

# STATION METEO

RESKILLING C/C++ Developpeur Embarqué  
pour AKKA TECHNOLOGIES  
Via AJC Formation

## Participants

Lucas SANER

Mickael ANTHEAUME

Stephane CUILLERDIER

# Timing (temporaire)

- 1/3 spec / recherche / equipe / choix technique : 10 min
- 1/3 architecture / conception serveur : 10 min
- 1/3 conception client / demo : 10 min



# Sommaire

24/06/2021

# LA SPECIFICATION

PRESENTATION DU PROJET / LES PREREQUIS ET FINALITE

# Intitulé du projet :

- L'objectif de ce projet est de concevoir une application PC Station Météo.
- On souhaite afficher sur cette station Météo des informations météorologiques de 2 points géographiques différents:
  - en mer, ce qu'on appellera la « Balise Mer »
    - Température en °C/°F de -40°C à 50°C
    - Résolution : 0.1°C relevé toutes les 10 minutes
    - Taux d'humidité
    - Pression
  - d'une ville choisie, ce qu'on appelle la « Balise Ville »
    - Température en °C/°F de -40°C à 50°C
    - Résolution : 0.1°C relevé toutes les 10 minutes
    - Gestion de l'affichage de pictogrammes associés
    - Affichage de la Ville
    - Graphique prévisionnel pour les 5 jours suivants
- Ainsi que l'affichage de l'heure et de la date

Il faudra aussi créer une partie d'administration permettant de configurer certains paramètres :

- Format de l'heure 12H ou 24H
- Choix de la Ville
- Unité de Température Fahrenheit ou Celsius
- Possibilité de choisir les styles d'affichages :
  - Famille de Police
  - Couleur : (Chaque style sera décliné en Mode Jour/Nuit)
  - Choix de la langue •: Anglais / Français

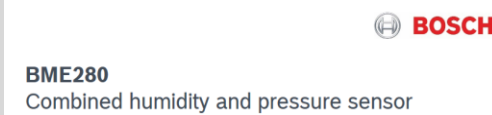
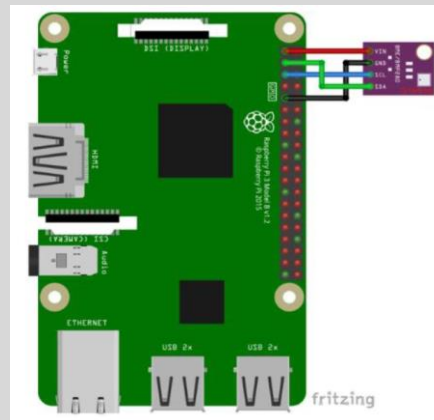
Une partie facultative, S'il reste du temps :

- vous enregistrerez toutes les heures les informations de la balise au sein d'une base de données. Le but sera d'afficher la température moyenne des 12 dernières heures et de l'afficher au sein de votre station.

# Matériel mis a disposition :

- La balise Mer (en plein cœur de la mer) ce compose :
  - Raspberry Pi 3 Model B+
  - Marque: U:Create
  - Processeur: ARM
  - Vitesse du processeur: 1.40 GHz
  - Nombre de cœurs: 4
  - Taille de la mémoire vive: 1GB
  - Type de technologie sans fil: 802.11bgn, 802.11ac
  - Nombre de ports USB 2.0: 4
  - Accès en TCP ou HTTP

Branchement du capteur  
sur le Raspberry Pi en I2C



Et d'un Capteurs d'humidité BME280:

Capteur de temperature :

Temperature: -40...85°C

Precision : 0,01°C

Capteur d'humidité

Humidité : 0...100%

Temps de réponse : 1 s

Precision : ±3%

Capteur de pression

Pression: 300...1100 hPa

Bruit de mesure : 0.2 Pa

Interface : I2C



Un programme mis a disposition permet de tester l'état de fonctionnement du capteur

```
pi@pi:~ $ python bme280.py
Chip ID      : 96
Version      : 0
Temperature  : 23.59 c
Pressure     : 984.583980249 hPa
Humidity     : 34.0987183674 %
```

24/06/2021

# L'EQUIPE

PRESENTATION / ROLE / REPARTITION



# Participants au projet :

- Lucas SANER
- Mickael ANTREAUME
- Stéphane CUILLERDIER

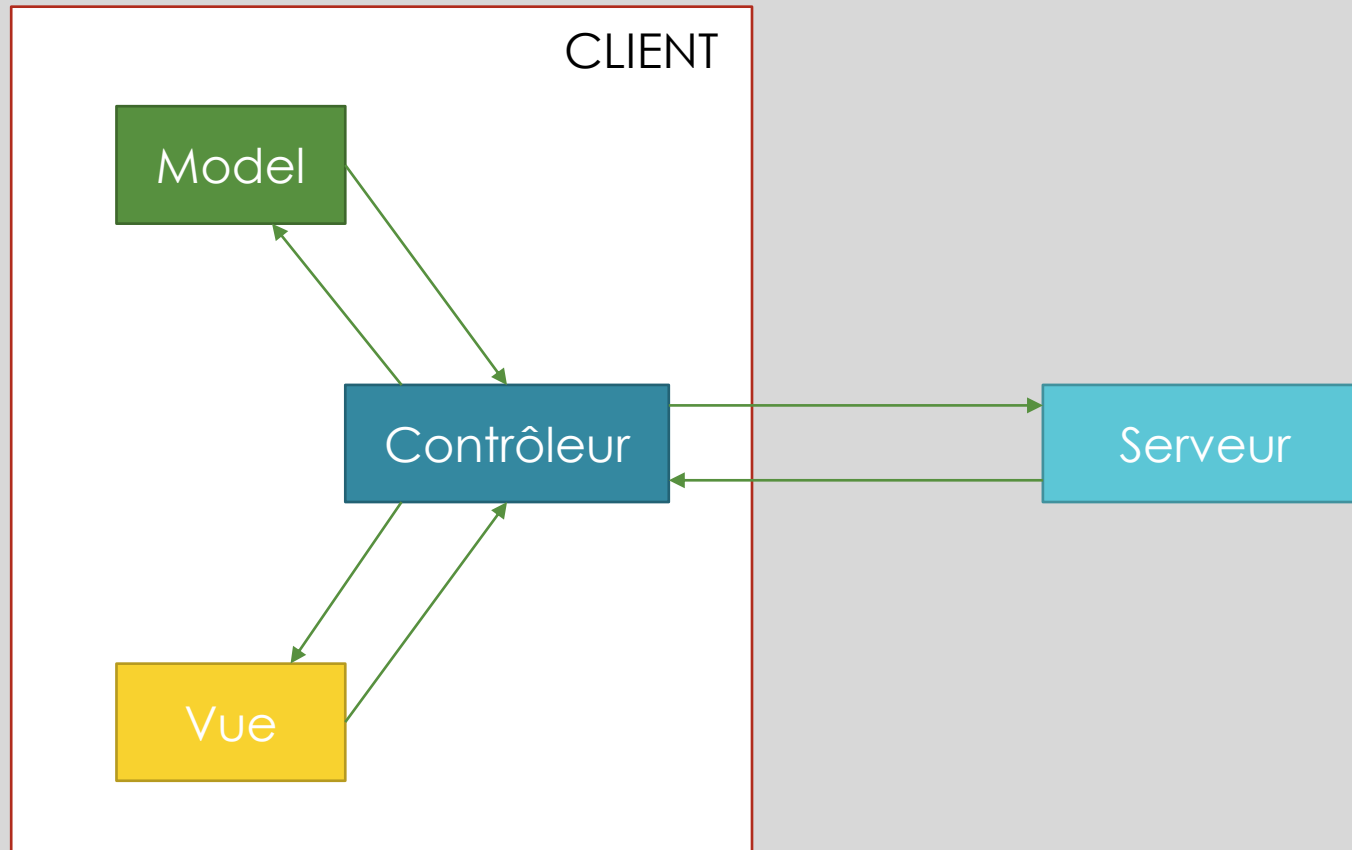


24/06/2021

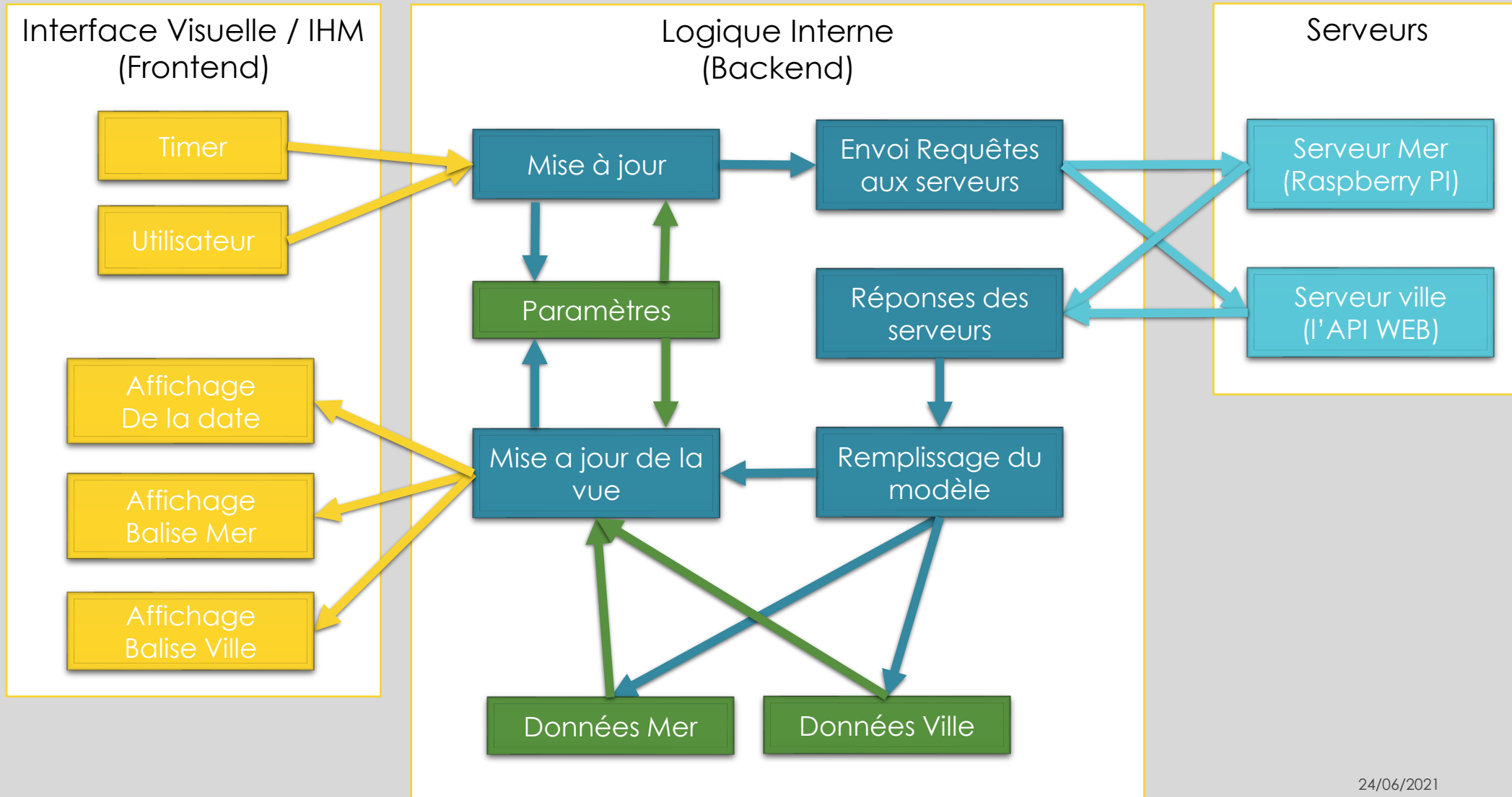
# L'ARCHITECTURE DU PROJET

MVC / SCHEMAS LOGIQUES / FONCTIONNEMENT CLIENT-SERVEUR

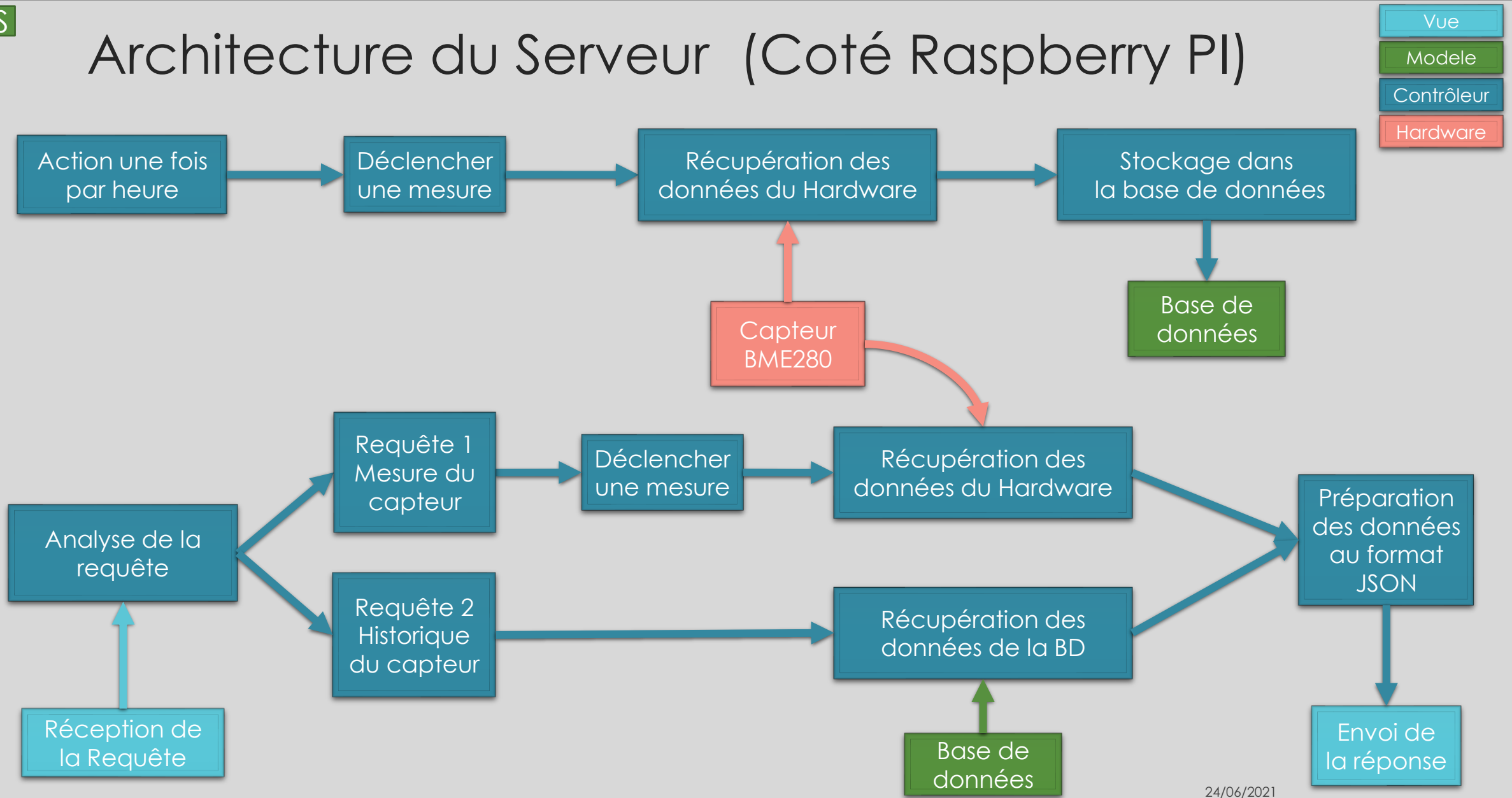
# MVC : Model-View-Contrôleur dans un contexte Client-Serveur



# Architecture du Client (Coté utilisateur)



# Architecture du Serveur (Coté Raspberry PI)



24/06/2021

# CHOIX TECHNIQUES

MATERIELS / OUTILS / PLATEFORMES



## Comparaison des apis web trouvées:

APIS:	infoclimat	meteomatics	accuweather	openweathermap	meteoconcept
Nombre d'appel serveur autorisé	5000/jr	1000/14jr (version d'essai 14jr )	50 appels /jr	1000/jr	500/jr
gratuité service	gratuit	Payant (essai 14jr)	gratuit	gratuit	gratuit
Type de recherche(par ville.....)	Coordonnées géographiques (lat-long)	Coordonnées géographiques (lat-long)	Ville code postal coordonnées géographiques (lat-long)	Ville/code postal/coordonnées géographiques (lat-long)	ville
Données: Température/pression humidité/icône	Pas de logo	oui	oui	oui	oui
Forecast 5 jours	oui	oui	oui	oui	oui



## Comparaison des apis web trouvées:

APIS:	infoclimat	meteomatics	accuweather	openweatherm ap	meteoconcept
Nombre d'appel serveur autorisé	5000/jr	1000/14jr (version d'essai 14jr )	50 appels /jr	1000/jr	500/jr
gratuité service	gratuit	Payant (essai 14jr)	gratuit	gratuit	gratuit
Type de recherche(par ville.....)	Coordonnées géographiques (lat-long)	Coordonnées géographiques (lat-long)	Ville/code postal/coordonnées géographiques (lat-long)	Ville/code postal/coordonnées géographiques (lat-long)	ville
Données: Température/pression humidité/icône	Pas de logo	oui	oui	oui	oui
Forecast 5 jours	oui	oui	oui	oui	oui

# Choix du Serveur WEB HTTP

SERVEURS  
NGINX/APACHE2  
+  
APPLICATION  
CGI



Trop lourd pour notre besoin  
  
Pas besoin de sécurité vu la  
nature des données

APPLICATION  
+  
Libuv / uv-cpp  
(Librairie HTTP)



Très léger  
  
Facilement déployable



# Choix par fonctionnalité

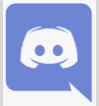
- Contrôle du Capteur BME280 : Driver officiel Bosch
- Base de données : Sqlite
- Format de donnée pour les échanges HTTP : JSON
- Envoi/Réception de requêtes HTTP : libUV
- Système Multilingue : QT Translator
- Affichage de graphique : Création d'un Composant personnalisé
- Gestion des thèmes visuel : QT Stylesheet
- Interface graphique : QT + Designer intégré
- Chargement / Sauvegarde des paramètres : fichier INI

# Outils / Librairie / Framework / Matériel

## Outils de productivité :



GitHub



Discord



Powerpoint



Mobaxterm : connexion en ssh

## Matériel :



Raspberry Pi



Capteur BME280

## Outils de développement :



c/c++



Framework QT fait c++



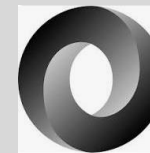
Libuv : Serveur HTTP



Sqlite3



Open Weather Map (Api Web)



Json

24/06/2021

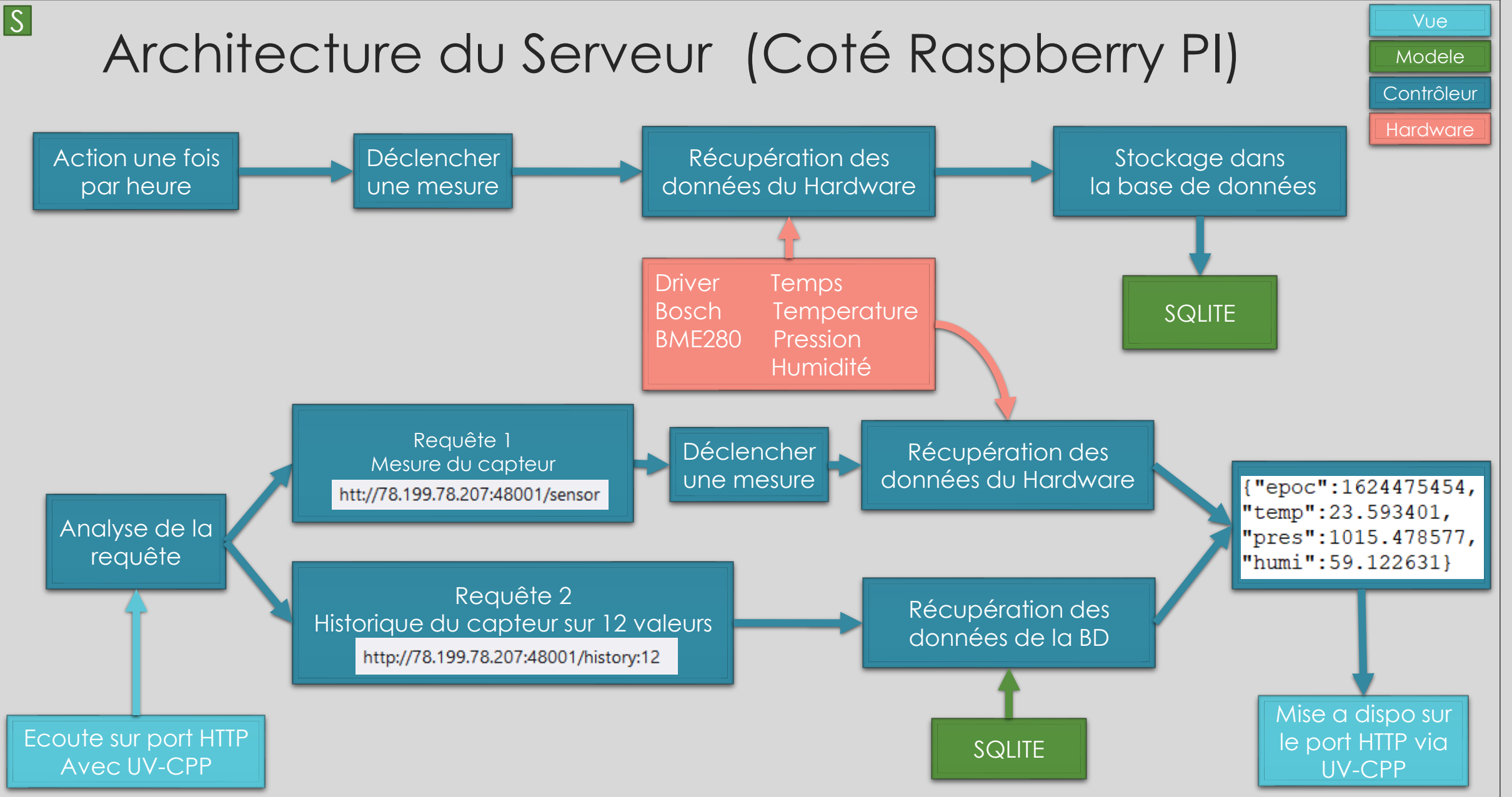
# REALISATION

IMPLEMENTATION ITERATIVE / TESTS / DEBUG / AJOUT DE FONCTIONALITES

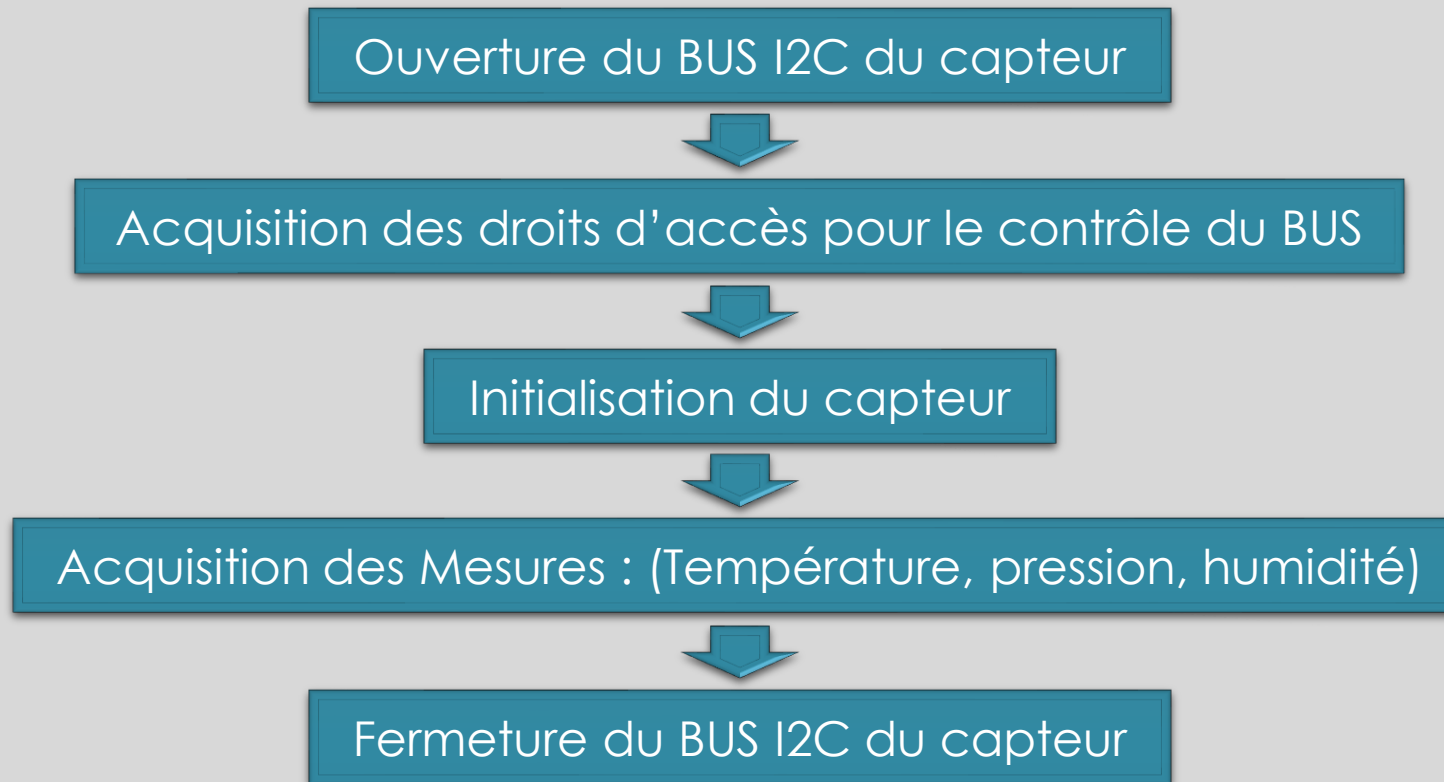
24/06/2021

# CONCEPTION DU SERVEUR

# Architecture du Serveur (Coté Raspberry PI)



# Déclenchement d'une mesure du capteur en I2C



# Extraction depuis la base de données

Requête SQL pour extraire les 12 dernière mesures depuis la base de donnée Sqlite :

```
select * from tbl_bme280_sensor_history order by epoc_time desc limit 12;
```

```
pi@pi:/usr/share/bme280Server $ sqlite3 database.db3
SQLite version 3.27.2 2019-02-25 16:06:06
Enter ".help" for usage hints.
sqlite> .tables
tbl_bme280_sensor_history
sqlite> select * from tbl_bme280_sensor_history order by epoc_time desc limit 12;
1624474569|23.0199|1015.53|56.6401
1624470969|23.8253|1015.11|61.2338
1624467369|23.5236|1015.29|59.2408
1624463769|23.389|1015.35|59.6951
1624460169|23.4263|1015.3|60.2304
1624456569|23.4887|1015.4|60.138
1624452969|23.5784|1015.48|59.959
1624449369|23.5859|1015.66|59.8327
1624445769|23.6956|1015.31|59.5925
1624442169|23.7804|1015.06|59.3967
1624438569|23.5984|1014.96|60.577
1624434969|24.2766|1014.98|59.9618
sqlite> █
```

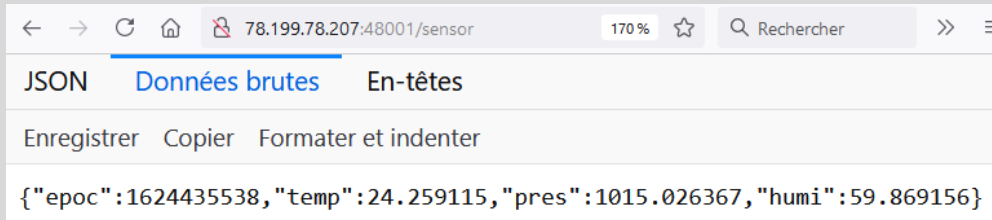


Envoi des données par le port http

```
{
  "count": 12,
  "history": [
    {
      "epoc": 1624474569,
      "temp": 23.0199,
      "pres": 1015.53,
      "humi": 56.6401
    },
    {
      "epoc": 1624470969,
      "temp": 23.8253,
      "pres": 1015.11,
      "humi": 61.2338
    },
    {
      "epoc": 1624467369,
      "temp": 23.5236,
      "pres": 1015.29,
      "humi": 59.2408
    },
    {
      "epoc": 1624463769,
      "temp": 23.389,
      "pres": 1015.35,
      "humi": 59.6951
    },
    {
      "epoc": 1624460169,
      "temp": 23.4263,
      "pres": 1015.3,
      "humi": 60.2304
    },
    {
      "epoc": 1624456569,
      "temp": 23.4887,
      "pres": 1015.4,
      "humi": 60.138
    },
    {
      "epoc": 1624452969,
      "temp": 23.5784,
      "pres": 1015.48,
      "humi": 59.959
    },
    {
      "epoc": 1624449369,
      "temp": 23.5859,
      "pres": 1015.66,
      "humi": 59.8327
    },
    {
      "epoc": 1624445769,
      "temp": 23.6956,
      "pres": 1015.31,
      "humi": 59.5925
    },
    {
      "epoc": 1624442169,
      "temp": 23.7804,
      "pres": 1015.06,
      "humi": 59.3967
    },
    {
      "epoc": 1624438569,
      "temp": 23.5984,
      "pres": 1014.96,
      "humi": 60.577
    },
    {
      "epoc": 1624434969,
      "temp": 24.2766,
      "pres": 1014.98,
      "humi": 59.9618
    }
  ]
}
```

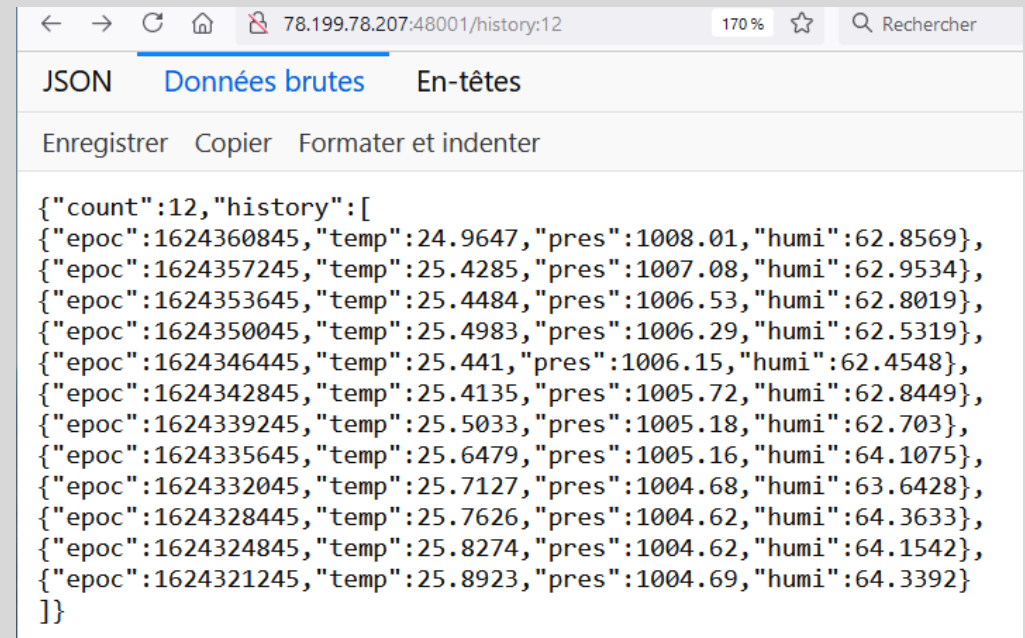
# Url's HTTP disponibles

`http://ip:port/sensor`



`http://ip:port/history:N`  
(n est un nombre de 1..1e6)

Ici les 12 dernières mesures





# Url's HTTP disponibles

<http://ip:port/>

The screenshot shows the BME280 HTTP Server web interface. At the top, it says 'BME280 HTTP Server'. Below that, it states 'This Server expose JSon Datas of I2C Sensor BME280'. There is a section titled 'Available datas per measures' with a list of sensors: Epoch time (seconds), Temperature (Celsius), Pressure (hPa), and Humidity (%). Below this is a section titled 'Availables Urls queries' with three examples: 'Get infos' with the URL 'http://ip:port/infos', 'Start a measure of the sensor' with the URL 'http://ip:port/sensor', and 'Retrieve an history of last N hourly sensor measures' with the URL 'http://ip:port/history.N (here n is 12)'. At the bottom, it says 'Happy Logging :)'.

**BME280 HTTP Server**

This Server expose JSon Datas of  
I2C Sensor BME280

**Available datas per measures**

- Epoch time (seconds)
- Temperature (Celsius)
- Pressure (hPa)
- Humidity (%)

**Availables Urls queries**

Get infos  
Ex : <http://ip:port/infos>

Start a measure of the sensor  
Ex : <http://ip:port/sensor>

Retrieve an history of last N hourly sensor measures  
Ex : <http://ip:port/history.N> (here n is 12)

Happy Logging :)

<http://ip:port/infos>

The screenshot shows the BME280 HTTP Server web interface with the JSON response for the /infos endpoint. The browser address bar shows '78.199.78.207:48001/infos'. The response is displayed in a JSON format with the following fields:

- server\_version: "0.1.55.X32"
- server\_command: "/usr/local/bin/bme280Server -b /dev/i2c-1 -d /usr/share/bme280Server/database.db3 -t 3600 -p 80"
- database\_records: 14955
- database\_version: "3.27.2"
- database\_path: "/usr/share/bme280Server/database.db3"
- os\_sys\_name: "Linux"
- os\_release: "5.10.17-v7+"
- os\_version: "#1403 SMP Mon Feb 22 11:29:51 GMT 2021"
- os\_machine: "armv71"

**JSON** Données brutes En-têtes

Enregistrer Copier Tout réduire Tout développer Filtre le JSON

server\_version: "0.1.55.X32"

server\_command: "/usr/local/bin/bme280Server -b /dev/i2c-1 -d /usr/share/bme280Server/database.db3 -t 3600 -p 80"

database\_records: 14955

database\_version: "3.27.2"

database\_path: "/usr/share/bme280Server/database.db3"

os\_sys\_name: "Linux"

os\_release: "5.10.17-v7+"

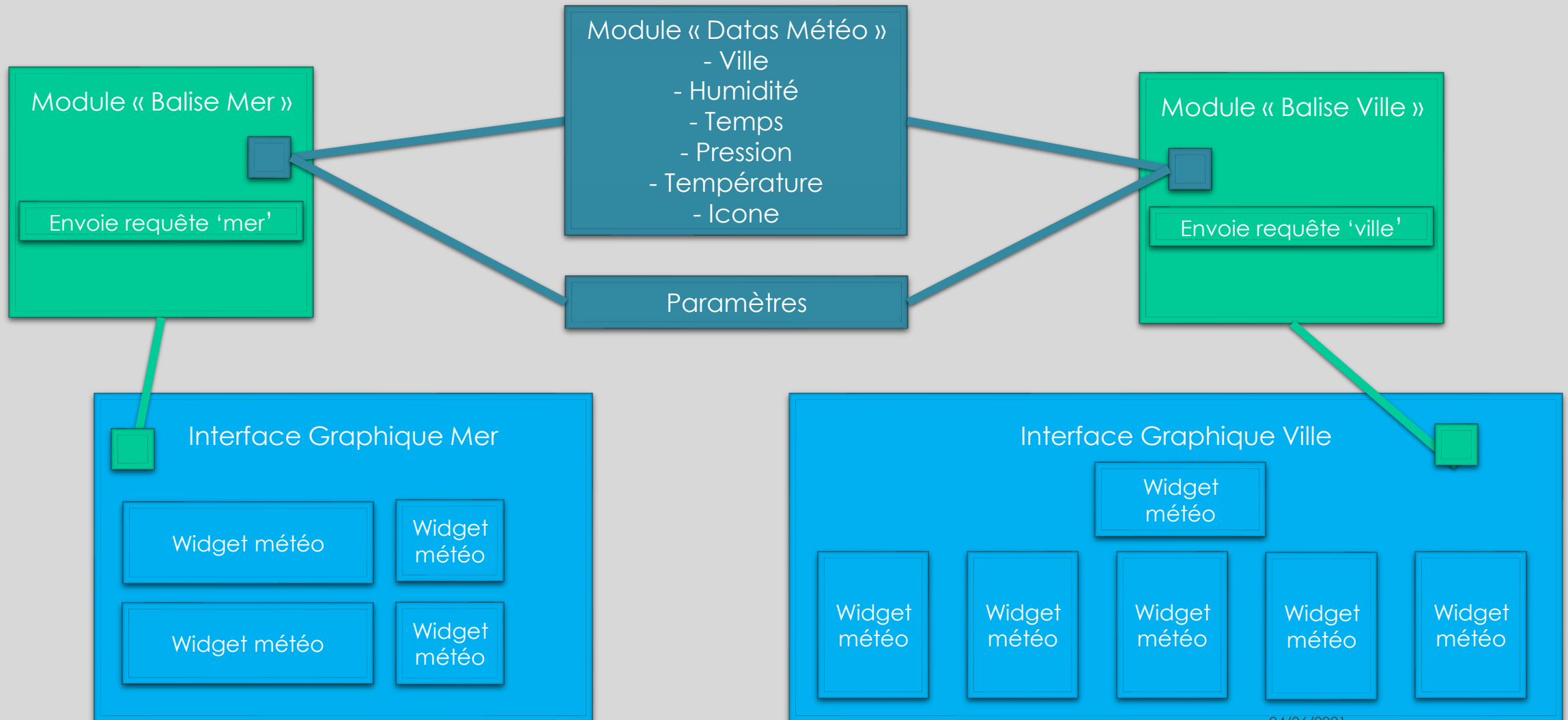
os\_version: "#1403 SMP Mon Feb 22 11:29:51 GMT 2021"

os\_machine: "armv71"

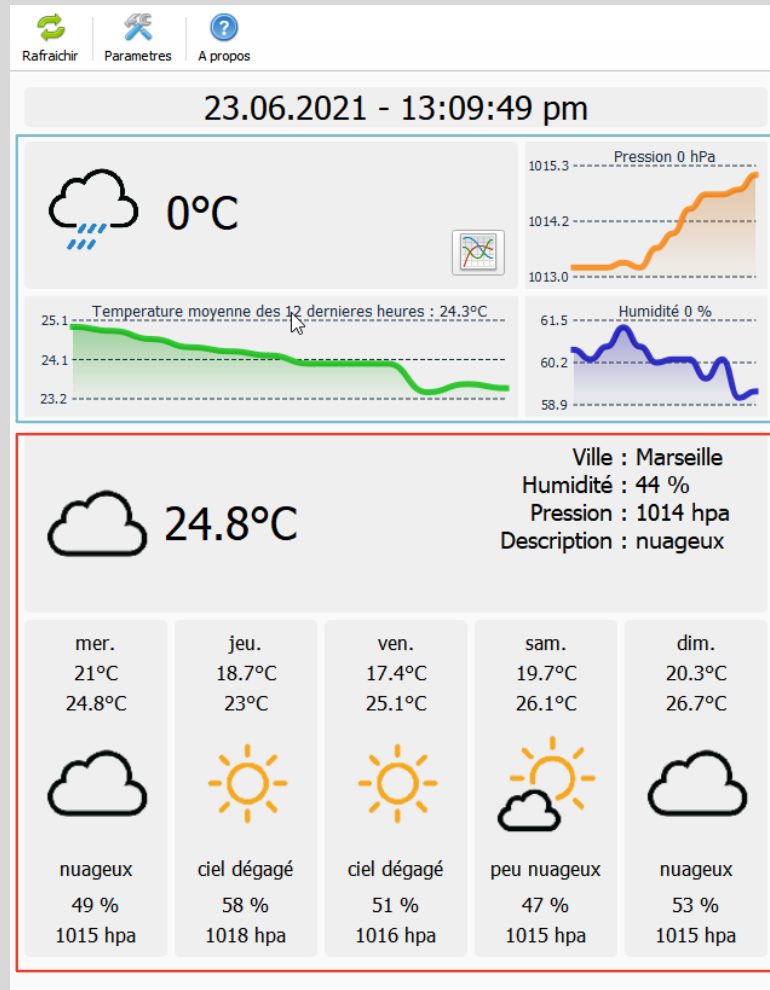
24/06/2021

# CONCEPTION DU CLIENT

# Architecture de l'application



# Zones de l'interface graphique



Balise Mer



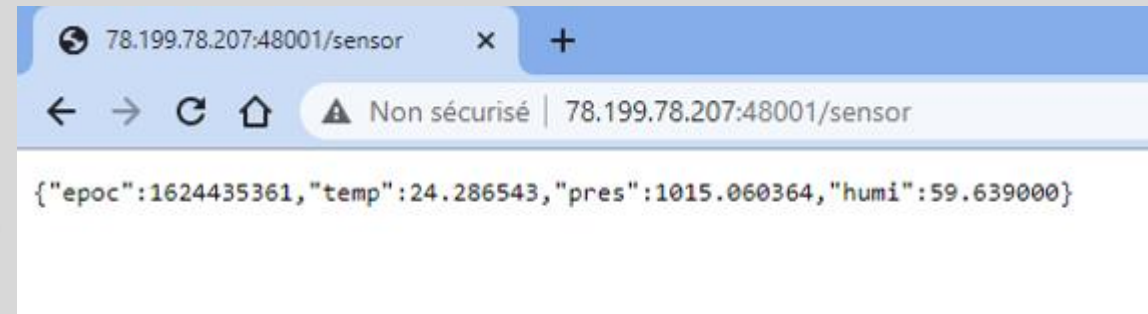
Balise Ville

# Balise Mer : Récupération des mesures

Récupération des paramètres température, pression, humidité

Application Météo

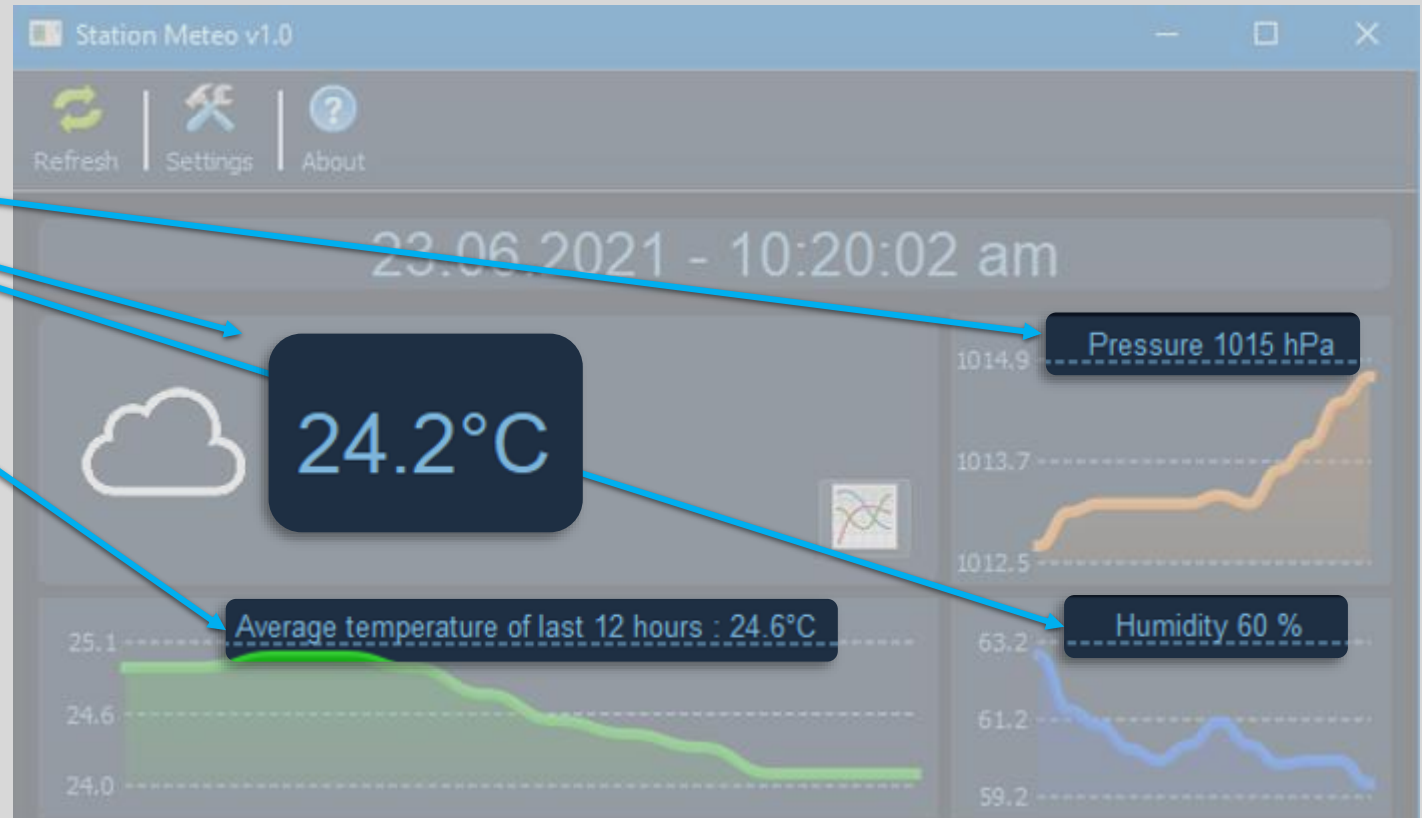
Envoie requête  
serveur



L

# Balise Mer : Récupération des mesures

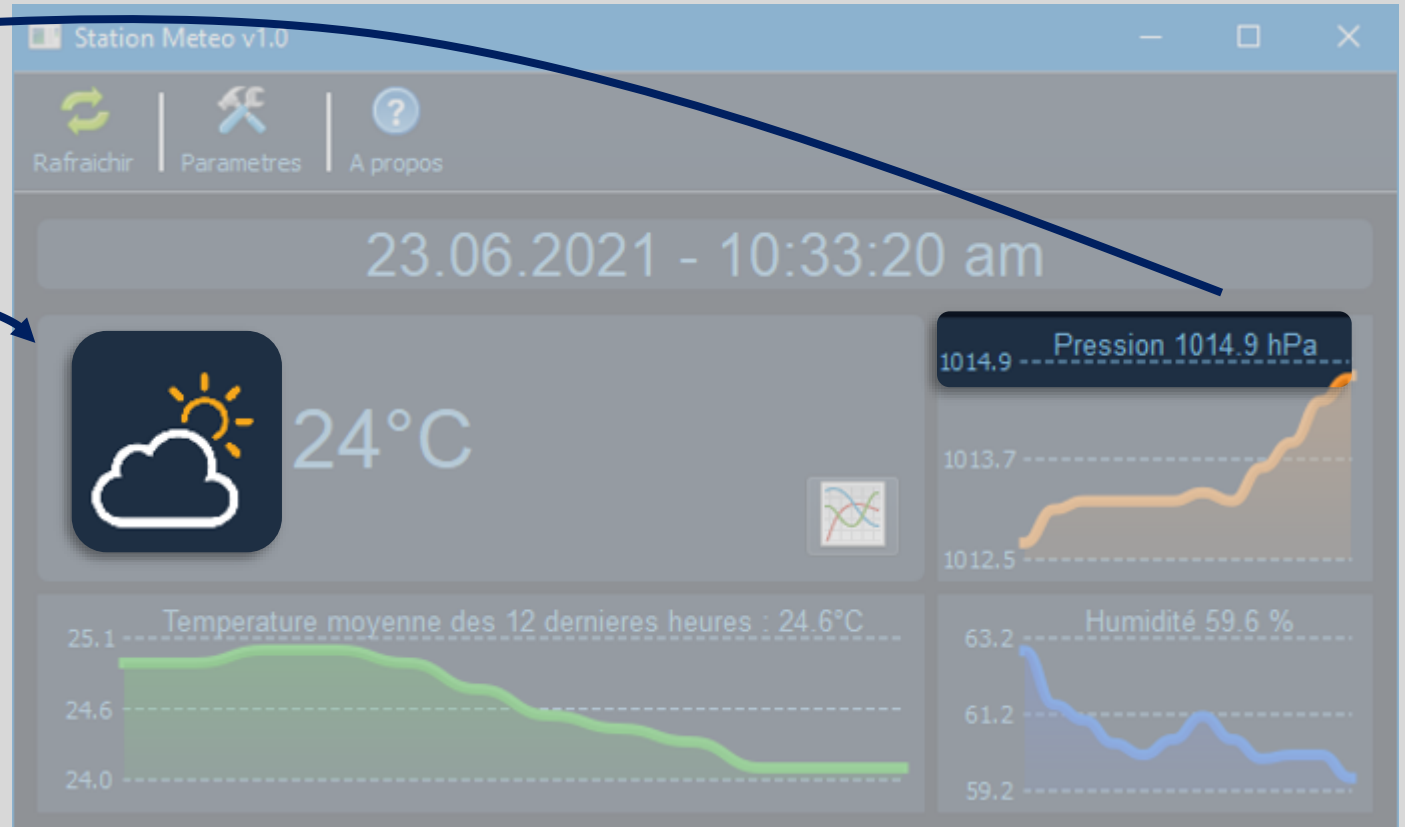
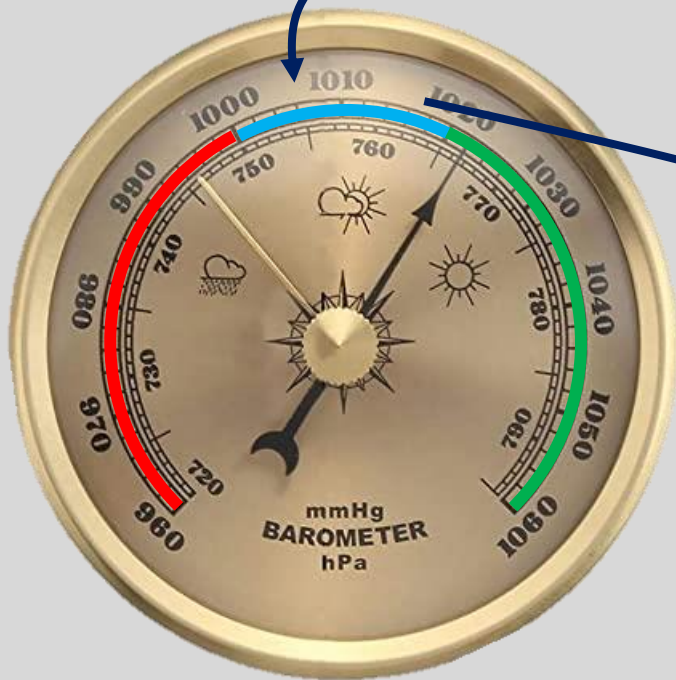
Récupération des paramètres température, pression, humidité



L

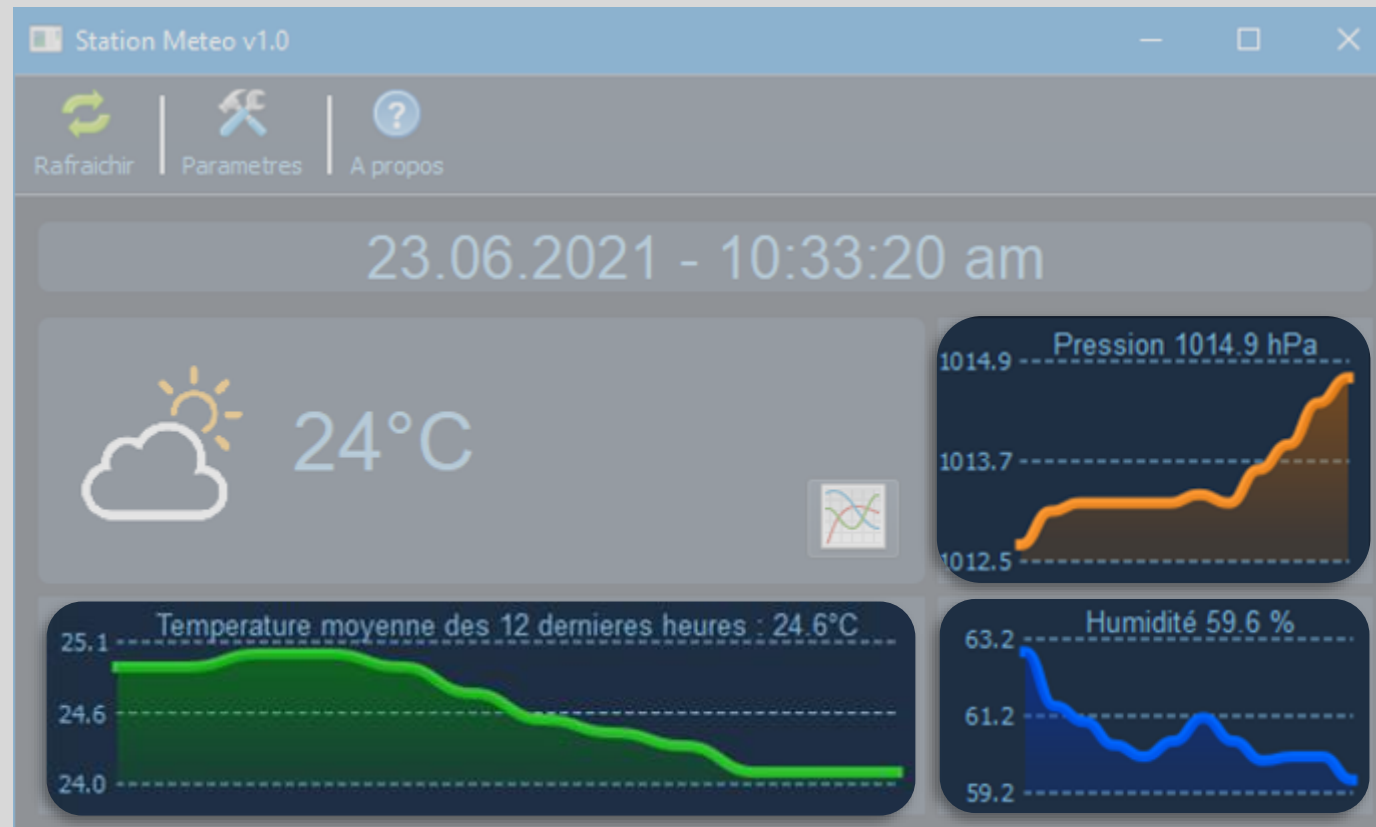
# Balise Mer : Illustration fonction de la pression

Exploitation des données de pression



# Balise Mer : Graph des mesures

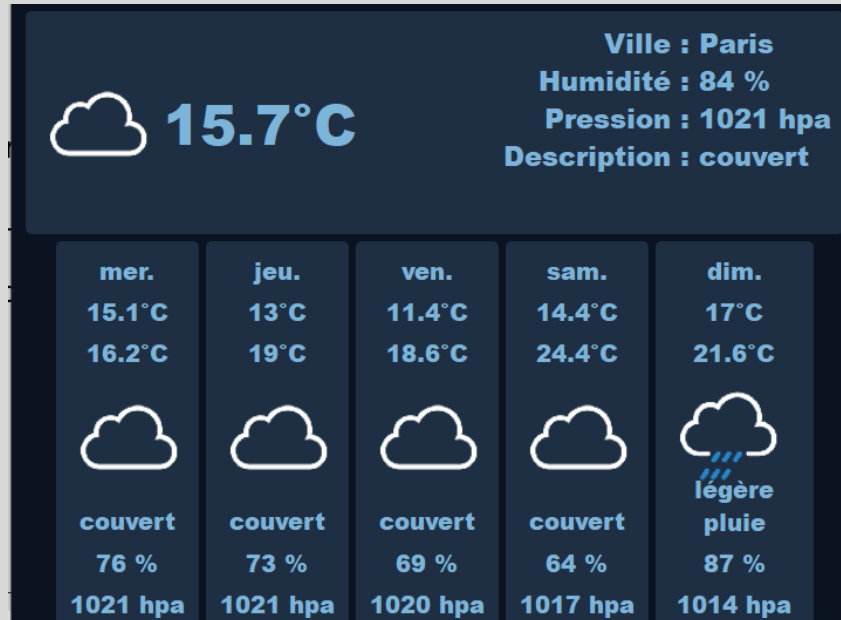
Représentation graphique des données







# Balise Ville : Forecast et mesures instantanées



Mesures instantanées

prévisionnel

Nous utilisons un url par fonction



# Balise Ville : Descriptif url

- url mesures actuelles:

- <https://api.openweathermap.org/data/2.5/weather?q=paris,fr&appid=58e08b52cadfc9c96fc8354666cec712&lang=fr>

- url previsionnel

- <https://api.openweathermap.org/data/2.5/forecast?q=paris,fr&appid=58e08b52cadfc9c96fc8354666cec712&lang=fr>



# Balise Ville : Descriptif fichier json de l'api

JSON	Données brutes	En-têtes
Enregistrer	Copier	Tout réduire Tout dével
coord:	{...}	
weather:		
0:		
id:	804	
main:	"Clouds"	
description:	"couvert"	
icon:	"04d"	
base:	"stations"	
main:		
temp:	288.98	
feels_like:	288.79	
temp_min:	287.74	
temp_max:	290.66	
pressure:	1020	
humidity:	83	
visibility:	10000	
wind:		
speed:	0.89	
deg:	42	
gust:	4.02	
clouds:	{...}	
dt:	1624454739	
sys:	{...}	
timezone:	7200	
id:	2988507	
name:	"Paris"	
cod:	200	


**Ville : Paris**  
**Humidité : 81 %**  
**Pression : 1020 hpa**  
**Description : couvert**








# Balise Ville : changement de ville

Grace au paramètre de changement de ville disponible dans notre url .

Nous avons rajoute une option de changement de ville dans notre menu paramètre qui viendra changer la ville recherche dans l'adresse url afin d'en afficher les mesures,

 **20.7°C**

**Ville : Marseille**  
**Humidité : 62 %**  
**Pression : 1016 hpa**  
**Description : nuageux**

jeu.	ven.	sam.	dim.	lun.
18.3°C	17.9°C	19.5°C	20.6°C	20.7°C
22.7°C	24.8°C	26°C	27°C	25.8°C
				
ciel dégagé	ciel dégagé	peu nuageux	couvert	ciel dégagé
58 %	51 %	48 %	51 %	57 %



Paramètres ? X

Général

Format de l'heure ☐ 24H ☒ 12H

Unité de Temperature ☒ Celsius ☐ Fahrenheit ☐ Kelvin

Police

Styles ☐ Jour ☒ Nuit

Langue

Delai de mise à jour (Min)

Balise Mer

IP

Port


Balise Ville

Ville






Ville

☒ ☐ ☐



 **15°C**

**Ville : Paris**  
**Humidité : 78 %**  
**Pression : 1021 hpa**  
**Description : couvert**

jeu.	ven.	sam.	dim.	lun.
13.1°C	11.7°C	13.8°C	16.1°C	14.7°C
18.4°C	21.6°C	25.4°C	19.4°C	22.5°C
				
couvert	nuageux	couvert	légère pluie	légère pluie
70 %	74 %	74 %	88 %	69 %
1021 hpa	1020 hpa	1015 hpa	1013 hpa	1016 hpa



# Balise Ville : logos/icones



L'api que nous avons choisi nous renvoie des codes icones et des descriptions afin de connaitre le temps qu'il fait durant chacune de ses mesures (couvert ensoleillé...)

Nous avons donc récupérer des icones puis les avons lié à ces codes au niveau du programme afin de le utiliser .

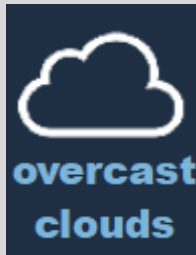




# Balise Ville : changement de langue:

Notre API nous permet de modifier la langue (par défaut anglaise)  
des descriptions meteo reçues,

Nous avons donc utilisé cette fonction dans notre programme  
Cette fonction sera utilise lorsque nous modifierons la langue dans le menu parametre



Paramètres ? X

Général

Format de l'heure ☐ 24H ☒ 12H

Unité de Temperature ☒ Celsius ☐ Fahrenheit ☐ Kelvin

Police Arial Black

Styles ☐ Jour ☒ Nuit

Langue FR

Delai de mise à jour (Min) 10

Balise Mer

IP 78.199.78.207

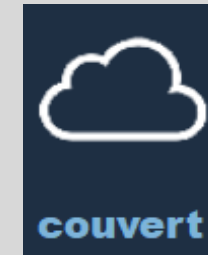
Port 48001

Balise Ville

Clef API 58e08b52cadfc9c96fc8354666cec712

Ville paris

✓ ✗ 💾










# Balise Ville : Affichage jour

Dans les infos renvoyées par l'api nous avons bien sur la date de chaque mesures qui nous est transmise

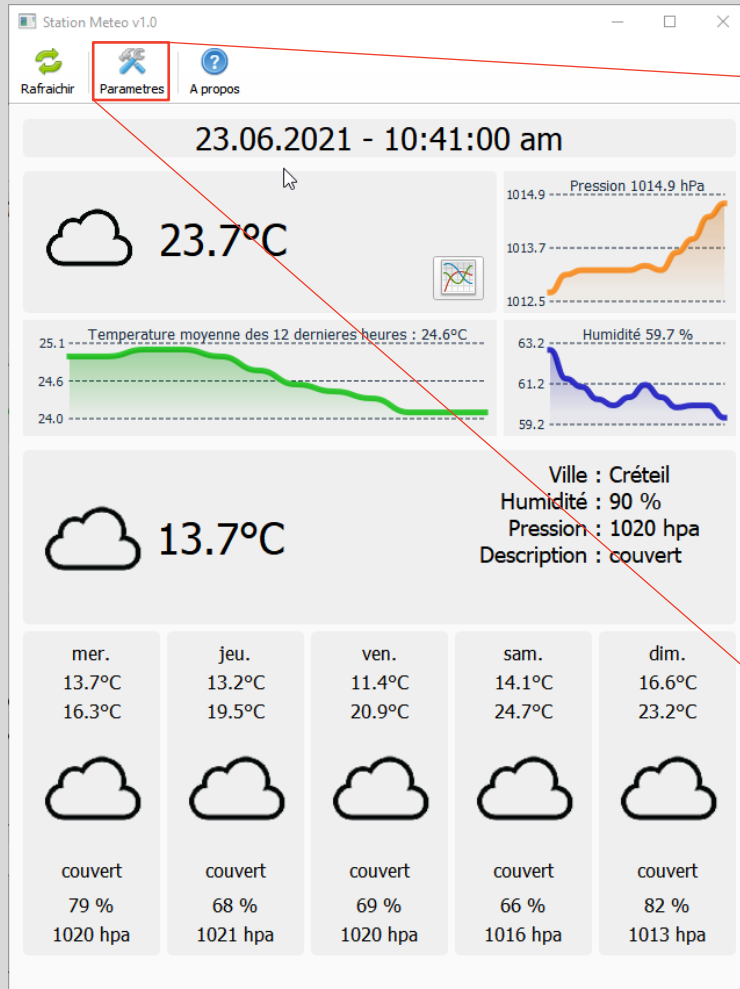
Grace a elles ci et a la fonction Qdate nous avons eu la possibilité d'afficher le jour correspondant et a le traduire selon la langue sélectionnée

Afin d'afficher des mesures les plus précises au niveau des prévisions nous avons optes pour l'affichage de la température minimale et de la maximale plutôt qu'une moyenne journalière

Minimale:  
Maximale:

mer.	jeu.	ven.	sam.	dim.
14.6°C	13°C	11.2°C	14.5°C	16.8°C
16.3°C	19.7°C	20.2°C	24.4°C	23.2°C
				
couvert	couvert	couvert	couvert	couvert
76 %	68 %	71 %	68 %	80 %
1021 hpa	1021 hpa	1020 hpa	1016 hpa	1013 hpa

# Fonctionnalité : Interface D'administration



Paramètres

Général

Format de l'heure ☐ 24H ☒ 12H

Unité de Temperature ☒ Celsius ☐ Fahrenheit ☐ Kelvin

Police MS Shell Dlg 2

Styles ☒ Jour ☐ Nuit

Langue FR

Delai de mise à jour (Min) 10

Balise Mer

IP 78.199.78.207

Port 48001

Balise Ville

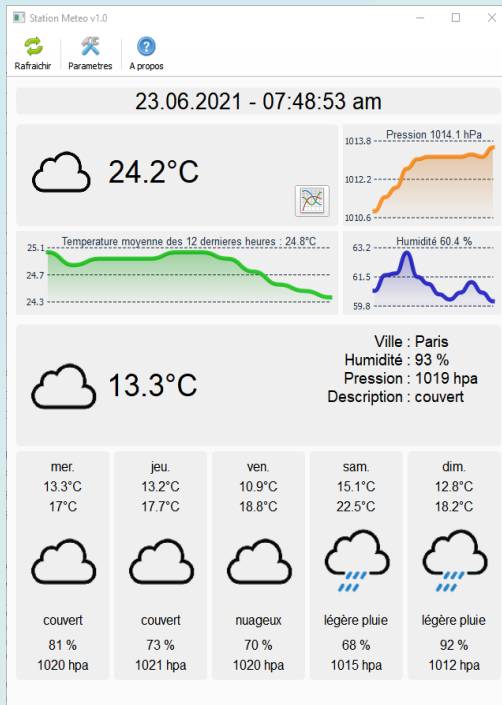
Clef API 467fdde738c8df563afa33a978b53563

Ville Creteil

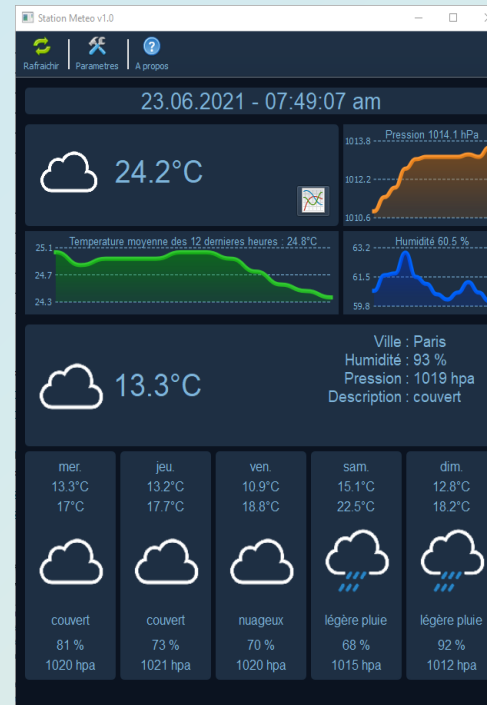


# General : changement de thème Jour / Nuit

## Thème Jour



## Thème Nuit



## Configuration par fichier de style QSS

```

QMainWindow { background-color: #0B1320; }
QDialog { background-color: #0B1320; }
QToolBar { background-color: #1E2F43; }
QPushButton { color: #7DB4DA; }
QPushButton:hover { background-color: #1F4083; }
QPushButton {
    background-color: #1E2F43;
    color: #7DB4DA;
}
QPushButton:hover
{
    background-color: #1F4083;
    color: #E0E0E0;
}
QLabel{ color: #7DB4DA; }
QRadioButton
{
    color: #7DB4DA;
    padding: 0 5 0 0;
}
QRadioButton::indicator:checked
{
    height: 10px;
    width: 10px;
    border-style: solid;
    border-radius: 5px;
    border-width: 2px;
    border-color: #7DB4DA;
    background-color: #7DB4DA;
}
QRadioButton::indicator:!checked
{
    height: 10px;
    width: 10px;
    border-style: solid;
  
```

# General : Chargement / Sauvegarde des paramètres

## Chargement depuis un fichier INI

```
void GlobalSettings::SettingsStruct::Load(const QString& vFilePathName)
{
    GlobalSettings::SettingsStruct _default;

    QSettings config(vFilePathName, QSettings::IniFormat);

    m_FormatHourEnum = (FormatHourEnum)config.value("FormatHourEnum", (int)_default.m_FormatHourEnum).toInt();
    m_TemperatureUnit = (TemperatureUnitEnum)config.value("TemperatureUnit", (int)_default.m_TemperatureUnit).toInt();
    m_Ville = config.value("Ville", _default.m_Ville).toString();
    m_FontFamily = config.value("FontFamily", _default.m_FontFamily).toString();
    m_ApiKey = config.value("ApiKey", _default.m_ApiKey).toString();
    m_IP = config.value("IP", _default.m_IP).toString();
    m_Port = config.value("Port", _default.m_Port).toString();
    m_Language = config.value("Language", _default.m_Language).toString();
    m_Style = (StyleEnum)config.value("Style", (int)_default.m_Style).toInt();
    m_RefreshDelayInMinutes = config.value("RefreshDelayInMinutes", _default.m_RefreshDelayInMinutes).toUInt();
}
```

## Sauvegarde dans un fichier INI

```
void GlobalSettings::SettingsStruct::Save(const QString& vFilePathName)
{
    QSettings config(vFilePathName, QSettings::IniFormat);

    config.setValue("FormatHourEnum", (int)m_FormatHourEnum);
    config.setValue("TemperatureUnit", (int)m_TemperatureUnit);
    config.setValue("Ville", m_Ville);
    config.setValue("FontFamily", m_FontFamily);
    config.setValue("ApiKey", m_ApiKey);
    config.setValue("IP", m_IP);
    config.setValue("Port", m_Port);
    config.setValue("Language", m_Language);
    config.setValue("Style", (int)m_Style);
    config.setValue("RefreshDelayInMinutes", m_RefreshDelayInMinutes);

    config.sync();
}
```

## Le contenu du fichier INI

```
[General]
FormatHourEnum=0
TemperatureUnit=0
Ville=marseille
FontFamily=MS Shell Dlg 2
ApiKey=467fdde738c8df563afa33a978b53563
IP=78.199.78.207
Port=48001
Language=FR
Style=0
RefreshDelayInMinutes=10
```

# General : Gestion Multilingue

Traduction de l'application

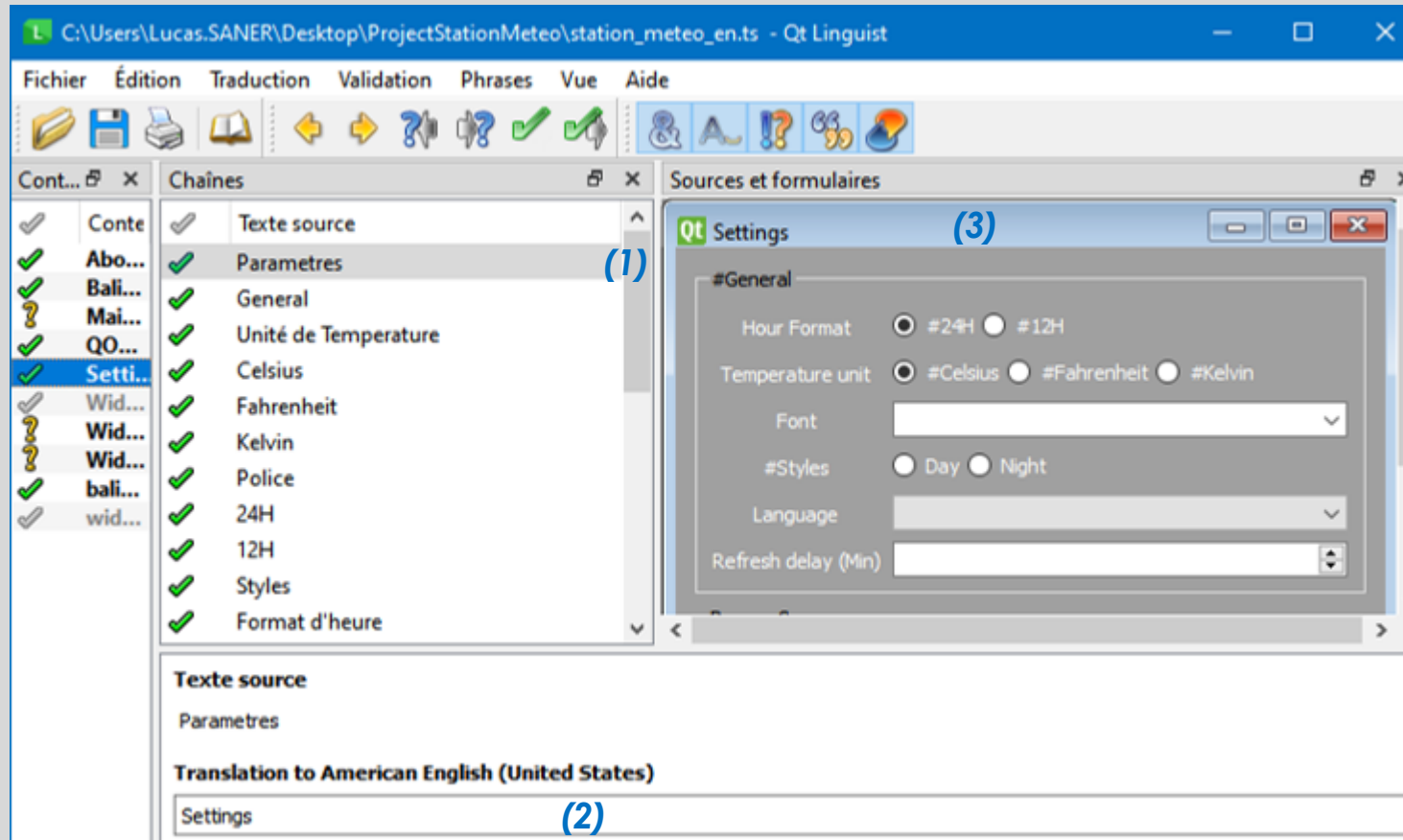
```
ui->customGraphPressure->SetSerieName("Pression ");
```



```
ui->customGraphPressure->SetSerieName(QObject::tr("Pression "));
```

# General : Gestion Multilingue

Traduction de l'application



# General : Gestion Multilingue

Traduction de l'application

Paramètres ? X

Général

Format de l'heure ☐ 24H ☒ 12H

Unité de Temperature ☒ Celsius ☐ Fahrenheit ☐ Kelvin

Police Arial

Styles ☐ Jour ☒ Nuit

Langue EN

Delai de mise à jour (Min) 10

Balise Mer

IP 78.199.78.207

Port 48001

Balise Ville

Clef API 58e08b52cadfc9c96fc8354666cec712

Ville Paris

✓ ✗ 💾



Settings ? X

General

Hour Format ☐ 24H ☒ 12H

Temperature unit ☒ Celsius ☐ Fahrenheit ☐ Kelvin

Font Arial

Styles ☐ Day ☒ Night

Language EN

Refresh delay (Min) 10

Beacon Sea

IP 78.199.78.207

Port 48001

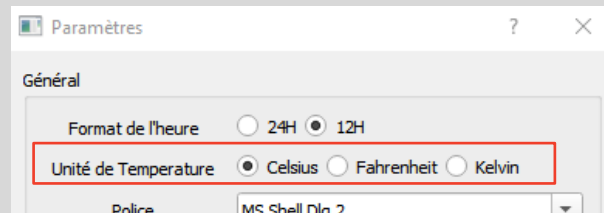
Beacon City

Api Key 58e08b52cadfc9c96fc8354666cec712

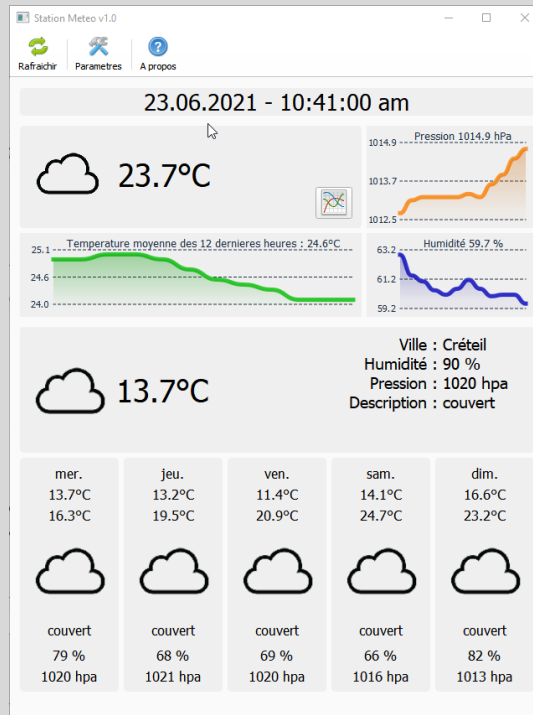
City Paris

✓ ✗ 💾

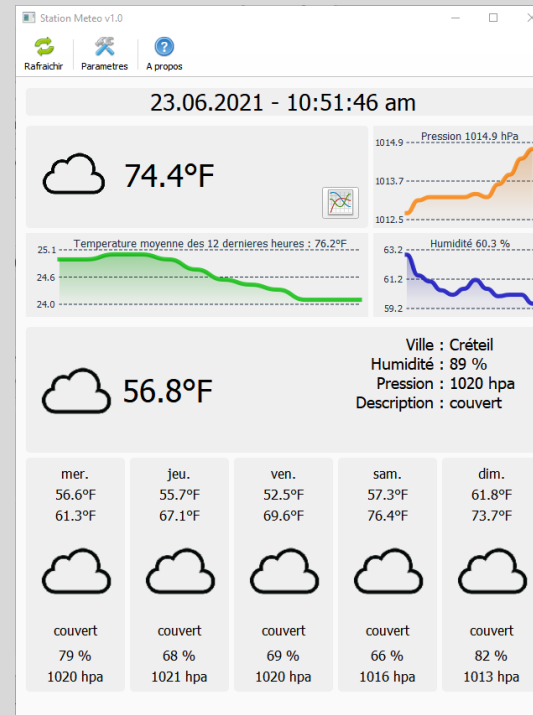
# General : Changement de l'unité des mesures



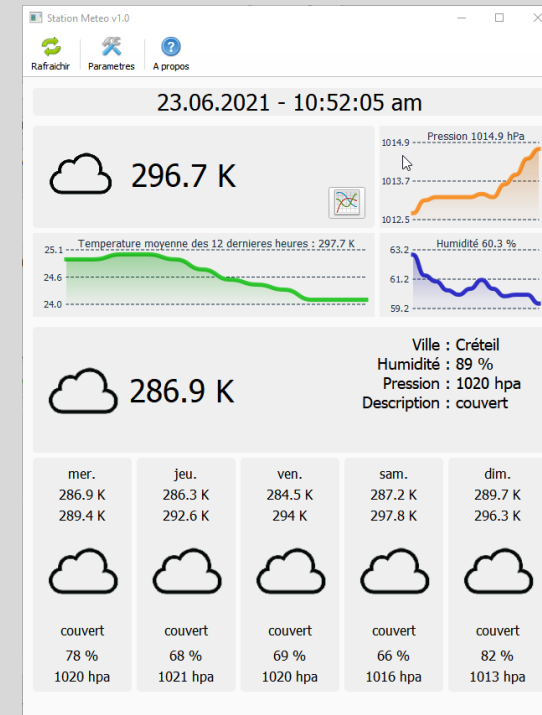
Affichage en Celsius  
(default)



Affichage en Fahrenheit

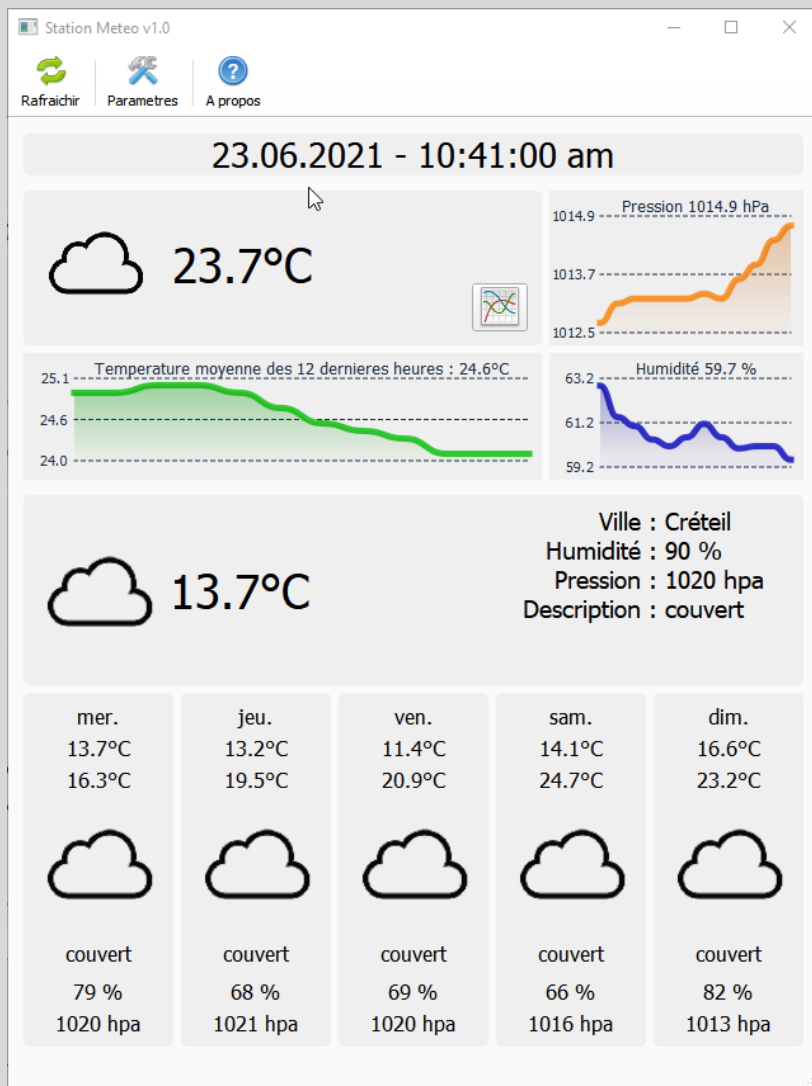


Affichage en Kelvin





# General : changement de police



Sélection de la police Mistral

Parametres

Général

Format de l'heure ☐ 24h ☒ 12h

Unité de Temperature ☒ Celsius ☐ Fahrenheit ☐ Kelvin

Police **Mistral**

Styles ☒ Jour ☐ Nuit

Langue FR

Delai de mise à jour (Min) 10

Balise Mer

IP 78.199.78.207

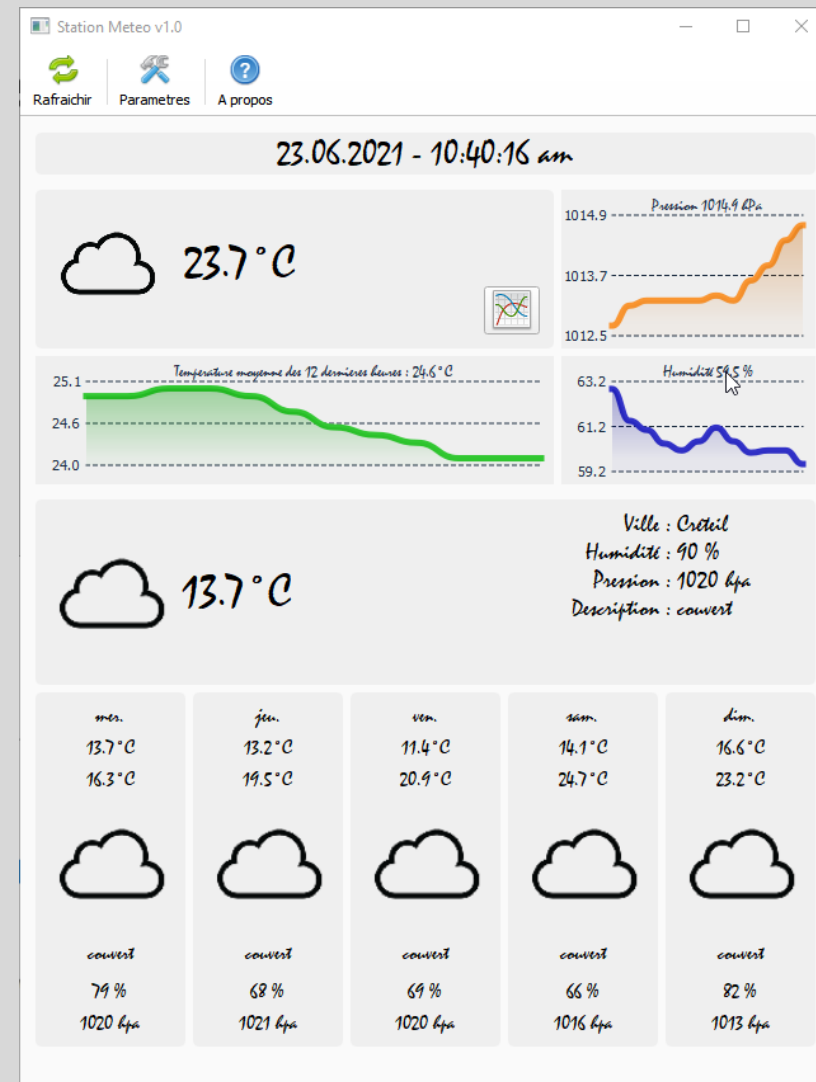
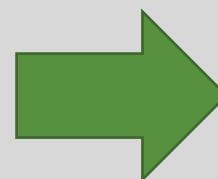
Port 48001

Balise Ville

Clef API 467fdde738cdf563afa33a978b53563

Ville Créteil

✓ ✗ 💾



24/06/2021

# CONCLUSION

PROBLEMES RENCONTRES / EVOLUTIONS POSSIBLES / APPORTS PERSONNELS



# Conclusion

- Evolutions possible :
  - Convertir les requêtes serveur en asynchrone
  - Rajouter d'autres langues
  - pouvoir proposer a la lecture + de paramètres sur l'api web
  - Proposer un graphique prévisionnel des 5 jours
  - Pouvoir se connecter a plus d'un ip dans le cas de la balise mer
  - Customisation du thème par l'utilisateur

# Remerciements

24/06/2021

# DEMONSTRATION

DEMO DU LOGICIEL FINI