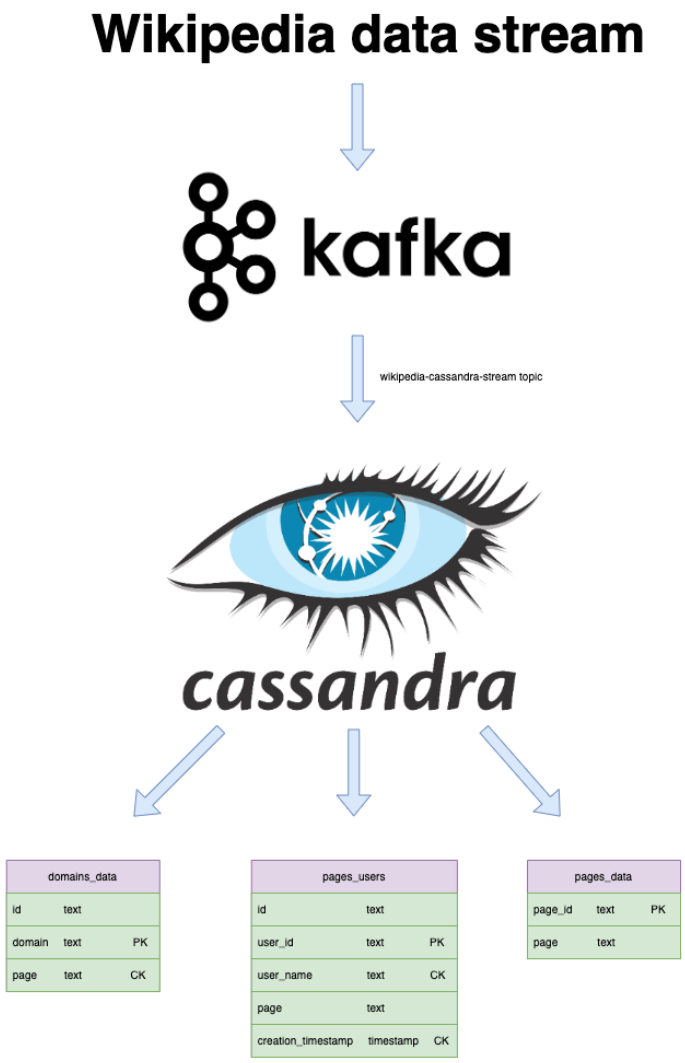
Wikipedia data stream processing project

Design description

System design looks like this:



1. To process Wikipedia stream data there was decided to use **Kafka** and write data that has key “data” to the messages queue.
2. From the message queue the consumer gets the data and loads it to **Cassandra** database.

3. To get needed answers for questions there was created **API** that has 5 different endpoints that gets needd data from table in Cassandra.

Kafka Usage

For the initial process of stream data, **Kafka** was chosen.

Here was decided to use **Kafka** because of such reasons:

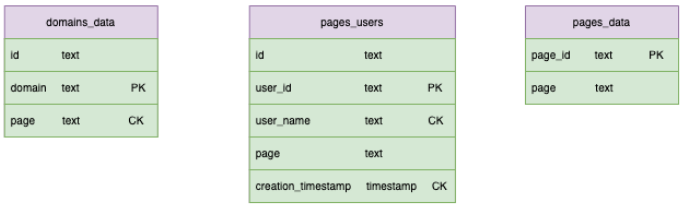
1. It is suitable for real time data processing.
2. It creates a great pipeline of getting messages from the stream, adding it to the queue, where they will wait for consumer.
3. It is a reliable system to save messages.

Tables description

For saving the data **Cassandra** database management system was chosen.

The main reasons for choosing Cassandra are:

1. It is flexible and scalable.
2. It is suitable to handle a large stream of data. In our case there is a big stream of Wikipedia data and when the system will work for a long time there will be a great amount of data.
3. It has great performance and is rather fast. When choosing great PRIMARY KEY and CLUSTERING KEY we will be able to get the necessary data quickly enough.



There was created **3** tables:

1. **domains_data** - contains information about the domains in which were created pages. Here PRIMARY KEY is **domain** as it is suitable to have nodes where one node contains one domain data and in another one - another. To sort our data in partition and to provide uniqueness for domain name and page there was added CLUSTERING KEY that is **page.** This table is used to query for all existing domains for which pages were created and for the number of articles created for a specified domain.
2. **page_users** - contains information about the user that created a page, the page itself and creation time of the article. As a PRIMARY KEY here is used **user_id** as it is suitable to have information about one user in one node. As for the CLUSTERING KEYS, they are **user_name** and **creation_timestamp.** They are used to create a great order of data and easy to get all the pages that were created by specified use and id, name and number of pages created by the user in specific time range.
3. **pages_data** - contains information about the created page, its id and its name. There was chosen PRIMARY KEY as **page_id,** as there is a need to get page name by the **page_id.**

API

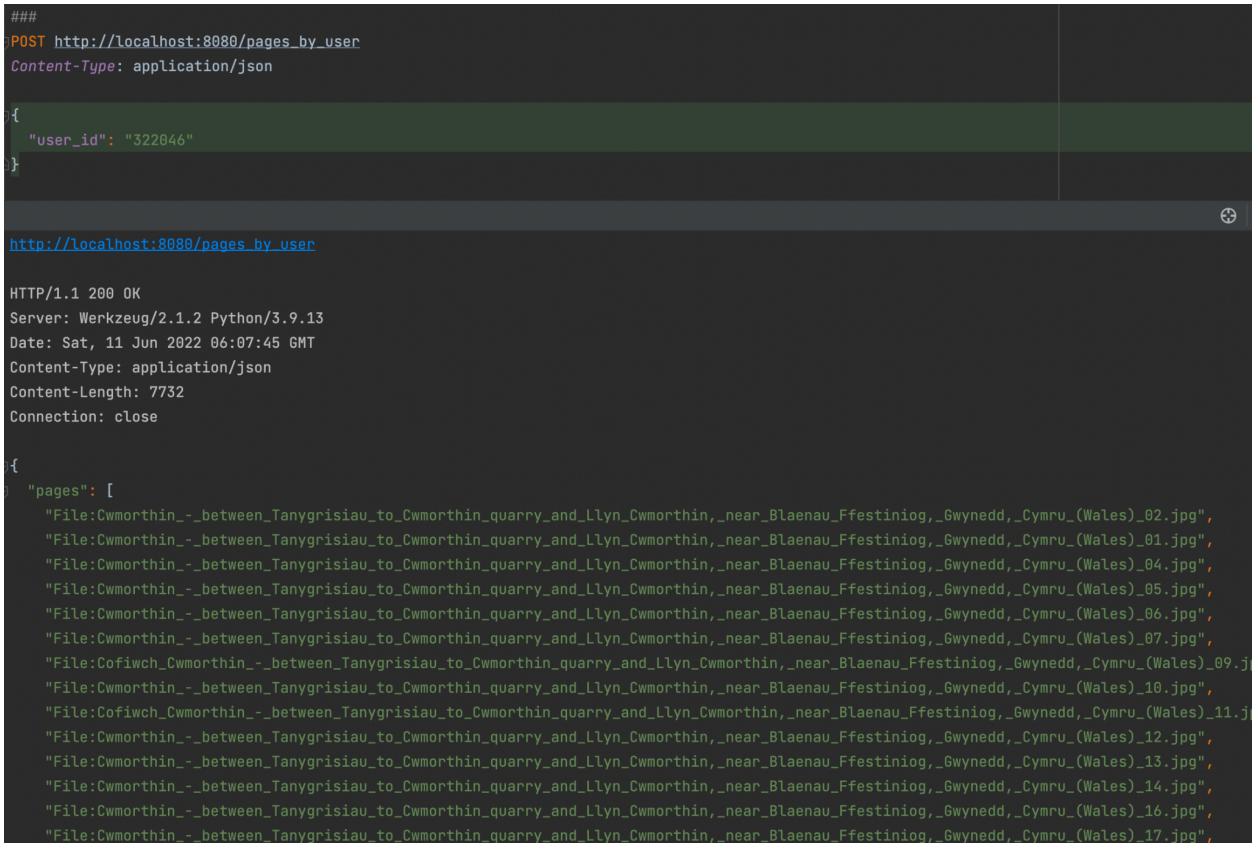
There are 5 types of queries that are now available:

1. Get the list of existing domains for which pages were created - http://localhost:8080/domains_by_pages.

```
###
POST http://localhost:8080/domains_by_pages
Content-Type: application/json

{}
```

```
http://localhost:8080/domains_by_pages

HTTP/1.1 200 OK
Server: Werkzeug/2.1.2 Python/3.9.13
Date: Sat, 11 Jun 2022 06:07:15 GMT
Content-Type: application/json
Content-Length: 1725
Connection: close

{
  "domains": [
    "el.wikipedia.org",
    "fa.wiktionary.org",
    "ht.wikipedia.org",
    "te.wikisource.org",
    "ar.wikipedia.org",
    "id.wikipedia.org",
    "meta.wikimedia.org",
    "or.wikipedia.org",
    "nl.wikipedia.org",
    "ko.wikisource.org",
    "uk.wiktionary.org",
    "www.mediawiki.org",
    "th.wikibooks.org",
    "sv.wiktionary.org",
    "lv.wikipedia.org",
```

2. Get all the pages which were created by the user with a specified user_id - http://localhost:8080/pages_by_user.



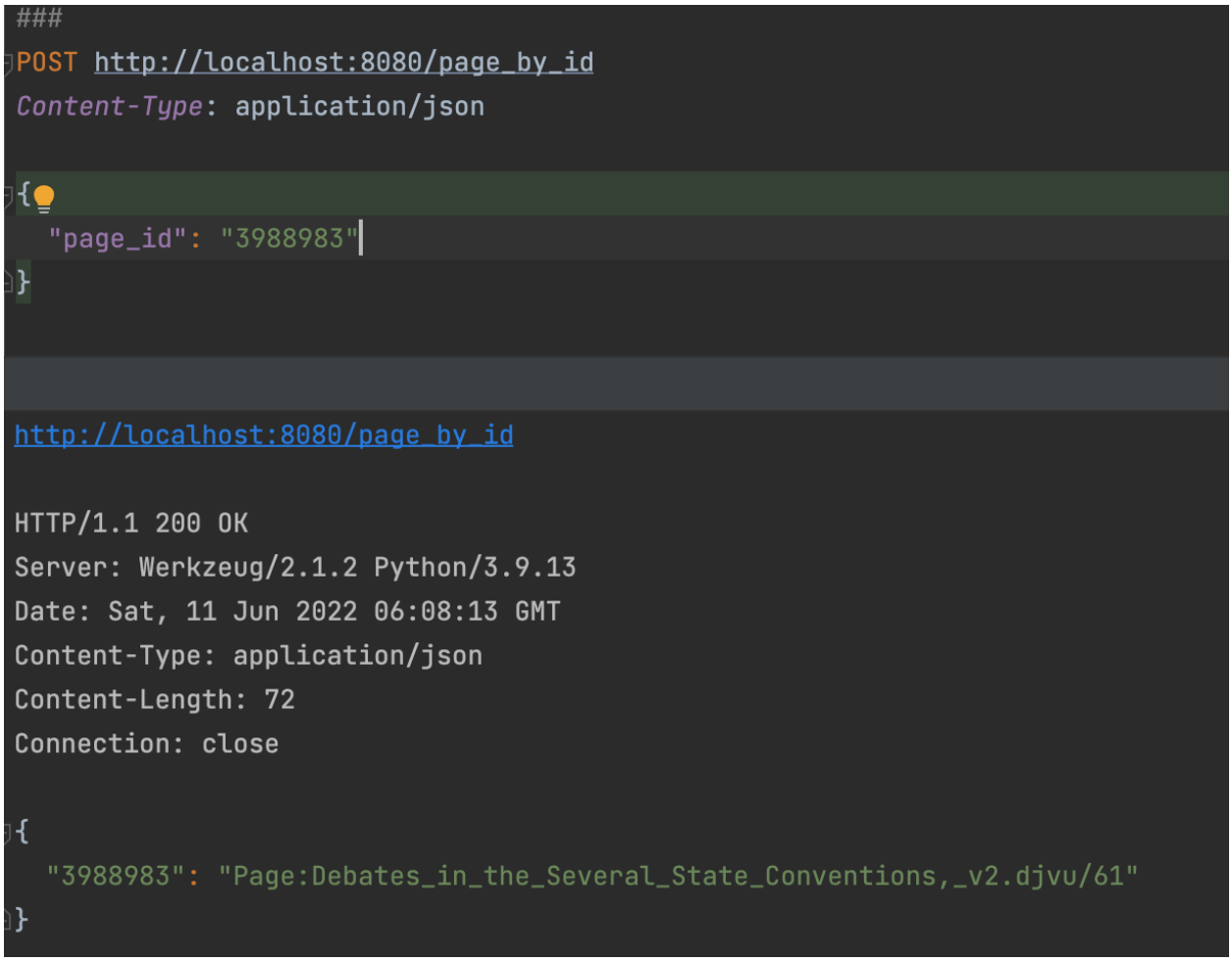
```
###
POST http://localhost:8080/pages_by_user
Content-Type: application/json

{
  "user_id": "322046"
}
```

```
http://localhost:8080/pages_by_user

HTTP/1.1 200 OK
Server: Werkzeug/2.1.2 Python/3.9.13
Date: Sat, 11 Jun 2022 06:07:45 GMT
Content-Type: application/json
Content-Length: 7732
Connection: close

{
  "pages": [
    "File:Cwmorthin_-_between_Tanygrisiau_to_Cwmorthin_quarry_and_Llyn_Cwmorthin,_near_Blaenau_Ffestiniog,_Gwynedd,_Cymru_(Wales)_02.jpg",
    "File:Cwmorthin_-_between_Tanygrisiau_to_Cwmorthin_quarry_and_Llyn_Cwmorthin,_near_Blaenau_Ffestiniog,_Gwynedd,_Cymru_(Wales)_01.jpg",
    "File:Cwmorthin_-_between_Tanygrisiau_to_Cwmorthin_quarry_and_Llyn_Cwmorthin,_near_Blaenau_Ffestiniog,_Gwynedd,_Cymru_(Wales)_04.jpg",
    "File:Cwmorthin_-_between_Tanygrisiau_to_Cwmorthin_quarry_and_Llyn_Cwmorthin,_near_Blaenau_Ffestiniog,_Gwynedd,_Cymru_(Wales)_05.jpg",
    "File:Cwmorthin_-_between_Tanygrisiau_to_Cwmorthin_quarry_and_Llyn_Cwmorthin,_near_Blaenau_Ffestiniog,_Gwynedd,_Cymru_(Wales)_06.jpg",
    "File:Cwmorthin_-_between_Tanygrisiau_to_Cwmorthin_quarry_and_Llyn_Cwmorthin,_near_Blaenau_Ffestiniog,_Gwynedd,_Cymru_(Wales)_07.jpg",
    "File:Cofiwch_Cwmorthin_-_between_Tanygrisiau_to_Cwmorthin_quarry_and_Llyn_Cwmorthin,_near_Blaenau_Ffestiniog,_Gwynedd,_Cymru_(Wales)_09.j
    "File:Cwmorthin_-_between_Tanygrisiau_to_Cwmorthin_quarry_and_Llyn_Cwmorthin,_near_Blaenau_Ffestiniog,_Gwynedd,_Cymru_(Wales)_10.jpg",
    "File:Cofiwch_Cwmorthin_-_between_Tanygrisiau_to_Cwmorthin_quarry_and_Llyn_Cwmorthin,_near_Blaenau_Ffestiniog,_Gwynedd,_Cymru_(Wales)_11.j
    "File:Cwmorthin_-_between_Tanygrisiau_to_Cwmorthin_quarry_and_Llyn_Cwmorthin,_near_Blaenau_Ffestiniog,_Gwynedd,_Cymru_(Wales)_12.jpg",
    "File:Cwmorthin_-_between_Tanygrisiau_to_Cwmorthin_quarry_and_Llyn_Cwmorthin,_near_Blaenau_Ffestiniog,_Gwynedd,_Cymru_(Wales)_13.jpg",
    "File:Cwmorthin_-_between_Tanygrisiau_to_Cwmorthin_quarry_and_Llyn_Cwmorthin,_near_Blaenau_Ffestiniog,_Gwynedd,_Cymru_(Wales)_14.jpg",
    "File:Cwmorthin_-_between_Tanygrisiau_to_Cwmorthin_quarry_and_Llyn_Cwmorthin,_near_Blaenau_Ffestiniog,_Gwynedd,_Cymru_(Wales)_16.jpg",
    "File:Cwmorthin_-_between_Tanygrisiau_to_Cwmorthin_quarry_and_Llyn_Cwmorthin,_near_Blaenau_Ffestiniog,_Gwynedd,_Cymru_(Wales)_17.jpg",
```

3. Get the number of articles created for a specified domain - http://localhost:8080/pages_count_by_domain.

```
POST http://localhost:8080/pages_count_by_domain
Content-Type: application/json

{
  "domain": "commons.wikimedia.org"
}

http://localhost:8080/pages_count_by_domain

HTTP/1.1 200 OK
Server: Werkzeug/2.1.2 Python/3.9.13
Date: Sat, 11 Jun 2022 06:08:02 GMT
Content-Type: application/json
Content-Length: 31
Connection: close

{
  "commons.wikimedia.org": 1018
}
```

4. Get the page with the specified page_id - http://localhost:8080/page_by_id.



```
###
POST http://localhost:8080/page_by_id
Content-Type: application/json

{
  "page_id": "3988983"
}


http://localhost:8080/page_by_id

HTTP/1.1 200 OK
Server: Werkzeug/2.1.2 Python/3.9.13
Date: Sat, 11 Jun 2022 06:08:13 GMT
Content-Type: application/json
Content-Length: 72
Connection: close

{
  "3988983": "Page:Debates_in_the_Several_State_Conventions,_v2.djvu/61"
}
```

5. Get the id, name, and the number of created pages of all the users who created at least one page in a specified time range - http://localhost:8080/users_data.

```
###
POST http://localhost:8080/users_data
Content-Type: application/json


{
  "start_time": "2022-06-10",
  "end_time": "2022-06-12"
}
```

```
http://localhost:8080/users_data

HTTP/1.1 200 OK
Server: Werkzeug/2.1.2 Python/3.9.13
Date: Sat, 11 Jun 2022 06:08:34 GMT
Content-Type: application/json
Content-Length: 16276
Connection: close

{
  "1": [
    "35088",
    "Cabayi",
    1
  ],
  "10": [
    "29077096",
    "Crowsus",
    1
  ],
```