INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 06, June 2024, pp: 2079-2085

Impact Factor: 5.725

STREAMLINING ACCESS TO CLINICALTRIALS.GOV API'S RETRIEVAL OF BULK DATA AUTOMATION

Rohith Khanna S[1], Dhanush Chandrasekar[2], Pavan Kalyan T[3], Thanneermalai C[4]

[1,2,3,4]Department of Computer Science and Engineering SRM Institute of Science and Technology, Ramapuram Chennai, India.

DOI: https://www.doi.org/10.58257/IJPREMS35189

ABSTRACT

Data and the information that can be gained from it, have permeated every aspect of human consumption. This significance is most observed within the domain of medicine and medical research, where we notice that the accuracy, reliability, consistency, and accessibility of well-documented information wields profound implications for the outcomes obtained. At the heart of this, we find that well-reviewed, curated, and authenticated data that is relevant is where the hinges of progress hang. The notion holds significance as it also allows sculpting new outcomes from the insights obtained from the various studies, trials, and tribulations leading towards shaping treatment protocols, informing healthcare policy, and guiding scientific trends. However, despite this pivotal role that raw data plays in advancing medical knowledge, accessing very pertinent medical trial results can often prove to be a formidable challenge when there exists a large volume and complexity of data housed within repositories for public use which has the largest collection of clinical trial data, such as ClinicalTrials.gov can overwhelm even the most seasoned researchers, clinicians, and those who partake in interaction and attempting to get results from it. Thus, while the demand for timely and comprehensive data continues to soar, the ability to harness its full potential remains hindered by barriers to accessibility and usability. The primary objective of this paper is to focus on a particular side of data retrieval for those without experience in this field.

Keywords- Clinical Trial Data, API Request Optimization, Bulk Data Retrieval

1. INTRODUCTION

In an era dominated by the apparent omniscience of data, this storage has various systems in place due to the sheer amount obtained in every single moment. Using this requires various data mining methods operated by different fields, in different quantities, and for varying utilities. By blindly diving deep into the raw data available with efficiency and a goal in mind, data mining reveals hidden insights invisible to traditional methods that are missed otherwise. However, a successful data mining operation requires specifics in the form of high quality, relevancy, and organization, forming the foundation that requires careful collection, cleaning, and processing before applying data mining techniques. The power of data mining lies in turning raw data into usable information. It empowers organizations to make informed decisions based on evidence, optimize processes, identify risks and opportunities, and ultimately achieve their goals. This process is exemplified within the medical research community.

Data Mining applications seem to be spanning across industries, from healthcare to finance, marketing, and education. This is observed the most in education [4], as data mining can enhance learning experiences by tailoring one's instruction, identifying the students needing extra support, and improving the teaching methods. It can also be seen that currently, there is minimal use of technology in the landscape of medical studies to support students and educators with the necessary tools, apart from the bare basics that would help in improving the medical understanding of a student for a particular research or projects and sample sizes. one of the major reasons for this observed minimal usage of data mining is the paradoxical complexity of the simple concept, of data validity. This is due to the inaccuracy or false output and difficulty interpreting the results of the studies done every day.

By undertaking the herculean task ourselves, we only delay the inevitable truth that no matter what study tells us today, we plan to proactively identify and support any other projects that could lend us a more efficient outcome, at the same time providing a better understanding of the factors that cause such changes or behavior. But, this veritas is already being proven for the time being on a large data scale by various websites and API (Application Programming Interface) such as ClinicalTrials.gov by extracting meaningful information from raw data and validating the need, then compiling them all into a very in-depth generation of arrays and dictionaries.

Still, it also comes with its detriment because the vast amounts of data are coming over to just around an average of 30,000 characters to 50,000 characters per study. And adding onto the fact that there is an estimate of 494,289 studies existing as of May 9, 2024; there is valid data for every kind of medical study from various resources and organizations that one could use to emphasize the growth in either pharmaceutical or analysis work. By addressing these studies, the groundwork can be laid for the future implementation of the medical approaches [7], ultimately maximizing the

subsequent achievements. However, it is held back by its labyrinthian structure as an API for the uninitiated, posing a threat to anyone being a novice in API request generation for getting the needed data due to this nature. This study proposes a simple approach to answer this problem, utilizing various programming tactics within an even friendlier language such as Python to parse through and request data with either specific parameters, or by getting all the needed data in bulk storage into .csv (Comma Separated Values) files. By delving deeper into these factors, the research seeks to unveil potential interventions for enhancing overall efficiency for the common user aiming to get data buried deep within the labyrinthian structure boasted by the database.

The remainder of this work is organized as follows: Section II reviews the works related to this topic and that which is relevant in short. The proposed approach to the methodology is explained in Section III with a description of the program used along with the modularity of the work done and Section IV has the program's finer structure explained more in detail. Section V provides the results and a discussion of the findings with a retrospective analysis of various factors which might hint at future applications, or enhancing the current methods. Section VI brings the study to a conclusion following the previous sections and the insight gained.

2. RELATED WORK

The paper [1] authored by Danielle G. T. Arts, MSc, Nicolette F. de Keizer, PhD, and Gertjan Scheffer, MD, PhD, this paper explores the critical nature of storing medical data and other roles the quality of data plays within these storage registries. It delves into the understandable rise in the number of registries in medical research and highlights the evident importance of data quality as a major factor in helping find their efficacy and utility going forward. Following an in-depth literature review of numerous research papers, it offers insights into the various aspects found in data quality and suggests strategies for enhancing it within existing registries. Moreover, the authors present a compelling case study to evaluate challenges and opportunities often faced in the real world in practice on improving data quality, thus providing valuable knowledge for beginners and professionals alike. It culminates in a generic framework that shows the optimal method for defining and improving data quality offering a roadmap for future research and implementation.

The paper [2] by Philipp Gemmeke et al. addresses the ever-evolving nature of the healthcare sector where the study addresses the cause for the increase in digitalization in the storage of patient records in recent times. Along with the integration of electronics in the context of patient care, and the subsequent increase in the number of sensors used in tandem, contributing to the staggering rise in the volume of medical data, it also recognizes the potential of such Linked Data and Web APIs in the automation of pre-processing medical images, offering insights automatically into innovative approaches that can be used for handling the data in the digital age without human intervention.

Written by William J. Gordon and Robert S. Rudin, the paper [3] delves into the significance of ' Application Programming Interfaces ' (APIs) in strengthening the existing systems of healthcare and the available data in bulk. The paper attempts to predict the use cases, the hurdles to be wary of, and the innumerable opportunities that can follow the use of APIs, especially in healthcare, helping to organize and parse through data that can be obtained easily and understood the same. This sheds light on the latent potential of APIs in facilitating almost seamless data exchange and utility across the plethora of existing healthcare systems, which in turn paves the way for improved patient care and the respective outcomes.

The paper [5] penned by François Bocquet, Mario Campone, and Marc Cuggia, dives into the different complications, and hurdles following the use and difficulties accompanying the handling of huge volumes of any data that is present in the clinical analysis due to its volatile and iterative nature when organized in bulk or having multiple forms. It also shows the imperative and urgent need to implement comprehensive and well-structured clinical data warehouses in hospitals and healthcare infrastructures, while highlighting any challenges related to data analysis, verification, and relevance that will determine its validity. By examining the complexities lurking in managing vast amounts of this data, they proceed to offer insights into strategies for overcoming hurdles and for optimizing the utilization of clinical data for research and healthcare decision-making, concluding with more suggestions to improve the same.

Authors Elena Pavlenko, Daniel Strech, and Holger Langhof delve into the critical and impervious need in the paper [6] to authenticate the integrity and ethical aspects of sourcing data from numerous places that will be used within clinical research processes. This is a systematic review exploring the procedures that are being employed for checking any paper for either aspect of plagiarism, any unlawful duplication, or determining the validity of their sourced data. This process sheds light on the structure of the best practices and policies for maintaining data integrity optimally not only in clinical data warehouses, but that which could be used for other purposes too.

Published by Kenneth D. Mandl et al., their intensive study in their paper [8] contains a detailed overview of the architecture, structuring, and intricacies of the 'Scalable Collaborative Infrastructure for Learning Healthcare System' (SCILHS). It allows for a review into the primary fund source of SCILHS through the 'Patient-Centred Outcomes

www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

Vol. 04, Issue 06, June 2024, pp: 2079-2085

e-ISSN : 2583-1062

Impact Factor: 5.725

Research Institute' (PCORI) and depicts its framework, which leads to it contributing towards advancing the existing collaborative research infrastructure in the healthcare industry.

A pre-emptive review of existing literature in this niche field reveals a very critical gap within medical research-based APIs – the glaring absence of a robust architecture that effectively returns the needed data at a moment's notice or in large amounts which has been curated and stored already. This paper aims to address this dearth by developing such an architecture-included system that has reliable accuracy as well as provides knowledgeable insight into the minute details stored in the API responses as well within these databases.

3. PROPOSED METHOD

a. Recognizing the API's Structure and Functionality:

Creating a program from scratch that can request data from the ClinicalTrials.gov's API needs an understanding of the documentation, the basis of its documentation, its structure, parameters, and responses to the request that is returned. This is pivotal for gaining clarity on how to organize the API requests and in interpreting the data that is then returned by the API. A JSON request module is chosen for communicating with the ClinicalTrials.gov API due to its universality, flexibility, and ease of use in numerous programming languages as it is standardized within the modules already. JSON (JavaScript Object Notation) is a data format that is widely supported all around and easy to parse due to its array or dictionary-like structure, making it sought after for transmitting data between a client/user and a web server. Having JSON request modules helps in fast and fluent communication with any kind of API, allowing for HTTP requests and responses in JSON format. This is a streamlined approach, facilitating well-modelled and efficient data transmission which also contains a robust and reliable data retrieval system.

b. API Data Structure Identification:

The focused ClinicalTrials.gov API organizes its data in an array manner, holding various nested dictionaries, which is a structure that enhances the organization and accessibility of the data when large amounts of data on a single study are stored at once. Using nested dictionaries allows ordered related data elements to be grouped linearly, which makes it easier to identify specific dictionary values within the array. For example, to identify values that are stored for the name of the disease or condition that has undergone this rigorous study, we have to first identify the name of that which we require. If we require any name or a specific set of names that are within, the following will be the structure that is observed.

{
"**studies**": [
{"**protocolSection**": { "…": {…},
"**conditionsModule**": {
"**conditions**": [
"**Condition Name(s)**", "**…**"]
}
}
"**nextPageToken**": "**Token for next page**"
}

By grouping related data elements within nested dictionaries, the API facilitates efficient querying and filtering based on specifics and different criteria. To make for a sample test case, the stored clinical trials can have data that may include information such as the study titles, the different conditions, the phases of a study conducted, and their subsequent outcomes. By organizing this information within nested dictionaries and having them be in different numeral values of an array, the requests can be modified to easily navigate and fetch specific data elements, such as the trial outcomes or the names assigned to the studies. Also understanding that due to the multiple nested dictionaries and arrays, the structure can be difficult to identify at a glance.

c. Automating the Retrieval Process:

Seeing that the limitations of manual data retrieval include bulk data, the program should prioritize the consistency of the process, ensuring the accuracy of the data retrieved. Singular retrieval of data from the API would be impractical and time-consuming for the machine, the server, and the user, particularly for those with little to no experience with APIs. It should also offer a more efficient way which is also reliable in consistently getting systematic data. This should also apply across a wide range of possible queries. Holding f-strings as the template for the data which can dynamically change, allows for a more efficient manner of looping data inputs, generating more outputs in a single execution. The

produced output should also boast reproducibility from any degree of iterations, resulting in the same if no change in input is observed.

To remove any human error in the inputs that could fault the execution a function should consider any word as its input and ignore sentences, making for simpler criteria to use. A different function needs to convert all special characters into their respective 'American Standard Code for Information Interchange' (ASCII) values to allow for errorless use in the template f-string as well. Going ahead in including a sleep timer for the requests would prove to be a precautionary safety mechanism to avoid timeouts or being tagged as spam requests. This sleep timer must only go off after a set number of valid targets have been achieved, where a valid target is any Uniform Resource Locator (URL) that has an array with readable values within and not just blank, allowing for checking of database values en masse.

4. PROGRAM MODULES

The proposed program is made in the Python language to be more accessible and understandable to the common populace. This program consists of three major modules: Initialization, Organization, and Iteration where each module plays a vital role in the successful execution of the program.

A. Initialization:

The module starts the program with the necessary header files and the needed functions that are required for the program to run efficiently and effectively without any errors or unnecessary runtime issues. This module defines the structural components upon which the rest of the program will hinge. The functions defined include that which clears the ' Comma Separated Values ' (CSV) file upon which the program will implement and print data. This will also help to make sure that every time the program is run, it does not override its data iteratively, ensuring that each run is unique in comparison. This also prevents redundant data from being printed into the file. Other functions include opening a .txt Notepad file which has the feature of entering a large list of data into the document which contains a list of values for which the data will read and automatically parse through printing the values. A function for stripping unnecessary data and replacing them with needed values is requested to run for every iteration or every execution of the program.

Finally, the sample program now holds different variables which will be implemented in f-strings for the program to look at when executing as they can be changed by the user such as the whitelist for the data to be searched for, or the name of the CSV file which will be created or used to save the data.

B. Organization:

This module is imperative in working for a well-structured program as it includes all the initialization of the loops that are going to be used during the execution of the program. There are also variable strings present in this section as they are included to act as quick executory values in the program if you enter the values separated by commas within those strings, only they will be used for implementing the output.

Once all of the setup is completed, the loops begin where it checks a list of conditions where if activated, it would operate differently. This includes the presence of a whitelist to be read through, priority variables that are set, or variables that are set to be skipped over entirely during the runtime of this program. This results in the cumulative counter of phases being reset for every loop which contains a unique value of an 'organizer' and the 'biomarker' which is the unique identification set for any studies. After applying the two values to a template set to check as per the URL template which is used to store data by the API, it checks if the needed values are present or if they are not. If they are present, it tags it as a valid link and the increment to the counter of valid links is done. If not, the URL is changed for another and the loop continues.

C. Iteration:

This part of the program focuses on identifying the different values set to be found within the program and loops through each valid link for these values, adding them onto dynamically changing variables and storing them in the memory. After each successful run, it opens the function of writing into the CSV file and printing the values into it.

When the counter for valid links hits a certain number of values, it goes into a rest phase where there will be no requests to the API itself, avoiding access prevention within a single session. This is done for even small values as due to the nature of the two lists used in executing the sample program, there are large amounts of possible outcomes. This ends up in exponential values for larger datasets therefore it is necessary to implement a safety mechanism beforehand.

The program is linked below in a GitHub repository, where the concept behind the programming is focused on easy understandability to even the least knowledgeable user of the program with all the process descriptions, the needed modules, and instructions on how to use the program are embedded as comments wherever deemed necessary.

The customizable GitHub program is in the below Link: https://github.com/streamliningapi/ClinicalTrials.gov-optimizations.

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 06, June 2024, pp: 2079-2085

Impact Factor: 5.725

5. RESULTS AND DISCUSSION

The experimental methods were implemented in Python, through careful consideration of the user's ease of use and access, the below-displayed figures are of the representative interfaces, the snippets of code used, and sample outputs for certain 'Biomarkers' and 'Companies'. The various schematics of the program will include different arrays one can choose from the plethora of data given within every study, choosing to select particular sets of data or exclude particulars.

This level of customization can also extend to blacklisting certain 'Biomarkers' or 'Organizers' which will exclude them from bulk data collection. The sample output shown below is the culmination of two different inputs and specifics given to the program, every data set found for the condition 'Cancer' and the respective phase data from the arrays for every study done by the various organizations. The other is for every single study done by the company 'Pfizer' and their phase count data, separated into four phases Phase 1 through Phase 4. There also exists a Total counter, responsible for counting the cumulative values for any organization or condition which is useful when retrieving data for a large combined dataset of multiple values for both 'Company' and 'Biomarker' being used.

Due to the modularity of the program in use, we can easily modify the program to print only those studies that have at least a single-phase value within the array, leading to even further optimization of the output .csv file having only valid values that have coherent values that are not just null. This is avoided in this program output to print all possible outcomes. The program can also be modified if you need other values within the JSON responses, by declaring the array type and positions within the program and printing them similarly.

Organization	Condition	Phase 1	Phase 2	Phase 3	Phase 4	Total
Novartis	Cancer	0	1	0	0	1
Veeda Oncology	Cancer	0	1	0	0	1
Betta Pharmaceuticals Co., Ltd.	Cancer	0	1	0	0	1
ImmuneOnco Biopharmaceuticals (Shanghai) Inc.	Cancer	1	0	0	0	1
Dana-Farber Cancer Institute	Cancer	0	0	0	0	0
Spanish Cooperative Group for the Treatment of Digestive Tumours (TTD)	Cancer	0	1	0	0	1
First Affiliated Hospital of Zhejiang University	Cancer	0	0	0	0	0
CureVac	Cancer	1	1	0	0	2
VA Office of Research and Development	Cancer	0	1	0	0	1
University of Texas Southwestern Medical Center	Cancer	0	0	0	0	0
Erasmus Medical Center	Cancer	0	0	0	0	0
Case Comprehensive Cancer Center	Cancer	0	0	0	0	0
SWOG Cancer Research Network	Cancer	0	1	1	0	2
University of Louisville	Cancer	1	1	0	0	2
Herlev and Gentofte Hospital	Cancer	0	0	0	0	0
Children's Oncology Group	Cancer	1	0	0	0	1
U.S. Army Medical Research Acquisition Activity	Cancer	0	0	0	0	0
Henan Cancer Hospital	Cancer	0	0	0	0	0
Tomsk National Research Medical Center of the Russian Academy of Sciences	Cancer	1	0	0	0	1
Federal University of São Paulo	Cancer	0	1	1	0	2
University of California, Davis	Cancer	0	0	0	0	0
Amsterdam UMC, location VUmc	Cancer	1	0	0	0	1
Fudan University	Cancer	0	0	0	0	0
Institute of Hematology & Blood Diseases Hospital, China	Cancer	1	0	0	0	1
Sidney Kimmel Comprehensive Cancer Center at Johns Hopkins	Cancer	0	1	0	0	1
AdventHealth	Cancer	0	0	0	0	0

Fig.1. Condition Search Results (Cancer)

Organization	Condition	Phase 1 Count	Phase 2 Count	Phase 3 Count	Phase 4 Count	Total Phases
Pfizer	Invasive Candidiasis	0	0	1	0	1
Pfizer	Postherpetic Neuralgia	0	1	0	0	1
Pfizer	Advanced Breast Cancer Female	0	0	1	0	1
Pfizer	Healthy Subjects	1	0	0	0	1
Pfizer	Major Depressive Disorder	0	0	1	0	1
Pfizer	Carcinoma, Non-Small-Cell Lung	0	0	1	0	1
Pfizer	Breast Neoplasms	1	0	0	0	1
Pfizer	Pain	0	1	0	0	1
Pfizer	Carcinoma, Non-Small Cell Lung	0	1	0	0	1
Pfizer	Neovascular Age-related Macular Degeneration	0	0	0	0	0
Pfizer	Spondylitis, Ankylosing	0	0	1	0	1
Pfizer	Osteoarthritis	0	0	1	0	1
Pfizer	Multiple Myeloma, Plasma Cell Leukemia	1	0	0	0	1
Pfizer	Invasive Candidiasis	0	0	0	0	0
Pfizer	Candidiasis	0	0	0	0	0
Pfizer	Gastroesophageal Reflux	0	0	1	0	1
Pfizer	Major Depressive Disorder	0	0	0	1	1
Pfizer	Crohn Disease	0	0	0	0	0
Pfizer	Healthy	1	0	0	0	1
Pfizer	SARS-CoV-2 Infection	0	0	0	0	0
Pfizer	Macular Degeneration	0	0	0	0	0

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 06, June 2024, pp: 2079-2085

Impact Factor: 5.725

Pfizer	Advanced, Androgen Receptor Positive Triple Negative Breast Cancer	0	1	0	0	1
Pfizer	Metastatic Pancreatic Ductal Adenocarcinoma	0	1	0	0	1
Pfizer	Rheumatoid Arthritis	0	0	0	0	0
Pfizer	Major Depressive Disorder	1	0	0	0	1
Pfizer	Intra-abdominal Infection	0	0	0	1	1
Pfizer	Methodology Study	1	0	0	0	1
Pfizer	Rheumatoid Arthritis	0	0	0	0	0
Pfizer	Multiple Myeloma	0	0	0	1	1
Pfizer	Healthy Participants	1	0	0	0	1
Pfizer	Tinea Pedis	1	0	0	0	1
Pfizer	Neoplasms	1	0	0	0	1

Fig.2. Organization Search Results (Pfizer)

Going forward on our aim to improve upon existing systems, our aim moving forward is to redefine the code such that the structure can return even larger datasets more efficiently, with a faster execution time as getting only certain data can be easily accomplished. By enabling the program created to read data from a .txt notepad file, we have already achieved increasing in its scalability and made it more adaptable to diverse sources of data. This upgrade will simplify the process of retrieving data, reducing the need for manual input and minimizing potential errors.

Additionally, allowing for the cross-verification of different data within different files to identify already existing values, making them null and void in the current execution can be implemented to improve efficiency in running programs halfway through. Planning to integrate the system's data retrieval capabilities into everyday scripts which can be automated through other means can prove to streamline workflow and eliminate redundancy by only appending new data to existing files instead of running the data from the very beginning.

The possibility of adding a web interface to enlist the same requests from users allows for even faster execution as it removes the tedious task of running a programming application in the background and attempting to run them simultaneously while not knowing if the process has been done successfully. A sample of this web interface/website is depicted in figures at the end of this section.

Moreover, we envision the implementation of parallel processing, which will allow us to compare outputs from different datasets simultaneously instead of different output files. This will enable more sophisticated trend analysis and provide users with deeper insights into clinical trial data which has been collected across different organizations, differing locations, and varying timespans.

Through these processes, this study aims to make the retrieval of clinical study data more accessible, empowering users to extract valuable insights with ease, allowing for not only the common man to find useful but also for the development of more efficient medicines through these studies.

fig.3. Streamlining Clinical Trials Home Page

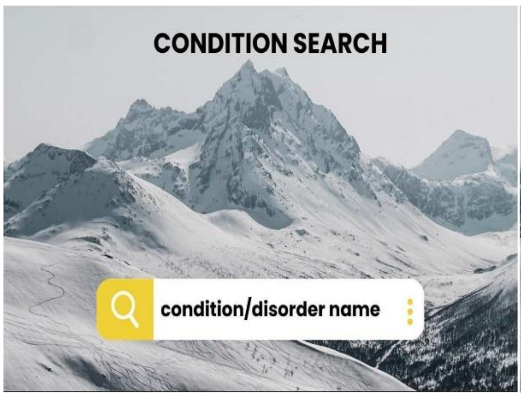

Fig.4. Company Search Page

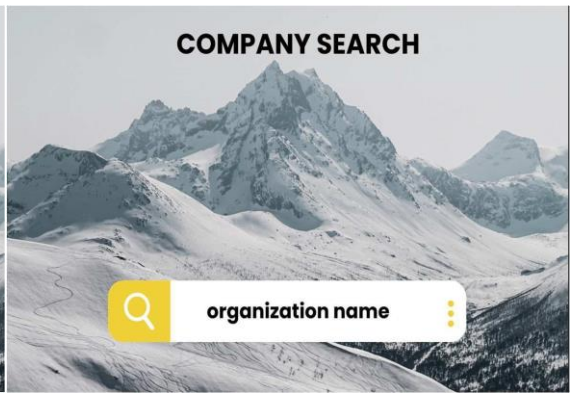

Fig.5. Condition Search Page

6. CONCLUSION

In summary, this study has acknowledged the complex challenges that are associated with accessing and documenting reliable and curated clinical trial data, while also simultaneously proposing accessible methods to surpass these challenges within API data retrieval of databases that have well-structured data. By a meticulous investigation of such a data architecture and by API utilization through JSON request automation, we have managed to allow access to pivotal medical research information to be made easier by those less acquainted with the topic. This includes introducing user-friendly interfaces, an easily accessible programming language, automation methods that are easily understood, and parallel processing capabilities, we have endeavored to empower researchers, clinicians, and stakeholders with the tools requisite for extracting invaluable insights and catalyzing consequential progress in healthcare. Keeping in mind that forward, this research incites potential through the continued refinement and the expansion of its various functionalities present within, a more streamlined, and transparent approach to the utilization of data from clinical trials for other purposes is possible. By allowing ourselves to harness the full potential of clinical trial data readily available at a moment's notice, we can allow for advancements that will promise to shape the future landscape of healthcare for the better.

7. REFERENCES

[1] Danielle G. T. Arts, MSc, Nicolette F. De Keizer, PhD, Gert-Jan Scheffer, MD, PhD, "Defining and improving data quality in medical registries: a literature review, case study, and generic framework", J Am Med Inform Assoc. 2002 Nov-Dec;9(6):600-11, doi: 10.1197/jamia.m1087, PMID: 12386111; PMCID: PMC349377.

[2] Philipp Gemmeke, Maria Maleshkova, Patrick Philipp, Michael Gotz, Christian Weber, Benedikt Kampgen, Marco Nolden, Klaus Maier-Hein, and Achim Rettinger," Using linked data and web APIs for automating the pre-processing of Medical Images", Fifth International Workshop on Consuming Linked Data (COLD2014), January 2014.

[3] William J. Gordon and Robert S. Rudin., "Why APIs? Anticipated value, barriers, and opportunities for standards-based application programming interfaces in healthcare: perspectives of US thought leaders", JAMIA Open, 2022 Apr 6 ;5(2): ooac023, doi: 10.1093/jamiaopen/ooac023, PMID: 35474716; PMCID: PMC9030107.

[4] Zhang, J. Chen, W. Zhang, Q. Xu, and J. Shi, "Education data mining application for predicting students' achievements of Portuguese using ensemble model", Science Journal of Education, vol. 9, no. 2, p. 58, Jan 2021, doi: 10.11648/j.sjedu.20210902.16.

[5] François Bocquet, Mario Campone, and Marc Cuggia, "The challenges of implementing comprehensive clinical data warehouses in hospitals", International Journal of Environmental Research and Public Health. 2022.

[6] Elena Pavlenko, Daniel Strech, and Holger Langhof, "Implementation of data access and use procedures in clinical data warehouses. A systematic review of literature and publicly available policies", BMC Medical Informatics and Decision Making, 2020, Cortez, Paulo, Student Performance. UCI Machine Learning Repository, 2014, https://doi.org/10.24432/C5TG7T.

[7] Lamberti MJ, Kubick W, Awatin J, McCormick J, Carroll J, Getz K, "The use of real-world evidence and data in clinical research and post approval safety studies", Ther Innov Regul Sci. 2018 Nov;52(6):778-783, doi: 10.1177/2168479018764662, Mar 28 2018, PMID: 29714579.

[8] Kenneth D Mandl, Isaac S Kohane, Douglas McFadden, Griffin M Weber, Marc Natter, Joshua Mandel, Sebastian Schneeweiss, Sarah Weiler, Jeffrey G Klann, Jonathan Bickel, William G Adams, Yaorong Ge, Xiaobo Zhou, James Perkins, Keith Marsolo, Elmer Bernstam, John Showalter, Alexander Quarshie, Elizabeth Ofili, George Hripcsak, Shawn N Murphy., "Scalable collaborative infrastructure for a learning", J Am Med Inform Assoc. 2014 Jul-Aug, 21(4):615-20, doi: 10.1136/amiajnl-2014-002727, Epub 2014 May 12, PMID: 24821734; PMCID: PMC4078286.