

Cahier des charges CLOUD OF THINGS **SMART GREEN HOUSE**



Réalisé par :
CHEBAANE ZEINEB
ZOUARI RAMI

Encadré par :
MR. MOHAMED BÉCHA
KAANICHE



Table des matières

1 Concept	ii
1.1 Context général du projet	ii
1.2 Problématique	1
1.3 Solution proposée	1
2 Clients Objectifs et limites	2
2.1 Clients	2
2.2 Objectifs	2
2.3 Limites	2
3 Choix technologique	3
3.1 Côté serveur	3
3.2 Middleware	4
3.3 Côté Client	4
3.4 Les composants IoT	5
3.5 Architecture	5
4 Modèle commercial	7
4.1 Modèles commerciales	7
4.2 Business Model Canvas	8
5 Livrables	10
6 Contraintes	11
6.1 Méthodologie de travail	11
6.2 Contrainte de temps	13
6.3 Diagramme de Gantt	13
7 Diagramme de déploiement	16

Concept

1.1 Context général du projet

L'un des grands défis auquel doit faire face un pays est la satisfaction, en matière de nutrition, de la population. La solution à ce problème passe nécessairement par le développement du secteur agricole. L'activité agricole existe depuis la sédentarisation des populations humaine. Depuis son existence cette activité n'a de cesse de se développer avec le développement de l'humanité. C'est devenu un symbole de prospérité de beaucoup de civilisation.

Ce développement c'est, bien sûr, fait grâce au développement de moyens techniques et technologiques. La nécessité d'augmenter la production alimentaire pose alors une problématique remarquable. Cependant, l'émergence des technologies avancées touchant tous les domaines de vie présente une solution efficace pour le secteur d'agriculture.

L'IoT semble d'ailleurs d'une grande aide au secteur agricole. L'un des plus grands domaines d'application de l'Iot dans le secteur de l'agriculture est le Smart Green House : Est une percée dans la technologie agricole qui crée un climat pour une croissance constante des plantes à l'aide de la technologie des capteurs. Il optimise la croissance des plantes en surveillant les conditions climatiques (température et humidité) dans une serre et de mettre en œuvre des actions correctives pour maintenir des conditions optimales pour la croissance des plantes.

Dans ce cadre, nous proposons de concevoir une solution intelligente pour les serres dite "Smart Green House" permettant d'assurer une productivité optimale.

1.2 Problématique

Une serre est conçue à l'origine comme un simple abri, ou une enseinte destinée à la culture et à la protection des plantes en exploitant le rayonnement solaire. Elle est devenue un local industriel de production de la matière végétale où l'on tente d'adopter l'environnement immédiat de la plante de façon à améliorer sa productivité et sa qualité en l'affranchissant du climat extérieur, du sol local et des saisons.

les facteurs climatiques qui influencent le climat intérieur de la serre sont la température et l'humidité. Chacun de ces facteurs engendre une combinaison d'effets qui peuvent être favorables ou non au fonctionnement de la serre selon les conditions locales qui prévalent. La température intervient d'une façon prépondérante dans la croissance et le développement de la végétation, ainsi le taux d'humidité joue aussi un rôle dans le processus de la photosynthèse (fixation du dioxyde de carbone et de l'eau dans les feuilles afin de produire des sucres utilisés pour l'énergie et la croissance).

Un contrôle alors bien maîtrisé du plan énergétique du climat permet donc de gérer ces paramètres et d'améliorer le fonctionnement des plantes.

1.3 Solution proposée

Ce projet consiste à construire une serre intelligente créant un microclimat adapté à la croissance des plantes grâce à l'utilisation de capteurs (capteur d'humidité et de température) qui permet de :

- Surveiller le taux d'humidité, la valeur de température et la concentration de CO₂.
- Affirmer l'agriculteur à propos des conditions climatiques dans la serre.
- Donner une alerte lorsque les valeurs dépassent le seuil ou sont inférieures au seuil à travers l'application mobile.
- Activer l'aération automatiquement en cas de besoin pour réguler les paramètres de la serre.

Clients Objectifs et limites

2.1 Clients

Le profil de notre clientèle cible qui va utiliser notre solution sera les jeunes agriculteurs.

2.2 Objectifs

Notre solution **Smart Green House** consiste à contrôler et surveiller d'une façon sophistiquée le climat dans les serres pour garantir un taux de croissance élevé de la culture à travers :

- La surveillance de la température au fond de la serre.
- La surveillance de taux d'humidité du sol.
- La surveillance de la concentration de CO2. - Assurance de l'aération automatique de la serre pour régler la température au fond d'elle par rapport à la température ambiante.

2.3 Limites

Le transfert et l'acquisition des données de différentes serres passe toujours par le Cloud et qui est en relation direct avec le réseau Internet utilisé. En effet, un mauvais débit d'internet peut conduire à un mauvais suivi des paramètres de mesure de notre solution. Dans ce cas, il est nécessaire l'acquisition d'un réseau performant qui résoudre le problème le transfert de données avec un délai de retard négligeable pour que notre solution fonctionne d'une manière performante.

Choix technologique

3.1 Côté serveur

- **MongoDB :** C'est un système de gestion de base de données orienté documents, répartissable sur un nombre quelconque d'ordinateurs et ne nécessitant pas de schéma prédéfini des données. Elle est distribuée et sécurisée par défaut, et disponible en tant que service entièrement géré sur AWS, Azure et Google Cloud. Le choix de la base de données MongoDB est justifié par les points forts suivants :

- Base de données NoSQL
- Une orientation documents
- Performance excellente
- Un système totalement dynamique
- Une très bonne flexibilité
- Un système de gestion en JSON et BSON (JSON binaire)

- **Mosquitto MQTT Broker :** Mosquitto est un serveur MQTT Open Source (Broker) qui facilite la communication entre objets connectés (M2M). Il réduit les taux de mise à jour en secondes et le protocole Publish/Subscribe collecte plus de données avec moins de bande passante.

- **Node Red :** C'est un outil de développement basé sur les flux pour la programmation visuelle développé pour connecter ensemble des périphériques matériels, des API et des services en ligne dans le cadre de l'Internet des objets.

3.2 Middleware

- **Jakarta EE** : C'est une spécification pour la plate-forme Java d'Oracle, destinée aux applications d'entreprise. La plate-forme étend Java Platform, Standard Edition (Java SE) en fournissant une API de mapping objet-relationnel, des architectures distribuées et multitiers, et des services Web 3.0. La plate-forme se fonde principalement sur des composants modulaires exécutés sur un serveur d'applications.

3.3 Côté Client

- **Flutter** : Flutter est un kit de développement logiciel d'interface utilisateur open-source. Il est utilisé pour développer des applications pour Android, iOS, Linux, Mac, Windows, Google Fuchsia et le web à partir d'une seule base de code.

Le choix d'utiliser flutter est justifié par ces points forts suivants :

- Flutter permet de simplifier et réduire le temps de développement.
- Flutter est Open source
- Flutter utilise un langage de programmation facile à apprendre et à mettre en œuvre, appelé Dart, qui est le langage de programmation général de Google.
- Le moteur de rendu de Flutter est développé en C++, qui est un langage offrant de meilleures performances que celui de React Native, développé en Javascript.

- **Angular** : est un framework pour clients, open source, basé sur TypeScript. Angular est une réécriture complète d'AngularJS, cadriel construit par la même équipe. Il permet la création d'applications Web et plus particulièrement d'applications Web monopages : des applications Web accessibles via une page Web unique qui permet de fluidifier l'expérience utilisateur et d'éviter les chargements de pages à chaque nouvelle action. Le framework est basé sur une architecture du type MVC et permet donc de séparer les données, le visuel et les actions pour une meilleure gestion des responsabilités.

Le choix d'utiliser angular est justifié par les points forts suivants :

- Découvrez une façon de créer des applications avec Angular et réutilisez le code pour créer

des applications pour n'importe quelle cible de déploiement. Pour le Web, le Web mobile, le mobile natif et le bureau natif.

- L'avantage de Angular par rapport React est la disponibilité des packages. En effet, il existe un énorme référentiel de packages open-source disponibles pour les développeurs Angular comme sont NgBootstrap, Angular Google Maps, NgRx, NgTranslate, AngularFire, NgxTextEditor, Angular Material, Ng2 Pdf Viewer, NgxCharts

- **Capacitor :** Capacitor est un environnement d'exécution natif open source pour la construction d'applications Web natives. Créez des applications Web multiplateforme iOS, Android et progressives avec JavaScript, HTML et CSS.

3.4 Les composants IoT

- **Raspberry Pi 4 :** Le Raspberry Pi 4 est un nano-ordinateur pouvant se connecter à un moniteur, à un ensemble clavier/souris et disposant d'interfaces Wi-Fi, Bluetooth et Ethernet. Il fonctionne depuis une carte micro-SD et fonctionne avec un système d'exploitation basé sur Linux ou Windows 10 IoT.

- **Les capteurs nécessaires :**

- Capteur de température et d'humidité DHT22 : Ce capteur de température et d'humidité a une précision de température de +/- 0,5 C et une plage d'humidité de 0 à 100 . Il est simple à connecter au Raspberry Pi.
- Capteur de CO2 "Atlas ezo CO2" pour mesurer la concentration de CO2 dans la serre.

3.5 Architecture

Notre système est basé principalement sur des capteurs qui peuvent fonctionner confortablement et d'une manière efficace dans l'agriculture dans chaque serre.

Il contient les différents types de capteurs pour différents cas d'utilisation comme : capteur d'humidité, capteur de température, capteur de mesure du taux de CO2 dans la serre.

CHOIX TECHNOLOGIQUE

Notre architecture prend le modèle suivant décrit ci-dessous :

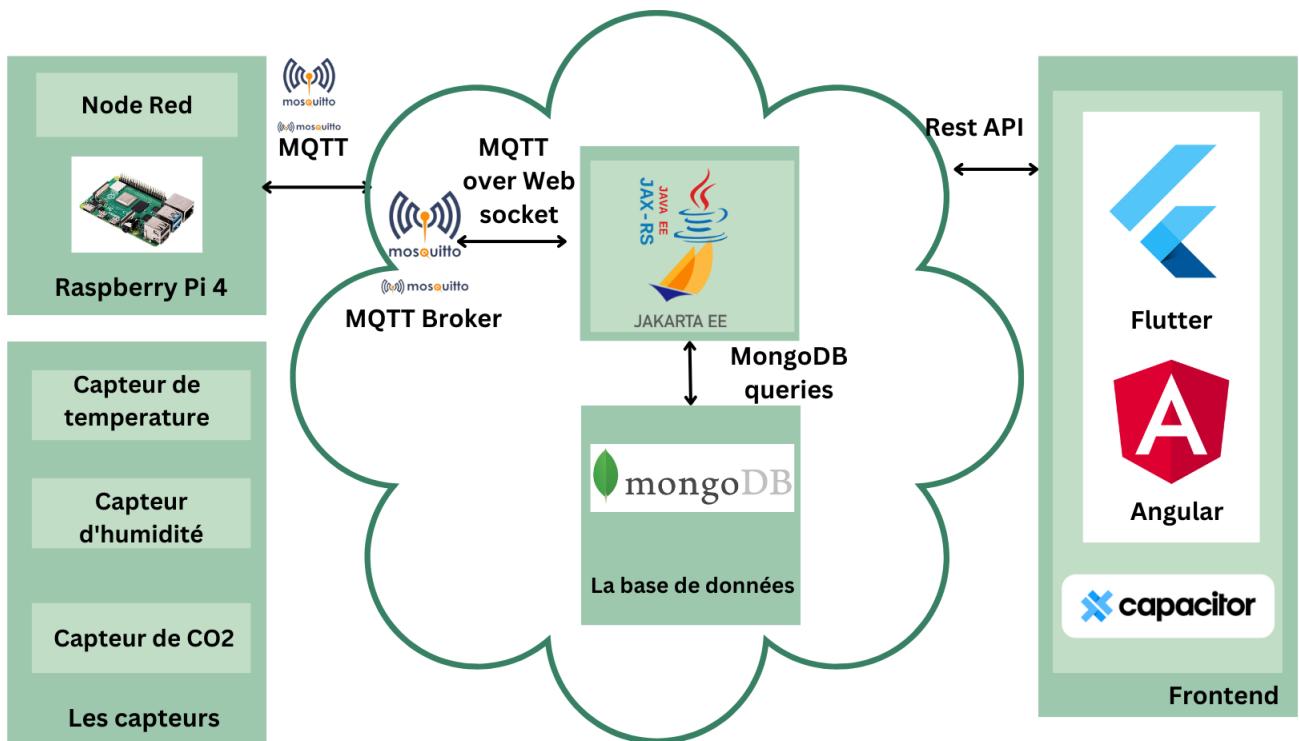


Figure 1 - Architecture du système

Modèle commercial

4.1 Modèles commerciales

Nous allons adopter le concept du 4P du marketing.

Les 4P du marketing visent à proposer le bon produit au bon moment, au bon endroit et à bon prix en tenant compte des habitudes des consommateurs. Il permet aussi bien de fixer la manière du suivie de l'évolution de ces cotumes d'une façon synchrone dans une entreprise.

Intimement liés les uns aux autres, ces 4P sont :

- **Politique Produit** : Notre solution est considérée comme le centre de notre stratégie Marketing.

Marque : Notre Solution “Smart Green House” qui est à la base d'une application hybride composée essentiellement par les deux parties hardware et software. Services liés au produit :

- Des mises à jour au niveau de notre solution, des nouvelles fonctionnalités, des nouvelles propriétés et des évolutions quotidiennes.
- Des garanties allant jusqu'à 2 ans.

Les caractéristiques : Désignation par une variété qui est relative aux fonctionnalités de notre produit, les options qui le caractérise, le design, les formes...

- **Politique Prix** : Le prix est considéré comme l'élément clé et le centre d'établissement de la communication.

Le prix de notre solution dépend des exigences que le client désire les implémentées dans sa serre.

La politique de réduction commerciale :

- Il aura des promotions et des remises dans le cas où il y a plus que 2 fonctionnalités à

implémenter.

- Des réductions particulières auront lieu pour des problèmes liés à la qualité et efficacité de la solution.

- Le paiement s'effectue soit par le virement bancaire ou par le paiement en ligne.

- Politique de distribution : La distribution s'intéresse aux moyens mis en disposition pour commercialiser les produits.

- Les canaux de distribution : Les sites de vente en ligne ou les sites de vente de matériel agricole.

- Politique de Communication : La politique de communication est sans aucun doute la politique à laquelle les créateurs d'entreprise pensent le plus lorsqu'ils souhaitent mettre sur le marché un produit ou un service. La communication est là encore un domaine assez vaste pour lequel il faudra prendre en compte l'ensemble des possibilités offertes pour établir votre stratégie commerciale :

- Publicité

- Les réseaux sociaux

4.2 Business Model Canvas

Le business model canvas est un outil que l'on utilise d'une façon générale pour retranscrire de manière simple le modèle économique d'une entreprise. Il est parfaitement adapté à la phase de création, et peut aussi être utilisé pour le lancement d'un produit ou d'un service.

La figure ci dessous présente le Buisiness Model Canvas pour notre projet.

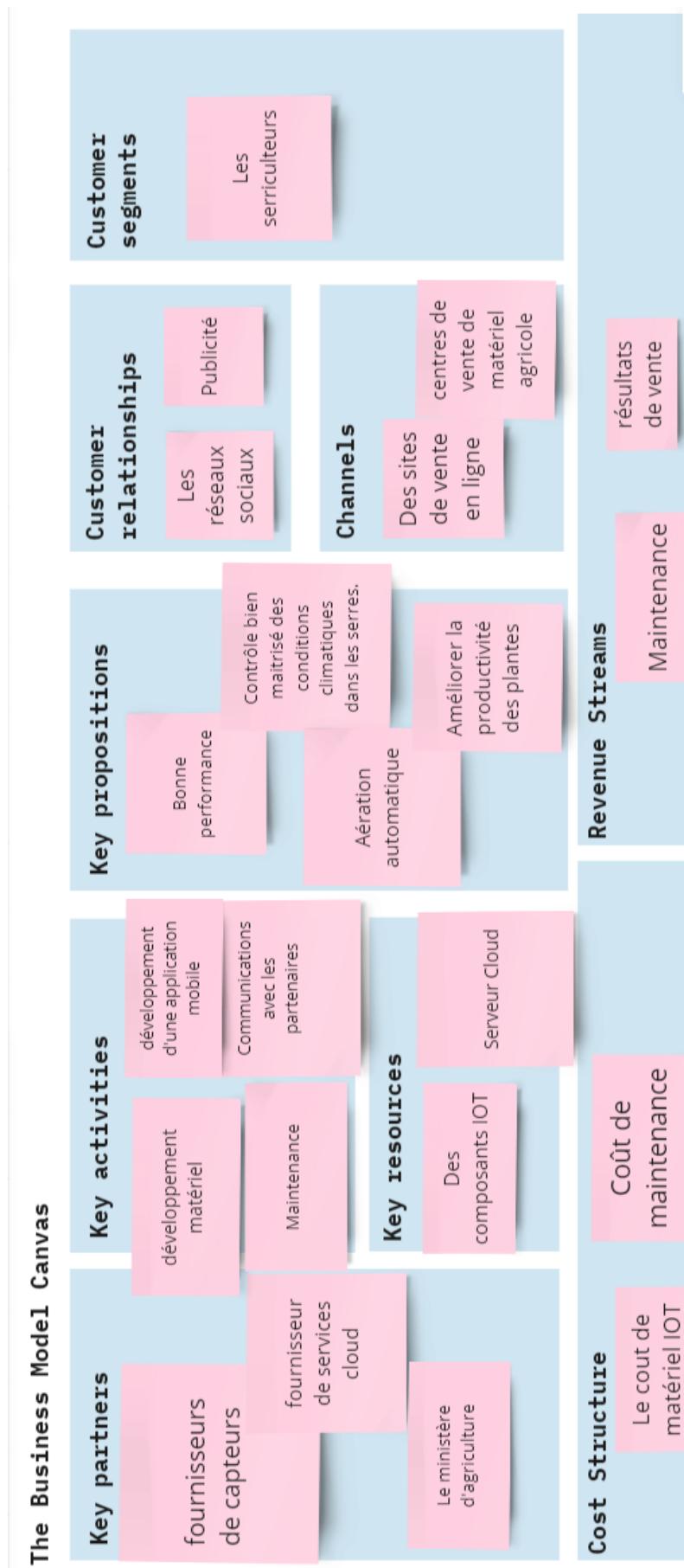


Figure 2 - Structure BMC

Livrables

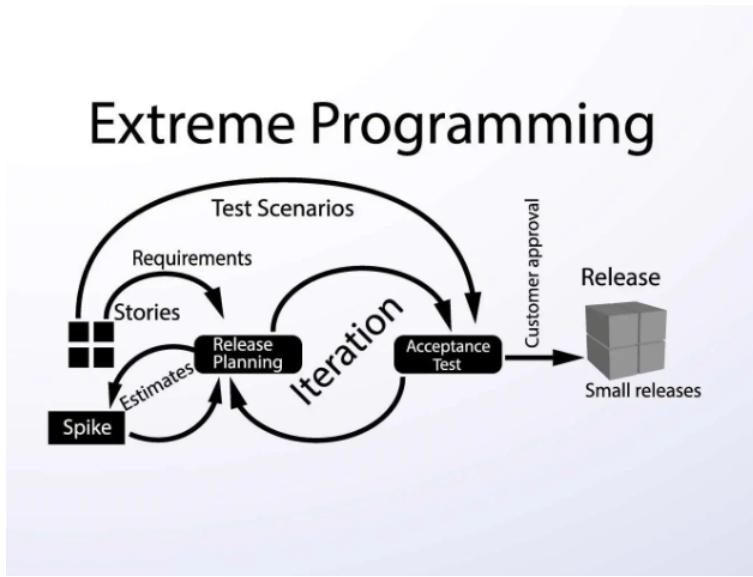
- **Cahier conceptuel** : Ce document présente de manière détaillée et structurée les spécifications, les services à rendre.
- **Exécutables et Sources** : Des fichiers dans un répertoire en Github contenant le code de la solution IoT et de l'application mobile. développée.
- **Documentation technique** : La totalité des bibliographie et technologies utilisées dans le développement de cette solution ainsi que les références utilisées.
- **Vidéo de démonstration** : Une vidéo sous format mp4 qui contient une démonstration de la solution proposée.

Contraintes

6.1 Méthodologie de travail

Nous allons utiliser comme méthodologie de travail Extreme Programming (XP) qui est un cadre de développement logiciel agile qui vise à produire des logiciels de meilleure qualité. l'Extreme Programming est basé sur les exigences du client. Le développement de logiciels classiques ne peut pourtant répondre aux désirs des clients que dans une certaine mesure, les choses se compliquent notamment quand ces désirs changent régulièrement. XP tente par ailleurs d'exciter la créativité des développeurs et accueille les erreurs comme un facteur évident dans le processus de travail.

- **Principe de la méthode XP :** Les principes de la méthode XP ne sont pas nouveaux puisqu'il s'agit de ceux des méthodes agiles. La différence et l'originalité consistent dans le fait qu'elles sont poussées à l'extrême. La méthode XP se base principalement sur :
 - Une forte réactivité au changement des besoins du client.
 - Un travail par binome.
 - La qualité du travail fourni.
 - La qualité des tests effectués.



- Côté Client

L'extreme programming agit de manière très orientée client, ce qui peut aller jusqu'à intégrer le client comme un membre de l'équipe et à avoir au moins un représentant sur site (On-Site Customer).

Le client pose ses exigences pour le produit, mais n'indique que partiellement la façon d'atteindre les objectifs. Seule la hiérarchisation des secteurs partiels relève de son domaine de compétences. Il doit ici également réussir à faire comprendre ses propres désirs.

- Côté Développeurs

L'équipe de développeurs n'est pas sous-divisée. Autrement dit : toute personne créant activement le produit prend la casquette de développeur. L'équipe regroupe donc non seulement les programmeurs, mais aussi d'autres personnes participant à la création, en fonction des exigences du projet. Outre le travail de développement effectif, la mission des développeurs est également de réagir aux désirs du client : évaluer la charge de travail, établir un calendrier, planifier la mise en œuvre.

- Côté Manager

Le rôle du manager consiste à établir le lien entre les développeurs et les clients. Les personnes de ce groupe amènent les deux autres à une même table et animent par exemple le planning game. Le manager veille ici à ce que les règles définies au préalable ainsi que les conventions générales d'une discussion constructive soient respectées.

- Côté Coach

Pour toute équipe qui inclus le client doit savoir gérer l'extrême programming(XP) et pouvoir appiquer cette méthode de travail de manière cohérente. Un coach peut contribuer à ce que tous se fassent une même idée des procédures. Il n'a rien à voir avec le développement du produit à proprement parler et ne sert que d'aide externe, de manière très similaire à un Scrum Master. Le coach accompagne l'équipe idéalement pendant toute la phase de développement et il est disponible pour répondre aux questions et aide à éclaircir des points obscurs. Rôles : Dans l'extrême programming, les rôles servent à répartir les tâches et les compétences entre tous les intervenants, tant les développeurs que les clients.

6.2 Contrainte de temps

Une période de 7 semaines ne sera pas suffisante pour bien terminer le développement de cette solution avec toutes les spécifications nécessaires et obtenir une version stable de l'application.

6.3 Diagramme de Gantt

La méthode Gantt se base sur la détermination de la meilleure manière de fixation des différentes tâches du projet à exécuter, sur une période déterminée, en fonction :

- Des durées de chacune des tâches.
- Des contraintes d'antériorité existant entre les différentes tâches.
- Des délais exigés à respecter.

Le développement de notre solution sera étalé sur 9 parties :

- Planification : 6 jours de 02/11/2022 à 09/11/2022.
- Conception : 5 jours de 10/11/2022 à 15/11/2022.
- Développement : 25 jours de 16/11/2022 à 24/12/2022.
- Partie Hardware : 9 jours de 16/11/2022 à 26/11/2022.
- Partie Software : 20 jours de 29/11/2022 à 24/12/2022.
- Test : 5 jours de 27/12/2022 à 31/12/2022.

CONTRAINTE

- Déploiement : 4 jours de 03/01/2023 à 06/01/2023.
- Documentation : 3 jours de 07/01/2023 à 11/01/2023.

Nom	Date de début	Date de fin
● Planification	02/11/22	09/11/22
● Conception	10/11/22	15/11/22
● Développement	16/11/22	24/12/22
● Partie Hardware	16/11/22	25/11/22
● Partie Software	26/11/22	24/12/22
● Test	26/12/22	31/12/22
● Déploiement	03/01/23	06/01/23
● Documentation	07/01/23	11/01/23

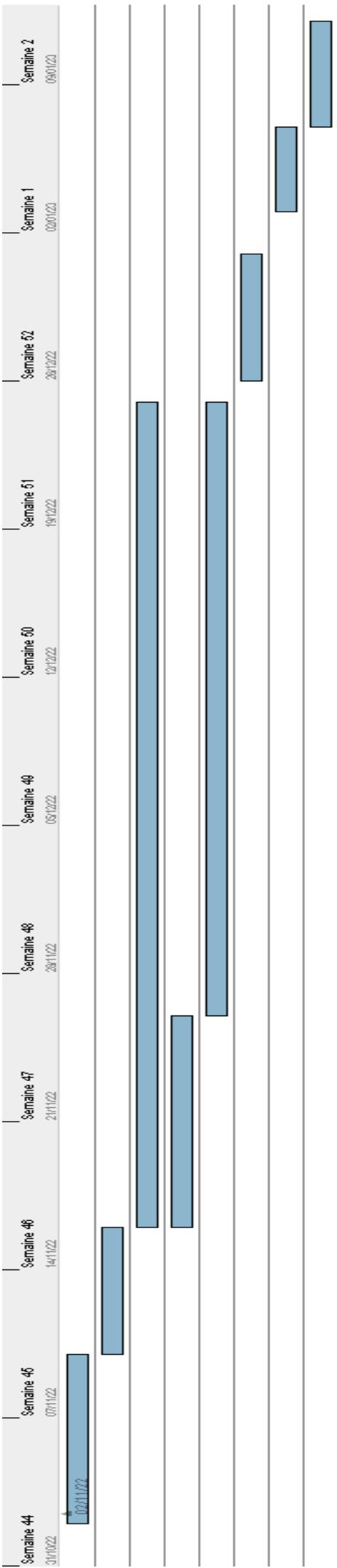


Figure 3 - Diagramme de Gantt

Diagramme de déploiement

Le diagramme de déploiement est un diagramme UML qui sert à décrire l'architecture d'exécution d'un système, y compris les nœuds tels que les environnements d'exécution matériels ou logiciels qui les relie.

Il permet aussi de décrire le déploiement physique des informations générées par le logiciel sur des composants matériels.

DIAGRAMME DE DÉPLOIEMENT

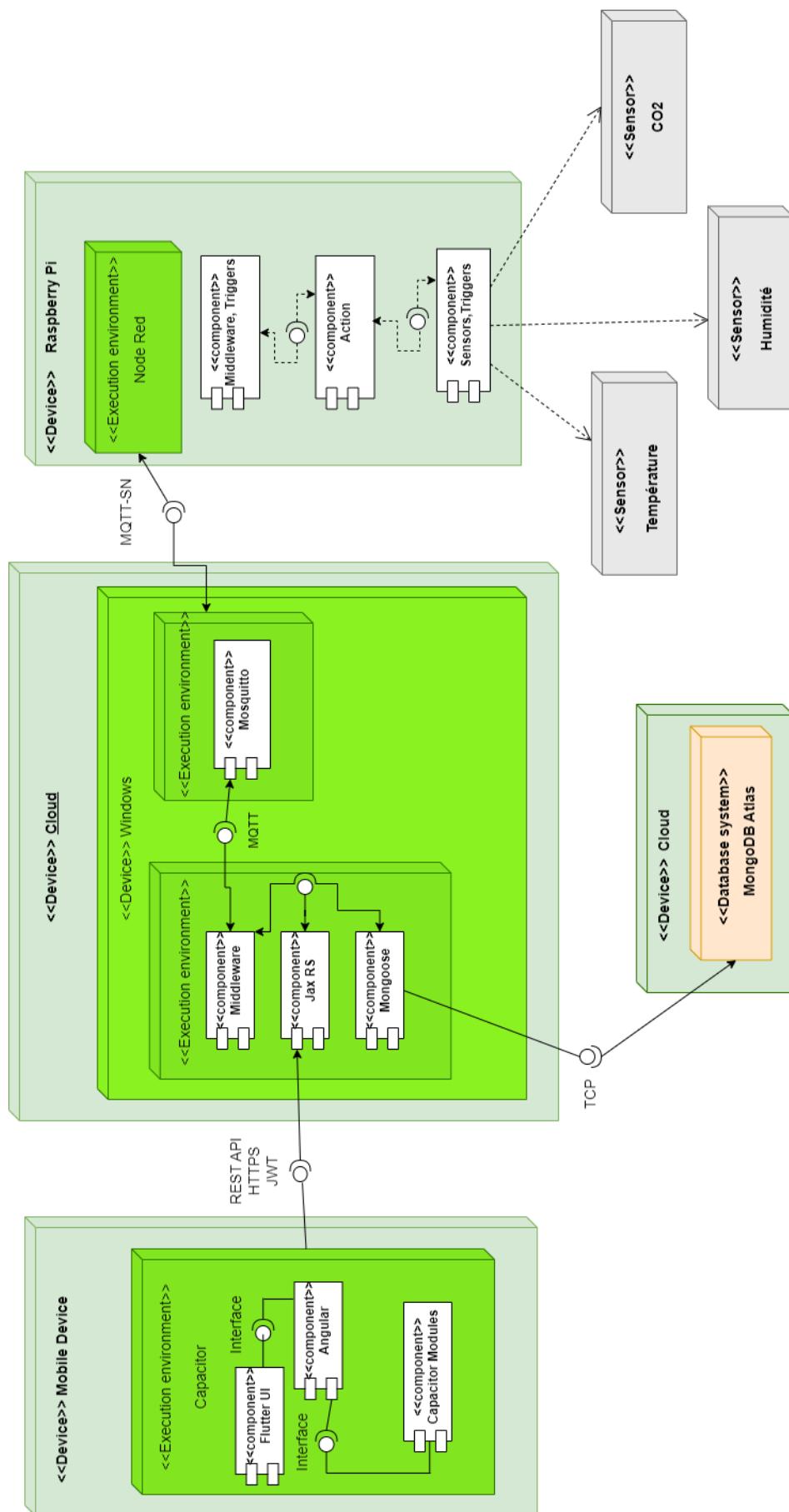


Figure 4 - Diagramme de déploiement