Heterogeneous Information Resources Handler (**He***In***H***and*)

Project Documentation

Summer: 2019/2020

CONTENT

1. Project Vision

1.1) Software Product Vision

The main idea behind the heterogeneous Information Retrieval (IR) System, is to deal with the representation, storage, organization of , and access to information items we are interested in. This makes adding information, updating current information already added to the table, and removing information not needed within the system.

1.2) Software Process Vision

The aim of **information retrieval** is to provide the user with the “best possible" **information** from a database. The problem of **information retrieval** is determining what constitutes the best possible **information** for a given user query.

Without information retrieval systems, they will be information overload. This information overload is usually caused by the existence of multiple sources of information, overabundance of information, difficulty in managing information, irrelevance of the received information and scarcity of time on the part of information users want to analyze and understand.

How does an Information retrieval system solve the issue of information overload? When the process begins when a user enters a query into the system. Queries are formal statements of **information** needs, for example search strings in web search engines. In **information retrieval** a query **does** not uniquely identify a single object in the collection

The vision within the project is to be able to create a system, able to combine different tables, scatter around into one general useful application that is convenient for each business, to work with.

a. Project Specifications (some directions):
   - o Interactive, GUI oriented, Event Driven software Tool
   - o The Client Architectural model:
     - External Client Architecture - dynamic collection of GUI components:
       - Tool Handling Component:

- Artifact Handling Component (repository metamodels and repository instances handler)
- Help Componente;

▪ Internal Component Architecture:

- MVC (Model-View-Controller) - Architectural Pattern
- Design Patterns (Observer, Singleton, State, Composite, Factory Method, Abstract Factory, Proxy, Bridge, Command)

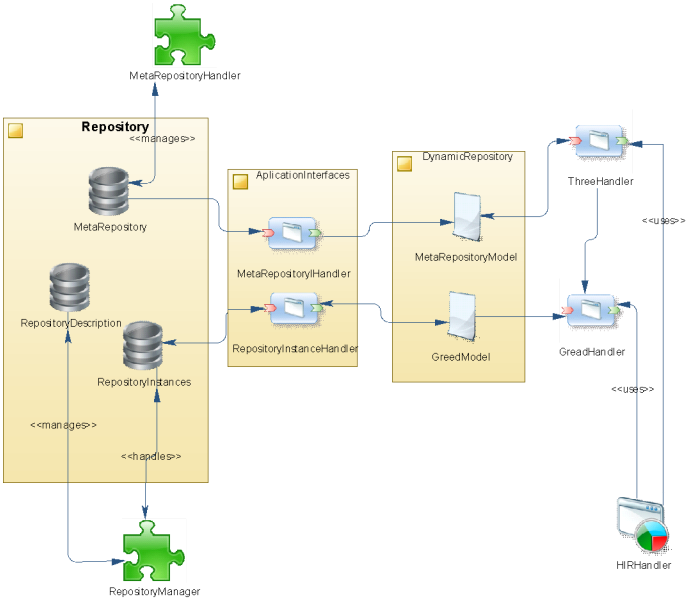
▪ Tool Settings (Parametrisation, Localisation and Personalisation)

- Logotype, Language and About
  - Algorithms and Data Structures

▪ Information Resource Abstraction

▪ Create, Update, Delete (CRUD) and Search Operations

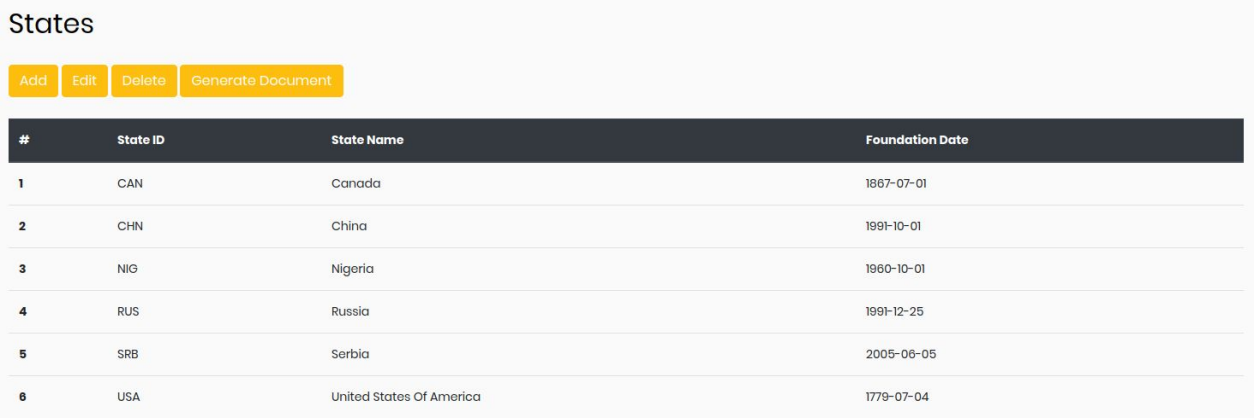
o Prototype - Evolutionary Prototype based on Object Oriented Paradigm (arbitrary programming language, GUI library, and development environment).



Heterogeneous Information Resources Handler(He*in*H*and*) - EAM

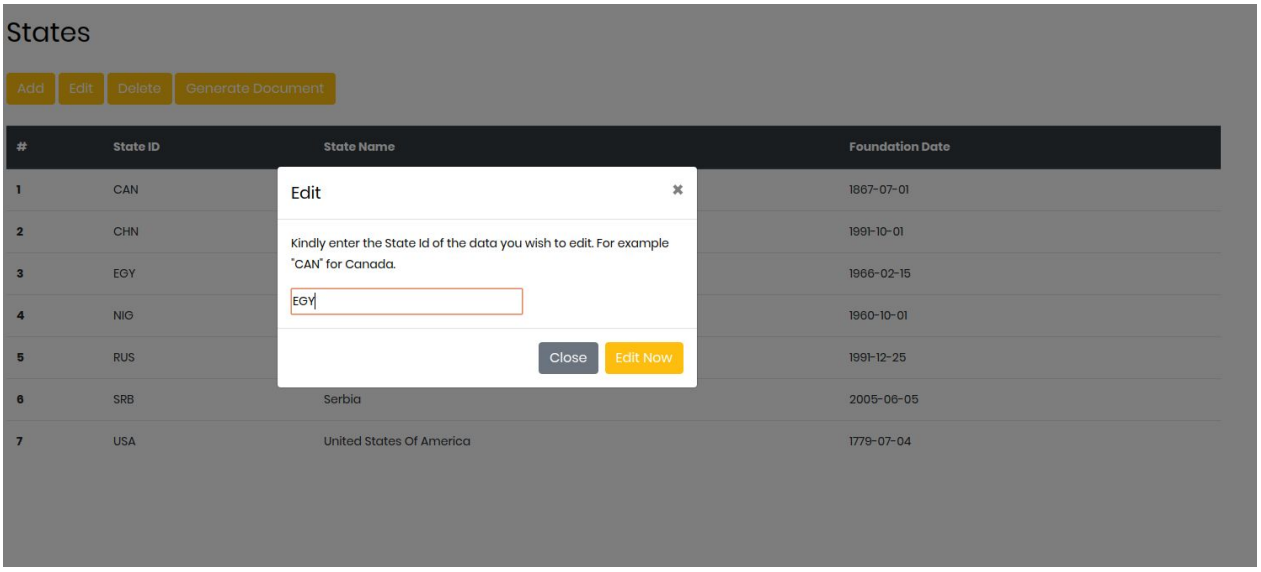
2. The Class Model of Heterogeneous Information Resources Handler(HeI*n*H*and*)

In this section of the project, the user is able to perform CRUD operations. This includes operations that support the ability of storing / retrieving / mining of relational data.

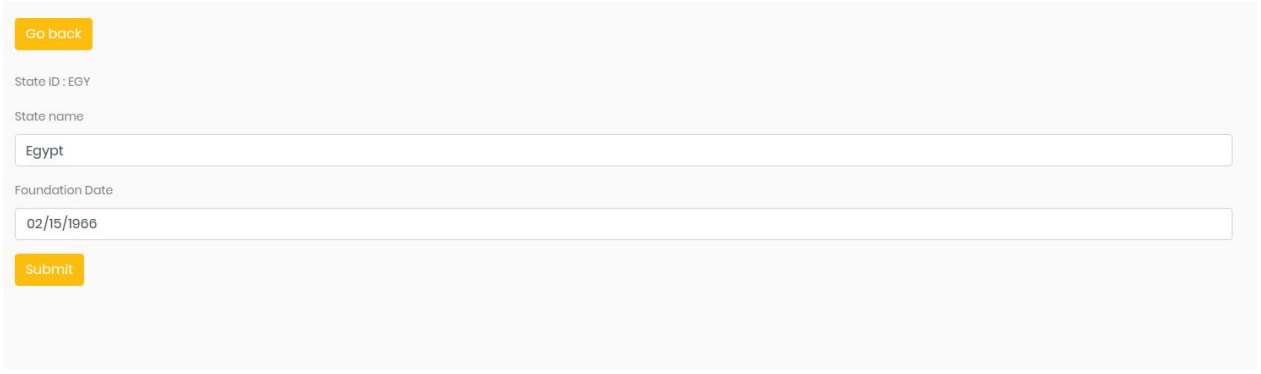


With the click of the Add button, The user is able to add store data into the database, such as the State Id, State name, and the foundation date.

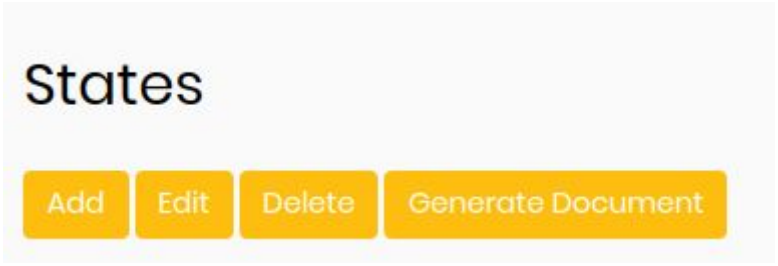
Once that is complete, if necessary changes are required to be performed, the user has the privilege to use the Edit button to correct the mistakes already made. Once the Edit button is clicked he/she must know the State Id, he or she wants to make changes to, then click the Edit now button.

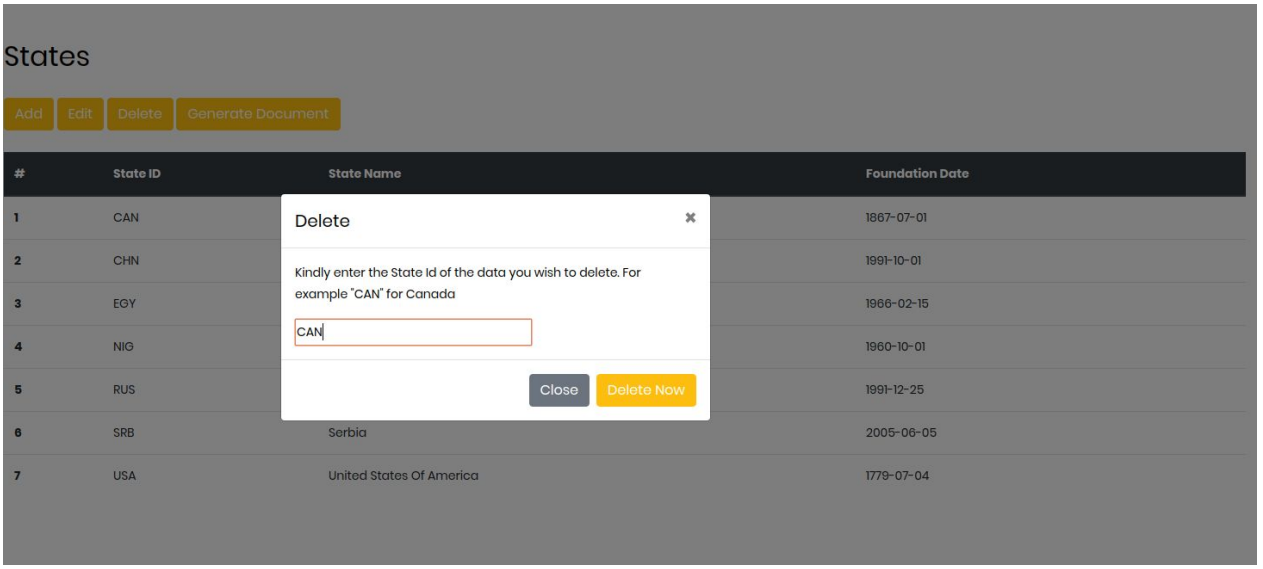


Once the edit button is clicked, he/she may make the necessary changes needed to be made, then can now, update, the Name of the State and the Foundation Date. Then click the Submit button.



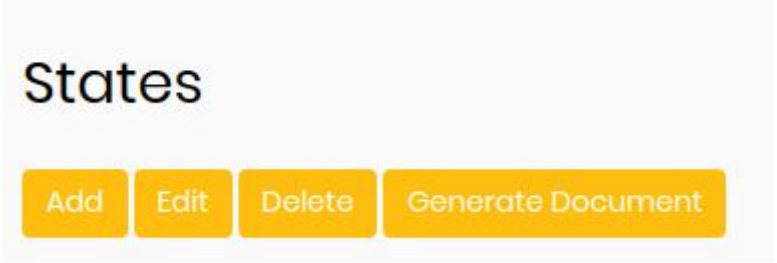
If the user doesn't want a row on their table, he/she must click the Delete Button, Once the Delete button is clicked the user must know the State Id, he/she wants to remove and specify that State ID, then click the Delete now button.





Then finally, delete an existing row from our table by specifying the State ID of that state.

With the click of the Generate Document button, the user is able to create documents, this comes in the form of a JSON type document, to list out all the States name, State Foundation Date along with their State Id.



document 38 - Created at: "2020-07-04T09:16:05.150Z"


```
{
    "_id": "5f0048d5bdaace3059cf3a15",
    "created_at": "\"2020-07-04T09:16:05.150Z\"",
    "allStates": [
        {
            "stateId": "CAN",
            "stateName": "CanadaA",
            "stateFoundationDate": "1867-07-01"
        },
        {
            "stateId": "CHN",
            "stateName": "China",
            "stateFoundationDate": "1991-10-01"
        },
        {
            "stateId": "EGY",
            "stateName": "Egypt",
            "stateFoundationDate": "1966-02-15"
        },
        {
            "stateId": "NIG",
            "stateName": "Nigeria",
            "stateFoundationDate": "1960-10-01"
        },
        {
            "stateId": "RUS",
            "stateName": "Russia",
            "stateFoundationDate": "1991-12-25"
        },
        {
            "stateId": "SRB",
            "stateName": "Serbia",
            "stateFoundationDate": "2005-06-05"
        },
        {
            "stateId": "USA",
            "stateName": "United States Of America",
            "stateFoundationDate": "1779-07-04"
        }
    ]
}
```

3. The Document model for MongoDB

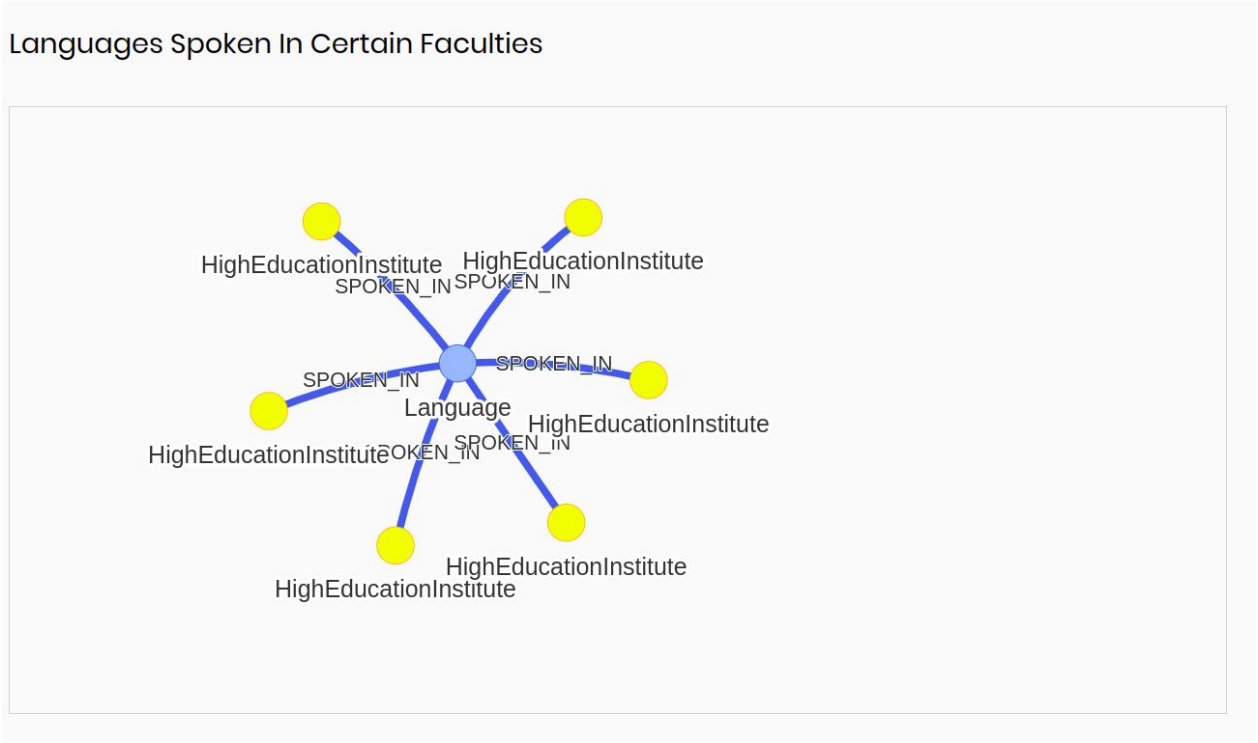
In the json specification document we can see the relationship between each country and cities. Information such as the id of the country, the numbers of cities within the country and also the cities within this country are saved. Within the cities section, we also have more specific details like the id, name of the cities, as well as the zip code.

document 139 - Created at: "2020-07-04T09:18:55.347Z"


```
{
    "_id": "5f00497fbdaace3059cf3a17",
    "created_at": "\"2020-07-04T09:18:55.347Z\"",
    "highEducationInstitutes": [
        {
            "id": 1,
            "instituteName": "British International School",
            "instituteType": [
                {
                    "TIP_UST": "HS",
                    "TIP_NAZIV": "High School"
                }
            ],
            "states": [
                {
                    "id": "NIG",
                    "name": "Nigeria",
                    "foundationDate": "1960-10-01",
                    "numberOfCities": 1,
                    "cities": [
                        {
                            "id": 1,
                            "name": "Abuja",
                            "zip": "74523"
                        }
                    ]
                }
            ],
            "ownershipType": [
                {
                    "VV_OZNAKA": "PR",
                    "VV_NAZIV": "Private"
                }
            ]
        },
```

4. The Graph model for Neo4j

In the graph visualization, we were able to model the relationship between the language spoken in specific institutes. For the graph below, we made a visualization of where the Serbian language is spoken at.



5. Zipped working application with running instructions.

In order to run the project first you will have to unzip the project. After unzipping, the next step is to install all the required libraries and applications needed to run the project. These applications are mysql-server, mongodb, and neo4j.

Installing MySQL

Let's start by installing MySQL. MySQL is an open-source database management system, commonly installed as part of the popular LAMP (Linux, Apache, MySQL, PHP/Python/Perl) stack. It uses a relational database and SQL (Structured Query Language) to manage its data.

The short version of the installation is simple: update your package index, install the mysql-server package, and then run the included security script.

- sudo apt update
- sudo apt install mysql-server
- sudo mysql_secure_installation

Installing MongoDB

Next, we have to install MongoDB Community on your Ubuntu system, these instructions will use the official mongodb-org package, which is maintained and supported by MongoDB Inc. The official mongodb-org package always contains the latest version of MongoDB, and is available from its own dedicated repo.

1) Import the public key used by the package management system

- wget -qO - https://www.mongodb.org/static/pgp/server-4.2.asc | sudo apt-key add -

The operation should respond with an OK.

However, if you receive an error indicating that gnupg is not installed, you can:

Install gnupg and its required libraries using the following command:

- sudo apt-get install gnupg

Once installed, retry importing the key:

- wget -qO - https://www.mongodb.org/static/pgp/server-4.2.asc | sudo apt-key add -

2) Create a list file for MongoDB

Create the list file /etc/apt/sources.list.d/mongodb-org-4.2.list for your version of Ubuntu.

Click on the appropriate tab for your version of Ubuntu. If you are unsure of what Ubuntu version the host is running, open a terminal or shell on the host and execute lsb_release -dc.

The following instruction is for Ubuntu 18.04 (Bionic). For Ubuntu 16.04 (Xenial), click on the appropriate tab.

Create the /etc/apt/sources.list.d/mongodb-org-4.2.list file for Ubuntu 18.04 (Bionic):

- echo "deb [arch=amd64,arm64] https://repo.mongodb.org/apt/ubuntu bionic/mongodb-org/4.2 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-4.2.list

3) Reload local package database. Issue the following command to reload the local package database:

- sudo apt-get update

4) Install the MongoDB packages

You can install either the latest stable version of MongoDB or a specific version of MongoDB. To install the latest stable version, issue the following

- sudo apt-get install -y mongodb-org

Optional. Although you can specify any available version of MongoDB, apt-get will upgrade the packages when a newer version becomes available. To prevent unintended upgrades, you can pin the package at the currently installed version:

- echo "mongodb-org hold" | sudo dpkg --set-selections
- echo "mongodb-org-server hold" | sudo dpkg --set-selections
- echo "mongodb-org-shell hold" | sudo dpkg --set-selections
- echo "mongodb-org-mongos hold" | sudo dpkg --set-selections
- echo "mongodb-org-tools hold" | sudo dpkg --set-selections

To start MongDB, run the command, sudo systemctl start mongod

Verify that MongoDB has started successfully

- sudo systemctl status mongod
- Stop MongoDB - sudo systemctl stop mongod

Installing Neo4j

First we'll add the repository key to our keychain.

- wget --no-check-certificate -O - https://debian.neo4j.org/neotechnology.gpg.key | sudo apt-key add -

Then add the repository to the list of apt sources.

- echo 'deb http://debian.neo4j.org/repo stable/' | sudo tee /etc/apt/sources.list.d/neo4j.list

Finally update the repository information and install Neo4j.

- sudo apt update
- sudo apt install neo4j

The server should have started automatically and should also be restarted at boot. If necessary the server can be stopped with

- sudo service neo4j stop

To start the project, you will need to run, open the project with some editor. In our case, Visual Studio. Then run the following command,

- nodemon index.js.

Then go directly to your browser, and type in the domain name
http://localhost:3000/