EDUEASY – SMART LEARNING ASSISTANT SYSTEM

Pavani Demindie Gallage

IT17407458

BSc. (Hons) Degree in Information Technology

Specializing in Information Technology

Department of Information Technology

Sri Lanka Institiuete of Information Technology

Sri Lanka

Septrmber 2020

EDUEASY – SMART LEARNING ASSISTANT SYSTEM

2020-009

Final Project Thesis

Pavani Demindie Gallage

IT17407458

Supervisor – Dr.Anuradha Karunasena

BSc. (Hons) Degree in Information Technology

Department of Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

September 2020

DECLARATION

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

Name	Student ID	Signature
P.D.Gallage	IT17407458	

The above candidates are carrying out research for the undergraduate Dissertation under my supervision.

Signature of the supervisor Date

…………………………… …………………………

(Dr. Anuradha Karunasena)

Abstract

The usage of smart learning concepts has been increased rapidly all over the world as better teaching and learning methods. Most of the educational institutes such as universities are experimenting those concepts with their students. In the university lecture room environment, the lecture method is the most popular method of teaching. In the lecture method, the lecturer presents the content mostly using lecture slides and the students make their notes based on the content presented.

Referring the relevant reference materials is very important to all the students when they aim for the best grade in any subject. It helps to understand difficult lessons, formulas, theories, etc. The lecturer can’t teach everything regarding a lesson within the two-hour lecture time. So, students should search for more information relating to lessons by themselves. Most of the students skip the process of understanding lessons by referring to references and they memorize the difficult lessons as it is due to the limited time that they get during exam periods.

Developing an automatic reference finder is a good way to overcome the above problems and to support the students by finding exact reference materials relevant to the given content automatically. The purpose of this research is to create an effective and efficient tool for automatically search reference materials on the internet according to the selected keywords or keyphrases. Furthermore using content similarity algorithm the tool can generate a similarity value by comparing the original text with the reference text and the student can get an idea the web results are how much similar to the original text. Finally, the student can refer to the most relevant and rated web references using the Reference Finder.

Keywords - Smart learning concepts, Automatic reference search, Reference Finder, Content similarity, keyword extraction

ACKNOWLEDGEMENT

Working on this project on “EduEasy - Smart Learning Assistant system” was a source of immense knowledge for us. We would like to express my thanks of gratitude to all those who provided me the possibility to complete this project. A special gratitude we give to Dr. Janaka Wijeykoon, the Lecture-in-charge of Comprehensive Design and Analysis Project, Dr. Anuradha Karunasena, our project supervisor & Ms. Pradeepa Bandara, our co-supervisor, whose contribution in stimulating suggestions and encouragement helped us to complete the research project.

The guidance and support received from our families, friends, and fellow group members were more helpful to complete the project successfully. Furthermore, we would like to acknowledge all the others who offered their full support and encouragement for the completion of this research project.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

LCA	Lowest Common Ancestor
CSUN	California State University Northridge
IDE	Integrated Development Environment
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
WSGI	Web Server Gateway Interface

1. INTRODUCTION

1.1. Background Literature

Smart learning may be a broad term for education in today’s digital age. It reflects how advanced technologies are enabling learners to digest knowledge and skills more effectively, efficiently, and conveniently [1]. Smart Learning or intelligent education includes new educational contexts and the experts are thinking that it is important to focus on the student’s use of technology at their fingertips. It does not only depend on the software and hardware available but on how they are using those concepts and techniques in the classes. With the increase of popularity of using new technical devices, educational institutes have begun to adopt modern teaching methods to their lecture room environment. Most of the researchers who study, write, and invent new concepts about smart learning have dedicated their time to make popular those smart learning concepts among schools and other educational institutions such as universities. Globally many countries have launched projects focused on smart learning and everyday researchers have sought out new technologies, concepts, and methods to improve smart learning. Many people still find the concept unclear, although the ongoing global movement towards smart learning marks an important transformation in modern learning.

Society is normally resistant to change. If a person comes up with a weird or unique concept that is totally different from normal thinking, the majority of society makes that person ridicule. Though that new concept makes a huge benefit to society, they are not going to accept it as it is and this thinking pattern of a typical person continues from history to today without any changes. Since smart learning is in relatively growing stages, it faces several barriers from society. The main challenge is that smart learning goes against the non-flexible nature of traditional education. Furthermore, insufficient time to learn e-learning tools and implement them into teaching practice, lack of recognition, teachers’ low self-efficiency in educational technology knowledge, shortage of knowledge, and motivation among students [2] are other barriers that occur when it comes to implementing smart learning concepts inside the educational institutes.

Rather than the decisive task of managing learning, today’s institutions are more focused on managing personnel, buildings, and finance. The academic community and the students should embrace smart learning as a solution to many problems. For this solution to become effective, both teachers and students need to change how they recognize learning, and accept that traditional education simply cannot keep up with today’s rapidly changing world. The way the entire higher education system works should be change for the modern learner to gain more benefits from it. Smart learning should be flexible for all, regardless of age, religion, gender, physical abilities, and other groupings.

Recently the education sector has seen a detectable increase in educational standards, advancement, and innovations. Schools and other educational institutions have begun to adopt modern teaching methods through interactive intelligent boards, projectors, and smart notebooks, etc. These technological updates make the education system more interactive and easier. Students like better to use tablets, phones, laptops, and other devices to share and transfer study material that saves their time and effort. Not only that, but this smart technology also helps teachers to show boring lectures into an attractive session. It is not almost sharing information but also creating a far better online presentation by avoiding the necessity to write down manually.

Besides the above benefits, smart learning concepts enhance the learning experience and it creates an interactive learning experience in the classroom. Smart learning gives easy access to online resources, increases productivity, simplified teaching, cost reduction, and it saves time for both students and lecturers [3].

In the university lecture room environment, the lecture method is the most popular method of teaching. In the lecture method, the lecturer presents the content mostly using lecture slides and the students make their own notes based on the content presented. The majority of the students are not keen to make their own notes or they rely only on the lecture slides. There are several issues when it comes to a traditional lecture room environment. The students’ notes may not be complete due to the teacher’s accent and the speed of speech. Most of the students skip the process of referring the relevant reference materials because it is difficult to find suitable references. The majority of the students struggle to find practice questions relevant to the course contents because of their tight schedules and it is difficult to search suitable answers for different questions regarding particular lessons. The students waste their time seeking the appropriate lecture slides while referring to the lecture note for short notes.

“EduEasy Samart Learning Assistance System” is introduced to overcome the above lecture room environment’s problems. There are four main components included in EduEasy Samart Learning Assistance System that have addressed four different main problems students face during the lecture room. The first component is “Note Taker”. It records the lecturer’s voice into an audio file and transcribes into a text file as the first part of the component. In the second part of the component, the generated text file is summarized into a clear short note. The second component is “Reference Finder”. It finds exact relevant reference materials automatically according to the summarized note and it also generates a content similarity value for comparing how much similar the content of reference materials to the content of the summarized note. The third component is “Question Presenter”. It creates simple WH questions according to the summarized note. Wh- question usually starts with a word beginning with wh-, and "how" is also a wh-question type. What, when, where, who, whom, which, whose, why, and how are normally used to begin wh-questions [5]. It also generates the appropriate answers to those wh-questions. The fourth component is “Slide Matcher”. It navigates the students to the correct lecture slides according to the different parts of the summarized note.

The Reference Finder is developed to overcome the difficulty of finding exact relevant reference materials automatically according to the summarized note. Keyword extraction is the most used technique when developing the reference finder component. Some various researches and studies had been done to identify the techniques to extract keywords or keyphrases from documents or web pages.

One of the researches [6] had explained how to use LCA-based keyword search for effectively retrieving “information unit” from web pages. The authors had discussed the problems of identifying the “Information Unit” of relevant pages containing all the input keywords as the answer. So they take a set of most related web pages as the search target and then model it as a tree structure. For each set of web pages that contain all the given keywords, the smallest and most meaningful subset is selected as a whole to answer the query. Then they come up with an LCA-based (Lowest Common

Ancestor) algorithm to identify the most compact subset of the web pages containing all the keywords**.**

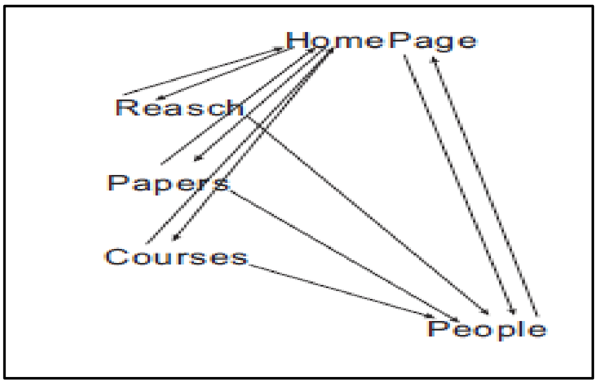


Figure 1.1.1: Link relation inside a small Cluster

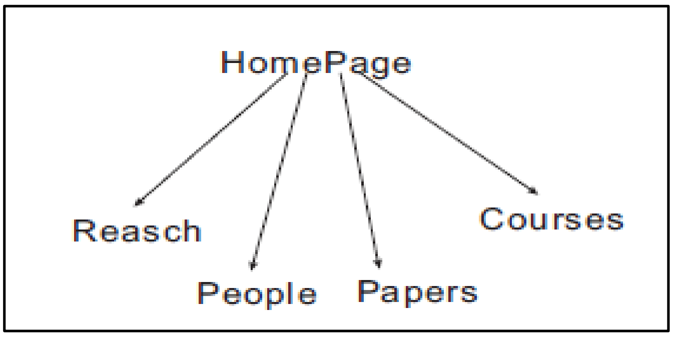


Figure 1.1.2: Breadth-First Traversed Tree

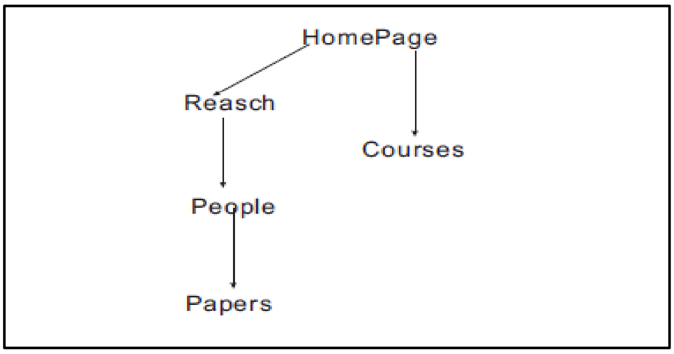


Figure 1.1.3: -Depth First Traversed Tree

There are two ways to traversing the Cluster respectively are Depth First traversal (DFT) and Breadth-First traversal (BFT). Figure 1.1.2 and figure 1.1.3 show the output of the two methods traversing the Cluster shown in figure 1.1.1. Since they store the Dewey Code of each node in the server, the BFT method will cost less space than the DFT method, as a result, the former method is preferred. Dewey Code is based on

Dewey Decimal Classification developed for general knowledge classification. With Dewey Code, each node is assigned a code that represents the path from the root of the tree to the node. Each component of the path represents the local order of an ancestor node. They have used two algorithms for this process. In the first algorithm, they have presented how to model a tree from cluster and encoding with Dewey Code. In the second algorithm, they have presented the LCA-Based algorithm of retrieving "Information Unit" from a Cluster.

The authors of "Web Information Extraction and classification using Vector Space Model Algorithm" [7] research had done based on text mining and the VSM algorithm used for text classification has more accuracy, better applicability, and less manual interference. Text mining is the technique that helps users to seek out useful information from an outsized number of digital text documents on the online or databases. a primary step toward any Web-based text mining effort would be to gather a big number of Web mentions of a topic. Thus, the challenge becomes not only to seek out all the topic occurrences, but also to filter just people who have the specified meaning. This paper discusses the algorithm of the way to follow the appointed website or website consistent with users request and obtains data of particular homes in Internet by extraction on web mining. The VSM algorithm used for text classification has more accuracy, better applicability , and fewer manual interference.

R. Lawson [8] explains how to apply web scraping techniques using the BeautifulSoup library [9] to extract content from web sites easily. Furthermore, one of the research [10] had conducted about web scraping explains how the web scraper draws the desired links from the web, and then the data is extracted to get the data from the source links and finally stowing that data into a CSV file.

Comparing two documents can be done in many ways. Re-processing data is very important before the comparison. Natural Language Processing (NLP) is a part of computer science and AI which deals with human languages. One of the researches [11] done about Natural Language Processing algorithms described deeply about the functions of NLP and how it can be used to re-process the data. There are many functions in NLP which help to re-process the data. The study [11] has explained about the main steps of re-processing data using NLP. Get the data in clean and standard format for further analysis is very important. Tokenization, stop words, and stemming are mostly use for data re-processing. Tokenization can break a complex sentence into words, understand the importance of each word to the sentence, and produce a structured description on an input sentence. J. J. Webster and C. Kit [12] have explained the significance and complexity of tokenization, the beginning step of NLP. Notions of word and token has been discussed and defined from the viewpoints of lexicography and pragmatic implementation, respectively

One of the researches [13] explains a solution using K-mean algorithms and Hierarchical Clustering Dendrogram. Here compared two sentences should be converted into vectors first and there are methods explained "Count Vectorizer method" and "Word Embeddings". Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space that measures the cosine of the angle between them [14]. This cosine similarity algorithm is mostly used for comparing two contents and finally, it generates a similarity value using vectors. When comparing documents this similarity value is more important to get an idea about how similar the two documents are. Jaccard similarity algorithm and cosine similarity are mostly used in the EduEasy Smart Learning Assistance System to compare the content similarity between the lecture note and the lecture slides and generate similarity value by comparing online references and the lecture note.

1.2. Research Gap

Table 1.2.1: Comparison of existing products

	SlidePlayer	CSUN | Oviatt Library	**EduEasy Reference Finder**
Search results in any place on the internet	✗	✗	✓
Show user-targeted recommendations	✓	✓	✓
Search every type of reference materials	✗	✗	✓
Compare the similarity of the content	✗	✗	✓

There are some tools and software that are made to find some reference materials. They have different functionalities compare to the EduEasy Reference Finder. According to Table 1.2.1, the SliePlayer application only provides presentation slides as the output when searching for reference materials. On the other hand, the CSUN Ovit Library only provides e-books and pdf. Both of the applications can not search for references from anywhere on the internet. But both applications show user target recommendations. They have only a limited database for finding references. And also both of the applications do not present every type of reference material and they can not generate the similarity value by comparing two documents as the Reference Finder does.

The Reference Finder can find any type of reference material automatically and it can be searched anywhere on the internet. And also it shows user target recommendations in separate tabs in the search engine. As the most important point, the Reference Finder can generate a similarity value. The above facts show that there is no such application that has all the functionalities similar to the Reference Finder application as one product.

1.3. Research Problem

Motivate students to study smart instead of study hard by using advanced technologies and smart learning concepts is a good way to be successful in their education. Studying smart is a challenge without correct support and the Reference Finder can support the students by finding exact reference materials relevant to the given topic automatically.

Referring the relevant reference materials is very important to all the students when they aim for the best grade in any subject. It helps to understand difficult lessons, formulas, theories, etc. The lecturer can't teach everything regarding a lesson within the two-hour lecture time. So, students should search for more information relating to lessons by themselves.

Most of the students skip the process of understanding lessons by referring to references and they memorize the difficult lessons as it is due to the limited time that they get during exam periods. Sometimes it is hard to find exact relevant reference materials according to the lecture note in a short period. Some students prefer to watch videos to understand difficult lessons rather than reading articles or pdf. Because of that it more important to search for different kinds of reference materials such as YouTube videos, pdf, articles, papers, and books in one go as well. Referring to the reading materials such as articles, pdf, and papers is time-consuming, and as well as it is difficult to determine how much similar the reference material to the lecture note. So there should be a way to choose the most appropriate article to make the time more meaningful than wasting.

Developing an automatic Reference Finder is a good way to overcome the above problems. The Reference Finder has two functionalities to provide solutions for two main obstacles that occur when finding references.

1.4. Research Objectives

As a whole product including four main components with more than eight functionalities, the main objective of the EduEasy Smart Learning System is to develop an E-learning application for students to effectively learn and revise lectures done at the university. In order to reach the main objective of the EduEasy Smart Learning System, the specific objectives of each component need to be fulfilled accurately. When it comes to the Reference Finder component it has two main objectives.

The first objective is to find the exact reference materials automatically on the internet according to the selected keywords or keyphrases. The keywords are extracted from the summarized note that is generated in the Note Taker component as the output. Students can refer to the references that appear in separate google tabs and the results appear as e-books, articles, papers, and YouTube videos.

The second objective is to generate a similarity value by comparing the original summarized note with the reference material. The similarity between the two contents is increased when the similarity value is higher. So the student can get an idea the web results are how much similar to the summarized note. It makes students easier to choose a good reference material among all the suggested references.

In addition to the above main two objectives, the Reference Finder saves students' time and effort to find reference materials on the internet. Finally, it helps students to achieve successfully their academic goals.

2. METHODOLOGY

2.1. Methodology

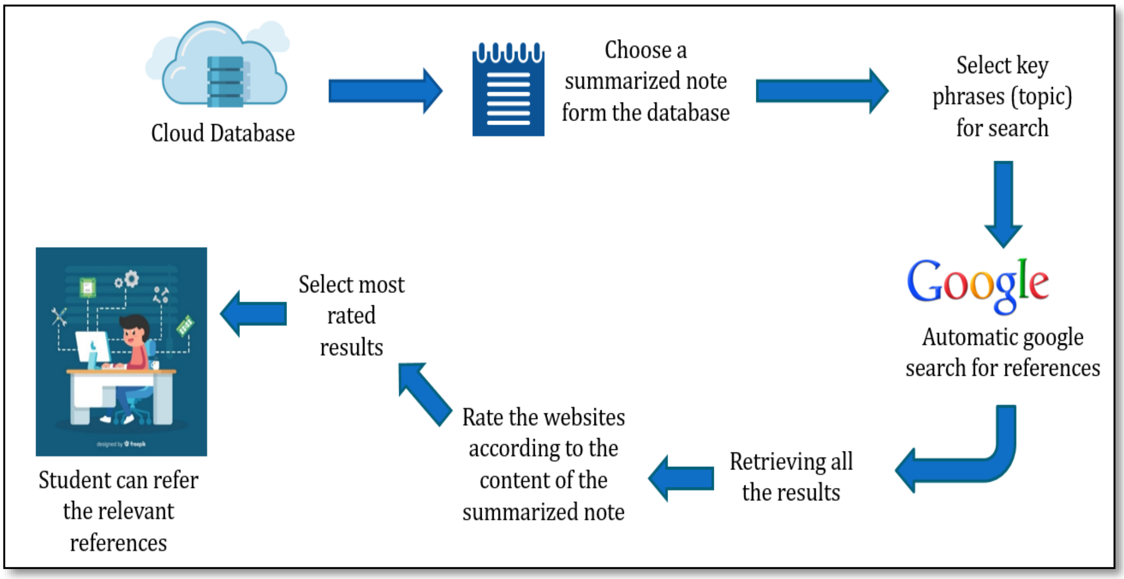
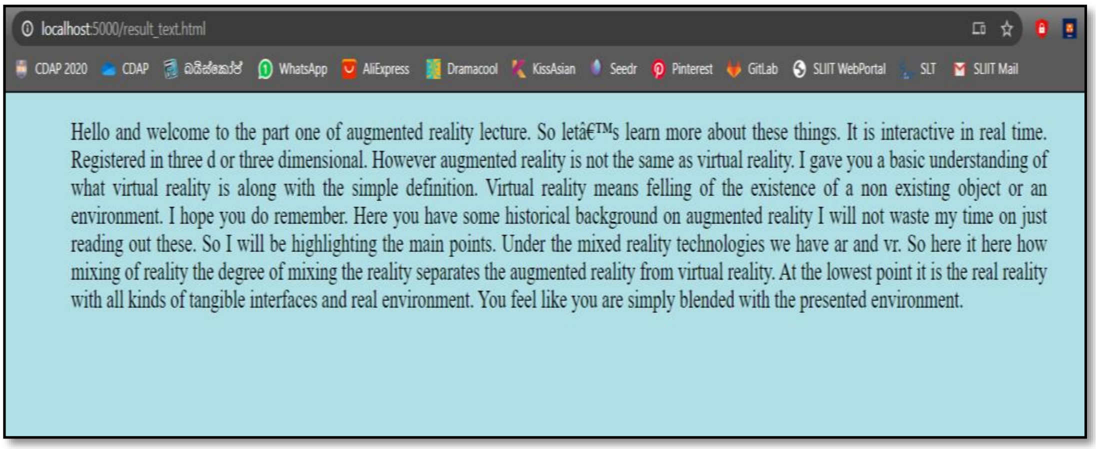

Cloud Database
Choose a summarized note form the database
Select key phrases (topic) for search
Google
Automatic google search for references
Retrieving all the results
Rate the websites according to the content of the summarized note
Select most rated results
Student can refer the relevant references

Figure 2.1.1: The system flow diagram of Reference Finder component

Figure 2.1.1 demonstrates an overview of the Reference Finder component. The automatic reference finding process begins after the student select a summarized note which is stored in the system database. Then the Reference Finder extracts the keyphrases from the summarized note and automatically searches the references according to the keyphrases. The references appear in the separate tabs in the search engine and the results contain YouTube videos, web articles, e-books, pdf, etc. Then using one of the reference materials that appear in the search engine, the system compares the content similarity between the summarized note and that web reference. The choosing reference should not be any kind of video material. After that, the similarity algorithm generates a similarity value and the student can get an idea the web results are how much similar to the summarized note. Finally, the student can refer to the most relevant and rated references using the reference finder component.

Figure 2.1.2: Tools and technologies

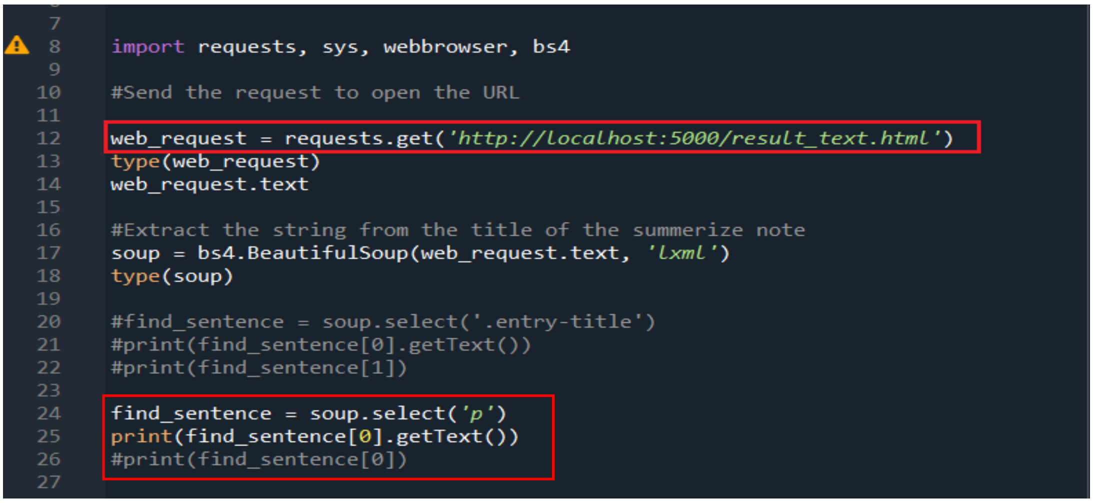
Figure 1.2.2 depicts the tools and technologies used to develop the Reference Finder component. Python version 3.7 is used to develop the whole Automatic Reference Finder application. All the python implementation is done by using "Anaconda Spyder" IDE. Spyder (Scientific Python Development Environment) is a free integrated development environment (IDE) that is included with Anaconda. And also It includes editing, interactive testing, debugging, and introspection features. When developing the web application "JetBrains PhpStorm 2017.2" IDE is used along with HTML, CSS, and Bootstrap. "Flask" is a lightweight WSGI web application framework and it is used to connect the web application with the back-end python functions. "GitLab" is used to manage the EduEasy project and its source files, as they are changing over time.



Hello and welcome to the part one of augmented reality lecture. So letâ€™s learn more about these things. It is interactive in real time. Registered in three d or three dimensional. However augmented reality is not the same as virtual reality. I gave you a basic understanding of what virtual reality is along with the simple definition. Virtual reality means felling of the existence of a non existing object or an environment. I hope you do remember. Here you have some historical background on augmented reality I will not waste my time on just reading out these. So I will be highlighting the main points. Under the mixed reality technologies we have ar and vr. So here it here how mixing of reality the degree of mixing the reality separates the augmented reality from virtual reality. At the lowest point it is the real reality with all kinds of tangible interfaces and real environment. You feel like you are simply blended with the presented environment.

Figure 2.1.3: The input of the Reference Finder

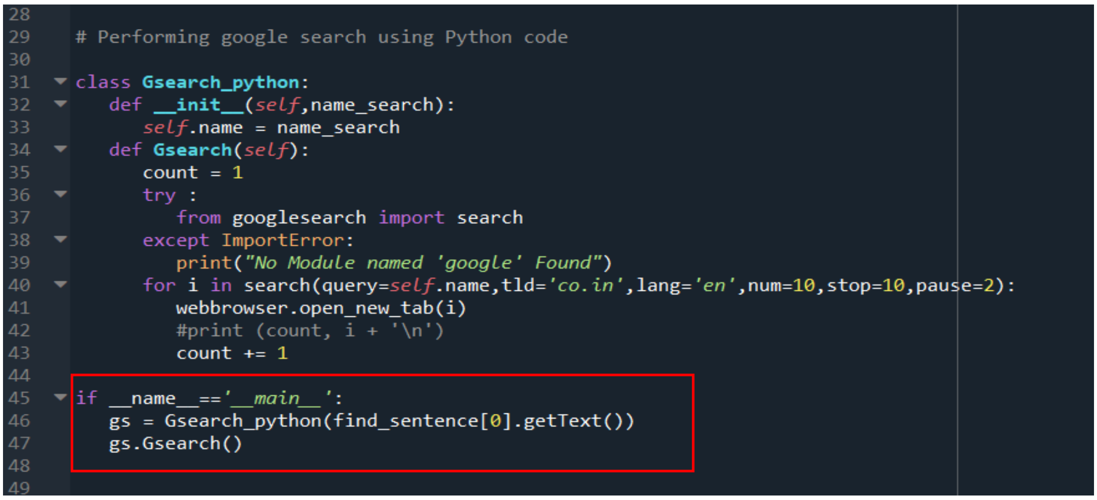
As shown in Figure 2.1.3, this is the input of the Automatic Reference Finder that is generated from the Note Taker component. The web scraping technique is used for the keyphrase extraction process and the automatic google search process.



```
import requests, sys, webbrowser, bs4

#Send the request to open the URL

web_request = requests.get('http://localhost:5000/result_text.html')
type(web_request)
web_request.text

#Extract the string from the title of the summerize note
soup = bs4.BeautifulSoup(web_request.text, 'lxml')
type(soup)

#find_sentence = soup.select('.entry-title')
#print(find_sentence[0].getText())
#print(find_sentence[1])

find_sentence = soup.select('p')
print(find_sentence[0].getText())
#print(find_sentence[0])
```

Figure 2.1.4: Request the summarized note URL to extract the keyphrases

According to Figure 2.1.4 first, import the requests library, sys library, bs4 library, and webbrowser library to the project. Then copy the URL of the summarized note and paste it in front of the “web_request” method. The requests library is used for requesting the URL. After that using bs4 (BeautifulSoup library) library the application can extract any content that is inside the particular HTML tags or CSS classes. The contents are extracted in the paragraph tags of the summarized note as shown in Figure 1.2.4. The extracted content is saved under the “find_sentence[0].getText()” variable.



```
# Performing google search using Python code

class Gsearch_python:
    def __init__(self,name_search):
        self.name = name_search
    def Gsearch(self):
        count = 1
        try :
            from googlesearch import search
        except ImportError:
            print("No Module named 'google' Found")
        for i in search(query=self.name,tld='co.in',lang='en',num=10,stop=10,pause=2):
            webbrowser.open_new_tab(i)
            #print (count, i + '\n')
            count += 1

if __name__=='__main__':
    gs = Gsearch_python(find_sentence[0].getText())
    gs.Gsearch()
```

Figure 2.1.5: Automatic Reference Finder

According to Figure 1.2.5, using the “find_sentence[0].getText()” as the input for the automatic reference finder application, it searches the references and opens them in separate tabs in the search engine. The webbroweser library is used to open the tabs in the search engine.

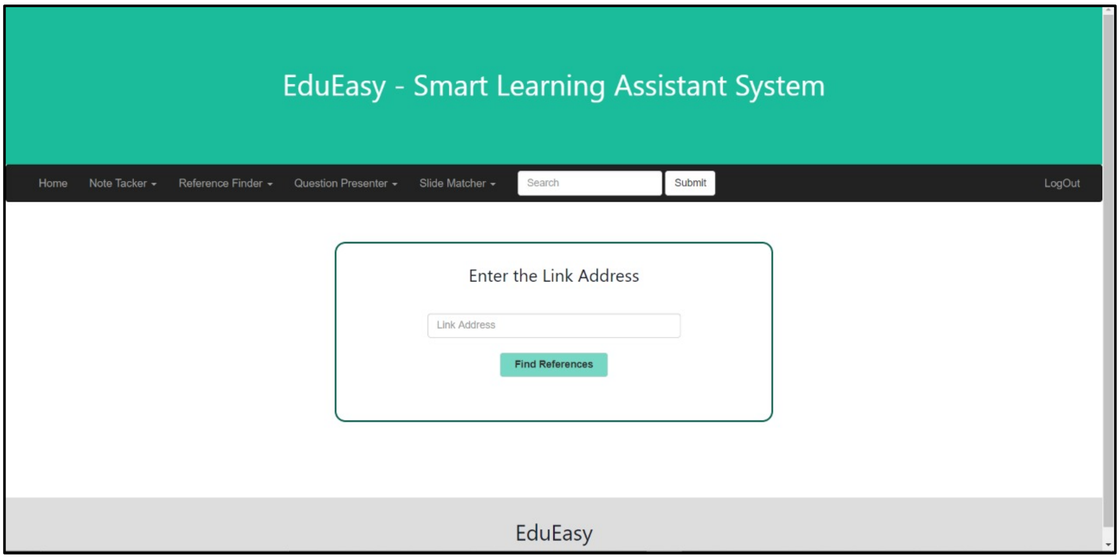


Figure 2.1.6: User Interface of “Finding References”

Figure 2.1.6 depicts the user interface of the first part of the Reference Finder component. The student copies the summarized note’s URL and pastes it in the blank and press the “Find References” button. Then the references appear in the separate tabs in the search engine.

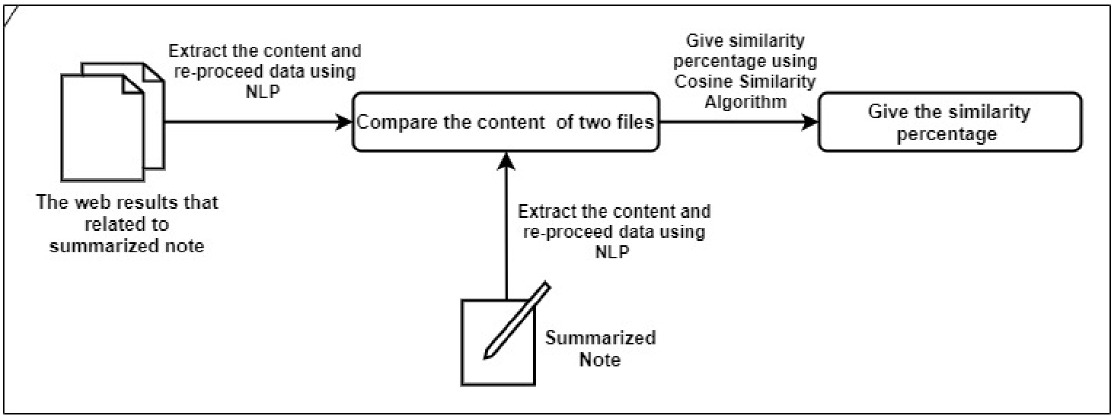
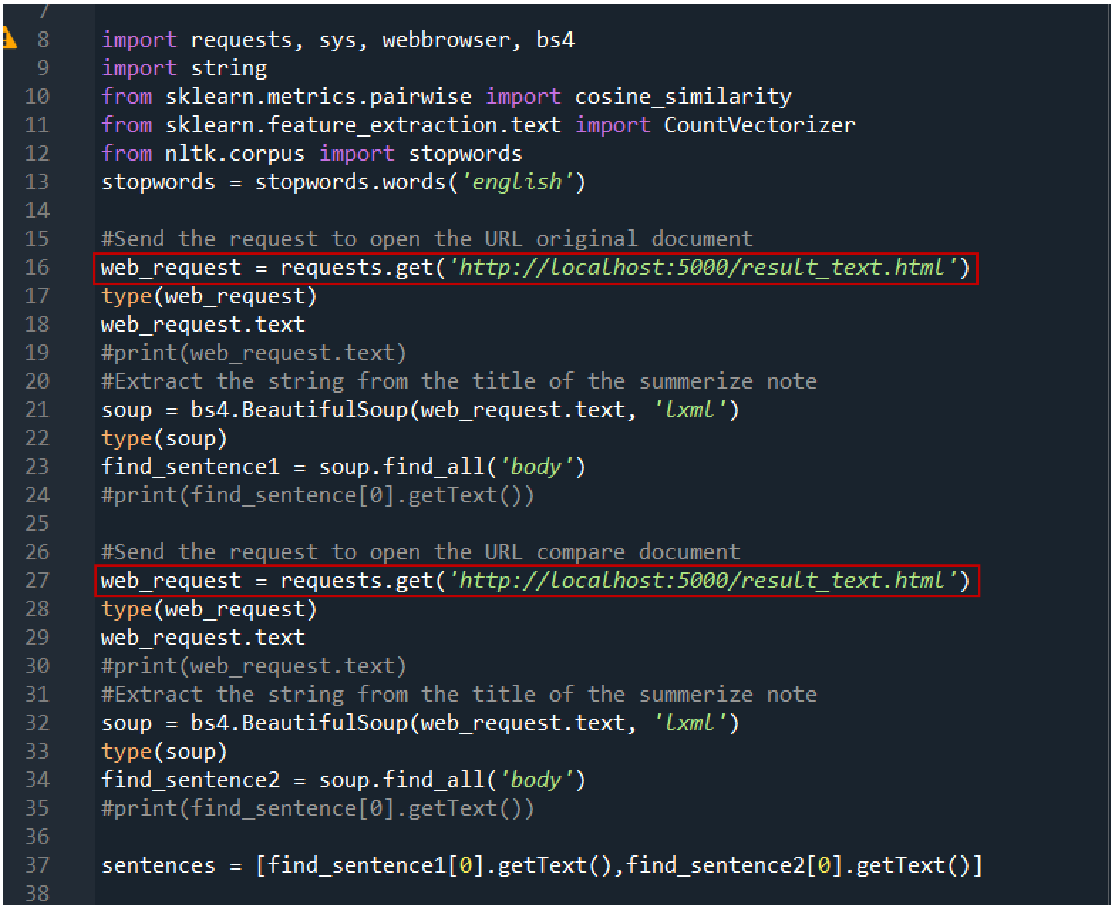

Extract the content and re-proceed data using NLP
The web results that related to summarized note
Compare the content of two files
Give similarity percentage using Cosine Similarity Algorithm
Give the similarity percentage
Extract the content and re-proceed data using NLP
Summarized Note

Figure 2.1.7: System diagram for the second part of the Reference Finder component

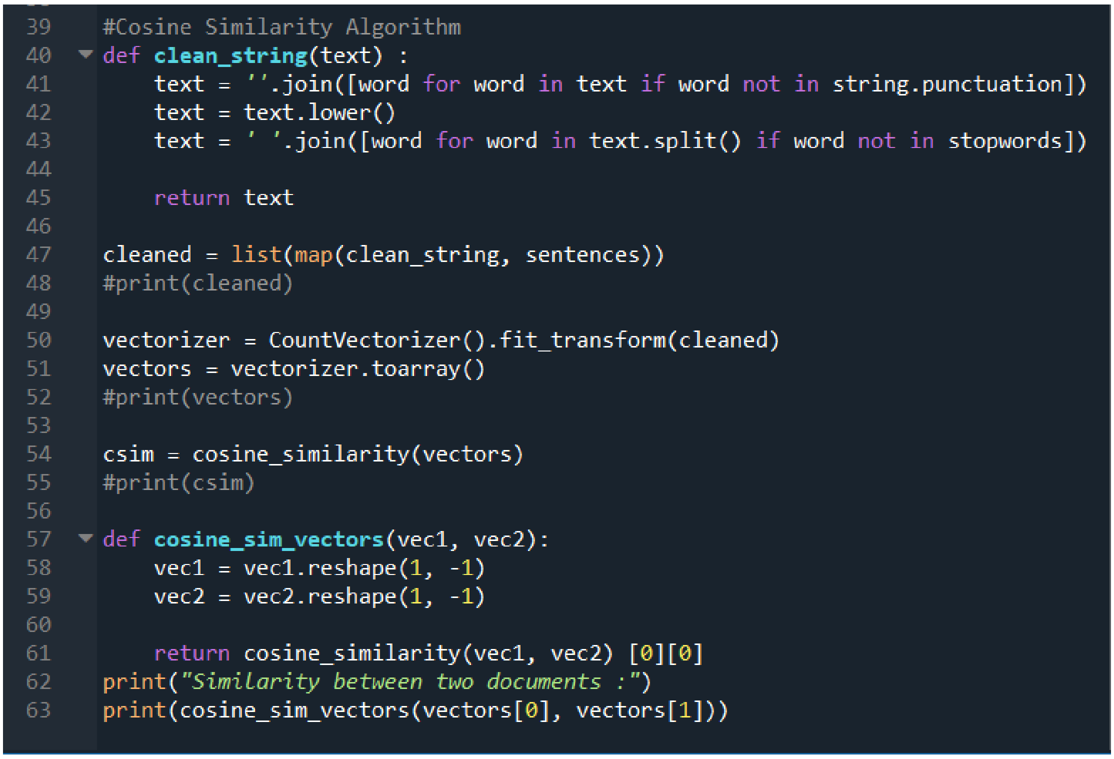
Figure 2.1.7 demonstrates how the second part of the Reference Finder Component is worked. The second half of the Reference Finder component begins with inputting the summarized note's URL and the one of the reference material's URL into the application as shown in Figure 2.1.8. The WebScrapping technique is used to do this also. Then the application extracts the whole content of the two web pages and stores them separately under two variables.



```python
import requests, sys, webbrowser, bs4
import string
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.feature_extraction.text import CountVectorizer
from nltk.corpus import stopwords
stopwords = stopwords.words('english')

#Send the request to open the URL original document
web_request = requests.get('http://localhost:5000/result_text.html')
type(web_request)
web_request.text
#print(web_request.text)
#Extract the string from the title of the summerize note
soup = bs4.BeautifulSoup(web_request.text, 'lxml')
type(soup)
find_sentence1 = soup.find_all('body')
#print(find_sentence[0].getText())

#Send the request to open the URL compare document
web_request = requests.get('http://localhost:5000/result_text.html')
type(web_request)
web_request.text
#print(web_request.text)
#Extract the string from the title of the summerize note
soup = bs4.BeautifulSoup(web_request.text, 'lxml')
type(soup)
find_sentence2 = soup.find_all('body')
#print(find_sentence[0].getText())

sentences = [find_sentence1[0].getText(),find_sentence2[0].getText()]
```

Figure 2.1.8: Get the two URLs to compare the similarity

Before going through the compare content similarity process the content data have to re-process using Tokenization and removing stop words techniques. Finally, according to Figure 2.1.9, the Reference Finder compares the content of the summarized note and the web results that are generated as the reference in google and generates a similarity value using the cosine similarity algorithm.



```
#Cosine Similarity Algorithm
def clean_string(text) :
    text = ''.join([word for word in text if word not in string.punctuation])
    text = text.lower()
    text = ' '.join([word for word in text.split() if word not in stopwords])

    return text

cleaned = list(map(clean_string, sentences))
#print(cleaned)

vectorizer = CountVectorizer().fit_transform(cleaned)
vectors = vectorizer.toarray()
#print(vectors)

csim = cosine_similarity(vectors)
#print(csim)

def cosine_sim_vectors(vec1, vec2):
    vec1 = vec1.reshape(1, -1)
    vec2 = vec2.reshape(1, -1)

    return cosine_similarity(vec1, vec2) [0][0]
print("Similarity between two documents :")
print(cosine_sim_vectors(vectors[0], vectors[1]))
```

Figure 2.1.9: Compare similarity algorithm

Here vectors are used to calculate the similarity value. If the user inputs the same URL as the two inputs the similarity value must be 1.000. If the user inputs different URLs that are relevant to different topics (as an example one document can be related to data mining and the other document can be related to heart patients) the similarity value must be 0.000. Finally, the student can refer to the most relevant and rated references using the reference finder component.

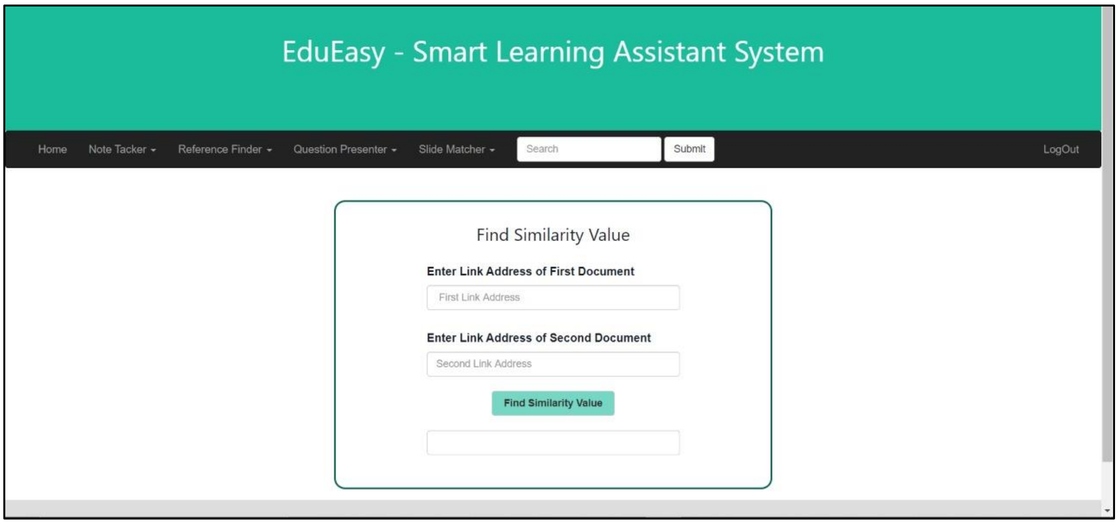


Figure 2.1.10: User Interface of “Similarity Value”

Figure 2.1.10 depicts the user interface of the second part of the Reference Finder component. First, the student has to copy the two URLs and paste them in suitable spaces. After press the “Find Similarity Value” button, the application generates the similarity value.

2.2. Commercialization aspects of the product

EduEasy E-learning web application is mainly designed for the university lecture room environment. Students who want to study smart, EduEasy system will help them to make their works easy and correctly. Students are struggling near exams to find proper lecture notes, relating references, sample questions, and the course materials such as lecture PPTs. EduEasy application is developed to overcome the above major problems.

EduEasy application is going to be introduced in three main packages: Platinum, Gold, and Silver. All the functionalities are included in the platinum package. Gold package is included all functions except the reference finding function. And Silver package has only the lecture summarization function and one month free trial will be given to all new users of the EduEasy system.

Features provided by the EduEasy system

- Proving a summarize note of lecture recording.
- Generate the questions and answers for the summarized note automatically.
- Generate exact relevant references automatically for additional reading.
- Navigate to the exact lecture slide from the summarized note.

Benefits of EduEasy system

- Easy to find the summarized note and no need to write down notes manually.
- Easy to find similar questions and answers regarding a specific lesson.
- Can find any type of reference material automatically by searching anywhere on the internet.
- Can navigate to the relevant lecture slide automatically.

2.3. Testing and Implementation

The software testing process lets the developer find and bugs before the program becomes usable, which decreases the chance of failure of the program. Many techs typically don't work on their own. When the research project is completed, the end-user can still have an eye on the potential situations that the end-user could experience. Accuracy and reliability are also properly observed, which means that the end consumer will be able to operate it productively.

For the testing of the application unit testing had been done after implementing an independently testable portion of the tool. Alpha testing and beta testing is being conducted. A small group of students would have the application set up on their devices.

Table 2.3.1: Test Case 1

Test ID	01
Description	Insert the URL of the summarized note into the given text field.
Steps	1. Open the "Reference Finder" interface 2. Navigate to the database where the summarized notes are stored. 3. Copy the link address of the chosen summarized note. 4. Paste the link address of the summarized note in the given space. 5. Press the "Find References" button.
Expected outcome	Display the reference materials in separate tabs on the search engine.
Actual outcome	Display the reference materials in separate tabs on the search engine.

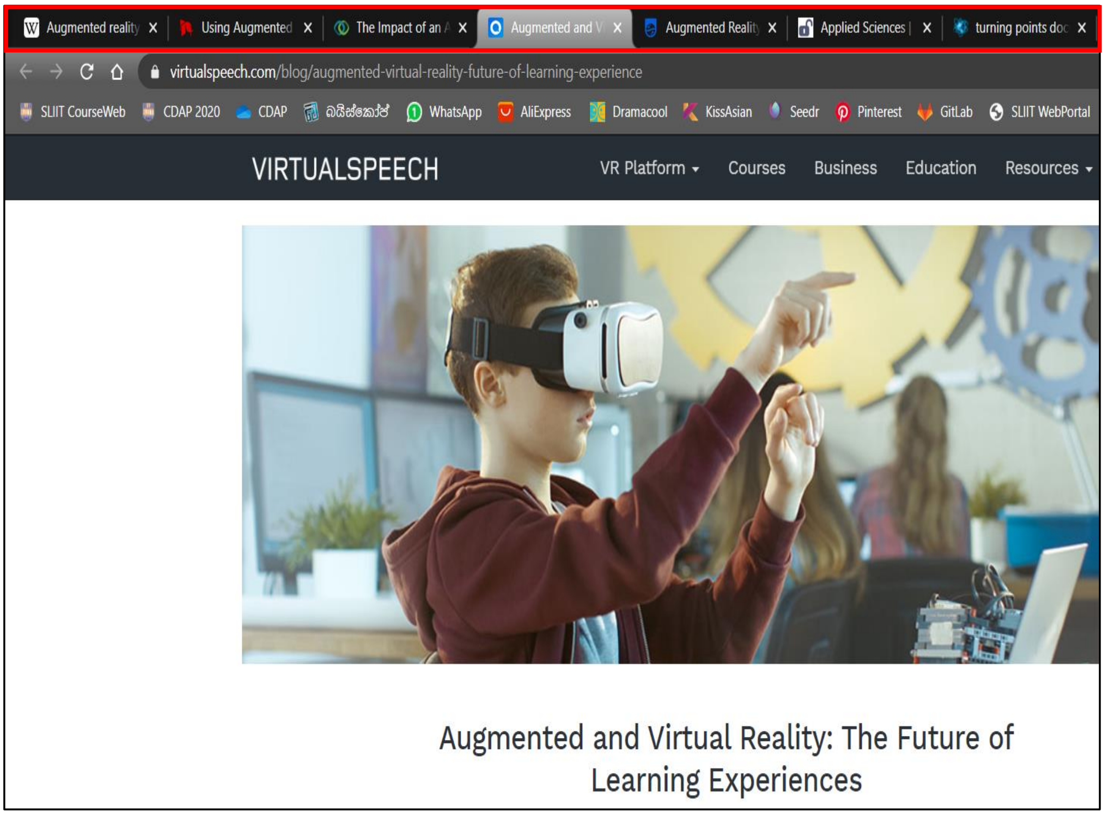
Table 2.3.2: Test Case 2

Test ID	02
Description	Insert two URLs into the given text fields and generate the similarity value.
Steps	1. Open the “Find Similarity Value” interface 2. Navigate to the database where the summarized notes are stored. 3. Copy the link address of the chosen summarized note. 4. Paste the link address of the summarized note in one of the given space. 5. Copy the link address of another reference material and paste it in the other space. 6. Press the “Find Similarity Value” button.
Expected outcome	Display a similarity value.
Actual outcome	Display a similarity value.

3. RESULTS AND DISCUSSION

3.1. Results

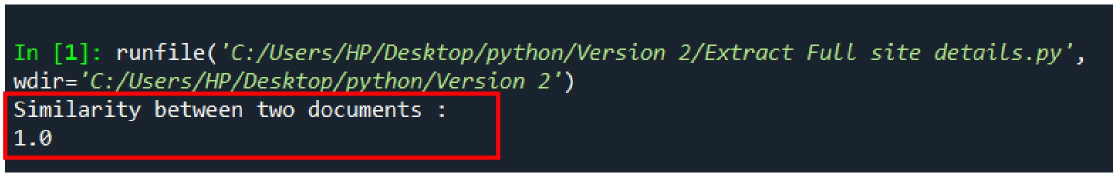
There are two main functionalities and two main results in the Reference Finder component. The first part of the reference finding process is started by the extraction of the topic from the summarized note. The reference finder component automatically searches the references according to the keyphrases and demonstrated all the references in separate tabs in the search engine as shown in Figure 3.1.1. The results appear as YouTube videos, articles, books, and blogs, etc.



Augmented and Virtual Reality: The Future of Learning Experiences

Figure 3.1.1: Automatic reference finder results

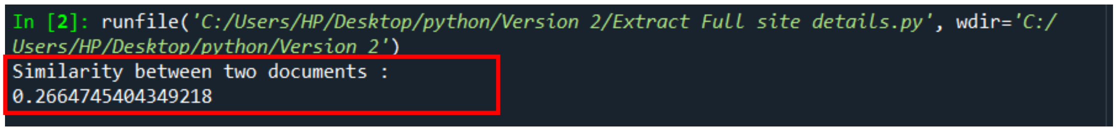
The second part of the component was started by using one of the references that appear as the result in the search engine as the input. The input reference material must not be a video material as a YouTube video. Then the component extracted the content of the web reference and re-processed the data before comparing the content similarity with the summarized note. The similarity algorithm generates a similarity value and the student can get an idea the web results are how much similar to the summarized note.



```
In [1]: runfile('C:/Users/HP/Desktop/python/Version 2/Extract Full site details.py',
wdir='C:/Users/HP/Desktop/python/Version 2')
Similarity between two documents :
1.0
```

Figure 3.1.2: Inputting the same URL as the inputs

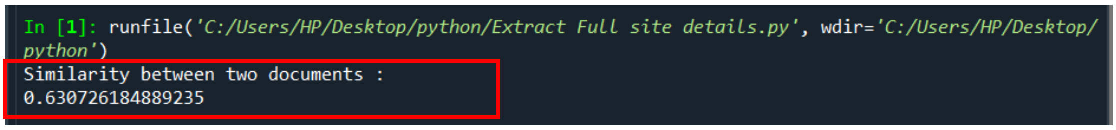
The output that depicts in Figure 3.1.2 is generated by inputting the same document's URL as the two inputs. It generates a 1.0 (1.00%) similarity value.



```
In [2]: runfile('C:/Users/HP/Desktop/python/Version 2/Extract Full site details.py', wdir='C:/
Users/HP/Desktop/python/Version 2')
Similarity between two documents :
0.2664745404349218
```

Figure 3.1.3: Inputting different URLs as the inputs

The output that depicts in Figure 3.1.3 is generated by inputting two different document's URLs as the inputs. It generates a 0.26647 (0.266%) similarity value.



```
In [1]: runfile('C:/Users/HP/Desktop/python/Extract Full site details.py', wdir='C:/Users/HP/Desktop/
python')
Similarity between two documents :
0.630726184889235
```

Figure 3.1.4: Inputting the summarized note's URL and the reference document's URL as the inputs

The output that depicts in Figure 3.1.4 is generated by inputting the summarized note's URL and the reference document's URLs as the inputs. It generates a 0.63072 (0.630%) similarity value.

According to the results, the similarity between the two contents is increased when the similarity value is higher. Finally, the student can refer to the most relevant and rated references using the reference finder component.

3.2. Research Findings

- The Reference Finder application is the main new research finding. Because there is no such application that has all the functionalities similar to the Reference Finder application as one product according to the former studies.
- In addition to that, the automatic reference finder that searches all the types of reference materials such as YouTube videos, pdf, articles, and e-books according to key phrases is one of the research findings. There are no such tools or applications that search all types of reference materials anywhere on the internet. And also it shows user target recommendations in separate tabs in the search engine.
- The other research finding is finding the most relevant reference material by comparing the reference material with the summarized note. The application generated a similarity value and the students can easily measure similarity and refer the suitable reference.

3.3. Discussion

There were so many challenges faced when developing the Reference Finder component. It was very hard to find algorithms and techniques. But the final outcome is more successful and the users can get a whole new experience by using the Automatic Reference Finder.

4. CONCLUSION

Referring the relevant reference materials is very important to all the students when they aim for the best grade in any subject. It helps to understand difficult lessons, formulas, theories, etc. The lecturer can’t teach everything regarding a lesson within the two-hour lecture time. So, students should search for more information relating to lessons by themselves.

Developing an automatic Reference Finder is a good way to overcome the above problems. The Reference Finder has two functionalities to provide solutions for two main obstacles that occur when finding references.

The reference Finder finds the exact reference materials automatically on the internet according to the selected keywords or keyphrases. The keywords are extracted from the summarized note that is generated in the Note Taker component as the output. Students can refer to the references that appear in separate google tabs and the results appear as e-books, articles, papers, and YouTube videos.

It generates a similarity value by comparing the original summarized note with the reference material. The similarity between the two contents is increased when the similarity value is higher. So the student can get an idea the web results are how much similar to the summarized note. It makes students easier to choose a good reference material among all the suggested references.

The fully functional Reference Finder tool can make student's life easier and help to succeed in their academic achievements.

REFERENCES

[1] M. A. Awar, “Pioneering smart learning,”. [Online]. Available: https://www.ellucian.com/emea-ap/insights/pioneering-smart-learning. [Accessed: May 4, 2020].

[2] A. Jokiaho, B. May, M. Specht and S. Stoyanov, “Barriers to suing E-leaning in an Advanced Way,” in International Journal of Advanced Corporate Learning, 2018, vol. 11, no. 1, pp. 17-22.

[3] D. Kaur, “How Smart Class Technology is Benefiting Education Sector,” entrepreneur.com, 2018. [Online]. Available: https://www.entrepreneur.com/article/322587. [Accessed: Aug. 10, 2020].

[4] CAE Team, “What is Smart Learning and why does it interest educational centers?,” cae.net, 2020. [Online]. Available: https://www.cae.net/what-is-smart-learning-and-why-does-it-interest-educational-centers/. [Accessed: Feb. 21, 2020].

[5] Cambridge University Press, “Meaning of wh- question in English,” cambridge.org, [Online]. Available: https://dictionary.cambridge.org/dictionary/english/wh-question. [Accessed: Sep. 12, 2020].

[6] X.Somg, J.Feng, G.Li, and Q.Hong, “LCA-based Keyword Search for Effectively Retrieving Information Unit from Web Pages,” Ninth International Conference on Web-Age Information Management, Zhangjiajie Hunan, China, July, 2008.

[7] G. Patil, A. Patil, "Web Information Extraction and classification using Vector Space Model Algorithm," In International Journal of Emerging Technology and Advanced Engineering, 2011.

[8]R. Lawson, Web Scraping with Python, 1st ed. 2015. [E-book] Available: Amazon.com.

[9] L. Richardson, "Beautiful Soup Documentation," [Online]. Available: https://www.crummy.com/software/BeautifulSoup/bs4/doc/. [Accessed June 05, 2020].

[10] D. M. Thomas, and S. Mathur, "Data Analysis by Web Scraping using Python," In 2019 3rd International Conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India, June, 2019.

[11]L. Logeswaran, H. Lee, D. Radev "Sentence Orderingand Coherence Modeling Using Recurrent Neural Networks," Department of Computer Science & Engineering, University of Michigan Department of Computer Science, Yale University, The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18).

[12]J. J. Webster, and C. Kit, "Tokenization As The Initial Phase in NLP," In the 15th International Conference on Computational Linguistics, 1992.

[13]X. Yang, J. Liu, "Semi-supervised learning of dialogue acts using sentence similarity based on word embeddings" Department of Electronic Engineering, Tsighua University Beijing100084, China.

[14] Dario Radecic,"Calculating String Similarity in Python," towardsdatascience.com, Oct.30,2019. [Online]. Available: https://towardsdatascience.com/calculating-string-similarity-in-python-276e18a7d33a. [Accessed: Aug.10, 2020].