RÉGION ACADÉMIQUE
PAYS DE LA LOIRE
Liberté
Égalité
Fraternité

Lycée TOUCHARD WASHINGTON

Dossier Final

BTS CIEL SESSION 2025 - IR

E6 – PROJET TECHNIQUE

Système de Surveillance météorologique connecté

Lycée : **Touchard Washington**	Ville : **LE MANS**	
Nom du projet :	**Système de surveillance météorologique connecté**	
N° du projet :	**TW3**	
Projet nouveau	Oui▣ Non▢	Projet interne : Oui▣ Non▢
Délai de réalisation	**150 heures**	Statut des étudiants : Formation initiale ▣ Apprentissage ▢
Spécialité des étudiants	ER▢ IR▣ Mixte▢	Nombre d'étudiants : **3 étudiants**
Professeurs responsables	**Philippe CRUCHET**, François MARTIN, Anthony LE CREN, Jilali KHAMLACH, Saïd LAHSIKA.	

Sommaire

1.Introduction au système

Le projet consiste à développer un système de surveillance météorologique connecté, basé sur une infrastructure réseau et un ensemble de capteurs pour collecter et afficher des données climatiques. Son objectif est de permettre un suivi en temps réel des conditions météorologiques locales à distance, de réaliser un stockage de ces données pour une analyse historique. Les domaines d'application sont nombreux, l'agriculture, la surveillance environnementale, ou même simplement pour les particuliers désireux de connaître précisément la météo

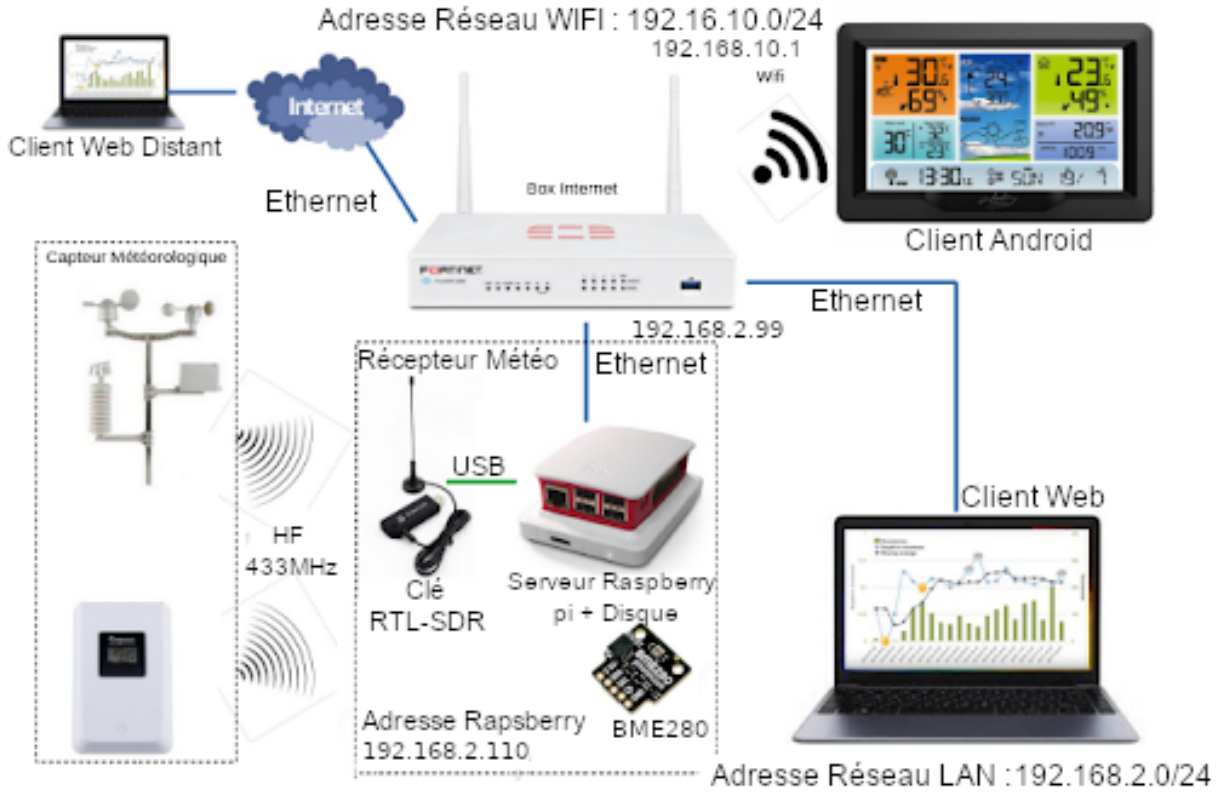


Image 1: Synoptique du système

Comme présenté ci-dessus, le système météorologique est centré autour d'un serveur basé sur un Raspberry, qui réceptionne les données des capteurs, les traitera afin de les stocker et de les diffuser en temps réel. Pour la réception des données, est faite à partir d'une clé RTL-DSR et de l'application RTL_433 qui permettra la traduction des trames. Les données seront donc stockées au sein de la Raspberry dans une base de données. La communication réseau est réalisée grâce à une box Internet. Les clients pourront se connecter à partir d'un navigateur web(en local et à distance) ou d'un appareil Android. Les clients recevront les données en temps réel, alors que les clients web auront accès aux données présentes dans la base de données.

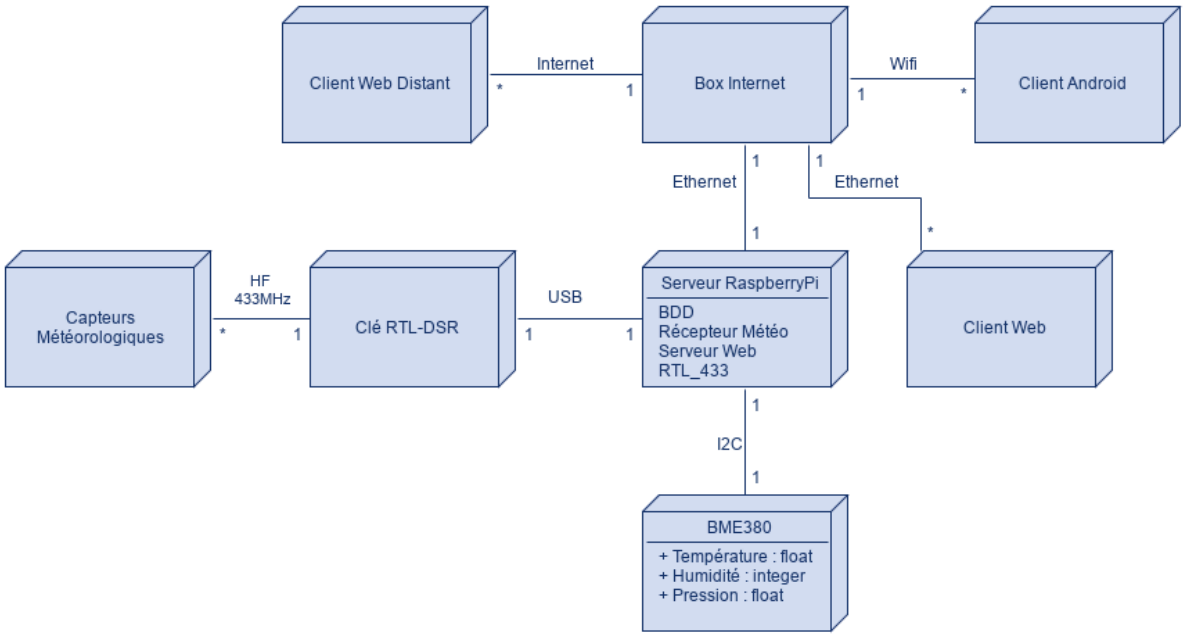


Image 2 : "Diagramme de déploiement UML"

Dans ce diagramme de déploiement on peut y voir les différents éléments qui vont constituer notre projet.

Afin de mieux comprendre ce diagramme voici une description des principaux éléments composant le système:

- **Box Internet** : Cette box est l'élément central de ce réseau, elle permet la communication entre le Serveur Raspberry et les éléments extérieurs au réseau. Elle est donc unique, mais plusieurs clients (Web et Android) peuvent s'y connecter.
- **Serveur Raspberry** : Le serveur raspberry permet la connexion entre la Box Internet et les capteurs météorologiques. Il est relié à une clé RTL afin de recevoir les données des capteurs. Celui-ci est unique car il permet de centraliser les données car il a aussi le rôle de base de données.
- **Clients** : il y a différents type de clients:
    - **Clients Android**: se connecte via une application mobile et accède aux informations en temps réel
    - **Clients Web**:
        - **Local**: les clients locaux peuvent accéder à l'historique des données relevé par les capteurs, en accédant à la base de données.
        - **Distant**: les clients distants ont les mêmes options que les clients locaux, cependant ils doivent s'authentifier avant d'y accéder

2.Descriptions des capteurs et des actionneurs

2.1.Fine Offset

2.1.1.Hygro/Thermomètre

2.1.1.1.Descriptif

Référence	WS-2300
Rôle	Déterminer la température, le taux d'humidité et alimenter le pluviomètre (WS-2300-16) et l'anémomètre (WS-2300-15)
Caractéristiques	➢ Dimensions : 71 x 73 x 136 mm ➢ Alimentation : Pile LR6, 1.5V

2.1.1.2.Entrées physiques

Grandeurs Physiques	Unité	Échelle de mesure	Précision	Type de grandeur
Température	°C	-29.9 à 69.9	0.1°C	Analogique
Humidité	%	20 à 95%	1%	Analogique

2.1.2.Anémomètre/Girouette

2.1.2.1.Descriptif

Référence	WS-2300-15
Rôle	Déterminer la vitesse et la direction du vent
Caractéristiques	➢ Dimensions : 60 x 197 x 290 mm ➢ Alimentation : Filaire (connexion à l'hygro/thermomètre (WS-2300))

2.1.2.2.Entrées physiques

Grandeurs Physiques	Unité	Échelle de mesure	Précision	Type de grandeur
Vitesse	km/h ou m/s	0km/h à 180km/h ou 1m/s à 50m/s	0.1m/s	Analogique
Direction	°	0 à 360	22.5° (à vérifier)	Analogique

2.1.3.Pluviomètre

2.1.3.1.Descriptif

Référence	WS-2300-16
Rôle	Déterminer le niveau de précipitation.2

Caractéristiques	➢ Dimensions : 140 x 70 x 137 mm ➢ Alimentation : Pile LR6, 1.5V

2.1.3.2.Entrées physiques

Grandeurs Physiques	Unité	Échelle de mesure	Précision	Type de grandeur
Précipitations	mm	0 à 999.9	0.1mm	Numérique

2.2.Capteurs Hygro/Thermo

2.2.1.Nexus-TH

2.2.1.1.Descriptif

Référence	Nexus-TH
Rôle	Déterminer la température et le taux d'humidité
Caractéristiques	➢ Dimensions : 140 x 70 x 137 mm ➢ Alimentation : Filaire (connexion à l'hygro/thermomètre (WS-2300))

2.2.1.2.Entrées physiques

Grandeurs Physiques	Unité	Échelle de mesure	Précision	Type de grandeur
Température	°C	-40 à 65	1°C	Analogique
Humidité	%	1 à 99%	5%	Analogique

2.2.2.Oregon

2.2.2.1.Descriptif

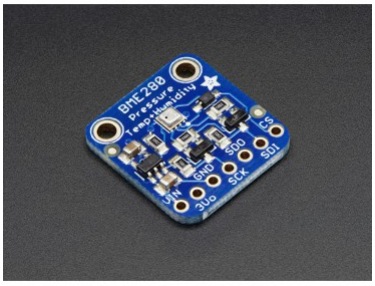
Référence	Oregon
Rôle	Déterminer la température et le taux d'humidité
Caractéristiques	➢ Dimensions : 140 x 70 x 137 mm ➢ Alimentation : Filaire (connexion à l'hygro/thermomètre (WS-2300))

2.2.2.2.Entrées physiques

Grandeurs Physiques	Unité	Échelle de mesure	Précision	Type de grandeur
Température	°C	30 à 60	0.1°C	Analogique
Humidité	%	5 à 95%	1%	Analogique

2.3.Capteur Local

2.3.1.Descriptif



Référence	BME280
Rôle	Déterminer la pression atmosphérique, la température et l'humidité au niveau du serveur
Caractéristiques	➢ Température: -40°C à +85°C: ±1°C ➢ Humidité: 0% à 100%: ±3% ➢ Pression Atmosphérique: 300 hPa à 1100 hPa: ±1Pa

2.3.2.Entrées physiques

Grandeurs Physiques	Unité	Échelle de mesure	Précision	Type de grandeur
Température	°C	-40 à 85	1°C	Analogique
Humidité	%	0 à 100	3%	Analogique
Pression atmosphérique	Pa	300 hPa à 1100 hPa	1 Pa	Analogique

3.Diagramme BPMN du récepteur et des clients Android

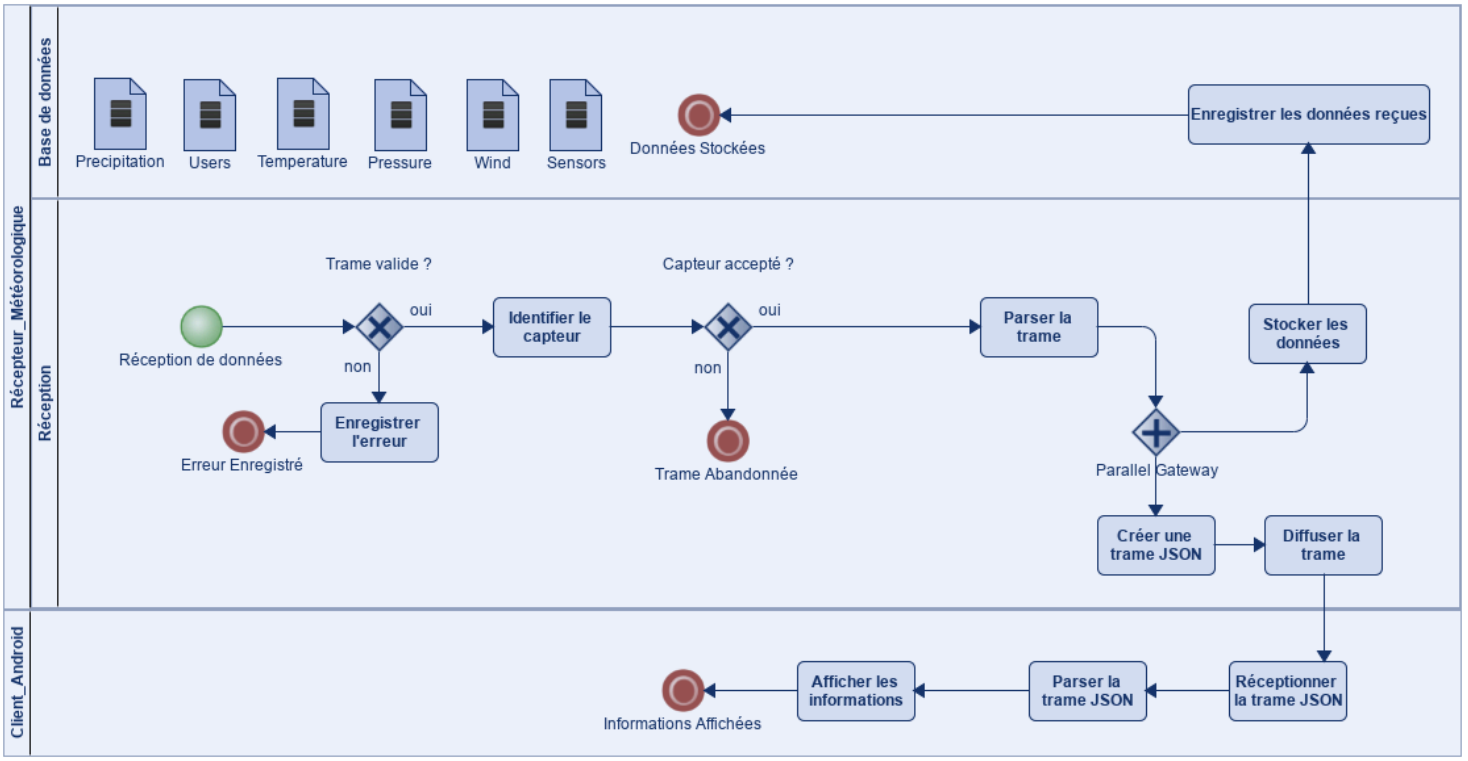


Image 3: Diagramme BPMN "Récepteur et Client Android"

Ce diagramme BPMN (Business Process Model and Notation) décrit le processus métier impliqué dans la réception, le traitement, le stockage et la diffusion des données météorologiques au sein d'un système. Le processus met en relation deux acteurs principaux :

- **Le récepteur de données météo** : Il collecte des informations climatiques en temps réel, telles que la température, l'humidité, la direction et la vitesse du vent, ainsi que les précipitations. Ces données sont ensuite traitées et enregistrées dans une base de données structurée.
- **Le client Android** : Une application cliente qui reçoit les données sous forme de trame JSON, les interprète et les affiche à l'utilisateur.

Le processus détaillé comprend les étapes suivantes :

- **Réception et traitement des données** : Les données météorologiques sont reçues, analysées et transformées en un format standardisé prêt à être stocké.
- **Stockage dans la base de données** : Les informations traitées sont sauvegardées dans une table de données dédiée.
- **Génération et diffusion de la trame JSON** : Une trame JSON contenant les données météo est générée et envoyée au client Android.
- **Traitement par le client Android** : Le client Android décode et affiche les informations pour l'utilisateur final.

4.Diagramme BPMN du client web

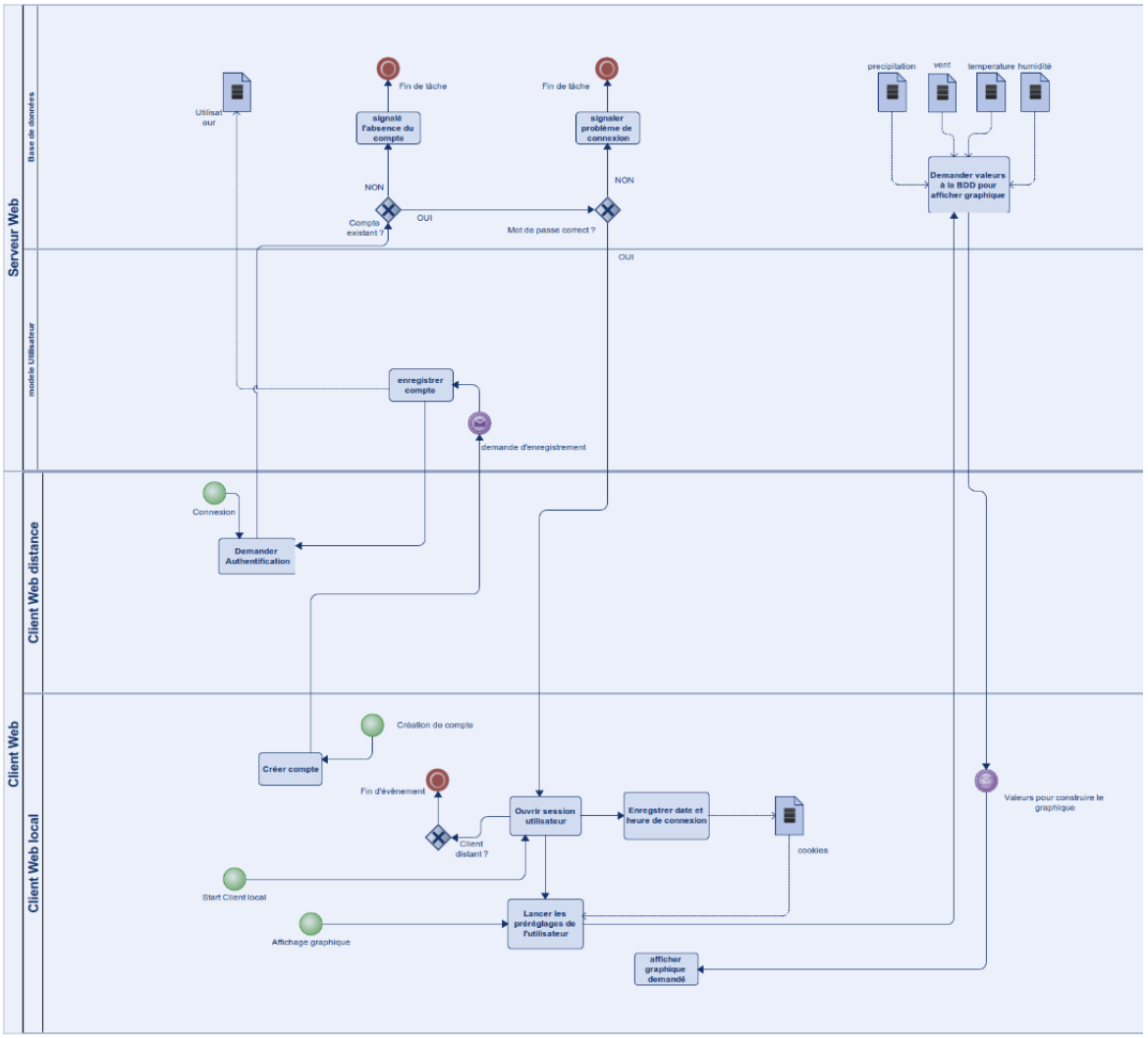


Image 4: Diagramme BPMN "client web"

Ce diagramme est un diagramme de processus BPMN (Business Process Model and Notation) qui représente le fonctionnement d'un système de gestion de compte et d'affichage de données météorologiques. Il est divisé en plusieurs pools et lanes correspondant aux différents acteurs et systèmes impliqués.

Structure du diagramme :

- **Pools et Lanes :**
  - **Serveur** : Il gère l'enregistrement des comptes, l'authentification et l'affichage des données.
  - **Client Web local** : S'occupe du lancement du client et de l'affichage graphique
- **Principaux processus représentés :**
  - **Connexion et authentification** : Un utilisateur tente de se connecter.Si le mot de passe est incorrect, une erreur est signalée.Sinon, la session utilisateur s'ouvre.

➢ **Affichage des données météorologiques** :

- Le système demande des valeurs à la base de données (température, humidité, vent, précipitation).
- Les valeurs sont utilisées pour afficher des graphiques.

Conclusion :

Ce diagramme BPMN modélise un processus d'authentification et de gestion de compte utilisateur avec une interaction entre un serveur, un client web distant et un client web local.

5.Dictionnaire des acteurs

Nom de l'acteur	Description
Capteurs météorologiques	Envoient des données météorologiques.
Utilisateur sous Android	Visualise les données localement en temps réel et configure des seuils d'alerte.
Utilisateur Web local	Visualise l'historique des différentes grandeurs météorologique sous forme graphique ou sous forme de tableaux
Utilisateur Web Distant	Visualise l'historique des différentes grandeurs météorologique sous forme graphique ou sous forme de tableaux mais doit s'authentifier au préalable.

6.Cas d'utilisations

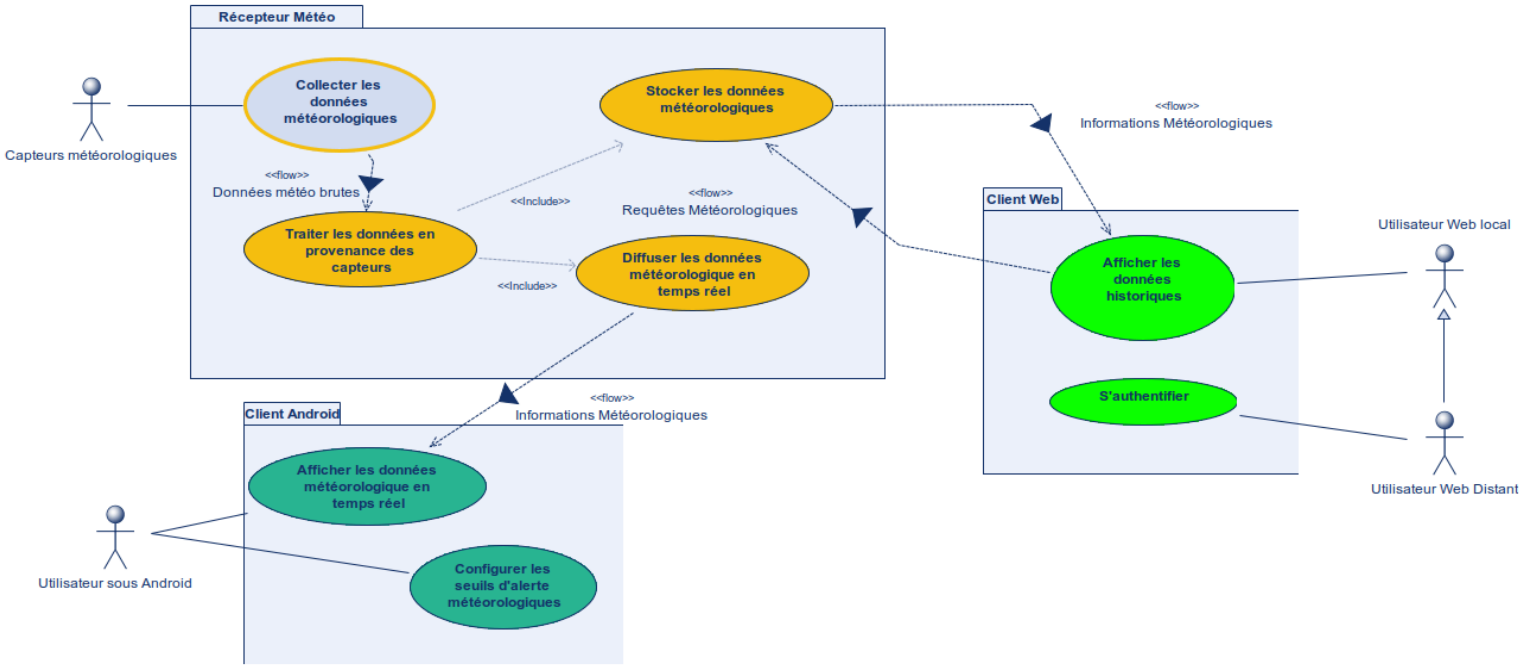


Image 5: Diagramme des cas d'utilisation

Sur ce diagramme de de cas d'utilisation se trouve trois systèmes avec les différents acteurs entrant en interaction avec le système. Il y a tout d'abord le récepteur météo les acteurs en lien avec ce cas sont les capteurs météorologiques. Celui-ci va collecter les données météorologiques par le biais d'un logiciel, celles-ci vont ensuite être traitées pour au final être à la fois stockées dans une base de données local au récepteur et être diffusées à des clients Android. Ensuite il y la classe client Android avec l'utilisateur sous android comme acteur et enfin la classe Client Web avec ces deux acteurs, utilisateur web distant et local.

7.Base de données

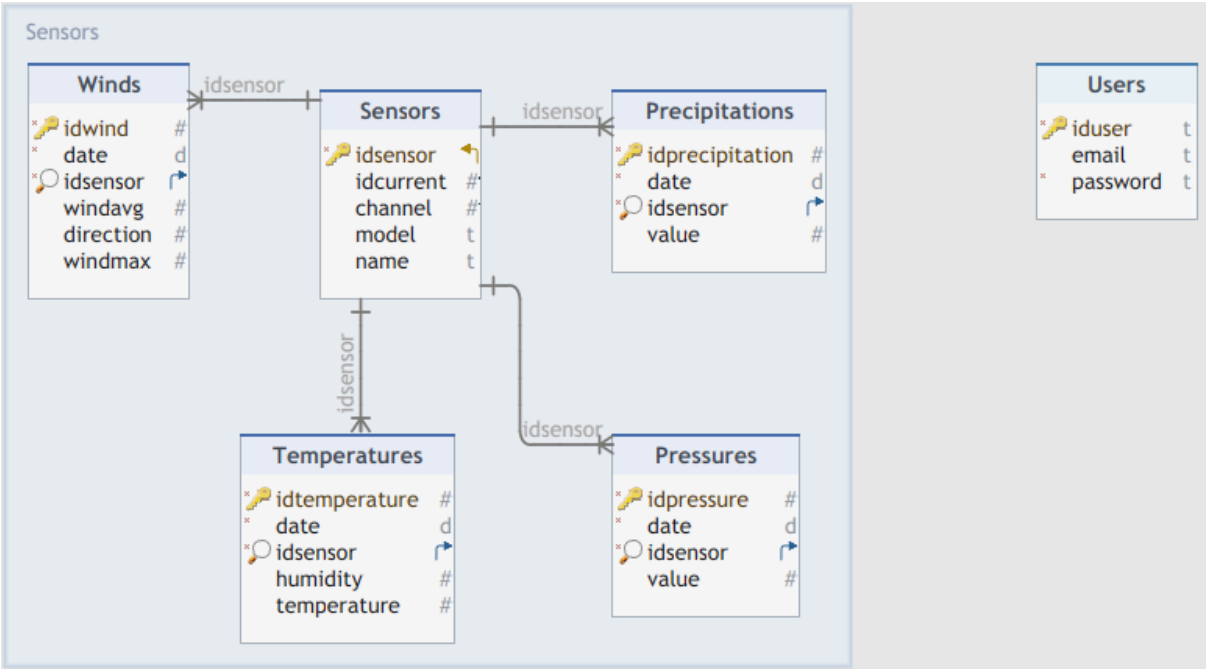


Image 6: Illustration de la Base de Données

La base de données est conçue pour centraliser et historiser les mesures issues de différents capteurs météorologiques. Elle repose sur une organisation relationnelle permettant de structurer les données par type de mesure tout en les reliant à leur capteur d'origine.

Structure générale :

- *La table **Sensors** contient les informations générales de chaque capteur (modèle, canal, nom, etc.). Elle sert de point central vers lequel toutes les mesures sont reliées.*
- *Les données mesurées sont réparties dans quatre tables spécialisées selon le type de paramètre mesuré :*
  - ***Temperatures** : stocke température et humidité.*
  - ***Pressures** : stocke la pression atmosphérique.*
  - ***Winds** : stocke la direction, la vitesse moyenne et la vitesse maximale du vent.*
  - ***Precipitations** : stocke la quantité de précipitations.*
- *Chaque table de mesure contient :*
  - *un identifiant propre à la mesure,*
  - *une clé étrangère vers le capteur (idsensor),*
  - *la date de la mesure,*
  - *les valeurs mesurées.*

*Table **Users** :*

*Une table **Users** est également présente pour gérer l'accès des utilisateurs à l'application (interface web par exemple), avec un identifiant, une adresse email et un mot de passe.*

Avantages de cette structure :

- *Séparation logique des types de données, ce qui facilite les requêtes spécifiques (ex. : afficher uniquement les températures).*
- *Historisation complète avec horodatage précis.*
- *Flexibilité : possibilité d'ajouter facilement de nouveaux types de capteurs ou mesures.*
- *Sécurité et gestion des droits via la table Users.*

8.Répartition du temps de travail

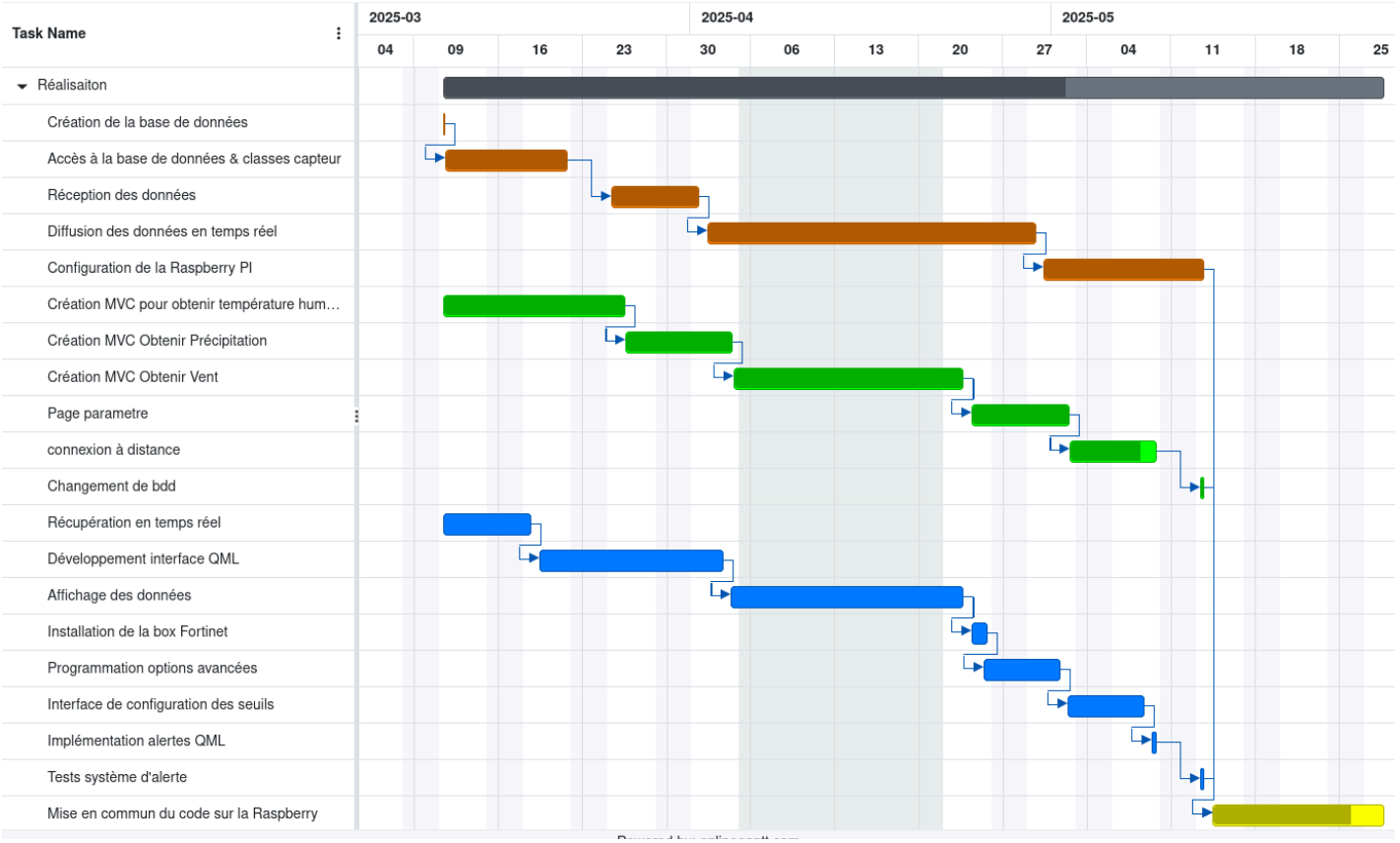


Image 7: Diagramme de Gantt

Étudiant 1:

Création de la base de données:
Cette tâche consiste à créer la structure initiale de la base de données, incluant les tables pour chaque type de données météorologiques et les relations nécessaires entre elles.

Accès à la base de données & classes capteur:
Développement des classes en Qt permettant d'accéder à la base de données via QSqlDatabase et de structurer les données des capteurs en objets exploitables pour le traitement et l'affichage.

Réception des données:
Mise en place de la lecture des trames issues de la clé RTL-SDR grâce à QProcess et parsing via QJsonObject, pour les transformer en données exploitables dans le programme.

Diffusion des données en temps réel:
Développement d'un serveur WebSocket pour transmettre en temps réel les données météorologiques aux clients Android ou aux interfaces Web.

Configuration du Raspberry Pi:
Installation des logiciels nécessaires, configuration réseau, gestion des services et automatisation du lancement de l'application au démarrage du Raspberry Pi.

Étudiant 2 :

Création MVC pour obtenir température :

Cette tâche consiste à développer la chaîne complète de récupération et d'affichage des températures et humidités, en suivant une architecture MVC. Cela inclut la création du modèle pour interroger la base MariaDB, le contrôleur en PHP pour formater les données en JSON, et la vue HTML/JS utilisant Highcharts pour l'affichage graphique.

Création MVC obtenir précipitations :

Même principe que la tâche précédente, mais appliqué aux données de précipitations. Le but est de permettre un affichage clair et dynamique des précipitations selon différents filtres temporels (par heure, jour, mois), en utilisant des requêtes adaptées côté modèle et un graphique à barres côté vue.

Création MVC obtenir vent :

Cette tâche concerne l'affichage des données de direction et de vitesse du vent. Elle a nécessité la mise en place d'un graphique polaire interactif avec Highcharts, en lien avec des données récupérées depuis la base. L'ensemble respecte la logique MVC avec des requêtes SQL spécifiques et un traitement PHP adapté.

Page paramètre :

Conception d'une interface permettant de modifier ou supprimer les capteurs. Cette page utilise un contrôleur pour envoyer les requêtes de mise à jour/suppression à la base via PHP, et une vue interactive en HTML/JavaScript avec mises à jour dynamiques.

Connexion à distance :

Implémentation d'un système d'accès sécurisé permettant de consulter et interagir avec le site météo à distance grâce à l'adresse IP.

Mise en commun du code sur la Raspberry:
Regroupement de l'ensemble des modules, tests d'intégration sur la Raspberry Pi, et finalisation du déploiement du projet avec synchronisation Git.