Jax-Y 1.1

Github Project : https://github.com/rac021/Jax-Y

Package Download : https://sourceforge.net/projects/jax-y

Demo :

I) Install demo Database :

```
1 )   chmod +x db-script/db-planes.sh

2 )   ./db-script/db-planes.sh
```

II) Run The executable Jar which deploy endpoint at http://localhost:8080/rest/resources

```
1 )   cd server

2 )   java -jar jax-y-swarm.jar
```

III) Wildfly swarm Server Administration

```
1 ) http://localhost:8080/console/

2 ) Test jax-y deployment
```

IV) Run the GUI Client :

```
1 )   java -jar GUI/jaxy-ui.jar
```

V) Tests (Public serivce) :

```
1 ) invoke the serviceDiscovery
    http://localhost:8080/rest/resources/infoServices

2 ) invoke infoServices with XML/Encrypted - JSON/Encrypted

3 ) invoke the service planes
    http://localhost:8080/rest/resources/planes ( XML / JSON )

4 ) Filter on total_pssengers > 300  :  total_passengers=_>_300

5 ) Keep Only model                  :  model
```

```
      6 ) Keep Only model + distance_km     :   model - distance_km
```

VI) Add new Secured Service (customSignOn authentication) :

```
     1 ) Stop the server

     2 ) Uncomment vip_planes service

     3 ) Uncomment customSignOn authentication in the serviceConf.yaml file

     4 ) restart the server

     5 ) invoke the serviceDiscovery
         http://localhost:8080/rest/resources/infoServices

     6 ) invoke the service vip_planes  :
         http://localhost:8080/rest/resources/vip_planes
         ( XML / JSON / XML-ENCRYPTED / JSON-ENCRYPTED )

     7 ) Test authentication by changing login - password - timeStamp. Test timeOut

     8 ) Test SQL type inference capacity

     9 ) Decrypt data locally

    10 ) Filter on total_passengers=_>_300
                   total_passengers=_>_300&model='Airbus A340-500'
                   total_passengers=_>_300&model=_not_'Airbus A340-500'

    10 ) Keep Only model                    :   model

    11 ) Keep Only model + distance_km    :   model - distance_km

    12 ) Change Tags using "AS" in SQL Queries

    13 ) Test CBC Cipher ( Explain IV )
```

VII) Test SSO authentication with KeyCloak (should works with HTTPS) :

```
     1 ) Start KeyCloak SERVER ( 127.0.0.1:8180 )

     2 ) Stop the jaxy server

     3 ) Comment customSignOn authentication in the serviceConf.yaml file

     4 ) Uncomment SSO authentication in the serviceConf.yaml file

     5 ) restart the server

     6 ) Go to the SSO panel

     7 ) invoke the serviceDiscovery
         http://localhost:8080/rest/resources/infoServices
```

```
    -

8 )  invoke the service vip_planes
     http://localhost:8080/rest/resources/vip_planes
     ( XML - JSON - XML/Encrypted - JSON/Encrypted )

9 )  Test authentication by changing login - password - clientID - secretID

10 )  Filter On                         : total_passengers=_>_300

11 )  Keep Only model                   :  model

12 )  Keep Only model + distance_km     :  model - distance_km

13 )  Add New User in Keycloak ( name + password + login )

14 )  Test acces of the new user to the vip_planes service

15 )  Check logs in KeyCloak server for the user admin
```

VIII) Test Https

```
1 )  For customSignOn authentication

    - Enable HTTPS in serviceConf.yaml

    - Set Self Signed Certificate vs Existing one

    - Restart Server

    - Go to : http://localhost:8443/rest/resources/infoServices

    - Tests

2 )  For SSO

    - Restart Keyloak using https mode ( Default port : 8545 )

    - Uncomment Keycloak_https.json int the serviceConf.yaml

    - Restart Jax-Y server ( Default https port : 8443 )

    - Go to : http://localhost:8443/rest/resources/infoServices

    - Test Connections + Authentication for different users
```

IX) Generate Shell-script for automation

```
1 )  Generate script

2 )  test Script
```

X) Stress test - apache benchmark

```
    1 ) Generate script for stress test

    2 ) Run AB

    3 ) Riddle ??
```

Upcoming Features :

```
    * Swagger-Angular-Client intergration

    * Runtime Algo Choice for Encryption ( AES - DES ... ) ( Done ! )

    * Global Configuration Supports ( Thread pool size, nb of threads by service ,
      data queue size ) ( Done ! )

    * Authentication server using HTTPS  ( Done ! )

    * GUI Https supports  ( Done ! )

    * Generate Jar Client for specific Configuration

    * Add Real Time decryption ( For Stream + Large data )

    * Add support Let's Encrypt
```