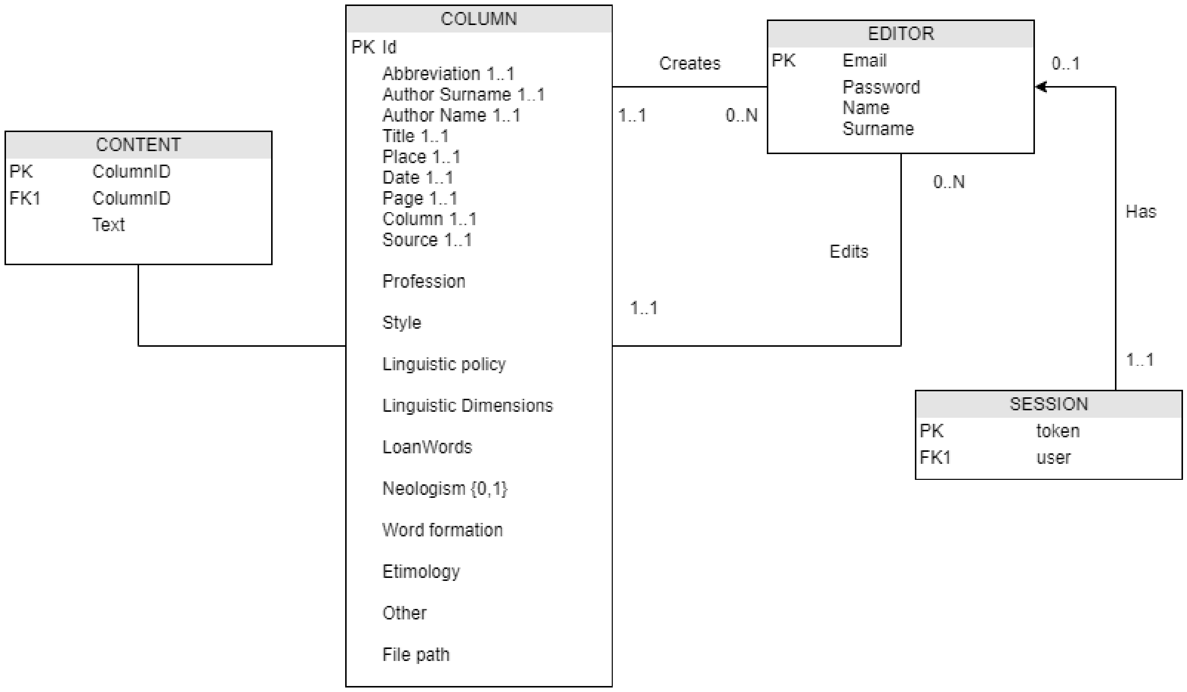
The aim of this report is to explain the components, development planning and overall functionality of the web application for this investigation project.

The web application will be a RESTful service programmed width actual web development languages and tools (HTML 5, CSS 3, JavaScript 5 and PHP 7). This approach to the task will enable us to isolate the tasks that we will perform in the client-side area, such as request sending and handling or dynamic resources management, and in the server-side area, which will be defining later.

We would like to warn that this is not a definitive version of the software and changes will be suffered during the development process.

I. DATABASE STRUCTURE

The database used for this project will follow the following Entity – Relationship diagram:



As we can see, the database will handle all the required data specified in the ChroQué Protocol and will also manage Editor's (users who are allowed to create and update the column's information) sessions in order to maintain security in our application.

The CONTENT entity viability is being evaluated due to performance issues and information retrieval indexing. It's including in the database is not assured at this point of the development.

We are also evaluating the possibility of creating a role of Administrator in order to have some user management and database interactions log.

II. **SERVER - SIDE SOFTWARE TASKS**

The server side area will be programmed in PHP 7.0 language and will handle several HTTP requests.

Depending on the request method and URL, it will perform the following tasks:

GET:

- **/rest/GET/column?param1="value"&param2="value..."**:

  It will make a SELECT statement to the database with the values sent with the url. The result retrieved will be resent to the Client in order to visualize them.

- **/rest/GET/login**

  It will check if the user exists in the database as an editor and will initiate session.

POST:

- **/rest/POST/proccessText**

  It will receive a form with a text (in DOC, DOCX or PDF format) where it will parse automatically the data specified by the ChroQué Protocol. This data will be resent to the user for further confirmation.

- **/rest/POST/createColumn**

  It will receive a form with all the data that has to be created in the database, the server will order to make the entry and will announce the success or failure in the task to the user.

- **/rest/POST/updateColumn**

  It will receive a form with all the data that has to be updated in the database, the server will order to change the entry and will announce the success or failure in the task to the user.

- **/rest/POST/eraseColumn**

  It will receive a order to delete the column of the database. Dependent files will be erased as well.

III. CLIENT - SIDE SOFTWARE

The Client-Side software will be programmed with HTML 5, CSS3 and JavaScript 5.

It will be responsible of sending and managing requests. The technology used for such purpose will be AJAX (Asyncronous JavaScript and XML), which will send all the requests to the server and will be waiting to a response without blocking the code execution.

The communication between the Client Side and Server Side will be with a structured data format: JSON (JavaScript Object Notation). The notation of JSON is the following:

```
{
        Key 1 : "Value 1"
        Key 2 : "Value 2"
}
```

This will allow the client to retrieve efficiently the data from the server and use it to show it to the user.

In terms of user interface design, we will implement a custom Bootstrap 4 framework in order to guarantee a comfortable User Experience.

Following this, we will explain some details of the functioning of several user interfaces in the web application:

Data insertion interface

The data insertion interface will follow two processes.

First, it will be an input where the user can upload or drag and drop a text (.doc, .docx, .pdf extensions are only allowed) to be analysed by the server. Once this task is done, the webpage will dynamically generate a form with the data parsed from the text, in order to check if everything has been done correctly, and the additional fields where the user can introduce more information about the column he is uploading.

Then, the form will be sent to de server in order to insert this data into de database.

Research interface

In order to achieve a good User Experience, we are thinking about profiting JavaScript asyncronous tasks and tools to make dynamic searches, every time the user inserts or changes data in the main form, there will be shown the results that agree on them.

If it results tedious, there will be an option to unable this feature and make the research in a traditional way (the user inserts the data and there has to be clicked a button to trigger the research and retrieve the results.