

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

Thennakoon T.M.K.H.B

Ekanayake P.M.D.P

Ranasinghe R.M.A.K

Marapana S.K.C.W.K.M.R.T.S.B

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**

Thennakoon T.M.K.H.B

Ekanayake P.M.D.P

Ranasinghe R.M.A.K

Marapana S.K.C.W.K.M.R.T.S.B

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

We declare that this is our 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 our 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, We hereby grant to Sri Lanka Institute of Information Technology, the nonexclusive right to reproduce and distribute the dissertation, in whole or in part in print, electronic or other medium. We retain the right to use this content in whole or part in future works (such as articles or books).

Signatures:

Date:

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

Signature of the supervisor:

Date:

ABSTRACT

The World Wide Web contains a wide range of material from a variety of fields. However, when concerns towards the computer science domain, information users find on the internet may not be up-to-date due to the rapid pace of change and having to spend less time on the internet for researching and debugging tasks is an added luxury. Having an expertise level while providing answers through a platform is convenient for users, yet when a user signs into a platform, the user must start from the beginning, regardless of the level of competence in the field. Moreover, not having a proper way to evaluate the existing programming knowledge is another obstacle. To address mentioned complications, researchers of this paper have introduced a new e-learning platform- 'ProbExpert'. The platform has been constructed with machine learning and deep learning approaches such as NLP, keyword extraction, semantic information analysis, cosine similarity, and information summarization. With aforesaid technologies, ProbExpert provides systems in automated answering, optimized answer generation, structured question-based quiz evaluation together with a fully automated portfolio generation with a novel user ranking algorithm based on the bell curve.

keywords — answer automation, portfolio generation, quiz scoring model, user ranking

ACKNOWLEDGEMENT

I would like to take the procession of this section to express my sincere gratitude to all the individuals who guided me through this journey since day one. First of all, I would like to thank the Sri Lanka Institute of Information Technology (SLIIT) for according this opportunity to release innovative ideas through this project as a compulsory requirement of the course. Also, I take this opportunity to thank each faculty member and lecturer who lent their hand in guidance and support throughout this research project.

It was an honour to have a supervisor who assisted us to get back in the right direction when we chose the wrong path. So, I would like to express my heartiest gratitude to Ms. Dinuka Wijendra, who willingly agreed to supervise this research through the year and provided advice to enhance the worth of end result. I would like to thank our co-supervisor Ms. Anjalie Gamage, who willing to supervise this project throughout the year.

Finally, my gratitude would be paid to the colleagues of my team, my friends and my family members who encourage and support to strengthen.

Last but not least, I would like to thank all others whose names are not listed here, but have given their utmost encouragement and support in every possible manner

TABLE OF CONTENTS

| | |
|--|-----|
| Declaration..... | i |
| Abstract..... | ii |
| Acknowledgement..... | iii |
| List of Figures..... | vi |
| List Of Tables | vi |
| List Of Equations | vi |
| List Of Abbreviations..... | vii |
| 1 Introduction..... | 1 |
| 1.1 Background..... | 3 |
| 1.2 Research Gap | 5 |
| 1.3 Research Problem..... | 10 |
| 1.4 Research Objectives..... | 14 |
| 1.4.1 Main Objectives | 14 |
| 1.4.2 Sub Objectives..... | 14 |
| 2 Methodology | 17 |
| 2.1 System Overview..... | 17 |
| 2.2 Data Gathering..... | 18 |
| 2.2.1 Developers' activity extraction..... | 18 |
| 2.2.2 Stackoverflow answer scraping | 18 |
| 2.2.3 GitHub Scraper and API..... | 19 |
| 2.2.4 Other data related to automated answers | 21 |
| 2.3 Data Processing..... | 22 |
| 2.3.1 Tokenization..... | 22 |
| 2.3.2 Stop word removal..... | 22 |
| 2.3.3 Text Embedding | 23 |
| 2.4 Keyword Extraction..... | 25 |
| 2.5 Similarity Checking with Cosine Similarity | 26 |
| 2.5.1 Similar question finding..... | 27 |
| 2.5.2 The similarity between answers..... | 28 |
| 2.6 Generating the automated answer..... | 28 |
| 2.7 Summarization | 29 |
| 2.8 Quiz Evaluation Technique..... | 29 |
| 2.9 User Ranking Technique..... | 30 |

| | | |
|--------|---|----|
| 2.10 | Scorer Models | 30 |
| 2.11 | Automated Portfolio Generation..... | 32 |
| 2.12 | Commercialization..... | 33 |
| 2.12.1 | Premium Accounts for Hiring Managers..... | 33 |
| 2.12.2 | CV generate option as a premium service | 34 |
| 2.12.3 | Section for display Open job Opportunities..... | 34 |
| 2.12.4 | Display Sponsored Advertisements | 35 |
| 3 | Testing and Implementation | 36 |
| 3.1 | Testing..... | 36 |
| 3.2 | Implementation..... | 38 |
| 3.2.1 | Portfolio Front-end | 38 |
| 3.2.2 | Portfolio Back-end..... | 38 |
| 3.2.3 | Web Scrappers..... | 38 |
| 3.2.4 | Deployment..... | 39 |
| 3.2.5 | Telegram Bot..... | 41 |
| 4 | Results and discussion..... | 42 |
| 4.1 | Results and findings..... | 42 |
| 4.1.1 | Expert Identification | 42 |
| 4.1.2 | CV Shortlisting..... | 42 |
| 4.1.3 | As guidance for newbies..... | 42 |
| 4.1.4 | Automated answer..... | 42 |
| 4.2 | Discussion..... | 44 |
| 4.3 | Summary of student Contribution..... | 45 |
| 5 | Conclusion..... | 47 |
| | REFERENCES..... | 48 |

LIST OF FIGURES

| | |
|---|----|
| Figure 1.1: Number of resources visited by users to find an answer..... | 12 |
| Figure 1.2: Time spent on internet resources, to find an answer..... | 12 |
| Figure 2.1: High-level system Diagram | 17 |
| Figure 2.2: Flow of Stackoverflow scraper | 18 |
| Figure 2.3: GitHub issues of ReactJS codebase..... | 20 |
| Figure 2.4: stopwords removing comparison..... | 23 |
| Figure 2.5: BERT keywords embedding..... | 24 |
| Figure 2.6: Cosine similarity illustration..... | 26 |
| Figure 2.7: Flow of finding similar questions..... | 27 |
| Figure 2.8: Commercialization strategy of ProbExpert..... | 34 |
| Figure 3.1: Jest code coverage results..... | 36 |
| Figure 3.2: Testing API endpoint via Postman | 37 |
| Figure 3.3: Sample user document on MongoDB..... | 39 |
| Figure 3.4: Frontend deployment on Vercel..... | 40 |
| Figure 3.5: Backend deployment on Heroku | 40 |
| Figure 3.6: Telegram Bot sample image | 41 |

LIST OF TABLES

| | |
|---|----|
| Table 1.1: Prevailing algorithms' appropriateness to assess developers' skills..... | 5 |
| Table 1.2: Gap between ProbExpert and other platforms..... | 7 |
| Table 2.1 : User classes distribution | 32 |
| Table 4.1: Summary of student contribution | 45 |

LIST OF EQUATIONS

| | |
|-----------|----|
| 2.1 | 26 |
| 2.2 | 29 |
| 2.3 | 30 |
| 2.4 | 32 |
| 4.1 | 43 |

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

In recent years, answers on Q&A sites such as Stack Overflow[1] and Quora have become a key source of information for developers to address their technical issues.[2] In most cases, developers submit their inquiries to a search engine, returning a list of relevant postings that may contain answers. Then, developers must review the answers and acquire knowledge to find the solutions to their issues. However, the sheer volume of questions and answers on the mentioned Q&A websites makes it difficult to find information.

As developers face such problems, it may take a long time to get the exact response they require. It is also difficult to locate answers in long articles since there are vast amounts of noise and repeated material online, and the answer they find may only fix part of the problem. 'ProbExpert' is developed to address these concerns, which is more advanced and more equipped to handle challenges than the current Q&A systems available.

ProbExpert consists of some advanced and novel concepts that help users to get more than answers. One of the main functions of ProbExpert is that it will offer developers a fully fledged auto-generated portfolio out of the box. It will track and highlight all the contributions committed to leading Dev communities on the internet. Therefore, users do not need to update them with their recent works. The proposed platform is smart enough to identify and update itself according to the user's recent contributions. Since all the information is precisely optimized for search engines, these portfolios will appear in the search results of major search engines such as Google, Bing, etc. Finally, the developer's Coding, Q&A, and Professional scores will be displayed next to their profile picture.

ProbExpert also includes the functionality of the Question-and-Answer system as a crucial component of the system for sharing information and receiving answers. In addition to receiving answers from people, the system will construct an answer on its own using publicly available data. Stack Overflow, GitHub, Medium, Dev, and YouTube are information resources for the automatic answer. The information gathered will be processed and will go through numerous phases before the response

is finalized. To limit the number of duplicate questions, the platform will have a similar question-finding mechanism. As a public dev platform, anyone can answer the questions posted here, but the downside of this is that one question can get several duplicate answers, which inevitably leads to wasting the time of the person, and that is an area of focus with this research, 'reducing the time spent on finding the solution.' Therefore, to reduce this type of redundancy, an optimized answer is displayed. This process is focused on bringing all the alternative solutions into one optimized answer without harming the semantic information.

By considering only the above-stated specifications, it is clear that ProbExpert is not a traditional Q&A website. While it helps achieve the primary cause, another out-of-the-box functionality is that the platform can act as a medium to improve individuals' programming knowledge. Being such a platform leads to having an enormous dataset of day-to-day programmer's questions and received reliable, optimized answers, which can be easily turned into a platform for theoretical knowledge checking. In this scenario, structured-type online quizzes are considered an effective method[4] and have proven a positive influence on academics[3]. The questions are presented based on the user level, therefore enabling the adaptiveness to everyone for increased engagement and motivation[4]. Presented questions will be generated by the platform's questions. Users can type the answer for each question, and they will receive an unbiased score to assess their current knowledge level.

Overall, this paper covers how ProbExpert helps users develop user profiles, acquire optimized responses, generate quizzes based on relevant areas, and generate automatically summarized answers from the publicly collected information.

1.1 Background

Within the last two decades, plenty of research have been carried out to detect experts in question-answer communities and social coding platforms. In 2010, Wei-Chen et al. proposed a novel hybrid approach [5] that considers user subject relevance, user reputation, and authority of a category in finding experts. It is the first model up to that time that considers both users' reputation and the users' domain knowledge to the target questions. In 2014, Zhao proposed a topic-sensitive probabilistic model [6] for detect experts in the question-answer communities. On the other hand, evaluating developers in social coding platforms such as GitHub and GitLab are useful because they have the real-world software development activities of developers. Also, employers and HR managers often scan candidates' GitHub profiles in the interview process to learn more about their skills and interests. In reference [7], Capiluppi et al. identified four main insights that employers can derive from developers' GitHub profiles by doing an interview-based study with several IT employers. Those are 1. Shared open-source values, 2. Community acceptance of works and contribution quality, 3. Project management skills, and 4. Passion for coding. Since GitHub provides a REST API to its full data set, more researchers tend to use it as a fine resource to evaluate developers and their activities. In 2013, Saya et al. [8] and in 2016, Constantinou et al. [9] studied the characteristics of developer's activities and the expertise level. In 2019, Eduardo et al. proposed an unsupervised machine learning approach to identify experts in software libraries and frameworks among GitHub users [10]. Nevertheless, these studies are platform-dependent, and they are solely focused on identifying the experts. However, previous works did not classify users according to their skill level (e.g., beginner, intermediate and advanced).

Different technologies, such as keyword extraction, text embedding, similarity analysis, and web scraping, generate automated answers and similar question findings. Numerous studies on the mentioned technologies have been conducted in recent years. Almost all of the research has been done on individual technology. Because the ProbExpert question and answering platform uses different technologies, this paper contains studies on the bindings of those specific studies.

Automatic text summarization is a complex subject that is of increasing importance these days. When an input is applied, an automatically summarized result is produced. Although research on the Automated Text Summary began in IBM Research Laboratories in the 1950s,[12] the area of Text Summarization has grown exponentially in recent years due to the internet. Because there are ample amounts of material on the internet, manually summarizing large text pages is quite challenging. As a result, it is critical to sift through many available records to find the proper document. The goal of the text review is to condense the source text into a condensed version that keeps the material's content and overall meaning[13]. To make it easier for users to locate the solutions, the platform is equipped with finding a summarized answer utilizing the overall answers to the ProbExpert platform. A great deal of research has been done on answer summarizing[12]–[18]. A summary is a text derived from one or more texts that contain an approximate amount of the information contained in the original text but not more than half of the original text[13]. According to another guide, a text summary is a process of distilling the most relevant details from a source to create abbreviated versions for a particular customer and objective [12].

With modern-day knowledge shifting towards the internet, all learners can access vast information with several clicks. However, there are times that they cannot rely on the internet alone because, as learners, all the essential facts should be stored in memory. The best way to retain information is by taking quizzes in spaced repetition[11]. According to research[12], the quizzes method is more effective. According to the paper, reading information and then consolidating and testing the quiz form's knowledge helps retain the information[12]. Also, regarding a study based on students which spans five years, the conclusion was the online quizzes are a proven positive influence on students' academics[3]. With the above facts in mind, our platform ProbExpert has implemented a quiz system utilizing the platform's user questions and brilliant answers such questions received. Individuals will be exposed to everyday programming problems, preferably according to the user's knowledge level, which will be helpful in upcoming interviews or exams, and an unbiased score to evaluate the user's knowledge over their desired study area.

1.2 Research Gap

Since plenty of research have been performed to detect experts in the questions answer communities and social coding platforms within the last two decades, most of these studies focused on evaluating developers on individual platforms. Furthermore, they are focused on just finding the experts. No studies focused on classifying users according to their skill level (e.g., beginner, intermediate and advanced). The Prevailing algorithms that developed to detect experts on Q&A platforms only focused on the general type of Q&A platforms such as Quora and Yahoo Answers, and No study focused on the platforms in the specialized field such as programming. Therefore, it isn't idle to use previous research outcomes directly to evaluate developers' proficiency in specialized platforms such as Stack Overflow [9],[13], and [10] are the most famous studies conducted to assess developers on social coding platforms like GitHub. As in the Q&A platform-based research, these are also solely focused only on GitHub.

Table 1.1: Prevailing algorithms' appropriateness to assess developers' skills

| Algorithm | Suitability to assess developers |
|-------------------------------------|--|
| Page Rank | Not suitable (Assess users via link analysis) |
| HITS | Not suitable (Assess users via link analysis) |
| Mutual Reinforcement approach | Considers how many people are involved and who has helped whom. |
| Hybrid approach of Wen-Chen | Considers five types of relationships between users. |
| Topic sensitive probabilistic model | Considers the interests, expertise, and reputation of users. (Extended version of Page Rank) |
| Purposed model | Assess, score and rank developers based on their opensource contribution statistics. |

In this era, advanced knowledge-sharing social platforms have increasingly attracted people's attention due to their importance and impact on society. These platforms combine a broad range of knowledgeable data, including discussion forums (like Quora, Stack Overflow), blogging networks (Medium, DEV), code hosting platforms

(GitHub, Gitlab, Bitbucket), and professional networking sites such as LinkedIn. With the unstoppable rise of popularity, these platforms have moved beyond personal use. They have been increasingly used by organizations when recruiting employees. Therefore, these platforms start to focus more on generating a detailed profile for their users. These profiles contain users' reputations, contributions, activities, etc. But the problem is that these profiles are platform dependent. Therefore, up to today, there are no methods users can use to showcase all of their works and contributions in a single place (single platform).

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. 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 1.2: 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

Learning can be overwhelming in the present day with the support of technology, especially the internet, as there is a vast amount of information for a single subject. When learning a particular area, initially, a proper path is vital, and then during the process, it is inevitable to face numerous questions [1]. Many people are wondering about the path they had chosen to reach their individual goals. Once on the wrong path without proper guidance is identified, the final result occurs with wastage of money, time and energy.

After being futile with finding learning materials and starting to self-learn, people tend to seek the professional help of an expert in the field, mostly expecting expert-approved answers [2] or try to follow the path of a particular expert who has already reached their desired goals [3]. Also, individuals admit that experts have the knowledge to solve issues within considerably less time and present profound guidance. There is no specific technology for addressing these issues.

People have to look at public forums, even without knowing whether the content provided by other peers is right or wrong. Due to that particular reason, people may become uncertain. From the view of an expert, no matter how many contributions have been made for a specific platform, there is less reputation when moving from different platforms. Because every platform evaluates users' reputation by only considering the contribution towards the platform regardless of the expertise, every user has to earn reputation (points) from the ground up. Especially experts, who will not receive a proper reputation for proficiency at the beginning tend to quit or neglect such platforms as building a fitting reputation is time-consuming [5]. Without having an adequate recognition of the knowledge, contributions to questions may be often overlooked by other users misunderstanding the expert as a novice.

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 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 Zhanget al. 2019 [14] outdated answers could be categorized into two classes: legacy or invalid [15]. 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 [14]. 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

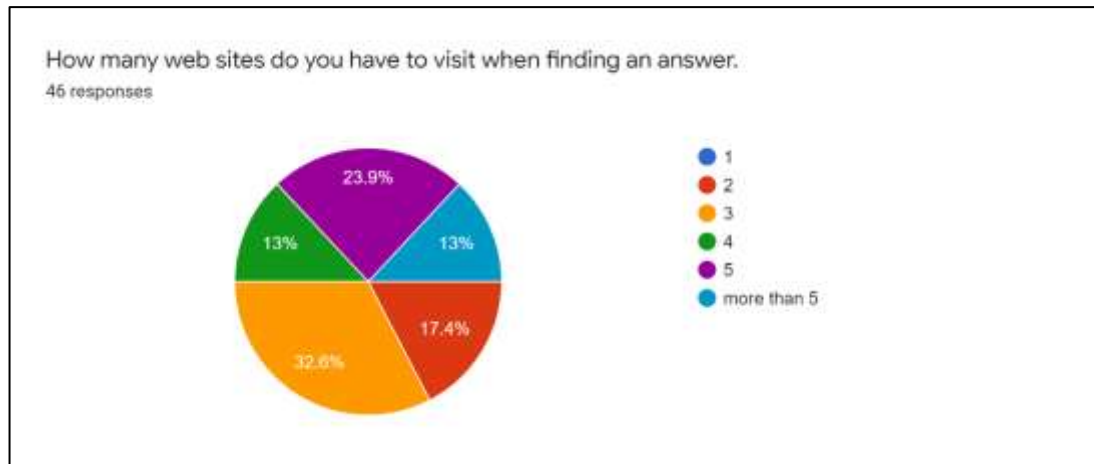


Figure 1.1: Number of resources visited by users to find an answer.

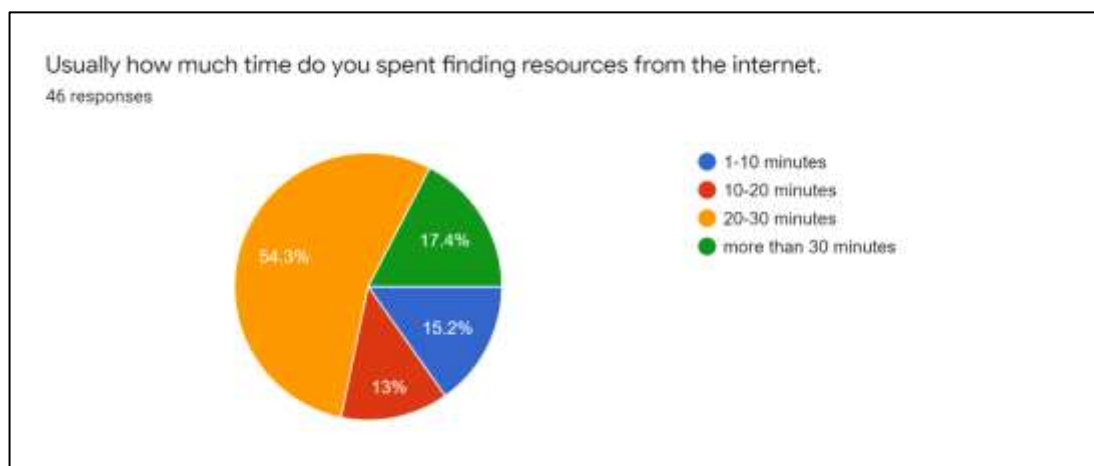


Figure 1.2: Time spent on internet resources, to find an answer

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 1.1 shows, majority of users have to visit at least 3 web sites and as shown in Figure 1.2, 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 that 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.

Another problem of such platforms is the availability of numerous answers to a single question. Some questions may have received virtually a perfect answer, but in order to form an ideal solution, users will have to survey a few other answers as an optimization method to form an optimal answer is absent in these platforms. When going through answers, the user will be weary as the same answer may be repeated in multiple locations. In another scenario, when a question or a problem is encountered, most are used to google the answer and end up having innumerable information or resources as answers are scattered throughout the internet. There is no proper way to acquire summarized and prioritized answers to a question in a single location.

Therefore, the problem of not having a platform to help users to learn new programming technologies in an ideal way, to connect with experts to clarify doubts and answers and to have an optimal answer for questions, which have been mentioned should be addressed in an efficient way.

1.4 Research Objectives

1.4.1 Main Objectives

To eliminate the problems stated above, an e-learning platform powered by Machine Learning and Artificial Intelligence is proposed, enabling personalized learning path creation for users and determining individual learning to be more resourceful and compelling. The ML-powered platform has the capability to find answers to users' subject related problems/questions across the internet archives. This facility is a great help for users as this process saves valuable time of the user by bringing answers and references to a single place with more accurate information. The system will be intelligent to generate an optimal answer from up-voted answers to form a complete answer. Additionally, with the help of previously answered questions, platform offers a quiz option to either refresh or solidify users' knowledge on their subject of interest. Apart from the above features, a user can get support from an expert on their field through video conferencing as well. The platform has the ability to accurately measure users' proficiency by evaluating the users' contribution to other platforms using machine learning to rank them in the platform. This evaluation method is a great opportunity for the users as well because this process generates a valuable portfolio of the user which can be used to showcase their skills/talents to the outside world.

1.4.2 Sub Objectives

In order to reach the main objective, specific objective mentioned below has to be fulfilled.

1.4.2.1 Detecting users' proficiency level along with an auto-generated portfolio

In this specific objective, we propose an approach that identifies developers' expertise according to their behaviors and activities on community questions answers platforms and social code management platforms. Advanced custom-made web crawler will be used to mine the developers' information. The proposed model will predict the developers' proficiency level (Newbie, Intermediate, Professional, Expert) and

generate a portfolio by filtering out unnecessary details from the gathered data set. This portfolio contains references to their research papers, blog articles, git contributes, academic qualifications, employment history, etc. Since these details are ordered precisely on a timeline, it will give newbies a solid idea of how they should shape their path accordingly to pursue their dreamed profession or career goals by seeing an expert's portfolio. Furthermore, this portfolio will help recruiters to identify the skills and proficiency level of the candidates, and candidates can also use this to showcase their skills at the interviews.

1.4.2.2 Form an answer automatically using public data for user's questions.

One of the primary objectives of the ProbExpert platform, 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.

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.

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

1.4.2.3 Optimal answer generation through up-voted answers

Users can openly post their questions in the thread section to get answers from different expert users. Once a question gets several answers, the 4 top voted answers will be selected to formulate an optimal answer for the viewers, analyzed answers will be processed to check whether answers share similar content. If answers share similar content server will merge them to reduce redundancy. Then the server will save the answer marking it as a formulated answer. Then it will be shown as the optimal answer to the question.

1.4.2.4 Formulation of structured-type questions for knowledge checking, using existing answered questions of the platform based on the user level.

To generate a quiz, the user has to select a specific topic to sort out the related questions that are already posted on the topic. Generation of the generic questions will be done by removing unnecessary words and components of selected questions. Then identification of each of the questions' difficulty level, according to the users' expertise level will be executed. (If a beginner level user asks the original question, which is used to formulate a quiz question, the question's difficulty level will be marked as easy), and if there are no questions related to a user's topic, then questions and answers will be formulated using external resources. These questions also will be generalized. Formulated questions will be structured type questions. The system will use both generated optimal answers, and the voted top four answers to increase the accuracy as the given answers will be different and mostly 'user unique'. Then the system will determine the similarity between the above answers and the user's answer to assign a similarity level\score and based on that metric; marks will be given. The answers getting from external resources will also go through the similarity checking process

2 METHODOLOGY

2.1 System Overview

The ProbExpert developer ranking system consists of three individual scorer models to evaluate developers' skills in three different major categories: Coding, Q&A, and developer professional achievements. GitHub, Stack Overflow, and LinkedIn are used respectively to identify and evaluate the areas mentioned above of a developer. In these three communities, GitHub is the only platform that offers a public REST API to its complete data set. Furthermore, when evaluating a developer, coding skills and contributed projects play a significant role. Therefore, the ProbExpert ranking algorithm gave the 80% of the total weight to the GitHub scorer model while giving only 20% to the other two models when finalizing the total score of a developer.

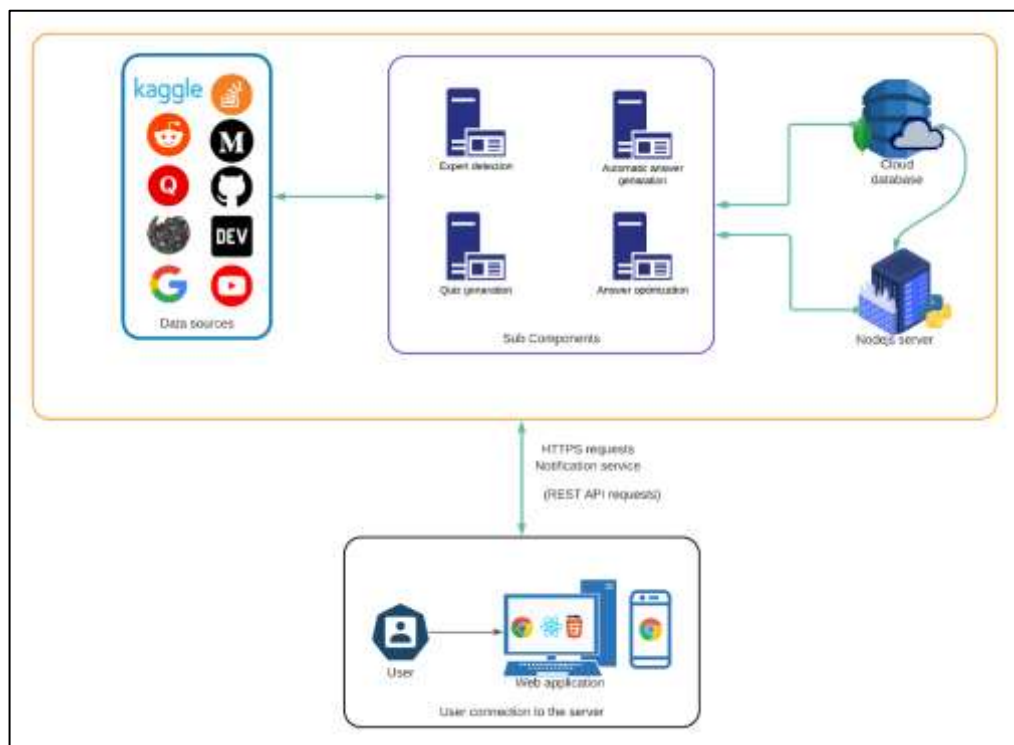


Figure 2.1: High-level system Diagram

ProbExpert platform is mainly composed of 4 components. In the upcoming sections above technologies have been discussed thoroughly.

2.2 Data Gathering

2.2.1 Developers' activity extraction

Three different scrapping tools have been developed from the ground up to collect the developers' data from GitHub, Stack Overflow, and LinkedIn. Since Stack Overflow and LinkedIn do not provide a public API to gather their users' data, BeautifulSoup4 and Selenium were used to build these two scrappers. However, unlike others, GitHub provides a GraphQL API to query all of its public data. Since it has a strict rating limit (1000 requests for an hour), and these response query results not changing frequently, A tool called Git-Response-Cacher to stop duplicate requests within 24hrs is developed to address this.

2.2.2 Stackoverflow answer scraping

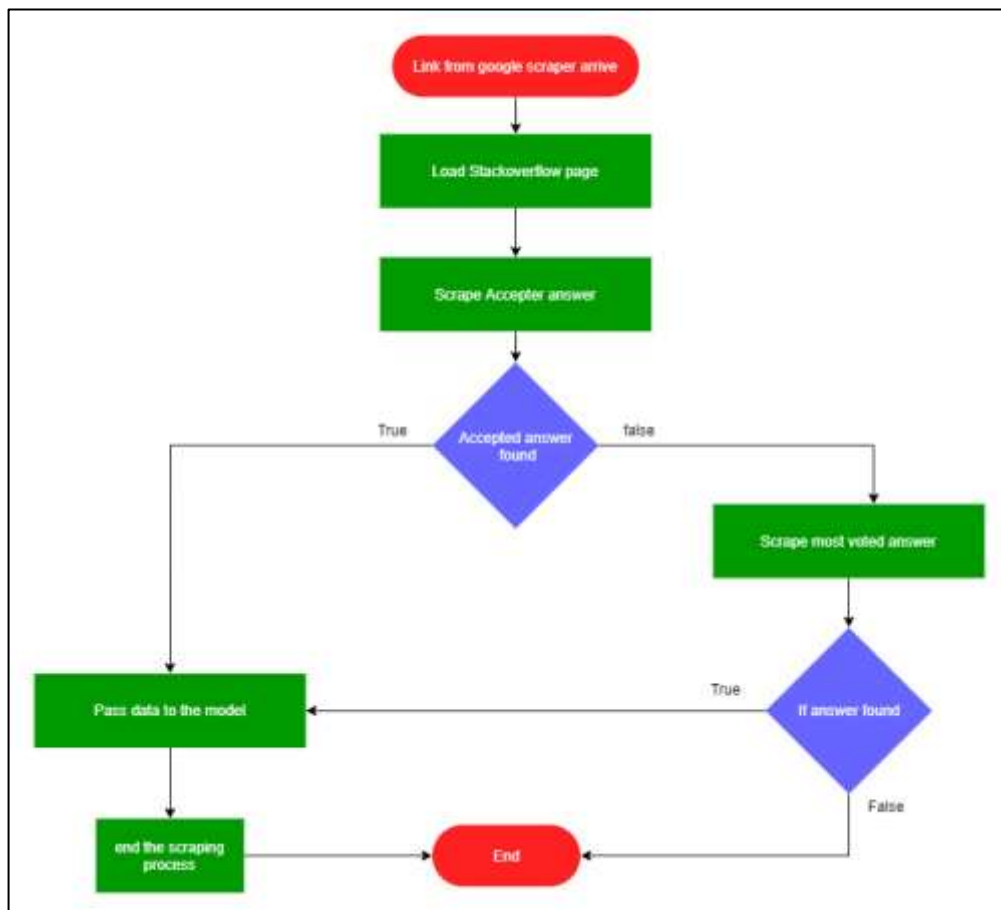


Figure 2.2: Flow of Stackoverflow scraper

for Stackoverflow answer scraping, a custom model has been built with BeautifulSoup4 and Lxml. The model will first try to scrape a verified answer if there is one. If the model finds a verified answer, it will process and filter relevant content out. If the model cannot find a verified answer, it will search for the most voted answer of the Stackoverflow question. Then it will perform the filtering process. Another application of mentioned Stackoverflow answer scraping is applied in quiz generation. If users' desired questions are not on our platform, they can search and obtain from Stackoverflow. Flow of Stackoverflow scraper can be seen in Figure 2.2

2.2.3 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 [16] 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 2.3, 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.

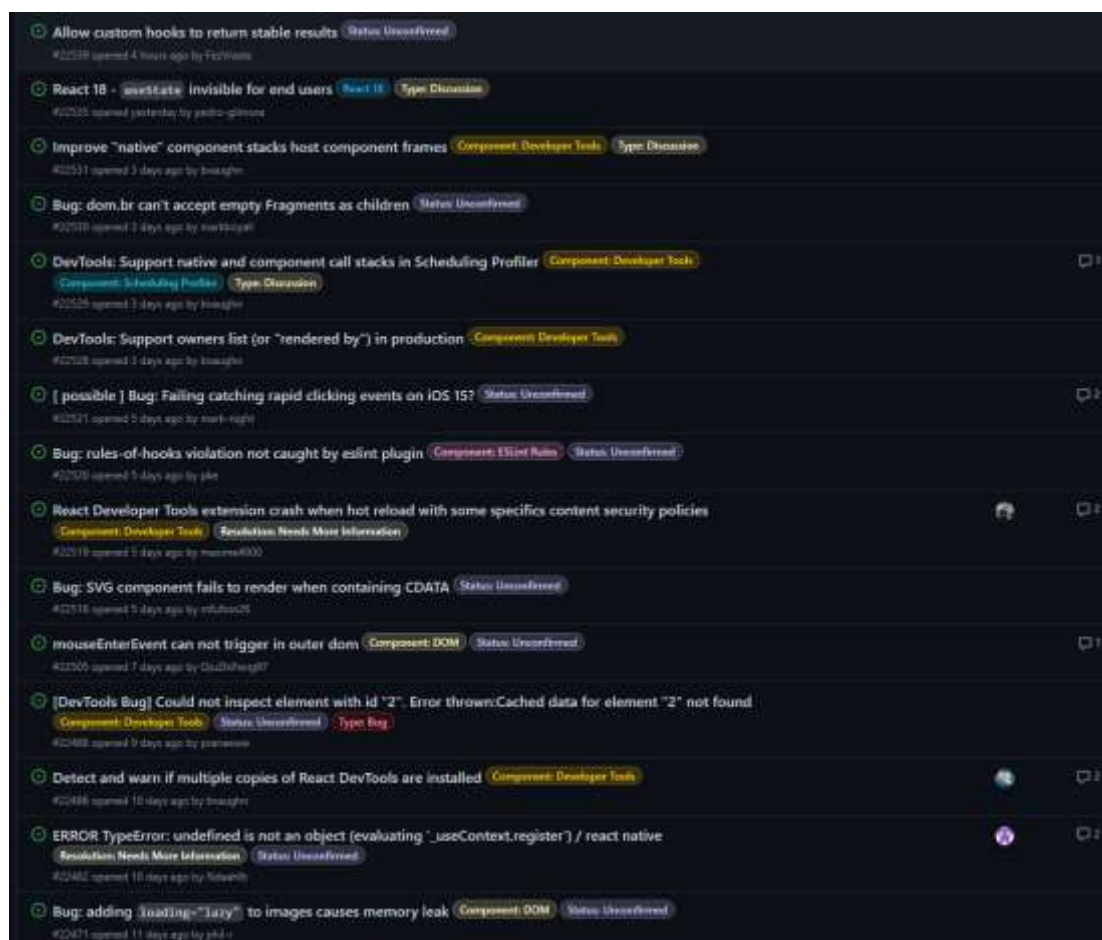


Figure 2.3: GitHub issues of ReactJS codebase

2.2.4 Other data related to automated answers

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 **Error! Reference source not found.** 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 Data Processing

A large amount of raw data was used to generate information in ProbExpert. As the first task, the data must be processed by removing unnecessary data and converting the data into a numerical representation. There are three steps in data processing.

2.3.1 Tokenization

In this process, the text is first tokenized into small individual tokens such as words, punctuation. This process is done by the implementation of rules specific to each language. Based on the specified pattern, the strings are broken into tokens using regular expressions, The patterns used in this work remove the punctuation in the

2.3.2 Stop word removal

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. Therefor removing the stopwords will help text similariton model to perform well and quick. Comparison of a normal text and stopwords removed text can be shown in Figure 2.4.

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 2.4: stopwords removing comparison

2.3.3 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 **Error! Reference source not found.**). To do the text embeddings of ProbExpert platform, this study has used BERT based approach.

2.3.3.1 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 [17].

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 [18].

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.

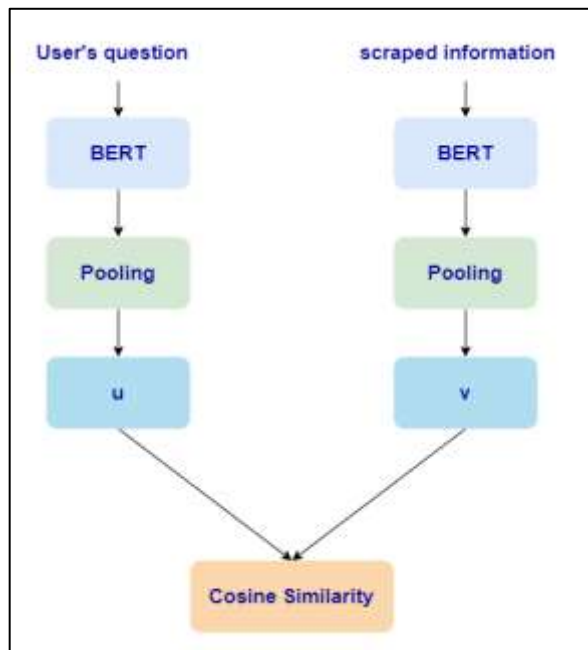


Figure 2.5: BERT keywords embedding

2.4 Keyword Extraction

Keyword Extraction (KE) is the automated extraction of single or multiple-token phrases from a textual document that best expresses all critical aspects of its content and can be seen as the automated generation of a short document summary[19]. In the ProbExpert platform, those extracted keywords will be used to calculate the relevancy of the information.

In this study, the Embedding-based keyword extraction method has been used to extract keywords. This exploits document embeddings and cosine similarity to identify candidate keywords. First, a document embedding is computed, then word n-grams of different sizes are generated, which are subsequently ranked along with their similarity to the embedding of the document. We can effectively achieve this task using pre-trained BERT models' transformer-based sentence embeddings[20][21]. KeyBERT is a minimal and easy-to-use keyword extraction technique that leverages BERT embeddings made by Maarten Grootendorst[19][22]. Once the desired question/answer is applied to KeyBERT, the following operations will be carried out to extract keywords.

- Extraction of candidate Keywords.
- Convert the documents/texts to numerical data.
- Calculate similarity.
- Diversification using Max Sum Similarity and Maximum Marginal Relevance algorithms.

2.5 Similarity Checking with Cosine Similarity

To find similar questions, the similarity between answers, and the relevancy of the information on this platform, 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 **Error! Reference source not found.**

$$\text{Cosine}(\vec{v}, \vec{w}) = \frac{\vec{v} \cdot \vec{w}}{|\vec{v}| |\vec{w}|} = \frac{\sum_{i=1}^N v_i \times w_i}{\sqrt{\sum_{i=1}^N v_i^2} \sqrt{\sum_{i=1}^N w_i^2}} \quad 2.1$$

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 in Figure 2.6

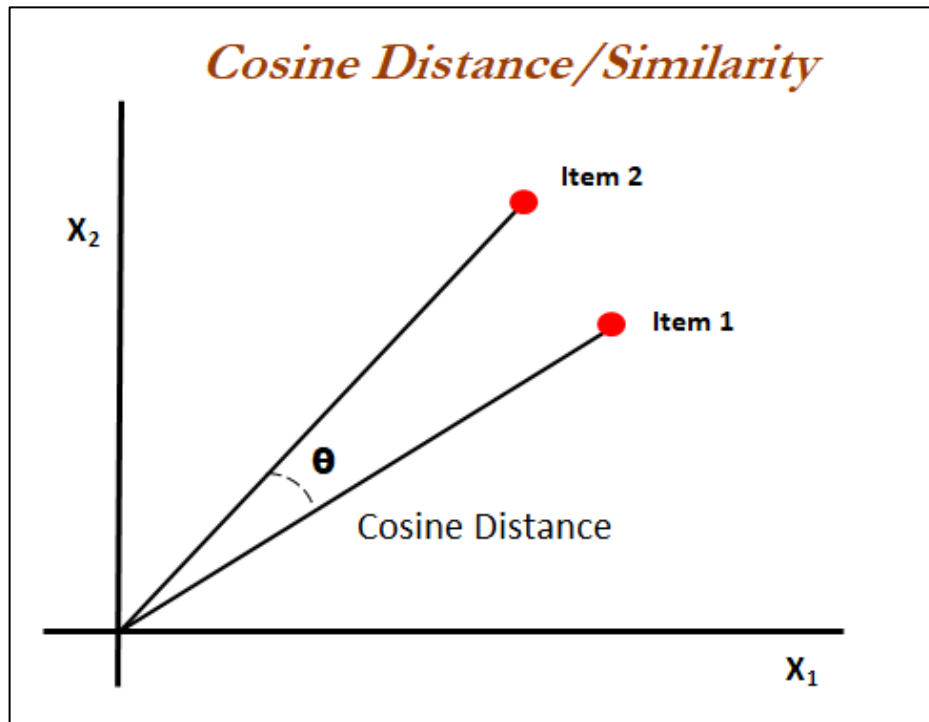


Figure 2.6: Cosine similarity illustration

2.5.1 Similar question finding

Cosine similarity was used to calculate the similarity between two questions (vectorized values). The user's question will be embedded first, followed by the question list. Then, for each question, cosine similarity will be determined against the user's question. Questions with more than 80% cosine similarity will be classified as similar in the last phase. Flow of similar question finding has been illustrated in Figure 2.7

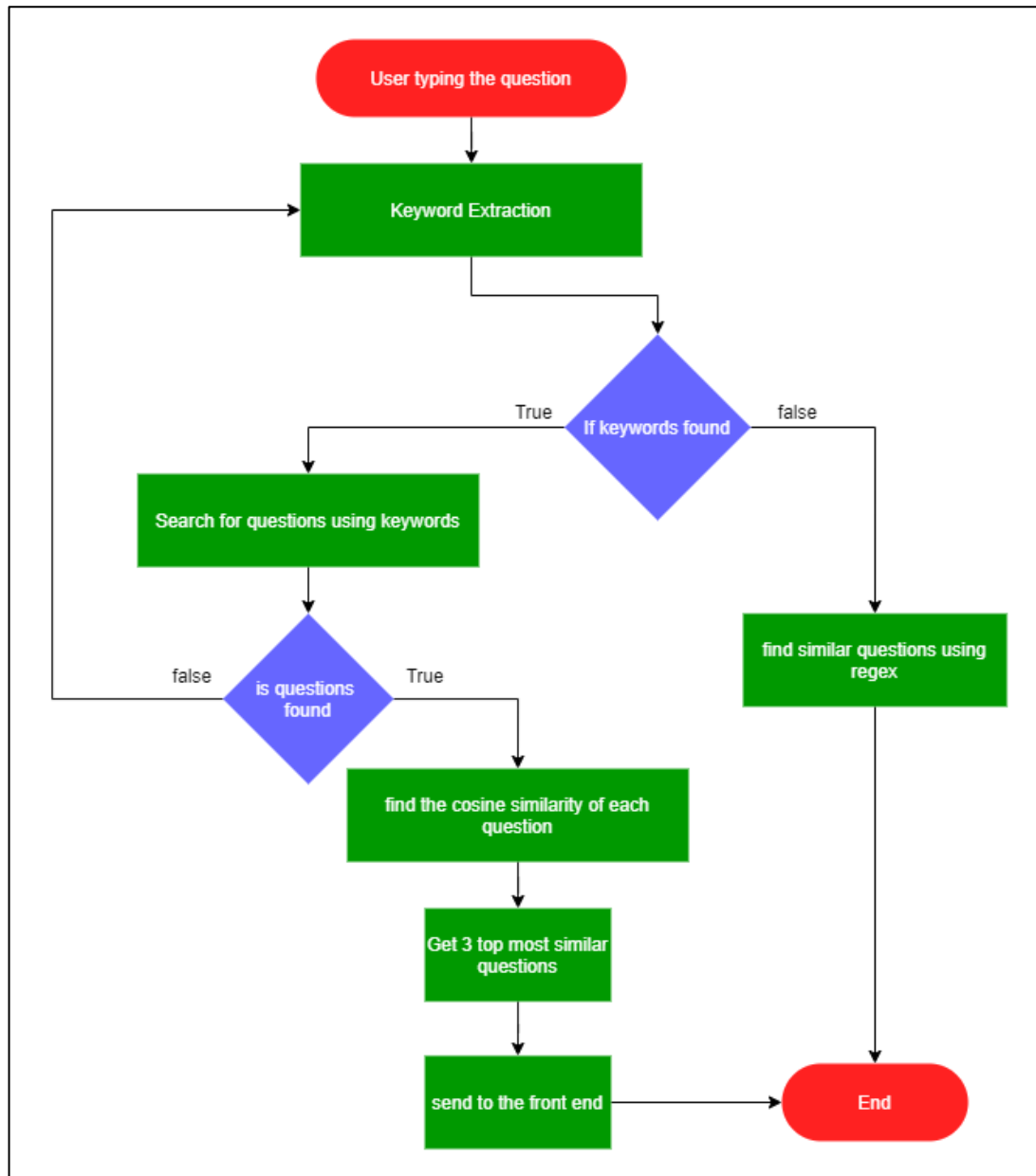


Figure 2.7: Flow of finding similar questions

2.5.2 The similarity between answers

Cosine similarity is used to calculate the similarity between user answers against the platform's verified answer. We used the BERT sentence transformers model to map sentences and paragraphs to a 768-dimensional dense vector space, enabling tasks like clustering and semantic search. After tokenizing, embeddings, and performing mean pooling to take attention mask into account for correct averaging, applying the cosine similarity between the two vectors was done, which generated a similarity score between the above-mentioned two answers.

A text document is represented as a vector of terms in this metric. The similarity between two documents can be calculated using this model by calculating the cosine value between the term vectors of the two documents[23]. Given two N dimension vectors \vec{v} and \vec{w} the cosine similarity between them can be calculated as follows:

2.6 Generating the automated answer

material scraped from web resources will be preprocessed before being utilized to calculate the relevance to the query using cosine similarity[23]. If the scraped information from a resource does not have a similarity value greater than 80%, it is considered irrelevant to the user's inquiry. Information with a more than 80% similarity value will be kept in a document database with the question id as the reference to the user's question. Material scraped from web resources will be preprocessed before being utilized to calculate the relevance to the query using cosine similarity[23]. If the scraped information from a resource does not have a similarity value greater than 80%, it is considered irrelevant to the user's inquiry. Information with a more than 80% similarity value will be kept in a document database with the question id as the reference to the user's question.

2.7 Summarization

The Q&A portion of ProbExpert is primarily focused on obtaining a semantically optimized answer to be more efficient and save time when using the platform. So, it is critical to use an appropriate summarizing technique for this. The most often utilized methods of summary in the current era are extractive and abstractive summarization. We discovered that abstractive approaches are superior since they are more human-friendly than reading a series of lines directly from the selected answers[24]. In order to do abstractive summarization, ProbExpert employed a mix of Transformers and BERT libraries, which are commonly used in machine learning. BERT (Bidirectional transformer) is used to overcome RNN and other neural network restrictions, such as long-term dependencies. It is a bidirectional model that has been pre-trained.

2.8 Quiz Evaluation Technique

The platform uses the returning score of the calculated cosine similarity [D] method for the quizzes' evaluation method. The process consists of calculating the above score for three scenarios.

- The cosine similarity score between the platform's optimal answer and the user's answer. (C_o)
- The cosine similarity score of the summarized user answer and the platform's optimal answer. (C_s)
- Cosine similarity of user answer's extracted keywords and platform's answer's extracted keywords. (C_k)

ProbExpert is equipped with three different BERT models for answer evaluation (distilroberta-base, bert-base-nli-mean-tokens, t5-base), which eliminates the dependency of a single model and eliminates any single model BERT transformer model related inconsistencies[25]. Once all the similarity scores are obtained, the mean score will be presented as the final core. If the calculated cosine similarity score can be represented in C, the formula is as follows

$$Score = \frac{C_o + C_s + C_k}{3} \quad 2.2$$

2.9 User Ranking Technique

The ProbExpert developer ranking technique consists of three individual scorer models to evaluate developers' skills in three different major categories: Coding, Q&A, and developer professional achievements. GitHub, Stack Overflow, and LinkedIn are used respectively to identify and evaluate the areas mentioned above of a developer. In these three communities, GitHub is the only platform that offers a public REST API to its complete data set. Furthermore, when evaluating a developer, coding skills and contributed projects play a significant role. Therefore, the ProbExpert ranking algorithm gave the 80% of the total weight to the GitHub scorer model while giving only 20% to the other two models when finalizing the total score of a developer.

2.10 Scorer Models

The primary Scorer model, which is GitHub scorer consists of 4 stages. In stage one, it gathers all required features from GitHub's GraphQL API for the given username. These gathered features are briefly described below. Then it calculates the penalty and reward score for each feature.

High Contributions (HC): HC measures the developer's contribution count to the repositories, with over 1000 stargazers. U/K gives the total commit percentage of the developer while S gives the total stargazers of the repository, and n gives the total repository count, which has over 1000 stargazers.

$$HC = \sum_{i=1}^n \left(s \frac{u}{k} \right) \quad 2.33$$

Commit Frequency (CF): Here, CF measures the current commit frequency of a developer by dividing the total number of commits of the last 365 days by 365. This value helps to determine how active the developer is currently.

Low Contributions (LC), Number of stargazers (TS), total followers (TF), total repository count (TR), number of used languages (TL), and the number of issues (NI)

opened by the developer are the other activities that are taken into account as the rewarding factors for the scorer model. When determining a developer's programming skill, each factor is multiplied by a specific weight according to its hardness [10]. The method of finding suitable weights for the selected features will be discussed later in this paper.

Since This scorer model is based on the Bell Curve graph [26], considerations were applied in some activities as penalty factors to balance the distribution of the scores. Therefore, if profile age, total commit count, and the number of used languages is under a specific limit, they will be considered penalty activities. These features also get multiplied by the computed weights based on their significance. After generating the score for each feature, it sums up all scores into one score called the total dev score (TDS). Total rank weights (TR) and assigned weight for each feature were calculated by consuming the data of 60 GitHub profiles extracted using the GitHub-scrapper tool. Since feature weights do not relate with the population [10], a balanced data set which is based on stratified balanced sampling that included undergraduates, software engineers, senior software engineers, open-source contributors, GitHub star developers, and beginner profiles aged less than one year, which are not included to any of above stratum, was used as a balanced sample. Two annotators were used for labelling the profiles into the classes, and then by plotting each feature against their class, weights were calculated.

Then to determine the rank of the selected developer, the normalized score is calculated based on the Bell-curve by consuming the derived TDS value from the previous step. The normal cumulative distribution function (normalCDF) has been used to derive the normalized dev score from the generated score instead of using empirical rule. since it does not give the precise value for the result, ranking users based on those values will not be accurate. However, it is impossible to calculate the cumulative distribution directly in JavaScript because it does not provide any built-in method to calculate the error function (erf) [27]. Therefore, by using the erf method of math.js opensource library, the normalCDF function was implemented.

$$NDS = \frac{1}{2} \operatorname{erf} \left(\frac{TDS - TW}{\sqrt{2} TR} \right) \quad 2.44$$

NDS gives the normalized developer score using TDS, summed values of assigned weight for each factor (TW), and the summed value of the weight of each rank (TR). Finally, by considering the left tailed p-value from the generated bell curve, a user is classified into the related class according to the values of table I.

Stack Overflow and LinkedIn scorer models have also been developed using the same manner used to develop the GitHub scorer model. User Reputation, Total answered tags count, Total Number of Badges with their types, the number of endorsements, and the number of peer connections have been used as the features for these two models. Finally, the developer's overall score is calculated by giving 80% of the weight to the GitHub score and 10% each for the other two scorers.

2.11 Automated Portfolio Generation

By consuming the given usernames of GitHub, Stack Overflow, and LinkedIn of a developer, ProbExpert's portfolio system will first scrape and collect the developer activities and calculate the developer score as explained in the previous sections. Then it generates the developer portfolio by showcasing the developer score and other valuable and significant developer activities using timeline progress bars and graphs.

Table 2.1 : User classes distribution

| Class | Range |
|--------------|--------|
| Newbie | 100-81 |
| Beginner | 80-56 |
| Intermediate | 55-46 |
| Advanced | 45-26 |
| Expert | 25-5 |
| Guru | 5-0 |

Furthermore, it includes a leaderboard that assists recruiters and other users in finding out the experts on the platform. Since these portfolios needed to be optimized for

search engines, Next.js [28] has been used on the top React.js library to enable static rendering. Therefore, all the generated developer portfolios can be cached on all the major content delivery networks (CDNs), and also, they are 100% accessible by web spiders to rank them on major search engines. Also, Next.js hybrid-static page generation method is used for loading the portfolios without any server-side computation by serving the pre-rendered pages stored in the server.

2.12 Commercialization

Introducing ProbExpert into the market is a significant goal in this project. Below Figure 2.8: Commercialization strategy of ProbExpert illustrate the marketing plan and strategy we made for ProbExpert commercialization purpose.

2.12.1 Premium Accounts for Hiring Managers

As discussed in the literature survey and research gap, recruiting managers still do not have a single platform to evaluate all their candidates in a single place. Therefore, they have to go into each platform like GitHub, Stackoverflow, and LinkedIn to evaluate each candidate. They can get rid of all the time-consuming hard work by just creating a recruiters premium account in ProbExpert platforms. Using the recruiter premium, they can get all the advantages listed below. Utilizing the following perks, organizations can save countless minutes on CV shortlisting and find the top unemployed developers in the field in no time.

- Easily identify all the unemployed top developers using the leaderboard.
- Evaluate their candidates' proficiency and contributions to the software field via generating the profile within a single click.
- Compare all their candidates by looking at the generated score for each of them.

2.12.2 CV generate option as a premium service

By using ProbExpert portfolio system, developers can maintain their online portfolio as they want. To attract more users into the system and for the platform's popularity, this service will be offered to developers as a completely free service. However, downloading their online portfolio as PDF can be provided to the users as a premium option by considering users' feedback.

2.12.3 Section for display Open job Opportunities

In ProbExpert, we could provide a separate space for organizations to post their job opportunities. This implementation also has a positive effect on getting more developers to the system and turning them into daily users. This service could provide to the organizations as a pay-as-you-go service.

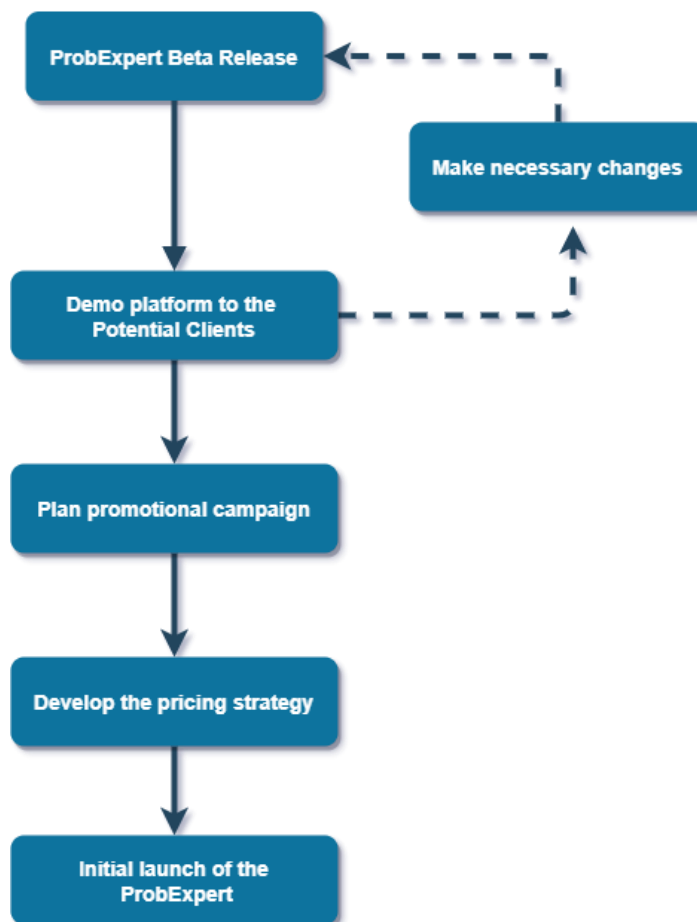


Figure 2.8: Commercialization strategy of ProbExpert

2.12.4 Display Sponsored Advertisements

Since this platform has a specific set of users, Organizations that want to promote their new technologies and services can post their advertisements on this platform. Since advertising has a negative effect on the platform's reputation, this is not a finalized strategy yet.

3 TESTING AND IMPLEMENTATION

This section describes how the system was implemented and tested under several testing techniques to minimize the issues.

3.1 Testing

For any application, testing plays a significant role in the success of the application. Quality testing procedures can reduce the number of bugs in the application and identifying them before the release is essential and cost-effective.

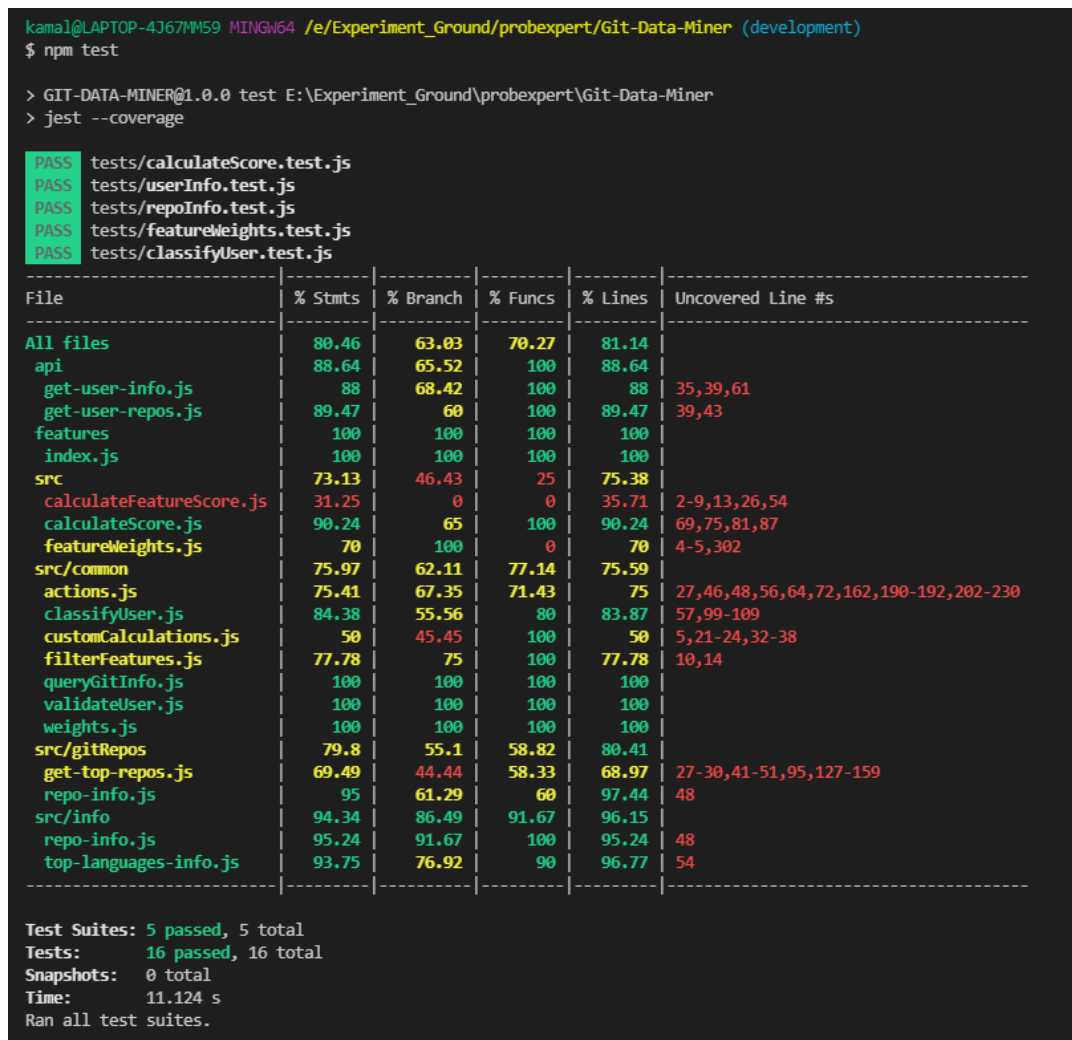


Figure 3.1: Jest code coverage results

Therefore, we introduced the Jest testing framework to our repositories to maintain the error-free codebase. Also using Jest, we can collect all the code coverage information of the entire project. **Error! Reference source not found.** shows the tested test cases and the code coverage information of the Git-Data-Miner repository.

Also, Postman API testing tool has been used to test all the API endpoints, and its collections feature was used to manage all the endpoints in one place. **Error! Reference source not found.** shows testing backend API request via the Postman application.

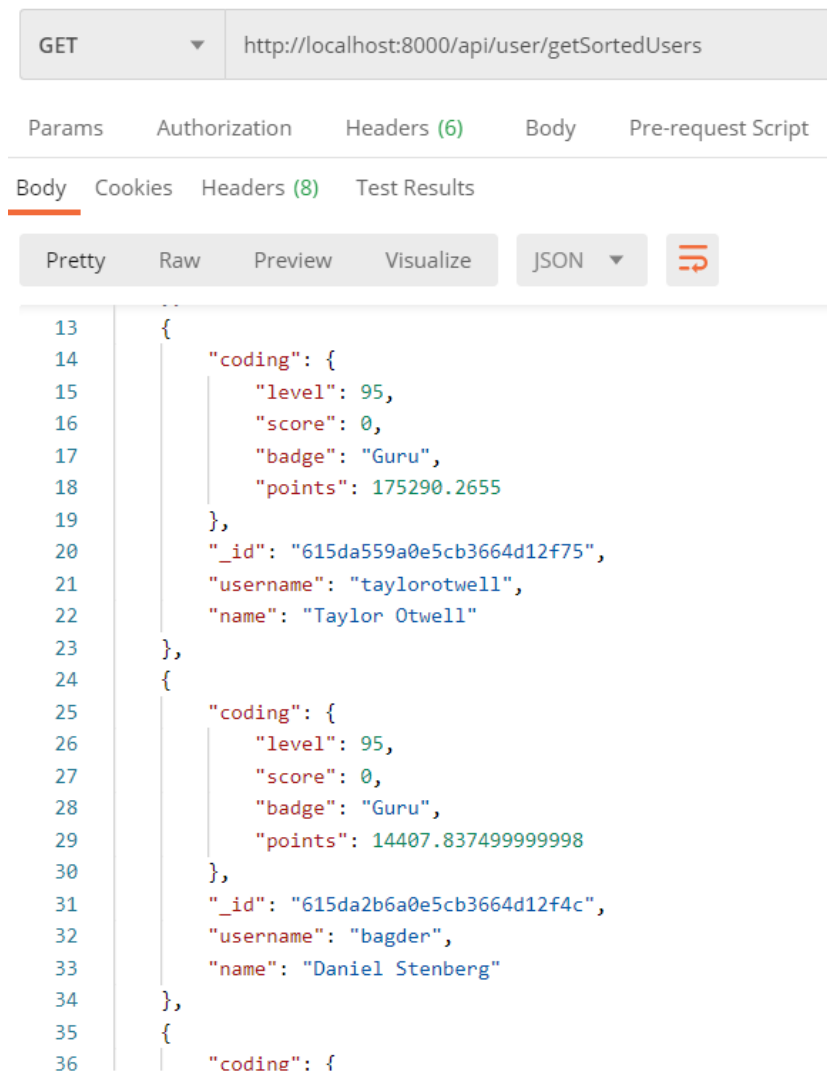


Figure 3.2: Testing API endpoint via Postman

3.2 Implementation

3.2.1 Portfolio Front-end

ProbExpert portfolio front end was developed using Next.js on top of React.js. Material-Ui component library has been used to develop the frontend components and stylings. The main reason to select Next.js to implement the frontend is that it provides server-side rendering (SSR) feature out of the box[27]. This technology has been used to render all the generated portfolio on the server side. This means that once the HTML has been delivered to the client (the user's browser), nothing else needs to happen for the user to be able to read the content on the page. This makes page loading times appear much faster to the user. SSR also make it easier to make the pages indexable and crawlable. Therefore, the web crawler can easily access all the pages for indexing and ranking them on major search engines like Google, Bing, etc. The below figure shows the user interface of an actual portfolio in our application.

3.2.2 Portfolio Back-end

The portfolio backend was developed as a RESTful API service to easily communicate with the ProbExpert frontend. Express.js library was used on top of Node.js to build the backend. Since Express.js provides a highly advanced routing mechanism, It helps to reduce the development time massively. Also, its built-in debugging mechanism helps pinpoint the exact part of the code that contains the bugs.

Since the portfolio user model consists of many data and most of them are not available for all the users, a database with a strict schema is harder to maintain. Therefore, the MongoDB Atlas database has been used as the central database of the portfolio system.

3.2.3 Web Scrappers

Two separate scrappers have been developed to scrape user information from Stackoverflow and LinkedIn. Beautiful Soup 4 python library was used as the main technology to them. Since LinkedIn block unauthorized visitors from viewing user profiles, Selenium was used to automate the Login process in LinkedIn. Since the

portfolio backend has been built on the Node server, the Node.js child process feature was used to run the python scripts.

```
_id: ObjectId("615de483d4586252b4c72358")
  coding: Object
    level: 95
    score: 0.017514309298222175
    badge: "Guru"
    points: 741.803
  > contribute_high_repos: Array
  > contribute_mid_repos: Array
  > contribute_low_repos: Array
    0: "lerrua/remote-jobs-brazil"
    1: "wesbos/awesome-uses"
    2: "prest/prest"
    3: "ossf/scorecard"
    4: "editor-bootstrap/vim-bootstrap"
    5: "openshift/osin"
    6: "ergochat/ergo"
    7: "mattn/goveralls"
    8: "golang/gofrontend"
  > individual_high_repos: Array
  > individual_mid_repos: Array
  > individual_low_repos: Array
    class: "GitHubStar"
  > git_langs: Array
    username: "avelino"
    name: "Avelino"
    avatar_url: "https://avatars.githubusercontent.com/u/31996?u=77bedf6b738458f691ee6e..."
    bio: "open source engineer"
    company: "@decodebuzz"
    location: "Kailua-Kona, Hawaii"
    created_at: "2008-10-31T14:06:07Z"
    total_prs: 553
    total_issues: 910
    total_stars: 70111
    total_commits: 11334
    contributed_to: 54
    total_repos: 170
    total_followers: 3223
    is_hireable: true
    profile_age: 12.9
    total_discussions: 44
    answered_discussions: 4
    latest_update: 2021-10-06T18:01:39.934+00:00
    __v: 0
```

Figure 3.3: Sample user document on MongoDB

3.2.4 Deployment

Since the portfolio frontend was developed using Next.js, it has been deployed on their native Vercel's cloud platform. Vercel is a deployment and collaboration platform for deploy client-side applications and web services. These servers scale automatically according to the traffic, and they have an inbuilt caching system for all the applications. Therefore, Vercel is the optimal solution to host the frontend of the ProbExpert platform.

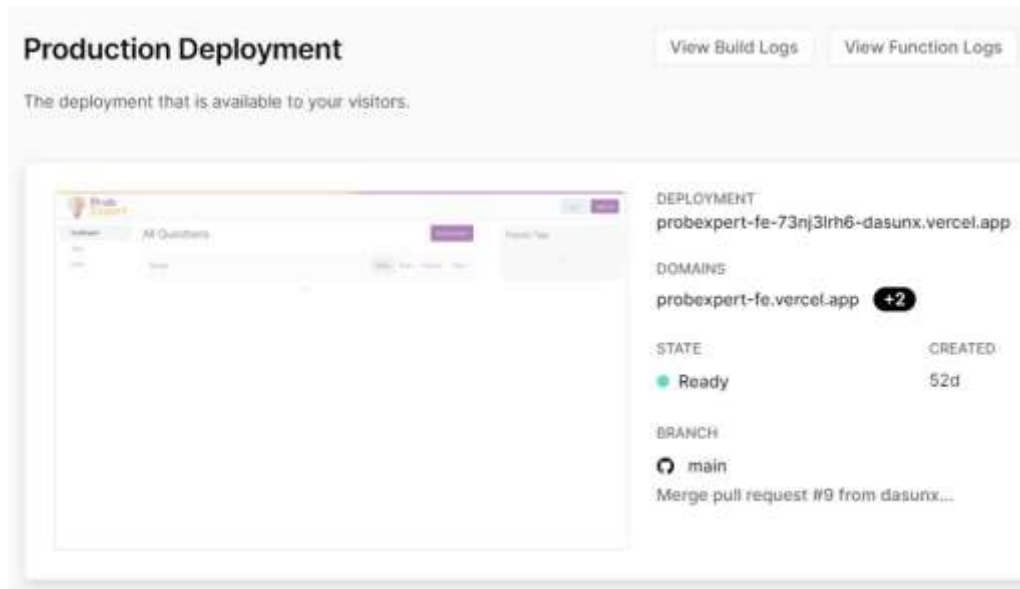


Figure 3.4: Frontend deployment on Vercel

the Git-Data-Miner tool that is used to communicate with GitHub GraphQL API has also been deployed on Vercel. Since all the methods of Git-Data-Miner were made as serverless functions, Vercel's serverless service has been used for the deployment. ProbExpert's backend has been deployed on Heroku using a free dyno. Since ProbExpert's Node.js server consisted of python scripts and Selenium, three external build packs: python, chrome-driver, and chrome-browser build pack, have been added to the dyno.

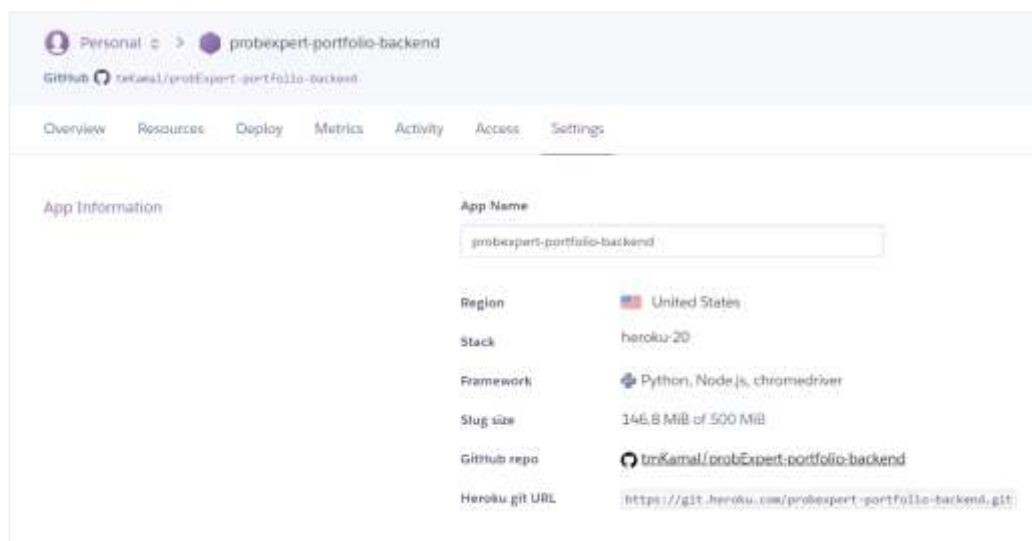


Figure 3.5: Backend deployment on Heroku

3.2.5 Telegram Bot

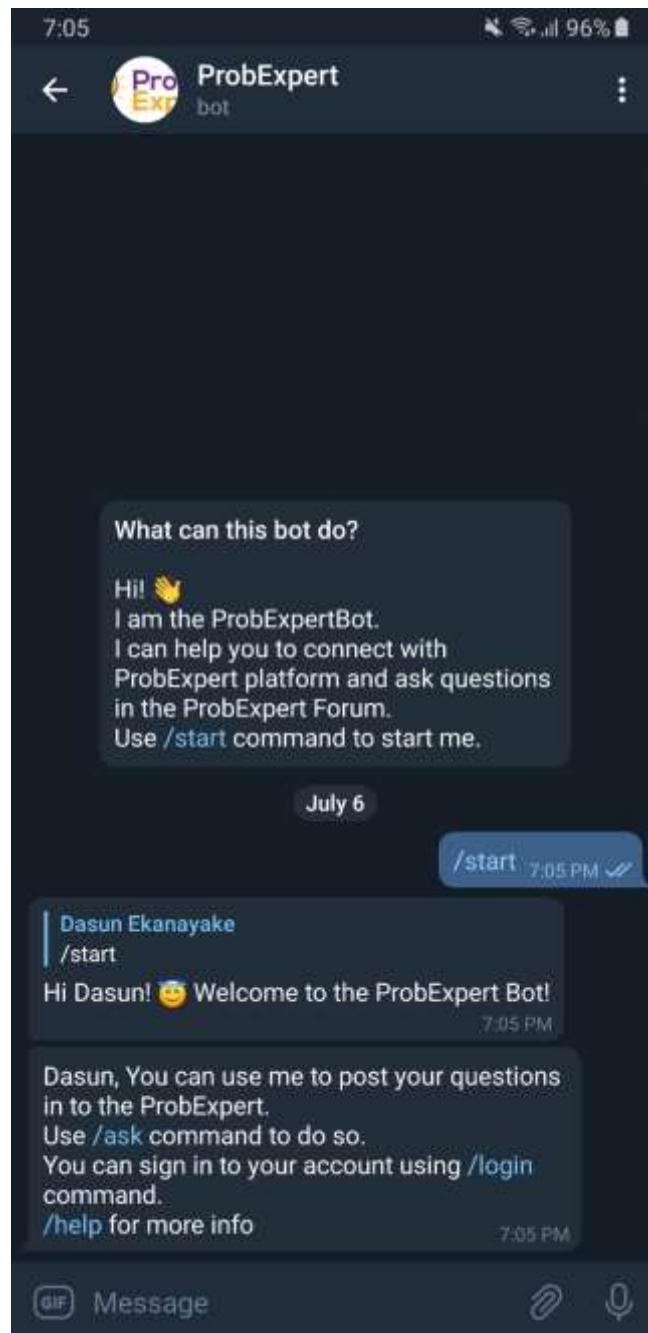


Figure 3.6: Telegram Bots sample image

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.

4 RESULTS AND DISCUSSION

This section discusses the results and findings of the ProbExpert developer ranking algorithm and automated Portfolio generation implementation.

4.1 Results and findings

This section performs a qualitative comparison of the proposed ProbExpert Q&A platform with the currently available other platforms.

4.1.1 Expert Identification

Since all the developers are listed on the leaderboard according to their rankings, hiring managers and potential buyers can find the unemployed or free-lancing developers in no time by filtering the leaderboard by using the available to hire tag.

4.1.2 CV Shortlisting

Recruiters can utilize ProbExpert's portfolio generating section to quickly generate and determine developers' scores and activities. This will save a recruiter a significant amount of time, which they would otherwise spend manually going through a developer's social networks such as LinkedIn, GitHub, and Stack Overflow while evaluating a candidate.

4.1.3 As guidance for newbies

Since developer portfolios include significant developer activities, viewing an expert's portfolio will give novices a solid idea of how they should shape their path accordingly to pursue their dream profession or career goals.

4.1.4 Automated answer

The system will identify multiple resources from all five separate resource websites (Stackoverflow, GitHub, YouTube, medium, dev) in the best-case scenario of automated answer generation. If the user were to visit those resource websites manually, they would need to view 14 web pages, which would take 9.4 minutes (5).

The average loading time for a desktop webpage is 10.3 seconds [31] ($lt = 10.3$). The average loading time for a desktop webpage is 10.3 seconds [31] ($lt = 10.3$). The user's time to determine the relevance of the resource has been calculated to be 30 seconds ($rt = 30$).

$$total\ time = \sum_{k=0}^n (lt + rt) \quad 4.1$$

Optimized answer: Instead of checking each answer individually, users may easily apply the optimized answer solution to receive the most accurate and effective answer for their problems. This process will save a significant amount of time while also ensuring a working solution to the problem.

Structured type theoretical Quizzes from platform questions: Since the platform contains questions raised by general programmers, the quizzes made by platform's questions can be used as material to update and check the current knowledge. This facility can be helpful to individuals as well as larger organizations to evaluate or train their current employees or recruits.

4.2 Discussion

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 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. When the predicted time is compared to the existing average time of generating automated answers, the percentage difference is $100 - \frac{30.90}{240} \times 100\%$ 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.

4.3 Summary of student Contribution

Table 4.1: Summary of student contribution

| Student | Functionality | Description |
|----------------------|---|---|
| Ekanayake P.M.D.P | ProbExpert Platform | <ul style="list-style-type: none"> System design Database design Backend structure design API structure design Overall UI developments Common UI component developments |
| | Automatic answer generation | <ul style="list-style-type: none"> Custom Data scraping models Information similarity models UI developments |
| | Similar question finding | <ul style="list-style-type: none"> Keyword extraction Question filtering UI developments |
| Thennakoon T.M.K.H.B | Detecting users' proficiency level along with an auto-generated portfolio | <ul style="list-style-type: none"> Serverless RESTful API to retrieve data from Github GraphQL service. |

| | | |
|----------------------------------|--|--|
| | | <ul style="list-style-type: none"> ❑ Web Scrappers to gather necessary developer's statistics ❑ Scoring model to generate a score for developers activities and behaviours ❑ Ranking algorithm to classify developers into suitable classes |
| S.K.C.W.K.M.R.T.S.B. Marapana | Structured type quiz generation for knowledge checking by utilizing the answered platform question and introduce an unbiased scoring method for evaluating answers | <ul style="list-style-type: none"> ❑ Design user interface ❑ Web scrapper for gathering questions and answers ❑ Generate user scores for quizzes |
| Ranasinghe R.M.A.K. | Answer optimization | <ul style="list-style-type: none"> ❑ Design user interface ❑ Capture data ❑ Create Training model ❑ Generate an optimized answer |

5 CONCLUSION

Due to the scarcity of data, questions with cosine similarity values greater than 80% are now considered similar; however, as data sets grow, cosine similarity values can be increased to greater than 90%. A higher cosine similarity score between questions indicates that the system will find nearly identical questions. 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. Furthermore, a blockchain system will be applied at a later stage. The primary purpose of this implementation is to introduce a digital currency for transactions within the platform, which can be used to do various tasks, such as hiring an expert for a dedicated session.

REFERENCES

- [1] B. Vasilescu, V. Filkov, and A. Serebrenik, "StackOverflow and GitHub: Associations between Software Development and Crowdsourced Knowledge," in *2013 International Conference on Social Computing*, 2013, pp. 188–195, doi: 10.1109/SocialCom.2013.35.
- [2] K. Mao, Y. Yang, Q. Wang, Y. Jia, and M. Harman, "Developer recommendation for crowdsourced software development tasks," in *Proceedings - 9th IEEE International Symposium on Service-Oriented System Engineering, IEEE SOSE 2015*, Jun. 2015, vol. 30, pp. 347–356, doi: 10.1109/SOSE.2015.46.
- [3] L. Salas-Morera, A. Arauzo-Azofra, and L. García-Hernández, "Analysis of Online Quizzes As a Teaching and Assessment Tool," *J. Technol. Sci. Educ.*, vol. 2, no. 1, 2012, doi: 10.3926/jotse.30.
- [4] B. Ross, A. M. Chase, D. Robbie, G. Oates, and Y. Absalom, "Adaptive quizzes to increase motivation, engagement and learning outcomes in a first year accounting unit," *Int. J. Educ. Technol. High. Educ.*, vol. 15, no. 1, pp. 1–14, 2018, doi: 10.1186/s41239-018-0113-2.
- [5] W. C. Kao, D. R. Liu, and S. W. Wang, "Expert finding in question-answering websites: A novel hybrid approach," in *Proceedings of the ACM Symposium on Applied Computing*, 2010, pp. 867–871, doi: 10.1145/1774088.1774266.
- [6] G. Zhou, J. Zhao, T. He, and W. Wu, "An empirical study of topic-sensitive probabilistic model for expert finding in question answer communities," *Knowledge-Based Syst.*, vol. 66, pp. 136–145, 2014, doi: 10.1016/j.knosys.2014.04.032.
- [7] A. Capiluppi, A. Serebrenik, and L. Singer, "Assessing technical candidates on the social web," *IEEE Softw.*, vol. 30, no. 1, pp. 45–51, 2013, doi: 10.1109/MS.2012.169.
- [8] S. Onoue, H. Hata, and K. I. Matsumoto, "A study of the characteristics of developers' activities in GitHub," *Proc. - Asia-Pacific Softw. Eng. Conf. APSEC*, vol. 2, pp. 7–12, 2013, doi: 10.1109/APSEC.2013.104.
- [9] E. Constantinou and G. M. Kapitsaki, "A study of the characteristics of developers' activities in GitHub," in *Proceedings - Asia-Pacific Software Engineering Conference, APSEC*, 2013, vol. 2, pp. 7–12, doi: 10.1109/APSEC.2013.104.
- [10] J. E. Montandon, L. L. Silva, and M. T. Valente, "Identifying Experts in Software Libraries and Frameworks among GitHub Users Software Developers Expertise View project Software Defects View project Identifying Experts in Software Libraries and Frameworks among GitHub Users."
- [11] L. Averell and A. Heathcote, "The form of the forgetting curve and the fate of memories," *J. Math. Psychol.*, vol. 55, no. 1, pp. 25–35, 2011, doi: 10.1016/j.jmp.2010.08.009.
- [12] H. L. Roediger, A. L. Putnam, and M. A. Smith, *Ten Benefits of Testing and Their Applications to Educational Practice*, vol. 55. 2011.
- [13] E. Constantinou and G. M. Kapitsaki, "Identifying Developers' Expertise in Social Coding Platforms," in *Proceedings - 42nd Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2016*, Oct. 2016, pp.

- 63–67, doi: 10.1109/SEAA.2016.18.
- [14] 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.
 - [15] M. Piteira and C. Costa, *Learning computer programming: Study of difficulties in learning programming*. 2013.
 - [16] GitHub, “GitHub public repository search.” [Online]. Available: <https://github.com/search?q=is:public>.
 - [17] 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.
 - [18] N. Reimers and I. Gurevych, “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks,” 2019, [Online]. Available: <http://arxiv.org/abs/1908.10084>.
 - [19] 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.
 - [20] S. Desai and G. Durrett, “Calibration of Pre-trained Transformers,” pp. 295–302, 2020, doi: 10.18653/v1/2020.emnlp-main.21.
 - [21] T. Wolf *et al.*, “Transformers: State-of-the-Art Natural Language Processing” pp. 38–45, 2020, doi: 10.18653/v1/2020.emnlp-demos.6.
 - [22] M. Grootendorst, “Keyword Extraction with BERT,” 2020. .
 - [23] B. Li and L. Han, “Distance weighted cosine similarity measure for text classification,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 8206 LNCS, pp. 611–618, 2013, doi: 10.1007/978-3-642-41278-3_74.
 - [24] D. Demner-Fushman and J. Lin, “Answer extraction, semantic clustering, and extractive summarization for clinical question answering,” in *COLING/ACL 2006 - 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, 2006, vol. 1, doi: 10.3115/1220175.1220281.
 - [25] A. Chernyavskiy, D. Ilvovsky, and P. Nakov, “Transformers: ‘The End of History’ for NLP?,” 2021.
 - [26] “The Bell Curve: Intelligence and Class Structure in American Life - Richard J. Herrnstein, Charles Murray - Google Books.” .
 - [27] A. Soranzo and E. Epure, “Simply Explicitly Invertible Approximations to 4 Decimals of Error Function and Normal Cumulative Distribution Function,” no. 2, pp. 3–5, Jan. 2012.
 - [28] L. Hussein and S. Jubell, “Hur rendering påverkar prestanda och sökmotoroptimering: – NextJS vs React,” 2021.