

Université privée de Marrakech  
Filière : génie informatique

# Projet de Fin de semestre

Application Web de gestion et d'optimisation  
énergétique d'un hôtel

Réalisé par:

Youssra Boughazi

FatimaEzzahra Boutaghzout

Hajar Hachman

Hamid Marbouh

Mohamed Ayoub EL Ghazali

Encadré par :

M.Said GOUNANE

# Remerciements

Nous tenons à exprimer notre sincère gratitude à notre encadrant, le professeur Said GOUNANE, pour son soutien, ses conseils précieux et son accompagnement tout au long de ce projet. Son expertise et sa disponibilité ont été d'une aide inestimable pour nous guider à chaque étape, depuis la conception jusqu'à la réalisation de cette application. Sans son encadrement bienveillant et ses orientations éclairées, ce projet n'aurait pas pu aboutir avec autant de succès. Merci, Monsieur, pour votre confiance et votre engagement à nos côtés.

# Résumé

Ce projet consiste à concevoir et développer une application web intelligente de gestion énergétique destinée à un hôtel, intitulée Smart Energy Hotel Manager. L'objectif principal est d'assurer un suivi précis de la consommation d'énergie, d'optimiser l'utilisation des équipements énergivores et de réduire les coûts énergétiques tout en améliorant l'efficacité opérationnelle.

L'application permet de superviser la consommation énergétique par étage et par chambre, en s'appuyant sur des capteurs intelligents (température, luminosité et mouvement) installés dans chaque chambre. Ces capteurs collectent des données en temps réel afin de détecter les situations de surconsommation, d'inoccupation ou d'anomalie technique.

Le système repose sur deux acteurs principaux :

- L'Administrateur, chargé de la gestion globale du système (étages, chambres, capteurs, utilisateurs, paramètres énergétiques et tableaux de bord).
- Le Technicien, responsable du suivi des alertes, de la maintenance des équipements et de la résolution des incidents techniques.

Développée selon une architecture moderne Frontend / Backend / Base de données, l'application utilise des technologies web actuelles permettant une interface intuitive, sécurisée et évolutive. Ce projet s'inscrit dans une démarche de transition énergétique, en contribuant à une gestion intelligente, durable et automatisée des ressources énergétiques dans le secteur hôtelier.

# Abstract

This project focuses on the design and development of a smart web-based energy management system for hotels, named Smart Energy Hotel Manager. The main objective is to monitor, analyze, and optimize energy consumption in order to reduce operational costs and improve energy efficiency.

The system enables real-time supervision of energy usage at both floor and room levels through the use of intelligent sensors, including temperature, light, and motion sensors installed in each room. These sensors collect and transmit data that allow the detection of energy anomalies, unoccupied rooms, and technical malfunctions.

The application involves two main user roles: the Administrator, who manages the overall system configuration, floors, rooms, sensors, and energy monitoring dashboards; and the Technician, who is responsible for handling alerts, performing maintenance tasks, and resolving technical issues.

Built on a modern web architecture combining a secure backend, an interactive frontend, and a relational database, the proposed solution supports scalability, reliability, and ease of use. This project contributes to sustainable development by promoting intelligent and automated energy management in the hospitality sector.

# Table des matières

Remerciements .....	2
Résumé .....	3
Abstract .....	4
Table des matières .....	5
Table des annexes .....	7
<b>Chapitre 1 : Contexte général du projet .....</b>	<b>8</b>
1.Cadre général du projet .....	9
2.Problématique .....	9
3.Objectifs du projet .....	10
4.Analyse des besoins .....	11
5.Organisation et planning du projet .....	11
<b>Chapitre 2 : État de l'art et solutions existantes .....</b>	<b>12</b>
1.Évolution des systèmes de gestion énergétique .....	13
2.La gestion énergétique dans le secteur hôtelier .....	13
3.Étude comparative des solutions existantes.....	13
4.Limites des solutions actuelles .....	14
<b>Chapitre 3 : Analyse et conception du système .....</b>	<b>15</b>
1.Analyse des besoins .....	16
2.Besoins fonctionnels .....	16
3.Besoins non fonctionnels .....	17
4.Conception architecturale .....	17
5.Architecture globale du système .....	17
6.Architecture technique .....	18
7.Méthodes d'analyse et de conception .....	19
8.Approche orientée objet et UML et modélisation .....	19
9.Architecture MVC et ses adaptations .....	21
10.Design Patterns et bonnes pratiques .....	22
<b>Chapitre 4 : Réalisation et implémentation .....</b>	<b>23</b>
1.Technologies et outils utilisés .....	24
2.Gestion des capteurs et équipements .....	24
3.Gestion de la consommation énergétique et des alertes .....	28
<b>Chapitre 5 : Tests, résultats et validation .....</b>	<b>30</b>
1.stratégie de test .....	31
3.Tests fonctionnels .....	31
4.Présentation de l'application .....	32
5.Conclusion générale et perspectives .....	37

# Table des Annexes

Figure 1 : Interface de communication MQTT (Broker et topics)

Figure 2: Environnement Arduino – Lecture des données des capteurs

Figure3 : Structure et contenu de la base de données MariaDB

Figure 4 : Affichage des données des capteurs dans l'environnement Arduino

Figure 5 : Interface de communication MQTT pour la réception des données des capteurs

Figure 6 : Implémentation du backend – Traitement des données MQTT (Laravel)

Figure 7: Tableau de bord Administrateur du système Smart Energy Hotel Manager

Figure 8: Tableau de bord Administrateur du système Smart Energy Hotel Manager

# Chapitre 1 :

## Contexte générale du projet

## 1. Cadre générale :

Dans un contexte marqué par l'augmentation continue de la consommation énergétique et par les enjeux environnementaux actuels, la maîtrise de l'énergie est devenue une priorité stratégique pour les établissements hôteliers. Ces derniers utilisent quotidiennement de nombreux équipements énergivores tels que les systèmes de climatisation, d'éclairage et de chauffage, ce qui engendre des coûts élevés et une forte dépendance énergétique. Avec l'évolution des technologies numériques et de l'Internet des Objets (IoT), il est désormais possible de collecter, analyser et exploiter des données énergétiques en temps réel à l'aide de capteurs intelligents. Ces technologies offrent de nouvelles opportunités pour optimiser la gestion énergétique, améliorer l'efficacité opérationnelle et réduire l'impact environnemental des hôtels. C'est dans ce cadre que s'inscrit le projet Smart Energy Hotel Manager, qui vise à concevoir et développer une application web intelligente dédiée à la supervision et à l'optimisation de la consommation énergétique au sein d'un hôtel. Le système permet de suivre la consommation d'énergie par étage et par chambre grâce à l'utilisation de capteurs de température, de luminosité et de mouvement installés dans les différentes zones de l'établissement.

## 2. Problématique

Dans les établissements hôteliers, la gestion de la consommation énergétique représente un défi majeur en raison de la diversité des équipements utilisés, de la multiplicité des chambres et des étages, ainsi que de la variation constante du taux d'occupation. Une grande partie de l'énergie est souvent consommée de manière inefficace, notamment dans les chambres inoccupées ou en dehors des périodes d'utilisation réelle. Les méthodes traditionnelles de gestion énergétique reposent généralement sur des contrôles manuels ou sur des systèmes peu automatisés, ce qui rend difficile la surveillance en temps réel, la détection rapide des anomalies et la prise de décisions optimales. De plus, l'absence d'un suivi détaillé par étage et par chambre limite la capacité des responsables à identifier précisément les sources de surconsommation. Par ailleurs, le manque d'outils centralisés permettant de collecter, analyser et visualiser les données issues des capteurs énergétiques complique le travail du personnel technique. Les interventions sont souvent réactives plutôt que préventives, entraînant des retards dans la résolution des pannes et une augmentation des coûts de maintenance.

Face à ces constats, il devient nécessaire de mettre en place une solution intelligente, centralisée et automatisée permettant de superviser la consommation énergétique en temps réel, de détecter les anomalies, d'optimiser l'utilisation des équipements et d'assister le personnel hôtelier dans la prise de décision. Cette problématique constitue le point de départ du projet Smart Energy Hotel Manager.

### 3. Objectifs

Le projet Smart Energy Hotel Manager a pour objectif principal de concevoir et développer une application web intelligente permettant de superviser, analyser et optimiser la consommation énergétique au sein d'un établissement hôtelier. Cette solution vise à assurer un suivi en temps réel de l'énergie consommée par étage et par chambre, à centraliser la gestion des équipements énergétiques, à détecter les anomalies et les situations de surconsommation, et à faciliter les interventions techniques grâce à un système d'alertes et de tableaux de bord. À travers cette approche, le projet contribue à la réduction des coûts énergétiques, à l'amélioration de l'efficacité opérationnelle et à la promotion d'une gestion durable et responsable des ressources énergétiques.

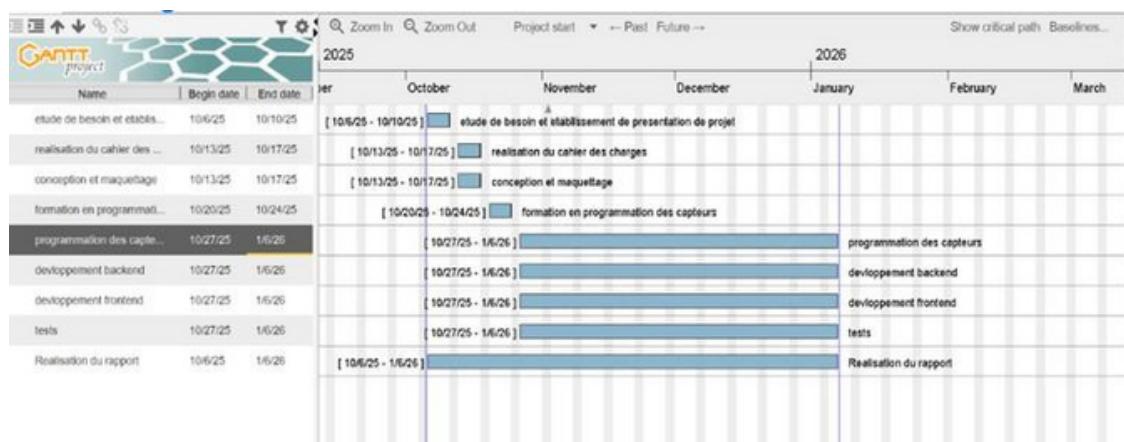
Dans le cadre de ce projet, une attention particulière est accordée à l'utilisation des technologies modernes telles que les capteurs intelligents et les applications web interactives afin d'assurer une gestion énergétique précise et automatisée. Le système repose sur une architecture sécurisée et évolutive, permettant une collecte fiable des données, une visualisation claire des informations énergétiques et une adaptation aux besoins futurs de l'établissement. De plus, la solution est conçue pour faciliter le travail du personnel hôtelier, notamment l'administrateur et le technicien, en améliorant la coordination, la réactivité face aux incidents et la qualité de la maintenance des équipements.

## 4. Analyse des Besoins

L'analyse des besoins vise à identifier les fonctionnalités essentielles du système de gestion énergétique hôtelier. L'application doit permettre une gestion centralisée des étages, des chambres et des capteurs afin d'assurer un suivi précis de la consommation énergétique. Les données collectées à partir des capteurs de température, de luminosité et de mouvement doivent être analysées en temps réel pour détecter les anomalies et les situations de surconsommation. Le système doit répondre aux besoins de l'administrateur pour la configuration et la supervision globale, ainsi qu'aux besoins du technicien pour le traitement des alertes et la maintenance. Il est également nécessaire de garantir la sécurité des données, la performance du système et la fiabilité des informations. Enfin, l'application doit être évolutive et facile à utiliser afin de s'adapter aux besoins futurs de l'établissement.

## 5. Planning du projet

Le projet est planifié selon une organisation progressive en plusieurs phases afin d'assurer une réalisation structurée et efficace. La première phase est consacrée à l'analyse des besoins et à la rédaction du cahier des charges. Ensuite, une phase de conception est réalisée pour définir l'architecture du système et les différents diagrammes UML. La phase de développement comprend l'implémentation du backend et du frontend de l'application. Des tests fonctionnels et techniques sont ensuite effectués afin de valider le bon fonctionnement du système. Enfin, la dernière phase concerne la documentation du projet et la préparation de la présentation finale.



## Chapitre 2 : État de l'art et solutions existantes

## 1. Évolution des systèmes de gestion énergétique

Les systèmes de gestion énergétique ont connu une évolution significative avec l'avancement des technologies numériques. Initialement basés sur des contrôles manuels et des relevés périodiques, ces systèmes sont devenus progressivement automatisés grâce à l'intégration de capteurs et de systèmes informatisés. L'apparition de l'Internet des Objets (IoT) a permis la collecte de données en temps réel et une meilleure analyse de la consommation énergétique. Aujourd'hui, les solutions modernes offrent des fonctionnalités avancées telles que la supervision à distance, l'automatisation et l'optimisation intelligente de l'énergie.

## 2. La gestion énergétique dans le secteur hôtelier

Dans le secteur hôtelier, la gestion énergétique représente un enjeu majeur en raison de la consommation élevée liée à l'éclairage, au chauffage et à la climatisation. La variation du taux d'occupation des chambres complique davantage le contrôle de l'énergie. Une gestion inefficace entraîne une surconsommation et des coûts élevés. Ainsi, les hôtels ont de plus en plus recours à des systèmes intelligents permettant de suivre la consommation par chambre et par étage, d'optimiser l'utilisation des équipements et de réduire l'impact environnemental.

## 3. Étude comparative des solutions existantes

Plusieurs solutions commerciales de gestion énergétique existent actuellement sur le marché, proposant des fonctionnalités telles que le suivi de la consommation, la gestion des équipements et la génération de rapports. Certaines solutions intègrent des capteurs intelligents et des tableaux de bord analytiques. Cependant, ces systèmes sont souvent coûteux, complexes à déployer ou peu adaptés aux besoins spécifiques de chaque établissement. Le benchmark permet ainsi d'identifier les fonctionnalités pertinentes à intégrer et les limites à éviter lors de la conception du système proposé.

## 4. Limites des solutions actuelles

Malgré l'existence de plusieurs solutions de gestion énergétique dans le secteur hôtelier, celles-ci présentent encore un ensemble de limites qui freinent leur efficacité et leur adoption à grande échelle. Tout d'abord, la majorité des systèmes actuels ne proposent pas une supervision

centralisée de l'ensemble des équipements énergétiques. Les données sont souvent dispersées entre plusieurs outils ou interfaces, ce qui complique le suivi global de la consommation et rend la prise de décision moins efficace.

Ensuite, le manque d'automatisation intelligente constitue une contrainte majeure. De nombreuses solutions se limitent à un simple affichage des données sans proposer d'actions automatiques en cas de surconsommation ou d'anomalie. Cela oblige les gestionnaires et les techniciens à intervenir manuellement, entraînant une perte de temps et une augmentation des coûts.

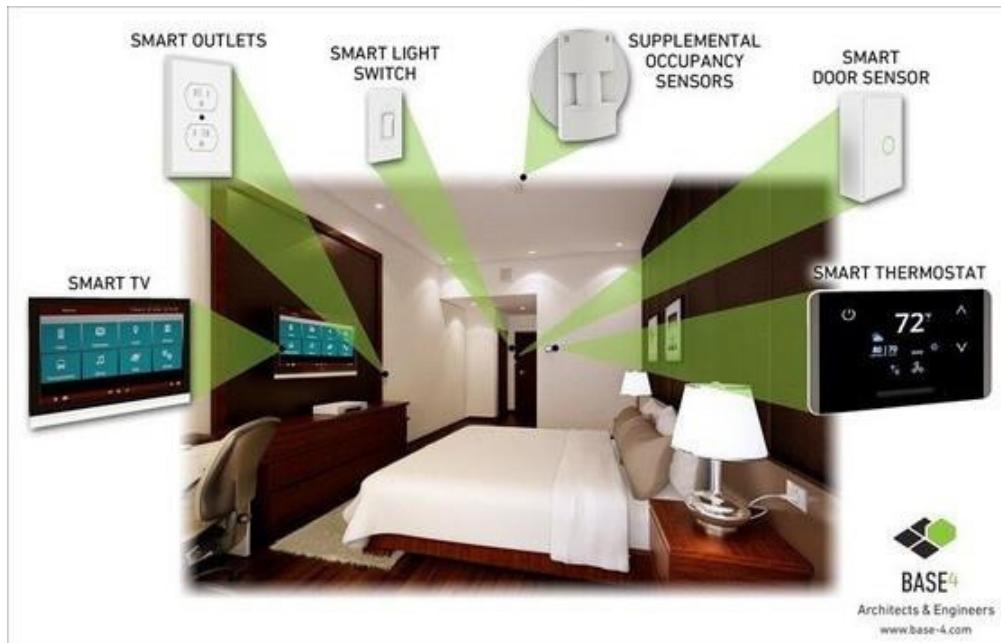
Par ailleurs, l'absence de détection proactive des pannes est un problème fréquent. Les systèmes existants identifient souvent les incidents après leur apparition, au lieu de les anticiper. Cette approche réactive peut provoquer des interruptions de service prolongées et un gaspillage énergétique important.

De plus, la complexité des interfaces représente un frein pour les utilisateurs techniques.

Certaines plateformes sont difficiles à prendre en main, nécessitent une formation avancée ou manquent de tableaux de bord clairs et intuitifs, ce qui réduit leur efficacité opérationnelle.

Enfin, le manque de flexibilité et d'évolutivité limite l'adaptation des solutions existantes aux besoins spécifiques de chaque hôtel. L'intégration de nouveaux capteurs, l'ajout de fonctionnalités ou l'évolution vers des architectures modernes sont souvent coûteux et techniquement contraignants.

Face à ces limites, le projet Smart Energy Hotel Manager propose une solution moderne, centralisée et évolutive, intégrant une automatisation intelligente et une gestion simplifiée, répondant ainsi aux insuffisances des solutions actuelles.



# Chapitre3 : Analyse et conception

## 1. Analyse des besoins

L'analyse des besoins constitue une étape essentielle dans la réalisation du projet Smart Energy Hotel Manager. Elle permet d'identifier précisément les attentes des utilisateurs, les fonctionnalités attendues ainsi que les contraintes techniques et organisationnelles du système. Cette analyse vise à garantir que la solution développée répond efficacement aux problématiques liées à la gestion et à l'optimisation de la consommation énergétique dans le secteur hôtelier. Le système repose sur deux acteurs principaux :

- Administrateur : responsable de la supervision globale du système, de la gestion des ressources énergétiques et de la prise de décisions stratégiques.
- Technicien : chargé du suivi technique, de la maintenance des équipements et de la résolution des incidents.

## 2. Besoins fonctionnels

L'analyse des besoins constitue une étape essentielle dans la réalisation du projet Smart Energy Hotel Manager. Elle permet d'identifier précisément les attentes des utilisateurs, les fonctionnalités attendues ainsi que les contraintes techniques et organisationnelles du système. Cette analyse vise à garantir que la solution développée répond efficacement aux problématiques liées à la gestion et à l'optimisation de la consommation énergétique dans le secteur hôtelier. Le système repose sur deux acteurs principaux :

- Administrateur : responsable de la supervision globale du système, de la gestion des ressources énergétiques et de la prise de décisions stratégiques.
- Technicien : chargé du suivi technique, de la maintenance des équipements et de la résolution des incidents.

L'administrateur doit disposer d'un ensemble de fonctionnalités lui permettant de gérer et superviser l'ensemble du système énergétique :

- Se connecter de manière sécurisée au système
- Gérer les comptes des techniciens
- Gérer les chambres (création, modification, suppression)
- Gérer les capteurs énergétiques associés aux chambres
- Superviser la consommation énergétique globale
- Consulter un tableau de bord interactif
- Configurer des modes automatiques (Éco, Nuit, Maintenance)
- Recevoir et gérer les alertes de surconsommation ou d'anomalie
- Générer des rapports énergétiques et statistique

Le technicien doit pouvoir assurer efficacement la maintenance et le suivi technique :

- Se connecter au système
- Consulter les données des capteurs
- Recevoir et consulter les alertes
- Identifier les anomalies et pannes techniques
- Signaler et résoudre les incidents
- Suivre l'historique des interventions
- Consulter les statistiques énergétiques par zone ou par équipement

### 3.Besoins non fonctionnels

En plus des fonctionnalités, le système doit répondre à des exigences non fonctionnelles garantissant sa qualité et sa fiabilité :

- Sécurité : authentification sécurisée et gestion des droits d'accès
- Performance : affichage fluide et mise à jour rapide des données
- Disponibilité : système accessible en continu
- Ergonomie : interface claire et facile à utiliser
- Scalabilité : possibilité d'ajouter de nouveaux capteurs ou fonctionnalités
- Maintenance : code structuré, documenté et versionné

### 4. Conception architecturale

La conception architecturale définit la structure globale du système, la manière dont ses composants interagissent ainsi que les technologies utilisées. Elle vise à garantir une application modulaire, évolutive, maintenable et sécurisée, capable de traiter des données énergétiques issues de capteurs et de les présenter sous forme de tableaux de bord et d'alertes.

### 5. Architecture globale du système

L'architecture globale de Smart Energy Hotel Manager repose sur une séparation claire entre les couches de présentation, de traitement et de stockage. Le système est conçu autour de deux profils utilisateurs : Administrateur et Technicien.

Le fonctionnement global est le suivant :

1. Authentification et contrôle d'accès
2. Les utilisateurs (Admin/Technicien) s'authentifient via un mécanisme sécurisé. Le système applique ensuite les permissions selon le rôle.
3. Collecte et traitement des données énergétiques
4. Les données provenant des capteurs (température, luminosité, mouvement, etc.) sont envoyées au backend via des endpoints API. Le backend enregistre les mesures et peut déclencher des alertes en cas d'anomalie (surconsommation, incohérence, capteur inactif...).

## 6. Architecture technique (Frontend / Backend / Base de données)

Le système adopte une architecture 3-tiers (3 couches), basée sur une API REST :

### A. Frontend (React.js)

Le frontend représente la couche de présentation. Il assure :

- l'interface utilisateur (Admin/Technicien),
- l'affichage des dashboards (graphiques, statistiques),
- la consultation et gestion des alertes,
- la gestion des chambres et capteurs (selon rôle),
- la communication avec le backend via Axios,
- un design responsive grâce à Tailwind CSS.
- 

### B. Backend (Laravel 11 – API REST)

Le backend constitue la couche métier. Il gère :

l'authentification sécurisée (JWT/Sanctum),  
la gestion des rôles et permissions (Admin/Technicien),  
les règles métiers (consommation, anomalies, alertes),  
les endpoints API (CRUD chambres, capteurs, utilisateurs, alertes, interventions),  
la validation des données et la sécurité (middlewares, validation requests).

- 

### C. Base de données (MySQL)

La base de données assure la persistance et l'historisation des informations :

- utilisateurs (admin/technicien),
- chambres,
- capteurs,
- mesures énergétiques,
- alertes,
- interventions,
- rapports/statistiques (si stockés).

Avantages de cette architecture :

- Séparation claire des responsabilités
- Facilité de maintenance (frontend et backend indépendants)
- Évolutivité (ajout futur de nouveaux capteurs ou modules)
- Sécurité renforcée via contrôle d'accès côté API
- Performance grâce à une API optimisée et des requêtes SQL structurées

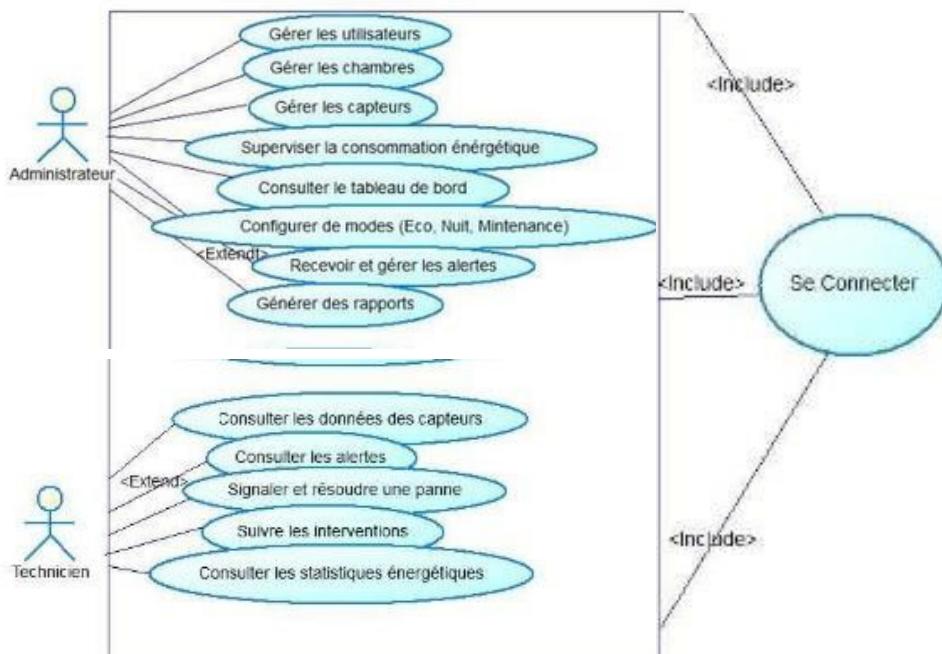
## 7. Méthodes d'analyse et de conception

Dans le cadre du projet Smart Energy Hotel Manager, des méthodes d'analyse et de conception reconnues ont été adoptées afin de garantir la qualité, la cohérence et la maintenabilité du système. Ces méthodes permettent de structurer les besoins, de modéliser le système et de faciliter sa mise en œuvre technique.

## 8. Approche orientée objet et UML

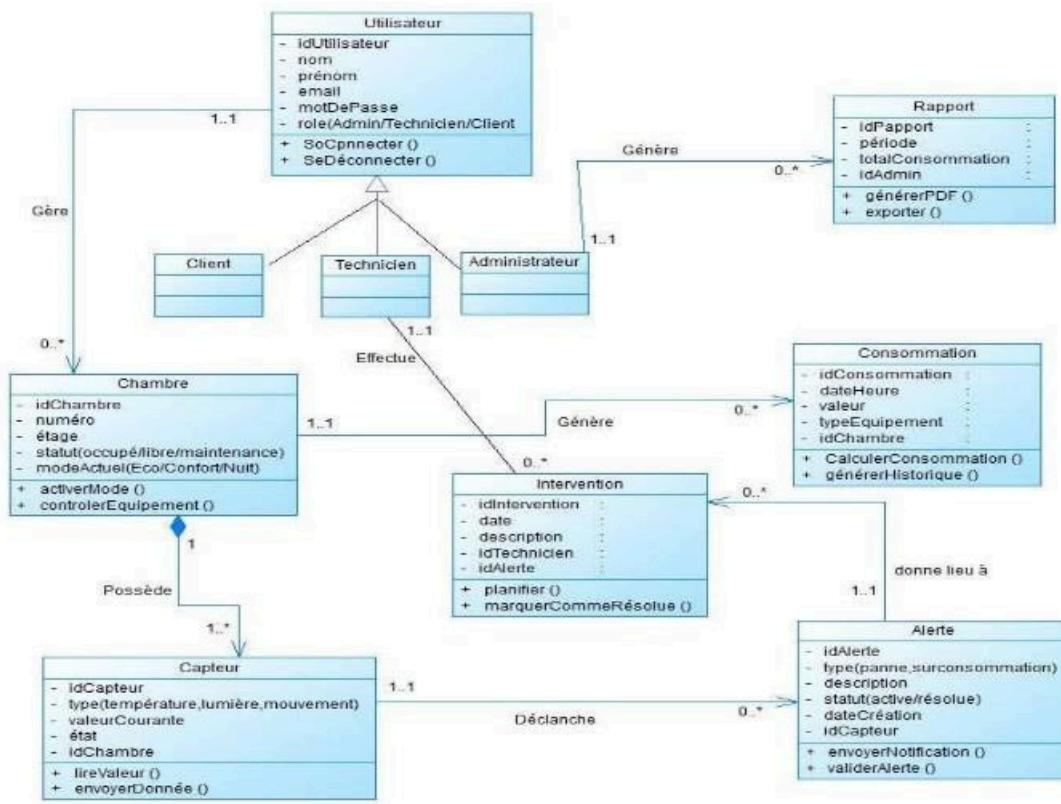
L'approche orientée objet a été retenue pour analyser et concevoir le système, car elle permet de représenter le projet sous forme d'entités réelles interagissant entre elles. Chaque composant du système est modélisé comme un objet possédant des attributs et des comportements, ce qui facilite la compréhension, l'évolution et la maintenance du système. Dans ce cadre, le langage de modélisation UML (Unified Modeling Language) a été utilisé pour représenter graphiquement les différents aspects du système. Les principaux diagrammes UML exploités sont :

### Diagramme de cas d'utilisation :



Il permet d'identifier les interactions entre les acteurs du système (Administrateur et Technicien) et les fonctionnalités offertes par l'application.

### Diagramme de classes :

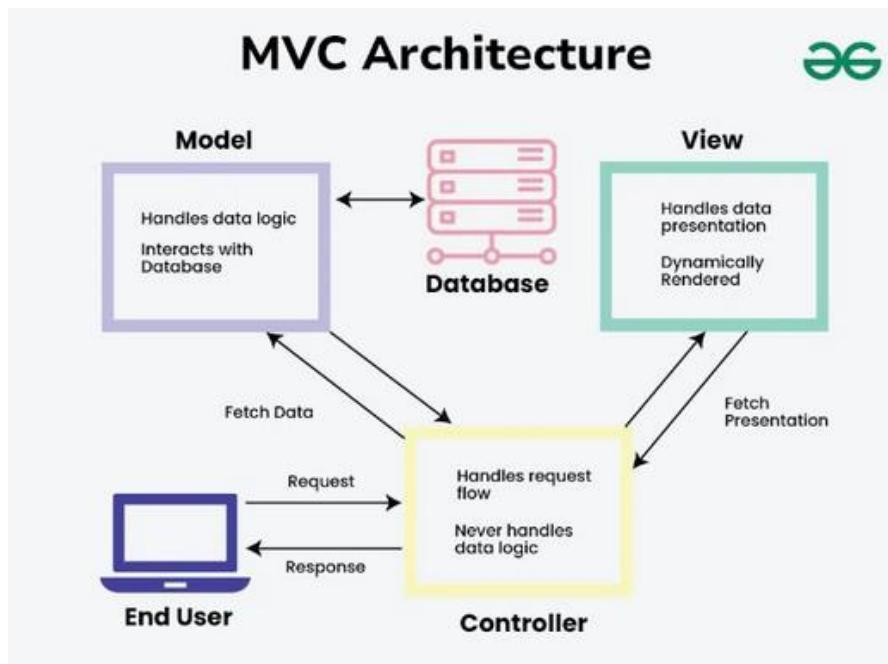


Il décrit la structure statique du système en mettant en évidence les classes principales (Utilisateur, Chambre, Capteur, Consommation, Alerte, Intervention, etc.), leurs attributs, méthodes et relations.

## 9. Architecture MVC et ses adaptations

Pour la réalisation du système, l'architecture MVC (Model – View – Controller) a été adoptée, car elle favorise une séparation claire des responsabilités et améliore la maintenabilité de l'application.

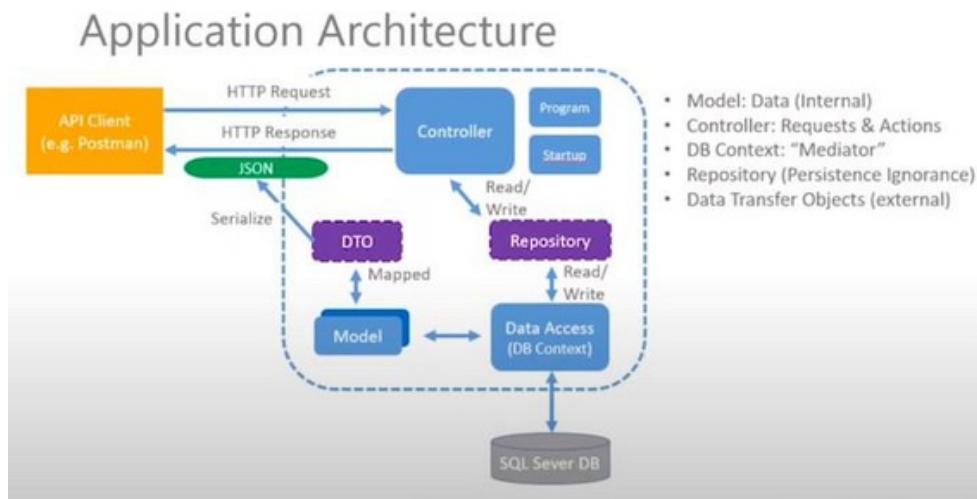
- Model :
- Représente les données et la logique métier du système. Dans le projet, les modèles Laravel correspondent aux entités telles que les utilisateurs, chambres, capteurs, consommations et alertes. Ils interagissent directement avec la base de données MySQL.
- View :
- Correspond à la couche de présentation. Elle est assurée par le frontend React, qui se charge d'afficher les interfaces utilisateurs (tableaux de bord, formulaires, graphiques) de manière dynamique et responsive.
- Controller :
- Joue le rôle d'intermédiaire entre les modèles et les vues. Les contrôleurs Laravel reçoivent les requêtes du frontend, appliquent les règles métier, puis retournent les réponses via des API REST.



Architecture MVC (Model – View – Controller)

## 10. Design Patterns et bonnes pratiques

Dans le projet Smart Energy Hotel Manager, l'utilisation de design patterns et de bonnes pratiques de développement permet d'assurer une application structurée, maintenable, évolutive et sécurisée. Ces choix facilitent la compréhension du système, la maintenance du code et l'ajout de nouvelles fonctionnalités à long terme.



Architecture applicative basée sur une API REST (Laravel)

# Chapitre 4 : Réalisation et implémentation

## 1. Technologies et outils utilisés

La réalisation du projet repose sur des technologies modernes et largement utilisées dans le développement d'applications web :

Backend : Laravel 11

Framework PHP utilisé pour le développement de l'API REST, la gestion de la logique métier, la sécurité et l'accès aux données.

Frontend : React.js

Bibliothèque JavaScript utilisée pour la création d'interfaces utilisateur dynamiques et réactives.

MariaDB (Base de données)

Système de gestion de base de données relationnelle pour le stockage des informations.

## 2. Gestion des capteurs et équipements

La gestion des capteurs constitue un élément central du système Smart Energy Hotel Manager. Elle permet la collecte, la transmission et l'analyse des données énergétiques afin d'assurer une supervision efficace et une optimisation de la consommation. Dans ce projet, les capteurs BMP, AH et le microcontrôleur ESP32 sont utilisés pour mesurer et transmettre les informations environnementales.

## Arduino (ESP32 – IoT)

L'environnement Arduino est utilisé pour programmer le microcontrôleur ESP32, qui joue le rôle de passerelle entre les capteurs physiques et le système informatique.

Rôles principaux :

- lecture des données issues des capteurs (BMP, AH),
- traitement et formatage des mesures,
- connexion au réseau Wi-Fi,
- envoi des données vers le serveur via le protocole MQTT.

L'utilisation d'Arduino permet un développement simple, rapide et fiable des composants IoT du système.

## MQTT (Message Queuing Telemetry Transport)

MQTT est un protocole de communication léger, spécialement conçu pour les systèmes IoT. Il est utilisé pour assurer la transmission des données entre l'ESP32 et le serveur.

Fonctionnement dans le projet :

- l'ESP32 publie les données des capteurs sur des topics MQTT,
- le serveur (backend) s'abonne à ces topics,
- les données sont reçues en temps réel et stockées dans la base de données.

Avantages de MQTT :

- faible consommation de bande passante,
- communication rapide et fiable,
- idéal pour les systèmes distribués et temps réel.

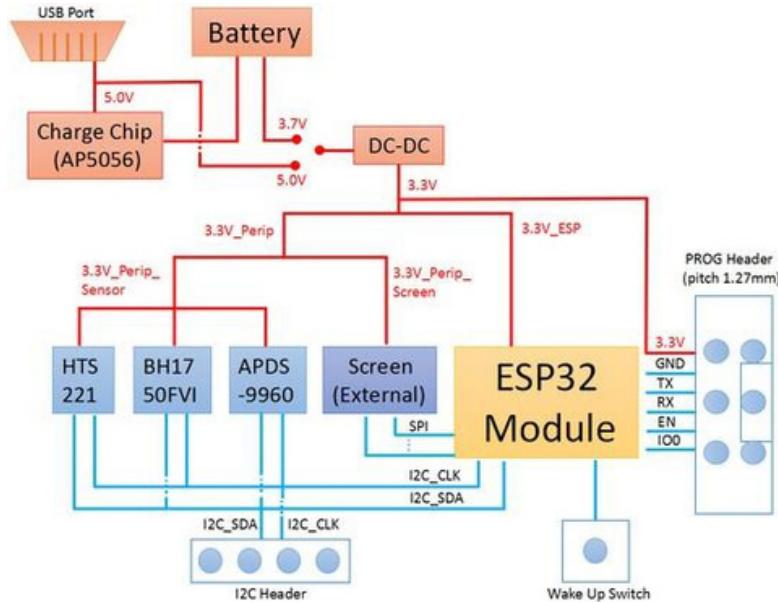
## Visual Studio Code (VS Code)

Visual Studio Code est l'éditeur de code principal utilisé pour le développement du projet.

Il est utilisé pour :

- le développement du backend (Laravel),
- le développement du frontend (React),
- l'édition du code Arduino,
- la gestion des fichiers et du versionnement Git.

Grâce à ses extensions (PHP, JavaScript, Arduino, Git), VS Code offre un environnement de développement productif et centralisé.



Architecture matérielle et alimentation du module ESP32 avec capteurs

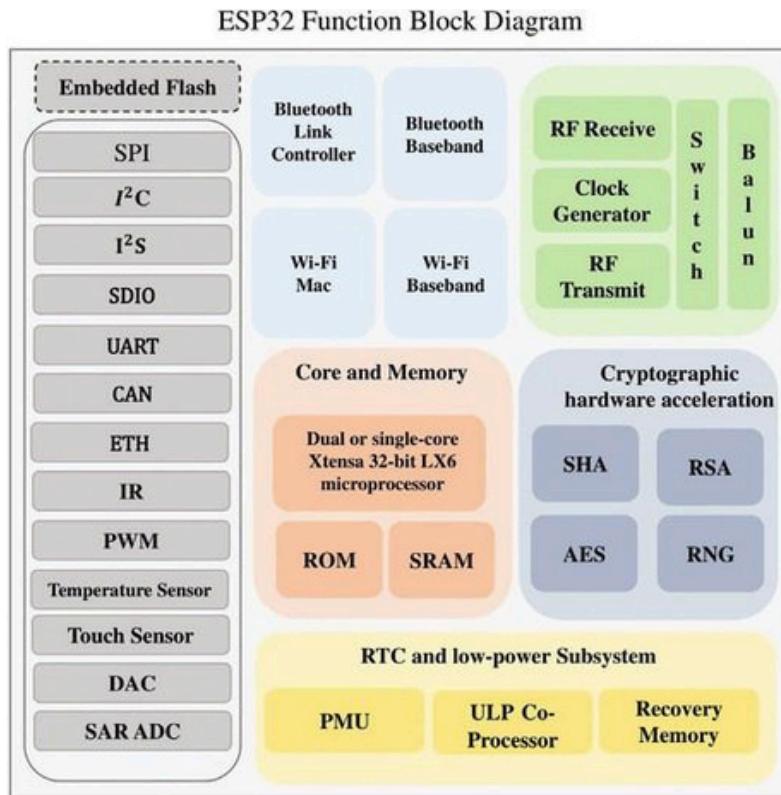


Diagramme fonctionnel interne du microcontrôleur ESP32

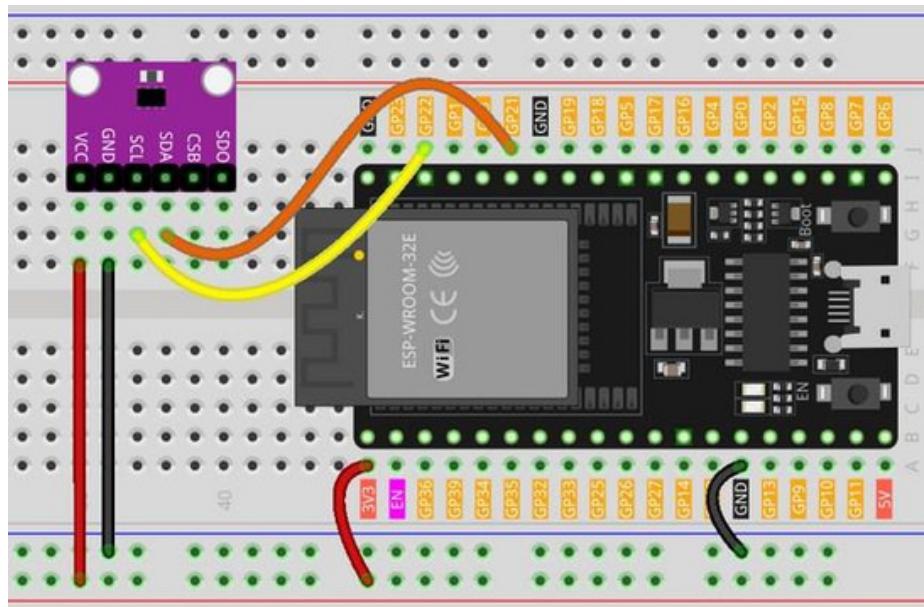


Schéma de câblage du microcontrôleur ESP32 avec capteur environnemental sur breadboard

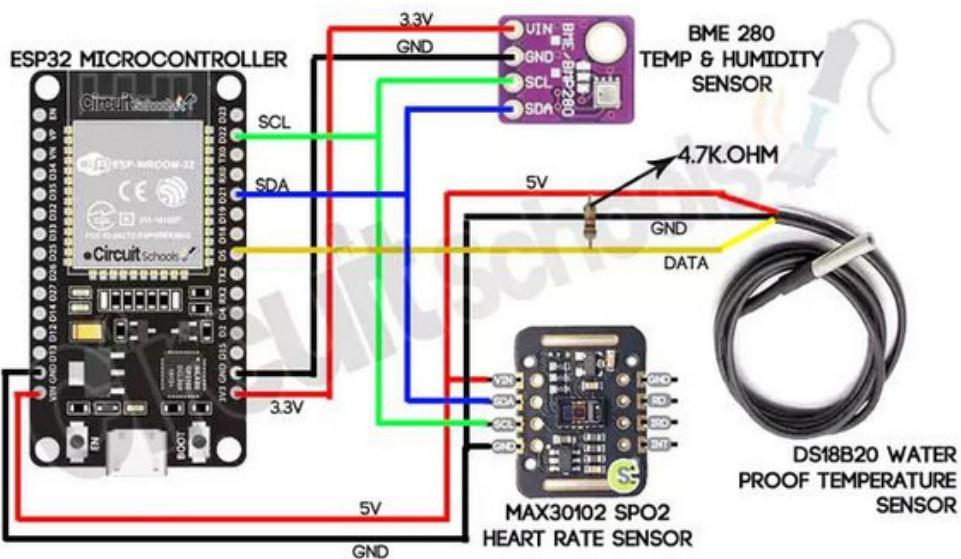


Schéma de connexion des capteurs (BME280, DS18B20, MAX30102) au microcontrôleur ESP32

### 3.Gestion de la consommation énergétique et des alertes

La gestion de la consommation énergétique et des alertes constitue une fonctionnalité essentielle du système Smart Energy Hotel Manager. Elle permet de surveiller en continu l'utilisation de l'énergie, d'identifier les situations anormales et de réagir rapidement afin de réduire les coûts énergétiques et d'améliorer la maintenance des équipements.

#### Collecte et traitement des données énergétiques

Les données énergétiques sont collectées à partir des capteurs installés dans les chambres et les différentes zones de l'hôtel. Ces données incluent notamment la température, l'humidité et les états des équipements énergétiques.

Le traitement des données s'effectue selon les étapes suivantes :

- réception des mesures envoyées par les capteurs via le microcontrôleur ESP32,
- stockage des données dans la base de données MySQL,
- analyse des mesures pour calculer la consommation énergétique par chambre et par période,
- agrégation des données pour produire des statistiques globales.

#### Calcul de la consommation énergétique

Une fois générées, les alertes sont :

- enregistrées dans le système,
- affichées sur le tableau de bord,
- notifiées aux techniciens responsables.

Le technicien peut :

- consulter la liste des alertes,
- analyser les informations associées,
- intervenir sur l'équipement concerné,
- clôturer l'alerte après résolution.

## Calcul de la consommation énergétique

Le système intègre un mécanisme intelligent de détection des anomalies énergétiques. Une alerte est générée automatiquement lorsque :

- la consommation dépasse un seuil prédéfini,
- une variation anormale est détectée,
- un capteur cesse de fonctionner correctement,
- un équipement présente un comportement inhabituel.

Chaque alerte contient des informations détaillées telles que la zone concernée, le type d'anomalie, la date et le niveau de priorité.

## Gestion et suivi des alertes

Le système intègre un mécanisme intelligent de détection des anomalies énergétiques. Une alerte est générée automatiquement lorsque :

- la consommation dépasse un seuil prédéfini,
- une variation anormale est détectée,
- un capteur cesse de fonctionner correctement,
- un équipement présente un comportement inhabituel.

Chaque alerte contient des informations détaillées telles que la zone concernée, le type d'anomalie, la date et le niveau de priorité.

## Chapitre 5 : Tests, résultats et validation

## 1. Stratégie de test

La stratégie de test mise en place vise à garantir la fiabilité, la sécurité et la conformité du système par rapport aux besoins définis dans le cahier des charges. Les tests ont été réalisés tout au long de la phase de développement afin de détecter les anomalies le plus tôt possible. Les principaux objectifs de la stratégie de test sont :

- vérifier le bon fonctionnement des fonctionnalités,
- valider la communication entre le frontend et le backend,
- assurer la sécurité des accès selon les rôles,
- garantir la cohérence des données stockées,
- évaluer la stabilité globale du système.

Les tests ont porté sur :

- les API backend (Laravel),
- l'interface frontend (React),
- la gestion des capteurs et des alertes,
- les scénarios d'utilisation des administrateurs et techniciens.

## 2. Tests fonctionnels

Les tests fonctionnels ont été réalisés afin de vérifier que chaque fonctionnalité du système répond correctement aux exigences définies.

Exemples de tests fonctionnels réalisés :

- Authentification
  - Connexion réussie avec un compte administrateur
  - Connexion réussie avec un compte technicien
  - Refus d'accès en cas d'identifiants incorrects
- Gestion des chambres
  - Ajout, modification et suppression d'une chambre
  - Association correcte des capteurs à une chambre
- Gestion des capteurs
  - Réception et enregistrement des données des capteurs
  - Affichage des données sur le tableau de bord
- Gestion de la consommation énergétique
  - Calcul correct de la consommation par période
  - Affichage des statistiques énergétiques
- Gestion des alertes
  - Génération automatique d'alertes en cas d'anomalie
  - Consultation et clôture des alertes par le technicien

### 3. Présentation de l'application

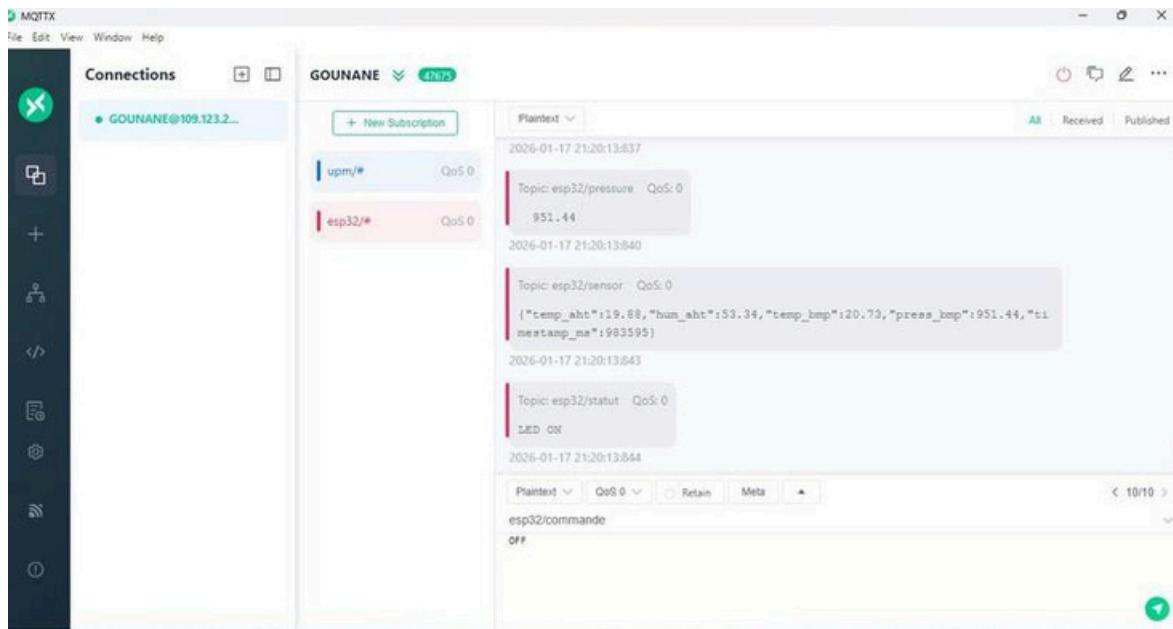


Figure 1 : Interface de communication MQTT (Broker et topics)

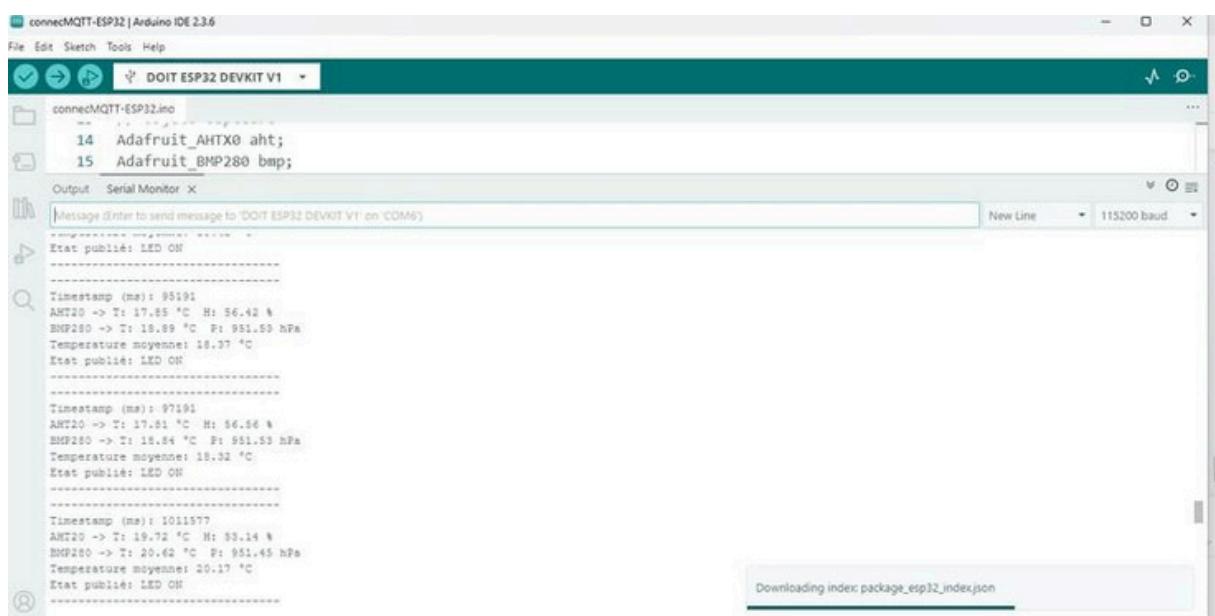


Figure 2: Environnement Arduino – Lecture des données des capteurs

Figure3 : Structure et contenu de la base de données MariaDB

The screenshot shows the Arduino IDE interface with the following details:

- Title Bar:** connectMQTT-ESP32 | Arduino IDE 2.3.6
- Menu Bar:** File Edit Sketch Tools Help
- Toolbar:** Includes icons for Save, Undo, Redo, Print, and others.
- Project Name:** DOIT ESP32 DEVKIT V1
- Code Editor:** Displays the sketch code for connecting to an MQTT broker.

```
connectMQTT-ESP32.ino
23
24 // ----- WIFI -----
Output Serial Monitor X
Message [Enter to send message to DOIT-ESP32-DEVKIT V1 on COM0]
-----[OK] MQTT connecté !
Abonnez się à esp32/commande
-----
```
- Serial Monitor:** Shows the following output:

```
Connexion au broker MQTT 109.123.243.44:1883
Client ID: ESP32-3ADC
[OK] MQTT connecté !
Abonnez się à esp32/commande
-----
```

A red box highlights the "LECTURE CAPTURES" section.

```
LECTURE CAPTURES
-----
```

Below this, sensor data is displayed:

```
ANHT20 -> Temp: 14.00 °C || Hum: 69.35 %
BH220 -> Temp: 19.01 °C || Press: 946.01 hPa
✓ Données publiées sur MQTT
```
- Status Bar:** Shows "Estat LED: Allumée".
- Text Input:** A text input field containing "Prochaine lecture dans 40 secondes |".
- Log Area:** Shows the MQTT disconnection log:

```
[MARS] MQTT déconnecté, reconnexion...
-----
```

A red box highlights the "[MARS]" entry.

```
Connexion au broker MQTT 109.123.243.44:1883
Client ID: ESP32-3ADC
[OK] MQTT connecté !
```

Figure 4 : Affichage des données des capteurs dans l'environnement Arduino

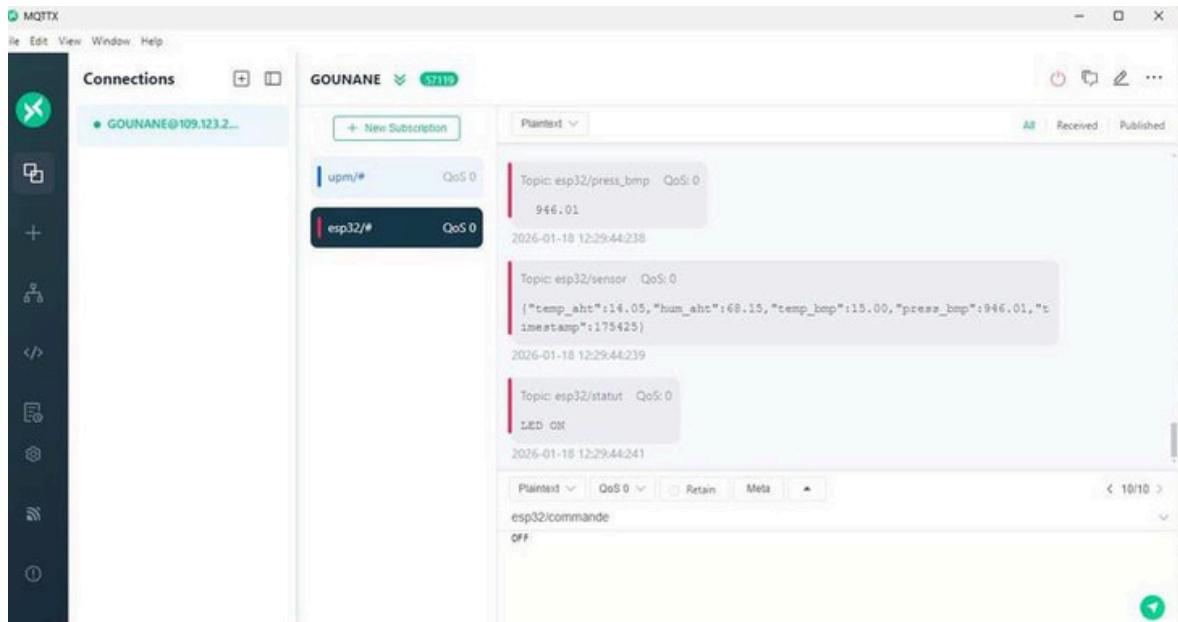


Figure 5 : Interface de communication MQTT pour la réception des données des capteurs

```

File Edit Selection View Go Run ... SmartN
DatabaseSeeder.php Sensor.php ListenToMqtt.php SensorController.php Sensors.txt 3
backend> app > Console > Commands > ListenToMqtt.php > PHP Intelephense > ListenToMqtt @ handle
10
0 references | 0 implementations
11 class ListenToMqtt extends Command
12 {
13     0 references
14     protected $signature = 'mqtt:listen';
15     0 references
16     protected $description = 'Récupère les données du capteur via MQTT toutes les 20 secondes et les enregistre en DB';
17     0 references | 0 overrides
18     public function handle(): never    The routine never returns. It seems it contains an infinite loop
19     (
20         $server = '109.123.243.44'; // IP de ton broker MQTT
21         $port = 1883; // Port MQTT
22         $clientId = 'laravel_listener'; // Identifiant unique
23
24         // Paramètres de connexion
25         $connectionSettings = (new ConnectionSettings)
26             ->setUsername(username: 'shaari') // Remplace par ton user MQTT
27             ->setPassword(password: 'shaari2023') // Remplace par ton mot de passe MQTT
28             ->setKeepAliveInterval(keepAliveInterval: 60)
29             ->setLastWillTopic(lastWillTopic: 'esp32/status')
30             ->setLastWillMessage(lastWillMessage: 'Laravel listener disconnected')
31             ->setLastWillQualityOfService(lastWillQualityOfService: 1);
32
33         $this->info(string: "Démarrage du listener MQTT (toutes les 20 secondes)...");
34
35         while (true) {
36             try {
37                 $mqtt = new MqttClient(host: $server, port: $port, clientId: $clientId);
38                 $mqtt->connect(settings: $connectionSettings, useCleanSession: true);
39                 $this->info(string: "Connecté au broker MQTT !");
40             }
41             catch (Exception $e) {
42                 $this->error(string: "Erreur lors de la connexion au broker MQTT : " . $e->getMessage());
43             }
44         }
45     }
46 }

```

Figure 6 : Implémentation du backend – Traitement des données MQTT (Laravel)

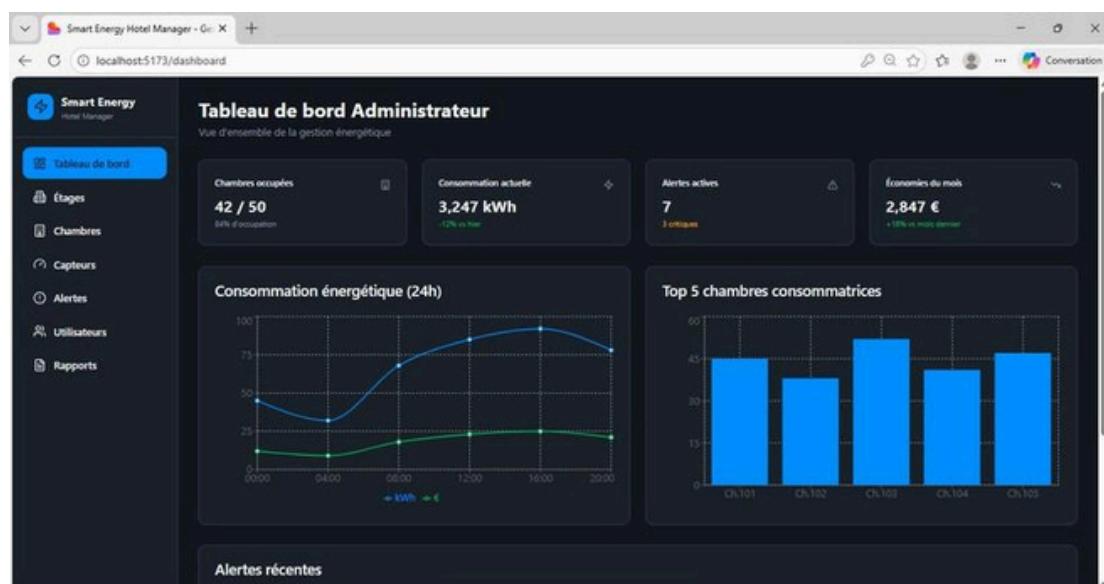


Figure 7: Tableau de bord Administrateur du système Smart Energy Hotel Manager

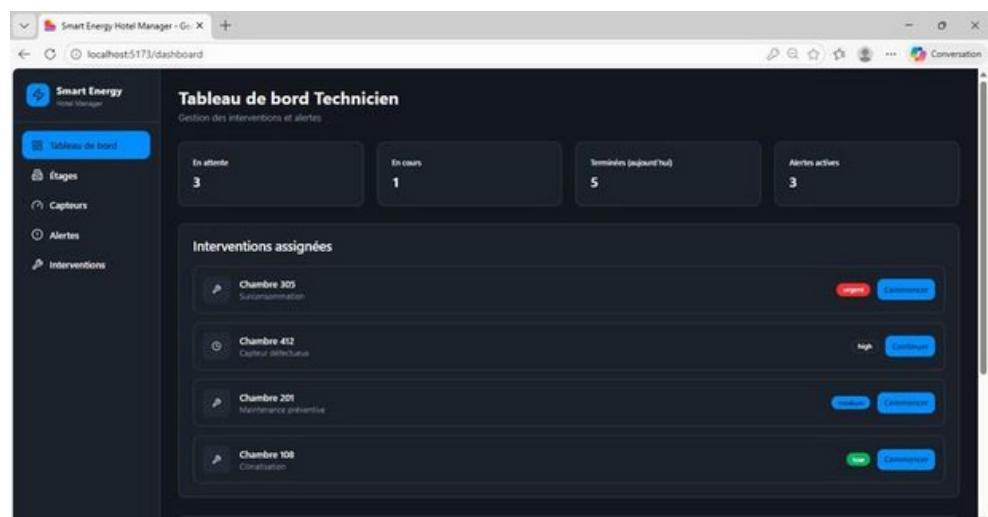


Figure 7: Tableau de bord Administrateur du système Smart Energy Hotel Manager

# CONCLUSION

Ce projet avait pour objectif de concevoir et de réaliser une application intelligente de gestion énergétique pour le secteur hôtelier, intitulée Smart Energy Hotel Manager. Face aux enjeux économiques et environnementaux liés à la consommation d'énergie, la solution proposée vise à offrir une supervision centralisée, une analyse précise des consommations et une gestion proactive des anomalies énergétiques.

Tout au long de ce travail, une démarche structurée a été adoptée, начиная par l'analyse des besoins, la conception architecturale et la modélisation du système, puis la phase de réalisation et d'implémentation. L'architecture retenue, basée sur une API REST développée avec Laravel, un frontend dynamique en React et une base de données MariaDB, a permis de garantir une séparation claire des responsabilités, une bonne maintenabilité et une évolutivité du système. L'intégration des technologies IoT, notamment l'utilisation du microcontrôleur ESP32, des capteurs environnementaux et du protocole MQTT, a renforcé l'aspect intelligent et temps réel de la solution.

Les résultats obtenus à l'issue des tests démontrent que l'application répond efficacement aux exigences fonctionnelles définies. Le système permet de collecter et traiter les données énergétiques, de visualiser les informations via des tableaux de bord clairs, de détecter automatiquement les situations异常和 de générer des alertes facilitant l'intervention des techniciens. Ces fonctionnalités contribuent directement à l'optimisation de la consommation énergétique et à l'amélioration de la maintenance des équipements.

Cependant, malgré les résultats satisfaisants, certaines limites subsistent. Le système dépend fortement de la qualité et de la fiabilité des capteurs utilisés, et les tests de performance à grande échelle n'ont pas encore été explorés en profondeur. De plus, les mécanismes d'analyse restent principalement basés sur des seuils prédéfinis.

En termes de perspectives, plusieurs axes d'amélioration peuvent être envisagés. Il serait possible d'intégrer des algorithmes d'intelligence artificielle afin de réaliser une analyse prédictive de la consommation énergétique et d'anticiper les pannes. L'ajout de nouveaux types de capteurs ( $\text{CO}_2$ , luminosité avancée, qualité de l'air) permettrait d'enrichir les données collectées. Par ailleurs, le déploiement du système sur une infrastructure cloud et l'amélioration des mécanismes de sécurité renforcerait la robustesse et la scalabilité de la solution.

En conclusion, le projet Smart Energy Hotel Manager constitue une base solide pour une gestion énergétique intelligente dans le secteur hôtelier. Il démontre l'intérêt des technologies web et IoT combinées pour répondre aux défis énergétiques actuels et ouvre la voie à de futures évolutions vers des systèmes encore plus autonomes et performants.

