# PRIMED: A Medicine Search System

### Pedro Simões
up202403063@up.pt
Faculdade de Engenharia da Universidade do Porto
Porto, Portugal

### Miguel Garrido
up202108889@up.pt
Faculdade de Engenharia da Universidade do Porto
Porto, Portugal

### Emanuel Maia
up202107486@up.pt
Faculdade de Engenharia da Universidade do Porto
Porto, Portugal

### Guilherme Martins
up202403106@up.pt
Faculdade de Engenharia da Universidade do Porto
Porto, Portugal

## Abstract

The **PRIMED** project aims to enhance access to structured pharmaceutical data for healthcare and research by collecting and processing information on medicines, diseases, manufacturers, and user reviews. This paper details the latest milestone, which involves data collection from diverse sources such as Kaggle and Wikipedia, followed by a comprehensive data pipeline implemented in `Python`. Key steps in this pipeline include data cleaning, text normalization, and standardization of formats, ensuring the data is structured and easily searchable. The model stores data in a `JSON` format, making it compatible with future integration into larger systems. Preliminary results indicate that the processed data improves accessibility and organization, providing a valuable resource for healthcare professionals and researchers in making informed decisions.

## CCS Concepts

• **Information systems** → Information retrieval query processing.

## Keywords

Medicine, Treatments, Sickness, Pipeline, Data, Gathering, Scraping, Preparation, Search Engine

## 1 Introduction

Nowadays, accurate and comprehensive medicine information plays a vital role in decision-making on the healthcare and investigation sectors. Having access to clean and relevant information is crucial to enhance the efficiency and safety of treatments.

The **PRIMED** project aims to compile and organize such data so that it can be accessed in an easy and organized way. It provides

insights that can be applied both in practical cases or simply in developing new health safety politics.

In this part of the project, data from various sources was collected, processed and analysed with the goal of creating a solid basis for a pharmaceutical system. The whole process is described in this document, from the choice of the theme itself up to the analysis of gathered data and its subsequent classification.

## 2 Theme Selection

The choice to focus on medicines as the central theme for this project stems from the critical role they play in modern healthcare. Medication is the primary tool for treating multiple health problems and conditions, making them an essential component in both public healthcare systems and individual patient care. The data surrounding these substances, such as active substances, applicable cases and clinical trials, offers valuable insights that can improve decision-making in medical practice and pharmaceutics.

## 3 Data Collection

This chapter outlines the data selection process and the methods utilized to gather it.

### 3.1 Selection of Data

One of the main challenges of the healthcare sector is ensuring that accurate and up-to-date information on medication is available to healthcare professionals. Given these needs, the selection of data to be collected was made:

- `Medicines`: The main component of the data, consisting of medicines and their respective relevant, intrinsic information.
- `Diseases`: To complement the collected data, information on some diseases was collected so that it would be possible to get more information to allow for an easier medicine selection process.
- `Pharmaceutical Companies`: Some pharmaceutical companies are more trusted than others due to their credibility and higher quality products, which can impact the decision-making process.
- `Reviews`: It is important to know the success rate of the presented medicine and how the people who use it feel about it.

With this data, the aim is to provide the required information to the users.

## 3.2 Gathering

Finding data suitable for the project proved to be a challenge; not only did it have to be relevant and accurate, but it also had to meet some criteria in terms of quantity and quality. Therefore, the data had to be gathered from multiple sources using different methods - most of the data came from prepared datasets found on Kaggle[1], while the rest came from scraping Wikipedia[2].

*3.2.1 Medicines.* The *Medicines* dataset[3], retrieved from Kaggle, contains a list of pharmaceutical treatments and some relevant, mostly textual, information about the cases where it is used.

The present information on this dataset is:

- `Medicine Name`: The name of the medicine.
- `Composition`: The active substance present in the medicine.
- `Uses`: A list of cases where the medicine is used (specific diseases, for example).
- `Side Effects`: Lists possible side effects resulting from the medicine's usage.
- `Manufacturer`: The name of the company responsible for producing the medicine.
- `Reviews`: Three additional columns containing the percentage of "Excellent", "Average" and "Poor" reviews for each medicine's treatment results.

This dataset, which contains 11824 different medicines, possesses a *CC0 1.0 Universal*[4] license, which means the data is part of the public domain, allowing for the copying and modification of the data.

*3.2.2 Diseases.* This dataset was scraped from the tables of a Wikipedia page containing a list of autoimmune diseases[5]; by gathering this data, the goal was to complement the previous dataset's "Uses" and "Side Effects" columns by collecting more information on this specific subset of diseases.

The information extracted from this dataset consists of:

- `Disease`: The name of the disease.
- `Primary Organ/Body Part Affected`: Information on the organs or body parts affected by the disease.
- `Autoantibodies`: The antibodies associated with each specific disease.
- `Acceptance as an Autoimmune Disease`: Classification for each disease related to its acceptance as an autoimmune condition, based on the current scientific consensus and level of evidence supporting its autoimmune nature.
- `Prevalence Rate (US)`: The percentage of people affected by the disease in the United States of America.

The gathered dataset contains about 110 lines of diseases, being available under the *Creative Commons Attribution-ShareAlike 4.0 International*[6] license, which allows for the sharing and adaptation of the contents.

*3.2.3 Pharmaceutical Companies.* To complement the data on companies present in the *Medicines* dataset, more data on pharmaceutical companies was gathered through another round Wikipedia scraping - this time from a page containing an extensive list of pharmaceutical companies[7]. Approximately 700 companies' names and founding dates were gathered, alongside a short description from each one's Wikipedia article.

The collected information follows this structure:

- `Company Name`: The company's name.
- `Year`: The year the company was created and, if available, when the company was shut down.
- `Description`: A short description of each company, its values and some extra information.

This dataset falls under the same license as the previous *Diseases* dataset (*Creative Commons Attribution-ShareAlike 4.0 International*), as the data was collected in a similar way.

*3.2.4 Reviews.* Lastly, a dataset containing reviews for the collected medication data with more personal descriptions from users[8] was retrieved from UC Irvine's Machine Learning Repository[9].

The data collected had the following structure:

- `Unique ID`: The unique identification of the review.
- `Drug Name`: Name of the drug/treatment.
- `Condition`: The condition of the patient where it was used.
- `Review`: Written review of the experience of taking the drug.
- `Rating`: Number from 0-10 that expresses the quality of the drug.
- `Date`: Date of when the drug was taken.
- `Useful Count`: Similar to a "like" system, this showcases the number of people who found this review useful.

Each review can only be associated with a single condition/disease, which is a limitation of the dataset itself.

This dataset contains around 215000 entries and is covered by the *Attribution 4.0 International*[10] license, which allows for the sharing and adaptation of the datasets for any purpose, provided that the appropriate credit is given.

## 4 Pipeline Description

As mentioned in section 3.2 of this report, **PRIMED**'s data comes from various sources. Due to the often unstructured nature of this data, it is vital to have a streamlined and automated way of normalizing and processing all the information into similar formats, which is the main role of the data pipeline, present in figure [1].

The pipeline consists of Python[11] scripts utilizing libraries such as pandas[12], unidecode[13], html[14], json[15] and csv[16] with the crux of the data processing occurring on the to_json.py script, which converts the CSV files into JSON format.

### 4.1 Elimination of Null Values and Rows

When processing data, another key aspect to consider is that not all data may be correct or even present. For this reason, before doing anything else with the CSV source files, the pipeline uses pandas *dataframes* to check for and remove rows containing only null values, or rows in which key values, such as `Medicine Name`, for instance, aren't present.

### 4.2 Text Normalization

When scraping information from websites, it is important to make sure all the text is normalized. The data pipeline ensures, via Python's unidecode() function, that the dataset only contains ASCII characters. This helps prevent future issues when searching the datasets for information.

data_pipeline.png

**Figure 1: Data Pipeline**

Another character normalization problem resulting from the scraping process arises due to HTML's nature - more specifically, escape codes used to represent special characters. To convert these codes into ASCII characters, the unescape()[14] function from Python's html module, wrapped inside a call to the previously mentioned unidecode() function, to convert the Unicode output into ASCII.

### 4.3 Standardization of Formats

For the best possible result, formats such as dates should be standardized; therefore, part of the pipeline deals with transforming this data into yyyy-mm-dd format, which makes the process of sorting and searching substantially easier.

### 4.4 Data Storage

The final output of the pipeline (the processed data that the search engine will be working with) is stored in JSON format. This decision stems from the requirement of a document-based data storage model, where each JSON object represents a document encapsulating all the relevant fields and values for a particular record - in this case, a specific medicine. This approach provides flexibility in handling unstructured data, since there are no hard constraints in place.

This model also ensures the search engine can query and retrieve data efficiently, based on specific fields within these documents, ensuring that the system can scale as the dataset grows; it also supports complex data relationships and makes it easier to analyse information across different records.

## 5 Conceptual Data Model

Having finalized the data collection and the subsequent transformation and storage processes in the pipeline, a conceptual data model for the combined dataset was developed.
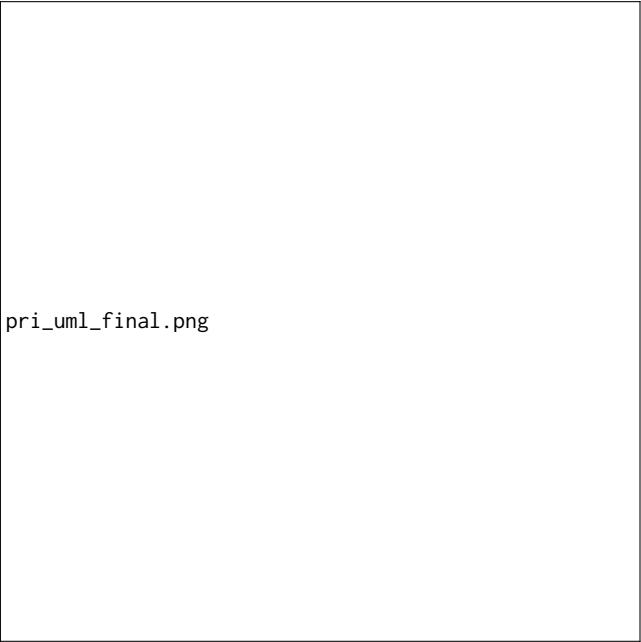
pri_uml_final.png

**Figure 2: Conceptual Data Model**

Each medicine is associated with one and only one pharmaceutical company; a company, however, can produce multiple medicines. A single medicine can be linked to multiple diseases and reviews, though the reverse is true only for diseases - a disease can be associated to many medicines (and many reviews). A review, however, can only be associated with a single medicine and a single disease.

## 6 Dataset Characterization

With regard to the characterization of the dataset, the data resulting from the pipeline was analysed. Graphs, tables and even word clouds, despite not being the most reliable method, were created to make it easier to understand the structure of the data and the patterns that guide the system itself.

### 6.1 General Description

The final version of the dataset, representing the collection of documents, contains 11498 entries. Each entry represents a medicine along with its related information, such as associated diseases and possible side effects, user reviews, ratings and manufacturer details.

### 6.2 Distribution of Drug Manufacturers

By analysing figure [3], which shows the distribution of the main pharmaceutical companies responsible for manufacturing medicines where the five that possess the largest share are highlighted, we are given a clearer understanding of each manufacturers' production capabilities.

## 6.3 Distribution of Reviews

A box plot, present in figure [4], is utilized to analyse the distribution of "Excellent", "Average" and "Poor" review percentages across the different medicines. The graph shows that, for instance, "Excellent" reviews percentages are more evenly spread than the "Average" reviews percentages which, despite having a lower interquartile range[17], possesses a higher number of outliers. The "Poor" review percentages have the largest variations, but also the lowest values.

In figure [5], we can see the average percentage of reviews for the top ten use cases, which supports the findings from figure [4].

## 6.4 Distribution of Side Effects

By analysing figure [6], we can identify the most common side effects caused by medication - nausea, headaches, diarrhea - and how many medicines are associated to each one of them. As expected, the data shows that side effects that occur more frequently are those that are milder in nature.

## 6.5 Common Uses Word Cloud

Figure [7] contains a word cloud with the most common use cases for medicines. "Bacterial Infections" and "Hypertension", for example, are clearly highlighted in the graph, being conditions that are found and treated more frequently.

## 6.6 Most Common Drug Compositions

Figure [8] identifies the the most common drug combinations available in the market, with "Levocetirizine + Montelukast" being the most prevalent, followed by other combinations such as "Luliconazole" and "Domperidone + Rabeprazole".

## 6.7 Years with the Most Companies Foundations

By analysing figure [9], it's possible to observe the years in which the most pharmaceutical companies were founded, with 2003 standing out. This temporal analysis gives us some information about periods of significant growth in the pharmaceutical industry.

## 7 Information Needs

The information needs can vary depending on the person using **PRIMED**. If the individual who is using the tool is, for example, a doctor or a pharmacist, these needs might be related to the compositions of a medicine or cases where it can be applied. If the user is a patient (an average person), they might be more interested in checking which company manufactures the medicine to check if it can be trusted or to assess other user's reviews of that treatment, as well as possible side effects, to have a better understanding of what can happen to them.

Here is a list of possible information needs, explaining what type of data is needed, why and for whom:

- Medicine Compositions: The composition of medication is an important aspect since some might be harmful to the patient, depending on what kind of active substances are present.
- Uses: Not every medicine has clearly defined use cases; some may have a primary use while also being beneficial for other less common conditions.

- Side Effects: The users should be aware of possible side effects resulting from their treatment, so that they can be informed and prepared for potential symptoms or strange events.
- Manufacturer: Some people have a preference for specific manufacturers who they trust more than others, or can simply want to gather more information on the company producing the medicine they were prescribed.
- Reviews: For all possible types of users, existing reviews are always important - whether they are textual or not. These indicate other people's experiences with a particular medicine and can even report some peculiar cases where the patients experienced rare effects from that treatment.
- Diseases' Primary Organs/Autoantibodies: In some cases, experts might need to know what part of the body or organs are most affected by a disease, or what antibodies are linked to it, in order to prescribe the best solution.
- Prevalence Rate: It can also be relevant to be aware of how common a certain disease is in a population, whether that influences the diagnosis of the patient or just for statistical purposes.

Based one these needs, we can answer some questions like:

⇒ Which medicines are more commonly used to treat the common cold?
⇒ Trustworthy companies that provide medicines for Alzheimer's disease.
⇒ How does the composition of medicine for diabetes vary between manufacturers like Novo Nordisk, Eli Lilly, and Sanofi?
⇒ What are the most effective treatments for managing rheumatoid vasculitis pain and inflammation?
⇒ What is the best treatment for persistent migraine symptoms, including nausea and light sensitivity?
⇒ Is weight gain a common side effect of antidepressants?
⇒ What side effects have other patients experienced with medications like rituximab or methotrexate for treating vasculitis?

As previously stated, a wide range of people can take advantage of this type of information - from regular patients to medicine experts and even investigators, everyone can benefit from having access to the data present in **PRIMED**.

## 8 System Overview

Apache Solr[18] is a search and analysis platform which allows the indexation of big data volumes, while also enabling quick and efficient queries on that same data. It works by indexing documents from a collection and then efficiently associating words with the respective documents they appear in. Thanks to this process, the queries performed are faster, since when the user searches for a word, Solr checks the index; Solr also allows the usage of other "tools" via query parameters, such as boosters that enhance the querying process by prioritizing certain terms or fields, or fuzziness to try and mitigate the effects of typos on user queries.

## 8.1 Project Objectives

The primary objective of the project is to give users access to structured information for healthcare and research, focusing on

medicines, associated diseases, user reviews or even manufacturers. The use of Solr means to allow the system to deliver an efficient querying process to the **PRIMED** data collection able to rapidly retrieve the desired results. Additionally, utilizing Solr's capabilities offers the opportunity to enhance the document collection itself by boosting specified fields in search results and creating of custom data types, among other features.

## 8.2 System Architecture

The search engine receives a document collection in JSON format, generated by the data pipeline previously implemented in Python. This collection is then indexed by Solr according to a preconfigured schema[? ]. When a user submits a query to the system, it is sent to Solr via a HTTP request[? ], which returns the results for the query in a JSON file. Figure [10] illustrates the system's overall architecture.

## 9 Document Analysis

In this section, the structure of the documents is analysed, as well as the systems utilized to index the document collection and parse the queries made by the users, which are treated as a variable in each of the aforementioned system's configurations.

## 9.1 Document Characteristics

The final version of a document in the data collection, generated as a result of the previously implemented data pipeline, contains the following fields:

- drug: The name of the medicine in question.
- composition: The active substance(s) present in the drug.
- applicable_diseases: A list of diseases for which the medicine is usually taken or applied.
- diseases_info: A list of information on the medicine's applicable diseases.
- possible_side_effects: A list of possible side effects provoked by the drug when taken.
- excellent_review_perc: Percentage of reviews with a rating superior to 7.
- average_review_perc: Percentage of reviews with a rating between 4 and 7.
- poor_review_perc: Percentage of reviews with a rating inferior to 4.
- reviews_average_rating: The average review score for the medicine, based on all the reviews it received (rounded to two decimal places).
- reviews: Reviews associated with each medicine (in some specific cases, the association is made based on the active principle due to the reviewer not revealing the medicine's name itself).
- manufacturer: The name of the company responsible for producing a specific medicine.
- manufacturer_desc: A short description of each company, containing some extra information.
- manufacturer_start: The year the company was created.
- manufacturer_end: The year the company was shut down, if available.

However, only a few of these document attributes are utilized in the retrieval process; the query is performed solely on the diseases-_info, reviews and manufacturer_desc fields in the index, since these are attributes which primarily contain unstructured text.

## 9.2 Schema Definition

Solr stores details about the field types and which fields it is expected to understand in a schema. A schema allows for a specific search system to be configured on the indexing process. For system and performance evaluation purposes, a decision was made to create two distinct systems.

As a result, two different schemas were created - one for the baseline system, with minimal configuration, and another, more advanced schema for the improved system.

Since every field has the *stored* property set to *true* in both schemas, it won't be displayed in the upcoming (advanced) schema table. Of all the fields present, only two are not indexed (have the *indexed* property set to *false*): manufacturer_start and manufacturer_end, since they are only used to display additional information about a medicine's manufacturer and are not actively used in any search tasks. The only fields with the *multiValued* property enabled are those that store a list of strings: reviews, diseases_info, applicable_diseases and possible_side_effects.

For the basic schema, only native Solr data types were used - no additional field types were created. For this reason, no tokenizers[? ] or filters[? ] were added in the basic schema; however, the text fields utilize the *text_general* type, which tokenizes with a standard tokenizer. The percentage fields were assigned the *pint* type, while the field representing the average rating of the reviews was assigned the *pdouble* type.

For the advanced schema, since the goal is to perform queries on textual field, a new field type, **textBoosted**, was created, based on the *TextField* class, which makes use of both tokenizers and filters - while the tokenizers are responsible for breaking field data into lexical units (tokens), filters examine the stream of tokens and keep them, discard them or transform them depending on the specific filter being used. The **textBoosted** field type includes:

- ⇒ **StandardTokenizerFactory:** Tokenizer that uses whitespaces and punctuation as delimiters to split text into tokens.
- ⇒ **ASCIIFoldingFilterFactory:** Filter that converts Unicode characters like special characters or characters with accents into their ASCII equivalents, if one exists.
- ⇒ **LowerCaseFilterFactory:** Filter that converts all letters in a token to lowercase.
- ⇒ **SnowballPorterFilterFactory:** Filter that applies stemming to tokens during text processing by generating pattern-based word stemmers; while less accurate than table-based stemmers, it is faster and less complex. It was also chosen for its aggressive stemming, making it suitable to normalize word variations in the reviews and manufacturer descriptions.

Two additional field types, similar to **textBoosted**, were also created:

- **diseasesBoosted**: Very similar to **textBoosted**, but utilizes the **EnglishMinimalStemFilterFactory** filter for stemming instead, since it's less powerful and, therefore, more suitable

to handle fields with more technical terms - particularly the diseases' information.

- ***shortText***: Very similar to ***textBoosted*** but without a stemming filter, as it is applied to short textual fields that contain very specific terms.

For each of three field types, the same tokenizer and filters were used in both the query analyser and the index analyser.

The table below provides an overview of the advanced schema's structure, detailing the field names, their respective types and whether they are indexed or multi-valued.

| Field | Type | Indexed | Multi-Valued |
|---|---|---|---|
| drug | shortText | ✓ | ✗ |
| composition | shortText | ✓ | ✗ |
| applicable_diseases | shortText | ✓ | ✓ |
| diseases_info | diseasesBoosted | ✓ | ✓ |
| possible_side_effects | shortText | ✓ | ✓ |
| excellent_review_perc | pint | ✓ | ✗ |
| average_review_perc | pint | ✓ | ✗ |
| poor_review_perc | pint | ✓ | ✗ |
| reviews_average_rating | pdouble | ✓ | ✗ |
| reviews | textBoosted | ✓ | ✓ |
| manufacturer_desc | textBoosted | ✓ | ✗ |
| manufacturer | text_general | ✓ | ✗ |
| manufacturer_start | text_general | ✗ | ✗ |
| manufacturer_end | text_general | ✗ | ✗ |

**Table 1: Advanced Schema**

## 9.3 Retrieval Process

With both schemas implemented, the next step in the system development was to configure the query parsers and their respective parameters. Both systems utilize the Extended DisMax (eDisMax) query parser[? ], which is an improved version of the DisMax query parser[? ].

For the query parameters used by the aforementioned schemas, the following common ones are utilized:

- q: Defines the raw input string for the query, inserted by the user.
- q.op: Applies a logical OR/AND to query operations.
- sort: Specifies the way the documents returned are to be sorted.
- start: Specifies an offset applied to the documents returned from a query being displayed (decides whether Solr should skip over a set number of documents).
- rows: Sets the maximum number of documents to be returned as a result.

Additionally, other parameters focused on optimizing search results are used:

- qf (**Field Boosts**): Represents list of fields that are assigned a particular boost factor, which increases (or decreases) its importance in the query.
- pf (**Phrase Match**): Once the list of matching documents has been identified by the qf parameter, pf can be used

to boost the score of documents in which all of the terms present in the user's query appear in close proximity.

- ps (**Phrase Slop**): Represents the amount of phrase slop[? ] to apply to queries specified with the pf parameter.
- bf (**Independent Boosts**): Used to apply additional boosting to the documents based on certain fields or functions, independent of a document's score calculated by other parameters.

Other techniques, external to the query parameters, were explored via direct application to the user's query within the system. However, due to constraints in the system specifications and query handling, these techniques could not be implemented in the final version of the advanced system:

- **Term Boosts**: A specific set of terms relevant that may be relevant to the user are boosted individually when present in a query.
- **Fuzziness**: Enables approximate matching of terms, which makes it easier to handle typos or misspellings.

The baseline system exclusively uses the common parameters, along with the qf parameter, to process a query and attempt to find matches in the designated query fields (diseases_info, reviews and manufacturer_desc); however, all three fields have the same weight in the ranking process. The system then returns the top 30 ranked documents, sorted in descending order based on their average review rating.

| Parameter | Value |
|---|---|
| q | $query |
| q.op | AND |
| sort | reviews_average_rating desc |
| start | 0 |
| rows | 30 |
| qf | diseases_info reviews manufacturer_desc |

**Table 2: Simple Query Parser**

Alternatively, the advanced system makes use of the all the common and optimization parameters, allowing for theoretically better results to be achieved when processing a query. In this case, the common parameters have the same values as the ones in the baseline system, but the designated query fields, albeit similar in structure to those in the baseline system, have different weights in the ranking process: the reviews field has a weight of 4 and the diseases_info field has a weight of 3, as they are the most important sources of information; manufacturer_desc maintains a weight of 1.

The document receives an additional boost if all of the terms in the query are found in close proximity of each other (exact phrase match) within the reviews field, due to the more informal nature its text. Nevertheless, to account for some variation in word order or the presence of extra words, a phrase slop of 2 is applied to allow for more flexibility when searching for the phrase matches.

Finally, an independent boost is added, based on the `excellent_review_perc` and `poor_review_perc` fields, with values of 1.5 and 0.5, respectively. This adjusts the scoring of each document based on its respective percentage of positive and negative reviews.

| Parameter | Value |
|---|---|
| q | $query |
| q.op | AND |
| sort | reviews_average_rating desc |
| start | 0 |
| rows | 30 |
| qf | diseases_infoˆ3 reviewsˆ4 manufacturer_desc |
| pf | reviewsˆ3 |
| ps | 2 |
| bf | excellent_review_percˆ1.5 poor_review_percˆ0.5 |

**Table 3: Advanced Query Parser**

## 10 System Evaluation

In this section, the information retrieval performance is evaluated, seeking to understand if the querying system is effective and works as expected.

### 10.1 Ground Truth Development

### 10.2 Performance Metrics

*10.2.1 Manual evaluation.* To evaluate the effectiveness of retrieved information, some metrics were established and analyzed. One example are the defined queries to be evaluated. These are derived from information needs and are:

⇒ **Q1**: Companies with treatment for Malaria.
⇒ **Q2**: Reviews from Eli Lilly.
⇒ **Q3**: Weight gain from antidepressants.
⇒ **Q4**: Medicine for migraines.

These queries were tested and evaluated, the evaluation being considerate of the used schema, the simple and advanced.

| Query | Simple results | Advanced results |
|---|---|---|
| Q1 | 1 | 60 |
| Q2 | 1 | 61 |
| Q3 | 1 | 1573 |
| Q4 | 1 | 4842 |

**Table 4: Queries evaluation table**

**Companies with treatment for Malaria**: From the advanced query, 60 results were found, all having the companies who produced it and mostly with more information about it. The simple schema produced the same output.

**Reviews from Eli Lilly**: There were 61 entries found using the advanced schema, where only drugs from Eli Lilly were retrieved and ordered from the best to worst reviews average rating.

**Weight gain from antidepressants**: 1573 results were found in the advanced schema query where both side effects and/or reviews would include weight gain/antidepressants.

**Medicine for migraines**: Trying to find a solution for migraines, 4842 solutions were found on the advanced schema.

*10.2.2 Precision@k.* To evaluate the performance of our data and schemas the P@k metric is used and a K of 10 was defined. This metric selects a number of results and within those, the number of relevant and irrelevant ones are accounted for. In this case, we decided to test this metric on the advanced schema for the third query example (Weight gain from antidepressants). Here, 5 of the results were deemed relevant. The way this was decided was by searching the reviews for people who took antidepressants and the possible side effects for that medication was weight gain. Antidepressants whose side effects did not include weight gain were deemed irrelevant, although weight gain was reported in some other reviews.

$$P@10 = \frac{5}{10} = 0.5$$

*10.2.3 Mean Average Precision.* For this precision metric, the Average Precision for each result is to be calculated and then the mean of every relevant result is taken into account. Here, since there are too many results for each query, only the top 5 for each one were accounted, having for each query the averages

- **Q1**: Here, the R1, R2 and R3 are relevant and R4 and R5 not relevant, having the AP of the first 3 results of 100 percent and the final AC of 1/3 as the last 2 were not deemed relevant.
- **Q2**: Since this result is the return of reviews this AC is 100 percent, taking into account that all results are relevant.
- **Q3**: This AC was almost all calculated in the previous subsection (P@K) with a result of 50 percent, having the
- **Q4**: R2, R3, R4

### 10.3 Performance Analysis

## 11 Conclusions

TODO: Having completed this milestone of the project marks a very important step in the development of our system. The work carried out to this date has made it possible to transform an initial set of diverse, loosely related data into a cohesive and structured basis for **PRIMED**. By carrying out data cleaning operations, we have been able to maintain the quality of the information while simultaneously reducing inconsistencies and duplicates. Outfitted with this data, **PRIMED** will be able to offer users a practical source for consulting medical information; however, we will keep an eye on the dataset in case there is a need for further enhancements during future milestones.

Pedro Simões, Miguel Garrido, Emanuel Maia and Guilherme Martins

## A  Annexes



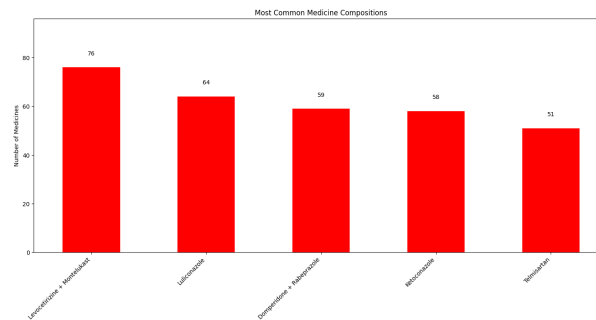Figure 3: Top Drug Manufacturers



Figure 6: Distribution of Side Effects



Figure 4: Distribution of Reviews



Figure 7: Common Uses Word Cloud



Figure 5: Mean of Reviews Percentages by Uses



Figure 8: Most Common Drug Compositions
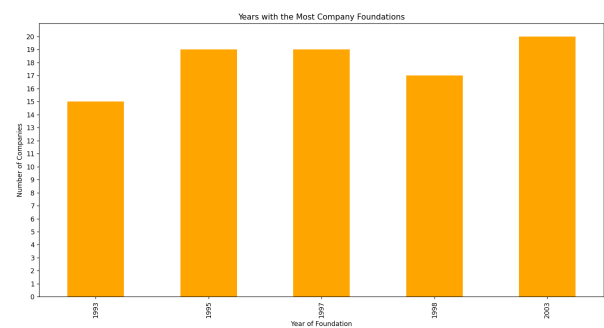
**Figure 9: Years with the Most Companies Foundations**



**Figure 10: System Architecture**

## References

[1] Kaggle. https://www.kaggle.com, 2024.
Accessed on: November 9, 2024.
[2] Wikipedia. https://www.wikipedia.org/, 2024.
Accessed on: November 9, 2024.
[3] Kaggle. 11000 medicine details. https://www.kaggle.com/datasets/singhnavjot2062001/11000-medicine-details, 2024.
Accessed on: November 9, 2024.
[4] Creative Commons. Cc0 1.0 universal. https://creativecommons.org/publicdomain/zero/1.0/.
Accessed on: November 9, 2024.
[5] Wikipedia. https://en.wikipedia.org/wiki/List_of_autoimmune_diseases, 2024.
Accessed on: November 9, 2024.
[6] Creative Commons. Creative commons attribution-sharealike 4.0 international license. https://en.wikipedia.org/wiki/Wikipedia:Text_of_the_Creative_Commons_Attribution-ShareAlike_4.0_International_License.
Accessed on: November 9, 2024.
[7] Wikipedia. https://en.wikipedia.org/wiki/List_of_pharmaceutical_companies, 2024.
Accessed on: November 9, 2024.

[8] Surya Kallumadi and Felix Grer. Drug reviews. https://archive.ics.uci.edu/dataset/462/drug+review+dataset+drugs+com, 2018. UC Irvine Machine Learning Repository.
[9] UC Irvine. https://archive.ics.uci.edu/.
Accessed on: November 9, 2024.
[10] Creative Commons. Attribution 4.0 international. https://creativecommons.org/licenses/by/4.0/legalcode.
Accessed on: November 9, 2024.
[11] Python Software Foundation. *Python Programming Language.* Python Software Foundation, 2023. Version 3.x
Accessed on: November 9, 2024.
[12] The pandas development team. *pandas: Python Data Analysis Library*, 2023. Version 2.x
Accessed on: November 9, 2024.
[13] Takahiko Wada. *unidecode: Text transliteration for Python*, 2023. Version 1.x
Accessed on: November 9, 2024.
[14] Python Software Foundation. *html: HTML and XHTML processing for Python*, 2023. Python Standard Library
Accessed on: November 9, 2024.
[15] Python Software Foundation. *json: JSON encoder and decoder for Python*, 2023. Python Standard Library
Accessed on: November 9, 2024.
[16] Python Software Foundation. *csv: CSV file reading and writing for Python*, 2023. Python Standard Library
Accessed on: November 9, 2024.
[17] Pritha Bhandari. How to find interquartile range (iqr) | calculator & examples. https://www.scribbr.com/statistics/interquartile-range/, 2023.
Accessed on: November 9, 2024.
[18] Sematext. *Getting Started with Apache Solr.* solr introduction and explanation.