Université de Rennes

ESIR ECOLE SUPERIEURE D'INGENIEURS DE RENNES

Projet FabLab IoT -

Développement d'une interface de pilotage de périphérique domotique

Auteurs :
Thibault GUERINEL
Elise MÉTAYER
Ismael SYLLA
Aubry TONNERRE

Janvier 2024

Table des matières

Table des Figures

I. INTRODUCTION

Ce projet s'inscrit dans le cadre de notre formation en ESIR2 Spécialité Informatique parcours IoT. Il s'étalera sur toute l'année, nous consacrerons le semestre 7 à la gestion de projet et à sa conception et le semestre 8 à sa réalisation. Ce projet a pour but de développer l'esprit de synthèse, le travail en équipe, d'apprendre à gérer un projet dans son ensemble (organisation, respect des délais, satisfaction du client) et de développer de nouvelles compétences (méthodologies, langage de programmation, protocoles de communication).

Nous allons travailler sur le développement d'une interface de pilotage de périphérique domotique. Nous avons choisi de travailler sur ce projet car il permet un rendu concret grâce à l'interaction des objets du quotidien entre eux. Nous avions envie d'en apprendre plus sur la domotique qui est un domaine dans lequel nous pourrions travailler et qui est amené à perdurer.

La domotique est l'ensemble des techniques de l'électronique, de physique du bâtiment, d'automatisme, de l'informatique et des télécommunications utilisées dans les bâtiments. Elle a pour but d'être plus ou moins « interopérable » et de permettre de centraliser le contrôle des différents systèmes et sous-systèmes de la maison et de l'entreprise.

L'avantage principal de la domotique est sa capacité à faire d'importantes économies d'énergie à l'intérieur d'une maison. Si elle est bien conçue, une installation domotique permet de réaliser entre 25 et 30% d'économie énergétique par an. Elle vise à apporter des solutions simples et intuitives aux problématiques d'économie d'énergie, de sécurité et de confort en centralisant le contrôle des différents équipements d'une maison ou d'un bâtiment.

Nous allons commencer par présenter le projet, décrire le cahier des charges, les solutions techniques existantes et celles que nous allons utiliser. Nous allons ensuite présenter l'équipe projet et la planification de celui-ci. Nous terminerons sur une conclusion exposant l'état du projet, et nos ressentis personnels.

II. CADRAGE DE PROJET

a. PRÉSENTATION DU PROJET

Notre projet prendra la forme d'une application web permettant à son utilisateur de gérer les modules domotiques de sa maison. L'efficacité de l'application dépend en grande partie de la simplicité de son interface utilisateur. Notre problématique est donc de concevoir une interface intuitive et accessible, permettant aux utilisateurs de contrôler facilement et de comprendre rapidement le fonctionnement des différents modules domotiques connectés. C'est aussi la légèreté de notre application qui nous démarque car elle utilise peu de technologies. Nous avons pour objectif de créer une gestion de scénarios intuitive et personnalisable facilement avec des templates existants.

Notre projet se concentre spécifiquement sur l'intégration de modules communicants en Zigbee, une technologie sans fil économe en énergie et offrant une connectivité fiable.

b. CAHIER DES CHARGES

Au travers de cette partie, nous allons présenter le cahier des charges de notre projet. Nous aborderons les grands points de notre projet, son objectif, ses aspects et son périmètre.

i. Diagramme pieuvre :

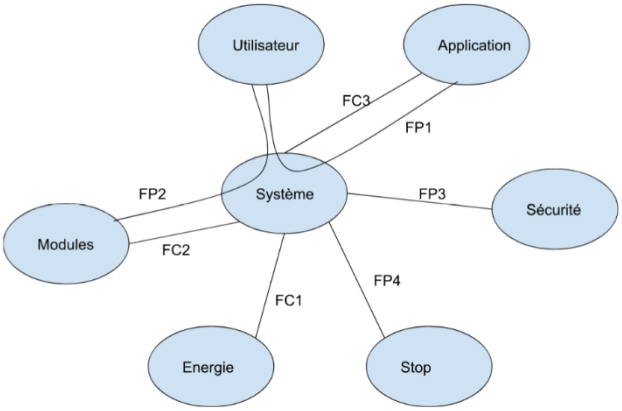


Figure 1 - Diagramme Pieuvre du projet

FP1 : Créer/activer des scénarios
FP2 : Contrôler les modules à distance (mais chez soi)
FP3 : Sécuriser la connexion (ne pas laisser à n'importe qui accéder aux actionneurs)
FP4 : Arrêt manuel d'urgence
FC1 : Être autonome (énergie)
FC2 : Ajout des différents modules sur l'appli
FC3 : Accessibilité (facile à utiliser)

ii. Objectif du projet

Nous allons répondre aux différentes questions du QQOQCP (Qui ? Quoi ? Où ? Quand ? Comment ? Pourquoi ?)

Qui : S'adresser aux adultes, et aux enfants. Mais le système est équipé d'une protection par mot de passe pour la réalisation de certaines tâches.

Quoi : Permettre la gestion de modules domotiques depuis son ordinateur.

Où : Accéder aux systèmes uniquement depuis la maison ou l'appartement.

Quand : Ensemble prêt pour juin 2024.

Comment : Accéder depuis une page web par un serveur exécuté sur un Raspberry.

Pourquoi : Contrôler aisément les éléments de sa maison et de les automatiser.

iii. Périmètre du projet

Le projet est destiné aux adultes sachant se servir d'un smartphone. Les enfants pourront l'utiliser mais avec un usage plus restreint afin de limiter leur pouvoir sur le système. Par exemple, on ne va pas vouloir que l'enfant change la température de la maison, mais il pourra allumer ou éteindre la lumière. L'utilisateur ne pourra pas utiliser tous les capteurs, il y aura une liste prédéfinie de capteurs qui seront compatibles avec notre système. Le serveur sera conçu pour être déployé sur un Raspberry. Le système aura pour but d'avoir une faible consommation énergétique. L'outil ne pourra être contrôlé que depuis un ordinateur.

iv. Description fonctionnelle des besoins

- Créer scénario

L'utilisateur pourra créer lui-même ses scénarios afin d'avoir un système qui répond parfaitement à son besoin. Il a donc la possibilité de créer un système entièrement personnalisable.

- Tableau de bord

L'utilisateur aura un résumé de ses équipements, des scénarios créés et de l'état de son système.

- Template scénario déjà existant

L'utilisateur pourra utiliser des templates de scénarios déjà existants s'il le souhaite.

- Arrêt d'urgence

Un bouton d'arrêt d'urgence sera aussi disponible dans l'application pour arrêter le système domotique, ainsi l'utilisateur pourra contrôler son environnement "manuellement".

c. ETUDE DE L'ART

L'objectif de cette partie va être de faire une étude sur l'existant. En effet, il existe déjà plusieurs solutions payantes ou open source qui permettent de piloter au travers d'une interface des périphériques IoT et d'en retirer des informations.
Nous nous intéresserons principalement à quatre applications ici à savoir Jeedom, Home Assistant ainsi que ESP-Home et tuya smart.

i. Jeedom

Figure 2 - Logo JeeDom

Jeedom (*Jeedom* [sans date]) se positionne comme une solution domotique **open source** axée sur la simplification du contrôle des capteurs IoT. Cette plateforme offre une **interface web** permettant la gestion et la **programmation de scénarios** pour automatiser diverses tâches. Développée avec une compatibilité étendue, Jeedom prend en charge diverses marques et normes de communication, assurant une intégration fluide avec un large éventail de dispositifs. En outre, elle offre des fonctionnalités de **sécurité avancées et de chiffrement** pour protéger le compte utilisateur et les communications permettent une personnalisation approfondie de l'interface.
Jeedom propose également des **solutions payantes** qui étendent encore davantage les fonctionnalités de la plateforme. Ces offres payantes fournissent des caractéristiques avancées telles que des plugins spécialisés, un support technique prioritaire et des mises à jour exclusives. Cette approche de modèle économique permet à Jeedom de soutenir le développement continu de sa plateforme tout en offrant aux utilisateurs une variété d'options pour personnaliser leur expérience en fonction de leurs préférences et de leurs besoins.

ii. Home Assistant

Figure 3 - Logo Home Assistant

Home Assistant (Assistant [sans date]) une deuxième solution dans le domaine de la domotique, se distingue par son approche **open source** centrée sur l'automatisation intelligente et la gestion centralisée des appareils connectés. Conçue pour être accessible à un large public, Home Assistant propose une interface utilisateur intuitive qui permet le contrôle facile des dispositifs IoT. Cette plateforme se caractérise par sa flexibilité, offrant la **compatibilité avec un large éventail de protocoles** et de marques comme pour Jeedom. Les utilisateurs peuvent créer des automatisations en utilisant un langage de script déclaratif. De plus, Home Assistant s'aligne sur les principes de la sécurité informatique en proposant des **fonctionnalités avancées de chiffrement** et de gestion des accès. Bien qu'Home Assistant soit à la base une solution open source. Il propose aussi des **abonnements** comme pour Jeedom proposant à l'utilisateur une assistance ainsi qu'un accès à des fonctionnalités en avance.

iii. ESP-Home

Figure 4 - Logo ESPHome

ESPHome (*ESPHome* [sans date]) se distingue en tant que **framework open source** dédié à la programmation et à la gestion des dispositifs IoT basés sur des microcontrôleurs **ESP8266 et ESP32**. Contrairement à Jeedom et Home Assistant qui sont des plateformes domotiques complètes, ESPHome se concentre spécifiquement sur la création et la gestion des firmwares pour les appareils connectés. L'un de ses principaux avantages réside dans sa simplicité et son **orientation vers le bricolage**, permettant aux utilisateurs de personnaliser le fonctionnement de leurs dispositifs selon leurs besoins spécifiques en utilisant le langage de configuration YAML.
La principale différence réside dans le niveau de complexité et d'extensibilité. Alors que Jeedom et Home Assistant offrent des solutions plus complètes avec des interfaces graphiques, des **automatisations avancées**, et des intégrations étendues, ESPHome s'adresse davantage aux utilisateurs qui souhaitent une approche plus légère et focalisée sur la création de firmwares pour des dispositifs spécifiques. Cela en fait une solution idéale pour les projets DIY et pour les personnes qui aiment avoir un contrôle direct sur le code de leurs objets connectés.

iv. Tuya Smart

Figure 5 - Logo Tuya

Tuya (*Tuya Smart - Global IoT Developer Service Provider* [sans date]) Smart est une plateforme IoT qui se consacre à fournir une connectivité intelligente à divers appareils domestiques et industriels. En tant que **fournisseur de services IoT**, Tuya Smart propose des outils de développement, tels que des modules matériels, des kits de développement logiciel (SDK) et une plateforme cloud, pour faciliter l'intégration de fonctionnalités intelligentes dans les produits. La polyvalence de la plateforme se manifeste par sa **compatibilité avec un large éventail de dispositifs**, tels que les éclairages, les thermostats, les caméras de sécurité, et bien d'autres. Les utilisateurs bénéficient d'une expérience centralisée grâce à l'application mobile Tuya Smart, permettant le contrôle à distance, l'automatisation des tâches et la personnalisation des scénarios. Tuya Smart s'est affirmé comme un acteur important dans la connectivité intelligente, en offrant des **solutions complètes** aux fabricants et aux utilisateurs finaux, contribuant ainsi à la création d'un écosystème solide d'appareils IoT interconnectés.

Nous avons examiné des solutions permettant à chaque utilisateur de déployer sa propre solution domotique en utilisant des applications accessibles et open source, avec la possibilité d'ajouter des services supplémentaires moyennant un coût financier.

d. VEILLE TECHNOLOGIQUE

i. Angular

Figure 6 - Logo Angular JS

AngularJS (*AngularJS — Superheroic JavaScript MVW Framework* [sans date]) est un framework **JavaScript** développé par Google qui est principalement utilisé pour créer des **applications web monopages** (SPA). Basé sur une architecture **MVC** (modèle vue contrôleur), il offre une liaison de données bidirectionnelle, des directives pour une extensibilité HTML, et un système d'injection de dépendances pour la modularité. AngularJS facilite les tests unitaires, la gestion des états de l'application, et l'intégration avec des services web Rest.

ii. VueJS

Figure 7 - Logo VueJS

Vue.js (*Vue.js - The Progressive JavaScript Framework | Vue.js* [sans date]) est un **framework JavaScript** progressif, mettant l'accent sur la construction d'interfaces utilisateur réactives. Sa conception incrémentale permet une **intégration facile** dans des projets existants, tandis que sa syntaxe concise et son architecture de composants favorisent la **simplicité et la maintenabilité** du code. Avec un système réactif permettant une liaison bidirectionnelle des données, Vue.js simplifie la gestion des états de l'application. Les directives **facilitent la manipulation** du DOM (Document Object Model)[1] de manière déclarative, tandis que l'écosystème robuste et l'interface en ligne de commande (CLI) simplifient le développement, la configuration et le déploiement des projets Vue.js.

iii. ReactJS

Figure 8 - Logo ReactJS

React.js (*React* [sans date]) est un **framework JavaScript** populaire axé sur la construction d'interfaces utilisateur réactives. Conçu par Facebook, React se distingue par son DOM et son **approche déclarative**. Il favorise la création d'applications modulaires à l'aide de composants réutilisables, **simplifiant ainsi la structuration et la maintenance du code**. La gestion unidirectionnelle des données facilite le suivi des états de l'application. Avec une communauté active et un écosystème riche, React offre des outils tels que Redux pour la gestion d'état et Next.js pour le développement d'applications web robustes.

En tenant compte des contraintes de notre projet, la première étape consiste à piloter exclusivement des périphériques utilisant le protocole MQTT et Zigbee.

[1] “Le Document Object Model est une interface de programmation normalisée par le W3C, qui permet à des scripts d'examiner et de modifier le contenu du navigateur web.” (https://fr.wikipedia.org/wiki/Document_Object_Model)

À la suite de notre recherche documentaire, nous avons identifié une solution capable de gérer le protocole MQTT au sein d'un système largement utilisé par les différentes solutions précédemment évoquées. Cette solution est Zigbee2MQTT.

iv. Zigbee2MQTT

Figure 9 - Logo ZigBee2MQTT

Zigbee2MQTT (*Home | Zigbee2MQTT* [sans date]) est une solution open source qui facilite la gestion du protocole MQTT dans les systèmes domotiques, améliorant ainsi l'interopérabilité des dispositifs Zigbee au sein de l'Internet des Objets. En agissant comme un **pont entre les dispositifs Zigbee et le protocole MQTT**, cette plateforme offre une approche unifiée pour la communication et l'intégration au sein de différentes plateformes domotiques. Son développement est **open source** et permet aux utilisateurs de personnaliser la solution selon leurs besoins, tandis que la communauté assure un développement continu et la prise en charge de nouveaux périphériques. Les fonctionnalités incluent une configuration simple des appareils Zigbee, la gestion des groupes, et la compatibilité avec une variété de capteurs et d'actionneurs.

III. CONCEPTION & RÉALISATION DU PROJET

Notre projet est centré autour de l'accessibilité qui nous démarque de nos concurrents qui sont déjà bien implantés sur le marché, notamment homeAssistant qui est un logiciel gratuit sorti en 2013. Notre objectif est donc de faire quelque chose de simple qu'un utilisateur lambda pourra prendre en main très facilement.

Notre projet sera divisé en deux grandes parties, le front avec principalement l'interface homme/machine et le back avec le serveur et la base de données. Pour visualiser l'ensemble voici un schéma de la structure de notre projet :

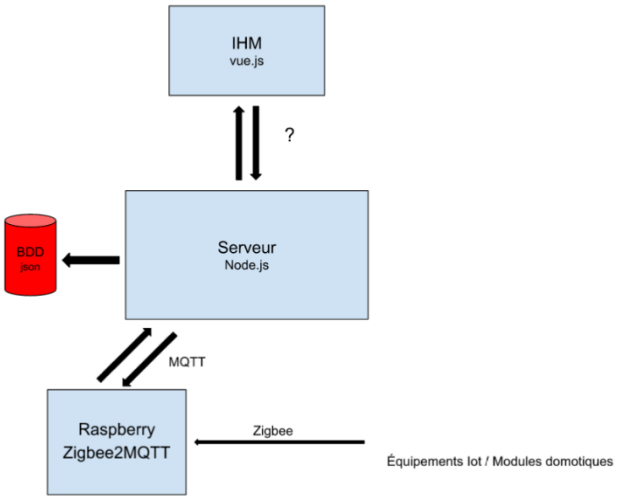


Figure 10 - Organisation générale du projet

La structure de notre projet est donc composée d'une IHM avec laquelle l'utilisateur interagit.

Cette IHM sert d'affichage uniquement, il sera sous forme d'une page Web. L'ensemble des données sont traitées par le serveur. Le serveur sert d'intermédiaire entre la base de données, Zigbee2MQTT, et l'IHM.

Nous allons maintenant décrire comment nous récupérons les informations de Zigbee.

Description du scénario d'envoi d'informations à un module

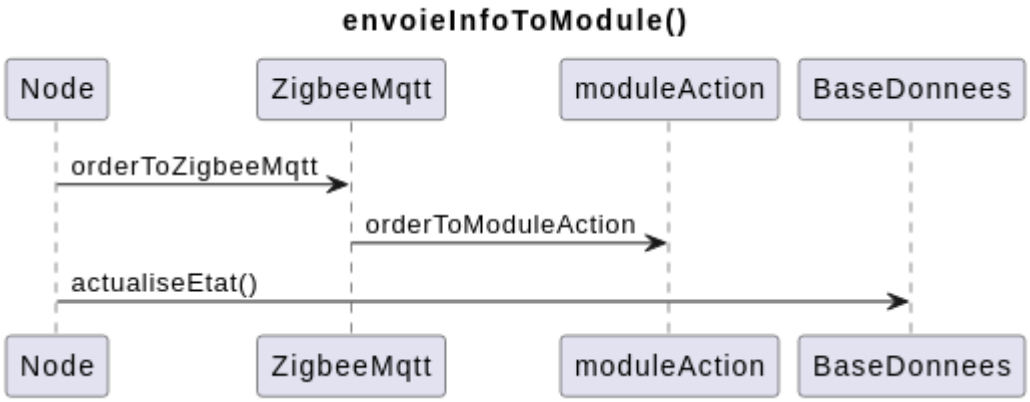


Figure 11 - Exemple d'un scénario lors d'un envoie de module

Le diagramme de séquence illustre comment nous acheminons les données vers un module tout en synchronisant notre base de données en simultané. Initialement, notre serveur Node envoie des données à Zigbee2Mqtt, lequel les transmet aux actionneurs pour les mettre en marche. En parallèle, l'événement actualiseEtat() s'occupe de mettre à jour les informations dans la base de données.

IV. GESTION DE PROJET

a. PRÉSENTATION DE L'ÉQUIPE

Fiche des compétences de chacun :

Personnes / Compétences	Thibault	Ismaël	Elise	Aubry
Zigbee				
Compréhension du MQTT				x
Compétences en électronique	x	x	x	x
Configuration Raspberry Pi	x	x		x
Installation de capteurs Zigbee				
Programmation web(HTML, CSS, VueJS, NodeJS)		x	x	x
Gestion de bases de données		x	x	
Configuration MQTT Broker				x
Sécurité logicielle				
Déjà travaillé sur une maison intelligente ?	x	x	x	

Les compétences variées de chaque membre de l'équipe s'entrecroisent et se complètent, couvrant ainsi un large éventail des besoins du projet de maison intelligente utilisant Zigbee et MQTT.

b. PLANIFICATION

Afin de planifier au mieux notre travail, nous avons décidé d'utiliser Click Up. C'est un outil de gestion de tâche. Nous avons ainsi les annexes 2 et 3 de ce rapport qui présente les

diagrammes Gant, rassemblant l'ensemble du travail qui a été effectué et le travail qui sera fourni lors du deuxième semestre.

c. AVANCEMENT PROJET

i. Partie Serveur

À ce stade, nous avons mené des essais de lecture des logs à partir d'un fichier, nous permettant de détecter de nouvelles connexions/déconnexions d'objets connectés ainsi que de recevoir des données de nos capteurs. L'analyse de ce fichier permet d'extraire les informations pertinentes concernant les capteurs qui serons ensuite insérées dans une base de données. Il est à noter que notre base de données actuelle prend la forme d'un fichier JSON, toutefois, cette dernière demeure temporaire. Nous envisageons une transition vers un autre type de stockage, peut-être sous forme de bases de données telles que SQLite ou MongoDB.

De plus, nous avons mis en place un serveur Express afin de gérer les routes. Ces routes nous permettent d'envoyer et de récupérer des informations depuis notre base de données. Bien qu'aucun lien concret n'ait encore été établi avec la partie frontale du projet (interface utilisateur), nos tests sur les routes et les contrôleurs intégrés fonctionnent. Nous avons donc réussi à créer des profils utilisateurs dans le cadre d'un processus d'authentification reposant sur les identifiants et les mots de passe, procédant à des essais fructueux par le biais de Postman.

Voici un exemple de test que nous avons eu à réaliser :

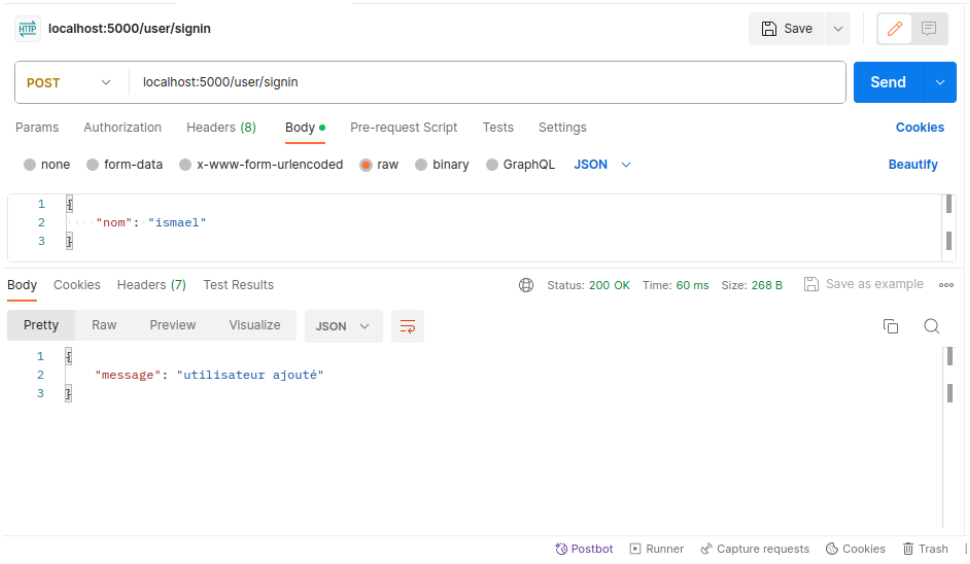


Figure 12 - Exemple de requête POST avec une interface Postman

En somme, nous avons réussi à lire les logs, à les intégrer dans la base de données, et à mettre en place un système de connexion pour les utilisateurs. Ces avancements constituent une étape significative dans l'évolution de notre projet.

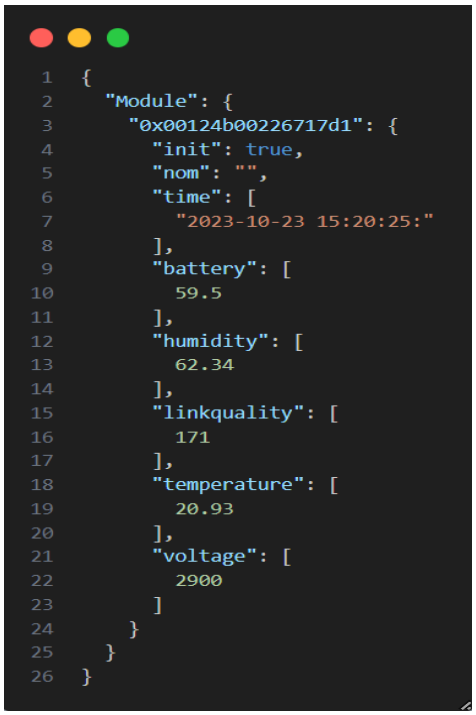


Figure 13 - Exemple éléments base de données JSON

Cette image illustre l'état actuel de notre base de données JSON, on présente ici la table Module qui a comme clés les identifiants de nos appareils connectés. De ce fait, nous stockons toutes les informations qui sont liées à cet appareil (capteur ou actionneur) dans cette table. Nous stockons dans cette table le nom du module, ainsi que les événements détectés par notre module concernant la batterie, la température, et l'humidité, c'est à dire toutes les nouvelles données concernant le nouvel état de notre module sont ajoutées aux données déjà existantes dans notre table, c'est la raison pour laquelle les valeurs lies aux clés comme température sont des tableaux, la date à laquelle ces données sont enregistrées sont elles aussi enregistrées.

ii. Partie client

L'objectif du projet pour la partie IHM est de développer une interface web qui soit capable de répondre aux différents besoins décrits précédemment à savoir la possibilité de vérifier le statut des différents capteurs, de programmer différents scénarios ou encore la possibilité de paramétrer l'interface.

La veille technique que nous avons fait nous a permis de découvrir trois Framework javascript différents qui ont des caractéristiques propres à chacun. Notre objectif à terme est de déployer une **application web légère**. Le choix de le faire avec le Framework Vu JS était plus simple car lors de nos tests de développement sur les différents Framework, nous avons tous remarqué que ceux vu précédemment étaient plus simple à mettre en place et accessible. De plus, nous trouvions que ça syntaxe était plus accessible que Angular (qui nécessite

d'apprendre du Typescript ainsi que des concepts plus avancés de programmation notamment avec le pattern MVC) ou encore avec React ou nous avions plus de difficulté avec la syntaxe et la méthode de programmation. VueJS nous permet de remettre en application le concept vu en TWEB et le Framework est reconnu pour être plus accessible que React ou encore Angular.

Actuellement, la partie IHM est développée en mode single Page. Cette caractéristique nous permet de manipuler une application légère et simple. L'objectif est d'avoir quatre menus accessibles sur la même page.

Voici un premier visuel de l'IHM.

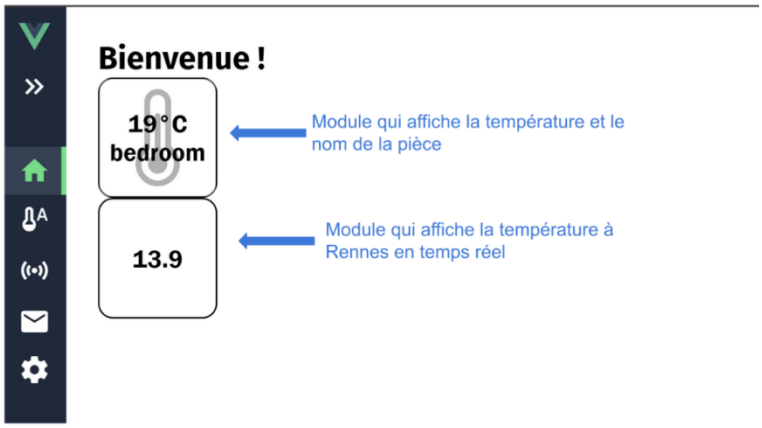


Figure 14 - Pages Principale de l'application

Nous avons pour le moment une page principale qui accueillera par la suite des modules contenant les informations des capteurs.

Nous avons aussi mis en place d'autre menu dans un volet déroulant mais qui mène tous à une page en travaux qui accueilleront de futures fonctionnalités.

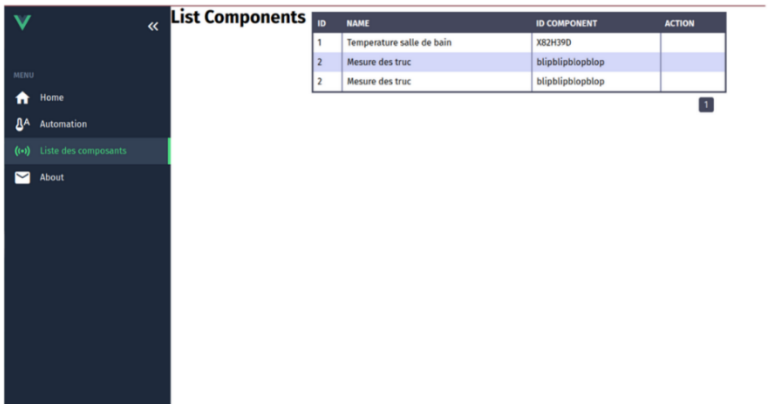


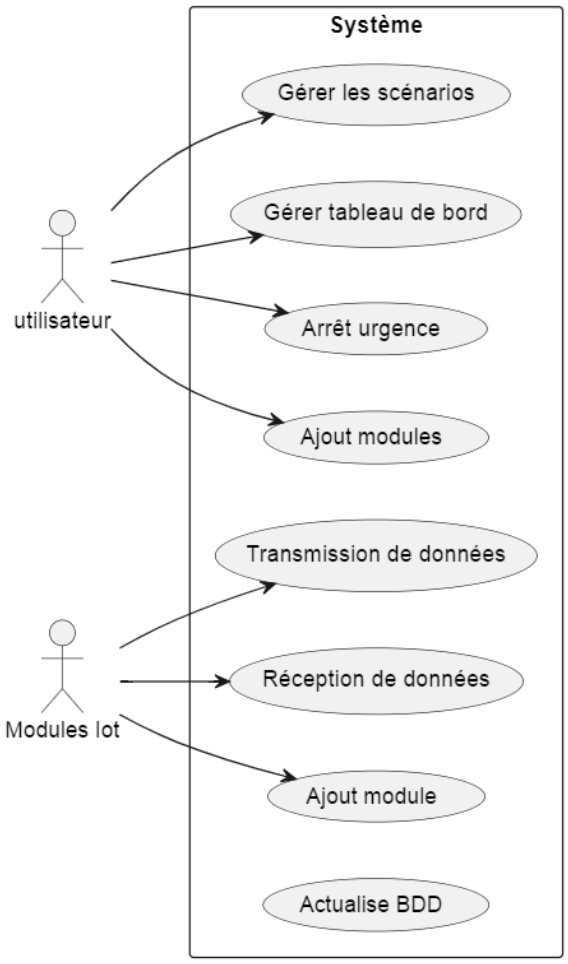
Figure 15 - Page concernant les composants

V. CONCLUSION

En conclusion, ce semestre a été marqué par une amorce dans le développement de notre interface de pilotage de périphérique domotique. La diversité des compétences au sein de notre équipe a été un atout majeur, nous permettant de couvrir un large éventail des besoins du projet. Les tâches ont été repartit selon la volonté et les compétences de chacun. Une base solide a été créée définissant les tenants et aboutissants du projet. Notre projet est organisé et prévu sur le long terme, ce qui nous permet d'avoir une vision globale de nos objectifs.

VI. ANNEXES

a. Annexe 1 - Diagramme UML Use Case



- GÉRER LES SCÉNARIOS :
  - BUT : Pouvoir créer et activer/désactiver des enchaînements d'actions prédéfinies.
  - DÉBUT : clic sur 'créer un scénario'
  - FIN : scénario activé/désactivé
  - ACTEURS : utilisateur
  - ENCHAÎNEMENT : clic sur 'créer un scénario', indiquer les actions à réaliser, définir des horaires, enregistrer le scénario, scénario activé/désactivé
  - ALTERNATIVE : activer manuellement les modules
  - EXCEPTIONS : un module du scénario n'a plus de pile/batterie,

- GÉRER LE TABLEAU DE BORD :
  - BUT : Gestion des éléments, esthétique de l'écran d'accueil, visibilité des informations
  - DÉBUT : ouverture de l'application
  - FIN : fermeture de l'application
  - ACTEUR : utilisateur
  - ENCHAÎNEMENT : ouverture de l'application, ajout de widgets/de blocs d'informations, enregistrer, fermer l'application
  - ALTERNATIVE : Tableau de bord de base

    - EXCEPTIONS :

- ARRÊT D'URGENCE :
    - BUT : arrêter le système en cas de problème
    - DÉBUT : clic sur 'arrêt d'urgence' (physique ou appli)
    - FIN : arrêt du système
    - ACTEUR : utilisateur
    - ENCHAÎNEMENT : clic, arrêt
    - ALTERNATIVE : couper le courant
    - EXCEPTION : problème de réseau

- AJOUT DES MODULES :
    - BUT : ajouter de nouveaux objets connectés
    - DÉBUT : clic sur 'ajouter des modules'
    - FIN : module ajouté
    - ACTEUR : utilisateur
    - ENCHAÎNEMENT : clic sur 'ajout module', clic sur le mode appairage du module, détection du nouvel appareil, sélection du nouvel appareil, module ajouté
    - ALTERNATIVE : utiliser les modules déjà ajoutés
    - EXCEPTION : problème de réseau / interférence / limite d'ajout à 40 (100 si débridage)

- TRANSMISSION DES DONNÉES :
    - BUT : les modules envoient des données vers le système
    - DÉBUT : module connecté
    - FIN : module déconnecté/ arrêt du système
    - ACTEUR : modules
    - ENCHAÎNEMENT : connecteur du module, envoie des données, réception des données par le système, déconnexion du module / arrêt système

b. Annexe 2 - Diagramme de Gantt IHM

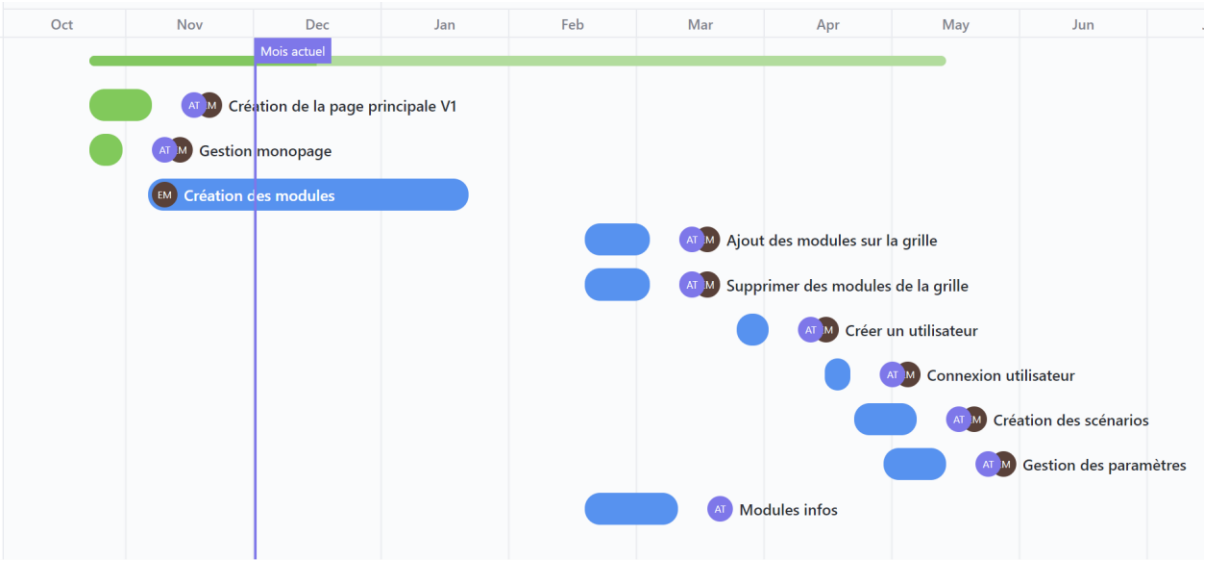


Figure 16 - Diagramme Gantt Partie IHM

c. Annexe 3 - Diagramme de Gant Serveur

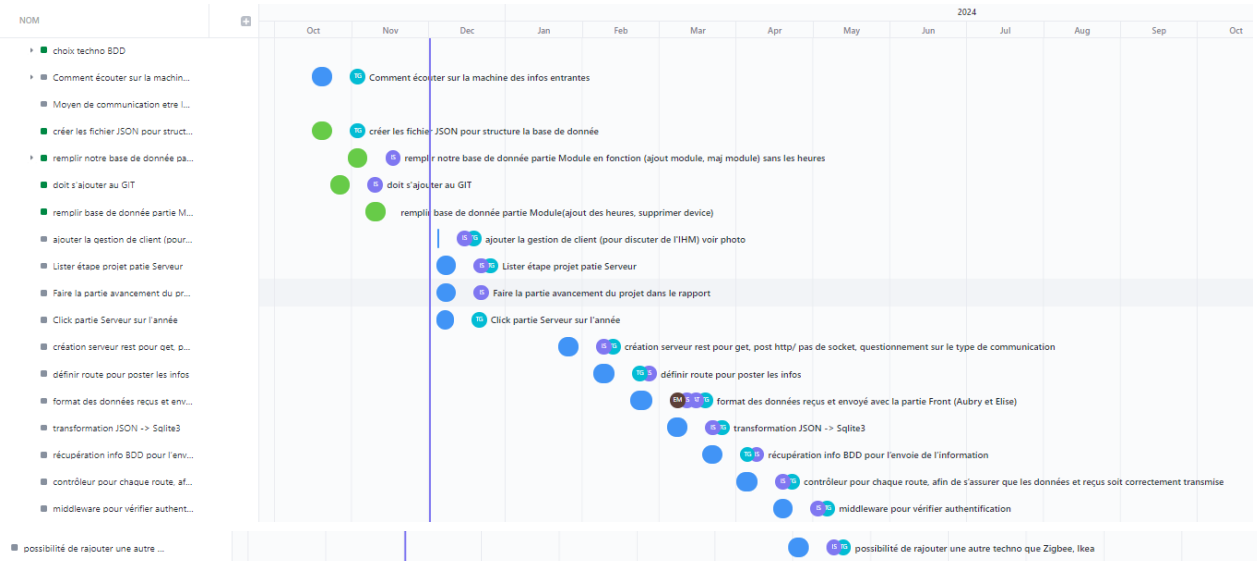


Figure 17 - Diagramme Gantt Partie Serveur

VII. BIBLIOGRAPHIE

AngularJS — Superheroic JavaScript MVW Framework, [sans date]. [en ligne]. [Consulté le 11 décembre 2023]. Disponible à l'adresse : https://angularjs.org/

ASSISTANT, Home, [sans date]. Home Assistant. *Home Assistant* [en ligne]. [Consulté le 11 décembre 2023]. Disponible à l'adresse : https://www.home-assistant.io/

ESPHome, [sans date]. *ESPHome* [en ligne]. [Consulté le 11 décembre 2023]. Disponible à l'adresse : https://esphome.io/index.html

Home | Zigbee2MQTT, [sans date]. [en ligne]. [Consulté le 11 décembre 2023]. Disponible à l'adresse : https://www.zigbee2mqtt.io/

Jeedom, [sans date]. [en ligne]. [Consulté le 11 décembre 2023]. Disponible à l'adresse : https://www.jeedom.com/fr

React, [sans date]. [en ligne]. [Consulté le 11 décembre 2023]. Disponible à l'adresse : https://fr.react.dev/

Tuya Smart - Global IoT Developer Service Provider, [sans date]. [en ligne]. [Consulté le 11 décembre 2023]. Disponible à l'adresse : https://www.tuya.com/

Vue.js - The Progressive JavaScript Framework | Vue.js, [sans date]. [en ligne]. [Consulté le 11 décembre 2023]. Disponible à l'adresse : https://vuejs.org/