RepoAI: A Novel Approach Towards Automated Project Reporting

Manikiran P[1], Abhay Subramanian K[2], Kshithij R. Kikkeri[3]
[1]Department of Information Technology, Alliance University, Bangalore
[2]Department of Computer Science & Engineering, Dayananda Sagar Academy of Technology and Management, Bangalore
[3]Department of Computer Science & Engineering, BNM Institute of Technology, Bangalore

***Abstract*— A report is fundamentally a document that gives out information on a given topic in a methodical fashion. Writing the report is not as overwhelming as some may initially think, more so, for school students. A student's interaction with prevailing literature often requires perusing a substantial amount of text. In this paper, we have proposed a novel method to summarize project reports automatically so that students can perform their tasks by consulting shorter summaries instead of entire literature. This model produces summaries that are statistically better than summaries produced by existing models. We found that summaries helped the students save time and provide them a guiding light, that there was no evidence that accuracy degraded when summaries were used and that most students preferred working with the proposed model to working by themselves browsing the internet.**

Keywords: Android, Data Mining, Web Scrapping, Google NLP

I. INTRODUCTION

School education poses no lack of hurdles to students. Some of the issues faced by students can appear to be so frustrating that the students might find it impossible to cope with. Most of the times, it is the case that the students are barely expected to pull out their part of the work. However, the task might look straightforward at hindsight, a multitude of students feel that it is a laborious errand to classify the topics and writing relevant content under it in a specified format as they utterly fail to throw some light on the bird's eye view of the topic and make some generic points on the given topic before dealing with the problem statement. They become unsuccessful to give an apt problem statement; they fail to furnish a relationship between the problem statement and all of the topic sentences in the research paper; they fail to reinforce the research paper with enough statistics or facts that are relevant only to the topic of the research paper under consideration. A great deal of students faces challenges with the formulation of the research paper as they do not completely scan through and understand all the data available at the student's disposal. As a consequence of this, the choice is left to the student whether to choose a generic topic or a specific topic for the amount of content required. For example, the topic of marine pollution can be a 200-page thesis because there is a cornucopia of data available on marine pollution. Notwithstanding this, the general topic of marine pollution by itself, is extremely humongous for a 10-page research paper. Drafting any report takes a lot of time, energy, and effective organization of contents in the specified format. To prevent pullulation of key wrongdoings and maintain the naturalistic exactitude, one must devote adequate time to do research in a proper manner and jot down the findings, support one's report with adequate data

and statistics, and ensure that the report is in the correct format, within the report and in the appendices. Most students, however, fall short in these metacognitive skills (Graesser & Person 1994). Afolabi (1992) identified some of the most common problems that students have when writing a literature review, including not being sufficiently critical, lacking synthesis, and not discriminating between relevant and irrelevant materials. Hence the proposed model was developed to provide students a guiding light as to how to draft reports and the model is justified by extensive experimentation.

II. BACKGROUND

A. Artificial Intelligence (AI)

According to the father of Artificial Intelligence, John McCarthy, it is "*The science and engineering of making intelligent machines, especially intelligent computer programs*" [1]. AI is a way of instructing a computer, a computer-supervised bot, or a program to think logically, in the parallelly as the intellectual human processes his thoughts.

B. Frontend and Backend

Frontend and backend are terminologies that describe the user interfaces and program services that are work together to provide the complete final application. A frontend application is a piece of software that users interactive with directly. For example, considering login page, the form which user gets to enter, edit and submit is treated as frontend. Frontend applications typically comprise of HTML, CSS and JavaScript. A backend application or program is generally a script or piece of code that runs without the user knowledge to provide support of the front-end services, typically by residing closer to the key resources such as database and can directly communicate with the required resources. Backend is a program that maybe invoked directly by the frontend or indirectly with the help of intermediate programs. Backend is built using languages such as JSP, PHP or Python.

C. Hybrid Application

Hybrid applications are generally a set of web scripts or programs running within a a browser shell of the application which has privileges to utilize the native features of the mobile, such as camera, notifications and storage. Hybrid applications have a lot of advantages compared to pure native applications such as wide range of platform support, reduced development time, and support for using third party libraries or scripts.

D. Ionic framework

According to the Ionic Official website, "*Ionic Framework is an open source UI toolkit for building performant, high-quality mobile and desktop apps using web technologies (HTML, CSS, and JavaScript)*" [2]. It is a mobile application development framework targeted at building hybrid mobile apps.

E. Database

A database is an assortment of data that is systematized so that it can be effortlessly retrieved, managed and restructured. There are multiple types of databases, such as SQL, MySQL, NoSQL, MS Access, MongoDB, etc.

F. API

An application programming interface (API) is a set of communication rules and commands that facilitate in building applications. In general, an API is used to provide unified and simple access to resources, either belonging to the same project/organization or any other third party provided resources.

G. The Cloud Natural Language API

Google Cloud's Natural Language API provides users with the structure and context of original text using a powerful pretrained machine learning models through a simplified endpoint or REST API. The documents are classified into general categories such as news, technology and entertainment. It facilitates features like entity recognition, sentiment analysis, entity sentiment analysis and other text annotations to users of the API.

III. PREVIOUS WORK

Though several intricate and absorbing models of project reporting structure presently exist, a persistent issue has been how to identify the text present in a given website in the report of a particular model in a new piece of writing. In the field of discourse analysis, researchers have generally resorted to using trained people who rate or specialist informants from the target field; this is a time-consuming process [4], [5], [6]. inexperienced readers in the classroom, on the other hand, need to gain a more immediate view of the different structural moves used in a text. Similarly, if novice writers are able to monitor the structural changes in their own compositions, they can make corrections to the flow of a text where necessary [7] There are several research papers on automatic bug reporting and unsupervised bug reports categorization. Also, there are many research papers on automatic report generation for smart office/inventory management. Boris et al have come up with a model which automatically generates project reports which can be edited and stored in a SQL database. They are useful for project managers.[8] Zalte et al have proposed an automatic question paper generator which is streamlined and secure and is mostly inclined toward academia and hence found its mention here.[9] Most of the existing systems focus on the industries and none on the academia. Hence, we have developed a system that concentrates on the students need for report generation which will save students' time and energy to a large extent.

IV. IMPLEMENTATION

A. *Backend*

The development and deployment of backend part is done in a number of steps which is described in great detail

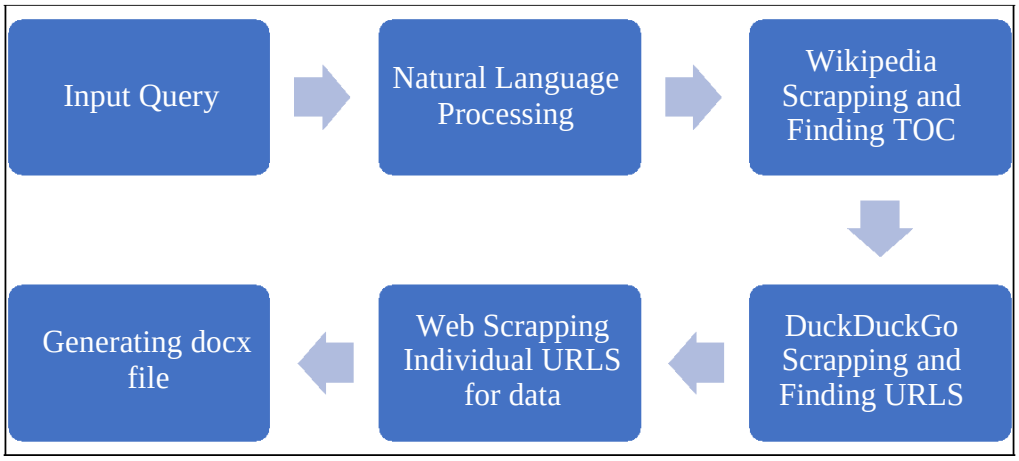
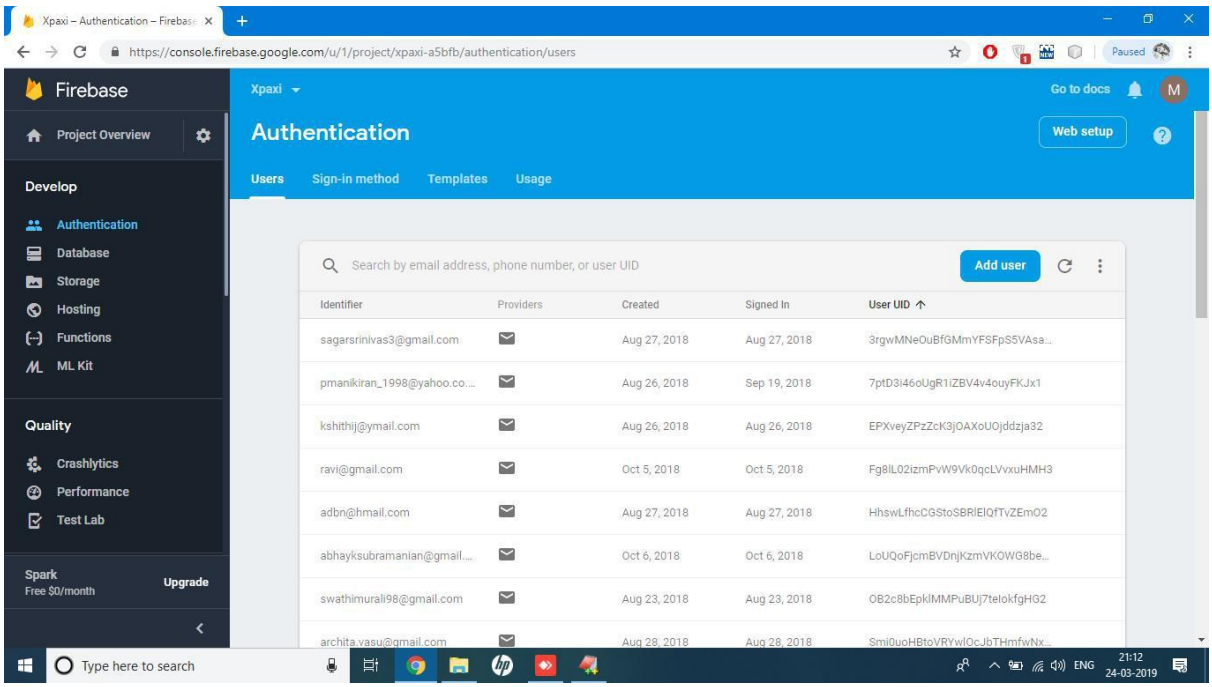


Fig 1. Backend Workflow

1. Login and Signup

The data entered by the user is filtered to check for SQL Injections and then passed on to the FireBase API. The response from the FireBase API is sent back to the user through our API.



2. Google Scrapping

The backend development starts with the usage of Python. Some background checks are done at this stage. If an article on the input query is present on Wikipedia, then it jumps to step 3. Else, a predefined set of topic headings are employed namely [“Introduction”, “Cause”, “Effect”, “'Precautions”, “Applications”, “Conclusion”]. Subsequent Google searches will be performed on the predefined headings. Taking an example of ‘Marine Pollution’ as the input string, Google searches on “Marine pollution introduction”, “Marine pollution causes” etc., will be performed.

3. NLP & Wikipedia Scrapping

Google Cloud Natural Language Processing from the “google.cloud” library is used to extract the main topic query from a set of long input query. Instead of having predefined topics, a Wikipedia search is performed to check if there is any related topic, and scrapped the TOC (Table of Contents) to get the topics for the particular query. Then the search is run similar to step 2.

The input query which is received by the API backend script is cleaned by using NLP for filtering out unwanted words and leaving behind only essential keywords. These keywords are used to find related Wikipedia pages, if any, and the Table of Contents of that particular Wikipedia page is scrapped and a list of necessary headings again after cleaning and filtering is returned back to the user.

4. DuckDuckGo Scrapping

Initially, Google Custom Search API was employed for fetching query result links, but due to the limitation of 100 query searches per day, we moved on to DuckDuckGo scraping . The searching efficiency was improved by using DuckDuckGo instead of Google, as Google blocks bots which try to search a lot of queries within a short period of time. The URLs from the scraped results were refined by filtering out the ads, and giving more priority to popular websites likes Wikipedia and other similar websites.

5. Individual URL Scrapping

Since each website has a different html DOM structure, it is really hard to scrape using same technique. Alternately, the route chosen was to grab the text from the entire webpage. Now a linear search algorithm is run to find the match of text from the DuckDuckGo result summary. From the initial position the matching text found to the required length of string/paragraphs, that is specified by the user in the app, is scrapped.

6. Using Flask & Generating .docx file

Once this functionality is complete, “Flask” library of Python is incorporated to implement a real-time backend server/API endpoint. Finally, the data is collected and organized which was scrapped from the WWW, and is exported into docx file format, using the Python library

"docx". This file is saved in the server, and also the path to this file is saved in database, for users to see their past generated documents.

B. *Frontend*

1. First Stage – Ionic

A hybrid app was built using Ionic framework. Ionic framework is built over Cordova to generate apps for both Android and iOS. The user gets a login and signup options, for storing previous search results for future purposes. The user has in his prerogative to do a part of the report on his phone and the other part on his desktop. This exhibits a macaronic charm and has an edge over other similar applications.

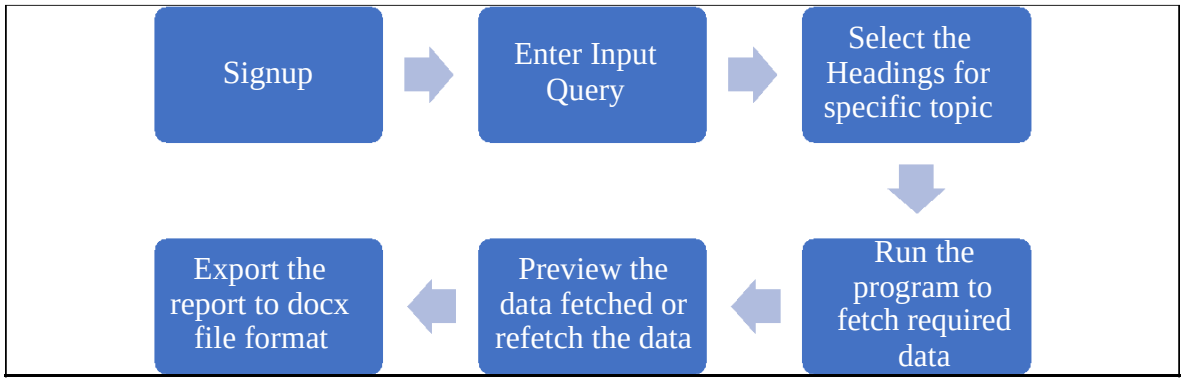


Fig 2. Frontend workflow

2. Input Query

Prima facie, the user enters any search query in the given text field. A new FormData object is created including the search query which is sent as a HTTP POST request to the API which is located at http://api.repoai.ml. The response thus obtained is a list of heading which is further filtered and processed using our cleaning techniques including NLP.

3. Selecting the Headings for specific topic

The user is displayed with all the topics fetched from the API request in a list format with option to select, add, edit and remove them. The user can add new heading(s) of his/her choice if they find it missing from the Wikipedia TOC and our predefined headings set. They can also edit the headings, if they find any mistakes or irrelevancy.

4. Generating Data and Reviewing it

Once a user selects any particular heading, the data gets fetched automatically and is shown to the user to check for mistakes or do any modifications. If he/she feels that the content is irrelevant, they can click regenerate option to fetch data from different sources. The same procedure is followed with rest of the headings. The user is also provided with the option to reorder the headings to match his/her requirements.

5. Exporting Report

The user can click on Generate Report button to select the file format he/she desires to export it to, either PDF or Docx file format. Once they click on a particular file format, the request is sent to the API along with the entire data, and a path to the generated file is returned back. The app uses this path to download using the default download manager.

6. Solving local storage issue

A plugin called "cordova-sqlite-storage" was used which helps in maintaining an SQLite database in a persistent storage location (phone storage).

7. Saving Recent Searches

An algorithm was written to save N recent searches on local storage, similar to that of LRU (Least Recently Used) cache which is a cache that, when low on memory, evicts least recently used items. LRU is an eviction policy that makes a lot of sense for the typical kind of cache we all deal with on a daily basis. [10]

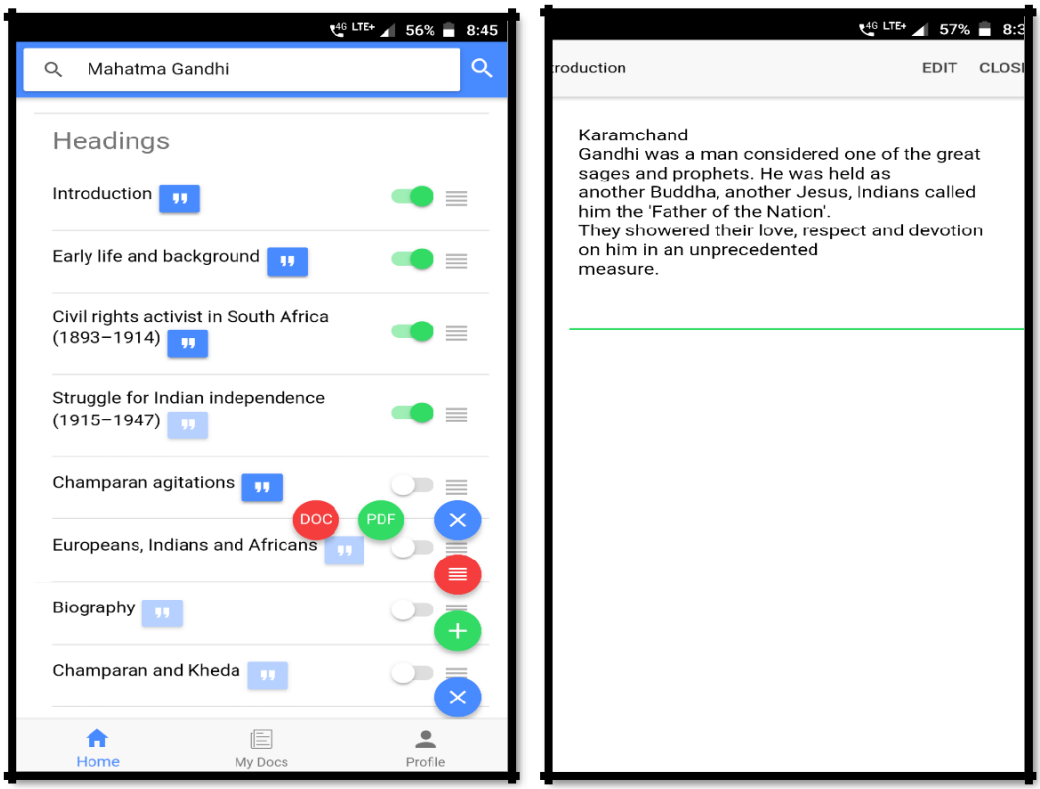


Fig 3. Screenshots

V. CONCLUSION AND FUTURE ENHANCEMENTS

This project has an immense scope in the coming days as there are a lot of functionalities and modules which can be incorporated into it, but the scope of the project is limited due to various constraints. Some of the areas in which there can be enhancements are listed below

1. Addition of word limit to the report so that the student can comply with the word limit given to him by the teacher.
2. Addition of images in the report so that it becomes more presentable and get a clearer picture of what the report is conveying.
3. Implementation of figures and statistics which help increase the readability and gives an idea about the past, present and future scenarios of the topic under consideration.
4. Support of multiple formats prescribed by different institutions/organizations like IEEE, IJERT etc, also formats prescribed by different universities for project reports.
5. Google is working hard to provide cloud services that facilitate human-computer interaction through tools that are able to consume human language like GCP (Google Cloud Platform), hence it is provides a path for wider enhancement of user experience thereby increasing the student's usage of the app.

ACKNOWLEDGMENT

Many professors have mentored and guided us during the course of developing our application. Their input was extremely useful in the subsequent development of the application and without their help it would have been an uphill task. Several people have given us constructive feedback on different modules of the application. Their help is hereby sincerely acknowledged. We now understand why family members are unfailingly mentioned in this segment. Without our families' love and support, this project wouldn't have been successful. Many thanks to them for being our pillars of support.

REFERENCES

[1] https://www.tutorialspoint.com/artificial_intelligence/artificial_intelligence_overview.htm

[2] https://ionicframework.com/docs/intro/concepts

[3] J. M. Swales, Genre Analysis: English In Academic and Research Settings. Cambridge, U.K.: Cambridge Univ. Press, 1990.

[4] L. Anthony, "Characteristic features of research article titles in computer science," IEEE Trans. Prof. Commun., vol. 44, pp. 187–194, Sept. 2001.

[5] L. Anthony, "Writing research article introductions in software engineering: How accurate is a standard model?," IEEE Trans. Prof. Commun., vol. 42, pp. 38–46, Mar. 1999

[6] Laurence Anthony AND George V. Lashkia, "Mover: A Machine Learning Tool to Assist in the Reading and Writing of Technical Papers" IEEE TRANSACTIONS ON PROFESSIONAL COMMUNICATION, VOL. 46, NO. 3, SEPTEMBER 2003

[7] Boris Veytsman, Maria Shmilevich: Automatic report generation with Web, TEX and SQL. TUGboat, Volume 28 (2007), No. 1 — Proceedings of the Practical TEX 2006 Conference

[8] Zalte S.V., Jadhav C.C, Mangire A.A, Hole A.D.4 and Tulshi A.R: Automatic Question Paper Generator System, International Journal of Advanced Research in Computer and Communication Engineering Vol. 7, Issue 3, March 2018

[9] https://www.openmymind.net/Writing-An-LRU-Cache/

AUTHORS

Manikiran P is currently a student at the department of Information Technology, Alliance University, Bangalore, India. His research interests include Natural Language Processing and Machine Learning. He can be contacted at pmanikiran1998@gmail.com

Abhay Subramanian K is currently a student at the department of Computer Science & Engineering, Dayananda Sagar Academy of Technology and Management, Bangalore, India. His research interests include Game Theory and Machine Learning. He can be contacted at abhaysubramaniank@gmail.com

Kshithij R. Kikkeri is currently a student at the department of Computer Science & Engineering, BNM Institute of Technology, Bangalore, India. His research interests include Natural Language Processing and Machine Learning. He can be contacted at kshithij@ymail.com