**Projet sgbd-D1C1**

**REVERSE ENGINEERING**

**L’url du dépôt sur github :** **https://github.com/khadyLo/Projet-SGBD-DIC1**

**MEMBRES**

* **Fatou Kane LO**
* **Khady** **LO**

**DIC1TR**

**PLAN**

**INTRODUCTION**

1. **VALIDATION**
2. **EXTRACTION**
3. **GENERATION**

**CONCLUSION**

**INTRODUCTION**

Afin de faciliter la structuration des données de type XML et JSON, on a mis en place une solution d’ingénierie générant en sortie un fichier de type SVG.

Pour y arriver, les étapes suivantes sont indispensables :

* **Validation**
* **Extraction**
* **Génération**

Suivant le langage choisi (python dans notre cas), on aura besoin de différentes classes et modules.

1. **VALIDATION**

* **Fichier Json**
* **Modules : json, request**
* **Les fonctions**
* **load\_json\_local\_file(file)**
* **load\_json\_remote\_file(url)**
* **Fichier XML**
* **Modules : xml.etree.ElementTree as ET request**
* **Les fonctions**
* **Load\_xml\_local\_file(file)**
* **Load\_xml\_remote\_file(url)**

1. **EXTRACTION**

* **Fichier Json**
* **Les fonctions**
* **Extraction\_data\_json(data)**
* **Fichier XML**
* **Modules : xml.etree.ElementTree as ET**
* **Les fonctions**
* **Extraction\_data\_xml(root)**

1. **GENERATION**

* **Module : Os**
* **Pour la fonction principale (XJ\_convertor)**
* **Os,**
* **getopt,**
* **sys,**
* **request,**
* **xml.etree.ElementTree as ET.**
* **Les fonctions :**
* **Generate\_mcd**
* **Generate\_svg**
* **Pour la fonction principale (XJ\_convertor)**
* **XJ\_convertor(argv)**

**CONCLUSION**