



# RAPPORT FINAL

Projet TPI

## RESUME

Production d'un outil pour la génération de Dashboard sur des données massives liées à l'analyse et l'optimisation des performances opérationnelles dans un contexte d'excellence opérationnelle

Manar AFIFI

Arnold perez FAHO MBATCHOU

Batiste GROS

## Table des matières

<b>1 Cahier des charges</b>	3
<b>2 Etat de l'art :</b>	5
<b>3 Solutions existantes de type commercial</b>	5
<b>3.1. Tableau</b>	5
<b>3.2. Power BI</b>	6
<b>3.3. Qlik Sense</b>	7
<b>4 Solutions existantes de type Open Source</b>	9
<b>4.1. Grafana</b>	9
.....	9
<b>4.2. Metabase</b>	10
<b>4.3. Apache Superset</b>	12
<b>5 Résultat de l'état de l'art 1</b>	15
<b>6 Cahier de charges pour centraliser les données</b>	16
<b>7 Solutions pour les ETL</b>	17
<b>7.1. Airbyte</b>	17
<b>7.2. Talend</b>	18
<b>7.3. Fivetran</b>	19
<b>8 Comparaison des bases de données</b>	20
<b>9 Résultat de l'état de l'art 2</b>	21
<b>10 Choix de la réalisation</b>	21
<b>10.1 Spring Boot pour le Backend</b>	21
<b>10.2 PostgreSQL (Base de données)</b>	22
<b>10.3 Metabase (intégration de visualisations)</b>	22
<b>10.4 React pour le frontend</b>	23
<b>10.5 Airbyte : une solution moderne et modulaire pour l'intégration de données</b>	24
<b>10.6 Docker et Docker Compose : une infrastructure standardisée et scalable</b>	24
<b>11 Résultat du développement</b>	25
<b>11.1 Présentation de l'interface</b>	25
<b>11.2 Petite démo représentant quelques fonctionnalités :</b>	25

<b>12 Difficultés rencontrées.....</b>	<b>32</b>
<b>13 Architecture du projet.....</b>	<b>34</b>
<b>14 Conclusion.....</b>	<b>35</b>

## 1 Cahier des charges

Catégorie	Sous-catégorie	Description
Contexte du Projet	–	Développement d'une plateforme de Dashboard pour l'analyse et l'optimisation dans l'aéronautique et la maintenance.
	Objectif principal	Fournir une solution pour analyser les données et optimiser les performances opérationnelles.
	Secteurs concernés	Aéronautique, Maintenance industrielle.
Utilisateurs : Parties prenantes	Administrateurs	Gestion globale : utilisateurs, permissions, configuration de la plateforme.
	Les ingénieurs/ Techniciens	Suivi des tâches de maintenance et visualisation des données.
	Managers	Analyse des performances via des KPI pour la prise de décision stratégique.
Fonctionnalités Principales	Gestion des données	Intégration de données issues de capteurs IoT, bases SQL/NoSQL, Systèmes ERP
	Personnalisation	Modification des Dashboard : widgets, filtres, plages de données, des options de personnalisations pour les couleurs, polices, arrière-plan des Dashboard.
	Visualisation	Graphiques dynamiques : camemberts, histogrammes, cartes, lignes, Kanban, etc.
	Alertes et notifications	Notifications automatiques sur dépassement de seuils critiques.

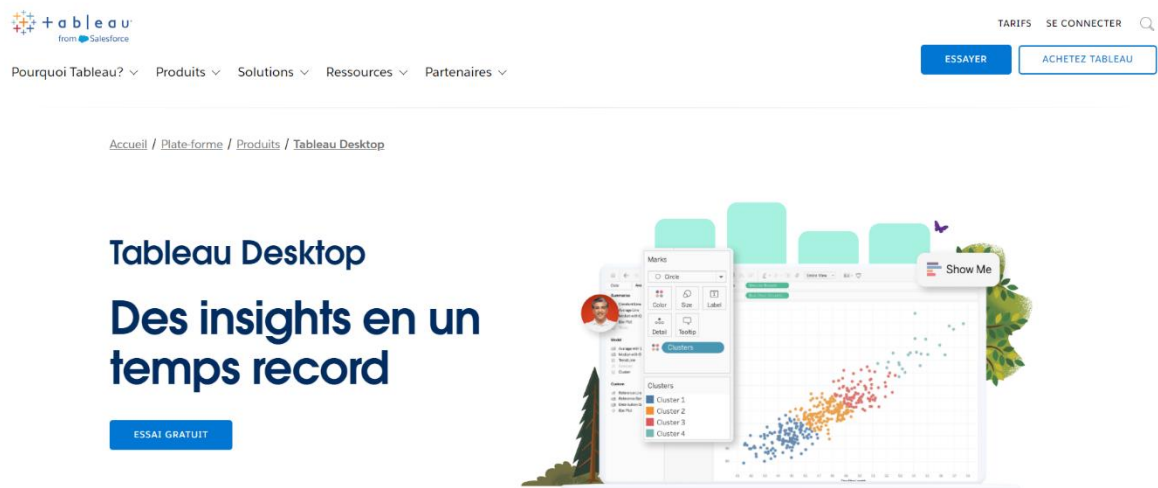
	Rapports automatisés	Génération périodique de rapports en PDF ou Excel.
	Partage	Partage de Dashboard avec des équipes ou utilisateurs spécifiques.
	Traçabilité	Suivi des modifications des actions des utilisateurs.
Contraintes Techniques	Accessibilité	Compatibilité avec les ordi, mobile, et tablette.
	Temps de réponse	Chargement des Dashboard en moins de Ts secondes.
Sécurité	Confidentialité des données	Chiffrement des données sensibles en transit et au repos.
	Gestion des accès	Contrôle basé sur les rôles des utilisateurs pour limiter les permissions.
Expérience	Ergonomie	Interface intuitive pour utilisateurs techniques et non techniques.

## 2 Etat de l'art :

Du cahier des charges ci-dessus, nous allons tirer les principaux critères que nous utiliserons pour évaluer les différentes solutions techniques existantes. Nous utiliserons la comparaison de ces critères pour établir un classement des solutions.

## 3 Solutions existantes de type commercial

### 3.1. Tableau



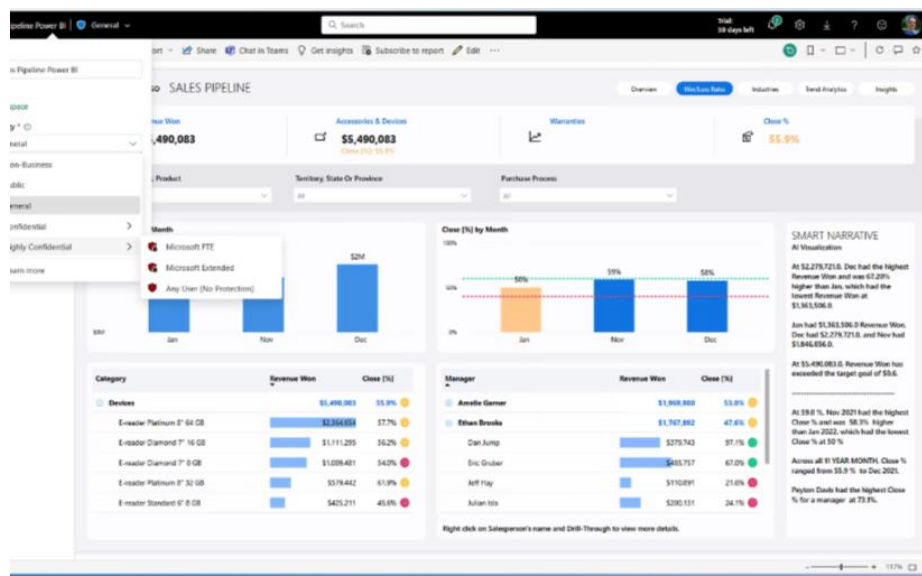
Critères	Notes/5	Description
Utilisateurs à différents niveaux d'accès	4	Gestion complète des rôles et permissions par utilisateur.
Gestion des données	5	Intégration multi-source : SQL, Cloud, application dans le cloud comme Salesforce. Nettoyage de donnée sans avoir besoin de coder.
Personnalisation	4,5	Création rapide des calculs puissants à partir de données existantes. L'IA est intégré pour faciliter la prise de décision ... Des insights approfondis plus rapidement
Visualisation	5	Les data scientists peuvent intégrer et visualiser les résultats obtenus avec R, Python, MATLAB et d'autres extensions, pour déployer leurs modèles à l'échelle de l'organisation.

		Graphiques dynamiques avancés : camemberts, cartes, kanban, histogrammes, etc
Alertes et notifications	4	Alertes configurables en cas de dépassement de seuils critiques.
Rapports automatisés	5	Export en PDF et Excel, génération automatique intégrée.
Partage	5	Partage collaboratif via Tableau Server/Cloud.
Traçabilité	4	Suivi des modifications et actions utilisateur via journaux d'activité.
Accessibilité	5	Compatible avec desktop, mobile et tablette.
Temps de réponse	4	Temps de réponse rapide pour données de taille modérée.
Confidentialité des données	5	Chiffrement intégré des données en transit et au repos.
Gestion des accès	5	Contrôle avancé des permissions basé sur les rôles utilisateurs.
Ergonomie	5	Interface intuitive et facile à prendre en main, adaptée aux même des débutants.

## 3.2. Power BI



# Power BI



Critères	Notes/5	Description
Utilisateurs à différents niveaux d'accès	4	Gestion des permissions native via Azure Active Directory.
Gestion des données	4	Intégration avec SQL, NoSQL et autres outils de l'écosystème Microsoft.
Personnalisation	4,5	Options de personnalisation modérées avec des styles prédéfinis.
Visualisation	4	Options de visualisation complètes mais moins flexibles que Tableau.
Alertes et notifications	5	Notifications configurables via Power Automate pour les seuils critiques.
Rapports automatisés	5	Rapports Excel natifs, export PDF via plugins tiers.
Partage	4,5	Intégration native avec Microsoft Teams et SharePoint pour partage.
Traçabilité	5	Journaux intégrés pour suivi des modifications et actions utilisateur.
Accessibilité	4	Compatible avec desktop et mobile, support natif limité pour tablette.
Temps de réponse	4	Performances optimales dans l'écosystème Microsoft Azure.
Confidentialité des données	5	Sécurisé avec chiffrement, conforme aux normes Microsoft.
Gestion des accès	5	Gestion centralisée via Azure AD.
Ergonomie	4	Convient aux utilisateurs familiers avec Microsoft Office.

### 3.3. Qlik Sense







Critères	Notes/5	Description
Utilisateurs à différents niveaux d'accès	4	Gestion des rôles et permissions via l'administration de Qlik Management Console.
Gestion des données	5	Intégration multi-source : SQL, NoSQL, fichiers plats, ERP, et cloud.
Personnalisation	4	Bien que Qlik Sense offre de nombreuses options, la personnalisation graphique est plus limitée que Tableau.
Visualisation	5	Visualisation dynamique avec exploration associative et graphiques interactifs.
Alertes et notifications	4	Disponible via l'extension Qlik Alerting ou via API pour alertes avancées.
Rapports automatisés	4	Intégration native avec NPrinting, mais nécessite une licence séparée.
Partage	4	Les utilisateurs peuvent partager leurs analyses facilement dans Qlik Cloud.
Traçabilité	5	Qlik Sense fournit une bonne gestion des logs utilisateurs dans la console d'administration.
Accessibilité	5	Les dashboards sont parfaitement adaptés à différents appareils sans configuration supplémentaire.
Temps de réponse	5	Le moteur Qlik est conçu pour offrir des performances élevées même avec des datasets volumineux.
Confidentialité des données	5	Conforme aux normes de sécurité modernes, adapté aux grandes entreprises.
Gestion des accès	5	La gestion fine des accès est un atout majeur de Qlik Sense.
Ergonomie	3	Interface intuitive, mais nécessite une formation pour exploiter toutes les fonctionnalités.

## 4 Solutions existantes de type Open Source

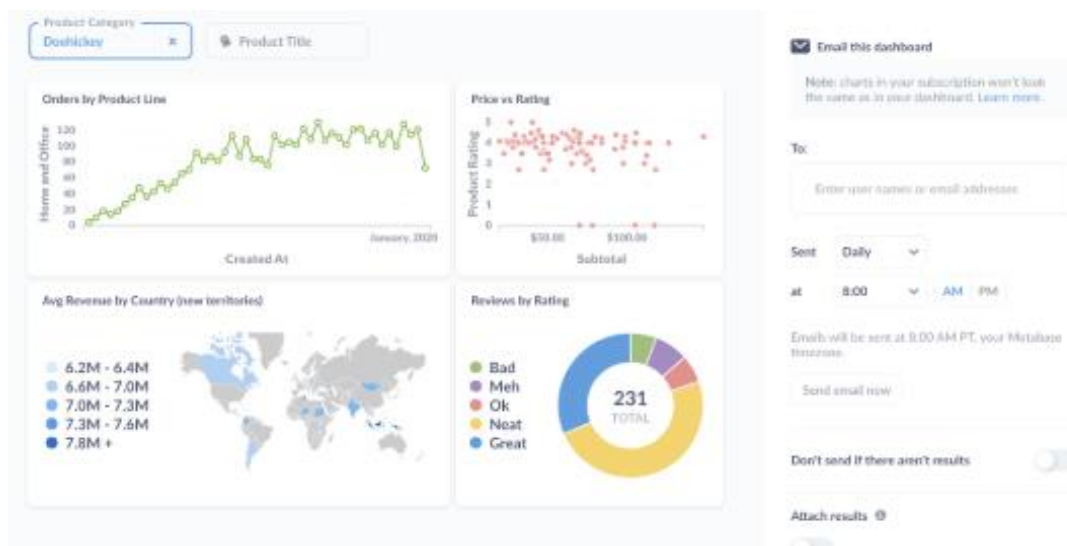
### 4.1. Grafana

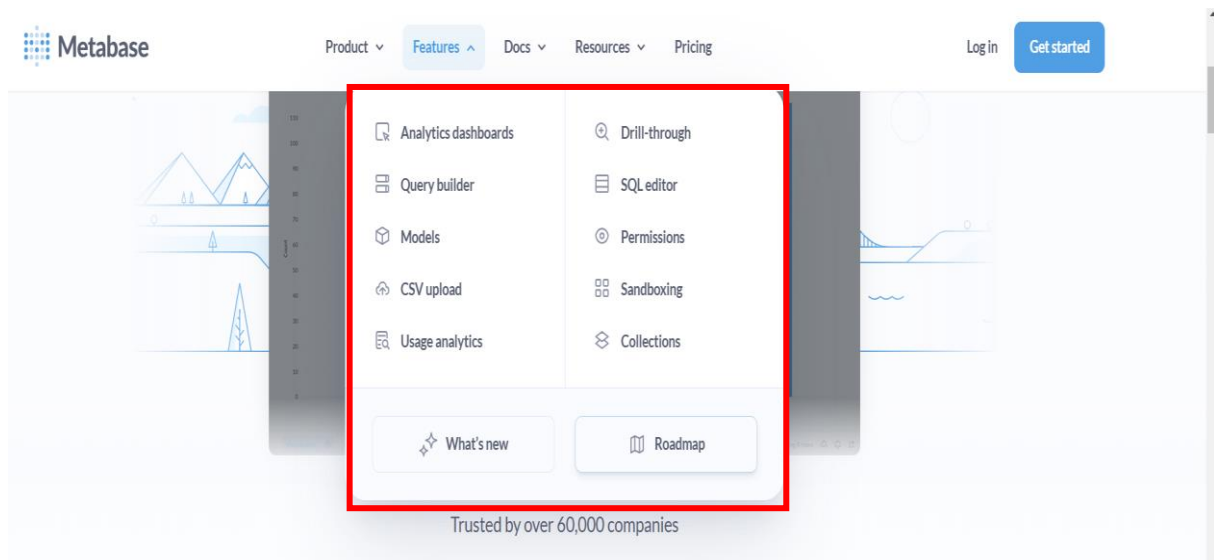


Critères	Notes/5	Description
Utilisateurs à différents niveaux d'accès	3	Gestion des rôles basique avec permissions définies pour chaque utilisateur.
Gestion des données	4	Intégration native avec des bases SQL, NoSQL, IoT, et autres sources externes.
Personnalisation	3	Options limitées pour personnaliser l'apparence graphique (thèmes prédéfinis).
Visualisation	5	Graphiques interactifs pour monitoring : barres, lignes, heatmaps, cartes. Excellente pour des analyses en temps réel avec une visualisation fluide et performante.
Alertes et notifications	5	Alertes robustes configurables (email, Slack, PagerDuty, Webhooks, etc.).

Rapports automatisés	3	Export Grafana Enterprise supporte des rapports PDF, mais c'est payant et limité dans la version open source.
Partage	3	Partage limité à des URL ou des instantanés statiques de dashboards.
Traçabilité	3	La traçabilité est limitée et dépend souvent de l'infrastructure sous-jacente.
Accessibilité	4	Interface responsive adaptée à desktop, mobile via navigateur.
Temps de réponse	5	Optimisé pour le monitoring en temps réel, même avec de grandes quantités de données.
Confidentialité des données	3	Grafana ne gère pas directement la sécurité des données, mais peut s'appuyer sur TLS et les systèmes externes.
Gestion des accès	3	Fonctionnalités de gestion des accès limitées pour des cas d'utilisation avancés.
Ergonomie	4	Grafana est adapté aux experts techniques, mais peut être intimidant pour les débutants.

## 4.2. Metabase

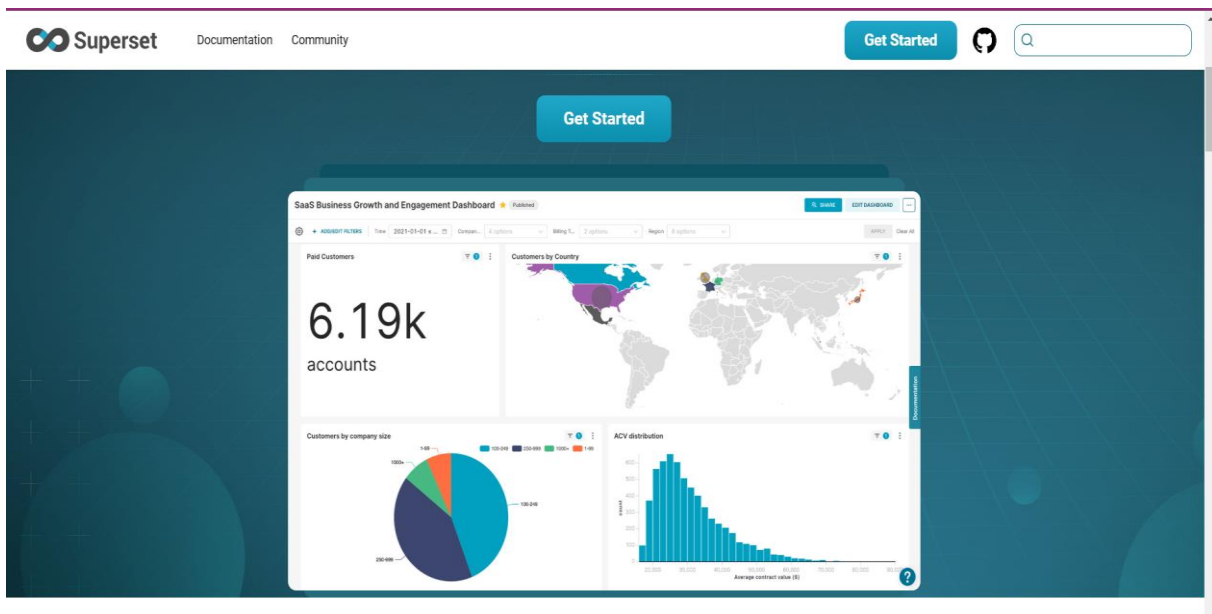




Critères	Notes/5	Description
Utilisateurs à différents niveaux d'accès	3	Gestion basique des rôles : Administrateur, Utilisateur, et Visualisateur.
Gestion des données	5	Metabase supporte les bases courantes comme MySQL, PostgreSQL, MongoDB, etc., mais est limité pour les systèmes distribués
Personnalisation	3	Personnalisation limitée aux champs visibles et filtres interactifs : Pas de contrôle avancé sur l'apparence ou les couleurs des visualisations.
Visualisation	4	Supporte des graphiques simples : barres, lignes, tableaux, cartes, etc : Convient aux besoins de visualisation de base, mais manque d'options avancées comme Kanban ou heatmaps.
Alertes et notifications	3	Alertes configurables basées sur des seuils ou conditions simples : Notifications disponibles par email, mais pas de support intégré pour Slack ou Webhooks.
Rapports automatisés	4	Export de graphiques ou tableaux en CSV, Excel, ou PDF : Rapports basiques programmables, mais fonctionnalités limitées pour des besoins avancés.
Partage	4	Les dashboards et questions peuvent être partagés via des liens ou exports.
Traçabilité	3	Pas d'audit avancé, dépend des logs système pour des suivis détaillés.

Accessibilité	4	Bien adapté à desktop et mobile, mais pas d'application dédiée pour les appareils mobiles.
Temps de réponse	4	Performances acceptables pour des bases relationnelles de taille modérée : Convient pour des datasets petits à moyens, mais ralentit avec de grands volumes de données.
Confidentialité des données	3	Pas de sécurité native comme le chiffrement des données, nécessite TLS externe.
Gestion des accès	3	Peut gérer l'accès par projet ou par utilisateur, mais reste simpliste.
Ergonomie	5	Interface conviviale, facile à utiliser même pour les non-techniciens.

## 4.3. Apache Superset



The screenshot shows the Superset web interface. On the left, there's a sidebar with 'DATABASE' (gsheets), 'SCHEMA' (main), and 'SEE TABLE SCHEMA' (SaaS Product Sample Data). The main area displays a SQL query editor with a query that filters data from 'main.SaaS Product Sample Data' based on 'first\_open' and 'first\_open' dates. Below the query editor, there's a 'RUN' button and a 'LIMIT: 1000' dropdown. The results section shows a table with 100 rows returned, including columns like 'user', 'first\_open', 'Year', 'Month', 'dayofweek', 'Total users', 'Company size', and 'screen\_list'.

The screenshot shows the 'Supported Databases' page in the Superset interface. It features a grid of logos for various databases and data sources, including PostgreSQL, Google Big Query, snowflake, MySQL, amazon REDSHIFT, amazon ATHENA, druid, databricks, CSV, ClickHouse, ROCKSET, dremio, trino, ORACLE DATABASE, pinot, presto, IBM DB2, SAP HANA, and SQL Server.

Critères	Notes/5	Description
Utilisateurs à différents niveaux d'accès	5	Gestion des rôles avancée via RBAC (Role-Based Access Control).
Gestion des données	5	Supporte des connecteurs variés : MySQL, PostgreSQL, Presto, BigQuery, etc.
Personnalisation	4,75	Support de thèmes et graphiques personnalisés, mais nécessite parfois du code.
Visualisation	5	Large choix de graphiques interactifs : barres, lignes, cartes, diagrammes, etc.
Alertes et notifications	4,5	Intégration possible avec Slack, Email, ou Webhooks pour des notifications avancées.
Rapports automatisés	3	Fonctionnalités basiques d'automatisation des rapports, pas aussi riches que Tableau.
Partage	4	Partage des dashboards via des liens publics ou accès restreint.

Traçabilité	3,5	La traçabilité est limitée par défaut et nécessite une configuration supplémentaire.
Accessibilité	4	Accessible via navigateur web avec une interface responsive.
Temps de réponse	4	Optimisé pour des bases SQL et data lakes de taille moyenne à grande.
Confidentialité des données	4	Dépend de la configuration du backend et des connexions chiffrées (TLS).
Gestion des accès	4	Fonctionnalités avancées pour des environnements collaboratifs, mais peut nécessiter du temps pour être configuré.
Ergonomie	4	Interface intuitive, mais nécessite une certaine expertise technique pour des visualisations avancées.

Solutions	Avantages	Inconvénients
Tableau	<ul style="list-style-type: none"> <li>- Large choix de visualisations interactives et avancées.</li> <li>- Intégration avec de multiples sources de données.</li> <li>- Facilité d'utilisation grâce à une interface glisser-déposer.</li> <li>- Fonctionnalités de partage et collaboration robustes.</li> <li>- Fonctionnalités analytiques avancées.</li> </ul>	<ul style="list-style-type: none"> <li>- Coût élevé, surtout pour les grandes équipes.</li> <li>- Nécessite une formation pour maîtriser les fonctionnalités avancées.</li> <li>- Dépendance au serveur Tableau pour des déploiements à grande échelle.</li> </ul>
Power BI	<ul style="list-style-type: none"> <li>- Intégration native avec Microsoft 365 et Azure.</li> <li>- Visualisations interactives et partage facilité.</li> <li>- Fonctionnalités d'alertes basées sur les seuils définis.</li> <li>- Idéal pour les utilisateurs de l'écosystème Microsoft.</li> </ul>	<ul style="list-style-type: none"> <li>- Performances limitées pour des datasets volumineux.</li> <li>- Courbe d'apprentissage pour des analyses complexes.</li> <li>- Nécessite une configuration locale pour des connexions à certaines sources.</li> </ul>
Qlik Sense	<ul style="list-style-type: none"> <li>- Modèle de données associatif unique permettant une exploration libre.</li> <li>- Intégration avec SQL, NoSQL, ERP.</li> <li>- Visualisations interactives adaptées à des analyses complexes.</li> <li>- Support d'analyses prédictives et de machine learning.</li> </ul>	<ul style="list-style-type: none"> <li>- Licence coûteuse.</li> <li>- Nécessite une courbe d'apprentissage pour configurer des modèles complexes.</li> <li>- Interface utilisateur moins intuitive pour les non-techniciens.</li> </ul>

Grafana	<ul style="list-style-type: none"> <li>- Open source et gratuit (version de base).</li> <li>- Spécialisé dans le monitoring en temps réel.</li> <li>- Intégration avec des bases SQL, NoSQL, et flux de données (Prometheus, InfluxDB).</li> <li>- Notifications robustes via email, Slack, PagerDuty, etc.</li> <li>- Interface hautement personnalisable.</li> </ul>	<ul style="list-style-type: none"> <li>- Limité pour des visualisations complexes ou des rapports statiques.</li> <li>- Nécessite des plug-ins payants pour des fonctionnalités avancées.</li> <li>- Non adapté pour les utilisateurs non techniques.</li> </ul>
Metabase	<ul style="list-style-type: none"> <li>- Open source et gratuit.</li> <li>- Interface intuitive et facile à utiliser pour les non-techniciens.</li> <li>- Connexion rapide avec des bases relationnelles et fichiers CSV.</li> <li>- Permet la création rapide de visualisations simples.</li> </ul>	<ul style="list-style-type: none"> <li>- Fonctionnalités limitées pour les analyses complexes ou graphiques avancés.</li> <li>- Pas adapté pour des charges importantes de données.</li> <li>- Personnalisation graphique limitée (pas de thèmes avancés).</li> </ul>
Apache Superset	<ul style="list-style-type: none"> <li>- Open source et extensible.</li> <li>- Large choix de visualisations interactives (barres, lignes, cartes, etc.).</li> <li>- Compatible avec SQL et bases NoSQL.</li> <li>- Notifications configurables avec intégrations externes.</li> <li>- Extensible via plug-ins.</li> </ul>	<ul style="list-style-type: none"> <li>- Courbe d'apprentissage pour les utilisateurs non techniques.</li> <li>- Requiert des compétences techniques pour l'installation et la configuration.</li> <li>- Moins performant pour les requêtes interactives sur des bases volumineuses.</li> </ul>

## 5 Résultat de l'état de l'art 1

Critères	Tableau	Power BI	Qlik Sense	Grafana	Metabase	Apache Superset
Utilisateurs à différents niveaux	4.0	4.0	4.0	3.0	3.0	5.0
Gestion des données	5.0	4.0	5.0	4.0	5.0	5.0
Personnalisation	4.5	4.5	4.0	3.0	3.0	4.75
Visualisation	5.0	4.0	5.0	5.0	4.0	5.0
Alertes et notifications	4.0	5.0	4.0	5.0	3.0	4.5
Rapports automatisés	5.0	5.0	4.0	3.0	4.0	3.0
Partage	5.0	4.5	4.0	3.0	4.0	4.0
Traçabilité	4.0	5.0	5.0	3.0	3.0	3.5
Accessibilité	5.0	4.0	5.0	4.0	4.0	4.0
Temps de réponse	4.0	4.0	5.0	5.0	4.0	4.0
Confidentialité des données	5.0	5.0	5.0	3.0	3.0	4.0



Gestion des accès	5.0	5.0	5.0	3.0	3.0	4.0
Ergonomie	5.0	4.0	3.0	4.0	5.0	4.0
Score pondéré final	4.8	4.3	4.23	3.83	4.12	4.26

## 6 Cahier de charges pour centraliser les données

Catégorie	Sous-catégorie	Description
Contexte du Projet	Objectif principal	Mettre en place une solution de Data Warehouse pour centraliser, stocker, et analyser de grandes quantités de données provenant de sources hétérogènes.
	Besoins	<ul style="list-style-type: none"> <li>- Consolider les données issues de multiples systèmes (bases SQL/NoSQL, IoT, ERP).</li> <li>- Fournir un accès rapide et fiable pour les analyses et visualisations de données.</li> <li>- Améliorer la scalabilité et réduire la complexité de la gestion des données.</li> </ul>
Sources de données	Types de données	<ul style="list-style-type: none"> <li>- Relationnelles : MySQL, PostgreSQL, SQL Server.</li> <li>- Non relationnelles : MongoDB, Cassandra, Firebase.</li> <li>- Données semi-structurées : JSON, XML.</li> <li>- Données IoT : logs de capteurs, flux en temps réel.</li> <li>- Fichiers plats : CSV, Excel, PDF</li> </ul>
Fonctionnalités Principales	Stockage de données	<ul style="list-style-type: none"> <li>- Capacité à gérer de grands volumes de données (pétaoctets).</li> <li>- Historisation des données sur plusieurs années.</li> </ul>

	Séparation calcul/stockage	Optimisation des ressources pour permettre un ajustement indépendant des performances de calcul et des capacités de stockage.
	Scalabilité	<ul style="list-style-type: none"><li>- Capacité à évoluer horizontalement pour répondre aux besoins croissants.</li><li>- Gestion automatique ou semi-automatique des ressources.</li></ul>
	Traitement SQL avancé	Support des requêtes complexes et analyses ad hoc pour extraire des insights décisionnels.
	Sécurité et conformité	<ul style="list-style-type: none"><li>Chiffrement des données (au repos et en transit).</li><li>- Gestion des accès basée sur les rôles (RBAC).</li><li>- Conformité avec les normes GDPR, SOC 2, ISO 27001.</li></ul>
Contraintes Techniques	Temps de réponse	<ul style="list-style-type: none"><li>- Requêtes simples : &lt; 3 secondes.</li><li>- Requêtes complexes sur grands volumes : &lt; 10 secondes.</li></ul>

## 7 Solutions pour les ETL

### 7.1. Airbyte



Critères	Notes /5	Description
Types de données	5/5	Supporte SQL, NoSQL (MongoDB, PostgreSQL), API, fichiers plats, cloud (S3, Google Cloud).
Connecteurs	5/5	Plus de 300 connecteurs natifs vers bases de données, API, warehouses cloud.
Stockage de données	5/5	Intégration avec PostgreSQL, MongoDB, Snowflake, BigQuery, etc.
Séparation calcul/stockage	5/5	Basé sur Docker et Kubernetes, scalable indépendamment du stockage.
Scalabilité	5/5	Adapté aux architectures cloud et microservices, extensible facilement.
Sécurité et conformité	5/5	Chiffrement des données, conformité GDPR, SOC 2, gestion avancée des permissions.

Avantages	Inconvénients
Open-source et gratuit.	Jeune projet, documentation encore en évolution.
Très facile à installer et à configurer (Docker, Kubernetes).	Interface utilisateur parfois instable sur certaines versions.
Support natif de MongoDB et PostgreSQL.	Moins d'optimisations sur les très gros volumes comparé à des solutions cloud natives.
300+ connecteurs pour bases de données, API, et fichiers.	Nécessite une maintenance active pour suivre les mises à jour.

## 7.2. Talend



Critères	Notes /5	Description
Types de données	5/5	Supporte SQL, NoSQL, API, fichiers, applications cloud (Salesforce, SAP).
Connecteurs	5/5	Grand nombre de connecteurs, mais beaucoup nécessitent la version payante.
Stockage de données	4/5	Compatible avec plusieurs systèmes, mais nécessite une configuration avancée.
Séparation calcul/stockage	3/5	Moins flexible qu'Airbyte sur la gestion des ressources.
Scalabilité	4/5	Bonne, mais demande un tuning manuel.
Sécurité et conformité	5/5	Sécurité avancée, certifications GDPR, ISO 27001.
Temps de réponse	4/5	Performant, mais les jobs peuvent être lourds à exécuter.

Avantages	Inconvénients
Solution complète avec une grande communauté.	Version open-source limitée, nécessite un abonnement pour les fonctionnalités avancées.
Très bon support des bases SQL et NoSQL.	Interface moins intuitive, prise en main plus longue.
Sécurité et conformité élevées.	Déploiement plus complexe comparé à Airbyte.
Automatisation avancée avec des workflows complexes.	Temps d'exécution parfois long pour les gros flux.

### 7.3. Fivetran

Critères	Notes /5	Description
Types de données	4/5	Principalement orienté bases SQL, peu de support NoSQL.
Connecteurs	5/5	Très nombreux connecteurs cloud, mais payants.
Stockage de données	5/5	Intégration cloud native avec BigQuery, Snowflake, Redshift.
Séparation calcul/stockage	5/5	Complètement découplé grâce à son architecture cloud.
Scalabilité	5/5	Très scalable en SaaS.
Sécurité et conformité	5/5	Sécurité avancée, certifications GDPR, SOC 2.
Temps de réponse	4/5	Performant mais latence sur certaines intégrations.

Avantages	Inconvénients
Déploiement ultra-rapide, service cloud natif.	Payant, coûteux à grande échelle.
Excellente intégration avec Snowflake, BigQuery et Redshift.	Moins flexible pour des pipelines personnalisés.
Sécurité et conformité au top.	Pas de version open-source.
Maintenance simplifiée, peu d'administration requise.	Moins adapté aux workflows complexes et aux bases NoSQL.

## 8 Comparaison des bases de données

Critères	MongoDB (NoSQL)	PostgreSQL (SQL)	MySQL (SQL)	Redis (Key-Value)
Modèle de données	Documents (JSON/BSON)	Relationnel (tables)	Relationnel (tables)	Clés-valeurs
Flexibilité du schéma	Dynamique (NoSQL)	Rigide (SQL)	Rigide (SQL)	Clé-Valeur strict
Scalabilité	Horizontale (sharding)	Verticale principalement	Verticale principalement	Extrêmement rapide en mémoire
Performances en lecture	Excellentes (indexation optimisée)	Bonne mais dépend des requêtes	Bonne mais dépend des requêtes	Ultra-rapide en mémoire
Performances en écriture	Très rapide (sans transactions complexes)	Plus lent avec transactions	Plus lent avec transactions	Ultra-rapide

<b>Support des transactions ACID</b>	Partiel (MongoDB 4.0+)	Complet	Complet	Pas de transactions
<b>Stockage et gestion des données</b>	NoSQL, adapté aux gros volumes et aux données évolutives	SQL structuré, adapté aux relations complexes	SQL structuré, adapté aux relations complexes	Stockage en mémoire, pas adapté aux gros volumes
<b>Requêtage</b>	Flexible (JSON, agrégations avancées)	Puissant (SQL, JSONB)	Standard SQL	Limité (recherches par clé)

## 9 Résultat de l'état de l'art 2

Critères	Airbyte	Talend	Fivetran
Types de données	5.0	5.0	4.0
Connecteurs	5.0	5.0	5.0
Stockage de données	5.0	4.0	5.0
Séparation calcul/stockage	5.0	3.0	5.0
Scalabilité	5.0	4.0	5.0
Sécurité et conformité	5.0	5.0	5.0
Temps de réponse	5.0	4.0	4.0
Score pondéré final	5.0	4.28	4.7

## 10 Choix de la réalisation

### 10.1 Spring Boot pour le Backend

- **Pourquoi Spring Boot ?**

Spring Boot a été choisi pour son **écosystème complet**, sa **flexibilité**, et sa **rapidité de développement**. Spring Boot facilite la création d'applications Java robustes avec une configuration minimale. Il fournit des outils et des configurations par défaut qui accélèrent le développement tout en garantissant une architecture modulaire et facilement testable.

- **Avantages spécifiques :**

➔ **API RESTful** : Spring Boot permet de créer des API RESTful facilement, ce qui est idéal pour fournir des services backend pour une interface frontend basée sur React.

➔ **Sécurité** avec Spring Security (authentification, autorisation).

→ Modulaire et testable.

## 10.2 PostgreSQL (Base de données)

- **Pourquoi PostgreSQL ?**

Nous avons choisi **PostgreSQL** pour sa **fiabilité**, sa **scalabilité** et ses **fonctionnalités avancées**. PostgreSQL offre des performances exceptionnelles pour les applications nécessitant des transactions complexes et des requêtes de données relationnelles. Il est parfaitement adapté pour gérer les tables complexes nécessaires au stockage des données de dashboards et d'utilisateurs.

- **Avantages spécifiques :**

→ **Support des types de données avancés** : PostgreSQL permet d'utiliser des types de données complexes et des fonctionnalités comme les **transactions** et **index** avancés, utiles pour une application qui traite des données volumineuses.

→ **Fiabilité et scalabilité** : C'est un choix parfait pour gérer une application à grande échelle avec un grand nombre d'utilisateurs et de données.

## 10.3 Metabase (intégration de visualisations)

- **Pourquoi Metabase ?**

Metabase a été choisi pour ses capacités puissantes de création de visualisations et de dashboards interactifs, sans nécessiter une expertise en programmation. Grâce à sa simplicité d'utilisation et son intégration fluide avec des bases de données comme **PostgreSQL**, Metabase permet de générer des graphiques, des cartes et des rapports basés sur les données extraites, facilitant ainsi la visualisation des informations par les utilisateurs finaux.

- **Avantages spécifiques de Metabase dans le projet :**

1. **Visualisations prêtes à l'emploi :**

- Metabase offre une **large gamme de visualisations** prêtes à l'emploi (graphique en barres, camemberts, lignes, etc.) qui peuvent être facilement intégrées dans les dashboards.
- Ces visualisations sont automatiquement générées à partir des **données stockées dans PostgreSQL**, ce qui permet aux utilisateurs de rapidement comprendre les tendances, les performances et d'autres aspects clés des données sans nécessiter une configuration complexe.

## 2. Facilité d'intégration avec Spring Boot :

- L'intégration de **Metabase** avec le backend **Spring Boot** est facilitée par l'utilisation d'**APIs RESTful**. Spring Boot permet de servir des **données dynamiques** depuis la base PostgreSQL sous forme de **requêtes API**, que Metabase peut ensuite consommer pour générer des dashboards et des rapports interactifs.
- APIs RESTful permettent une **communication fluide** entre le serveur backend Spring Boot et le frontend Metabase, où les utilisateurs peuvent interagir en temps réel avec les données.

## 3. API de Metabase pour automatiser la génération de dashboards :

- Une autre grande force de Metabase est son **API REST** qui permet de créer, modifier et récupérer des **dashboards et des cartes** (visualisations). Cela permet une **automatisation de la gestion des rapports**, sans avoir à passer par l'interface graphique de Metabase.
- **Dans le cadre de notre projet**, l'utilisation de l'API Metabase permet non seulement de **générer dynamiquement** des dashboards basés sur les requêtes provenant de Spring Boot, mais aussi d'automatiser la mise à jour des visualisations en fonction des nouvelles données.

## 4. Sécurité et accessibilité :

- **Metabase** garantit la sécurité des données en s'intégrant avec les **protocoles d'authentification** déjà présents dans Spring Boot, tout en permettant de gérer l'accès aux dashboards et aux données de manière fine (utilisateurs autorisés, rôles).

# 10.4 React pour le frontend

- **Pourquoi React.js ?**

**React** a été sélectionné pour sa capacité à créer des interfaces utilisateur dynamiques et réactives. Avec React, chaque modification des données entraîne un re-rendering rapide et optimisé de l'interface sans recharger la page entière, créant ainsi une expérience fluide et réactive pour l'utilisateur.

- **Avantages spécifiques :**

➔ **Composants réutilisables** : React permet de créer des composants réutilisables et modulaires, ce qui rend le code plus maintenable et plus extensible.

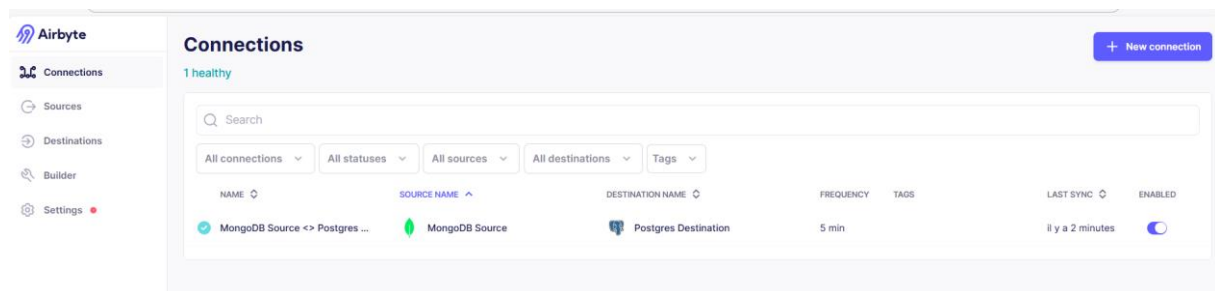
➔ **Gestion de l'état avec hooks** : React utilise les hooks pour gérer l'état et l'effet des composants, facilitant la gestion de l'état complexe de l'application.



## 10.5 Airbyte : une solution moderne et modulaire pour l'intégration de données

L'un des défis majeurs de notre projet réside dans la nécessité d'intégrer et de transformer des données issues de plusieurs sources. Pour répondre à cet enjeu, nous avons choisi Airbyte, une solution d'ETL (Extract, Transform, Load) moderne et open-source. Il permet une connexion à de nombreuses bases de données et permet l'ajout de ses propres connecteurs, de plus il se configure par api.

Dans notre cas, nous allons utiliser airbyte pour transférer la base MongoDB dans la table Postgres. Airbyte doit être installé via le script ABCTL fournis. On l'ajoute donc dans le projet afin que le setup auto puisse connaître les commandes. Start.sh permet de tout télécharger et de lancer les dockers. Il appelle ensuite airbyte-setup pour configurer la connexion de mongo à postgres, les données seront actualisées toutes les 5 minutes. On peut visualiser la connexion dans l'interface airbyte :



## 10.6 Docker et Docker Compose : une infrastructure standardisée et scalable

L'utilisation de Docker dans notre projet répond à un **double objectif** : garantir une uniformité entre les environnements de développement et de production, et simplifier le déploiement des services. En isolant chaque composant applicatif dans un conteneur dédié, Docker évite les conflits de dépendances et assure une meilleure portabilité du système.

Docker Compose joue un rôle clé dans l'orchestration de ces conteneurs, permettant ainsi de démarrer et configurer l'ensemble des services avec un simple fichier YAML. Cette approche nous assure une gestion centralisée et efficace des différents composants, qu'il s'agisse de la base de données, du backend, de l'ETL ou encore de la couche de visualisation.

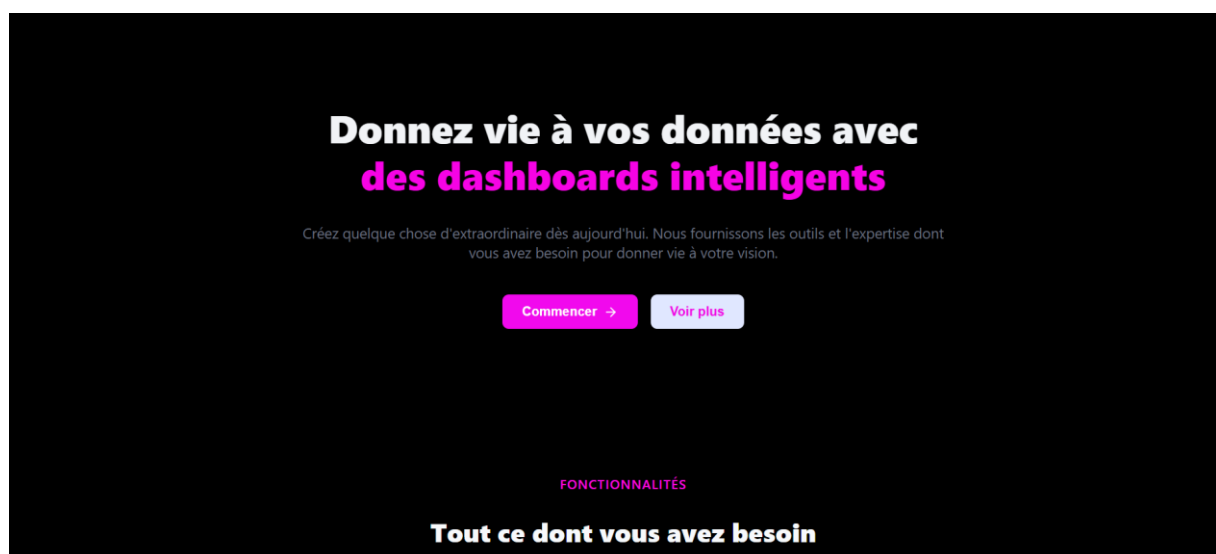
## 11 Résultat du développement

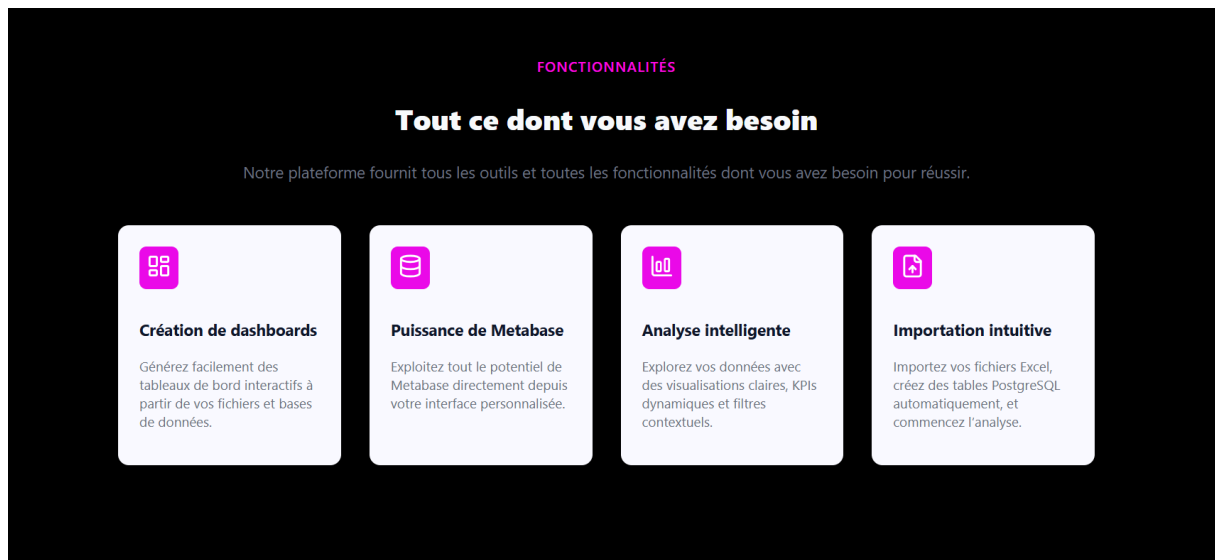
### 11.1 Présentation de l'interface

L'interface du tableau de bord permet à l'utilisateur de visualiser et interagir avec les données via différents types de graphiques (barres, lignes, camemberts, etc.). Elle inclut une barre de recherche pour filtrer les dashboards et un système d'alertes intelligentes qui détecte automatiquement les anomalies dans les données. L'utilisateur peut personnaliser les axes X et Y des graphiques selon ses besoins. Des KPI (nombre d'utilisateurs, dashboards, tables) sont affichés sous forme de cartes. Le menu latéral offre un accès rapide aux différentes sections de l'application comme la gestion des utilisateurs, la création de dashboards, et l'importation de données.

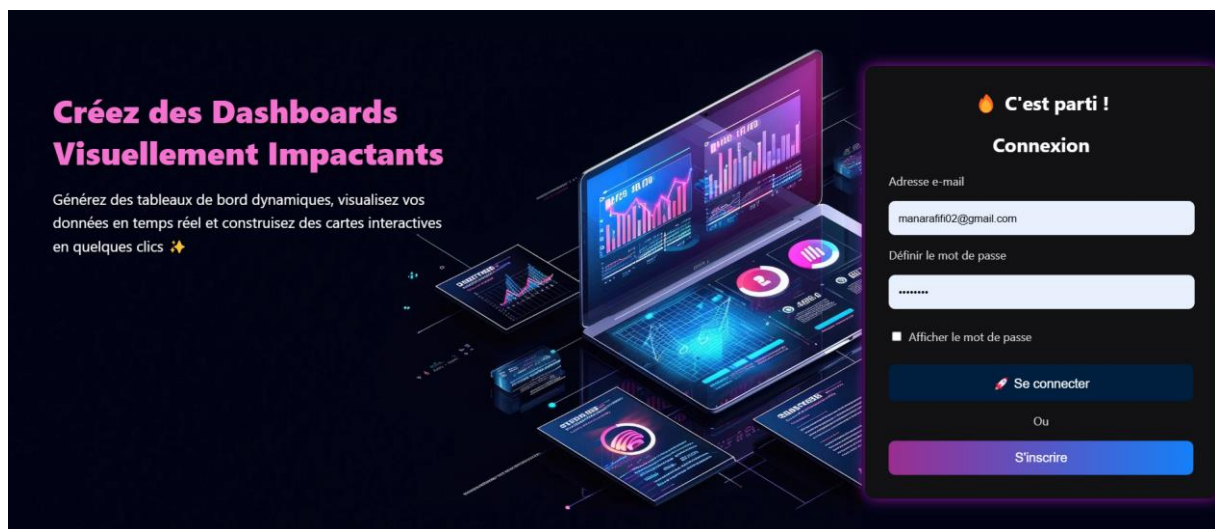
Elle intègre également des fonctionnalités liées à Metabase, permettant de visualiser, créer et personnaliser des cartes (questions) et des dashboards. Grâce à l'intégration avec Metabase, l'utilisateur peut consulter des graphiques dynamiques générés directement à partir des données des cartes Metabase. Il peut également obtenir des URL d'intégration pour embarquer des cartes ou dashboards dans l'application. Le système d'alertes est également intelligent, détectant automatiquement des anomalies dans les données des cartes Metabase et affichant des notifications en fonction des valeurs aberrantes ou significatives.

### 11.2 Petite démo représentant quelques fonctionnalités :

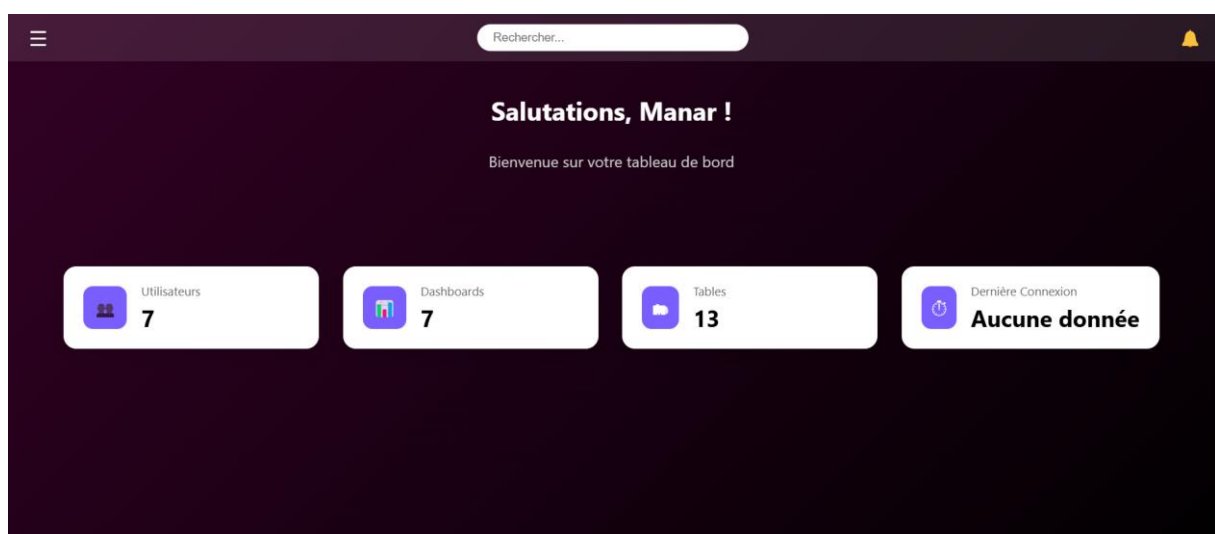




Page de connexion :



Vue d'ensemble :



Importation des données :


## Import de Fichiers

[← Retour au Dashboard](#)

Progression  
**100%**

Types de fichiers  
**1**

Taille totale  
**60.03 kB**




Glissez-déposez vos fichiers ici

ou

Sélectionner des fichiers

Tous types de fichiers acceptés

Fichiers (3)




Glissez-déposez vos fichiers ici


ou


Sélectionner des fichiers

Tous types de fichiers acceptés

Fichiers (3)

 **clients\_plus.xlsx**  
12.55 kB

 **iot\_sensor\_data\_1000.xlsx**  
41.54 kB

 **achats.xlsx**  
5.94 kB

Génération des kpi de ces données importées :

### Fichiers importés

- clients\_plus.xlsx
- iot\_sensor\_data\_1000.xlsx
- achats.xlsx

### KPIs du fichier : clients\_plus.xlsx

Moyenne de ID\_CLIENT  
53.00

Somme de ID\_CLIENT  
5565.00

Catégorie fréquente de NOM  
Dupont

Catégorie fréquente de PRENOM  
Anna

Catégorie fréquente de CIVILITE  
M.

Catégorie fréquente de ADRESSE  
15 rue des Lilas

Catégorie fréquente de VILLE  
Berlin

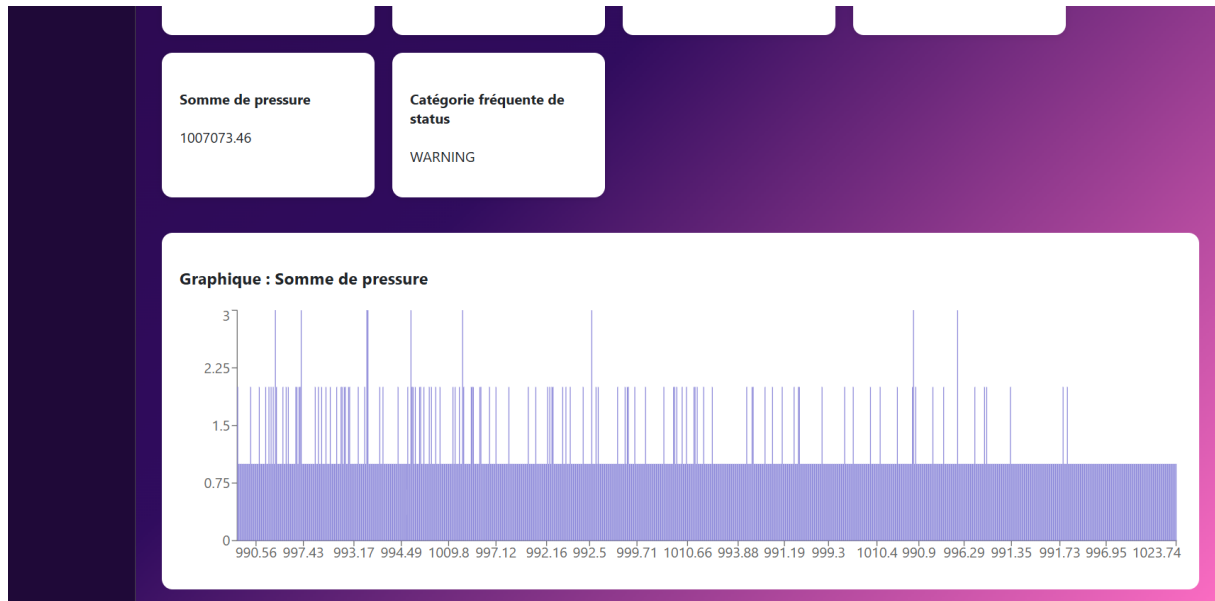
Moyenne de CODE\_POSTAL  
51684.30

Somme de CODE\_POSTAL  
5426851.00

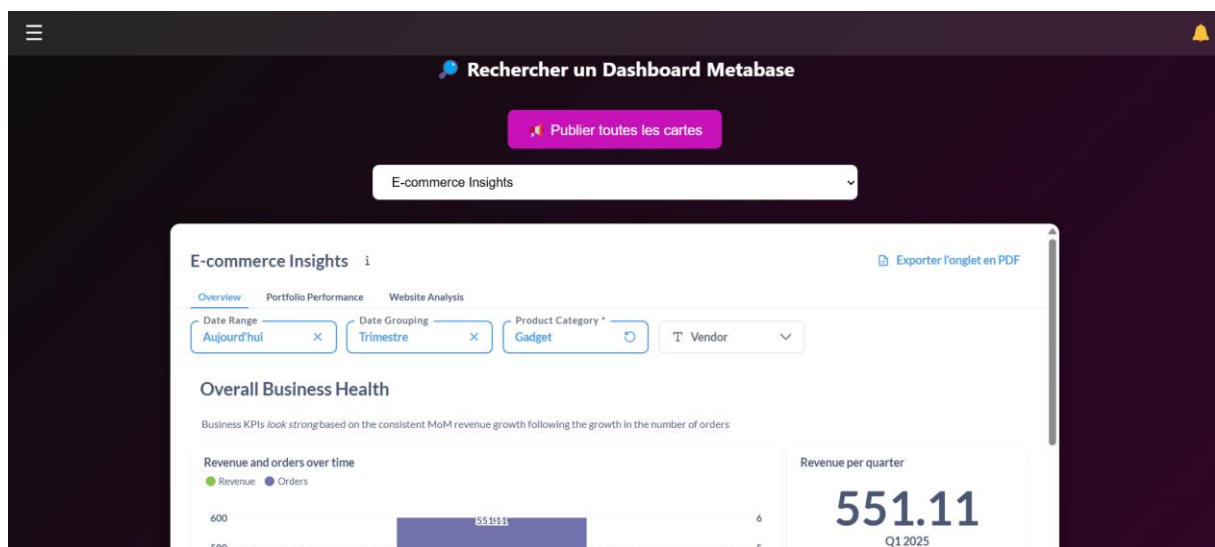
Catégorie fréquente de PAYS  
Italie

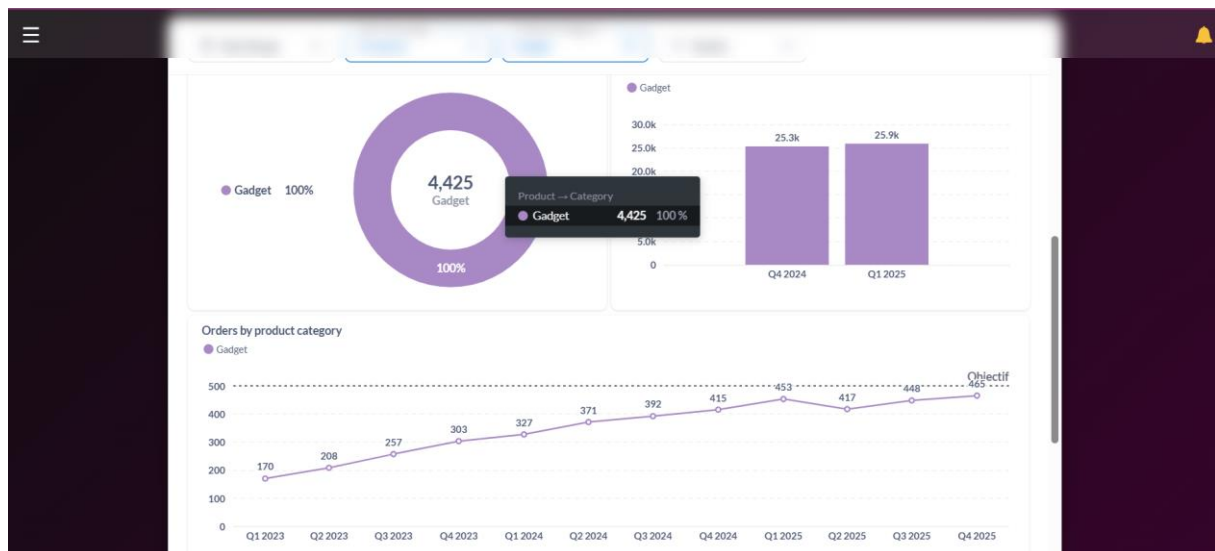
Moyenne de MOBILE  
9392062541.48

Somme de MOBILE  
986166566855.00

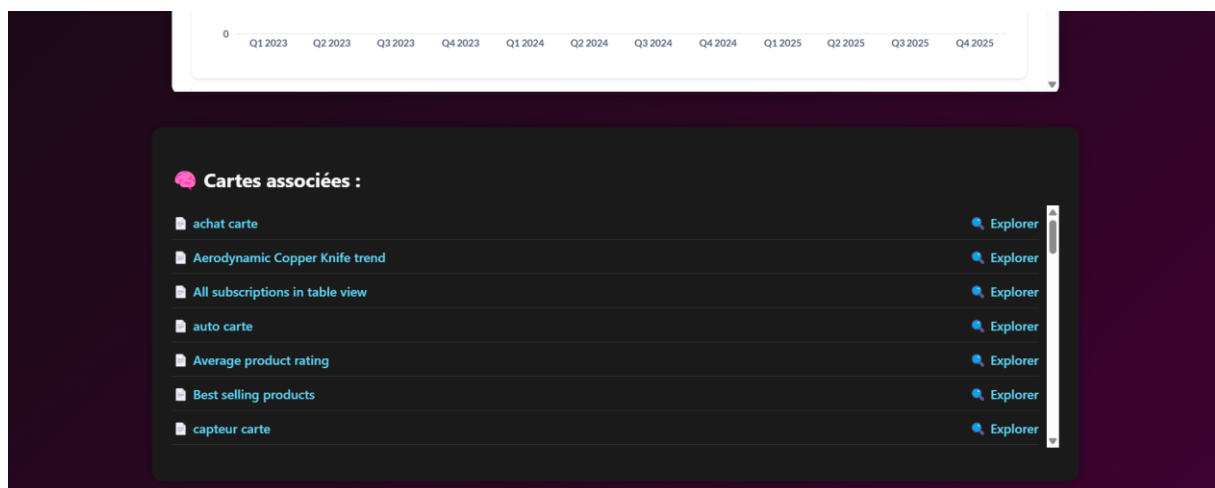


## Récupération des dashboards Metabase :





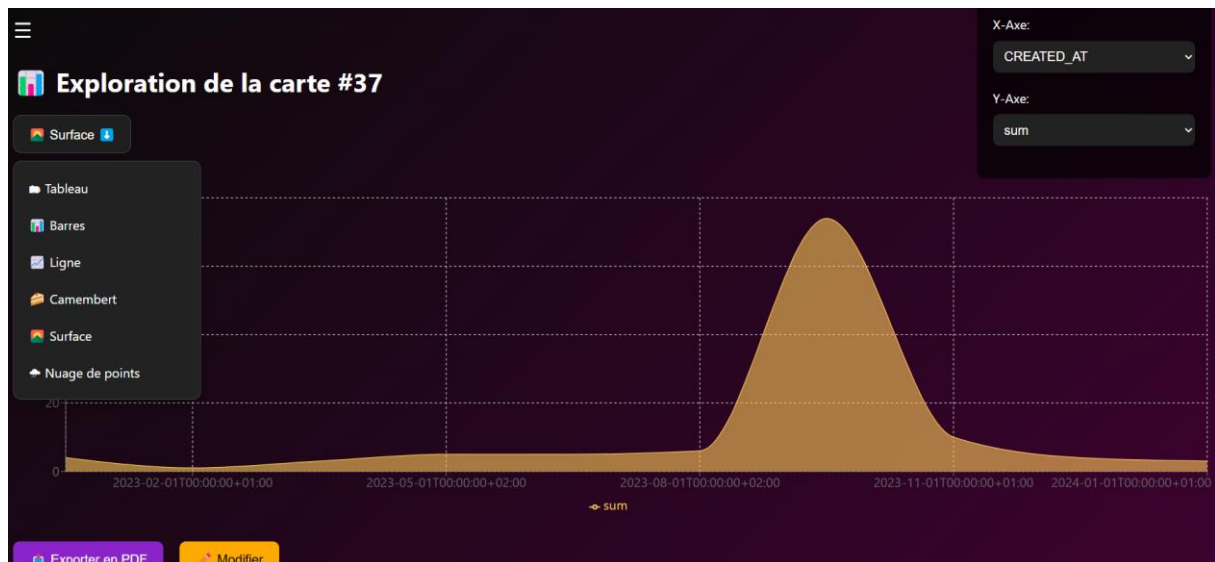
Affichage des cartes Metabase et les cartes créées via notre interface :



Visualisation des dashboards :

The screenshot shows the 'Exploration de la carte #37' dashboard in Tableau. The X-Axis is set to 'CREATED\_AT' and the Y-Axis is set to 'sum'. The data is displayed in a table with the following columns: 'CREATED\_AT' and 'SUM'.

CREATED_AT	SUM
2022-10-01T00:00:00+02:00	4
2023-02-01T00:00:00+01:00	1
2023-04-01T00:00:00+02:00	3
2023-05-01T00:00:00+02:00	5
2023-06-01T00:00:00+02:00	5
2023-08-01T00:00:00+02:00	6
2023-09-01T00:00:00+02:00	74



Possibilité de modifier les données :

Modifier Carte #84

Rechercher...

+ Nouvelle Colonne + Nouvelle Ligne

date	produit	quantite	prix_unitaire	total	fournisseur
2025-03-16	Ecran	4	11.47	45.88	BureauPro
2025-03-28	Souris	3	281.28	843.84	OfficePlus
2025-03-27	Imprimante	2	141.45	282.9	OfficePlus
2025-03-27	Bureau	8	11.37	90.96	OfficePlus
2025-03-25	Webcam	8	26.22	209.76	TechStore
2025-03-07	Souris	4	129.99	519.96	Fournitech

Importer des données pour les insérer dans la base postgresQL et Metabase

Importer un Excel et Créer une Table Automatiquement

Nom de la table PostgreSQL

Choisir un fichier Aucun fichier choisi

Importer

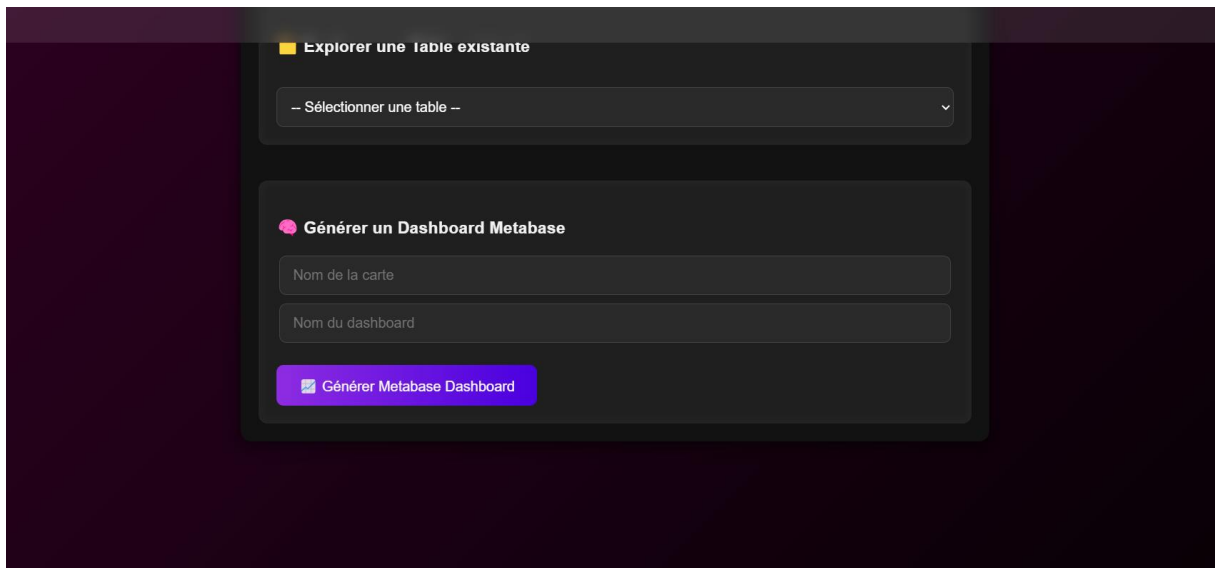
Explorer une Table existante

Sélectionner une table


Générer un Dashboard Metabase

Nom de la carte

Création des dashboards et des cartes :



Créer des alertes :

 **Créer une Alerte**

Remplissez le formulaire pour définir une nouvelle règle d'alerte.


-- Sélectionner une carte --


▼

Supérieur à

▼

0

 **Créer l'alerte**

 **Mes Règles enregistrées**



The screenshot shows a web application interface with a light blue background. At the top, there is a form for creating an alert. It includes a dropdown menu labeled 'Supérieur à' with a downward arrow, and a text input field containing the number '0'. Below this is a text input field labeled 'Message d'alerte'. A prominent purple button with a white envelope icon and the text 'Créer l'alerte' is positioned below the form. A horizontal line separates the form from the 'Mes Règles enregistrées' section. This section has a title with a document icon and contains a list of registered rules. Each rule is displayed in a white box with a light blue border. The first two rules are identical: they feature a yellow bell icon, the title 'Ventes critiques', the condition 'montant\_total > 100000' (or 'total\_\_ > 100000' for the second), and a notification message 'Ventes très élevées détectées !' accompanied by a red envelope icon. A vertical scrollbar is visible on the right side of the rules list.

## 12 Difficultés rencontrées

### 1. Intégration entre React, Spring Boot et Metabase :

**Gestion des erreurs d'API :** L'utilisation de plusieurs services d'API (Spring Boot et Metabase) a entraîné des erreurs liées à des appels API non réussis, des erreurs de CORS, ou des problèmes de disponibilité du serveur Metabase. Cela a parfois ralenti l'intégration et la mise en place d'un système de gestion des erreurs robustes.

**Par exemple :** si les données d'une carte Metabase étaient envoyées sous un format inattendu ou si la réponse JSON contenait des erreurs, le backend Spring Boot lançait des exceptions. Nous avons dû implémenter une gestion des erreurs et des exceptions dans Spring Boot afin de capturer ces erreurs et renvoyer des réponses appropriées au frontend. Cela a impliqué l'ajout d'une gestion des exceptions globales pour garantir que les utilisateurs recevaient des messages d'erreur clairs et appropriés au lieu d'exception brutes.

### 2. Sécurisation des API Metabase :

- **Authentification et autorisation :** Metabase expose des API sensibles pour récupérer les données de ses dashboards. Gérer l'authentification et les permissions sur ces API n'a pas été une tâche simple. En l'absence de

mécanismes de sécurité natifs dans Metabase pour gérer les rôles et permissions dans le cadre d'une intégration avec un backend personnalisé (Spring Boot), des efforts supplémentaires ont été nécessaires pour sécuriser l'accès aux dashboards et aux données via l'API de Metabase.

### **3. Difficulté d'intégration et d'affichage des cartes Metabase dans l'application**

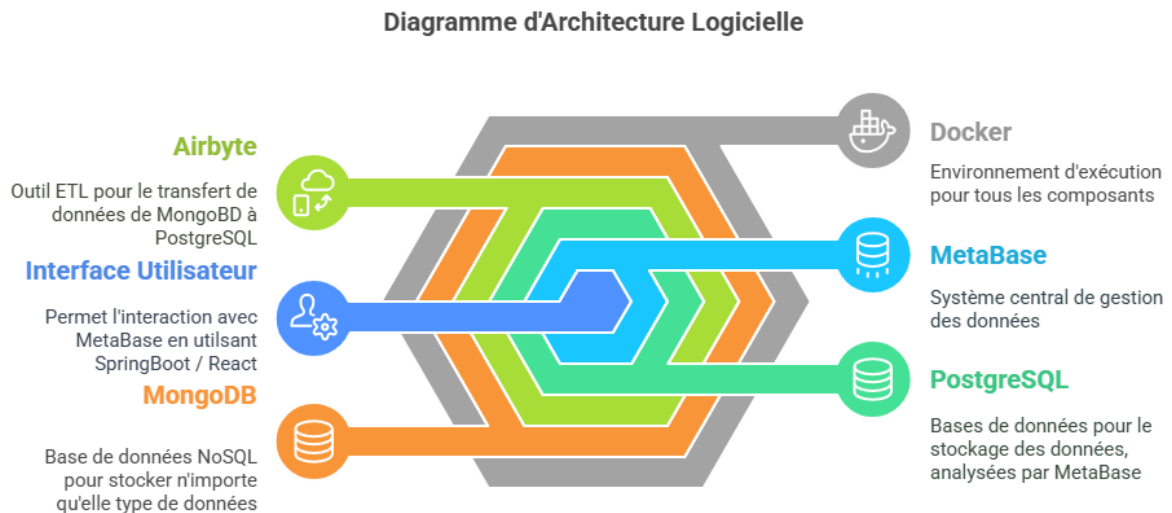
Le problème résidait dans l'intégration de l'iframe de la carte Metabase dans notre frontend React. Même après avoir ajouté la carte dans le dashboard via l'API de Metabase et avoir récupéré l'ID de la carte, l'affichage dans l'application n'était pas réussi. Cela était dû à des erreurs liées à l'URL de l'iframe, la gestion du token d'authentification pour l'intégration de Metabase dans l'iframe, ou encore un problème de CORS, empêchant le bon chargement de la carte.

Nous avons dû prendre plusieurs mesures pour résoudre ce problème, telles que :

- Vérifier que l'URL de l'iframe était correcte et contenait les paramètres nécessaires (comme les tokens d'authentification) pour permettre un affichage sans erreurs.
- Configurer Metabase pour autoriser l'intégration d'iframes via son paramétrage de sécurité.
- Ajouter des mécanismes de gestion des erreurs côté frontend pour intercepter toute erreur lors du chargement de la carte et afficher un message approprié aux utilisateurs en cas d'échec.

Cela a entraîné un travail supplémentaire pour s'assurer que l'intégration entre Metabase, l'API Spring Boot et le frontend React soit fluide, sécurisée et fonctionnelle.

## 13 Architecture du projet



L'architecture logicielle mise en place repose sur des technologies open-source, interopérables et entièrement configurables via API. Ce choix garantit flexibilité, évolutivité et adaptabilité à différents types de projets ou d'entreprises.

Le système capte automatiquement tout type de données dans une base MongoDB, capable de stocker des données hétérogènes. Ces données sont ensuite transférées automatiquement dans une base PostgreSQL via Airbyte, un outil ETL moderne, simple à intégrer dans un pipeline de données.

Ensuite, Metabase analyse la base PostgreSQL en détectant automatiquement les schémas et relations, ce qui permet de générer dynamiquement tous les graphiques d'analyse pertinents.

Une interface utilisateur personnalisée, développée en Spring Boot + React, permet d'interagir avec Metabase via son API : les utilisateurs peuvent visualiser, filtrer, exporter ou enregistrer les graphiques au format PDF, et enrichir l'expérience de visualisation des données de façon intuitive.

Enfin, l'ensemble de la solution est conteneurisé avec Docker, ce qui permet un déploiement rapide et reproductible sur n'importe quel environnement. Un simple fichier Start.sh permet de lancer tous les services et de configurer automatiquement l'ensemble du système.

Cette architecture constitue ainsi un pipeline complet de la donnée, allant de la captation brute à l'analyse visuelle avancée, tout en offrant une interface de pilotage simple et puissante.

## 14 Conclusion

L'architecture technologique adoptée repose sur des choix réfléchis visant à concilier robustesse, flexibilité et évolutivité. Spring Boot offre un cadre de développement efficace et modulaire, tandis que l'utilisation combinée de PostgreSQL et MongoDB nous permet de gérer efficacement des données relationnelles et semi-structurées. Metabase facilite l'exploitation des données et leur visualisation, rendant les analyses accessibles aux utilisateurs non techniques.

L'intégration d'Airbyte constitue un atout majeur pour la gestion des flux de données, offrant une solution d'ETL performante et adaptable aux exigences du projet. Enfin, Docker et Docker Compose garantissent une infrastructure homogène et facilement déployable, simplifiant ainsi la mise en production et l'orchestration des services.

Ces choix technologiques nous permettent d'assurer une architecture à la fois robuste et évolutive, en adéquation avec les exigences du projet et les défis futurs qu'il pourra rencontrer.