

# Implementing Disability Knowledge Graph and Advocating Platform

Submitted by

Dharmalingam Dhayananth

Supervised by

Pierre Maret

Dennis Diefenbach

Christo El Morr

Academic Supervisor

Pierre Maret



Université Jean Monnet



York University



The QA Company

QA Company

# Table of Contents

1.	Introduction to the project .....	iv
2.	The objective of the internship .....	v
3.	Presentation of the context.....	vi
3.1	Presentation of the problem .....	vi
3.2	Related Work .....	vii
4.	Selected solution and implementation .....	xi
4.1	Wikibase .....	xi
4.2	QAnswer .....	xii
4.3	Knowledge Graph .....	xii
5.	Development Stack and Tools .....	xiv
6.	Disability data modeling .....	xvi
7.	Implementation.....	xvii
7.1	Setting up Wikibase infrastructure .....	xvii
7.2	Document upload and analyzing.....	xx
7.3	Upload edits to Wikibase .....	xxi
7.4	Integration with Qanswer .....	xxii
7.5	Search function .....	xxii
7.6	Continuous glossary management.....	xxiii
7.7	Global overview.....	xxiv
8.	Supportive implementation.....	xxv
9.	Status of in-use implementation .....	xxvi
10.	Conclusion.....	xxvi
11.	Future work .....	xxvii
12.	Appendix.....	xxviii
12.1	Search results (Question answers).....	xxviii
12.2	Ontology model for document upload .....	xxx
12.3	Reconciliation configuration file (Important properties).....	xxxi
12.4	OpenRefine manifest file.....	xxxii
	References .....	xxxiii

## Table of Figures

Figure 1: Representation of the statement and qualifier in the Wikibase data model .....	xi
Figure 2: RDF Graph consist of single triplet with subject, predicate and object .....	xiii
Figure 3: : Ontology diagram of Disability wiki project.....	xvi
Figure 4: Architecture and different components of the Wikibase platform.....	xvii
Figure 5: Convention on the Rights of Persons with Disabilities wiki page from disability wikibase.....	xviii
Figure 6: CRPD Article 11 wiki page from disability wikibase .....	xix
Figure 7: CRPD Article 11 properties from disability wikibase .....	xix
Figure 8: Architecture diagram for document upload process flow.....	xx
Figure 9: Sequence diagram for document upload and classification .....	xxi
Figure 10: Search function flow of disability wiki website.....	xxii
Figure 11:Sequence diagram for retrain process of algorithm with new glossary terms .....	xxiii
Figure 12: Platform architecture of Disability wiki project.....	xxiv
Figure 13: Test 1: Question answer results from disability wiki website .....	xxviii
Figure 14: Test 2: Question answer results from disability wiki website .....	xxviii
Figure 15: Test 3: Question answer results from disability wiki website .....	xxix
Figure 16: Test 4: Question answer results from disability wiki website .....	xxix

## Abstract

**Disability acts and rights** prevent disability discrimination and help to promote equal rights, equal opportunity, and equal access for people with disability conditions. Most of the countries have already implemented different acts and laws to promote disability rights and equal opportunity. Also, many organizations and governments work together to implement such laws and implications to support people with disabilities. (CDC, 2020)

Many convention documents are presenting different pieces of information to support disability communities and people with disabilities, such as ***Convention on the rights of person with disability (CRPD)***, ***Disability rights promotion international (DRPI)***, and so on. Some documents might contain more than 100 pages and are filled with much conventional information. It is very difficult to search for a specific piece of information in such documents. Also, some information only relevant to some regions and most of the documents are specific to the country. People might find it difficult to search and differentiate information from these documents.

In this context, the goal of my internship is to develop a platform (one or more connected systems) to promote disability rights, conventions, and legislation that provide easy access to information from such conventional documents. Initially, the design of an ontology schema has been discussed and developed based on the requirements. This ontology schema helps to map different entities and their relations using a graph-based concept. The knowledge graph has been developed based on the ontology schema diagram.

Followingly, Wikibase infrastructure has been configured with custom settings and hosted to the internet to publish the linked open data (Knowledge Graph). The MediaWiki software tool has been configured by following all the best practices in terms of security, scalability, and access privilege.

In continuation of this, the wikibase bot program has been developed, that uses wikibase API endpoints and data store endpoint to store and retrieve data. This program was developed to import data from Excel files into the data store.

As the third mission of this internship, A web application has been developed with a search feature for the public where people can access disability-related information with ***Natural Language*** questions. This feature has been achieved by integrating disability knowledge graphs into the QAnswer platform.

Lastly, we developed a document management application that allows users to upload documents, analyze the document and upload information directly to Wikibase. This application uses a classification algorithm that has been integrated from another internship project.

# 1. Introduction to the project

This report presents the work done during the internship project with York University, Canada, QA Company, France, and University Jean Monnet, France. The goal of this project is to develop a disability knowledge graph for the promotion of disability information. I was involved in the development, deployment, and infrastructure management for the software applications of the project. The main tasks are as follow,

1. Configure and run a data store to store, update, and retrieve information.
2. Develop a program to store and retrieve information from the implemented data store.
3. Develop a smart platform for search and document analysis.
4. Enable the question-answering feature to the system where users can find required information by asking human-like questions.

**Disability** is any condition of physical or mental impairment that makes it the person difficult to do certain activities (activity limitation) and interact with the society around them (participation restrictions). People with a disability is a diverse group with a wide range of special needs. Many people face significant barriers in work, study, sport, and doing everyday activities. Disability rights and public conventions explain activities, conventions, and legal requirements to secure equal opportunities and equal rights for all people with disabilities (CDC, 2020).

Many people do not aware of these conventional rights and legal documents. So, it is necessary to provide a simplified platform to access information easily and rapidly. The project aims to provide a platform that digitalizes the disability rights documents and enables easy access to the information with question-answering features to help people with needs. The project introducing a platform that will be enabled with all necessary modules to automate the information search on documents. The smart platform aims to increase the efficiency of information search with *natural language* questions.

Users can upload documents to the smart platform and it will be analyzed and tagged with the provided glossary of terms<sup>1</sup> with the help of *Machine Learning (ML)* and *Natural Language Processing (NLP)*. Additionally, it is integrated with many other modules such as user management, authorization, and authentication management, glossary management, document management, train model and evaluation, and so on.

---

<sup>1</sup> Glossary terms are domain specific vocabulary prepared by domain experts

## 2. The objective of the internship

- Process and prepare structured data from free-text documents
- Logically represent relations between information and objects. Some documents are related to a specific region. The document contains paragraphs and paragraphs talks about different disability rights-related topics. So, this logical connection between data should be represented for a better search of the information.
- Easy and advanced search. Disability community people might not computer scientists and sometimes they might not have experience with computers. Thus, the search feature should be simple enough and should be able to search with natural language.
- Enabling ability to ask question-oriented search instead of keyword search.
- Interlinking data from different knowledge sources is one of the future objectives of the project.
- Build a scalable project with the aim of easy integration.

In addition to this, one more objective is to integrate the deliverable of the co-intern who is working on the document analyzing and classification problem. The solution needs to be integrated and the outcome of these solutions should be used interchangeably within the system.

### 3. Presentation of the context

In addition to classical web technologies, semantic web technology provides additional capability to implement “web of things” and “web of linked data”, that enables machines and human to utilize the web. The ultimate goal is to enable machines to do more powerful tasks, integrate data from a different source, and develop systems that can support interactions over the network. This vision can be achieved by integrating different technologies of linked data such as RDF, SPARQL, OWL, and SKOS. (W3C Org, 2014)

York University, University Jean Monnet, and The QA Company together started a project to implement a disability question answering platform using semantic web technologies. York University and University Jean Monnet are research universities that are involved in researches in many contexts to solve various social and scientific problems. The QA Company is a young digital solution company located in France (QACompany, 2021). The company already dealing with similar problems and offers many different tools to solve such problems using artificial intelligence and machine learning.

#### 3.1 Presentation of the problem

Different documents promote disability conventional information in different forms. These documents contain the most valuable information that is required to be processed with high concern. Documents consist of free-text information and that is not structured data. Search for one piece of information with constraints is not possible with these documents. Sometimes the documents might contain many pages and relevant to some states of a country or country itself. Due to these reasons, it is less convenient for people with disabilities and disability communities to access the information.

The automatic document processing is tedious because the documents do not contain any universal standard or document format. The information is in the form of free text, hence the digitalization for the documents required some more technical tasks. A successful digitalization process requires proper paperwork, management, and record-keeping. The communities and organizations might spend a massive amount of time analyzing and access the information on a conventional disability information document. Also, such information is hard to represent in a form of structured data where a machine can understand. An ideal solution should capable to solve the above-mentioned issues and provide easy access to the information.

## 3.2 Related Work

### Researches

Documents contain free text and information. This information is not understandable by systems since it is not structured data. Different researches have shown different techniques to classify and extract this information into structured data. According to (Weikang Li, Wei Li, Yunfang Wu, 2018), research has been done to propose a method for question answering over free-text documents based on a unified neural network model. The proposed solutions contain three main considerations. Firstly, the model will get a general understanding of a document by reading the title and summary of the document. Secondly, the recurrent neural network method is applied to link the hidden meaning of the document summary and the respective question. Thus, the question will become closer to the document. Then, the author has employed a hierarchical recurrent neural network structure to obtain the document-level representation with the new question encoded vector. Finally, the model decides which is more appropriate sentence could answer the question. The author has proposed different techniques to get the general information about the document using Latent Dirichlet allocation (LDA) and Latent semantic analysis LSA. Although the proposed solution shows effective results with WikiQA English and Chinese data set, it is capable to retrieve only a sentence or word vector as an answer along with the encoded question phrase. This is not suitable for the Disability wiki project, since it is expected to get the complete paragraph of the document that talks about the searched glossary term<sup>2</sup> (topic). Also, the proposed method is capable to represent the logical relation between documents, countries, organizations, and so on.

According to (Marco Anteghini, Jennifer D'Souza, 2020), the authors have presented a novel method for creating a knowledge graph from scientific medical assays. The proposed approach is a hybrid method where the system uses a neural network for identifying critical information in the given document and prepares triples from the extracted information. Then the expert action is needed to clarify the generated triples. The algorithm extract information such as research problems, temperature value, different roles (detergent, assay provider), title, assay type, and so on (ORKG, 2021). A higher concern is given to generate research problems. The extracted information smantified and uploaded to the triple store. The algorithm also uses different metadata to prepare triples that are given by the contributor who uploading the document. The approach requires expert input to manage generated triplets. An expert will select correctly predicted triples, define a new triple, and deletes the incorrect one. This step can be omitted in the future with a well-trained model. This is very much similar to the disability wiki project and proposes a similar process flow. Although the method is very potential and promising, it requires a computer scientist and a medical expert to validate the generated triplet, because both expertise is required for validation. The proposed approach is only for creating and storing semantic data and no method has been proposed or considered for the information retrieval.

---

<sup>2</sup> List of social model terms that used to tag the paragraph based on its semantic meaning



## **Related platform and tools**

As mentioned in (Aitenders, 2021), Aitenders provides a platform that automates the tender answering process to help contractors. The organization introducing a platform that enabled all necessary modules to automate the workload for processing a tender document. The platform aims to increase the efficiency of the tender management and it deals with a project that contains dozens of documents, themselves contains hundreds of pages, to present their content to the clients with a smart platform.

The platform extracts different information from the tender document with the help of *Machine Learning (ML)* and *Natural Language Processing (NLP)*. The information will be classified with the help of a neural network model and stored in the database. The information of the tender document is classified into main categories such as Information, Functional requirement, Constraint, and so on. Followingly, these main classes are further classified into subclasses. This classified information can be retrieved by relevant users of the organization used to prepare a tender response. Even though the platform solves a similar problem, this solution will not support the disability wiki project. Because the platform highly target tender document and it is highly domain-specific. The platform does not support the continuous improvement on classification labels (continuously increase the glossary terms). Additionally, the extracted information of the document will be stored in a tabular structure database. It is difficult to represent the logical relation between data in tabular data stores.

According to (Adlib org, 2021), Adlib provides a software tool for innovative document classification service (SaaS). The organization uses Artificial intelligence, Machine learning, and Computer vision for identifying, classifying, categorizing, and separating different document types. The platform mainly supports business contracts, customer claims, business orders, and other commercial documents. Also, the users can train the algorithm with a domain-specific feature set for a more customized experience. Users can upload text in various formats, such as email, pdf, document, image, and so on. The platform also maintains a link between document corpus and provides a nice user interface to view the data in customized and prioritized visuals based on user settings. The platform is enabled with automated language detection and supports many other languages than English. There are many other features like confidential data management, security over documents, risk mitigation, and so on. This advanced document clustering platform helps organizations to transform a large volume of unstructured data into intelligent data for decision making.

The platform is very much targeting commercial documents and provides an intelligent business solution. This platform helps to structurize the document data and cluster it. The solution does not provide any components to represent data with its logical relation. Also, the platform does not retrieve information with a question-answering feature that is one of the requirements of a disability project. The platform performs analysis based on the document metadata and not based on the contents of the document. But, Paragraph level analysis is required for the disability wiki project. The platform can solve a few problems of the disability wiki projects, but not all the requirements of the project are satisfied with the Adlib platform.

As stated in (Exrtact corp, 2020), Extract is another intelligent document classification platform in the market. This platform can recognize and categorize many types of documents by analyzing their content and automatically extracts relevant index data per document. The document will extract and store information in the database based on the analyzed category. The index data extraction allows the platform to more intelligently, and consistently route documents to desired levels within a database or any other data management system. The document process is fully automated within the platform where it starts from uploading documents and ends when it is stored in the right place accurately and securely. The benefit of the platform is that allows workflows to be automated by intelligently indexing data. The platform is specialized in electronic medical record (EMR), land records system, course system, enterprise resource planning documents (ERP), and more.

The platform supports different formats of documents and structures. The document type and structure are automatically analyzed with machine learning and required data is located by the system. It is not required to point out this information to the system. This feature is one of the key features of the platform. Security is a high concern in the platform since it deals with confidential data such as medical records, and financial documents of the organizations.

This platform solves a similar problem but does not correctly match with the disability project. The disability project not only expects document indexing, but also information analysis of the document content, semantic relation between information, and question answering over the data. All these requirements are unsatisfied with the Extract platform.

As described in (QAnswer, 2021), QA Company offers one of the leading question answering platforms in the market that works by connecting different intelligent components. The question answering over dataset is a difficult task and it needs domain-specific adoption for better performance. QA Company offers a product named QAnswer that is designed with the concern of such cases. Hence, anyone can upload their raw structured or free-text data and can start asking questions without much configuration and adoption. The platform is designed to learn from feedback continuously. This platform supports different data formats, such as knowledge graph, excel files, free text data, and even modern website (HTML).

This advanced question answering system offers an efficient service with accurate results. It is possible to retrieve the required set of data from a given data set when a pattern of questions is asked. Also, users can advance the efficiency by configuring the uploaded data sets by introducing mappings, Mapping is a feature that allows mapping of a specific keyword to a data field in the dataset. So, the system will retrieve the answer with high accuracy. This system learns and acts much faster than the other platforms in the field. QAnswer can help to solve the data retrieval part of the disability wiki project with its advanced question-answering search feature.

Wikibase is one of the best tools that offer connected infrastructure to implement and maintain a large-scale knowledge graph. MediaWiki foundation offers this Wikibase tool which is an open-source software application that is free to use. Popular open-source data content platforms like Wikidata and Wikipedia have also been developed using Wikibase software. The Wikibase is enabled with multi-lingual capability to support many languages. Wikibase helps to implement knowledge graphs with RDF standards. Any domain can be easily represented with the Wikibase data model since it is represented using the logical relationship between data. Thus, the documents and the information of the documents can be represented with a knowledge graph by interlinking with disability social model topics (Ex: Discrimination, and Prevention of life etc). Also, the graph can be extended continuously and Wikibase is highly scalable to a large amount of data. Wikibase offers SPARQL endpoint to query data using SPARQL query language. This allows retrieving information with reasoning and filtering. This tool helps to implement linked data for the disability wiki project (MediaWiki org, 2020).

### **A guiding platform**

The enslaved organization has developed a platform with open-linked data similar to the disability wiki project. The platform allows exploring the data and life stories about the lives of the enslaved according to the *Journal of Slavery* and *Data preservation*. The organization building robust data to explore and discover around 600,000 people records and contains 5 million data points. The data is enriched from different data sources like spreadsheets, research articles, and other surveys. It is possible to search for people with community or group names and read about the life story of that group. The project is growing with the help of scholars and other interested communities. This project uses open-linked data with the help of Wikibase (Enslaved org, 2021). Although the project is very much similar, the disability wiki project is integrated with many other features like question answering, document upload, and analysis. Enslaved search is directly applied to Wikibase API that performs a flat search on entity labels. Due to the many similarities with our needs this platform has been chosen as a guideline product for the disability project.

However we noticed that the enslaved project doesn't have any automatic data upload workflow, it always requires computer experts to feed information directly to the data store, and the search is keyword based.

## 4. Selected solution and implementation

### 4.1 Wikibase

Wikibase has been selected for the disability project since it provides a connected infrastructure to implement an RDF knowledge graph. Knowledge graph helps to logically represent the relation between the data, which is one of the requirements of the disability project. Following, we discuss the Wikibase data model.

The objects are represented as an **item** in the wikibase model. An example would be Document is represented as the item. Relationships between items are described using **properties**. Items and the properties are **entities** in the Wikibase model and they are identified with a URI<sup>3</sup>. Items URI's contains a prefix 'Q' and property URI's contains a prefix 'P'. Additionally, an Entity can hold the following properties,

- Label: Name of the entity. Label can be defined differently for each language.
- Description: Description provides the detailed meaning of the entity and defined per language.
- Aliases: alternative names for the entity and defined per language.
- Statements: Statement also used to enrich the information about the entity. Statements contain property and value. Property shows the relationship between the entity and the value. Value can be a text or another entity. (Special cases can include no-value or unknown). Each statement can hold qualifiers that further describe the statements. For example, the validity period of the statement.

(MediaWiki org, 2021)

Figure 1 shows how the wikibase statement and corresponding qualifier are modeled within the Wikibase.

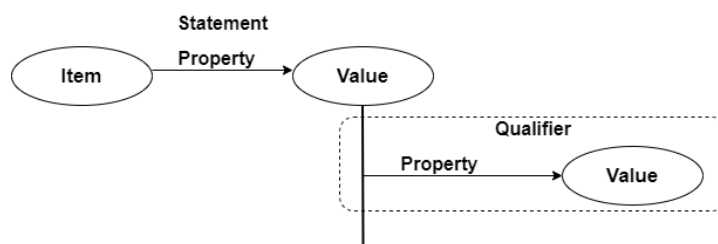


Figure 1: Representation of the statement and qualifier in the Wikibase data model

---

<sup>3</sup> Unified Resource Identifiers

## 4.2 QAnswer

QAnswer solution has been selected for the disability wiki project by considering all the features and services provided by the QA Company. The current requirement of the disability wiki project can be accomplished with the QAnswer query engine platform. The search engine capable to answer questions that are natural language questions. The system learns from the feedback of the user and continuously improves its answering capability. QAnswer allows querying the knowledge graph with natural language questions and analyses each relation from the graph and retrieves the most appropriate answers back to the user.

QAnswer requires a dataset in one of its supported formats. The selected dataset for this project is RDF<sup>4</sup> data format. RDF is a data structure that helps to represent the knowledge graph.

## 4.3 Knowledge Graph

A knowledge graph represents a collection of entities that are described with interlinked properties, objects, events, or concepts. The data together form a connected graph and put data in context via linking with semantic metadata. This provides a framework for data integrity, analytics, and reasoning capability over data. (Ontotext org, 2020)

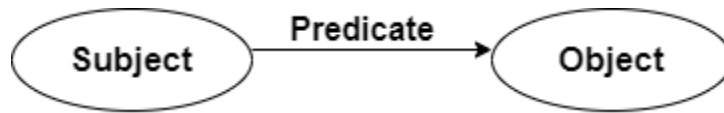
Knowledge graph represents the semantic interrelation between human-like concepts and entities to enable a more accurate interpretation of the data. This helps to form a meaningful and understandable data structure in the form of a graph. The links represent structured metadata that enables better search over graphs and analytics. Entities can be enriched with more linked properties that make it is much more valuable for analysis, visualization, and reporting.

This linked data is created by following different terminology of structured data that allows the specification of relations between different concepts. Due to the use of such constraints and inference rules, information can be queried efficiently using SPARQL protocol (a Query language) and machines can able to perform automated reasoning by understanding the logical relationships of the data. (W3C SPARQL Working Group., 2019)

Resource Description Framework is a standard for model and represents the linked data. An RDF Graph is a set of triplets, each comprised of a subject, a predicate, and an object. An object will be connected to a subject using a predicate (see *figure 2*). A set of statements can be created using such a relation. This structured representation of the data can help to represent the relation between disability document contents and glossary terms. Also, it is possible to link documents with regions, countries, subtopics, and so on. Also, the information can be retrieved with the different constraint that helps to search accurate information in the disability wiki project. Also, QAnswer supports RDF data format for question answering. Therefore the RDF knowledge graph data format has been selected as one of the solutions for the disability project.

---

<sup>4</sup> Resource Description Framework



*Figure 2: RDF Graph consist of single triplet with subject, predicate and object*

Wikibase makes its data available in the RDF data format. The data can be accessed in terms of the individual entity using HTTP requests (Ex: <http://disabilitywiki.univ-st-etienne.fr/wiki/Special:EntityData/Q21.ttl>) or the complete data dump can be downloaded using provided extensions.

# 5. Development Stack and Tools

## 1. Data Store

A datastore is a repository for storing and managing data within a system. The data store is capable of storing all types of data that are produced and used by different programs. They use metadata to structure the schema that helps to store data in a given structure (Techopedia, 2016). These data stores follow different architectural principles to store the data. The most popular architectures are relational databases and triplet (graph) based databases.

### Relational database

Relational databases store information in tabular format. The tables contain a unique column named as the primary key that helps to identify the row. The tables can interlink using foreign key columns based on the relational model (Oracle org, 2018). The relational model is defined using data description language (DDL). Structured Query Language (SQL) is used to insert, update, and delete the data within a database.

The tables are structured and the columns are predefined in a relational database. It is not flexible to introduce a new field to the table in a production environment. The search is limited by columns and tables. It is not possible to search unknown data only by knowing the relation within the database. This shows the limitation of the search facility. The data schema might become very complex when the number of tables and fields increases. It will also difficult to manage and maintain. The performance of the database highly depends on the amount of data and the number of tables. It may report poor performance with a large volume of data. (Roomi, 2021)

### Triple store

A triple store is a data store that stores triples. Triples are descriptive statements composed of three parts, a subject, a predicate, and an object. This pattern helps to describe any logical relations for a given domain. This ability allows reasoning on the dataset and retrieves the information.

The search can be made everywhere on the triple graph. It is not necessarily required to know entities and relations. The data that exist in the triple store is one single corpus and the search query is applied to the complete graph. Also, it is possible to search only a selected named graph if optionally specified. Triple store enriches the search with the reasoning ability to search particular information based on previous search results and constraints. It is capable of handling a large volume of data with complex relations. The performance is comparatively much ahead compared to relational databases, especially with a large volume of data. The triplet format is much effective to represent information in connected graph format. Also, introducing new relations to entities is very easy and does not require modifying the schema structure. The triple store can be queried along with the results of another triple store. Thus, it is possible to integrate different data sources around the world (Data Persée, 2021). Also, it is recommended by the QA Company to use a triple store database for efficient question answering experience. Due to the above reason, the triplestore datastore has been selected for this particular project.

## 2. Programming stack

The major programming stacks are

- Wikibase.
  - Wikibase software is used by thousands of the organization to develop their knowledge graph. we use the Wikibase to store disability knowledge graph. This is already recommended by the QA Company. (MediaWiki org, 2020)
- QAnswer.
  - QAnswer has been utilized to enable intelligent question-answering features to the platform. The further discussion explains how QAnswer integrated to Disability wiki project. (QACompany, 2021)
- *Python*.
  - Python offers a good compromise and it has been used in many enterprise applications. It helps to develop cross-platform applications and server-side applications. Also, the QA Company already uses python to feed data to the datastore. Python is the main programming language in ***Disability-Rights-Wiki*** project and *Flask* is the selected framework. (Python community, 2021)
- *React JS*
  - ReactJS is used in the front-side code base of the platform. ReactJS is a JavaScript-based library that allows developing front side (SPA) Single Page Application. It is simpler and very flexible. This framework is suitable for disability projects since the user interface is not much complex and does not include many modules. (React Team, 2020)
- *GitHub*.
  - GitHub is used for code collaboration to manage, collaborate, and maintain the versioning of the codebase of the platform. This tool was used in this project to maintain source code versioning. (Github org, 2020)
- *MySQL Workbench*.
  - MySQL is used for the database with a relational database schema. This allows the platform to manage data and maintain its integrity. This database is used to store temporary document information before it's get semantified. (MySQL Workbench DevTeam, 2020)



## 6. Disability data modeling

The important goal of the project is modeling and importing information from disability promotional documents and feed into Wikibase. This data modeling part includes identifying the correct ontology for storing document data and find appropriate relations for mapping the data within the wikibase. It is important to define the entities and the relations appropriately to match the current requirement and the domain. *Figure 3* shows the corresponding mappings and relationships among the entities using the ontology diagram.

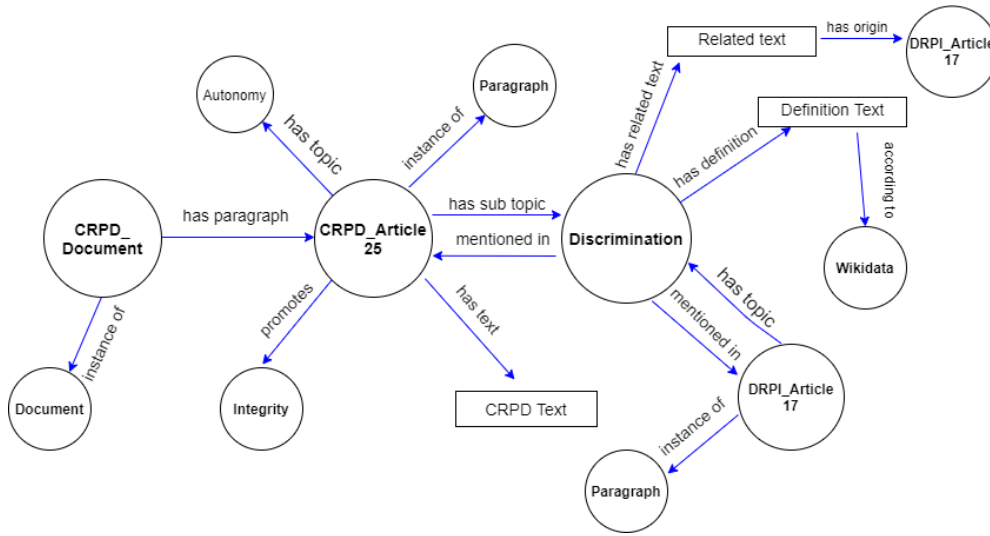


Figure 3: : Ontology diagram of Disability wiki project

As is shown in *figure 3*, Documents, topics, and sources are concepts. Relations are ‘*has topic*’, ‘*has paragraph*’, ‘*according to*’, ‘*mentioned in*’, ‘*has text*’, ‘*has definition*’, ‘*has origin*’ and ‘*instance of*’. The document is an ‘*instance of*’ a document. These documents are split into a list of paragraphs. Paragraphs are mapped to the document using the ‘*has paragraph*’ relation. Each paragraph has one or more topics and subtopics. This relation is indicated using *has ‘has topic’* and has ‘*has sub topic*’ relations. The **paragraph** ‘*has text*’ from the document. The topics are further described using definitions from different sources. The definition mapped using ‘*has definition*’ relation and the source is identified using ‘*according to*’ relation. In this way, it is possible to give many definitions to a specific topic from different sources (*See example: Ontology model for document upload*).

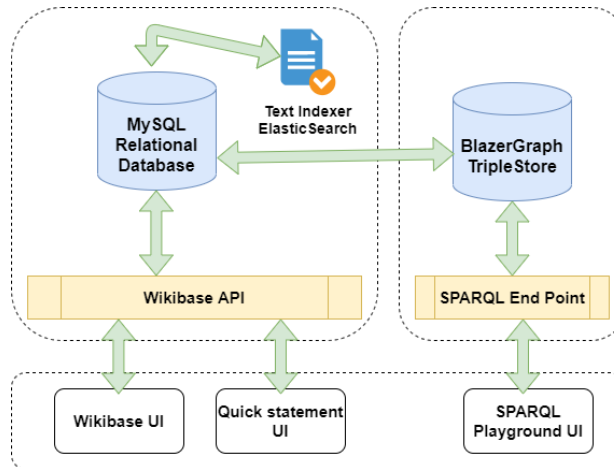
The topics are reverse mapped using ‘*mentioned in*’ relation to the corresponding paragraph. Topics or subtopics can contain related text from a paragraph. This relation has denoted using ‘*has text related*’ relation. That statement is further described using ‘*has origin*’ relation. *Has origin* relation shows the source item of the corresponding text. Wikibase data model is prepared accordingly and the data has been imported by following the given data model.

## 7. Implementation

Disability wiki platform has been implemented by developing a platform. The platform contains different software components that work together to bring required features to the users. The process starts from document upload and flows until information search. Different architecture patterns and products were involved within this process flow. The developed platform helps to extract information from the uploaded document and transform it into semantified data, then access information using natural language question. This automated process capable of enriching the data with new documents. The further discussion presents sub-processes of the implemented platform and then the global overview of the platform.

### 7.1 Setting up Wikibase infrastructure

Wikibase is composed of different components. Each component provides different services and managed to work together to bring the platform live. *Figure 4* shows the architecture of the Wikibase.



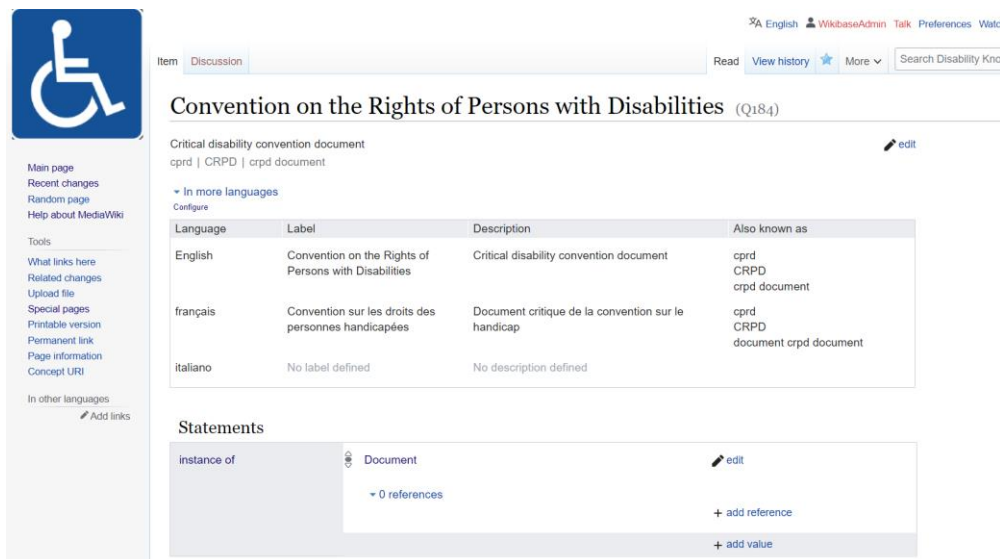
*Figure 4: Architecture and different components of the Wikibase platform*

- **Wikibase Front-end (UI):** Wikibase UI provides a graphical user interface to the platform and allows interaction with the data model. Users can log in to the Wikibase and can create, delete, or update entities, and properties.
- **SPARQL Playground:** SPARQL Playground UI allows to perform SPARQL query operations against the triple store of Wikibase. Users can download or visualize the query results in a different format.
- **MySQL Database:** MySQL database components stores the data model in tabular data. The information is stored as Items, Pages, Statements, Properties, and Qualifiers.

- **ElasticSearch:** ElasticSearch indexes the entities that help to search quickly the concepts and properties within the Wikibase UI.
- **BlazerGraph Triple Store:** Blazer graph store is used as a triple store in Wikibase infrastructure. This stores the create entities, statements, and qualifiers in RDF format and provides knowledge graph data for the created data model. A Wikibase updater monitors the changes that happened in the MySQL database and keeps sync the graphs with data.
- **Wikibase API:** Wikibae API provides several programming endpoints to programmatically edit the data within wikibase. For example, it is possible to create a bot program that creates, searches, queries, edits, and deletes the entities within the Wikibase using provided API.

MediaWiki provides all these components as managed and connected docker containers. Each container connected locally and share the services within them. (Docker org, 2021) Docker-compose is a tool that helps to run several docker containers in a connected and managed environment.

Wikibase platform can be configured using YAML file. This file helps to customize the application hostnames, dependencies between containers, upload local files to containers, define environment variables for containers, and set up different volumes for each container. This complete infrastructure has been configured with custom settings to adapt to the disability wiki project using docker-compose.yml file. The docker-machine has been installed in a Ubuntu server environment and configured with the custom settings as described above. Also, the service is publically available with the following addresses. Wikibase-UI: <https://disabilitywiki.univ-st-etienne.fr/>, Sparql-UI: <http://query-disabilitywiki.univ-st-etienne.fr/>. Figures 5, 6 and 7 are showing the configured arbitrary wikibase for disability wiki project.



The screenshot displays the Wikibase interface for the entity 'Convention on the Rights of Persons with Disabilities' (Q184). The page includes a sidebar with navigation links, a top navigation bar with user options, and a main content area. The main content area features a table with translations and a section for statements.

Language	Label	Description	Also known as
English	Convention on the Rights of Persons with Disabilities	Critical disability convention document	cprd CRPD cprd document
français	Convention sur les droits des personnes handicapées	Document critique de la convention sur le handicap	cprd CRPD document cprd document
italiano	No label defined	No description defined	

**Statements**

instance of Document + 0 references + add reference + add value

Figure 5: Convention on the Rights of Persons with Disabilities wiki page from disability wikibase



## 7.2 Document upload and analyzing

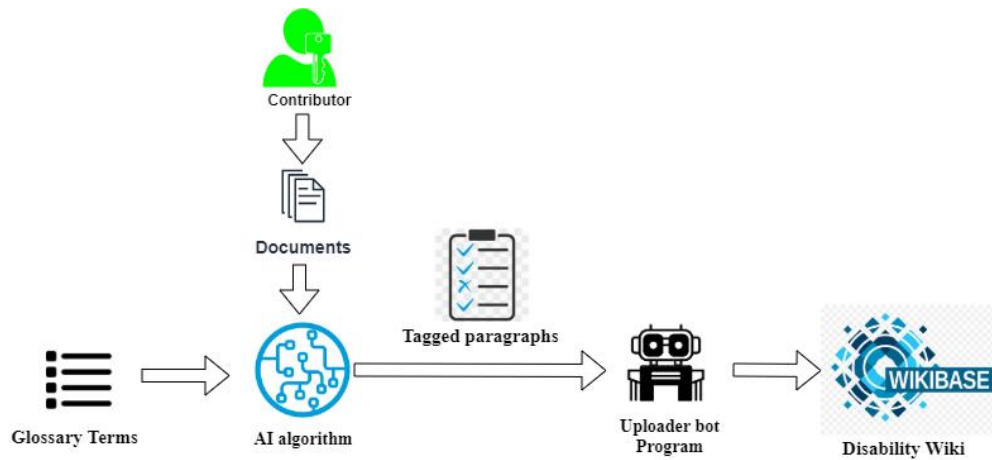


Figure 8: Architecture diagram for document upload process flow

The document upload component allows users to upload documents to the platform (*see figure 8*). The application contains several modules to manage and modify the uploaded documents and classification results. This application is secured using an authentication and authorization mechanism. Only registered users are allowed to upload the document. The uploaded document will be extracted paragraph by paragraph using a paragraph extractor. Then each paragraph will be analyzed by the integrated Machine learning algorithm (Project of the co-intern candidate). The algorithm uses a glossary of health informatics terms that are prepared by domain experts to tag the paragraph based on their semantic meaning. Then the output of the algorithm will be stored in the local database (MySQL). The uploaded documents can be downloaded or viewed in the system. The document status will be displayed based on the current process (Ex: Classified, Pending, Uploaded, etc). Users can able to view and modify the classification results within the application. Finally, the completed results will be requested to upload to the Wikibase.

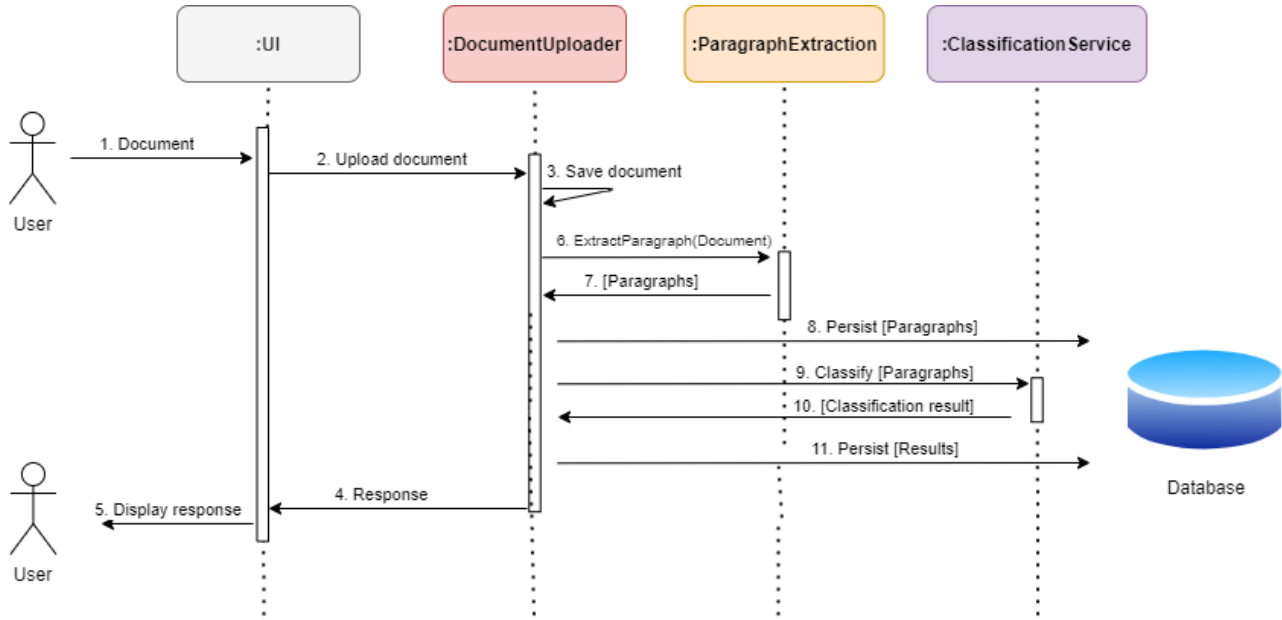


Figure 9: Sequence diagram for document upload and classification

Figure 9 shows the sequence of process call for the document upload process. As it is showing, the uploaded document is firstly extracted into a list of paragraphs and then classified using the classifier. The classified results will be stored in the database. The classification results can be further modified by the authorized user.

## 7.3 Upload edits to Wikibase

Wikibase upload requests are shown to the platform administration users. Only the administrator can approve or reject an upload request. The uploader is a bot program (Pywikibot) that is configured within the web application. The uploader takes the final classification results of the document and semantifies it using the given schema. This semantification is done based on the prepared wikibase data model. Any changes to the wikibase data model will impact this upload process. It is a tightly coupled process with the wikibase data model. Finally, the bot program uploads the prepared data to wikibase according to the given ontology model (*Ontology model for document upload*). This bot program takes configuration from the application *config.py* file and defines the entity classes and relations accordingly.

## 7.4 Integration with Qanswer

Wikibase knowledge graph needs to be uploaded and synchronized continuously to the QAnswer data store. This task is done by uploading a shell script file to the wikibase docker container. This script runs once per day and keeps synchronize the data. The steps for uploading the data to QAnswer as follow,

1. Obtain authentication token from QAnswer platform
2. Dump the Wikibase data (Generates dump.ttl from BlazerGraph triple store)
3. Download the **Flavors**. Flavours can be namespaces or constructed triplets with customized relations using 'SPARQL CONSTRUCT' query. (W3C Org, 2013) 'CONSTRUCT' Queries returns a single graph from triplestore based on the given graph template. The result is itself a graph formed by taking each query solutions from the 'SELECT' query and combining the triples into a single graph by union,
4. Merge flavour data with dump.ttl.
5. Upload the dump.ttl to QAnswer using HTTP POST request.
6. Index the uploaded dataset by sending a HTTP GET request to QAnswer.

## 7.5 Search function

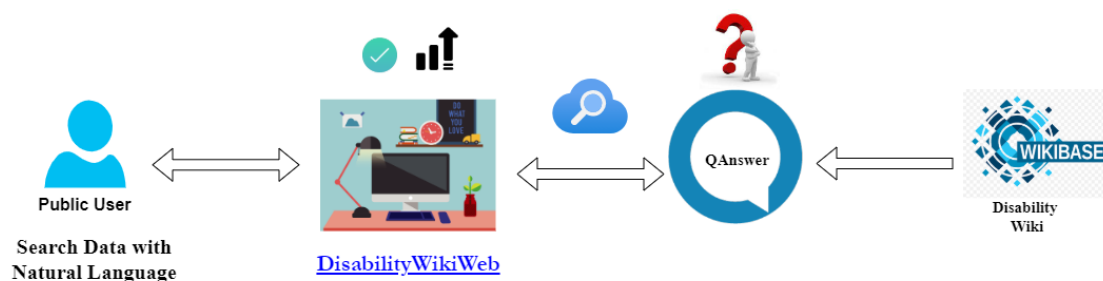


Figure 10: Search function flow of disability wiki website

Disability wiki website has a search feature that allows searching with natural language (*see figure 10*). This front application is connected with QAnswer search engine to facilitate the natural language question-answering feature. QAnswer platform exposed several API endpoints to support question answering over a specific dataset. Thus, anyone can upload their dataset and can start questioning it. As already discussed, QAnswer uses a knowledge graph from the Disability wikibase. The disability wiki website uses QAnswer API endpoints to send the questions and retrieve answers. Initially, the application uses **Autocomplete** feature of QAnswer platform that gives several pre-completed questions while user typings. Thus, the user can able to select these autocomplete options instead of typing complete questions.

This is a very useful feature that automatically finds the entities and their relations and helps to ask questions with correct labels and relations. Because asking wrong questions with incorrect grammar and keywords would result in wrong answers. Also, this feature helps avoiding users from typing malformed questions and incorrect keywords. However, QAnswer is mature enough to answer accurately even with malformed questions.

Once the search button has clicked, the question will be sent to the QAnswer API to get the results. QAnswer sends answers in three formats (Table, Text, Graph). Also, QAnswer might return a list of answers. Currently, only the text answers are shown to the users, Table and Graph answers are ignored.

## 7.6 Continous glossary management

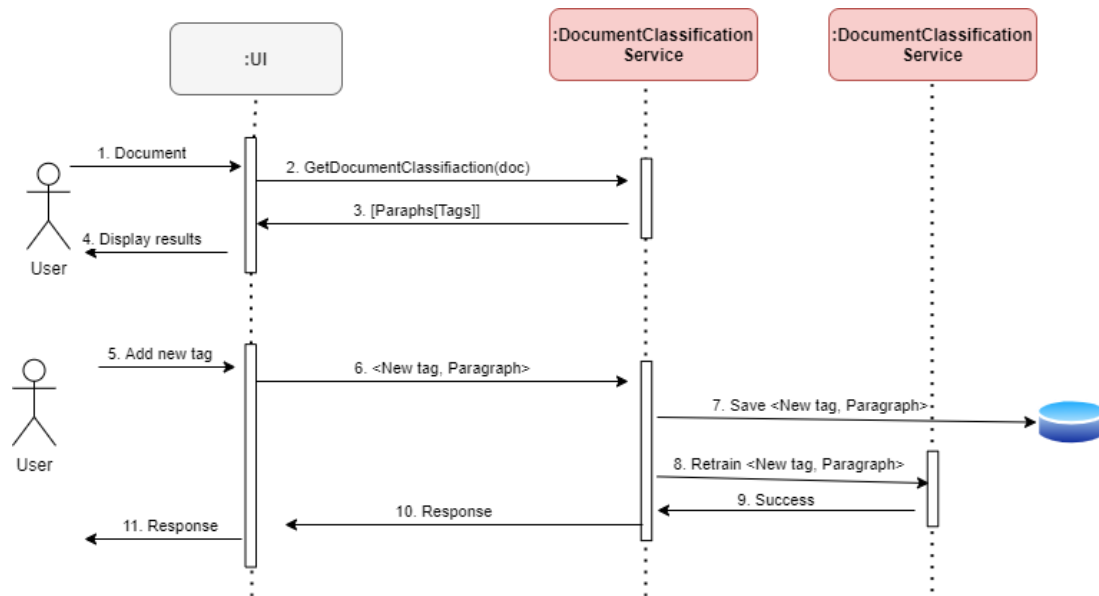


Figure 11: Sequence diagram for retrain process of algorithm with new glossary terms

Figure 11 shows the sequence of program flow for glossary management. Glossary terms are the terms that are prepared by experts and used to tag the paragraph in the process of classification. These terms evolve continuously based on user actions. The classification results contain a list of paragraphs and corresponding tagged glossary terms. Users are allowed to edit the glossary terms. During this process, users can introduce new glossary terms that do not exist in the database. These terms are unknown to the machine learning algorithm. So, it is necessary to introduce the new glossary terms to the algorithm. This process is called retraining. The system stores these new terms along with the paragraph text. Hence, it is possible to retrain the machine learning algorithm with new terms. Thus, the algorithm will use these new terms for the classification process in the future. The retraining part is not yet completed and added to the future release.



## 7.7 Global overview

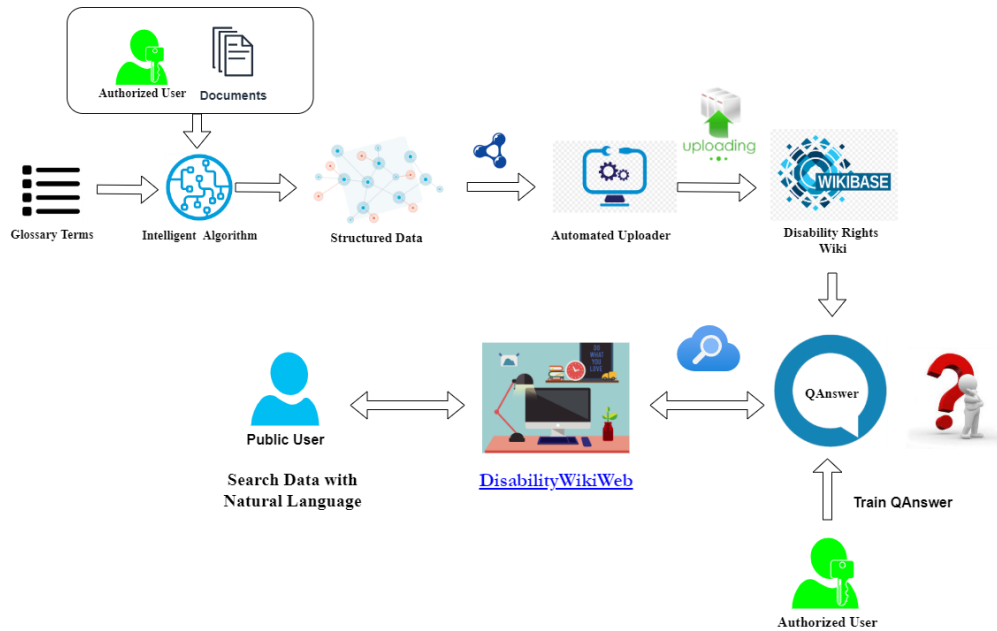


Figure 12: Platform architecture of Disability wiki project

Figure 12 shows the complete architecture of the platform. As it's shown, several components work together and provide different service support to the platform. The process starts with the document upload component. Only authorized users are allowed to upload documents to the platform. The document upload component is integrated with the document classification service and glossary management service. All these components prepare structured data from uploaded documents. An automatic updater is a bot program that uploads this structured data to wikibase. Wikibase is running on docker and contains several other components. The data from wikibase will be synchronized with QAnswer. QAnswer requires training. Training can be done by an authorized user. Disability wiki website is a search interface that allows to search information with natural language. This website connected with the QAnswer search engine and retrieve results from it. This website is public and accessible to all general users.

## 8. Supportive implementation

In addition to the main task, I also performed a supportive task that involved demonstrating the integration process of the OpenRefine tool. OpenRefine is a tool for working with messy data to clean it, transform it, and upload it to the wikibase data store. It is possible to upload structured data like CSV, Excel files directly to wikibase using this tool. It is a very flexible and efficient tool for this process. This task is involved in demonstrating two different tools.

### 1. Reconciliation service

(Installing the reconciliation service, 2020), OpenRefine reconciliation service is a service that has developed using Python. This service connects with configured wikibase and helps reconciliation of the data such as search and link entities and import corresponding properties of the matched entities. This tool uses both wikibase API and the Sparql endpoint of a given wikibase instance. The tool offers a powerful search by applying different search criteria and constraints. Originally, the tool is designed to work for the Wikidata platform and later released with a feature to support arbitrary wikibases. This service is also available as a docker container.

### 2. OpenRefine front application

This is a web application developed using Java and Javascript, together bring user interface to interact with reconciliation service. It is important to configure this application to communicate with a reconciliation service. Users can upload CSV, and Excel files to the wikibase using this application (OpenRefine, 2020).

### Configuration steps:

1. Configure and run Reconciliation service to the Disability wikibase.
  - Pull docker container.
  - Create a configuration file with properties of disability wikibase (see *Reconciliation configuration file*).
  - Edit docker-compose file to copy the configuration file.
  - Up and run the docker container with prepared configurations.
2. Configure and run the OpenRefine front application in a way that can communicate with the Reconciliation service.
  - Create a configuration file with the properties of the Reconciliation service.
  - Create a wikibase manifest file to point disability wikibase (see *OpenRefine manifest file*).
  - Run the application with the above configuration files.

## 9. Status of in-use implementation

This report presented the work done to develop the disability project where users can search for information easily and efficiently. The developed project has been validated by supervisors from University Jean Monnet, York University, and the QA Company. This project is expected to officially launch in 2022. The current implemented version is a pilot prototype. This version of the system is hosted in the University Jean Monnet infrastructure. This will be used by community people for test purposes and to define further roadmap and milestones accordingly. Based on it, project development will be continued and further improvement will be performed to improve the functionalities and performance.

## 10. Conclusion

The implemented platform helps to upload and semantifying the convention documents. Users can upload documents to the platform using a user-friendly interface with few clicks. The semantified information will be uploaded to the wikibase without user interaction. Thus, the data and information will be enriched enormously and will be used by many organizations around the world. The platform allows public users to search for information with advanced natural language question-answering features. This helps different communities and organizations to search and conveniently explore document information. The interested communities can search for information in one place rather than spending time on the internet and large text documents. The result shows good answers and much accurate information. Continuous glossary management helps to improve the classification results. It keeps tracking the new glossary terms introduced by the contributors and prepares training data behind the process. Thus, it eliminates the manual retraining of the model with a list of new glossary terms. The model gets improved over time and reduces the need for domain experts to validate the results. The wikibase data is auto synchronized everyday night with the QAnswer platform. This feature assures that the data is up to date and the information is valid. Thus, all uploaded documents are accessible through the search engine within one day. The current version of the application is helpful to understand the actual need of the project and the user preferences.

The current implementation only considered few relations from developed ontology (ex: has topic, has subtopic). It is required to introduce other relations to the system such as 'prevents', 'supports', 'aware of', and so on. Otherwise, it can be less interesting to the communities and organizations. The load balancing <sup>5</sup>feature is not yet considered for this version of the application. However, is it the most needed feature, because classification service will take longer to execute with a 100-pages document. Thus, decoupling the long task is necessary. Otherwise, the user will wait longer with the unresponsive website. Apart from these factors, the current application is efficient enough to be demonstrated as an initial prototype for the project. Further development will improve the usability of the system.

---

<sup>5</sup> Create optimized approach to handle thousands of upload requests without breaking the system

# 11. Future work

The current project requires many improvements and feature enhancements. Firstly, the document data is semantified using few relations such as ‘has paragraph’, ‘has topic’, and ‘mentioned in’. Future work will include other relations such as ‘comply with’, ‘supports’, ‘has legislation’, and so on. It is also required to update the machine learning classifier to extract such semantic information from the uploaded document. Uploaded documents should link with their corresponding country. Some documents are country and region-specific. The current version of the application contains such properties in the front application. However, it is not considered in the wikibase data model. It is important to semantically link documents with country and region. This can be done by importing countries and regions from Wikidata to disability wikibase. Then, if the uploaded document contains region or country options, it is possible to link between the document entity and country entity using appropriate relation.

Document upload platform should be optimized to adopt a large number of requests. There can be a situation where many users may upload documents simultaneously. The web application should be capable enough to ‘load balance’ these multiple requests and optimize the performance. This can be done using a message broker architecture where the main web application and classification service are decoupled. Both services should be communicated using an asynchronous message broker in ‘**Queue**’ based service protocol. Another most essential improvement is universal accessibility.

Universal accessibility helps people with disabilities to access information as normally as other users. Many improvements are required to achieve this goal such as screen reader integration, voice search integration, magnifier screen feature, and so on. This is very important because of the targeted community. The current version of the application is targeted only at English documents. It is necessary to consider other languages since different communities will upload and search information in a different language. Many use cases were discovered during the first prototype demonstration such as a content summary that helps family people to understand the contents, user-centric information, and so on. All above-stated considerations and improvements will be discussed and developed in the future release of the system to give a better experience to the target community.

## 12. Appendix

### 12.1 Search results (Question answers)

Question: Documents about human rights

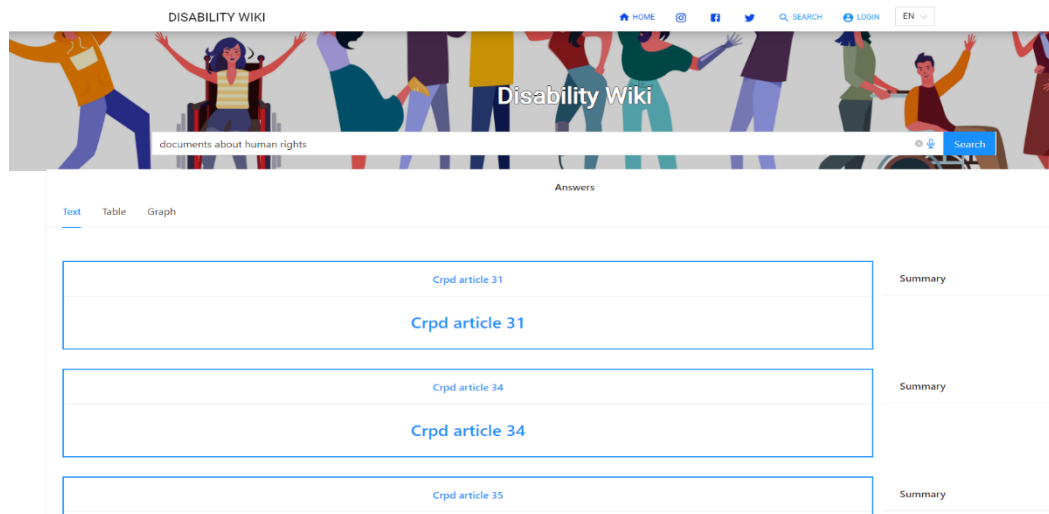


Figure 13: Test 1: Question answer results from disability wiki website

Question : Test about equity from documents

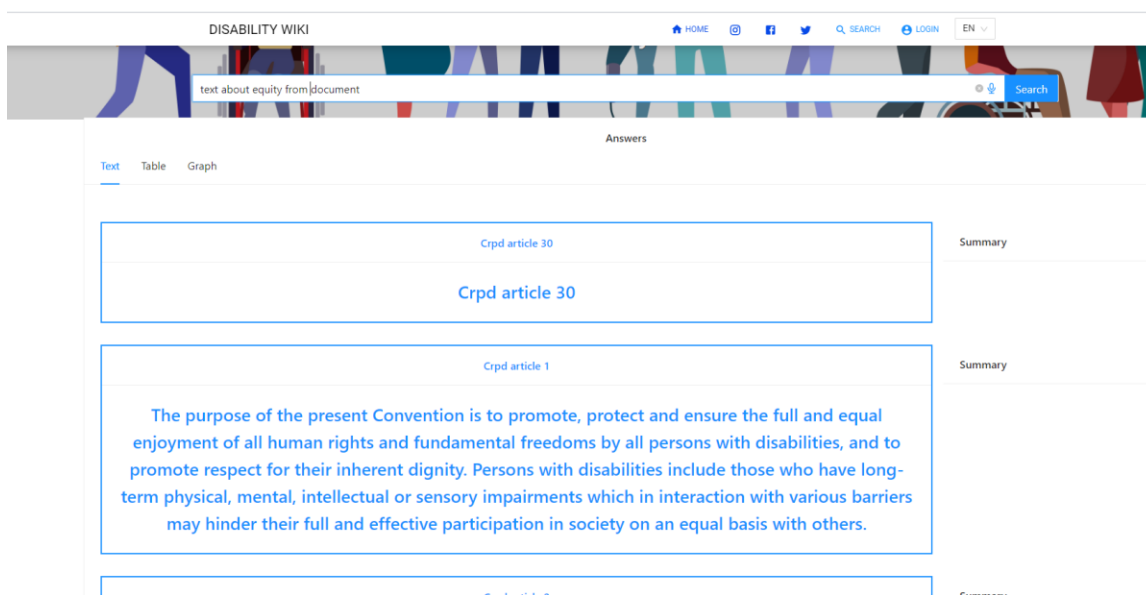


Figure 14: Test 2: Question answer results from disability wiki website

Question : Topics of crpd article 11

The screenshot shows the Disability Wiki interface. At the top, there's a header with the text "Disability Wiki" and a search bar containing the query "topics of crpd article 11". Below the header, there's a section titled "Answers" with tabs for "Text", "Table", and "Graph". The "Text" tab is selected. The results are displayed in two main boxes. The first box is titled "Humanitarian emergencies" and contains the text "Humanitarian emergencies". The second box is titled "Natural disaster" and contains the text "Natural disaster". To the right of these boxes, there's a "Summary" section. The summary for "Humanitarian emergencies" is empty. The summary for "Natural disaster" contains the following information: "mentioned in Crpd article 11", "has subtopic Natural disaster", "instance of Topic", "http://www.w3.org/1999/02/22-rdf-syntax-ns#type http://wikiba.se/ontology#Item", and "has subtopic Natural disaster".

Figure 15: Test 3: Question answer results from disability wiki website

Question : Documents about discrimination

The screenshot shows the Disability Wiki interface. At the top, there's a header with the text "Disability Wiki" and a search bar containing the query "documents about discrimination". Below the header, there's a section titled "Answers" with tabs for "Text", "Table", and "Graph". The "Text" tab is selected. The results are displayed in two main boxes. The first box is titled "Crpd article 25" and contains the text "Crpd article 25". The second box is titled "Crpd article 2" and contains the text "Communication includes languages, display of text, Braille, tactile communication, large print, accessible multimedia as well as written, audio, plain-language, human-reader and augmentative and alternative modes, means and formats of communication, including accessible information and communication technology Language includes spoken and signed languages and other forms of". To the right of these boxes, there's a "Summary" section. The summary for "Crpd article 25" is empty. The summary for "Crpd article 2" contains the text "Communication includes languages, display of text, Braille, tactile communication, large print, accessible multimedia as well as written, audio, plain-language, human-reader and augmentative and alternative modes, means and formats of communication, including accessible information and communication technology Language includes spoken and signed languages and other forms of".

Figure 16: Test 4: Question answer results from disability wiki website

## 12.2 Ontology model for document upload

...

**Document Name: Cameroon Disability Rights**

**Belongs to >> Canada**

instance of >> **Document**

has\_paragraph >> **Cameroon Disability Rights\_paragraph001**

has\_paragraph >> **Cameroon Disability Rights\_paragraph002**

has\_paragraph >> **Cameroon Disability Rights\_paragraph003**

**Cameroon Disability Rights\_paragraph001**

**Belongs to >> Canada**

has\_text >> "text of the paragraph"

has\_topic >> **Education rights**

**Health**

mentionned\_in >> **Cameroon Disability Rights\_paragraph001**

...

## 12.3 Reconciliation configuration file (Important properties)

```
mediawiki_api_endpoint = 'http://host.docker.internal:8181/w/api.php'
wikibase_sparql_endpoint = 'http://host.docker.internal:8989/bigdata/namespace/wdq/sparql'
wikibase_main_page = 'http://host.docker.internal:8181/wiki/Main_Page'
wikibase_namespace_id = 0
wikibase_namespace_prefix = ''
qid_url_pattern = 'http://host.docker.internal:8181/wiki/Item:{{id}}'
default_type_entity = 'Q35120'
type_property_path = 'P31'
wdt_prefix = 'wdt:'
sparql_query_to_fetch_unique_id_properties = """
SELECT ?pid WHERE { ?pid wdt:P35/wdt:P302* wd:Q240641 }
"""
```



## 12.4 OpenRefine manifest file

```
{
  "version": "1.0",
  "mediawiki": {
    "name": "my wiki",
    "root": " https://disabilitywiki.univ-st-etienne.fr/wiki/",
    "main_page": " https://disabilitywiki.univ-st-etienne.fr/wiki/Main_Page",
    "api": "http://dev.com:8181/w/api.php"
  },
  "wikibase": {
    "site_iri": " https://disabilitywiki.univ-st-etienne.fr/entity/",
    "maxlag": 5,
    "properties": {
      "instance_of": "P38",
      "subclass_of": "P279"
    }
  },
  "reconciliation": {
    "endpoint": " https://disabilitywiki-reconciliation.univ-st-etienne.fr/${lang}/api"
  }
}
```

# References

1. Adlib org. (2021, 01 01). *Revolutionize Document Classification with AI & ML*. Retrieved from Adli: <https://www.adlibsoftware.com/products-and-services/offerings/classification>
2. Aitenders. (2021, 01 01). *Document Control*. Retrieved from Aitenders: <https://www.aitenders.com/documents-consistency-check/>
3. CDC. (2020, 09 06). *Disability and Health Overview*. Retrieved from Center for disease control and prevention: [https://www.cdc.gov/ncbddd/disabilityandhealth/disability.html#:~:text=A%20disability%20is%20any%20condition,around%20them%20\(participation%20restrictions\).](https://www.cdc.gov/ncbddd/disabilityandhealth/disability.html#:~:text=A%20disability%20is%20any%20condition,around%20them%20(participation%20restrictions).)
4. Data Persée. (2021). *What is a triplestore?* Retrieved from Data Persée: <http://data.persee.fr/understanding/what-is-a-triplestore/?lang=en>
5. Docker org. (2021, 06 15). *Overview of Docker compose*. Retrieved from docker docs: <https://docs.docker.com/compose/>
6. Enslaved org. (2021). *about*. Retrieved from enslaved: <https://enslaved.org/about>
7. Exrtact corp. (2020, 12 30). *AUTOMATED DOCUMENT CLASSIFICATION & INDEXING SOFTWARE*. Retrieved from extract: <https://www.extractsystems.com/automated-document-classification-and-indexing-software>
8. Github org. (2020, 06 18). *Github*. Retrieved from Github: <https://github.com/>
9. *Installing the reconciliation service*. (2020, 01 10). Retrieved from Openrefine-wikibase: <https://openrefine-wikibase.readthedocs.io/en/latest/install.html>
10. Marco Anteghini, Jennifer D'Souza. (2020). Representing Semantified Biological Assays in. *ResearchGate*, 2-8.
11. MediaWiki org. (2020, 05 06). *MediaWiki is a collaboration and documentation platform brought to you by a vibrant community*. Retrieved from MediaWiki: <https://www.mediawiki.org/wiki/MediaWiki>
12. MediaWiki org. (2020, 12 05). *Wikibase/Install*. Retrieved from MediaWiki: <https://www.mediawiki.org/wiki/Wikibase/Install>
13. MediaWiki org. (2021, 02 02). *Wikibase/Datamodel*. Retrieved from Mediawiki: <https://www.mediawiki.org/wiki/Wikibase/DataModel>
14. MySQL Workbench DevTeam. (2020, 06 25). *MySQL Workbench*. Retrieved from MySQL Workbench: <https://www.mysql.com/products/workbench/>

15. Ontotext org. (2020). *What is a Knowledge Graph?* Retrieved from Ontotext:  
<https://www.ontotext.com/knowledgehub/fundamentals/what-is-a-knowledge-graph/>
16. OpenRefine. (2020). *Welcome*. Retrieved from OpenRefine: <https://openrefine.org/>
17. Oracle org. (2018). *what is relational database*. Retrieved from Database:  
<https://www.oracle.com/database/what-is-a-relational-database/>
18. ORKG. (2021). *Contribution comparison*. Retrieved from ORKG:  
<https://www.orkg.org/orkg/comparison?contributions=R48195,R48179,R48147>
19. Python community. (2021). *Python*. Retrieved from python org: <https://www.python.org/>
20. QACompany. (2021). *Team*. Retrieved from QA Company: <https://the-qa-company.com/team/>
21. QAnswer. (2021, 06 13). *Accessing your Knowledge via Natural Language*. Retrieved from QAnswer:  
<https://www.qanswer.eu/>
22. React Team. (2020, 02). *React org*. Retrieved from Getting Started: <https://reactjs.org/docs/getting-started.html>
23. Roomi, M. (2021, 04 14). *Advantages and Disadvantages of Relational Database | Limitations & Benefits of Relational Database*. Retrieved from Hightechwhizz: <https://www.hitechwhizz.com/2021/04/6-advantages-and-disadvantages-limitations-benefits-of-relational-database.html>
24. Techopedia. (2016, 05 13). *Datastore*. Retrieved from techopedia:  
<https://www.techopedia.com/definition/23343/datastore>
25. W3C Org. (2013, 03 26). *SPARQL Query Language for RDF*. Retrieved from W3C:  
<https://www.w3.org/TR/rdf-sparql-query/#construct>
26. W3C Org. (2014). *SEMANTIC WEB*. Retrieved from W3C:  
<https://www.w3.org/standards/semanticweb/#:~:text=The%20term%20%E2%80%9CSemantic%20Web%E2%80%9D%20refers,SPARQL%2C%20OWL%2C%20and%20SKOS.>
27. W3C SPARQL Working Group. (2019, 03 19). *SPARQL 1.1 Overview*. Retrieved from Sparql:  
<https://www.w3.org/TR/sparql11-overview/>
28. Weikang Li, Wei Li, Yunfang Wu. (2018). A Unified Model for Document-Based Question Answering. *The Thirty-Second AAAI Conference* (pp. 1-6). China: Peking University, MOE,.