EUL: A Unified Research Repository and SDG Classification Framework using KNN and Cosine Similarity

A Thesis Presented to

The Faculty of the School of Computer Studies

Department of the

University of San Jose-Recoletos

Cebu City, Philippines

In Partial Fulfillment

Of the Requirements for the Degree

Bachelor of Science in Computer Science

By

Bohol, Cristopher

Premacio, Paul Joshua

Dr. Ma. Lorna D.Miro

Adviser

May 2023

TABLE OF CONTENTS

ENDORSEMENT

ACKNOWLEDGEMENT

ABSTRACT

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

RATIONALE OF THE STUDY

THEORETICAL BACKGROUND

REVIEW OF THE RELATED STUDIES

PROJECT OBJECTIVES

PROJECT SCOPE AND LIMITATIONS

SIGNIFICANCE OF THE STUDY

RESEARCH METHODOLOGY

CHAPTER II

SOFTWARE REQUIREMENTS AND DESIGN SPECIFICATIONS

USE CASE DIAGRAM

USE CASE NARRATIVE

ACTIVITY DIAGRAM

CLASS DIAGRAM

USER INTERFACE DESIGN

CHAPTER III

SOFTWARE DEVELOPMENT AND TESTING

DEVELOPMENT AND TESTING PROCESS

Development Process

Testing Process

CHAPTER IV

SUMMARY, CONCLUSION, AND RECOMMENDATIONS

Summary of Findings

Conclusion

Recommendations

BIBLIOGRAPHY

ABSTRACT

This paper aims to develop an institution-wide digital research repository system that automatically classifies approved research papers into 17 UN Sustainable Development Goals (SDG). It emphasizes the research contributions for the empowerment of sustainable development by serving as a comprehensive knowledge repository and an SDG classifier. The classifier uses the K-nearest neighbor as its main algorithm in classifying research papers. This classifier is composed of TF-IDF, for text vectorization and knowing the relevance of a particular word in a corpus; cosine similarity to calculate the similarity scores of each document based on the TF-IDF vectors, and finally, the KNN algorithm to get the nearest neighbor and output the predicted labels.

CHAPTER I

INTRODUCTION

RATIONALE OF THE STUDY

Classification is a process of assigning objects into different classes or categories based on shared qualities or characteristics. Humans have done this process even before the invention of modern computers, Medieval army commanders sorting their formation based on the roles of the unit, pikes up-front, archers at the back, or a renaissance doctor categorizing medicines on a shelf and labeling them. The goal is simple: to easily manage and analyze the information at hand. A simple task no doubt, but doing it manually, with a large amount of information, let's say documents, is a tedious task. Manually managing documents is not ideal, especially if there are hundreds of research documents from the repository. To improve this process, we enlist the help of machines in the form of Artificial Intelligence (AI).

The advancement of modern computing gave birth to AI and eventually its underlying fields, Machine Learning (ML), and Natural Language Processing (NLP). Artificial Intelligence technology has been around since the 1940s [1]. It's been fine-tuned throughout the past decades. There are a lot of real-world applications that use ML and NLP. For instance: chatbots, language translators, email classification and filtering. With the help of AI and its related fields, document classification can now be done automatically [2].

In 2015, all United Nations member states adopted a resolution that calls for a shared blueprint for peace and prosperity, they call this, 'The 2030 Agenda for Sustainable Development. At its heart are the 17 Sustainable Development Goals (SDGs), in which developed and developing countries around the world are called to participate. With this plan in mind, classifying research according to the UN SDGs makes the research presently relevant.

This study aims to develop a system that an educational institution can benefit from by creating a web and mobile application that serves as a repository of all institutional research.

THEORETICAL BACKGROUND

Machine Learning

ML is defined as a type of AI that allows software applications to become more accurate at predicting outcomes without being explicitly programmed to do so [3]. This can be done by using algorithms or models to analyze and draw inferences from patterns in the data. There are three types of Machine learning: *supervised*, *unsupervised*, and *reinforcement learning*.

Supervised Learning is an approach to creating artificial intelligence (AI), where a computer algorithm is trained on input data that has been labeled for a particular output. The model is trained until it can detect the underlying patterns and relationships between the input data and the output labels, enabling it to yield accurate labeling results when presented with never-before-seen data [4]. It can be further down into two categories, ***classification*** and ***regression*** algorithms.

A **classification** algorithm is defined as a technique that is used to identify the category of new observations on the basis of training data. In Classification, a program learns from the given dataset or observations and then classifies new observations into a number of classes or groups [5]. Examples of classification are SVM and Naive Bayes.

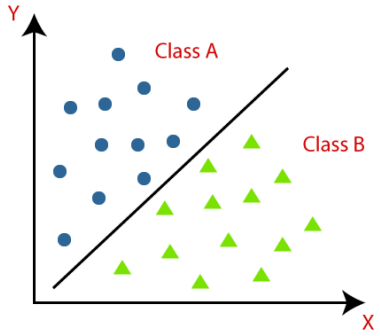


Figure 1. Sample Visualization of a Classification Algorithm

K-Nearest Neighbor Algorithm also known as KNN or k-NN, is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point.

Term Frequency - Inverse Document Frequency (TF - IDF)

TF-IDF is an example of text vectorization. It counts the frequency of a word in a document to measure its relevance to a document in a collection of documents. Text Vectorization is the process of converting text into numerical representation [9].

Term Frequency (TF) is simply just the frequency of a word in a document.

Inverse Document Frequency (IDF) is the frequency of the word across all documents. This gets the rarity of the word. The closer it is to 0, the more common the word is. Multiplying these two numbers results in the TF-IDF score of a word in a document. The higher the score, the more relevant that word is in that particular document [10].

Word	TF		IDF	TF*IDF	
	A	B		A	B
The	1/7	1/7	log(2/2) = 0	0	0
Car	1/7	0	log(2/1) = 0.3	0.043	0
Truck	0	1/7	log(2/1) = 0.3	0	0.043
Is	1/7	1/7	log(2/2) = 0	0	0
Driven	1/7	1/7	log(2/2) = 0	0	0
On	1/7	1/7	log(2/2) = 0	0	0
The	1/7	1/7	log(2/2) = 0	0	0
Road	1/7	0	log(2/1) = 0.3	0.043	0
Highway	0	1/7	log(2/1) = 0.3	0	0.043

Figure 4. TF-IDF sample

Mathematically it can be described as,

$$tf\ idf\ (t, d, D) = tf\ (t, d)\ .\ idf\ (t, D)$$

Where,

$$tf\ (t, d) = log\ (1 + freq\ (t, d))$$

$$idf\ (t, D) = log\ (\ \frac{N}{count\ (d \in D : t \in d)}\ )$$

Natural Language Processing

Computers can't understand human language. It only understands 0s and 1s or binary information. For computers and humans to *communicate directly,* NLP is used. NLP is described as a branch of computer science—and more specifically, the branch of artificial intelligence or AI—concerned with giving computers the ability to understand text and spoken words in much the same way human beings can [11].

Cosine Similarity

Cosine similarity measures the similarity between two vectors of an inner product space. It is measured by the cosine of the angle between two vectors and determines whether two vectors are pointing in roughly the same direction. It is often used to measure document similarity in text analysis.

REVIEW OF THE RELATED LITERATURE

Research paper classification systems based on TF-IDF and LDA schemes

A research paper classifier [13], uses TF-IDF as its text vectorization algorithm and LDA scheme in topic modeling. This study also uses the K-means clustering algorithm to finally classify its test data. While this paper is similar to EUL, the study in question only used the abstract part of the research as its source of data, while EUL uses the abstract, the rationale of the study, and the research methodology as its inputs.

Subject classification of research papers based on interrelationships analysis

A similar study of classifying research papers uses a novel supervised approach for subject classification of scientific articles based on an analysis of their interrelationships [14]. This study focuses more on scientific articles, citations, common authors, and common references to assign subjects to papers, whereas EUL accepts any form of the research topic as long as its authors are affiliated with USJ-R.

PROJECT OBJECTIVES

This study intends to develop a mobile and web application for managing USJ-R's research repository. Specifically, this study aims to:

- Have a single research repository for the entire USJ-R community.
- Classify papers based on the UN's 17 Sustainable Development Goals (SDG).
- Extract information from the research papers' relevant information like Approved Date, Title, Authors, Department, and Abstract.

PROJECT SCOPE AND LIMITATIONS

EUL is a web and mobile development project study.

This study focuses on research papers from faculty and students of USJ-R. The papers are expected to follow USJ-R's approved format. It will extract needed information from source documents and classify the document according to UN SDGs.

The study requires the device to be connected constantly to the internet as there are features of the system that fetches data from the cloud. As there are constant data processing and algorithms involved in the study, the machine that runs the processing should be at least a Ryzen 5 or intel core i5 with at least 8 GB of RAM. The accepted format is only PDF files to give uniformity in the repository and the scraping tool the study is using is PDF-specific.

Its mobile component requires Android v4.1 (Jellybean) and above. The system only supports English as its main language. As of now, the application will only be used by USJR faculty and students.

SIGNIFICANCE OF THE STUDY

Presently, there is a lack of an institutional-wide research repository system. The study not only is a research repository but also a system that automatically classifies research in accordance with the United Nation's Sustainable Development Goals (UN SDG).

Having a digital repository that classifies research papers based on UN Sustainable Development Goals promotes goal-oriented, and efficient knowledge discovery which enhances the institutions' contribution to sustainable development efforts on a global scale.

This also makes collaborations among departments and students much easier and more efficient because it is accessible through mobile and web applications.

RESEARCH METHODOLOGY

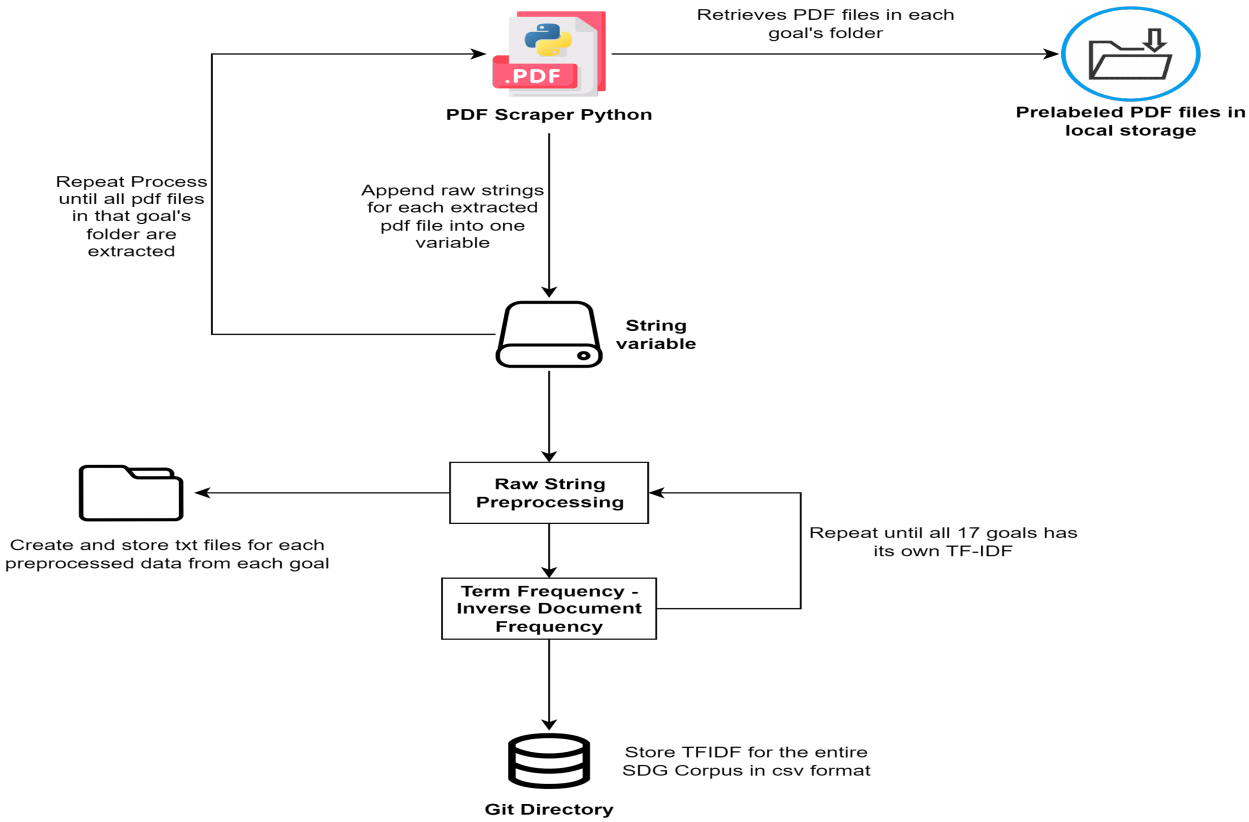


Figure 5. A conceptual framework for model training

Training the Model

The study uses a supervised technique in classifying documents and as such, the study already has predefined training data for each goal. Each goal consists of 5 pdf files that are manually classified as part of that goal. The entire training corpus consists of 85 pdf files. To limit the computational time of training the model, each pdf file in the training set is limited to only 3-5 pages. These pages are manually chosen to be the candidate for training as most of the relevant keywords are found on these pages. The pdf files are fetched directly from google scholar's wide variety of research journals and papers.

Figure 5 shows the process of training the model based on predefined labels.

Once the training starts, the first step is the scraper will extract all of the words in a single pdf file per goal. Once all pdf files in that goal have been scraped, the next process will start.

The second process is text processing, in this area, the appended string of all pdf files in one goal is cleaned of all unnecessary text, like stop words, punctuations, and numbers. Just like in any machine learning model, text preprocessing plays an integral part in the accuracy of the model as it removes unwanted or unimportant texts. Regular Expression and Lemmatization techniques are used in this process.

Another process is to store preprocessed strings and store them into their respective text files labeled with respect to their goal. This process is important in classification.

Once preprocessing is done, words are assigned a numerical value, representing the frequency of the word in the document. This process is called text vectorization. Term Frequency- Inverse Document Frequency is the most common form of text vectorization. There are other common techniques like bag of words, but the advantage of using TF-IDF is, it gives a numerical value or in this case, IDF, of a particular word in the entire corpus, meaning, it calculates how relevant that word is with respect to the SDG corpus. The lesser the value, the more common it is in the SDG corpus and vice versa.

Since the study needs to classify document based on 17 UN SDG, the system generates TF-IDF for each SDG, and afterward create a single TF-IDF that represents the entire SDG corpus.

After creating the TF-IDF for the SDG corpus, the result is then stored locally in a CSV format. This is a one-time process and can only be triggered once the system fails to locate the stored TF-IDF CSV file or by manually calling a function that trains the model.

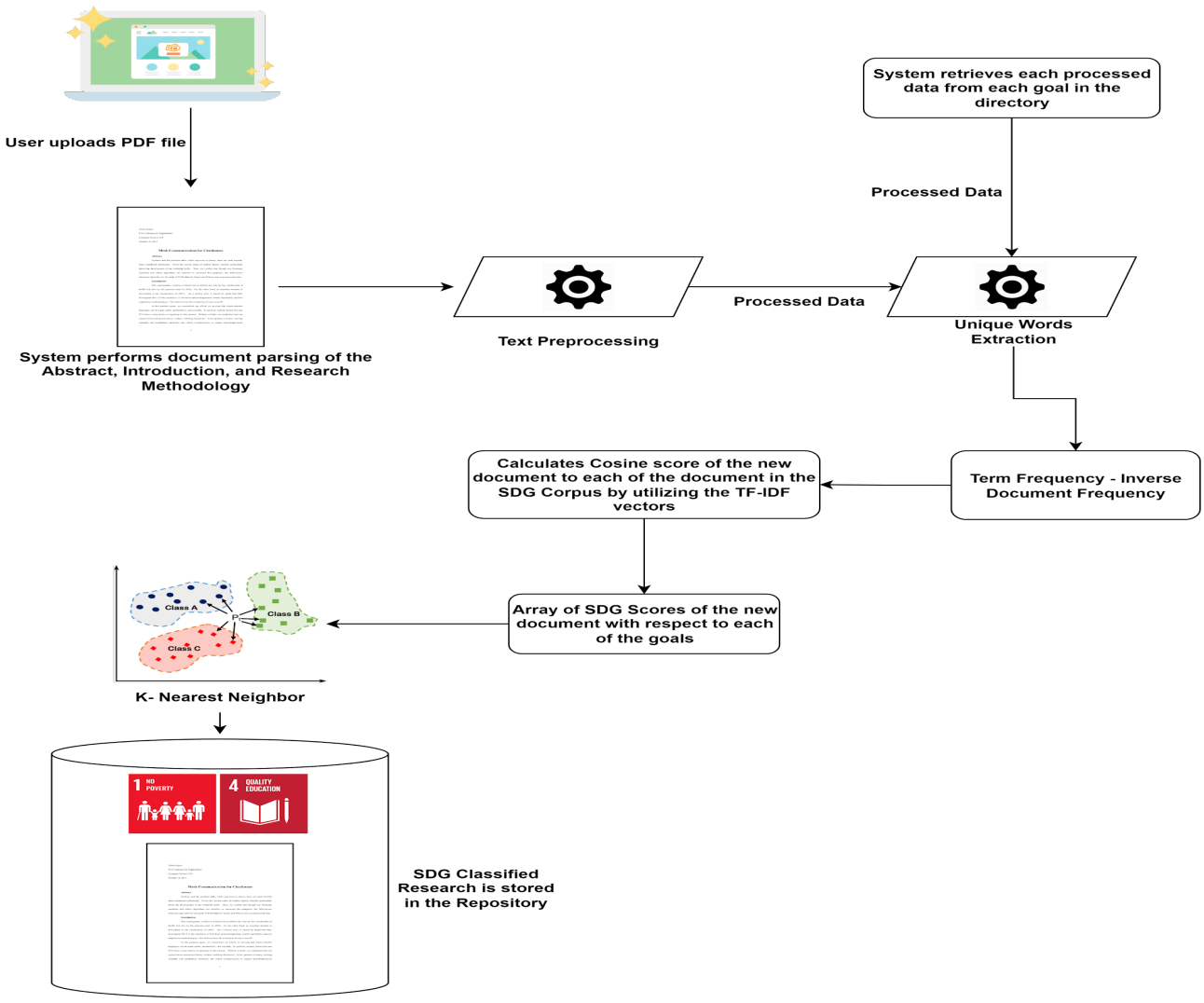


Figure 6. A conceptual framework for SDG Classification

Classification

The Abstract, the Rationale of the study, and the Research Methodology were chosen to be used as the primary source of input data since most of the relevant keywords needed for the system to classify can be found in these documents. In the first step, the user uploads a file (pdf), and the system will perform document parsing to extract texts from the Abstract, the Introduction, and the Research Methodology.

This appended data is composed of the extracted Abstract, Introduction, and Research Methodology and represents the new document to be classified. This appended data will undergo the same process when the system trains the model.

The system will perform the usual text preprocessing of the appended data. Once done, the system will retrieve the preprocessed data for all goals stored in the directory from the training process. These data retrieved from the directory alongside the preprocessed appended data are then used to extract unique words which will then be used to update the TF-IDF feature set.

After preprocessing, the system will now calculate the TF-IDF vectors for every 18 documents, the last document now represents the document to be classified.

Once the TF-IDF is done creating, the system will calculate the cosine similarity score of the last document to each of the documents in the TF-IDF. This will give an array of cosine scores that represents the similarity of the new document to each of the documents in the SDG corpus.

The cosine similarity scores will then be used as input to the KNN classifier. The KNN classifier performs the majority voting scheme to predict the labels or goals of the new document. The KNN classifier has a hyperparameter called K, this K represents the number of data points taken into account for classifying the new document. The optimal value of K depends on the nature of the data set. In this study, there are only 17 documents, and as such the value of K is set to 5.

Information Extraction

The system also incorporates information extraction every time the user uploads a PDF file. This technique automatically extracts the Approved Date, Research Title, Research Authors, and Department/School.

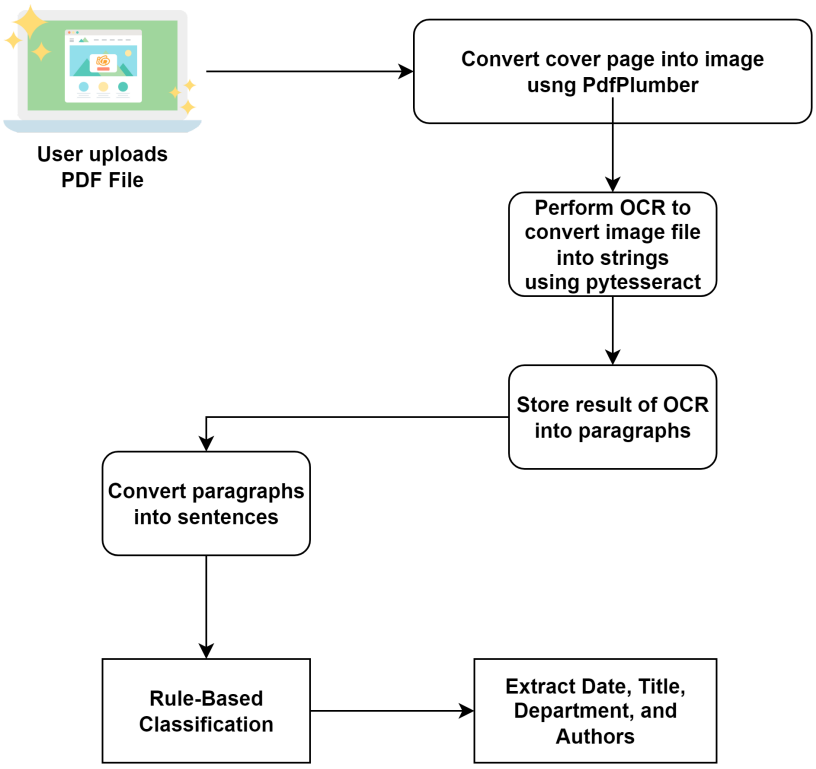


Figure 7. A conceptual framework for Information Extraction

Figure 7 visualizes how the system extracts information when the user uploads a PDF file. The first step is to convert the cover page into an image using pdfplumber. Once the image has been created, the system performs OCR using pytesseract to convert the image file into strings. After performing OCR, the result is then transformed into paragraphs, after transforming the paragraphs are then split into sentences. After splitting, a rule-based classification is used to extract the Date, Title, Authors, and Department.

This process makes the extraction of the raw text accurate and removes unwanted spacing if the usual pdfplumber is used. This makes the rule-based identification method an accurate and clean result.

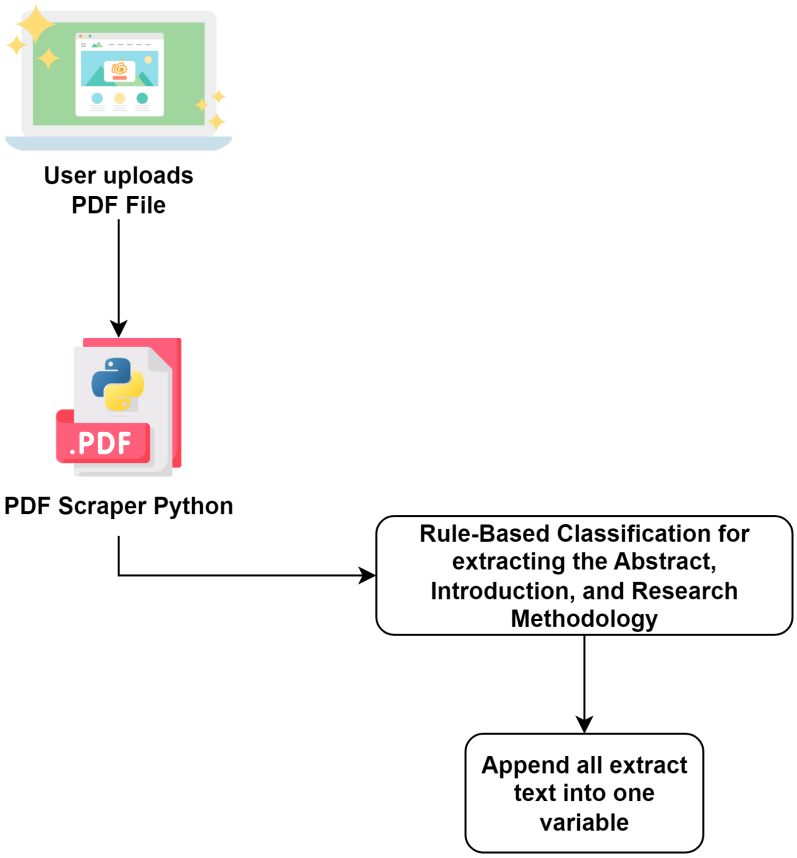


Figure 8. A conceptual framework for Extracting Abstract, Introduction, and Research Methodology

Figure 8 shows how the system extracts the Abstract, Introduction, and Research Methodology to be used as input to the SDG classifier. The pdfplumber is used here to extract text from pdf files instead of OCR since the result of extraction is clean.

A rule-based classification is then applied to identify which part of the extracted text is an Abstract, Introduction, and Research Methodology. To lessen the computational process, the system only limits the scraping to 15 pages as these 3 sections are usually found in these parts.

CHAPTER II

SOFTWARE REQUIREMENTS AND DESIGN SPECIFICATIONS

This chapter specifies the user and system requirements that are expected to be accomplished as well as the structure and process of achieving these. It contains sections for the Use Case Diagram, Use Case Narrative, Activity Diagram, Class Diagram, and User Interface Design,

USE CASE DIAGRAM

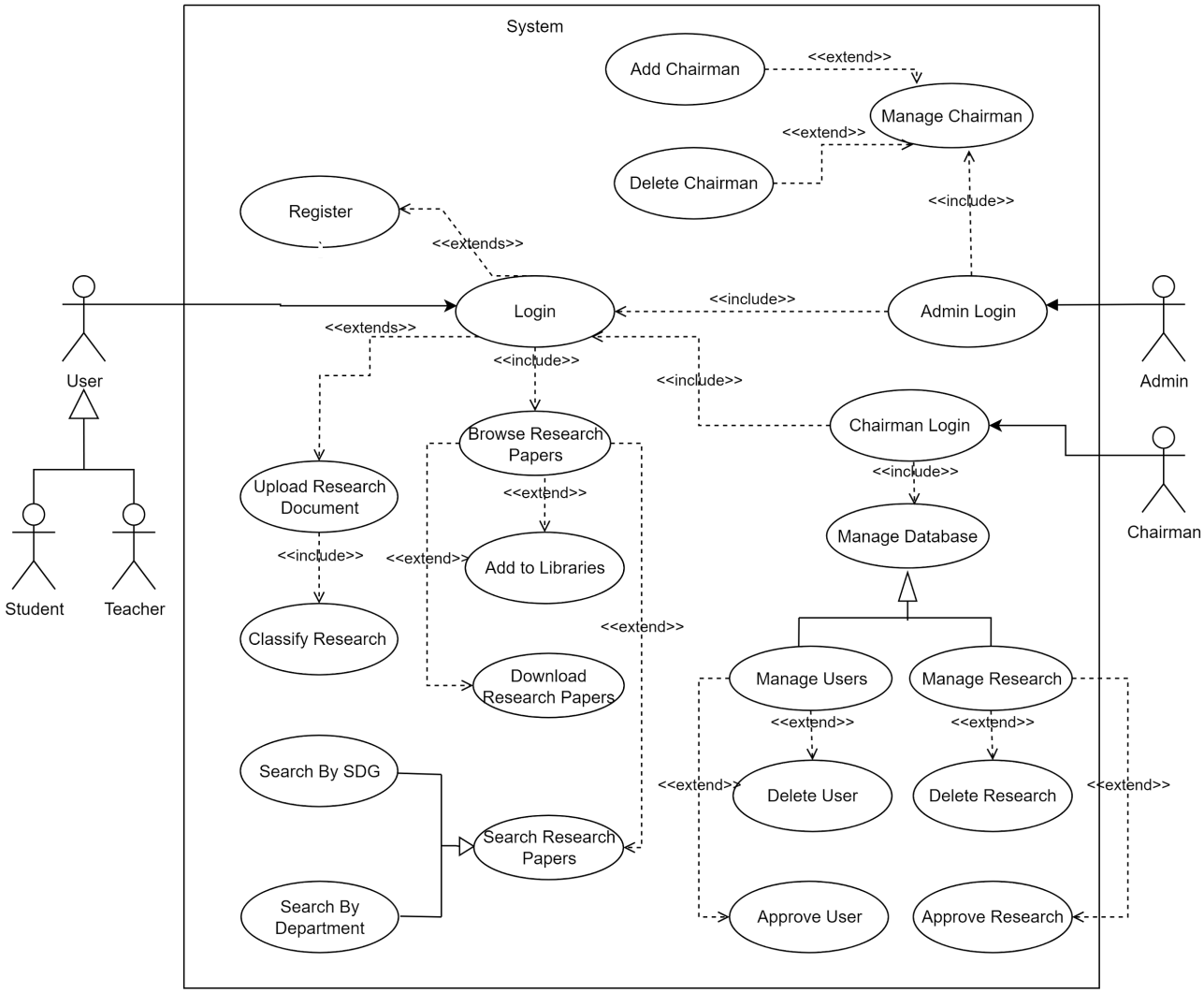


Figure 6 Defines the different use cases in which the system performs when students, teachers, and admin query the research repository on the Application.

This diagram maps how different actors or users interact with our system. A user except for the admin can upload their approved research paper and store it in the repository.

USE CASE NARRATIVE

The use case narrative shows the main scenarios and alternative flows of a use case. It provides more details about the use case.

Use Case 001	Verify User
Actor(s)	Admin, Chairman
Overview	Admin will send an API request to verify the user
Precondition(s)	User is not yet approved by the admin.
Post Condition(s)	The user is removed from the list of pending users.
Flow of Events	
Actor Action(s)	System Response
2. The admin will check if new request has been submitted on dashboard.	1. The system will display a list of new users on the screen
3. The admin will approve the request of the user	
Alternate Flow of Events: A2. The user is not affiliated	
1. If the submitted form from the user is not recognizable by the chairman, the chairman will delete the user	
Alternate Flow of Events: A2. User info is incorrect	
1. If the input value is incorrect, the admin can alter the info provided by the user	

Use Case 002	Manage Database
Actor(s)	Admin
Overview	The admin will check and manage the database to avoid malicious attacks on the database where it securely stores private accounts of teachers and students and the submitted outputs of the Josenian Community.
Precondition(s)	The admin should be logged in already before managing the database.
Post Condition(s)	The app can connect and fetch data from the database.
Flow of Events	
Actor Action(s)	System Response
	1. Performs CRUD operations as requested by the admin.

Use Case 003	Add to Libraries
Actor(s)	Student, Teacher, System
Overview	Users can add research to their respective libraries.
Precondition(s)	Teachers or Students should be login with their respective accounts
Post Condition(s)	Selected research is saved in the library.
Flow of Events	
Actor Action(s)	System Response
	1. After clicking the add button, the research will be saved in their profile.

Use Case 004	Upload Research Document
Actor(s)	Admin, Student, System
Overview	Admin, and student can upload a soft copy of research to the database where it securely stores and classifies research according to what categories of Sustainable Development Goals the research should be classified.
Precondition(s)	Admin or student should be login to access the features upload button.
Post Condition(s)	Research has been successfully classified and stored in the research repository.
Flow of Events	
Actor Action(s)	System Response
	1. The user taps the button to upload research
	2. The system classifies the research.
	3. After classifying the research into its respective categories, the result screen would display the category result and the information of the research paper to be displayed on the front page such as author, adviser, and, etc.
Alternate Flow of Events: A1 Wrong Format	
	1. The system will display "Wrong Format! PDF only!".
	2. The user will pick another file until the user uploaded the PDF format.
	3. Continue Process UC4 num 2.

Use Case 005	Search Research
Actor(s),	Teacher, Student
Overview	To enable users to search the research repository.
Precondition(s)	Actors should be login with their respective accounts for them to access the search function
Post Condition(s)	none
Flow of Events	
Actor Action(s)	System Response

	1. Check if the user specifies categories or filters. 2. Check research in the repository based on user search.
	3. Displays a list of research based on the user's search.
Alternate Flow of Events: Alternate Flow A1: No research found.	
1. System displays text that no research is found.	

Use Case 006	Register
Actor(s),	Teacher, Student
Overview	Actors are expected to create their accounts before they can log in and use exclusive features of the app.
Precondition(s)	none
Post Condition(s)	none
Flow of Events	
Actor Action(s)	
	1. From home, the user will need to go to registration page
	2. The user will enter the necessary information and then click the register button
	3. The system will check if the entered email or id has already been registered before.
	3. If the account is not yet registered, the user is successfully registered to the website.
Alternative Flow of Events: A1: Exist Account	
If the system detects it is already existing, the system will pop up an error message	
The user will ask to input again.	

Use Case 007	Login
Actor(s)	Student, Teacher
Overview	To let the actors access private features in the app.
Precondition(s)	none
Post Condition(s)	Redirected to homepage
Flow of Events	
Actor Action(s)	
1. From homepage, actors need to click sign in to redirects to sign in page.	3. If credential is valid and the account is approved, the actors will be redirected to private homepage which is based on what role they have as a user status.
2. The actors will then input their account information like email and password.	
Alternate Flow of Events: A1: Credentials is not valid	

	1. System will display the error message then the system will ask to input again.
	2. If submitted credentials is correct, redirect to private homepage.
	2. If the system detects that the number of attempt made by user is equal to 3, then the system will notify the user wants to reset password.
Alternate Flow of Events: A1: Credentials is valid but account was not yet approved by admin.	
	1. System will display the error message to wait for confirmation message.

Use Case 007	Login
Actor(s)	Student, Teacher
Overview	To let the actors access private features in the app.
Precondition(s)	none
Post Condition(s)	none
Flow of Events	
Actor Action(s)	
	1. From the homepage, actors need to click sign in to redirect to the sign-in page.
	2. The actors will then input their account information like email and password.
	3. If the credential is valid and the account is approved, the actors will be redirected to the private homepage which is based on what role they have as a user status.
Alternate Flow of Events: A1: Credentials are not valid	
	1. The system will display the error message then the system will ask to input again.
	2. If the submitted credentials are correct, redirect to the private homepage.
	2. If the system detects that the number of an attempt made by the user is equal to 3, then the system will notify the user who wants to reset the password.
Alternate Flow of Events: A1: Credentials are valid but the account was not yet approved by admin.	
	1. System will display the error message to wait for a confirmation message.

Use Case 008	Classify Research
Actor(s)	System
Overview	It automatically classifies what kind of category the research belongs to.
Precondition(s)	The file uploaded has successfully extracted information
Post Condition(s)	Teacher successfully gave updates and added new instructions to the teams.

Flow of Events	
Actor Action(s)	System Response
	1. The system reads the text in the docs or pdf uploaded by the user.
	2. System performs preprocessing of the document.
	3. System performs the process indicated in the Classification Diagram.
	4. System successfully classified research.

Use Case 009	Add Comment
Actor(s)	Student, Teacher
Overview	This use case allows a student or teacher to add a comment to a research paper
Precondition(s)	
	The user is logged into the system.
	The user has selected a research paper to view.
	The research paper has comments enabled.
Post Condition(s)	
	The comment is added to the research paper in the system.
	The user is returned to the research paper view page.
Flow of Events	
Actor Action(s)	System Response
	1. The system displays a form for the user to enter their comment.
2. The user enters their comment and submits the form.	3. The system adds the comment to the research paper and displays it on the page.
	4. The user is returned to the research paper view page.
Alternate Flows	
2a. The user cancels the comment.	
1. The system discards the comment and returns the user to the research paper view page.	
2b. The user submits an empty comment.	
1. The system displays an error message indicating that the comment cannot be empty.	
2. The user is prompted to enter a valid comment.	
Exceptions:	
1. The user is not logged into the system.	
2. The user selects a research paper that does not have comments enabled.	
3. The system encounters an error while adding the comment.	

Use Case 010	Update User
Actor(s)	Admin, Chairman
Overview	This use case allows the admin to update a user's information in the library management system.

Precondition(s)	
	The actors is logged into the system.
	The actors has selected the "Update User" option from the dashboard panel.
	The actors to be updated has been selected.
Post Condition(s)	
	The user's information has been updated in the system.
Flow of Events	
Actor Action(s)	System Response
	1. The system displays a form with the current user information.
2. The user enters their comment and submits the form.	3. The system adds the comment to the research paper and displays it on the page.
	4. The user is returned to the research paper view page.
Alternate Flows	
3a. The admin cancels the update.	
1. The system discards the changes and returns the admin to the previous screen.	
Exceptions:	
1. The user is not logged into the system.	
2. The system encounters an error while updating the user information.	

Use Case 011	Browse Research Papers
Actor(s)	Admin, Chairman, Teachers, Students
Overview	This use case allows a user to browse the research papers in the library management system.
Precondition(s)	
	The actors is logged into the system.
	The actors has selected either one of the sdg icon in the home menu
Post Condition(s)	
	The user has viewed the selected research paper
Flow of Events	
Actor Action(s)	System Response
	1. The system displays a list of sdg in the home page
2. The user clicks one of the icon corresponds to the specific SDG category	3. The system will redirect the user to corresponding sdg where the sytem filter out the list of researches according to category
4. The users select read more to view	4. The user has viewd the selected research paper

the specific research	
Alternate Flows: none	
Exceptions:	
1. The user is not logged into the system.	
2. There are no research papers available in the system	

Use Case 012	Edit/Delete Coment
Actor(s)	Student, Teacher
Overview	This use case allows a user to edit or delete their comment on a research paper in the library management system.
Precondition(s)	
	The user is logged into the system.
	The user has selected a research paper to view.
	The user has previously added a comment to the research paper.
Post Condition(s)	
	The comment is added to the research paper in the system.
	The user is returned to the research paper view page.
Flow of Events	
Actor Action(s)	System Response
	1. The system displays the research paper information and the user's comment.
3. The user selects the "Edit/Delete Comment" option.	2. The system displays the edit or delete button.
4. The user edits or deletes their comment and submits the form.	5. The system updates or deletes the comment and displays a message indicating that it was updated or deleted successfully.
Alternate Flows: None	
Exceptions:	
1. The user is not logged into the system.	
2. The selected research paper does not exist.	
3. The user's comment does not exist.	
4. The user is not the author of the comment.	

Use Case 013	Search Research Papers
Actor(s)	Admin, Chairman, Teachers, Students
Overview	This use case allows a user to search for research papers in the library management system.
Precondition(s)	

	The user is logged into the system.
	The user has selected the "Search" option from the side navigation menu
Post Condition(s)	
	The comment is added to the research paper in the system.
	The user is returned to the research paper view page.
Flow of Events	
Actor Action(s)	System Response
	1. The system displays a search form with a list of all research
2. The user enters search keywords	3. The system displays a list of research papers that match the search criteria.
4. The user selects a research paper to view.	5. The system displays the selected research paper and its details.
Alternate Flows	
3a. The search returns no results.	
1. The system displays a message indicating that no results were found.	
Exceptions:	
1. The user is not logged into the system.	
2. There are no research papers available in the system.	
3. The search criteria are invalid.	

ACTIVITY DIAGRAM

The activity diagram shows the workflow behavior of the system by describing the sequence of actions in the process.

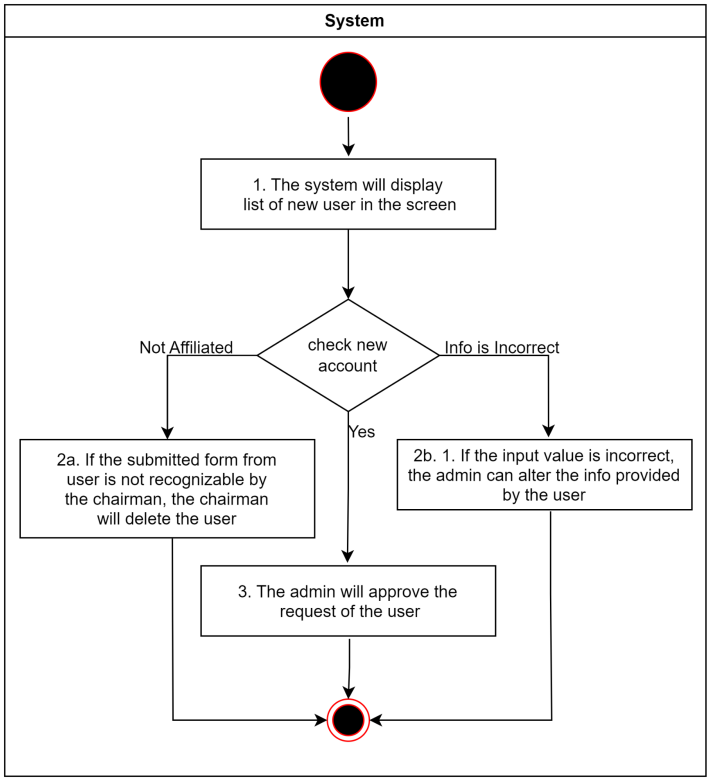


Figure 4. UC001 Verify User

Figure 4 shows the visualization when the admin verifies the request of the user. The system will check the inputted Student ID if it is correct, and the system will permit the user to access the app.

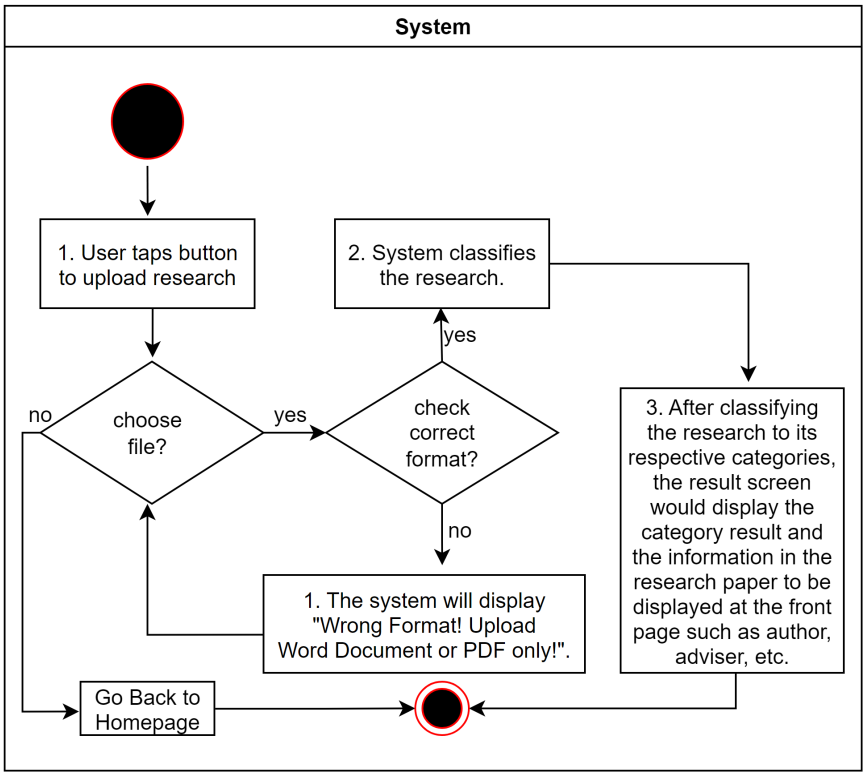


Figure 5. UC004 Upload Research Document

Figure 5 shows how the system behaves when the users upload research. The system will check the type of file being uploaded until it is a pdf file.

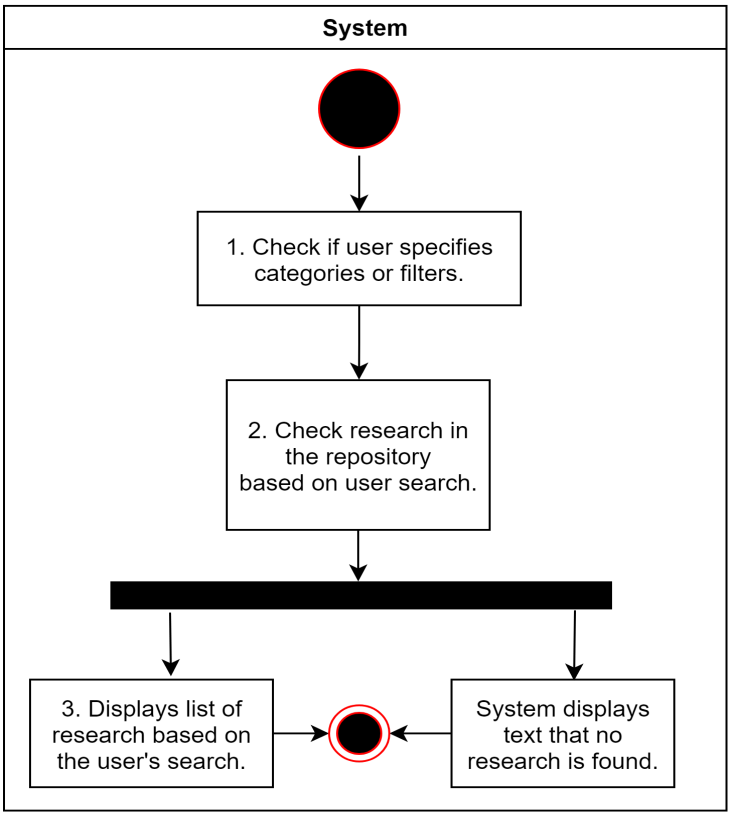


Figure 6. UC005 Search Research

Figure 6 shows the system's behavior when users either use filters with their search results or not. Research display is based on what categories are closer to the user input. It would only display no research found if stack categories plus the search term don't match up with the research in the repository.

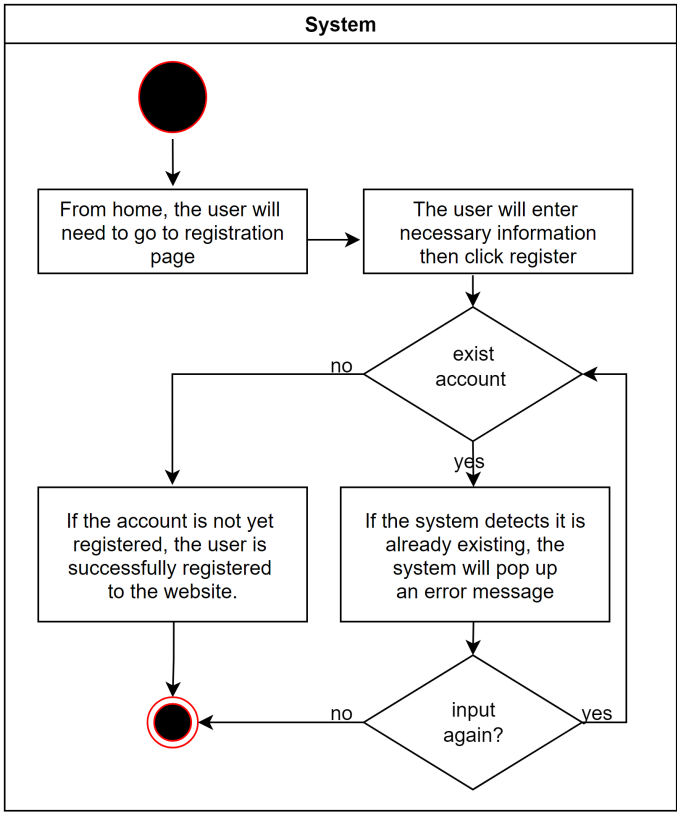


Figure 7. UC006 Register

Figure 7 shows the registration process of the user when using the application. If the account the user wants to register already exists then the user is required to enter again unless the user gives registers their account. Else, they will wait for the admin's approval of their account.

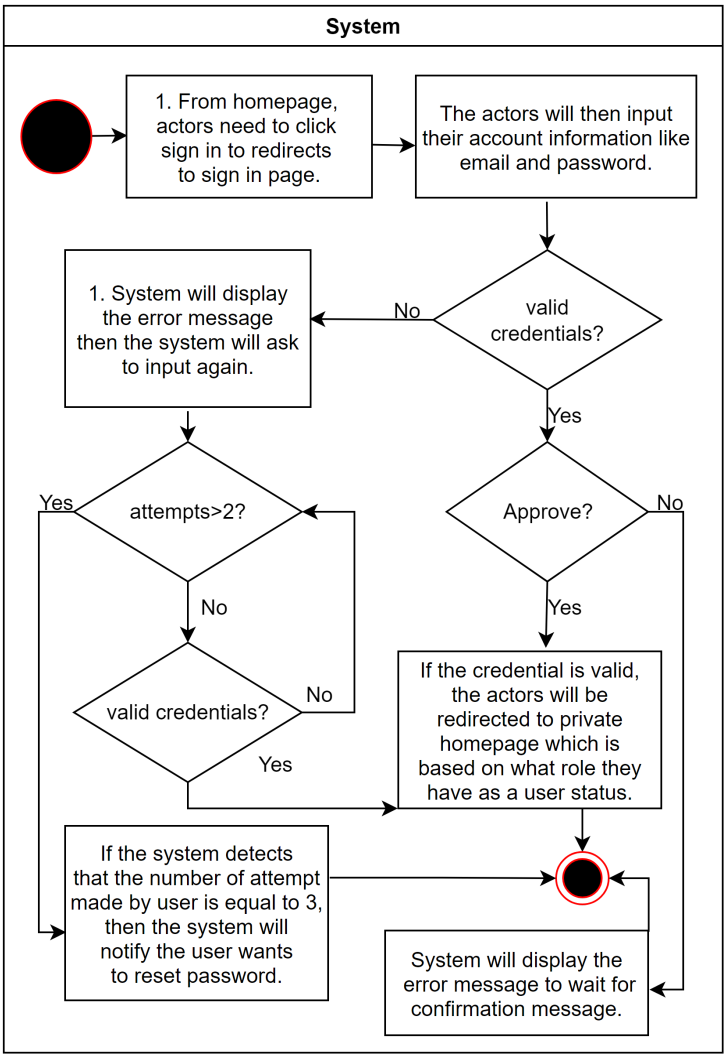


Figure 8. UC007 Login

Figure 8 shows the login process of the user when using the application. If the user fails to enter the correct information 3 times then the system will notify the user that he/she needs to reset the password using the user's email. Also, users without approval from the admin may not be able to log in.

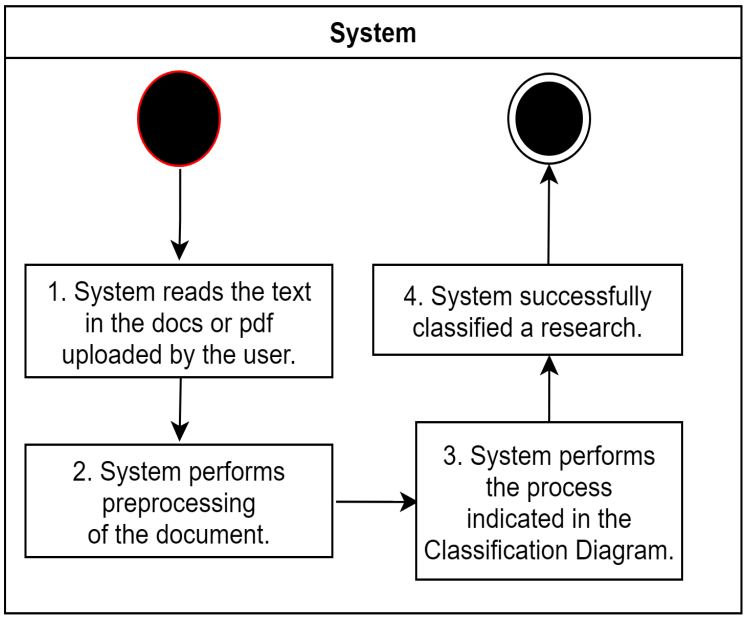


Figure 9. UC008 Classify Research

Figure 9 shows the behavior of the system when performing the classification of the research uploaded by the user. Extracted keywords from the Abstract and the rationale of the study will be used in order for the algorithm to be classified according to the predicted result. After classifying the research and, keywords, classified categories will be used in future use such as finding similar research.

CLASS DIAGRAM

The class diagram shows the structure of the system as classes with their attributes, operations, and relationships among other classes.

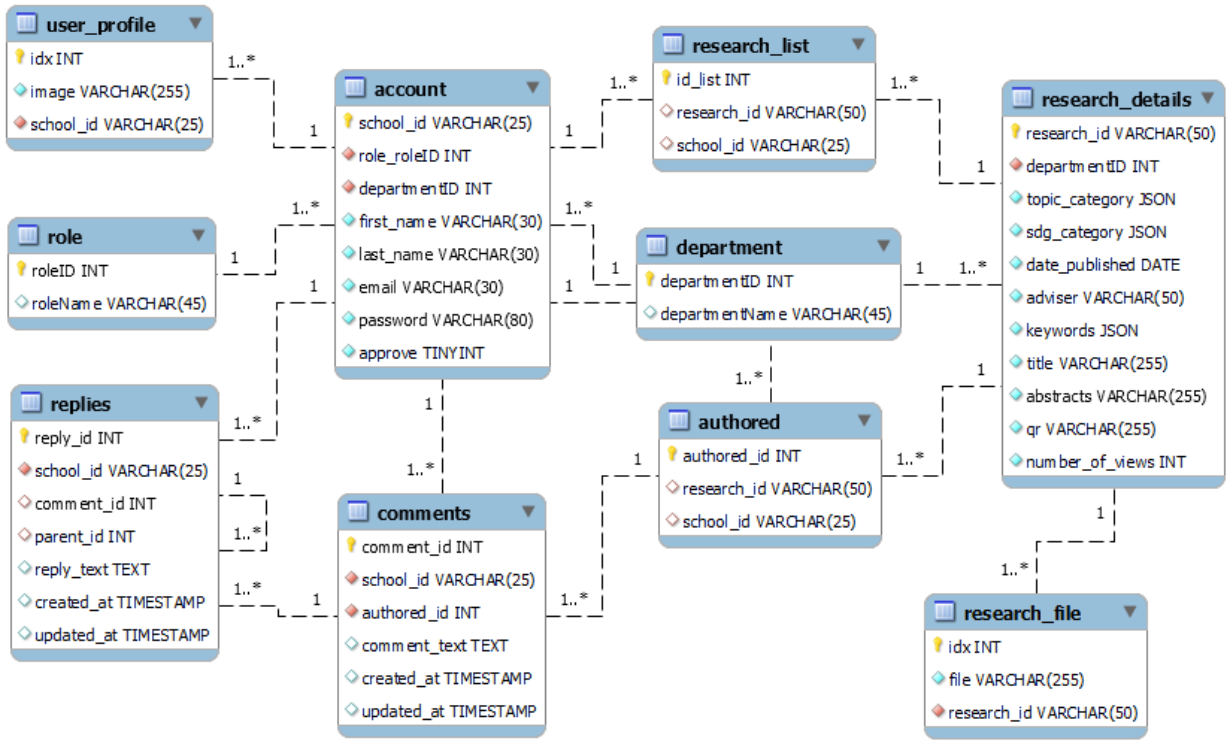
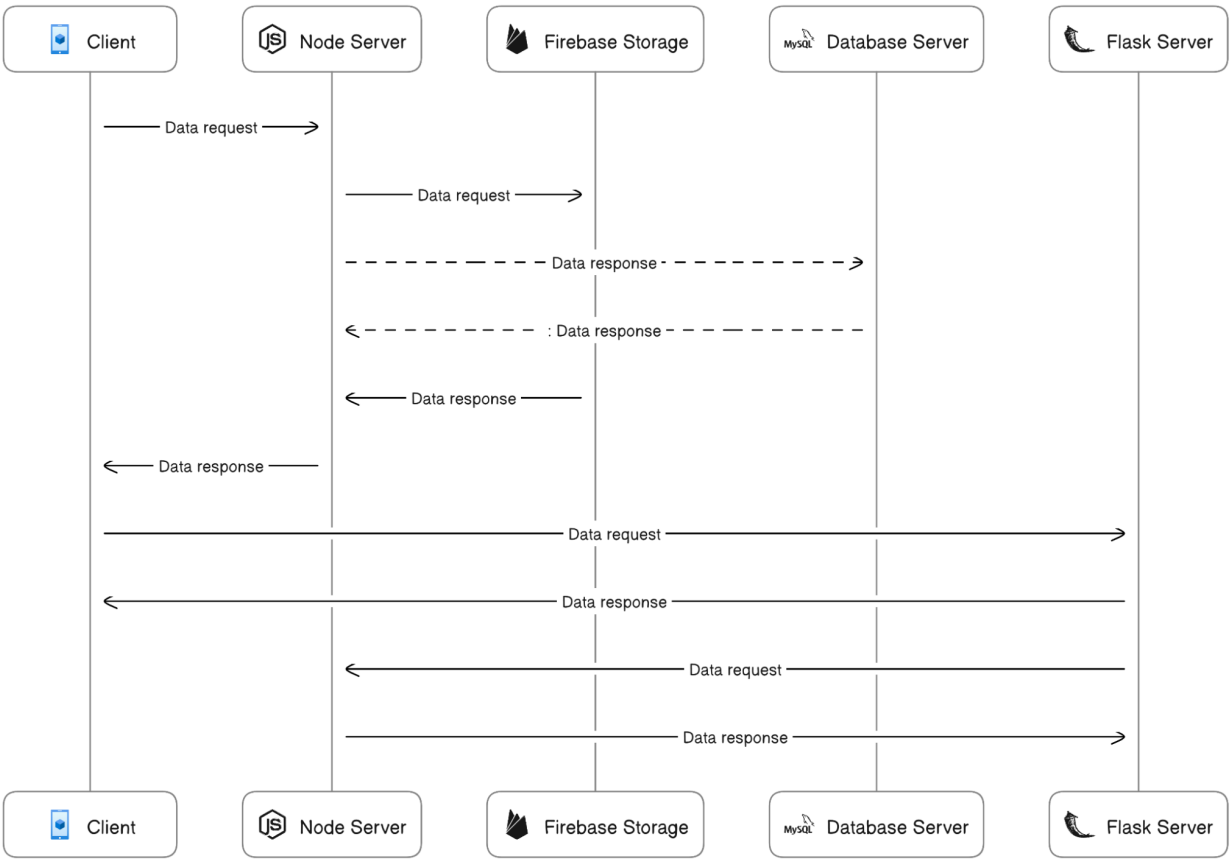
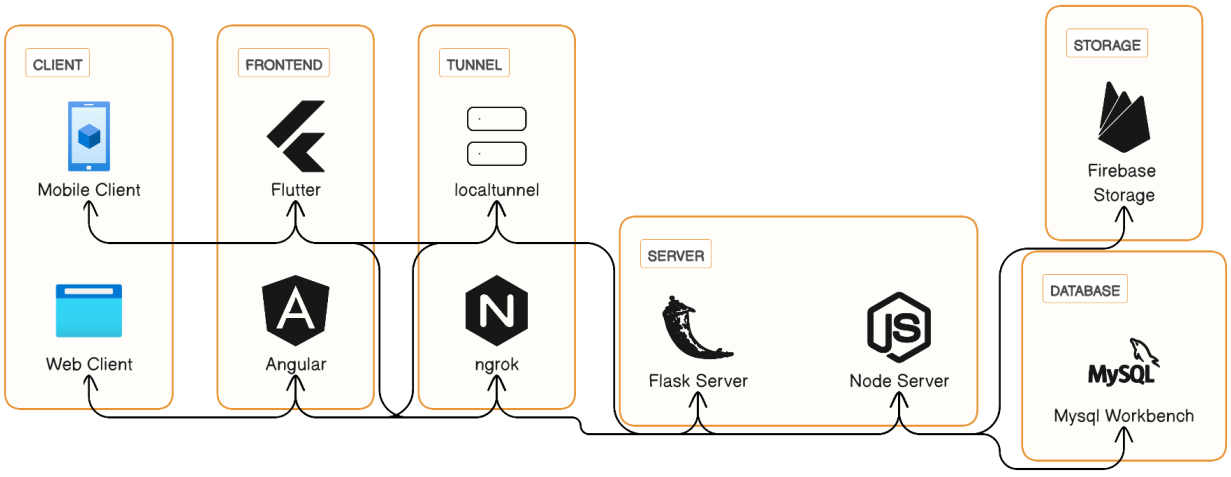


Figure 10. Class Diagram of the EUL


Client
Node Server
Firebase Storage
Database Server
Flask Server
Data request
Data request
Data response
: Data response
Data response
Data response
Data request
Data response
Data request
Data response
Client
Node Server
Firebase Storage
Database Server
Flask Server


CLIENT
Mobile Client
Web Client
FRONTEND
Flutter
Angular
TUNNEL
localtunnel
ngrok
SERVER
Flask Server
Node Server
STORAGE
Firebase Storage
DATABASE
Mysql Workbench

USER INTERFACE DESIGN

The user interface design is the process of creating interfaces that are expected to provide insights into how the user can interact with the system.

Mobile Application

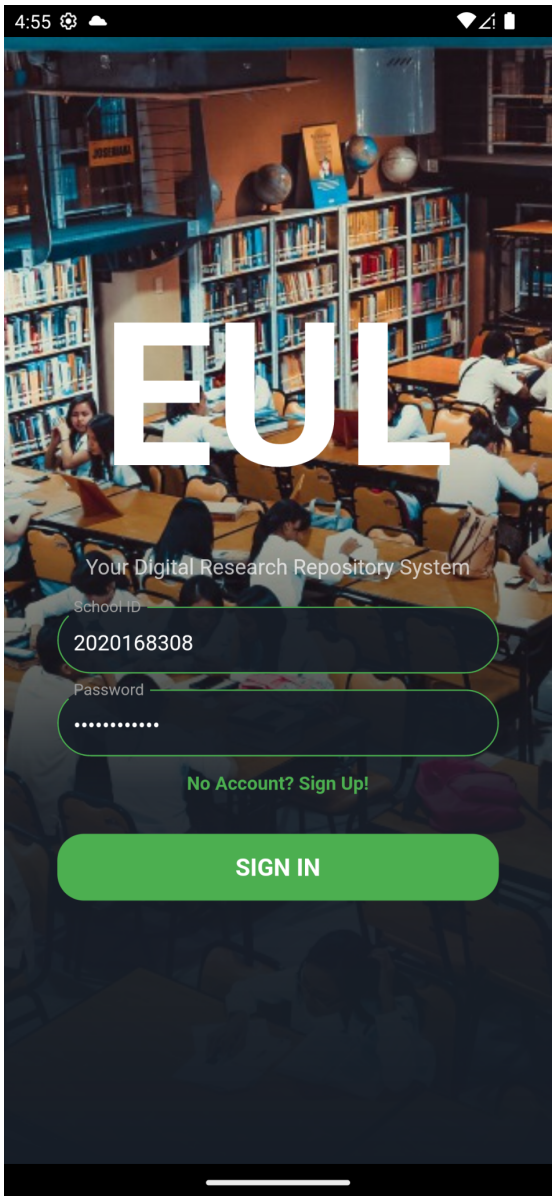


Figure 11 shows the mobile's login screen

Figure 12 shows the login screen where the Student ID and password are required.

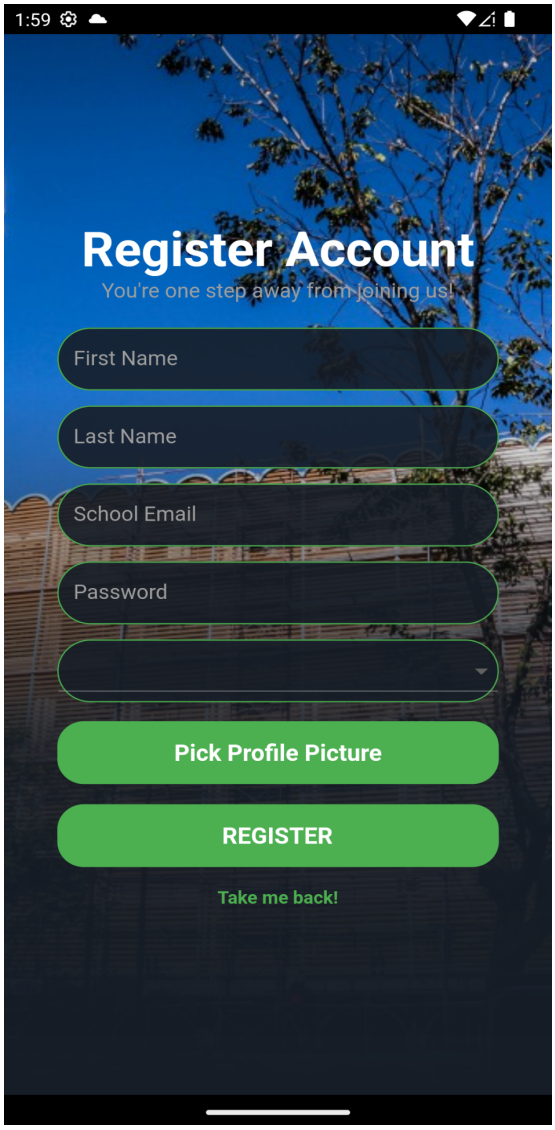


Figure 12 shows the mobile's registration page

This screen shows how the user can create an account with EUL. Users can register by inputting the necessary information on the screen.

Figure 13 shows the mobile dashboard

This is the main dashboard of the mobile application, after login the user will be redirected to this screen. This screen shows all of the 17 UN SDGs. Here users can also search for a particular research or sort papers based on categories.

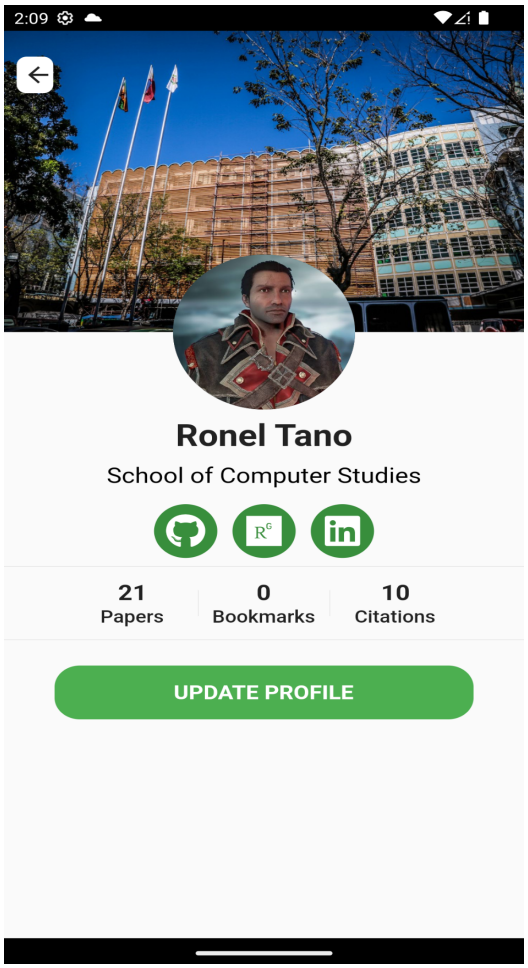


Figure 14 shows the update profile page

This screen shows the user's information as well as the number of research papers and bookmarks. Users can also update their profiles by changing their names.

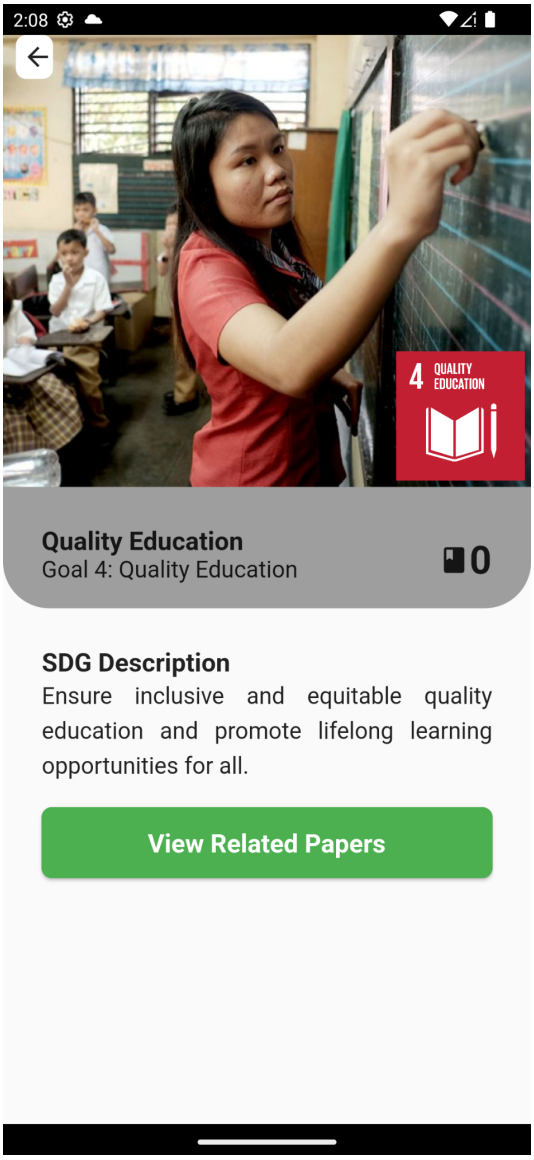


Figure 15 shows the screen for a particular SDG

This screen is shown when the user taps on a particular SDG icon. This screen contains the SDG name and description as well as a button to redirect users to research papers that are part of the goal.

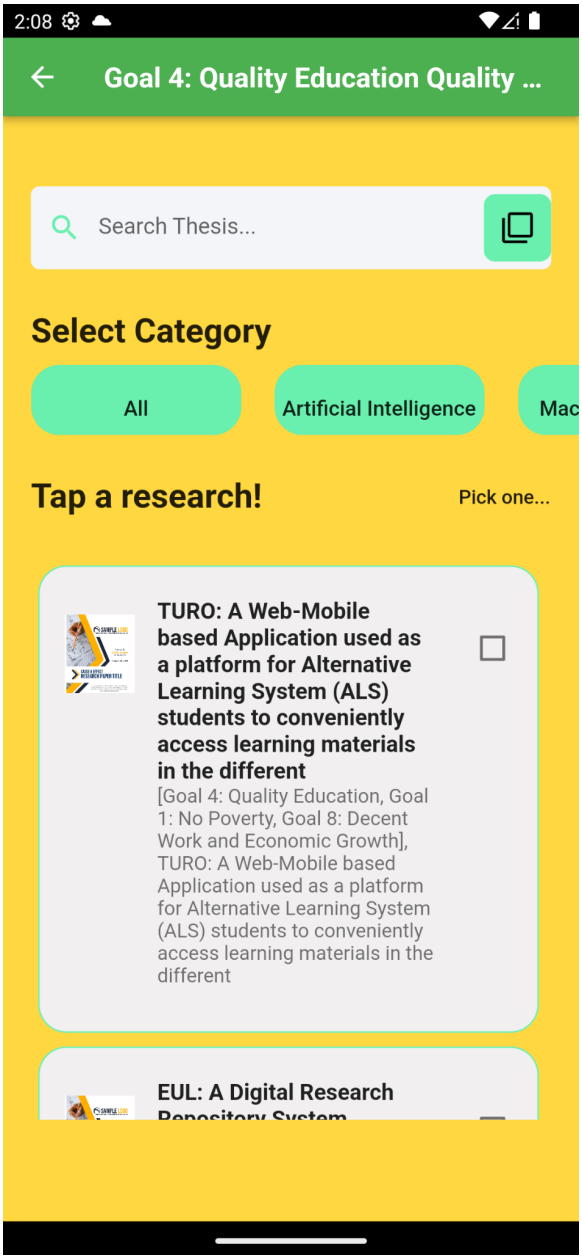


Figure 15 shows the dashboard for a particular SDG

This screen shows the dashboard when users click the button for viewing related papers from a particular SDG. This lists all research papers that are classified to that SDG.

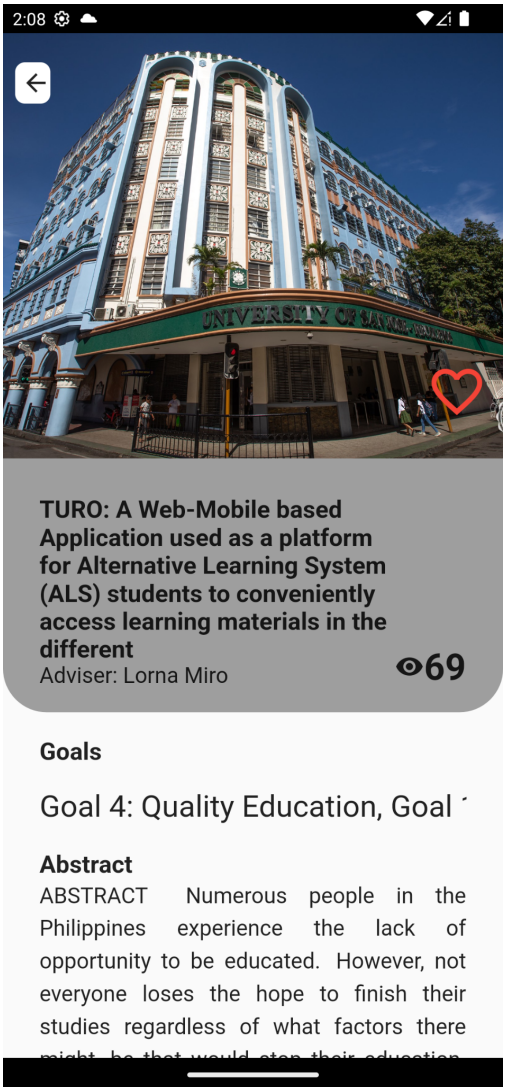


Figure 16 shows the view research paper's screen

This screen is shown when users click a research paper. The user can view the title, goals, and abstract, users can also view the PDF file. There's also a button for bookmarks.

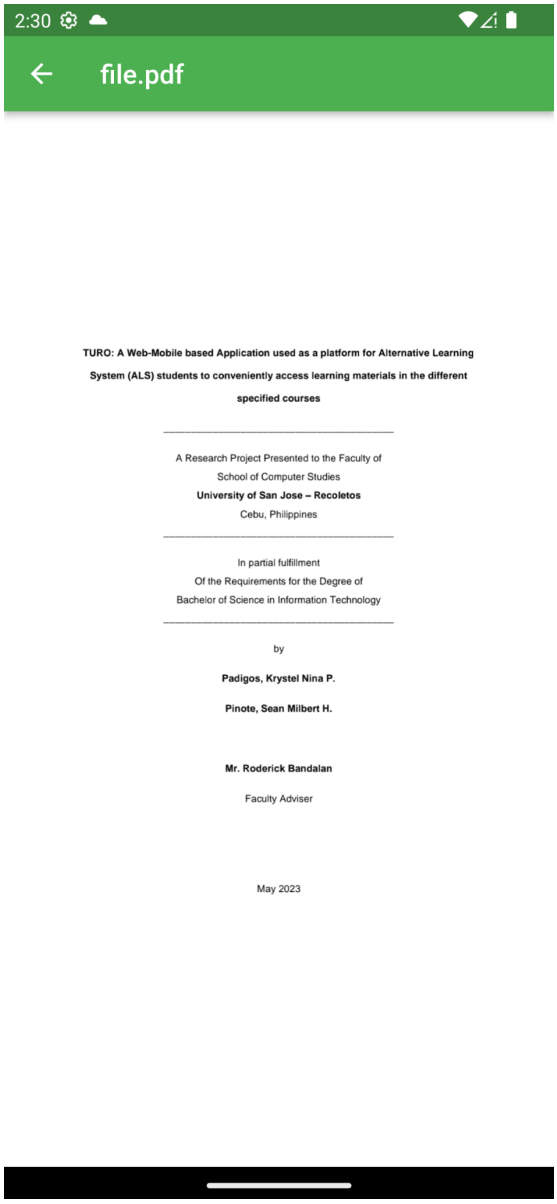


Figure 17 shows the screen for viewing PDF files

This screen allows the user to view and read the pdf file for a particular research paper.

Figure 18 shows the dashboard of the user's uploaded papers

This screen shows all of the user's research papers. Users can navigate and choose which of the research papers the user wants to view.

Figure 19 shows the dashboard of all user's bookmarks

This screen allows the users to view their bookmarked research papers.

Figure 20 shows the first screen in the upload process

This screen is shown when the user taps the upload button in the main dashboard. Here the user is introduced to the upload process of EUL.

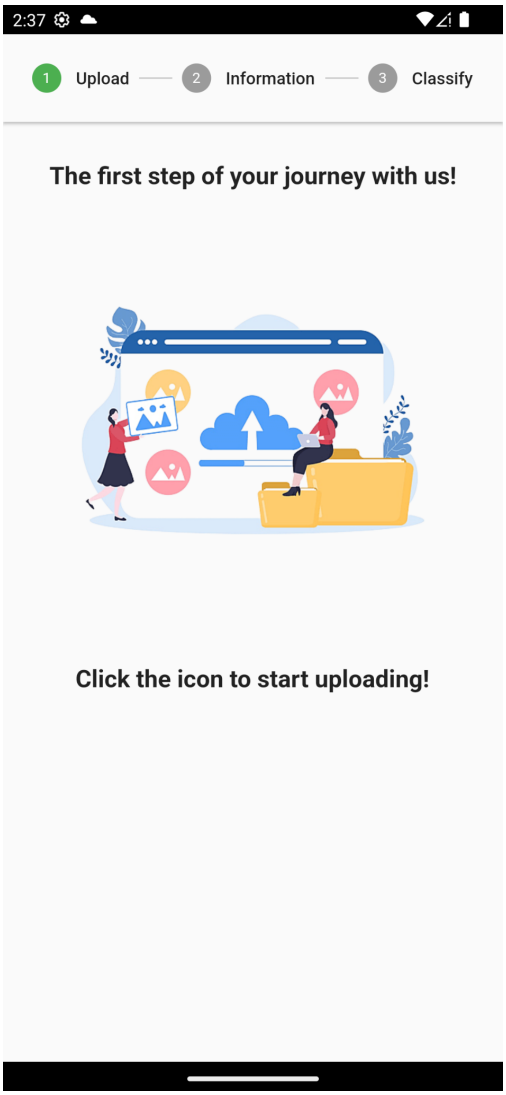


Figure 21 shows the first page of the stepper in uploading the paper

This screen shows how the user can upload a PDF file into the system. Users can tap the image to start uploading research papers.

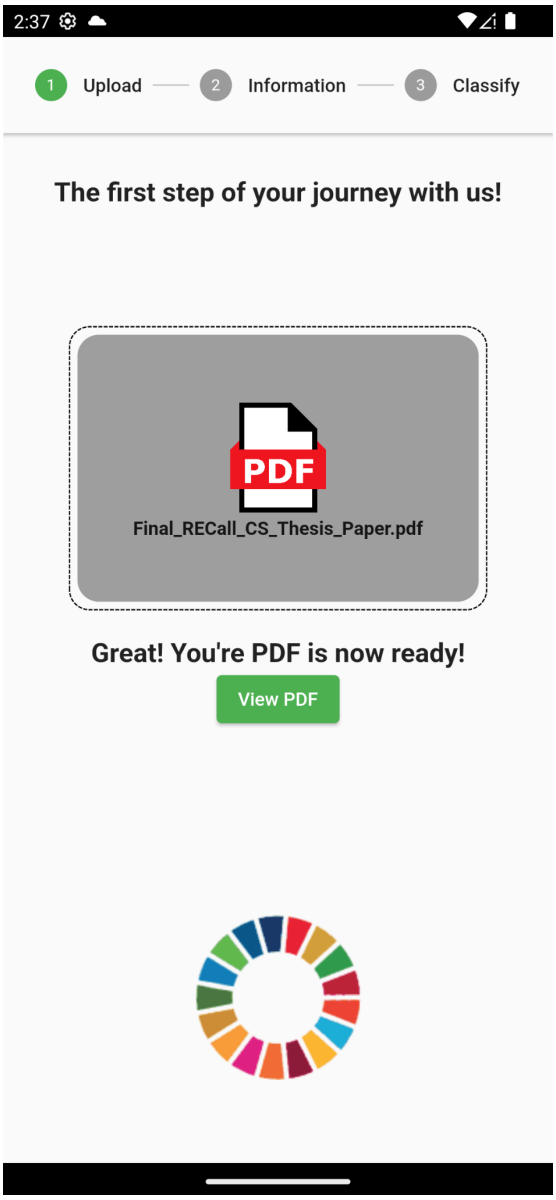


Figure 22 shows the screen when the user picks a file to upload

This screen is shown when the user picks a pdf file from their phone. If fetched from the system successfully a new button will appear to proceed to the next step.

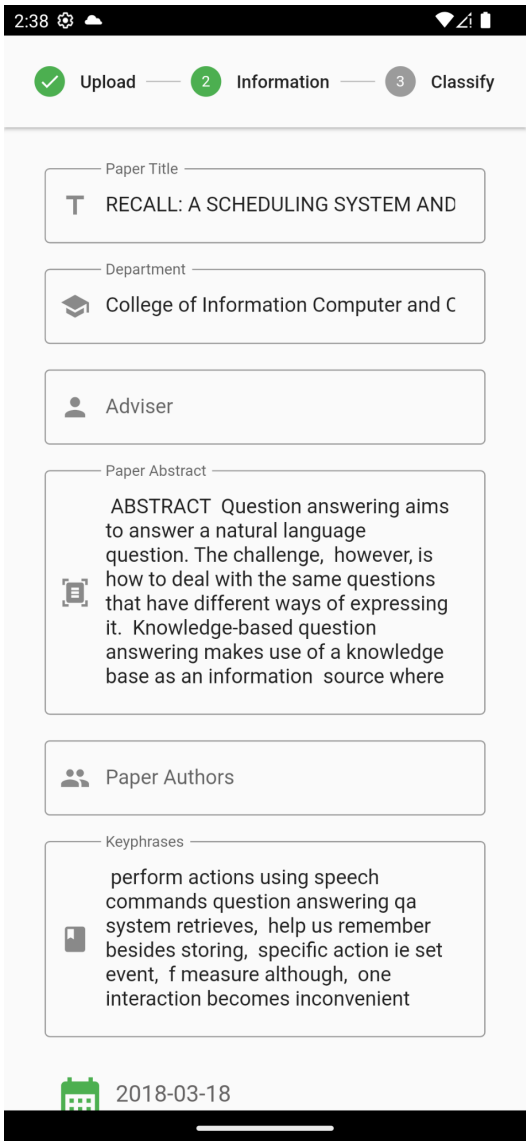


Figure 23 shows the screen when the user is done uploading a file.

This screen will be shown to the user once the paper uploaded will be accepted by the system. This screen shows the extracted text from the uploaded file.

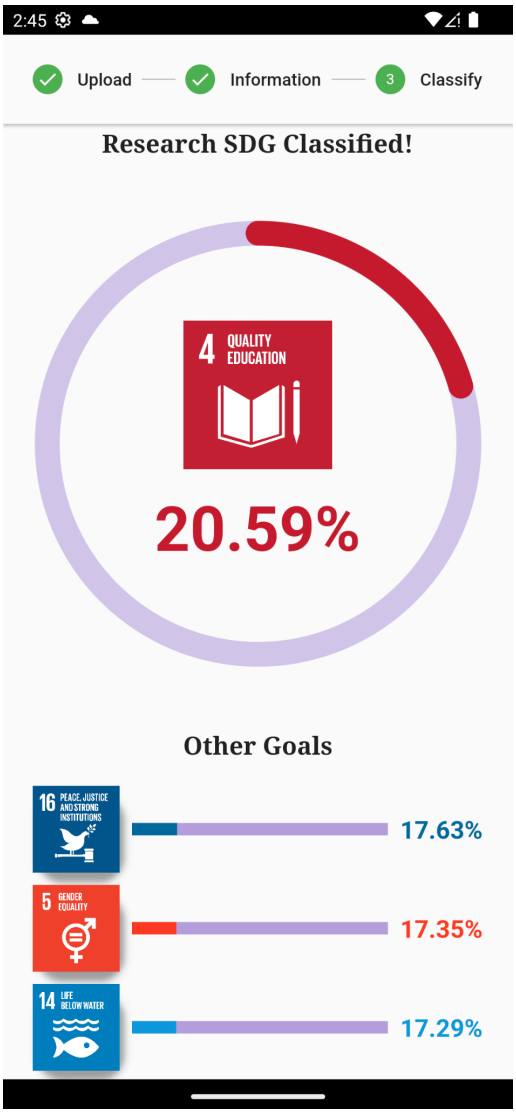
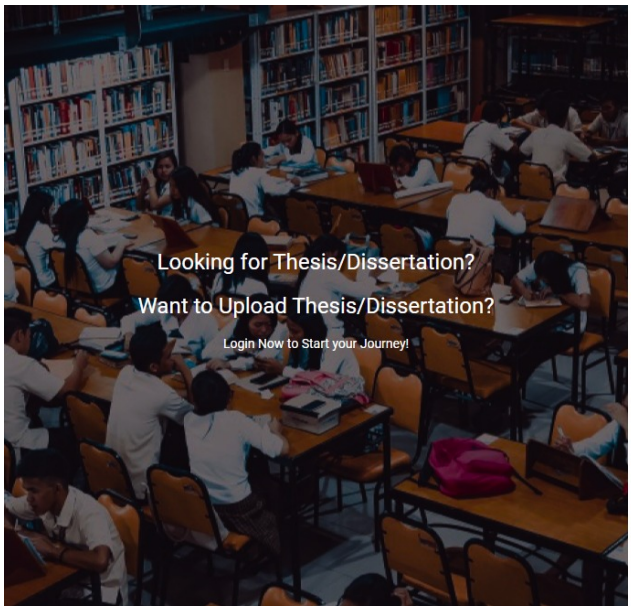
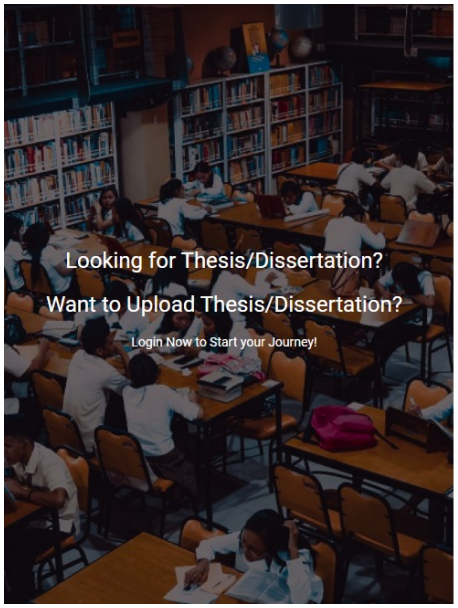


Figure 24 shows the final screen of the upload process

This screen is shown after the user clicks the button to classify research. This displays the top 4 goals predicted by the algorithm. The uppermost shows the highest score of all the goals.

Web Application





Registration Page

1 Personal Information — Login Information — 3 Done

Last Name *

First Name *

Email *

Enter your School ID *

0/10

Next Back to Login

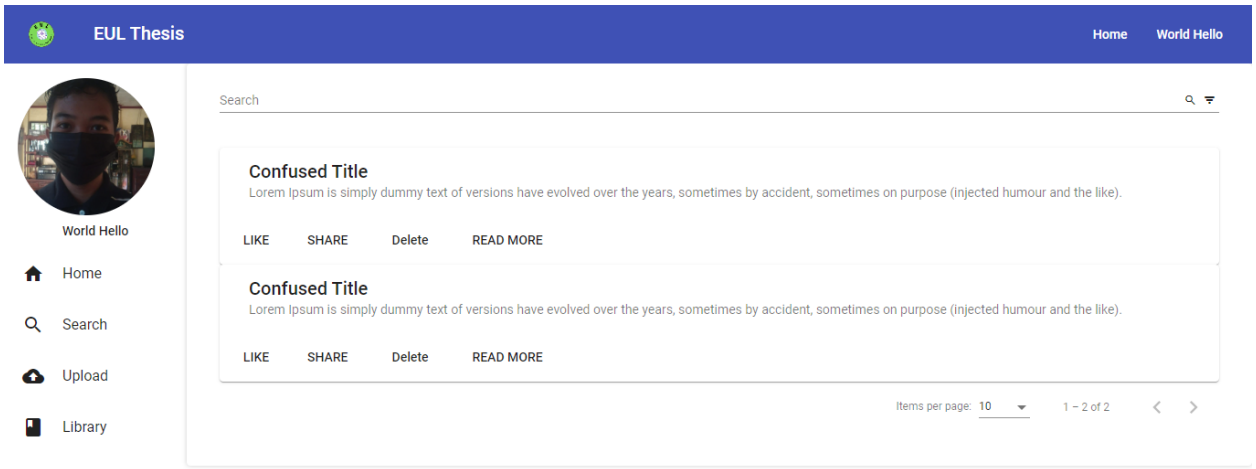
Your account is not approved yet. Please contact your administrator.

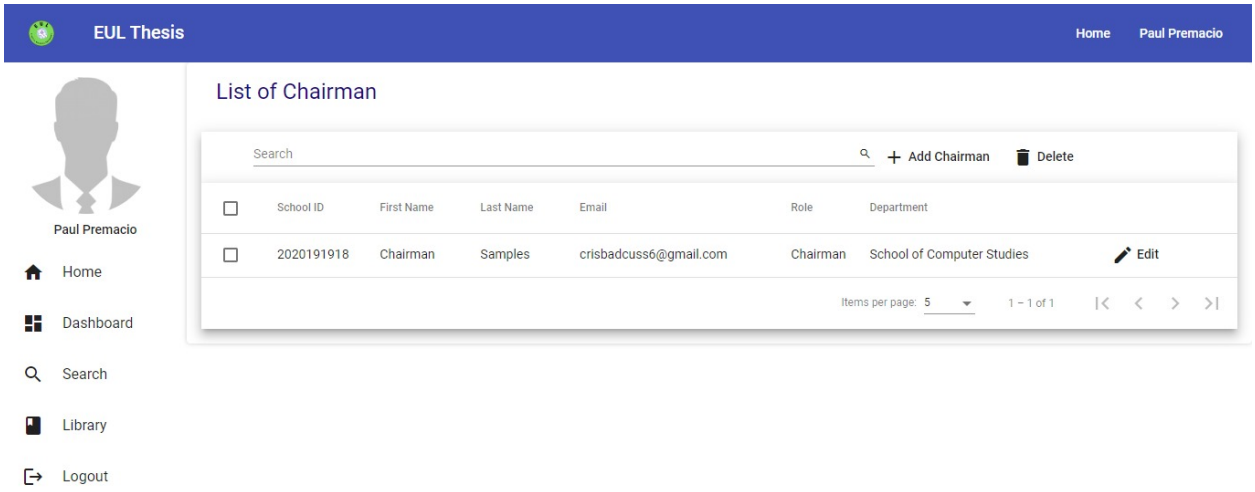
IMPORTANT NOTICE

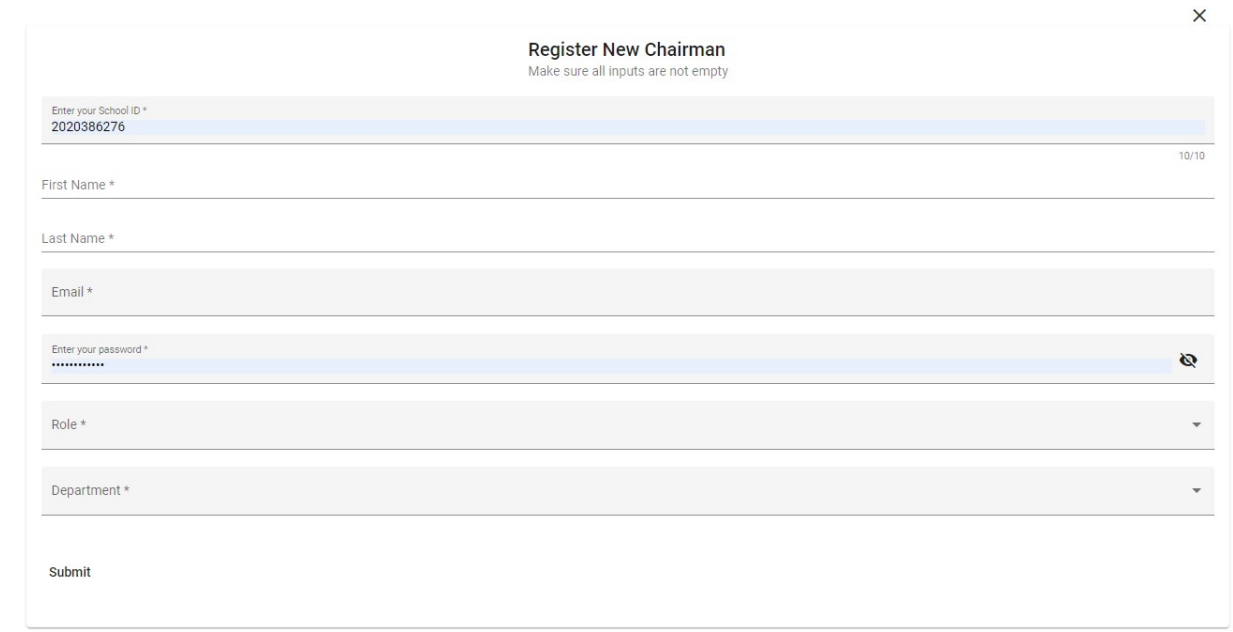
Thank you for registering. Your account is currently pending approval from the chairman. You will be notified via email once your account has been approved.

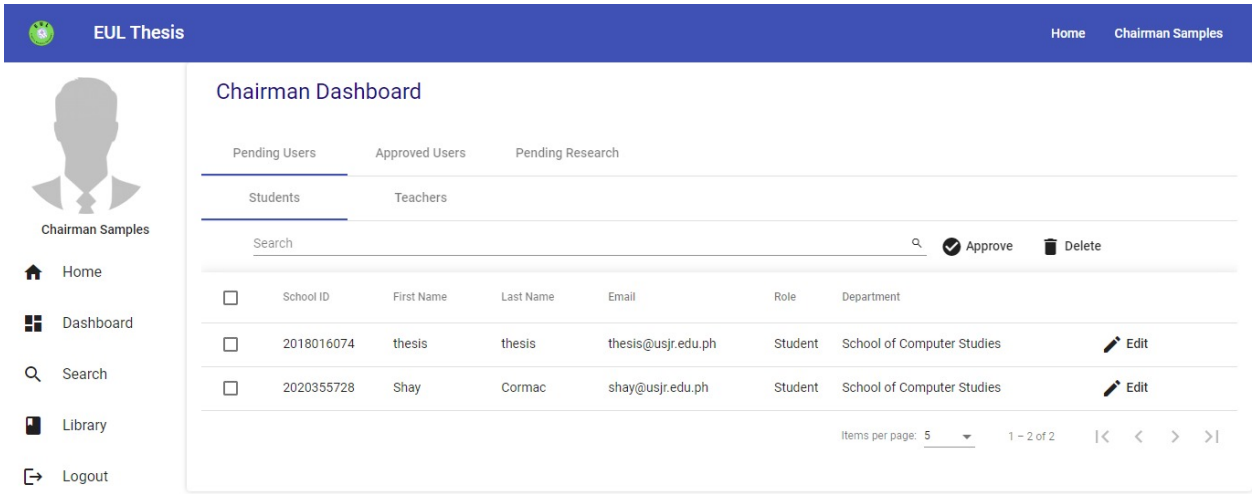
Login


EUL Thesis
Home
World Hello
World Hello
Home
Search
Upload
Library
My Research
Logout
POVERTY
2 ZERO HUNGER
3 GOOD HEALTH AND WELL-BEING
4 QUALITY
5 GENDER EQUALITY
Prev
1/4
Next


EUL Thesis
Home
World Hello
World Hello
Home
Search
Upload
Library
My Research
Logout
Search
Confused Title
Lorem Ipsum is simply dummy text of versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like).
LIKE
SHARE
Delete
READ MORE
Confused Title
Lorem Ipsum is simply dummy text of versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like).
LIKE
SHARE
Delete
READ MORE
Items per page: 10
1 – 2 of 2


EUL Thesis
Home
Paul Premacio
List of Chairman
Search
Add Chairman
Delete
School ID
First Name
Last Name
Email
Role
Department
2020191918
Chairman
Samples
crisbadcuss6@gmail.com
Chairman
School of Computer Studies
Edit
Items per page: 5
1 – 1 of 1
Paul Premacio
Home
Dashboard
Search
Library
Logout


Register New Chairman
Make sure all inputs are not empty
Enter your School ID *
2020386276
10/10
First Name *
Last Name *
Email *
Enter your password *
Role *
Department *
Submit


EUL Thesis
Home
Chairman Samples
Chairman Samples
Home
Dashboard
Search
Library
Logout
Chairman Dashboard
Pending Users
Approved Users
Pending Research
Students
Teachers
Search
Approve
Delete
School ID
First Name
Last Name
Email
Role
Department
2018016074
thesis
thesis
thesis@usjr.edu.ph
Student
School of Computer Studies
Edit
2020355728
Shay
Cormac
shay@usjr.edu.ph
Student
School of Computer Studies
Edit
Items per page: 5
1 – 2 of 2

Chairman Samples

- Home
- Dashboard
- Search
- Library
- Logout

Chairman Dashboard

Pending Users | Approved Users | Pending Research

Students | Teachers

Search

Approve | Delete

	ResearchID	Adviser	Date Published
☐	9ddb12f0-7064-4199-9741-c238df7b1586	Venice Steffen	February 2, 1974

Title

Est dolorum libero blanditiis.

Voluptate qui reprehenderit. Maxime quia nihil. Expedita eius quam. Voluptatem delectus ut. Aliquam debitis sit. Ipsum incidunt sit. Aperiam ab ut. Veniam et sint; velit corporis exercitationem?

Topic Category:

Hello World | SDG | SDG 2 | Goal

SDG Category:

Goal 5: Gender Equality

29.82%

Goal 4: Quality Education

28.52%

Goal 8: Decent Work and Economic Growth

32.66%

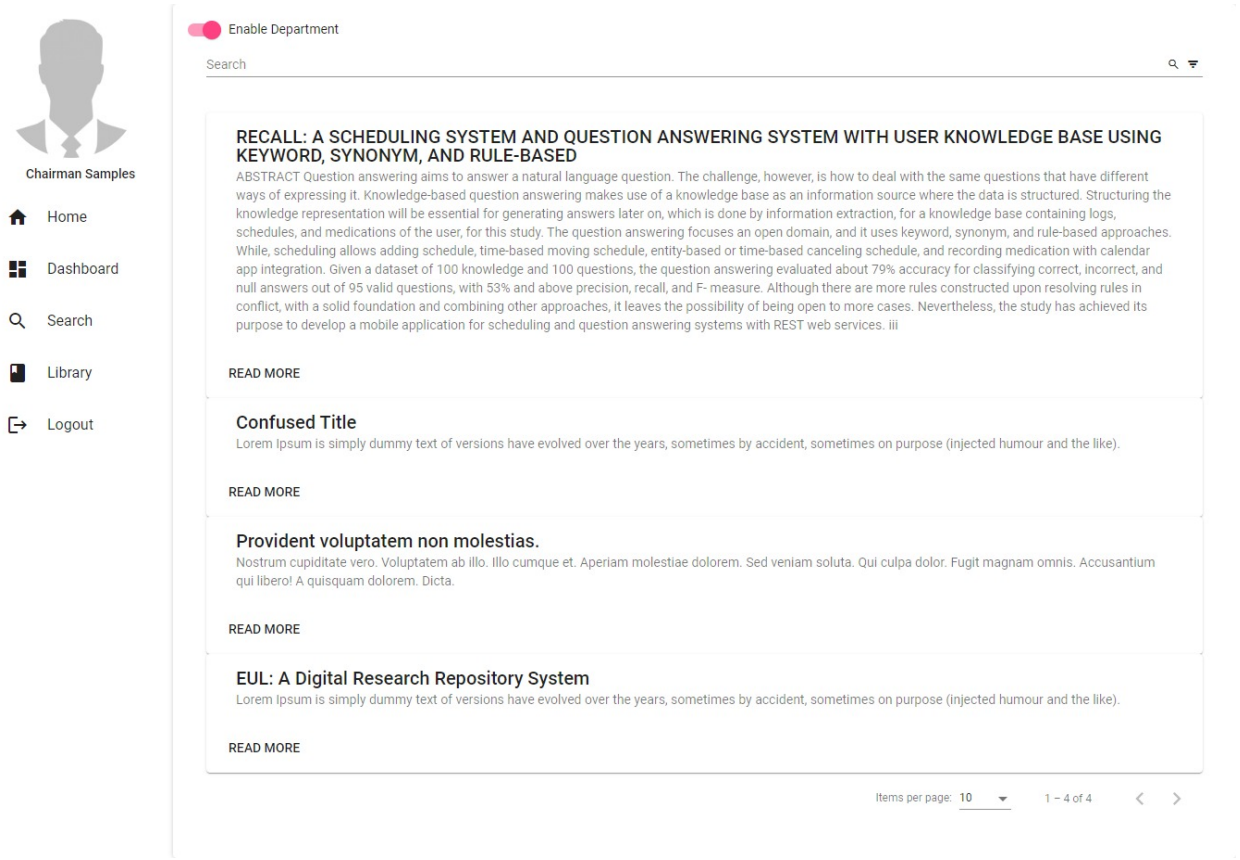
Keywords:

Hello World | SDG | SDG 2 | Goal

Authors:

Register ACcount | Paul Josua Premacio

Items per page: 5 | 1 – 1 of 1


Chairman Samples
Home
Dashboard
Search
Library
Logout
Enable Department
Search
RECALL: A SCHEDULING SYSTEM AND QUESTION ANSWERING SYSTEM WITH USER KNOWLEDGE BASE USING KEYWORD, SYNONYM, AND RULE-BASED
ABSTRACT Question answering aims to answer a natural language question. The challenge, however, is how to deal with the same questions that have different ways of expressing it. Knowledge-based question answering makes use of a knowledge base as an information source where the data is structured. Structuring the knowledge representation will be essential for generating answers later on, which is done by information extraction, for a knowledge base containing logs, schedules, and medications of the user, for this study. The question answering focuses an open domain, and it uses keyword, synonym, and rule-based approaches. While, scheduling allows adding schedule, time-based moving schedule, entity-based or time-based canceling schedule, and recording medication with calendar app integration. Given a dataset of 100 knowledge and 100 questions, the question answering evaluated about 79% accuracy for classifying correct, incorrect, and null answers out of 95 valid questions, with 53% and above precision, recall, and F- measure. Although there are more rules constructed upon resolving rules in conflict, with a solid foundation and combining other approaches, it leaves the possibility of being open to more cases. Nevertheless, the study has achieved its purpose to develop a mobile application for scheduling and question answering systems with REST web services. iii
READ MORE
Confused Title
Lorem Ipsum is simply dummy text of versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like).
READ MORE
Provident voluptatem non molestias.
Nostrum cupiditate vero. Voluptatem ab illo. Illo cumque et. Aperiam molestiae dolorem. Sed veniam soluta. Qui culpa dolor. Fugit magnam omnis. Accusantium qui libero! A quisquam dolorem. Dicta.
READ MORE
EUL: A Digital Research Repository System
Lorem Ipsum is simply dummy text of versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like).
READ MORE
Items per page: 10
1 – 4 of 4

yehey!

EUL Thesis

Home Cristopher Bohol Logout

Cristopher Bohol

- Home
- Dashboard
- Search
- Library
- Logout

Go Back

Search

All | Department's Research | Teacher's Research | Student's Research

Unde voluptatem iste ullam eum.

Voluptas et perspiciatis. Non quod recusandae! Quia doloremque mollitia. Eveniet officia dignissimos! Recusandae hic natus. Omnis vitae consectetur. Numquam omnis iste; nihil deleniti debitis. Esse.

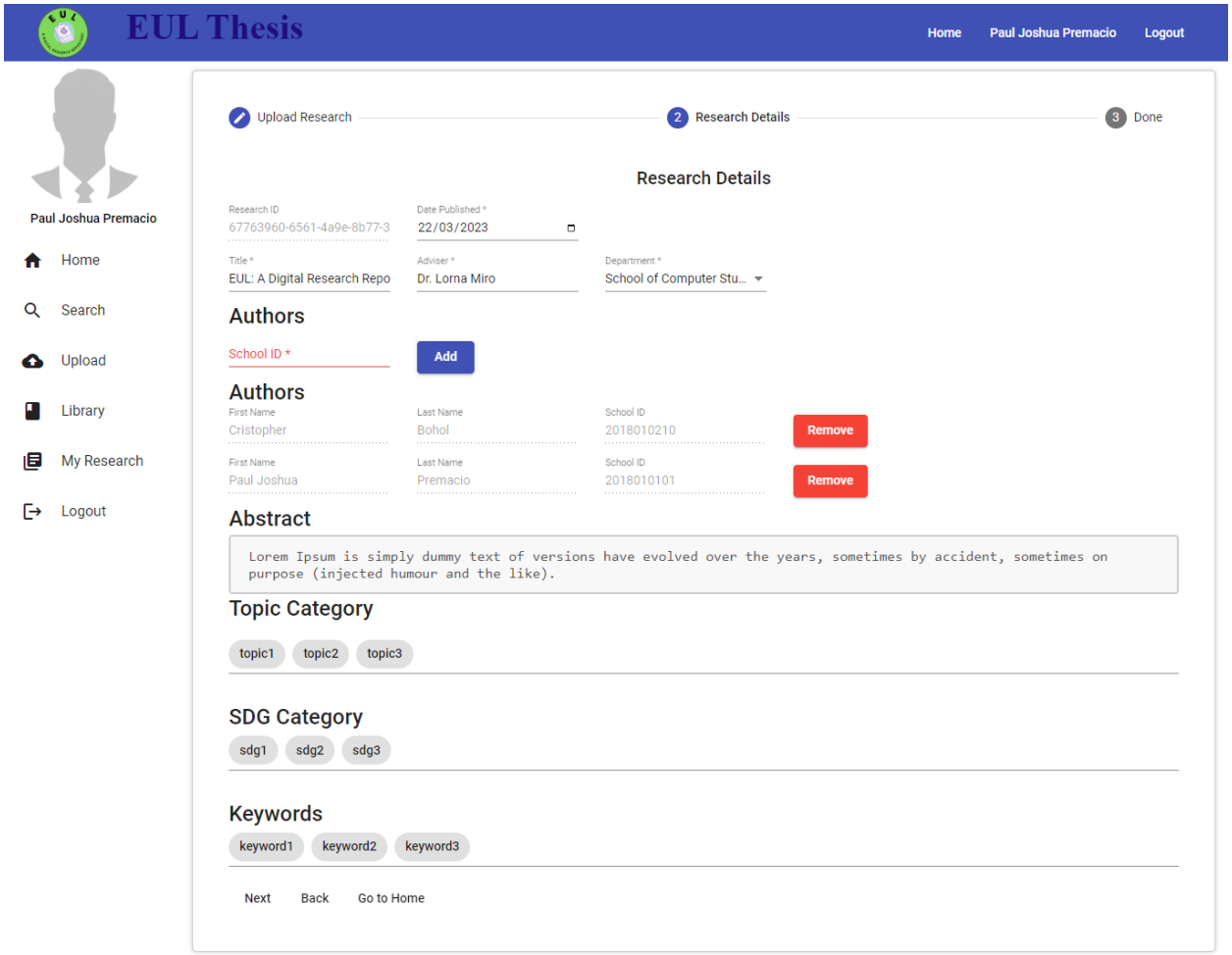
READ MORE

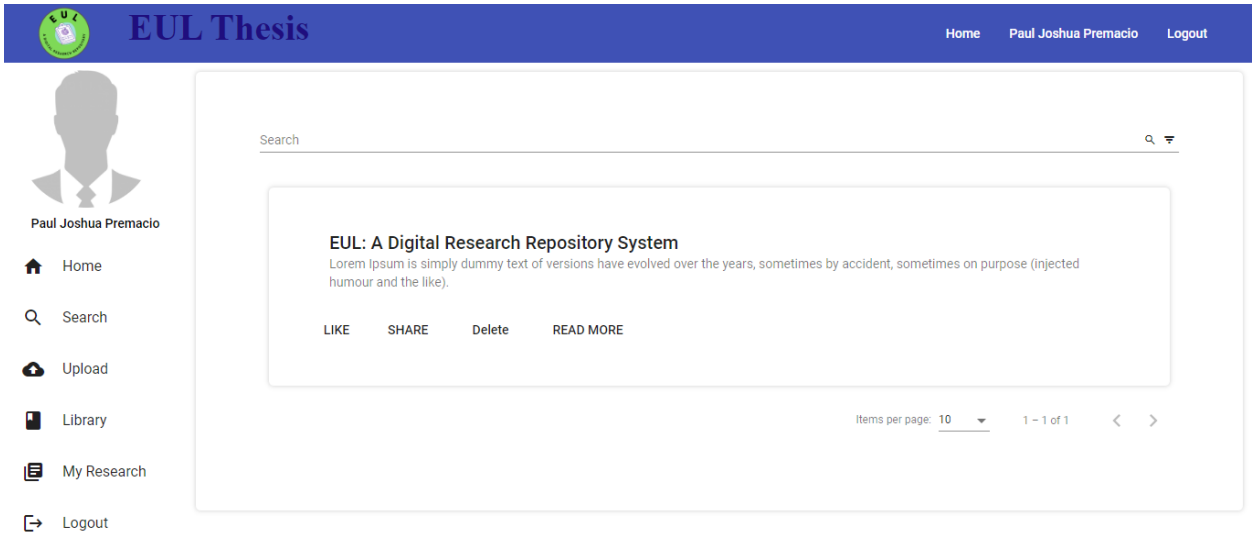
Vero iste consectetur aut atque.

Nihil voluptatem amet. Aliquam rem vero. Eum odio necessitatibus. Dolores perspiciatis magnam. Perferendis repellendus corporis. Nesciunt rerum molestias. Nam sit labore. Suscipit neque consequuntur;

READ MORE

Items per page: 10 1 – 2 of 2


EUL Thesis
Home
Paul Joshua Premacio
Logout
Paul Joshua Premacio
Home
Search
Upload
Library
My Research
Logout
Upload Research
2 Research Details
3 Done
Research Details
Research ID
67763960-6561-4a9e-8b77-3
Date Published *
22/03/2023
Title *
EUL: A Digital Research Repo
Adviser *
Dr. Lorna Miro
Department *
School of Computer Stu...
Authors
School ID *
Add
Authors
First Name
Cristopher
Last Name
Bohol
School ID
2018010210
Remove
First Name
Paul Joshua
Last Name
Premacio
School ID
2018010101
Remove
Abstract
Lorem Ipsum is simply dummy text of versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like).
Topic Category
topic1
topic2
topic3
SDG Category
sdg1
sdg2
sdg3
Keywords
keyword1
keyword2
keyword3
Next
Back
Go to Home


EUL Thesis
Home
Paul Joshua Premacio
Logout
Paul Joshua Premacio
Home
Search
Upload
Library
My Research
Logout
Search
EUL: A Digital Research Repository System
Lorem Ipsum is simply dummy text of versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like).
LIKE
SHARE
Delete
READ MORE
Items per page: 10
1 – 1 of 1

CHAPTER III

SOFTWARE DEVELOPMENT AND TESTING

This chapter describes the implementation of the project in development, the various tools used to create and run the application, and testing methods to evaluate the question-answering results. It contains sections for the Development and Testing Process.

Development Software Platforms, Development Environments, and Tools

The system can be accessed via mobile and web platforms. Each platform has its own technologies implemented. The process is a hybrid of the Agile and Prototyping Models. An initial prototype is created and refined as the development progressed.

For the web platform, these are the tools and development environment used:

- Visual Studio Code - a lightweight IDE commonly used by developers. This is used to run the REST API services and the web frontend.
- Node.js - a javascript framework used in this system as the backend environment.
- Angular 14 - a javascript framework used by this system as its frontend technology.

For the mobile platform, these are the tools and development environment used:

- Flutter Framework 3.3.7 - a relatively new framework for developing mobile, web, and desktop applications developed by Google. This framework is using Dart as its main language.
    - http v0.13.4 - an open-source package by Flutter used primarily in handling http requests.
    - json_serializable v6.3.1 - an open source package for easy handling of JSON, for example converting JSON to Maps, or Maps to JSON.
    - path_provider v2.0.11 - an open-source package for accessing and finding commonly accessed locations in the file system.

- Android Studio version 2021.2 - is used by this system to run the emulator for debugging and testing.
- Visual Studio Code - the primary IDE used by the developer for fast and smooth development.

For the Python Backend, these are the tools and development environment used:

- Visual Studio Code - IDE used for developing and running Python scripts.
- python 3.10.5 - the language used for creating the algorithms used by this system.
- Python Flask 2.2.3 - a Python framework for building web applications. This system used the framework in serving the Python scripts to be used by the web and mobile platforms in the form of http requests.
- nltk 3.7 - Natural Language Toolkit, a package used for handling all features related to natural language processing like lemmatization, POS Tagger, etc.
- pandas 1.4.3 - a python library for handling data analysis and structures. This library is used by the system primarily for converting dictionaries to CSV files.
- pdfplumber 0.7.5 - a Python library for handling all tasks related to PDF files.
- pytesseract 0.3.10 - a Python library used for handling OCR tasks.
- numpy 1.24.3 - a Python library used for handling mathematical operations on arrays. It is used for efficient calculations on arrays.

For Database Server:

- MySQL Workbench - digital filing cabinet for information, helps create, edit, view tables, run queries, manage user accounts, and server settings. Useful for database admins and developers.

DEVELOPMENT PROCESS

Preparing the Data for Training

The study uses a supervised learning algorithm to classify papers thus the first step is to carefully chose which papers fetched from google scholar or other trusted research portals to include in the training set. Here, it is already manually renamed to which goal it belongs.

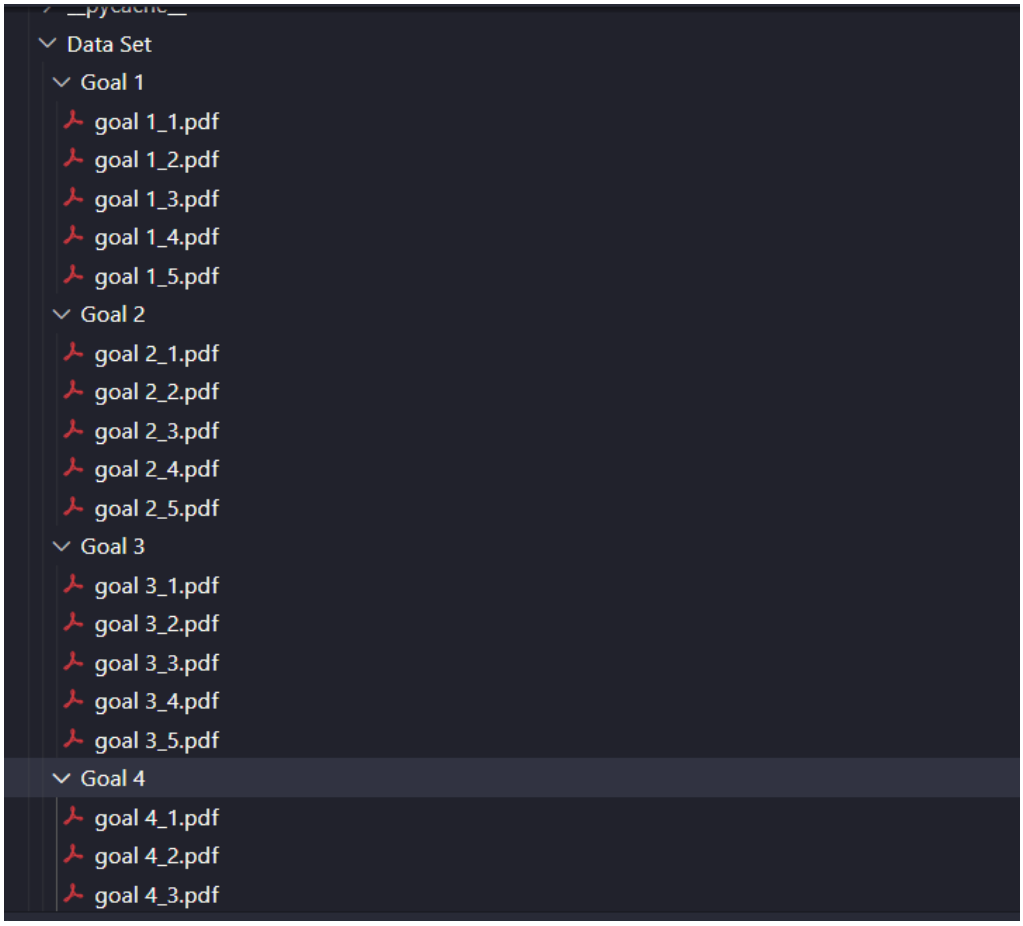


Figure N: Directory of the Training Set with prelabeled PDF files

```python
def extractAllPDF(self, goal):
    directory = (glob.glob("tfidf/Data Set/" + goal + "/*.pdf"))
    extractedText = " "
    finalText = " "
    for file in directory:
        with pdfplumber.open(file) as pdf:
            for page in pdf.pages:
                extractedText = page.extract_text()
                finalText = finalText + extractedText
    return finalText
```

Figure N: Function for opening the directory on which the training data are located

The function will automatically navigate to the defined directory with the name of the goal. Ex. tfidf/Data Set/Goal 1/. After locating the goal's folder, the function identifies which of the files inside the goal folder is a pdf. All of the files are already in PDF format so everything inside will be used for extraction.

Figure N: Goal 1 Folder with all of its PDF files

It will iterate over each of the files, extract all strings inside and store it into the finalText variable. This is repeated until the function reaches the final folder which is Goal 17.

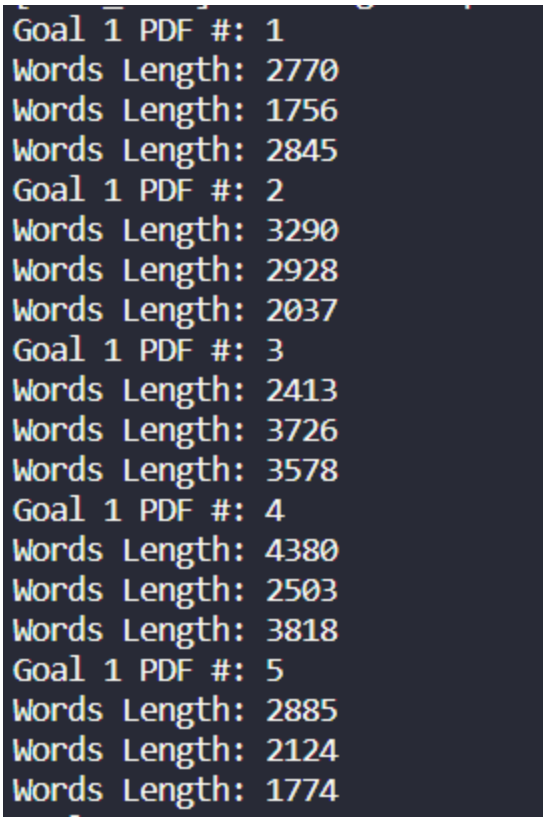


Figure N: Word Counts per PDF

This visualizes the word count for each PDF file inside the Goal 1 Folder. The PDF files are trimmed to only contain 3 pages each to maximize computational efficiency.

```
    Human SettlementsThe Evolution of Urban Water Management
To effectively address the emerging challenges in water management in cities, it is instructive to first
understand the historical evolution of urban water management regimes (i.e. infrastructure and
institutions). Traditionally, urban water has been managed in a technocratic way, based on principles of
predictability and control. Brown et al. (2009) investigated the evolution of urban water management
in cities over the last 200 years and considered a series of sustainable futures perspectives.5 As shown
in Figure 1, they developed a typology of six dominant water management regimes that represent a
nested continuum of socio-political drivers and service delivery responses, i.e. water supply, sewered,
drained, waterways, water cycle and water sensitive cities. This framework has been used to inform the
development of many urban water management strategies in Australia, (notably Water for Victoria6,
South Australia's Water for Good7), and internationally (e.g. Asian Development Bank (2013)8).
Figure 1: Evolution of Urban Water Management Regimes
(source: Asian Development Bank (2013) adapted from Brown et al 2009)
The first three regimes represent the historical development of water servicing in response to the need
to provide (i) clean and reliable water supplies, (ii) better public health outcomes and (iii) protection
from flooding. In these regimes, water services are provided through large, centralised infrastructure
and the associated yet siloed administrative/governance systems managed on the community's behalf
by utilities. The corresponding urban water infrastructure in almost all developed cities exhibit these
characteristics, with low community water literacy9 and urban water services largely invisible and typically
taken for granted.
Managing the water cycle in this segmented and linear way, whereby wastewater and stormwater
are swiftly channeled outside of the city and into receiving waterways, has given rise to a range of
unintended consequences, including environmental degradation.
5  Brown, R.R., Keath, N., & Wong, T.H.F. (2009), 'Urban water management in cities: historical, current and future regimes', Water, Science and
Technology, 59(5), 847-855.
6  http://delwp.vic.gov.au/water/a-new-water-plan-for-victoria
7  www.environment.sa.gov.au/files/.../water/water-for-good-full-plan.pdf
8  http://www.adb.org/sites/default/files/publication/30190/asian-water-development-outlook-2013.pdf
9  http://watersensitivecities.org.au/are-australians-water-literate/
3
A Framing Paper for the High-Level Panel on Water     This overwhelming demand on natural capital has left many cities with a reduced environmental capacity
to assimilate and process pollution, which in turn compromises water supply security (especially for
downstream urban environments) and urban liveability. The existing legacy of environmental damage
is especially problematic, with the majority of developed cities characterised by an ecological footprint
much larger than the physical footprint of the city.10 These challenges are further exacerbated by rapidly
increasing urban population that is not well supported by aging infrastructure and inadequate ongoing
investment in its augmentation, and the unpredictable and diverse nature of climate change impacts,
causing drought in some places (e.g: many Australian, Indian and Brazilian cities etc in recent times)
and severe flooding in others (e.g: the recent floods in Jakarta, Chennai, Houston, cities in Great Britain,
just to name a few).
The next two regimes, the Waterways and Water Cycle cities, reflect the response of urban water
systems (particularly in the developed world) to new challenges and socio-political drivers characterised
```

Figure N: Raw Data

After going through all of the pdf files in a goal's folder. This is what the raw extracted text looks like. As expected of raw data, it still contains a lot of unnecessary text.

Once this is done, the result will then be stored on a list. This process will repeat until all goals are scraped and stored on a list.

```
for goal in goals:
    trainingData = self.extractAllPDF(goal)
    trainingDocs.append(trainingData)
```

Figure N: Iterate each goal

The trainingDocs list represents the goals in the corpus, this is still raw data and has not been preprocessed yet.

Once all goals are appended to the list, the system performs text preprocessing.

```
preProcessedDocs = self.preprocess_documents(documents)
unique = self.getUniqueWords(preProcessedDocs, False)
```

Figure N: Preprocessing and Getting Unique Words

The variable documents represents the list of unprocessed text from each goal. The list contains 17 items as there are 17 goals in the entire corpus.

```
def preprocess_documents(self, docs):
    stop_words = set(stopwords.words("english"))
    lemmatizer = WordNetLemmatizer()
    lemmatized_tokens = []
    for doc in docs:
        # Tokenize document
        tokens = word_tokenize(doc)
        # Remove stopwords and punctuations, and convert to lowercase
        filtered_tokens = [
            token.lower() for token in tokens if token.isalnum() and token not in stop_words]
        # Lemmatize tokens
        lemmatized_tokens.append([lemmatizer.lemmatize(
            token) for token in filtered_tokens])

    return lemmatized_tokens
```

Figure N: Preprocessing

Preprocessing includes removing of stop words, punctuations, and numbers, converting to lowercase, and lemmatization. All of these are done with the help of the nltk library. Each of the items in the list of unprocessed text represents each goal, and this function iterates over that list and performs text preprocessing. Once done, the system will create a new list of preprocessed data.

```
overview a world free from child poverty however d
the sustainable development goal offer tremendous
policy programme meet child poverty reduction goal
worked build global best practice provide support
```

Figure N: Snippet of Preprocessed Data

After preprocessing, the text is now normalized and ready to be used for the next step.

```
def getUniqueWords(self, preProcessedDocs, type):
    unique = {}
    for str in preProcessedDocs:
        for str2 in str:
            unique[str2] = 0
    return unique
```

Figure N: Snippet of Preprocessed Data

The next step for training the model is the identification of the unique words in the entire 17 documents. These words represent the feature set.

	Value
overview	0
a	0
world	0
free	0
from	0
...	...
flower	0
color	0
categorize	0
venation	0
optical	0

Figure N: Dictionary of Unique Words

These unique words are then stored in a dictionary data structure for easy retrieval and storage.

Text Vectorization

After creating the dictionary of unique words, the next step is the calculation of the TF-IDF. The first step in this process is to create the Term Frequency of each word with respect to its parent goal. All words in Goal 1 have their own Term Frequency, the same as Goal 2 until Goal 17.

```
for listOfTokens in preProcessedDocs:
    tf.append(self.getTermFreq(
        unique, len(listOfTokens), listOfTokens))
    tv.append(self.getTerm(unique, len(listOfTokens), listOfTokens))
```

Figure N: Iterating over the entire Preprocessed List

The formula for term frequency is word frequency / total count of words in the document. In this case, Goal 1 has a total count of 4019 words.

```
            Value
overview        2
a               7
world          25
free            3
from            4
...           ...
flower          0
color           0
categorize      0
venation        0
optical         0

[10393 rows x 1 columns]
Total Number of Words in Goal 1:  4019
```

Figure N: Term Vectors for Goal 1

This visualizes the term vectors of goal 1. Each word is assigned a value that represents the occurrence of that word in the Goal 1 document.

	Value
overview	0.000498
a	0.001742
world	0.006220
free	0.000746
from	0.000995
...	...
flower	0.000000
color	0.000000
categorize	0.000000
venation	0.000000
optical	0.000000

Figure N: Term Frequency for Goal 1

After getting the occurrence of the words in Goal 1, the system will proceed in calculating its term frequency. By applying the formula mentioned before, the result will look like this. There are words that have zero values because these words are not present in the Goal 1 document.

```
def inverse(self, unique, preProcessedDocs, tf=[{}]):
    num_of_docs = len(preProcessedDocs)
    idf, finalIDF = {}, {}
    for str in unique:
        idf[str] = 0
    for str in idf:
        for str2 in tf:
            if str in str2:
                idf[str] = idf[str] + str2[str]
    for str in idf:
        finalIDF[str] = math.log10(num_of_docs / idf[str])
    return finalIDF
```

Figure N: Code Snipper for IDF

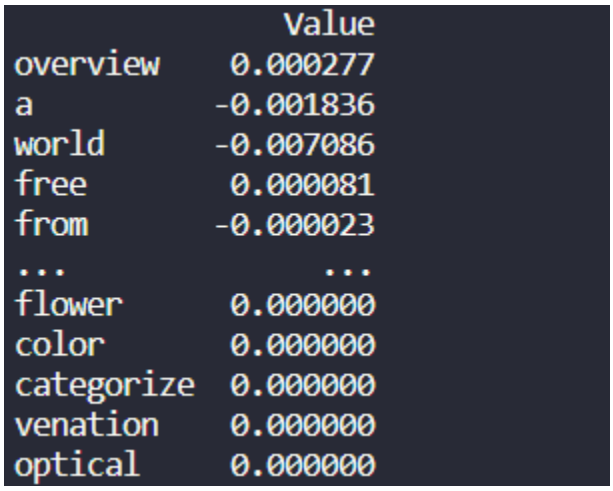
After getting all Term Frequencies in the entire list of processed data, the next step is getting the inverse document frequency

	Value
overview	0.556303
a	-1.054358
world	-1.139179
free	0.109144
from	-0.023481
...	...
flower	1.255273
color	0.954243
categorize	1.255273
venation	1.255273
optical	1.255273

Figure N: Result of Inverse Document Frequency for Goal 1

The formula for IDF is log(N/DF), where N is the total number of documents, and DF is the document frequency.

Inverse Document Frequency is a measure that helps identifies the relevance of a particular term in the entire collection of documents. The lesser the value, the more common it is, and the higher, the more rare or unique the word is. As in the image snippet, the term *world* has an IDF of -1.139179, while the term flower has an IDF value of 1.2555, this means that the term *world* appears more frequently across the 17 documents and is very irrelevant, whilst the term flower appears more frequent across all documents.



	Value
overview	0.000277
a	-0.001836
world	-0.007086
free	0.000081
from	-0.000023
...	...
flower	0.000000
color	0.000000
categorize	0.000000
venation	0.000000
optical	0.000000

Figure N: Final TF-IDF Values

After getting the IDF values, the final step is to get the Final TFIDF, which has a very straightforward calculation, TF x IDF. The image represents the TFIDF value for Document 1 which in this case is Goal 1, the term overview has a value of 0.000277, while terms like flower, color, etc. have values of 0, which means that these values don't have any relevance to Goal 1 whatsoever.

```
def calculateTFIDF(self, listofDict, idf, tf_idf):
    temp = {}
    count = 1
    for list in listofDict:
        temp = list
        for features in idf:
            if temp.__contains__(features):
                temp[features] = temp[features] * idf[features]
        tf_idf.append(temp)

    return tf_idf
```

Figure N: TF-IDF calculation

The last process is to store the final TF-IDF for the entire SDG corpus into a CSV file. This CSV file visualizes how the TFIDF looks after appending all of the 17 documents.

	overview	a	world	free	from	child	poverty	however	despite	urgency	availability	proven	approach	measure	respond	received	relatively	little	attention	global	struggle	the
0	0.000277	-0.00184	-0.00709	8.15E-05	-2.34E-05	-0.01535	-0.0491	-0.00134	-0.00037	0.000237	-0.00022	0.000119	-0.00145	-0.00191	0.000175	-5.00E-05	-6.22E-05	7.49E-05	-0.00014	-0.0079	0.000475	-0.0221
1	0	-0.00267	-0.00173	0	0	-0.00555	-0.00132	-0.00151	-0.00012	0	-5.08E-05	8.05E-05	-0.00098	-0.0006	0	-2.26E-05	-0.00013	0	-3.24E-05	-0.0024	0	-0.00916
2	0	-0.00504	-0.00306	0	0	-0.00031	0	-0.00214	0	0	-8.99E-05	0.000143	-0.00087	-0.00053	0	0	0	8.99E-05	-5.73E-05	-0.00098	0	-0.01867
3	8.65E-05	-0.00279	-0.00284	1.70E-05	-1.10E-05	-0.00959	-0.0014	-0.00251	-0.00063	0.000148	-4.68E-05	0	-0.00136	-0.00018	5.48E-05	0	-1.94E-05	0	-8.95E-05	-0.00358	0	-0.01867
4	0	-0.00686	-0.0121	3.74E-05	-8.04E-06	-0.00247	-0.00691	-0.00031	0	0	0	0	0	-0.0002	0.000121	-2.29E-05	0	0.000103	-6.57E-05	-0.0015	0	-0.02704
5	0	-0.00299	-0.00539	1.72E-05	-7.40E-06	-0.00049	-0.00018	-0.00212	-5.80E-05	0	-4.74E-05	0	-0.00023	-0.00019	0	-3.17E-05	-5.91E-05	4.74E-05	0	-0.00138	0	-0.01426
6	0	-0.00167	-0.00253	0	0	-0.00033	-0.00036	-0.00227	0	0	-9.54E-05	0	0	-0.00093	0.000223	0	-3.96E-05	0	-6.08E-05	-0.00625	0	-0.01563
7	0	-0.00269	-0.00087	2.79E-05	-1.20E-05	-0.00184	-0.00143	-0.00252	-0.00028	0	0	0	-0.0013	-0.00226	0	-1.71E-05	-6.38E-05	7.68E-05	-0.00024	0	0	-0.0214
8	0.000159	-0.0012	-0.00228	0	0	0	-0.00128	-0.00179	0	0	-8.60E-05	0	-0.00166	0	0.000101	-3.83E-05	0	0	-0.00011	-0.00031	0	-0.01739
9	0	-0.00285	-0.00476	5.37E-05	0	-0.00152	-0.00221	-0.00066	-0.00072	0	0	0	0	0	0	0	-6.14E-05	0	-9.44E-05	-0.00188	0	-0.01213
10	0	-0.00265	-0.00597	0	-1.97E-05	-0.00043	-0.00094	-0.00113	-7.71E-05	0	-0.00025	0	-0.00274	-0.00037	0	-5.61E-05	-2.62E-05	6.31E-05	-8.04E-05	-0.00413	0	-0.01447
11	0	-0.00408	-0.00206	0	0	0	0	-0.00092	-0.00019	0	-0.00016	0	-0.00112	-0.0003	0	-1.73E-05	-3.22E-05	0	0	-0.00621	0	-0.01738
12	0.000119	-0.00293	-0.00488	0	0	0	0	-0.00077	-0.00016	0	-0.00019	0.000102	-0.00062	-0.00013	0	0	-0.00011	0	-4.11E-05	-0.0082	0	-0.01689
13	0	-0.00233	-0.0028	0	0	-0.00025	-0.00028	-0.00132	-0.00018	0	-0.00015	0	-0.00036	0	0	0	-3.07E-05	0	0	-0.0035	0	-0.02262
14	0	-0.00255	-0.00138	0	-4.72E-06	-0.00021	-0.00045	-0.00325	-7.40E-05	0	-0.00073	0	-0.00088	-0.00047	0	-2.69E-05	0	0.000182	-7.72E-05	-0.0033	0	-0.01092
15	0	-0.00148	-0.00124	8.51E-05	-7.33E-06	-0.00176	-0.00035	-0.00126	-0.00017	0	0	0.000149	-0.00102	-0.0011	5.49E-05	-2.09E-05	0	0	-0.00012	-0.00051	0	-0.02283
16	0	-0.00362	-0.00554	0	0	0	-0.00032	-0.00205	-0.00021	0	-0.00034	0	-0.00021	-0.00017	0	0	0	0	0	-0.00439	0	-0.01647

Figure N: TF-IDF in CSV format

In the CSV file, each row represents each goal in SDG. Here we can visualize properly the relevance of a particular word in the entire SDF corpus.

Information Extraction

Abstract, Introduction, and Research Methodology

After creating the TF-IDF for the entire SDG corpus, the system is now ready to accept new documents from users. Classifying papers based on everything it contains isn't ideal, as it will greatly affect the computational performance of the classifier. The solution for this is to extract sections of the paper that contains information that greatly describes the paper. Abstract alone isn't enough as there are papers that have fewer words in the abstract. To augment this, the Introduction and Research Methodology are also extracted.

```
def main_logic(self, filename):
    appendedData = ""
    abstract = self.getFromPDFAbstract(filename)
    introduction = self.getFromPDFIntro(filename)
    method = self.getFromPDFMethod(filename)
    appendedData = abstract + introduction + method
    return {'abstract': abstract, 'introduction': introduction, 'method': method, 'appendedData': appendedData}
```

Figure N: Extracting Data Set for Classifying New Document

It is also important to remember that not all papers in the institution contain all of the sections, some papers only have an Abstract, and an Introduction but no Research Methodology. As long as both the Abstract and the Introduction are present, the classifier will still predict accurately the SDG labels.

```
def getFromPDFAbstract(self, filename):
    count = 1
    finalText,final_abstract = " ", " "
    limitPages,currentPage = 10,0
    with pdfplumber.open('assets/upload/' + filename) as pdf:
        for page in pdf.pages:
            extractFromPDF = page.extract_text()
            finalText = finalText + extractFromPDF
            checkAbs = self.getAbstract(finalText, count)
            if (checkAbs):
                final_abstract = finalText
                final_abstract = self.cleanString(final_abstract)
                break
            if (currentPage == limitPages):
                break
            count += 1
            currentPage += 1
            final_abstract = " "
            finalText = " "
    return final_abstract
```

Figure N: Handler for Extracting Abstract

To improve the computational time in extracting the Abstract, the search is limited to only 10 pages as most Abstract in the institution's format can be found around these sections.

```python
def getAbstract(self, processedText, page):
    count = 0
    pageAbstract = 0
    abstract = False
    if (("ABSTRACT" in processedText or "Abstract" in processedText)
            and ("TABLE OF CONTENTS" not in processedText and "Table of Contents" not in processedText)):
        if (count == 0):
            abstract = True
            pageAbstract = page
        count += 1
    return abstract
```

Figure N: Logic for Extracting Abstract

Extracting the Abstract is a straightforward method since papers contain a table of contents that has the word Abstract, the function will check if the word Abstract is in the string, and if found, it will check if the word table of contents is not in the string. If both conditions (Abstract in String and Table of Contents not in String) are True, then that specific string is an Abstract.


```
{
  "abstract": "   ABSTRACT    Counseling is one of the important services that a school
should have. It helps students  whenever they have problems and also provides services
that can also help the mental  health of students. However, during the pandemic, face-to-
face counseling was somehow  prohibited. Because of this, a system was designed to help
counselors, teachers, and  students connect with each other. Follow App is a web
application that will help teachers  manually refer students virtually, counselors manage
referrals, and automatically refer  students based on data of EDP through the
institution's School Information Service and  Learning Management System (LMS). The system
will schedule students for a counseling  session and will provide a platform for video
meetings. As a result of this endeavor, the  researchers ensured that the system was ready
for deployment; however, the system can  still  be  improved  further.
```


Figure N: Extracted Abstract using Insomnia

Some abstracts may contain noisy texts and may need to be cleaned but as long as the format is in-lined with the institution's standard there won't be any problem extracting.

```
def getFromPDFIntro(self, filename):
    count = 1
    finalText,final_intro = " "," "
    limitPages,currentPage = 10,0
    with pdfplumber.open('assets/upload/' + filename) as pdf:
        for page in pdf.pages:
            extractFromPDF = page.extract_text()
            finalText = finalText + extractFromPDF
            checkAbs = self.getIntroduction(finalText)
            if (checkAbs):
                final_intro = finalText
                final_intro = self.cleanString(final_intro)
                break
            if (currentPage == limitPages):
                break
            count += 1
            final_intro = " "
            currentPage += 1
            finalText = " "
    return final_intro
```

Figure N: Handler for Extracting Introduction

The process of extracting the Introduction is similar to the Abstract. Limit the pages to scrape to 10.

```
def getIntroduction(self, processedText):
    count = 0
    introduction = False
    if (("INTRODUCTION" in processedText or "Introduction" in processedText)
        and ("TABLE OF CONTENTS" not in processedText and "Table of Contents" not in processedText)):
        if (count == 0):
            introduction = True
        count += 1
    return introduction
```

Figure N: Logic for Extracting Introduction

The logic for extracting the introduction is also similar to that of the abstract extraction.

```
  "introduction": "   CHAPTER I      INTRODUCTION    Rationale of the Study   The Guidance
Office plays a vital role in every school. The guidance  office  comprises guidance
counselors in charge of counseling students whenever the latter  encounter personal and
academic-related problems and conflicts. The function of school  counselors in ensuring
student achievement is fundamental. (Lapan, Gysbers, & Kayson,  2007; Stone & Dahir,
2006).  Most counseling in schools is done face-to-face through teachers' referrals or
individual appointments in which the students need to go to the office to have the
counseling session. At the University of San Jose-Recoletos (USJ-R), students are given
the appointment slip and are called to the guidance office for their session, and then
feedback is given to each teacher. But during the pandemic, the system of counseling  has
changed. Due to the fact that there were restrictions for physical encounters, the
implementation of online classes has been introduced. In lieu of this, counseling session
has been shifted to online as well to meet the student's need even in the online set-up.
Follow App is a web application for counseling that will help teachers, counselors,  and
students have the session in an online set-up. The application focuses on scheduling
students for the session, allowing counselors to give feedback online and refer students
based on academic performance and behavior.  10   ",
```


Figure N: Extracted Introduction using Insomnia

This is the structure of the introduction after extracting, it still contains noisy texts but will be removed later after text preprocessing.

```python
def getFromPDFMethod(self, filename):
    count = 1
    finalText = " "
    final_method = " "
    limitPages = 10
    currentPage = 0
    with pdfplumber.open('assets/upload/' + filename) as pdf:
        for page in pdf.pages:
            extractFromPDF = page.extract_text()
            finalText = finalText + extractFromPDF
            checkAbs = self.getMethodology(finalText)
            if (checkAbs):
                final_method = finalText
                final_method = self.cleanString(final_method)
                break
            count += 1
            if (currentPage == limitPages):
                break
            currentPage += 1
            final_method = " "
            finalText = " "
    return final_method
```

Figure N: Handler for Methodology Extraction

The handler is the same as the previous two extractions.

```
def getMethodology(self, processedText):
    count = 0
    methodology = False
    if (("Research Methodology" in processedText or "RESEARCH METHODOLOGY" in processedText)
            and ("TABLE OF CONTENTS" not in processedText and "Table of Contents" not in processedText)):
        if (count == 0):
            methodology = True
        count += 1
    return methodology
```

Figure N: Logic for Methodology Extraction

The process for extracting the methodology is similar to those previous two extractions.

```
    "method": " "
```

There are instances where there will be no methodology extracted and the data appended data will only be composed of the Abstract and the Introduction. The methodology in this case is an optional requirement.

Extraction of these data is done once the user uploads their approved research paper, but before the user can view the extracted information, the system will first check if the paper uploaded is an authentic approved research paper.

```
def acceptanceChecker(self, filename):
    go = False
    endorsement = " "
    if (self.checkPages(filename) >= 5):
        endorsement = self.endorsementExtraction(filename)
        if ("PASSED" in endorsement):
            go = True
        else:
            os.remove("assets/upload/" + filename)
    else:
        os.remove("assets/upload/" + filename)
    return go
```

Figure N: Logic for Checking Acceptance

The system will perform checking of the paper and it will scrape the file until it finds the endorsement page, if it returns true, the paper uploaded is approved and if it's false, the paper is rejected, and removed from the local directory.

Published Date, Title, and Department

The system incorporates a feature that automatically extracts research information from the user once a file is uploaded to the system.

```
def extract_text_from_pdf(self):
    # Get the directory containing the script
    script_dir = os.path.dirname(os.path.abspath(__file__))
    # Get the parent directory of the script directory
    parent_dir = os.path.dirname(script_dir)
    # Construct the full path to the PDF file
    pdf_path = os.path.join(parent_dir, "assets",
                            "upload", self.document_path)
    # Open PDF file in binary mode
    with open(pdf_path, 'rb') as pdf_file:
        # Create a PDFPlumber object
        pdf_reader = pdfplumber.open(pdf_file)
        # Extract text from the first page of the PDF
        page = pdf_reader.pages[0]
        pdf_text = page.extract_text()
        # Convert PDF page to image
        x0, y0, x1, y1 = page.cropbox or (0, 0, page.width, page.height)
        image = page.to_image(resolution=300)
        image_file = 'temp_image.png'
        image.save(image_file, format='png')
        # Specify the path to Tesseract executable
        tesseract_path = os.path.join(os.getenv('PROGRAMFILES'), 'Tesseract-OCR', 'tesseract.exe')
        # Perform OCR using pytesseract
        ocr_text = pytesseract.image_to_string(image_file)
        # Extract paragraphs from extracted text
        paragraphs = self.extract_paragraphs_from_text(ocr_text)
        os.remove(image_file)

        return paragraphs
```

Figure N: Converting Cover Page to Image File

The first step in this process is to convert the cover page into an image format. The goal here is to provide better extraction results and more code readability. After converting the cover page into an image, the system calls the pytesseract.image_to_string to perform OCR. The role of the OCR here is just simply to extract all strings. This removes the unnecessary spacing provided by the default extractor, pdfplumber. Although the system is still using the default extractor for other extraction functionalities.

```python
def process_extracted_text(self, input_text, fromNode):
    information = {}
    information['title'] = self.extract_title(input_text)
    information['department'] = self.extract_department(input_text)
    information['authors'] = self.extract_names(input_text, fromNode)
    information['published_date'] = self.extract_published_date(input_text)
    return information
```

Figure N: Handler for Information Extraction

```python
def extract_title(self, input_text):
    title = ''
    if len(input_text) > 0:
        # Extract the first item as the title
        title = input_text[0].strip()
    return title
```

Figure N: Title Extraction

As per the pattern in the format, research titles are usually found at the beginning of the cover page.

```python
def extract_department(self, input_text):
    departments = [
        'School of Law', 'School of Business and Management',
        'School of Computer Studies', 'Senior High School',
        'School of Arts and Sciences', 'RITTC',
        'School of Allied Medical Sciences',
        'School of Engineering', 'School of Education', 'College of Information Computer and Communications Technology'
    ]
    extracted_department = ''
    for text in input_text:
        for department in departments:
            if department in text:
                extracted_department = department
                break

        if extracted_department:
            break
```

Figure N: Department Extraction

Extracting the department is also a pretty straightforward approach and mostly involves rule-based identification. It contains the names of all the departments in the university.

```
def extract_published_date(self, input_text):
    extracted_date = None
    current_date = datetime.datetime.now()  # Get current date and time
    for text in input_text:
        extracted_date_str = re.findall(r'\b\d{1,2}/\d{4}\b', text)
        if extracted_date_str:
            extracted_date = date_parser.parse(
                extracted_date_str[0], fuzzy=True)
        if not extracted_date:
            date_formats = [
                '%B %d, %Y',
                '%B %Y',              # Month Year (e.g. March 2020)
                '%m/%Y',              # Month/Year (e.g. 03/2020)
                '%b %Y',
                # Month Year without slash (e.g. 03 2020)
                '%m %Y',
                'date %Y',            # Custom date format (e.g. date 2023)
                # Custom date format (e.g. June 20, 2022)
                '%B %d, %Y',
                # Custom date format with time (e.g.June 20, 2022 12:34)
                '%B %d, %Y %H:%M',
            ]
            for line in text.split('\n'):
                try:
                    # Attempt to parse date from line using date_formats
                    for date_format in date_formats:
                        parsed_date = date_parser.parse(line, fuzzy=True, yearfirst=True,
                                                        default=current_date)
                        if parsed_date:
                            extracted_date = parsed_date
                            break
                    if extracted_date:
                        break
                except ValueError:
```

Figure N: Date Extraction

The function will check if the string given is of a date-time format. The function also utilizes rule-based classification to check if the token is a date or not.

SDG Classification

Applying TF-IDF on the New Document

The KNN algorithm involves the creation of training/test data, identifying which distance/similarity metric to use, and majority voting by its neighbors or the K. One of the common distance metrics used in KNN is the Euclidean, Manhattan, Minkowski, and Hamming distance. The study focuses on the use of cosine similarity as its main similarity metric since this technique focuses more on the semantic meaning between two documents rather than their geometric proximity.

```python
def classifyResearch(self, data):
    count = 0
    trainingDocs,newDocs = [],[]
    goals = ['Goal 1', 'Goal 2', 'Goal 3', 'Goal 4', 'Goal 5',
             'Goal 6', 'Goal 7', 'Goal 8', 'Goal 9', 'Goal 10', 'Goal 11', 'Goal 12',
             'Goal 13',
             'Goal 14', 'Goal 15', 'Goal 16', 'Goal 17'
            ]
    if (self.checkDataSet() == False):
        for goal in goals:
            trainingData = self.extractAllPDF(goal)
            trainingDocs.append(trainingData)
    else:
        trainingDocs = self.extractTraining()
        newDocs.append(data)
        newData = self.preprocess_documents(newDocs)
        data = newData[0]
    trainingDocs.append(data)
    values = self.getTFIDF(trainingDocs)
    count += 1
    if (self.checkLastData()):
        self.removeNewData()
    return self.getCosine(values, count)
```

Figure N: Handler for Classifier

To avoid retraining the model every time there's a new document to classify, the system provides some rules to handle that particular scenario.

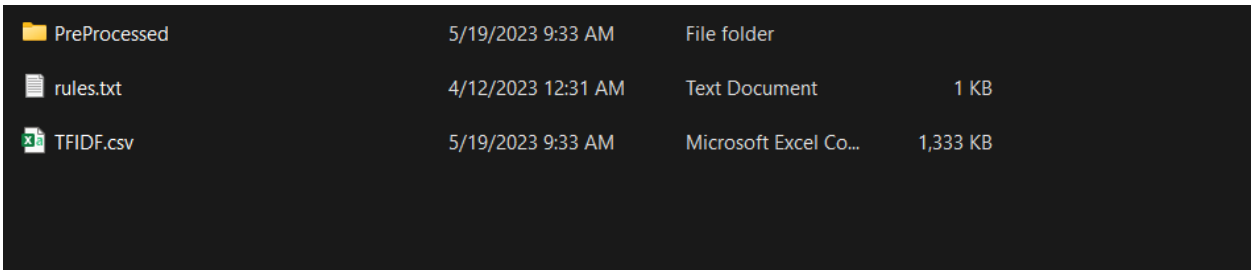


Figure N: Directory of TFIDF.csv

First, the system will check if the TF-IDF.csv file exists in the directory. If it returns false, that's the time the retraining will happen, and if not, the system will retrieve all the preprocessed data into a list. These preprocessed data are stored beforehand during the training of the model.

```
def extractTraining(self):
    index = 17
    string = ""
    extractedTraining = []
    for i in range(index):
        with open(r"tfidf/Results/PreProcessed/PreProcessed " + str(i+1) + ".txt", 'r', encoding="utf8") as f:
            for line in f:
                string = line.split()
                extractedTraining.append(string)
                string = ""
    return extractedTraining
```

Figure N: Retrieving Preprocessed Data

These data are stored separately based on their goals, in total, there are 17 text files in the directory that contains all of these preprocessed data. This step plays a crucial role in decreasing the computational time of the classification as it skips the process of extracting raw data from all 17 goals and applying text processing.

```
else:
    trainingDocs = self.extractTraining()
    newDocs.append(data)
    newData = self.preprocess_documents(newDocs)
    data = newData[0]
```

Figure N: New Document PreProcessing

The new document will be the only one that undergoes preprocessing. After preprocessing the new document. The newly uploaded document will be stored in the temporary 18th row of the SDG corpus.

```
def getTFIDF(self, documents):
    tv, tf, final = [{}], [{}], [{}]
    index = 1
    if (self.checkDataSet() == False):
        preProcessedDocs = self.preprocess_documents(documents)
        unique = self.getUniqueWords(preProcessedDocs, False)

        for token in preProcessedDocs:
            self.writeListToTxt(' '.join(token), index)
            index += 1
        self.addChecker()
    else:
        preProcessedDocs = documents
        unique = self.getUniqueWords(preProcessedDocs, True)

    for listOfTokens in preProcessedDocs:
        tf.append(self.getTermFreq(
            unique, len(listOfTokens), listOfTokens))
        tv.append(self.getTerm(unique, len(listOfTokens), listOfTokens))
    tf.pop(0)
    tv.pop(0)
    idf = self.inverse(unique, preProcessedDocs, tv)
    final = self.calculateTFIDF(tf, idf, final)
    final.pop(0)
    values = []
    for doc in final:
        values.append(doc.values())
    tf_idf = self.convertingToDP(final)
    return values
```

Figure N: Apply TF-IDF to the Newly Uploaded Document

The process of applying TF-IDF is a similar process to that when training the model, the only difference is there's a new document appended.

RAW ABSTRACT Plants are categorized into smaller groups according to their shared characteristics, which can be daunting given their complexity. While experts can quickly recognize f amiliar plants, identifying potentially harmful or toxic ones, particularly in medicine, can be challenging. Botanists possess the expertise to distinguish such plants, but with millions of species featuring similar parts (roots, stems, leaves), they must devise a system to classify them effectively. Living Green aims to expound botanical research through a Mobile Botanical Identifier mobile ap plication for finding an unknown plant's captured or uploaded photo with a barter system. It is a mobile application that connects users with plant enthusiasts and plant experts to aid in identifying a plant name. 3 CHAPTER I INTRODUCTION Rationale of the Study Our world is primarily made up of vegetation, so it is called a "green planet." In addition to the relationship between pla ental methods, and another method called the optical method-more advantageous than other techniques-were all used in identifying leaves. 6

PreProcessed ['abstract', 'plant', 'categorized', 'smaller', 'group', 'according', 'shared', 'characteristic', 'daunting', 'given', 'complexity', 'while', 'expert', 'quickly', 'recognize', 'familiar', 'plant', 'identifying', 'potentially', 'harmful', 'toxic', 'one', 'particularly', 'medicine', 'challenging', 'botanist', 'posse', 'expertise', 'distinguish', 'plant', 'million', 'specie', 'featuring', 'similar', 'part', 'root', 'stem', 'leaf', 'must', 'devise', 'system', 'classify', 'effectively', 'living', 'green', 'aim', 'expound', 'botanical', 'research', 'mobile', 'botanical', 'identifier', 'mob ile', 'application', 'finding', 'unknown', 'plant', 'captured', 'uploaded', 'photo', 'barter', 'system', 'it', 'mobile', 'application', 'connects', 'user', 'plant', 'enthusiast', 'plant', 'expert', 'aid', 'identifying', 'plant', 'name', '3', 'chapter', 'i', 'introduction', 'rationale', 'study', 'our', 'world', 'primarily', 'made', 'vegetation', 'called', 'green', 'planet', 'in', 'addition', 'relatio nship', 'plant', 'food', 'medicine', 'furniture', 'essential', 'understand', 'conceive', 'world', 'without', 'oxygen', 'plant', 'supply', 'because', 'argue', 'plant', 'foundation', 'life', 'plant', 'cl assified', 'smaller', 'group', 'based', 'characteristic', 'share', 'recognition', 'plant', 'might', 'challenging', 'since', 'complex', 'the', 'process', 'recognizing', 'familiar', 'plant', 'simple', 'e xpert', 'sometimes', 'particularly', 'medicine', 'need', 'identify', 'prejudiced', 'toxic', 'plant', 'botanist', 'quickly', 'identify', 'plant', 'must', 'find', 'way', 'classify', 'many', 'different', 'specie', 'since', 'million', 'plant', 'specie', 'composed', 'similar', 'part', 'root', 'stem', 'leaf', 'designing', 'plant', 'recognition', 'system', 'necessary', 'save', 'time', 'money', 'numerous', 'study', 'focused', 'leaf', 'identify', 'plant', 'comparison', 'plant', 'part', 'leaf', 'crucial', 'conveying', 'plant', 'characteristic', 'fruit', 'flower', 'also', 'always', 'present', 'plant', 'seas onal', 'size', 'shape', 'color', 'change', 'develop', 'numerous', 'study', 'employed', 'leaf', 'categorize', 'different', 'plant', 'type', 'based', 'shape', 'texture', 'venation', 'color', 'chemical', 'approach', 'instrumental', 'method', 'another', 'method', 'called', 'optical', 'advantageous', 'used', 'identifying', 'leaf', '6']

Figure N: The Raw Data and Preprocessed Data of the New Document

This data represent the new document to be classified. The abstract and the Introduction are extracted and preprocessed.

overview	0
a	0
world	2
free	0
from	0
...	...
flower	1
color	2
categorize	1
venation	1
optical	1

Figure N: Term Vectors of New Document

new	Value
overview	0.000000
a	0.000000
world	0.009174
free	0.000000
from	0.000000
...	...
flower	0.004587
color	0.009174
categorize	0.004587
venation	0.004587
optical	0.004587

Figure N: Term Frequency of New Document

	Value
overview	0.556303
a	-1.054358
world	-1.139179
free	0.109144
from	-0.023481
...	...
flower	1.255273
color	0.954243
categorize	1.255273
venation	1.255273
optical	1.255273

Figure N: Inverse Frequency of New Document

[10393 rows x 1 columns]

	Value
overview	0.000000
a	-0.000000
world	-0.010451
free	0.000000
from	-0.000000
...	...
flower	0.005758
color	0.008755
categorize	0.005758
venation	0.005758
optical	0.005758

Figure N: TF-IDF of New Document

The process of applying TF-IDF on the newly uploaded document. Words below the three dots are all high values, this indicates that these words in the new document are unique with respect to the SDG corpus

After getting the TF-IDF of the new document and appending it to the TF-IDF of the SDG corpus, the next step is the cosine similarity scoring.

Cosine Similarity Scoring

$$similarity(A,B) = \frac{A \cdot B}{\|A\| \times \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}$$

Figure N: Cosine Similarity Formula

Mathematically, cosine similarity measures the cosine of the angle between two vectors in an inner product space or multi-dimensional space. In the context of this study, the vectors are arrays of TFIDF values of each document.

New Document:	
overview	0.000000
a	-0.000000
world	-0.010451
free	0.000000
from	-0.000000
...	...
flower	0.005758
color	0.008755
categorize	0.005758
venation	0.005758
optical	0.005758

Figure N: TFIDF values of New Document

Goal 1 TF-IDF:	
overview	0.000277
a	-0.001836
world	-0.007086
free	0.000081
from	-0.000023
...	...
flower	0.000000
color	0.000000
categorize	0.000000
venation	0.000000
optical	0.000000

Figure N: TFIDF values of Goal 1: No Poverty

Since we already have our arrays of TF-IDF values. The first step in getting the cosine similarity score is to apply the formula where A represents the New Document, and B the document in the TF-IDF. We first need to calculate the dot product of New Document (A), with the first document in TF-IDF (B). The dot product measures the alignment of two vectors. If the result is positive, this means that the two vectors are pointing in a similar direction, and the opposite is negative. A zero value means that the two vectors are perpendicular to each other or in document classification's case, no similarity at all.

Dot Product of New Document and First Document in TF-IDF:

*Dot Product = (0.000000 x 0.000277) + (- 0.00000 * -0.001836) + (-0.010451 * -0.007086) until the last value of both the Arrays.*

The next step is getting the product of the magnitudes of both vectors compared. This is to normalize the value of the dot product so we can have values from -1 and 1, representing the similarity/dissimilarity between the vectors.

||A|| x ||B||, where:

New Document (A) = [0.00000,-0.00000,-0.010451]

First Document in TFIDF (B) = [0.000277,-0.001836,-0.007086]

A = sqrt(0.00000^2 + -0.00000^2 + -0.010451^2)

B= sqrt (0.000277^2 + -0.001836^2 + -0.007086^2)

*Magnitude = A * B*

So, the final result is Dot Product / Magnitude = Cosine Score

```
Dot Product:  0.0005002948157214309 Magnitude:  0.0031891493840120098
Dot Product:  0.0006354694623225191 Magnitude:  0.004224129279961806
Dot Product:  0.0006712936983007098 Magnitude:  0.005097953019234838
Dot Product:  0.0004649926236958813 5 Magnitude:  0.005640819885505017
Dot Product:  0.0005225613445220877 Magnitude:  0.006258342413813219
Dot Product:  0.0005404088936799382 Magnitude:  0.004154221245468342
Dot Product:  0.0004621314348072816 Magnitude:  0.003297559919403429 5
Dot Product:  0.0003566790319039243 Magnitude:  0.00330276782326685
Dot Product:  0.0003773478244656774 7 Magnitude:  0.003663257124483432
Dot Product:  0.0005613008904062513 Magnitude:  0.004501854206457432
Dot Product:  0.0005504168288468866 Magnitude:  0.004048207131901482
Dot Product:  0.000580391169196893 Magnitude:  0.003524819031213985
Dot Product:  0.0010632756729797711 Magnitude:  0.003515086136476789 7
Dot Product:  0.0004538545572882027 Magnitude:  0.003127476495624967 4
Dot Product:  0.0003941483144570214 Magnitude:  0.004478422214645667
```

Figure N: Dot Product and Magnitude of New Document with respect to each Document

```
New Document x  Goal 1: No Poverty :  0.08088
New Document x  Goal 2: Zero Hunger :  0.18037
New Document x  Goal 3: Good Health and Well-Being :  0.15687
New Document x  Goal 4: Quality Education :  0.15044
New Document x  Goal 5: Gender Equality :  0.13168
New Document x  Goal 6: Clean Water and Sanitation :  0.08243
New Document x  Goal 7: Affordable and Clean Energy :  0.0835
New Document x  Goal 8: Decent Work and Economic Growth :  0.13009
New Document x  Goal 9: Industry, Innovation, and Infrastrucuture :  0.14014
New Document x  Goal 10: Reduced Inequalities :  0.10799
New Document x  Goal 11: Sustainable Cities and Communities :  0.10301
New Document x  Goal 12: Responsible Consumption and Production :  0.12468
New Document x  Goal 13: Climate Action :  0.13597
New Document x  Goal 14: Life Below Water :  0.16466
New Document x  Goal 15: Life on Land :  0.30249
New Document x  Goal 16: Peace, Justice and Strong Institutions :  0.14512
New Document x  Goal 17: Partnership for the Goals :  0.08801
```

Figure N: Final Result of Cosine Similarity Scoring

After applying cosine scoring on each of the documents, all the results are normalized and output values closer to 1, this means that all documents have a sense of similarity to the new document. This is a predicted outcome of the values since there are words in the new document that exists in all of the documents, the question is how relevant that word is to the entire SDG corpus, and that was the purpose of getting the TF-IDF. We can see that Goal 15: Life on Land outputs the highest score, 0.30249 or 30%. This is an accurate result since the new document we tested actually talks about plants and other specie of it.

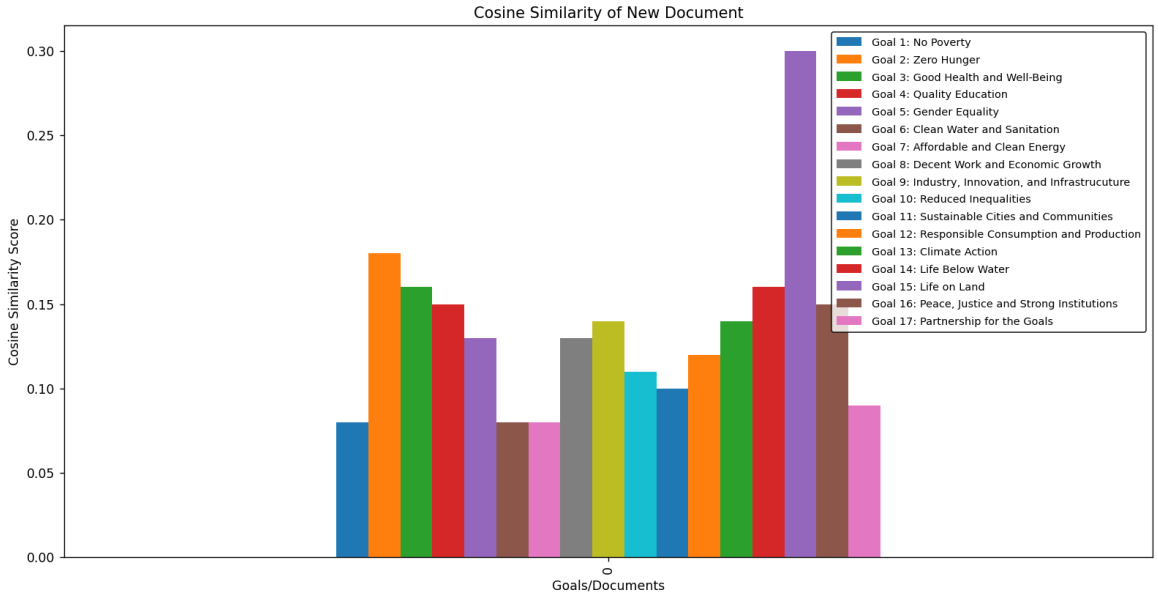


Figure N: Visualization of the Final Cosine Scores Result

```
def getCosine(self, oldDoc, count):
    newVector, cosine = oldDoc[len(oldDoc)-1], []
    counter = 0
    classifier = {}
    del oldDoc[-1]
    goals = ["Goal 1: No Poverty", "Goal 2: Zero Hunger", "Goal 3: Good Health and Well-Being", "Goa
             "Goal 9: Industry, Innovation, and Infrastrucuture", "Goal 10: Reduced Inequalities",
             ]
    for val in oldDoc:
        val2 = val
        vector1, vector2 = [], []
        dotProduct, magnitude, magnitude1, magnitude2 = 0, 0, 0, 0
        for newvec in newVector:
            vector1.append(newvec)
        for oldVar in val2:
            vector2.append(oldVar)
        dotProduct = np.dot(vector1, vector2)
        magnitude1 = math.sqrt(
            sum(component ** 2 for component in newVector))
        magnitude2 = math.sqrt(sum(component ** 2 for component in val2))
        magnitude = magnitude1 * magnitude2
        cosine.append(round(dotProduct/magnitude, 5))
        print("New Document x ",
              goals[counter], ": ", cosine[counter])
        percent = round(
            (dotProduct / magnitude) * 100, 2)
        classifier[goals[counter]] = percent
        counter += 1
    sorted_dict = dict(
        sorted(classifier.items(), key=lambda item: item[1], reverse=True))
    return sorted_dict
```

Figure N: Code Snippet for Getting the Cosine Scores

Here we can see how getting the cosine scores is implemented, first, we have a variable oldDoc that is a list of values we got from our TF-IDF, under the hood, it contains 17 items. The variable newVector holds the TF-IDF values of the new document. Once we all have 2 variables, we then calculate the dot product of newVector to the first index of the oldDoc, once we get the dot product we then proceed to get the magnitude. The variable magnitude1 stores the value of the newVector and the magnitude2 variable stores the value of the first element of oldDoc. After that, we simply multiply both and store it to variable magnitude. Finally, we then divide dotProduct and magnitude to get the cosine score of New Document and Goal 1. This process is repeated until the loop iterates over the entire oldDoc list.

At the end of the loop, the values are then stored in the dictionary which has the goal name as the key, and the cosine score as the value.

The last step in classification is the KNN classifier. This function requires the cosine dictionary and the hyperparameter K. The value of K depends on the number of documents we have. The rule of thumb is to take the square root of the total number of documents in the corpus. In our case, we have 17 so the sqrt(17) = 4.1231, but we adjusted it to 5 to give more optimal performance of the classifier.

```python
def knn_classifier(self, cosine_similarity, k):
    cosine_similarityList = list(cosine_similarity.values())
    goals = list(cosine_similarity.keys())
    nearest_labels = goals[:k]
    nearest_scores = cosine_similarityList[:k]
    weighted_votes = Counter()
    for i, label in enumerate(nearest_labels):
        weighted_votes[label] += nearest_scores[i]
    predicted_label = weighted_votes.most_common(4)
    resultDictionary = dict((x, y) for x, y in predicted_label)
    return resultDictionary
```

Figure N: K-nearest neighbor classifier

The KNN performs the voting by the neighbors (K) to give the predicted labels. Here we can see that list is sliced so that only the k values are included in the voting. This is the function's way of determining which neighbor has the right to vote for the predicted label.

```
Vote:  30.25
Vote:  18.04
Vote:  16.47
Vote:  15.69
Vote:  15.04
Final Votes Result:  [('Goal 15: Life on Land', 30.25), ('Goal 2: Zero Hunger', 18.04), ('Goal 14: Life Below Water', 16.47), ('Goal 3: Good Health and Well
-Being', 15.69)]
```

Since we need to have the top four goals, we set the most common to output the top four values in the voting. The voting scheme used is majority voting since this is the most applicable and easy-to-implement voting scheme. So the final result would look like this,

```json
{
  "Goal 14: Life Below Water": 16.47,
  "Goal 15: Life on Land": 30.25,
  "Goal 2: Zero Hunger": 18.04,
  "Goal 3: Good Health and Well-Being": 15.69
}
```

Figure N: Final Classification Result

TESTING PROCESS

ACCURACY TESTING

CHAPTER IV

SUMMARY, CONCLUSION, AND RECOMMENDATIONS SUMMARY OF FINDINGS

CONCLUSION

The use of TF-IDF, Cosine Similarity Scoring, and the KNN Algorithm proves to perform well in supervised document classification. The SDG classifier gives accurate results when there's a variety of data in each goal. The TF-IDF of the SDG corpus demonstrated a key role in the classification process. Using the cosine similarity scores as input to the KNN algorithm established good results in classifying research papers according to UN SDGs.

Utilizing a rule-based approach in handling extracted text to categorize it into different labels shows justifiable results.

RECOMMENDATIONS

To alleviate the problem when extracting information the usage of OCR is recommended.

To improve the computational performance of the classifier, various optimization techniques may be used just like multi-threading. Adding more PDF files to each goal in the training data will greatly improve the accuracy of the classifier.

To make the SDG classifier more flexible in processing huge chunks of data, other classification algorithms like SVM or Random Forest may be included.

BIBLIOGRAPHY

A. Online Website Resources

[1] History of artificial intelligence - javatpoint. www.javatpoint.com. (n.d.). Retrieved July 25, 2022, from https://www.javatpoint.com/history-of-artificial-intelligence

[2] Editor. (2021, November 17). *Document classification with machine learning: Computer vision, OCR, NLP, and other techniques*. AltexSoft. Retrieved August 4, 2022, from https://www.altexsoft.com/blog/document-classification/

[3] Burns, E. (2021, March 30). *What is machine learning and why is it important?* SearchEnterpriseAI. Retrieved July 25, 2022, from https://www.techtarget.com/searchenterpriseai/definition/machine-learning-ML

[4] Petersson, D. (2021, March 26). *What is supervised learning?* SearchEnterpriseAI. Retrieved August 4, 2022, from https://www.techtarget.com/searchenterpriseai/definition/supervised-learning

[5] *Classification algorithm in Machine Learning - Javatpoint*. (n.d.). Retrieved August 4, 2022, from https://www.javatpoint.com/classification-algorithm-in-machine-learning

[6] Pratt, M. K. (2020, July 8). *What is unsupervised learning?* SearchEnterpriseAI. Retrieved July 28, 2022, from https://www.techtarget.com/searchenterpriseai/definition/unsupervised-learning

[7] Webb, G. I. (1970, January 1). *Naïve bayes*. SpringerLink. Retrieved July 27, 2022, from https://link.springer.com/10.1007%2F978-0-387-30164-8_576

[8] (LEDU), E. E. (2018, September 12). *Understanding K-means clustering in machine learning*. Medium. Retrieved August 4, 2022, from https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning-6a6e67336aa1

[9] Chen, S. (2020, May 26). *Getting started with text vectorization*. Medium. Retrieved August 1, 2022, from https://towardsdatascience.com/getting-started-with-text-vectorization-2f2efbec6685#:~:text=Text%20Vectorization%20is%20the%20process,(L1)%20Normalized%20Term%20Frequency

[10] *Understanding TF-ID: A simple introduction*. MonkeyLearn Blog. (2019, May 10). Retrieved August 3, 2022, from https://monkeylearn.com/blog/what-is-tf-idf/#:~:text=TF%2DIDF%20

[11] By: IBM Cloud Education. (n.d.). *What is natural language processing?* IBM. Retrieved August 4, 2022, from https://www.ibm.com/cloud/learn/natural-language-processing

[12] *What is optical character recognition (OCR)?* IBM. (n.d.). Retrieved July 28, 2022, from https://www.ibm.com/cloud/blog/optical-character-recognition

[13] Kim, S.-W., & Gil, J.-M. (2019, August 26). *Research paper classification systems based on TF-IDF and LDA Schemes - human-centric computing and Information Sciences*. SpringerOpen. Retrieved August 4, 2022, from https://hcis-journal.springeropen.com/articles/10.1186/s13673-019-0192-7

[14] Mohsen Taheriyan University of Southern California, Taheriyan, M., California, U. of S., Nebraska, U. of, Army, U. S., Massachusetts, U. of, Oklahoma, U. of, Saic, Maryland, U. of, Nasa, & Metrics, O. M. V. A. (2011, August 1). *Subject classification of research papers based on Interrelationships Analysis: Proceedings of the 2011 Workshop on Knowledge Discovery, modeling and Simulation*. ACM Conferences. Retrieved July 30, 2022, from https://dl.acm.org/doi/10.1145/2023568.2023579

[15] Maheshwari, A. (2018, July 17). *Report on text classification using CNN, RNN & Han*. Medium. Retrieved August 28, 2022, from https://medium.com/jatana/report-on-text-classification-using-cnn-rnn-han-f0e887214d5f#:~:text=Text%20Classification%20Using%20Convolutional%20Neural,inspired%20by%20animal%20visual%20cortex.