelle le broker MQTT se trouve mais je pouvais avoir accès aux mêmes données que sur le serveur donc je pouvais effectuer mes fonctions et mes traitements en local sans être freiné puisque ce sont les mêmes données.

5.2 2e solution

La seconde solution est tout simplement d'anticiper à l'avance combien de temps ça me prendrait pour développer un service dont 2 parties (Gateway/Ingestor) ce qui veut dire que je dois développer sur 2 conteneurs docker différents donc je dois envoyer une demande à un autre département informatique de me fournir ces images afin qu'ils puissent me fournir les 2 images dans un petit laps de temps. Et je peux qu'attendre car je ne peux pas passer de tests sans passer par des conteneurs.

6. Système de gestion des alertes (Projet auxiliaire)

6.1 Explication

Le système de gestion d'alertes consiste à pouvoir envoyer des alertes spécifiques (en particulier du capteur air) en fonction des données qui sont susceptibles à être des informations compromettantes pour les clients de chez Finemeca. Ce système est fait pour le capteur de détection de fuite d'air.

Ce système enregistre lorsque la pression courante est inférieur la pression d'ouverture ce qui signifie qu'il y a une fuite d'air dans leur système d'installation. Finemeca pourra facilement alerter leurs clients d'un problème éventuel.

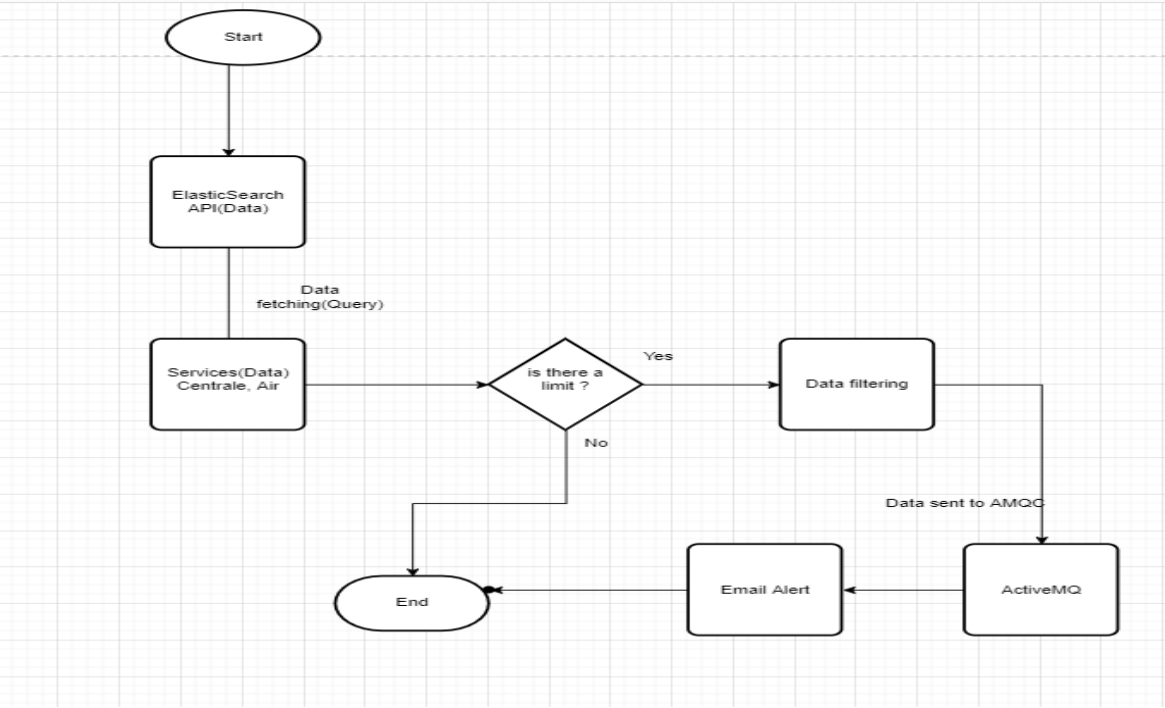
6.2 Fonctionnement

D'un point de vue technique, c'est qu'à partir de l'API provenant de ElasticSearch, qu'on a la possiblité de faire des requêtes sur les données dont on a besoin et à partir de ces données de les filtrer.

Lorsqu'une donnée jugée importante selon Finemeca dépasse un certain seuil préétablis, celle-ci sera envoyé sur AMQC et ensuite transmise à Finemeca sous forme d'alerte mail sous forme no-reply.

Ce qui pourra en théorie aider Finemeca à maintenir les capteurs de leurs clients et faciliter la maintenance sur du long terme.

6.3 Diagramme de flux



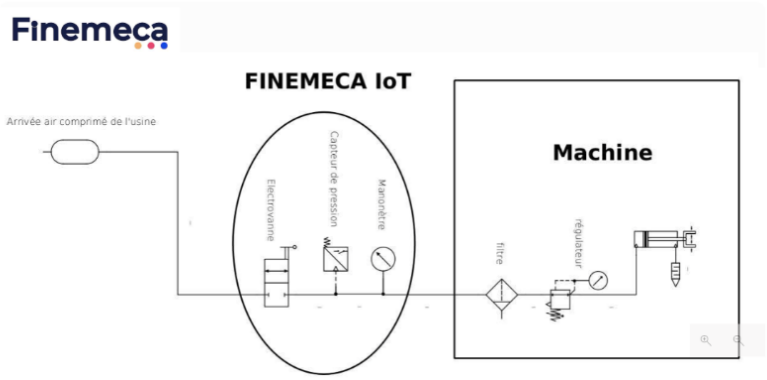
6.3.1 Explication

Ce diagramme permet de visualiser comment le système d'alarme déclenche les alertes au client.

On communique à la base de données à travers l'API d'Elasticsearch afin de récupérer toutes les informations en fonction d'une date sélectionnée, et à partir de ça on va voir si certaines de nos données jugé significatives ont dépassé la limite imposée par Finemeca.

Si la donnée dépasse la limite, Finemeca recevra une alerte avec la donnée en question. Dans le cas contraire, on ne fera rien.

7. Système de gestion des alertes (Projet auxiliaire)



7.1 Fonctionnement

Le but de cette page web est de pouvoir regrouper les données essentielles du capteur Air.

Cette page a été construite avec le framework Flask dont je l'utilise principalement car c'est assez simple de créer sa propre API.

J'envoie la donnée de la pression courante dans l'API et du côté client, avec du javascript on gère l'animation de la page d'accueil, à chaque nouvelle valeur détecté la valeur deviendra verte et l'actualisation se mettra à jour pour correspondre avec l'heure auquel la valeur aura été mise à jour.

8. Conclusion

Au bout de 15 semaines, l'objectif qui était de pouvoir réaliser des services et création de dashboards comme un développeur engagé à temps plein a donc été réalisé.

J'ai su faire des corrections, des améliorations et de rajouter de nouvelles fonctionnalités sur la partie backend. Cela m'a permis aussi de développer des services ainsi qu'une petite page web, ainsi que de la maintenir.

Ce stage fut une expérience tant enrichissante que positive au point de vue professionnel et humain. J'ai pu mettre en pratique et approfondir les acquis théoriques étudiés pendant mon parcours scolaire à l'École Supérieur en Informatique.

J'ai également pu acquérir de nombreuses connaissances durant ce stage. À savoir, une connaissance bien plus approfondie de Python dans un contexte de création de services, la découverte de nouveaux outils technologiques tel que Docker, Portainer, etc…

J'ai donc été confronté à la réalité du monde professionnel et du travail en équipe. Ces apprentissages constituent, pour moi, une excellente expérience professionnelle.

9. Annexes

9.1 Sources

https://fr.wikipedia.org/wiki/MQTT

https://www.redhat.com/fr/topics/containers/what-is-docker

https://activemq.apache.org/

https://www.elastic.co/fr/what-is/elasticsearch

https://www.elastic.co/fr/kibana/

https://fr.wikipedia.org/wiki/Flask_(framework)

https://blog.back4app.com/fr/la-signification-du-backend-et-tout-ce-que-vous-devez-savoir-a-son-sujet/

9.2 Carnet de bord

9.2.1 Semaine 1

Objectif(s) :

- Me familiariser avec l'équipe et surtout avec le développeur qui me supervise durant ma période de stage.

- Comprendre les outils de développement

Compte rendu explicatif :

Je devais me documenter concernant les outils de développement tel que MQTT, l'environnement Azure avec le dépôt git embarqué, Docker, Portainer, etc… afin de pouvoir avoir une compréhension plus claire.

Pouvoir également prendre part au discussion avec l'équipe et Finemeca dans le but d'apporter une solution à un problème.

Notes et remarques personnelles :

J'ai pu essayer de développer une version plus propre du code source de base dans une version de tests puisque je n'ai pas encore accès à tous les outils de développement étant donné que je suis encore nouveau au sein de l'équipe.

9.2.2 Semaine 2

Objectif(s) :

- Prendre la main sur le projet Finemeca

- Améliorer le projet en cours + rajouter des fonctionnalités en python

- Apprendre à utiliser les technologies (ElasticSearch,Kibana,Amqc,Mqtt,Docker)

- Rencontre du client(Finemeca) à travers Zoom

Compte rendu explicatif :

Le lundi, j'ai pu rencontrer à travers Zoom, le client, le patron de chez Finemeca. Il expliquait ses attentes par rapport au projet qui est déjà en cours, il m'a également accueilli en expliquant le but de leur entreprise ainsi que leur collaboration avec Equans.

Comme ce projet était déjà en cours, la récolte de données via MQTT était déjà présentes. Ma tâche dans ce projet est de devoir parser ses données à travers le protocol amqc qui distribue cette objet en json et je dois le parser dans le code afin de l'envoyer dans la base de données afin de pouvoir faire une étude sur cette données.

Notes et remarques personnelles :

/

9.2.3 Semaine 3

Objectif(s) :

- Permettre de faire tourner le code sur un conteneur Docker

- Tester le code

- Créer un dashboard via Kibana(Optionnel)

Compte rendu explicatif :

Cette semaine consistait à faire des tests unitaires du projet en cours(Finemeca) afin de faire valider le projet y compris de faire tourner les services via un docker afin de pouvoir importer les images docker sur les serveurs azure de chez Equans.

Également, si j'ai le temps de devoir également de commencer à me familiariser avec la base de données Elasticsearch ainsi que Kibana le visualiseur de données.

Notes et remarques personnelles :

/

9.2.4 Semaine 4

Objectif(s) :

- tester les méthodes

- déployer les services(Docker) sur le serveur(Portainer)

- créer un service sur le topic Air

Compte rendu explicatif :

Cette semaine consistait à terminer mes tests unitaires sur les méthodes unitaires que j'utilise dans mes services ainsi que de tester que les 2 services que j'ai crée fonctionnent bien en production sans déclenchement d'erreur.

Ainsi, il fallait aussi que je déploie mes services Docker sur aws pour qu'ils puissent être dans le Portainer(gestionnaire de service Docker) de l'entreprise.

Dernièrement, je devais réfléchir pour que je puisse déjà commencer à établir un nouveau service pour Finemeca à propos du nouveau topic (Air).

Notes et remarques personnelles

/

9.2.5 Semaine 5

Objectif(s) :

- création de 2 dashboards

- Sélection de données pour l'affichage des premiers dashboards

Compte rendu explicatif :

Cette semaine consistait à créer à partir des services que j'ai développé jusqu'à

maintenant de créer des dashboards grâce aux données récupérées et envoyées par les services.

Par la suite, je devais créer des graphes par rapport à la donnée que le client aimerait mettre en avant. J'ai dû organiser une réunion avec l'équipe pour juger quelles données valaient la peine d'être afficher dans un dashboard.

Notes et remarques personnelles :

/

9.2.6 Semaine 6

Objectif(s) :

- Ajout des éléments dans le dashboard (Kibana)

- Faire en sorte que le client puisse avoir accès à ces dashboards

Compte rendu explicatif :

Cette semaine a débuté par une réunion avec le groupe en expliquant ce qu'on a réalisé la semaine passée. Ensuite, j'ai eu une réunion avec Finemeca pour m'indiquer ce qu'il voudrait que j'ajoute dans le dashboard.

Après ça il fallait que j'ajoute l'application(dashboard) sur le portail afin que Finemeca puisse y avoir accès depuis sa machine.

Notes et remarques personnelles :

/

9.2.7 Semaine 7

Objectif(s) :

- Maintien de services

- Gestion de services crées

Compte rendu explicatif :

Cette semaine était une semaine relativement calme car la société pour laquelle je développe l'application (Finemeca) est en train de tester et de réparer leurs capteurs à cause de multiples oublis de données que je leur ai fait part.

Du coup, il n'y avait plus de données en entrée et je ne pouvais pas créer comme convenu dans le planning de cette semaine, le service pour leur capteur Air d'autant plus que je devais attendre qu'on me donne mes dépôts Docker dans lequel je peux ajouter mon service.

Étant donné cet imprévu , je devais me charger de la maintenance et de la vérification des logs pour éviter un bug éventuel en production.

Notes et remarques personnelles :

/

9.2.8 Semaine 8

Objectif(s) :

- Déploiement du service Air

- Réalisation d'un dashboard(kibana) pour le service Air

Compte rendu explicatif :

Cette semaine, comme du côté de chez Finemeca, ils ont pu régler le soucis avec leur capteur, du moins temporairement et qu'également j'ai pu recevoir les images docker dans lequel je peux faire tourner mes services alors je pouvais enfin me tourner vers le développement du service du module air .

Après que le service a pu tourner impeccablement sur le serveur et que les données ont été envoyées à la base de données ElasticSearch, je pouvais passer au dashboarding avec les données qui ont été traitées pour réaliser des affichages graphiques.

J'ai également eu une réunion en fin de semaine avec des développeurs car on est

en train de penser à comment implémenter un système qui envoie des alertes par mail lorsque les données atteignent un certain seuil. Ça pourra donner à Finemeca un retour au niveau de ses capteurs (devices). On aura encore plus d'indication par rapport à ça car c'est une autre équipe qui gère tout le maintien des serveurs et des services mails.

Notes et remarques personnelles :

/

9.2.9 Semaine 9

Objectif(s) :

- Déploiement du service Air

- Réalisation d'un dashboard(kibana) pour le service Air

Compte rendu explicatif :

Cette semaine, comme du côté de chez Finemeca, ils ont pu régler le soucis avec leur capteur, du moins temporairement et qu'également j'ai pu recevoir les images docker dans lequel je peux faire tourner mes services alors je pouvais enfin me tourner vers le développement du service du module air .

Après que le service a pu tourner impeccablement sur le serveur et que les données ont été envoyées à la base de données ElasticSearch, je pouvais passer au dashboarding avec les données qui ont été traitées pour réaliser des affichages graphiques.

J'ai également eu une réunion en fin de semaine avec des développeurs car on est en train de penser à comment implémenter un système qui envoie des alertes par mail lorsque les données atteignent un certain seuil. Ça pourra donner à Finemeca un retour au niveau de ses capteurs (devices). On aura encore plus d'indication par rapport à ça car c'est une autre équipe qui gère tout le maintien des serveurs et des services mails.

Notes et remarques personnelles :

/

9.2.10 Semaine 10

Objectif(s) :

- Faire passer mes données d'alerte dans le protocole AMQC (en cours).

- pouvoir récolter les données jugées importantes pour qu'on puisse tester le système d'alerte. (Cette tâche est en pause tant que je n'ai pas les alertes à disposition étant donné le congé)

Compte rendu explicatif :

Cette semaine était relativement calme car la plupart des collègues étaient en congé, je ne pouvais donc pas trop avancer. J'avais besoin qu'on puisse me fournir les accès pour que je puisse commencer à travailler avec les système d'alerte mais je devais attendre que le département IT puisse me les fournir.

J'ai donc consacrée du temps de faire la récolte de données et le filtrage de données selon le besoin de Finemeca. C'est une étape importante, il faut passer par ça avant d'afficher une alerte quelconque.

Notes et remarques personnelles :

/

9.2.11 Semaine 11

Objectif(s) :

- Discussion par rapport aux soucis du côté de Finemeca

- Interactive Hub(discussions de projets)

Compte rendu explicatif :

Cette semaine était assez calme, ça se résumait à des réunions pour les futures projets et comment on compte les implémenter en fonction des profils de clients.

J’ai eu également des appels du client Finemeca afin qu’il m’explique si j’ai bien compris ses attentes.

Finalement, on avait passé ce vendredi 22 avril à Louvain dans un interactive Hub afin qu’on puisse réaliser un exercice avec le département pour réaliser une simulation entre la conception d’une visualisation graphique pour un CFO (pour le domaine financier) et également le CTO (dans le domaine technique).

Notes et remarques personnelles :

Après qu’on a finit notre travail dans l’interactive Hub durant la matinée, on avait passé une journée sympa à Louvain. On a pu visiter Louvain car ça faisait partie du programme du vendredi.

9.2.12 Semaine 12

Objectif(s) :

- Suite de l’implémentation du système d’alerte de données

Compte rendu explicatif :

Cette semaine, je poursuis l’implémentation du système d’alerte en attendant que le département IT me fournisse les accès. J’ai toujours pas de nouvelles du département IT, ce qui devient un peu problématique car sans ces accès, je ne peux pas plus tester et potentiellement résoudre de potentiels problèmes.

Notes et remarques personnelles :

/

9.2.13 Semaine 13

Objectif(s) :

- Attente des accès pour le système d’alerte

- Documentation sur le framework Flask

Compte rendu explicatif :

Comme je suis toujours en attente pour qu’on me fournisse les accès pour le

système d'alerte car il me reste juste l'implémentation du SDK d'amazon afin que je puisse lier mon code de base avec les accès. Par manque de temps, j'ai décidé de faire une autre tâche qui fait partie de mes priorités, la page dynamique.

Du coup, je me suis un peu renseigné sur le framework Flask afin de pouvoir créer un petit site web d'une démonstration d'un système de dashboard dynamique. Flask me sera fortement utile pour créer une petite API que je pourrais utiliser pour afficher des données du côté client.

Notes et remarques personnelles :

/

9.2.14 Semaine 14

Objectif(s) :

- Suite de l'implémentation du dashboard/image dynamique

Compte rendu explicatif :

Cette semaine consistait à discuter mon approche de base sur l'implémentation d'un page dynamique. Cette page doit contenir le schéma de base du capteur d'air ainsi qu'en dessous, de contenir les informations comme la pression courante actuelle et le temps d'actualisation de cette pression.

Afin de faire cette tâche, j'ai du utiliser flask pour créer une API avec les données dont j'en aurai besoin. Ensuite, le javascript pour le côté client pour afficher mes différentes informations à travers l'usage de mon API.

Notes et remarques personnelles :

/

9.2.15 Semaine 15

Objectif(s) :

- Suite de l'implémentation du dashboard/image dynamique

- Maintenance des services

- correction de rapport de stage

Compte rendu explicatif :

Durant cette semaine, je me focalisé à créer une petite API qui permet d'afficher dans une petite page web, la pression courante actuelle du capteur et de l'incorporer dans la page web en utilisant le framework Flask.

Ensuite, je devais effectuer des vérifications que mes services tournent parfaitement au niveau du serveur car j'ai déjà eu une panne donc je regarde de temps en temps. Notamment, en consultant les logs sur Portainer et d'appliquer des corrections en cas de bugs.

Ensuite, je devais contrôler que mes services tournent parfaitement au niveau du serveur car j'ai déjà eu de temps en temps des pannes.

Finalement, j'ai corrigé la plupart des erreurs qui se trouvaient dans mon rapport de stage.

Notes et remarques personnelles :

/

9.3 Evaluation continue

HE2B ESI

STAGE EN INFORMATIQUE : EVALUATION CONTINUE DU STAGE
PAR LE RESPONSABLE AU SEIN DE L'ORGANISME.

Document du stagiaire

Nom de l'Organisme Fabricom (Equans)
Personne qui évalue le stage
Nom, Prénom Geoffrey Anthoon
Nicolas Ruhland
Tél/GSM 0478832518

Concerne l'étudiant: Nom Ezeagwula Prénom Emmanuel

Echelle de valeurs proposées
Insuffisant --- Faible -- Très Moyen - Moyen M Bon + Très bon ++ Excellent +++

Au terme du stage l'étudiant a atteint les seuils de compétences suivants :

	mi Mars	mi Avril	mi Mai
FORMATION			
Connaissances théoriques	--- -- - M **+** ++ +++	--- -- - M **+** ++ +++	--- -- - M **+** ++ +++
Analyse	--- -- - M **+** ++ +++	--- -- - M **+** ++ +++	--- -- - M + **++** +++
Pratique de la programmation	--- -- - **M** + ++ +++	--- -- - **M** + ++ +++	--- -- - **M** + ++ +++
TRAVAIL FOURNI			
Quantité de travail fourni	--- -- - M **+** ++ +++	--- -- - M **+** ++ +++	--- -- - M **+** ++ +++
Qualité du travail fourni	--- -- - M **+** ++ +++	--- -- - M **+** ++ +++	--- -- - M **+** ++ +++
Adaptabilité au système et à l'environnement	--- -- - M + **++** +++	--- -- - M + **++** +++	--- -- - M + **++** +++
Capacité d'auto+apprentissage	--- -- - M + **++** +++	--- -- - M + **++** +++	--- -- - M + **++** +++
Capacité d'organisation	--- -- - **M** + ++ +++	--- -- - **M** + ++ +++	--- -- - **M** + ++ +++
COMPORTEMENT			
Aptitudes à la communication	--- -- - M **+** ++ +++	--- -- - M + **++** +++	--- -- - M + **++** +++
Sociabilité	--- -- - M **+** ++ +++	--- -- - M **+** ++ +++	--- -- - M **+** ++ +++
Réceptivité aux recommandations	--- -- - M **+** ++ +++	--- -- - M **+** ++ +++	--- -- - M **+** ++ +++

a une vue plus complète et globale qu'au début du stage

Comment remplir la grille
Après une période de +/- 4 semaines, un entretien est réalisé entre le superviseur du stage et le/la stagiaire. Chacun des items de cette grille sont évalués et explicités par le superviseur.
Le/la stagiaire est tenu de prendre note des remarques émises en les transcrivant dans son "Carnet de bord de Stage".

Emmanuel a bien répondu à la problématique que nous lui avons fourni pour son stage, il a su surmonter les problèmes qu'il a rencontré et a proposé des solutions adéquates, avec une bonne communication dans l'ensemble.