Description du Serveur

Réalisé par :

- Islam Guettouche
- Lounis Rabhi

1-Introduction

Le but de notre projet est de Créer une API REST pour la gestion des transactions bancaires et de le déployé sur un hébergeur de type Heroku.

2- Adresse du service REST

L'adresse du service REST est : https://servicerestserveurbyguetfinalv.herokuapp.com/

La page d'accueil de serveur est :

3- Description des requêtes

URL	Méthode	Description
/Depot	**POST**	Déposer une nouvelle transaction en l'introduisant sous format d'un flux XML, une erreur est affiché si le flux n'est pas valide
/Resume	**GET**	permet d'obtenir une courte description des transactions stockées en base
/Detail	**GET**	Affichage de la liste des transactions détaillées sous format d'un flux XML
/Stats	**GET**	Affichage des statistiques telles que le nombre de transactions et le montant total
/trx/id	**GET**	Renvoie un flux XML décrivant le détail de la transaction d'identifiant id avec id = idenPaim

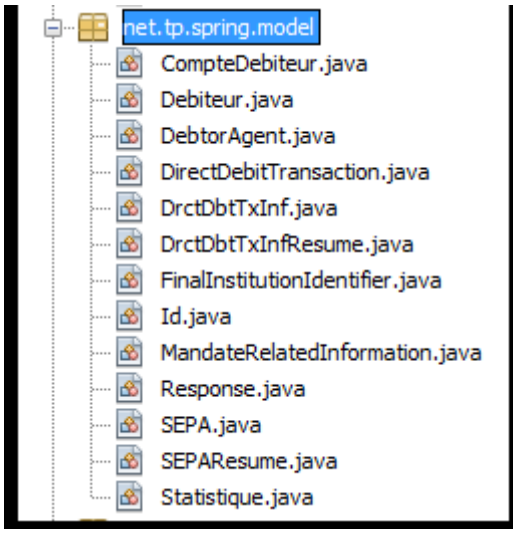
4-Liste des technologies utilisées

1 .Spring MVC Framework:

Nous avons utilisé le Framework spring MVC, les éléments correspondant aux spring MVC sont :

-Le model :

C'est les différentes classes définissant les éléments d'une transaction SEPA qui sont :



-Les vues

On a utilisé la vue accueil.jsp pour l'affichage de l'interface de la page d'accueil de notre serveur qui une interface ergonomique, on a opté pour une interface ergonomique en introduisant du code css a la vue afin de la rendre plus agréable à voir.

-les contrôleurs

On a utilisé deux contrôleurs qui sont :

HomeController.java : il nous permet de visualiser la page d'accueil contenant les différents services ainsi qu'une petite explication de ces derniers.

SepaController.java : il nous permet d'accéder aux différents services d'une transaction SEPA qui sont la recherche, le dépôt, les statistiques, le résumé d'une transaction.

2. JdbcTemplate et MYSQL:

Nous avons utilisé MySQL come Système de Gestion de Base de Données (SGBD) pour sa facilité à mettre en œuvre. Et aussi jdbctemplate pour nous faciliter l'exécution des requêtes SQL et la connexion à notre base de données. Notre base contient une table « » qui est définit comme suit :

```
CREATE TABLE `transaction` (
    `transaction_id` int(11) NOT NULL,
    `num` varchar(6) COLLATE utf8_unicode_ci NOT NULL,
    `PmtId` varchar(35) COLLATE utf8_unicode_ci NOT NULL,
    `InstdAmt` double NOT NULL,
    `MndtId` varchar(35) COLLATE utf8_unicode_ci NOT NULL,
    `DtOfSgntr` varchar(10) COLLATE utf8_unicode_ci NOT NULL,
    `BIC` varchar(11) COLLATE utf8_unicode_ci NOT NULL,
    `Nm` varchar(35) COLLATE utf8_unicode_ci NOT NULL,
    `IBAN` varchar(34) COLLATE utf8_unicode_ci NOT NULL,
    `RmtInf` varchar(50) COLLATE utf8_unicode_ci NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

5-Tutoriel de déploiement

Nous avons utilisé la plateforme heroku pour déployer notre service, nous avons utilisé CLI pour uploader et déployer notre projet automatiquement.

-Nous avons d'abord créé une application en tapant la commande suivante :

heroku create servicerestserveurbyguetfinalv

-Ensuite afin de générer le fichier war il faut se mettre sur le dossier serveur et lancer la commande suivante : mvn war :war

```
C:\Users\Guettouche\Desktop\M1 GIL\S02\Web 2\Projet_Web_Final_Vr\Projet_Web_By_GI\Serveur>mvn war:war
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] Building SepaByGuettouche 1.0
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] --- maven-war-plugin:2.2:war (default-cli) @ Serveur ---
[INFO] Packaging webapp
[INFO] Assembling webapp [Serveur] in [C:\Users\Guettouche\Desktop\M1 GIL\S02\Web 2\Projet_Web_Final_Vr\Projet_Web_By_G
\Serveur\target\SEPA]
[INFO] Processing war project
[INFO] Copying webapp resources [C:\Users\Guettouche\Desktop\M1 GIL\S02\Web 2\Projet_Web_Final_Vr\Projet_Web_By_GI\Serv
ur\src\main\webapp]
[INFO] Webapp assembled in [952 msecs]
[INFO] Building war: C:\Users\Guettouche\Desktop\M1 GIL\S02\Web 2\Projet_Web_Final_Vr\Projet_Web_By_GI\Serveur\target\SE
PA.war
[INFO] WEB-INF\web.xml already added, skipping
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 4.372 s
[INFO] Finished at: 2017-05-01T23:31:51+02:00
[INFO] Final Memory: 9M/113M
[INFO] ------------------------------------------------------------------------
```

Le fichier war est créé et se trouve dans le dossier target.

-Enfin afin de lancer le déploiement on a exécuté en se situant sur le dossier target la commande suivante :

Heroku deploy: war –war SEPA.war –app servicerestserveurbyguetfinalv

```
C:\Users\Guettouche\Desktop\M1 GIL\S02\Web 2\Projet_Web_Final_Vr\Projet_Web_By_GI\Serveur\target>heroku deploy:war --war
 SEPA.war --app servicerestserveurbyguetfinalv
Uploading SEPA.war
-----> Packaging application...
       - app: servicerestserveurbyguetfinalv
       - including: webapp-runner.jar
       - including: SEPA.war
-----> Creating build...
       - file: slug.tgz
       - size: 16MB
-----> Uploading build...
       - success
-----> Deploying...
remote:
remote: -----> heroku-deploy app detected
remote: -----> Installing OpenJDK 1.8... done
remote: -----> Discovering process types
remote:        Procfile declares types -> web
remote:
remote: -----> Compressing...
remote:        Done: 64.5M
remote: -----> Launching...
remote:        Released v4
remote:        https://servicerestserveurbyguetfinalv.herokuapp.com/ deployed to Heroku
remote:
-----> Done
```

-pour accéder au serveur on tape lien ci-dessous :

https://servicerestserveurbyguetfinalv.herokuapp.com/

6- Conclusion

La réalisation de ce projet nous a permet d'apprendre comment réaliser une application avec le Framework Spring MVC et la manière de la déployer sur un hébergeur de type Heroku.