**Atli Antoine**

**Aly Youssof**

**Al Massati Soufiane**

**Romeih Youssef**

**Rapport de Conception : Plateforme Big Data pour GoodAir - MSPR3**



# **Introduction**

Dans le cadre du bloc de compétence n°3 de la certification professionnelle « Expert en Ingénierie des Données », ce document constitue le rapport de conception d'une plateforme Big Data développée pour le projet GoodAir. Ce dernier est une initiative portée par TotalGreen, entreprise française œuvrant dans le secteur des énergies renouvelables, qui ambitionne de créer un laboratoire de recherche consacré à l’étude de la qualité de l’air et de l’eau en France. Le laboratoire, baptisé GoodAir, a pour objectif d’évaluer les effets du changement climatique, de proposer des recommandations de santé publique basées sur des observations environnementales précises, mais aussi de sensibiliser la population par l'intermédiaire de plateformes interactives et scientifiques.

La plateforme que nous concevons devra répondre à plusieurs enjeux critiques : tout d’abord, elle devra permettre la collecte automatisée de données issues de sources ouvertes fiables. Ensuite, elle devra offrir une architecture de traitement et de stockage évolutive, permettant de manipuler et analyser de grands volumes de données en temps quasi-réel. Cette architecture devra également intégrer des mécanismes de sécurité stricts, respecter la législation européenne en matière de protection des données, et inclure des outils d’analyse pour les experts métiers du laboratoire. Enfin, le système devra pouvoir, à terme, intégrer des algorithmes d’intelligence artificielle afin de détecter automatiquement des anomalies ou formuler des prédictions sur les évolutions environnementales.

Ce rapport vise à présenter la réflexion architecturale menée dans ce cadre, en détaillant les choix technologiques effectués, les différentes étapes du pipeline de traitement des données, ainsi que les méthodologies employées pour garantir la qualité, la sécurité et l’évolutivité de la solution mise en œuvre. Il s'agit d'une vue d'ensemble critique de la solution proposée, dans une logique de développement agile.

# 

Sommaire

[**Introduction 2**](#_heading=h.vmlcgc1mg44o)

[**I. Contexte et Objectifs 4**](#_heading=h.xxe56uqthiks)

[**II. Étapes du Projet 5**](#_heading=h.nm69f5q1fki0)

[**A. Sécurité de l’information (RGPD) 5**](#_heading=h.wq8xba69vfuv)

[**B. Modélisation des données 6**](#_heading=h.qupd8zso58r7)

[1. Architecture « fact-dimension » 8](#_heading=h.i0h2dare89yi)

[2. Pourquoi un data warehouse ? 9](#_heading=h.946keswa97k9)

[3. Flux ELT 9](#_heading=h.6sd97y3k1083)

[**C. Récolte des données 10**](#_heading=h.8s26ydwcix2)

[**D. Stockage des données 11**](#_heading=h.sutrxicuk3gu)

[**E. Intégration des données 12**](#_heading=h.jw4evz5ndye3)

[**F. Visualisation des données 13**](#_heading=h.6iilj0es0kdw)

[**G. Prédiction et segmentation des données 14**](#_heading=h.n4yzvgvfsr8x)

[**III. Choix des technologies 15**](#_heading=h.9xc5cjx4e7t2)

[**V. Architecture applicatif 22**](#_heading=h.ej1k7mfx976n)

[**VI. Conclusion 26**](#_heading=h.yet4x1tyqjnj)

[**VII. Perspectives 27**](#_heading=h.nxm10vqriwyt)

# **Contexte et Objectifs**

Le projet s’inscrit dans un cadre global où les données environnementales occupent une place essentielle dans la compréhension des phénomènes climatiques, sanitaires et écologiques. Face à ces enjeux, TotalGreen, entreprise française spécialisée dans les énergies renouvelables, souhaite créer un laboratoire de recherche — GoodAir — consacré à l’étude de la qualité de l’air et de l’eau en France. Ce laboratoire, réunissant climatologues, data analyst et data ingénieur, devra s’appuyer sur une base de données environnementales abondante, fiable et régulièrement actualisée.

Dans une logique de rationalisation des coûts et de gain de temps, le projet repose sur une stratégie d’exploitation de données ouvertes. Plutôt que d’installer un réseau de capteurs physiques, GoodAir s’appuiera sur des sources de données existantes via des APIs publiques, notamment celles d’OpenWeatherMap et de AQICN, permettant l’accès à des données météorologiques et de qualité de l’air à haute résolution.

L’objectif principal est donc de concevoir une plateforme capable de collecter, stocker et sécuriser ces données, tout en en facilitant le traitement, l’analyse et la visualisation. Cela implique la mise en place d’un pipeline complet, allant de collecte automatique des données jusqu’à leur restitution via des outils de visualisation interactifs et accessibles. Une attention particulière sera accordée à la qualité, à l’intégrité et à la traçabilité des données, afin de permettre des analyses fiables, tant historiques que prédictives, soutenant ainsi les activités de recherche et d’aide à la décision.

# **Étapes du Projet**

# 

# **Sécurité de l’information (RGPD)**

Afin de garantir la qualité et la conformité du traitement des données, celui-ci doit être encadré par des normes juridiques strictes, notamment le Règlement Général sur la Protection des Données (RGPD). Dans le cadre de l’analyse des données relatives à la qualité de l’air, une étude de conformité a été conduite à partir du dictionnaire des données. Celle-ci révèle l’absence de toute donnée à caractère personnel, directe ou indirecte (telles que le nom, le prénom, l’adresse IP ou tout identifiant relatif à une personne physique). Les données traitées se limitent à des indicateurs environnementaux (tels que l’AQI, les concentrations de particules PM2.5 et PM10), à des identifiants techniques de stations (idx), ainsi qu’à des informations agrégées (data, status). Le fondement juridique du traitement repose sur l’intérêt légitime, en lien avec des objectifs de surveillance environnementale, et peut s’inscrire, le cas échéant, dans le cadre d’une mission d’intérêt public lorsqu’il est effectué par une autorité ou une collectivité. Le principe de minimisation est respecté, les données collectées étant strictement nécessaires à la finalité poursuivie. Ces finalités sont clairement définies : surveillance de la qualité de l’air, diffusion d’indicateurs à destination des citoyens, et appui à la décision publique en matière environnementale. En matière de conservation, il est recommandé d’adapter la durée de stockage aux seuls besoins opérationnels, en conservant les données agrégées aussi longtemps que nécessaire et en supprimant les données brutes lorsqu’elles ne sont plus utiles.

D’un point de vue sécuritaire, l’analyse s’appuie sur les recommandations formulées par l’Agence nationale de la sécurité des systèmes d'information (ANSSI). L’accès aux données s’effectue via une interface API protégée par un système de token, dont la sécurisation et le renouvellement périodique doivent être garantis. Il est par ailleurs conseillé de mettre en œuvre une gestion des accès fondée sur les rôles (RBAC) pour les utilisateurs internes. L’intégrité des données est assurée par la journalisation des accès et des extractions, ainsi que par l’utilisation de fonctions de hachage (comme SHA-256) permettant de vérifier la cohérence des fichiers. L’hébergement des données doit être réalisé dans des infrastructures sécurisées, idéalement situées au sein de l’Union européenne, bénéficiant de dispositifs de protection adaptés (pare-feu, antivirus, mises à jour régulières). Les échanges de données doivent être chiffrés via le protocole HTTPS, en évitant toute transmission en clair. La supervision du système ainsi que la mise en place d’audits réguliers et de mécanismes d’alerte en cas de comportement anormal permettent de renforcer la robustesse du dispositif. Enfin, selon le modèle D/I/C/PAT (Disponibilité, Intégrité, Confidentialité, Preuve/Auditabilité/Traçabilité), il est essentiel de prévoir un plan de continuité d’activité (PCA) et un plan de reprise d’activité (PRA) afin de garantir la résilience du système en cas d’incident.

# **Modélisation des données**

Ci-dessous le schéma du Data Lake :



Ce datalake a été conçu pour centraliser de façon souple et scalable l’ensemble des données externes relatives à la qualité de l’air et à la météo, avant toute transformation métier.

**Pourquoi un datalake ?**

* **Flexibilité** : en stockant brut “à la pipeline” (JSON/API) sans imposer de schéma figé, on peut facilement absorber de nouveaux champs ou sources sans interrompre le flux.
* **Évolutivité** : stockage append-only, optimisé pour ingérer de gros volumes d’appels API à haute fréquence, tout en garantissant performance et montée en charge.
* **Reproductibilité** : chaque entrée porte un timestamp d’ingestion, ce qui permet de rejouer intégralement les pipelines ELT sur n’importe quelle période, pour audits ou corrections.

**Quels types de données ?**

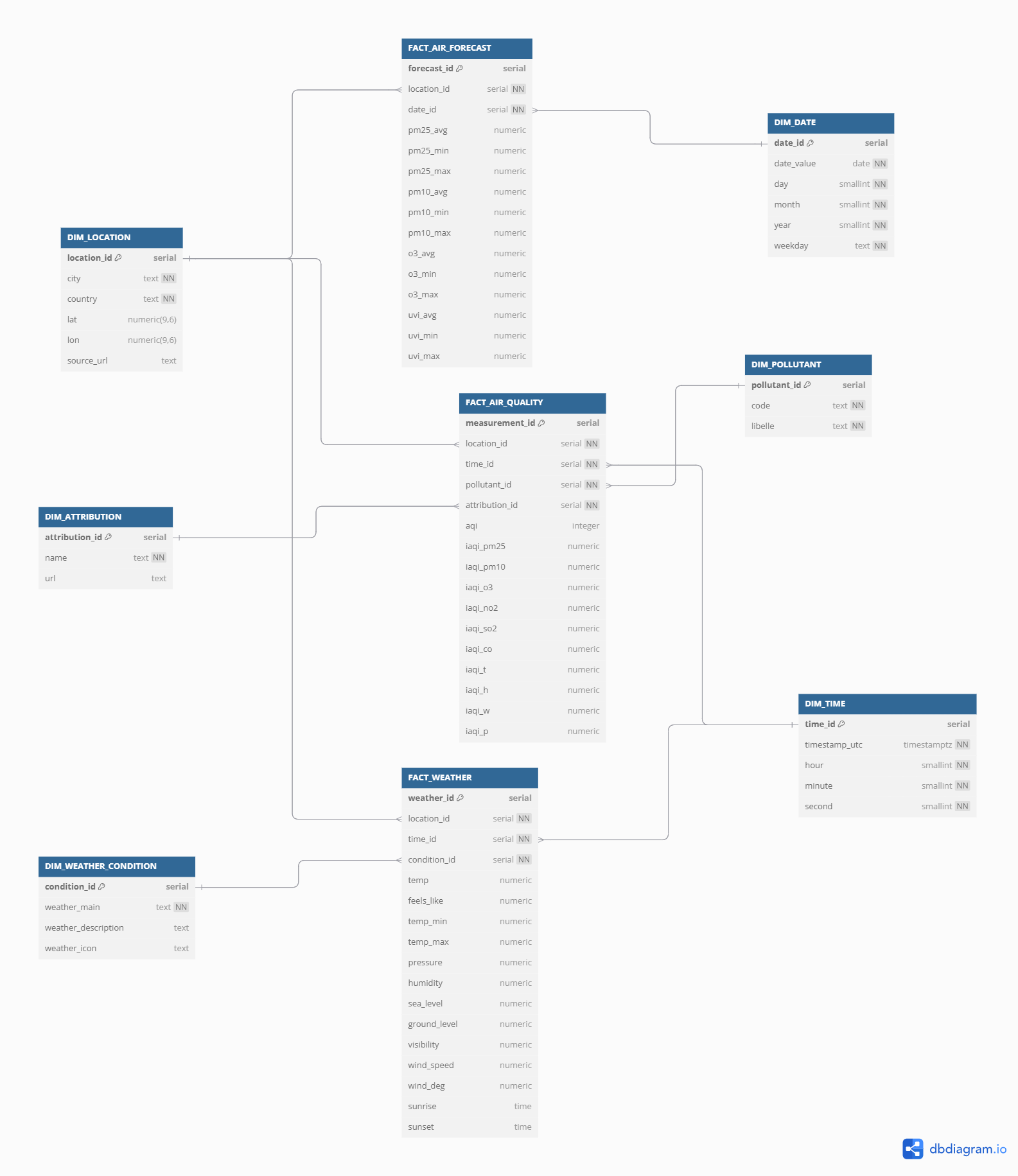
1. **Qualité de l’air instantanée** (AQI et composantes IAQI) pour suivre en temps réel les polluants majeurs.
2. **Prévisions de pollution journalières** (PM₂.₅, PM₁₀, O₃, UVI) pour planifier la prévention santé et environnementale.
3. **Relevés météo courants** (température, pression, vent, humidité, visibilité, etc.) pour contextualiser les variations de qualité de l’air.

**Comment ça marche ?**

* **Landing zone immuable** : on n’édite jamais les lignes en raw, on ne fait qu’ajouter, assurant intégrité et traçabilité.
* **Schéma-on-read** : tout est capturé en VARCHAR, chaque clé JSON trouve sa colonne, sans risques de casting ou de rupture. Le typage (FLOAT, DATE, INT) et la validation se font ensuite dans des couches « bronze/silver ».
* **Segmentation fonctionnelle** : chaque catégorie de données (air, prévisions, météo) a sa propre table et sa propre fréquence d’ingestion, simplifiant la maintenance et permettant d’appliquer des SLA distincts.

En haute vue, ce datalake offre un socle stable et adaptable, prêt à alimenter des couches de nettoyage, d’enrichissement et des tableaux de bord analytiques, tout en garantissant traçabilité et agilité face à l’évolution des APIs.

Ci-dessous le schéma du Data Warehouse :



Ce Data Warehouse est organisé selon un schéma en étoile, optimisé pour l’analyse et la génération de rapports.

### **1. Architecture « fact-dimension »**

* **Tables de faits**
  + **Fact\_Air\_Quality** : enregistrements horodatés de qualité de l’air, granularité heure × polluant × emplacement.
  + **Fact\_Weather** : relevés météo détaillés (température, pression, vent, humidité…) à l’heure, par emplacement.
  + **Fact\_Air\_Forecast** : prévisions journalières de pollution (PM 2.5, PM 10, O₃, UVI) par date et par lieu.
* **Tables de dimensions** (conformes et réutilisables)  
  + **DIM\_Location** : données géographiques (ville, pays, coordonnées, source API).
  + **DIM\_Date** et **DIM\_Time** : dates et heures décomposées (année, mois, jour, heure, minute…) pour faciliter le slicer/dicer temporel.
  + **DIM\_Pollutant** : catalogage des différents polluants (code, libellé).
  + **DIM\_Attribution** : origine ou agence ayant fourni la mesure (nom, URL).
  + **DIM\_Weather\_Condition** : typologie des états météo (ensoleillé, pluvieux, nuageux, etc.).

### **2. Pourquoi un data warehouse ?**

* **Performance** : structures indexées et typées (NUMERIC, DATE, TIME) pour des agrégations rapides, sans coût de parsing JSON.
* **Clarté du modèle** : séparations nettes entre faits (mesures numériques) et dimensions (contextes métiers), facilitant la lecture et la maintenance des requêtes SQL.
* **Conformité & Réutilisation** : dimensions partagées assurent la cohérence (une même date ou un même lieu ne sont jamais dupliqués).
* **Support BI/Reporting** : schéma en étoile prêt pour cubes OLAP, outils de reporting et dashboards, avec des grains bien définis (heure vs. jour).

### **3. Flux ELT**

1. **Extraction** : on puise dans la couche RAW du datalake (JSON bruts).
2. **Transformation / Bronze→Silver** : on cast, nettoie et normalise – conversion en types natifs, géocodage, dédoublonnage.
3. **Chargement** : on remplit les dimensions puis, grâce à leurs clés substituts (surrogate keys), les tables de faits.
4. **Aggrégations Gold** : vues matérialisées ou tables d’agrégats (moyennes glissantes, seuils AQI, comparatifs journaliers) directement consommables par les utilisateurs finaux.

En synthèse, ce data warehouse offre un **modèle optimisé pour l’analyse** : grain clair, dimensions standardisées, données typées et facilement exploitables par les outils BI.

# **Récolte des données**

La première étape du pipeline consistera à interroger périodiquement les API d’AQICN et d’OpenWeatherMap. Cette collecte sera orchestrée via le langage Python. Nous paramétrerons des workflows qui déclencheront des appels API à des horaires définis, tout en respectant les quotas imposés par les offres gratuites.

Chaque requête renverra un jeu de données au format JSON, incluant des mesures horodatées (température, humidité, taux de CO2, PM10, etc.) pour une liste de villes définie. Les réponses seront stockées temporairement dans un environnement de type Data Lake, en attendant leur traitement ultérieur.

# **Stockage des données**

Le stockage des données brutes se fera dans un Data Lake, qui constituera un point d’entrée flexible pour tous les flux non structurés ou semi-structurés issus des sources externes. Ce lac de données jouera un rôle fondamental dans l’architecture globale, car il permettra de centraliser l’ensemble des informations collectées, qu’elles soient exploitables immédiatement ou destinées à des traitements ultérieurs. Chaque enregistrement sera rigoureusement horodaté, géolocalisé et catégorisé par type de mesure afin de permettre une organisation efficace et une historisation fiable. Ces données, conservées sans transformation initiale, offriront la possibilité d’effectuer des analyses exploratoires a posteriori, d’alimenter des modèles de machine learning, ou de revenir sur les données originales en cas de besoin d’audit ou de recalcul. Le Data Lake permettra également de faire coexister différents formats de fichiers (JSON, CSV, Parquet), assurant ainsi une grande souplesse dans la gestion des flux hétérogènes et facilitant les étapes d’intégration vers les outils en aval.

En parallèle, un entrepôt de données structuré, de type Data Warehouse, sera mis en place pour permettre des traitements analytiques rapides et fiables. Cet entrepôt constituera le cœur décisionnel de la plateforme : il accueillera les données transformées et consolidées, prêtes à être interrogées par les experts métiers. Pour cela, nous nous appuierons sur une base relationnelle robuste, dimensionnée pour gérer efficacement des volumes croissants de données tout en assurant une latence minimale dans les requêtes. La modélisation adoptée reposera sur un schéma en étoile, dans lequel les faits (mesures de pollution, enregistrements climatiques) seront reliés à des dimensions (temps, localisation, type de capteur). Ce type de structuration permettra non seulement d’optimiser les performances de requêtage mais aussi de faciliter la création de vues personnalisées pour chaque usage métier. Qu’il s’agisse de suivre l’évolution des concentrations de particules fines au fil des mois, de comparer les niveaux de pollution entre différentes agglomérations, ou de générer des tableaux croisés dynamiques sur les épisodes extrêmes, l’entrepôt offrira une base solide pour la valorisation des données environnementales.

# **Intégration des données**

Les données collectées devront être nettoyées et validées avant d’être exploitées. Cette étape sera cruciale pour éviter les erreurs de calcul ou d’analyse. Elle inclura plusieurs sous-tâches : vérification des types, gestion des valeurs nulles, détection d’outliers, harmonisation des unités, etc.

Nous prévoirons un système de règles conditionnelles pour traiter les cas ambigus : par exemple, une température aberrante ou une valeur AQI manquante seront soit imputées à l’aide de valeurs historiques, soit écartées si leur taux d’anomalie dépasse un seuil tolérable. Une fois nettoyées, les données seront converties dans un format optimisé (Parquet) en vue de leur intégration dans le Data Warehouse.

Une fois les données intégrées, elles seront soumises à des transformations plus poussées dans une logique ELT (Extract, Load, Transform). Nous créerons des modèles de données adaptés aux besoins spécifiques du laboratoire : séries temporelles agrégées, tableaux de croisement ville/paramètre, moyennes mobiles, etc.

Les outils de transformation, comme dbt (Data Build Tool), permettront de versionner les scripts SQL, d’automatiser les tests de cohérence, et de documenter chaque transformation. Ce processus garantira la reproductibilité des résultats et facilitera la maintenance du pipeline dans le temps.

# **Visualisation des données**

Les résultats issus du pipeline de traitement devront être rendus accessibles et lisibles à des non-techniciens. C’est pourquoi une couche de visualisation interactive sera développée à l’aide d’outils comme Looker Studio. Cette interface permettra de générer des tableaux de bord dynamiques, filtrables par ville, par période ou par type de polluant.

Chaque tableau de bord affichera des indicateurs synthétiques, des graphiques en séries temporelles, des heatmaps géographiques ou encore des alertes visuelles. Des outils de téléchargement (CSV, PDF) seront proposés pour permettre aux chercheurs de réutiliser les données dans leurs propres outils d’analyse.

Nous prévoirons également l’édition automatique de rapports hebdomadaires ou mensuels, qui seront envoyés aux membres du laboratoire par mail ou déposés dans un espace partagé sécurisé.

# **Prédiction et segmentation des données**

L’analyse statistique constitue une étape fondamentale du projet, car elle permettra de transformer les données environnementales brutes en informations décisionnelles. Cette phase aura pour but de mettre en lumière les relations entre variables, d’identifier des anomalies, et de produire des indicateurs synthétiques exploitables par les chercheurs.

Nous commencerons par le calcul d’indicateurs clés, tels que la moyenne journalière des concentrations de polluants (PM2.5, PM10), les valeurs maximales et minimales enregistrées sur des périodes définies, ou encore les taux de dépassement des seuils réglementaires. Ces KPI seront ensuite enrichis par des calculs de tendance et de dispersion afin de mesurer la variabilité intra-journalière ou inter-ville.

Nous chercherons également à explorer les corrélations entre variables, comme celles entre humidité et pollution, ou entre vent et taux d’ozone. Des coefficients comme Pearson ou Spearman permettront de quantifier l’intensité de ces relations. Si certaines corrélations s’avèrent statistiquement significatives, elles pourraient orienter les recommandations futures du laboratoire.

Enfin, l’analyse des valeurs extrêmes permettra de détecter automatiquement des événements inhabituels : pics de pollution, canicules, inversions thermiques. Ces situations seront identifiées grâce à des méthodes comme le z-score ou les quantiles, puis archivées dans une base dédiée.

# **Choix des technologies**

Le langage Python sera utilisé comme base pour le développement de l’ensemble des scripts du projet. Il sera notamment mobilisé pour la phase d’ingestion des données, grâce à ses bibliothèques comme requests pour les appels API et json pour le traitement de la réponse. Pour le nettoyage et la transformation des données, Python offrira aussi toute la puissance de pandas et numpy, permettant de manipuler les jeux de données en mémoire avec souplesse et rapidité. Sa simplicité de syntaxe et la richesse de son écosystème scientifique en feront un allié essentiel tout au long du pipeline.

Pour le stockage des données brutes, nous mettrons en place un environnement de type Data Lake, reposant sur PostgreSQL. Ce niveau de stockage constituera la couche d’archivage principale, garantissant la traçabilité et la possibilité de relancer des analyses à partir des données initiales.

Le stockage structuré sera assuré via un entrepôt de données, en envisageant soit une instance PostgreSQL, soit une solution cloud comme Snowflake. Ces bases permettront de modéliser les données sous forme de schémas relationnels, avec des capacités d’indexation et d’optimisation de requêtes adaptées à la volumétrie croissante des données environnementales. Le modèle en étoile que nous concevrons facilitera les agrégations par dimension géographique ou temporelle.

Afin d’assurer l’intégration fluide des données du Data Lake vers le Data Warehouse, nous utiliserons la plateforme Talend. Cet outil d’intégration de données permettra de concevoir graphiquement des flux ETL robustes et réutilisables. Il facilitera l’extraction des fichiers bruts depuis le Data Lake, leur transformation selon les modèles définis (filtrage, agrégation, normalisation) puis leur chargement structuré dans l’entrepôt. Talend offrira également des options de suivi des performances et de gestion des erreurs, essentielles pour la fiabilité du pipeline.

La transformation avancée des données, dans une logique ELT, sera prise en charge par dbt, un outil spécialisé dans le versionnage et la documentation des modèles de données. En écrivant des transformations directement en SQL dans le Data Warehouse, nous pourrons maintenir un haut niveau de performance tout en assurant la clarté, la reproductibilité et la traçabilité des processus de traitement.

Pour la visualisation et la restitution des résultats, Looker Studio s’imposera comme une solution adaptée. Grâce à ses connecteurs vers des bases SQL ou des fichiers en ligne, il permettra de concevoir des dashboards interactifs et esthétiques. Ces tableaux de bord seront essentiels pour donner aux chercheurs une lecture directe des évolutions environnementales dans les villes surveillées.

Enfin, les notebooks Jupyter seront proposés comme environnement complémentaire à l’usage des analystes et chercheurs. Ils offriront un espace flexible où coder, tester et visualiser des analyses statistiques ou prédictives, tout en conservant une documentation dynamique. Ce cadre favorisera l’exploration libre des données et l’expérimentation de modèles dans un contexte semi-industriel.

En complément des outils techniques centrés sur la donnée, plusieurs outils collaboratifs seront employés pour garantir une organisation de projet structurée et transparente. Pour la gestion des tâches, nous utiliserons Trello, un outil de gestion de projet en mode kanban, permettant de suivre l’évolution de chaque étape du développement, d’attribuer les responsabilités, et de documenter les livrables intermédiaires. Ce cadre visuel simplifiera la coordination entre les membres de l’équipe, tout en offrant un suivi clair à nos encadrants pédagogiques ou commanditaires.

La conception visuelle du pipeline de traitement sera réalisée à l’aide de Miro, une plateforme de tableau blanc collaboratif en ligne. Cet outil nous permettra de construire, commenter et itérer sur le schéma de l’architecture cible, en facilitant les échanges d’idées et les validations itératives autour du flux de données. Miro offrira ainsi une représentation graphique évolutive du projet, compréhensible aussi bien par des profils techniques que non techniques.

Enfin, pour centraliser le code source, les scripts, les documents techniques et les versions successives du projet, nous mettrons en place un espace de travail sur GitHub. Ce référentiel partagé garantira à la fois la traçabilité des contributions, la gestion des versions du code, et la possibilité de travailler en parallèle tout en sécurisant les échanges. Cette organisation outillée renforcera la cohérence du projet et assurera une livraison plus fluide et plus rigoureuse.

1. **Justification des outils utiliser**

1. Python

| **Élément** | **Python** | **Concurrent : R** |
| --- | --- | --- |
| **Avantages** | Langage polyvalent, vaste écosystème, facile à apprendre | Excellente pour les statistiques et visualisation |
| **Inconvénients** | Moins performant pour certains traitements lourds | Moins polyvalent hors analyse statistique |

**Justification :**

J’ai choisi Python pour sa polyvalence et sa compatibilité avec de nombreux outils d’analyse de données, d’ETL et de visualisation. Il est aussi très bien documenté.

2. VS Code

| **Élément** | **Visual Studio Code** | **Concurrent : PyCharm** |
| --- | --- | --- |
| **Avantages** | Léger, rapide, riche en extensions | IDE complet pour Python |
| **Inconvénients** | Moins d’outils intégrés par défaut | Plus lourd, consommation de ressources |

**Justification :**VS Code offre un excellent compromis entre légèreté et puissance. Son système d’extensions me permet d'adapter l'environnement à mes besoins spécifiques.

3. Postman

| **Élément** | **Postman** | **Concurrent : Insomnia** |
| --- | --- | --- |
| **Avantages** | Interface intuitive, gestion des environnements, collaboration | Léger, open-source |
| **Inconvénients** | Peut devenir lent avec de gros projets | Moins de fonctionnalités avancées |

**Justification :**J’utilise Postman pour tester les APIs (comme celles de données environnementales). Il facilite la création de requêtes complexes et leur partage.

4. Talend

| **Élément** | **Talend Open Studio** | **Concurrent : Apache NiFi** |
| --- | --- | --- |
| **Avantages** | Interface visuelle, nombreuses connexions de sources | Temps réel, streaming performant |
| **Inconvénients** | Moins adapté au temps réel | Moins intuitif à prendre en main |

**Justification :**Talend est idéal pour créer des workflows ETL complexes sans coder, ce qui accélère la mise en œuvre du pipeline de données.

5. DBT (Data Build Tool)

| **Élément** | **DBT** | **Concurrent : Airflow + SQL Scripts** |
| --- | --- | --- |
| **Avantages** | Versionning SQL, transformations fiables, testabilité | Plus de contrôle mais plus complexe |
| **Inconvénients** | Limité aux transformations SQL | Plus de configuration requise |

**Justification :** J’ai choisi DBT pour sa capacité à gérer les transformations de données dans un cadre de développement logiciel (versionnage, documentation, test…).

6. PostgreSQL

| **Élément** | **PostgreSQL** | **Concurrent : MySQL** |
| --- | --- | --- |
| **Avantages** | Puissant, open-source, support JSON, géodonnées | Facile à configurer, plus rapide en lecture simple |
| **Inconvénients** | Moins performant sur très gros volumes sans optimisation | Moins riche en fonctionnalités avancées |

**Justification :** PostgreSQL est une base robuste et flexible, particulièrement adaptée à des traitements analytiques et à la gestion de données structurées et semi-structurées.

7. Looker Studio

| **Élément** | **Looker Studio** | **Concurrent : Power BI** |
| --- | --- | --- |
| **Avantages** | Gratuit, facile à connecter aux APIs Google, partage web | Plus d’interactivité, connecteurs variés |
| **Inconvénients** | Moins de possibilités d’interactivité avancée | Coût d’abonnement |

**Justification :** J’ai retenu Looker Studio pour sa simplicité d’usage, son intégration native avec les services Google, et sa capacité à générer des dashboards interactifs rapidement.

8. Trello

| **Élément** | **Trello** | **Concurrent : Asana** |
| --- | --- | --- |
| **Avantages** | Interface intuitive, méthode Kanban simple, gratuit | Gestion avancée de projets, intégrations |
| **Inconvénients** | Limité pour les projets complexes | Moins visuel et un peu plus rigide |

**Justification :** J’ai choisi Trello pour sa simplicité et sa clarté visuelle. Il est parfait pour organiser les tâches de manière agile en suivant l’évolution du projet étape par étape.

9. Miro

| **Élément** | **Miro** | **Concurrent : Lucidspark** |
| --- | --- | --- |
| **Avantages** | Idéal pour le brainstorming, collaboration en temps réel | Fonctionnalités similaires, bon pour le mapping |
| **Inconvénients** | Certaines fonctions avancées sont payantes | Moins de templates disponibles |

**Justification :** Miro permet une excellente collaboration visuelle à distance. C’est un outil de choix pour planifier, modéliser et conceptualiser l’architecture de données ou le pipeline.

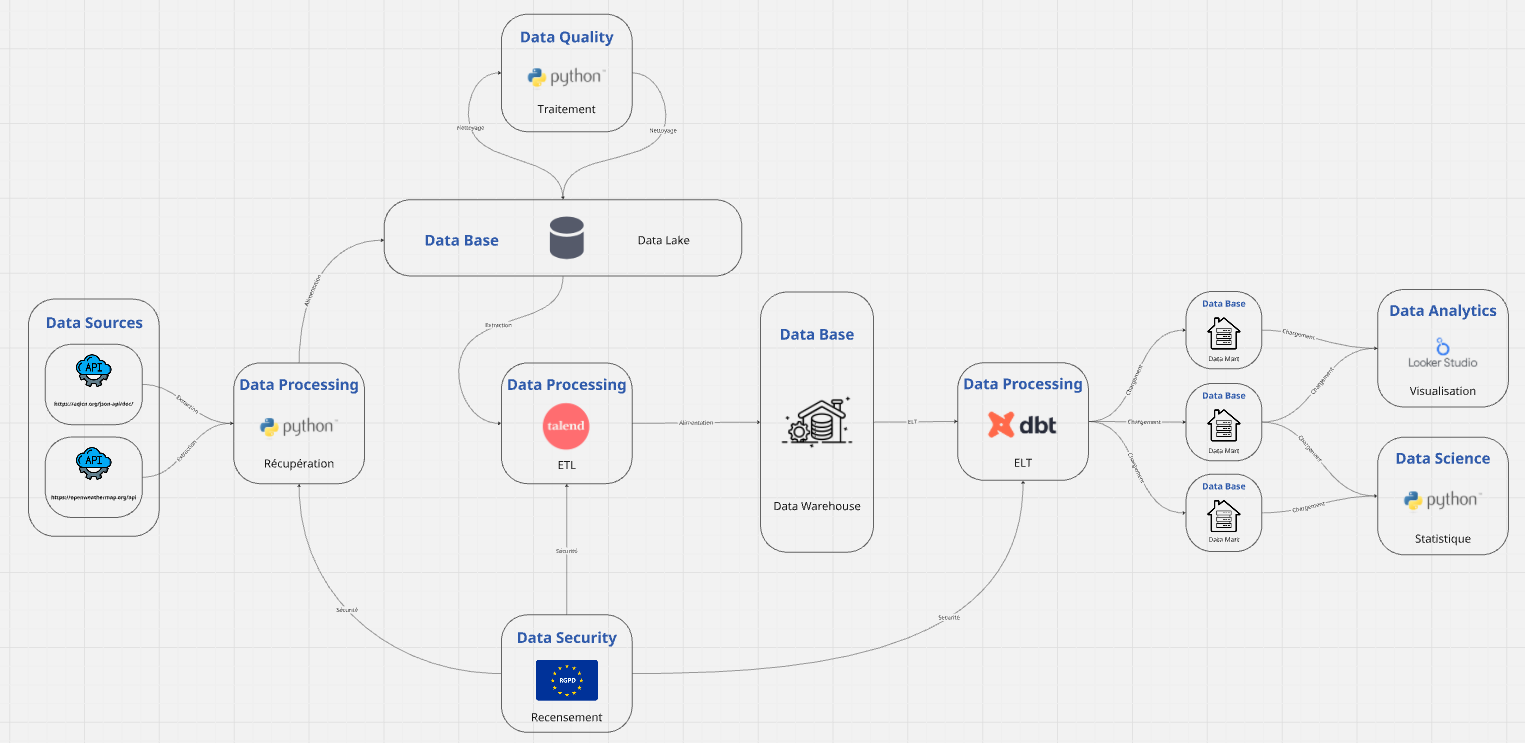
10. GitHub

| **Élément** | **GitHub** | **Concurrent : GitLab** |
| --- | --- | --- |
| **Avantages** | Leader du marché, large communauté, CI/CD intégrée | Plus de personnalisation, déploiement privé facile |
| **Inconvénients** | Fonctionnalités avancées parfois complexes | Moins répandu en entreprise |

**Justification :** GitHub est essentiel pour versionner le code du projet (Python, DBT, etc.). Sa popularité et ses intégrations le rendent incontournable.

# **Architecture applicatif**

Ci-dessous le schéma applicatif :



**Ingestion des données** Nous utilisons des scripts Python pour interroger les API externes et ingérer des fichiers plats (CSV, JSON, logs), puis déposons ces flux bruts dans un Data Lake. Cette zone de stockage centralisée conserve l’historique complet et facilite la ré-ingestion si nécessaire.

**Nettoyage et contrôle qualité** Des jobs Python scrutent les données brutes pour détecter les valeurs manquantes, les doublons et les formats incorrects, puis appliquent des règles de normalisation et des enrichissements (par exemple géocodage). Le résultat est un jeu “curated” fiable, prêt à alimenter le système en aval.

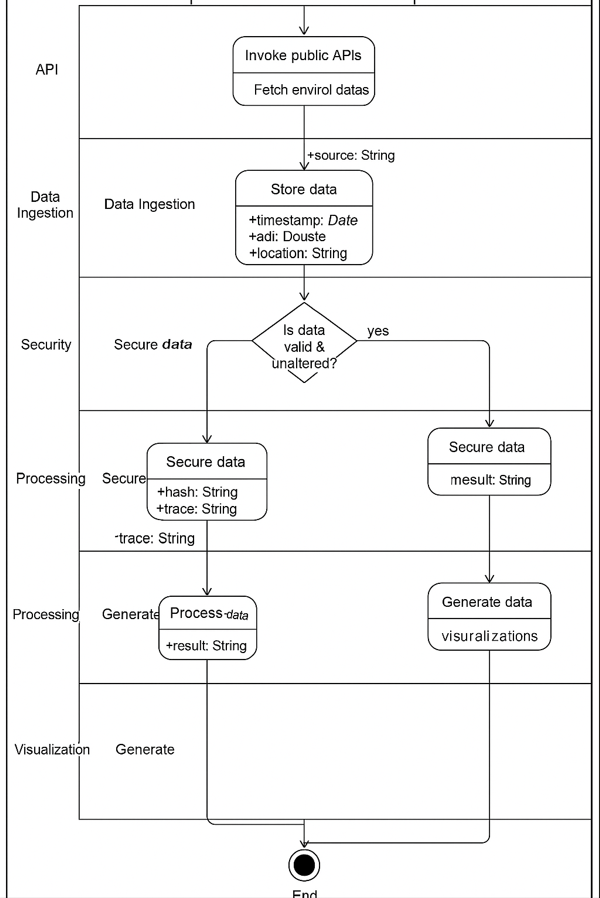
**ETL vers le Data Warehouse** Avec Talend, nous orchestrons les extractions depuis le Data Lake, réalisons des transformations lourdes (jointures, agrégations) et chargeons les tables dans votre entrepôt de données relationnelles. Talend offre une interface graphique pour piloter facilement les workflows et gérer les erreurs ou relances.

**Entrepôt central (Data Warehouse)** Le Data Warehouse est structuré selon un schéma en étoile (faites et dimensions) afin d’optimiser les requêtes analytiques. Il constitue le point de vérité unique pour l’ensemble des indicateurs clés de votre activité.

**ELT et modélisation métier** Nous utilisons dbt pour versionner et documenter les transformations SQL directement dans le Warehouse. dbt exécute des tests automatiques et génère des Data Marts thématiques (Ventes, Marketing, Finance) afin de livrer des vues métiers prêtes à consommer.

**Restitution et Data Science** Les Data Marts se connectent en direct à Looker Studio pour créer des tableaux de bord interactifs et partagés en self-service. Parallèlement, vos équipes Data Science accèdent via Jupyter/Python aux mêmes sources pour mener des analyses statistiques et prototypage de modèles.

**Gouvernance & sécurité** Tout le flux est protégé par chiffrement au repos et en transit, avec pseudonymisation des données sensibles et traçabilité des accès. Cette approche garantit la conformité RGPD et renforce la confiance de vos utilisateurs.



Le diagramme d’activité UML ci-dessus illustre le déroulement opérationnel du projet GoodAir, de manière séquentielle et structurée. Il permet de visualiser les différentes étapes du traitement des données environnementales, tout en identifiant les acteurs impliqués à chaque niveau du processus.

Le processus débute par l’**initiation du projet**, marquant la mise en place des objectifs et des ressources. Dans une logique d’exploitation de données ouvertes, les informations sont ensuite **collectées via des APIs publiques** telles qu’OpenWeatherMap ou AQICN. Ces données brutes sont ensuite **stockées dans une infrastructure centralisée**, garantissant leur accessibilité et leur archivage.

Une phase de **sécurisation des données** est réalisée, principalement par le data analyst, afin d’assurer leur intégrité, leur traçabilité et leur conformité aux exigences réglementaires. Les données sont ensuite **traitées** par le data engineer : cela comprend le nettoyage, la normalisation et la structuration des jeux de données pour les rendre exploitables. Vient ensuite l’étape d’**analyse**, qui peut prendre la forme d’analyses statistiques ou prédictives, selon les besoins du laboratoire.

Enfin, les résultats sont **visualisés** à travers des outils interactifs, à destination des climatologues et autres chercheurs, facilitant l’interprétation des données et soutenant la prise de décision scientifique et environnementale.

Ce diagramme offre ainsi une vue synthétique et rigoureuse du pipeline de données mis en place par GoodAir, en soulignant la complémentarité des rôles techniques et scientifiques tout au long du processus.

# **Conclusion**

La plateforme GoodAir s’appuie sur une architecture conçue pour répondre à des exigences élevées en matière de performance, de fiabilité et d’adaptabilité. Grâce à un pipeline de données automatisé et modulaire, elle garantit une collecte régulière, un traitement structuré et un stockage sécurisé des données environnementales. L’usage de technologies modernes, largement éprouvées dans le monde de la donnée, confère au projet une stabilité essentielle tout en assurant une conformité avec les normes réglementaires en vigueur.

Mais au-delà de sa fonction actuelle, la plateforme est pensée comme une base évolutive. Elle ouvre la voie à des développements futurs, tels que l’intégration de modèles d’intelligence artificielle, l’analyse prédictive ou encore l’enrichissement avec de nouvelles sources de données. Elle constitue donc un socle robuste, à la fois pour la recherche scientifique, pour les analyses stratégiques et pour l’aide à la décision en matière de santé publique et d’environnement.

# **Perspectives**

Dans la continuité du développement, une prochaine étape clé consistera à mettre en place une phase de testing structurée. Cette phase visera à :

* Valider la fiabilité des flux de données (détection d’anomalies, vérification de la fraîcheur des données)
* Tester la résilience de la plateforme face à des interruptions ou des pics de charge
* Évaluer la précision des visualisations et tableaux de bord dans différents scénarios d’usage
* Recueillir les retours des utilisateurs (chercheurs, analystes, partenaires) pour affiner l’ergonomie et les fonctionnalités

Ce processus de test permettra non seulement de fiabiliser l’environnement technique, mais aussi de s’assurer que la plateforme répond efficacement aux besoins concrets des parties prenantes. Il s’agira également d’un tremplin pour tester les premières briques d’IA prédictive, en se basant sur des cas d’usage ciblés (par exemple : prédiction de pics de pollution ou corrélation entre météo et qualité de l’air).