WeatherStation

LOUVEAU Julien - MARTINOT Romain - MENTEN Simon
MERCENIER Louis & STREIGNARD Rémi

01

Introduction

Présentation de l'architecture globale

02

IoT

Implémentation de l'IoT dans le projet

03

API

Fonctionnalités apportées par l'API Java

04

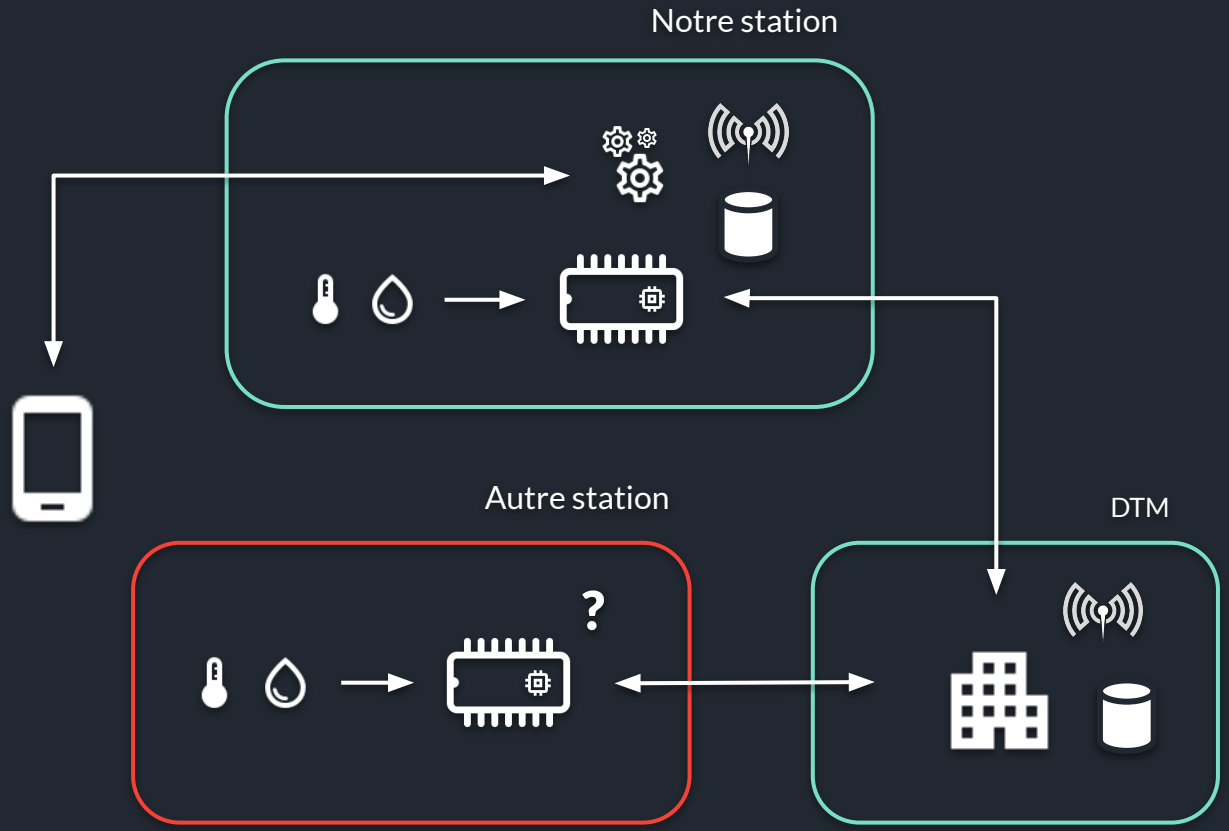
Application

Analyse de la solution mobile

01 Introduction

Présentation de l’architecture globale

Architecture globale



API

Service MQTT

Base de données

Capteurs

Application mobile

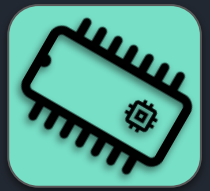
Microcontrôleur

02

IoT

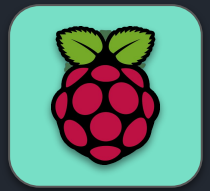
Implémentation de l'IoT dans le projet

Les composants



ESP32

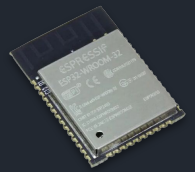
Récupération des
données des capteurs



Raspberry Pi 3

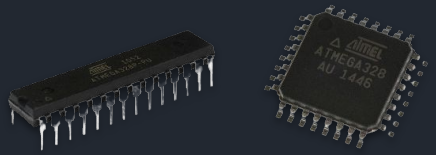
Traitement et
communication des valeurs

ESP32 - Alternatives microcontrôleurs



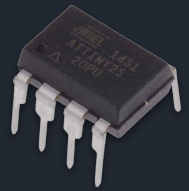
ESP32

~ 3,20 €
~ 35 IO
32 Mb de mémoire
Connexion wifi et Bluetooth



atmega328 (arduino uno)

~ 1.60 €
~ 19 IO
32 kB de mémoire

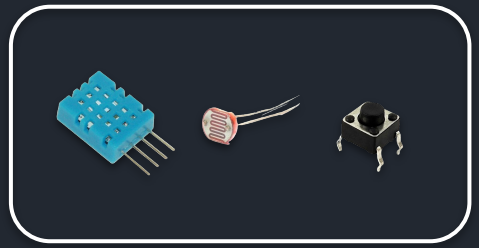


attiny25

~ 0.9 €
~ 6 IO
2 kB de mémoire

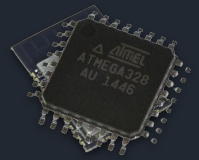
ESP32

capteurs



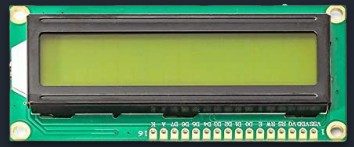
ATMEGA328P

ESP32

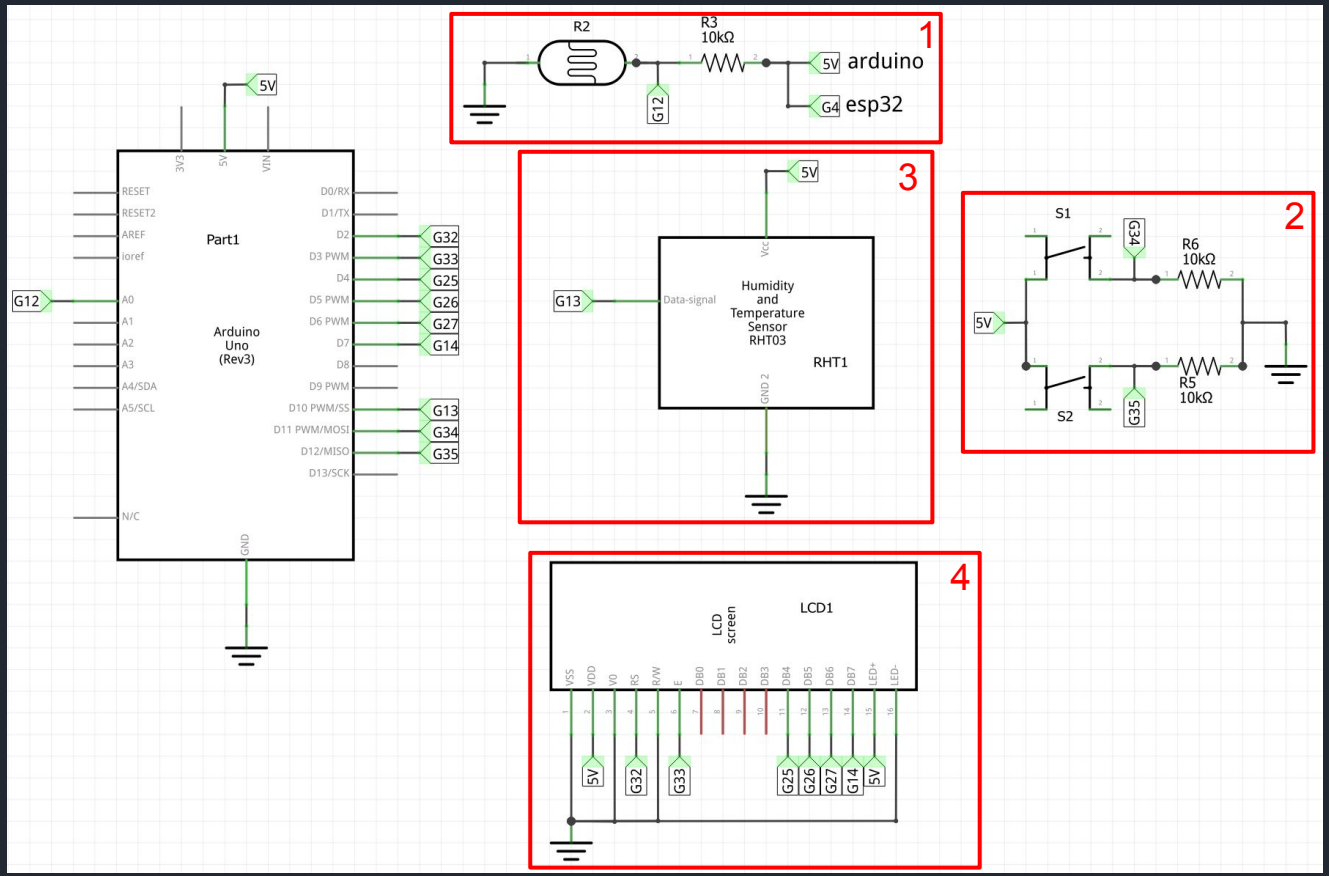


Communication Bluetooth

écran LCD



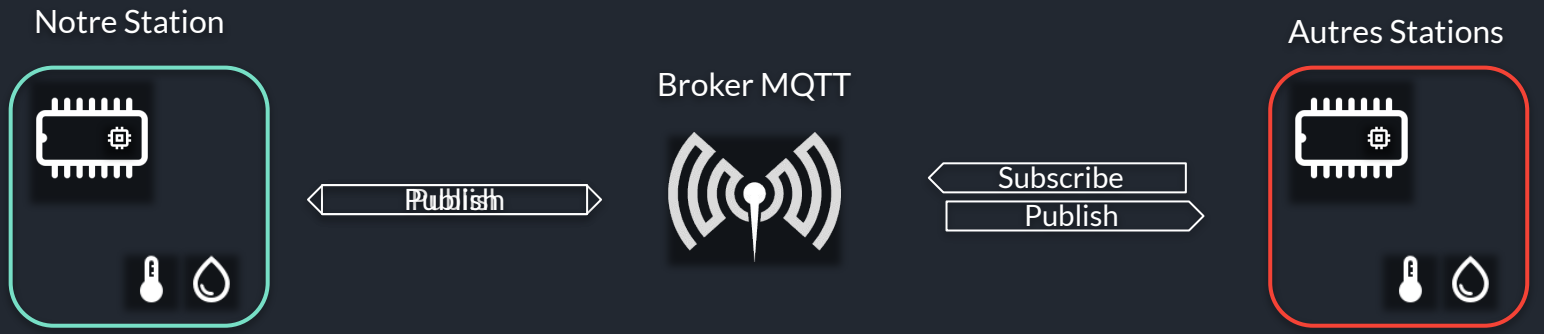
ESP32 - Station météo



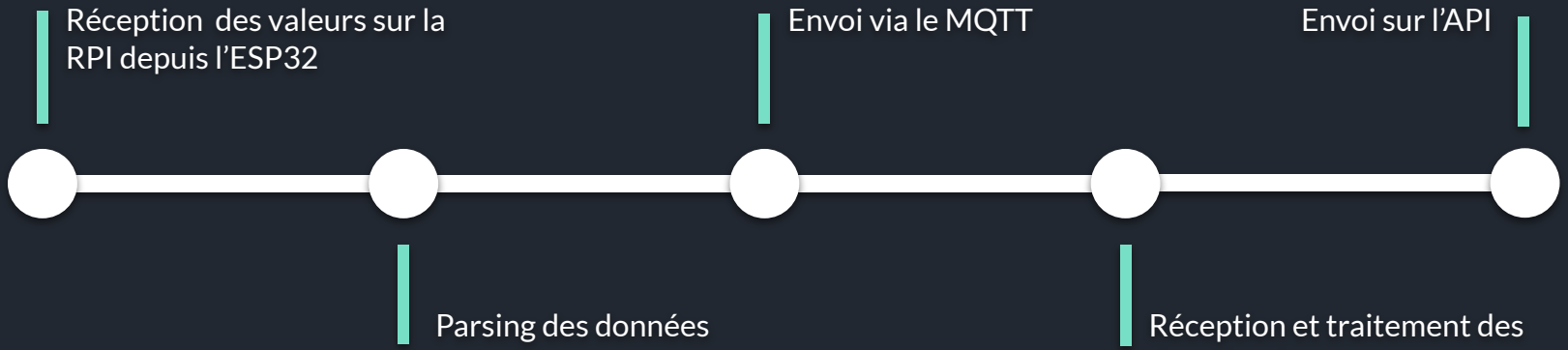
Raspberry Pi 3 - Comparatif

	Raspberry Pi 3	Ordinateur classique
Portabilité	++	-
Performances	-	++
Consommation	~3W	80W+
Discrétion	Très discret	Visible
Avantages/Prix	++	-

MQTT - Implémentation



Traitement des données - IoT



03

API

Fonctionnalités apportées par l'API Java

Choix technologiques 01

Avantages API

- Données personnalisées
- Contrôles supplémentaires
- Séparation client / base de données

Avantages Java

- Portabilité
- Plus rapide que d’autres langages interprétés
- Nombreux outils disponibles

Choix technologiques 02

Qu’est ce que REST ?

- REpresentational State Transfer
- Style d’architecture web
- Expose des ressources
- Requêtes HTTP
- Formats standards (XML, JSON)
- Stateless

Avantages

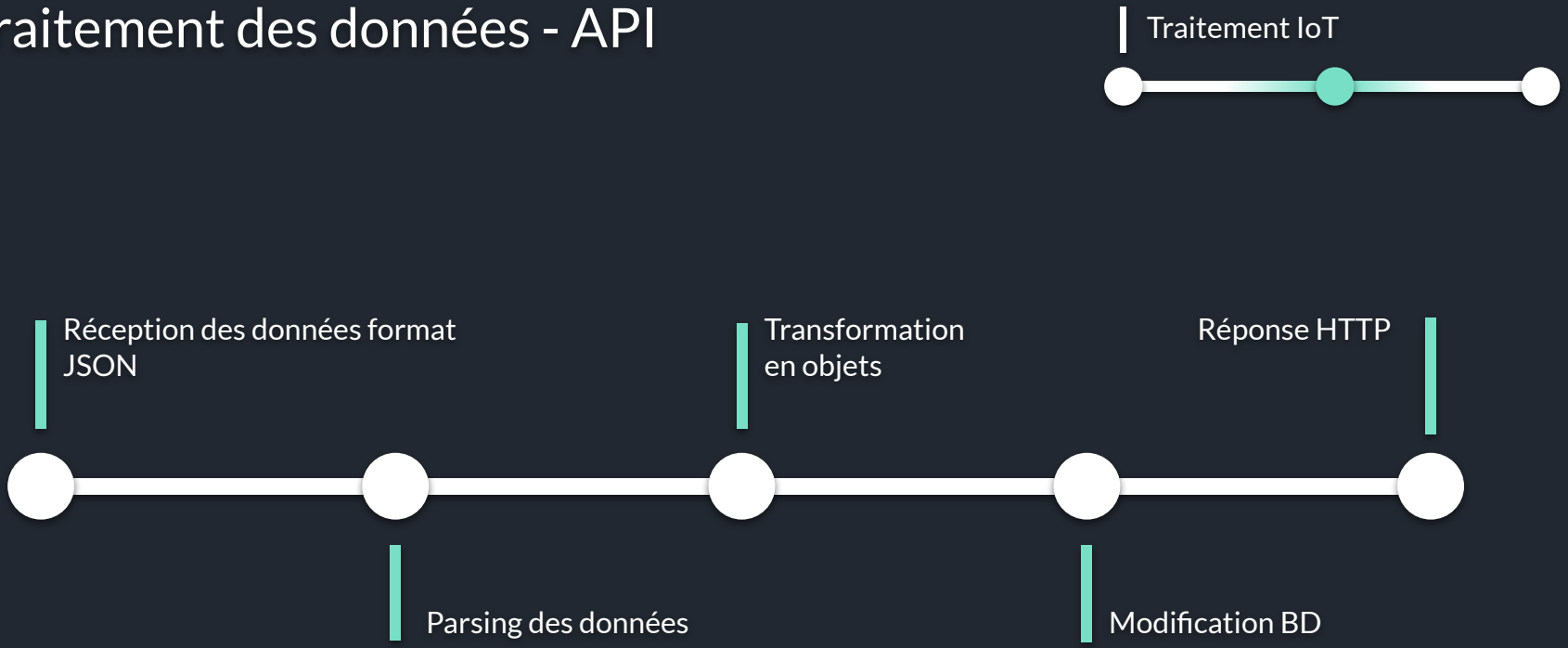
- Indépendance client / serveur
- Load balancing
- Formats compatibles dans le temps

Communication avec l'API

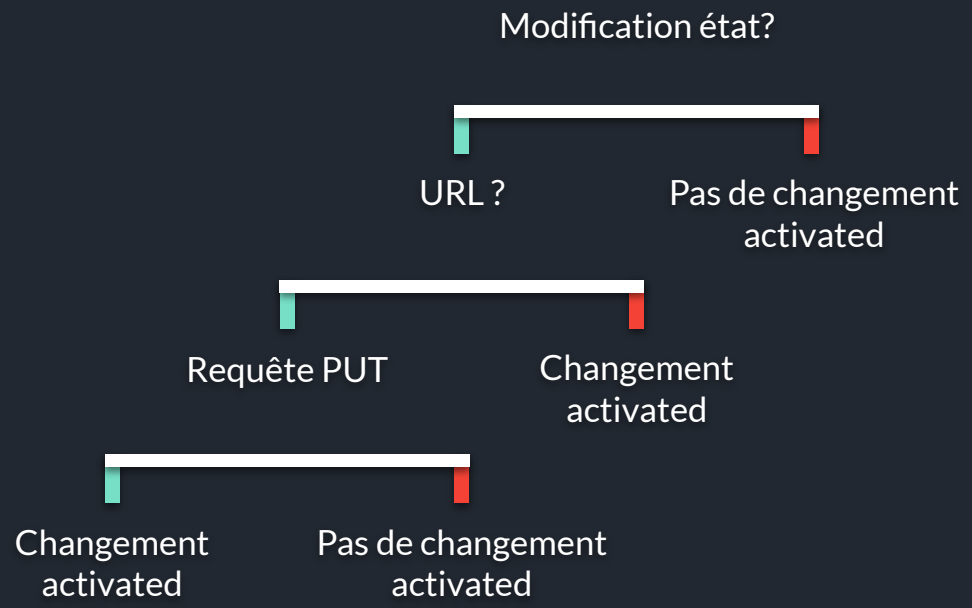
GET	/sensor	Obtenir les informations de tous les capteurs
GET	/sensor/{id}	Obtenir les informations du capteur qui possède l'identifiant {id}
POST	/sensor	Créer un nouveau capteur
PUT	/sensor/{id}	Modifier le capteur qui possède l'identifiant {id}
DELETE	/sensor/{id}	Supprimer le capteur qui possède l'identifiant {id}

GET	/sensor/unassigned	Obtenir les capteurs qui ne sont pas assignés à une pièce
POST	/sensor/{id}/value	Enregistrer une valeur pour le capteur qui possède l'identifiant {id}
GET	/sensor/{id}/value	Obtenir la dernière valeur pour le capteur qui possède l'identifiant {id}

Traitement des données - API



Communication API - Devices



devices

id BIGINT

activated BIT(1)

is_fav BIT(1)

name VARCHAR (255)

room_id BIGINT (FK)

url VARCHAR (255)

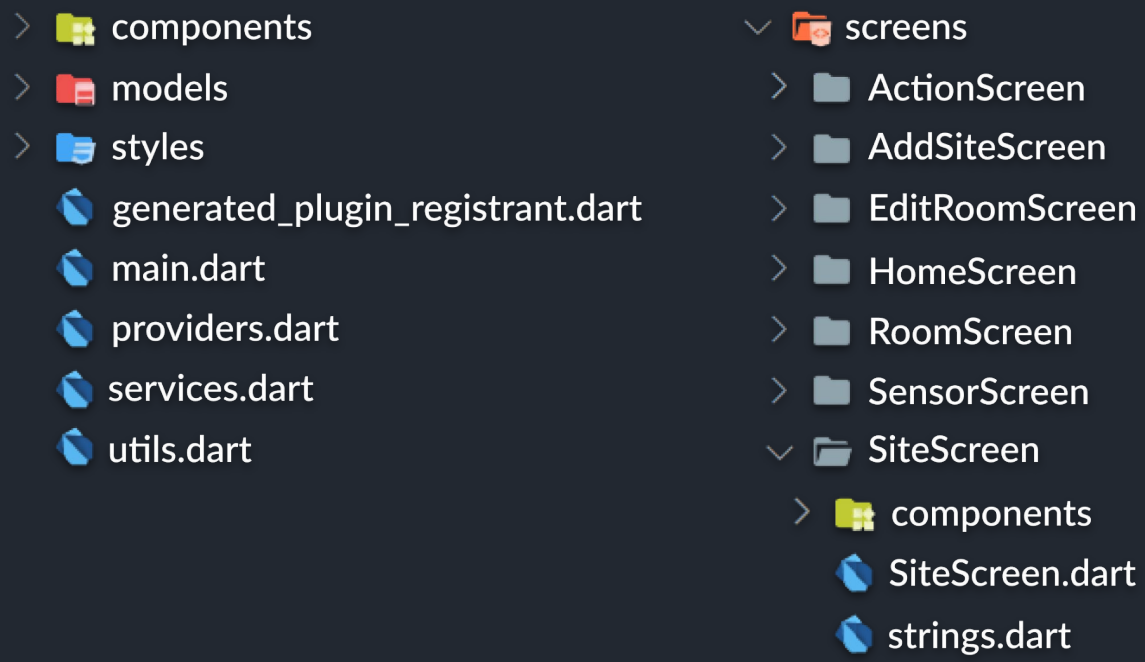
04 Application

Analyse de la solution mobile

Mobile - Choix technologique

	Flutter	React Native	Android
Type de l'app	Hybride (natif)	Hybride (basé sur un container web)	Natif
Langage	Dart	JavaScript & HTML	Java / Kotlin
Plateformes ciblées	Web & Mobile & Desktop (Beta)	Mobile, UWP & Web (React uniquement)	Mobile (seulement Android)
Coûts, facilité de développement	-	++	--
Performances	++	-	++
Design	Réutilisable, facile à personnaliser	Réutilisable, difficile à personnaliser	Réutilisable, et drag & drop

Organisation du code



Widgets - Gestion de l'état

StatelessWidget

```
class StatelessExample extends StatelessWidget {

@override
 Widget build(BuildContext context) {
   return Container( );
 }
}
```

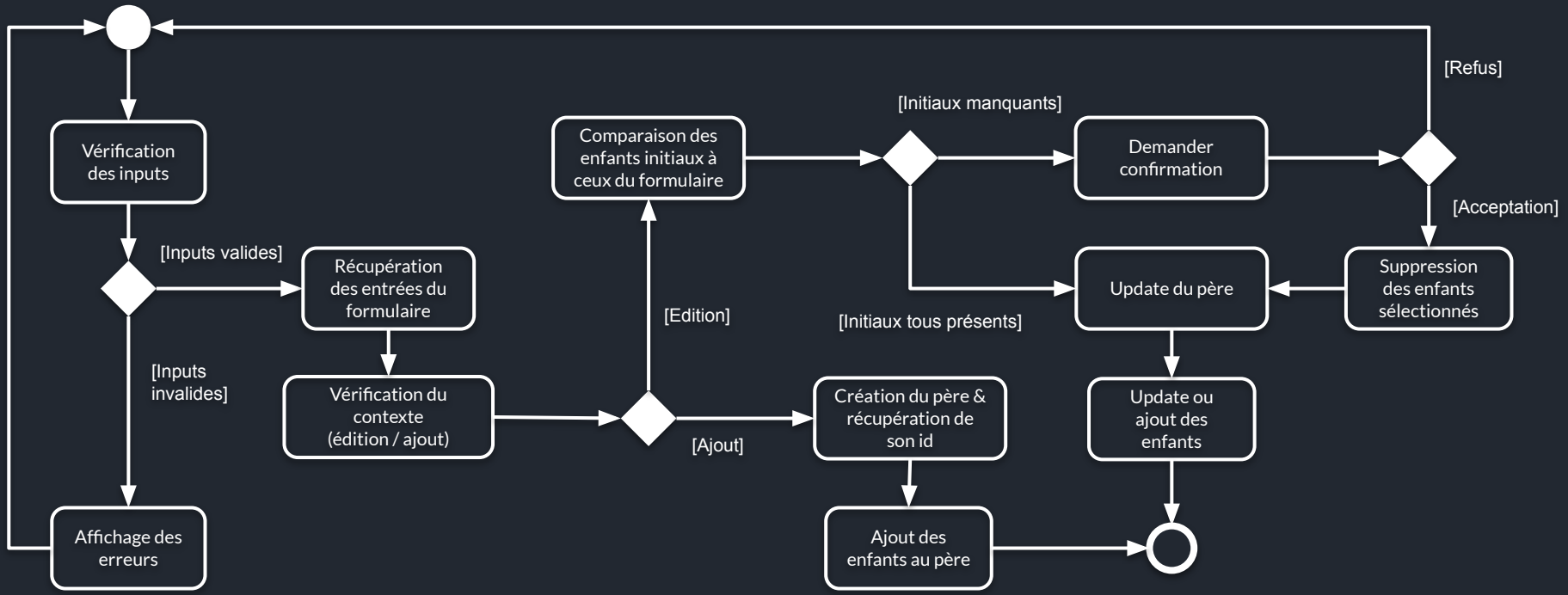
StatefulWidget

```
class StatefulExample extends StatefulWidget {
 @override
 _StatefulExampleState createState() => _StatefulExampleState();
}

class _StatefulExampleState extends State<StatefulExample> {

@override
 Widget build(BuildContext context) {
   return Container( );
 }
}
```

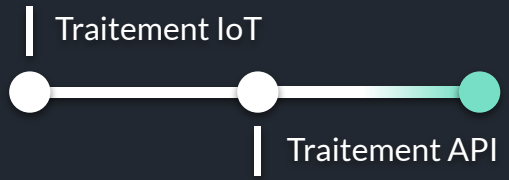
Fonctionnalités

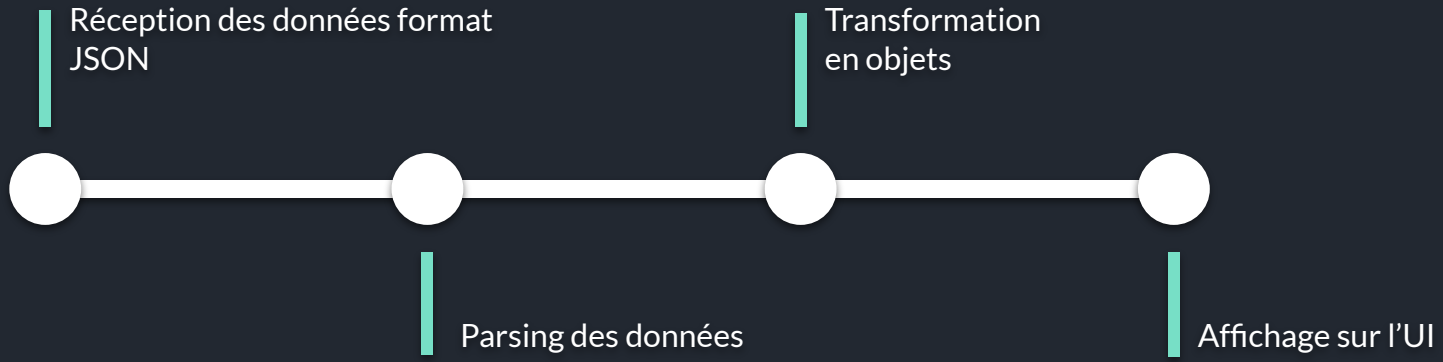
- Barre de recherche
- Favoris
- Refresh
- Interaction en live avec les devices
- Full CRUD sur les sites
- Full CRUD sur les rooms
- Modifications des liaisons avec les sensors et devices
- Création et suppression d’événements

Logique des formulaires - Diagramme



Traitement des données - Application



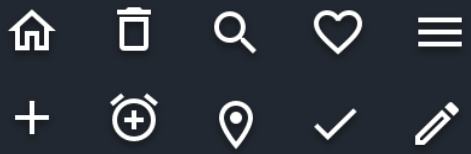


Confort utilisateur

Affichage des données

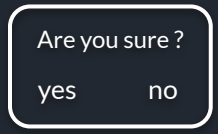
Attrait des couleurs

Icônes conventionnelles



Bulles d'information

This field is required.

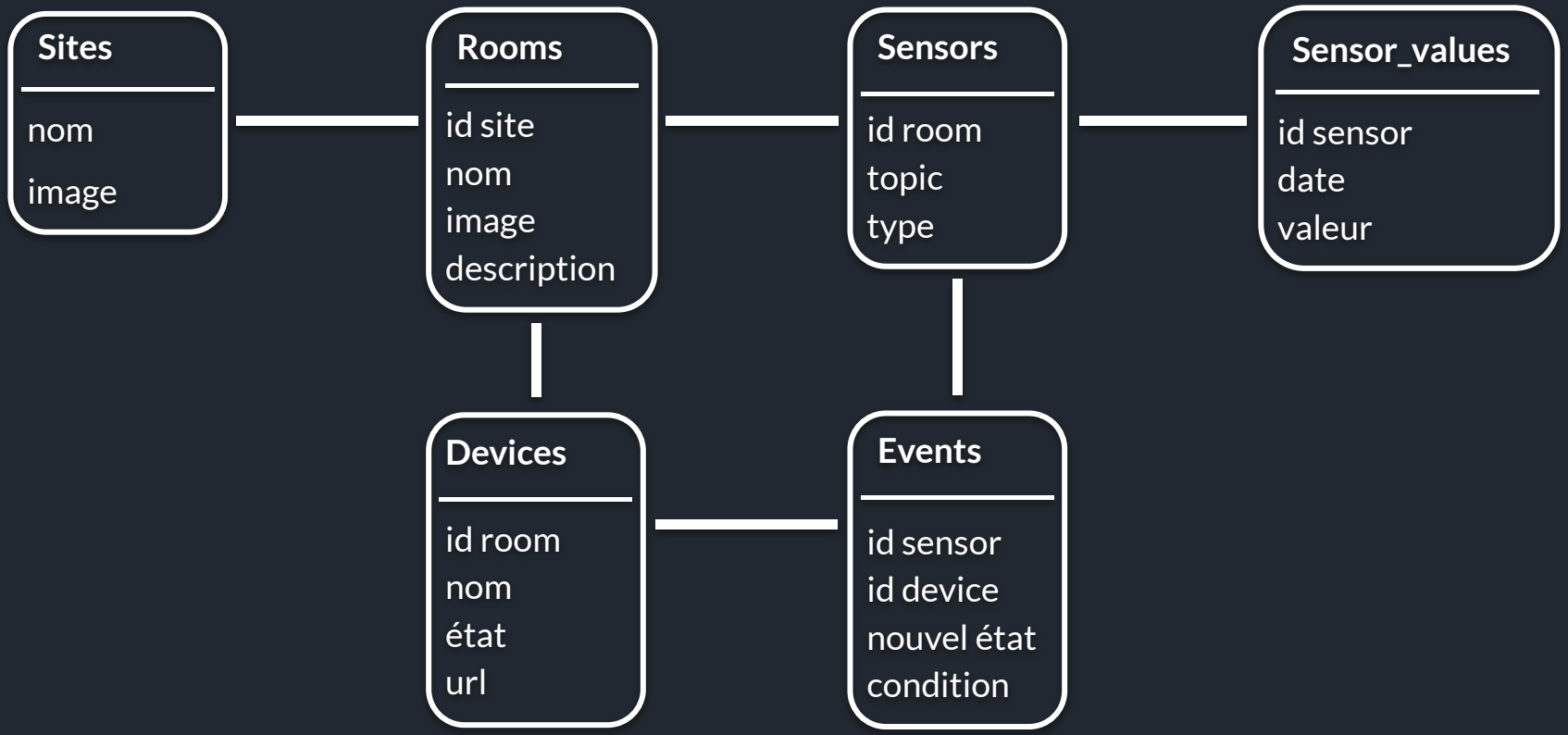


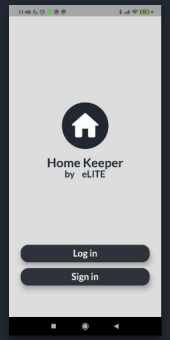
Sélections en évidence

Positionnement intuitif

Démonstration

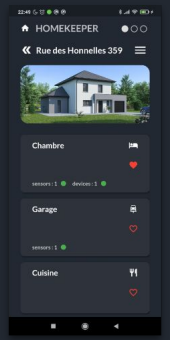
Base de données



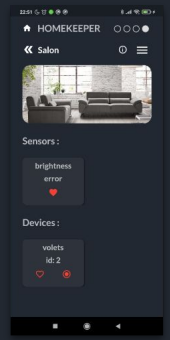

Home Keeper
by eLITE
Log in
Sign in

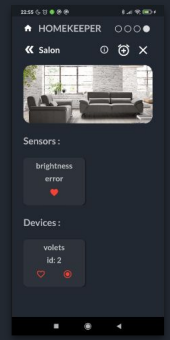

Enter a search term
Rue des Honnelles 359
DTM
Rue Haute 294

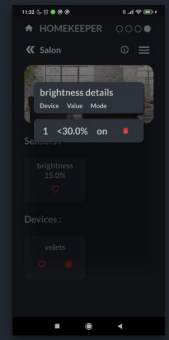

Enter a ...
Rue des Honnelles 359
DTM
Rue Haute 294

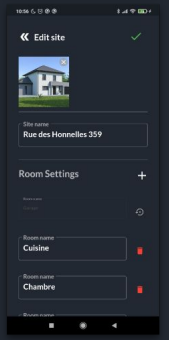

HOMEKEEPER
Rue des Honnelles 359
Chambre
Garage
Cuisine

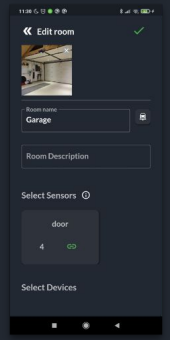

HOMEKEEPER
Rue des Honnelle...
Chambre
Garage
Cuisine

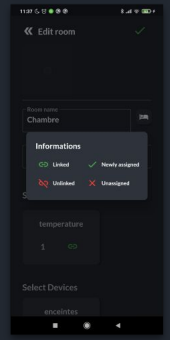

HOMEKEEPER
Salon
Sensors :
brightness
error
Devices :
volets
id: 2


HOMEKEEPER
Salon
Sensors :
brightness
error
Devices :
volets
id: 2

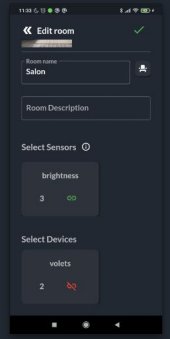

HOMEKEEPER
Salon
brightness details
Device
Value
Mode
1
<30.0%
on
Devices :
volets

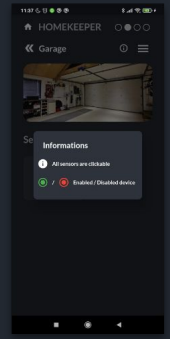

Edit site
Site name
Rue des Honnelles 359
Room Settings
Room name
Cuisine
Room name
Chambre


Edit room
Room name
Garage
Room Description
Select Sensors
door
4
Select Devices


Edit room
Room name
Chambre
Informations
Linked
Newly assigned
Unlinked
Unassigned
temperature
1
Select Devices
enceintes


Add action
Select sensor :
brightness
Activation logic :
Less
More
Activation value :
Value here
0
Select device :
enceintes
volets
Action :
OFF
ON


Edit room
Room name
Salon
Room Description
Select Sensors
brightness
3
Select Devices
volets
2


HOMEKEEPER
Garage
Informations
All sensors are clickable
Enabled / Disabled device