Documentation Projet PASIFAE PRO

Présentation du projet :

PASIFAE PRO est une application dédiée à la gestion du prélèvement à la source (PAS) des traitements et salaires.

Elle permet de collecter les fond prélevés par les entreprises (Tiers collecteur) et de transmettre les ordres de paiement à PSAR pour prélèvement auprès de la Banque de France.

Logiciels utilisés :

Open VPN :

Eclipse IDE : Développement Java : ajout de fonctionnalités, modifications de classes et implémentation des nouvelles règles de gestion

SQL Developper : Création et exécution de scripts SQL pour gérer les ordres de paiement, les retours arrière, et l'intégration des données dans la base.

FilesZilla : Transfert de fichier entre les plateformes notamment pour l'envoi vers le simulateur SPEP

Objectif du stage :

-Automatiser les processus de traitement et de gestion des fichiers de paiements

-Ajouter de nouvelles balises dans les fichiers XML pour enrichir les données

-Modifier et adapter les classes Java associées aux services et traitements(ServiceIntegrationsspepImpl, DbBatchIntegrationSpepImpl)

-Tester les fichiers dans un environnement de simulation (bouchon SPEP) pour garantir leur conformité

Étapes du projet et contributions :

1) Création du script d'ordre de paiement (de 0 à 9).

Générer des ordres de paiement structurés pour les 10 tables (0 à 9) utilisées pour paralléliser le traitement dans PASIFAE PRO

```
ALTER TABLE ORDRE_PAIEMENT_0
ADD (
    ADRESSE VARCHAR2(90),
    COMPLEMENTADRESSE VARCHAR2(80),
    CODEPOSTAL VARCHAR2(20),
    VILLE VARCHAR2(40),
    PAYS VARCHAR2(40)
);
```

2)Script de retour arrière (SQL)

Mettre en place un script SQL permettant de revenir à un état antérieur des données en cas d'erreur ou d'échec dans le traitement des ordres de paiement.

-Développement d'un script SQL pour restaurer les données dans la base.

-Tests pour s'assurer que le retour arrière s'exécute sans perte de données.

```
ALTER TABLE ORDRE_PAIEMENT_0 DROP COLUMN (ADRESSE, COMPLEMENTADRESSE, CODEPOSTAL, VILLE, PAYS);

ALTER TABLE ORDRE_PAIEMENT_1 DROP COLUMN (ADRESSE, COMPLEMENTADRESSE, CODEPOSTAL, VILLE, PAYS);

ALTER TABLE ORDRE_PAIEMENT_2 DROP COLUMN (ADRESSE, COMPLEMENTADRESSE, CODEPOSTAL, VILLE, PAYS);

ALTER TABLE ORDRE_PAIEMENT_3 DROP COLUMN (ADRESSE, COMPLEMENTADRESSE, CODEPOSTAL, VILLE, PAYS);

ALTER TABLE ORDRE_PAIEMENT_4 DROP COLUMN (ADRESSE, COMPLEMENTADRESSE, CODEPOSTAL, VILLE, PAYS);

ALTER TABLE ORDRE_PAIEMENT_5 DROP COLUMN (ADRESSE, COMPLEMENTADRESSE, CODEPOSTAL, VILLE, PAYS);

ALTER TABLE ORDRE_PAIEMENT_6 DROP COLUMN (ADRESSE, COMPLEMENTADRESSE, CODEPOSTAL, VILLE, PAYS);

ALTER TABLE ORDRE_PAIEMENT_7 DROP COLUMN (ADRESSE, COMPLEMENTADRESSE, CODEPOSTAL, VILLE, PAYS);

ALTER TABLE ORDRE_PAIEMENT_8 DROP COLUMN (ADRESSE, COMPLEMENTADRESSE, CODEPOSTAL, VILLE, PAYS);

ALTER TABLE ORDRE_PAIEMENT_9 DROP COLUMN (ADRESSE, COMPLEMENTADRESSE, CODEPOSTAL, VILLE, PAYS);
```

3) Modification du fichiers XML (XSD)

Enrichissement des fichiers XML d'ordre de paiement avec de nouvelles balises : adresse, complementAdresse, codePostal, ville, pays

```
<xs:complexType name="versementOrdrePaiementPasDto">
   <xs:sequence>
      <xs:element name="idVersementOrdrePaiementPas" type="xs:base64Binary" minOccurs="0"/>
      <xs:element name="bic" type="xs:string" minOccurs="0"/>
      <xs:element name="dateSignatureMandat" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="iban" type="xs:string" minOccurs="0"/>
      <xs:element name="montantPaye" type="xs:decimal" minOccurs="0"/>
      <xs:element name="rum" type="xs:string" minOccurs="0"/>
      <xs:element name="siretPayeur" type="xs:string" minOccurs="0"/>
      <xs:element name="titulaire1" type="xs:string" minOccurs="0"/>
      <xs:element name="titulaire2" type="xs:string" minOccurs="0"/>
      <xs:element name="adresse" type="xs:string" minOccurs="0"/>
      <xs:element name="complementAdresse" type="xs:string" minOccurs="0"/>
      <xs:element name="codePostal" type="xs:string" minOccurs="0"/>
      <xs:element name="ville" type="xs:string" minOccurs="0"/>
      <xs:element name="pays" type="xs:string" minOccurs="0"/>

    </xs:sequence>
</xs:complexType>
```

Modifications dans les classes Java :

-La classe OrdrePaiement permet la gestion des paiements encapsulant des informations cruciales telle que les détails bancaires, les informations sur le bénéficiaire et des données administratives.

-L'objectif principal de cette modification est d'ajouter des attributs relatifs à l'adresse du bénéficiaire afin de garantir une gestion complète sur le bénéficiaire et des données relatifs

-Les attributs suivants ont été ajoutés à la classe OrdrePaiement :

-adresse : Adresse du titulaire

-complementAdresse : Complément adresse du titulaire (étage, appartement)

-codePostal : Code postal de l'adresse du bénéficiaire

-ville : Ville du titulaire

-pays : Pays du titulaire

-L'intégration de ces nouveaux attributs améliore la précision des informations relatives au bénéficiaire, facilitant ainsi :

La validation des paiements

La conformité réglementaires

L'intéropérabilité

```java
/**
 * Class OrdrePaiement
 */
public class OrdrePaiement extends BaseObject implements Serializable
{

 private String adresse;

 private String complementAdresse;

 private String codePostal;

 private String ville;

 private String pays;



 public String getAdresse() {
    return adresse;
}

public void setAdresse(String adresse) {
    this.adresse = adresse;
}
```

-La classe SpepOrdrePaiement est utilisée pour gérer les informations liées aux paimeents. Elle contient des données essentielles comme l'identité du titulaire, les informations bancaires, aini que d'autre détails relatif au paiement.

-Les modifications apportées visent à enrichir la classe avec des informations supplémentaires concernant l'adresse du bénéficiaire. Cela permet de mieux gérer les paiements,surtout lorsqu'il s'agit de transactions internationales ou des vérifier la validité des informations de paiement.

-Les nouveaux attributs ajoutées sont :

-adresse

-complementAdresse

-codePostal

-ville

-pays

-L'ajout de ces informations permet de :

Optimiser l'exécutions des paiements

Assurer la conformité

Gérer les anomalies

```java
private SPEPOrdrePaiement recupererVersement(Element eltVersement) {
    SPEPOrdrePaiement ordrePaiement = new SPEPOrdrePaiement();
```

```
// ALO le 17/01/2025
// Ajout des attributs adresse
    ordrePaiement.setAdresse(eltVersement.getChildText("adresse"));
    ordrePaiement.setComplementAdresse(eltVersement.getChildText("complementAdresse"));
    ordrePaiement.setCodePostal(eltVersement.getChildText("codePostal"));
    ordrePaiement.setVille(eltVersement.getChildText("ville"));
    ordrePaiement.setPays(eltVersement.getChildText("pays"));
```

- La classe ServiceIntegrationSpepmpl est responsable de l'intégration des ordres de paiement dans un système transactionnel. Cette classe interagit avec des fichiers XML contenant des ordres de paiement, les traite et les intègres dans une base de données tout en gérant les erreurs, les rejets et les anomalies.

- Les imports de la classe incluent des bibliothèques nécessaires pour :

    -Manipulation des fichiers XML : stocke

    -Gestion des services et de la transaction :

    -Gestion des exceptions et des erreurs :

-Les attributs principaux :

    DbBatchIntegrationspep : Service de gestion des interactions avec la base de données

    ServiceIntegrationSpepTransactionnel : Service pour gérer les transactions

    Listes des ordres de paiement : liste des ordres de paiement à traiter ou rejeter

    Gestion des erreurs : Suivi des erreurs et paiements rejetés

-Ajout des attributs

```
public void TransformerSpepRejetEnRejetVersement(String numSeqFichier) throws GlobalException {
    RejetOrdreVersement rejet = null;
    for (SPEPEnteteOrdrePaiement uneEntete : this.listeRejetEntete) {
        for (SPEPOrdrePaiement versement : uneEntete.getListeVersements()) {
            rejet = new RejetOrdreVersement();
            rejet.setNumeroSeqFichier(numSeqFichier);
```

```
private SPEPOrdrePaiement controlerElementVersementOrdrePaiementPAS(SPEPOrdrePaiement eltVersementPas, boolean rejetOrdre, String codeAnomalie) throws GlobalException {
    List<String> listeAnomalie = new ArrayList<String>();
    String message = null;
    String anomalie = new String();
    String anoRetour = null;
    String listeCaracteres = constituerListeCaracteres();
```

```
String adresse = eltVersementPas.getAdresse();
adresse = Texte.transformerCaracteresAccentues(adresse);
adresse = Texte.filtrerCaracteresInterdits(adresse, " ");
eltVersementPas.setAdresse(adresse);
/* Controle adresse */
if (((eltVersementPas.getAdresse() == null || eltVersementPas.getAdresse().length() == 0) && montant != null && !montant.estMontantNul()) || (eltVersementPas.getComplementAdresse() != null &&
    anomalie = ("30");
    listeAnomalie.add(anomalie);
    message = chercherCorrespondanceMessageAnomalie(anomalie);
    CPLog.getLogger().bilan(message);
```

-La classe DbBatchIntegrationSpepImpl joue un rôle essentiel dans le processus d'intégration des ordres de paiement dans le système financier en exploitant des fichiers de paiement. Elle permet d traiter, valider et enregistrer les données de paiement en assurant une intégration fluide dans la base de données.

-Elle prend en charge la lecture des fichiers de paiement en provenance de diverses sources. Ces fichiers contiennent des informations détaillées concernant les paiements, telle que :

* -Le nom du fichier

  -Le contenu du fichier

  -Le numéro de séquence

  -Le nombre de paiements

  -La date de création et de enregistrement

  -Le montant total des paiements

-Lors du traitement des ordres de paiement, la classe DbBatchintegrationSpepImpl gère également l'intégration des informations d'adresse liées aux bénéficiaires de paiement. Ces informations sont extraites des données de paiement qui comprennent : Adresse, complementadresse, codePostal, ville, pays.

-Elle assure également la gestion des erreurs qui peuvent survenir dans le cas de l'insertion des ordres de paiement et des informations associés. En cas de violation de contraintes, comme des doublons ou des erreurs de formats, les erreurs sont capturées et les transactions sont annulées pour éviter toute corruption de données.

```
List<OrdrePaiement> listeOrdrePaiement = (List<OrdrePaiement>) (pParametre);

batchValues = new ArrayList<>(listeOrdrePaiement.size());

for (OrdrePaiement ordrePaiement : listeOrdrePaiement)
{

    if (ordrePaiement.getMontant().estMontantNul())
    {

        batchValues.add(
            new MapSqlParameterSource(
                "idTech", ordrePaiement.getIdTechEntete())
```

```
//ALO le 20/01/2025
//Ajout des données adresse
.addValue("adresse", ordrePaiement.getAdresse())
.addValue("complementAdresse",ordrePaiement.getComplementAdresse())
.addValue("codePostal",ordrePaiement.getCodePostal())
.addValue("ville", ordrePaiement.getVille())
.addValue("pays", ordrePaiement.getPays())

.getValues());

 }
 else
 {

     batchValues.add(
         new MapSqlParameterSource(
             "idTech", ordrePaiement.getIdTechEntete())

  //ALO le 20/01/2025
  //Ajout des données adresse
  .addValue("adresse", ordrePaiement.getAdresse())
  .addValue("complementAdresse",ordrePaiement.getComplementAdresse())
  .addValue("codePostal",ordrePaiement.getCodePostal())
  .addValue("ville", ordrePaiement.getVille())
  .addValue("pays", ordrePaiement.getPays())

  .getValues());

              }
          }
          break;

   }

   return batchValues;

//ALO le 20/01/2025
//Ajout des données adresse
pDataSetIn.setValue(":adresse", unOrdre.getAdresse());
pDataSetIn.setValue(":complementadresse", unOrdre.getComplementAdresse());
pDataSetIn.setValue(":codepostal", unOrdre.getCodePostal());
pDataSetIn.setValue(":ville", unOrdre.getVille());
pDataSetIn.setValue(":pays", unOrdre.getPays());


//ALO le 20/01/2025
//Ajout des données adresse
pDataSetIn.setValue(":adresse", unOrdre.getAdresse());
pDataSetIn.setValue(":complementadresse", unOrdre.getComplementAdresse());
pDataSetIn.setValue(":codepostal", unOrdre.getCodePostal());
pDataSetIn.setValue(":ville", unOrdre.getVille());
pDataSetIn.setValue(":pays", unOrdre.getPays());

 }
 else
 {
```

4)Intégration des fichiers dans PASIFAE PRO

Les fichiers reçu de SPEP doivent être intégrés dans les tables de la base de données PASIFAE PRO

-Intégration des fichiers dans les 10 tables via SQL Developer

-Attribution du statut T aux fichiers intégrés pour indiquer qu'ils sont prêts pour les traitements

5) Tests et validation des balises adresses PASIFAE PRO

Le but du test est de vérifier la gestion des erreurs lors de l'intégration des données en testant les balises d'adresse avec un dépassement du nombre de caractère autorisé ou un caractère non autorisé. L'objectif est de s'assurer que le système détecte correctement les anomalies et génère les codes d'anomalie correspondants.

Pour chaque champ d'adresse, nous entrons une valeur dépassant la limite prévue afin de vérifier si le code anomalie correspondant est bien généré.

GRILLE DE TESTS PASIFAE PRO

Test	balise	erreur	résultat attendu
contrôle longueur > 90 caractères	adresse	Anomalie 30	Code 30
contrôle caractère interdit	adresse	Anomalie 30	Code 30
contrôle balise absente ou nulle alors que montant >0	adresse	Anomalie 30	Code 30
contrôle longueur > 90 caractères	complementAdresse	Anomalie 31	Code 31
contrôle caractère interdit	complementAdresse	Anomalie 31	Code 31
contrôle balise absente ou nulle alors que montant >0	complementAdresse	Anomalie 31	Code 31
contrôle longueur > 20 caractères	codePostal	Anomalie 32	Code 32
contrôle caractère interdit	codePostal	Anomalie 32	Code 32
contrôle balise absente ou nulle alors que montant >0	codePostal	Anomalie 32	Code 32
contrôle longueur > 40 caractères	ville	Anomalie 33	Code 33
contrôle caractère interdit	ville	Anomalie 33	Code 33
contrôle balise absente ou nulle alors que montant >0	ville	Anomalie 33	Code 33
contrôle longueur > 40 caractères	pays	Anomalie 34	Code 34
contrôle caractère interdit	pays	Anomalie 34	Code 34
contrôle balise absente ou nulle alors que montant >0	pays	Anomalie 34	Code 34

Le test implique l'interaction avec plusieurs tables de la base de données, notamment :

Table FICHIER

Cette table stocke les fichiers XML intégrés dans le système.

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	NOMFICHIER	VARCHAR2(50 BYTE)	Yes	(null)	1	(null)
2	DATEENREGISTREMENT	DATE	Yes	(null)	2	(null)
3	STATUTFICHIER	VARCHAR2(1 BYTE)	Yes	(null)	3	(null)
4	CONTENUFICHIER	CLOB	Yes	(null)	4	(null)
5	DATECREATIONREMISE	DATE	Yes	(null)	5	(null)
6	NOMBREPAIEMENT	NUMBER(11,0)	Yes	(null)	6	(null)
7	MONTANTTOTALPAIEMENT	NUMBER	Yes	(null)	7	(null)
8	CODEERREUR	NUMBER(2,0)	Yes	(null)	8	(null)
9	NUMEROSEQUENCE	VARCHAR2(32 BYTE)	No	(null)	9	(null)
10	DATETRAITEMENT	DATE	Yes	(null)	10	(null)

Table ORDRE PAIEMENT :

Cette table contient les ordres de paiement extraits des fichiers.

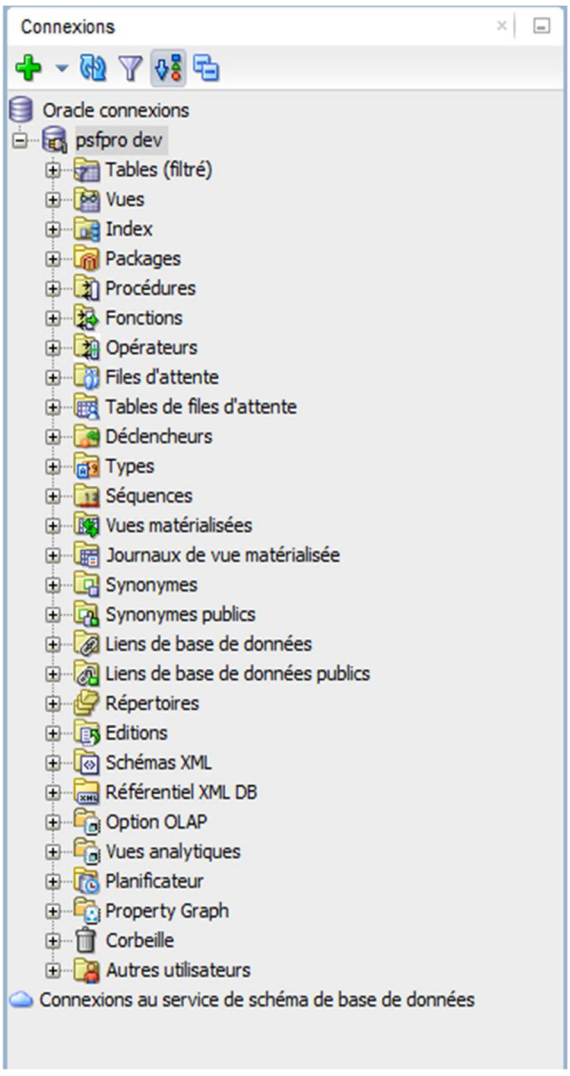
	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	IDTECHENTETE	VARCHAR2(32 BYTE)	Yes	(null)	1	(null)
2	IDENTIFIANTORDREPAIEMENT	VARCHAR2(24 BYTE)	No	(null)	2	(null)
3	IDENTIFIANTDOSSIER	VARCHAR2(24 BYTE)	Yes	(null)	3	(null)
4	MONTANT	NUMBER(11,2)	Yes	(null)	4	(null)
5	BIC	VARCHAR2(11 BYTE)	Yes	(null)	5	(null)
6	IBAN	VARCHAR2(34 BYTE)	Yes	(null)	6	(null)
7	NOMTITULAIRE1	VARCHAR2(15 BYTE)	Yes	(null)	7	(null)
8	NOMTITULAIRE2	VARCHAR2(117 BYTE)	Yes	(null)	8	(null)
9	DATESIGNATUREMANDAT	DATE	Yes	(null)	9	(null)
10	RUM	VARCHAR2(35 BYTE)	Yes	(null)	10	(null)
11	SIRETPAYEUR	VARCHAR2(14 BYTE)	Yes	(null)	11	(null)
12	ENVOYEBDF	VARCHAR2(1 BYTE)	Yes	(null)	12	(null)
13	INDICATEURTRAITEMENTECV	VARCHAR2(1 BYTE)	Yes	(null)	13	(null)
14	INDICATEURTRAITEMENTREFCB	VARCHAR2(1 BYTE)	Yes	(null)	14	(null)
15	NUMEROSEQUENCE	VARCHAR2(32 BYTE)	Yes	(null)	15	(null)
16	ADRESSE	VARCHAR2(90 BYTE)	Yes	(null)	16	(null)
17	COMPLEMENTADRESSE	VARCHAR2(80 BYTE)	Yes	(null)	17	(null)
18	CODEPOSTAL	VARCHAR2(20 BYTE)	Yes	(null)	18	(null)
19	VILLE	VARCHAR2(40 BYTE)	Yes	(null)	19	(null)
20	PAYS	VARCHAR2(40 BYTE)	Yes	(null)	20	(null)

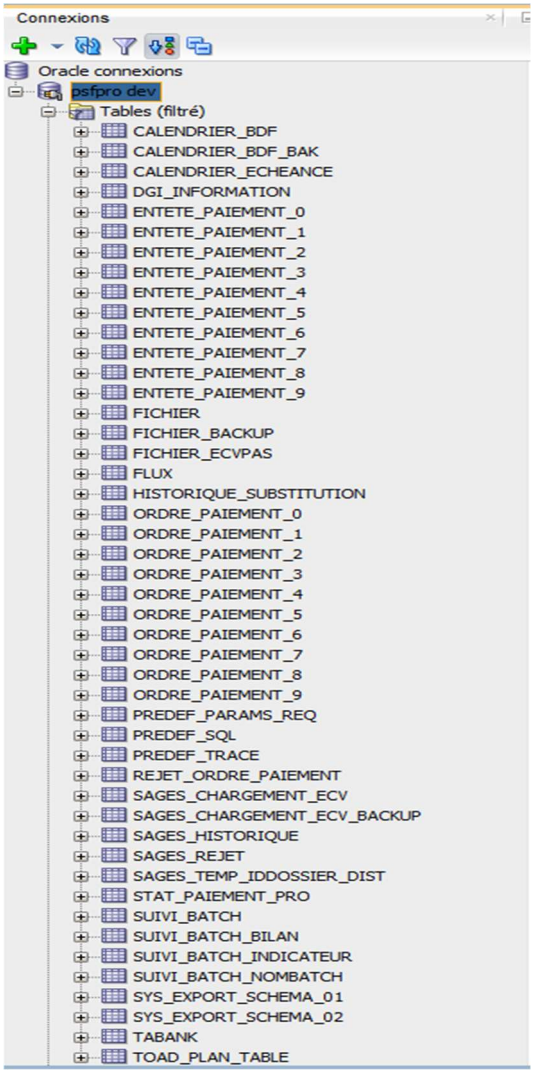
Table REJET_ORDRE_PAIEMENT :

Cette table recense les ordres de paiement rejetés ainsi que le code anomalie correspondant

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	IDENTIFIANTDOSSIER	VARCHAR2(32 BYTE)	Yes	(null)	1	(null)
2	IDENTIFIANTORDREPAIEMENT	VARCHAR2(32 BYTE)	No	(null)	2	(null)
3	NUMEROSEQUENCE	NUMBER	Yes	(null)	3	(null)
4	CODEANOMALIE	VARCHAR2(3 BYTE)	Yes	(null)	4	(null)
5	VERSEMENT	VARCHAR2(500 BYTE)	Yes	(null)	5	(null)
6	INDICATEURTRAITEMENTECV	VARCHAR2(1 BYTE)	Yes	(null)	6	(null)
7	DATEENREGISTREMENT	DATE	Yes	(null)	7	(null)
8	IDTECHENTETE	VARCHAR2(32 BYTE)	Yes	(null)	8	(null)

Base de donnée psfpro:





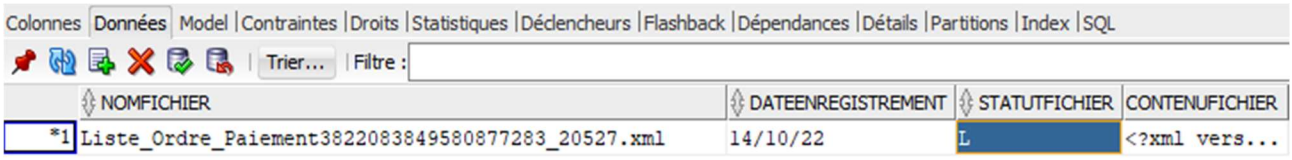
Étape 1 :

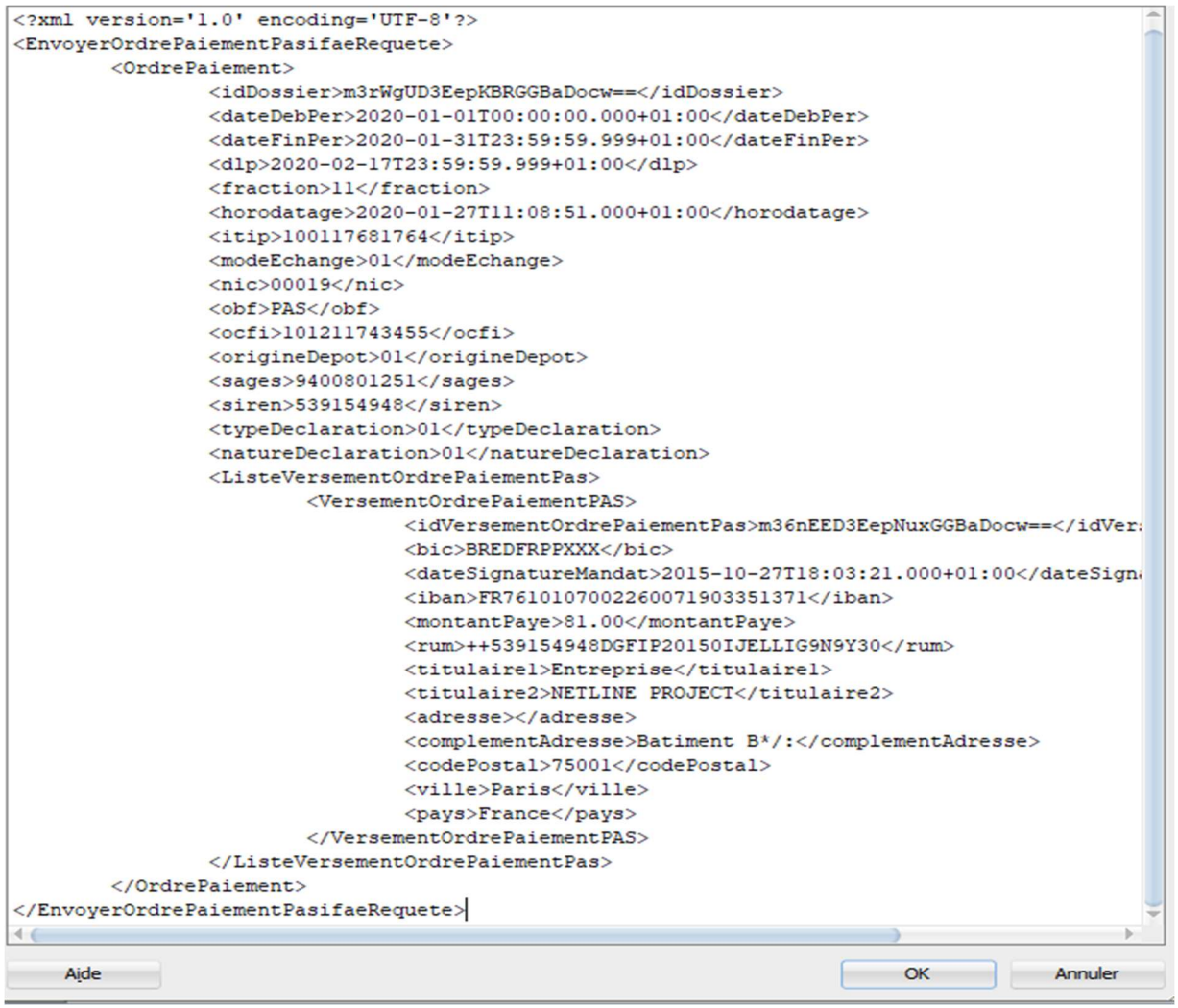
Ouvrir SQL Developer et localiser la table fichier

Modifier l'état du fichier qui est au statut T en le passant au statut L

Mettre à jour les données dans la table (Dépasser nombre de caractère autorisées et caractère non autorisés pour chaque champ d'adresse)

Valider les modifications en faisant un commit



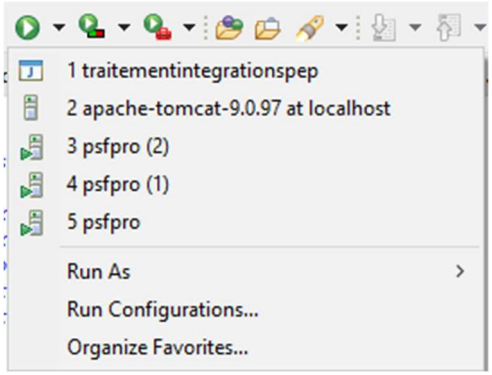


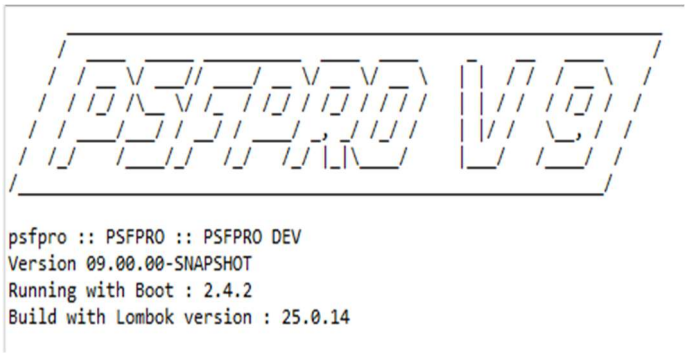
Étape 2 :

Lancer Eclipse et ouvrir le projet PASIFAE PRO

Exécuter Run as →Traitement intégration SPEP

Observer la console pour vérifier si le message d'erreur apparaît « Adresse non conforme »






```
2025-02-20 18:14:38.691 DEBUG M093-1918081 --- [          main] p.b.bilan                        :
Début traitement du fichier : Liste_Ordre_Paiement3822083849580877283_20527.xml nombre de versements: 1 montant des versements: 81,00
2025-02-20 18:14:38.692 DEBUG M093-1918081 --- [          main] p.b.bilan                        :
--------------------------------------------------------------------------------
2025-02-20 18:14:38.693 DEBUG M093-1918081 --- [          main] f.g.f.p.d.i.DbAccesCommunImpl    : requete=select DATEECHEANCE,  DATEBATCH from CALENDRIER_ECHEANCE ORDER BY  DATEECHEANCE
2025-02-20 18:14:39.002 DEBUG M093-1918081 --- [          main] f.g.f.p.d.i.DbAccesCommunImpl    : resultat=[{DATEECHEANCE=2019-03-15 00:00:00.0, DATEBATCH=2019-03-19 00:00:00.0}, {DATEEC
2025-02-20 18:14:39.079 DEBUG M093-1918081 --- [          main] p.b.bilan                        : Adresse du titulaire absente ou non conforme
2025-02-20 18:14:39.081 DEBUG M093-1918081 --- [          main] p.b.bilan                        : Complement d'adresse du titulaire absent ou non conforme
2025-02-20 18:14:39.082 DEBUG M093-1918081 --- [          main] p.b.bilan                        : Code postal du titulaire absent ou non conforme
2025-02-20 18:14:39.083 DEBUG M093-1918081 --- [          main] p.b.bilan                        : Ville du titulaire absente ou non conforme
2025-02-20 18:14:39.084 DEBUG M093-1918081 --- [          main] p.b.bilan                        : Pays du titulaire absent ou non conforme
2025-02-20 18:14:39.104 DEBUG M093-1918081 --- [          main] p.b.bilan                        : Nombre d'Entete en rejet à enregistrer: 1
2025-02-20 18:14:39.107 DEBUG M093-1918081 --- [          main] p.b.bilan                        : Nombre d'erreurs totales : 0
```

Etape 3 :

Consulter la table REJET_ORDRE_PAIEMENT pour confirmer l'apparition des codes anomalies (30 ,31, 32, 33, 34)

REJET_ORDRE_PAIEMENT

Colonnes | Données | Model | Contraintes | Droits | Statistiques | Déclencheurs | Flashback | Dépendances | Détails | Partitions | Index | SQL

Trier... | Filtre :

	IDENTIFIANTDOSSIER	IDENTIFIANTORDREPAIEMENT	NUMEROSEQUENCE	CODEANOMALIE	VERSEMENT
1	m3rWgUD3EepKBRGGBaDocw==	m36nEED3EepNuxGGBaDocw==	20527	30	BREDFRPPXXX;F
2	m3rWgUD3EepKBRGGBaDocw==	m36nEED3EepNuxGGBaDocw==	20527	31	BREDFRPPXXX;F
3	m3rWgUD3EepKBRGGBaDocw==	m36nEED3EepNuxGGBaDocw==	20527	32	BREDFRPPXXX;F
4	m3rWgUD3EepKBRGGBaDocw==	m36nEED3EepNuxGGBaDocw==	20527	34	BREDFRPPXXX;F
5	m3rWgUD3EepKBRGGBaDocw==	m36nEED3EepNuxGGBaDocw==	20527	32	BREDFRPPXXX;F
6	m3rWgUD3EepKBRGGBaDocw==	m36nEED3EepNuxGGBaDocw==	20527	33	BREDFRPPXXX;F
7	m3rWgUD3EepKBRGGBaDocw==	m36nEED3EepNuxGGBaDocw==	20527	33	BREDFRPPXXX;F
8	m3rWgUD3EepKBRGGBaDocw==	m36nEED3EepNuxGGBaDocw==	20527	34	BREDFRPPXXX;F
9	m3rWgUD3EepKBRGGBaDocw==	m36nEED3EepNuxGGBaDocw==	20527	31	BREDFRPPXXX;F
10	m3rWgUD3EepKBRGGBaDocw==	m36nEED3EepNuxGGBaDocw==	20527	30	BREDFRPPXXX;F
11	m3rWgUD3EepKBRGGBaDocw==	m36nEED3EepNuxGGBaDocw==	20527	30	BREDFRPPXXX;F