Rouen Normandie

Language web
XML
Rapport

API REST

Élèves :
DiarraMohamed El Habib
Haroun Bachar Amine

Enseignant :
HERAUVILLE STEPHANE
REIT ETIENNE

Table des matières

1 Introduction

1.1 Presentation du projet

Ce compte rendu présente la mise en œuvre du projet XML pour le service REST CV24. L'objectif est de fournir une gestion efficace et conforme des CVs en format XML, suivant un schéma XSD spécifique, avec des fonctionnalités de CRUD (Create, Read, Update, Delete) intégrées via une interface REST.

2 Versions et dependances

Le projet utilise Maven pour gérer ses dépendances, avec Spring Boot version 3.2.3 comme base. Les principales dépendances incluent :
Spring Boot Starter Web : pour les fonctionnalités RESTful et la gestion du serveur embarqué.

MySQL Connector Java et MariaDB Java Client : pour la connectivité à la base de données MySQL configurée sur Clever Cloud.

```
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>8.0.33</version>
</dependency>
<dependency>
    <groupId>org.mariadb.jdbc</groupId>
    <artifactId>mariadb-java-client</artifactId>
    <version>2.7.2</version>
</dependency>
```

Figure 1 – Dependance mysql

Spring Boot Starter Data JPA : pour la gestion de la persistance via JPA.
Jakarta XML Bind API :pour le traitement des données XML. D'autres librairies pour le support XML et le développement de tests.
Jackson data format :nécessaire pour la conversion en HTML.

```
<dependency>
    <groupId>com.fasterxml.jackson.dataformat</groupId>
    <artifactId>jackson-dataformat-xml</artifactId>
</dependency>
```

Figure 2 – Dependance jackson

2.1 Base de données

La base de données MySQL est configurée pour interagir avec l'application via Spring Data JPA, facilitant la gestion des entités CV et leur persistance. Les détails de connexion sont spécifiés dans le fichier application.properties. Cette base de données est hebergée sur le serveur clervercloud . Le choix de la bdd hebergée à été preferée sur un deploement à l'aide de conteneur car le deploiement s'effectue jamais lorsque nous utilisont des conteneurs et ceci meme malres la modification du fichier maven

2.2 Gestion de la persistance

La gestion de la persistance des données est réalisée avec Spring Data JPA, qui offre une interface simplifiée pour la manipulation des entités de la base de données.

Application.properties : Les détails de connexion à la base de données MySQL sont configurés dans application.properties, facilitant la gestion des connexions et des transactions.

```
# DataSource Configuration
spring.datasource.url=jdbc:mysql://b3k9ig4cm1xrnpm7oydn-mysql.services.clever-cloud
spring.datasource.username=umztbejdmqubcmrp
spring.datasource.password=7ngmJ0fcdBDgYRDnr0X4
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
```

Figure 3 – Application.properties

Repository :Les opérations de base de données sont abstraites via des interfaces JPA Repository, permettant des opérations d'insertion, suppression et de modification simplifiées sans nécessiter de SQL explicite.

interface Cv24Repository extends JpaRepository<CV24, Long> CV24 findCVById(Long id) ;

2.3 Choix de JPA et MySQL

JPA est choisi pour sa capacité à gérer de manière abstraite et efficace les opérations Object-Relational Mapping , tandis que MySQL est choisi pour son intégration facile avec Spring Boot et clervercloud en tant que add-ons.

3 Fonctionnalités par Route

3.1 Page d'acceuil

HomeController : Affiche la page d'accueil du projet, incluant le nom du projet, la version, les membres de l'équipe, et le logo de l'Université..

3.2 Page d'aide

HelpController : Fournit une page d'aide qui liste les opérations disponibles dans le service REST, avec des détails sur les URL, les méthodes attendues et un résumé des opérations.

3.3 Gestion des CV

3.3.1 resume/xml et /resume

Liste tous les CVs soit en format XML soit en HTML, utilisant JAXB pour générer les représentations requises.

3.3.2 ViewController /cv24/xml ?id= et /html ?id=

Renvoie les détails d'un CV spécifique soit en XML soit en HTML. En cas d'absence du CV, un message d'erreur est généré.

4 Gestion des Erreurs

GlobalExceptionHandler : :Gère les exceptions globalement pour l'application, retournant des messages d'erreur formatés en XML pour les requêtes qui échouent.

5 Validation XML avec Schéma XSD

La validation XML contre un schéma XSD est importante pour assurer l'intégrité et la conformité des données avant leur insertion dans la base de données. Cette validation est gérée par le XmlValidationService, qui utilise la bibliothèque JAXB pour le processus de validation :
Chargement du schemas XSD : À l'initialisation du service, le schéma XSD est chargé à partir des ressources du projet.

```
SchemaFactory factory = SchemaFactory.newInstance(XMLConstants.W3C_XML_SCHEMA_NS_URI);
try (InputStream xsdInputStream = new ClassPathResource("xsd/cv24.xsd").getInputStream())
    this.schema = factory.newSchema(new StreamSource(xsdInputStream));
```

Figure 4 – Chargement

Insertion selon la validation :Si le document est valide, le processus continue vers l'insertion ; sinon, une exception est levée et traitée pour informer l'utilisateur de l'invalidité du document XML fourni.

```
Validator validator = schema.newValidator();
validator.validate(xmlSource);
```

Figure 5 – Validation

Validation du Document XML : La méthode validateXml du XmlValidationService prend en entrée un StreamSource qui encapsule le document XML et utilise un Validator pour vérifier la conformité du document au schéma XSD.

6 Affichage et manipulation des CV

Les CVs sont gérés via des endpoints spécifiques dans le CV24Controller, utilisant des techniques de marshalling pour convertir les données entre formats XML et objets Java.

Marshalling et Unmarshalling avec JAXB : JAXB est utilisé pour convertir les objets Java en XML et vice versa. Cette opération est cruciale pour les requêtes GET et POST qui manipulent des données XML.

```
JAXBContext context = JAXBContext.newInstance(CV24.class);
Unmarshaller unmarshaller = context.createUnmarshaller();
JAXBElement<CV24> jaxbElement = unmarshaller.unmarshal(new StreamSource(xmlStream), CV24.class);
```

Figure 6 – Techniques de conversion XML-JAVA <-> JAVA-XML

Utilisation de Jackson pour la Conversion en HTML : Jackson, configuré pour gérer les formats XML, est utilisé pour la sérialisation et la désérialisation des objets Java en XML. Pour l'affichage en HTML et Spring MVC est utilisé pour générer dynamiquement des vues HTML à partir des données des CVs.

7 Test Postman

Flexibilité : Les tests postman etaient necessaires pour la verification et la la validation de la réponse en cas de succès ou d'erreur, et la persistance correcte des données en base.

7.1 Test de la page d'acceuil

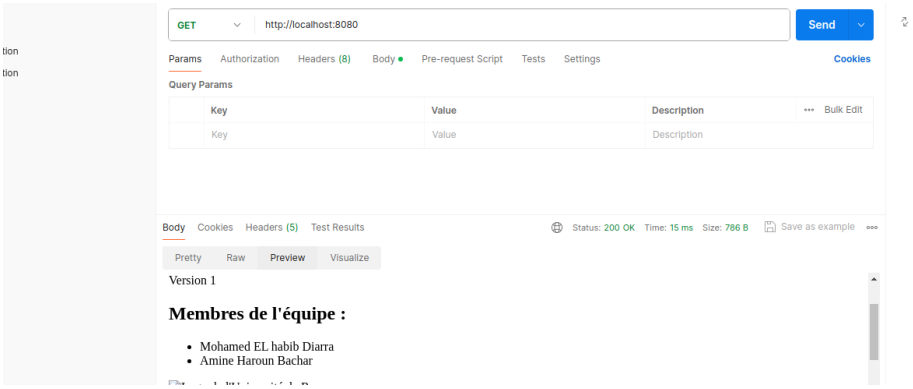


Figure 7 – Patron Command

7.2 Test L'insertion

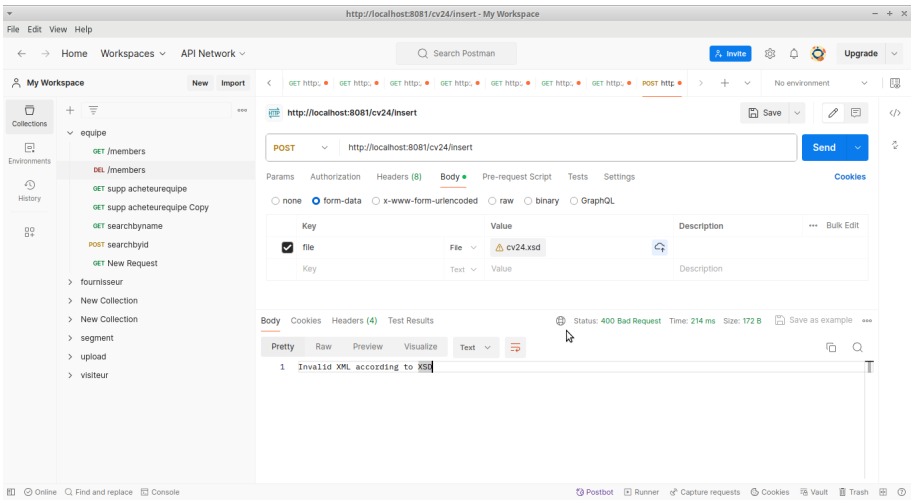


Figure 8 – Tentative d'insertion avec un schema non conforme

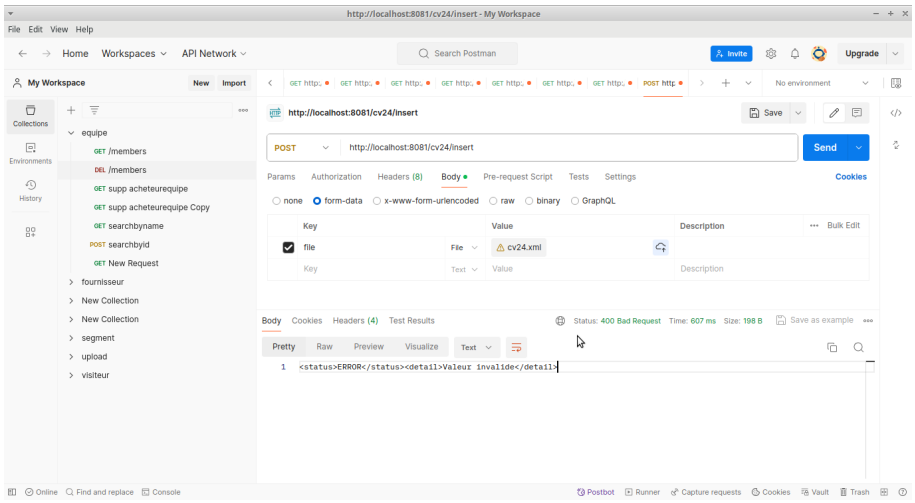


Figure 9 – Tentative d'insertion avec des valeurs invalides

7.3 Test de suppression

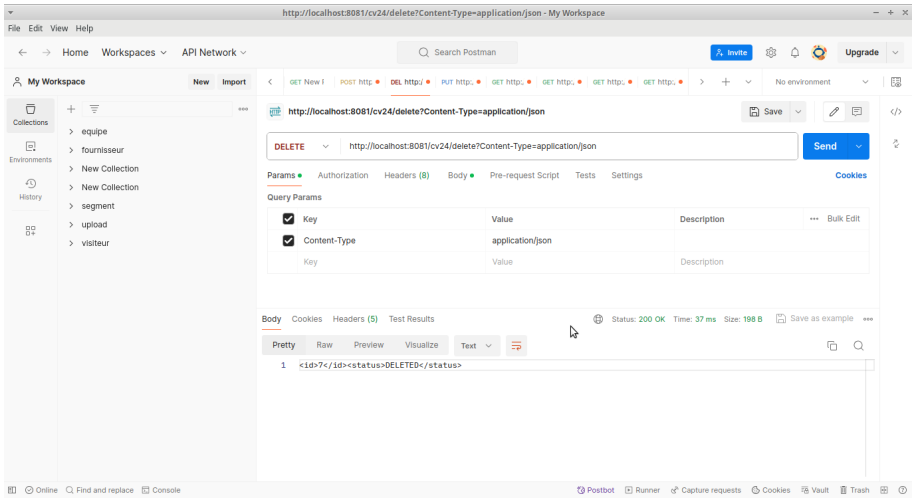


Figure 10 – Suppression de CV

7.4 Test de Mises à jour

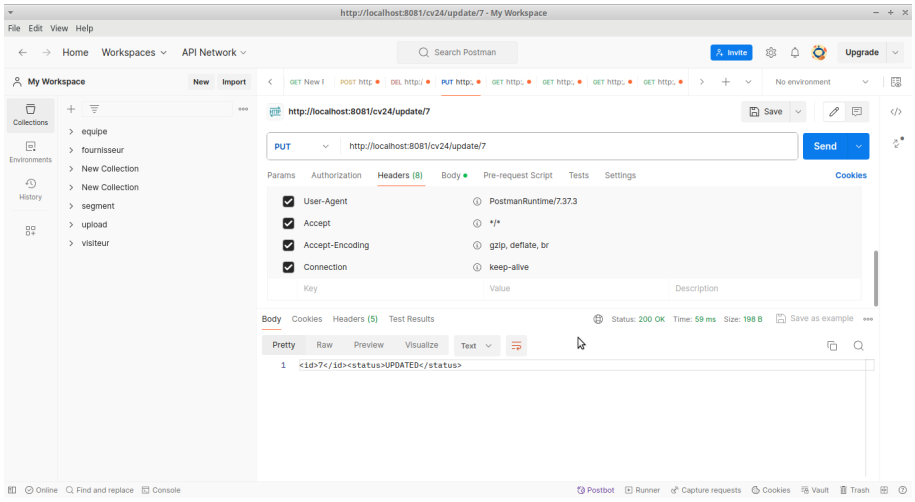


Figure 11 – Mise à jour des information d'un CV

7.5 Test d'insertion sans validation

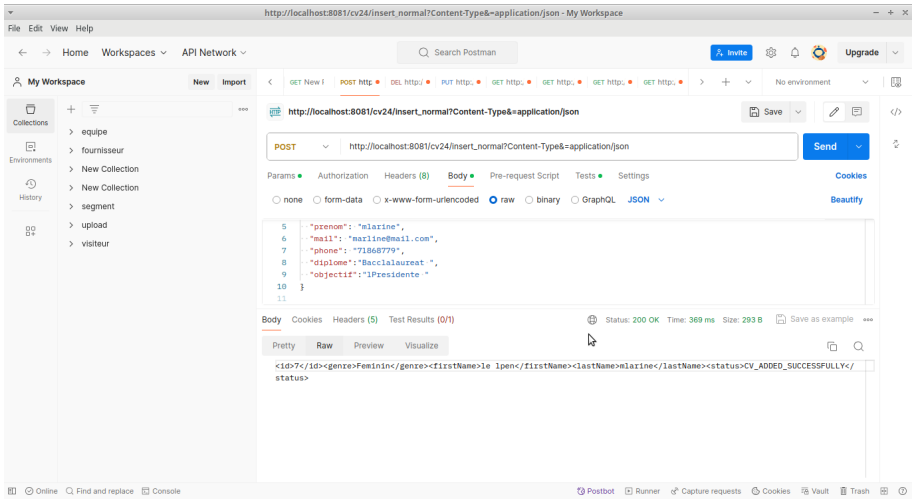


Figure 12 – Insertion d'un schemas sans validation

7.6 Test de recuperation de la page d'aide

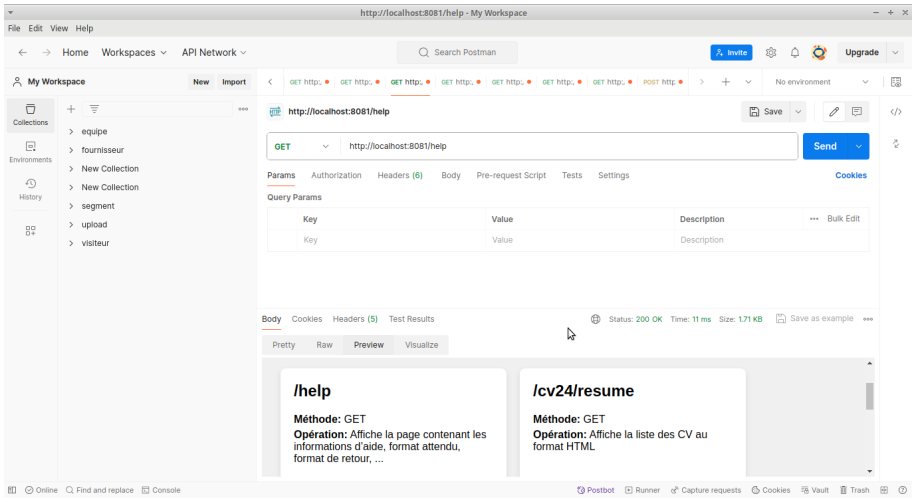


Figure 13 – Recuperation de la page d'aide

7.7 Test de recuperation de l'ensemble des cv au format XML

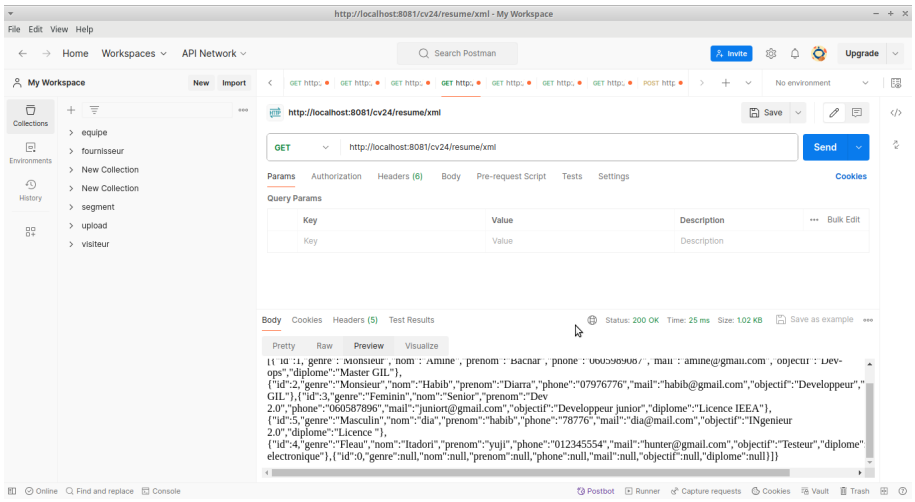


Figure 14 – Recuperation de la page d'aide

7.8 Test de recuperation de l'ensemble des cv au format HTML

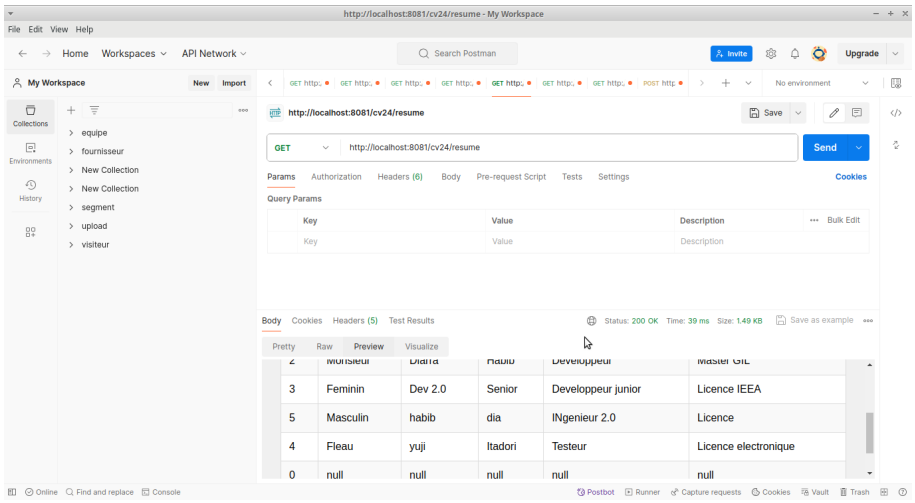


Figure 15 – Recuperation de l'ensemble des cv

7.9 Test de recuperation d'un cv par identifiant

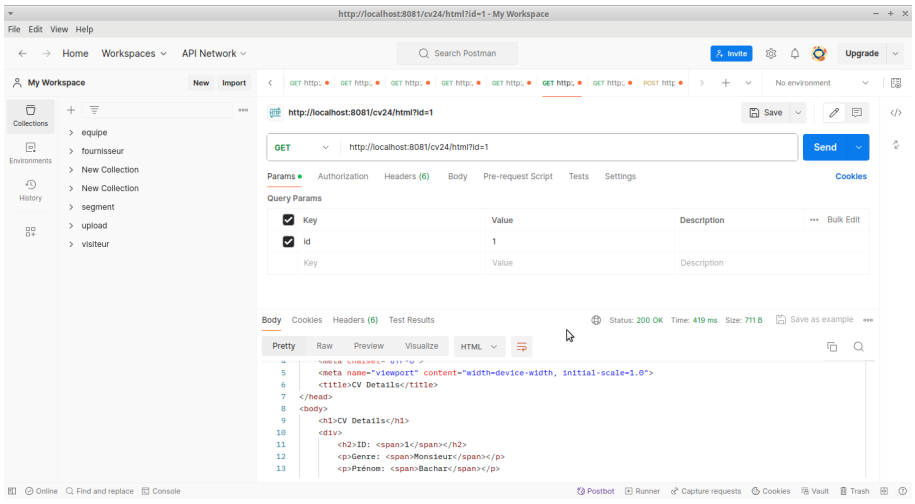


Figure 16 – Recuperation de l'ensemble des cv

7.10 Test de recuperation de donnée html par identifiant

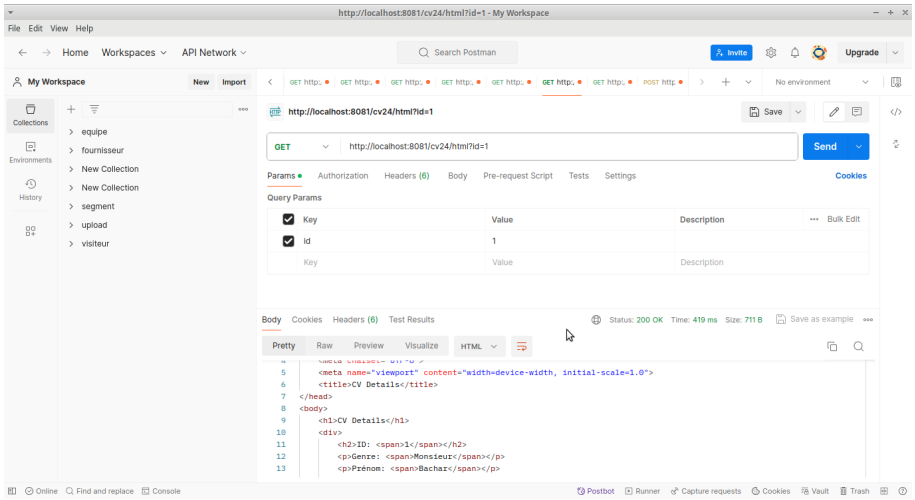


Figure 17 – Recuperation de l'ensemble des cv

7.11 Test de recuperation de donnée XML par identifiant

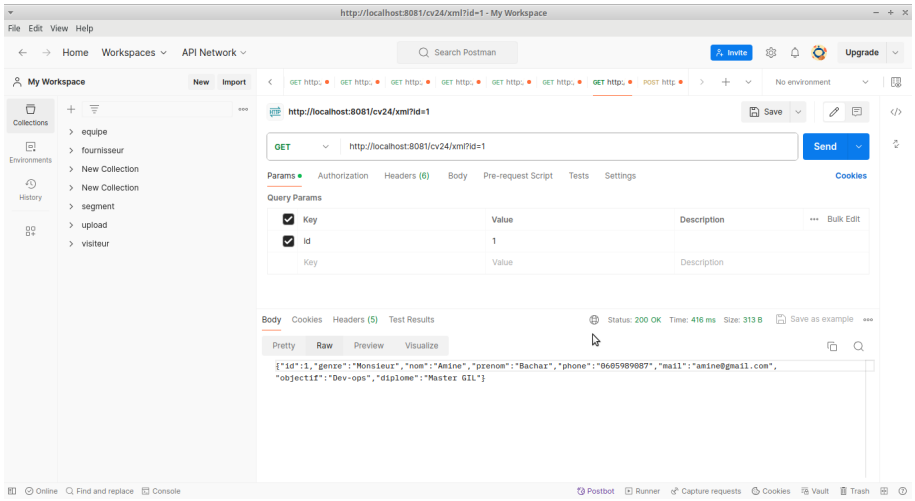


Figure 18 – Recuperation de l'ensemble des cv

8 Déploiement et Maintenance

8.1 Déploiement avec Docker et Contraintes avec Clever Cloud

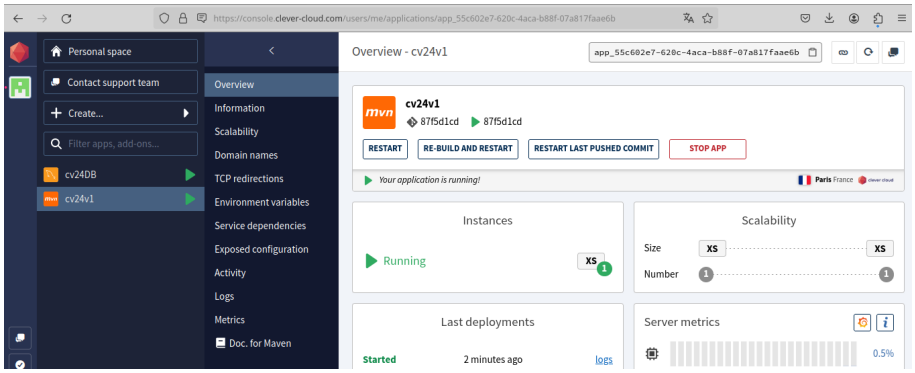


Figure 19 – Deploiément clevercloud

Le déploiement du projet CV24 sous Docker aurait offert de multiples avantages tels que l'amélioration de la portabilité, la facilité de gestion des dépendances,Clever Cloud est une plateforme robuste offrant une gamme de services pour le déploiement automatique et la gestion d'applications cloud, mais l'intégration de conteneurs Docker dans cette plateforme dans le contexte de notre projet présentait des défis spécifiques. Les contraintes de configuration réseau et de gestion du stockage persistant ont rendu l'option de déploiement sur Clever Cloud moins viable pour notre application.

8.2 Deploiement el local

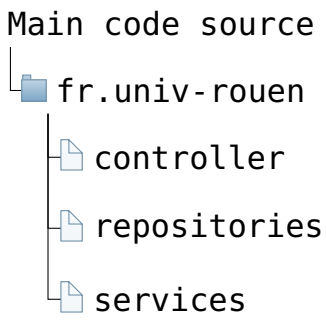
En revanche, le déploiement de l'application en local via Docker a été réalisé avec succès. Ce processus a impliqué la création d'un Dockerfile approprié pour construire l'image de l'application et l'utilisation de Docker Compose pour orchestrer l'application et ses services associés tels que MySQL

```
2  FROM openjdk:21
3
4  # Copier le fichier JAR généré dans l'image
5  COPY target/cv24database.war /cv24database.jar
6
7  ENTRYPOINT ["java", "-jar", "cv24database.jar"]
8
9  # Exposer le port sur lequel l'application sera accessible
10 EXPOSE 8080
```

Figure 20 – Dockerfile pour le deploiement

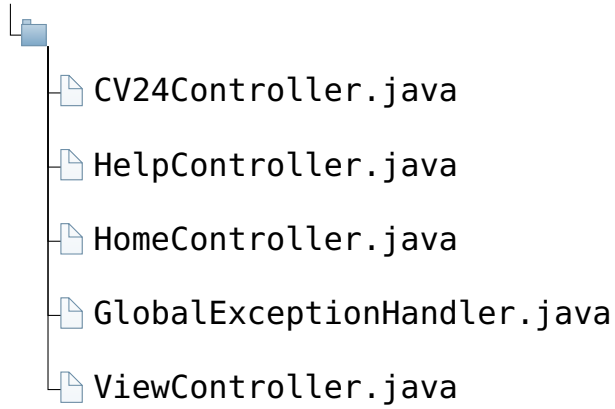
9 Répartition des paquetages

9.1 paquetage default



fr.univ-rouen : Contient les contrôleurs de l'application Spring Boot. Les contrôleurs agissent comme la couche de présentation dans l'architecture MVC (Modèle-Vue-Contrôleur), gérant les requêtes entrantes et renvoyant les réponses au client.

9.2 paquetage controller



Ce paquet contient les contrôleurs de votre application Spring Boot. Les contrôleurs agissent comme la couche de présentation dans l'architecture MVC (Modèle-Vue-Contrôleur), gérant les requêtes entrantes et renvoyant les réponses au client.

Analyzer.java : C'est une interface qui définit une méthode analyze prenant en paramètre le nom d'un fichier..

CV24Controller.java : Gère les opérations pour les CVs, telles que l'ajout, la suppression, la mise à jour, et la récupération des CVs.

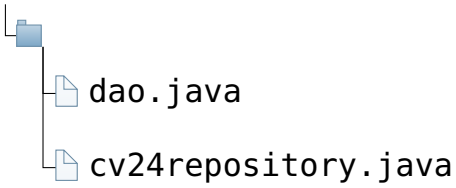
ViewController.java : Fournit des méthodes pour retourner des vues ou des données sous forme de HTML/XML, en fonction des requêtes de l'utilisa-

teur.

HelpControlle.java : Offre une page d'aide qui décrit les différentes routes et fonctionnalités disponibles dans l'application.

GlobalExceptionHandler.java : Cette classe gere les exceptions et retournes les erreurs appropriées.

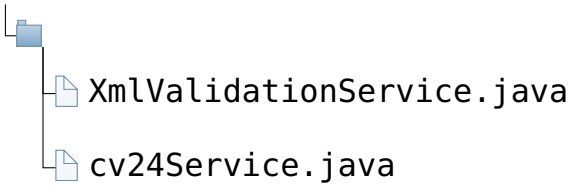
9.3 paquetage repositories



cv2'repository : Interface Spring Data JPA qui définit des méthodes pour l'accès aux données des CVs, telles que la recherche, l'insertion, la mise à jour et la suppression des CVs.

dao : Une entité qui représente un CV, avec des champs pour chaque propriété que le CV doit avoir, comme le nom, la date de naissance, les qualifications, etc.

9.4 paquetage service



XmlValidationService :S'occupe de la validation des documents XML contre un schéma XSD pour s'assurer que les données reçues sont valides.

cv24Service :Fournit des méthodes pour manipuler les données des CVs, en encapsulant la logique spécifique aux CVs.