

Rapport du projet de fin d'année

Thème :

**Développement d'un système domotique intelligent :
Conception et implémentation d'une application mobile
de gestion d'une maison connectée avec contrôle vocal**

Réalisé par :

Bannany Brahim

Mahrouch Mohamed

Ait M'hind Ouyhia Youssef

Amrani Zakariae

Spécialité : **Génie Informatique**

Sommaire :

Sommaire :	2
Liste des figures :	4
Remerciements	5
Résumé	6
Abstract	7
Introduction Générale	8
I. Contexte et Motivation :	8
II. Problématique :	8
III. Objectifs du projet :	8
IV. Démarche méthodologique :	9
V. Organisation du rapport :	9
Chapitre 2 : Étude de l'existant	10
I. La domotique : définition et enjeux :	10
II. Exemples de solutions existantes :	10
1. Google Home:	10
2. Amazon Alexa:	10
3. Home Assistant:	11
4. Tuya Smart:	11
III. Limites des solutions existantes :	11
IV. Positionnement de notre solution Smartie :	11
Chapitre 3 : Cahier des charges	12
I. Objectifs généraux :	12
II. Objectifs fonctionnels :	12
III. Objectifs non fonctionnels :	12
IV. Diagramme des cas d'utilisation :	13
V. Contraintes techniques :	14
VI. Diagramme de context :	14
VII. Diagrammes de sequence :	15

Chapitre 4 : Conception de l'architecture du système	16
I. Architecture générale du système :.....	16
II. Schéma global de l'architecture :	16
III. Description des composants :	17
1. Application mobile (Flutter) :	17
2. Interface web (Angular) :	17
3. Backend (FastAPI) :	17
4. Base de données (MongoDB Atlas) :	17
5. Serveur ESP32 :	17
6. Contrôleur Arduino :	17
IV. Communication entre les composants :	18
V. Gestion des commandes vocales :	18
Chapitre 5 : Développement et implémentation	19
I. Application mobile – Flutter :	19
1. Structure du projet :	19
2. Écrans principaux :	20
3. Commande vocale :	23
II. Backend – FastAPI :	23
1. Structure de l'API :	23
2. Traitement de commande vocale :	23
III. Microcontrôleurs – ESP32 & Arduino :	24
1. Fonctionnement de l'ESP32 :	24
2. Communication ESP32 → Arduino :	24
3. Structure des composants :	24
IV. Interface Web – Angular :	24
1. Modules principaux :	24
2. Design et UX :	25
Conclusion	27

Liste des figures :

Figure 1 : Diagramme de cas d'utilisation	13
Figure 2 : Diagramme de context	14
Figure 3 : Diagramme de sequence - Commandes vocales	15
Figure 4 : Diagramme de sequence - Gestion des appareils.....	15
Figure 5 : Architecture de l'application.....	16
Figure 6 : Stricture de l'applciation mobile	19
Figure 7 : Ecran d'inscription.....	20
Figure 8 : Ecran de gestion des appareils	20
Figure 9 : Ecran de gestion des appareils	21
Figure 10 : Ecran d'ajout d'un appareil	21
Figure 11 : Fonctionnalité de reconnaissance vocale	22
Figure 12 : Ecran des paramètres	22
Figure 13 : Stricture du serveur FASTAPI.....	23
Figure 14 : Stricture des composants ESP32 et Arduino	24
Figure 15 : Form d'authentification	25
Figure 16 : Form d'inscription.....	25
Figure 17 : Tableau de bord	25
Figure 18 : Page de gestion des appareils	26
Figure 19 : Page des paramètres.....	26

Remerciements

*Nous tenons tout d'abord à exprimer notre profonde gratitude à **Dieu Tout-Puissant** de nous avoir donné la force, la patience et la persévérance nécessaires pour mener à bien ce projet de fin d'année.*

*Nous adressons nos plus sincères remerciements à l'ensemble du corps professoral de la **Faculté des Sciences et Techniques d'Errachidia**, et plus particulièrement aux enseignants du **Cycle d'Ingénieur en Génie Informatique**, pour la qualité de la formation qu'ils nous ont dispensée tout au long de notre parcours académique.*

*Nous remercions chaleureusement **notre encadrant** pour sa disponibilité, son accompagnement, ses conseils précieux ainsi que pour la confiance qu'il nous a accordée durant les différentes phases de réalisation de ce projet. Son soutien nous a été d'une grande aide, tant sur le plan technique que méthodologique.*

*Nous tenons également à remercier **nos camarades de promotion** pour les échanges enrichissants, l'esprit d'entraide et la bonne ambiance qui ont régné tout au long de cette année.*

*Enfin, nous exprimons notre reconnaissance à **nos familles** pour leur soutien moral, leurs encouragements constants et leur patience tout au long de notre parcours universitaire.*

Résumé

Dans un monde en constante évolution technologique, la domotique s'impose aujourd'hui comme une solution innovante et intelligente pour améliorer le confort, la sécurité et la gestion énergétique au sein des habitations. Dans ce cadre, nous avons développé une application mobile nommée **Smartie**, qui permet de contrôler à distance les différents équipements d'une maison intelligente, tout en intégrant un système de commandes vocales pour une interaction plus intuitive.

Ce projet s'inscrit dans le cadre de notre Projet de Fin d'Année au sein du cycle d'ingénieur en Génie Informatique à la Faculté des Sciences et Techniques d'Errachidia. Il combine plusieurs technologies modernes, à savoir **Flutter** pour le développement mobile multiplateforme, **FastAPI** pour le backend, **MongoDB Atlas** pour la base de données, et un **microcontrôleur ESP32** agissant comme serveur HTTP, connecté à un **Arduino** chargé de l'exécution physique des commandes.

Nous avons également réalisé une interface web à l'aide du framework **Angular**, afin de permettre une gestion centralisée depuis différents terminaux.

L'objectif principal de Smartie est d'offrir une solution efficace, accessible et évolutive pour le pilotage des équipements domestiques, tout en offrant à l'utilisateur une expérience fluide et interactive. Le système mis en place démontre la faisabilité et l'efficacité de l'intégration des objets connectés dans un environnement domestique contrôlé par des interfaces intuitives et des commandes vocales.

Mots-clés :

Domotique, Maison intelligente, Flutter, FastAPI, MongoDB Atlas, ESP32, Arduino, Commandes vocales, Angular.

Abstract

In an era marked by rapid technological advancements, smart home systems have emerged as innovative solutions to enhance comfort, security, and energy management within residential environments. As part of our final year engineering project, we developed a mobile application named **Smartie**, designed to remotely control various smart home devices, with the added capability of **voice command integration** for a more natural user experience.

This project was carried out within the **Computer Engineering Program** at the **Faculty of Science and Technology of Errachidia**. It integrates a range of modern technologies, including **Flutter** for cross-platform mobile development, **FastAPI** for backend services, **MongoDB Atlas** for cloud-based data storage, and an **ESP32 microcontroller** acting as an HTTP server that communicates with an **Arduino board** responsible for executing physical device actions.

Additionally, we developed a **web interface** using **Angular** to allow centralized control from desktop environments.

The main objective of Smartie is to provide a flexible, efficient, and scalable solution for managing smart home equipment, offering users a seamless and interactive experience. The implemented system demonstrates the feasibility and effectiveness of integrating connected devices into a smart environment, controlled through intuitive interfaces and voice commands.

Key words:

Smart home, Home automation, Flutter, FastAPI, MongoDB Atlas, ESP32, Arduino, Voice control, Angular.

Introduction Générale

I. Contexte et Motivation :

L'évolution rapide des technologies de l'information et de la communication a profondément transformé notre manière d'interagir avec notre environnement. Parmi les domaines les plus impactés figure la domotique, qui vise à automatiser, contrôler et optimiser le fonctionnement des équipements domestiques dans le but de garantir confort, sécurité, et efficacité énergétique.

Dans un contexte mondial de transition numérique et de recherche de solutions intelligentes, les systèmes de maison connectée connaissent un essor important. Ces systèmes permettent aux utilisateurs de contrôler leurs appareils électroménagers, systèmes d'éclairage, capteurs de sécurité, et autres équipements via des interfaces numériques accessibles depuis des smartphones ou navigateurs web.

C'est dans cette optique que s'inscrit notre projet intitulé **Smartie**. Il a été réalisé dans le cadre de notre Projet de Fin d'Année au sein du **Cycle d'Ingénieur en Génie Informatique** à la **Faculté des Sciences et Techniques d'Errachidia**.

II. Problématique :

Malgré la diversité des solutions de domotique disponibles sur le marché, plusieurs limitations subsistent : coût élevé, manque de flexibilité, dépendance à des services cloud externes, ou encore absence de personnalisation. De plus, l'intégration des **commandes vocales** dans des solutions locales et accessibles demeure encore limitée.

Face à ces constats, la problématique centrale de notre projet peut se formuler ainsi :

Comment concevoir et mettre en œuvre une solution de maison intelligente, à la fois mobile, web et vocale, qui soit accessible, évolutive, et capable de piloter efficacement des équipements physiques en temps réel ?

III. Objectifs du projet :

Notre objectif principal est de développer une **application mobile intelligente**, accompagnée d'une interface web et d'un backend robuste, permettant de :

- ✚ Contrôler à distance les équipements d'une maison via des commandes simples (allumer, éteindre, etc.).
- ✚ Intégrer la **commande vocale** pour une interaction naturelle avec le système.

- ✚ Assurer une communication fluide entre l'application, le serveur, et les microcontrôleurs (ESP32, Arduino).
- ✚ Utiliser une architecture **modulaire** et **scalable** reposant sur des technologies modernes (Flutter, FastAPI, MongoDB Atlas, Angular).

IV. Démarche méthodologique :

Pour atteindre ces objectifs, nous avons adopté une approche **modulaire et itérative**, fondée sur les étapes suivantes :

- ✚ Étude de l'existant et analyse des besoins.
- ✚ Conception de l'architecture logicielle et matérielle.
- ✚ Implémentation des différentes couches du système.
- ✚ Intégration et tests des modules.
- ✚ Validation fonctionnelle et technique de l'application.

V. Organisation du rapport :

Ce rapport est structuré comme suit :

- ✚ **Chapitre 2** : Présente l'étude de l'existant en matière de domotique et de solutions intelligentes.
- ✚ **Chapitre 3** : Décrit le cahier des charges fonctionnel et technique du projet.
- ✚ **Chapitre 4** : Détaille l'architecture du système et la conception des différents modules.
- ✚ **Chapitre 5** : Explique le processus de développement et d'implémentation.
- ✚ **Chapitre 6** : Expose les tests réalisés, les résultats obtenus, et les problèmes rencontrés.
- ✚ **Chapitre 7** : Propose une conclusion générale et des perspectives d'amélioration.

Chapitre 2 : Étude de l'existant

Avant d'entamer la conception de notre propre solution domotique, il était essentiel de réaliser une étude approfondie de l'existant. Cela nous a permis d'analyser les systèmes et technologies actuels utilisés dans le domaine des **maisons intelligentes**, de comprendre les besoins des utilisateurs, d'identifier les principales limitations des solutions existantes, et enfin de positionner notre projet **Smartie** dans ce contexte technologique.

I. La domotique : définition et enjeux :

La domotique est l'ensemble des techniques permettant de **centraliser le contrôle** des différents systèmes de la maison (éclairage, chauffage, sécurité, électroménager, etc.). Elle repose sur l'intégration de technologies informatiques, électroniques et de télécommunications dans le cadre domestique afin de :

- ✚ Améliorer le confort des occupants.
- ✚ Renforcer la sécurité de l'habitation.
- ✚ Optimiser la consommation énergétique.
- ✚ Offrir des services personnalisés à travers des interfaces intuitives.

Ces dernières années, l'essor de l'Internet des Objets (IoT) a largement contribué à démocratiser la domotique, rendant les dispositifs plus accessibles et connectés.

II. Exemples de solutions existantes :

1. Google Home:



Google Home permet de contrôler des appareils intelligents via l'assis tant vocal Google Assistant. Il offre une grande compatibilité avec les objets connectés de diverses marques, mais repose fortement sur le **cloud** et nécessite une connexion internet permanente.

2. Amazon Alexa:



Semblable à Google Home, Alexa offre une commande vocale performante et une vaste compatibilité avec les appareils. Toutefois, comme Google Home, elle pose des questions liées à **la confidentialité des données** et **la dépendance aux serveurs cloud**.

3. Home Assistant:



Home Assistant est une plateforme open-source qui permet l'automatisation et le contrôle local des appareils. Très puissante et flexible, elle demande cependant une certaine expertise technique pour l'installation et la configuration.

4. Tuya Smart:



Ces applications permettent de gérer des objets connectés via des plateformes cloud. Bien que simples à utiliser, elles nécessitent souvent des serveurs distants hébergés en Chine, ce qui peut poser des questions de **latence** et de **sécurité**.

III. Limites des solutions existantes :

Bien que performantes, les solutions existantes présentent un certain nombre de **limitations** :

- + **Dépendance au cloud** : perte de fonctionnalité en cas de coupure internet ;
- + **Manque de personnalisation** pour les utilisateurs avancés ;
- + **Coût élevé** pour certaines plateformes commerciales ;
- + **Complexité technique** pour les solutions open-source auto-hébergées ;
- + **Problèmes de confidentialité** : collecte et stockage de données personnelles vocales et domotiques.

IV. Positionnement de notre solution Smartie :

Face à ces constats, notre projet **Smartie** propose une solution :

- + **Accessible** : basée sur des technologies modernes mais ouvertes (Flutter, FastAPI, MongoDB, etc.) ;
- + **Hybride** : permettant un **contrôle local** des appareils via un serveur ESP32 et une communication directe avec l'Arduino ;
- + **Multiplateforme** : disponible sur mobile et sur le web ;
- + **Interactive** : avec intégration de **commandes vocales intelligentes** ;
- + **Modulaire et évolutive** : facile à adapter pour différents types de capteurs, appareils, ou extensions futures.

L'étude de l'existant nous a permis d'identifier les besoins non couverts par les solutions actuelles. Notre projet Smartie vise ainsi à combler certaines de ces lacunes en offrant une solution flexible, personnalisable et centrée sur l'utilisateur, tout en gardant une architecture techniquement solide et maîtrisable par les développeurs.

Chapitre 3 : Cahier des charges

Le cahier des charges constitue une étape cruciale dans tout projet de développement logiciel. Il définit les objectifs, les besoins fonctionnels et non fonctionnels, ainsi que les contraintes techniques à respecter. Dans le cadre de notre projet **Smartie**, ce document nous a servi de référence pour guider les choix technologiques, la conception et la réalisation du système.

I. Objectifs généraux :

Le projet Smartie vise à développer un système complet de gestion d'une maison intelligente, basé sur une architecture mobile, web, backend et embarquée, intégrant également la **commande vocale**. L'objectif principal est de proposer une solution **simple**, **intelligente**, **accessible** et **modulaire**, permettant aux utilisateurs de :

- ✚ Contrôler à distance les équipements de leur maison.
- ✚ Utiliser leur voix pour déclencher des actions.
- ✚ Visualiser l'état des équipements via des interfaces conviviales (mobile et web).

II. Objectifs fonctionnels :

- **Contrôle des équipements** : Permettre d'allumer/éteindre des appareils connectés (lumières, ventilateurs, etc.).
- **Application mobile** : Interface Flutter pour le contrôle des équipements et affichage de leur état.
- **Interface web** : Tableau de bord Angular pour un accès depuis un navigateur.
- **Commande vocale** : Reconnaître des commandes vocales en local pour piloter les appareils.
- **Synchronisation** : Communication en temps réel entre l'application, le backend et les microcontrôleurs.
- **Gestion des appareils** : Ajout, modification et suppression des appareils via l'interface.

III. Objectifs non fonctionnels :

- **Performance** : Réponses rapides aux commandes utilisateur.
- **Sécurité** : Authentification sécurisée, protection des données utilisateurs.
- **Scalabilité** : Possibilité d'ajouter facilement de nouveaux équipements ou fonctionnalités.
- **Accessibilité** : Application mobile intuitive, interface web ergonomique.

- **Portabilité** : Compatible Android/iOS et navigateurs modernes.
- **Résilience** : Fonctionnement partiel possible en cas de coupure Internet locale.

IV. Diagramme des cas d'utilisation :

Voici une description textuelle des principaux cas d'utilisation :

+ Utilisateur :

- Se connecter / s'inscrire
- Accéder à la liste des appareils
- Allumer / éteindre un appareil
- Contrôler un appareil par la voix
- Ajouter / modifier / supprimer un appareil

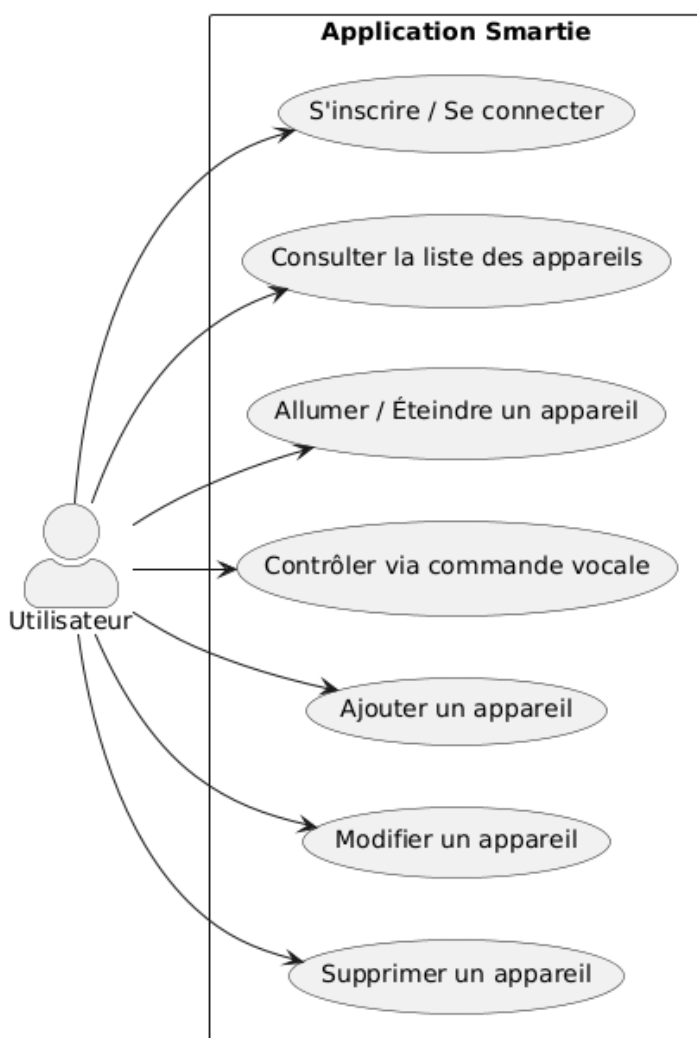


Figure 1 : Diagramme de cas d'utilisation

V. Contraintes techniques :

✚ Matérielles :

- ESP32 (Wi-Fi) connecté à un Arduino (action physique)
- Capteurs et actionneurs (LEDs, relais, etc.)

✚ Technologiques :

- **Frontend mobile** : Flutter
- **Frontend web** : Angular
- **Backend** : FastAPI (Python)
- **Base de données** : MongoDB Atlas (cloud)
- **Communication** : HTTP entre le backend et l'ESP32

VI. Diagramme de contexte :

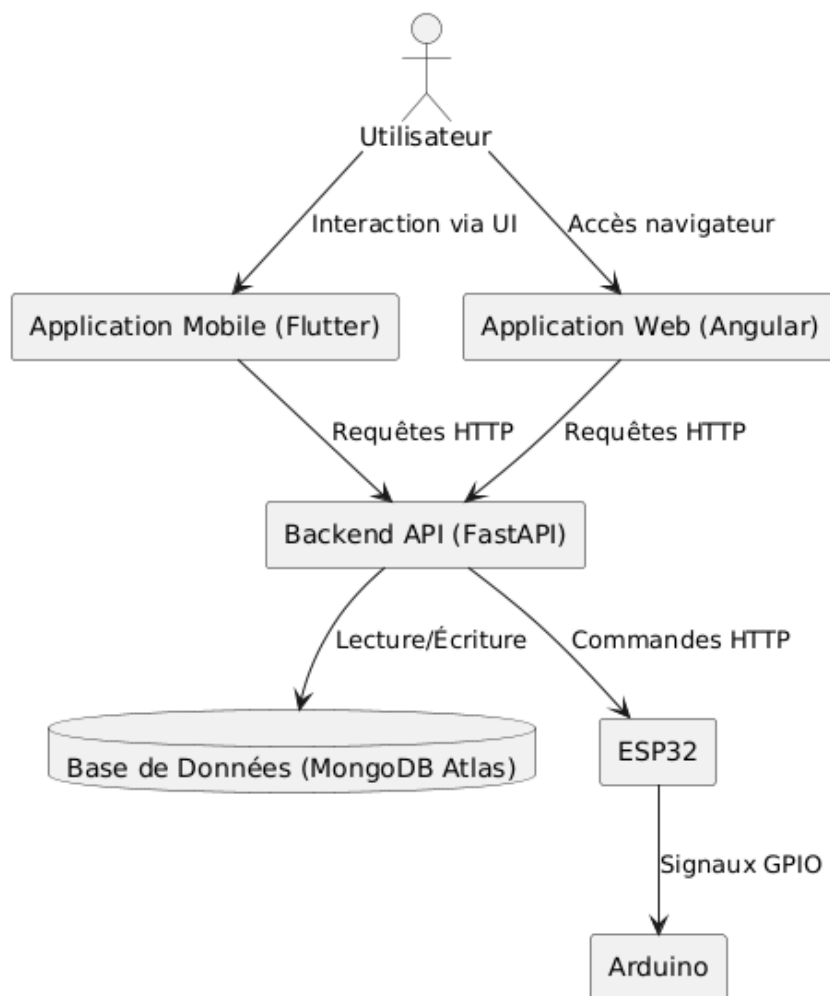


Figure 2 : Diagramme de contexte

VII. Diagrammes de sequence :

Commandes vocales :

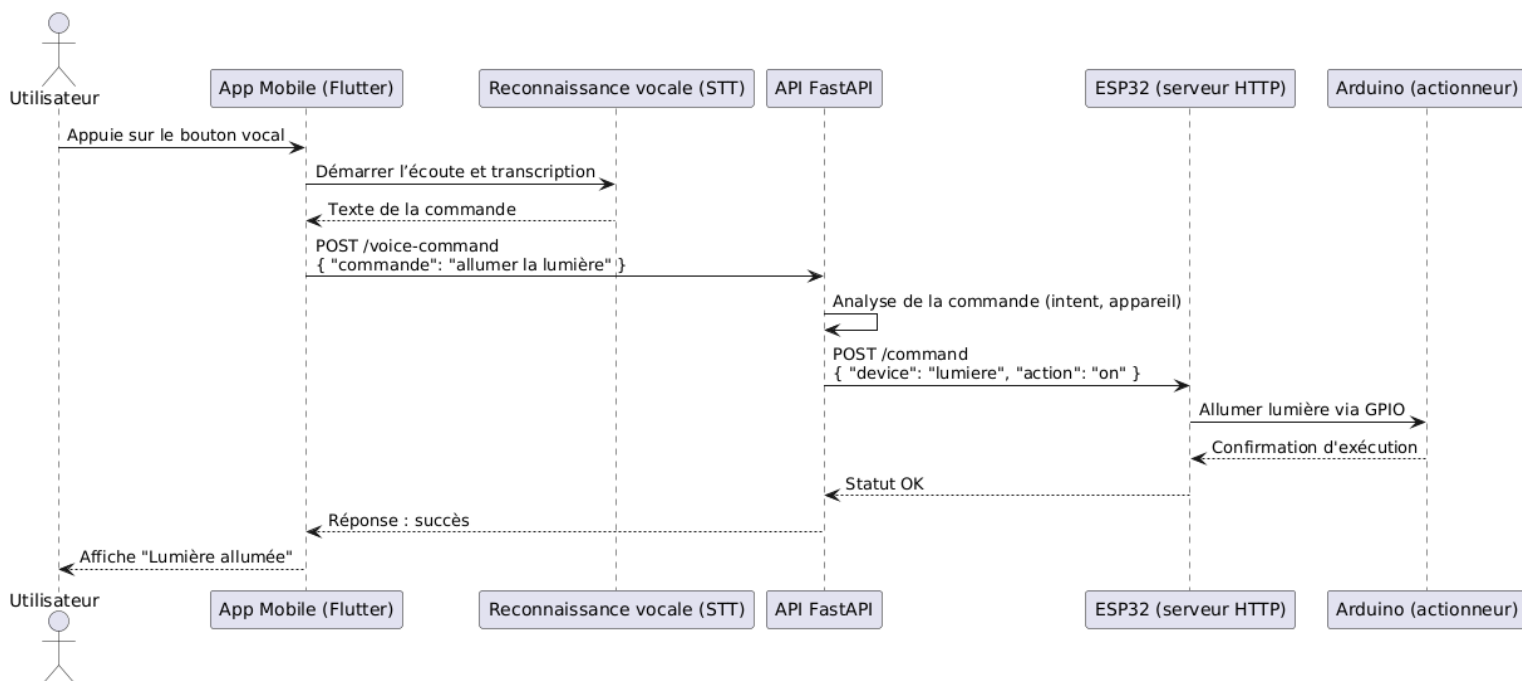


Figure 3 : Diagramme de sequence - Commandes vocales

Gestion des appareils :

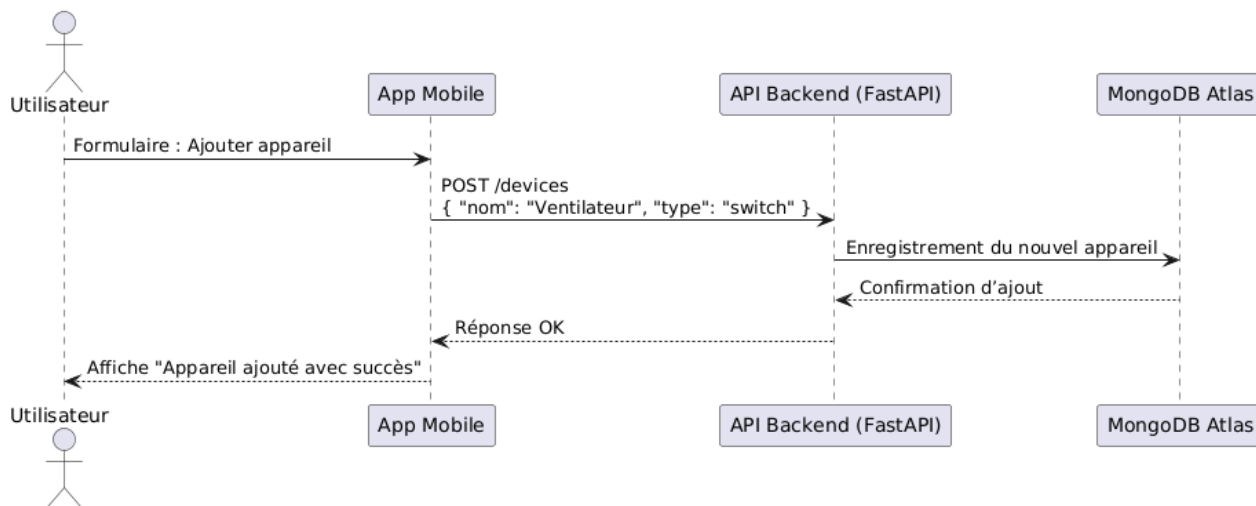


Figure 4 : Diagramme de sequence - Gestion des appareils

Ce cahier des charges a servi de base à toutes les décisions prises dans le cadre de ce projet. Il a permis de clarifier les besoins fonctionnels et techniques du système Smartie, tout en cadrant les choix d'implémentation. Il sera continuellement réévalué au fur et à mesure de l'évolution du projet.

Chapitre 4 : Conception de l'architecture du système

La conception de l'architecture est une étape essentielle qui précède toute phase de développement. Elle permet de définir les différents **composants logiciels et matériels** du système, ainsi que leurs interactions. Dans le cadre du projet **Smartie**, nous avons adopté une **architecture modulaire**, orientée services, qui repose sur l'intégration de plusieurs technologies complémentaires, tant du côté mobile que web, tout en incluant une couche matérielle de contrôle via microcontrôleurs.

I. Architecture générale du système :

Le système Smartie est composé de plusieurs couches distinctes :

- **Frontend mobile** développé avec **Flutter**.
- **Frontend web** développé avec **Angular**.
- **Backend API** développé avec **FastAPI** (Python).
- **Base de données NoSQL** MongoDB Atlas (hébergée dans le cloud).
- **ESP32**, servant de serveur HTTP local pour transmettre les commandes.
- **Arduino**, utilisé comme contrôleur physique (exécution d'actions : allumer/éteindre un appareil).

II. Schéma global de l'architecture :

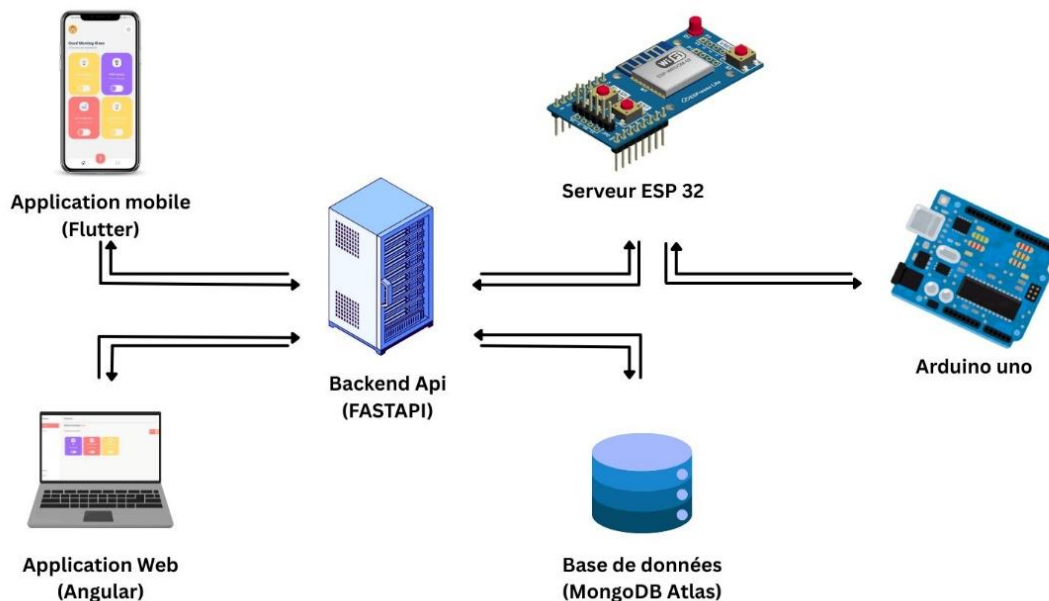


Figure 5 : Architecture de l'application

III. Description des composants :

1. Application mobile (Flutter) :

- ✚ Interface utilisateur moderne et responsive.
- ✚ Intégration de la reconnaissance vocale.
- ✚ Communication avec l'API via http.
- ✚ Visualisation en temps réel des équipements et de leur état.

2. Interface web (Angular) :

- ✚ Accessible depuis un navigateur.
- ✚ Permet de gérer les appareils et visualiser leur état.
- ✚ Connexion sécurisée au backend.

3. Backend (FastAPI) :

- ✚ API RESTful développée en Python avec FastAPI.
- ✚ Gère les utilisateurs, les appareils et les commandes vocales.
- ✚ Se connecte à MongoDB Atlas.
- ✚ Traduit les requêtes utilisateurs en commandes HTTP pour l'ESP32.

4. Base de données (MongoDB Atlas) :

- ✚ Base NoSQL orientée documents.
- ✚ Stocker les appareils et leurs états.
- ✚ Hautement disponible, hébergée sur le cloud.

5. Serveur ESP32 :

- ✚ Reçoit les requêtes HTTP depuis le backend.
- ✚ Joue le rôle de relais entre l'API et l'Arduino.
- ✚ Connecté au réseau Wi-Fi.

6. Contrôleur Arduino :

- ✚ Reçoit des signaux depuis l'ESP32 via GPIO.
- ✚ Exécute les actions physiques (allumage/extinction) sur les appareils.

IV. Communication entre les composants :

- **Flutter/Angular ↔ FastAPI** : Requêtes HTTP sécurisées (CRUD des appareils, commandes vocales).
- **FastAPI ↔ MongoDB** : Connexion MongoDB avec motor pour stocker et récupérer les données.
- **FastAPI ↔ ESP32** : Envoi de requêtes HTTP comme POST /command { "device": "lamp", "action": "on" }.
- **ESP32 ↔ Arduino** : Transmission via broches numériques (GPIO) ou série selon le montage.

V. Gestion des commandes vocales :

Le traitement vocal suit les étapes suivantes :


1. L'utilisateur clique sur le micro dans Flutter ;
2. La voix est transcrite en texte localement (ex. Whisper, Vosk ou STT de Google) ;
3. Le texte est envoyé au backend ;
4. Le backend analyse l'intention et l'appareil ciblé ;
5. Le backend envoie une commande à l'ESP32 ;
6. L'ESP32 active l'élément physique via l'Arduino.

L'architecture de **Smartie** repose sur une organisation bien définie de composants logiciels et matériels, facilitant la modularité, l'évolution et la maintenance du système. Chaque couche a été pensée pour garantir performance, sécurité et simplicité d'utilisation, tout en offrant une expérience utilisateur fluide et intelligente.

Chapitre 5 : Développement et implémentation

Après avoir défini l'architecture du système Smartie, la phase de développement nous a permis de concrétiser notre conception à travers des technologies modernes adaptées à chaque couche : **Flutter** pour l'application mobile, **FastAPI** pour le backend, **MongoDB Atlas** pour la base de données, **ESP32 et Arduino** pour le contrôle physique, et **Angular** pour l'interface web.

I. Application mobile – Flutter :



Flutter est un **framework open-source** développé par **Google** qui permet de créer des applications **mobiles, web et desktop** à partir d'un **seul code source**. Basé sur le langage **Dart**, Flutter se distingue par ses performances proches du natif et son système de widgets personnalisables qui facilite le développement d'interfaces utilisateur modernes, réactives et esthétiques. Grâce à sa fonctionnalité **hot reload**, il permet aux développeurs de visualiser instantanément les changements apportés au code, ce qui accélère le processus de développement.

1. Structure du projet :

- ✚ Architecture en widgets modulaires (Pages, Models, Services) ;
- ✚ Utilisation de Provider pour la gestion d'état ;
- ✚ Thèmes sombres et responsives.

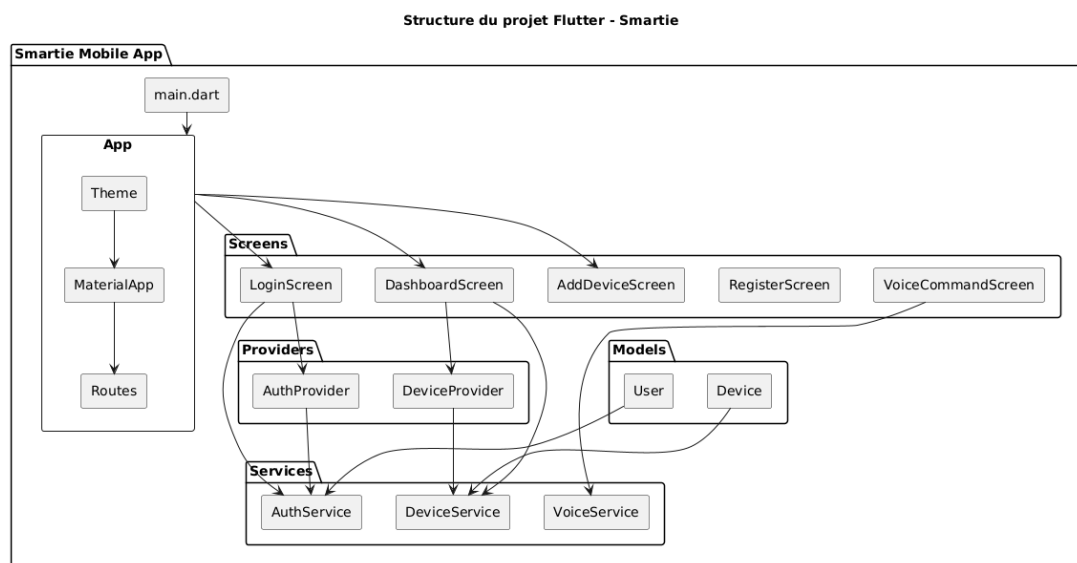


Figure 6 : Stricture de l'applciation mobile

2. Écrans principaux :

a. Écran de connexion / inscription :

L'écran de connexion / inscription permet aux utilisateurs de **créer un compte** ou de **se connecter** à l'application afin d'accéder de manière sécurisée à leurs données personnelles.

Figure 7 : Ecran d'inscription

b. Tableau de bord:

Le **tableau de bord** affiche la **liste des appareils connectés** avec des **interrupteurs (switchs) on/off**.

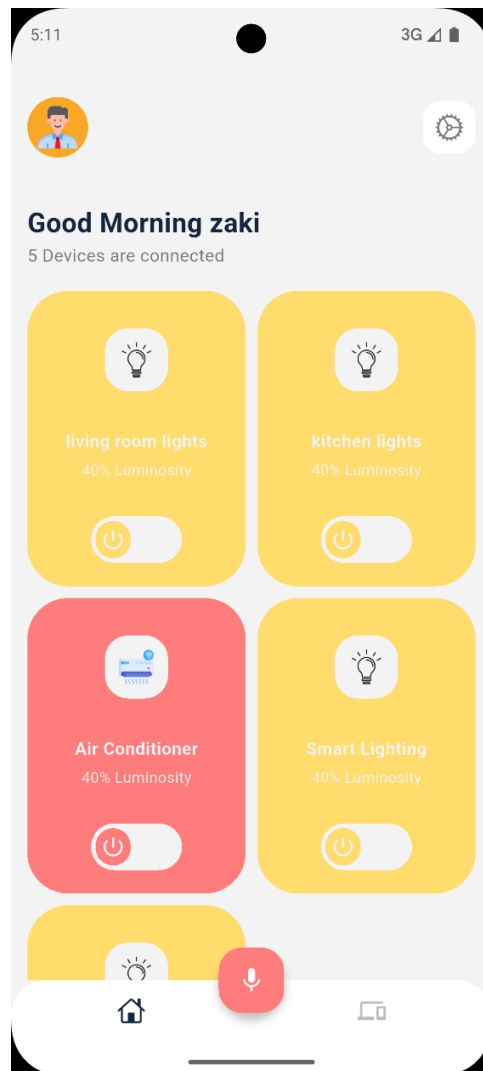


Figure 8 : Ecran de gestion des appareils

c. Écran de gestion des appareils :

L'écran de gestion des appareils permet à l'utilisateur de configurer un nouvel appareil ou de supprimer un ancien appareil.

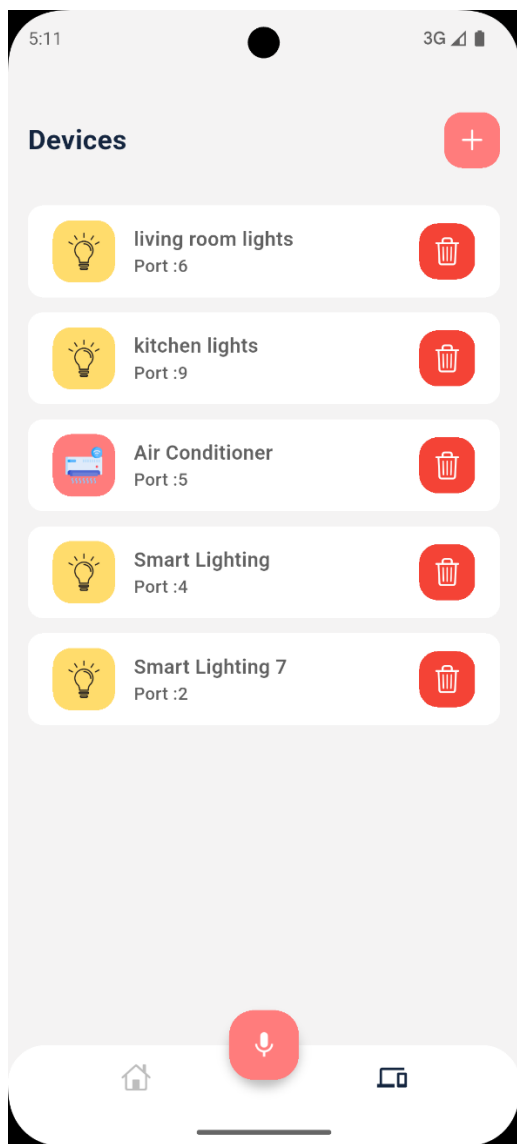


Figure 9 : Ecran de gestion des appareils

d. Ecran d'ajout d'un appareil :

L'écran d'ajout / modification d'un appareil permet à l'utilisateur de configurer un nouvel appareil.

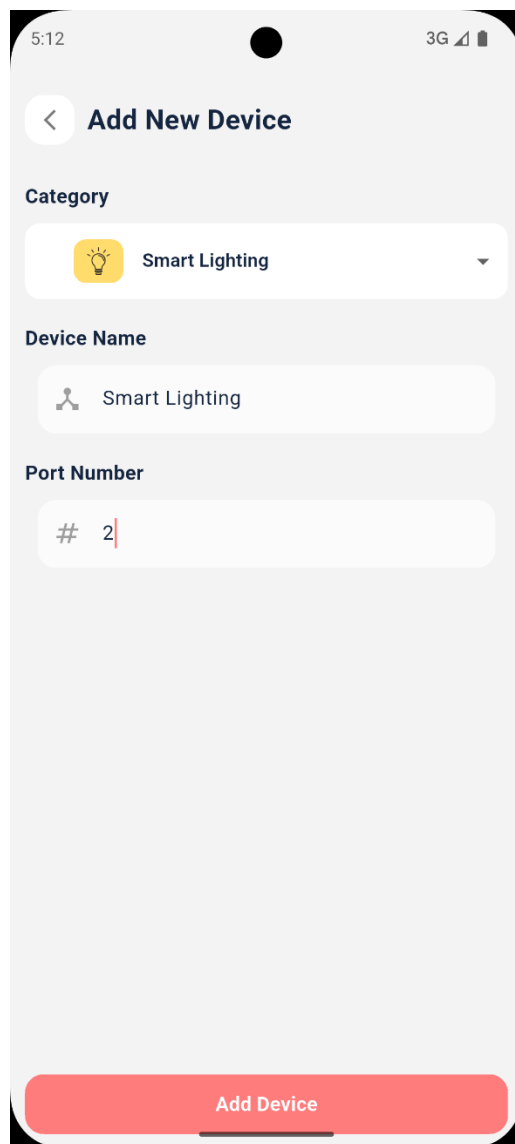


Figure 10 : Ecran d'ajout d'un appareil

e. Reconnaissance vocale :

La fonctionnalité de **reconnaissance vocale**, accessible via un **bouton micro**, permet à l'utilisateur de **contrôler les appareils par commande vocale**, offrant une interaction plus rapide et intuitive avec l'application.



Figure 11 : Fonctionnalité de reconnaissance vocale

f. Ecran des paramètres :

L'**écran des paramètres** permet à l'utilisateur de gérer de son compte.

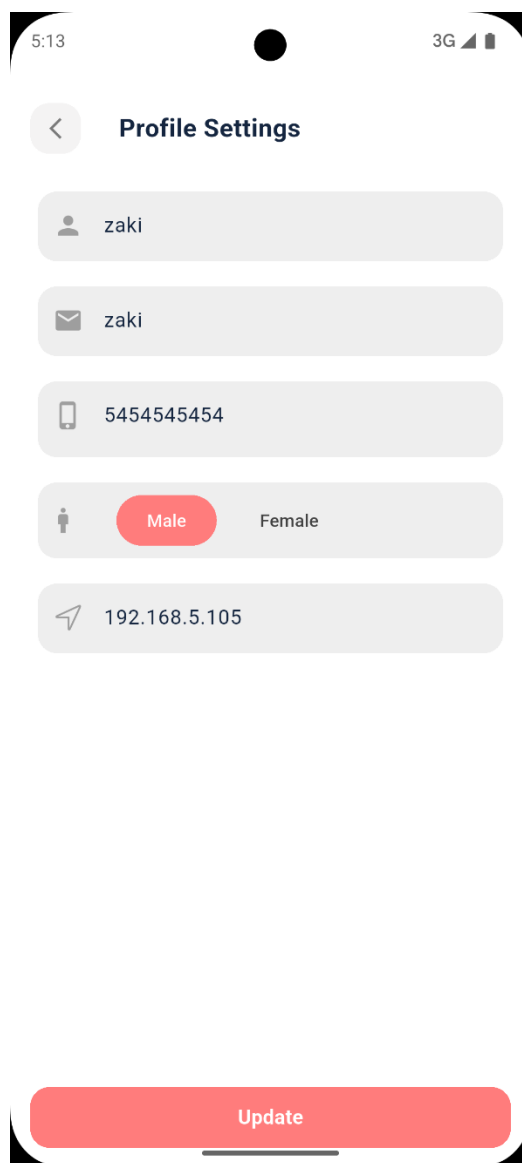


Figure 12 : Ecran des paramètres

3. Commande vocale :

- ✚ Utilisation de la reconnaissance vocale hors-ligne pour transformer la voix en texte.
- ✚ Envoi du texte à une route dédiée du backend (/predict_intent).
- ✚ Le backend interprète et déclenche la commande appropriée.

II. Backend – FastAPI :

1. Structure de l'API :

- ✚ **Routes** : /auth, /devices, /command, /voice-command.
- ✚ **Fichiers organisés** : main.py, models.py, routes/, services/, schemas/.

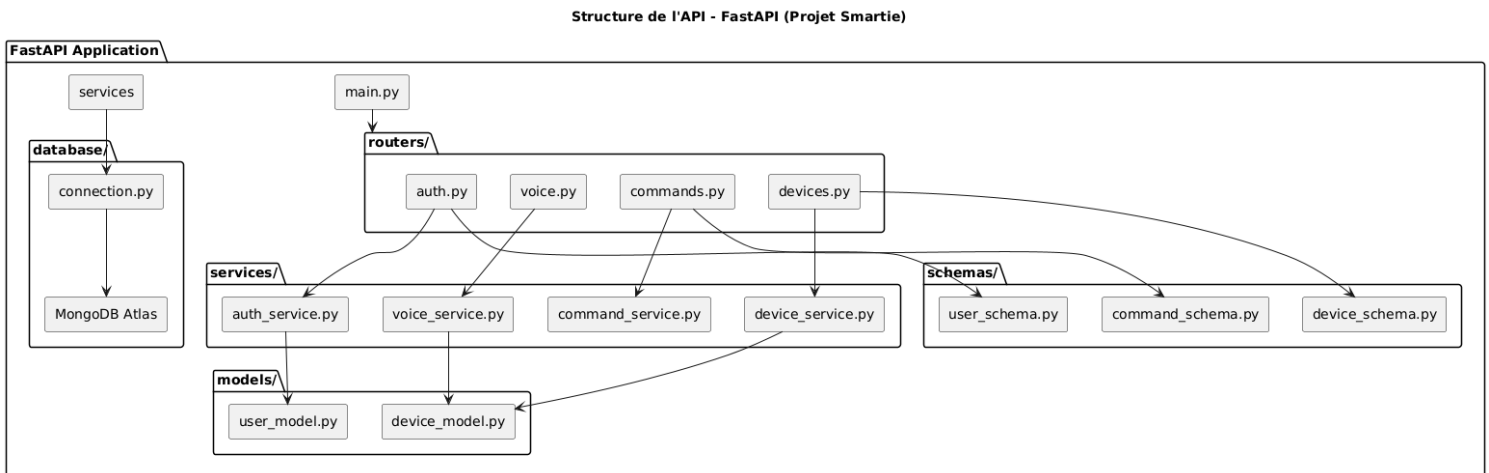


Figure 13 : Stricture du serveur FASTAPI

2. Traitement de commande vocale :

La capture de **l'audio**, qui est ensuite converti en **texte** grâce au modèle **Whisper**. Ce texte est ensuite analysé par un modèle **BERT**, préalablement entraîné sur un **dataset** annoté, pour identifier automatiquement les éléments clés de la commande tels que **l'intention** (ex. allumer, éteindre), **l'appareil** (ex. lumière, climatiseur) et le **lieu** (ex. salon, chambre).

Cette chaîne de traitement permet d'extraire les informations nécessaires afin d'exécuter les actions demandées par l'utilisateur.

III. Microcontrôleurs – ESP32 & Arduino :

1. Fonctionnement de l'ESP32 :

ESP32 agit comme serveur HTTP en Wi-Fi, recevant les commandes depuis FastAPI :

```
server.on("/command", HTTP_POST, [])(AsyncWebServerRequest *request){
    String device = request->arg("device");
    String action = request->arg("action");
    // GPIO logic here
});
```

2. Communication ESP32 → Arduino :

- ✚ **Via GPIO** : ESP32 active une broche HIGH/LOW connectée au relais ou à l'Arduino.
- ✚ **Arduino** exécute l'action (ex. allumer une LED, activer un moteur).

3. Stricture des composants :

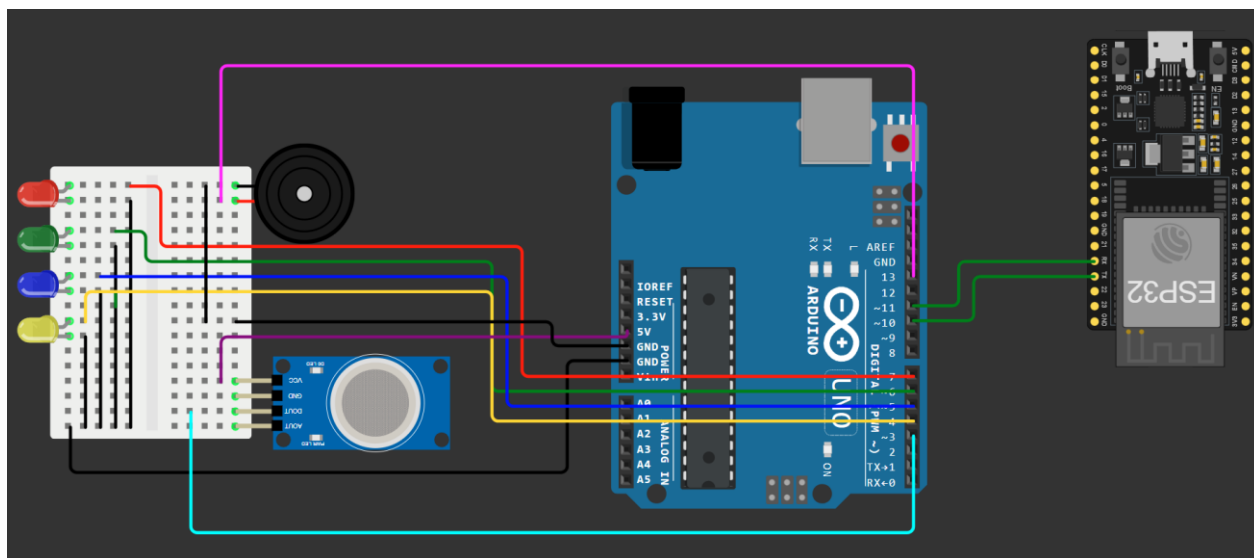


Figure 14 : Stricture des composants ESP32 et Arduino

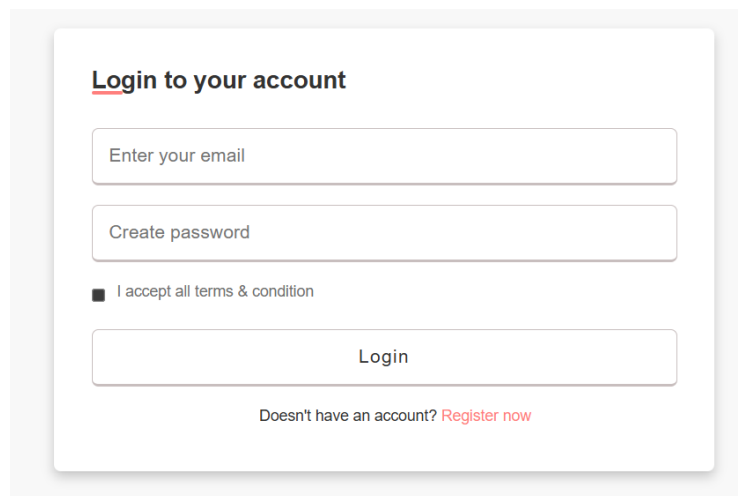
IV. Interface Web – Angular :

1. Modules principaux :

- ✚ Authentification
- ✚ Affichage de la liste des appareils ;
- ✚ Boutons on/off avec mise à jour en temps réel ;
- ✚ Formulaire d'ajout / édition d'appareil.

2. Design et UX :

a. Page de connexion / inscription :



Login to your account

Enter your email

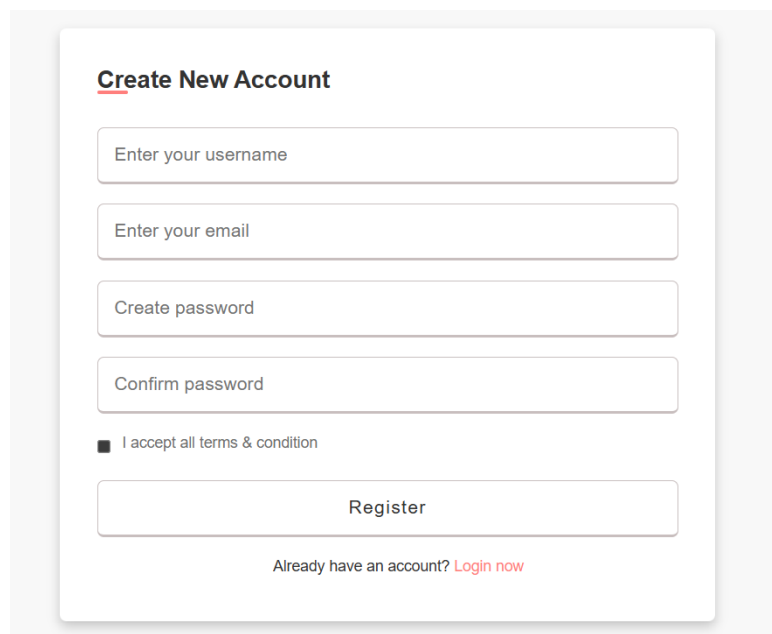
Create password

☐ I accept all terms & condition

Login

Doesn't have an account? [Register now](#)

Figure 15 : Form d'authentification



Create New Account

Enter your username

Enter your email

Create password

Confirm password

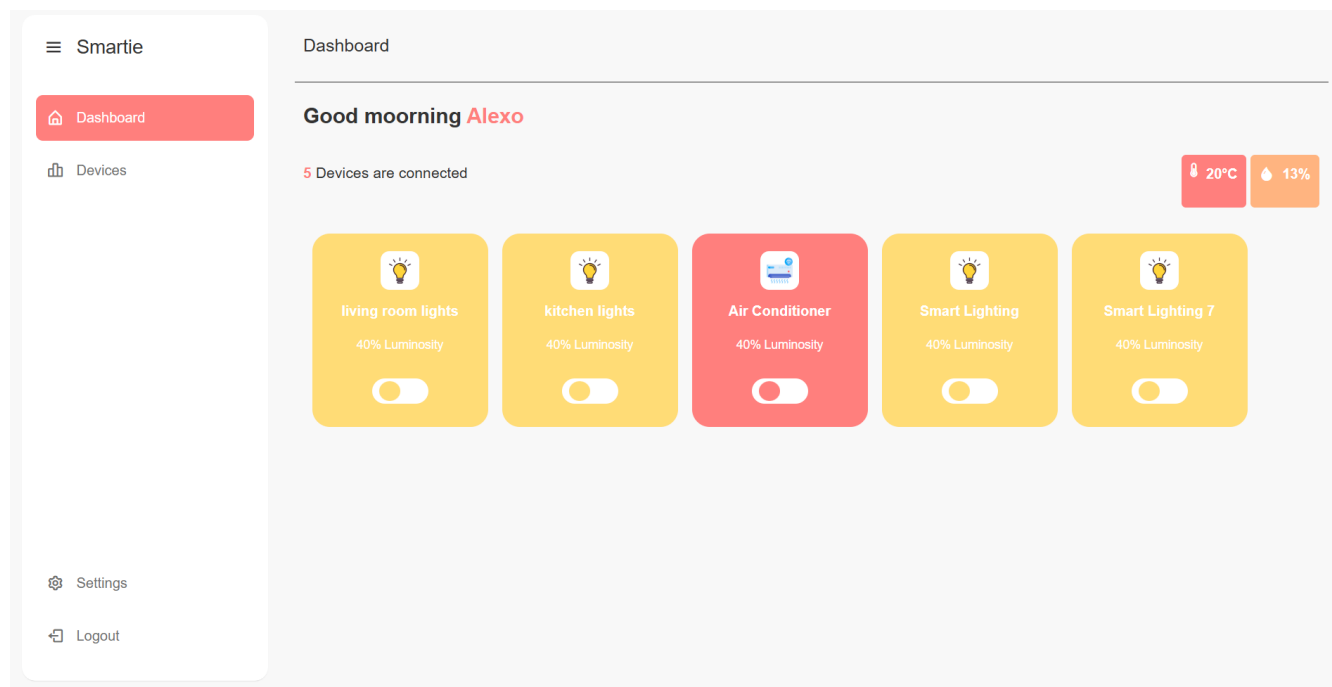
☐ I accept all terms & condition

Register

Already have an account? [Login now](#)

Figure 16 : Form d'inscription

b. Page du tableau de bord :



Smartie

Dashboard

Good morning Alexo

5 Devices are connected

20°C 13%

living room lights
40% Luminosity

kitchen lights
40% Luminosity

Air Conditioner
40% Luminosity

Smart Lighting
40% Luminosity

Smart Lighting 7
40% Luminosity

Settings

Logout

Figure 17 : Tableau de bord

c. Page de gestion des appareils :

Smartie

Dashboard

Devices











Settings

Logout

Devices

All **connected** devices

+ Add Device

Type Id	Name	Port	Status	Action
2	 living room lights	6	Off	
2	 kitchen lights	9	Off	
1	 Air Conditioner	5	Off	
2	 Smart Lighting	4	Off	
2	 Smart Lighting 7	2	Off	

d. Page des paramètres :

Figure 18 • Page de gestion des appareils

Smartie

Dashboard

Devices

Settings

Logout

Settings

Edit Profile

Alexo

alexo@gmail.com

Edit

Figure 19 : Page des paramètres

Conclusion

Le projet **Smartie** s'inscrit dans une démarche innovante visant à proposer une solution intelligente, moderne et accessible pour la gestion d'une maison connectée. Grâce à une architecture bien pensée combinant une application mobile développée en **Flutter**, un backend performant basé sur **FastAPI**, une base de données **MongoDB Atlas**, et une communication matérielle via **ESP32** et **Arduino**, nous avons pu concevoir un système complet et fonctionnel de **domotique intelligente**.

Notre solution permet aux utilisateurs de contrôler leurs appareils à distance, de recevoir des retours en temps réel et même de piloter leur maison à l'aide de **commandes vocales**, rendant l'expérience plus intuitive et plus naturelle. L'ajout d'une interface web en **Angular** vient renforcer la flexibilité du système en élargissant les possibilités d'accès.

Ce projet nous a permis de mettre en œuvre plusieurs compétences techniques apprises durant notre formation, notamment en développement mobile, architecture logicielle, API REST, gestion de base de données NoSQL, communication IoT, et intégration de l'intelligence vocale. Il nous a également confrontés à des défis concrets tels que la gestion des connexions réseau, la synchronisation entre différents systèmes et le traitement des commandes vocales, nous permettant ainsi de progresser tant sur le plan technique que sur le plan organisationnel.

En somme, **Smartie** est une preuve de concept solide d'un écosystème domotique intelligent, personnalisable, et orienté vers l'avenir. Ce projet ouvre la voie à plusieurs perspectives d'amélioration comme l'intégration de l'IA pour des recommandations intelligentes, le contrôle à distance via le cloud, ou encore la gestion d'énergie basée sur des données en temps réel.