



Projet Raleyd

-

Conception d'une application web

Présenté par l'équipe Raleyd

Loyd Simon
Jean Rafael Mendes
Léopold Ardouin

Sommaire

- I - Introduction
- II - Analyse fonctionnelle
- III - Gestion de projet
- IV - Technologie utilisés
- V - Concevoir : Base de données
- VI - Concevoir : Serveur web
- VII - Concevoir : API
- VIII - Concevoir : Interface web
- IX - Axe d'amélioration
- X - Conclusion

I - Introduction

La société "Lepetit" ayant racheté la startup "Atmos" qui commercialise des objets connectés pour la maison, envisage la modernisation d'un produit "phare" de la startup "Atmos".

cependant, depuis le rachat, les stations météorologiques connectées ne font que :

- perdre en popularité
- Coûte cher

Notre groupe vient d'être recruté dans cette importante compagnie afin de réaliser un nouveau **système de collecte de données météorologiques** et une maquette fonctionnelle de **l'interface web de visualisation des données**. Le groupe est composé de trois étudiants :

Loyd Simon, le chef de projet

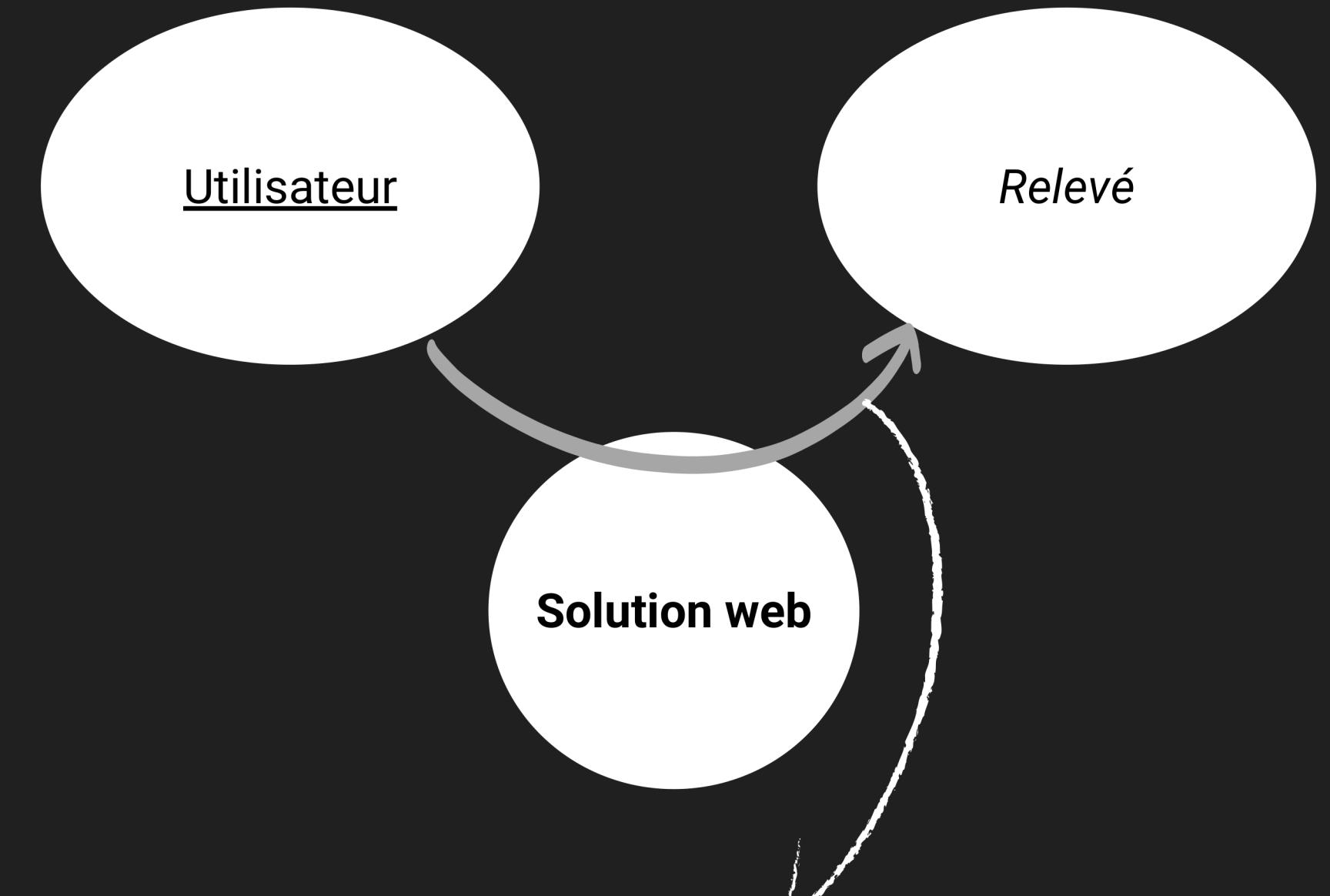
Jean Rafael Mendes

Léopold Ardouin

II - Analyse fonctionnelle

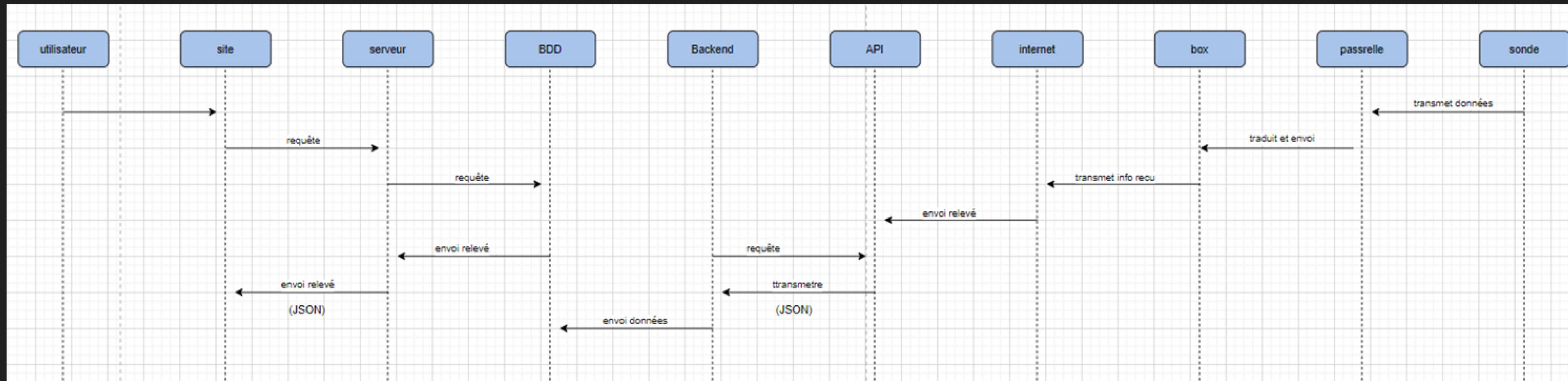
"Bête à corne"

La **solution web** rend service à l'utilisateur en agissant sur le *relevé* en satisfaisant sa demande d'accès.



La solution web permet d'accéder au relevé de température

Diagramme de séquence



donc



Récolte de données



mis à jours des relevés

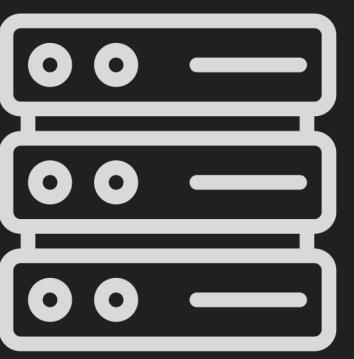


Interface web

Backend / Frontend



une BDD



Serveur



API

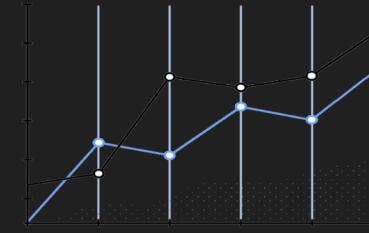


Interface

Fonctionnalités du site



Modification des
capteurs



Graphique



Carte



Batterie



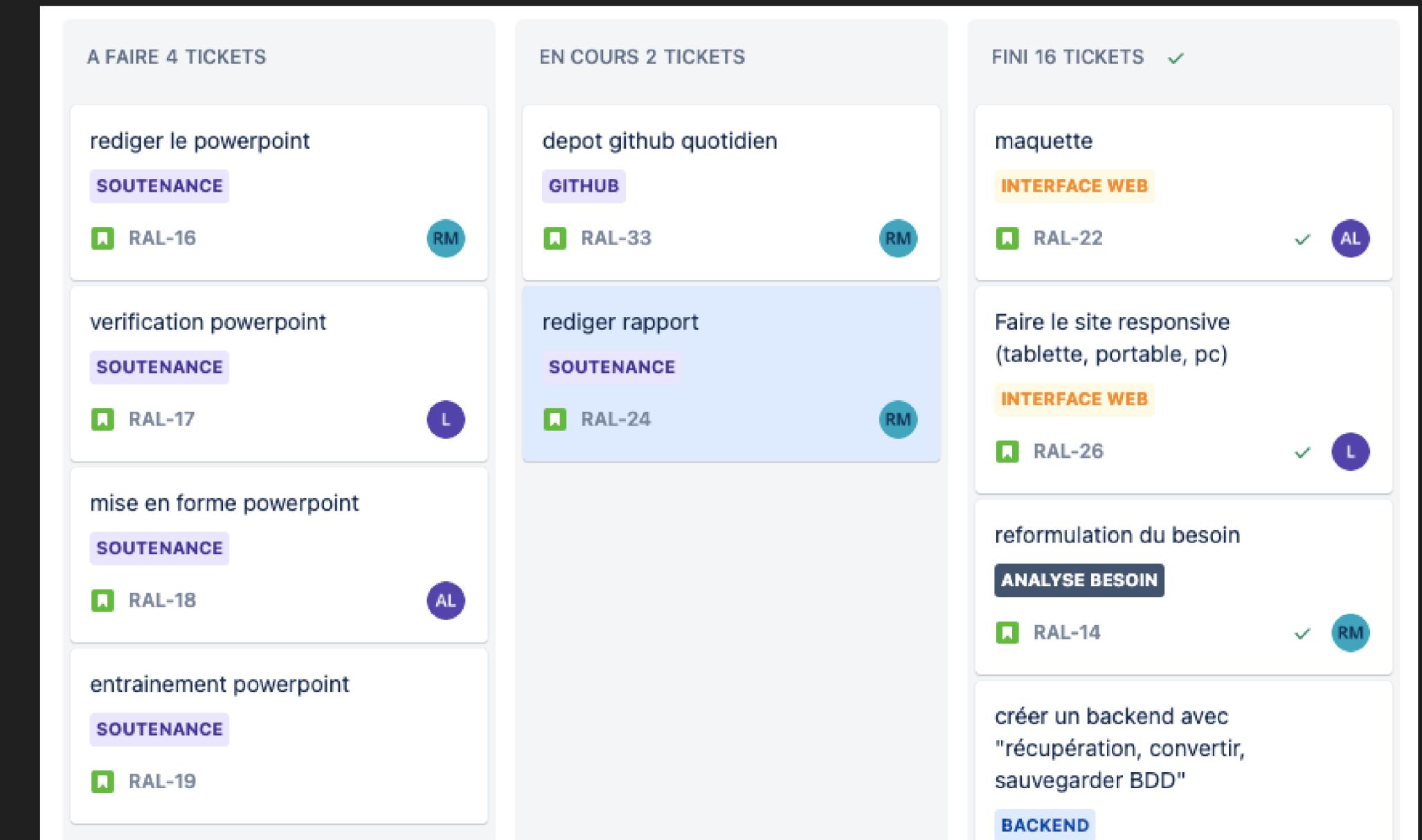
Historique

III - Gestion projet

Notre organisation avec
Jira :

Mendes Rafael : création GitHub,
rédaction du rapport

Ardouin Léopold : création des
maquettes, rédaction du
powerpoint



Simon Loyd: backend, Front-end

GitHub

Création d'un GitHub :

- Suivi du projet.
- Travail collaboratif.
- Versionnning.

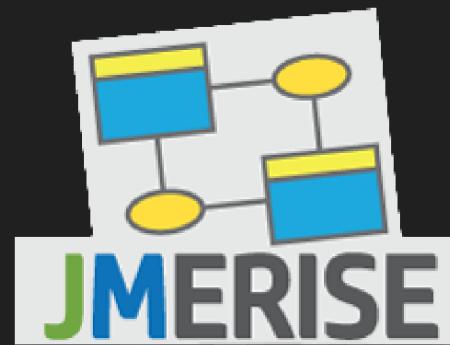
The screenshot shows a GitHub repository page for 'Rafaelllllll / ProjetMeteo' (Public). The main navigation bar includes links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, and Insights. Below the navigation is a summary bar showing 'main' branch, '3 branches', '0 tags', and buttons for 'Go to file', 'Add file', and 'Code'. The main content area displays a list of commits from 'loyds1' (commit html-css). The commits are as follows:

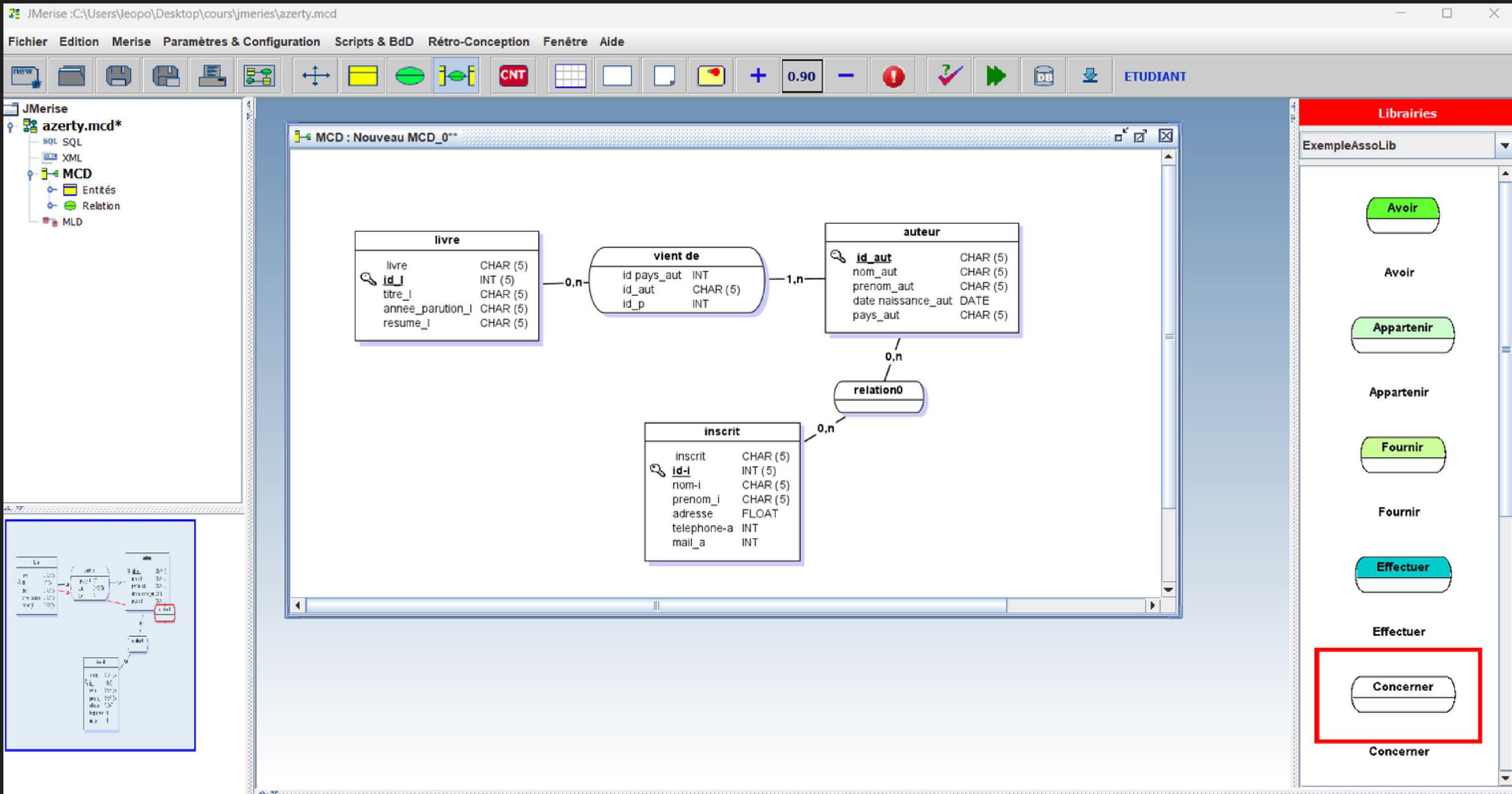
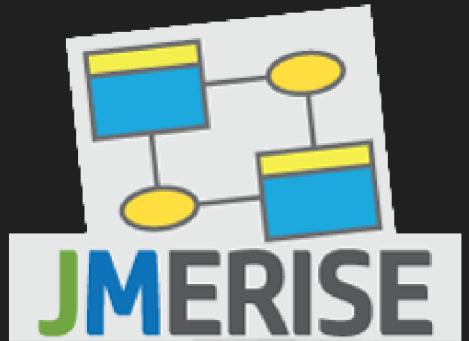
Commit	Message	Date	Actions
27877df	commit html-css	2 days ago	6 commits
	.idea	Premier dépôt	9 days ago
	Templates	html-css	2 days ago
	static	html-css	2 days ago
	.DS_Store	html-css	2 days ago
	Template_Specifications_Fonction...	Add files via upload	8 days ago
	backend.py	quatrieme commit	8 days ago
	backendApi.py	commit html-css	2 days ago
	main.py	html-css	2 days ago

Notre lien : <https://github.com/Rafaelllllll/ProjetMeteo>

IV - Technologie utilisées

Nos outils/logiciels pour concevoir.





Merise :

- Méthode d'analyse.
- Conception et gestion de projet informatique.

JMerise :

- Logiciel pour BDD.

Le SQL (Structured Query Language) :

- Langage de communication avec une base de données.
- Utilisation pour création des données du site web.





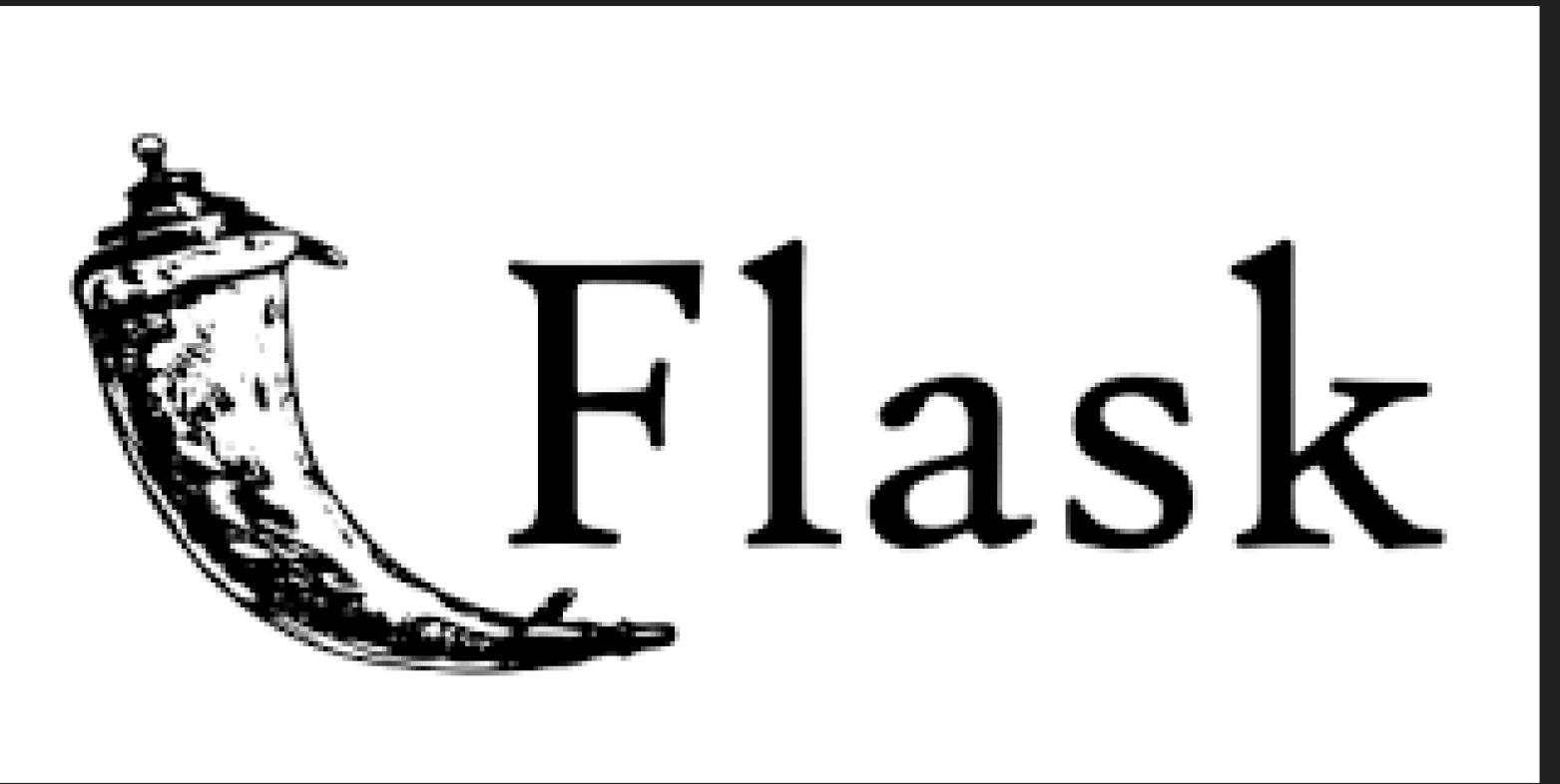
Screenshot of a Python development environment (VS Code) showing a project named "pythonProject". The code editor displays a file named "tp7.py" which contains the following Python script:

```
File Edit View Navigate Code Refactor Run Tools Git Window Help pythonProject - tp7.py
pythonProject > tp7.py
Project Commit Pull Requests Bookmarks Structure
pythonProject C:\Users\leopo\...
  liens parent
    lien
      sous lien
        nuage_atmos.png
        page 1 acceuil.html
        page 2.html
        style.css
        style.css.html
    ProjetMeteo
    Templates
    venv library root
      bloc2.py
      exe.py
      tp.py
      tp2.py
      tp3.py
      tp4.py
      tp5.py
      tp6.py
      tp7.py
      tp8.py
External Libraries
Scratches and Consoles
  dashboard.html x  page 1 acceuil.html x  tp8.py x  tp7.py x  style.css x  page 3 carte.html x  page 2.html x
1 chaine ="00101001CP10020XXCP20021XXETC"
2
3
4 #print (chaine.find("CP1"))
5 # ecrit la chaine + recherche CP1 pour avoir le positionnement
6 temp = chaine.find("CP1")+ 5
7 # cherche (degré) la position du 5 eme emplacement sur la chaine depuis la position de CP1
8 t=chaine_[temp: temp+2]
9 #print (temp)
10 # sur la commande au dessus on vient chercher la position de 20 degré mais on obtien que le
11 # ecrit le positionnement de la température trouvé
12 print_(la température du capteur 1 est de " + t + "°C")
13
14 #print (chaine.find("CP2"))
15 # la même chose qu'au dessus mais en changeant les variables pour pas qu'elles soient
16 temp2 = chaine.find("CP2")+ 5
17 t2=chaine_[temp2: temp2+2]
18 #print (temp2)
19 print_(la température du capteur 2 est de " + t2 + "°C")
20
21 if chaine.find("CP3"):
22     print_(Ooh yeah)
23
24 chaine ="00101001CP10020XXCP20021XXETCCP30025" # essaie avec Loyd
25
26 for i in range_(len(chaine)):
27     if chaine[i:i+2] == "CP":
28         tempA = chaine.find("CP") + 5
29         tA = chaine[tempA: tempA + 2]
30         print_(tA)
31
32 chaine ="00101001CP10020XXCP20021XXETCCP30025"
33
34 capteurs=["CP1", "CP2", "CP3"] # ca fonctionne en boucle
```

The code uses string manipulation to find specific substrings ("CP1", "CP2", "CP3") within a large binary-like string and prints their positions. It also demonstrates a loop that iterates through the string to find all occurrences of "CP".

- Un environnement de développement intégré.
- Adapté au langage Python.

- Framework open-source de développement web en Python.
- Nombreuses extensions permettant d'ajouter facilement des fonctionnalités.





Le HTML « HyperText
Markup Language »
nous a permis de constituer
notre page web et
de la structurer.

```
dashboard.html x style.css x page 1 acceuil.html x page 3 carte.html x page 2.html x
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8">
5     <meta http-equiv="X-UA-Compatible" content="IE=edge">
6     <meta name="viewport" content="width=device-width, initial-scale=1.0">
7     <title>Mon Site</title>
8   </head>
9   <body>
10    <p>relevé
11    </p>
12    {% for resultat in resultats %}
13      <p>L'id du capteur est : {{ resultat[0] }}</p>
14      <p>La température est de : {{ resultat[1] }}</p>
15      <p>L'humidité est de : {{ resultat[2] }}</p>
16      <p>Relevé effectué le :{{ resultat[3] }}</p>
17    {% endfor %}
18   </body>
19 </html>
```

HTML

```
dashboard.html x style.css x page 1 acceuil.html x page 3 carte.html x page 2.html x
1 body{
2   background:url("nuage_atmos.png") fixed no-repeat top center;
3 }
4 p{
5   color:black;
6   border-style:solid; /*bordure*/
7   border-color:orange;
8   border-radius:15px; /*arrondi de l'angle*/
9   padding-left:20px;
10  /*a 1h06 de la deuxième video y'a le truck pour defiler la page en gardent la l
11  */
12}
13
14}
```

CSS

Le CSS « Cascading StyleSheets » permet
de mettre en forme notre site et de
pouvoir le mettre en responsive.

V - Concevoir : Base de données

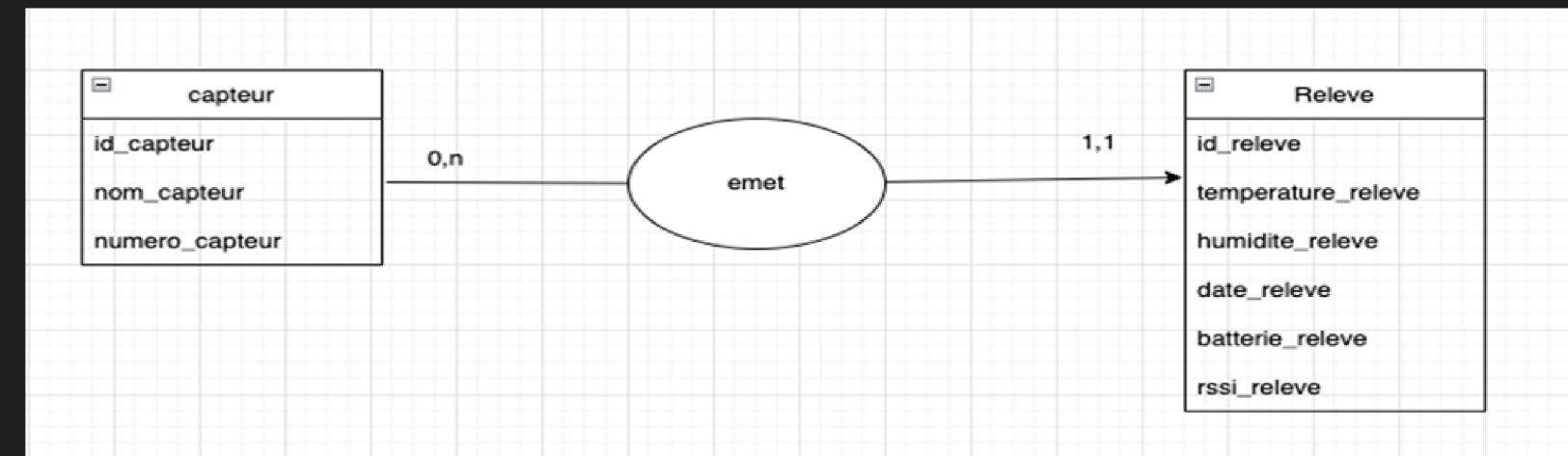
Le Modèle Conceptuel de Données sert à conceptualiser l'application. Il met en évidence deux éléments :

- Les entités.
- Les associations.

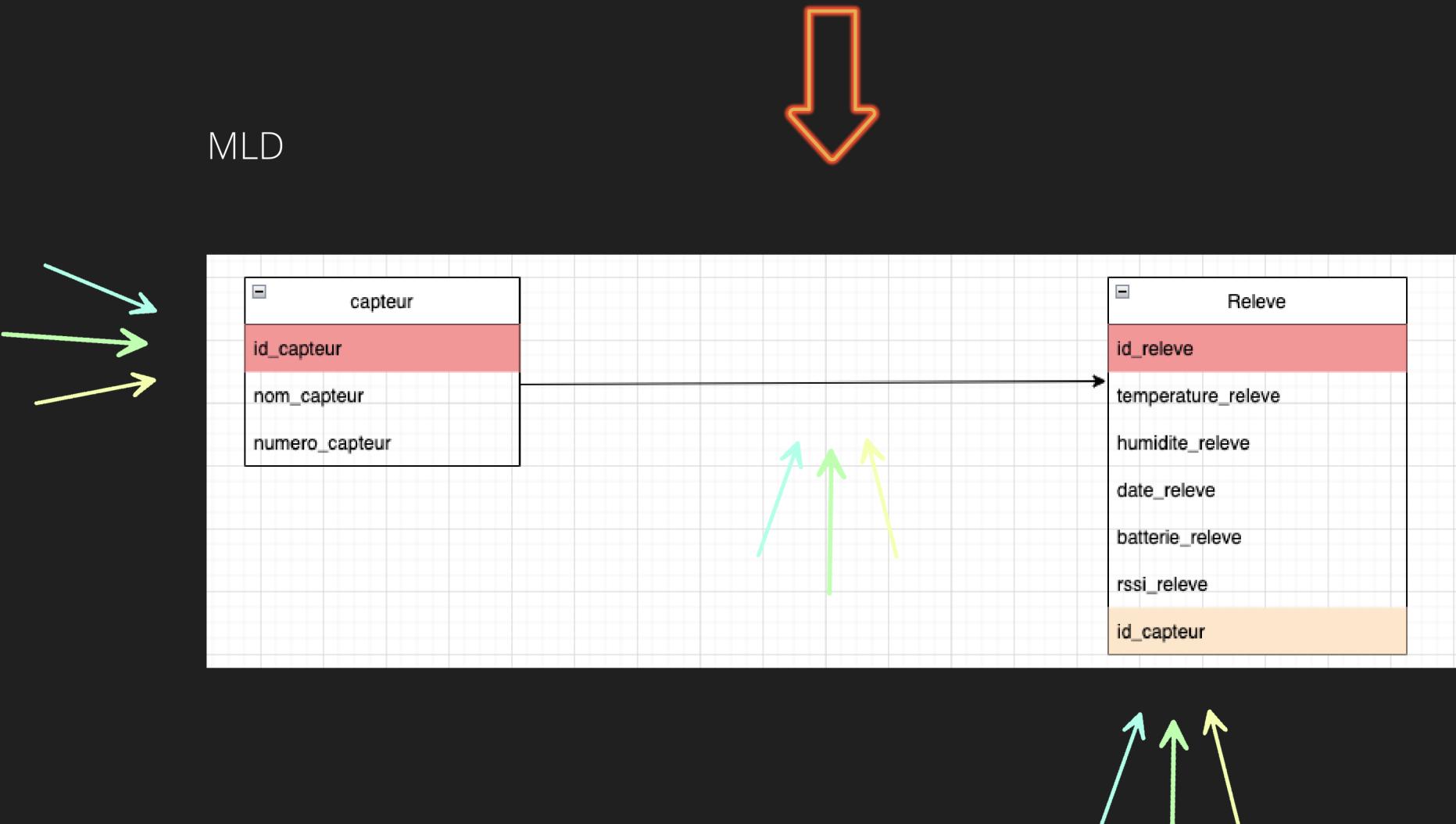
Le Modèle Logique de Données prend en compte la nature de l'outil logiciel avec lequel sera implanté la future base de données

- Les entités en tables.
- Les associations en clés étrangères/tables de jonction.

MCD



MLD



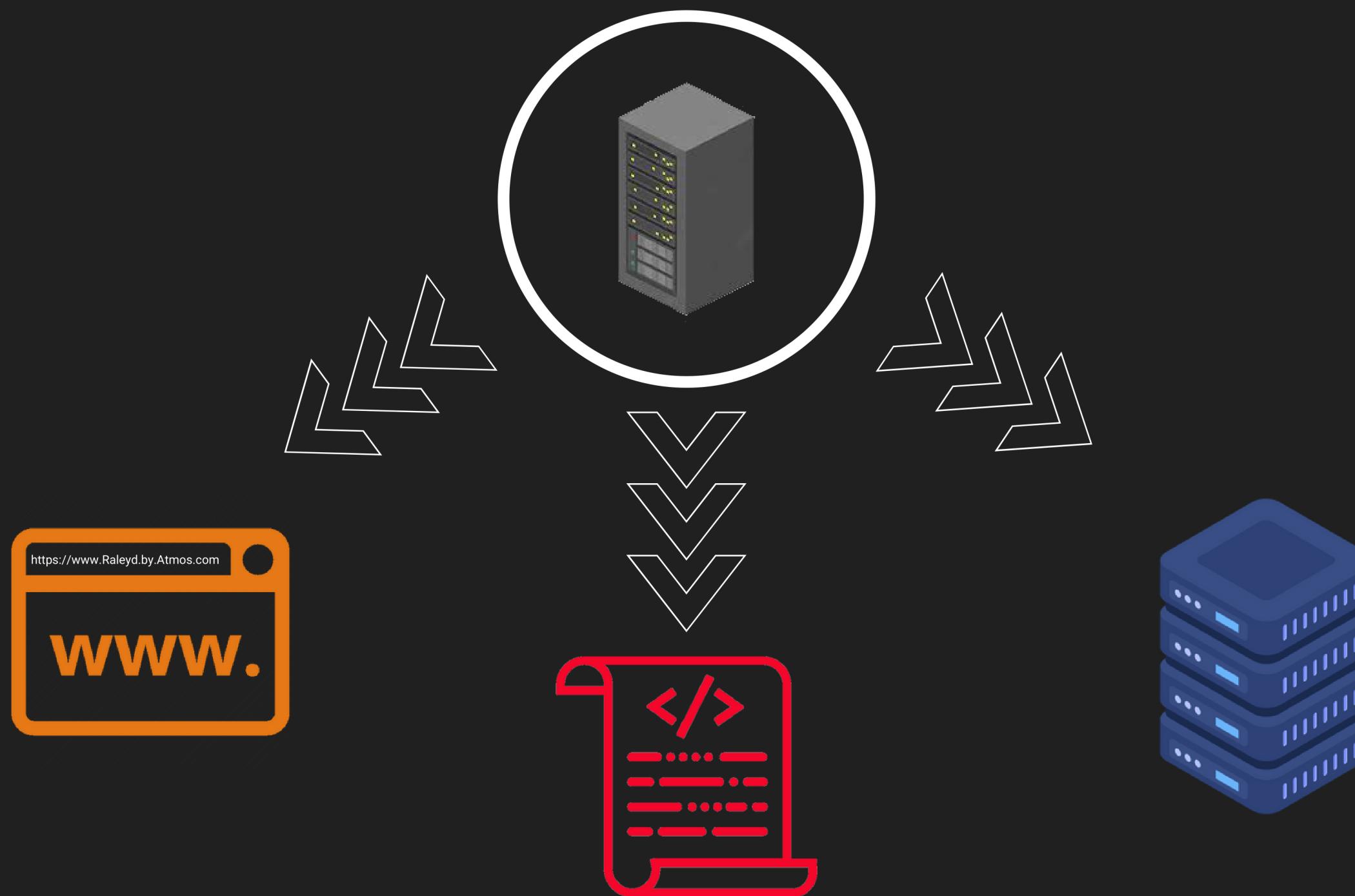
Nous allons ensuite passer le MLD en MPD ou Modèle Physique des Données.

Le MPD prend en compte la typologie concrète de la base de données.
Il est la transcription informatique du modèle originel (MCD)

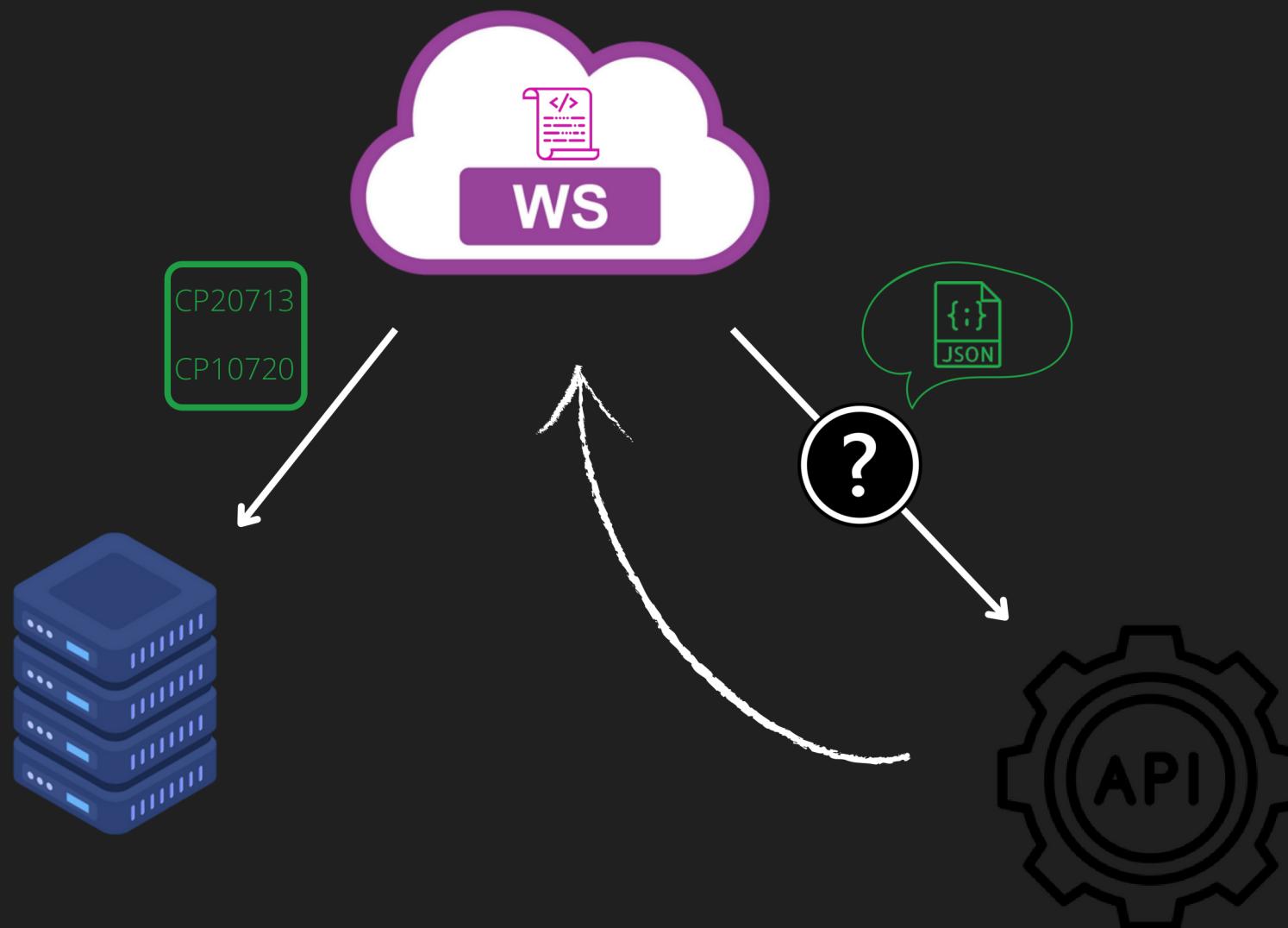


VI - Concevoir : Serveur web

Le système sera déployé sur un serveur, ce qui permettra de d'héberger un site web, une base de données et les scripts nécessaires au système.



VII - Concevoir : API



Via un script, l'**API** va :

- Recuperer les données de l'API_Artigala.
- Convertir les données.
- Sauvegarder dans la BDD.

Via un autre script, l'API va :



Pouvoir communiquer entre la base de données et le site web .

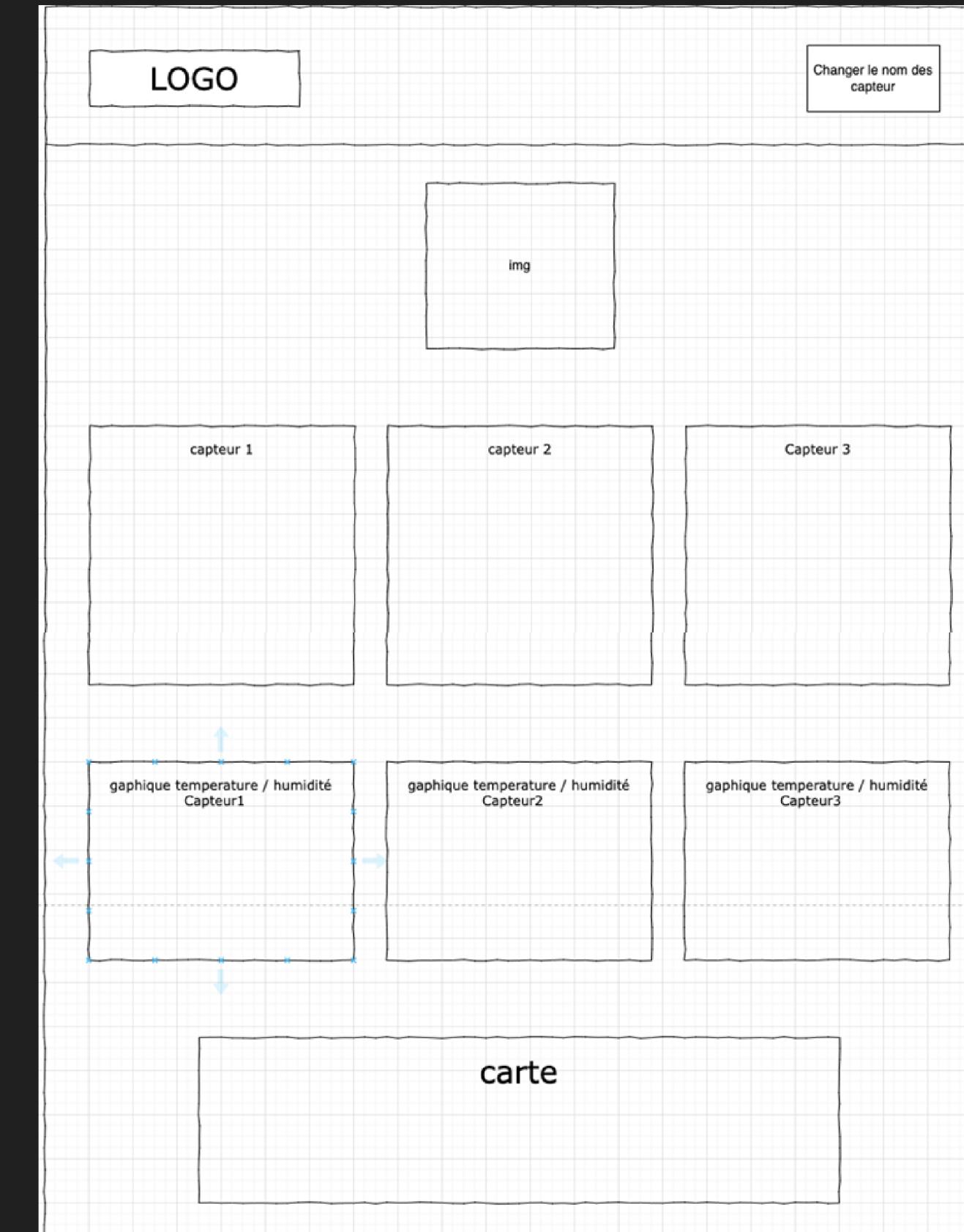
- Insert
- Select
- Delete
- Update

```
import pymysql.cursors\n\n# Loyd Simon *\n\ndef selectreleve():\n    # Connexion à la base de données\n    con = pymysql.connect(host='localhost', user="root", password="root", database='bloc2', port=8889)\n    cur = con.cursor()\n    mreq = "SELECT * FROM Capteur JOIN Releve ON Capteur.ID_C = Releve.ID_C WHERE Capteur.ID_C ="\n    cur.execute(mreq)\n    result = cur.fetchall()\n    return result\n\n# Loyd Simon *\n\ndef select():\n    con = pymysql.connect(host='localhost', user="root", password="root", database='bloc2', port=8889)\n    cur = con.cursor()\n    mreq = "SELECT * FROM `Capteur` "\n    cur.execute(mreq)\n    result = cur.fetchall()
```

VIII - Concevoir : Interface web

Wireframe c'est:

- Visualiser l'agencement de la page.
- L'architecture de l'information.
- Les parcours utilisateurs.
- Les fonctionnalités essentielles.

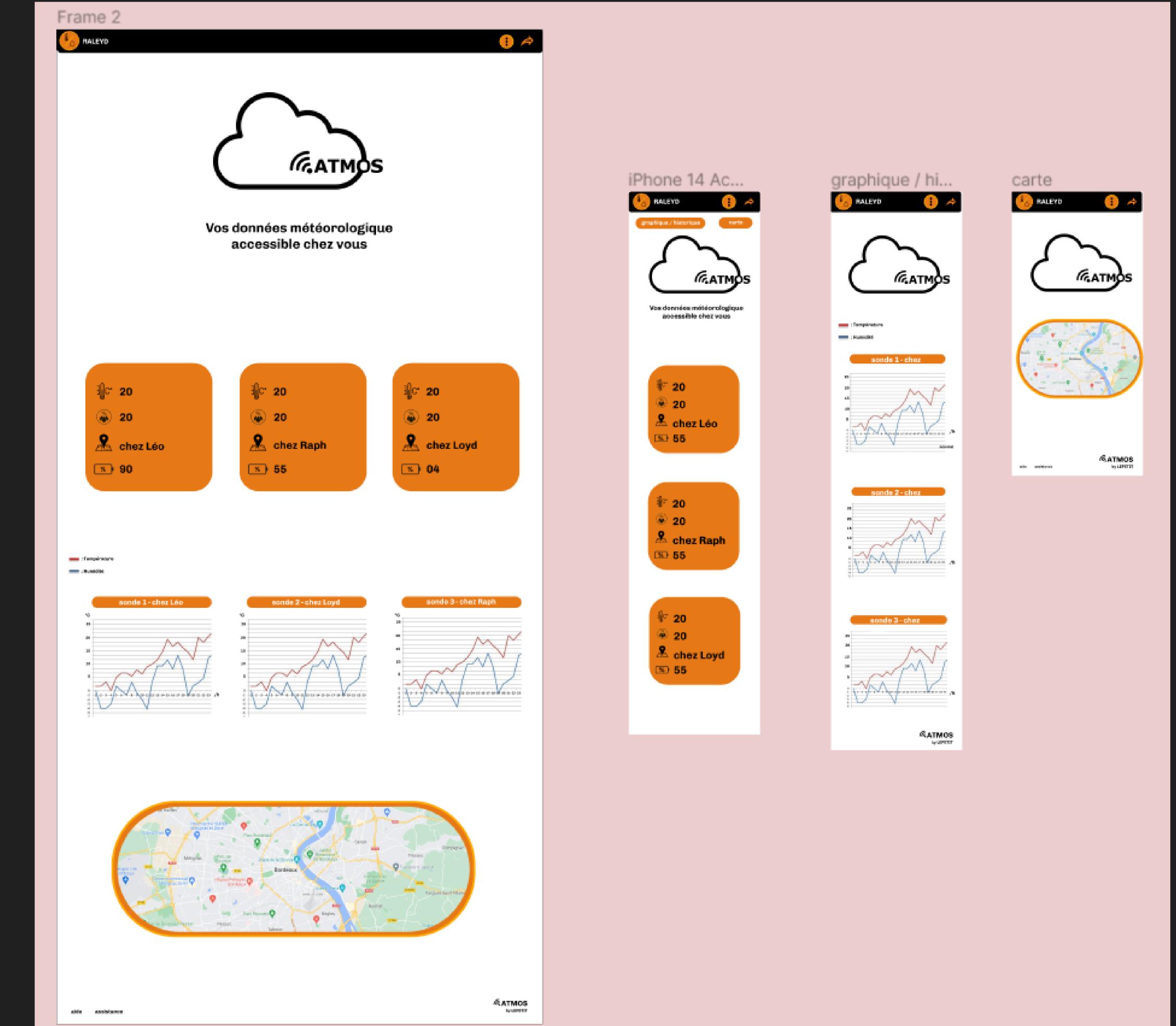


La maquette

La maquette nous permet de faire la conception d'une interface conformes à vos attentes.

Figma:

- Un éditeur graphique.
- Un outil de prototypage.
- Collaboration d'équipe en temps réel.
- Utilisable comme support de présentation avec les intervenants du projet.



interface responsive



Il est primordial d'avoir une interface responsive, c'est ce que l'on appelle Web réactive.

Il offre une consultation confortable sur des écrans de tailles très différentes.

IX - Axes d'améliorations



X - Conclusion



stable



responsive



rentable



open source

Merci 
de votre attention

L'ÉQUIPE RALEYD RESTE DISPONIBLE POUR RÉPONDRE À VOS
QUESTIONS.