Meeting 23/04/2024

- Rencontre avec Florent André (CTO MindMatcher)
- Suite RGPD, possiblité de récupérer ces informations d'experience professionel d'un fournisseur de service
    - ex: Linkedin à Malt
- Transformation de données d'un format JSON (donné par un dataprovider) vers du JSON-LD (Linked-Data)
- Présentation de la notion d'ontologie.
- Projet: Faire une API qui permet la transformation de données d'un dataprovider vers une ontologie pivot.
    - Posssibilité d'interroger une base GraphQL (en API) pour la mise en correspondance vers des frameworks pivot (ROME, RNCP, ESCO)
- Démonstration d'un outils de génération de transformation d'onthologie fait en StreamLit

Meeting 26/04/2024

- Rencontre avec Pierre Jacquin et Barthélémy Durette de la société MindMatcher
- Barthélémy travaille sur les ontologies et présentation des connaissances
- Pierre travaille sur l'alogo de trading

Les frameworks

il existe plusieurs frameworks de présentation des compétences (ROME, ESCO, GEN)
chaque framework présente des séries de compétences et skills qui peuvent être mises en relation.

L'ontologie

Définition

Il s'agit d'un ensemble de règles qui permettent de décrire l'information d'une manière structurée selon un modèle et de définir le contenu de la base de données en arrêtant les conditions d'existence et de validité des données.

stockage de L'ontologie et notion de graph

les outils de stockage des ontologies sont souvent des triplestores:

- RDF (Resource Description Framework) sont souvent appelées bases de données RDF ou triplestores. Ces bases de données sont optimisées pour gérer des triplets RDF, qui sont des enregistrements de données sous forme de sujet-prédicat-objet. Voici quelques exemples de triplestores populaires:
    - Apache Jena : C'est un framework open source pour le développement d'applications web sémantiques. Il inclut une base de données RDF et des outils pour traiter les données RDF.

- Virtuoso Universal Server : C'est une plateforme qui combine la gestion de données relationnelles et RDF. Elle permet de stocker des données RDF et de les interroger en utilisant SPARQL.
- Blazegraph : Une base de données graphique qui peut également fonctionner comme un triplestore, supportant le stockage RDF et les requêtes SPARQL.
- Stardog : Une base de données graphique qui supporte les données RDF. Elle permet de faire des requêtes complexes et offre de nombreuses fonctionnalités pour l'entreprise.
- AllegroGraph : Une base de données graphique qui se concentre sur la gestion efficace des triplets RDF et des requêtes SPARQL.

Ici, nous avons une base de données Elacticsearch qui est interrogeable par GraphQL en utilisant basé sur le fichier de description de l'ontologie.

Ontologie MindMatcher

GraphQL et Elasticsearch sont paramètré sur la base de ce model modèle.

```
ns:
  skos: http://www.w3.org/2004/02/skos/core#
  soo: https://competencies.be/soo/
import:
  -
triple:

#######SOO DATAMODEL##########

  soo:Experience:                                 # Describe an experience
whatever type : professional, educational, etc.
    skos:prefLabel: rdf:langstring                # The preferred label of the
experience
    soo:description: rdf:langstring               # A short paragraph describing
the experience
    soo:experienceType: skos:Concept              # The type of experience :
vocational, professional, personal, etc.
    soo:experienceStatus: skos:Concept            # The experience status : past,
ongoing, suggested
    soo:dateFrom: xsd:date                        # the start date if a time
period or the date of occurence
    soo:dateTo: xsd:date                          # the end date if a time period


  soo:Skill:                                      # Describe the skills attached
to an experience
    soo:experience: soo:Experience                # The experience that provided
or is likely to provide the skill
    soo:skillFamily: skos:Concept                 # The skill group as defined in
skos collections, e.g. hard skills/soft skills
    soo:skillLevel: soo:SkillLevel                # The skill level as defined as
a value on a scale
```

```
  soo:Polarity:                                # Polarity express the feeling
toward an experience or a skill
    soo:experience: soo:Experience             # The experience which
individual polarity is given
    soo:polarityScale: skos:OrderedCollection
    soo:polarityValue: skos:Concept            # The polarity defined as a
value on a scale
```

Principe de transformation ontologique et lexical

une première étape consiste à transformer la structure du DataProvider vers l'ontologie pivot sans changer les valeurs.
Dans cette étape, le DataProvider doit fournir la transformation (grace à l'interface de génération de règles faite en streamlit).
Des appels GraphQL permettront la transformation des données vers un framework cible (passé en paramètre) telle que l'a demandé le DataConsumer.
Dans le cas ou les appels graphQL ne permettrait pas de passer par du Training Enhancement.

à préciser La notion de DataSpace (Ariane?) a été évoqué.

Training Enhancement

C'est la partie que développe Pierre. Il est envisagé de faire un vecteur de 1024 bit/characters pour représenter le texte. Ce vecteur peut ensuite être envoyé directement à graphql pour faire une recherhe de distance entre mot (par projection, ou methode "cosinus").

Par ailleurs, il est aussi possible de récuperer les choix du dataprovider en terme de mapping.

Meeting 29/04/2024

Noua avons accès au répository gitlab MindMatcher.
Notre

OntoBridgeAPI Process

```
@startuml
title OntoBridgeAPI Process

actor DataConsumer
actor DataProvider
entity DataSpaceConnector
entity OntoBridgeAPI
entity InternalEngine
entity MachineLearning
entity GraphQL
database ElasticSearch
```

```
DataProvider --> DataSpaceConnector: JSON+Framework Name
DataSpaceConnector->OntoBridgeAPI:JSON+Framework Name
OntoBridgeAPI-->InternalEngine:JSON
InternalEngine-->ElasticSearch:DataProvider Document
ElasticSearch-->InternalEngine:Mapping Rules
InternalEngine-->InternalEngine:Generate
InternalEngine-->OntoBridgeAPI:JSON-LD
OntoBridgeAPI-->GraphQL:JSON-LD+FrameworkName
GraphQL-->ElasticSearch:Query
ElasticSearch-->GraphQL:Result
GraphQL-->OntoBridgeAPI:JSON-LD with Matched Terms
activate OntoBridgeAPI #blue
OntoBridgeAPI-->OntoBridgeAPI:**Check if present in Cache**
OntoBridgeAPI-->MachineLearning:Unmatched Terms
MachineLearning-->OntoBridgeAPI:List of Unmatched Vectors
OntoBridgeAPI-->GraphQL:flush cache
deactivate OntoBridgeAPI

OntoBridgeAPI-->GraphQL:Unmatched Vectors
GraphQL-->OntoBridgeAPI:Nearest Match terms
OntoBridgeAPI-->OntoBridgeAPI:consolidation Match+MachineLearning
DataSpaceConnector --> DataConsumer:JSON+Framework Name
@enduml
```