TEXT CLASSIFICATION USING HAN

UCS503 Software Engineering Project Report

End-Semester Evaluation

Submitted by:

(102003130) Arpit Sagar

(102003167) Rupinderpal Singh

(102003184) Madhvan Jindal

Team Name: Code Freaks

BE Third Year, COE-6

Group No: CO6SE1

Submitted to:

Dr.Anamika Sharma

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

Computer Science and Engineering Department

TIET, Patiala

December 2022

1.Project Selection Phase :

i. Software Bid

Software Bid/ Project Teams

UCS 503- Software Engineering Lab

Group : 3CO6 Dated:03/08/2022

Team Name: CODE FREAKS
Team ID (will be assigned by Instructor):

Please enter the names of your Preferred Team Members:
Arpit Sagar , Rupinderpal Singh , Madhvan Jindal

You are required to form a three to four person teams
Choose your team members wisely. You will not be allowed to change teams.

Name	Roll No	Project Experience	Programming Language used	Signature
Arpit Sagar	102003130	AI Prediction Model (Gold price prediction)	Python	
Rupinderpal Singh	102003167	AI Prediction Model(Gold price prediction)	Python	
Madhvan Jindal	102003184	AI Spam Mail Prediction Model	Python	

Programming Language / Environment Experience

List the languages you are most comfortable developing in, as a team, in your order of preference. Many of the projects involve Java or C/C++ programming.

1. C
2.C++
3.Python

Choices of Projects:

Please select 4 projects your team would like to work on, by order of preference: [Write at-least one paragraph for each choice (motivation, reason for choice, feasibility analysis, etc.)]

First Choice	Text Classification using Hierarchical Attention Networks(ML model with ui/ux) : Feasibilty analysis: It's estimated that around 80% of all information is unstructured, with text being one of the most common types of unstructured data. Because of the messy nature of text, analyzing, understanding, organizing, and sorting through text data is hard and time-consuming, so text Classification will help to structure, organize data and reduce time to analyse it. Applications include topic labeling,sentiment classification and spam detection.
	Its scalability,real-time analysis and unsurpassed accuracy using ML models leads to feasibility of this project.
Second Choice	CHATBOT(for covid 19) using python. Feasibilty analysis: People face many challenges when they visit websites or apps for covid 19 information. So we can create an automated chatbot that can help them with their queries regarding symptoms,vaccinations etc. and guide people to the particular information and solutions they are looking for.
Third Choice	Password generation/storage app website Feasibilty analysis: Large no. of Passwords and their uniqueness is a great issue of concern. It has become very important to regularly change password to maintain security and privacy. We regularly run out of secure passwords for our accounts. We even tend to forget passwords. Creating a website using react js which can suggest some easy passwords to remember but not that easy to guess around. Also further this website can store the passwords for different accounts. A structured user friendly app to store and organize passwords while also giving suggestions for strong unmatched keys.
Fourth Choice	Food ordering website (Using react JS) Feasibilty analysis: Considering the fact that different food ordering website already exists but what if we want to place order from different restaurants at the same time. So the hack is creating a website using React JS to order food from multiple restaurants in a single order unlike the existing food websites where we need to separately order for different restaurants. React Js technology helps us to build interactive websites and the implementation of food ordering and browsing web app with interactive interface is a great opportunity for learning and exploring web development as well as creative skills.

Additional Remarks/ Inputs

Please tell us about any other factors that we should take into consideration (e.g., if you really would like to work on a project for some particularly convincing reason).

We really would like to work on Machine learning project of text classification because it has the use of new and latest technology of hierarchical attention networks and its high-end usage in the era of large data and text makes this idea really effective and convincing for our project as it encourages us for research as well as for diving deep into concepts of machine learning and AI.

……

2.Planning Phase :

2.1 PROJECT WRITEUP

i. Project Overview:

Text Classification Model using Hierarchical attention Network is a machine Learning Model project to assign documents to classes or topics. It uses the hierarchical structure of documents (document - sentence - word).It can be implemented on various datasets for applications like topic labelling, tagging content, sentiment analysis etc. Using Keras library and data preprocessing, this ML model is compatible to be tested to classify any given document.
To demonstrate the attention mechanism in this project, News Classification is implemented wherein news articles are classified into categories, and short summaries of articles are made by extracting the most important sentences using sentence attention weights.

ii. Product Scope

Classification of Text using hierarchical attention networks filters the data making only relevant and important matter to be considered.An attention mechanism is used to find most important words and sentences in a document while keeping an eye on the context of words. Text classification can be used in a broad range of contexts such as classifying short texts (e.g., tweets, headlines, chatbot queries, etc.) or organizing much larger documents (e.g., customer reviews, news articles,legal contracts, longform customer surveys, etc.). Real life and industrial applications include sentiment analysis, news classifications, topic labeling ,spam/intent detections and corporate goals include classifiers like :-

- Product , workforce analytics
- Brand monitoring
- Market research
- Customer support

iii. Functional Requirements:

- Loading Dataset
- Data Preprocessing
- Deployment HAN Functional Model
    - Sentence Attention Model
    - Word Attention Model
    - Build Keras Attention Mechanism Model
- Classification Results

iv. Non Functional Requirements:

- **Availability:** The Model will provide 24/7 availability as we can upload and use any dataset for classification and analysis.

- **Scalability:** The application will be compatible in all browsers ,all versions of operating systems and configurations. The compilers do not affect the performance of the system.
- **User Friendly Interface:** The interface will be easy to use and beginner friendly. Easy to comprehend model specifications help users to add their own documents for appropriate classifications.
- **Extensibility:** Any further expansion of the model will depend on the improvement in performance by development of new algorithms.
- **Data integrity :** Datasets uploaded for model testing are verified and discrete.
- **Maintenance :** Updating Libraries ,inclusion of new parameters or removing synchronisation errors only when some new releases are there.
- **Reliability :** Software works all time in full functional mode and strategy for corrections and improvements include Tensorflow and Keras documentation review of functions.

2.2 ii. Feasibility Report :

1.Schedule Feasibility:

Estimated period of building the project: 2 Months

The development of the project contains following phases:

- Planning: 2 Weeks
- Designing: 2 Weeks
- Implementation : 2 Weeks
- Testing : 2 Weeks

2.Technical Feasibility:

- The frontend will be built using Python IDE compilers.
- The Model is compatible on all system configurations, interface and operating systems.

3.Economical Feasibility:

The making of the project is financially feasible and its maintenance cost is also very low. All the software that we require during the implementation of the project are open-source and the hosting and database services will be available at little or no prices.

4.Operational Feasibility:

The project will be fully operational and it can be operated across various platforms. The application will be developed with a view to provide the user a secure and friendly experience.

5.Legal Feasibility:

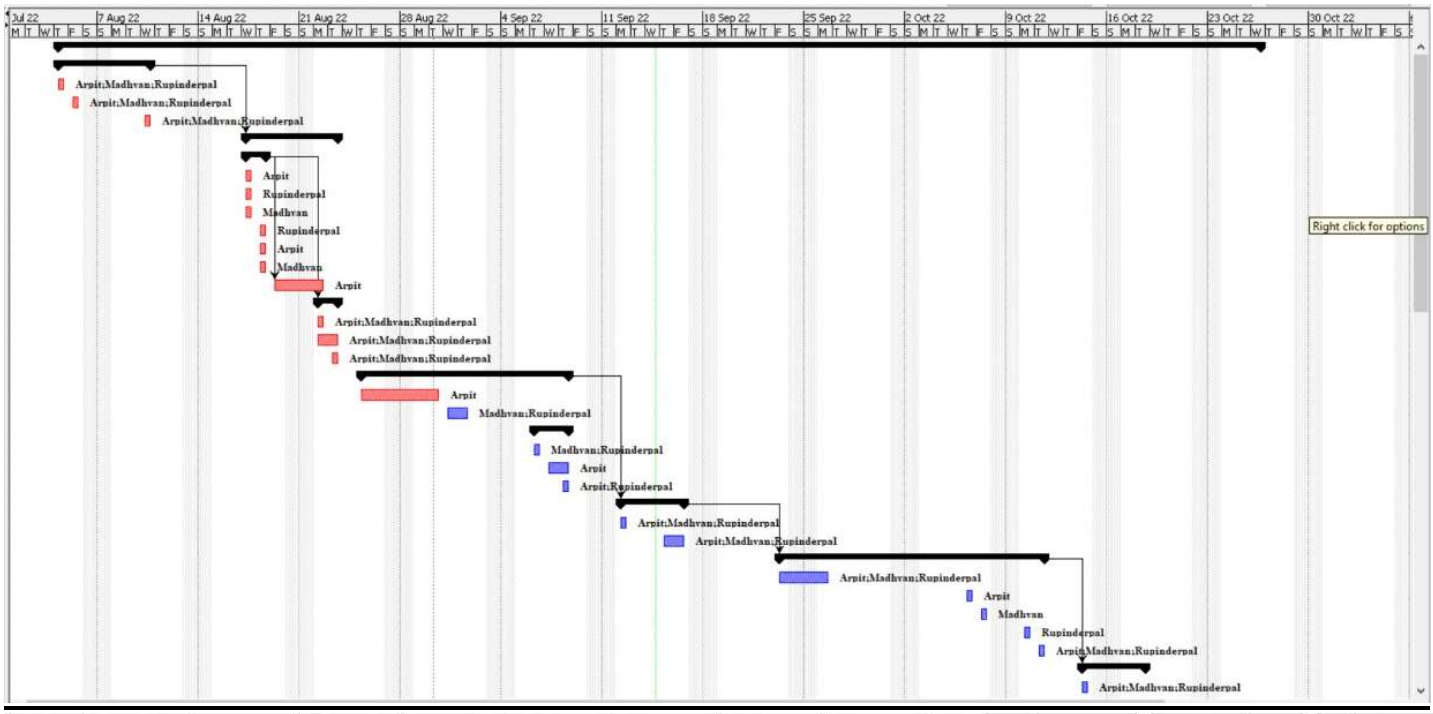
We are ensuring that we are not using any pirated stuff or any copyrighted stuff. If any resource is used while making the project, it will be cited in the documentation of the project in the references section.

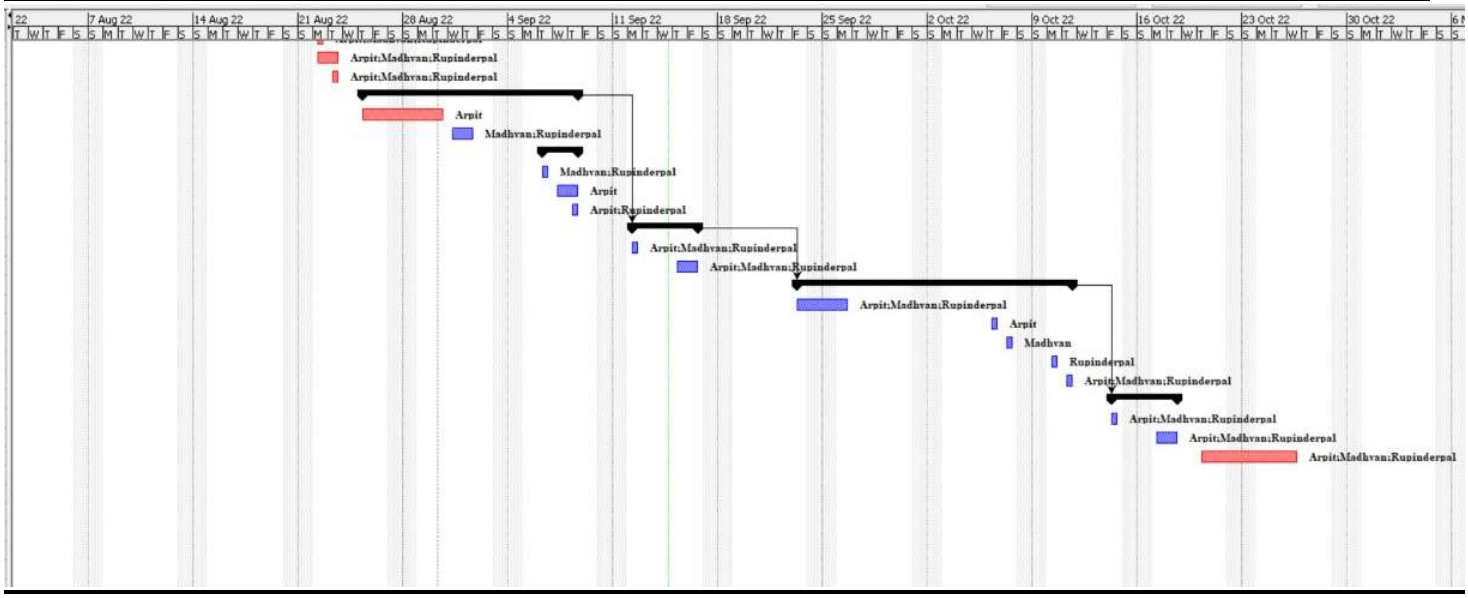
6.Cultural/Behavioural Feasibility:

The application will ensure connectivity between the people across an organization with utmost priority given to the security. The project is unbiased and will be equally accessible to all citizens within the provided territory within which the application will be accessible.

2.3 iii. Gantt Chart :

		Name	Duration	Start	Finish	Predecessors	Resource Names
1		**Text Classification**	**60 days**	**8/4/22 8:00 AM**	**10/26/22 5:00 PM**		
2		**Planning**	**5 days**	**8/4/22 8:00 AM**	**8/10/22 5:00 PM**		
3		Defining Strategy	1 day	8/4/22 8:00 AM	8/4/22 5:00 PM		Arpit;Madhvan;Rupinderpal
4		Literature Review	1 day	8/5/22 8:00 AM	8/5/22 5:00 PM		Arpit;Madhvan;Rupinderpal
5		Project Scope	1 day	8/10/22 8:00 AM	8/10/22 5:00 PM		Arpit;Madhvan;Rupinderpal
6		**Requirements**	**5 days**	**8/17/22 8:00 AM**	**8/23/22 5:00 PM**	**2**	
7		**Feasibility Study**	**2 days**	**8/17/22 8:00 AM**	**8/18/22 5:00 PM**		
8		Schedule Feasibility	1 day	8/17/22 8:00 AM	8/17/22 5:00 PM		Arpit
9		Technical Feasibility	1 day	8/17/22 8:00 AM	8/17/22 5:00 PM		Rupinderpal
10		Economic Feasibility	1 day	8/17/22 8:00 AM	8/17/22 5:00 PM		Madhvan
11		Operational Feasibility	1 day	8/18/22 8:00 AM	8/18/22 5:00 PM		Rupinderpal
12		Legal Feasibility	1 day	8/18/22 8:00 AM	8/18/22 5:00 PM		Arpit
13		Cultural/Beheviboural ...	1 day	8/18/22 8:00 AM	8/18/22 5:00 PM		Madhvan
14		Equipment Analysis	2 days	8/19/22 8:00 AM	8/22/22 5:00 PM	7	Arpit
15		**Construction Analysis**	**2 days**	**8/22/22 8:00 AM**	**8/23/22 5:00 PM**	**7**	
16		Functional Requirem...	1 day	8/22/22 8:00 AM	8/22/22 5:00 PM		Arpit;Madhvan;Rupinderpal
17		Non-Functional Requ...	2 days	8/22/22 8:00 AM	8/23/22 5:00 PM		Arpit;Madhvan;Rupinderpal
18		SRS Document	1 day	8/23/22 8:00 AM	8/23/22 5:00 PM		Arpit;Madhvan;Rupinderpal
19		**Design**	**11 days**	**8/25/22 8:00 AM**	**9/8/22 5:00 PM**		
20		Content Building	4 days	8/25/22 8:00 AM	8/30/22 5:00 PM		Arpit
21		Defining Data Hierarchy	2 days	8/31/22 8:00 AM	9/1/22 5:00 PM		Madhvan;Rupinderpal
22		**Develop Model frame**	**3 days**	**9/6/22 8:00 AM**	**9/8/22 5:00 PM**		
23		DFD	1 day	9/6/22 8:00 AM	9/6/22 5:00 PM		Madhvan;Rupinderpal
24		UML diagram	2 days	9/7/22 8:00 AM	9/8/22 5:00 PM		Arpit
25		ER diagram	1 day	9/8/22 8:00 AM	9/8/22 5:00 PM		Arpit;Rupinderpal
26		**Implementation**	**5 days**	**9/12/22 8:00 AM**	**9/16/22 5:00 PM**	**19**	
27		Deployment Phase	1 day	9/12/22 8:00 AM	9/12/22 5:00 PM		Arpit;Madhvan;Rupinderpal
28		Training Phase	2 days	9/15/22 8:00 AM	9/16/22 5:00 PM		Arpit;Madhvan;Rupinderpal
29		**Testing**	**13 days**	**9/23/22 8:00 AM**	**10/11/22 5:00 PM**	**26**	
30		Making final updates	2 days	9/23/22 8:00 AM	9/26/22 5:00 PM		Arpit;Madhvan;Rupinderpal
31		Unit Testing	1 day	10/6/22 8:00 AM	10/6/22 5:00 PM		Arpit
32		Integrated Testing	1 day	10/7/22 8:00 AM	10/7/22 5:00 PM		Madhvan
33		System Testing	1 day	10/10/22 8:00 AM	10/10/22 5:00 PM		Rupinderpal
34		Test Result	1 day	10/11/22 8:00 AM	10/11/22 5:00 PM		Arpit;Madhvan;Rupinderpal
35		**Accuracy Judgement**	**3 days**	**10/14/22 8:00 AM**	**10/18/22 5:00 PM**	**29**	
36		Measure Results	1 day	10/14/22 8:00 AM	10/14/22 5:00 PM		Arpit;Madhvan;Rupinderpal
37		Precision Analysis	2 days	10/17/22 8:00 AM	10/18/22 5:00 PM		Arpit;Madhvan;Rupinderpal
38		Documentation	5 days	10/20/22 8:00 AM	10/26/22 5:00 PM		Arpit;Madhvan;Rupinderpal


Jul 22
7 Aug 22
14 Aug 22
21 Aug 22
28 Aug 22
4 Sep 22
11 Sep 22
18 Sep 22
25 Sep 22
2 Oct 22
9 Oct 22
16 Oct 22
23 Oct 22
30 Oct 22
Arpit;Madhvan;Rupinderpal
Arpit;Madhvan;Rupinderpal
Arpit;Madhvan;Rupinderpal
Arpit
Rupinderpal
Madhvan
Rupinderpal
Arpit
Madhvan
Arpit
Arpit;Madhvan;Rupinderpal
Arpit;Madhvan;Rupinderpal
Arpit;Madhvan;Rupinderpal
Arpit
Madhvan;Rupinderpal
Madhvan;Rupinderpal
Arpit
Arpit;Rupinderpal
Arpit;Madhvan;Rupinderpal
Arpit;Madhvan;Rupinderpal
Arpit;Madhvan;Rupinderpal
Arpit
Madhvan
Rupinderpal
Arpit;Madhvan;Rupinderpal
Arpit;Madhvan;Rupinderpal
Right click for options

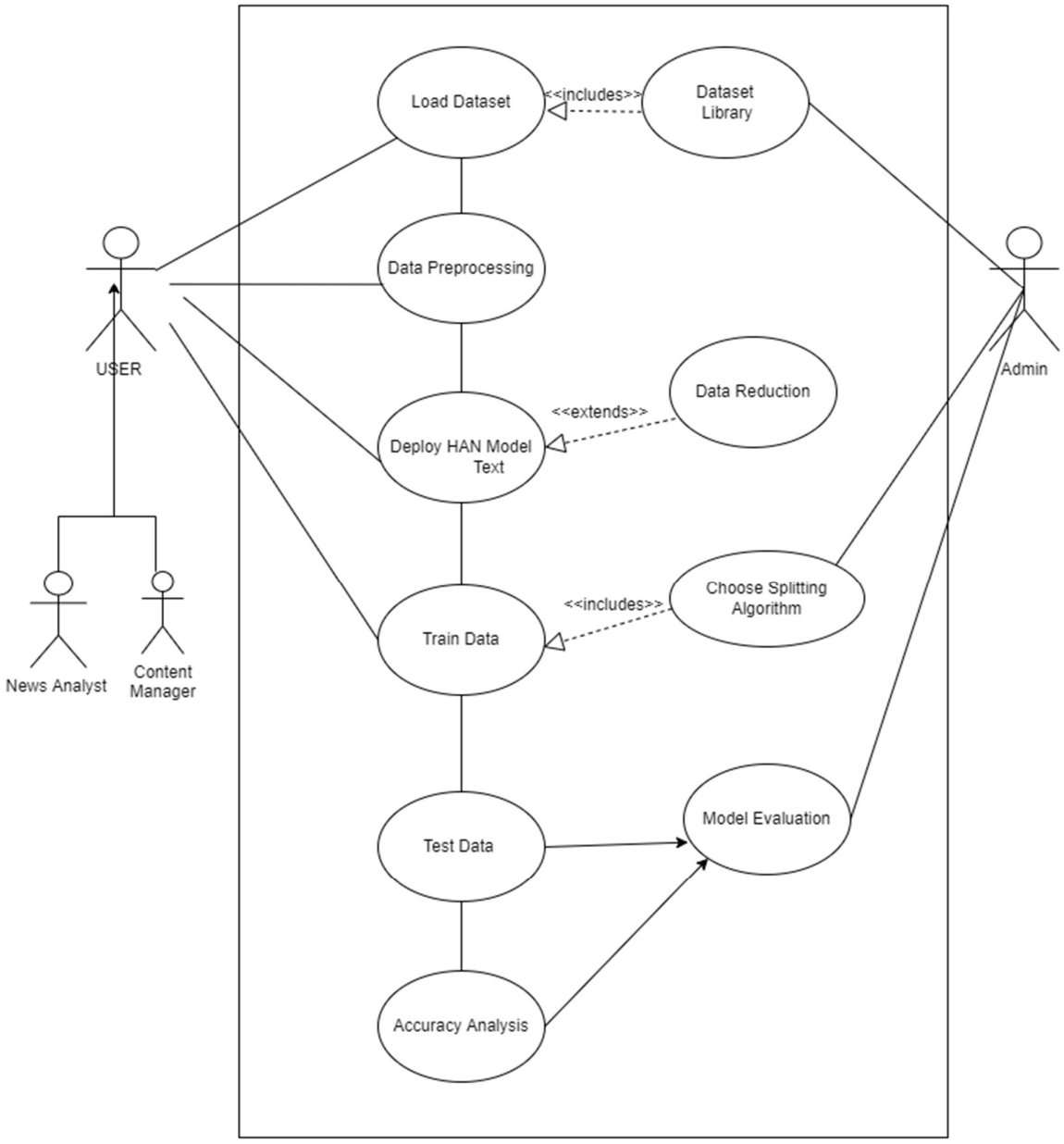

22
7 Aug 22
14 Aug 22
21 Aug 22
28 Aug 22
4 Sep 22
11 Sep 22
18 Sep 22
25 Sep 22
2 Oct 22
9 Oct 22
16 Oct 22
23 Oct 22
30 Oct 22
Arpit;Madhvan;Rupinderpal
Arpit;Madhvan;Rupinderpal
Arpit
Madhvan;Rupinderpal
Madhvan;Rupinderpal
Arpit
Arpit;Rupinderpal
Arpit;Madhvan;Rupinderpal
Arpit;Madhvan;Rupinderpal
Arpit;Madhvan;Rupinderpal
Arpit
Madhvan
Rupinderpal
Arpit;Madhvan;Rupinderpal
Arpit;Madhvan;Rupinderpal
Arpit;Madhvan;Rupinderpal
Arpit;Madhvan;Rupinderpal

Work Breakdown Structure :

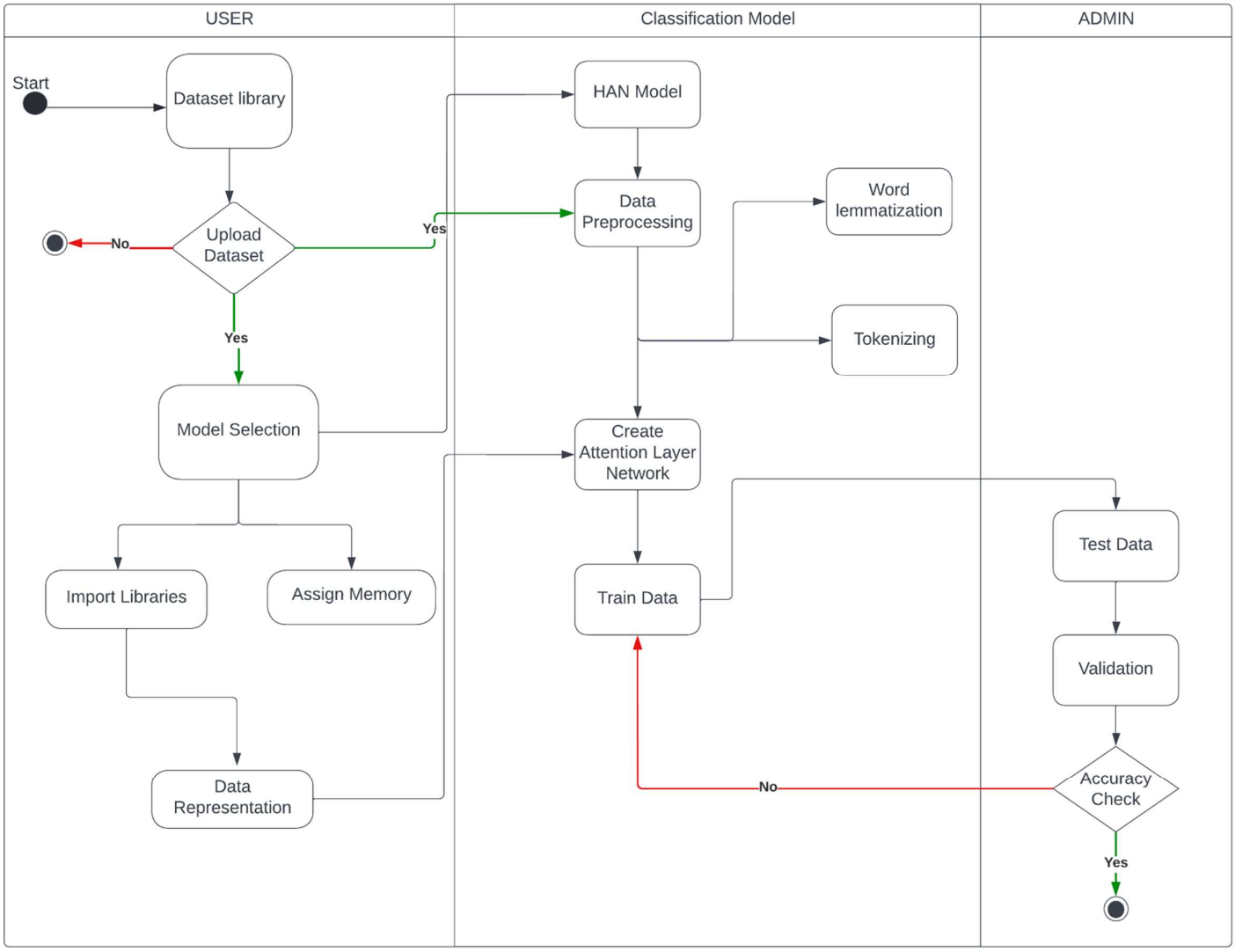
Serial No.	Task Name	Duration	Start Time	End Time
	Text Classification	**60 days**	**8/4/2022**	**10/26/2022**
1	**Planning**	**5 days**	**8/4/2022**	**8/10/2022**
1.1	Defining Strategy	1 day	8/4/2022	8/4/2022
1.2	Literature Review	1 day	8/5/2022	8/5/2022
1.3	Project Scope	1 day	8/10/2022	8/10/2022
2	**Requirements**	**5 days**	**8/17/2022**	**8/23/2022**
2.1	Feasibility Study	2 days	8/17/2022	8/18/2022
2.1.1	Schedule Feasibility	1 day	8/17/2022	8/17/2022
2.1.2	Technical Feasibility	1 day	8/17/2022	8/17/2022
2.1.3	Economic Feasibility	1 day	8/17/2022	8/17/2022
2.1.4	Operational Feasibility	1 day	8/18/2022	8/18/2022
2.1.5	Legal Feasibility	1 day	8/18/2022	8/18/2022
2.1.6	Cultural/Behavioural Feasibility	1 day	8/18/2022	8/18/2022
2.2	Equipment Analysis	2 days	8/19/2022	8/22/2022
2.3	Construction Analysis	3 days	8/19/2022	8/23/2022
2.3.1	Functional Requirements	1 day	8/19/2022	8/19/2022
2.3.2	Non-Functional Requiremnets	2 days	8/19/2022	8/22/2022
2.3.3	SRS Document	1 day	8/23/2022	8/23/2022
3	**Design**	**11 days**	**8/25/2022**	**9/8/2022**
3.1	Content Building	4 days	8/25/2022	8/30/2022
3.2	Defining Data Hierarchy	2 days	8/31/2022	9/1/2022
3.3	Develop Model frame	3 days	9/6/2022	9/8/2022
3.3.1	DFD	1 day	9/6/2022	9/6/2022
3.3.2	UML diagram	2 days	9/7/2022	9/8/2022
3.3.3	ER diagram	1 day	9/8/2022	9/8/2022
4	**Implementation**	**5 days**	**9/12/2022**	**9/16/2022**
4.1	Deployment Phase	1 day	9/12/2022	9/12/2022
4.2	Training Phase	2 days	9/15/2022	9/16/2022
5	**Testing**	**13 days**	**9/23/2022**	**10/11/2022**
5.1	Making final updates	2 days	9/23/2022	9/26/2022
5.2	Unit Testing	1 day	10/6/2022	10/6/2022
5.3	Integrated Testing	1 day	10/7/2022	10/7/2022
5.4	System Testing	1 day	10/10/2022	10/10/2022
5.5	Test Result	1 day	10/11/2022	10/11/2022
6	**Accuracy Judgement**	**3 days**	**10/14/2022**	**10/18/2022**
6.1	Measure Results	1 day	10/14/2022	10/14/2022
6.2	Precision Analysis	2 days	10/17/2022	10/18/2022
7	**Documentation**	**5 days**	**10/20/2022**	**10/26/2022**

3. Analysis Phase :

3.1 i) Use-Case Diagram :

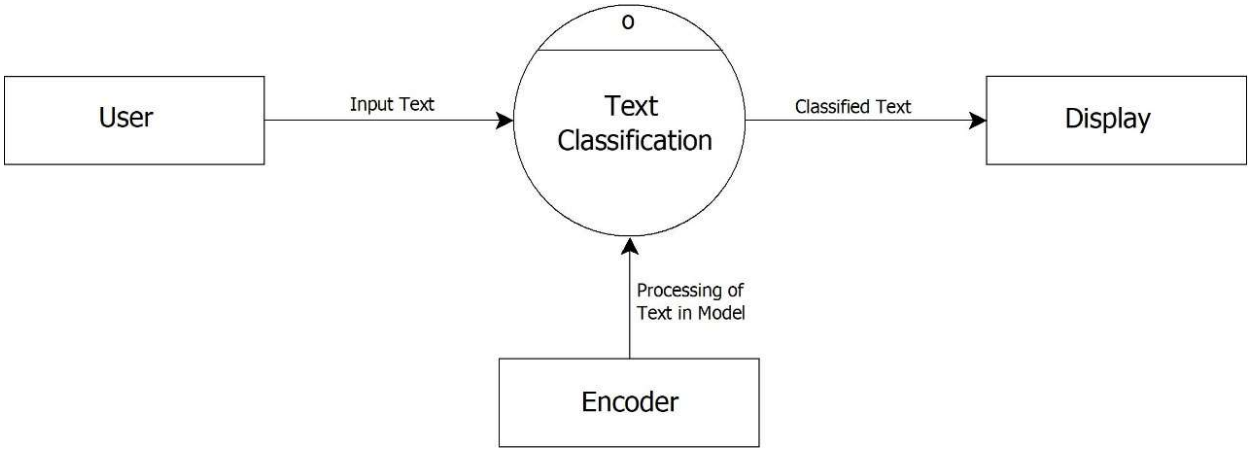


3.2 ii) Activity / Swimlane Diagram :

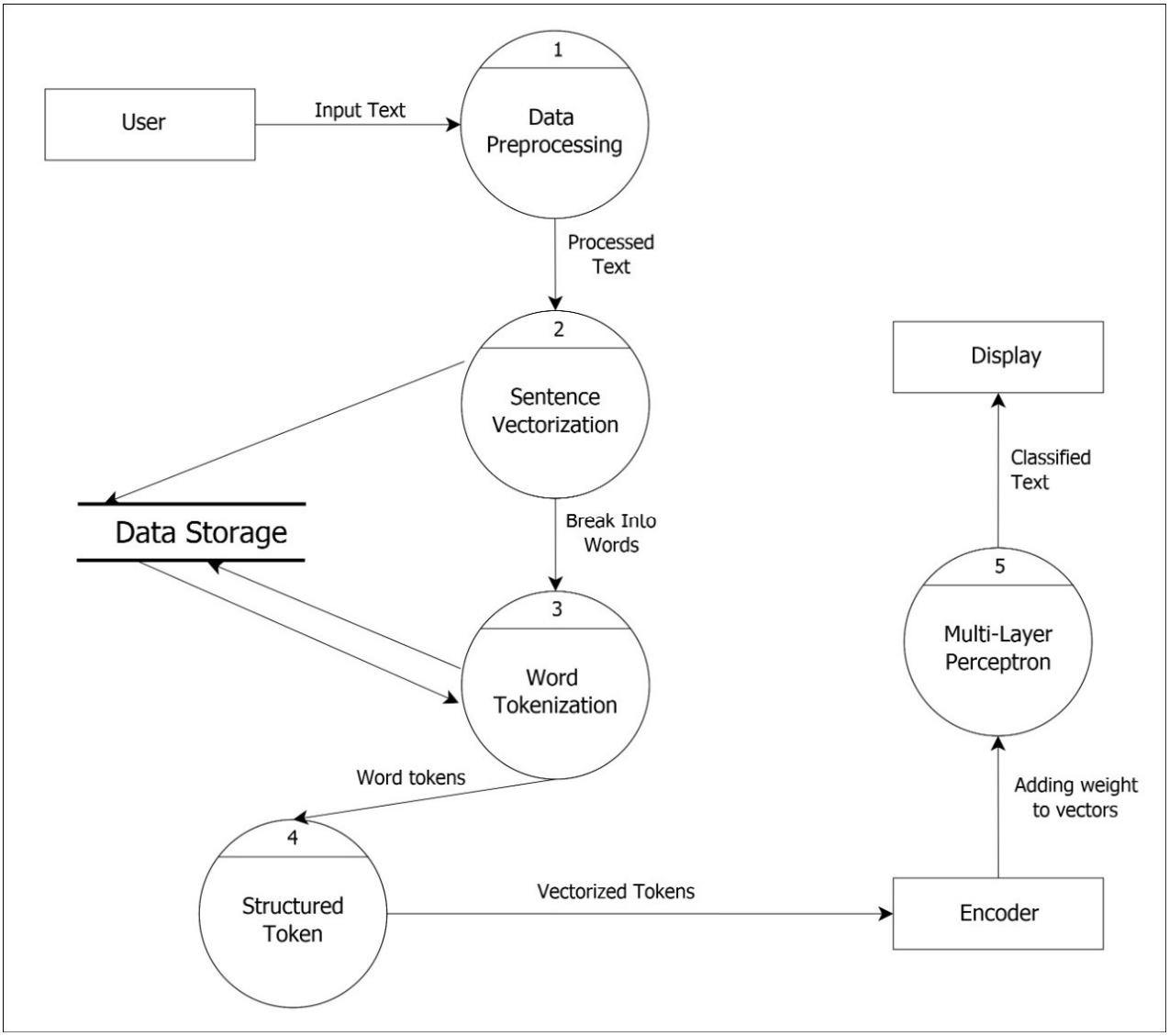


3.3 iii) Data Flow Diagrams :

Level 0:



Level 1:



3.4 iv Software Requirement Specification (IEEE Format)

1. Introduction

1.1 Purpose

The purpose of this SRS document is to provide a detailed overview of our project requirements, specifications and its goals. This document also specifies the intended audience and outlines the overall oof this project. The Text Classification Model classifies the given text using method of Hierarchical Attention Network from Yang et al. from 2016.The main aim is to assign unstructured data to one or multiple classes for further ease of data analysis.

1.2 Intended Audience and Reading Suggestions:

This document is intended for readers like documentation writers, news classifiers, testers, developers, marketing staff, and other data related portfolios where people work with classification.

1.3 Product Scope

Classification of Text using hierarchical attention networks filters the data making only relevant and important matter to be considered. The hierarchical structure of documents (document-sentences-words) is considered and an attention mechanism is used to find most important words and sentences in a document while keeping an eye on the context of words. Text classification can be used in a broad range of contexts such as classifying short texts (e.g., tweets, headlines, chatbot queries, etc.) or organizing much larger documents (e.g., customer reviews, news articles, legal contracts, long form customer surveys, etc.). Real life and industrial applications include sentiment analysis, news classifications, topic labeling, spam/intent detections and corporate goals include classifiers like:-

- Product analysis
- Brand monitoring
- Market research
- Customer support

1.4 References

[1] Text Classification overview. Link : https://monkeylearn.com/text-classification/

[2] Hierarchical Attention Network definition. Link : https://ojs.aaai.org/index.php/AAAI/article/view/4924

[3] Keras python library for model. Link : https://keras.io/

[4]Scope of Model. Link: https://www.latentview.com/blog/real-world-applications-of-text-classification/

[5] Model parameters and testing. Link: https://www.oreilly.com/library/view/practical-natural-language/9781492054047/ch04.html

[6] Dataset used for model : Link : https://www.kaggle.com/c/learn-ai-bbc

2. Overall Description

2.1 Product Perspective

Text classification is a machine learning technique that assigns a set of predefined categories to open-ended text. Text classifiers can be used to organize, structure, and categorize any kind of text – from documents, medical studies and files, and all over the web.

2.2 Product Functions

In this text classification model, the text data is sent to the system model where the text is now broken into sentences and later vectorized which is later used in Encoder. Now the sentence is broken into words where they are tokenized and then vectorized. Afterwards it is sent to Encoder where the Vectorized words are classified and then later co-joint into sentences to display the classified text.

2.3 Operating Environment

Python simulation environment like Jupyter, Spyder etc. can be used for executing this model. In this project, we use Google colab SAAS cloud environment to simulate our model as it supports all the libraries and models of Machine Learning.

2.4 Design and Implementation Constraints

Constraints include memory requirements in the compiler for inclusion of appropriate libraries and communication protocols include design conventions and python programming standards.

2.5 Assumptions and Dependencies

We assume the dataset to be discrete and consisting of words and sentences while the dependencies include keras library and other functional plot diagrams for perfect visualization of results and accuracy.Attention Layer Mechanisms and Tokenizing are other essential dependencies in the project.

3. External Interface Requirements

3.1 Hardware Interfaces

Hardware requirements include PC/laptop and input data could be in hard copy that can later be scanned while communication protocols include HTTP, HTTPS if data loaded from web servers and various websites.

3.2 Software Interfaces

Software interfaces include datasets from Kaggle, TensorFlow and other web-based documentaries or as per user's choice. Software tools include python compilers and editors for running the code and validation of model. Besides this we also need graphics libraries for plotting of analysis graphs and result prediction features.

3.3 Performance Requirements

Text Classification results are measured and validated using performance metrics of suitable parameters like Classification accuracy defined as number of correct predictions made as a ratio of all predictions made.

3.4 Software Quality Attributes

Quality attributes of the product include adaptability to any interface, correctness to algorithms, flexibility of input data ,maintainability of model parameters and reliability of functional libraries.

USER STORY CARD: Front side

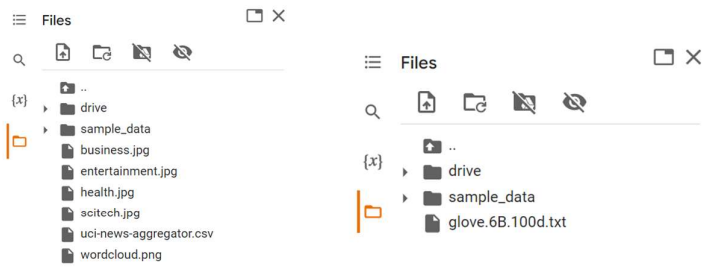
- Upload Document for Classification

CHOOSE FILES

```
from google.colab import files
uploaded=files.upload()
```

```
Choose Files  newstest.txt
• newstest.txt(text/plain) - 64 bytes, last modified: 11/26/2022 - 100% done
Saving newstest.txt to newstest.txt
```

- Successful Upload of files in Data Library



- Category News Labels

	ArticleId	Text	Category	CategoryId
0	1833	worldcom ex bos launch defence lawyer defendin...	business	0
1	154	german business confidence slide german busine...	business	0
2	1101	bbc poll indicates economic gloom citizen majo...	business	0
3	1976	lifestyle governs mobile choice faster better ...	tech	1
4	917	enron boss 168m payout eighteen former enron d...	business	0
...	...	...	...	...
1485	857	double eviction big brother model caprice holb...	entertainment	4
1486	325	dj double act revamp chart show dj duo jk joel...	entertainment	4
1487	1590	weak dollar hit reuters revenue medium group r...	business	0
1488	1587	apple ipod family expands market apple expande...	tech	1
1489	538	santy worm make unwelcome visit thousand websi...	tech	1

1490 rows × 4 columns

- Classified News Headlines Output

```
"Hooli stock price soared after a dip in PiedPiper revenue growth."
  - Predicted as: 'business'

"Captain Tsubasa scores a magnificent goal for the Japanese team."
  - Predicted as: 'sport'

"Merryweather mercenaries are sent on another mission, as government oversight groups call for new sanctions."
  - Predicted as: 'business'

"Beyoncé releases a new album, tops the charts in all of south-east Asia!"
  - Predicted as: 'entertainment'

"You won't guess what the latest trend in data analysis is!"
  - Predicted as: 'tech'
```

TEST CASE I

```
example1 = "test2.txt.txt"
file1 = open(example1, "r")
FileContent = file1.read()
FileContent
title_to_category(FileContent)
print('news title', '                              ', 'category','\n',FileContent, '      ', title_to_category(FileContent))
```

```
news title                                  category
 McDonald's February same-restaurant sales fall 0.3%
        business
```

NEWS TAGS

```
title_to_category(FileContent)
print('NEWS TITLE','                              ','CATEGORY','\n',FileContent,'------>      ',title_to_category(FileContent))
```

Python

```
NEWS TITLE                                  CATEGORY
 Taylor Swift Tops Music's Top 40 Money Makers 2014 List â€” How Much Did She  ...
 ------>       entertainment
```

USER STORY CARD: Back side

- **SUCCESS Conclusions**

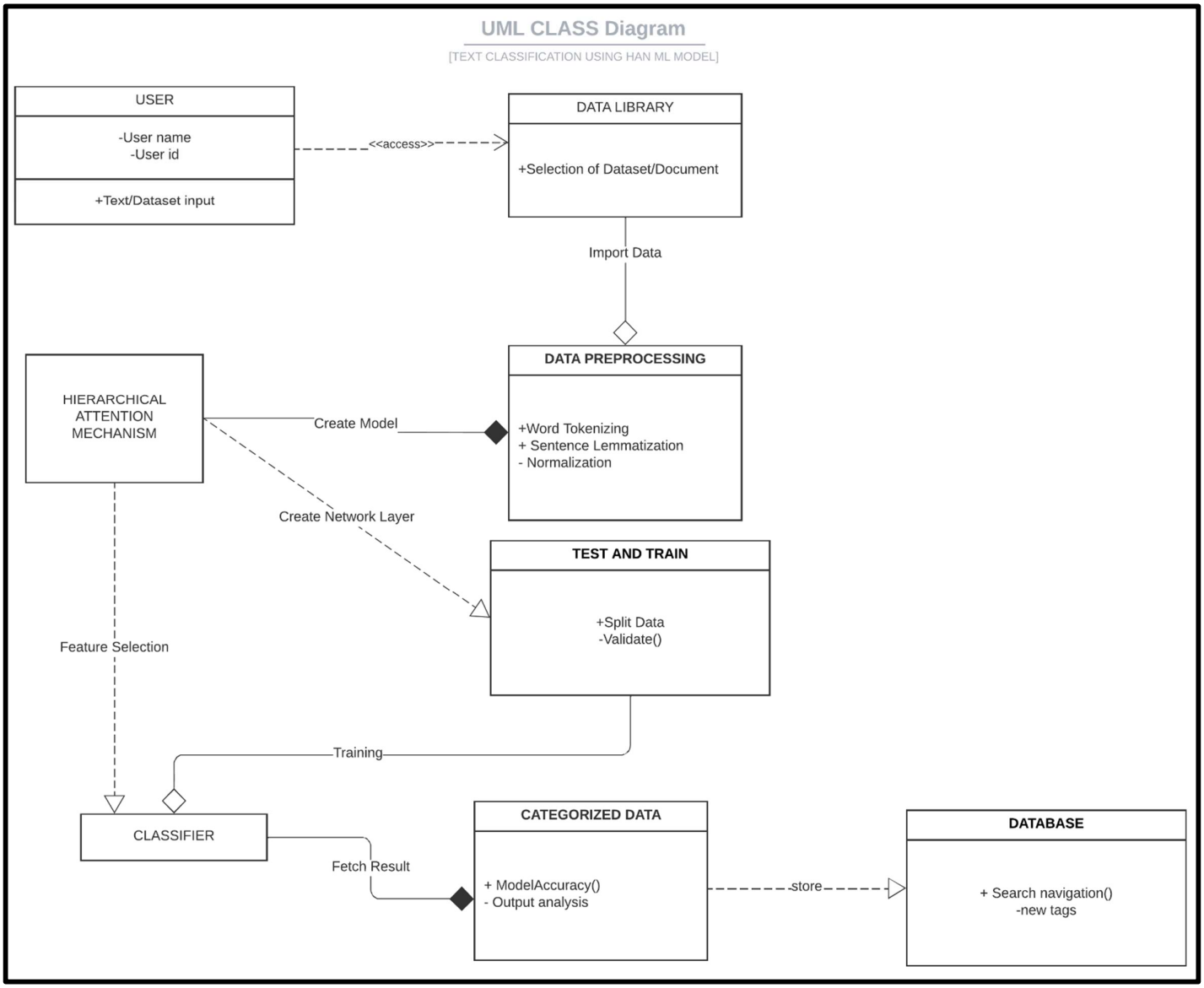
  - Document Uploaded Successfully.
  - Sentence and Word mechanism made on every line.
  - News Categories successfully deployed.
  - Classified the Document Headlines Precisely.
  - Storage of Tagged and Categorized News in Data Archive for further use.

- **FAILURE Possibilities**

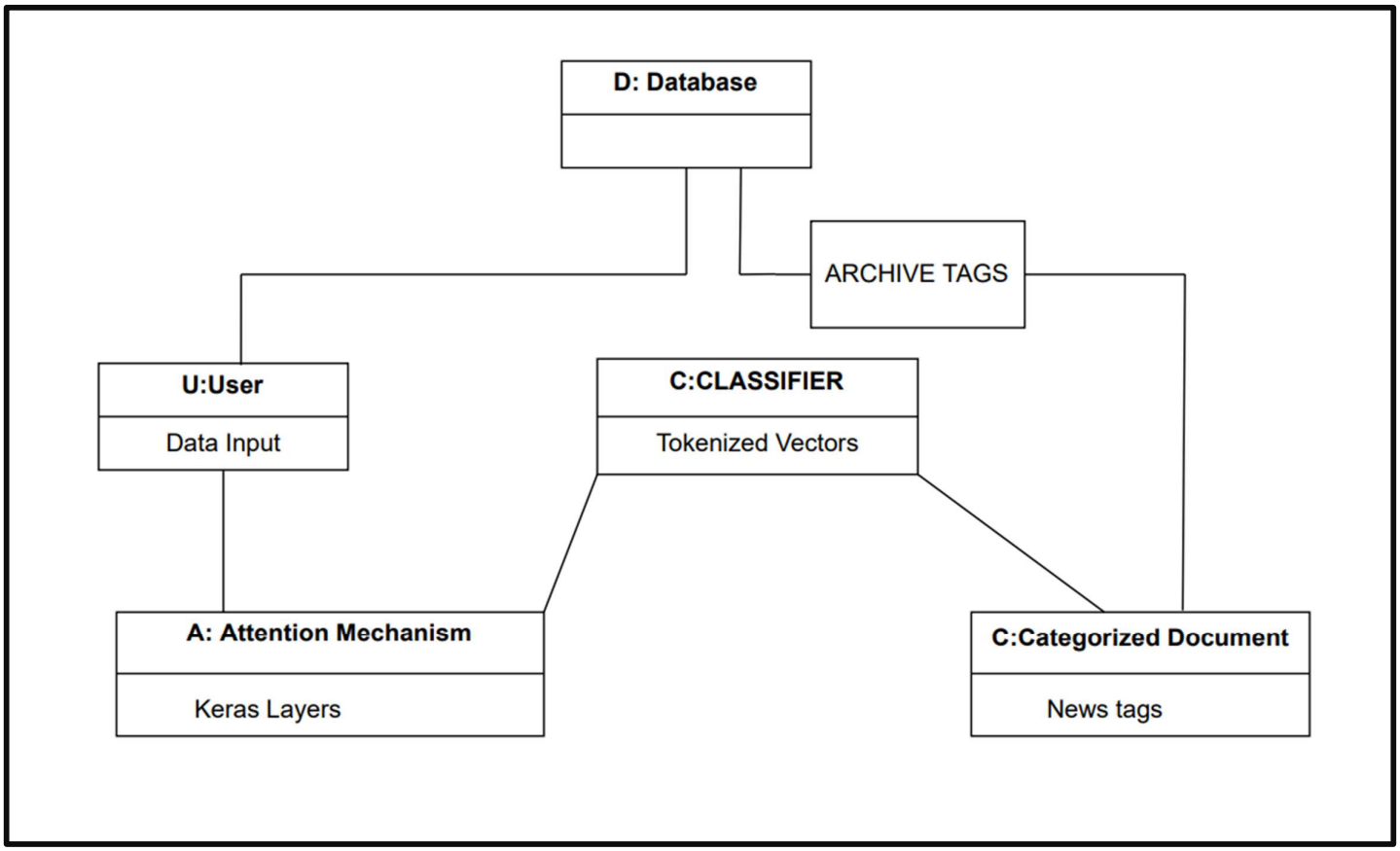
  - No File chosen for uploading.
  - File not compatible with interface.
  - Attention Layers not concatenated accurately.
  - Hierarchy of words unmatched with categories.
  - Irrelevant categorization of headline .

4. Design Phase

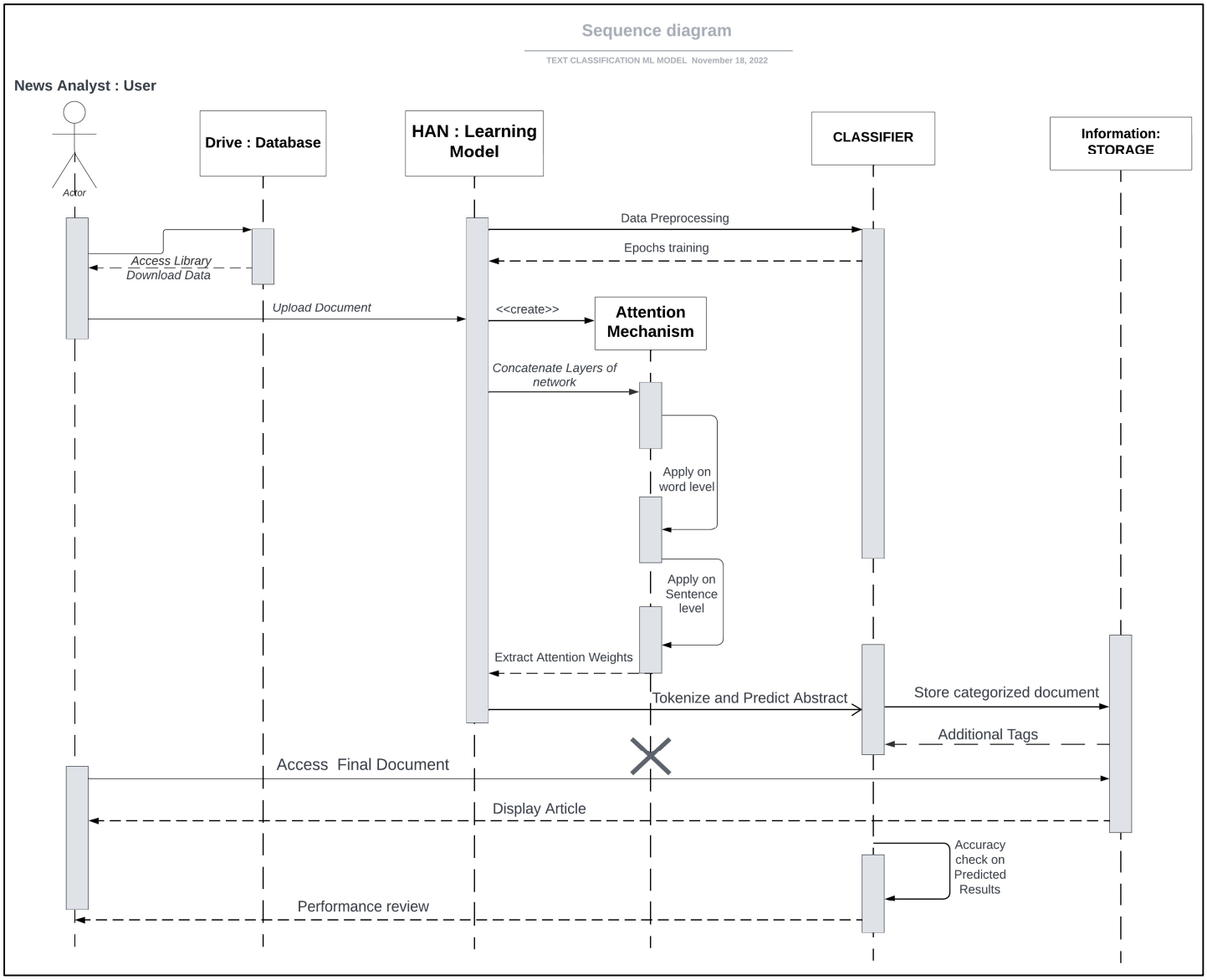
4.1 Class Diagram :



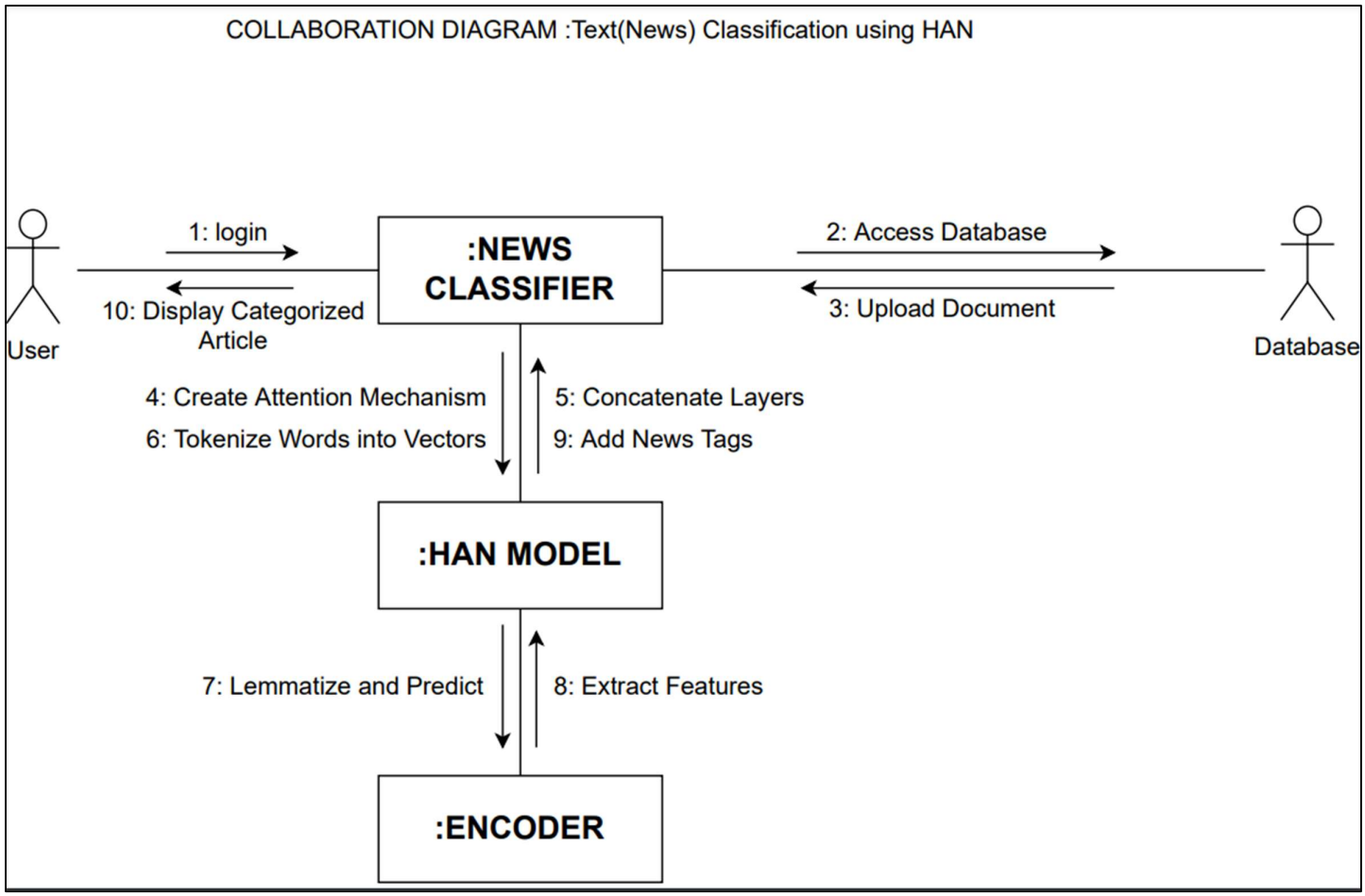
4.2 Object Diagram :



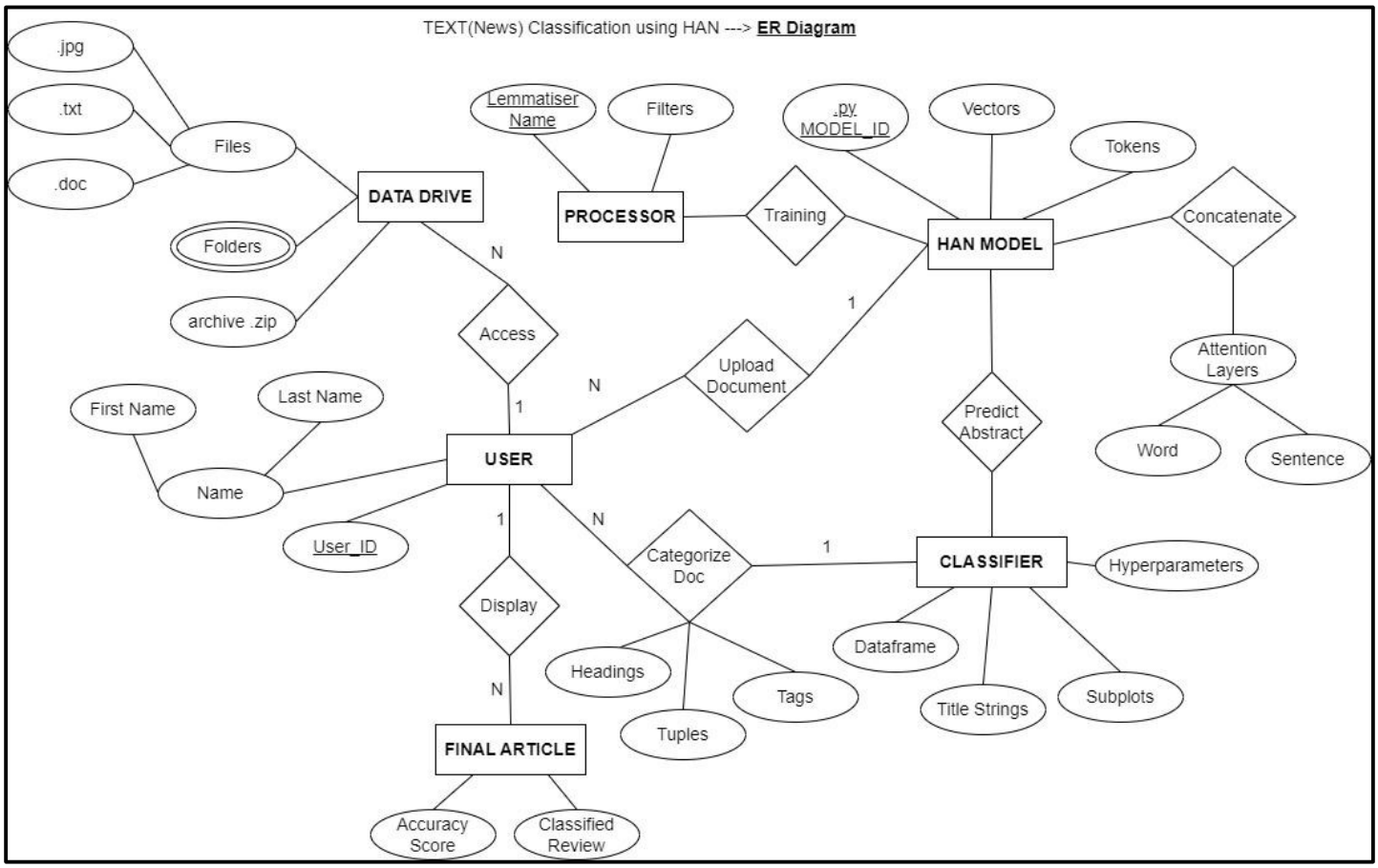
4.3 Sequence Diagram :



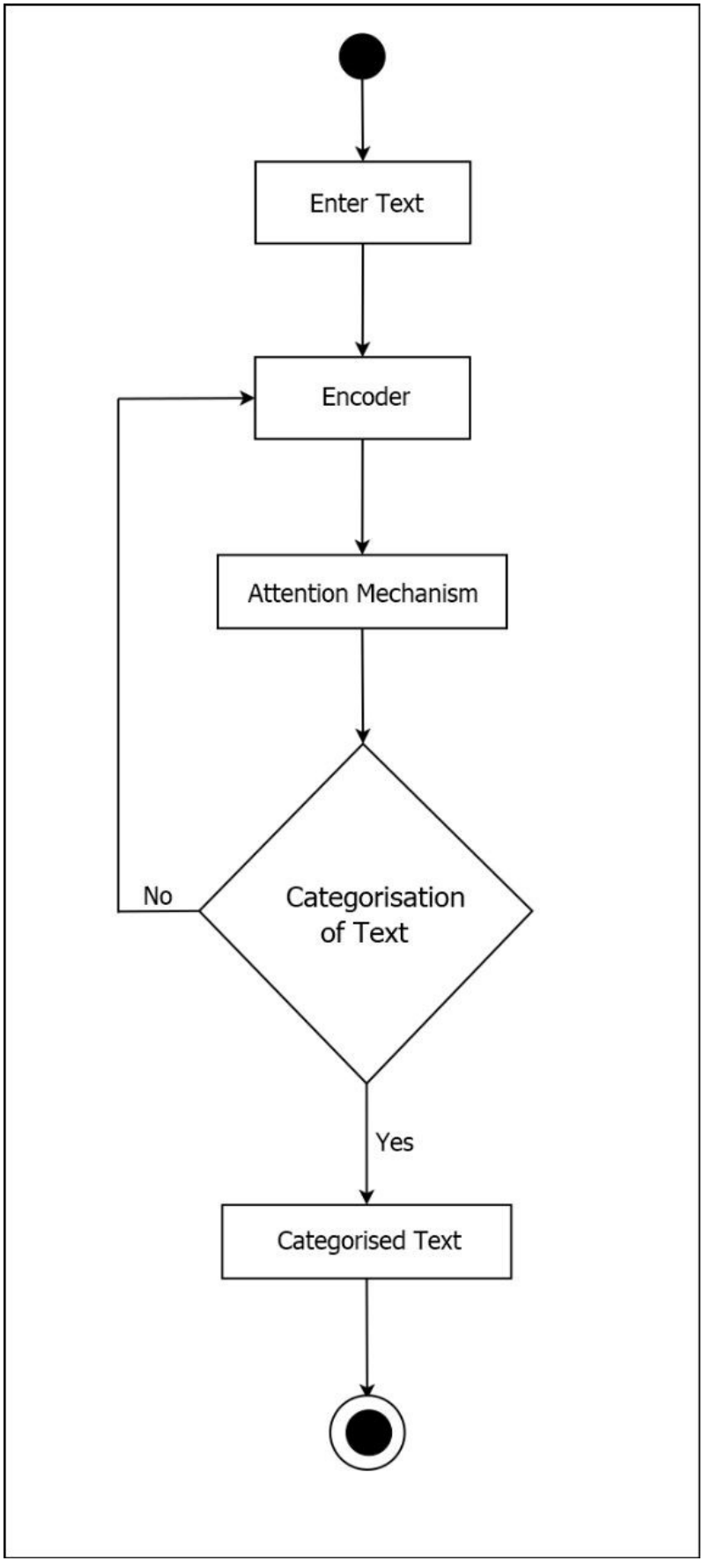
4.4 Collaboration Diagram :



4.5 Database Design : ER Diagram :

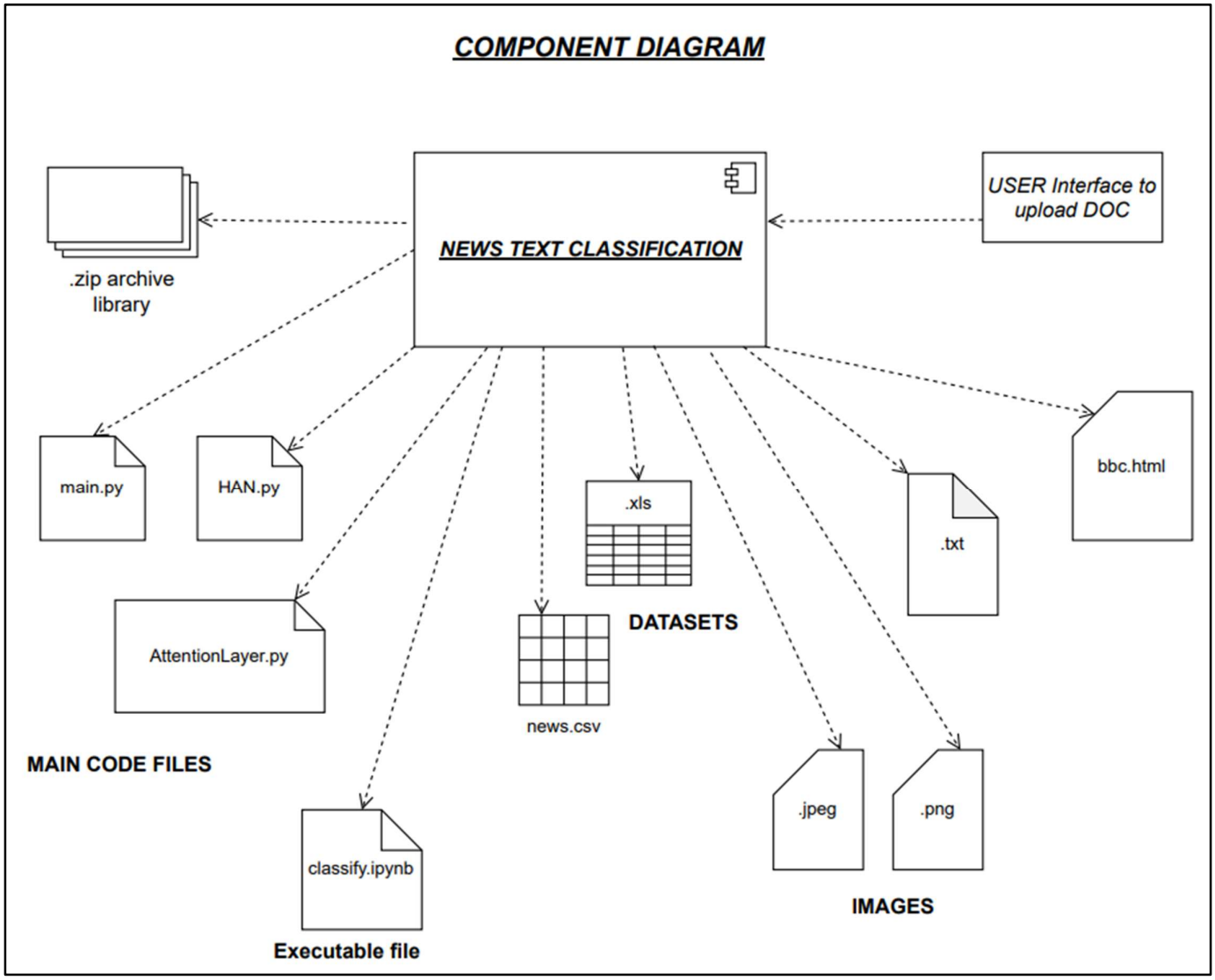


4.6 State Chart Diagram :

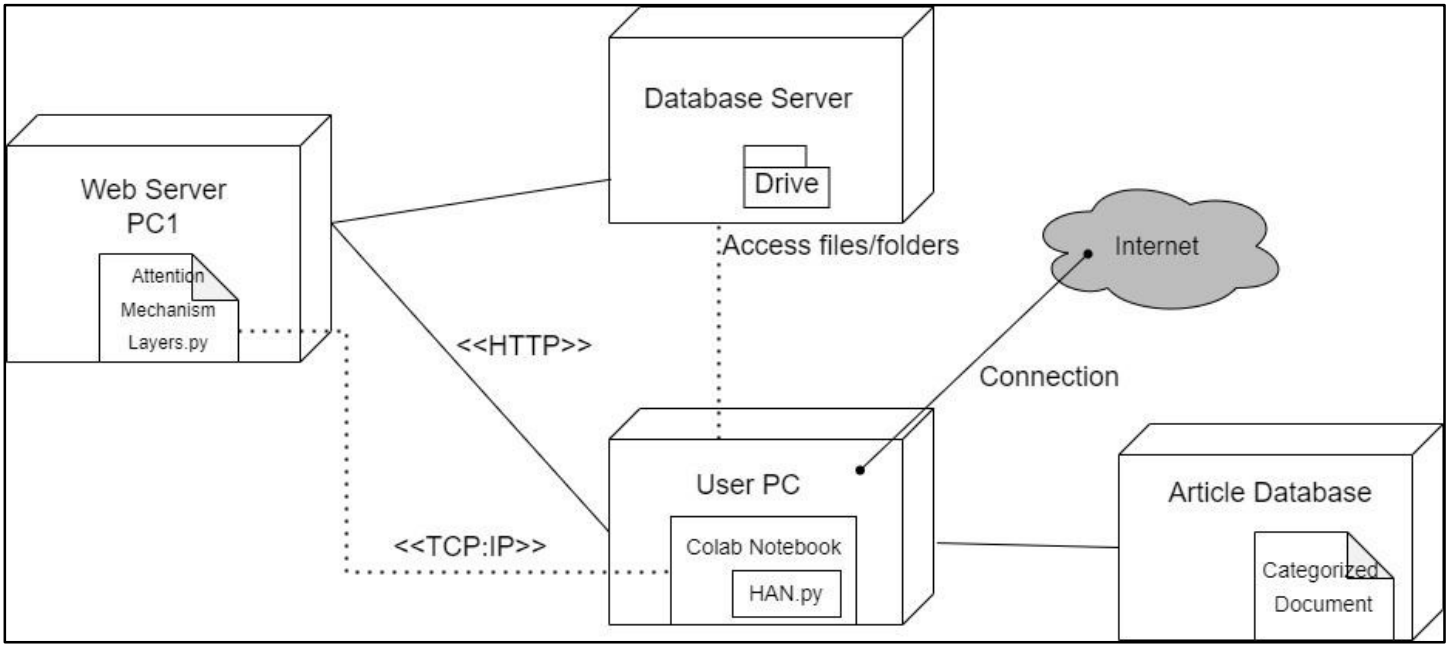


4. Implementation

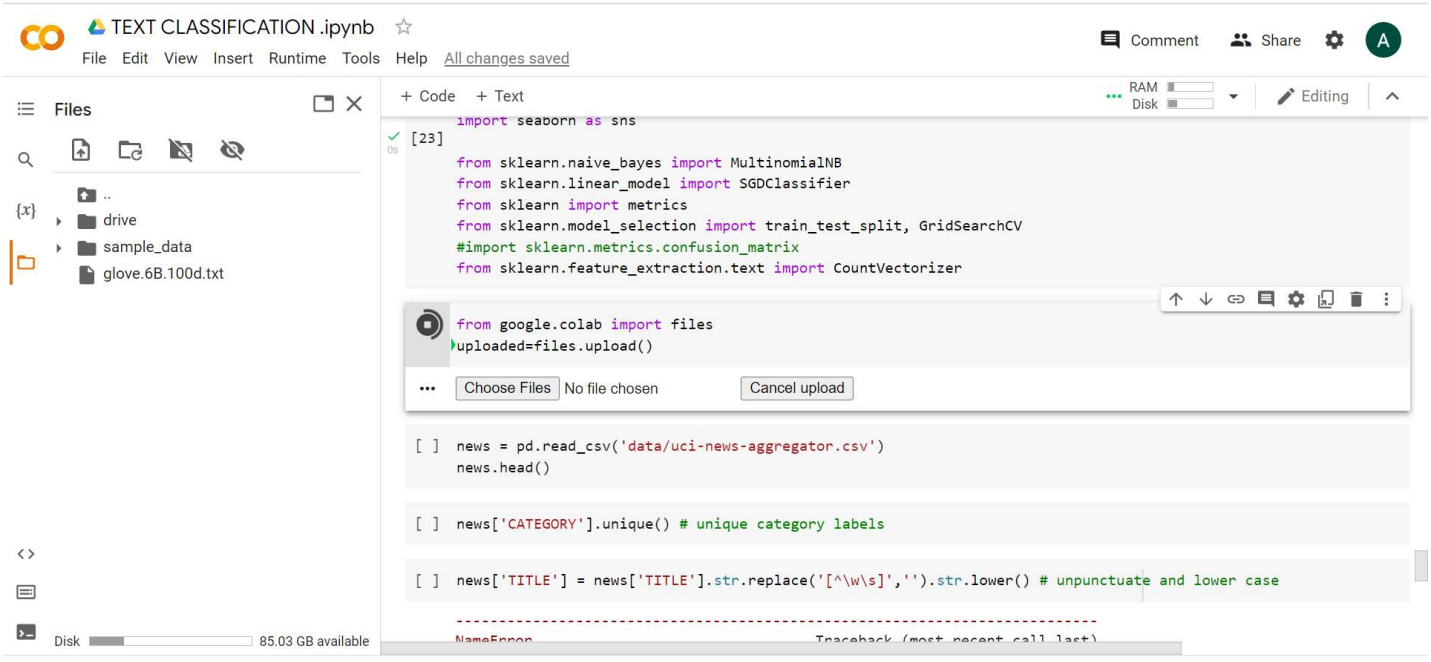
4.1 Component Diagram :



4.2 Deployment Diagram :



4.3 Screenshots of Working Project :



Upload Document Interface

```
from google.colab import files
uploaded=files.upload()
```

```
Choose Files newstest.txt
• newstest.txt(text/plain) - 64 bytes, last modified: 11/26/2022 - 100% done
Saving newstest.txt to newstest.txt
```

News Classification Output

TEST CASE IV

```
example1 = "T1.txt"
file1 = open(example1, "r")
FileContent = file1.read()
FileContent
title_to_category(FileContent)
print('NEWS TITLE',' ','CATEGORY','\n',FileContent,'------>  ',title_to_categ
```

```
NEWS TITLE                                CATEGORY
 Miley Cyrus Has A Backstage Tour-Drobe Drama ------>       entertainment
```

```
print('news title', '                                ', 'category', '\n'
      'Bank of England staff to go on strike', '     ', title_to_category('Bank of England staff to go on str
      'Trump stance could damage Earth - Hawking', ' ', title_to_category('Trump stance could damage Earth - H
      'Olivia de Havilland sues over TV show', '     ', title_to_category('Olivia de Havilland sues over TV sh
     )
```

```
news title                                category
Bank of England staff to go on strike     business
Trump stance could damage Earth - Hawking science and technology
Olivia de Havilland sues over TV show     entertainment
```

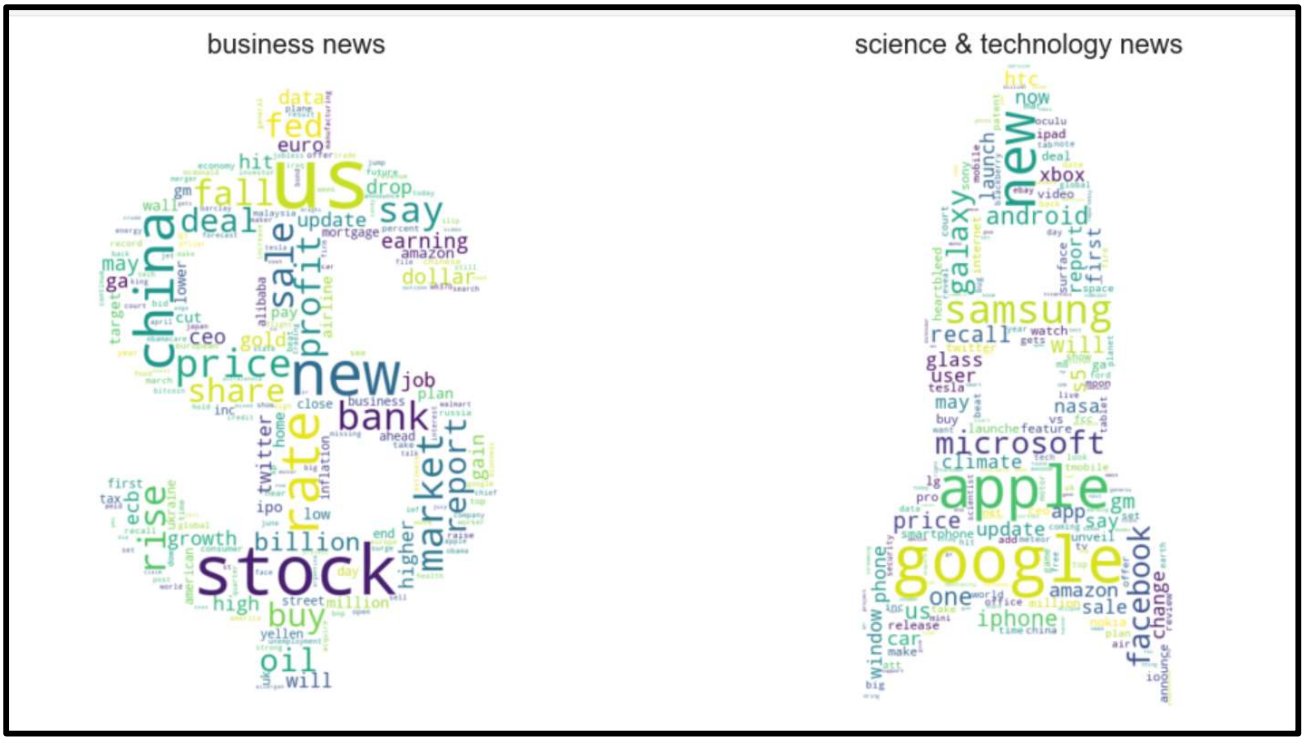
```
# Add to database
articleDB.loc[len(articleDB.index)] = [sent_list[0].rstrip('.'),
                                       ''.join(summary),
                                       article,
                                      cat,
                                      mostAtt_words]
display(articleDB)
```

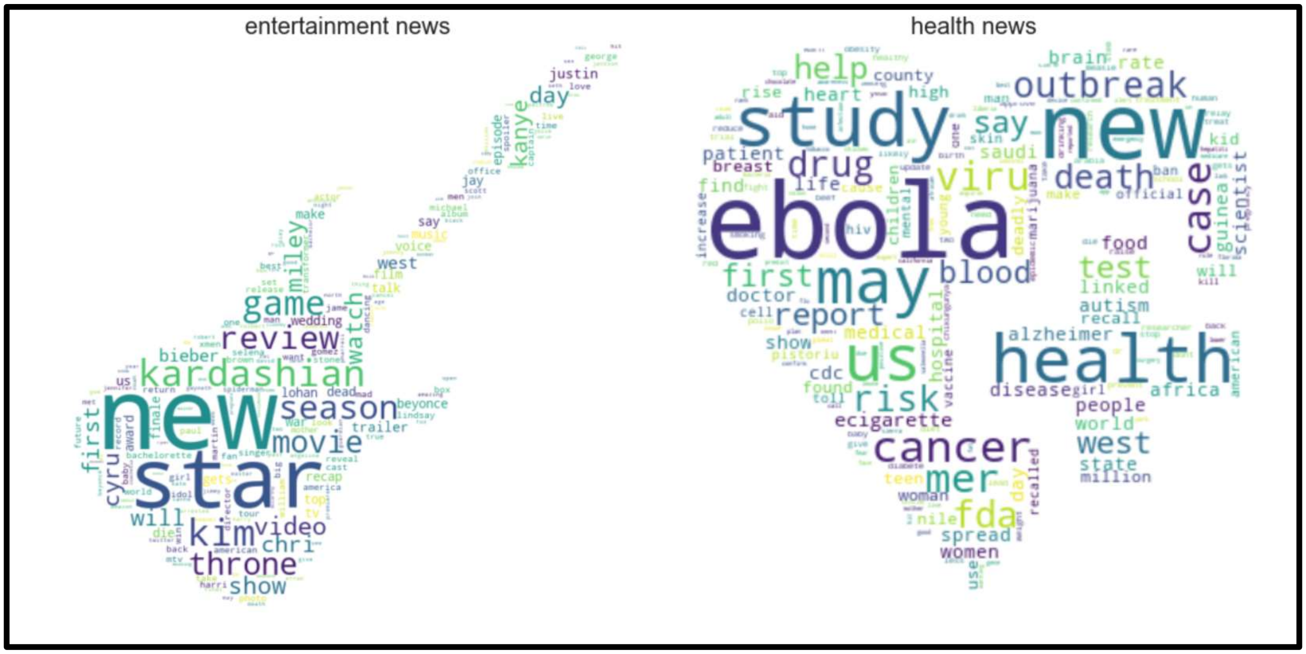
	Title	Summary	Text	Category	Tags
0	Holy recasting, Batman!	Affleck had been due to direct as well as star...	Holy recasting, Batman! The search is on for a...	entertainment	[upcoming, star, sequel, 1960s, tim]
1	Facebook said participating teens had provided...	It has also emerged that Google ran a similar ...	Facebook said participating teens had provided...	tech	[data, encrypted, device, access, apps]
2	In the final three months of 2018, the economy...	In the final three months of 2018, the economy...	In the final three months of 2018, the economy...	business	[eurozone, said, economic, data, economy]
3	That's the norm when it comes to Hollywood's b...	They've shared figures which reveal only 4% of...	That's the norm when it comes to Hollywood's b...	entertainment	[bfi, film, directed, event, screened]

```
example1 = "test3.txt"
file1 = open(example1, "r")
FileContent = file1.read()
FileContent
title_to_category(FileContent)
print('NEWS TITLE',' ','CATEGORY','\n',FileContent,'------>      ',title_to_category(FileContent))
```

[38] Python

```
NEWS TITLE                              CATEGORY
 Taylor Swift Tops Music's Top 40 Money Makers 2014 List â€” How Much Did She  ...
 ------>       entertainment
```




entertainment news
health news
new
star
kardashian
kim
throne
game
review
season
movie
video
show
ebola
study
new
health
may
us
cancer
outbreak
fda
risk
viru
death
case
drug
blood
first
report
test
mer
west

5.Testing

5.1 CYCLOMATIC COMPLEXITY

V(G)= P+1 , where P is the number of Predicate Nodes in the flow graph of G.

- **DataPreprocessing.py**
  V(G)=2+1
  V(G)=3
  {Since there are 2 for loops included in this function for cleaning the data }

```
def cleanString(review,stopWords):
    lemmatizer = WordNetLemmatizer()
    returnString = ""
    sentence_token = tokenize.sent_tokenize(review)
    idx_list = []
    for j in range(len(sentence_token)):
        single_sentence = tokenize.word_tokenize(sentence_token[j])
        sentences_filtered = [(idx,lemmatizer.lemmatize(w.lower())) for idx,w in enumerate(single_sentence)
                              if w.lower() not in stopWords and w.isalnum()]
        idx_list.append([x[0] for x in sentences_filtered])
        word_list = [x[1] for x in sentences_filtered]
        returnString = returnString + ' '.join(word_list) + ' . '

    return returnString, idx_list
```

```
Cleans raw data using the cleanString() function from above.
English stopwords are used from nltk library.
Cleaned dataset is saved in 'data_cleaned' pandas dataframe.
Labels are converted to numbers,
"""
articles = []
n = data_df['Text'].shape[0]
col_number = data_df.columns.get_loc('Text')
stopWords = set(stopwords.words('english'))
data_cleaned = data_df.copy()
for i in range(n):
    temp_string,idx_string = cleanString(data_df.iloc[i,col_number],stopWords)
    articles.append(temp_string)
    print(str(i+1)+' of '+str(n)+" articles cleaned.",end='\r')

data_cleaned.loc[:,'Text'] = pd.Series(articles,index=data_df.index)
data_cleaned.loc[:,'Category'] = pd.Categorical(data_cleaned.Category)
data_cleaned['Code'] = data_cleaned.Category.cat.codes
categoryToCode = dict( enumerate(data_cleaned['Category'].cat.categories))

data_cleaned.head()
```

- **LabelData.py**
  V(G)=4+1
  V(G)=5
  {Since there are 4 loops included in this function for getting the data from bbc news training set and using the four labels → business, science and tech, health and entertainment }

```
import os
cwd = os.getcwd()  # Get the current working directory (cwd)
filepath=('/content/drive/MyDrive/Colab Notebooks/bbc')
#filepath = os.path.join(cwd,"bbc")
articles = []
category_num = {}
count = 0
for item in os.listdir(filepath):
    category_num[item] = count
    count += 1
    if os.path.isdir(os.path.join(filepath,item)):
        sub_dir = os.path.join(filepath,item)
        files = [f for f in os.listdir(sub_dir)
         if os.path.isfile(os.path.join(sub_dir,f))]
        for file in files:
            with open(os.path.join(sub_dir,file)) as text:
                data = text.read()
                paragraphs = data.split('\n', -1)
                title = paragraphs[0] + '. '
                paragraphs[0] = title
                data = ''.join(paragraphs)
                articles.append((data,item))

data_df = pd.DataFrame(data=articles,columns=['Text',"Category"])
```

- **Tokenize.py**
  V(G)=2+1
  V(G)=3
  {Since there are 2 loops included in this function for tokenizing and calculating average words and sentences in the training document}

```
"""
Compute average number of words in each sentence and average number of sentences in each document.
"""
n_sent = 0
n_words = 0
for i in range(data_df.shape[0]):
    sent = tokenize.sent_tokenize(data_df.loc[i,'Text'])
    for satz in sent:
        n_words += len(tokenize.word_tokenize(satz))
    n_sent += len(sent)

print("Average number of words in each sentence: ",round(n_words/n_sent))
print("Average number of sentences in each document: ", round(n_sent/data_df.shape[0]))
```

```
Average number of words in each sentence:  25
Average number of sentences in each document:  19
```

5.2 TEST CASES AND TEST REPORTS

Test Case: 1.1	**Test Case Name:** Upload File
System : News Classifier	**Subsystem:** User Interface
Designed By: CO6SE1	**Design Date:** 20/11/2022
Executed By: CO6SE1	**Execution Date:**29/11/2022
Short Description: Uploading /Importing Invalid Document for Interface	

Pre-Conditions:

1. Drive is mounted and accessible.
2. Availability to choose files with good connectivity.

Step	Action	Expected System Response	Pass/ Fail	Comment
1	Click on Upload Files	The system displays a message asking the user for confirmation	Pass	
2	Select File	The system prompts a dialog box.	Pass	
3	Import Document	The system connects to the PC and shows options to browse and choose files.		
4	Click 'Upload' Button	The system displays invalid document message	Pass	
5	Check post-condition 1		Pass	

Post-Conditions:

Document is not uploaded and is not compatible.

Test Case: 1.2 **Test Case Name:** Wrong Prediction

System : News Classifier **Subsystem:** User Interface

Designed By: CO6SE1 **Design Date:** 20/11/2022

Executed By: CO6SE1 **Execution Date:**29/11/2022

Short Description: Inaccurate prediction of News Category

Pre-Conditions:

1. Document successfully uploaded.
2. Attention Mechanism Layers created.

Step	Action	Expected System Response	Pass/ Fail	Comment
1	Read file content	The system reads the file content and vectorises the sentences.	Pass	
2	Call the 'predictor' function	The system passes the vectors to the attention mechanism for prediction.	Pass	
3	Print Result	The system predicts Correct Title category	Fail	
4	Check post-condition		Pass	

Post-Conditions:

Title Category is incorrect and is depicted by inaccuracy metrics.

Test Case: 1.3 **Test Case Name:** Read File Error

System : News Classifier **Subsystem:** User Interface

Designed By: CO6SE1 **Design Date:** 20/11/2022

Executed By: CO6SE1 **Execution Date:**29/11/2022

Short Description: Updation of new document for categorization

Pre-Conditions:

1. HAN Model Layers successfully concatenated.
2. New Document Uploaded successfully.

Step	Action	Expected System Response	Pass/ Fail	Comment
1	Read file content	The system reads the file content and vectorises the sentences.	Pass	
2	Call the 'predictor' function	The system passes the vectors to the attention mechanism for prediction.	Pass	
3	Display category of New Document	The system calls the model function to update to new document.	Pass	
4	Predict Result	The system clears the previous output and updates the data sent.	Pass	

Post-Conditions:

Category Tag of newly uploaded document is printed clearing the previous output.