

CONCEPTION ET
DÉVELOPPEMENT D'UNE
APPLICATION DE
DOMOTIQUE

08-03-2023

LP ACSID 2022-2023- Projet Java

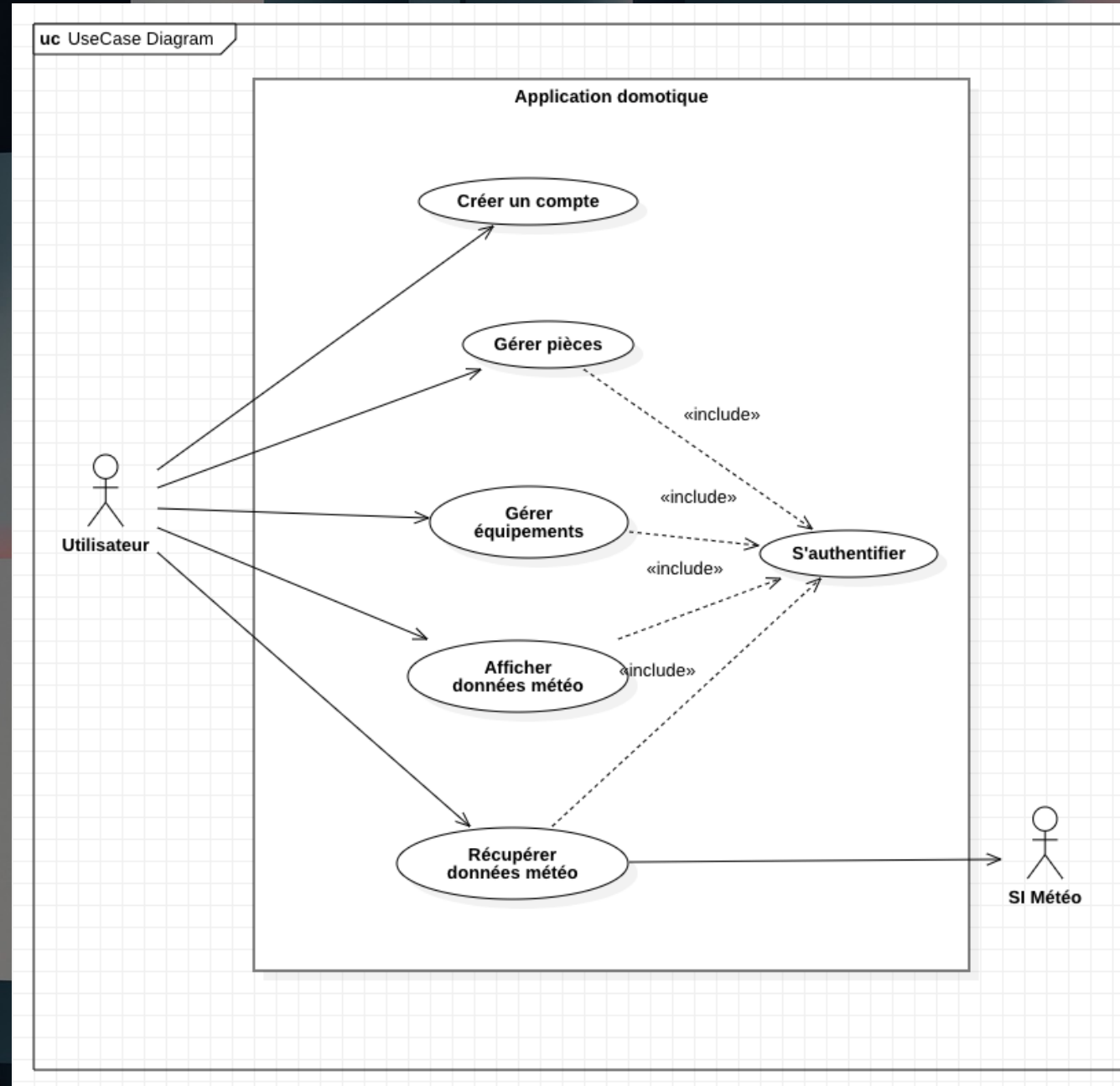
Par Mamitiana RAJAONSON – Yohan VOISIN
Enseignante : Tatiana AUBONNET

INTRODUCTION

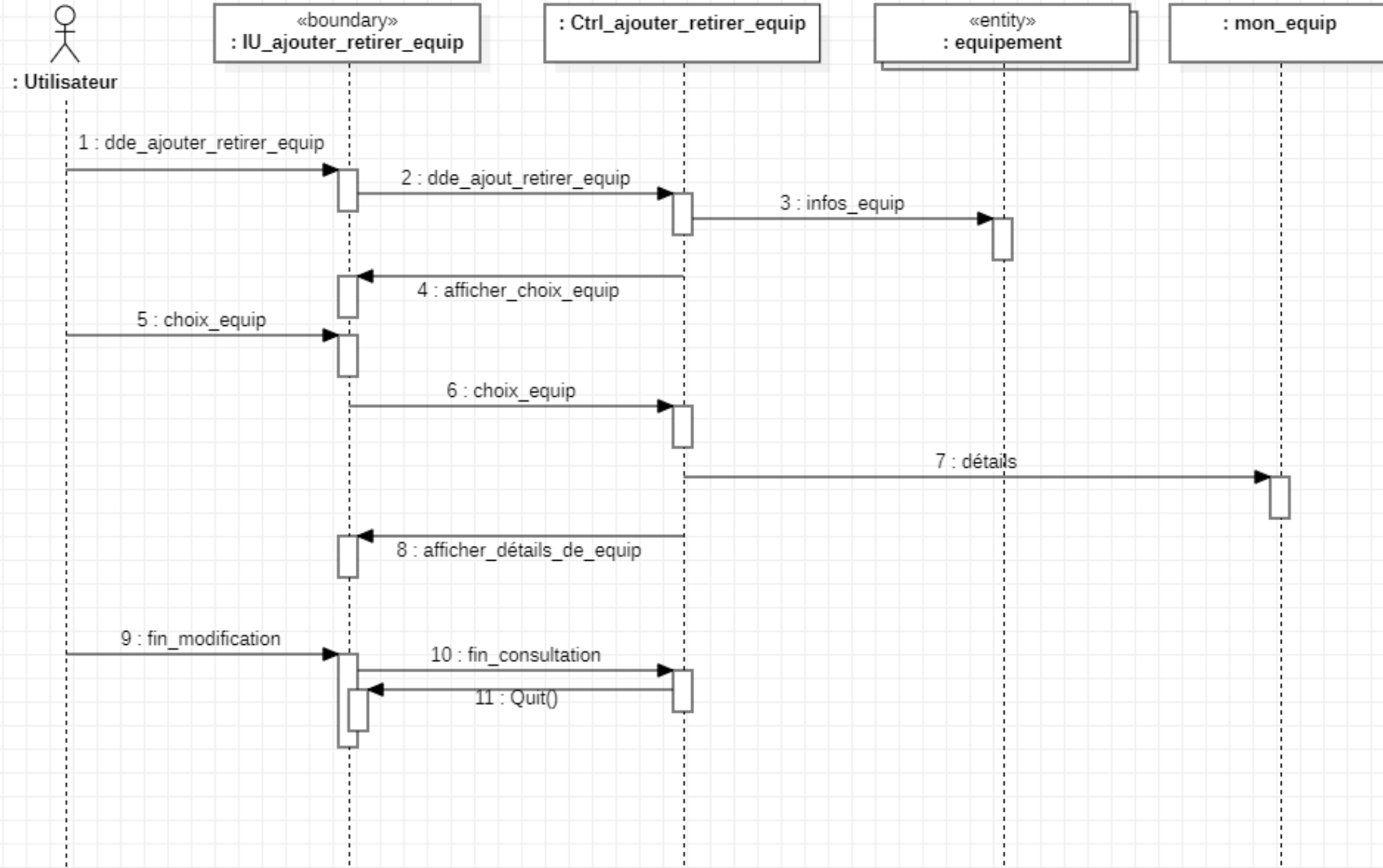
- ANALYSE ET CONCEPTION
- ENVIRONNEMENT DE DÉVELOPPEMENT
- IMPLÉMENTATION

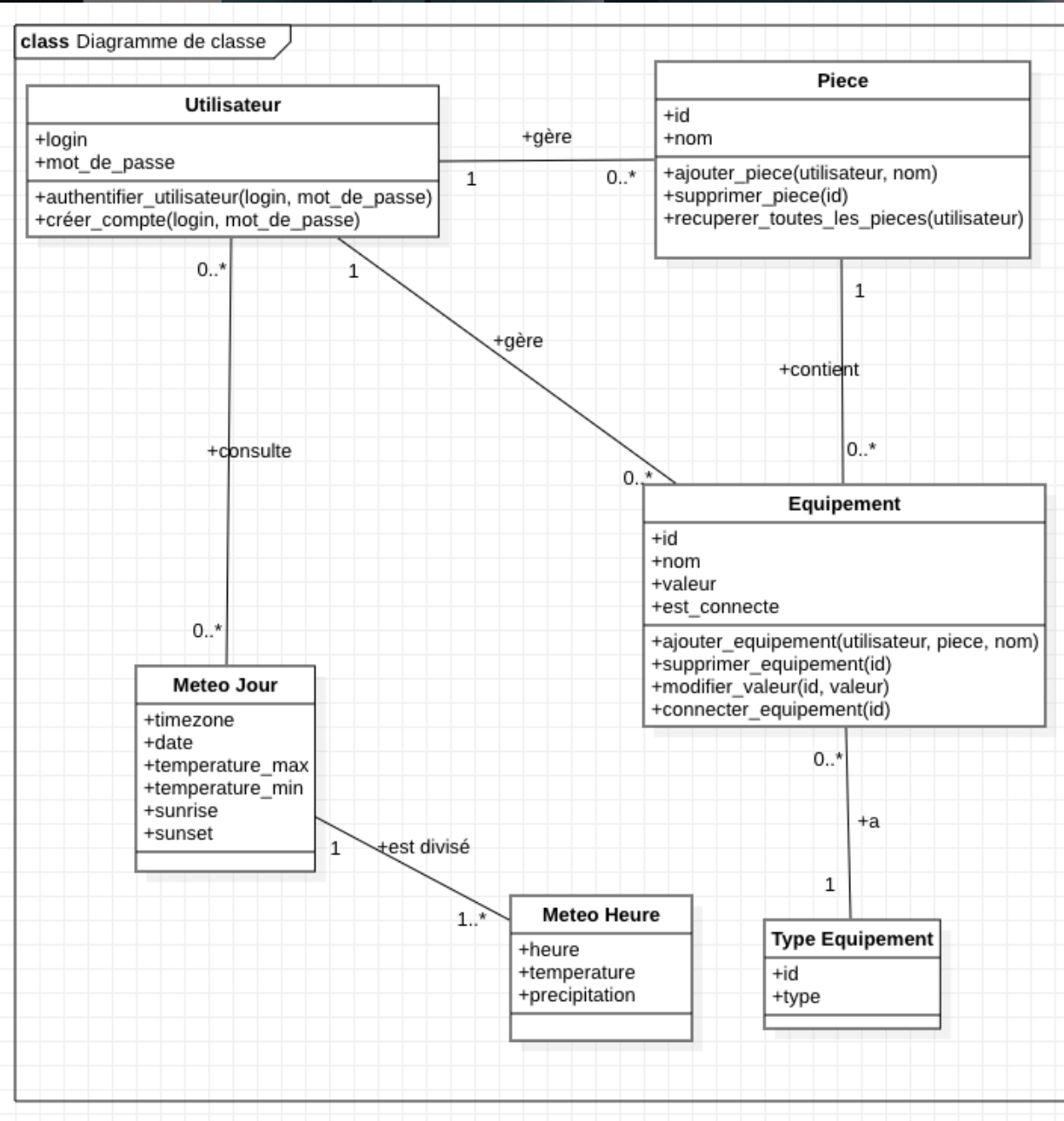
PARTIE I ANALYSE ET CONCEPTION

- Solution de domotique simple
- Surveiller, contrôler nos appareils électroniques à distances
- Accès via un compte utilisateur
- Afficher les données météo



sd ajouter équipement





Spécifications ouvertes et fermées :

- Authentification et autorisation
- Base de données
- Interface web
- Sécurité
- Performances
- Intégration API

PARTIE 2 ENVIRONNEMENT DE DÉVELOPPEMENT



Apache
NetBeans IDE



PARTIE 3 IMPLÉMENTATION

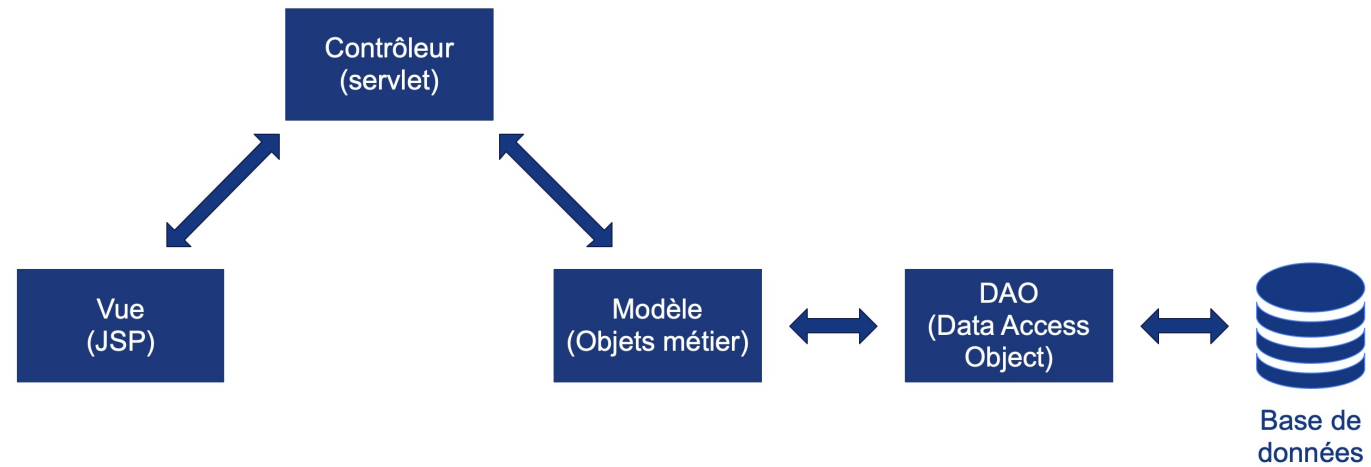


Java EE™

MVC : Modèle – Vue – Contrôleur
DAO : Data Access Object

Framework CSS : Bulma.io

Framework CSS icônes : Bootstrap Icons



PARTIE 3 - CLASSE « PIECE » ET SON INTERFACE DAO

```
public class Piece {  
  
    private String id;  
    private String utilisateur;  
    private String nom;  
  
    public Piece(int id, String utilisateur, String nom) {  
        this.id = "P" + id;  
        this.utilisateur = utilisateur;  
        this.nom = nom;  
    }  
  
    public String getId() {  
        return id;  
    }  
  
    public void setId(int id) {  
        this.id = "P" + id;  
    }  
  
    public String getUtilisateur() {  
        return utilisateur;  
    }  
  
    public void setUtilisateur(String utilisateur) {  
        this.utilisateur = utilisateur;  
    }  
  
    public String getNom() {  
        return nom;  
    }  
  
    public void setNom(String nom) {  
        this.nom = nom;  
    }  
}
```

```
public interface PieceDao {  
  
    public void createPiece(String utilisateur, String nom);  
  
    public Piece readPiece(String utilisateur, String nom);  
  
    public void updatePiece(Piece piece);  
  
    public void deletePiece(String utilisateur, String nom);  
  
    public void deletePieceById(String localId);  
  
    public List<Piece> getAllPieces(String utilisateur);  
  
}
```

PARTIE 3 – IMPLÉMENTATION D'UNE MÉTHODE DE L'INTERFACE

```
@Override
public void createPiece(String utilisateur, String nom) {
    Connection connexion = null;
    PreparedStatement preparedStatement = null;

    try {
        connexion = daoFactory.getConnection();
        String query = "INSERT INTO piece (utilisateur, nom) VALUES(?, ?);";
        preparedStatement = connexion.prepareStatement(query);

        preparedStatement.setString(1, utilisateur);
        preparedStatement.setString(2, nom);

        preparedStatement.executeUpdate();

        System.out.println("INSERT INTO piece " + utilisateur + " " + nom);
    } catch (SQLException e) {
        try {
            if (connexion != null) {
                connexion.rollback();
            }
        } catch (SQLException e2) {}
        Logger.getLogger(PieceDaoImpl.class.getName()).log(Level.SEVERE, null, e);
    } finally {
        try {
            if (preparedStatement != null) {
                preparedStatement.close();
            }
            if (connexion != null) {
                connexion.close();
            }
        } catch (SQLException e) {
            Logger.getLogger(PieceDaoImpl.class.getName()).log(Level.SEVERE, null, e);
        }
    }
}
```


Connexion

 Login

 Mot de passe

Se connecter

Connexion

Login et/ou mot de passe incorrect

 Login

 Mot de passe

Se connecter


```
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {

    HttpSession session = request.getSession();
    ServletContext contexte = getServletContext();
    RequestDispatcher dispatcher;

    String username = request.getParameter("login");
    String password = request.getParameter("pass");

    // Authentification du user
    Boolean isAuth = (Boolean) userDao.validateUser(username, password);

    System.out.println("isAuth " + isAuth);

    if (isAuth) {
        session.setAttribute("isAuth", isAuth);
        session.setAttribute("user", username);

        Cookie cookie = new Cookie("user", username);
        // Validité du cookie en secondes : 60s * 60m * 24h * 3j
        cookie.setMaxAge(60 * 60 * 24 * 3);
        response.addCookie(cookie);

        response.sendRedirect("Home");
    } else {
        request.setAttribute("loginError", "Login et/ou mot de passe incorrect");
        dispatcher = contexte.getRequestDispatcher("/jsp/auth.jsp");
        dispatcher.forward(request, response);
    }
}
```

Accueil
 [Configuration](#)
[A propos](#)

Déconnexion

07/03/2023 01:27:36

Température extérieure

07/0308/0309/03

Min

2.6°C5.0°C7.4°C

Max

9.3°C13.2°C11.0°C

Par heure

00h01h02h03h04h05h

5.6°C5.1°C4.5°C3.9°C3.7°C3.4°C

Ephémérides

07/0308/0309/03

07:1907:1707:15

18:4418:4518:47

CAPTEUR-FENETRE

DETECTEUR-FUMEE

↓

Garage - Porte

▼▲

CAPTEUR-PORTE

LUMIERE

Chambre 1 - Lampe de chevet 1

☐

Chambre 1 - Lampe de chevet 2

☐

Chambre 1 - Lumière

☐

Chambre 2 - Lumière

☐

Salon - Lustre 2

☐

RADIATEUR

19°C

Chambre 1 - Radiateur 1

▼▲

19°C

Chambre 2 - Radiateur

▼▲

18°C

Cuisine - Radiateur


▼▲


21°C

Salon - Radiateur 1

▼▲








18


[Accueil](#)
[Configuration](#)
[A propos](#)











[Déconnexion](#)


[Ajouter une nouvelle pièce](#)
[Ajouter un nouvel équipement](#)

Liste des pièces



Chambre 1	
Chambre 2	
Chambre 3	
Cuisine	
Garage	
Salle de bain	
Salon	


Liste des équipements

Chambre 1 - Lampe de chevet 1	
Chambre 1 - Lampe de chevet 2	
Chambre 1 - Lumière	
Chambre 1 - Radiateur 1	
Chambre 1 - Volet	
Chambre 2 - Lumière	
Chambre 2 - Radiateur	
Chambre 2 - Volet	
Cuisine - Radiateur	

 Open-Meteo

HomeFeaturesWeather APIs ▾Other APIs ▾Blog



Free Weather API

Open-Meteo is an open-source weather API with free access for non-commercial use.
No API key is required. You can use it immediately!

FeaturesDocumentation

Hourly 7-day forecast worldwide

Open-Meteo collaborates with national weather services providing open data with 1 to 11 km resolution. Our APIs select the best weather models for your location and provide data as a simple JSON API.

Super easy. Enter a location, select weather variables and get a

Forecast & CurrentLast 10 daysHistorical data

```
$ curl "https://api.open-meteo.com/v1/forecast?latitude=52.52&longitude=13.41&current_weather=true&hourly=temperature_2m,relativehumidity_2m,windspeed_10m"

{
  "current_weather": {
    "time": "2022-01-01T15:00"
    "temperature": 2.4, "weathercode": 3,
    "windspeed": 11.9, "winddirection": 95.0,
  }
}
```



```
public void fetchMeteo() throws IOException {  
    String url = "https://api.open-meteo.com/v1/meteofrance?latitude=48.85&longitude=2.35&  
hourly=temperature_2m,precipitation&  
daily=weathercode,temperature_2m_max,temperature_2m_min,sunrise,sunset&timezone=auto";  
    URL obj = new URL(url);  
    HttpURLConnection con = (HttpURLConnection) obj.openConnection();  
    con.setRequestMethod("GET");  
  
    int responseCode = con.getResponseCode();  
    System.out.println("fetch meteo " + responseCode);  
  
    BufferedReader in;  
    in = new BufferedReader(new InputStreamReader(con.getInputStream()));  
    String inputLine;  
    StringBuilder responseContent = new StringBuilder();  
    while ((inputLine = in.readLine()) != null) {  
        responseContent.append(inputLine);  
    }  
    in.close();  
    System.out.println("fetch meteo " + responseContent);  
  
    this.data = responseContent.toString();  
}
```

CONCLUSION ET PERSPECTIVES

Merci pour votre attention

Avez-vous des questions ?