



الكلية متعددة التخصصات - وورزازات  
+oX+eMHe+ - UoO%o%+  
FACULTÉ POLYDISCIPLINAIRE DE OUARZAZATE



الكلية متعددة التخصصات - وورزازات  
+oX+eMHe+ - UoO%o%+  
FACULTÉ POLYDISCIPLINAIRE DE OUARZAZATE

**Université Ibn Zohr**  
**Faculté Polydisciplinaire**  
**de Ouarzazate**  
**Filière Master : Intelligence**  
**Artificielle et Applications**

Module : Python pour le web

**Système de maintenance prédictive  
pour machines industrielles**

Projet Réalisé par :

**Hanane Iqli**  
**Hanan Bassou**  
**Maryame Khouya**

Encadré par :

**Pr. Rachid DAKIR**  
Professeure à la Faculté  
Polydisciplinaire de  
Ouarzazate

*Année universitaire : 2024-2025*

---

# Table des matières

<b>Remerciement</b>	<b>2</b>
<b>Résumé</b>	<b>3</b>
<b>Abstract</b>	<b>4</b>
<b>Introduction général</b>	<b>5</b>
<b>1 Système de maintenance prédictive pour machines industrielles</b>	<b>6</b>
1.1 Définition . . . . .	6
1.2 Fonctionnement typique . . . . .	6
1.3 Avantages . . . . .	7
1.4 Exemples de technologies utilisées . . . . .	7
1.5 Conclusion . . . . .	7
<b>2 Application de Surveillance de Température pour Moteur Industriel</b>	<b>8</b>
2.1 Définition . . . . .	8
2.2 Capteur de Température . . . . .	8
2.3 Composants essentiels . . . . .	8
2.4 Avantages de l'application . . . . .	9
2.5 Exemples d'applications . . . . .	9
<b>3 Conception</b>	<b>10</b>
3.1 Description de l'application . . . . .	10
3.2 Diagramme de cas d'utilisation . . . . .	10
3.3 Diagramme de classe . . . . .	12
3.4 Diagramme d'interaction . . . . .	14
3.5 Diagramme d'activités . . . . .	15
<b>4 Réalisation de l'Application</b>	<b>18</b>
4.1 Architecture du système . . . . .	18
4.2 Technologies utilisées . . . . .	19
4.3 Visualisation de l'application . . . . .	20
4.4 Visualisation de résultat . . . . .	21
<b>Conclusion</b>	<b>22</b>
<b>Références</b>	<b>23</b>

---

# Remerciement

*Tout d'abord, nous remercions Dieu tout-puissant pour les privilèges qu'il nous a accordés, l'opportunité d'étudier, et pour les capacités, compétences et patience qu'il nous a données pour réaliser ce travail. Nous remercions notre superviseur, le Docteur **Rachid DAKIR** pour son soutien, son aide et ses conseils. Nous n'oublions pas non plus de remercier et d'apprécier nos chers enseignants à toutes les étapes de notre vie, sans lesquels nous ne serions pas arrivés là où nous en sommes maintenant.*

---

# Résumé

Ce projet avait pour objectif de développer une application web de maintenance prédictive destinée à surveiller la température des moteurs industriels.

Après une présentation du concept de maintenance prédictive et de son importance dans l'industrie, une phase de conception a permis de modéliser les différentes fonctionnalités du système à l'aide de diagrammes UML (cas d'utilisation, classes, activités, interactions).

Ensuite, l'application a été réalisée en utilisant le framework Django, avec un front-end basé sur HTML, CSS et Bootstrap, et une base de données PostgreSQL. L'interface permet l'ajout, la visualisation et l'analyse graphique des relevés de température, tout en identifiant automatiquement les anomalies critiques.

## Mots-clés :

Maintenance prédictive, Surveillance de température, Moteurs industriels, Django, PostgreSQL, Diagrammes UML, Anomalies, Application web.

---

# Abstract

This project aimed to develop a web application for predictive maintenance designed to monitor the temperature of industrial engines.

After a presentation of the predictive maintenance concept and its importance in the industry, a design phase allowed for the modeling of the different functionalities of the system using UML diagrams (use cases, classes, activities, interactions).

Then, the application was built using the Django framework, with a front-end based on HTML, CSS, and Bootstrap, and a PostgreSQL database. The interface allows for the addition, visualization, and graphical analysis of temperature readings, while automatically identifying critical anomalies.

## **Keywords :**

Predictive maintenance, Temperature monitoring, Industrial motors, Django, PostgreSQL, UML diagrams, Anomalies, Web application.

---

# Introduction générale

## 1. Contexte général

Dans le cadre de l'amélioration de la performance et de la fiabilité des systèmes industriels, ce projet vise à développer une application web de surveillance de température pour moteurs industriels. Cette solution s'inscrit dans une logique de maintenance prédictive, une stratégie de plus en plus adoptée dans l'industrie pour anticiper les défaillances techniques et éviter les arrêts de production non planifiés.

L'application permettra de surveiller en temps réel la température des moteurs à partir de données collectées par des capteurs connectés, avec une interface permettant la visualisation, l'analyse et la gestion des alertes en cas de dépassement de seuils critiques.

## 2. Objectif de l'Application

L'application doit :

- Afficher en temps réel ou différé la température d'un ou plusieurs moteurs.
- Enregistrer les valeurs dans une base de données PostgreSQL.
- Permettre d'établir des alertes si seuil dépassé.
- Être accessible via un navigateur web (intranet ou internet).
- Ajouter des courbes ou graphiques de température dans le temps.(Optionnel)

---

## Chapitre 1

# Système de maintenance prédictive pour machines industrielles

## Introduction

La maintenance prédictive est une approche qui permet de mieux gérer les équipements industriels, en améliorant leur disponibilité, en réduisant les coûts et en prolongant leur durée de vie.

### 1.1 Définition

Un système de maintenance prédictive pour machines industrielles est une solution technologique qui utilise des données en temps réel, souvent collectées par des capteurs, pour prédire les défaillances et optimiser les interventions de maintenance. C'est une approche visant à éviter les arrêts de production inopinés et à prolonger la durée de vie des équipements.

### 1.2 Fonctionnement typique

Comment fonctionne un système de maintenance prédictive :

#### 1. Collecte de données :

Des capteurs sont installés sur les machines pour collecter des données telles que la température, les vibrations, la consommation d'énergie, etc. Ces données sont transmises en temps réel à un système d'analyse.

#### 2. Analyse des données :

Des algorithmes d'apprentissage automatique (machine learning) sont utilisés pour analyser les données, détecter les anomalies et identifier les tendances qui peuvent indiquer un problème imminent.

#### 3. Prédiction des défaillances :

Le système prédictif permet de prédire la durée de vie des composants, l'occurrence d'une panne ou la nécessité d'une intervention de maintenance.

#### 4. Planification de la maintenance :

Les techniciens peuvent alors planifier les interventions de maintenance de manière proactive, en fonction des prédictions du système.

### 1.3 Avantages

- **Réduction des coûts** : En évitant les pannes coûteuses et les arrêts de production inopinés, la maintenance prédictive permet de réduire les coûts de maintenance et de réparation.
- **Amélioration de la disponibilité des équipements** : En anticipant les pannes, la maintenance prédictive permet de maintenir les équipements en fonctionnement plus longtemps, ce qui améliore leur disponibilité.
- **Prolongement de la durée de vie des équipements** : En intervenant avant que les équipements ne tombent en panne, la maintenance prédictive permet de prolonger leur durée de vie.
- **Optimisation de la production** : La maintenance prédictive permet de mieux planifier la production, en évitant les arrêts de production inopinés et en optimisant l'utilisation des équipements.

### 1.4 Exemples de technologies utilisées

- **Analyse des vibrations** : L'analyse des vibrations peut aider à détecter des déséquilibres ou des décalages qui peuvent indiquer un problème avec les machines.
- **Analyse infrarouge** : La thermographie infrarouge peut aider à détecter des problèmes de surchauffe ou des fuites.
- **Analyse des fluides** : L'analyse chimique et physique des fluides peut aider à identifier les problèmes avec les lubrifiants ou les liquides de refroidissement.
- **Modèles d'apprentissage automatique** : Les modèles d'apprentissage automatique peuvent être utilisés pour prédire les défaillances et identifier les problèmes potentiels.

### 1.5 Conclusion

La maintenance prédictive s'impose comme une solution innovante et efficace pour anticiper les défaillances des machines industrielles. En combinant capteurs, analyses avancées et intelligence artificielle, elle permet de planifier les interventions au bon moment, d'éviter les arrêts imprévus et d'optimiser la performance des équipements. Cette approche contribue ainsi à une meilleure gestion industrielle, plus économique et plus fiable.



---

## Chapitre 2

# Application de Surveillance de Température pour Moteur Industriel

## Introduction

Le capteur de température du moteur est crucial dans l'industrie moderne. Il est fondamental pour surveiller la température du moteur afin de prévenir les pannes, optimiser les performances et garantir la sécurité des équipements.

### 2.1 Définition

Une application de surveillance de température pour moteurs industriels combine :

- des **capteurs** (thermocouples, RTD, etc.) qui mesurent la température,
- des **systèmes d'acquisition** qui transmettent ces données,
- un **logiciel** (plateformes comme *JRI-MySirius* ou *TIA Portal*) qui visualise, analyse et alerte sur les dépassements de seuils.

Ces systèmes préviennent les pannes, optimisent les performances et garantissent la sécurité en surveillant le moteur en temps réel et en planifiant la maintenance préventive.

### 2.2 Capteur de Température

Qu'est-ce qu'un capteur de température ?[1]

C'est un composant des systèmes de gestion du moteur, conçu pour mesurer avec précision la température, essentielle pour maintenir le moteur dans ses limites de température optimales.

En termes de **maintenance prédictive**, le capteur du moteur permet aux entreprises d'anticiper les problèmes avant qu'ils ne surviennent, réduisant ainsi les temps d'arrêt et les coûts de réparation.

En plus de protéger le moteur et d'améliorer l'efficacité opérationnelle, le capteur de température du liquide de refroidissement contribue également à la durabilité, en réduisant la consommation de carburant et les émissions.

### 2.3 Composants essentiels

#### 1. Capteurs de Température

- **Thermocouples et RTD / thermistances** : capables de mesurer des plages de température très larges, ils sont souvent reliés à des enregistreurs.
- **Capteurs spécifiques** : des sondes à visser pour la surveillance du liquide de refroidissement dans les moteurs diesel, par exemple.

### 2. Acquisition et Transmission des Données

- **Enregistreurs et transmetteurs** : collectent les données des capteurs et les envoient vers une plateforme centrale.
- **Connectivité** : les données peuvent être transmises sans fil vers le cloud ou des serveurs locaux, ou via des connexions numériques comme l'*IO-Link*.

### 3. Logiciel de Superviision

- **Plateformes centralisées** : des logiciels comme *JRI-MySirius* ou la plateforme *Siemens TIA Portal* permettent de gérer et visualiser les données de température.
- **Tableaux de bord et alertes** : l'interface intuitive affiche le statut des moteurs, avec des seuils d'alerte personnalisables et une gestion des utilisateurs et des droits d'accès.

### 2.4 Avantages de l'application

- **Maintenance préventive** : détecte les anomalies avant qu'elles ne conduisent à des pannes coûteuses, réduisant ainsi les temps d'arrêt imprévus.
- **Optimisation des performances** : assure le maintien de conditions de fonctionnement optimales, prolongeant la durée de vie du moteur et améliorant l'efficacité énergétique.
- **Sécurité accrue** : permet d'éviter les surchauffes dangereuses et garantit la sécurité des processus industriels.

### 2.5 Exemples d'applications

- **Industrie manufacturière** : surveillance des processus et des machines pour maintenir une qualité de production constante.
- **Automobile et ferroviaire** : contrôle de la température des moteurs, y compris des moteurs diesel des locomotives.
- **Applications de surveillance d'état** : les systèmes intégrés dans les servomoteurs peuvent surveiller la température, les vibrations et l'humidité pour détecter des défauts.

## Conclusion

Grâce au contrôle continu de la température, ce système permet la mise en œuvre de stratégies de maintenance prédictive, aidant ainsi à prévenir les pannes et à réduire les temps d'arrêt.

En maintenant la température du moteur dans des plages optimales, ce capteur contribue de manière significative à la **durabilité** et aux **performances** des équipements, rendant les opérations industrielles plus **sûres** et plus **efficaces**.

## Introduction

Ce chapitre présente la conception de l'application à travers différents diagrammes UML. Ils permettent de visualiser les fonctionnalités, les rôles des utilisateurs et l'organisation interne du système avant le développement.

### 3.1 Description de l'application

Ce projet consiste à développer une application web permettant de surveiller la température des moteurs industriels, détecter automatiquement les anomalies et fournir une interface d'administration pour consulter et gérer les relevés. L'application utilise Django pour le backend. Les anomalies (lorsque la température dépasse un seuil critique) sont automatiquement identifiées et affichées en rouge dans l'interface. Les administrateurs peuvent visualiser et filtrer les relevés via l'interface Django Admin. Un tableau de bord graphique est également disponible pour afficher le nombre d'anomalies. L'application répond à des exigences de performance, de sécurité et d'évolutivité, avec des options futures pour des notifications, des graphiques interactifs, et l'intégration d'une authentification.

### 3.2 Diagramme de cas d'utilisation

Les diagrammes de cas d'utilisation (DCU) sont des diagrammes UML utilisés pour une représentation du comportement fonctionnel d'un système logiciel. Ils sont utiles pour des présentations auprès de la direction ou des acteurs d'un projet, mais pour le développement, les cas d'utilisation sont plus appropriés. En effet, un cas d'utilisation (use cases) représente une unité discrète d'interaction entre un utilisateur (humain ou machine) et un système. Ainsi, dans un diagramme de cas d'utilisation, les utilisateurs sont appelés acteurs (actors), et ils apparaissent dans les cas d'utilisation.[2]

## Acteurs

Les acteurs sont des entités externes qui interagissent avec le système, comme une personne humaine ou un robot. Une même personne (ou robot) peut être plusieurs acteurs pour un système, c'est pourquoi les acteurs doivent surtout être décrits par leur rôle. Ce rôle décrit les besoins et les capacités de l'acteur. Un acteur agit sur le système (Figure 2.1).

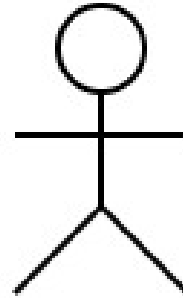


FIGURE 3.1 – Représentation d'un acteur UML

## Relations

Trois types de relations sont prises en charge par la norme UML et sont graphiquement représentées par des types particuliers de ces relations. Les relations indiquent que le cas d'utilisation source présente les mêmes conditions d'exécution que le cas issu. Une relation simple entre un acteur et une utilisation est un trait simple.

## Inclusions

Dans ce type d'interaction, le premier cas d'utilisation inclut le second et son issue dépend souvent de la résolution du second. Ce type de description est utile pour extraire un ensemble de sous-comportements communs à plusieurs tâches, comme une macro en programmation. Elle est représentée par une flèche en pointillé et le terme *include*.

## Extensions

Les extensions (*extend*) représentent des prolongements logiques de certaines tâches sous certaines conditions. Autrement dit un cas d'utilisation A étend un cas d'utilisation B lorsque le cas d'utilisation A peut être appelé au cours de l'exécution du cas d'utilisation B. Elle est représentée par une flèche en pointillée avec le terme *extend*. Ce type de relation peut être utile pour traiter des cas particuliers ou fonctions optionnelles, préciser les objectifs, ou encore pour tenir compte de nouvelles exigences au cours de la maintenance du système et de son évolution.

## Généralisations

La troisième relation est la relation de généralisation ou spécialisation. Le cas d'utilisation A est une généralisation de B, si B est un cas particulier de A c'est-à-dire lorsque A peut être substitué par B pour un cas précis. Ces relations sont des traits pleins terminés par une flèche en triangle.

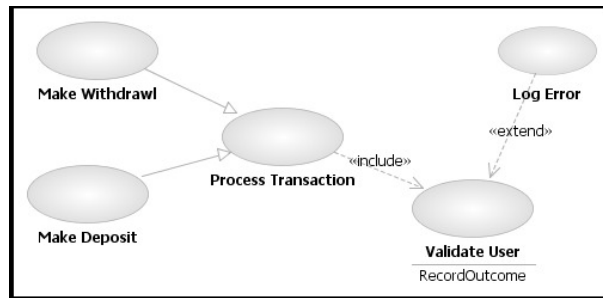


FIGURE 3.2 – Exemple de relations

voici le diagramme de cas d'utilisation de notre application de Surveillance de Température pour Moteur Industriel :

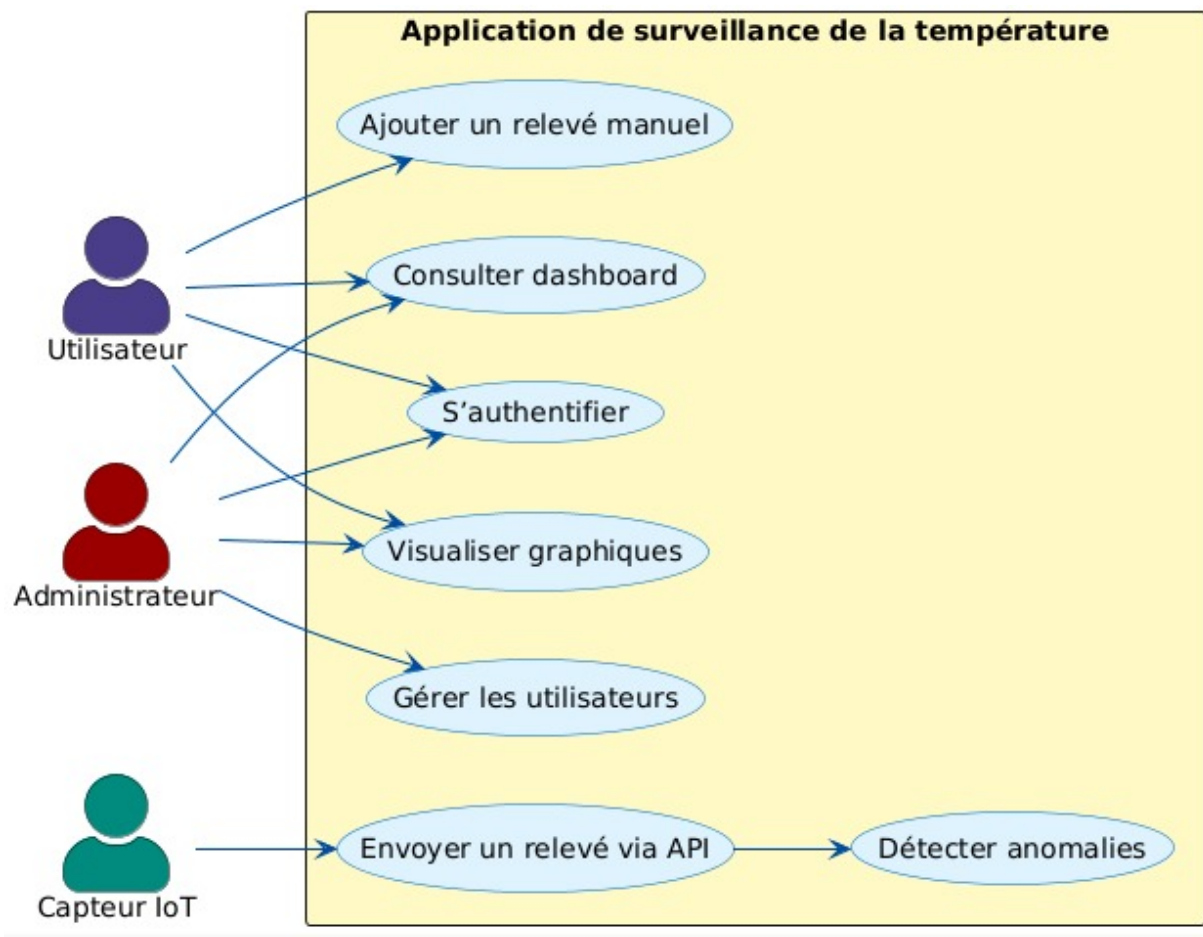


FIGURE 3.3 – Diagramme de cas d'utilisation

### 3.3 Diagramme de classe

Le diagramme de classes est un schéma utilisé en génie logiciel pour présenter les classes et les interfaces des systèmes ainsi que leurs relations. Ce diagramme fait partie de la partie statique d'UML, ne s'intéressant pas aux aspects temporels et dynamiques.

Une classe décrit les responsabilités, le comportement et le type d'un ensemble d'objets. Les éléments de cet ensemble sont les instances de la classe.

Une **classe** est un ensemble de fonctions et de données (attributs) qui sont liées ensemble par un champ sémantique. Les classes sont utilisées dans la programmation orientée objet. Elles permettent de modéliser un programme et ainsi de découper une tâche complexe en plusieurs petits travaux simples.[3]

## Schéma d'une classe

Une classe est représentée par un rectangle séparé en trois parties :

- la première partie contient le **nom de la classe**,
- la seconde contient les **attributs de la classe**,
- la dernière contient les **méthodes de la classe**.

La seconde et la dernière représentent le comportement de la classe.

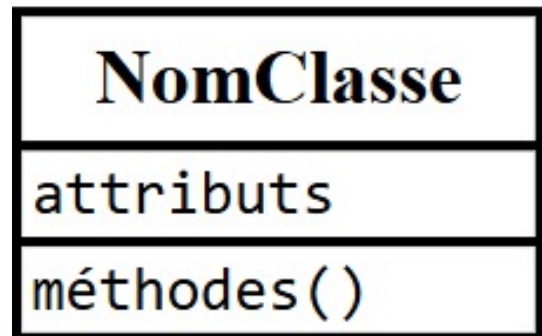


FIGURE 3.4 – Modèle d'une simple classe

voici le diagramme de classe de notre application de Surveillance de Température pour Moteur Industriel :

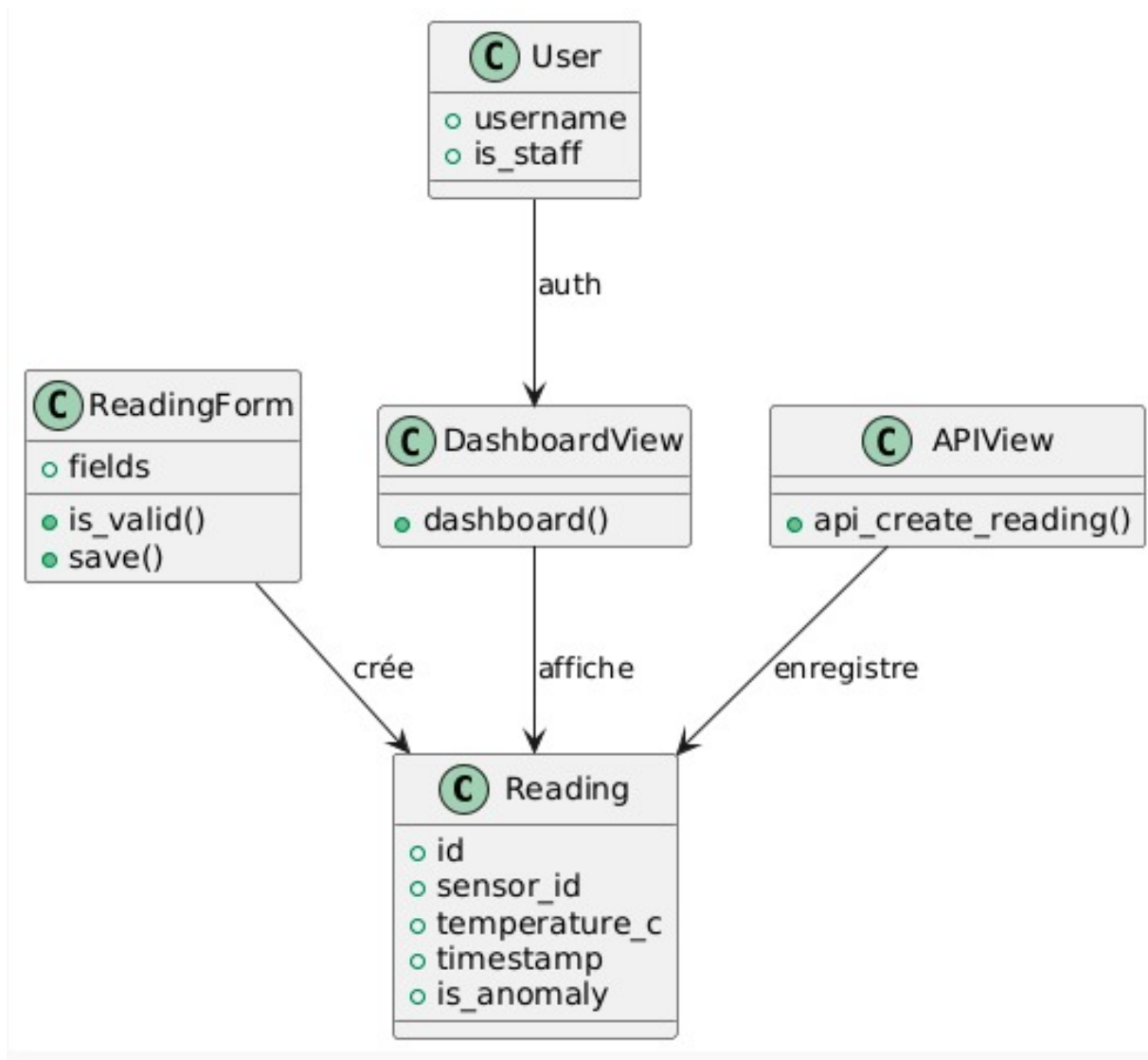


FIGURE 3.5 – Diagramme de classe

## 3.4 Diagramme d'interaction

Les diagrammes globaux d'interaction définissent des interactions par une variante des diagrammes d'activité, d'une manière qui permet une vue d'ensemble de flux de contrôle.

Ils se concentrent sur la vue d'ensemble de flux de contrôle où les nœuds sont des interactions ou `InteractionUses`.<sup>[4]</sup>

voilà les diagrammes d'interaction de notre application de Surveillance de Température pour Moteur Industriel :

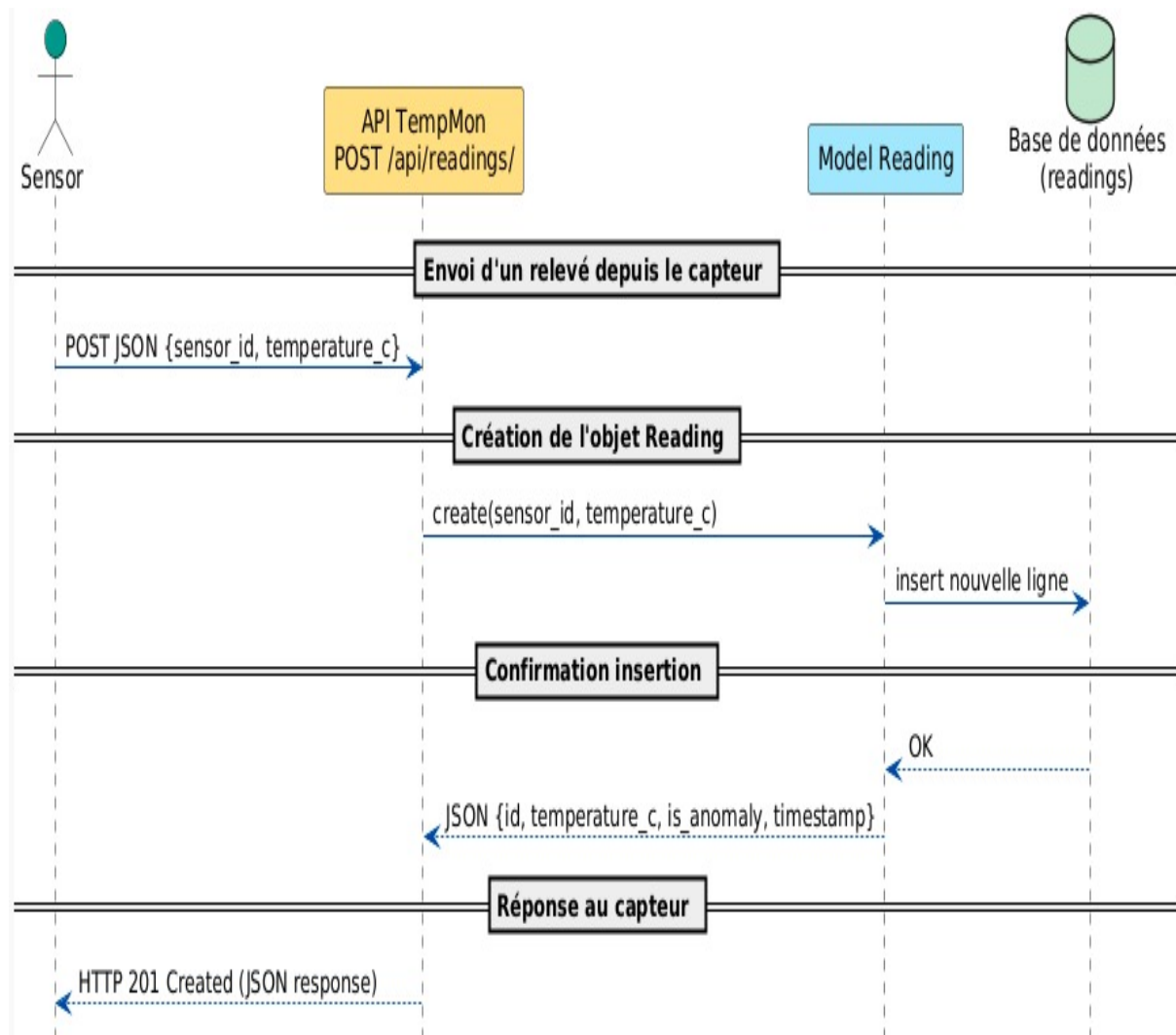


FIGURE 3.6 – Diagramme d'interaction global

### 3.5 Diagramme d'activités

Le diagramme d'activité est un diagramme comportemental d'UML, permettant de représenter le déclenchement d'événements en fonction des états du système et de modéliser des comportements parallélisables (multi-threads ou multi-processus). Le diagramme d'activité est également utilisé pour décrire un flux de travail (workflow).<sup>[5]</sup>

voici le diagramme d'activités de notre application de Surveillance de Température pour Moteur Industriel :



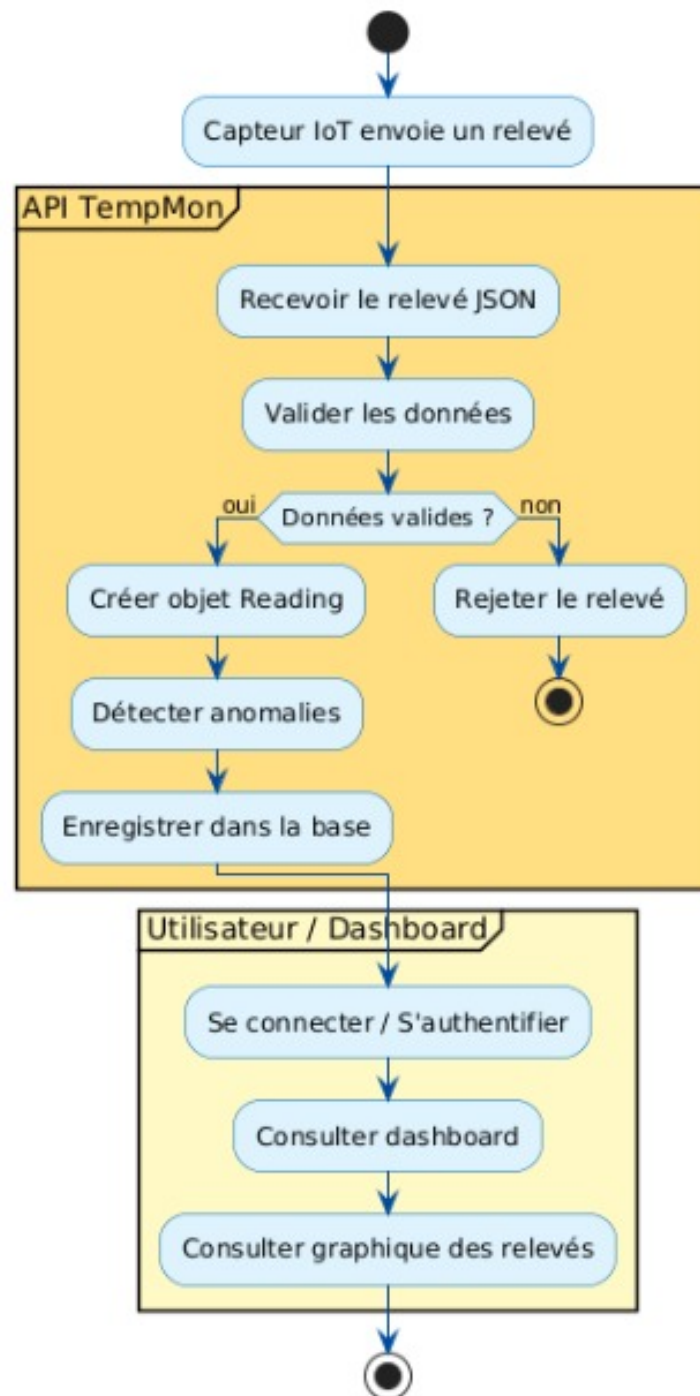


FIGURE 3.7 – Diagramme d'activités

## Conclusion

La modélisation réalisée dans ce chapitre offre une vue claire du fonctionnement de l'application. Elle facilite le développement et garantit une base solide pour l'évolution future du système.

## Chapitre 4

# Réalisation de l'Application

## Introduction

Pour réaliser un projet informatique sur un système de maintenance prédictive pour machines industrielles, il est essentiel de suivre une démarche structurée, allant de la définition du besoin à la mise en œuvre technique et la validation.

### 4.1 Architecture du système

Voici l'organisation des fichiers et dossiers du projet Django TempMon :

```
tempmon_project/ ..... Racine du projet
├─ manage.py ..... Script de gestion
├─ tempmon_project/ ..... Répertoire principal du projet
│   ├── __init__.py
│   ├── settings.py ..... Configuration
│   ├── urls.py ..... Routes globales
│   ├── wsgi.py
│   └─ asgi.py
├─ tempmon/ ..... Application principale
│   ├── __init__.py
│   ├── admin.py ..... Admin Django
│   ├── apps.py
│   ├── forms.py ..... Formulaires
│   ├── migrations/ ..... Migrations DB
│   │   └─ __init__.py
│   ├── models.py ..... Modèle Reading
│   ├── tests.py
│   ├── urls.py ..... Routes app
│   └─ views.py ..... Logique des vues
├─ templates/ ..... Dossier templates
│   ├── base.html ..... Template global
│   ├── registration/
│   │   └─ login.html ..... Page de connexion
│   └─ tempmon/
│       ├── dashboard.html ..... Page dashboard
│       └─ graph.html ..... Page graphique des températures
└─ static/ ..... Fichiers statiques
    ├── css/
    └─ style.css
```

### 4.2 Technologies utilisées

Le projet a été réalisé avec les technologies suivantes :

#### Backend :



**Django** est un framework web open source en Python. Il a pour but de rendre le développement d'applications web simple et basé sur la réutilisation de code. Développé en 2003 pour le journal local de Lawrence (État du Kansas, aux États-Unis).

#### Frontend :



Le HyperText Markup Language (HTML5) est le langage de balisage conçu pour écrire les pages web. Il permet de structurer le texte, de créer des formulaires et d'inclure des ressources multimédias (images, vidéos, programmes). Il reste aujourd'hui un standard incontournable du web.



Les feuilles de style en cascade (CSS – Cascading Style Sheets) décrivent la présentation des documents HTML et XML. Standardisé par le W3C depuis les années 1990, CSS est devenu essentiel pour la conception de sites web modernes, offrant un design responsive et cohérent sur tous les navigateurs.



**Bootstrap** est une collection d'outils utiles à la création du design (graphisme, animation et interactions avec la page dans le navigateur, etc.) de sites et d'applications web. C'est un ensemble qui contient des codes HTML et CSS, des formulaires, boutons, outils de navigation et autres éléments interactifs, ainsi que des extensions JavaScript en option. C'est l'un des projets les plus populaires sur la plate-forme de gestion de développement GitHub.

### Systeme de gestion de base de données SGBD :



**PostgreSQL** est un système de gestion de base de données relationnelle orienté objet puissant et open source qui est capable de prendre en charge en toute sécurité les charges de travail de données les plus complexes.

### 4.3 Visualisation de l'application

- **l'authentification :**

Page de connexion de l'application "TempMon" avec un formulaire pré-rempli pour l'utilisateur "admin" et un champ pour le mot de passe. Bouton "Connection" pour valider.

FIGURE 4.1 – Page d'authentification

- **Tableau de bord de température des moteurs :**

Tableau de bord de gestion des températures des moteurs. Permet d'ajouter des relevés et affiche un historique sous forme de tableau avec statuts et dates. Lien vers un graphique complet.

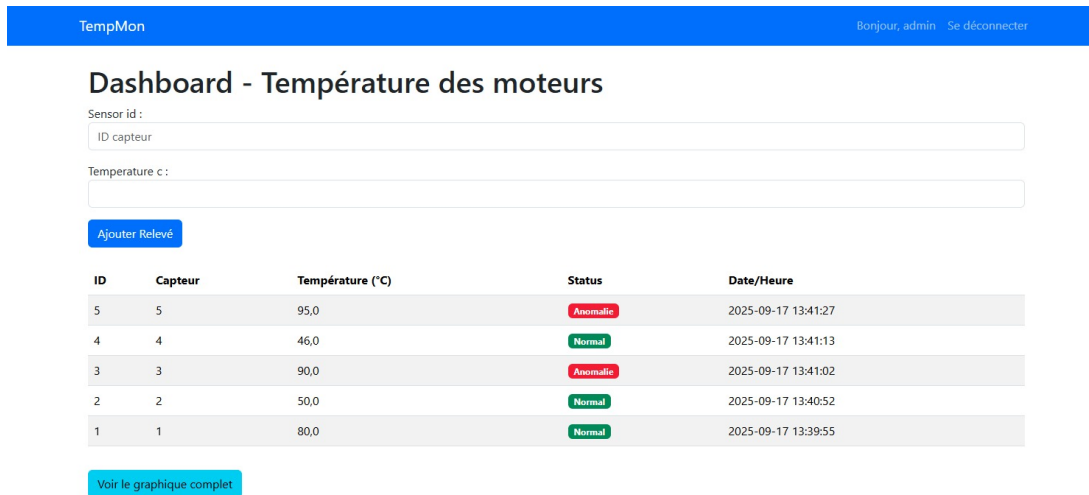


FIGURE 4.2 – Page premier d'application

## 4.4 Visualisation de résultat

### • Page de graphique des températures

Interface de graphique de températures de l'application. Affiche des courbes de température et de moyenne, avec un bouton de retour au tableau de bord.

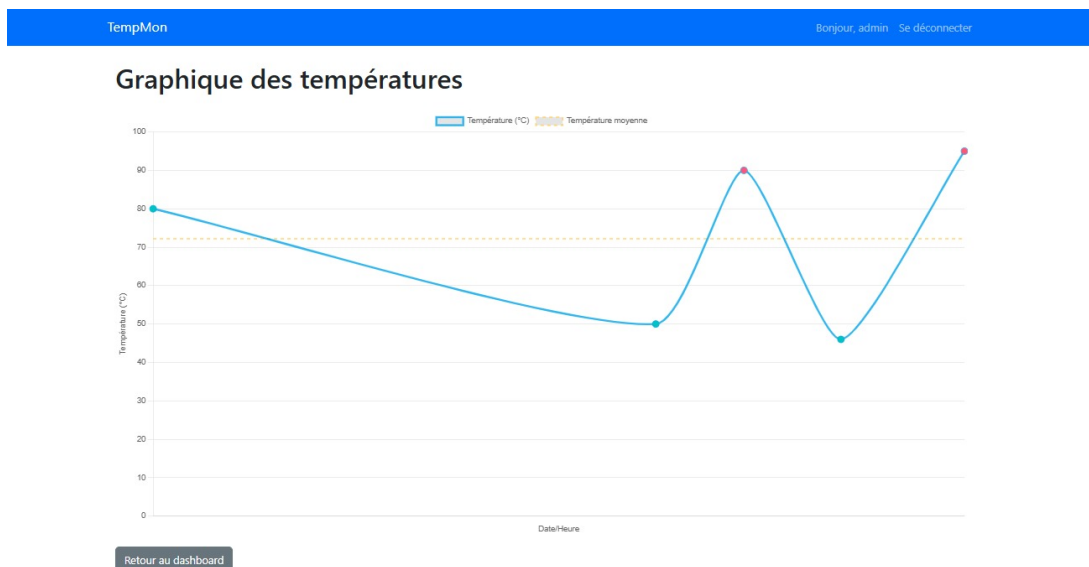


FIGURE 4.3 – exemple de graphe de température

## Conclusion

Ce chapitre a présenté la mise en œuvre technique de l'application TempMon. À l'aide de Django, d'outils web modernes et d'une base de données robuste, une solution fonctionnelle de surveillance des températures a été développée, avec une interface claire et intuitive.

---

## Conclusion générale

Le développement de l'application TempMon a démontré l'apport des technologies web dans la mise en place d'un système de maintenance prédictive simple, efficace et évolutif.

En combinant des outils modernes côté serveur et client, ainsi qu'une architecture bien pensée, l'application répond aux besoins de surveillance des équipements industriels.

Elle constitue une base solide pour de futures améliorations, telles que l'intégration de notifications en temps réel, de modèles prédictifs avancés ou de modules de reporting automatisé.

Ce projet ouvre ainsi la voie à des solutions plus intelligentes et automatisées dans le domaine de la maintenance industrielle.

---

## Références

1. <https://www.spainautoparts.com/fr/blog/qu-est-ce-que-le-capteur-de-te>
2. Wikipédia, *Diagramme de cas d'utilisation*,  
[https://fr.wikipedia.org/wiki/Diagramme\\_de\\_cas\\_d%27utilisation](https://fr.wikipedia.org/wiki/Diagramme_de_cas_d%27utilisation)
3. Wikipédia, *Diagramme de classes*,  
[https://fr.wikipedia.org/wiki/Diagramme\\_de\\_classes](https://fr.wikipedia.org/wiki/Diagramme_de_classes)
4. Wikipédia, *Diagramme global d'interaction*,  
[https://fr.wikipedia.org/wiki/Diagramme\\_global\\_d%27interaction](https://fr.wikipedia.org/wiki/Diagramme_global_d%27interaction)
5. Wikipédia, *Diagramme d'activités*,  
[https://fr.wikipedia.org/wiki/Diagramme\\_d%27activit%C3%A9](https://fr.wikipedia.org/wiki/Diagramme_d%27activit%C3%A9)
6. Définition de Django : [https://fr.wikipedia.org/wiki/Django\\_\(framework\)](https://fr.wikipedia.org/wiki/Django_(framework))
7. Définition de HTML : [https://fr.wikipedia.org/wiki/Hypertext\\_Markup\\_Language](https://fr.wikipedia.org/wiki/Hypertext_Markup_Language)
8. Définition de CSS : [https://fr.wikipedia.org/wiki/Feuilles\\_de\\_style\\_en\\_cascade](https://fr.wikipedia.org/wiki/Feuilles_de_style_en_cascade)
9. Bootstrap : [https://fr.wikipedia.org/wiki/Bootstrap\\_\(framework\)](https://fr.wikipedia.org/wiki/Bootstrap_(framework))
10. postgresql : <https://www.oracle.com/fr/database/definition-postgresql/>