

**PROBEXPERT: AN ENHANCED Q&A  
PLATFORM FOR REDUCING TIME SPENT ON  
LEARNING AND FINDING ANSWERS  
2021-155**

Ekanayake P.M.D.P

IT18013610

Bachelor of Science Special (Honors) in Information Technology  
Specializing in Software Engineering

Department of Information Technology

Sri Lanka Institute of Information Technology  
Sri Lanka

October 2021

**PROBEXPERT: AN ENHANCED Q&A  
PLATFORM FOR REDUCING TIME SPENT ON  
LEARNING AND FINDING ANSWERS  
2021-155**

Ekanayake P.M.D.P

IT18013610

Bachelor of Science Special (Honors) in Information Technology  
Specializing in Software Engineering

Department of Information Technology

Sri Lanka Institute of Information Technology  
Sri Lanka

October 2021

## DECLARATION

I declare that this is my own work, and this dissertation does not incorporate without acknowledgement any material previously submitted for a degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text. Also, I hereby grant to Sri Lanka Institute of Information Technology, the nonexclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).



Signature:

Date: 2021-10-14

The above candidate has carried out research for the bachelor's degree Dissertation under my supervision.

Signature of the supervisor:

Date:

## ABSTRACT

Developers and other associated individuals must keep up with current technologies due to the quick pace of development in technology-related industries; otherwise, they may not be able to function successfully. Keeping up with new information is no longer a difficult task, thanks to technological advancements. On the internet, there are numerous resources in a variety of media. While there is new information, there is also a lot of outdated and incorrect information. Experts in the field of technology may be able to recognize the new content, but novices may not. As a result, they must keep up with content sharing platforms like Medium and Dev, which are constantly updated. Developers are also increasingly using community forums and question-and-answer platforms to answer questions and share information. The problem with Q&A platforms is that users may have to wait for another user to respond, and the time it takes to receive an answer cannot be predicted. If the question asker's question is time critical, waiting a long period will be ineffective. This study developed an automatic answer creation system to address the issue of having to wait a long time for an answer. Various approaches to machine learning, natural language processing, and web scraping were used to provide such functionality. With that solution, a multi-resource answer can be generated in a matter of seconds.

***keywords:*** *automated answer generation, cosine similarity, keyword extraction, web scraping*

## **ACKNOWLEDGMENT**

I would like to use this section to offer my heartfelt gratitude to all of the people who have helped me along this journey since the beginning. First and foremost, I'd want to express my gratitude to the Sri Lanka Institute of Information Technology (SLIIT) for providing this opportunity to share new ideas through this project. I would also like to take this occasion to express my gratitude to each faculty member and lecturer that assisted me with this research endeavor by providing direction and support.

I want to convey my heartfelt gratitude specially to Ms. Dinuka Wijendra, who agreed to supervise this research throughout the year and offered suggestions to improve the value of the final product. I'd want to express my gratitude to our co-supervisor, Ms. Anjalie Gamage, for her willingness to oversee our project throughout the year.

Finally, I would like to express my thanks to my team members, friends, and family members who have encouraged and supported me in my efforts.

Last but not least, I'd like to express my gratitude to the people submitted their valuable ideas to the surveys and everyone else whose names aren't listed here but who have shown their unwavering support and encouragement in every way conceivable.

## Table of Contents

Declaration.....	i
Abstract.....	ii
Acknowledgment .....	iii
Table of Contents .....	iv
List of Figures.....	vi
List Of Tables .....	viii
Table of Equations.....	viii
List Of Abbreviations .....	ix
1 Introduction .....	1
1.1 Background.....	1
1.2 Research Gap .....	8
1.3 Research Problem.....	11
1.4 Research Objectives.....	13
1.1.1 Main Objectives .....	13
1.1.2 Sub Objectives.....	13
2 Methodology .....	14
2.1 System Overview.....	14
2.2 On Demand Data Scraping.....	18
1.1.3 Google Scraper.....	19
2.2.1. Stackoverflow Scraper.....	21
2.2.2. GitHub Scraper and API.....	23
2.2.3. Other Scrapers.....	25
2.3 Information Similarity Calculation .....	26
1.1.4 Removing Stopwords.....	28
1.1.5 Text Embedding .....	29
1.1.6 Cosine Similarity .....	31
2.4 Automated Answer Generation.....	32
2.5 Keyword Extraction.....	33
2.6 Finding Similar Questions.....	36
2.7 Commercialization.....	39
2.7.1 Premium User tier.....	39
2.7.2 Monthly Subscription for Novice Users .....	39
2.7.3 Advertisements.....	39

2.8	Testing and Implementation .....	40
2.8.1	Testing.....	40
2.8.2	Implementation.....	41
3	Results and discussion.....	45
3.1	Results and findings.....	45
3.2	Discussion.....	48
3.3	Summary of student Contribution.....	49
4	Conclusion.....	50
	References .....	51
	Appendices .....	53
	Appendix A.....	53

## LIST OF FIGURES

Figure 1.1: Usage of internet, when finding an answer .....	1
Figure 1.2: Usage of Q&A platforms to find answers.....	2
Figure 1.3: Resource usage when finding solutions. ....	3
Figure 1.4: Time taken to get an answer from another user.....	4
Figure 1.5: Percentage of people finding complete answer .....	4
Figure 1.6: YouTube video upload statistics per minute .....	5
Figure 1.7: Visitors count of medium.com.....	5
Figure 1.8: Mostly accessed categories of medium.com .....	6
Figure 1.9: Number of resources visited by users to find an answer.....	7
Figure 1.10 : Time spent on internet resources, to find an answer .....	7
Figure 2.1: System overview diagram.....	14
Figure 2.2: Flow of user submitting a question to ProbExpert.....	16
Figure 2.3: Simple flow chart of automated answer generation process.....	17
Figure 2.4: Hierarchy of the web scraping models in ProbExpert platform.....	19
Figure 2.5: Google Scraper flow.....	20
Figure 2.6: regex pattern sample .....	20
Figure 2.7: Stackoverflow accepted answer sample.....	21
Figure 2.8: Sample most voted answer .....	21
Figure 2.9: Stackoverflow scraper flow .....	22
Figure 2.10: GitHub issues of ReactJS codebase.....	24
Figure 2.11: Steps of text similarity calculation .....	27
Figure 2.12: stopwords removing comparison.....	28
Figure 2.13: BERT keywords embedding .....	30
Figure 2.14: Cosine similarity illustration.....	31
Figure 2.15: word cloud of an article about keyword extraction, words extracted by keyword extraction .....	33
Figure 2.16: Flow of finding similar questions.....	36
Figure 2.17: Question structure with sample data.....	37
Figure 2.18: NodeJS backend structure.....	41
Figure 2.19: Frontend Structure.....	42



Figure 2.20: Telegram Bot sample image .....	43
Figure 2.21: ProbExpert Deployments.....	44
Figure 3.1: Time spent to generate 30 automated answers.....	45
Figure 3.2: Similar question finding test execution times.....	47

## LIST OF TABLES

Table 1.1: Gap between ProbExpert and other platforms.....	9
Table 2.1: Keyword Extraction model comparison.....	35
Table 2.2: Model execution time sample.....	35
Table 2.3: Question similarity test data comparison .....	38
Table 3.1: Resource count vs generation time .....	46
Table 3.2: Summary of student contribution .....	49

## TABLE OF EQUATIONS

cosine similarity equation(1).....	31
time consumed equation(2).....	46

## LIST OF ABBREVIATIONS

Q&A	Question and answering
API	Application Programming Interface
REST	Representational state transfer
JS	Java Script
HTML	Hypertext Markup Language
URL	Uniform Resource Locator
regex	Regular Expression
BERT	Bidirectional Encoder Representations from Transformers
ML	Machine Learning
AI	Artificial Intelligence
DL	Deep Learning
NLP	Natural Language Processing

# 1 INTRODUCTION

## 1.1 Background

Software engineering field, or broadly the computer science related fields are evolving day by day, at extraordinary speed. People who work with such areas need to be updated with current technology and other pertinent knowledge to their relevant fields. Otherwise, they may encounter some significant problems with their day-to-day job. To keep up with the current knowledge related to computer science field and find answers for questions, individuals may follow blogs, tutorials, read books, watch videos, refer documentations, work on open-source projects, ask questions from others and follow other individuals in the community. To get information from individuals in computer science related fields, two surveys has been conducted for this study with different questions. According to one of the survey results in Figure 1, 69.6 % of users always use the internet to find answers to their problems, while the remaining 31.4% use the internet to find answers more frequently. The rise of new information and communication technology has significantly changed modern living. Digitalization and interconnected networks, as the primary drivers of this transformation, have resulted in services such as cellular networks and the internet. Because of the increased internet usage, people are more interested in multimedia resources than written ones; what this means is that the internet makes it easier to access more information resources related to the subject.

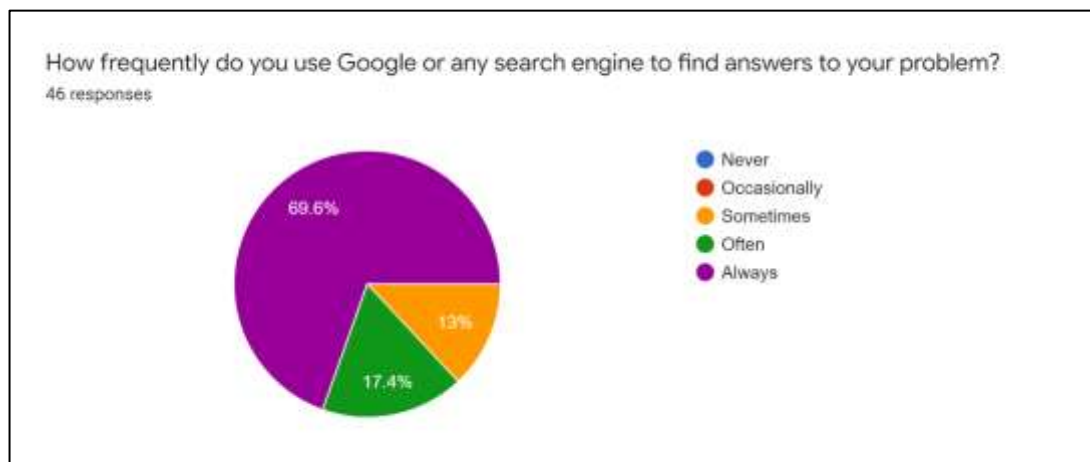


Figure 1.1: Usage of internet, when finding an answer

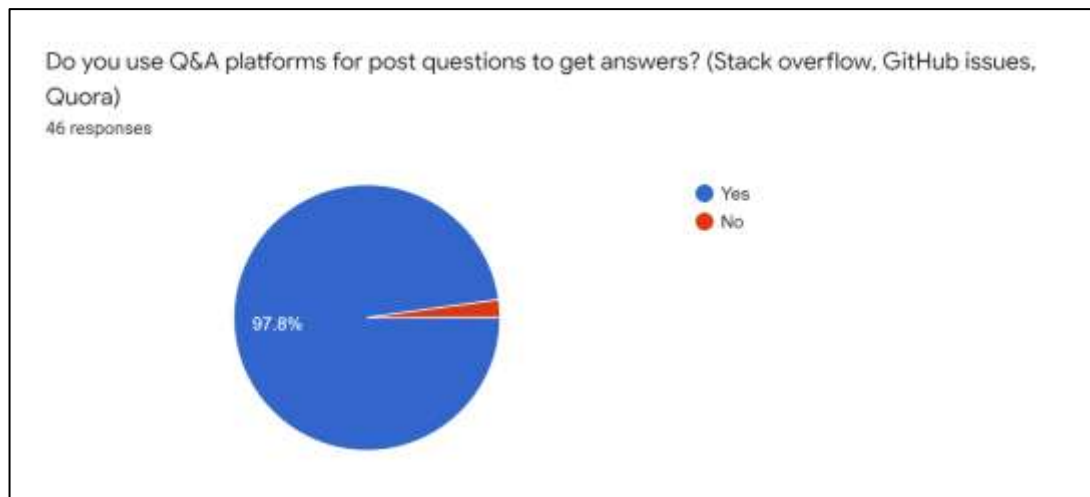


Figure 1.2: Usage of Q&A platforms to find answers

Modern life has been notably shaped by the rise of new information and communication technology. As the main drivers of this shift, digitalization and interconnected networks have led to services such as cellular networks and the internet [1]. Because of the increasing internet usage, individuals are more interested in multimedia resources than written ones, what the above indicates is with the internet, it is simpler to obtain more information resources connected to the subject.

This research paper considers the individuals relevant to the field of programming and programming is a highly important talent and may be a fulfilling profession. In recent years the need for programmers and student interest in programming have risen dramatically and beginning programming courses have become more popular. Learning to program is challenging, nevertheless. Novice programmers suffer from a broad variety of problems and deficiencies. Programming courses are typically considered as challenging, and frequently have the highest dropout rates. It is widely believed that it takes approximately 10 years of experience to convert a beginner into an experienced programmer [2]. They must have declarative and procedural knowledge, memory, comprehension, problem solving, abstraction and logical reasoning skills, among others.

Following up with current technologies will be a bit easy for people who are into the relevant field for quite few years, for novices reading books, blogs or articles may be extremely hard because they do not know which resources to utilize. Beginners may

be end up in a really appalling situations fast, and they may require some help from professionals in that field to solve such difficulties. Then there's the problem of finding true experts in the domain. Community-based forums, Q&A platforms, and mentoring programs are the best and easiest ways to locate experienced people. Finding a mentor will be the most difficult of those possibilities and Q&A platforms will be the easiest. As Figure 1.2 indicates, 97.8% candidates, is using Q&A platforms, where less than 3% is not using it. When finding solutions for questions, 90.2% use Q&A platforms as Figure 1.3 shows.

Q&A platforms and community-driven platforms have a lot in common and those can be advantages as well as disadvantages. As examples of benefits, people can share ideas and resources with people all over the world. As examples of negatives, those with opposing viewpoints will have difficulty interacting with others, and there may be language barriers as well. However, the main issues with Q&A platforms are not receiving correct/complete answers, not receiving answers quickly, and the need for the question asker to rely on another person. According to the survey, 64.4% of people had to wait at least an hour for an answer when using the Q&A platform, whereas only 8.9% of people received an answer within one hour, as shown in Figure 1.4. When it comes to correct/complete answers, most of the time (78.3 percent) people get the expected/correct answer as shown in the Figure 1.5, and none of the survey candidates got a wrong answer from the Q&A platforms at the time the survey was conducted.

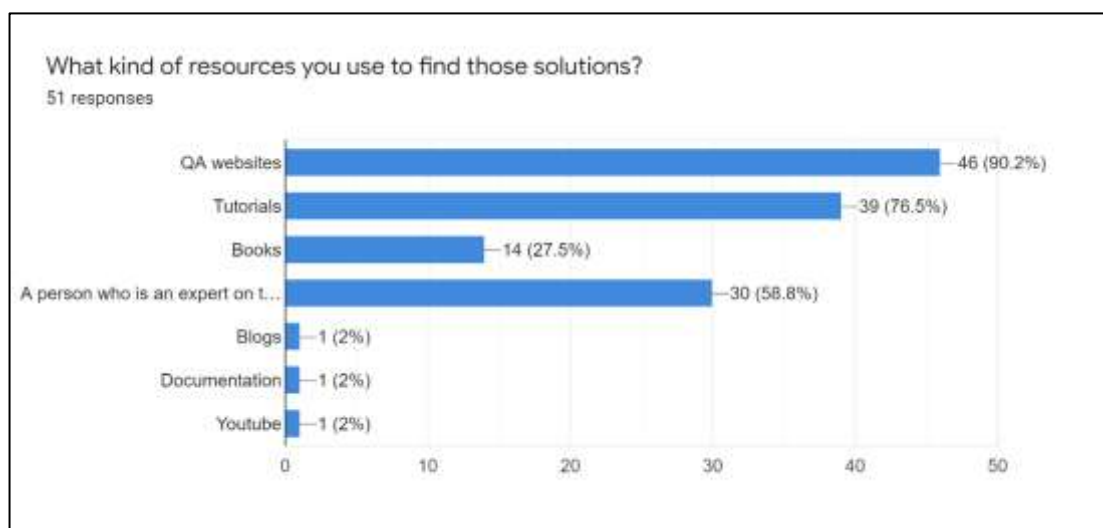


Figure 1.3: Resource usage when finding solutions.

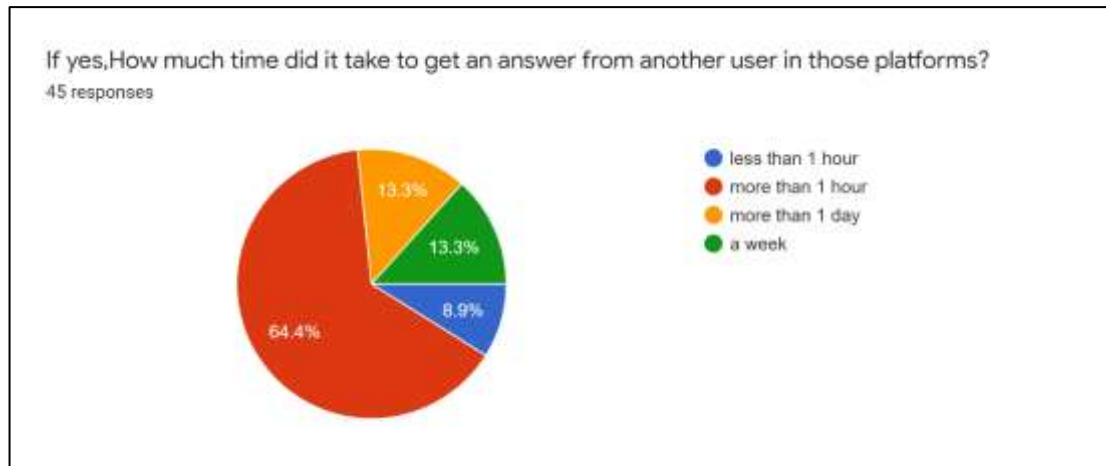


Figure 1.4: Time taken to get an answer from another user.

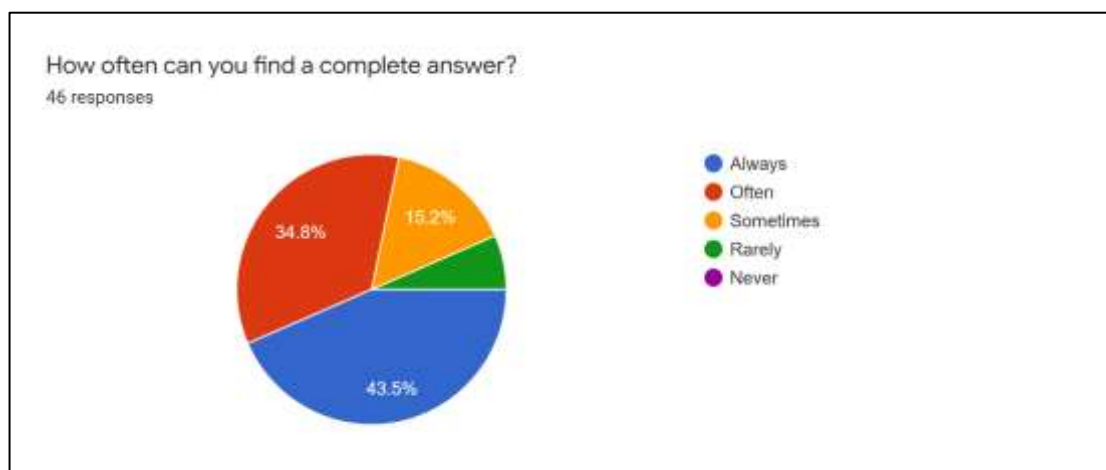


Figure 1.5: Percentage of people finding complete answer

When considering the computer science field, most used Q&A platform is Stackoverflow, Stackoverflow has 16 million registered users, and 22 million questions as of as of October 2021 [3].

Apart from Q&A platforms, individuals refer YouTube videos, blog articles and open-source projects to gain knowledge and find answers. Students belong to the Net Generation, prefer to learn from materials available on YouTube, Tutorials on Web. Also, they like to learn from examples [4]. As of February 2020, 500 hours of video are posted to YouTube per minute, as seen in Figure 1.6. YouTube maintains its reputation by imposing strict regulations and standards on users. In YouTube videos related to computer science on YouTube can be tutorials, theory explanations, coding examples, and a variety of different formats with the purpose of knowledge sharing.

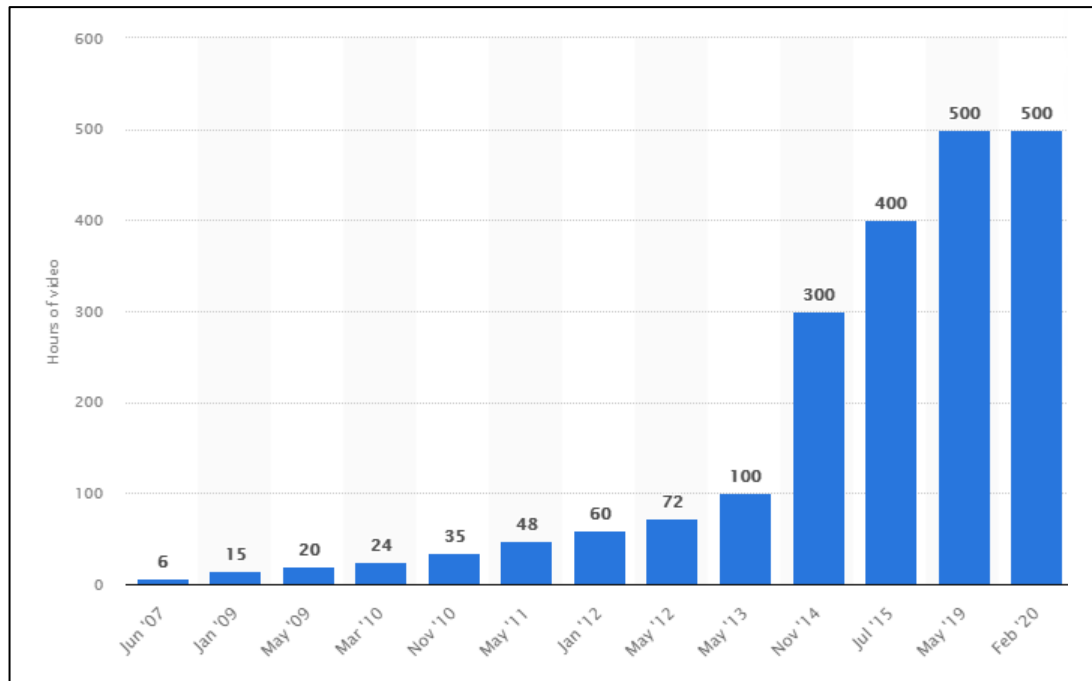


Figure 1.6: YouTube video upload statistics per minute

source: <https://www.statista.com/statistics/259477/hours-of-video-uploaded-to-youtube-every-minute/>

Blog articles, in addition to Q&A platforms and video resources, can be very useful in gaining additional knowledge and solving problems. Medium.com and dev.to will be the go-to blog platforms for computer science professionals among a plethora of blog article sites. According to Figure 1.7, Medium.com had over 165.36 million visits by September 2021. Despite the fact that medium.com has articles on a wide range of topics, the most popular categories are those linked to computer science, as seen in Figure 1.8. Unlike Medium.com, dev.to is dedicated to writings about computer science. Dev.to does not have as many visitors as medium.com, but it does have a respectable number of visitors per day.

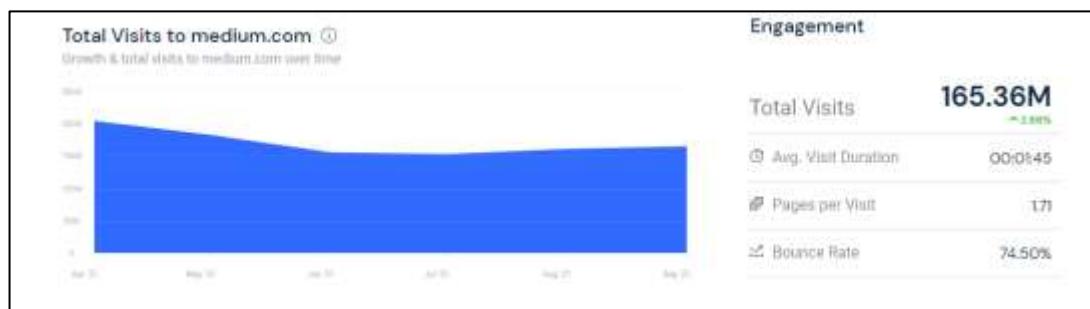


Figure 1.7: Visitors count of medium.com

source: <https://www.similarweb.com/website/medium.com/>





Figure 1.8: Mostly accessed categories of medium.com

source: <https://www.similarweb.com/website/medium.com/>

Individuals who are interested in programming, according to M. Piteira and C. Costa (2013) [4], want to learn from examples and study from other people's codes in addition to acquiring theoretical knowledge from the other resources mentioned above. Contributing to an open-source project is the best way to learn from other people's code; if someone isn't confident enough to participate, they may inspect the codebase and study it. One of the best places to find open-source projects is GitHub. As of January 2020, GitHub reports having over 66 million users [5] and more than 200 million repositories (including at least 44 million public repositories).[6] It is the largest source code host as of April 2020 [7]. Aside from project repositories, People can address their concerns with the help of GitHub issues. In each public code repository in GitHub, registered users can open issues that related to the repository and other people can easily see the open issues and provide relevant information to the issue. In GitHub, an open issue indicates that the issue's end goal has yet to be addressed, whereas a closed issue indicates that the issue has been resolved. Opened issues can be closed by providing replies and, if the issue is related to a new feature or a bugfix, by opening new pull requests.

With all these great resources, there is an issue for users. That is, they have to visit at least 2 to 3 resources to find an answer, and they have to spent at least 20 to 30 minutes visiting multiple web resources according to the survey results shown in Figure 1.9 and Figure 1.10. If the user's question is time-sensitive, wasting 20 to 30 minutes can be a major problem, and visiting numerous websites that do not provide a satisfactory answer to the user's question will typically cost time, network bandwidth, and computer resources.

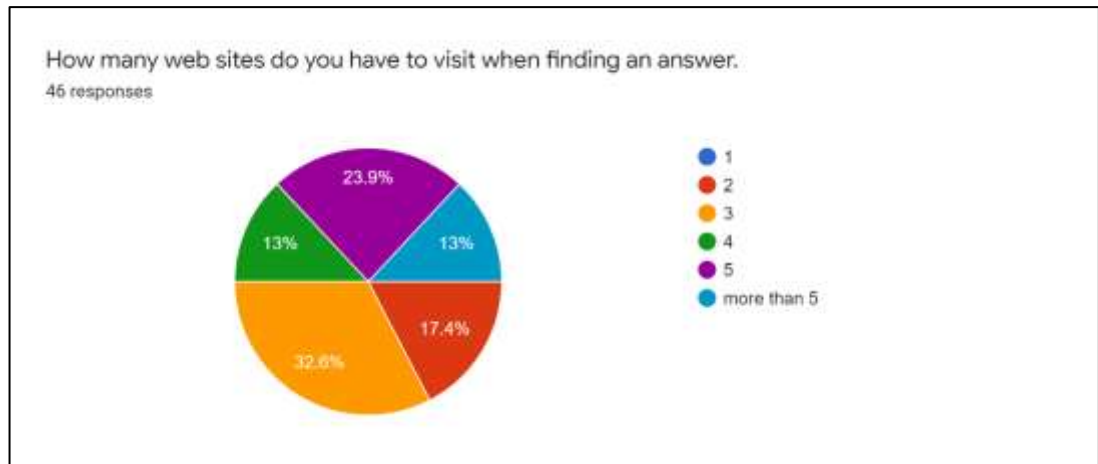


Figure 1.9: Number of resources visited by users to find an answer.

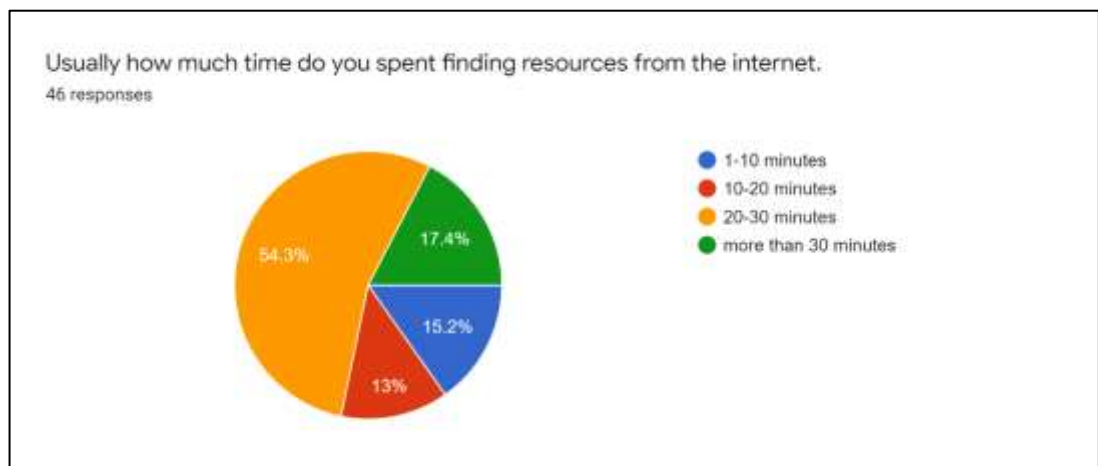


Figure 1.10 : Time spent on internet resources, to find an answer

This research has been focused on a solution that automatically generates answers from a variety of resources including Stackoverflow, GitHub, YouTube, Medium, and Dev.to, in order to reduce the number of multiple web resources that users must visit, and overall time spent on the internet when looking for an answer. To generate the answer automatically, this study used a variety of technologies, both related and unrelated to each other, such as web scraping technologies to scrape data from the internet, Natural Language Processing (NLP) to determine the relevance of the scraped information to the user's question, and several other Machine Learning (ML) and Deep Learning (DL) algorithms/models to aid the answer generation process and find similar questions in the database. The technique of finding similar questions in the database and the process of automatically creating an answer are detailed in depth in the remainder of this paper.

## 1.2 Research Gap

Since building a Q&A platform that automatically generates answers for users' questions and finding similar questions in the database in real-time are two of the main scopes of this research, the research gap can be evaluated from two perspectives: the gap between other related platforms and the gap between similar research work.

When it comes to tech-related Q&A platforms, the key platforms will be Stackoverflow, StackExchange, and Quora as a general platform. As the name indicates all of the above-mentioned platforms let users with account in the platform to post questions and get answers. The following are the distinctions between ProbExpert and the preceding platforms. The key difference between ProbExpert and other Q&A platforms is that none of the other platforms provide an automatic solution to users' questions, and ProbExpert will generate and supply a fully detailed answer with various resources in few seconds after a user posts a question. Various resources inside the automated answer can be in formats of text, images, videos, links, and code snippets.

Another key distinction between ProbExpert and other platforms is that ProbExpert offers a real-time similar question searching tool, which allows users to see similar questions while typing their question in the post-question form. This will make it easier for users to get answers and will also help to reduce data duplication within the platform. The divide between ProbExpert and other platforms in terms of similar question discovery is that while each platform provides some sort of similar question finding mechanism, none of them do it in real time. Other than generating automated answer and finding similar question, there are some other differences between ProbExpert, and other platforms can be seen in Table 1.1. Except for Quora, most platforms support code and provide unrestricted access. Users who are not registered on Quora are unable to look for other questions, but users on all other platforms are permitted to search for and read questions without registering. In other platforms, multiple resources with the answer are dependent on the person who provides the answer, however in ProbExpert automated answers, different resources are used.

Table 1.1: Gap between ProbExpert and other platforms

Platform	Feature				
	Code Support	Unrestricted access	Real-time similar question finding	Answer with multiple resources	Provide answer automatically
Stackoverflow	Yes	Yes	No	Not natively	No
StackExchange	Yes	Yes	No	Not natively	No
Quora	No	No	No	Not natively	No
ProbExpert	Yes	Yes	Yes	Yes	Yes

When defining the gap between technologies used in making of the ProbExpert platform. This study has used many of ML, DL and natural language processing methodologies, such as keyword extraction, semantic text analysis, weighted keyword analyses, and text similarity calculation, to provide an answer automatically and find similar questions. Natural language processing (NLP) is a set of theoretically motivated computer approaches for evaluating and modeling naturally occurring texts at one or more levels of linguistic analysis in order to achieve human-like language processing for a variety of activities and applications.

When existing study work is considered, there is not much with those technologies combined. Each approach has its own research work, but it has not been merged with other technologies. As an example, keyword extraction has its own research work done through TF-IDF, BERT, RAKE, and several other natural language processing approaches. Similarly, other technologies have had research studies conducted using a variety of ways, therefor this research work has analyzed and merged the aforementioned technologies to get the desired result.

Merging several technologies can occasionally cause delays in obtaining desired results; hence, users should use caution when integrating unrelated technologies. If done correctly, expected results can be more accurate than when utilizing a single strategy, and results can arrive faster. In addition to accuracy and execution time, runtime compatibility and resource usage must be considered. Some technologies may be able to run in certain environments while others may not. In that scenario, either the

technology or the runtime environment must be changed. In this study, when combining the technologies, all the above concerns has been addressed. Combined approaches are able to perform without getting any issues regardless of the runtime environments.

Aside from that, the technologies used in data collection and web scraping in this study have been specifically designed for this platform. The majority of earlier research used only one technology, such as BeautifulSoup4 for Python, and scraped data was immediately stored in a database as row data. Previous web scraping methods will not work effectively on modern websites that use JavaScript (JS) to render HTML content, because web scrapers grab the HTML content when the website is first loaded, whereas websites that use JS render their actual content after loading stub files of HTML. That means, the web browser first obtains the basic HTML material from the website, and then, after obtaining the JS functions, the web browser renders the actual content of the website.

In order to address rendering issues and reduce the web scraping time, multiple web scrapers have been built in this study to scrape web material from various web sites, and the majority of web crawlers of this work contain multiple web scraping technologies to boost speed and reduce scraping time. To fulfill the gap of not been able to scrape websites made with JS, in this study, asynchronous web scrapers have been built by combining different technologies using python. Scraped data from web scrapers is filtered before being stored in a database; if the filters filter out all of the scraped data, web scrapers will function again to find more information. Filtering the data will help ML and NLP models to perform better.

Overall, there is a significant gap between when comparing the end result of this study (ProbExpert Platform, answer auto generating and similar question finding in real-time) and the other available Q&A platforms. Furthermore, there is a considerable gap between the ML and NLP approaches as well as other technologies such as web scraping and information filtering used to build the platform, since previously those technologies has been used as it is and in this study most of the technologies are combined together to achieve the end result.

### 1.3 Research Problem

In this study, one main problem has been addressed with few relatively large other problems. The main problem is the need of large waiting time to get an answer from another user when using a Q&A platform and the lack of resources of the received answers. As Figure 1.4 shows, most of the users, 64.4% to be exact had to wait at least an hour to get an answer from another user after submitting a question into Q&A platform. When evaluating the time criticality of the user's problem, one hour can be a long period. In such circumstances, waiting that long will not be beneficial.

Assume the user has an answer to the question, but it is incorrect; if the user is fortunate, someone else may provide a different solution without requiring another hour or a significant amount of time. When presuming that the answer given by the user is valid but not the best answer to the question, it might lead to major problems in the locations where the answer has been applied.

Users may come across past questions related to the one they are about to ask, however the answers of the old question may be out of date or broken, depending on the age of the question. For example, a user may discover a piece of code that meets their requirements and reuse it in their own project. However, the user may be unaware that the APIs used in the code are outdated. Using such outdated APIs may result in software quality issues (for example, using an outdated security framework API). According to the study of Zhang et al. 2019 [8] outdated answers could be categorized into two classes: legacy or invalid [4]. Outdated answer can be considered as a legacy answer if it can still be used or applied, but it may not be recommended anymore since a newer answer might be better or more appropriate. On the other hand, an invalid answer indicates that the outdated answer is not valid or that it no longer works. Users who might have successfully applied the particular answer earlier would now run into errors or complete failures [8]. This issue is also addressed in this study, by filtering out the outdated content when providing the automated answer.

Regarding the lack of resources in user provided answer's, the question asker may not always obtain the perfect solution from Q&A platform since the answer is also coming

from another user; for example, the question asker may expect code examples in the answer, but the answer may just be in theory. Sometimes the theoretical parts may not provide a solution for the user's question.

When getting an answer, some people may be more interested in video resources than text resources, but the answer may not include any video resources. In such circumstances, people must search the theoretical sections on the internet to find their desired resources. Then one of the smaller concerns addressed by this research arises: the necessity to visit several websites to find the required response and spend a significant amount of time doing so. As Figure 1.9 shows, majority of users have to visit at least 3 web sites and as shown in Figure 1.10, the majority of the time accessing several websites takes more than 20 to 30 minutes.

Another modest problem that has been addressed in this study is not having real-time similar question finding mechanism for users to find questions that connected to their question. Without having such functionality duplicate questions on the platform will be comparatively high than having that functionality. When implementing similar question finding functionality, there are several values to taken into account, such how old the question is and, whether or not the question has any answers. Otherwise, there will be no use of recommending that question to the user as user cannot find the expected answer. In the case of old questions, users must need to carefully read the answer to see if it is still relevant, or they may ask the question again as a new one.

As a summary of research problems, this study has primarily addressed the issue of having to wait a long time for an answer from another user when using a Q&A platform, as well as the lack of resources in the received answers, by generating an answer automatically using up-to-date information and multiple types of resources. In addition, this study addressed a few minor issues, such as how to improve web scraping time and how to reduce duplicate questions in the database. In the next sections, technologies, tools, and other resources used to address those problems will be thoroughly explained.

## **1.4 Research Objectives**

### **1.1.1 Main Objectives**

1.1.1.1 Save users time spent on finding answers when using a Q&A platform.

One of the primary objectives of the ProbExpert platform, which is the final result of this study, is to develop an automatic answer generation approach that will save users time spent searching for answers on a Q&A platform. Automated response creation is a great help for users since it saves the user's important time by consolidating answers and references into a single location with more accurate and up-to-date information.

1.1.1.2 Make the automated answer resourceful with different type of resources.

To make the automatic response comprehensible, it required to include a variety of resources such as text, video, photos, and code samples. People who are not comfortable with one resource type can use knowledge from other resources when there are numerous resource types available.

1.1.1.3 Reduce duplicate questions in the platform.

Duplicate questions on the platform will not benefit any user. Regardless, it will make finding the answer a little more difficult for everyone. Answer searchers may need to see all of the duplicate questions to get an answer, and users who can supply an answer may be puzzled by duplicate questions. With the automated answer creation approach in ProbExpert, each question will generate an automated answer, which will consume more resources than finding similar questions in the database, if any exists. By applying duplicate question finding methods, more resources will be saved in addition to reducing duplicate questions.

### **1.1.2 Sub Objectives**

1.1.2.1 Scrape up-to-date information from the internet in a quick manner.

1.1.2.2 Find relevancy of scraped information to the user's question.



## 2 METHODOLOGY

### 2.1 System Overview

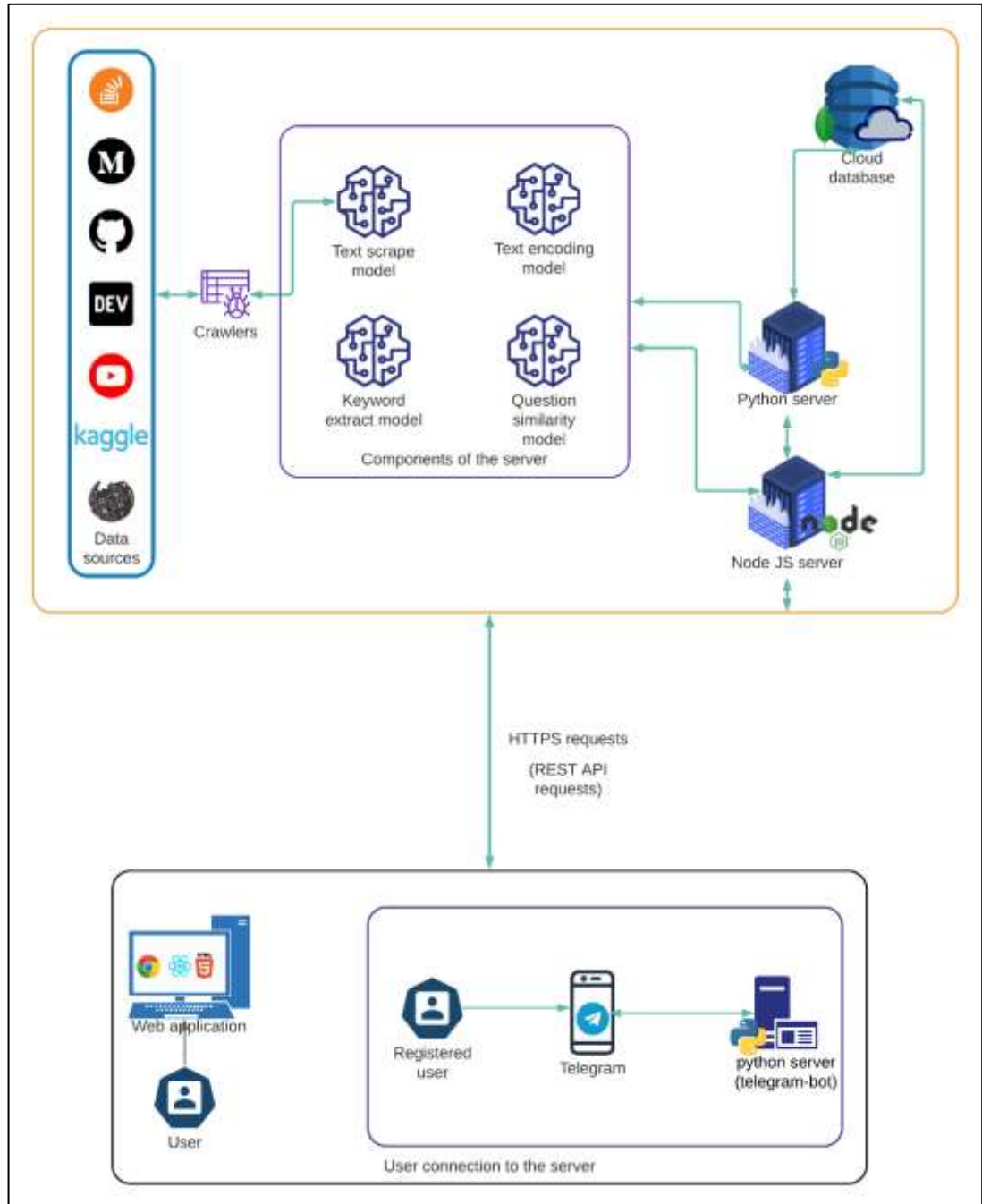


Figure 2.1: System overview diagram

In this section, overall system overview of automatic answer generation, real-time similar question finding, and web scraping technologies will be explained. Overall

system diagram is stated in Figure 2.1. To archive end result, ProbExpert platform contains two backend servers, one node JS and one python server, one cloud hosted Mongodb database, and to archive the end goal, there are four Python models: one for scraping information from the internet, one for extracting keywords from the retrieved data and users' questions, one for encoding text data, and one for calculating the similarity of questions with other questions or scraped information. End users can access the system using the web app or using the telegram bot.

Flow of user submitting a question to the ProbExpert platform has been demonstrated in Figure 2.2. The web app or Telegram bot can be used to do login and all other user-related inputs. Once the user begins typing the question in the web app, the ProbExpert platform will send the written information to the backend through REST API request. If the Python model finds a similar question to the typed question, it will include it in the response to the API call. Then, on the same screen, those similar questions will be displayed; if the user finds one of the suggested questions to be similar to the one, he/she is about to ask, the user can open the question and exit the question entering screen. If user does not open any suggested questions, this process will be repeated until, user submit the question.

After user submit the question into the platform, REST API request will be made to the NodeJS server. Then NodeJS server will perform two tasks, first it will save the question in the MongoDB database, and then it will trigger the automatic answer generation model. If everything went successful server will response as with a success message and user will be able to open the newly asked question in the ProbExpert platform.

If the user opens the question immediately after receiving a success response, the question title, body, and tags, as well as the loading indication, will be displayed in the automated answer area, as the answer generating process will take 3 to 10 seconds. When a user opens a question and the automatic answer is still being generated, the user does not need to refresh the screen to see whether there are any modifications. The platform will check the automatic answer creation status and display the answer if it has been generated.

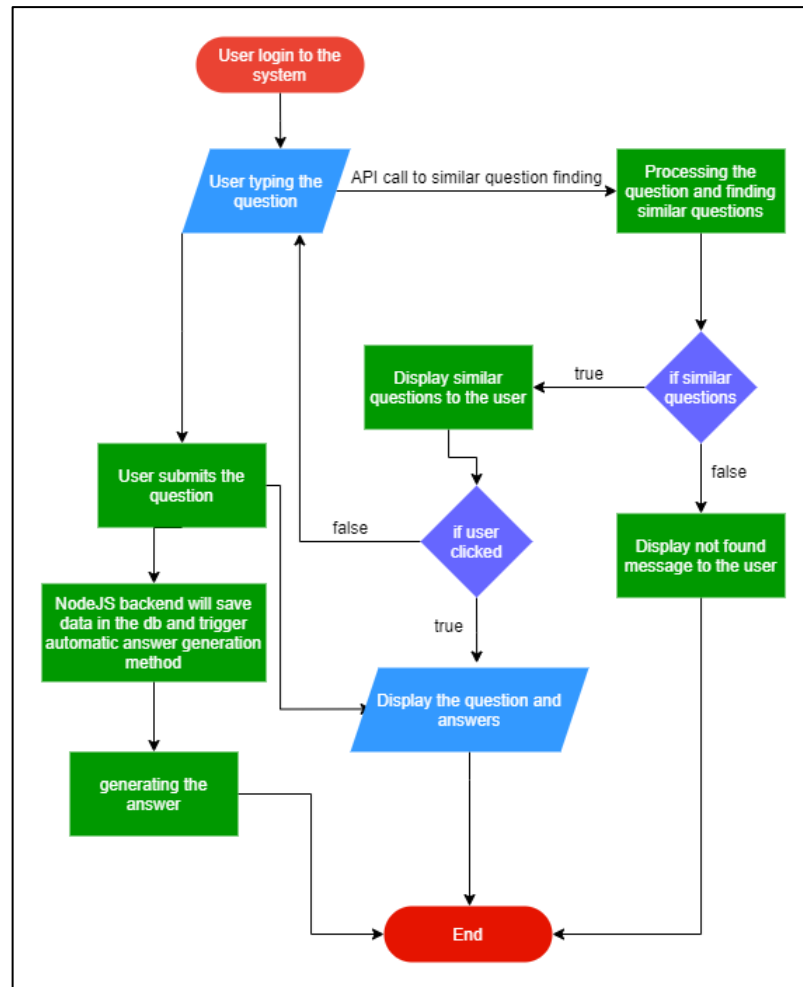


Figure 2.2: Flow of user submitting a question to ProbExpert

Flow chart of Figure 2.3 illustrates the complicated process of automated answer generation in simple phases. After the NodeJS server triggers the automated answer generation model with question data, the model will initiate a data scraping process using the data scraping model with initial data of the question, and the data scraping model will scrape required data from the resource web sites. Once the automated answer generating model has collected enough data, it will calculate the information's relevance to the user's question. If the scraped data has more than 80% resemblance, the automated answer will be generated. Otherwise, the scraping operation will be restarted with additional data from the user's question. The data scraping and relevancy calculation methods are complex procedures, which will be addressed within the next sections of this study.

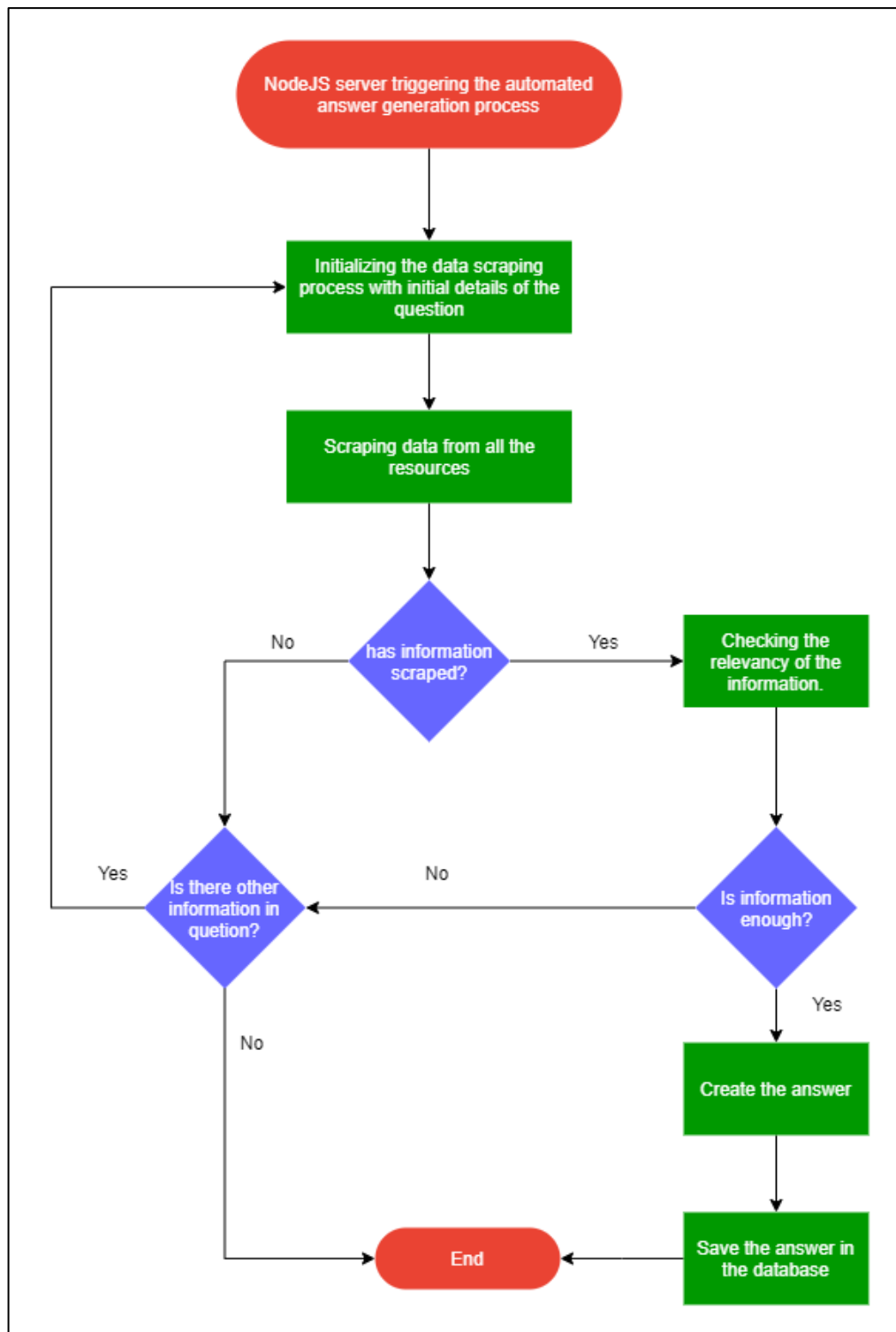


Figure 2.3: Simple flow chart of automated answer generation process

Models, logics, and other details of the system will be explained in this section and implementation and testing details of above technologies will be explained in forthcoming sections.

## 2.2 On Demand Data Scraping

In this section, all the sections related to data scraping, will be discussed. To generate an answer automatically, it needs up-to-date information related to the question. Since ProbExpert automated answer contains information from multiple variances, one scraper will not be enough. Therefore web scraping model contains few different web scrapers built with python.

In general, technologies and packages such as BeautifulSoup4, Scrapy, requests, LXML, Async-io, html-session, JSON, and headless browsing were used to build the scrapers. All of the above may not be required to scrape web sites that are not rendered using JS, because JS-based web sites do not render as soon as HTML content is loaded into the browser; asynchronous rendering is required. Async-io is used to enable asynchronous features in Python, and html-session, headless browser features, are used to render HTML content in an asynchronous manner using JS.

The resource web sites that will be scraped are, Google, Stackoverflow, YouTube, Medium, Dev and GitHub. As illustrated in the Figure 2.4 most of the resource web sites except YouTube and some parts of GitHub, scraping starts with a Google search result scraping. The reason for a google search instead of native platforms search for Stackoverflow, Medium and Dev is that search results can be filtered easily with the google search.

Google is the largest, and most used search engine as of September 2021, and it supports variety of techniques to filter search contents. Also, google search considers variety of factors before listing a website in the search results, According to Ziakis et al. 2019 [9] there are more than ten factors including keywords, age of the website, SSL, that will be considered by google to list a website in the search engine. On the other hand in the native search functionalities of some websites will not be as effective as google search. With that reason there is one significantly valuable feature in google search engine, and that is custom search filters functionality. Users can use symbols or words in the search to make the search results more precise. Search results can be filtered by date, site and other factors. [10].

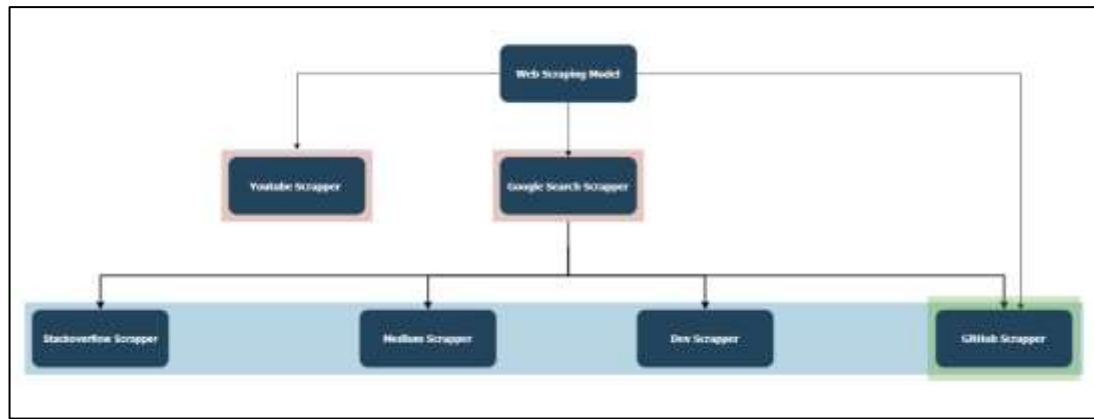


Figure 2.4: Hierarchy of the web scraping models in ProbExpert platform

Because YouTube's search engine is as good as Google's search engine, YouTube search is the world 2<sup>nd</sup> largest search engine, and YouTube is also a sub product of the Google company as of September 2021, a Google search is not required to find YouTube resources. For the GitHub resources, a scraper with GitHub and Google search was utilized because some GitHub features can be found more effectively with a Google search than a GitHub search. GitHub also provides an API for accessing data, and the API has been used in some portions of the GitHub scraper.

### 1.1.3 Google Scraper

As Figure 2.4 illustrated, before starting Medium, Dev and Stackoverflow scrapers, google search scraper has been performed to find up-to-date content of each web resource. To filter content from web site to web site, specific filter from google search engine has been used. To find up-to-date information, date limit has been set to two years of time period. With that filter, google search engine will list links to most relevant resources from desired resource sites. Flow of google search engine scraping has been illustrated in Figure 2.5.

All of the links listed by Google Search Engine are not direct links to other websites; instead, those links contain some information that may be useful for Google to track the listing's information. Those irrelevant parameters have to be eliminated from the URLs in our scenario. Otherwise, scrapers of other resource sites will not be able to function as expected.

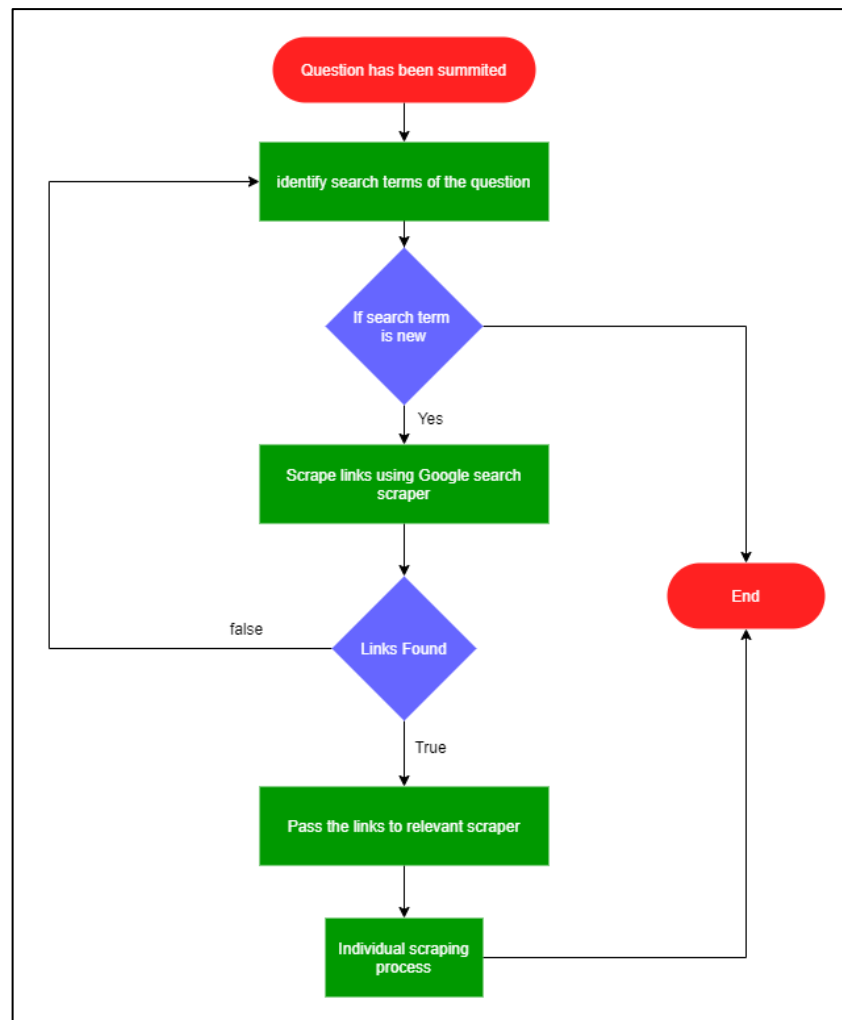


Figure 2.5: Google Scrapper flow

To remove unnecessary parameters from listed URLs, regular expression (regex) functionality has been used. A regex is a character sequence that describes a search pattern. String-searching algorithms typically use such patterns for "find" or "find and replace" operations on strings. Because of all the links scraped from google search scraper follows similar pattern as shown in Figure 2.6, regex was a perfect fit to find and remove the unwanted parameters from URLs. After refining the URLs received by google scraper, those links will be passed to relevant resource scrapers.

```

def get_clean_url(url):
    import re

    pattern = r"^.*?\&sa="
    return re.match(pattern, url).group(0)
  
```

Figure 2.6: regex pattern sample

### 2.2.1. Stackoverflow Scraper

After getting cleaned Stackoverflow URLs from Google scraper, Stackoverflow scraper will start scrape the Stackoverflow posts. First Stackoverflow scraper will try to find an accepted answer. Accepted answer means that, the answer has been accepted by question asker as the correct answer.

Reason for targeting an accepted answer is there is a more probability of working than a normal answer. When visiting Stackoverflow, accepted answer can be identified with the green checkmark under the voting buttons as shown in Figure 2.7.

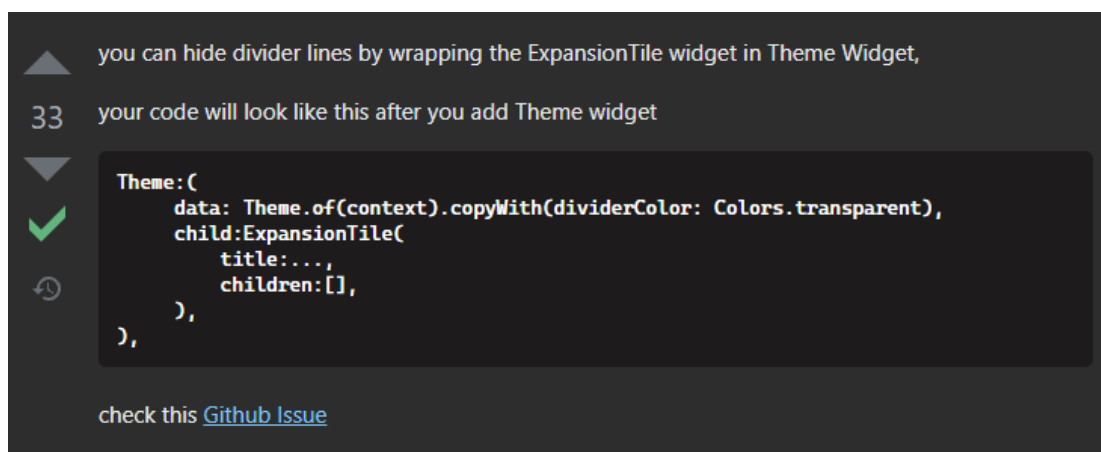


Figure 2.7: Stackoverflow accepted answer sample

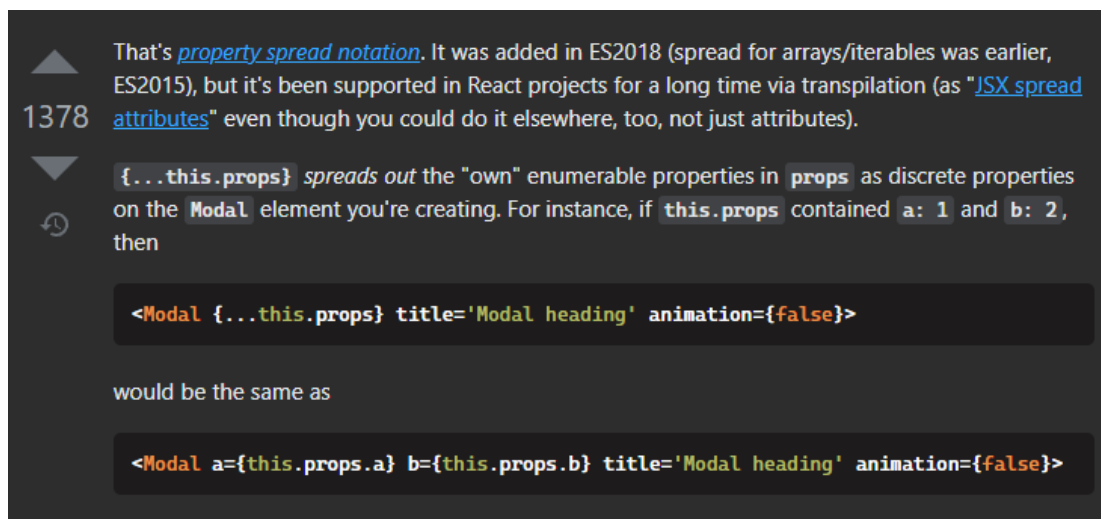


Figure 2.8: Sample most voted answer



If the scraper found an accepted answer from scraped page, it will filter out the content of accepted answer as HTML content. Although if there is no accepted answer, there can be a working answer among the answers, since the question asker may forget to mark the correct answer as the accepted one although the answer has large number of upvotes as illustrated in Figure 2.8, that usually means it's a working answer.

As Stackoverflow answer list goes, next answer in the top of the list will be the most voted one. When scraper cannot find an accepted answer, it will try to scrape the most voted answer. Scraping process will be similar to the accepted answer scraping, it will scrape all the HTML content of the answer. Flow of scraping Stackoverflow answer is shown in Figure 2.9.

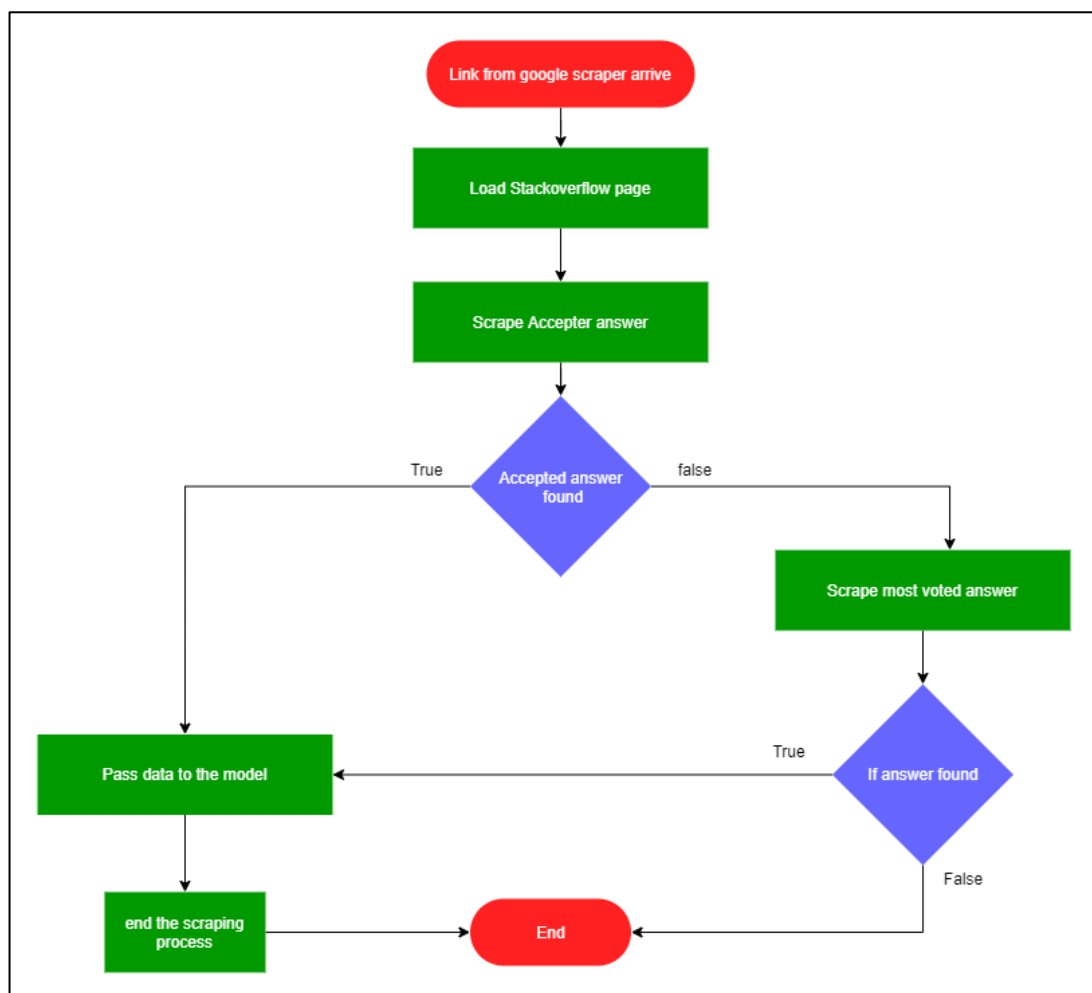


Figure 2.9: Stackoverflow scraper flow

Some parts of the Stackoverflow answer are being loaded by JS, because of that reason asynchronous scraping methods should be used, otherwise, code snippets from Stackoverflow answer may not be able scraped. If there is no answer in each link received by Google scraper, that means no question has get answered, although this scenario is extremely uncommon, sometimes it may occur. In that situation automated answer will not be able to contain an Stackoverflow answer.

Once Stackoverflow scraper has successfully find an answer, and successfully scraped answer's HTML content, it will pass the information to the next step of the automated answer generation process, which is the process of information similarity calculation. In upcoming sections, information similarity calculation will be explained.

### **2.2.2. GitHub Scraper and API**

GitHub is a service that hosts Git repositories, but it also adds many of its own features. While Git is a command-line tool, GitHub has a graphical user interface that can be accessed through the web. It also includes access control and a variety of collaboration tools, such as wikis and basic task management tools such as issue tracking, for each project.

For the automated answer, resources from GitHub will be project repositories and project issues that has been created by another users. Since GitHub contains at-least 44 million public repositories [6] and large number of issues on those repositories, more issues of a repository means that codebase is used by many, issues of GitHub can be seen as Figure 2.10, adding relevant resources from GitHub into the automated answer will be valuable.

People who are interested in contributing into open-source projects, will also be interested in GitHub. Users are free to watch, download or Fork – which is the term used by GitHub to copy someone else codebase into another user's account. Then with other functionalities such as Pull Request, users are able to contribute to the public repositories. Contributing to open-source projects may help individuals to get better understanding about coding practices, standards and other community related things.

To consume the GitHub resources, it has released a dedicated API. Almost all the resources from GitHub can be accessed through that API and in the GitHub scraping process, API has been consumed to get some information. GitHub API comes with a rate limit. GitHub REST API rate limit is 1000 request per hour. Rate limit is the count that someone can make requests to the GitHub server per hour. Once rate limit is exceeded users may not be able to make requests. To increase the rate limit, users may purchase enterprise cloud account which has over 15,000 requests.

Because of that rate limits, ProbExpert has limited consuming the GitHub API by moving the searching resource's part into Google scraper. Since Google scraper is excellent with searching resources, consuming the GitHub API will not be needed. Rate limit can be saved to get resource information of search results passed by Google scraper.

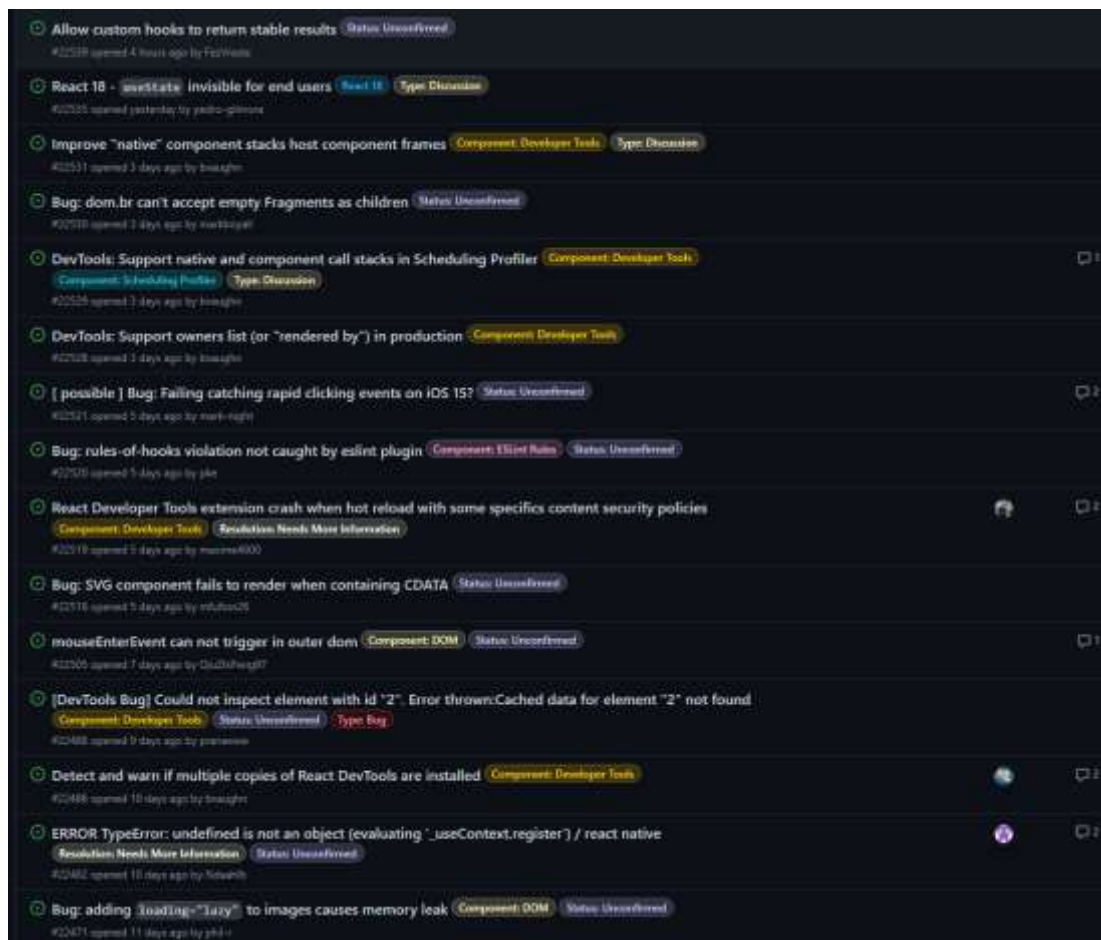


Figure 2.10: GitHub issues of ReactJS codebase

### 2.2.3. Other Scrapers

Beside the previous scrapers, details of YouTube, Medium and Dev scrapers will be addressed in this part. Among those scrapers, Medium and Dev scrapers are reasonably similar since both of them contains blog articles. Articles of Medium and Dev also contains images. To find the articles, as stated in the Figure 2.4 google search will be performed.

Because Medium and Dev have a significant number of articles and people submit new articles on a regular basis, Google search results can be sorted by date range to receive the most up-to-date information. After receiving cleaned links from google scraper, Medium and Dev scrapers will start the process. Issue with the Medium articles is some of them are marked as premium and without a subscription, users cannot read the articles. In such scenario, scraper will not be able to scrape the HTML content of the article and, only links to the articles will be passed to the next step. Google search scraper returns, at least ten links of articles, and the article scrapers will try to scrape ten articles each. Because of that there is a probability to find non premium labeled articles.

Because the Google search was done with a date range filter, all of the links returned are relatively new, and if scraping HTML content from those articles fails or article scrapers are unable to scrape data from the links, there is another method to scrape articles using a third-party API ("rss-to-json"). Because Medium and Dev have Really Simple Syndication (RSS) feeds, that users may use to find new articles on their respective websites, and all of the links returned by Google are relatively fresh, the rss-to-json api can be used to obtain article data. This is possible with both Medium and Dev.

Final scraper of the ProbExpert is the YouTube scraper, it scrapes data from YouTube such as video link, title, description, view count, comment count, like count and other relevant meta data. In this scenario, link, title, description and counts of view, and likes have been passed to the next step, information similarity calculation to determine the relevancy of the video to user's question.

### 2.3 Information Similarity Calculation

Information similarity calculation is the process of finding the relevance between scraped information and user's question. When considering the similar question finding method, information similarity calculation has been used to calculate the similarity between user's question and questions in the database.

Information similarity calculation model contains NLP approaches since same idea can be written in different words. With NLP approaches, such as semantic text analyzing, it is possible to compare the meaning of the sentences rather than word by word comparisons.

NLP is a field of study and application that investigates how computers might be used to comprehend and modify natural language text or speech in order to accomplish meaningful tasks. The goal of NLP researchers is to learn how humans perceive and use language so that appropriate tools and techniques may be developed to help computer systems understand and manipulate natural languages to execute desired tasks. The roots of NLP can be found in a variety of disciplines, including computer and information sciences, linguistics, mathematics, electrical and electronic engineering, artificial intelligence and robotics, and psychology. Machine translation, natural language text processing and summarization, user interfaces, multilingual and cross-linguistic information retrieval (CLIR), speech recognition, artificial intelligence, and expert systems are all examples of NLP applications.

As shown in Figure 2.11 text similarity calculation model contains few steps, text inputs, removing stop words, sentence tokenization, text encoding, cosine similarity calculation and finally finding the similarity. Text similarity value is basically come from cosine similarity calculation in this scenario. Cosine similarity and other steps will be explained in detail within upcoming sections.

In here text inputs means, the data that provide to the text similarity calculation model. in the scenario of automated answer generation, text inputs will be user's question and the text information scraped by scrapers. In similar question finding scenario, text inputs will be user's question and other questions in the database.

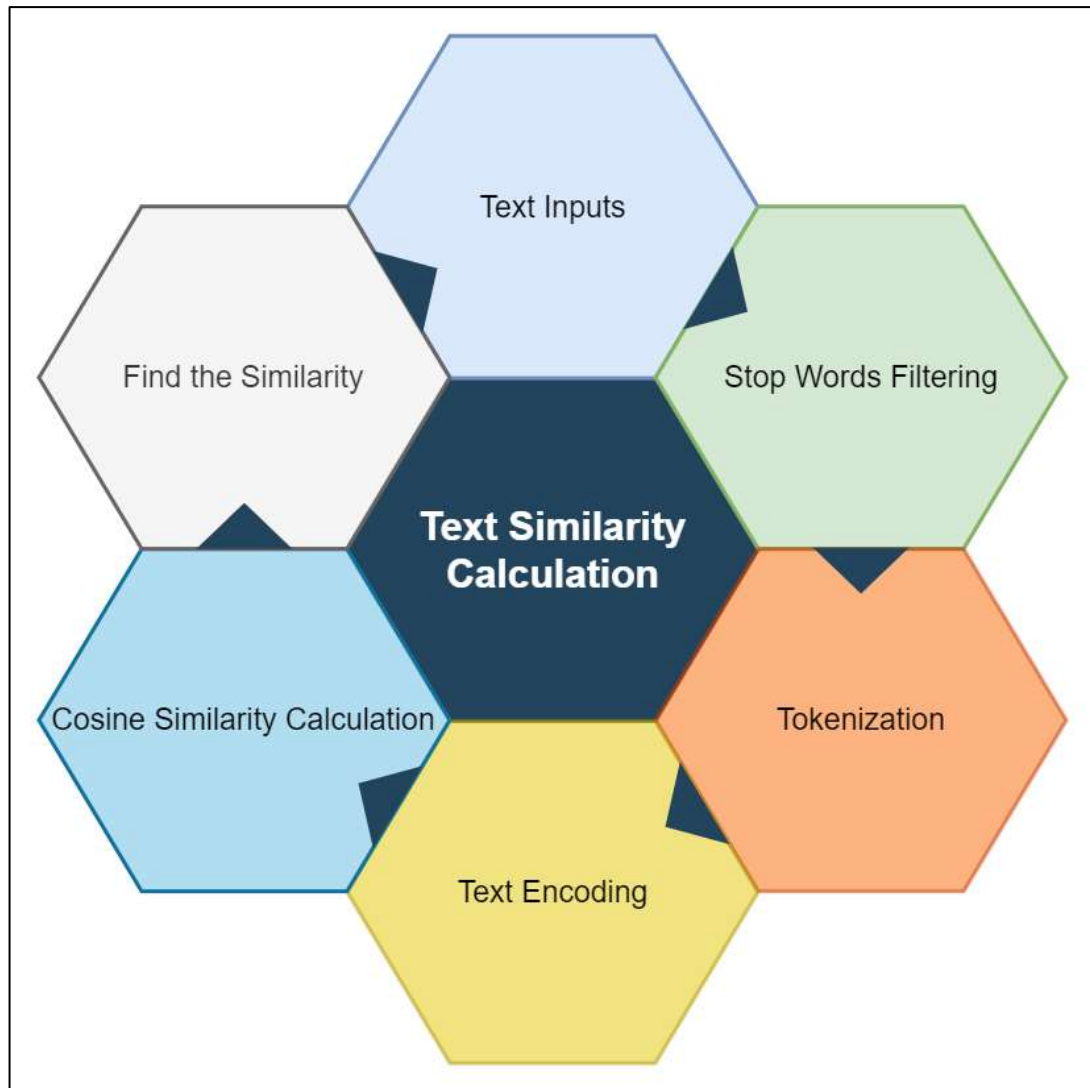


Figure 2.11 : Steps of text similarity calculation

Since most of the information scraped from YouTube are videos, and not text-based content inside the video has not been evaluated in this work. The title and description of the video are used to filter YouTube-related items. To discover the finest related YouTube videos for the user's question, metrics such as comment count, view count, and like count were taken into account. With title, and description of the video, similar process will be performed to calculate the similarity.

For all other scraped information, process of similarity calculation is identically similar. In forthcoming sections, models, logics, technologies and modules used to build each step illustrated in Figure 2.11 other than the text input step will be discussed in detail.

#### 1.1.4 Removing Stopwords

Stopwords are a collection of frequently used words in any language. As an example, stop words in the English language includes "the," "a," "is," and "are" and many other commonly used words. The idea behind removing these types of stopwords is that by removing common informative words from the text, the NLP model would focus more on the crucial information. (Technology related words in this paper). Stop words are often removed from the text before training deep learning and machine learning models since stop words occur in abundance, hence providing little to no unique information that can be used for classification or clustering.

In this paper, most of the technical words are focused since the platform targets the computer science related individuals. Therefore removing the stopwords will help text similarity model to perform well and quick. Comparison of a normal text and stopwords removed text can be shown in Figure 2.12.

Stopwords can be predefined, or can generate dynamic stopwords for documents with large number of words with technologies like TF-IDF. For information searching and text mining, TF-IDF is a weight model. It is a statistical metric that determines how valuable each word is in each text file. Since large text information is not processed in this platform, TF-IDF has not been used in this study.

Sample text	After removing stopwords
Python is a really awesome language	[python, really, awesome, language]
React is a frontend library	[react, frontend, library]

Figure 2.12: stopwords removing comparison

### 1.1.5 Text Embedding

A word embedding is a learned text representation in which words with similar meanings are represented similarly. In the same manner, text embeddings are dense vector representations of words in lower dimensional space. Embeddings are useful to find the keywords/similarity of the sentences. Similarity is not based on comparing the words in one sentence to another, similarity between two sentences need to be calculated upon the meaning of the sentence/text data. Once a sentence has been embedded it will be represented as a floating-point numbers vector (see Appendix A). To do the text embeddings of ProbExpert platform, this study has used BERT based approach.

#### 2.3.1.1 BERT and Sentence Bert

Previous language processing models were only capable of evaluating text inputs in a left-to-right or right-to-left order and could not do both at the same time. but when it comes to BERT, as its name suggests (Bidirectional Encoder Representations from Transformers) it can read in both directions at once. BERT was created and published in 2018 by Jacob Devlin and his colleagues from Google AI Language [11].

BERT set new state-of-the-art performance on various sentence classification and sentence-pair regression tasks. BERT uses a cross-encoder: Two sentences are passed to the transformer network and the target value is predicted. However, this setup is unsuitable for various pair regression tasks due to too many possible combinations. Finding in a collection of  $n = 10\,000$  sentences the pair with the highest similarity requires with BERT  $n \cdot (n-1)/2 = 49\,995\,000$  inference computations. On a modern V100 GPU, this requires about 65 hours [12].

With sentence BERT, fixed-sized vectors for input sentences can be derived. Using a similarity measure like cosine similarity or Manhattan/Euclidean distance, semantically similar sentences can be found. These similarity measures can be performed extremely efficient on modern hardware, allowing SBERT to be used for semantic similarity search as well as for clustering. The complexity for finding the most similar sentence pair in a collection of 10,000 sentences is reduced from 65 hours



with BERT to the computation of 10,000 sentence embeddings (~5 seconds with SBERT) and computing cosine similarity (~0.01 seconds).

To use the functionality of the Sentence BERT, in this study python module built by Ubiquitous Knowledge Processing Lab called “sentence-transformers”, and pretrained model bert-base-nli-mean-tokens. This model is able to convert sentences into 768-dimensional dense vector space. Then those vectors will be used to calculate the similarity value using cosine similarity, explained in the next section.

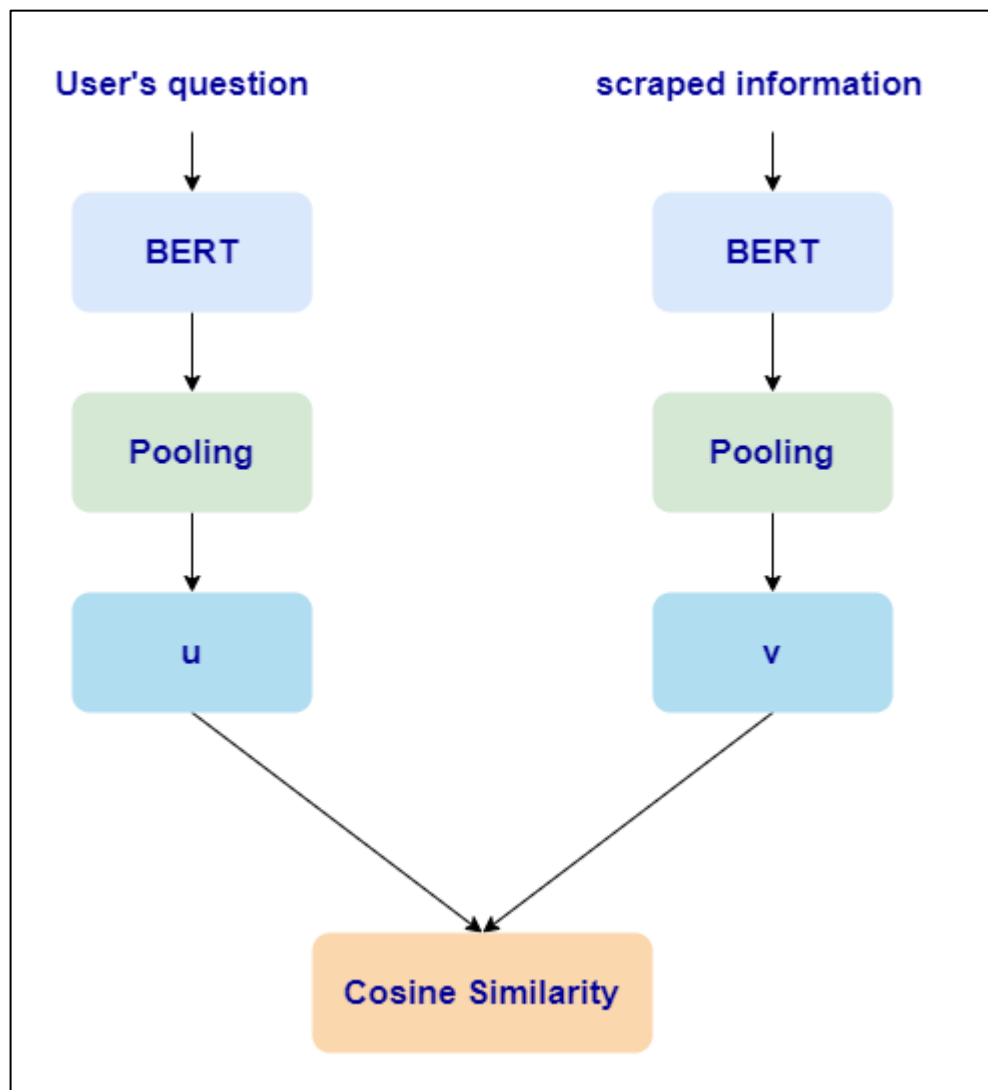


Figure 2.13: BERT keywords embedding

### 1.1.6 Cosine Similarity

After receiving dense vectors from text embedding model, cosine similarity model will start to find the similarity between targeted vectors. The vectors are usually non-zero and are located within an inner product space. Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space. As the cosine model gets 768-dimensional dense vector space from text embedding step, cosine similarity can be calculated. Equation of the cosine similarity can be written as (1).

$$\text{similarity}(u, v) = \frac{u \cdot v}{\|u\| \|v\|} = \frac{\sum_{i=1}^n a_n b_n}{\sqrt{\sum_{i=1}^n u_n^2} \sqrt{\sum_{i=1}^n v_n^2}} \quad (1)$$

cosine similarity value can be differed from 0 to 1, 0 been not similar and 1 been identical vectors. Since it is calculating the cosine value of the angels of the two vectors, illustration of cosine similarity can be seen as Figure 2.14. Usage of cosine similarity value will be discussed in the next section.

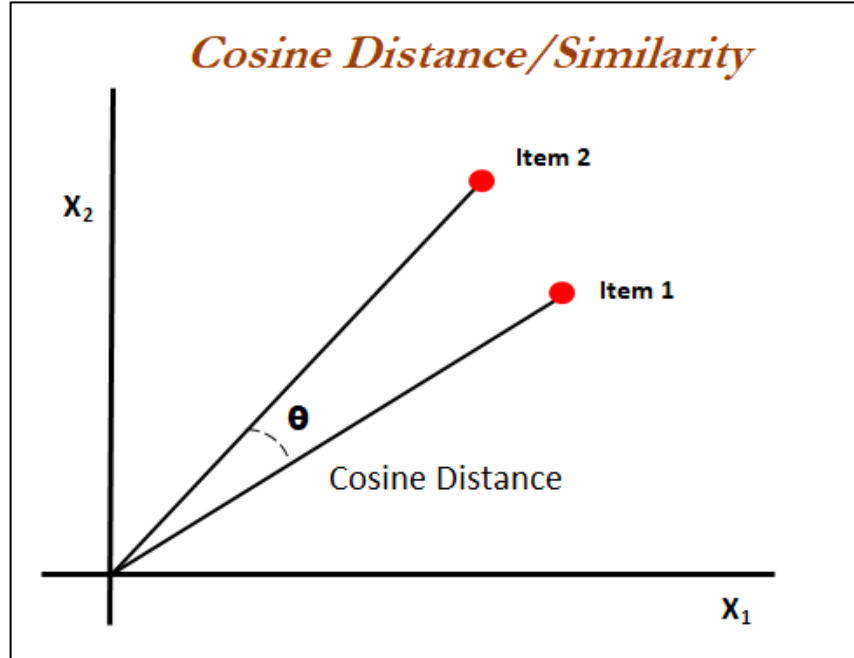


Figure 2.14: Cosine similarity illustration

source: <https://www.oreilly.com/library/view/statistics-for-machine/9781788295758/eb9cd609-e44a-40a2-9c3a-f16fc4f5289a.xhtml>

## **2.4 Automated Answer Generation**

Scraping data related to the question and finding the relevancy of the question is the hardest parts of the automated answer generation process. Information related to data scraping and information similarity calculation have been explained thoroughly in Section 2.2 above, and 2.3 above respectively.

After scraped data has been evaluated by information similarity, if the similarity value is greater than 80%, an automated response will be generated. The best-case scenario for automated answer creation is to have the greatest number of resources from all resource providers. The maximum resources for an automated answer are one Stackoverflow answer, two YouTube videos, five Medium and Dev blog articles, and finally a code sample from GitHub. If more than the maximum resources are collected, extra resources will be linked to the automated answer as external links. Users can visit certain resources by clicking on them.

Factors for best-case scenario to occur can be, popularity of the technology that user's question is referring to, generality of the question, it can be a common question that more users have come across, novelty of the technology and availability of the content regarding the technology.

For the worst-case scenario to occur, no resources should not be found, and there will be no external links as well. Probability of occurring such scenario is near to the zero. Let's consider about the most probable worst-case scenario that can be occurred, that is not having any actual resources like content of a Stackoverflow answer, articles of both Medium, and Dev, no YouTube videos, and no GitHub related content. In thus kind of scenarios, all the links found by various steps of data scraping related to the user's question will be attached, so that users can refer those links.

The rarity of the user's question, less popularity of the technology, that user has referenced in the question, obsolescence of the question and uniqueness of the question can be the reasons for the probable worst-case scenario. To avoid this, users can try to generalize the question.

## 2.5 Keyword Extraction

Keyword extraction (also known as keyword detection or keyword analysis) is a text analysis approach that extracts the most frequently used and important words and expressions from a document. Keywords extract from an article about keyword extraction can be seen in Figure 2.15 as a word cloud. Keyword extraction aids in the summarization of textual information and the identification of the central concerns presented.

ML, AI and NLP are used in keyword extraction to break down human language so that it can be interpreted and evaluated by machines. It is used to extract keywords from a wide range of material, including conventional documents and business reports, social media comments, internet forums and reviews, news items, and more.

Keyword extraction was applied in this study to extract keywords from user questions in order to assist in the identification of similar questions on the platform. Because similar questions are found in real-time, keyword extraction of the questions must be quick; otherwise, users may have to wait for the similar questions to appear. As fast as it is, keyword extraction must be accurate because the initial step in evaluating similarity questions is dependent on the extracted keywords.



Figure 2.15: word cloud of an article about keyword extraction, words extracted by keyword extraction

Keywords extracted from this step will be used to filter questions in the database by their keywords. That means, questions in the database have keywords added by the question asker, and keywords extracted from this step will be used as a query to search questions in the database. With precise and fast keyword extraction, most relevant questions can be found. In order to find the best key word extraction model, 3 different models have been compared (TF-IDF, RAKE, Spacy).

In When comparing the expected keywords and extracted keywords from TF-IDF and RAKE, there is a moderate difference. With each sample data, TF-IDF performed extremely well, and extracted the majority of the expected keywords. Since RAKE is more focused on extracting phrases, although extracted keywords in RAKE are not incorrect, keywords extracted from TF-IDF will be preferred over keywords extracted using RAKE. As a result of the usage of TF-IDF to extract keywords in this study, future actions after keyword extraction will be detailed in subsequent parts.

Table 2.1 there are two sample text inputs with expected keywords among the test data, and all three models have been utilized to extract keywords. Spacy has been used with the classification network "Roberta-large." Roberta is a self-supervised transformers model that was pretrained on a huge corpus of English data. This means that it was trained solely on raw texts, with no human labeling (thus its ability to use a large amount of publicly available data), and then used an automatic procedure to generate inputs and labels from those texts [13][14].

When it comes to model execution speed, Spacy is substantially slower than the other two, while the other two are roughly the same, albeit the TF-IDF model is slightly faster in some circumstances. Having slow execution speed can be a major issue, also Spacy model require more storage space than other two models as well. therefor Spacy with Roberta has not been used in the keyword extraction process of ProbExpert. Then there are two options left, in Table 2.2 there is a sample data from test execution, in there all the times are in seconds, and most of the times, time difference between those 2 models can be calculated in milliseconds, therefor choosing the best keyword extraction model will be dependent on quality of the extracted keywords in sample data.

When comparing the expected keywords and extracted keywords from TF-IDF and RAKE, there is a moderate difference. With each sample data, TF-IDF performed extremely well, and extracted the majority of the expected keywords. Since RAKE is more focused on extracting phrases, although extracted keywords in RAKE are not incorrect, keywords extracted from TF-IDF will be preferred over keywords extracted using RAKE. As a result of the usage of TF-IDF to extract keywords in this study, future actions after keyword extraction will be detailed in subsequent parts.

Table 2.1: Keyword Extraction model comparison

Sample text	5 Expected Keywords	Extracted Keywords		
		TF-IDF	RAKE	Spacy
Python module is a Python object with arbitrarily named attributes that you can bind and reference. Simply, a module is a file consisting of Python code. A module can define functions, classes and variables. A module can also include runnable code.	[Python, arbitrarily, module, object, classes]	[Python, Object, arbitrarily, named, attributes]	[python code, python object, file consisting, define functions, python module]	[python code, variables, code, python module, classes]
Summarization and key phrase extraction are two complementary techniques which, given a natural language text, extract the most important sentences and the most important words or phrases	['Summarization', 'Key phrase', 'extraction', 'natural', 'complementary']	['Summarization', 'Key phrase', 'extraction', 'two', 'complementary']	['key phrase extraction', 'important words', 'important sentences', 'summarization', 'phrases', 'given', 'extract']	['text', 'techniques', 'extraction', 'phrases', 'key phrase']

Table 2.2: Model execution time sample

Execution time
----------------

<b>TF-IDF</b>	<b>RAKE</b>
1.34s	1.46s
1.12s	1.19s
0.86s	1.02s
1.6s	1.74s
1.5s	1.58s

## 2.6 Finding Similar Questions

Finding similar questions in the database before user submit a question will be important for different factors. In users' side, question asker may save time, due to getting already answered question similar to the one they tried to post and also user who provide answers may not be exhausted with repeated questions.

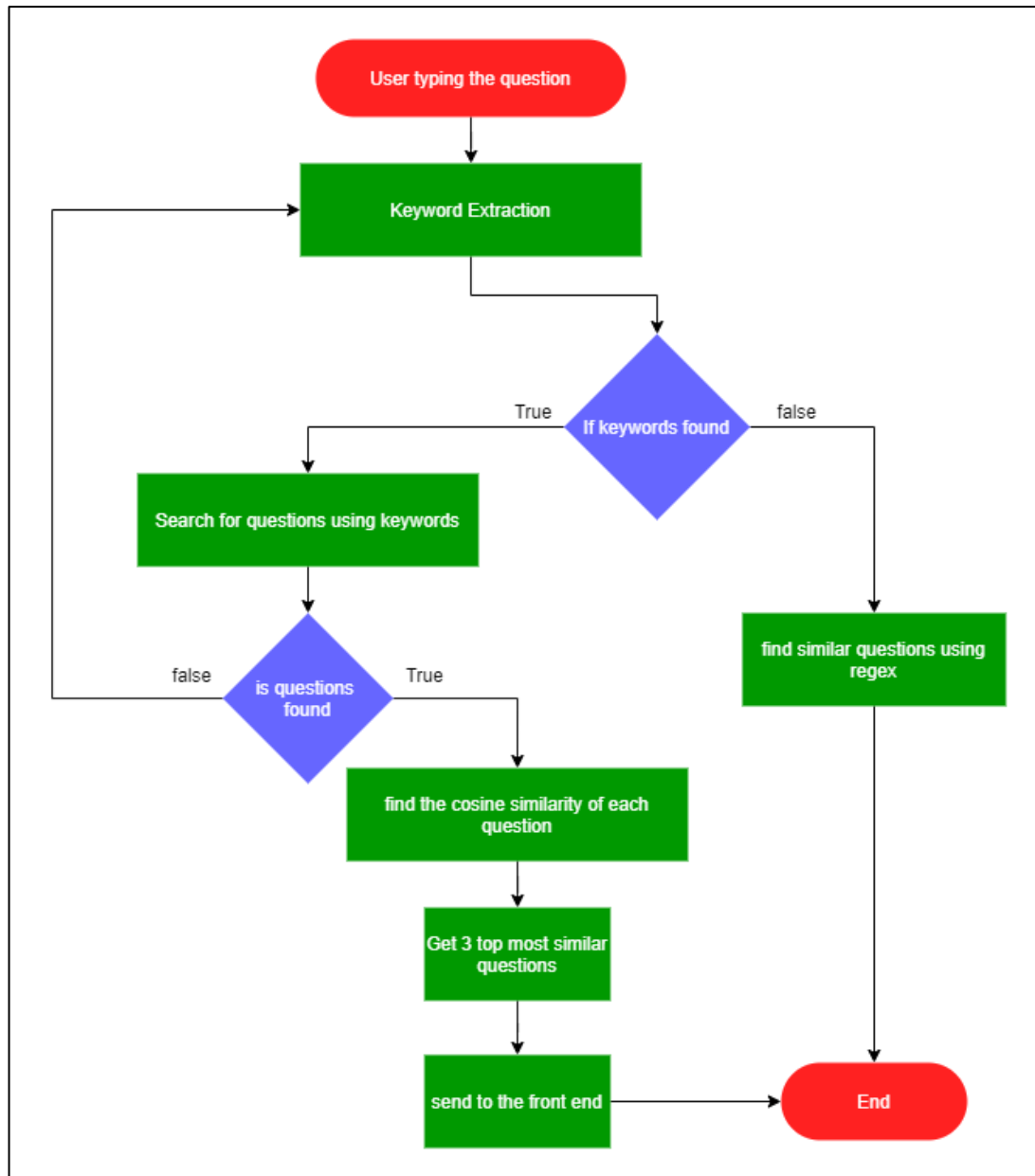


Figure 2.16: Flow of finding similar questions

since finding similar question is happening in real-time, user typed data will send to the backend before submitting the question. As shown in Figure 2.16, Once the partial text arrived at the similar question finding model, keywords are being extracted by keyword extractor model (explained in 2.5 above).

Structure of a question with sample data has been illustrated in Figure 2.17. There will be three major components for each question: title, description (text in database



structure), and keywords (tags in database structure), The extracted keywords will be used as a query to find questions in the database, and the tags in the database question will be queried with the keywords extracted in this step. Questions that require comparison will be reduced as a result of this step.

```
{
  "_id": {
    "$oid": "6087c107b8c06a179059dfa9"
  },
  "tags": [
    "python",
    "django",
    "web development"
  ],
  "score": {
    "$numberInt": "2"
  },
  "views": {
    "$numberInt": "24"
  },
  "title": "python django or flask for web development",
  "author": {
    "$oid": "6087bd93b8c06a179059dfa7"
  },
  "text": "I am very new to the web development in python. Please help me to choose python django or flask",
  "votes": [
    {
      "user": {
        "$oid": "6087bd93b8c06a179059dfa7"
      },
      "vote": {
        "$numberInt": "1"
      }
    },
    {
      "user": {
        "$oid": "60d72b7aa193310d34c1eb8a"
      },
      "vote": {
        "$numberInt": "1"
      }
    }
  ],
  "comments": [],
  "answers": [],
  "created": {
    "$date": {
      "$numberLong": "1619509511340"
    }
  },
  "_v": {
    "$numberInt": "2"
  }
}
```

Figure 2.17: Question structure with sample data

Once quantity of the questions, has been minimized, next step of the similarity question begins, that is to calculate the similarity between questions. For this step, technologies and models explained in information similarity calculation, Section 2.3 above have been used. Sample similarity value comparison can be seen in Table 2.3. In that table user has submitted a question related to CSS alignment of a HTML div node, then the similarity value has been calculated with few questions in the database,

most of the unrelated questions got similarity value below or near 0.3, and the largest similarity value one question has got over is 0.89, Although there is only two common words (excluding stopwords) between the users question and the chosen question, the idea meant by both questions are similar, and the developed model has been performed well to identify that. Since questions with similarity value over 0.8 considered as related, chosen question will be shown to the user as a similar question. If there is more than one question with similarity value over 0.8, top 3 questions will be shown to the user.

Table 2.3: Question similarity test data comparison

User submitted question	How to center a div element vertically
Questions in the database	Similarity value
python Django or flask for web development?	0.21996867656707764
ReactJS vs NextJS what is the difference?	0.3190786719322205
Why is python so widely used?	0.31442750692367554
what is null safety in dart?	0.31186968088150024
how can a python tuple differ from a dictionary?	0.17210909724235535
How do I undo the most recent local commits in Git?	0.3931351900100708
In CSS, how can I align a div vertically?	0.8966023588180542

## 2.7 Commercialization

### 2.7.1 Premium User tier

Since the process of automated answer generation is resource consuming, that feature can be limited for free tier user. One free user only has 2 auto generated answers for a week, if user need to have more than 2 automated answers, user will be able to upgrade

account to the premium tier and get unlimited automated answer generations. Also users who are not willing to upgrade to the premium tier will be able to purchase automate answer generation requests separately.

### **2.7.2 Monthly Subscription for Novice Users**

For new users having to find the answer in one place can be a really overwhelming feature, and it will be really helpful for their learning journey. If user need to generate answers quite often, users will be able to purchase a monthly subscription, this option will be more price saving and beginner friendly.

### **2.7.3 Advertisements**

#### **2.7.3.1 User based**

Free users will view user-based adverts, which will be shown only to free tier users, and advertisement suppliers will be Google or another reputable organization, so consumers will not be subjected to unpleasant advertisements.

#### **2.7.3.2 Sponsored Advertisements**

Organizations who wish to promote their new technology and services can post adverts on this platform because it has a specialized set of users. This is not a finalized approach because advertising has a detrimental impact on the platform's reputation.

## **2.8 Testing and Implementation**

This section covers how the system was implemented and tested using various testing methodologies in order to reduce problems.

### **2.8.1 Testing**

Testing is an important part of the success of any application. Quality testing processes can reduce the number of problems in a program and detecting them before release is critical and cost-effective. Some components of the platform have been developed based on test driven development (TDD). TDD means writing test case first before implementing the code, and then implement the code to pass the test case.

#### **2.8.1.1 Unit testing**

Unit testing is one of a smallest in size testing. It is a testing approach that test individual components of a system. The goal is to ensure that each component of the software code operates as planned. Unit tests have been built for most of the components in ProbExpert development, making it very easy to update a component to add a new feature because old functionality can be tested as well as adding new components to the system.

#### **2.8.1.2 API Testing**

Since API of the ProbExpert has been developed as a serverless backend, API testing is needed to ensure that all the endpoints are working without any issues. Security, availability and response time has been tested in API testing. With manually written test cases, APIs of ProbExpert platform has been tested.

#### **2.8.1.3 Integration Testing**

Since ProbExpert has built with multiple programming languages, and different technologies, integration testing is needed to make suer that overall system is working without any issues. Backend to front end integration, telegram bot integration has been tested in this step.

### **2.8.2 Implementation**

In this section, implementation details of components of the ProbExpert have been discussed. ProbExpert platform contains multiple services to perform the actions. Two

backend servers, front end application, machine learning models and a telegram bot has been implemented.

#### 2.8.2.1 Backend Servers

ProbExpert contains two backends implemented using python and NodeJS. Structure of the NodeJS server can be seen in Figure 2.18.

**NodeJS Server:** NodeJS server has been implemented to handle the basic routes of the system

**Python Server:** Python server has been implemented for the purpose of handling ML related requests, and NLP related requests.

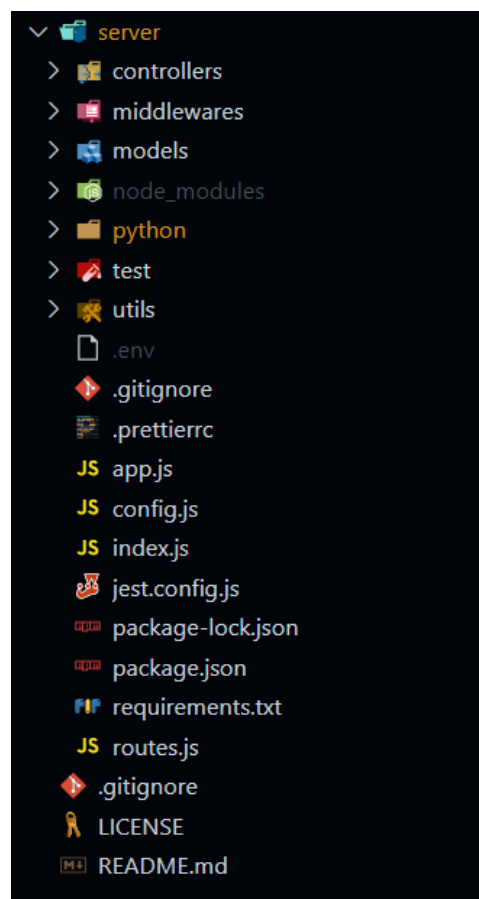


Figure 2.18: NodeJS backend structure

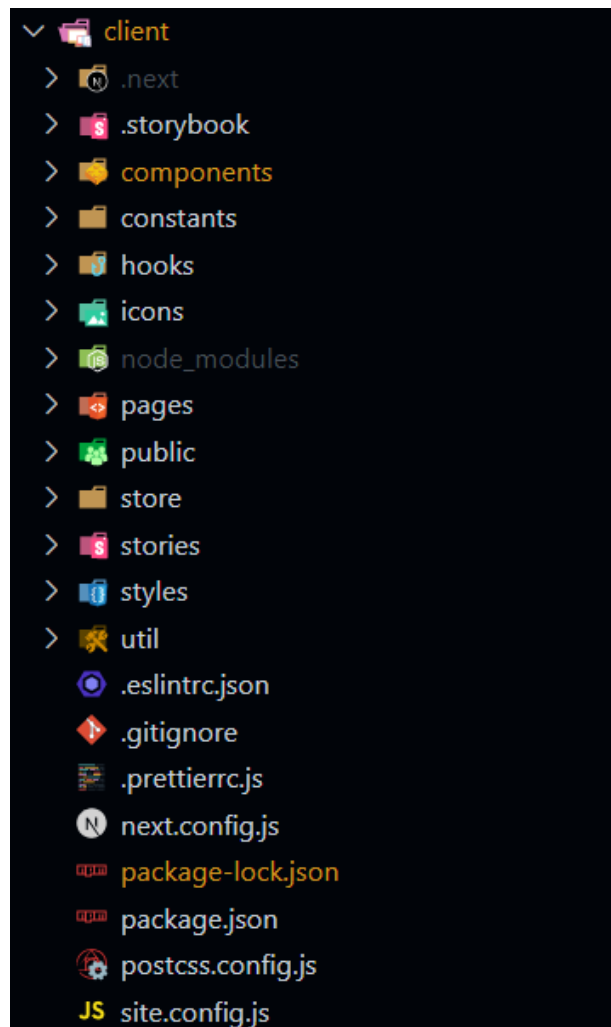


Figure 2.19: Frontend Structure

### 2.8.2.2 Frontend

Frontend application of the ProbExpert has been built with ReactJS and NextJS, Structure of the frontend application can be seen as in Figure 2.19

**ReactJS:** ReactJS has been used to develop all the components and base structure of the frontend. For styling custom themes has been used along with the custom CSS features like Grid, Flexbox and other things.

**NextJS:** The main reason to select Next.js to implement the frontend is that it provides server-side rendering (SSR) feature. SSR will help the frontend to load faster and indexed in a search engine like Google faster.

### 2.8.2.3 Telegram Bot

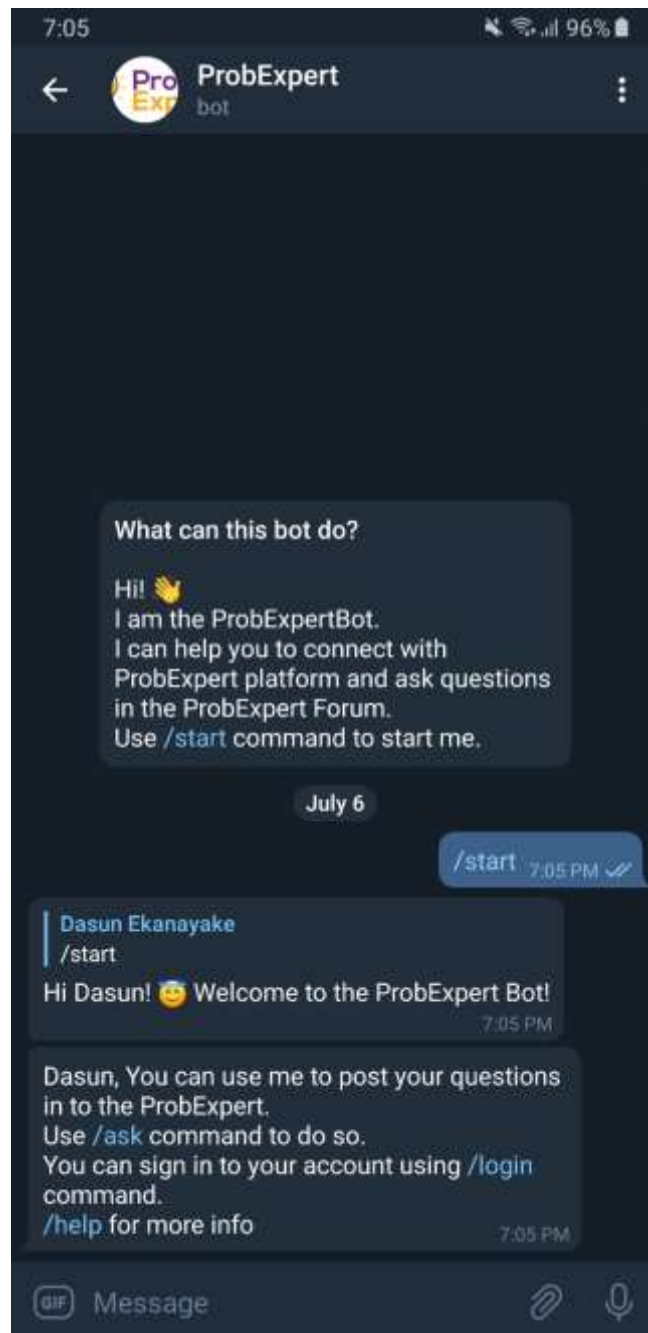


Figure 2.20: Telegram Bot sample image

Telegram bot has been developed using python, in order to archive the nonfunctional requirement of accessibility. Users with low network bandwidth and network coverage will be able to use the platform through the telegram bot. Along with python, telegram bot module has been used to build this.

#### 2.8.2.4 Database

MongoDB has been used as the primary database for the platform. Since it is a document database and no-sql database, it can be vertically scaled in any time. Since it is document-based database structure of the collections not need to be the same for each item, So that, in the future new values can be added to items, that old ones does not have.

#### 2.8.2.5 Deployments and Environment handling

Deployments of the ProbExpert services has been set automatically using features like CI/CD , Vercel hosting service, Heroku hosting service. For environments, ProbExpert has been implemented using two environments, prod and testing. All the development process has been processed in the testing environment. Once a set of features ready, one production release will be performed.

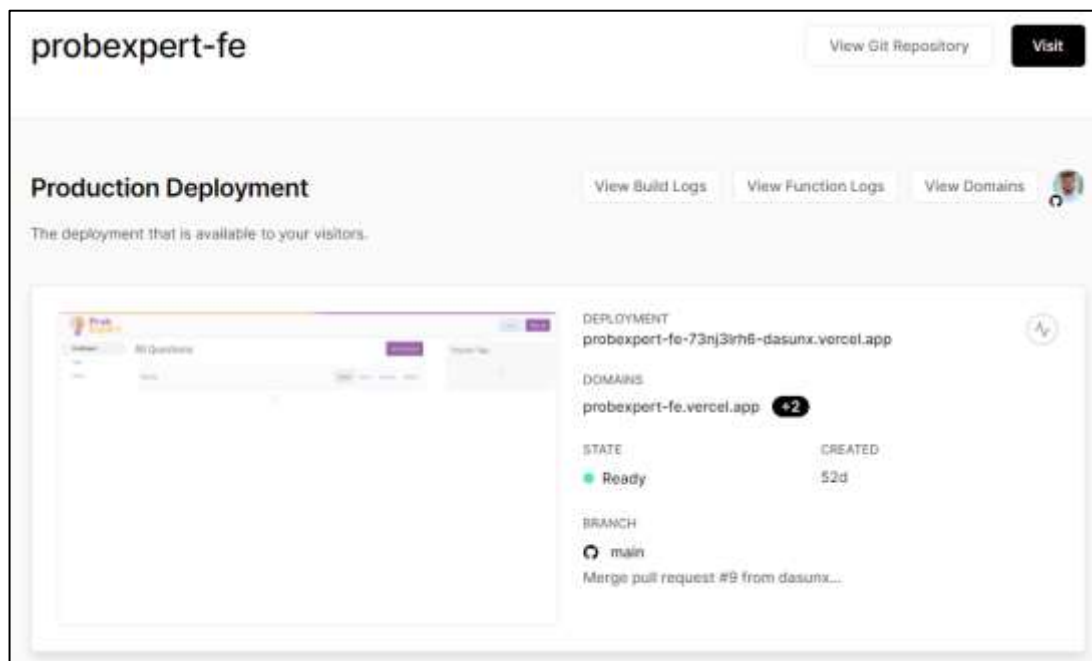


Figure 2.21: ProbExpert Deployments



### 3 RESULTS AND DISCUSSION

In this section, results, and findings of the ProbExpert automated answer generation system and real-time similar question finding system will be discussed.

#### 3.1 Results and findings

Developed components of the platform has been tested by asking 30 questions to find possible results of the automated answer generation approach and real-time similar question finding system. Because automated answer generation systems consume dynamic data rather than depending on fixed datasets, nearly all of the questions used to test the system received an answer automatically created with content inside the answer in addition to links to the information. The longest time it took to generate an automatic response was 30.9 seconds, while the smallest period was 5 seconds. The average time to generate an automatic response is between 10 and 20 seconds. Figure 3.1 illustrates the time spent on generating answers for 30 test cases.

With this test, we have found that the time it takes to generate an automated answer is proportional to the amount of information contained in the answer. The more content the answer has, the more time it takes to generate the answer. Table 3.1 includes resource count of an answer with respected time spent on generating the answer. First row of the table contains best answer with maximum number of resources.

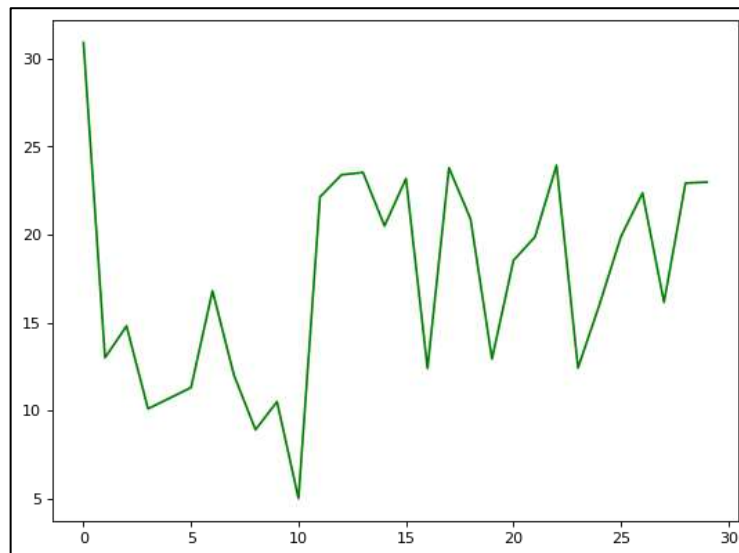


Figure 3.1: Time spent to generate 30 automated answers

Table 3.1: Resource count vs generation time

Resources count					Time (seconds)
Stackoverflow	YouTube	Medium	Dev	GitHub	
1	2	5	5	2	30.9
1	1	5	1	2	13.0
1	2	2	5	2	14.8
1	1	2	4	2	10.1
1	1	3	1	2	10.7
1	2	5	2	2	11.3
1	1	5	4	2	16.8
1	2	4	3	1	12.0
1	1	2	3	1	8.91
1	1	4	4	1	10.50
Average					13.90

If users had to visit all the resources in the best-case scenario answer, user need to visit 14 web pages. which would take 9.4 minutes according to (1). The average loading time for a desktop webpage is 10.3 seconds[15] ( $lt = 10.3$ ). The average loading time for a desktop webpage is 10.3 seconds[15] ( $lt = 10.3$ ). The user's time to determine the relevance of the resource has been calculated to be 30 seconds ( $rt = 30$ ).

$$total\ time = \sum_{k=0}^n (lt + rt) \quad (2)$$

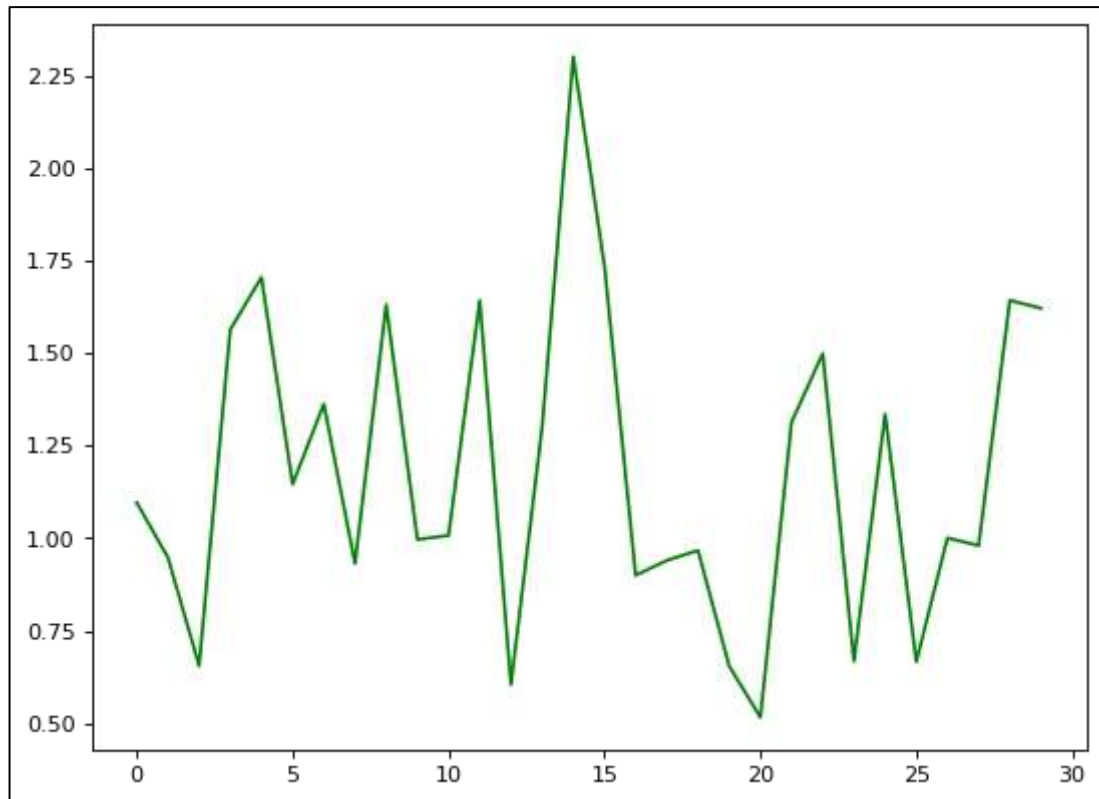


Figure 3.2: Similar question finding test execution times

Real-time similar question finding feature is also tested with the similar test data, As explained in Section 2.6 above, questions that have similarity value greater than 80% is considers as a similar question. Finding similar questions for each test case have been similar when considering the execution time. Maximum time discovered is 2.3 seconds and minimum time is 0.51 seconds, average for 30 test questions are 1.17 seconds. These times only includes the model execution times and retrieving data from database may add up few milliseconds to the total time.

While doing the tests and working with the platform, there are no major issues found, All the related systems and models related to the ProbExpert platform have worked as expected. Results from major components have been discussed in this section and in next section is about the research discussion.

### 3.2 Discussion

Most important finding of this research is the automated answer generation mechanism, that solve the problem of users having to wait large amount of time to get an answer from Q&A platforms.

Prior to conducting the study, the expected time to generate an automatic response was 2 to 4 minutes. When compared to the survey results, which showed that users had to wait at least 1 hour (Figure 1.4) for an answer, the projected 4 minutes duration was reasonable. After conducting the research and discovery of improved ways to complete each step of the automated answer generation model, the average time for generating an automated answer was 13.90 seconds, with a maximum time of 30.90 seconds, as indicated in the test results in Table 2.3. When the predicted time is compared to the existing average time of generating automated answers, the percentage difference is  $100 - 30.90/240\%$  is 87.12 %. This suggests that the findings of this study were 87.12% more successful than expected.

In terms of user experience, waiting for an hour has been reduced to 30.90 seconds in the test environment, a 99.14% percent reduction. When considering the time amount and resources is a significant success rate. Since these are test data, utilized in a test setting, values may vary slightly due to traffic and other factors, but there will be no significant variation because traffic and other resource-related things can be enhanced.

Because the resource sites are pre-defined, users only obtain content for answers from the same resources every time; this is not an issue; it is a limitation of current research, and it is an improvement that future research should focus on. Resource sites can be dynamically picked by constructing dynamic data scrapers. More future work and conclusion of this research can be found in the next section.

### 3.3 Summary of student Contribution

Table 3.2: Summary of student contribution

Student	Functionality	Description
Ekanayake P.M.D.P	ProbExpert Platform	<ul style="list-style-type: none"><li>• System design</li><li>• Database design</li><li>• Backend structure design</li><li>• API structure design</li><li>• Overall UI developments</li><li>• Common UI component developments</li></ul>
	Automatic answer generation	<ul style="list-style-type: none"><li>• Custom Data scraping models</li><li>• Information similarity models</li><li>• UI developments</li></ul>
	Similar question finding	<ul style="list-style-type: none"><li>• Keyword extraction</li><li>• Question filtering</li><li>• UI developments</li></ul>

## 4 CONCLUSION

Because time criticality and accuracy have become so crucial in computer science related fields, people need to stay up to date on information in their domains. As an outcome, community forums and Q&A platforms are becoming more popular among developers for finding answers to their questions and sharing knowledge. With more people asking questions, the time it takes to get a correct answer is a little longer. This study was carried out primarily to address the issue of users having to wait a long time for an answer from another user when using a Q&A platform.

There is substantial evidence that people in modern culture are becoming more involved in fast-paced activities. Waiting for someone to respond for an extended amount of time is not a positive experience in this case. By combining technologies from different domains such as ML, NLP and web scraping this study created a platform (ProbExpert), which includes an autonomous answer generation mechanism that generate answers for users' questions withing short amount of time. Aside from that, this research has resulted in the development of a method for finding comparable queries in real-time.

Future research into automated answer generation technologies should concentrate on developing more dynamic methods for gathering information that is publicly available on the internet rather than relying on information from predefined web sites. Furthermore, future research can be undertaken using various ML and NLP methodologies to get more accurate answers in a shorter amount of time.

## REFERENCES

- [1] T. Kilian, N. Hennigs, and S. Langner, “Do Millennials read books or blogs? Introducing a media usage typology of the internet generation,” *J. Consum. Mark.*, vol. 29, no. 2, pp. 114–124, Jan. 2012, doi: 10.1108/07363761211206366.
- [2] A. Robins, J. Rountree, and N. Rountree, “Learning and Teaching Programming: A Review and Discussion,” *Comput. Sci. Educ.*, vol. 13, no. 2, pp. 137–172, 2003, doi: 10.1076/csed.13.2.137.14200.
- [3] stackexchange, “All Sites - Stack Exchange,” *stackexchange*, 2021. <https://stackexchange.com/sites?view=list#questions>.
- [4] M. Piteira and C. Costa, *Learning computer programming: Study of difficulties in learning programming*. 2013.
- [5] GitHub User Search, “GitHub user search.” [Online]. Available: <https://github.com/search?q=type:user&type=Users>.
- [6] GitHub, “GitHub public repository search.” [Online]. Available: <https://github.com/search?q=is:public>.
- [7] G. Gousios, B. Vasilescu, A. Serebrenik, and A. Zaidman, “Lean {GHT}orrent: GitHub Data on Demand,” in *Proceedings of the 11th Working Conference on Mining Software Repositories*, 2014, pp. 384–387, doi: 10.1145/2597073.2597126.
- [8] H. Zhang, S. Wang, T.-H. Chen, Y. Zou, and A. E. Hassan, “An Empirical Study of Obsolete Answers on Stack Overflow,” *IEEE Trans. Softw. Eng.*, vol. 47, no. 4, pp. 850–862, 2021, doi: 10.1109/TSE.2019.2906315.
- [9] C. Ziakis, M. Vlachopoulou, T. Kyrkoudis, and M. Karagkiozidou, “Important Factors for Improving Google Search Rank,” *Futur. Internet*, vol. 11, no. 2, 2019, doi: 10.3390/fi11020032.
- [10] Google inc., “Refine web searches - Google Search Help.” [Online]. Available: <https://support.google.com/websearch/answer/2466433?hl=en>.
- [11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv Prepr. arXiv1810.04805*, 2018.
- [12] N. Reimers and I. Gurevych, “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks,” 2019, [Online]. Available: <http://arxiv.org/abs/1908.10084>.
- [13] Y. Liu *et al.*, “Roberta: A robustly optimized bert pretraining approach,” *arXiv*

*Prepr. arXiv1907.11692*, 2019.

- [14] J. Piskorski, N. Stefanovitch, G. Jacquet, and A. Podavini, “Exploring Linguistically-Lightweight Keyword Extraction Techniques for Indexing News Articles in a Multilingual Set-up,” *Proc. EACL Hackashop News Media Content Anal. Autom. Rep. Gener.*, pp. 35–44, 2021.
- [15] Backlinko, “web page speed analysis - 5.2 million desktop and mobile pages,” *Backlinko*, 2019. <https://backlinko.com/page-speed-stats> (accessed Aug. 28, 2021).



## APPENDICES

### Appendix A

Embedded representation of the sentence “This is an example sentence”

```
[ 3.20785761e-01  1.37235209e-01  2.59860307e-01 -9.81592014e-02
-3.63948137e-01  7.71966800e-02  3.69380236e-01 -3.12745571e-01
 1.33277312e-01 -2.93541998e-01  1.15458593e-01 -3.55825484e-01
 7.46323541e-02 -1.58448935e-01  1.35560036e-01  2.90623214e-02
-1.19101644e-01  2.45669022e-01 -3.30902040e-01  2.87550956e-01
 3.58330280e-01  4.96012777e-01  2.47232988e-01 -8.83323550e-02
 4.18649130e-02 -2.08248626e-02  2.40278855e-01 -4.19054449e-01
 4.51310933e-01 -2.65613288e-01 -1.54888928e-02  1.05151944e-01
 5.29058874e-01 -1.93584710e-01 -2.27550834e-01  3.09600145e-01
-7.52531812e-02 -2.09875740e-02  1.02610126e-01  1.09510422e-01
-6.37894496e-02 -5.12716174e-01  3.22104692e-02 -1.68447837e-01
-2.78715670e-01 -2.81573445e-01 -7.27939466e-03  4.53029156e-01
-1.75040886e-01 -1.90942615e-01  5.97975068e-02 -1.14427820e-01
 5.13522811e-02  2.61588097e-02  1.39900759e-01 -3.35936010e-01
-4.79327777e-04 -2.16055378e-01 -2.80452132e-01 -9.01873112e-02
-1.14230923e-01 -1.47018149e-01 -1.96869150e-01  4.47359324e-01
-2.24104170e-02  1.32173732e-01  4.94297713e-01  1.46769837e-01
-4.96936858e-01  4.49816622e-02  1.92679271e-01  1.17227145e-01
-1.75420761e-01  2.35334665e-01 -9.36276987e-02 -3.43187541e-01
 1.18700694e-02  7.65215084e-02 -1.60927668e-01  5.80342337e-02
 2.57425576e-01 -2.35456258e-01 -1.80083379e-01 -2.49861315e-01
-2.84124196e-01 -3.17099273e-01  1.10388212e-01 -2.69677132e-01
-1.32436603e-01 -7.37369061e-02 -2.89329588e-01 -2.64046967e-01
 4.75801259e-01 -2.97646463e-01  1.35365769e-01  9.06030312e-02
-2.30077729e-01  7.46968761e-02  1.31995648e-01  3.27475131e-01
 7.00818971e-02 -4.19959426e-02 -7.53934979e-02 -1.53492540e-01
 3.85508448e-01  2.87444293e-02 -5.29387593e-02 -3.10289413e-01
 1.33974999e-01  1.77829027e-01  2.14496091e-01 -1.05543442e-01
 1.60293728e-01 -1.78701237e-01 -5.27146123e-02  4.93296562e-03
-9.34184566e-02  7.39684701e-02 -2.57155988e-02  1.33839697e-01
 1.47296160e-01 -2.37625360e-01 -1.12931848e-01 -9.68184620e-02
 3.36583287e-01 -2.74174899e-01 -6.54706210e-02  4.92817201e-02
 1.31988838e-01 -4.43589687e-01 -1.06970947e-02 -4.84889336e-02
 1.75936908e-01  4.18258235e-02 -2.29800463e-01  1.55531272e-01
 9.68075469e-02 -6.19978718e-02 -6.72780722e-02  2.01664343e-01
 2.40504906e-01  1.58402666e-01 -8.88398886e-02 -2.84624040e-01
 2.17826337e-01  1.51687404e-02 -3.37094098e-01  8.40689614e-02
 2.85738647e-01 -9.28045362e-02 -1.36967516e-02 -1.84266698e-02
-4.77442853e-02 -4.09669816e-01  5.60709834e-03 -7.86342770e-02
 4.47715782e-02 -1.63302228e-01  3.03594857e-01 -5.78509364e-03
-1.69127747e-01  3.07288347e-03  2.40992531e-01 -1.12366877e-01
-6.86637387e-02 -1.63436174e-01  4.30761576e-01 -2.07451090e-01
-7.64148384e-02  1.73938945e-01 -3.71120870e-03  1.40606493e-01
-1.24495782e-01 -1.36831179e-01 -2.04466715e-01 -2.61337817e-01
 3.71090710e-01 -3.04821163e-01  4.69505221e-01 -3.01863611e-01
 1.10356934e-01 -5.00512958e-01 -2.32467607e-01 -6.82116672e-02]
```

-1.62878066e-01	-1.00999856e-02	2.02107981e-01	-1.46741077e-01
-7.48529062e-02	1.29013389e-01	-3.23341876e-01	1.52075076e-02
2.45560735e-01	-9.23012495e-02	-2.03286886e-01	-4.65878174e-02
4.88105297e-01	3.90794128e-03	1.31033406e-01	2.29238898e-01
4.68596816e-03	-2.30566472e-01	-3.05033296e-01	9.79674309e-02
2.37734079e-01	1.57506377e-01	-2.73724884e-01	-3.25381458e-01
-1.62799522e-01	-2.94019096e-02	1.50068283e-01	9.94747281e-02
-2.63197988e-01	-1.28741907e-02	2.52503812e-01	1.56142369e-01
2.77286526e-02	3.07224482e-01	-7.08917752e-02	-2.79852122e-01
-6.61465749e-02	1.63449809e-01	2.48402029e-01	-2.25151092e-01
2.03178808e-01	5.55712759e-01	-5.49423218e-01	-1.60272792e-01
-2.17318088e-01	-2.13816389e-01	2.99164116e-01	-3.37208658e-01
-2.11291283e-01	2.18243599e-02	-4.26608890e-01	-3.07555676e-01
1.65389016e-01	5.53647988e-02	3.58677715e-01	2.44155839e-01
-6.25228733e-02	3.81804496e-01	-2.54875392e-01	3.18901122e-01
-1.57904118e-01	1.69609874e-01	-3.86546850e-01	-3.15264225e-01
8.46402049e-02	1.04969963e-01	-3.98680151e-01	-2.33039245e-01
-1.69934742e-02	-1.49152175e-01	-2.56906092e-01	2.28837281e-01
9.48721841e-02	5.89506142e-02	-3.18252832e-01	2.65345752e-01
-1.56241253e-01	-1.86454326e-01	-7.94758797e-02	3.06922048e-01
1.02147833e-01	7.77629688e-02	6.35601804e-02	1.12111401e-02
1.67714190e-02	1.55028552e-01	-2.42936164e-01	-1.35720419e-02
4.19520810e-02	-1.46663278e-01	-1.28588304e-01	-1.26025125e-01
-1.79052040e-01	-1.78688481e-01	-2.58696843e-02	-4.14478242e-01
-2.21382782e-01	-3.12598377e-01	3.70721705e-02	3.77337545e-01
-6.11780100e-02	-1.59874067e-01	1.68886766e-01	8.85453820e-02
4.82252568e-01	1.93761662e-01	-5.98864667e-02	1.52439931e-02
4.03322279e-01	-1.68984398e-01	1.36139750e-01	-5.89164672e-03
-2.07116097e-01	-2.54944384e-01	-3.33225876e-02	-3.37102324e-01
2.86398470e-01	-1.41925007e-01	-9.24685150e-02	-4.32979539e-02
3.62478971e-01	5.97879058e-03	2.48066783e-01	1.14445195e-01
2.39894763e-01	-8.64451900e-02	-2.43712198e-02	1.18427597e-01
-2.26006538e-01	1.06434427e-01	1.46602124e-01	1.17185250e-01
7.15276152e-02	1.66553736e-01	-5.21074571e-02	-1.70544997e-01
1.13260243e-02	1.04481213e-01	2.68523805e-02	-2.58071631e-01
-6.96551949e-02	4.20858085e-01	1.81837037e-01	-6.45063892e-02
6.35846406e-02	2.33389344e-02	-5.41513562e-02	3.29986870e-01
4.55159128e-01	5.27775288e-02	-6.31922260e-02	4.73106690e-02
1.40821353e-01	-1.24132419e-02	2.03676075e-01	-1.38621494e-01
2.53020346e-01	6.68544248e-02	-1.15176968e-01	-1.60041705e-01
2.62127183e-02	1.22361556e-01	-2.35198736e-01	2.27538750e-01
9.03143063e-02	2.62368113e-01	-4.88039583e-01	1.70777097e-01
-2.72041652e-02	-1.49622843e-01	1.09966137e-01	-2.53704190e-02
1.39385045e-01	-1.34951323e-01	1.34330720e-01	-8.34580734e-02
1.29047018e-02	2.69696098e-02	-9.87128168e-02	3.18579614e-01
4.14020807e-01	-2.11029530e-01	-2.28444546e-01	6.75683767e-02
-3.47479492e-01	2.80165285e-01	7.19650388e-02	-3.62412304e-01
-6.52996302e-02	2.11518873e-02	1.09426178e-01	1.76790133e-01
4.92353261e-01	1.84055880e-01	-2.04466864e-01	-5.27974702e-02
3.04805804e-02	-1.36503428e-01	3.34767789e-01	1.12820566e-01]