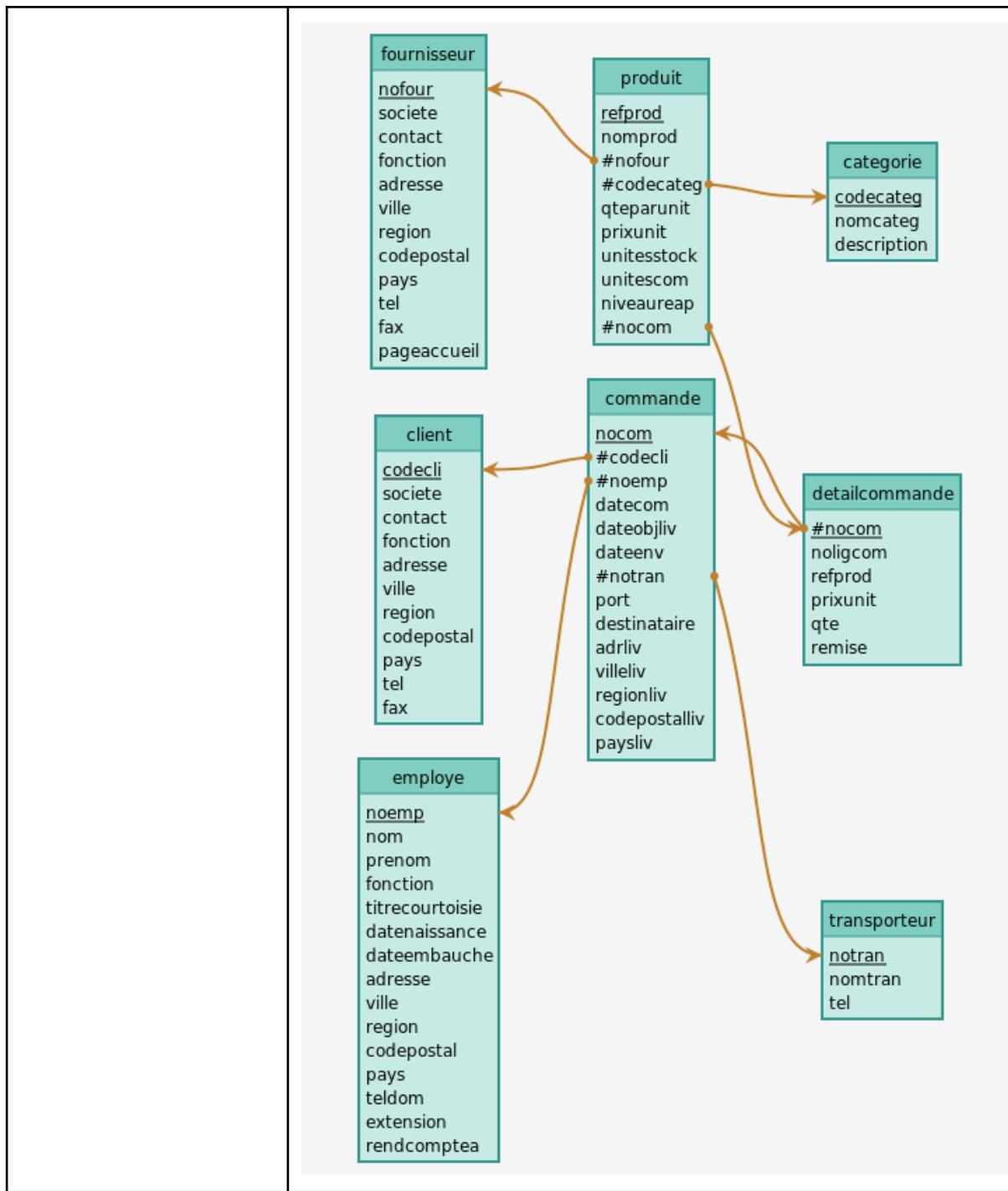




SAE 5-01 : Conception d'un outil décisionnel - Le cahier des charges

CONTEXTE	<p>Le responsable du service Achats de l'entreprise RIYOBRU nous contacte afin de mettre en place une application simplifiée intégrant certaines fonctionnalités de l'ERP Sage X3 pour avoir une meilleure gestion du carnet de commande.</p> <p>RIYOBRU est une entreprise de vente en ligne dans le secteur de l'aérospatial fondée en 1990, compte actuellement 1 million de clients et 110 employés en France.</p> <p>La problématique actuelle est que l'historique des commandes est mal référencé puisque tout est au format papier. Une première numérisation du carnet de commande 2025 a été mise en place en renseignant les commandes dans un fichier Excel. Or, Jean Dupont, responsable informatique chez RIYOBRU a publié l'article suivant sur le canal de communication commun de l'entreprise, ce qui a effrayé le service Achats :</p> <p><u>Comment Microsoft utilise vos données dans les fichiers Word et Excel ?</u></p> <p>De ce fait, le service informatique étant simplement orienté sur la partie support, Jean Dupont a fait appel à notre équipe pour répondre aux problématiques actuelles de l'entreprise concernant le carnet de commande. Le service Achats nous laisse un délai de trois mois et demi pour réaliser la demande.</p>
OBJECTIF	<p>L'objectif principal est de concevoir un datawarehouse sécurisé de sorte à préserver la confidentialité des données de l'entreprise et des informations clients. Le service Achats réclame un environnement de stockage sain, capable de respecter le Règlement Général sur la Protection des Données (RGPD).</p> <p>L'entreprise sera à même de consulter les données clientes via une application simple d'utilisation développée en parallèle.</p>
	Les personnes missionnées pour mener à bien le projet ci-dessus sont les suivantes :

	<ul style="list-style-type: none"> - Valentin Rieu : Ingénieur Génie Logiciel ; - Jimmy Yobo : Chargé de projets IT ; - Théo Brugel : Chargé de projets Business Intelligence. <p>Etant donné que notre équipe est chargée de traiter les données sensibles pour l'entreprise, il va falloir travailler avec l'idée de sécuriser au mieux les données des fournisseurs (en tant que personnes physiques, et non morales) et les données clients de sorte à respecter toutes les politiques de confidentialité et les règles établies par la CNIL via le RGPD.</p> <p>En effet, un certain nombre de mesures sont obligatoires.</p> <p>Dans un premier temps, il s'agira de créer le squelette de l'entrepôt de données.</p> <p>Une fois cette étape faite, on pourra insérer des données dans l'entrepôt de données sécurisé afin d'alimenter notre base de données.</p> <p>Tes</p> <p>Nous procéderons ensuite à une série de tests unitaires et d'intégrations dans un environnement de développement dédié, ainsi que des tests fonctionnels, de sécurité et de performance dans un environnement de test. L'objectif est de vérifier que l'application fonctionne correctement, que les fonctionnalités jouent leur rôle, etc...</p>
STRUCTURE	Les données des commandes ainsi que des clients sont actuellement stockés dans une base de données PostgreSQL avec le modèle de données suivant :



	<pre> erDiagram { "fournisseur" { string "nofour" string "societe" string "contact" string "fonction" string "adresse" string "ville" string "region" string "codepostal" string "pays" string "tel" string "fax" string "pageaccueil" } "produit" { string "refprod" string "nomprod" string "nofour" string "codecateg" number "qteparunit" string "prixunit" string "unitesstock" string "unitescom" string "niveaureap" } "categorie" { string "codecateg" string "nomcateg" string "description" } "client" { string "codecli" string "societe" string "contact" string "fonction" string "adresse" string "ville" string "region" string "codepostal" string "pays" string "tel" string "fax" } "commande" { string "nocom" string "codecli" string "noemp" string "datecom" string "dateobjliv" string "dateenv" string "notran" string "port" string "destinataire" string "adrliv" string "villeliv" string "regionliv" string "codepostalliv" string "paysliv" } "detailcommande" { string "nocom" string "noligcom" string "refprod" string "prixunit" string "qte" string "remise" } "employe" { string "noemp" string "nom" string "prenom" string "fonction" string "titrecourroisie" string "datenaissance" string "dateembauche" string "adresse" string "ville" string "region" string "codepostal" string "pays" string "teldom" string "extension" string "rendcompte" } "transporteur" { string "notran" string "nomtran" string "tel" } } "fournisseur" }o--o{ "produit" : "fournir" "produit" }o--o{ "categorie" : "appartenir" "produit" }o--o{ "commande" : "contenir" "client" }o--o{ "commande" : "effectuer" "commande" }o--o{ "detailcommande" : "décrire" "employe" }o--o{ "commande" : "préparer" "commande" }o--o{ "transporteur" : "expédier" </pre>
TECHNOLOGIE ET OUTILS TIERS	<p>L'application permettant d'accéder aux données du datawarehouse fonctionnant sur le modèle client-serveur, l'utilisation de plusieurs technologies est nécessaire.</p> <p>L'entrepôt de données, sous PostgreSQL, sera conservé et hébergé dans un serveur d'entreprise. Cette base de données sera accessible uniquement via le réseau local. Une nouvelle table sera créée contenant, pour chaque utilisateur de l'appli, un identifiant, un rôle et un mot de passe haché et salé avec la</p>

	<p>fonction de hachage Bcrypt.</p> <p>La consultation des données ainsi que la gestion de l'authentification d'un utilisateur sera gérée par une API REST développée en Java avec le framework Spring Boot. Cette API sera déployée sur un serveur d'entreprise et accessible depuis n'importe quel réseau. Une des qualités de cette méthode est que chaque membre de l'entreprise aura un accès à l'application (si elle en a les droits évidemment) et ce, quel que soit son système d'exploitation. L'idée de passer par l'active directory a été envisagée mais se limite à des utilisateurs Windows et imposerait donc une contrainte.</p> <p>Avoir une API REST peut aussi permettre dans le futur de créer d'autres outils utilisant les données du datawarehouse.</p> <p>L'interface sera développée avec des technologies web, HTML, CSS et la bibliothèque javascript React. Une fois déployée sur un serveur d'entreprise, elle sera accessible depuis n'importe quel réseau sur un navigateur web.</p>
DIAGRAMME DE DÉPLOIEMENT	<pre> graph TD Utilisateur --- Navigateur_web[Navigateur web] Navigateur_web --- Serveur_web[Serveur web] Serveur_web --- Serveur_d_entreprise[Serveur d'entreprise] Serveur_d_entreprise --- Base_donnees[Base de données PostgreSQL] Serveur_d_entreprise --- API_REST[API REST Spring Boot] </pre> <p>Le diagramme de déploiement illustre la architecture du système. Un utilisateur interactif (représenté par un symbole humain) communique avec un navigateur web. Ce dernier interagit avec un serveur web, qui à son tour fournit des services au serveur d'entreprise. Le serveur d'entreprise contient une base de données PostgreSQL et une application API REST développée avec Spring Boot.</p>

<h3>DIAGRAMME DE COMPOSANTS</h3>	<pre> graph TD Produit -- "Code article" --> Commande Commande -- "Details client" --> Clients Compte -- "Details compte" --> Commande Palement --> Commande Palement --> Compte </pre> <p>Detailed description: Ce diagramme de composants illustre la structure logique d'un système. Il commence par une entité 'Produit' qui fournit un 'Code article' à une 'Commande'. La 'Commande' possède des 'Details client' qui sont associés à des 'Clients'. De plus, la 'Commande' est reliée à un 'Compte' via des 'Details compte'. Enfin, il existe une relation 'Palement' qui peut être appliquée à la fois à la 'Commande' et au 'Compte'.</p>																																																																																																																																																						
<h3>DIAGRAMME DE GANTT</h3>	<table border="1"> <thead> <tr> <th>Date</th> <th>15 JANV</th> <th>24 JANV</th> <th>02 FEV</th> <th>18 FEV</th> <th>11 FÉV</th> <th>20 FÉV</th> <th>01 MARS</th> <th>10 MARS</th> <th>15 MARS</th> </tr> </thead> <tbody> <tr> <td>Analyse et conception</td> <td>En cours</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>Architectture</td> <td>En cours</td> <td>En cours</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>Développement base de données</td> <td></td> <td></td> <td>En cours</td> </tr> <tr> <td>Mise en place et test de la BDD</td> <td></td> <td></td> <td>En cours</td> </tr> <tr> <td>Focus sur la sécurité</td> <td></td> <td></td> <td>En cours</td> </tr> <tr> <td>Développement Backend</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td>En cours</td> <td>En cours</td> <td>En cours</td> </tr> <tr> <td>Configuration Spring Boot</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td>En cours</td> <td>En cours</td> <td>En cours</td> <td>En cours</td> </tr> <tr> <td>Développement API et authentification</td> <td></td> <td></td> <td></td> <td></td> <td>En cours</td> <td>En cours</td> <td>En cours</td> <td>En cours</td> <td>En cours</td> </tr> <tr> <td>Développement Frontend</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td>En cours</td> <td>En cours</td> <td>En cours</td> </tr> <tr> <td>Développement des interfaces</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td>En cours</td> <td>En cours</td> <td>En cours</td> </tr> <tr> <td>Setup React</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td>En cours</td> <td>En cours</td> <td>En cours</td> </tr> <tr> <td>Tests & Finalisation</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td>En cours</td> <td>En cours</td> <td>En cours</td> </tr> <tr> <td>Test unitaire</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td>En cours</td> <td>En cours</td> </tr> <tr> <td>Documentation finale</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td>En cours</td> </tr> </tbody> </table> <p>Detailed description: Ce diagramme Gantt fournit une vue temporelle des tâches. Les tâches sont classées par couleur et leur état est indiqué par une légende : 'En cours' (vert), 'En attente d'être démarré' (couleur pastel).</p>	Date	15 JANV	24 JANV	02 FEV	18 FEV	11 FÉV	20 FÉV	01 MARS	10 MARS	15 MARS	Analyse et conception	En cours									Architectture	En cours	En cours								Développement base de données			En cours	Mise en place et test de la BDD			En cours	Focus sur la sécurité			En cours	Développement Backend							En cours	En cours	En cours	Configuration Spring Boot						En cours	En cours	En cours	En cours	Développement API et authentification					En cours	Développement Frontend							En cours	En cours	En cours	Développement des interfaces							En cours	En cours	En cours	Setup React							En cours	En cours	En cours	Tests & Finalisation							En cours	En cours	En cours	Test unitaire								En cours	En cours	Documentation finale									En cours																						
Date	15 JANV	24 JANV	02 FEV	18 FEV	11 FÉV	20 FÉV	01 MARS	10 MARS	15 MARS																																																																																																																																														
Analyse et conception	En cours																																																																																																																																																						
Architectture	En cours	En cours																																																																																																																																																					
Développement base de données			En cours																																																																																																																																																				
Mise en place et test de la BDD			En cours																																																																																																																																																				
Focus sur la sécurité			En cours																																																																																																																																																				
Développement Backend							En cours	En cours	En cours																																																																																																																																														
Configuration Spring Boot						En cours	En cours	En cours	En cours																																																																																																																																														
Développement API et authentification					En cours																																																																																																																																																		
Développement Frontend							En cours	En cours	En cours																																																																																																																																														
Développement des interfaces							En cours	En cours	En cours																																																																																																																																														
Setup React							En cours	En cours	En cours																																																																																																																																														
Tests & Finalisation							En cours	En cours	En cours																																																																																																																																														
Test unitaire								En cours	En cours																																																																																																																																														
Documentation finale									En cours																																																																																																																																														
	<p>L'outil qui va être développé permettra la numérisation et la centralisation des commandes de l'entreprise RIYOBRU. Il assurera la transformation des archives papiers existantes en un système numérique structuré et facile à exploiter.</p> <p>Etant donné que nous allons rattacher notre base de données à notre application Web, cette dernière bénéficiera d'un contrôle d'accès de sorte à attribuer les rôles des utilisateurs et l'accès à certaines sources de données en fonction de leur poste au sein de l'entreprise. (en attendant raccord avec le reste des fonctionnalités)</p> <p>Un utilisateur non authentifié devra d'abord s'authentifier via une</p>																																																																																																																																																						

FONCTIONNALITÉS	<p>page de connexion. Toute requête (hors authentification) non authentifiée ne sera pas traitée. Le site étant public, la création de nouveau compte pourra seulement être effectuée par un administrateur via une page spéciale. La création d'un compte uniquement par des administrateurs permettra aussi que des utilisateurs légitimes ne s'octroient pas des priviléges trop élevés.</p> <p>L'authentification se fera via des JSON Web Token (JWT), ces jetons auront une durée d'expiration de 10 minutes. Toutes les 10 minutes, le jeton devra être rafraîchi via une requête au serveur. Le jeton sera stocké côté client dans le navigateur via l'API SessionStorage, avec sa date d'expiration. Cette approche permet de simplifier le développement mais est vulnérable aux attaques par Cross-site scripting, et devra être modifiée en production.</p> <p>Une fois l'authentification effectuée, le navigateur redirige l'utilisateur vers une page d'accueil affichant un message de bienvenue.</p> <p>Sur toutes les pages (sauf celle d'authentification), un menu permet d'accéder à chacune des 3 pages du site (page d'accueil, page du service finances et page du service logistique). Ce menu sera programmé à l'aide de composants web en utilisant React (JavaScript) afin de pouvoir l'appliquer sur les autres pages sans avoir à dupliquer de code. Nous pourrons appliquer nos règles de CSS une seule fois pour chaque page contenant le menu de la même manière.</p> <p>Cela représente un réel gain de temps et le code sera nettement plus clair et plus propre en matière de complexité.</p> <p>La page du service finances sera accessible uniquement par un utilisateur qui a le rôle du service finance. Cette page contient tous les rapports voulus par le service finance avec différents composants pour gérer les dates des rapports. L'idée serait de construire des tableaux HTML (Table) pour la totalité des informations nécessaires au service. Afin d'alléger la page, nous utiliserons un système de boutons auxquels nous appliquerons des actions pour afficher un tableau correspondant à l'intitulé. Cela permettra d'avoir une interface simple et intuitive pour un utilisateur lambda.</p> <p>La page du service finances contient initialement 8 rapports :</p> <ul style="list-style-type: none"> • F1 : Liste décroissante des produits avec les plus grosses remises avec le client concerné et l'information sur le nom de l'employé de la commande • F2 : Liste du chiffre d'affaires par produit avec sous-totaux par catégorie de produit • F3 : Liste du chiffre d'affaires par pays avec le total des frais de port • F4 : Liste du chiffre d'affaires transporté par transporteur
------------------------	--

	<ul style="list-style-type: none"> • F5 : Liste des commandes par employé avec son nom et les remises moyennes qu'il accorde par client • F6 : Liste des chiffres d'affaires et quantités réalisés avec les produits, par fournisseur • F7 : Cumul des frais de port par transporteur avec le nombre d'expéditions • F8 : Top 10 des fournisseurs avec le meilleur taux chiffre d'affaires/nombre de produits vendus. <p>La page du service logistique sera accessible uniquement par un utilisateur qui a le rôle du service logistique. Cette page contient tous les rapports voulus par le service logistique avec différents composants pour gérer les dates du rapport. Le fonctionnement de cette page sera le même que la page dédiée au service finance.</p> <p>La page du service logistique contient initialement 5 rapports :</p> <ul style="list-style-type: none"> • L1 : Liste des plus grands écarts entre la date d'objectif de livraison et la date d'envoi de la commande • L2 : Liste des produits dont le niveau de réappro est déclenché (unité en stock moins unité en commande est inférieur au niveau de réappro) • L3 : Quantités de produit livrées par région et par ville de livraison • L4 : Liste des commandes à livrer ayant un fournisseur dans la même région • L5 : Top 10 des transporteurs avec le plus de livraisons effectuées. <p>Sur les deux pages de rapport, les données doivent pouvoir être filtrées par semaine, mois, trimestre ou année.</p> <p>Des pages spéciales doivent être implémentées pour rediriger un utilisateur lors d'une erreur HTTP 401, 403 ou 404.</p>
DESCRIPTION DE L'API	<p>Toutes les routes qui nécessitent un corps de requête attendent un corps au format JSON. Toutes les réponses d'API seront également au format JSON. Toutes les routes authentifiées attendent dans l'entête HTTP le jeton d'accès JWT. Les routes préfixées par /finances sont uniquement accessibles aux utilisateurs du services finances, les routes préfixées par /logistique sont uniquement accessibles aux utilisateurs du service logistique.</p> <p>Toutes les routes préfixées par /\$SERVICE/rapport où \$\$SERVICE est le nom d'un service, prennent en paramètre dans l'URL, l'un des attributs facultatif suivant :</p> <ul style="list-style-type: none"> • week : Un entier représentant le numéro de la semaine voulue. • month : Un entier représentant le numéro du mois de l'année voulu. • trimester : Un entier représentant le numéro du trimestre de l'année voulu.

L'attribut **year** doit obligatoirement être renseigné, avec un entier représentant l'année voulue. Différentes erreurs HTTP peuvent être renvoyées, une erreur 401 pour un utilisateur non authentifié, une erreur 403 pour un utilisateur qui n'a pas le bon rôle ou bien une erreur 400 pour une erreur de syntaxe dans les paramètres de requêtes..

Routes non authentifiées :

- **POST /auth/login** : Prend en paramètre un corps qui contient le mot de passe et l'identifiant de l'utilisateur. Renvoie le jeton JWT ainsi que sa date d'expiration et le rôle de l'utilisateur.
- **GET /auth/logout** : Prend en paramètre dans l'entête HTTP le jeton JWT. Invalidate le jeton et renvoie un message de statut HTTP.
- **GET /auth/keepalive** : Prend en paramètre dans l'entête HTTP le jeton JWT, génère un nouveau jeton JWT et le renvoie ainsi que sa date d'expiration. Renvoie une erreur HTTP 419 si la date d'expiration a été dépassée.

Routes authentifiées :

- **GET /home** : Renvoie le texte de la page d'accueil personnalisé ou une erreur HTTP 401 pour un utilisateur non authentifié.
- **GET /user/role** : Renvoie le rôle de l'utilisateur actuel ou une erreur HTTP 401 pour un utilisateur non authentifié.
- **GET /finances/rapport/f1** : Renvoie les données pour le rapport F1 du service finances.
- **GET /finances/rapport/f2** : Renvoie les données pour le rapport F2 du services finances.
- **GET /finances/rapport/f3** : Renvoie les données pour le rapport F3 du service finances.
- **GET /finances/rapport/f4** : Renvoie les données pour le rapport F4 du services finances.
- **GET /finances/rapport/f5** : Renvoie les données pour le rapport F5 du service finances.
- **GET /finances/rapport/f6** : Renvoie les données pour le rapport F6 du services finances.
- **GET /finances/rapport/f7** : Renvoie les données pour le rapport F7 du service finances.
- **GET /finances/rapport/f8** : Renvoie les données pour le rapport F8 du services finances.
- **GET /logistique/rapport/l1** : Renvoie les données pour le rapport L1 du service logistique.
- **GET /logistique/rapport/l2** : Renvoie les données pour le rapport L2 du services logistique..
- **GET /logistique/rapport/l3** : Renvoie les données pour le rapport L3 du service logistique.
- **GET /logistique/rapport/l4** : Renvoie les données pour le rapport L4 du services logistique.

	<ul style="list-style-type: none"> • GET /logistique/rapport/L5 : Renvoie les données pour le rapport L5 du service logistique.
INTERFACE DE RESTITUTION	
CAHIER DE RECETTES DE TESTS	<p>Création d'une image Docker pour la mise en place des tests côté serveur.</p> <p>Plusieurs tests seront effectués :</p> <ul style="list-style-type: none"> • Tests de sécurité : vérification de l'authentification, vérification de la révocation des tokens, vérification de l'accès des différents rôles. • Tests unitaires : utilisation de JUnit pour effectuer des tests unitaires sur les différents composants de l'API. • Test de l'interface : vérification de l'intégrité du site sur plusieurs navigateurs, tests des composants de l'interface (sélecteurs de dates, boutons).
CONTRAINTEs	<p>contraintes de délais.</p> <p>Plusieurs contraintes juridiques doivent être respectées. En premier lieu, la conformité au RGPD (Règlement Général sur la Protection des Données) est impérative, particulièrement pour le traitement des données personnels des clients et des fournisseurs personnes physiques. Cela implique la mise en place de mesures techniques et organisationnelles appropriées pour garantir la sécurité des données, notamment leur chiffrement et leur pseudonymisation.</p> <p>L'entreprise RIYOBRU devra également maintenir un registre des activités de traitement détaillant de la nature des données collectées, leur finalités, et leur durée de conservation.</p> <p>Conformément aux exigences de la CNIL, des mesures renforcées doivent être implémentées pour prévenir toutes fuites de données, notamment dans la mise en place d'un système d'authentification robuste et une traçabilité des accès au datawarehouse. L'entreprise devra également s'assurer que les contrats avec les éventuels sous-traitants incluent des clauses spécifiques à la protection des données personnelles.</p>
BILAN	durée de réalisation, résultats du calendrier de recettes (tests unitaires, fonctionnels, etc...), retour client ?, respect des objectifs ?, visualisation des résultats obtenus.
RESSOURCES	<u>Merise et le cahier des charges</u> <u>Cahier des charges - Conception d'une base de données</u> <u>Ressources de développement logiciel</u>