

Rapport de Conception de Systèmes Cyber-Physique

Appel des présences semi-autonome multi dispositifs

Réalisé par

VIDAL-MAZUY Antoine
MARYNOWICZ Michael
BEAUCHET Quentin
FORNER Yann

Groupe 6

Encadré par

TIGLI Jean-Yves
LAVIROTTE Stéphane
COLOMB

Table des matières :

1. Introduction	3
2. Analyse centrée utilisateur	3
2.1. Problématiques utilisateurs :	3
2.2. Personas	3
2.2.1. Persona d'un professeur habile avec les nouvelles technologies :	4
2.2.2. Persona d'un professeur réticent aux nouvelles technologies:	5
2.2.3. Persona d'un élève :	6
2.2.4. Persona d'une employée de la Scolarité :	7
2.3. Services BluePrint :	7
2.3.1. Pour un professeur	8
2.3.2. Pour une secrétaire	10
2.3.3. Pour un étudiant	12
2.4. Pain Point :	14
2.5. Scénarios :	15
3. Architecture logicielle des prototypes	16
4. Liste des Services et des Équipements mis en œuvre et éléments techniques	17
4.1. Résumé de la liste des services et équipements mis en œuvre	17
4.1.1. Reconnaissance Faciale	17
4.1.2. Récupération des informations sur les cours depuis HyperPlanning	17
4.1.3. Lecteur de carte RFID	17
4.1.4. Détection des appareils étudiants par bluetooth	17
4.1.5. Base de données des étudiants	17
4.1.6. Génération de QR Code	17
4.1.7. Hotspot Wifi Privée	18
4.1.8. Notification à un étudiant	18
4.2. Descriptions techniques de services	18
4.2.1. Service de Détection par Bluetooth	19
4.2.2. Service Hyperplanning	19
4.2.3. Service de Reconnaissance Faciale - Caméra Netatmo	19
5. Achats	20
5.1. Liste des équipements à acheter	20
6. Composition logicielle de services (Cf. dessus)	20
6.1. Quelle plateforme de composition (i.e. de programmation n'impliquant que des clients vers les services du bus décrit ci-dessus) choisissez-vous ? (Node-Red, autre ?)	20
6.2. Décrire (avec logigramme, statechart, ...), la logique de composition entre les requêtes aux services du bus qui implémente l'application de votre projet.	20
7. Sprints du projet	21

1. Introduction

Dans le cadre de ce projet nous allons développer un prototype d'appel semi-autonome multi-dispositifs. Nous allons explorer plusieurs méthodes d'authentification des élèves. Certaines de ses méthodes sont facilement adaptable à une mise à l'échelle, d'autres représentent une preuve de concept.

L'appel des présences semi-autonome multi-dispositifs a pour objectif de faciliter la tâche des professeurs souhaitant faire l'appel durant leurs cours. Grâce à l'utilisation de multiples dispositifs, tels que le bluetooth, le wifi, une caméra ou encore un lecteur de carte.

Notre but est de détecter la présence d'un élève dans une classe. Cet appel doit être fiable et ne doit pas représenter une barrière à l'utilisation. Que ce soit d'un point de vue des professeurs, des élèves ou encore de la scolarité.

Grâce aux emplois du temps d'Hyperplanning, les feuilles de présence seront générées et remplies automatiquement. De plus, la fiche d'appel n'est transmise à la scolarité qu'après la validation par le professeur.

2. Analyse centrée utilisateur

2.1. Problématiques utilisateurs :

Ce projet va impacter plusieurs groupes de personnes. On a tout d'abord les 2 groupes d'utilisateurs primaires qui sont les professeurs ainsi que les étudiants. On a ensuite les utilisateurs secondaires tels que la scolarité. Nous allons étudier comment notre projet va s'intégrer à leurs habitudes et quel impact il aura une fois mis en place.

D'un côté les professeurs veulent que l'appel ne prenne pas plus de temps qu'à l'heure actuelle. Il doit être fiable et vérifiable par ce dernier.

D'un autre côté, les élèves doivent être assurés d'avoir leur présence correctement notifiée. A contrario, être notifié absent s'ils n'ont pas été présents lors d'un cours.

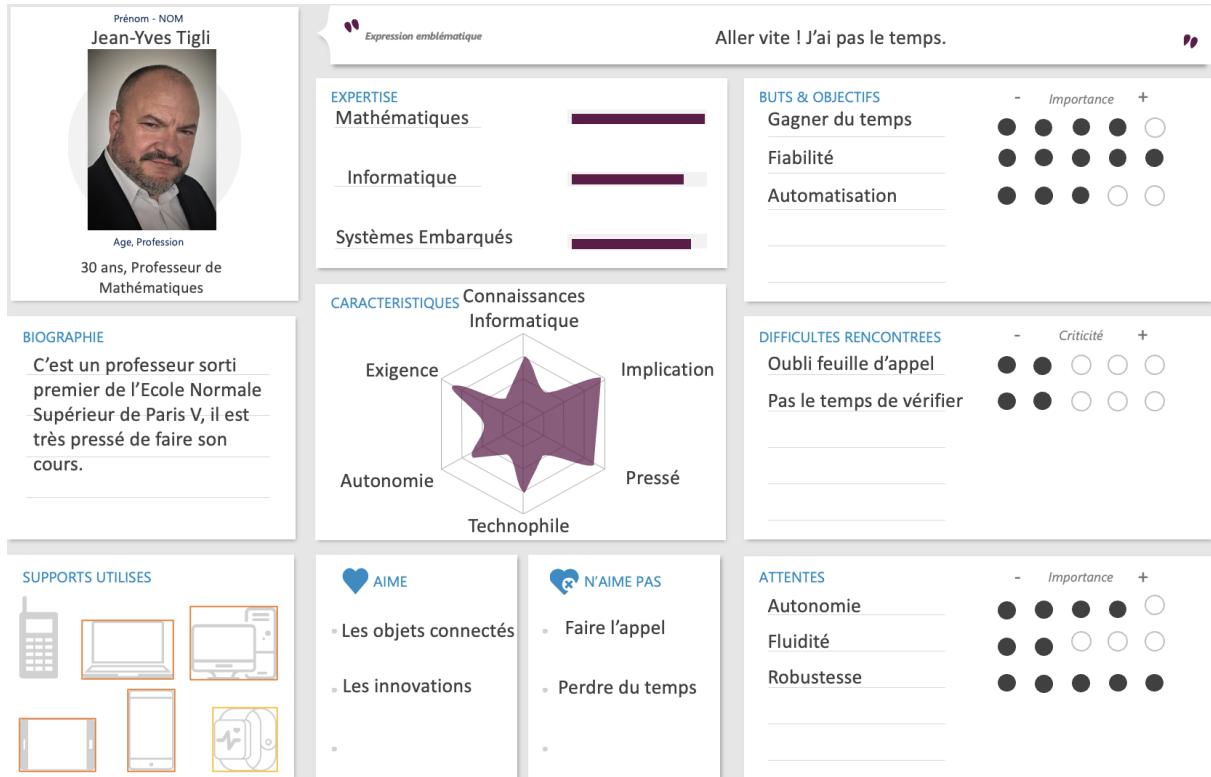
De plus, le travail effectué par la scolarité ne doit pas être impacté, et ne doit pas se complexifier. On cherche au contraire à réduire la charge de travail des feuilles d'appels.

2.2. Personas

Nous allons présenter quatre personas différents. Deux professeurs, un élève et une employée de la scolarité. Chacun possède une particularité et représente la logique métier d'un des trois groupes.

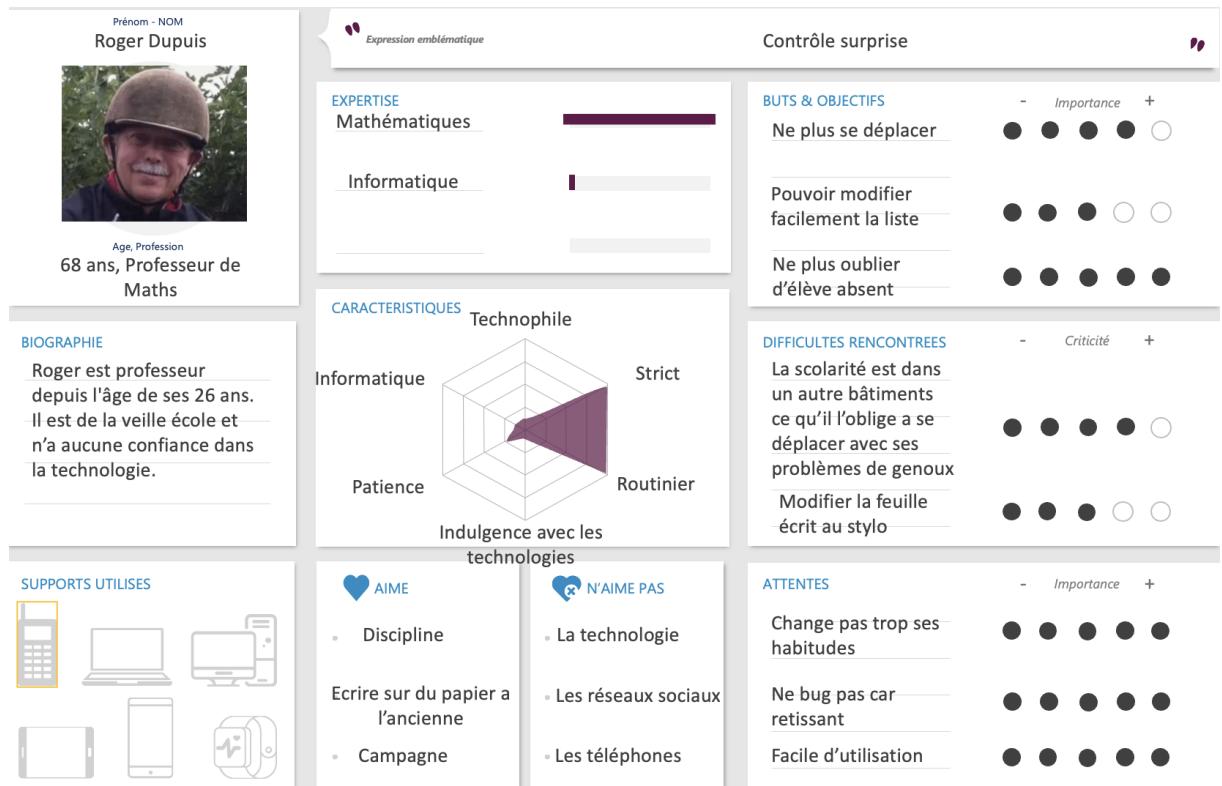
2.2.1. Persona d'un professeur habile avec les nouvelles technologies :

Pour ce premier persona nous prenons comme exemple un professeur habile avec les nouvelles technologies. Ce persona est un exemple type des utilisateurs primaire ciblés :



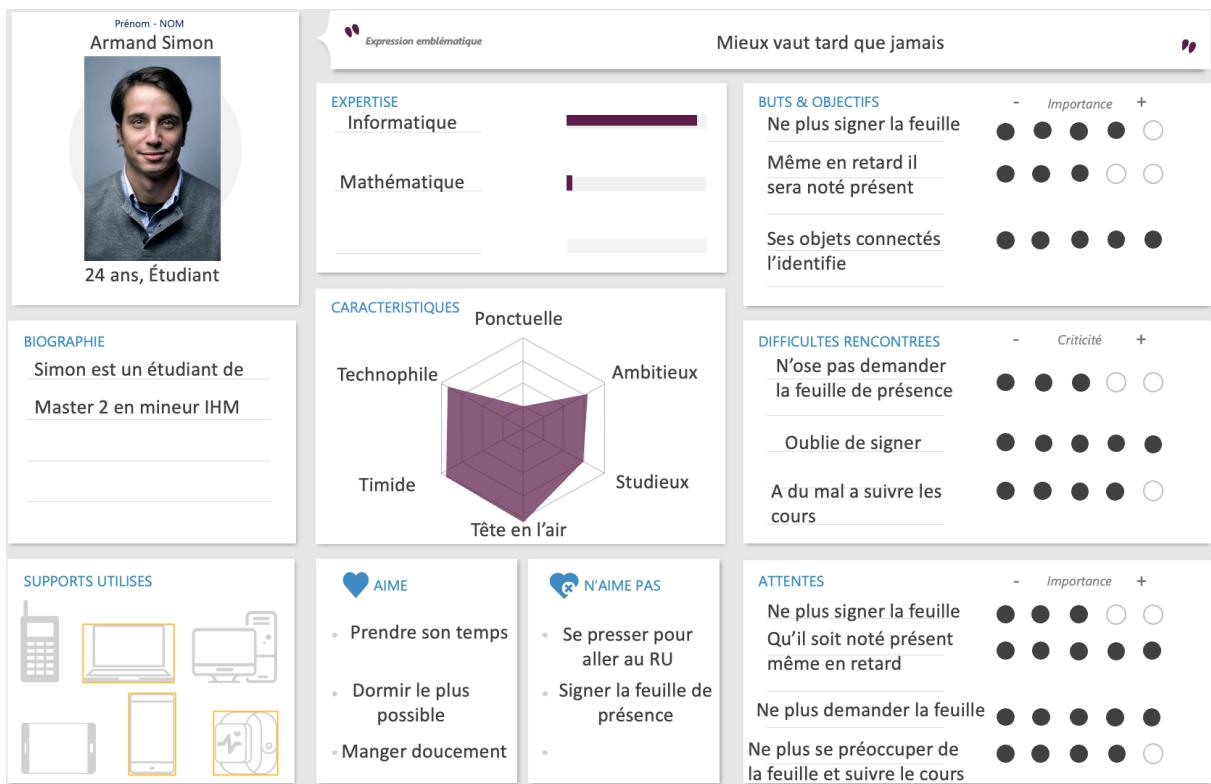
2.2.2. Persona d'un professeur réticent aux nouvelles technologies:

Ce personnage représente un deuxième type de professeur. Ce dernier est réticent aux nouvelles technologies et il n'est pas à l'aise avec celles-ci. Ce personnage est intéressant car il représente un réel défi. D'une part afin de le convaincre d'utiliser notre application. D'autre part, il faut que notre application soit la plus simple pour lui étant donné ses faibles compétences dans les nouvelles technologies.



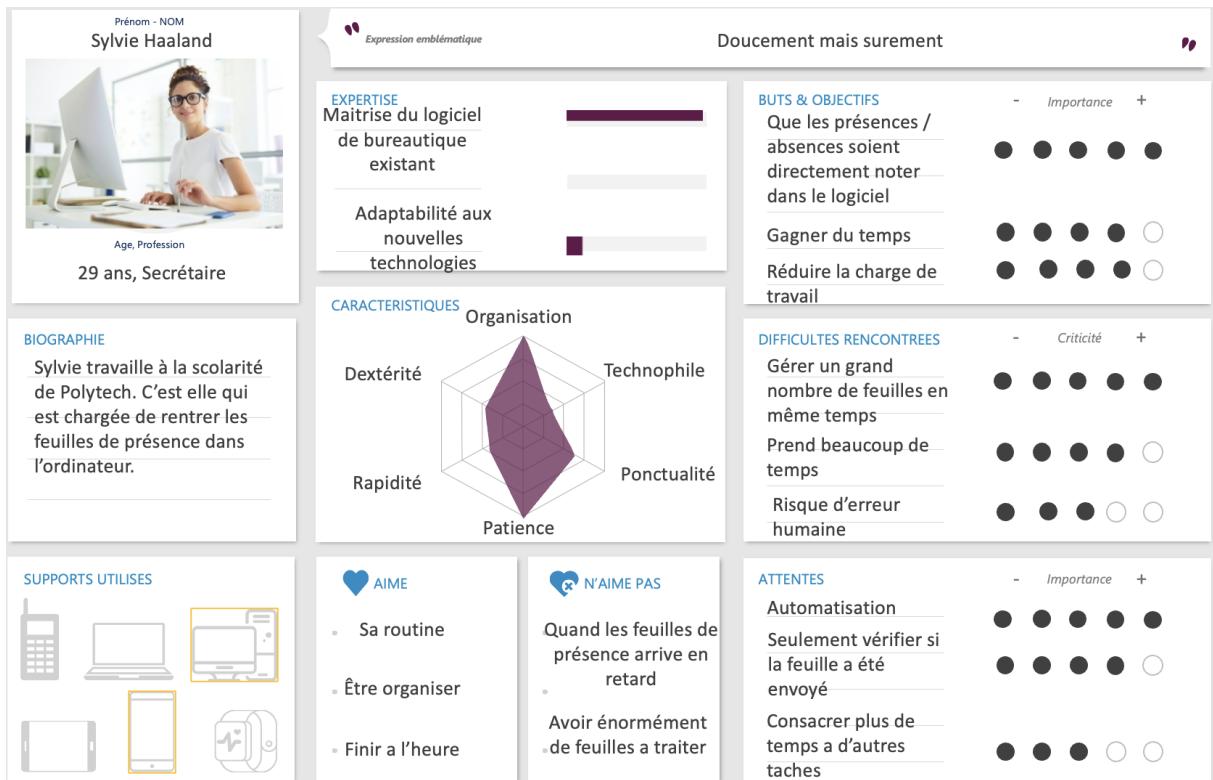
2.2.3. Persona d'un élève :

Armand est le persona qui représente un élève tête en l'air qui oublie souvent de signer la feuille de présence. Il sera donc utilisateur primaire.



2.2.4. Persona d'une employée de la Scolarité :

Sylvie est le persona qui représente la scolarité. Ce personna est intéressant car c'est un utilisateur secondaire. Il permet de montrer les difficultés actuelles que rencontre la scolarité et leurs attentes.



2.3. Services BluePrint :

Maintenant que nous avons défini nos personas, nous allons pour chacun d'entre eux, comparer leur quotidien, que ce soit avant ou après la mise en place de notre solution. Avant la mise en place, nous l'appelons "Service BluePrint as is", après la mise en place, "Service BluePrint to be". Grâce à ses différents pointeurs, nous allons pouvoir définir le cadre de notre sujet et présenter l'utilisabilité de notre solution.

2.3.1. Pour un professeur

Service BluePrint as is		Comme tous les jours, le professeur doit faire l'appel manuellement. Il perd du temps sur son cours en le faisant.	
	AVANT	PENDANT	APRES
OBJECTIFS	Récupérer la liste des étudiants.	Faire l'appel manuellement.	Rendre compte des personnes absentes.
ACTIONS	Doit aller chercher à la scolarité une feuille de présence	Appeler chaque étudiant.	Retourner à la scolarité pour donner la feuille.
POINTS DE CONTACTS	Scolarité	Papier dans la salle	Scolarité
EMOTIONS	 		 
OPPORTUNITES	Ne pas se déplacer à la scolarité, en l'ayant sur l'hypérplanning	N'a plus besoin de le faire, car c'est automatisé. Ou alors le faire partiellement.	Ne pas se déplacer à la scolarité, en l'envoyant à la scolarité automatiquement.
MOYENS d'INTERACTION	Déplacement en personne.	En parlant dans la salle.	Déplacement en personne.
Ligne de visibilité			
ACTIONS BACKSTAGE	La scolarité imprime la feuille de présence.	Les élèves signent la feuille de présence.	La scolarité, rentre les informations dans le système.
AUTRES ACTEURS	Scolarité.	Élèves.	Scolarité.
ETAT	Manuel ou papier.	Manuel ou papier.	Papier et informatique.
Outils internes			
OUTILS	Ordinateur et papier, Hypérplanning, imprimantes	Papier.	Ordinateur et papier, Hypérplanning

 Jean-Yves, Tigli	Service BluePrint to be		Comme tous les jours, le professeur doit faire l'appel manuellement. Il perd du temps sur son cours en la faisant.
	AVANT	PENDANT	APRES
	1	2	3
OBJECTIFS	Récupérer la liste des étudiants.	Faire l'appel manuellement.	Rendre compte des personnes absentes.
ACTIONS	Ne se préoccupe pas de l'appel.	Ne se préoccupe pas de l'appel.	Vérifier la liste d'appel
POINTS DE CONTACTS	Aucun	Aucun	Ordinateur / Smartphone
EMOTIONS	 		 
SOLUTIONS	Récupération automatique de la liste d'appel.	Système d'appel autonome à multi-facteurs (reconnaissance faciale, bluetooth...)	Vérifier sur un ordinateur / smartphone, et valider la feuille de présence.
MOYENS d'INTERACTION	Aucun	Aucun	Ordinateur / Smartphone
Ligne de visibilité			
ACTIONS BACKSTAGE	Récupération automatique de la liste d'appel.	Analyse de présence multi-facteurs Reconnaissance Faciale Badge étudiant Bluetooth Wifi Validation	Reception de la liste final par le système informatique de la scolarité.
AUTRES ACTEURS	Serveurs	Serveurs, RaspberryPI, Élèves	Serveurs
ETAT	Informatique	Informatique et humain	Informatique
Outils internes			
OUTILS	- Serveurs - Hyperplanning - Base de donnée	- Serveurs - Différents devices de reconnaissance - Application mobile	- Ordinateur / Smartphone - Serveurs

2.3.2. Pour une secrétaire

		Service BluePrint as is		
		AVANT	PENDANT	APRES
SYNTHÈSE	OBJECTIFS	Travailler sur ses tâches	Travailler sur ses tâches	Rendre compte des personnes absentes.
	ACTIONS	Imprimer feuille de présence	Travaille	Rentre manuellement les absences sur HyperPlanning.
	POINTS DE CONTACTS	Elèves / Scolarité		HyperPlanning
	EMOTIONS	 		
	OPPORTUNITÉS	Economiser du papier et du temps		Gagner du temps
	MOYENS d'INTERACTION	Remise de la feuille en main propre à un professeur ou un étudiant		Remise de la feuille en main propre à un professeur ou un étudiant
	Ligne de visibilité			
SYNTHÈSE	ACTIONS BACKSTAGE	Le professeur se déplace	Les élèves signent la feuille de présence.	Le professeur se déplace
	AUTRES ACTEURS	Le professeur	Élèves.	Le professeur
	ETAT	Manuel ou papier + imprimante	Manuel ou papier.	Papier et informatique.
Outils internes				
SYNTHÈSE	OUTILS	Ordinateur et papier, Hyperplanning, imprimantes		Ordinateur et papier, Hyperplanning

 <p>Sylvie Haaland</p>	Service BluePrint to be		Comme tous les jours, la scolarité doit imprimer des feuilles de présences pour chaque cours puis les rentrer dans HyperPlanning.
	AVANT	PENDANT	APRES
	1	2	3
	OBJECTIFS	Travailler sur ses taches	Travailler sur ses taches
	ACTIONS	Travaille	Travaille
	POINTS DE CONTACTS		HyperPlanning
	EMOTIONS		
Ligne de visibilité			
ACTIONS BACKSTAGE	Envoie automatique de la liste des étudiants	Remplissage de la liste de présence par le professeur	Envoie dans la BDD des étudiants absents / présents
AUTRES ACTEURS	Serveurs	Professeurs	Applications / Serveurs
ETAT	Informatique	Informatique et humain	Informatique et humain
Outils internes			
OUTILS	-Serveurs -Base de données -HyperPlanning	-Ordinateurs -Applications -Différents devices de reconnaissance pour détecter la présence	-Serveurs -Base de données -HyperPlanning

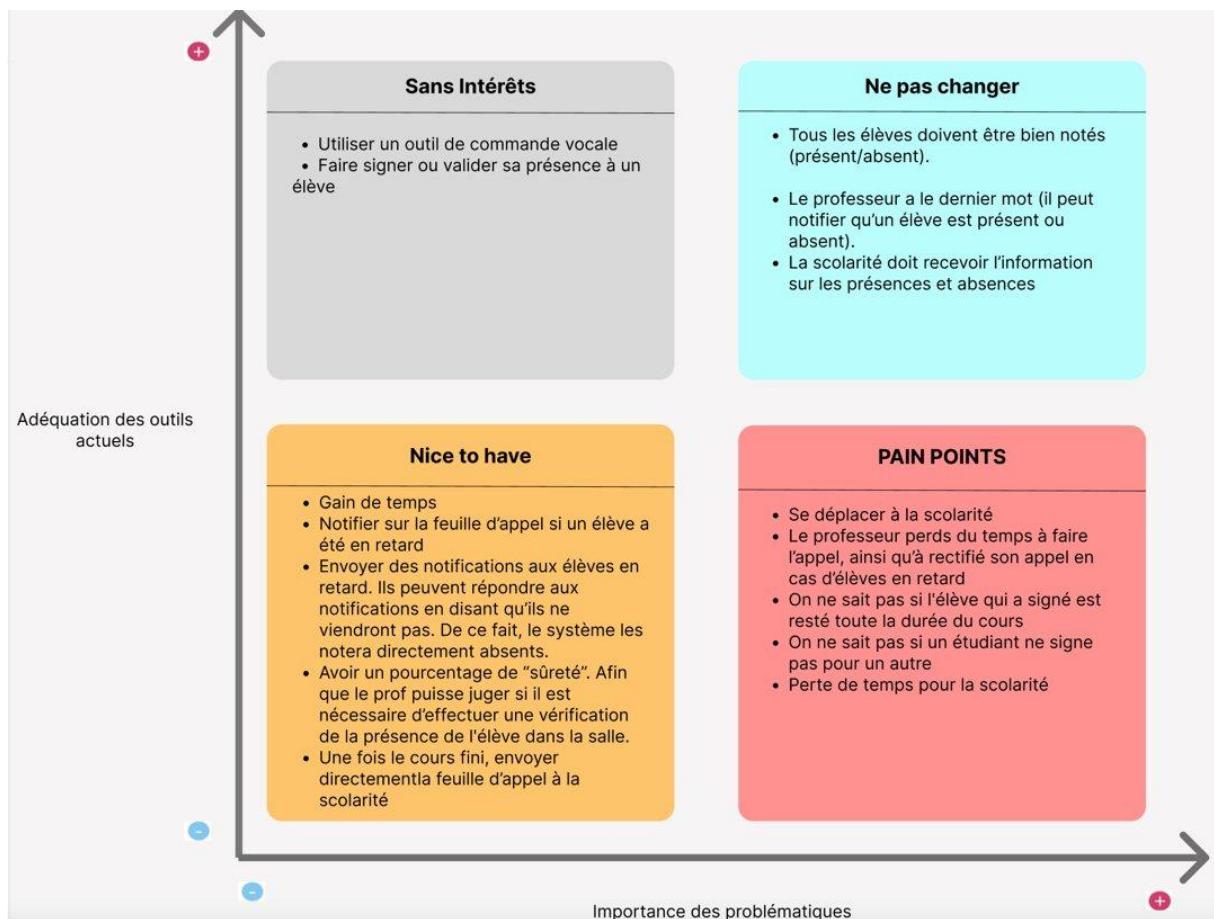
2.3.3. Pour un étudiant

Service BluePrint as is		Comme tous les jours, Simon doit signer la feuille de présence pour chaque cours.		
		AVANT	PENDANT	APRES
OBJECTIFS	Penser à signer la feuille	Trouver la feuille		
ACTIONS	Demander la feuille	Signer la feuille		
POINTS DE CONTACTS		Elèves		
EMOTIONS	 			
OPPORTUNITES	Ne plus penser à signer	Ne plus penser à signer		
MOYENS d'INTERACTION		Prendre la feuille en main propre à un professeur ou un étudiant		
Ligne de visibilité				
ACTIONS BACKSTAGE	Scolarité qui imprime la feuille et l'étudiant responsable qui la récupère	Les élèves signent la feuille de présence.	Le professeur se déplace	
AUTRES ACTEURS	Le professeur / ou l'élève responsable , la scolarité	Élèves.	Le professeur	
ETAT	Manuel ou papier + imprimante	Manuel ou papier.	Papier et informatique.	
Outils internes				
OUTILS	Ordinateur et papier, Hypperplanning, imprimantes	Stylo, papier	Ordinateur et papier, Hypperplanning	

 Simon Armand	Service BluePrint to be <i>Comme tous les jours, Simon doit signer la feuille de présence pour chaque cours.</i>		
	AVANT	PENDANT	APRES
	1	2	3
OBJECTIFS	Se consacrer sur les cours	Se consacrer sur les cours	Se consacrer sur les cours
ACTIONS	Travailler	Travailler	Travailler
POINTS DE CONTACTS		-Scanner le QR code -Valider la présence si besoin	
EMOTIONS			
SOLUTIONS	La fiche d'appelle est automatiquement envoyé au professeur sur l'application. Donc plus besoin d'y penser	L'appelle est en partie automatique, donc plus besoin de s'en soucier	
MOYENS d'INTERACTION		Informatique	
Ligne de visibilité			
ACTIONS BACKSTAGE	Envoie automatique de la liste des étudiants	Remplissage de la liste de présence par l'application et le professeur	Envoie dans la BDD des étudiants absents / présents
AUTRES ACTEURS	Serveurs	Professeurs / Application (qui contient tous les facteurs)	Applications / Serveurs
ETAT	Informatique	Informatique et humain	Informatique et humain
Outils internes			
OUTILS	-Serveurs -Base de données -HyperPlanning	-Ordinateurs -Applications -Différents devices de reconnaissance pour détecter la présence	-Serveurs -Base de données -HyperPlanning

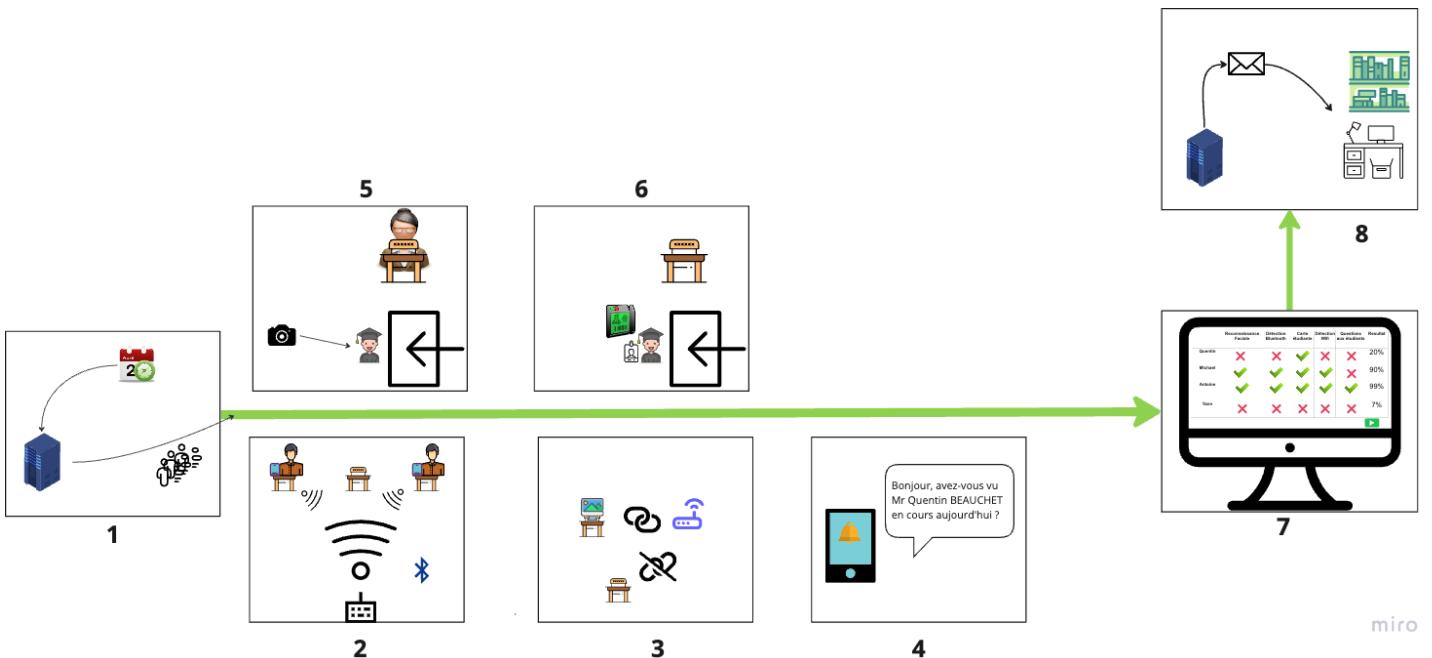
2.4. Pain Point :

Dans le cadre de la mise en place de notre projet, nous avons défini une matrice des différentes problématiques utilisateurs rencontrées. Cette matrice définit les cadres du projet. Elle permet de préserver les aspects actuels et de définir les réelles problématiques utilisateurs.



2.5. Scénarios :

Pour illustrer la mise en place du projet, nous avons fait un scénario qui couvre l'ensemble du système : du démarrage du cours jusqu'à la validation de la fiche d'appel à la fin d'un créneau de cours.



1. Le Serveur récupère la liste des cours sur l'hyper planning et y associe les données sur les étudiants censés être présents pour les envoyées aux différents moyens de reconnaissance.
2. Déetecter les appareils personnels à proximité pour savoir quels sont les élèves présents.
3. Lier un ordinateur au réseau wifi pour savoir qui est présent.
4. Notifier un étudiant et lui demander s'il a vu un certain élève.
5. La caméra essayera de reconnaître la tête de l'étudiant passant la porte.
6. L'élève scanne sa carte étudiante avant de rentrer en cours.
7. Le professeur verra apparaître un récapitulatif des étudiants ainsi que du pourcentage de chance que l'élève soit réellement présent en cours, il pourra ainsi l'envoyer si besoin est.
8. Le serveur récupère les données à la fin du cours et les enverra à la scolarité.

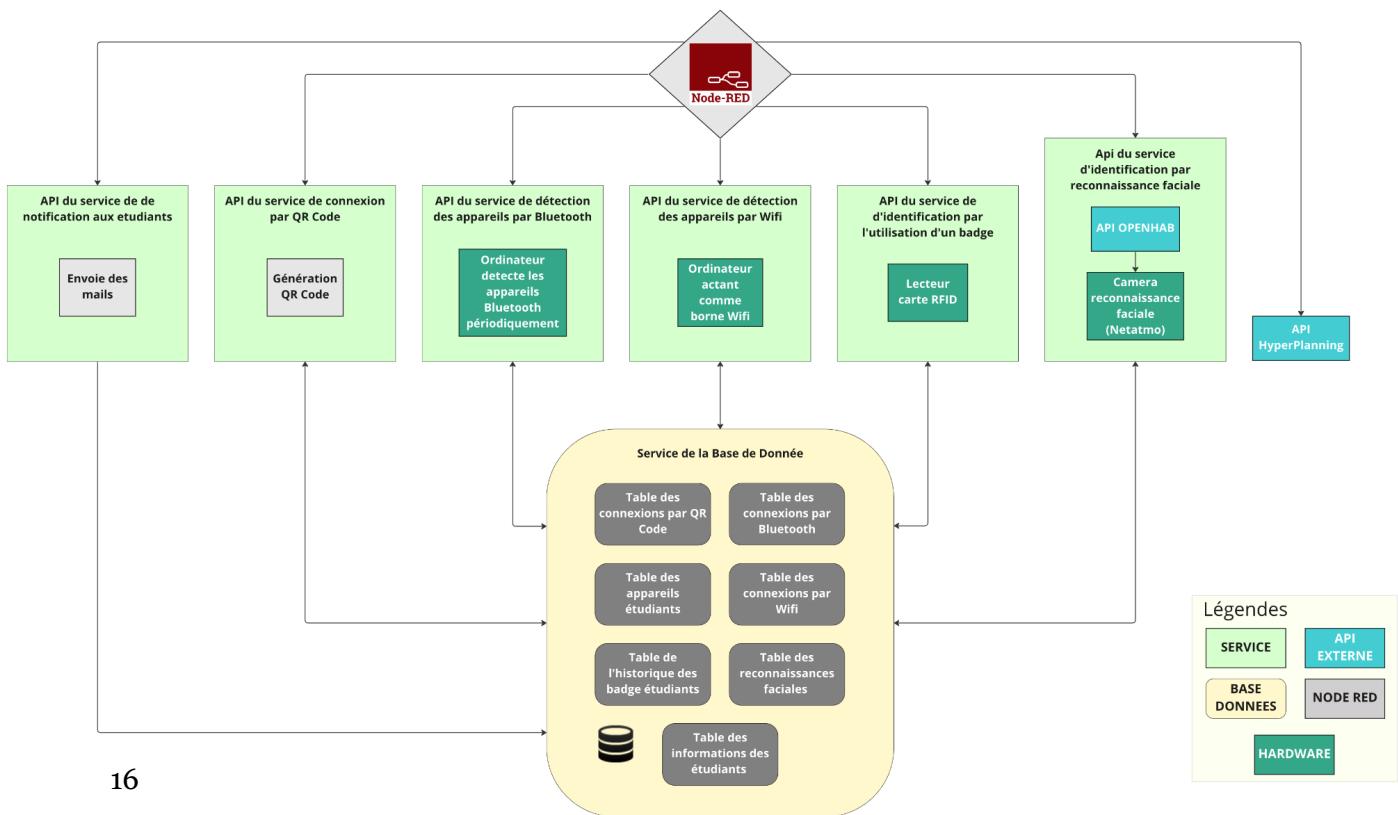
3. Architecture logicielle des prototypes :

L'architecture s'appuie sur une approche orientée microservices en utilisant Node Red. L'implémentation de la partie métier de l'application est alors une composition de service ne contenant que des clients de services du bus. A chaque élément de l'architecture logicielle sont associés des choix technologiques. L'interface client est quant à elle comprise dans la partie Node Red.

Chaque micro service offre une API qui permet:

- D'accéder à la caméra de reconnaissance faciale depuis OpenHab, ce qui nous permettra d'accéder à un historique de détection des étudiants.
- D'accéder au planning des différents cours depuis Hyperplanning.
- Récupérer un historique des événements capté par le lecteur de cartes RFID.
- Récupérer un historique des appareils connectés au réseau wifi privé.
- Récupérer un historique des appareils détectés sur le réseau bluetooth.
- De stocker et récupérer les informations dans une base de données en ligne, Node Red ne peut pas directement interagir avec la BD, il consulte ces données au travers des autres micro services.
- Récupérer les QR codes uniques ce qui permet aux étudiants de s'authentifier.
- De notifier les étudiants pour leur permettre de valider leur absence ou de signaler un retard.

Chaque micro-service est indépendant et fonctionne de manière autonome. Chaque événement (détection bluetooth, scan de carte, etc) sera géré de manière interne au service. Une fois tous les éléments stockés dans la base de données, nous pourrons par la suite récupérer les informations depuis Node-Red via les différents micro-services.



4. Liste des Services et des Équipements mis en œuvre et éléments techniques

4.1. Résumé de la liste des services et équipements mis en œuvre

4.1.1. Reconnaissance Faciale

Nous allons positionner la caméra Netatmo à l'entrée de la salle, celle-ci permet d'effectuer une reconnaissance faciale sur plus d'une trentaine d'étudiants. Nous allons utiliser le service d'Openhab pour accéder à la caméra.

Afin de déterminer la durée de présence des élèves, nous pouvons aussi rajouter une deuxième caméra qui sera positionnée au niveau de la sortie afin de détecter si un élève sort avant la fin du cours.

La caméra va de manière périodique récupérer les derniers événements via l'API d'OpenHab. Les éléments ainsi récupérés, seront stockés dans la base de données associées.

Nous allons ajouter une surcouche de plus haut niveau à ce service nous permettant de traiter les données issus de la caméra. Ainsi, si l'on veut accéder à aux données, nous allons envoyer une requête à l'api via **/camera/time/{secondes}/duration/{secondes}** qui nous permettra de récupérer une liste d'étudiants selon une intervalle de temps donnée avec leurs présences.

4.1.2. Récupération des informations sur les cours depuis HyperPlanning

Depuis l'API HyperPlanning nous allons pouvoir récupérer la liste des étudiants qui doivent être présents pour chaque cours. Nous allons aussi l'utiliser pour envoyer la feuille d'appel une fois que le professeur laura validé.

4.1.3. Lecteur de carte RFID

Nous allons positionner un lecteur de carte RFID à l'entrée de la salle pour permettre aux étudiants de badger en entrant dans celle-ci. Grâce au service que nous mettrons en place, nous pourrons récupérer la liste de tous les étudiants ayant badgé sur une période donnée.

À chaque événement détecté par le capteur RFID, le service s'occupera de stocker dans la base de données.

Pour ce service aussi, nous aurons une surcouche permettant de traiter ces données, depuis **/rfid/time/{secondes}/duration/{secondes}** nous pourrons récupérer la liste des étudiants ayant badgés dans une intervalle de temps donnée.

4.1.4. Détection des appareils étudiants par bluetooth

Un appareil Bluetooth dans la salle va scanner de manière périodique les équipements Bluetooth présents dans la salle pendant un créneau de cours. A chaque scan, le service va stocker les informations de présence dans la table associée de la base de données.

Nous allons créer une surcouche de plus haut niveau qui s'occupera de récupérer les données de présence lissée en fonction du temps. Nous y aurons accès via **/bluetooth/time/{secondes}/duration/{secondes}** afin d'avoir la liste des appareils détectés dans une intervalle de temps donnés. Un lissage est obligatoire dans le cas du bluetooth pour éviter les données incohérentes.

4.1.5. Base de données des étudiants

Cette base de données a pour but de lier un étudiant à divers biens lui appartenant. Nous pourrons lier un étudiant à un appareil grâce à son adresse MAC, celle-ci sera renseignée lors de la première connexion par le biais d'un QR code. Nous pourrons aussi lier un étudiant à une carte grâce au lecteur de carte RFID.

4.1.6. Génération de QR Code

Nous allons générer des QR Code uniques qui se renouveleront de manière périodique pour chaque cours. Les étudiants pourront les scanner puis s'authentifier avec leurs identifiants étudiants. Une fois authentifié un service de plus haut niveau permettra de récupérer les événements scannés, ce service nous donnera aussi la possibilité d'accéder la liste des étudiants ayant scanné le QR Code dans une intervalle de temps via **/qrCode/time/{secondes}/duration/{secondes}**.

4.1.7. Hotspot Wifi Privée

Grâce à une borne wifi privée que les étudiants pourront utiliser pour se connecter à internet, nous pourrons récupérer leurs adresses MAC et les identifier par le même principe que notre service Bluetooth. Il reposera aussi sur l'utilisation d'une surcouche pour accéder à ses données et nous pourrons consulter la liste des étudiants détectés sur le wifi dans une intervalle données via **/wifi/time/{secondes}/duration/{secondes}**.

4.1.8. Notification à un étudiant

Dès lors qu'un étudiant est détecté absent nous allons lui envoyer un mail pour savoir s'il compte venir ou s'il sera absent pour la totalité du cours.

4.2. Descriptions techniques de services

Service Name	Équipement associé O/N ? Si O (oui) rajouter le Nom et une URL de référence	URL du service	API et URL de documentation	Techno (MQTT, REST, AMQP etc.)	SDK avec codes sources clients (URL)	Via Openhab (O/N)
CameraNetatmo ¹	Oui https://www.openhab.org/addons/bin/dings/netatmo/	/camera	https://dev.netatmo.com/apidocumentation/security	REST		Oui
HyperPlanning	Non	/edt	https://www.index-education.com/fr/hyperplanning-info196-service-web.php	REST		Non
Lecteur de carte	Oui Un lecteur de carte RFID si Mr Lavirotte arrive à nous en fournir un	/rfid		REST		Non

Detection bluetooth	par	Oui Arduino avec une carte réseau	/bluetooth	https://www.npmjs.com/package/node-bluetooth	REST		Non
Hotspot Wifi	Privé	Oui Borne Wifi	/wifi	Dépendra de la borne Wifi utilisé			Non
Notification mail		Non	/mail	https://www.npmjs.com/package/mysql https://nodemailer.com/about/ https://nodejs.org/en/	SMTP		Non
Génération de QR Code		Non	/qr	https://nodejs.org/en/ https://www.npmjs.com/package/qrcode	REST		Non
Base de donné		Non	/db	https://dev.mysql.com/doc/	REST		Non

4.2.1. Service de Détection par Bluetooth

Nous utiliserons dans l'API les fonctions utilisées dans la photo ci-dessous. Ce qui nous permettra de récupérer les devices connectés au bluetooth.

find devices

```
device
.on('finished', console.log.bind(console, 'finished'))
.on('found', function found(address, name){
  console.log('Found: ' + address + ' with name ' + name);
}).scan();
```

will output

```
→ node-bluetooth git:(master) ✘ node example/index.js
Found: 22-22-a3-0d-63-09 with name Meizu MX4 Pro
Found: dc-2b-2a-82-76-29 with name Lsong's iPhone
Found: 38-bc-1a-37-2d-d4 with name MEIZU MX5
finished
```

4.2.2. Service Hyperplanning

Grâce à l'API d'HyperPlanning nous pourrons avoir accès à la liste des étudiants dans un cours. De plus, nous pourrons avoir accès à une photo de chaque étudiant afin de faire la reconnaissance faciale. Dans le cas où les informations des étudiants ne sont pas toutes accessibles (notamment la photo), il sera toujours possible de les ajouter manuellement.

Photo des étudiants			
PhotoEtudiant	Flux de la photo de l'étudiant de clé 'AEtudiant' au format 'AFormat'. La valeur de retour est vide si l'étudiant n'a pas de photo.	AEtudiant : THpSvcWCleEtudiant AFormat : THpSvcWFormatPhoto	base64Binary 2013.05.2

EtudiantsDuCours	Clés des étudiants du cours de clé 'ACours'	ACours : THpSvcWCleCours	THpSvcWTBlaueClesEtudiants	2007.0.6.1
------------------	---------------------------------------------	--------------------------	----------------------------	------------

4.2.3. Service de Reconnaissance Faciale - Caméra Netatmo

GET /getevents Returns all the events until the one specified in the request.

^

By default, it returns the 30 last events of the Home.
It can be combined with other parameters to go over the timeline of events:

- *person_id* = To retrieve events when a person was detected
- *device_id* = To retrieve events detected by a gateway
- *module_id* = To retrieve events detected by a module
- *offset* = Number of additional events to the one looked for

Required scope depending on the product:

- *read_camera* for Smart Indoor Camera and its accessory (Smart Siren, Smart Sensors)
- *read_presence* for Smart Outdoor Camera with or without Siren
- *read_doorbell* for Smart Video Doorbell
- *read_smokedetector* for Smart Smoke Detector
- *read_mx* for the BTicino Classe 300 EOS

L'API de la caméra, nous permet de récupérer les 30 derniers événements détectés. Plusieurs paramètres peuvent être donnés comme une personne en particulier ou encore un groupe de personnes

5. Achats

5.1. Liste des équipements à acheter

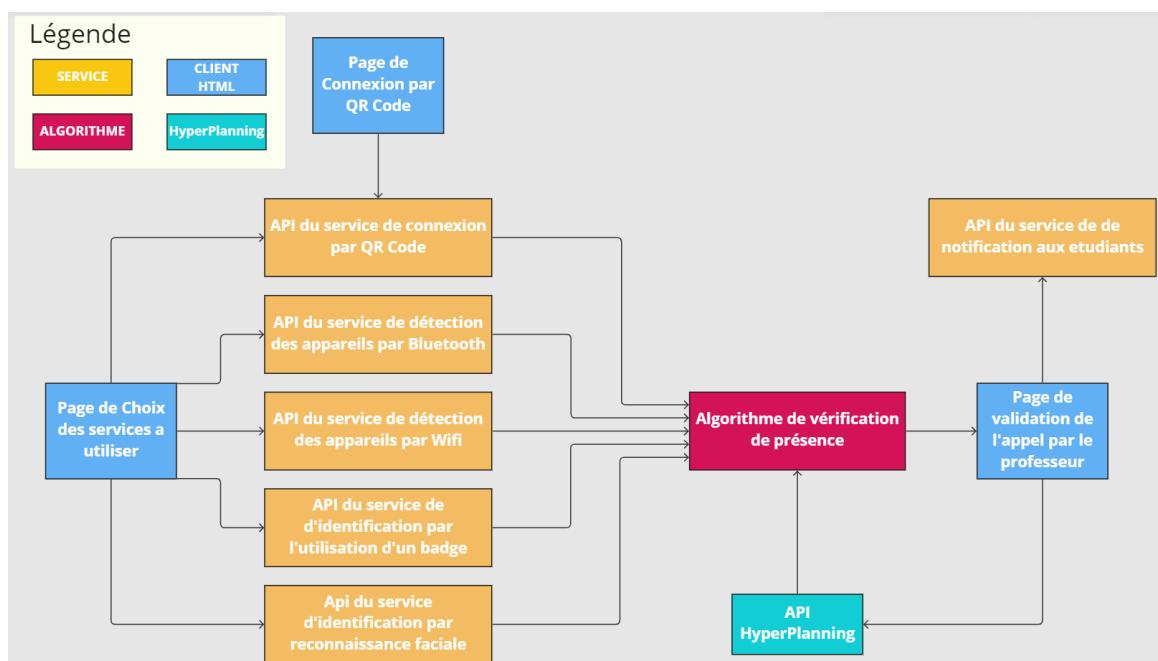
Equipement	Prix unitaire	Quantité	Référence	Fournisseur (URL)	Compatible OpenHab O/N ?
Lecteur carte RFID	16€	1		SIB Lecteur de contrôle d'accès de carte RFID de sécurité	Oui

6. Composition logicielle de services (Cf. dessus)

- 6.1. Quelle plateforme de composition (i.e. de programmation n'impliquant que des clients vers les services du bus décrit ci-dessus) choisissez-vous ? (Node-Red, autre ?)

Nous allons utiliser Node-Red car nous sommes sur un modèle de bus de service et nous l'avons déjà utilisé pour des anciens projets et nous avons donc quelques bases ce qui facilitera le développement.

- 6.2. Décrire (avec logigramme, statechart, ...), la logique de composition entre les requêtes aux services du bus qui implémente l'application de votre projet.



Le professeur accède à son interface client afin de choisir quels services il veut utiliser pour réaliser l'appel à l'aide de checkbox.

Une fois son choix réalisé, il peut lancer la recherche des informations dans les différents services sélectionnés grâce à leurs API, celles-ci sont ensuite transmises à l'Algorithme qui nous permet de déterminer le pourcentage de confiance de présence d'un élève. Pour cela l'algorithme va combiner ces informations avec la liste des étudiants récupérée depuis l'API HyperPlanning.

Le professeur va ensuite avoir sur sa page web une liste des étudiants avec leurs taux de confiance de leurs présence, il peut ensuite valider ou invalider un appel manuellement et une fois que celui ci lui est terminé il peut envoyer les informations vers l'API HyperPlanning qui se chargera de notifier la scolarité.

Tous les élèves notés absents se verront ensuite notifiés avec la possibilité de passer leurs présence en tant que retard qui deviendra effectif dès que sa présence sera détectée.

7. Sprints du projet

Les différents sprint que nous allons réaliser sont :

- Sprint 1 : Mise en place du service de récupération de la liste d'appels c'est-à-dire la validation de la feuille d'appel avec un simple client web. Mise en place de Node-Red.
- Sprint 2 : Mise en place de MVP des services d'authentification simples comme le bluetooth et le QR Code. Mise en place des bases de données respectives.
- Sprint 3 : Mise en place d'un MVP de détection d'équipements sur un hotspot wifi.
- Sprint 4 : Mise en place du service de reconnaissance faciale via l'API de la caméra Netatmo.
- Sprint 5 : Améliorations de MVP des services d'authentification existants. Mise en place dans Node Red des services existant
- Sprint 6 : Mise en place du service mail de notification aux étudiants. Développement d'un MVP du lecteur de carte RFID.
- Sprint 7 : Vérifications de l'efficacité des différents services existant.
- Sprint 8 : Améliorations des MVP restants, et s'assurer de la déployabilité des services.

Page de validation de l'appel par le professeur :

Liste d'appel

https://localhost:8080/appel

Bluetooth Wifi QR Code Camera Scan Carte

Search

Course : CSCP - 11/08/2022 - 13h30 to 17h45

Name	%	Presence
Surname Name	50	⚠️
Surname Name	100	✓
Surname Name	70	⚠️
Surname Name	100	✓
Surname Name	100	✓
Surname Name	0	✗
Surname Name	100	✓

Validate

Page de connexion des professeurs ainsi que pour l'authentification des étudiants après le scan du QR code :

