**UNIVERSITE JOSEPH KI-ZERBO**

**(UJKZ)**

**-=-=-=-=-**

**INSTITUT BURKINABE DES ARTS ET METIERS**

**(IBAM)**

****

**Master informatique**

**PROJET DE DEVELOPPEMENT A BASE DE COMPOSANT ET SERVICES WEB**

**GROUPE 6**

**EXERCICE 1** : **Application RESTful avec Sprint Boot**

**Objectif :** Développer une application RESTful en utilisant Spring Boot qui utilise Talend

Open Studio pour se connecter à une base de données MySQL/PostgreSql, avec monitoring

via Prometheus.

|  |  |
| --- | --- |
| **Membre du groupe** | **Professeur** |
| KINI Ingrid Jessica | Mr Ky Oumar |
| KOHOUN Rachel |  |
| KOFFI Amoin Amandine Josiane |  |
| TRAORE Dilobiè Djibril Daniel |  |
| SAWADOGO More |  |

**Année Académique 2023-2024**

[1.1 Objectifs du Projet 4](#_Toc177593610)

[1.2 Résultats attendus 4](#_Toc177593611)

[1.3 Présentation des technologies utilisées 4](#_Toc177593612)

[1.3.1 JDK (Java Development Kit) 4](#_Toc177593613)

[1.3.2 Spring Boot 3.3.3: 4](#_Toc177593614)

[1.3.3 STS (Spring Tool Suite-4.4.23.1) : 5](#_Toc177593615)

[1.3.4 Postman 7.36.7 5](#_Toc177593616)

[1.3.5 PostgreSQL 12: 5](#_Toc177593617)

[1.3.6 Talend Open Studio : 5](#_Toc177593618)

[1.3.7 Prometheus 5](#_Toc177593619)

[1.4 Architecture du système 5](#_Toc177593620)

[Fonctionnement de l’architecture : 6](#_Toc177593621)

[1.5 Installation et configuration des outils 7](#_Toc177593622)

[1.5.1 Postman 7](#_Toc177593623)

[1.5.2 Spring Boot 7](#_Toc177593624)

[1.5.3 Talend Open Studio 10](#_Toc177593625)

[1.5.4 PostgreSQL 13](#_Toc177593626)

[1.5.5 Prometheus 14](#_Toc177593627)

[1.6 Exécution et test de l’api 16](#_Toc177593628)

[1.6.1 Exécution de l’api 16](#_Toc177593629)

[1.6.2 Test sur postman 16](#_Toc177593630)

[1.6.3 Test sur Talend 19](#_Toc177593631)

[1.6.4 Surveillance avec prometheus 22](#_Toc177593632)

**Introduction**

Le mot data est de plus en plus utilisé au seins des entreprises, des écoles, voir même dans la sociéte ….,la donnée représente le nouveau pétrol. Devant donc l’afflue et la masse de données quotidiennes, nous devons régulièrement traité, analysé et exporté ces données pour aidé à la prise de décision en entreprise ; et pour ce faire nous devons utilisé un outils ETL(Extraction, Transformation et Chargement) qui s’appelle Talend.

Dans le sujet : développer une application RESTful utilisant Spring Boot qui utilise Talend Open Studio pour se connecter à une base de données Mysql/PostgreSql, avec monitoring via Prometheus, nous avons opté pour le developpement d’une solution permettant la gestion d’une bibliothèque. Les technologies utilisées sont : **Sprint boot** pour le developpement de l’api, **Talend Open Studion for Dtat Integration** pour l'ETL, **PostgreSql** pour la gestion de la base de données, **Prometheus** pour la surveillance (monotoring) de l’api et **Postman** pour tester les fonctionnalités de l’api.

Le présent rapport se structure en plusieurs sections. Après avoir défini les objectifs, nous décrierons l'architecture du système et les technologies utilisées, ensuite la configuration des outils utilisés et enfin, nous présenterons les tests réalisés avec Postman, Talend et prometheus.

## Objectifs du Projet

L’objectif principale du projet est de développer une application RESTful en utilisant Spring Boot qui utilise Talend Open Studio pour se connecter à une base de données MySQL/PostgreSQL, avec monitoring via Prometheus.et tests avec Postman.

De façon spécifique il s’agit de :

* Développer une API RESTful pour la gestion d’une bibliothèque (mais nous nous limiteront à deux table auteurs et livres),
* Implémenter des processus ETL avec Talend Open Studio pour l’extraction, la transformation et le chargement de données,
* Utiliser PostgreSQL pour la gestion de la base de données.
* Effectuer les tests de fonctionnalités avec Postman,
* Assurer la surveillance avec Prometheus.

## Résultats attendus

Les résultats attendus sont :

* Livrer une api de gestion (Created, Read, Updare, Delete) des livres et leurs auteurs fonctionnels
* Assurer un flux de données efficace entre l’api et la base de données à travers l’ETL
* Fournir un jeu de test.

## Présentation des technologies utilisées

## JDK (Java Development Kit)

C'est un ensemble d'outils de développement pour créer et exécuter des applications Java. Il inclut le compilateur Java, des bibliothèques et l'environnement d'exécution Java (JRE). Le JDK est essentiel pour tout développeur travaillant sur des projets Java.

Nous avons utilisé la version 17 pour Spring boot et la version 8 pour Talend

## Spring Boot 3.3.3:

C'est un framework basé sur Java qui facilite la création d'applications web et microservices. Spring Boot réduit le besoin de configuration manuelle en proposant une structure préconfigurée et des conventions. Il permet de démarrer rapidement un projet avec peu de configuration

## STS (Spring Tool Suite-4.4.23.1) :

C'est un IDE (environnement de développement intégré) basé sur Eclipse et spécialement conçu pour les développeurs travaillant avec le framework Spring. Il propose des outils et des plugins pour faciliter le développement avec Spring et Spring Boot.

## Postman 7.36.7

C'est un outil pour tester les API REST. Il permet de créer, tester et documenter facilement les requêtes API en envoyant des requêtes HTTP et en inspectant les réponses.

## PostgreSQL 12:

C’est un système de gestion de base de données relationnelle open source, réputé pour sa robustesse, son évolutivité et sa conformité aux standards SQL. Il prend en charge une large gamme de types de données, des transactions ACID, et est couramment utilisé pour les applications nécessitant une base de données fiable.

## Talend Open Studio :

C’est un outil open source d'intégration de données (ETL) qui permet d'extraire, transformer et charger des données entre différents systèmes. Il est utilisé pour automatiser les processus d'intégration de données dans divers environnements.

## Prometheus

C’est un outil open source de surveillance et d'alerting conçu pour collecter, stocker et analyser des données sous forme de séries temporelles (time-series). Il est principalement utilisé pour surveiller les performances des systèmes, applications, et infrastructures. Prometheus collecte les métriques en scrappant des endpoints définis à des intervalles réguliers, et offre un langage de requête puissant (PromQL) pour analyser les données. Conception et Architecture du Système

## Architecture du système

L’architecture est composée de 3 niveaux a savoir un client qui fournit l’interface Postman, un serveur d’application qui contient la logique métier et un serveur de base de données qui gère la persistance des données.

Les technologies utilisées nous a permis d’opter pour une architecture trois tiers a savoir :

* le client (postman),
* serveur d’application(Tomcat) ,
* un serveur de base de données (postgreSQl)











**Figure 1: Architecture du Système**

### Fonctionnement de l’architecture :

1. **Postman** : Utilisé pour tester les API exposées par Spring Boot (CRUD, ETL, etc.).
2. **Spring Boot** : Fournit des endpoints pour interagir avec la base de données PostgreSQL et exécuter des workflows via Talend.
3. **Talend** : Exécute des jobs ETL pour manipuler et traiter les données en se connectant à PostgreSQL.
4. **Prometheus** : Surveille les métriques de performance et de santé de l’api Spring Boot, ainsi que des jobs Talend.

## Installation et configuration des outils

### ****Postman****

##### **Installation** :

* Téléchargez la version de Postman pour votre système d’exploitation depuis le site officiel.
* Installez l'application en suivant les instructions propres à votre système (Windows, macOS, ou Linux).

##### **Configuration** :

* Après l'installation, créez un compte ou connectez-vous.
* Organisez les requêtes HTTP en collections pour faciliter les tests.

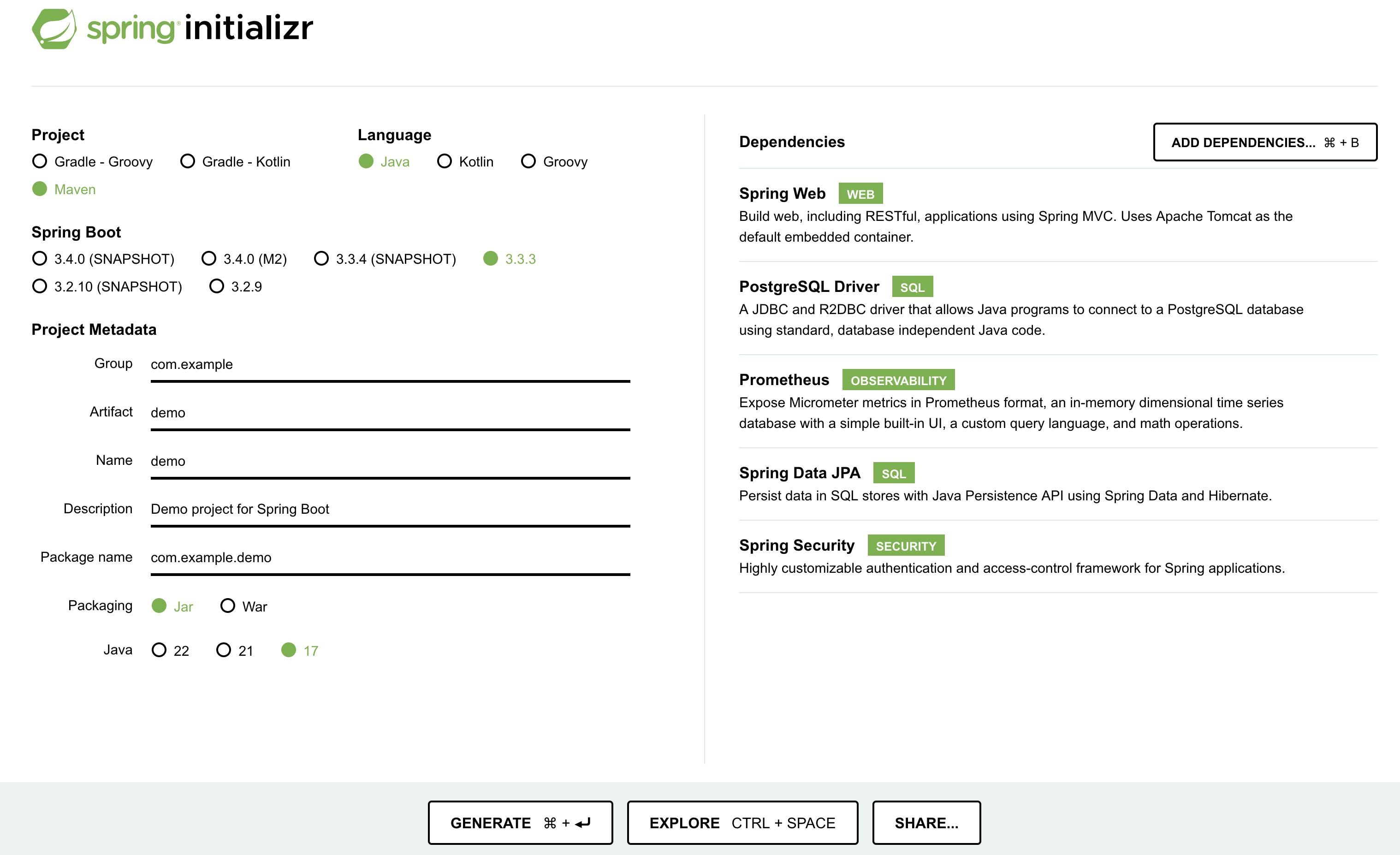
##### **Utilisation** :

* **Créer une requête** : Choisissez la méthode HTTP (GET, POST, PUT, DELETE, etc.), entrez l'URL et envoyez la requête.
* **Inspecter la réponse** : Analysez les réponses JSON, XML ou HTML fournies par l'API.

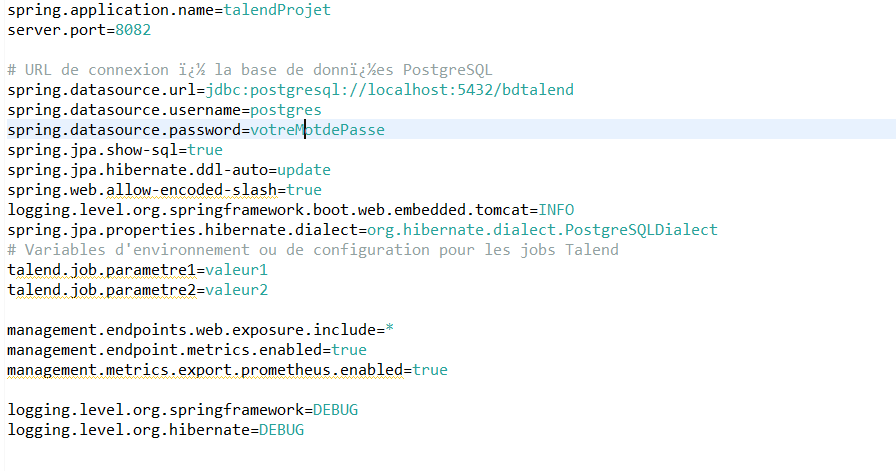
### ****Spring Boot****

##### **Installation** :

* Téléchargez et installez le **JDK** (Java Development Kit) version 17 ou supérieure depuis [Oracle](https://www.oracle.com/java/technologies/javase-jdk17-downloads.html).
* Utilisez **Spring Initializr** (<https://start.spring.io/>) pour créer un projet Spring Boot configuré.
* Ajouter les dépendances nécessaires pour le projet et cliquer sur générer pour générer le projet
* Importer le fichier (.zip) générer dans votre IDE

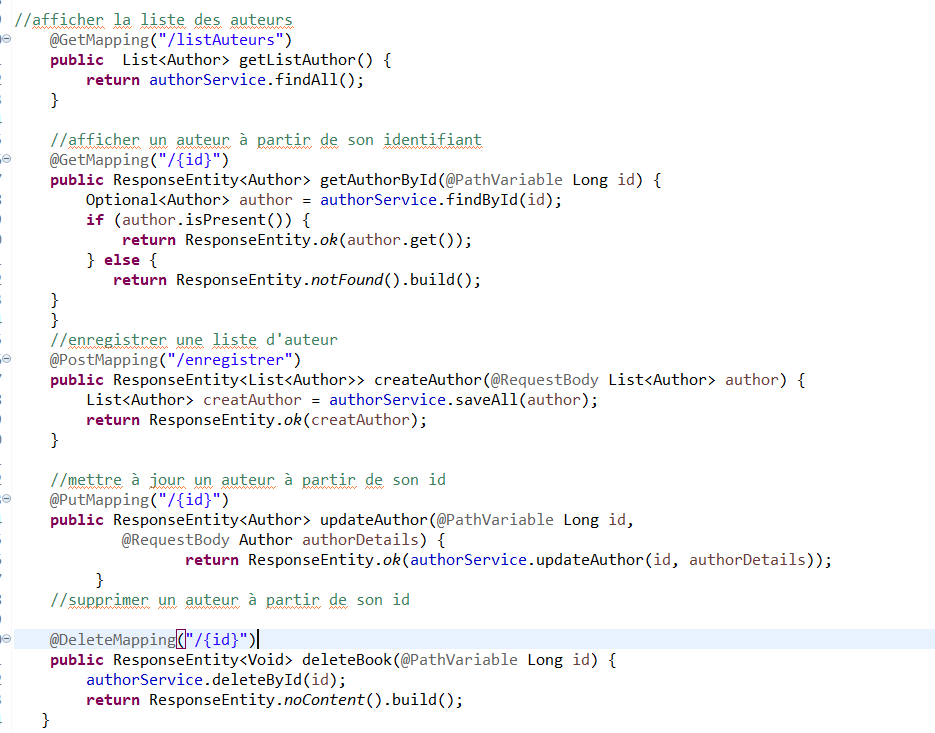


##### **Configuration** :

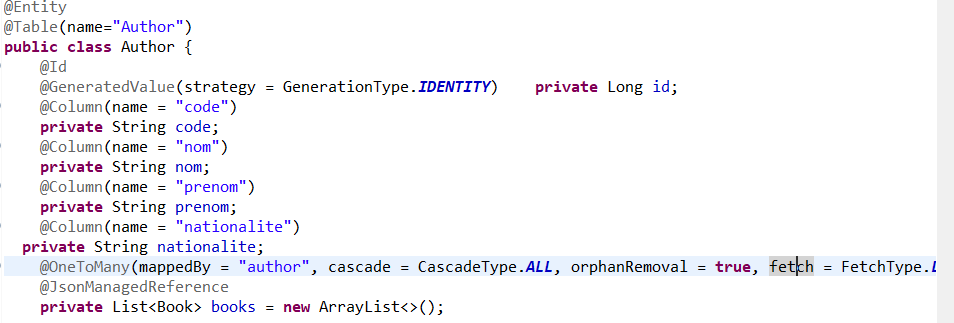
* Dans le fichier application.properties ou application.yml ajouter les propriétés de l'application (port, base de données, logs, etc.
* 

##### **Utilisation** :

* **Création d'une API REST** : Utilisez l'annotation @RestController pour créer des endpoints HTTP.



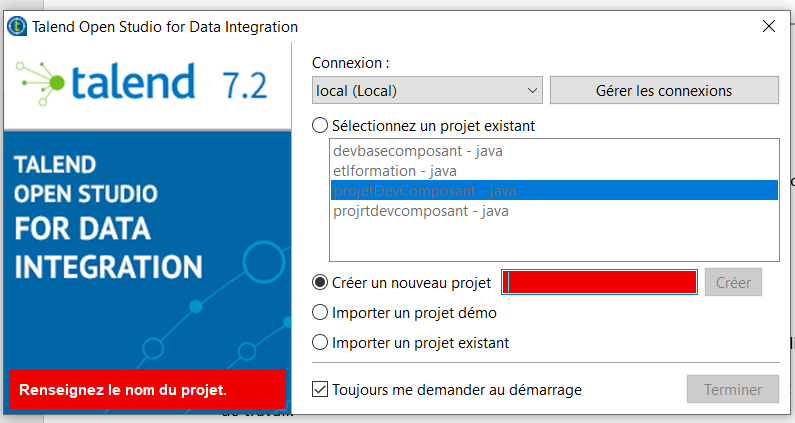
* **Explication des endpoints**
* @GetMapping("/listAuteurs")**:** pour récupérer tous les auteurs de la base de données.
* @PostMapping("/enregistrer")**:** pour créer un (ou plusieurs) auteurs .
* @PutMapping("/{id}")**:** pour mettre à jour un auteur existant.
* @DeleteMapping("/{id}")**:** pour supprimer un auteur.
* **Gestion de la persistance** : Configurez une base de données relationnelle avec JPA/Hibernate en intégrant les annotations @Entity, @Repository, etc.



### ****Talend Open Studio****

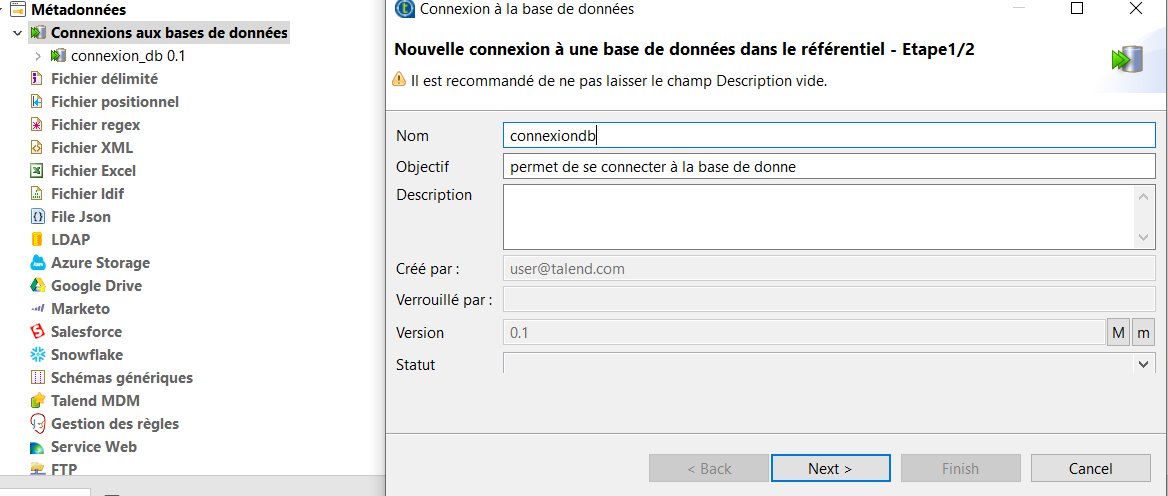
##### **Installation** :

* Téléchargez **Talend Open Studio** depuis le site officiel.
* Installez Java JDK si ce n'est pas déjà fait, car Talend dépend de Java pour fonctionner.
* Démarrez Talend et créez un nouveau projet(saisir le nom du projet) et cliquer sur créer.

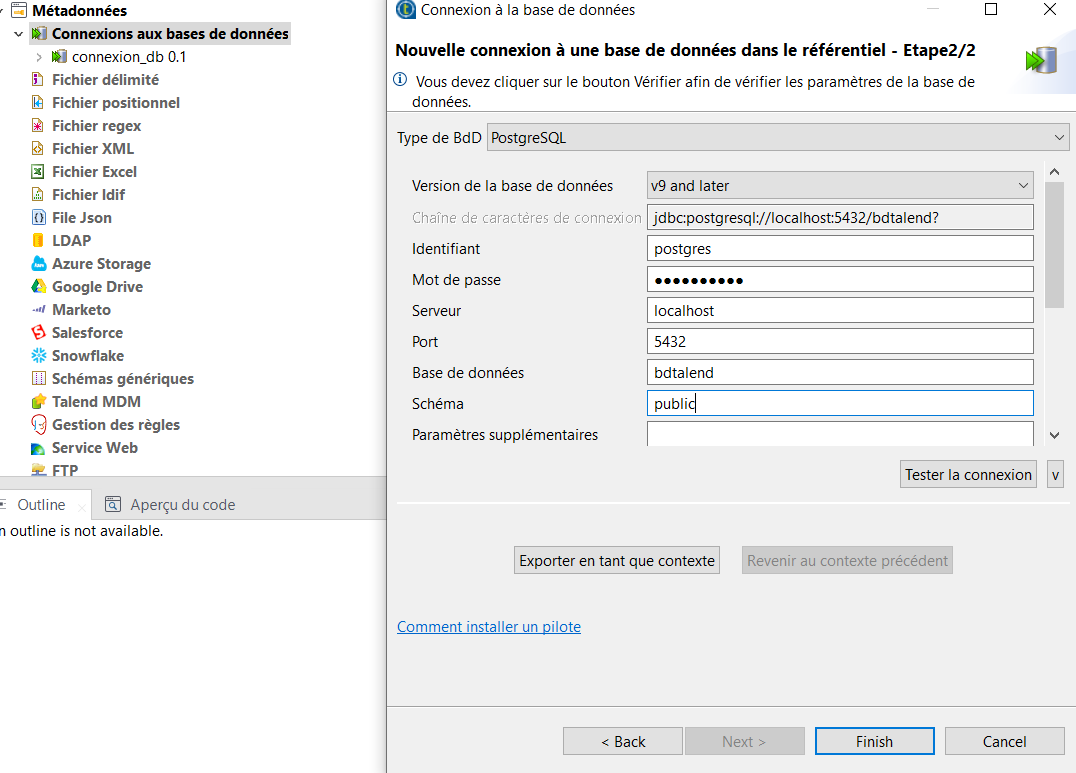


##### **Configuration** :

* Connectez-vous à une base de données comme PostgreSQL en utilisant le composant tDBConnection.
* Faire un clic droit sur métadonnées/connexion aux base de données (saisir le nom de la connexion)

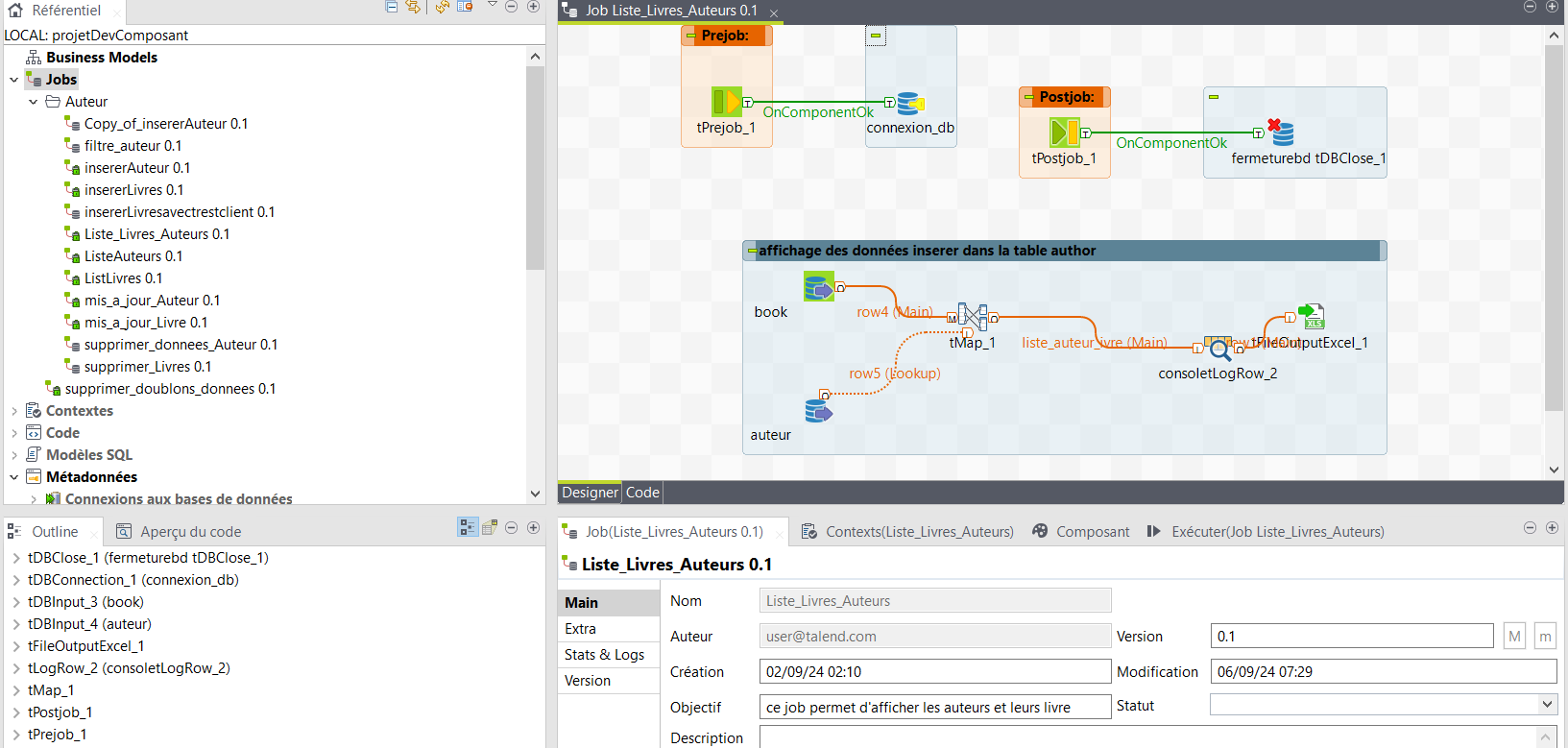


* Faire next pour obtenir la fenêtre ci-dessous :
* Sélectionné la base de donnée voulue et saisir vos paramètres de connexion à la base



##### **Utilisation** :

* **Création d’un job ETL pour l’extraction** : Faire clic droit sur job/créer job et glissez-déposez des composants comme tFileInput, tMap, tDBOutput, tRestClient. Double cliquer sur chaque composant utilisé pour les configurations spécifiques
* **tRestClient** : Utilisé pour appeler des services web via une API REST. Ce composant permet d'extraire des données au format JSON ou XML depuis une API.
* **tHttpRequest** : Permet d'envoyer des requêtes HTTP (GET, POST, etc.) et de recevoir des réponses de type JSON, XML ou texte brut.



* **Transformation de données** : Utilisez le composant tMap pour mapper et transformer les données entre plusieurs sources.
* **tMap** : Un composant clé pour mapper, transformer et filtrer les données extraites. Il permet de convertir les formats, effectuer des jointures, et modifier les structures de données avant de les charger dans la base de données.
* **Chargement** : Utilisez le composant tPostgresqlOutput ou tDBOutput .
* **tPostgresqlOutput** : Ce composant vous permet de charger les données transformées dans une base de données PostgreSQL. Vous pouvez insérer, mettre à jour ou supprimer des enregistrements dans une table.

### ****PostgreSQL****

##### **Installation** :

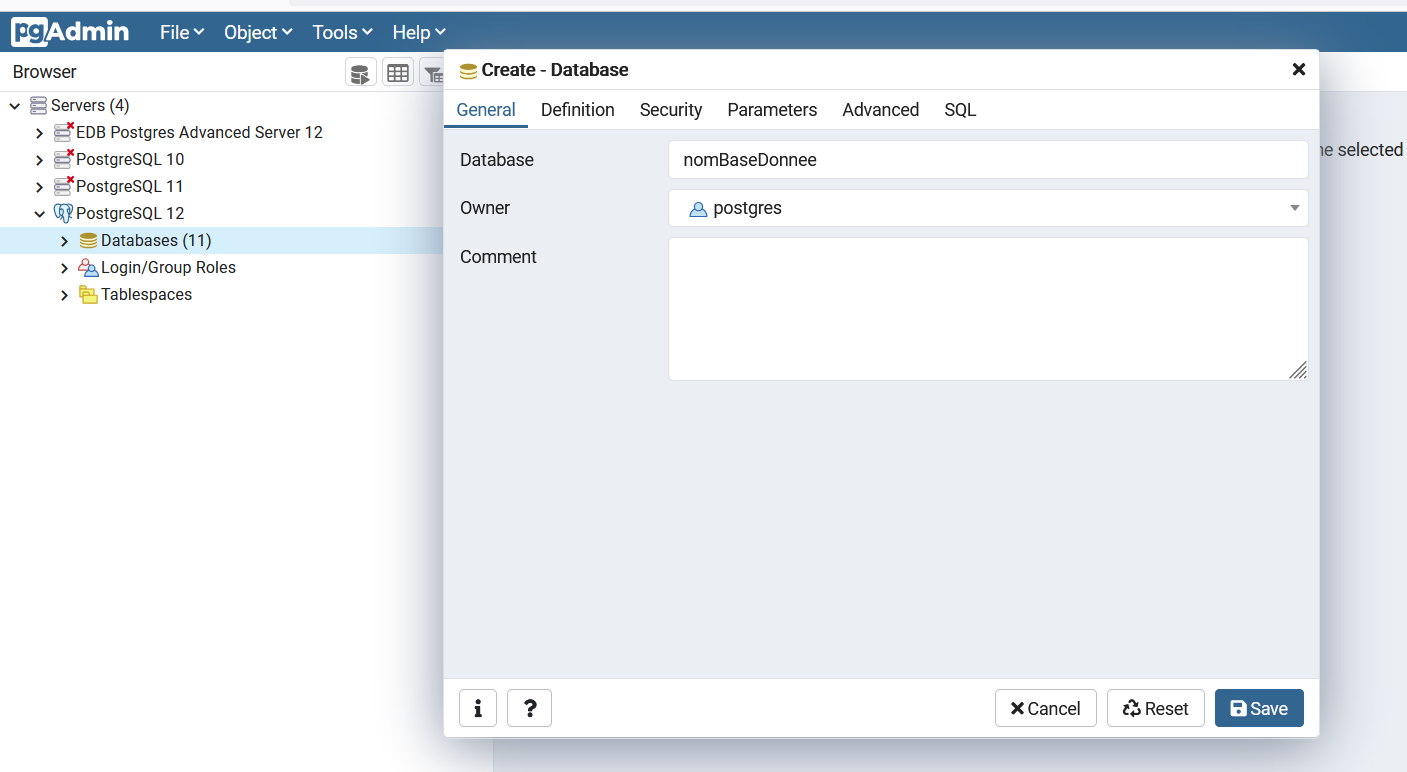
* Téléchargez la dernière version de PostgreSQL depuis [le site officiel](https://www.postgresql.org/download/).
* Suivez les étapes d'installation et assurez-vous que le service PostgreSQL démarre correctement.

##### **Configuration** :

* Configurez le fichier postgresql.conf et pg\_hba.conf pour ajuster les paramètres de connexion et les droits d'accès.
* Utilisez l'outil **pgAdmin** ou la commande psql pour interagir avec la base de données.

##### **Utilisation** :

* **Créer une base de données** : Utilisez l’interface graphique pour créer la base de données



* **Gestion des utilisateurs** : Gérez les utilisateurs et les droits avec CREATE USER et GRANT.

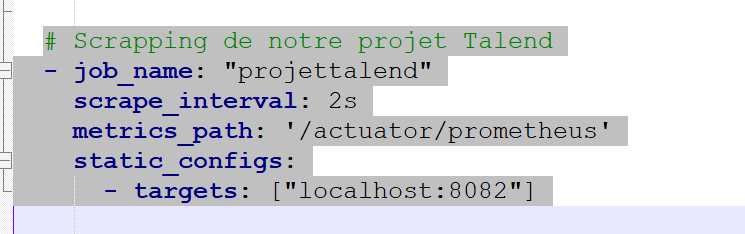
### ****Prometheus****

##### **Installation** :

* Téléchargez Prometheus depuis le site officiel.
* Extrayez et exécutez Prometheus en utilisant le fichier binaire téléchargé.
* Configurez le fichier prometheus.yml pour spécifier les cibles à surveiller.

##### **Configuration** :

* Ajoutez des jobs de scraping dans le fichier prometheus.yml pour surveiller les endpoints exposant des métriques (par exemple, localhost:8082/actuator/prometheus pour Spring Boot)



##### **Utilisation** :

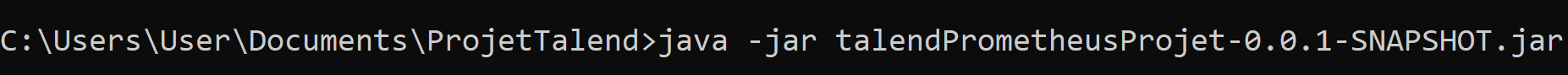
* **Surveiller une application** : Intégrez les bibliothèques client Prometheus dans votre application Spring Boot pour exposer des métriques via l’endpoint /actuator/prometheus.
* **Visualiser les données** : Utilisez **Prometheus Dashboard** pour interroger et visualiser les métriques collectées.

## Exécution et test de l’api

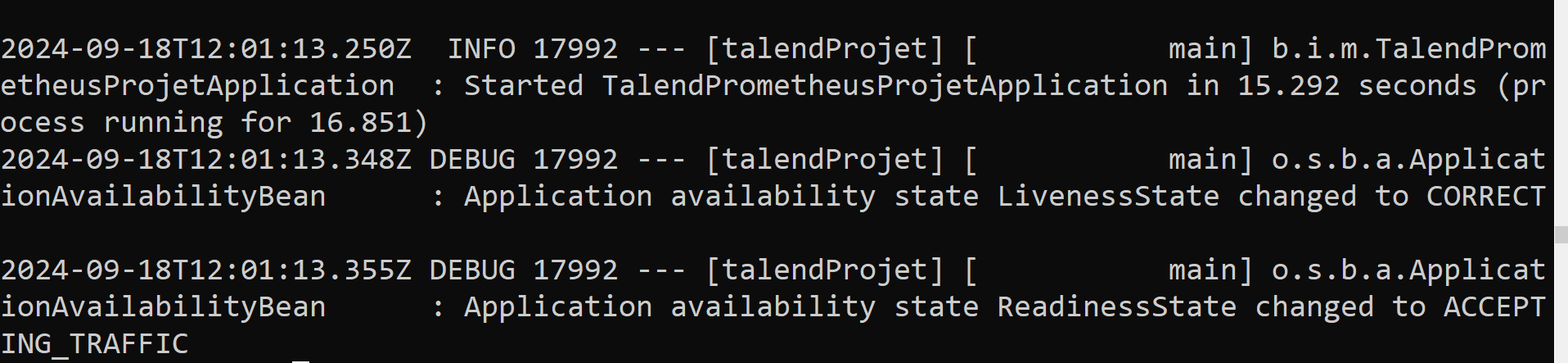
### ****Exécution de l’api****

Se positionner dans le dossier contenant le fichier jar et taper la commande **java –jar NomDuFichier.jar.**

Lorsque l’application démarre elle créé dans la base de données les tables(livre et auteur) contenues dans l’api



A la fin de l’exécution vous verra le message indiquant que l’api est démarrée (*started TalendPrometheusProjetApplication in 15.292 seconds*)

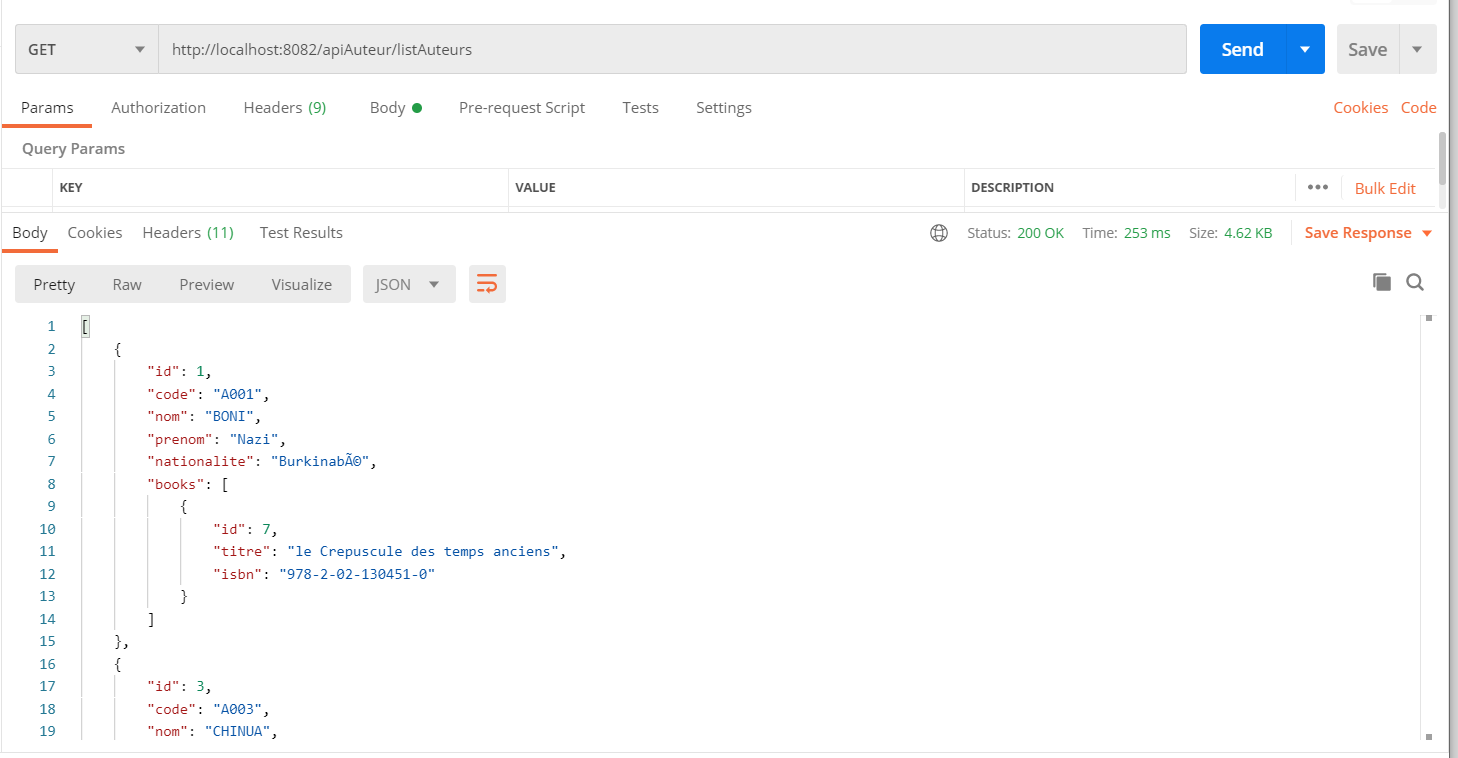


### ****Test sur postman****

Avant d’utiliser Postman, il faut s’assurer que l’APPi RESTful est demarré. Postman retourne les resultat sont format json

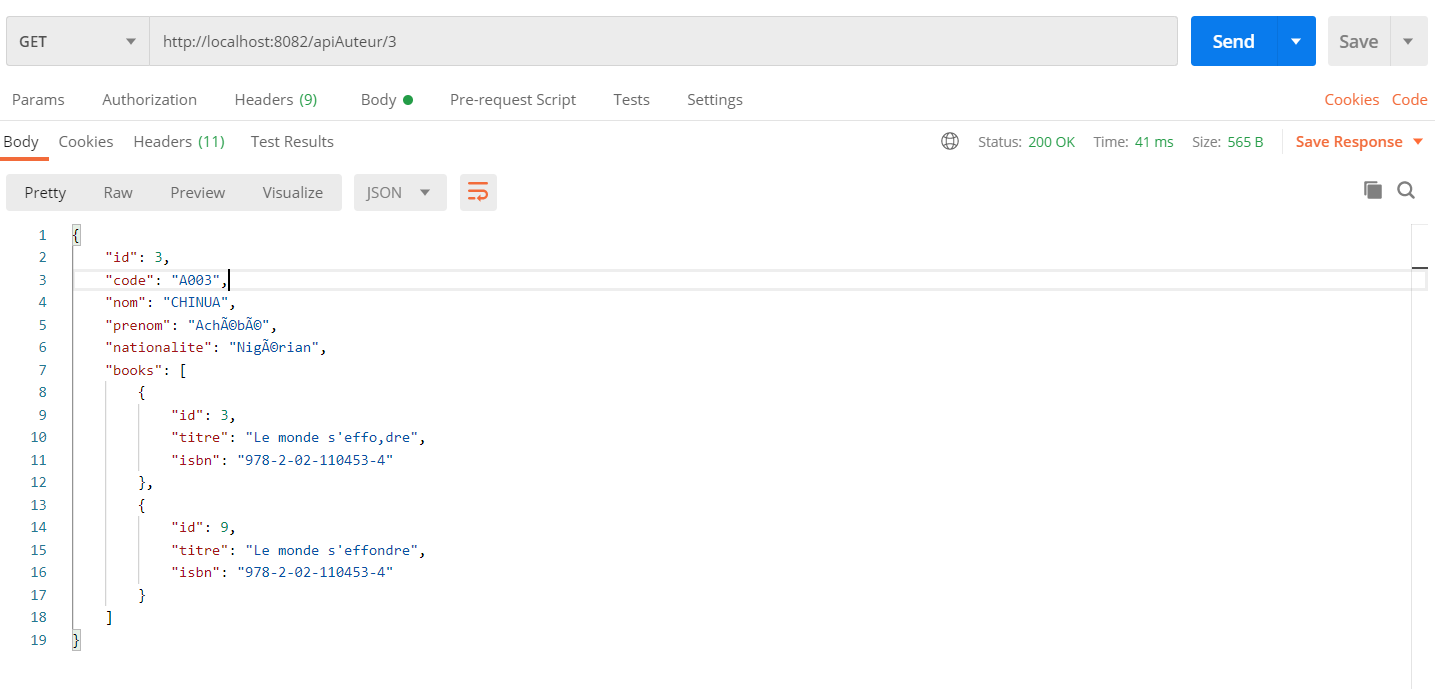
* **Test de l’endpoint qui retourne la liste des auteurs** :

Selectionner la methode **GET** et saisir cette url : <http://localhost:8082/apiAuteur/listAuteurs> **et** <http://localhost:8082/apiBook/listLivres> pour les livres



* **Test de l’endpoint qui retourne un auteur :**

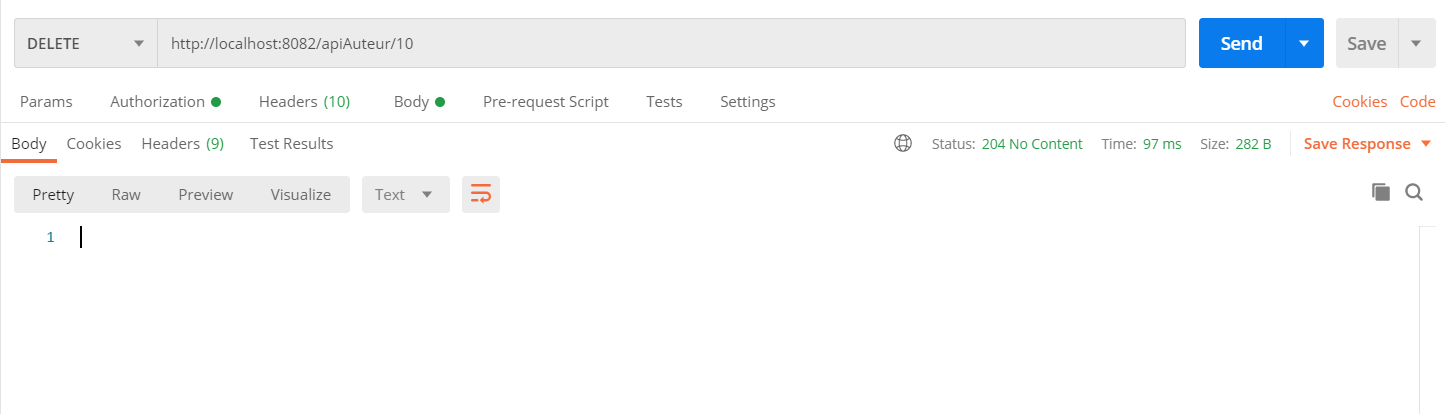
Sélectionner la méthode **GET** saisir cette url : <http://localhost:8082/apiAuteur/3>



* **Test de l’endpoint qui supprime un auteur** :

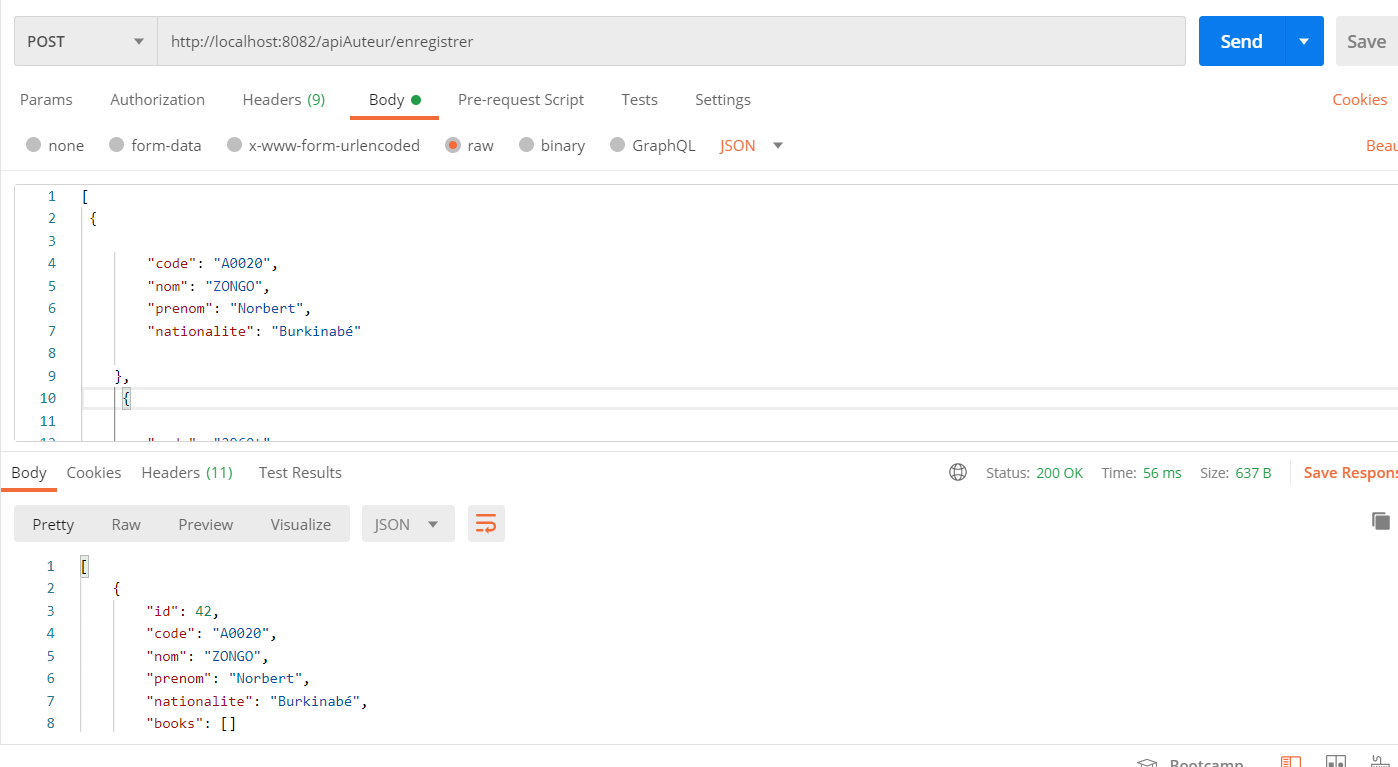
Sélectionner la méthode **DELETE** et saisir cette url : <http://localhost:8082/apiAuteur/10>

Pour supprimer l’auteur qui 10(dix) comme identifiant



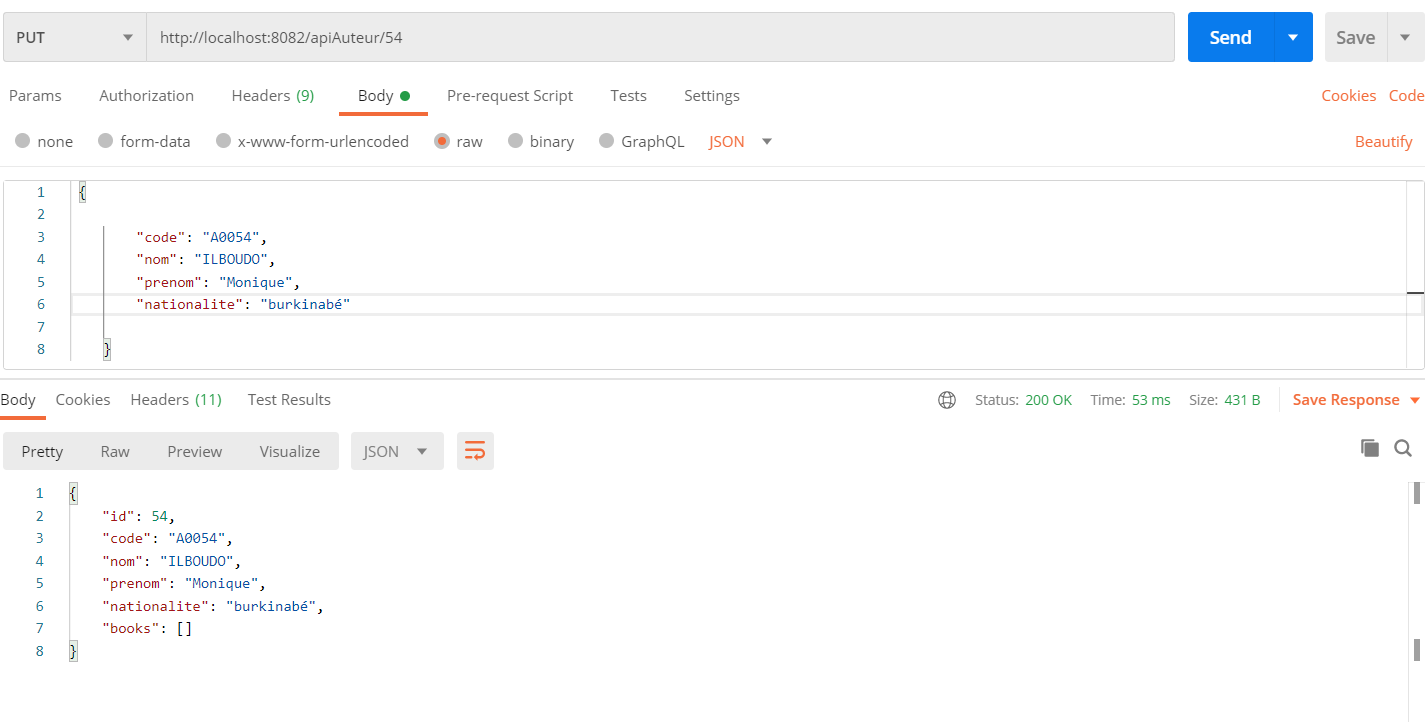
* **Test de l’endpoint qui enregistre liste des auteurs** :

Sélectionner la méthode **POST** et saisir cette url : <http://localhost:8082/apiAuteur/enregistrer> et <http://localhost:8082/apiBook/enregistrer> pour les livres.



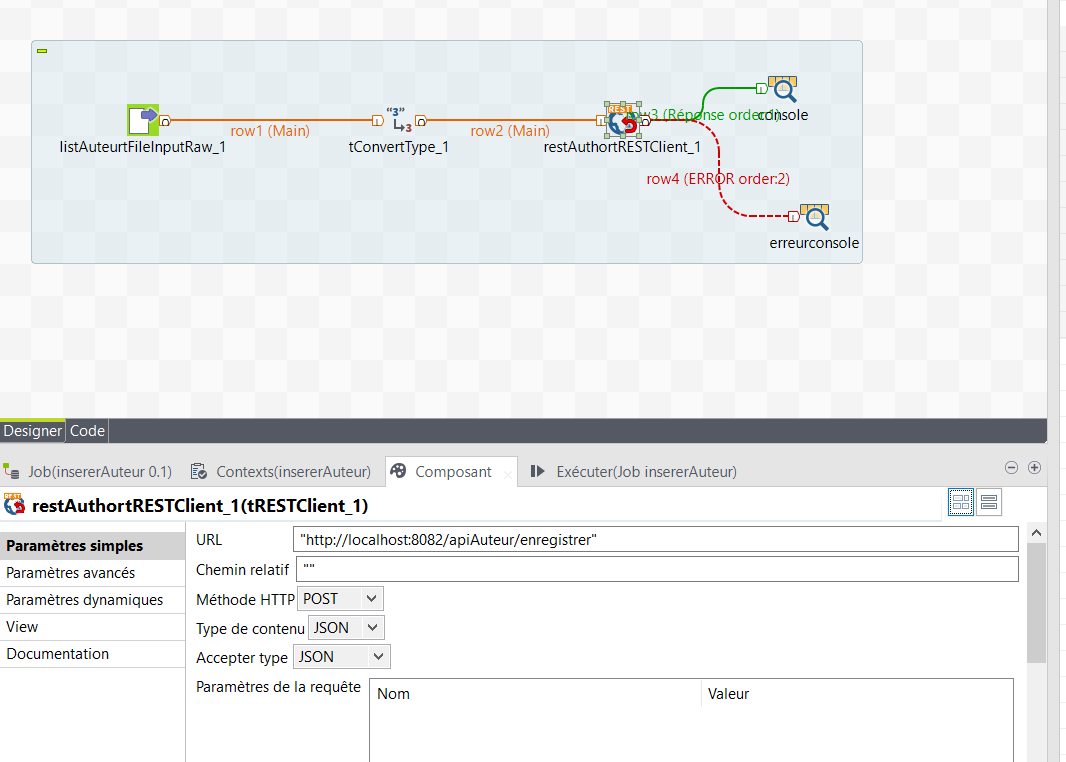
* **Test de l’endpoint qui met à jour un auteur donné** :

Sélectionner la méthode **PUT** et saisir cette url : <http://localhost:8082/apiAuteur/54> et saisir les nouvelle information pour l’acteur à mettre à jour comme indiqué ci-dessous

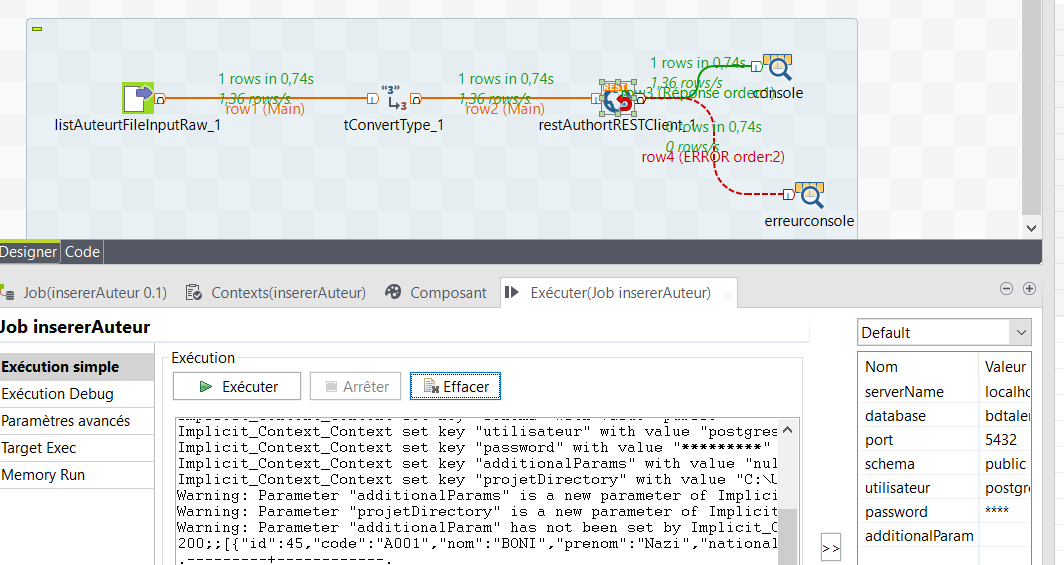


### ****Test sur Talend****

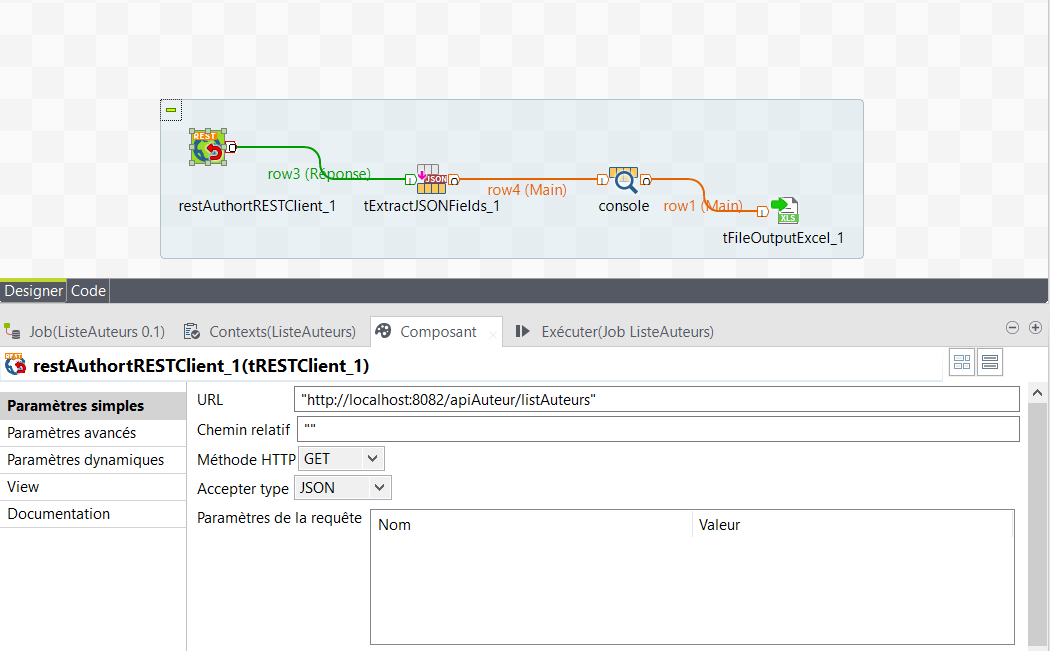
* **Chargement** des auteurs en utilisant le endpoint de l’api



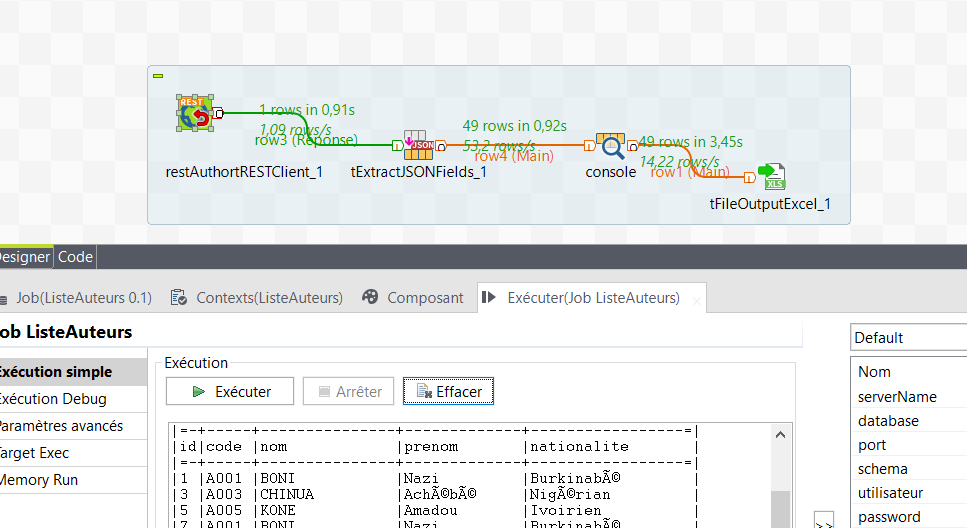
ci-dessous le résultat d’exécution.



* **Extraction** des auteurs à partir de l’api

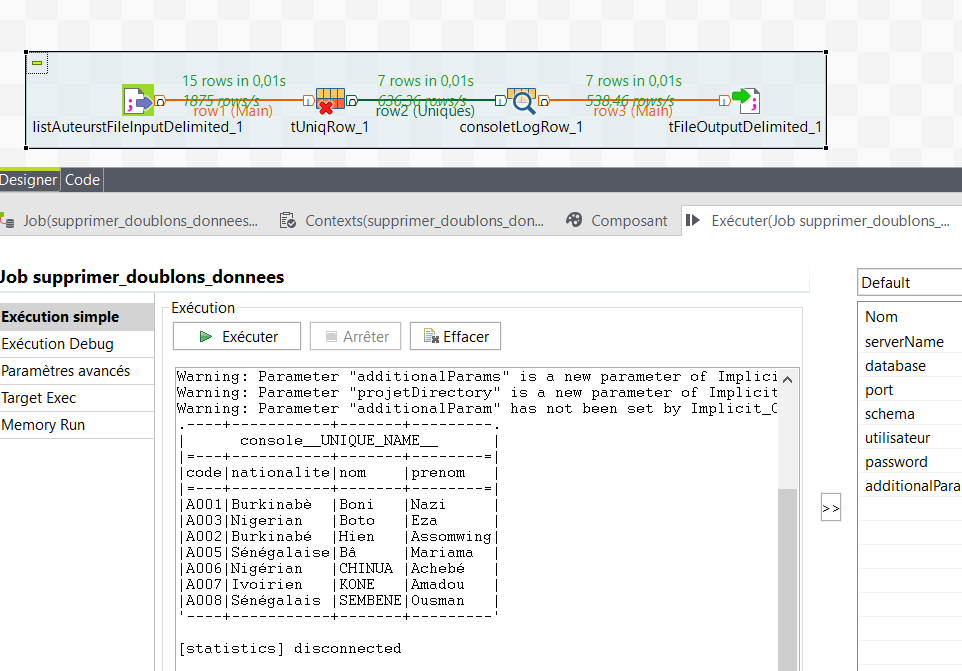


L’exécution donne le résultat ci-après :



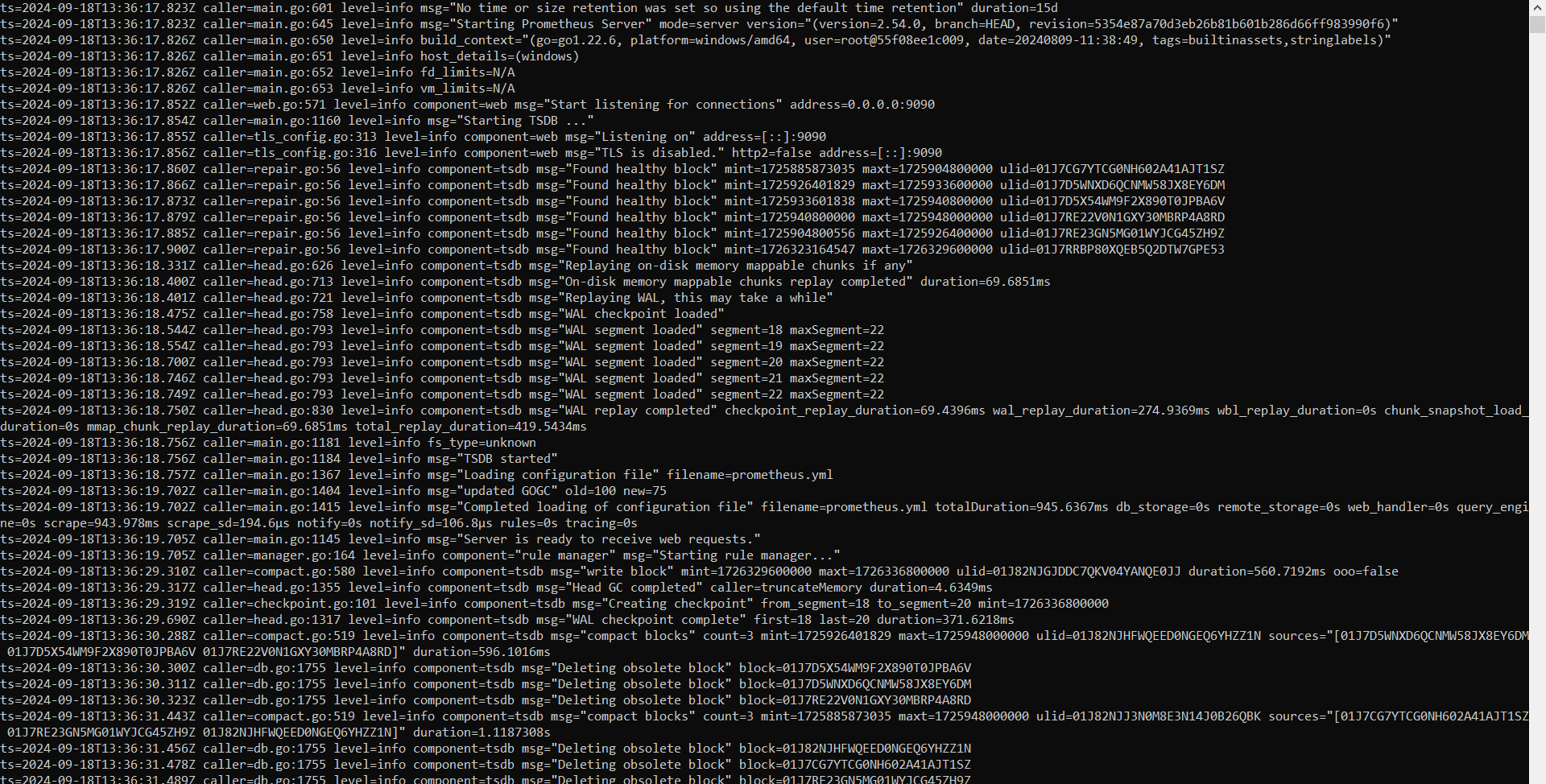
**Transformation** des données (suppression des données)

Dans la capture ci-dessous nous voyant que le fichier de départ contenait 15 lignes et que le résultat après suppression contient 7 lignes.

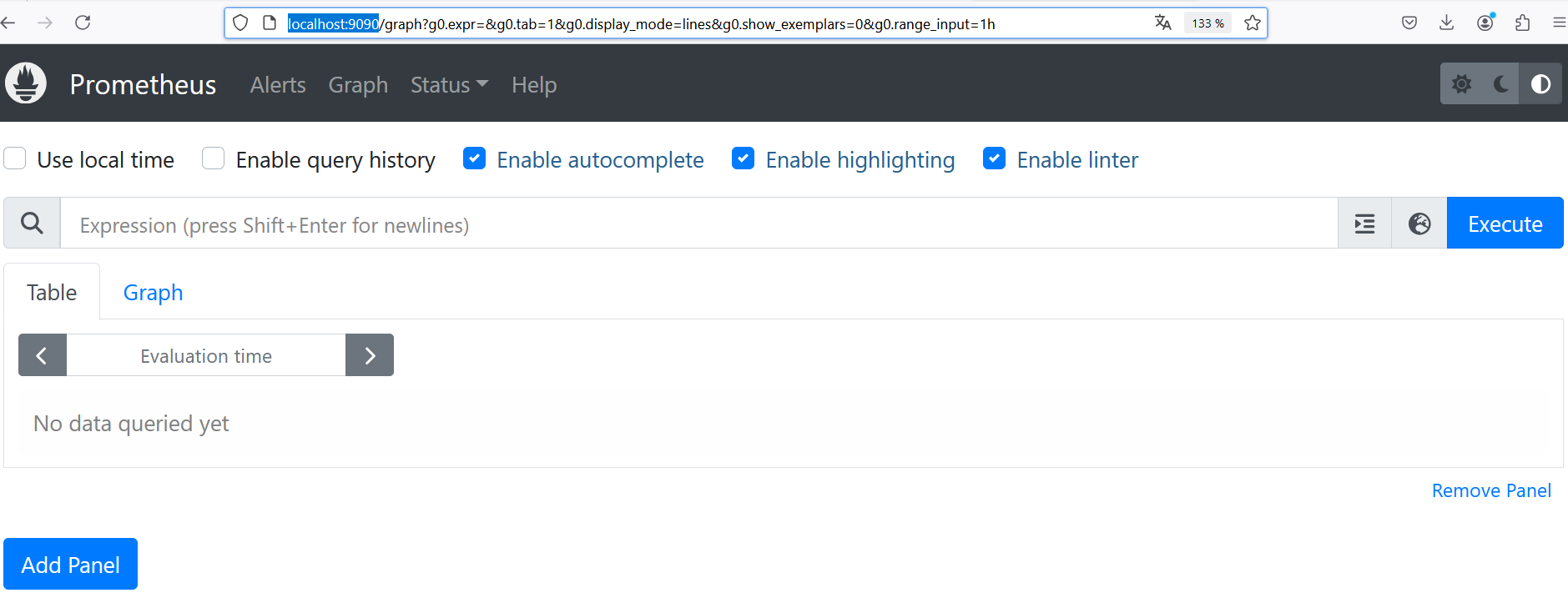


### ****Surveillance avec prometheus****

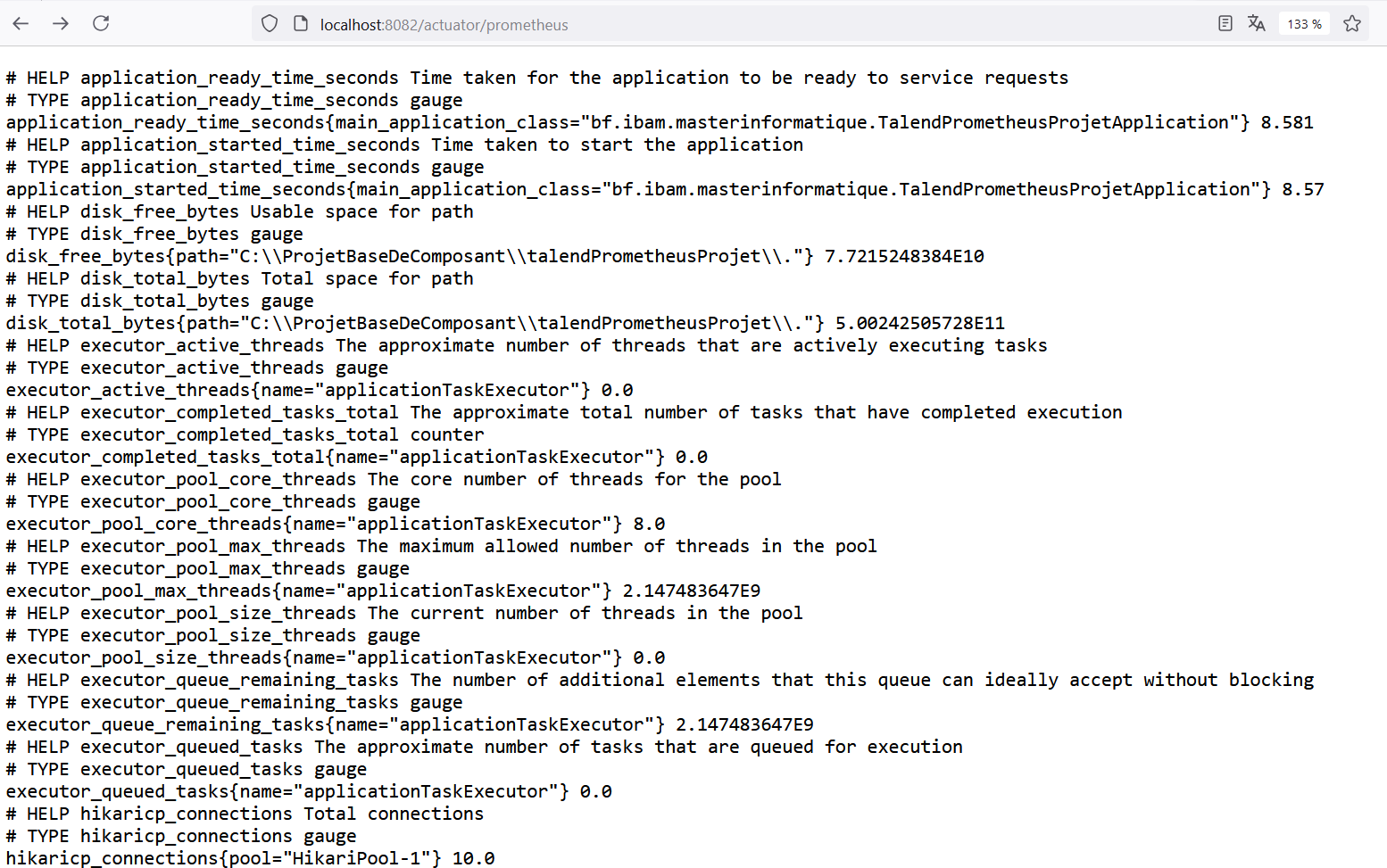
* Double cliquer sur l’exécutable



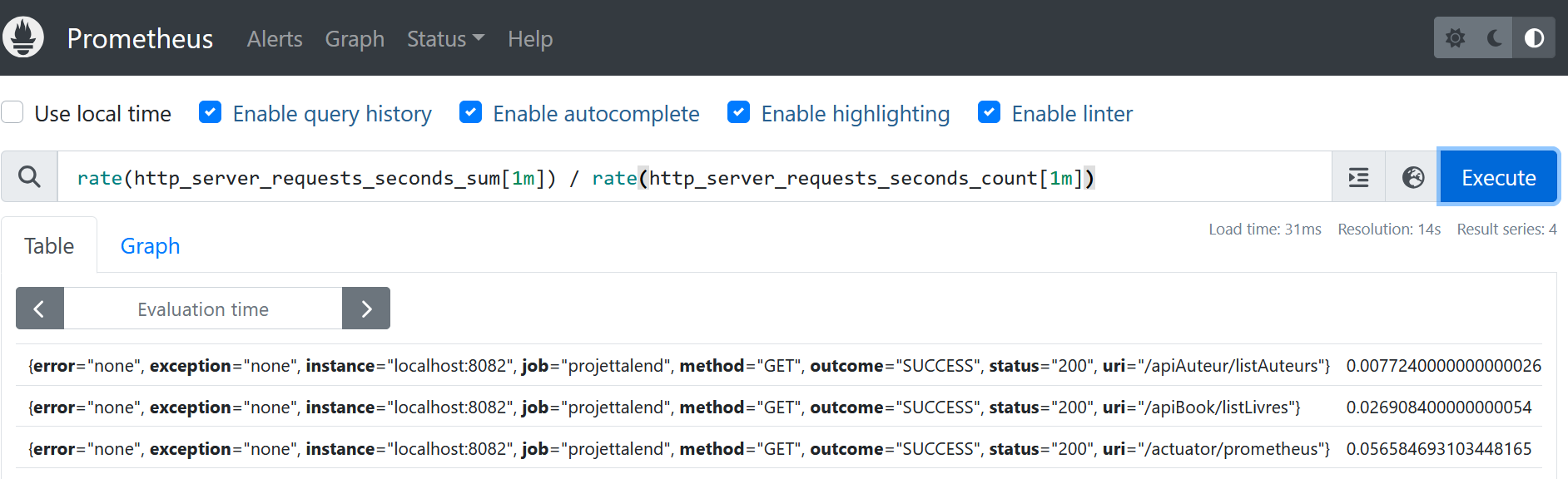
* Pour vérifier le bonne exécution de prometheus, taper cette url <http://localhost:9090> dans un navigateur pour obtenir le résultat ci-après



* Pour afficher les métriques de prometheus saisir cette url <http://localhost:8082/actuator/prometheus>



* La metric rate(http\_server\_requests\_seconds\_sum[1m]) / rate(http\_server\_requests\_seconds\_count[1m]) pour GET des endpoints <http://localhost:8082/apiAuteur/listAuteurs> , <http://localhost:8082/apiBook/listLivres> donne le résultat ci-après :



**Conclusion**

Ce projet a abouti au développement d'une solution intégrée pour la gestion des livres et auteur d’une bibliothèque utilisant des technologies à la pointe. La réalisation de ce projet nous a permis d’apprendre à utiliser les composant de Talend, de comprendre le processus d’intégration de données et de mettre en place des tâche d’intégration de données fiable et facile à maintenir et enfin de maitriser le monitoring d’une api.