A Web-Based Social Media Textual Posts Credibility Verification System using Natural Language Processing

**Derrick Nyaga Duncan**

**136788**

**Group: BBIT 4C**

**An Information System Project 2 documentation submitted to the School of Computing and Engineering Sciences in partial fulfilment of the requirements for the award of the bachelor’s degree in business information technology of   
Strathmore University**

**School of Computing and Engineering Sciences**

**Strathmore University**

**Nairobi, Kenya.**

**December 2023**

# Declaration

I declare that this work has not been previously submitted and approved for the award of a Bachelor’s degree by this or any other University. To the best of my knowledge and belief, the documentationcontains no material previously published or written by another person except where due reference is made in the documentationitself.

Student’s signature:

………………………………….………….. [Signature]

………………………………….………….. [Date]

Approval

The Information System Project 2 documentationof Derrick Nyaga Duncan was reviewed and approved (for examination) by:

Supervisor’s signature:

………………………………….………….. [Signature]

………………………………….………….. [Date]

# Abstract

In recent years, social media has become the primary means by which people received news and sourced for information, with many relying on social media to get updates on current events and information inquiries. However, due to this dependence, people have become prone to misinformation and fake news, resulting in misleading beliefs and decisions, the spread of fear and panic, and polarization. To address these negative effects, it was important to find ways to verify information quickly and accurately.

The study aimed to verify the credibility of information shared on social media and classify them into different categories. The study focused on Twitter, which had over 350 million active users worldwide. Although 80% of these users used Twitter as a source of information and news, it was often associated with the spread of misinformation. Therefore, the study aimed to provide social media users with a safeguard against the negative impacts of fake news and misinformation by enabling fact-checking of information. To achieve this, a transformer model was fine-tuned and applied to incorporate GloVe embeddings for classification. The system has an easy-to-use interface that allows users to enter statements. Following submission, the system cleans the input data and provides a classification indicating whether the statement is true or false. Furthermore, the system generates a predicted probability associated with the classification, enhancing the analysis' transparency. The test cases that were aligned with the functional requirements were passed. The study resulted in a web-based system that provides users with a platform for verifying the credibility of content consumed on social media.

***Keywords****: Social media, Misinformation, fake news, Transformer model, Glove, Web-based*

# Table of Contents

[Declaration ii](#_Toc152517874)

[Abstract iii](#_Toc152517875)

[Table of Contents iv](#_Toc152517876)

[List of figures viii](#_Toc152517877)

[List of Tables ix](#_Toc152517878)

[List of Abbreviations x](#_Toc152517879)

[Acknowledgment xi](#_Toc152517880)

[Chapter 1: Introduction 1](#_Toc152517881)

[1.1 Background 1](#_Toc152517882)

[1.2 Problem Statement 2](#_Toc152517883)

[1.3 Aim 2](#_Toc152517884)

[1.4 Specific Objectives 2](#_Toc152517885)

[1.5 Research Questions 2](#_Toc152517886)

[1.6 Justification 3](#_Toc152517887)

[1.7 Scope and Limitations 3](#_Toc152517888)

[Chapter 2: Literature Review 4](#_Toc152517889)

[2.1 Introduction 4](#_Toc152517890)

[2.2 The State of Fake News and Misinformation on social media 4](#_Toc152517891)

[2.2.1 Challenges Associated with Fake news and Misinformation on social media. 5](#_Toc152517892)

[2.3 Related Works 6](#_Toc152517893)

[2.3.1 Fake news detection using a transformer based approach 6](#_Toc152517894)

[2.3.2 Fake news Detection in social media with BERT-based deep learning approach 9](#_Toc152517895)

[2.3.2 Fake news detection using a graph neural network based approach 11](#_Toc152517896)

[2.4 Gaps in related works 13](#_Toc152517897)

[2.5 Conceptual Framework 13](#_Toc152517898)

[Chapter 3: Methodology 15](#_Toc152517899)

[3.1 Introduction 15](#_Toc152517900)

[3.2 Methodology 15](#_Toc152517901)

[3.2.1 Requirements gathering and analysis. 16](#_Toc152517902)

[3.2.2 Quick Design 16](#_Toc152517903)

[3.2.3 Build Prototype 16](#_Toc152517904)

[3.2.4 User Evaluation 16](#_Toc152517905)

[3.2.5 Refine Prototype. 16](#_Toc152517906)

[3.2.6 Implementation and Maintain 17](#_Toc152517907)

[3.3 System Analysis 17](#_Toc152517908)

[3.3.1 Activity Diagram 17](#_Toc152517909)

[3.3.2 Entity Relationship Diagrams 18](#_Toc152517910)

[3.3.3 System Sequence Diagrams 18](#_Toc152517911)

[3.3.4 Use Case Diagram 18](#_Toc152517912)

[3.3.5 Class Diagram 19](#_Toc152517913)

[3.4 Deliverables 19](#_Toc152517914)

[3.4.1 System Proposal 19](#_Toc152517915)

[3.4.2 System Design Diagrams 19](#_Toc152517916)

[3.4.4 Pre-processed Data 19](#_Toc152517917)

[3.4.5 Model 20](#_Toc152517918)

[3.4.6 User Interface 20](#_Toc152517919)

[3.5 Tools and Techniques 20](#_Toc152517920)

[3.5.1 Scikit-Learn 20](#_Toc152517921)

[3.5.2 Django 20](#_Toc152517922)

[3.5.3 GitHub 21](#_Toc152517923)

[3.5.4 JupyterLab 21](#_Toc152517924)

[3.5.5 Ngrok 21](#_Toc152517925)

[3.5.6 Superbase 21](#_Toc152517926)

[Chapter 4: System Analysis and Design 22](#_Toc152517927)

[4.1 Introduction 22](#_Toc152517928)

[4.2 System Requirements 22](#_Toc152517929)

[4.2.1 Functional Requirements 22](#_Toc152517930)

[4.2.2 Non-Functional Requirements 22](#_Toc152517931)

[4.3 System Analysis Diagrams 24](#_Toc152517932)

[4.3.1 Use Case Diagram 24](#_Toc152517933)

[4.3.2 Sequence Diagram 26](#_Toc152517934)

[4.3.3 Entity Relationship Diagram 27](#_Toc152517935)

[4.3.4 Activity Diagram 28](#_Toc152517936)

[4.4 System Design Diagrams 30](#_Toc152517937)

[4.4.1 Class Diagram 30](#_Toc152517938)

[4.3.4 Database Scheme 31](#_Toc152517939)

[4.3.4 System Architecture 31](#_Toc152517940)

[4.3.4 Wireframes 32](#_Toc152517941)

[Chapter 5: System Implementation and Testing 34](#_Toc152517942)

[5.1 Introduction 34](#_Toc152517943)

[5.2 Description of Implementation Environment 34](#_Toc152517944)

[5.2.1 Hardware Specifications 34](#_Toc152517945)

[5.2.2 Software Specifications 34](#_Toc152517946)

[5.3 Description of the Dataset 35](#_Toc152517947)

[5.4 Description of Training 38](#_Toc152517948)

[5.4.1 Training and Analysis of the Model 38](#_Toc152517949)

[5.5 Description of Testing 40](#_Toc152517950)

[5.5.1 Testing Paradigm 42](#_Toc152517951)

[5.5.2 Testing Results 43](#_Toc152517952)

[Chapter 6: Conclusion, Recommendations and Future Works 46](#_Toc152517953)

[6.1 Conclusion 46](#_Toc152517954)

[6.2 Recommendations 46](#_Toc152517955)

[6.3 Future Works 46](#_Toc152517956)

[References 47](#_Toc152517957)

[Appendices 49](#_Toc152517958)

[Appendix 1:Timeline of Activities 49](#_Toc152517959)

[Appendix 2: Test Cases 50](#_Toc152517960)

[Appendix 3: GitHub Analytics and Link 51](#_Toc152517961)

[Appendix 4: User Manual 52](#_Toc152517962)

[Appendix 5: Plagiarism Report 54](#_Toc152517963)

[Appendix 6: Marking Guide 55](#_Toc152517964)

# List of figures

[Figure 2.1 BART model architecture 7](#_Toc152433616)

[Figure 2.2 Word clouds of news content 12](#_Toc152433617)

[Figure 2.3 Conceptual Framework 14](#_Toc152433618)

[Figure 4.1 Use Case Diagram 24](#_Toc152433619)

[Figure 4.2 Sequence Diagram 27](#_Toc152433620)

[Figure 4.3 Entity Relationship Diagram 28](#_Toc152433621)

[Figure 4.4 Activity Diagram 29](#_Toc152433622)

[Figure 4.5 Class Diagram 30](#_Toc152433623)

[Figure 4.6 Database Scheme 31](#_Toc152433624)

[Figure 4.7 System Architecture 32](#_Toc152433625)

[Figure 4.8 Login Page 32](#_Toc152433626)

[Figure 4.9 Registration Page 33](#_Toc152433627)

[Figure 4.10 Make A Prediction Page 33](#_Toc152433628)

[Figure 5. 1 Dataset 35](#_Toc152433629)

[Figure 5. 2 Distribution of category 36](#_Toc152433630)

[Figure 5. 3 Category count per article 36](#_Toc152433631)

[Figure 5. 4 Cleaned dataset 37](#_Toc152433632)

[Figure 5. 5 Model Summary 39](#_Toc152433633)

[Figure 5. 6 Accuracy of model 39](#_Toc152433634)

[Figure 5. 7 Accuracy graph 40](#_Toc152433635)

[Figure 5. 8 Confusion matrix 41](#_Toc152433636)

[Figure 5. 9 White box testing 42](#_Toc152433637)

[Figure 5. 10 Blackbox testing 43](#_Toc152433638)

# List of Tables

[Table 2.1 Representation of BART confusion matrix 8](#_Toc152433458)

[Table 2.2 Evaluation metrics table 8](#_Toc152433459)

[Table 2.3 Evaluation of BART algorithm using a confusion matrix 8](#_Toc152433460)

[Table 2.4 Attributes in the fake news dataset 10](#_Toc152433461)

[Table 2.5 Evaluation for CNN with BERT confusion matrix 10](#_Toc152433462)

[Table 2.6 Evaluation scores using graph neural networks 13](#_Toc152433463)

[Table 5. 1 Hardware Specifications 34](#_Toc152433464)

[Table 5. 2 Software Specifications 34](#_Toc152433465)

[Table 5. 3 Testing Results 43](#_Toc152433466)

# List of Abbreviations

ACC – Accuracy

AP – Average precision

AUC- Area Under a Curve

BART - Bidirectional and autoregressive transformer

BERT- Bidirectional encoder from transformers

CNN – Convolutional neural network

ERD – Entity Relationship Diagram

FN- False Negative

FP- False Positive

GNN- Graph Neural Networks

NLP- Natural language processing

OOAD- object-Oriented Analysis and Design

Prec – Precision

Rec- Recall

TN- True Negative

TP- True Positive

# Acknowledgment

I'd like to show my gratitude and appreciation to everyone who made it possible for me to finish this project. First and foremost, I want to thank the Almighty God for providing me with the power, knowledge, and understanding to complete this undertaking. My sincere gratitude goes to my supervisor, Mr. Steve Kayugira, for his essential instruction, patience, and support from the beginning of this project. His helpful counsel, oversight, and critical participation when needed aided me immensely during this endeavour. I’d also like to express my gratitude to Strathmore University's School of Computing and Engineering Sciences for providing me with the opportunity to work on this project. It aided me in conducting extensive research, and I learned a great deal as a result. Finally, I'd like to express my gratitude to my family and friends for their unwavering support and assistance throughout this effort. Their insightful suggestions and encouragement have been extremely beneficial at various stages of this undertaking.

# Chapter 1: Introduction

## Background

Social media platforms in recent years emerged as sources of information and news. These platforms enabled users to consume and distribute information on a large scale, resulting in information spreading quickly and reaching a wide audience. However, the openness of social media resulted in the spread of misinformation, which could cause significant socioeconomic and political repercussions (Allcot, 2017). Lewandowsky, (2020) stated that deceptive or inaccurate information on social media could influence public opinion, election outcomes, and individual health choices.

False information and fake news on social media present a variety of challenges. The vast amount of information exchanged on these platforms makes it tedious and time-consuming for users to carefully check the validity of each post shared. The rate at which information circulates exacerbates this as false and misleading information can spread quickly without being sufficiently fact-checked (Guess, 2019).

The reliance on human resources to conduct manual verification of social media posts became impracticable, time-consuming, and expensive as the amount and pace of information grew. As a result, the capacity to automate systems to perform efficient and effective textual content verification provided reliability (shu, 2017). Natural language processing (NLP) presented interesting options for dealing with problems created by inaccurate content on social media. It encompassed a variety of approaches and algorithms that enabled computers to read, comprehend, and generate human language (Jursafsky, 2020). By utilizing NLP techniques, researchers and developers could build automated systems that could analyze textual content, find trends, and extract valuable information (Potthast, 2018).

The issue of credibility verification had been investigated previously in NLP research using a variety of techniques. Through approaches such as sentimental analysis, whereby emotional tone and sentiment represented in text were examined, plagiarism detection, whereby similarities and potential instances of plagiarism or content manipulation, algorithms were used to compare textual content with known reliable sources of information (Horne, 2018). With the existence of fact-checking sites such as PolitiFact that carried out credibility verification through a manual process, data gathered from it could be used to deploy a model on the web that could analyze textual content from Twitter posts. Therefore, leveraging the strengths of web scraping and natural language processing techniques, the verification process could be enhanced.

## Problem Statement

With social media increasingly having served as people’s main source of news and information, establishing the validity of the material posted on these platforms was a major concern. According to (Vosoughi, 2018) and (Pennycook, 2019), there was a risk that people formed false beliefs and made decisions based on inaccurate information due to the absence of mechanisms for evaluating quickly and accurately textual content on social media.

The enormous amount of information published on social media platforms in real-time was too much for current fact-checking techniques, including those used by organizations such as PolitiFact (Guess, 2020) . This limitation made it difficult to spot and verify fake news and misinformation in time, thus quick dissemination influenced the public’s beliefs. To enable social media users to verify information quickly and accurately, it was crucial for an approach that was both effective and efficient at determining the reliability of information posted on Twitter.

## Aim

To develop web-based social media textual post verification system using natural language processing.

## Specific Objectives

1. To study the state of fake news and misinformation on social media
2. To identify challenges associated with fake news and misinformation on social media
3. To investigate existing approaches in the verification of fake news and misinformation
4. To design and develop web-based social media textual post verification system using natural language processing
5. To test the developed web-based social media textual post verification system

## 1.5 Research Questions

1. What is the state of misinformation and fake news on social media?
2. What are the challenges associated with fake news and misinformation on social media?
3. How have the existing works approached verification of fake news and misinformation?
4. How will the system be used in verification of textual posts on social media?
5. How will the proposed system be tested?

## Justification

The inundation of fake news and misinformation on social media platforms raised urgent concerns throughout the globe. With consequences being severe on a variety of areas, including polarization, the spread of fear and panic, poor health decisions, and political outcomes. By mitigating this issue, a textual post credibility verification system could foster a more educated society and rebuild the credibility of information from social media.

While valuable and effective, the fact-checking of information by organizations such as PolitiFact became time-consuming and resource-intensive. By undertaking natural language processing techniques and enabling real-time analysis of textual posts on social media sites, the proposed system aimed to streamline the credibility verification process and reduce the time and effort required for verification.

The proposed research fit with the expanding need for novel remedies to counter false information. For the challenges brought about by fake news and misinformation, there was a dire need for creative solutions that made use of natural language processing and machine learning capabilities, as social media continued to evolve and play a major role in information transmission.

## Scope and Limitations

The goal of the study was to use natural language processing to create a web-based system for textual post credibility verification for social media. The system mainly examined text that had been posted on social media sites, mainly Twitter, and offered credibility verification. Other media such as photographs and videos were not covered by the proposed research topic. The study was limited to textual posts due to the lack of a dataset that encompassed other forms of media such as photographs and videos.

## Chapter 2: Literature Review

## 2.1 Introduction

The proliferation of fake news and misinformation became a pressing concern, particularly in social media. Extensive research had been conducted to tackle this problem, especially in classification and detection. Techniques such as machine learning, natural language processing, and deep learning emerged as valuable tools when it came to detection and classification. This chapter aimed to look at the state of misinformation and fake news at the time and various approaches that had taken place to try and combat the problem.

## 2.2 The State of Fake News and Misinformation on social media

Fake news and misinformation became prevalent issues in the era of social media, posing significant challenges to the credibility and reliability of information shared on platforms like Twitter. With recent technological advancements, more individuals spent time online and on social media. About 58.4% of the world's population used social media, with a daily average time spent being 2 hours and 27 minutes. (Chaffey, D., 2022) hence social media users' high time spent resulted in them highly encountering and being exposed to misinformation and fake news on a regular basis.

The scope and magnitude of fake news and misinformation on social media platforms is mind blowing. According to (Allcot, 2017) misleading news items travel rapidly and reach a larger audience than factual news stories. Another study carried out by (Guess, 2019) discovered that one in every four Americans visited websites that produced fake news during the United States 2016 election. Furthermore, (Vosoughi, 2018) discovered that bogus news reports were 70% more likely to be retweeted as compared to accurate stories, while Menczer (2020) highlights that false information on Twitter spreads six times faster than genuine information. Research conducted by shao, (2018) demonstrates that false information when it comes to dissemination of health and medical treatments results to negative impacts on people’s well- being and decision-making processes.

The rapid spread of fake news and misinformation presents issues for both fact-checking organization and users. The quantity of disinformation appearing on social media platforms has risen drastically in recent years, according to the prominent fact checking organization PolitiFact (Guess, 2020). Moreover, recommendation systems and algorithms implemented on social media platforms play a significant role in amplifying fake news and misinformation. These algorithms filter posts based on users’ choices and behavior which results to creation of filter bubbles and echo chambers as a result more users are likely to be exposed to post and content that promotes their existing opinions, spreading disinformation and generating a polarized information environment (Guess, 2019).

According to Cook (2017) fake news and disinformation had implications that go beyond the transmission of misleading information as they resulted to potentially huge societal and political consequences. Misleading information sway public opinion, influenced election, and eroded the trust institutions (Lewandowsky, 2020). Misinformation had been proven in studies to cause changes in behavior such as vaccine hesitation and adoption of conspiracy theories (Pennycook, 2019). This emphasizes the importance of mitigating the issue of fake news and misinformation as the implications can be far-reaching for people and the society as whole.

### 2.2.1 Challenges Associated with Fake news and Misinformation on social media.

Despite efforts to solve the dissemination of fake news and misinformation, various challenges remained. One of the difficulties was that flagging procedures were ineffective. Although social media sites had flagging techniques in place to allow users to report fake news and misinformation, these procedures frequently fell short of properly identifying and preventing propagation, as a result, false information easily evaded detection and ended up spreading to the masses.

Another challenge arose in the manual fact-checking of content on social media platforms. Organizations such as PolitiFact have shown to be effective in refuting erroneous information; however, the process is time-consuming and resource-intensive (Guess, 2020). The huge volume of information published on social media daily made it difficult for fact-checkers to keep up with the rapid spread of false information, hindering their ability to effectively identify and debunk misleading content. Furthermore, the spread of fake news and misinformation sometimes spread at a fast rate, outdoing the fact-checking process, resulting in the widespread dissemination of inaccurate information. This led to misinformation reaching large audiences before it could be fully disapproved.

Moreover, the ever-changing nature of fake news and misinformation presented a hurdle. Fact-checking procedures had to constantly adapt to keep up with new strategies adopted by purveyors of fake news as mechanisms used to propagate misinformation evolved. The manipulation of textual posts on social media, such as the use of deceptive language, inaccurate headlines, and forged quotes, complicated the fact-checking process and made distinguishing between authentic and fake information more difficult (Lazer, 2018).

Additionally, social media platforms’ recommendation systems and algorithms posed an obstacle in countering fake news and misinformation. These algorithms were intended for prioritizing social media posts based on users’ habits and behavior, which frequently resulted in the amplification of contentious content, including incorrect information. (Guess, 2020)

This algorithmic bias could result in the spreading of misinformation and false news by exposing people to content that confirmed their previous opinions, increasing confirmation biases, and creating echo chambers.

## 2.3 Related Works

Some of the related works in fake news and misinformation classification include:

### 2.3.1 Fake news detection using a transformer based approach

This study by Shaina Raza and Cheng Ding (2022) proposed a transformer-based approach to classify news and social contexts on news into fake or real news. It implemented a modified bidirectional and autoregressive transformer (BART) architecture. The BART model ability to integrate text generation and comprehension in both bidirectional and autoregressive was used as bidirectional encoder from transformers (BERT) model exceled in discriminative tasks but is limited to generative tasks while GPT-2 is suited for generative tasks but falls behind when it comes to discriminative.

The modification of BART architecture was part of the methodology that was used, a comprehensive variety of elements from news articles and social contexts were added into the encoder part, employing a multi-head attention to balance the relevance of various of information. Second, forecasting of the next token by masking some of the input tokens and calculating the item that follows based on temporal position using the decoder to generate predictions based on prior text sequences and user actions. Finally, a linear transformation and SoftMax layer were used to generate the output i.e., the target label.

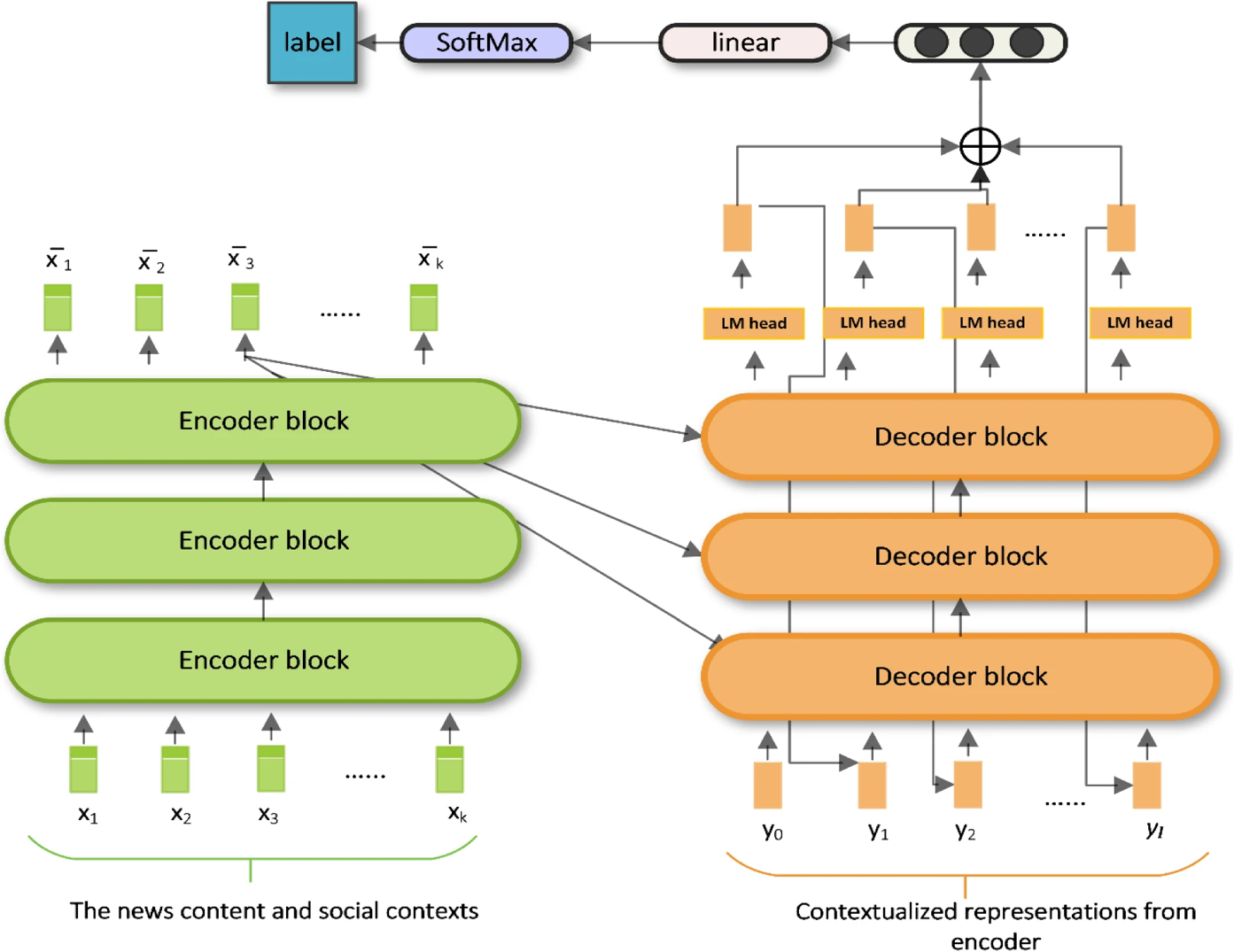


Figure 2.1 BART model architecture

The datasets used included target labels such as unreliable, mixed, reliable, true, satire, misleading content, imposter content, false and manipulated content. The first task carried out was to reduce the labels to real and fake to allow focus on binary classification. The datasets were then cleaned and pre-processed.

The model was evaluated using several key metrics. A confusion matrix was used to analyze the actual and expected classifications. True Positive (TP), True Negative (TN), False Positive (FP) ,False Negative (FN) values were represented in the matrix, offering insight into the models performance .

Table 2.1 Representation of BART confusion matrix

A picture containing text, screenshot, font, line

Description automatically generated

The True Positive (TP) represented number of false news stories detected as such. False Positive (FP) suggested the misclassification of authentic news stories as fake. True Negative (TN) indicated the correct identification of real news articles, whereas False Negative (FN) represented the misclassification of false news stories as real. Through the confusion matrix various evaluation metrics were calculated, including accuracy (ACC), precision (Prec), recall (Rec), F1-score (F1), area under the curve (AUC) and average precision (AP). These measures gave a thorough assessment of the model’s performance, considering proper classifications, trade-offs between true positives and false positives and precision and recall relationships. With the evaluation framework a comprehensive understanding of the model’s effectiveness in distinguishing between fake and real news was obtained.

Table 2.2 Evaluation metrics table

|  |  |
| --- | --- |
| Metrics | BART |
| False Positive Rate | 18 |
| F1 | 16 |
| Recall | 15 |
| Precision | 14 |
| Accuracy | 17 |

Table 2.3 Evaluation of BART algorithm using a confusion matrix

A picture containing text, screenshot, font, number

Description automatically generated

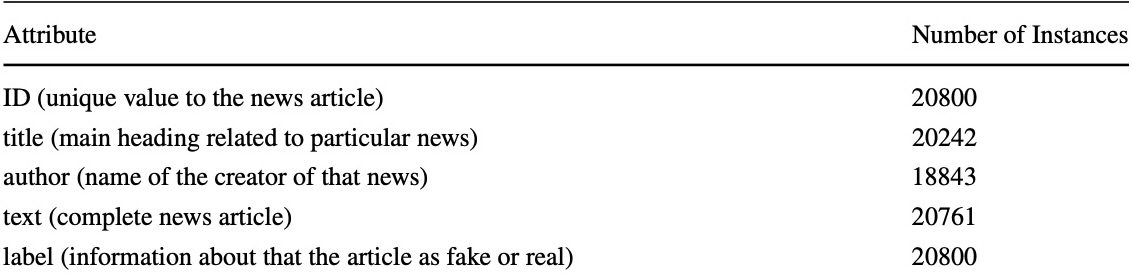
Based on Table 2.1, Table 2.2 and Table 2.3 , the accuracy of the model was determined to be 74.89% suggesting that more than 74% of the outcomes were right. The precision obtained was 72.40% showing the occurrence of few false positives in which actual news predicated be false. The recall value obtained was 77.68% indicating that there more true positives than false negatives. With that it was discovered that false negatives, in which false news were predicted to be true, were more destructive fake news detection. The obtained F1-score was 74.95% suggesting a good level of performance.

### 2.3.2 Fake news Detection in social media with BERT-based deep learning approach

This study by Rohit Kumar, Anurag Goswami, Pratik Narang (2021) proposed a BERT based deep learning approach to detect fake news in social media. Bidirectional encoder from transformers (BERT) was used as it was considered to offer a bidirectional training approach as well as offer improvements in classification performance with ability to capture semantic and long-distance decencies in sentences. The bidirectional encoder from transformers (BERT), with deep learning method based parallel block structure of the convolutional neural network (CNN) formed what was called FakeBERT which its aim was to be enhance the performance capabilities of the BERT model.

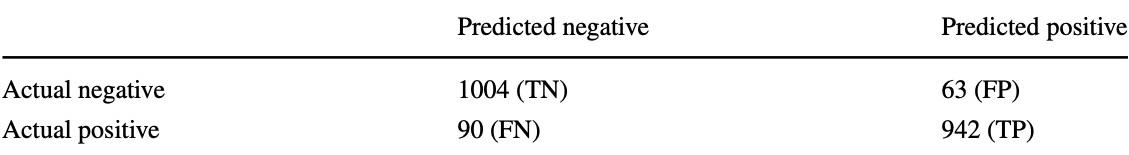
The dataset was collected during the 2016 US General election. It was made up of two files “train.csv ‘’ and “test.csv”. The “train.csv” file contains labelled examples of fake and real news, but the “test.csv” was a testing dataset that contains that dint contain labels. While the source wasn’t cited, its highlighted that the dataset captures the real-world aspect. Table 2.4 Shows the independent variables and dependant variable contained on the fake news dataset.

Table 2.4 Attributes in the fake news dataset



For data pre-processing data was truncated and segmented to fit within the length constraint.

Table 2.5 Evaluation for CNN with BERT confusion matrix



A confusion matrix was used to examine the CNN with BERT model evaluation results. True Negative (TN) had a value of 1004 in the confusion matrix, showing correct prediction of negative cases. The number of False Negative (FN) instances was 90, representing instances that were truly positive but were incorrectly labelled as negative. False Positive (FP) accounts for 63 cases when the outcome was negative but was wrongly projected as positive. True Positive (TP) has a rating of 942, indicating that positive instances were correctly predicted. The performance of the CNN with BERT model was evaluated using the confusion matrix in terms of its ability to reliably distinguish positive and negative instances.

The CNN with BERT model performed well in categorizing positive cases, according to the assessment criteria produced from the specified confusion matrix. With a recall value of 0.912, the model effectively detected a high proportion of genuine positive events in the previous evaluation. The precision score of 0.937 indicated that the model had a low rate of false positives, proving its ability to forecast positive events reliably. The high accuracy (ACC) of 0.927 indicated the model's general correctness across both positive and negative examples in the previous examination. Finally, the F1-score of 0.924 demonstrated a balanced performance in correctly detecting positive cases while avoiding false positives and false negatives. These evaluation metrics revealed, the effectiveness of CNN with BERT in accurately classifying instances and showed its reliability in text classification.

### 2.3.2 Fake news detection using a graph neural network based approach

The study by Pallabi Saitaki, Kshitij Gundale, Ankit Jain, Dev Jadeja, Harvi Patel and Mohendra Roy (2022) proposed a hybrid approach for fake news detection. Their hybrid approach intended to improve the accuracy to improve the accuracy of fake news identification by incorporating a graph neural network for news propagation analysis with bi-directional encoder representations from the transformer model for text characteristics.

The dataset was collected from popular data repository FakeNewsNet particularly from PolitiFact and GossipCop. PolitiFact concentrated on American politics and validated the truth of remarks made by various political figures, whereas GossipCop fact-checked celebrity reportage. Articles having ground truth labels assigned by independent journalists were included in the datasets. The data gathering method outlined in a prior work was used, which entailed extracting relevant information such as news content, tweets, retweets, and user profiles from tweet IDs connected with each news story. News stories that lacked text were eliminated. Several pre-processing stages were carried out on the obtained datasets. Figure 2.2 shows representations of both types of news that were provided.

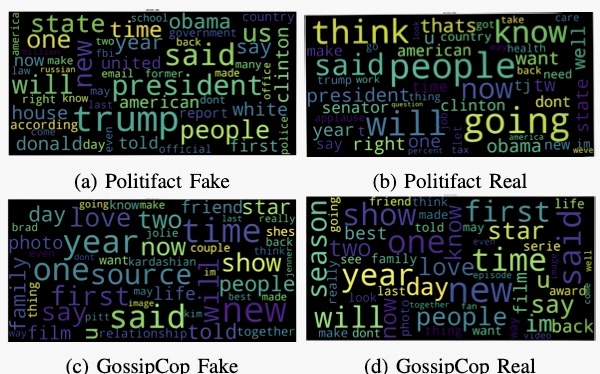
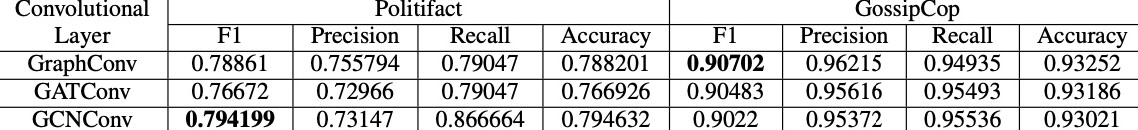


Figure 2.2 Word clouds of news content

The classification scores obtained using Graph Neural Networks (GNNs) on the PolitiFact and GossipCop datasets were the subject of the model evaluation. GraphConv and GCNConv were among the GNN models trained and evaluated. GCNConv achieved the best F1-score on the PolitiFact dataset, hitting 0.79. The GraphConv model received an F1-score of 0.907 on GossipCop. The research involved experimenting with different numbers of layers and embedding sizes for convolutional models, and it was discovered that employing 4 layers and 64 dimensions offered equivalent results while also reducing training time. The evaluation emphasized GNNs' ability to capture graph-level characteristics and their promise for boosting fake news classification performance. Table 2.6 shows the classification scores that were achieved using graph neural networks.

Table 2.6 Evaluation scores using graph neural networks



The combination of GNN-based features and text yielded superior results, showcasing the importance of using a multimodal approach for accurate detection of fake news.

## 2.4 Gaps in related works

The existing works extensively talked about the creation of models and algorithms for false news identification, but one key gap remained, and that was the development of a friendly user interface to access the veracity of information. While the existing research studies made significant advances in identifying and countering misinformation, there was still a need to transform this knowledge into practical tools that people could easily access and use.

Additionally, in the existing research works, it's important to note that some of the studies did not comprehensively conduct comparative analysis. While these research works contributed to the advancement of the field, the lack of a robust comparative analysis limited the effectiveness, as progress on how the research was progressing became unclear without benchmarks and comparison.

Given the rising importance of social media as a key source of news and information, it was critical to fill these gaps to demonstrate the legitimacy of content provided on these platforms. There was a risk of people establishing incorrect beliefs and basing their decisions on inaccurate information if methods for quickly and accurately analyzing textual content were not in place. Creating an effective and efficient method for determining the authenticity for platforms such as Twitter was critical for enabling social media users to check information in a timely and accurate manner.

## 2.5 Conceptual Framework

Data was web scraped from PolitiFact as it was the primary source of data. The application then fetched the data and stored it in a database. The data underwent pre-processing, training, and fine-tuning. Afterward, it was sent to the model, whereby classification was done based on the users' inputs, and the classification algorithm classified the input. The category of the input given was then sent back to the application to be displayed on the user interface.

A screenshot of a computer

Description automatically generated with low confidence

Figure 2.3 Conceptual Framework

# Chapter 3: Methodology

## 3.1 Introduction

This chapter outlined the method used to generate the proposed solution. It emphasized the system development methodology, system analysis and design approaches, and system development tools and procedures. It also addressed the proposed solution deliverables. For development, the system employed an object-oriented analysis and design methodology that was more data focused. OOAD was recommended because it allowed a problem to be divided into smaller sections that could be modified separately without affecting the surrounding components.

## 3.2 Methodology

The prototyping methodology was used in the development of the system. This methodology concurrently executed the analysis, design, and implementation phases to swiftly construct a simplified version of the proposed system and get reviews and feedback from users. The prototype of the system was a rapid version of the system that offered limited features. As shown in Figure 3. (Martin, 2020) , based on the feedback received, the developers subsequently engaged in reanalysis, redesign, and reimplementation of a second prototype, rectifying any flaws, and incorporating additional features. The iterative cycle continued until the users reached a consensus that the prototype delivered sufficient functionality to be implemented (Pressman, 2020). The methodology will be utilized due to its inherent flexibility in design, enabling easy modification and adjustments according to the users or developers’ preferences and needs. It also allows for the convenient identification of missing functionalities and early detection of errors, resulting in significant savings in effort and cost.



Figure 3.1 Prototyping methodology

### 3.2.1 Requirements gathering and analysis.

The prototyping model commenced with requirements analysis, where software requirements were defined as descriptions of the desired features and functionalities of the target system. These requirements specifically identify the functionalities that the system needs to possess to meet user’s satisfaction. (Bahill, 2019).

The requirements for the proposed system were gathered from Secondary data web scrapped from PolitiFact and hosted on Kaggle. The dataset served as the training data for the model utilized in the proposed system.

### 3.2.2 Quick Design

During the second stage, a preliminary or rapid system design was developed once the requirements were established. The design was not comprehensive but offered a concise representation of the system, focusing on the essential aspects that were visible to the user. The purpose of this stage was to provide the user with a high-level understanding of the system, creating a quick design in the development of the prototype that the user could test and give feedback.

### 3.2.3 Build Prototype

During this stage, a preliminary prototype was designed using the information gathered from quick design. This prototype typically served as a simplified version of the system, provided an approximation of key features and functionalities of the final product. It served as a functional model of the desired system.

### 3.2.4 User Evaluation

The users conducted a comprehensive evaluation of the first prototype, carefully assessing its strengths and weaknesses. They identify elements that needed to be added or removed based on their feedback. The developer collected and analysed the feedback from users to gain valuable insights.

### 3.2.5 Refine Prototype.

Following the user evaluation, if the user expressed dissatisfaction, the prototype was refined based on the requirements, user feedback, and suggestions. The refined prototype was then evaluated by the user, like the previous iteration. This iterative process continued as the prototype was adjusted to meet the user’s needs, allowing the developer to gain a deeper understanding of the necessary changes. Once the user was satisfied with the developed prototype, a final system was constructed based on the final prototype.

### 3.2.6 Implementation and Maintain

Once all the requirements were fully met and the user was entirely satisfied, the final product was designed and developed, considering the final prototype. The system then underwent rigorous testing, and if deemed ready, it was deployed into production. After implementation, the system went through routine maintenance to minimize downtime and prevent significant failures on a large scale.

## 3.3 System Analysis

System analysis involved the decomposition of a system into constituent parts, aiming to comprehend the function, nature, and interdependencies of those parts (Jeffrey Hoffer, 2021). Its purpose was to examine a system or its components to identify objectives. In the case of the proposed system, object-Oriented Analysis and Design (OOAD) were utilized, which employed several tools and system analysis. These tools include activity diagrams, entity relationship diagrams, system sequence diagrams, use case diagrams and class diagram.

### 3.3.1 Activity Diagram

An activity diagram provided a graphical representation of the flow or sequence of actions are performed to complete an activity or process (Brahma Dathan, 2016). This diagram showcased the workflow of a system, starting from a designated start point and leading to a finish point, while also outlining the various decision paths that could be taken as events occurred.

In the context of the proposed system, an activity diagram was utilized to describe the sequential flow of activities in a more detailed manner, providing visualization of use cases at a granular level. Additionally, it was used to visualize the sequence of activities and depict workflows within the system.

### 3.3.2 Entity Relationship Diagrams

An entity relationship diagram (ERD) was a diagrammatic representation that showcased the relationship between entities within a system. Entities could encompass people, objects, concepts, or events for which data could be stored (John Satzinger, 2015).

The ERD played a crucial role in identifying entities that were present in the proposed system and how they related to the objects and functions within the system, as well as to each other. It aided in the design and construction of the system’s database since ERDs could be easily translated into relational tables. This diagram facilitated the understanding of data organization and the establishment of relationships between entities, contributing to the effective implementation of the system’s databases.

### 3.3.3 System Sequence Diagrams

A system sequence diagram (SSD) was a specific type of sequence diagram that focused on illustrating input and output events. It depicted the sequence of use cases and defined their order, along with events taking place inside the system. The SSD tracked how function and use cases were performed within a system.

In the context of the proposed solution, a system sequence diagram was employed to model the interaction of the system with events. It illustrated the events generated by actors, their proper flow, and the potential connections and relationships between events. The diagram helped visualize and understand the sequence of events within the system and ensured that they were correctly executed and interconnected.

### 3.3.4 Use Case Diagram

A use case diagram provided a high-level overview of the interactions between users who were the actors and the system itself. It showed how different actors (users or external systems) interacted with the system through various use cases. The diagram helped in understanding the roles of different users and how they engaged with the system to achieve their goals or perform specific actions. It was an essential tool for visualizing and communicating the overall structure and functionality of a system in a clear and straightforward manner. (Dinesh Batra, 2019).

### 3.3.5 Class Diagram

A class diagram provided a comprehensive depiction of the object types present in a system and the static relationships that existed among them. It showcased the methods, attributes, relationships, and operations of objects as well as any constraints imposed on the system. (Pressman, 2020).

In the proposed system, a class diagram was utilized to visually represent the various types of objects in the system and illustrate the relationships that existed between them. Furthermore, the class diagram aided in the construction of executable and accurate code for the model application. It served as a valuable tool for designing and implementing the system by providing a clear representation of the object structure and their interactions.

## 3.4 Deliverables

System deliverables were the services or outputs that were expected to be supplied at various phases of system development as well as at the end of a project. These deliverables were critical in keeping projects on schedule and ensuring efficient time and resource allocation.

### 3.4.1 System Proposal

A system proposal is a document that explained the goals and needs of a project and sought approval prior to its formal start. It provided critical information regarding the value of the system, associated risks, validation requirements, and implementation scope.

### 3.4.2 System Design Diagrams

System design diagrams are visual representations of a system’s structure and components. They offered a high-level overview of the system’s architecture, enabling an understanding of how the various components interacted and worked together to achieve the set system's objectives.

### 3.4.4 Pre-processed Data

Pre-processed data was data that underwent a series of transformations to make it suitable for analysis or training of a model. This provided high-quality, usable, and effective data that would be used for training the model.

### 3.4.5 Model

A model refers to a mathematical or computational depiction of a real-world system or problem. It was developed using an algorithm that captured the patterns and correlations of the pre-processed data, thereby undergoing training.

### 3.4.6 User Interface

A user interface refers to a graphical interface that through it users interacts with a software application, website or any digital system. The model was integrated with a user interface enabling users to interact with it.

## 3.5 Tools and Techniques

The tools and techniques that were used in developing the proposed system included:

### 3.5.1 Scikit-Learn

Scikit- learn, often known as Sklearn, is an open- source machine learning framework that provide developers with a complete collection of tools and functions. It provided a user-friendly and legible syntax, making it easier to properly employ programming resources. (Géron, 2019) Sklearn was used in the suggested solution because of its wide collection of pre-built functions and advanced procedures that simplify the process of generating multiple models. Sklearn is well-known for its versatility and extensive range of machine learning algorithms, which enable quick model creation, evaluation, and fine tuning for the specified categorization task.

### 3.5.2 Django

Django is a high-level web framework for Python renowned for its simplicity and efficiency in web application development. Built on the principles of readability and the "Don't Repeat Yourself" (DRY) philosophy, Django accelerates the creation of web applications by minimizing redundancy and promoting code reuse. Following the Model-View-Template (MVT) pattern, Django offers a structured approach to organizing code, enhancing maintainability. Its built-in features, including an Object-Relational Mapping (ORM) system and an administrative interface, contribute to swift development and management of web applications. (Géron, 2019).

### 3.5.3 GitHub

GitHub is a web-based platform and cloud service that assist developers in storing, monitoring, and tracking changes to their codebase. (Mudholkar, 2017). The proposed solution code was pushed to GitHub to ensure effective version control. Using GitHub UI, developers can easily track changes and manage file revisions, making it a vital throughout the development process of the proposed solution.

### 3.5.4 JupyterLab

JupyterLab is a popular open-source web based integrated development environment (IDE) for data analysis, machine learning, and scientific research. It supports a variety of programming languages and has an easy-to-use interface for creating and sharing interactive notes. JupyterLab wide ecosystem and plugins enabled users to personalize and enhance their working environment to meet their individual requirements. (Jupyter Team, 2021).

### 3.5.5 Ngrok

Ngrok proves invaluable in deploying model APIs by simplifying the process of exposing locally hosted APIs to the internet (Ngrok team, 2023) . This tool enables developers to securely share and test their model APIs during the development phase. By creating a secure tunnel from a public endpoint to the locally running API server, Ngrok allows for easy access and interaction with the API from external systems or stakeholders. Its support for both HTTP and HTTPS ensures secure communication, and features like custom subdomains enhance the user experience when sharing model APIs. Ngrok's straightforward setup, coupled with authentication and access control features, streamlines the deployment of model APIs, making it a practical choice for developers aiming to showcase and test their machine learning models in real-world scenarios.

### 3.5.6 Superbase

Superbase is an open-source Firebase alternative that provides a backend as a service (BaaS) platform for building web and mobile applications. It offers a range of features, including a Postgres database, authentication, instant APIs, edge functions, real-time subscriptions, storage, and vector embeddings. (supberbase , 2023)

# Chapter 4: System Analysis and Design

## 4.1 Introduction

This chapter outlines how the system functional and non-functional requirements of the system and the different diagrams showing how the system will be designed and analyzed.

## 4.2 System Requirements

The functionalities that a system needs to meet user needs are identified by system requirements. They lower implementation costs and make it possible to suit user needs very effectively. The following are a few of the system requirements reviewed for the project:

### 4.2.1 Functional Requirements

Functional requirements explain the behaviour of the system under specified conditions and are the features that must be implemented for a user to be able to complete their tasks. They include:

1. User interface module - This module serves as the interface through which users engage with the system. Its purpose is to streamline user-system interactions, ensuring a user-friendly experience that demands minimal effort to achieve desired outcomes.
2. Predict feature- This feature enables users to complete a form, which is subsequently utilized by the classifier to generate predictions.
3. Database – The user details are stored here together with the predictions made for statements.
4. Classifier module- This module is fed with the statements that have passed by user to be verified as true or false together with predicted probability threshold

### 

### 4.2.2 Non-Functional Requirements

Non-functional requirements are a set of specifications that specify the system's operation capabilities and limits, with the goal of improving its functionality. They define a system's quality qualities and describe how it functions. Although they are not required, they are necessary for validating a system's performance.

The system non-functional requirements of the system include:

1. System reliability- Denotes the likelihood that the programme will operate without interruption for a particular amount of time. The system's reliability is assured through preventative maintenance, in which the hardware and software are subjected to regular and routine maintenance to keep them functioning and avoid unscheduled downtime due to unexpected failure.
2. System security - Ensures that the software and its data are safe from unauthorised access. Any user information in the system is encrypted, and the database is constantly backed up.
3. System usability - Describes how difficult it will be for a user to learn and utilise the system. The system's user interface is user-friendly, allowing the user to readily comprehend and navigate the system. Because the interface is simple, the user can do most tasks without assistance.
4. System performance – This is a quality attribute that describes the system's responsiveness to various user interactions with it. Poor performance results in a negative user experience. For optimal performance, the web pages in the system should load in less than five seconds and have an efficient throughput and reaction time.
5. System availability – This is determined by how long the system's features and services are available for use across all operations. The system's availability is ensured through shortening recovery times in the event of a failure. This includes debugging issues, replacing components as needed, and restarting the database service. The system is also fault-tolerant, which means that it can continue to operate without interruption even if one or more of its components fail. The information database is constantly backed up so that if the primary database fails, operations can be diverted to the backup database.

## 4.3 System Analysis Diagrams

#### The diagrams used for system analysis include:

### 4.3.1 Use Case Diagram

A use case diagram shows how various actors interact with the system.

Figure 4.1 shows the social media user, the administrator, and the tasks that each of the actors can perform. The user first registers and then logs in before making predictions by filling out a form that captures statements and their properties. Following that, a verdict for each statement is displayed. Users can also generate reports that display the verdict for their submitted statements. The administrator, on the other hand, gets access to a dashboard from which they manage users and view various statements submitted by users through reports generation.

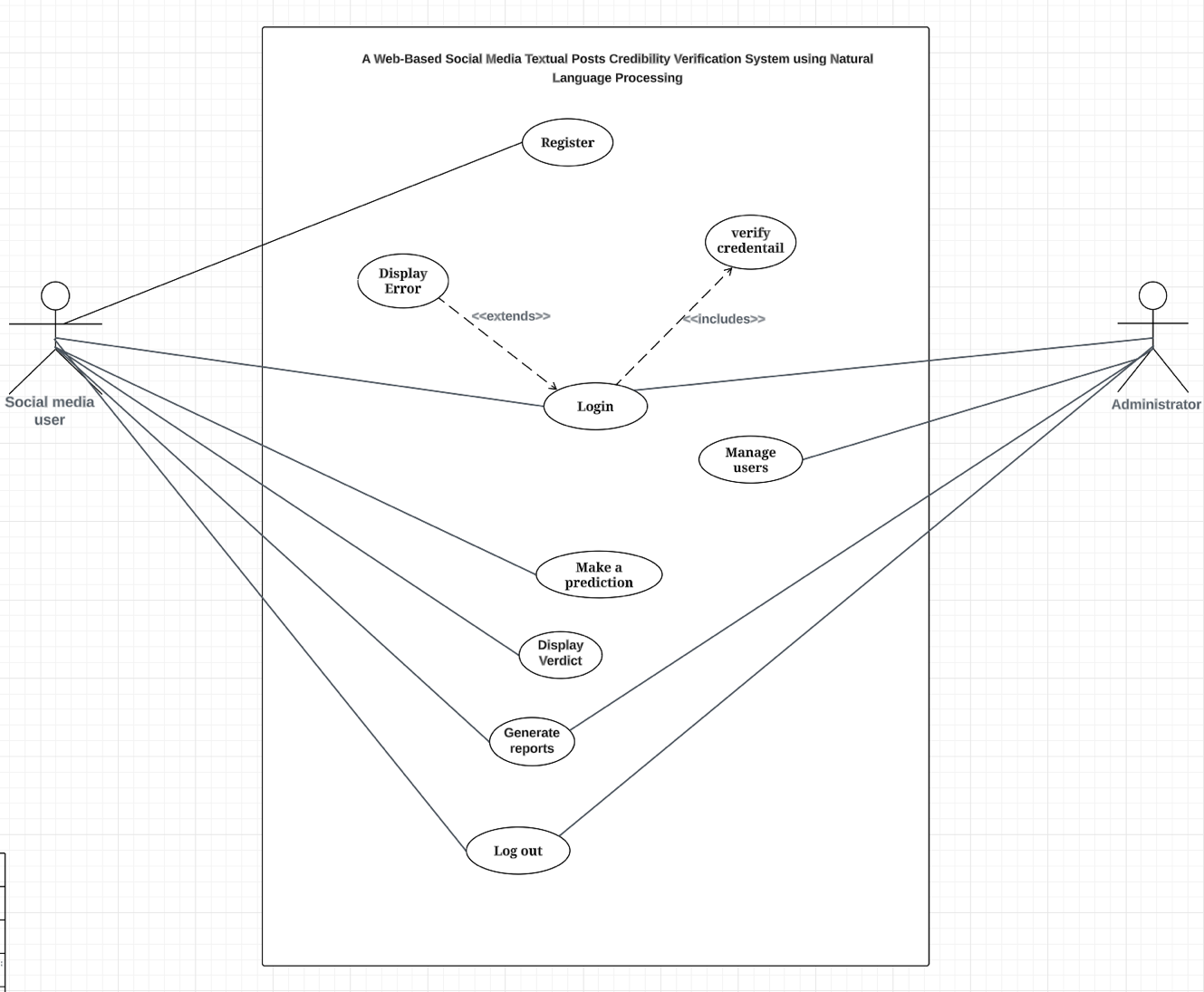


Figure 4.1 Use Case Diagram

Use Case Scenario one: Initiate Credibility Verification

|  |  |
| --- | --- |
| Use case | Initiate Credibility Verification |
| Actor | user |
| Pre-condition: | Be logged into the system All the data fields must have input |
| Post-condition: | The statements are categorized into one of 6 categories: true, mostly true, half true, mostly false, false, and pants on fire |
| Alternate condition/action: | All fields must have data otherwise the credibility verfication will not be initiated when the button is pressed pressed. |

Use Case Scenario Two: Input Values

|  |  |
| --- | --- |
| Use case | Input Values |
| Actor | user |
| Pre-condition | -Be logged into the system |
| Post-Condition | All the data fields should have a relevant value input before moving onto initiation of credibilty Verfication |
| Alternate condition/action | All fields must have data otherwise credibilty Verfication will not be initiated when the button is pressed |

Use Case Scenario Three: View verdict.

|  |  |
| --- | --- |
| Use case | View verdict |
| Actor | user |
| Pre-condition | Be logged into the system  A credibilty verdict must be present |
| Post-Condition | The result shoud be displayed |
| Alternate condition/action | The system should prompt the user to input values for the credibilty verdict to be given |

### 4.3.2 Sequence Diagram

#### A sequence diagram shows the interaction between objects in a sequential order.

They describe how and in what order the objects in a system work together to achieve the goal of the system. The social media user first fills a from that initiates a classification, the form's details are subsequently saved in the database and submitted to the classifier for classification. Following classification, a verdict is generated and displayed to the user, as well as being saved in the database.

A diagram of a diagram

Description automatically generated

Figure 5.2 Sequence Diagram

### 4.3.3 Entity Relationship Diagram

An Entity Relationship Diagram (ERD) is a graphical representation of the many entities in a system and the relationships that exist between them. Figure 4.4 shows the entities in the system how they relate to each other and their interaction. A user fills one or many forms that have one statement that are the classified by a model, each statement receives a one verdict.

A diagram of a diagram

Description automatically generated

Figure 6.3 Entity Relationship Diagram

### 4.3.4 Activity Diagram

An activity diagram depicts the flow of control in a system and refers to the steps involved in executing a use case. They are used to represent the sequential and concurrent actions that occur in the system. They aid in the visual representation of system workflows. In figure 4.3 we see the flow of activities in the system. A user first as to login and then fills a form that is used to make a prediction the form details are stored in the database then they are processed by a classifying model, the model then gives a verdict that concurrently its stored in a database.

A diagram of a process

Description automatically generated

Figure 7.4 Activity Diagram

## 4.4 System Design Diagrams

The diagrams used for system design include:

### 4.4.1 Class Diagram

A class diagram is a sort of static diagram that depicts the structure of a system by displaying the system's classes, characteristics, processes, and object relationships. Figure 4.5 shows the classes in the system which are the social media user, the form, the statement, the classifier and verdict. The characteristics and processes of each class are shown together with the relationships between classes.

A diagram of a computer

Description automatically generated

Figure 8.5 Class Diagram

### 4.3.4 Database Scheme

A database schema is the skeleton structure that represents the database's logical perspective. It outlines how the data is organised and how it may be related to other tables or data models.

As shown in figure 4.6 when the user registers the details are stored on the social\_media\_users table which are used for logging in. The form table was used to take account of the forms being filled and acts as a link for all the other tables as it contains the primary key for the other tables as foreign keys. Once a form was filled the details of the form are then stored in the statement tables and once a verdict was carried out for a statement the verdict table takes account of this.

A diagram of a data flow

Description automatically generated

Figure 9.6 Database Scheme

### 4.3.4 System Architecture

A system architecture diagram is an abstract representation of the component architecture of a system. It provides a quick overview of the system's component architecture to aid in component-component interactions and system functionality. It depicts the system's key operations as well as the relationships between the various system components. The figure 4.7 shows the user can make a request to access service from the system by send a request from the web browser after being connected to the internet. The web browser then requests the web server for the request via the internet from which the request is taken to the database server that responds accordingly to the request made.

A diagram of a cloud

Description automatically generated

Figure 10.7 System Architecture

### 4.3.4 Wireframes

Wireframes give a visual guide of how the web application will look like. Figure 4.8 shows the login page that a user inputs their username and password.

A screenshot of a login screen

Description automatically generated

Figure 11.8 Login Page

Figure 4.9 shows a registration form that a user filles to be able to login into the system to make a prediction.

A screenshot of a login form

Description automatically generated

Figure 12.9 Registration Page

Figure 4.9 Shows make a prediction page where the user fills a form and then does a prediction and

A screenshot of a computer screen

Description automatically generated

Figure 13.10 Make A Prediction Page

# Chapter 5: System Implementation and Testing

## 5.1 Introduction

This chapter discusses the system's implementation process, including how the model was trained and assessed, the dataset's description, and the implementation environment's description. Additionally discussed is the system testing procedure, along with an explanation of the test findings.

## 5.2 Description of Implementation Environment

This outlines the necessary hardware and software for the system to be implemented and run properly.

### 5.2.1 Hardware Specifications

Table 5. 1 Hardware Specifications

|  |  |  |
| --- | --- | --- |
| Component | Minimum | Recommended |
| Processor | 1.9 gigahertz (GHz) x86- or x64-bit dual core processor with SSE2 instruction set | 3.3 gigahertz (GHz) or faster 64-bit dual core processor with SSE2 instruction set |
| Hard disk storage | 128Gb | 256-GB RAM or more |
| RAM | 4gb | 8 GB and above |

### 5.2.2 Software Specifications

Table 5. 2 Software Specifications

|  |  |
| --- | --- |
| Software name | Description and Justification |
| Web Browser | An up-to-date web browser like recent versions of Arc ,Google Chrome, Mozilla Firefox, Apple Safari, Microsoft Edge and iOS and Android mobile browsers. |
| Operating system | Any Windows 7 or above and any Mac OS 10.8 or newer. |

# 5.3 Description of the Dataset

The dataset that was used to carry out training of the model was gotten from Kaggle.

The data was web scrapped from a site known as PolitiFact which is fact checking site for information and then added as part of Kaggle huge dataset library.

The data was downloaded from Kaggle in zipped folder. It was in csv format it contained over 40000 articles, it included and id column for each row which is unique to that specific row, a title column to capture the various articles headlines, text column to capture more details about the article, subject’s column to capture the domains that each article falls under, the date section to capture when the article was written , category to capture if the article is either true or false where by 1-true and 0-false.

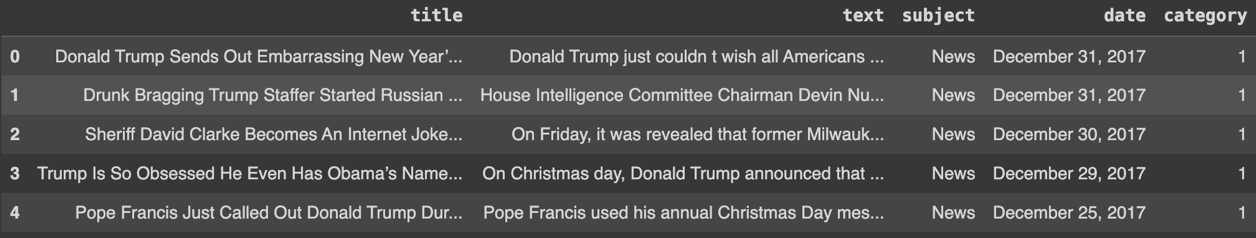


Figure 5. 1 Dataset

The dataset had no missing values and no duplicates. The distribution of category proved that there was no imbalance, and this can be seen through the following distributions:

A graph showing a number of bars

Description automatically generated with medium confidence

Figure 5. 2 Distribution of category

This above distribution shows the category of the article headlines as either true or false as per the chart we can see that there’s a good balance proving that the dataset is balanced.

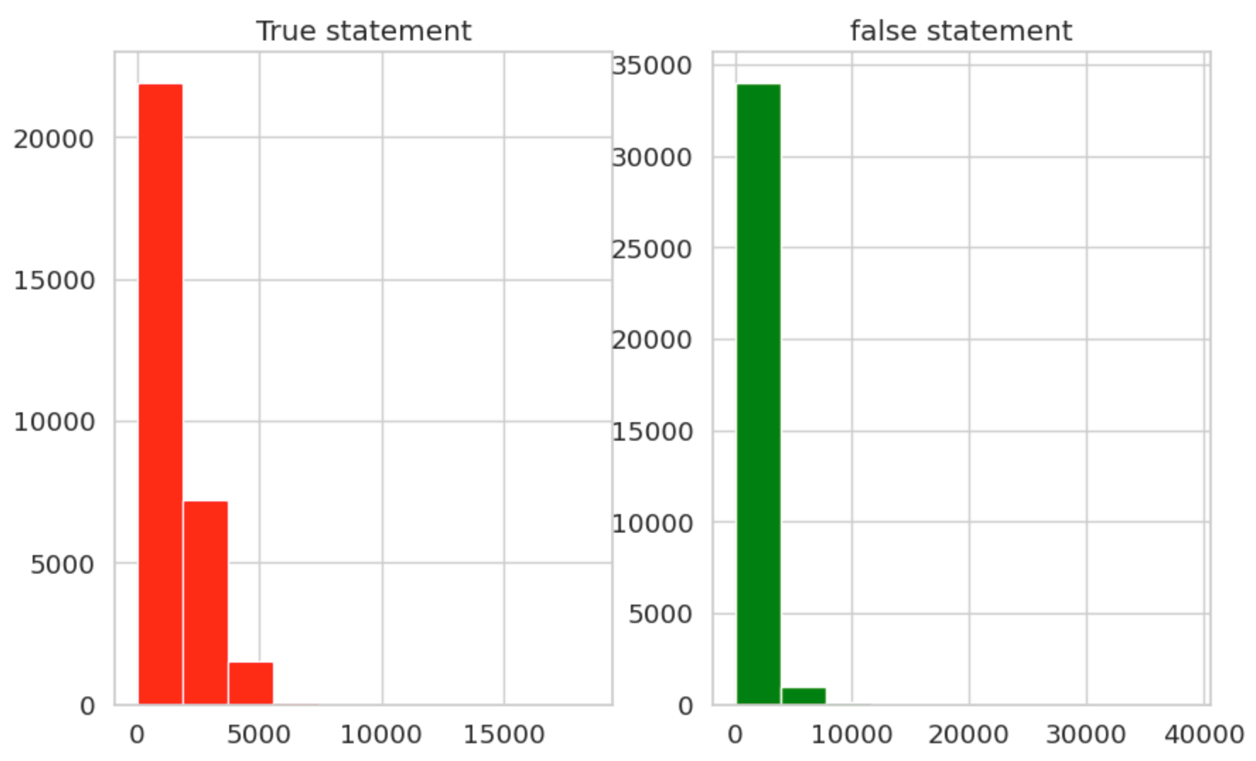


Figure 5. 3 Category count per article

The above distribution shows around 23000 are articles are true while around 34000 are false.

After caring out exploratory data analysis I had to now carry out feature engineering were

by I dropped and renamed columns, I renamed column text to statements category to verdict and dropped columns subject, date, title. This is how my dataset looked as per the figure below after carrying put feature engineering.



Figure 5. 4 Cleaned dataset

Data cleaning was performed to improve the quality of the dataset’s ‘statement' column. The NLTK library was used to remove common English stopwords during this extensive operation. Additionally, HTML tags were removed using BeautifulSoup, and regular expressions were used to remove superfluous text preceding the first phrase. The removal of terms within brackets, dashes and unnecessary symbols helped to refine the dataset even further. The combination of these cleaning stages into a denoising method produced a more polished text dataset, with the ‘statement' column changed to strings for consistency. The NLTK library, BeautifulSoup, and regular expressions performed critical roles in this data refining, jointly optimising the text data for more effective analysis.

## 5.4 Description of Training

This part will discuss how training was done, all the necessary imports and how the model was trained.

### 5.4.1 Training and Analysis of the Model

A systematic strategy was used to train and analyse a natural language processing model using the TensorFlow and Keras packages. The preparation of the data began with partitioning it into training and testing sets, with the ‘statement' column serving as input features and the ‘verdict' column acting as the target variable. To support consistent and fast model training, a maximum feature count of 20,000 and a sequence length of 300 were defined.

To ensure uniform input lengths, text data was tokenized, which is the process of translating words into number sequences. In addition, a denoising function was used to improve the quality of the text data in the testing dataset, making it more suitable for subsequent analysis. Glove, a pre-trained word embedding model, was used to represent words in a continuous vector space, allowing for a more detailed comprehension of semantic links within the text.

The neural network model's architecture was meticulously constructed. It had an embedding layer, an LSTM layer for sequence processing, and dense layers with dropout and L2 regularisation to reduce overfitting. The model was built with the Adam optimizer, with a learning rate of 0.01 and binary cross entropy as the loss function. During the training phase, the model was fed tokenized and padded training data, and its performance was measured using the validation set (test data). Callbacks, such as learning rate decrease and early halting, were used to improve training efficiency and minimise overfitting.

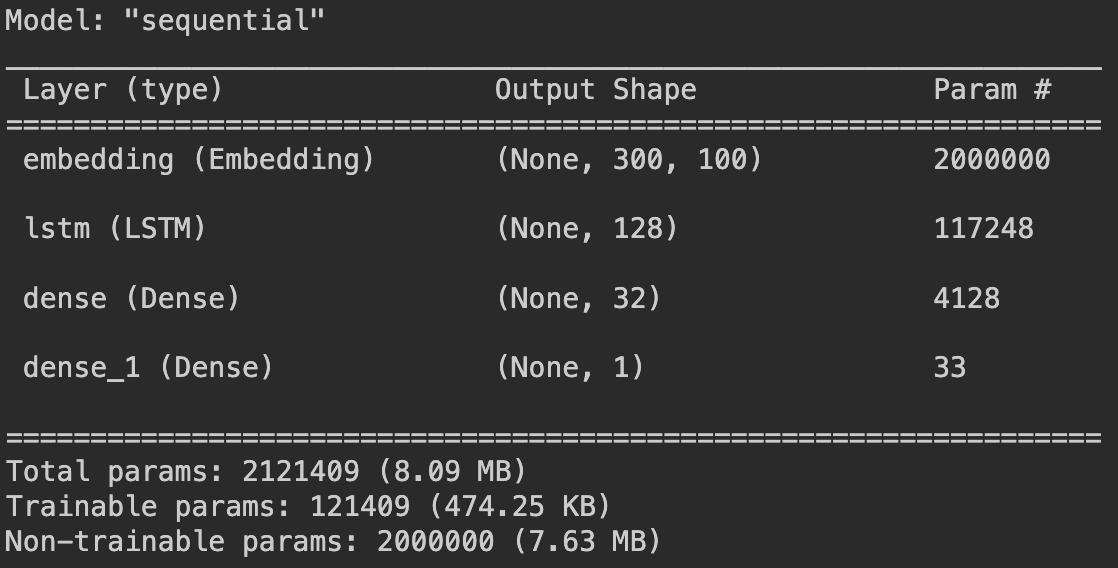


Figure 5. 5 Model Summary

The model summary gave an in-depth analysis of its design, including the number of parameters and layers involved. This comprehensive approach to model training, which included careful data preparation, the use of pre-trained embeddings, and the implementation of a well-designed neural network, aided in the building of a successful natural language processing model for the given task.

The model's performance on both the training and testing datasets was evaluated and found to be promising. Following a defined number of epochs for training, the accuracy on the training data was found to be 86.81%, demonstrating the model's proficiency in learning from the provided dataset. Similarly, the model's accuracy on testing data was 86.53%, demonstrating that it generalises effectively to previously unseen data.

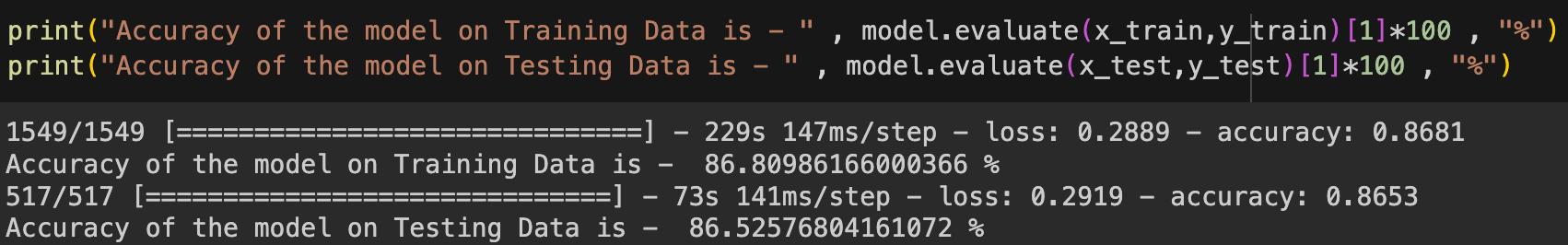


Figure 5. 6 Accuracy of model

A dual-plot graph was created to provide a visual depiction of the model's learning processes. The left plot depicts the trends in accuracy over epochs, with the green curve representing training set accuracy and the red curve representing testing set accuracy. This graphical representation depicts the model's convergence and stability during training. The right figure shows the equivalent loss values, with a decreased trend suggesting that the model's predictive performance is improving.

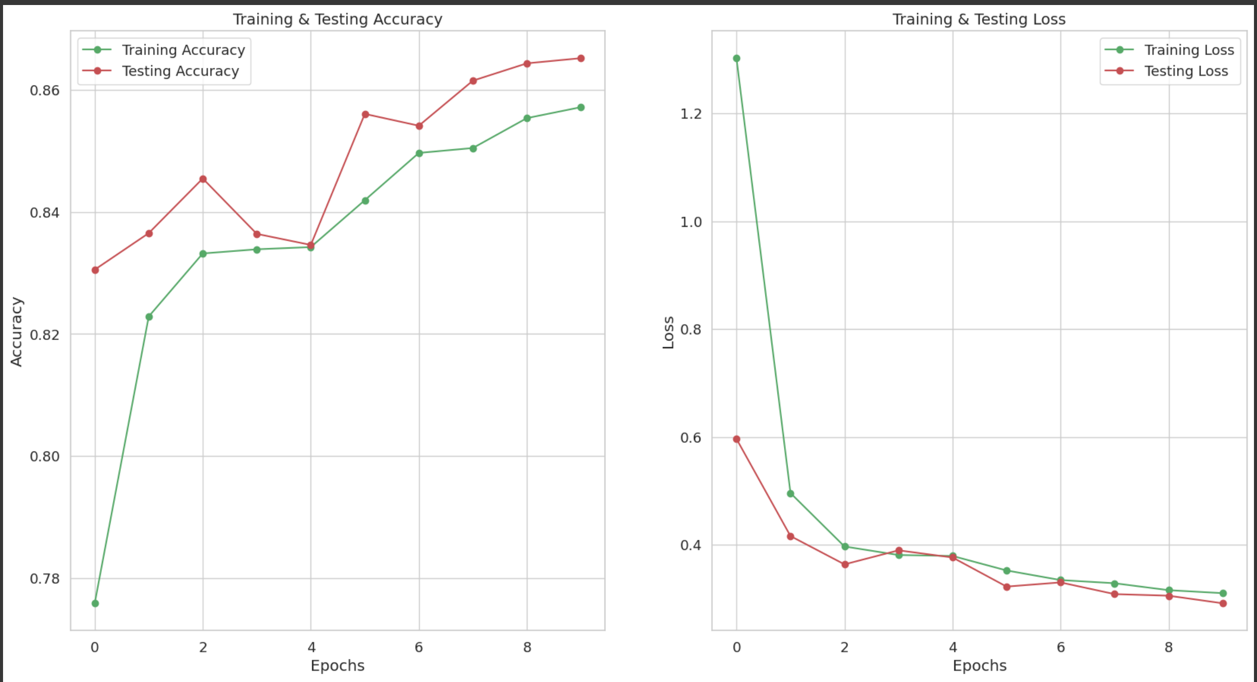


Figure 5. 7 Accuracy graph

Examining the accuracy graphs demonstrates that the model learns from the training data efficiently, achieving a consistent accuracy level. Notably, the testing accuracy roughly matches the training accuracy, indicating that the model can generalise well to new data. This observation is supported by the loss graphs, which show a constant drop in both training and testing losses. This reduction indicates that the model's predictive performance has steadily improved over the epochs.

## 5.5 Description of Testing

The trained model was applied to the test dataset during the testing phase to evaluate its prediction capabilities. Using the model, predicted probabilities were determined for each sample. Predict function. Following that, binary classifications were determined using a 0.5 probability threshold. The resulting matrix represented the first five samples, with a value of 1 indicating a forecast of 'True,' and a value of 0 indicating a prediction of 'False.'

A threshold of 0.5 was used to further interpret the predictions, and the predicted probabilities were transferred to class labels ('True' or 'False'). The first five samples' projected probability and labels were then printed. The model's confidence in its predictions is provided as a percentage probability for each sample in these outputs.

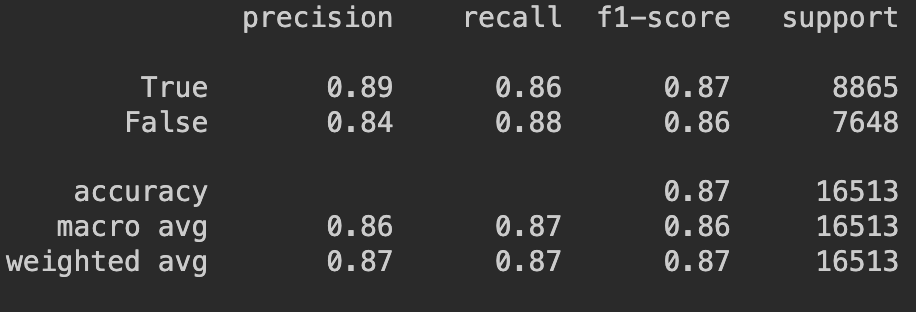


Figure 5. 8 Confusion matrix

The classification report produced by the classification report function gives a thorough evaluation of the model's performance on the test dataset. Precision, recall, and F1-score metrics for each class ('True' and 'False'), as well as overall accuracy, are included. The model attained a precision of 0.89 for 'True' and 0.84 for 'False,' a recall of 0.86 for 'True' and 0.88 for 'False,' and an overall accuracy of 87%, according to the research.

For both classes, the model displayed a high level of accuracy and balanced performance across precision, recall, and F1-score criteria. These results validate the model's performance in binary classification on the test dataset, offering vital insights into its prediction capabilities and generalisation to previously unseen data.

The model was then saved as a HDF5 file then integrated into a Django web app.

### 5.5.1 Testing Paradigm

#### Testing is the process of ensuring that a software product or application performs as expected. It aids in identifying problems and errors in the code and model so that they can be fixed. Thoroughly tested software ensures reliability, security, and excellent performance, as well as time savings, cost effectiveness, and customer happiness. This section covers how the model and application were tested, the testing paradigms employed, and the test results.

#### White box testing and black box testing were utilised as testing paradigms.

#### 

#### 5.5.1.1 White Box Testing

#### White box testing is an approach that examines the software's internal structure, internal design, code structure, and operation rather than only its functionality (Hamilton, 2020). The code is accessible to the testers here, and they must learn and understand it to understand the inner workings of the programme. The tester then checks the source code of the application for correct flow and organisation. A test case will be created for each procedure or combination of processes in the application by the tester. The model's white box testing was primarily done during training, when all the functions in the code were tested with different test cases.

#### A diagram of a computer code Description automatically generated

Figure 5. 9 White box testing

#### 5.5.1.2 Black Box Testing

Black box testing is a technique that tests the functionality of software applications without knowing the actual code structure, implementation details, or internal routes. A tester essentially offers an input and watches the system's output. It also enables a tester to replicate user activities and determine if the system fulfils its promises (Hamilton, 2020). This paradigm was employed in the prediction process, where the model was deployed on the web and used to predict on a sample statement. It was also used to test the web application's functionality when users interact with the model.

#### 

Figure 5. 10 Blackbox testing

### 5.5.2 Testing Results

#### This section shows the outputs and results of testing the various system modules. It also displayed the test cases utilised, their descriptions, and the test data.

Table 5. 3 Testing Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case | Description | Test Data | Result | Test Verdict |
| TC001 | Testing registration of a new user and verify that a user cannot register if they have not filled all required fields correctly. | Email is left  blank  Username –  Test1  Password –  test@123 | An error message is  displayed  showing the missing data field or the field that has not met the requirements. | Pass |
| TC002 | Testing registration of a new user providing all the required fields | Email – test@gmail.com  Username –  Test1  Password –  test@123 | The details of the new user are saved in the database and the user redirected to the login page. | Pass |
| TC003 | Testing login of a user where the user enters the correct credentials. | Username –  Test1  Password –  test@123 | The user is logged in and redirected to OTP page | Pass |
| TC004 | Testing login of a user where the user enters the wrong credentials. | Username – Test1  Password –  tes@3235 | Error message is displayed indicating that incorrect login details have been provided. | Pass |
| TC005 | Testing whether a user receives an OTP | Email  test@gmail.com | An OTP is sent to the users email address | Pass |
| TC006 | Testing whether a user enters the right an OTP as received in there mail | Email  test@gmail.com | A wrong OTP displays an error while a right OTP, the user is directed to the statement page | Pass |
| TC007 | Testing whether the model deployed gives a verdict | Statement  “Nairobi is in war’ | The model classifies this as false | Pass |
| TC008 | Testing if the model deployed gives a True Verdict | Statement  “[fuel](http://127.0.0.1:8000/admin/users/statementverdict/16/change/) prices have gone up in the world” | The model classified this as true | Pass |
| TC009 | Testing if the model deployed gives a False Verdict | Statement  “World War 3 is happening in Europe” | The model classified this as false | Pass |
| TC010 | Testing if user can view previous statement with their verdicts upon clicking dashboard and export the data | Export data | The user downloads statements with their verdicts | Pass |

# Chapter 6: Conclusion, Recommendations and Future Works

## 6.1 Conclusion

In conclusion, the research project focused on the development and evaluation of a machine learning model for credibility verification of information across various mediums, including social media, news articles, and headlines. The primary objective of the project was to create a web-based textual post credibility verification system, empowering users to swiftly determine the veracity of statements with a simple input.

The model is built on GloVe, a pre-trained word embedding approach that captures semantic associations between words. This method allows for a more detailed understanding of textual content by allowing the model to detect patterns associated with false or true statements.

By leveraging machine learning techniques, particularly natural language processing, the developed system contributes to the broader landscape of misinformation detection and prevention. The ease of use, accessibility, and quick turnaround time for information verification make it a valuable tool in the era of rapid information dissemination.

## 6.2 Recommendations

By using more data to carry out the training of the model including other languages not just limited to English would facilitate the robustness of the model in verifying information.

## 6.3 Future Works

The model's capabilities could be improved to include multilingual support, ensuring its effectiveness in tackling misinformation in a variety of linguistic contexts and appealing to a worldwide audience. Real-time verification techniques should be developed so that users may assess the reliability of information as it unfolds, especially in breaking news scenarios. Additionally, efforts should be put towards improving the model's explainability, giving users with insights into the elements driving credibility evaluations. Features for user feedback integration should be included to create a collaborative approach to refinement, allowing users to participate to the system's performance evaluation and improvement.

# References

Pennycook,G.,& Rand,D.G. (2019). The Implied Truth Effect: Attaching Warnings to a Subset of Fake News Stories Increases Perceived Accuracy of Stories Without Warnings. Managemnet Science , 67(11),4944-4957.

Vosoughi,S.,Roy,D., & Aral,S. (2018). The Spread of true and false online. Science, 359(6380),1146-1151.

Guess,A.,Nagler,J., & Tucker,J. (2020). Less than you think: Prevalence and Predictors of fake news . Science Advances , eaay3539.

Allcot,H., & Gentzkow, M. (2017). Social media and fake news in 2016 election . Journal of Economic Perspectives, 31(2), 211-236.

shu, K. D. (2017). FakeNewsNet: A Data Repository with News Content, Social Context and Dynamic Information for Studying Fake News on Social Media.

Guess, A. N. (2019). Exposure to untrustworthy websites in the 2016 US Election. Nature Human Behaviour, 3(4) , 308-313.

Jursafsky, D. &. (2020). Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition (3rd ed.). Pearson.

Potthast, M. H. (2018). A Plan for Actionable.

Horne, B. D. (2018). This just in: Fake news packs a lot in title, uses simpler, repetitive content in text body, more similar to satire than real news. Proceedings of the International AAAI Conference on Web and Social Media, 12(1), 607-616.

Wardle, C., & Derakhshan, H. ,. (2019). Information disorder: Toward an interdisciplinary framework for research and policy making. Council of Europe report, 1(0), 1-63.

Lazer, D. M. (2018). The science of fake news. Science. 359(6380), 1094-1096.

Chaffey, D. (2022, June 1). Global social media statistics research summary. Retrieved from Smart Insights: https://www.smartinsights.com/social-media-marketing/social- media-strategy/new-global-social-media-research/

Menczer. (2020). The science of fake news. Science. 359(6380), 1094-1096.

Lewandowsky, S., Ecker, U. K., & Cook, J. . (2020). Beyond Misinformation: Understanding and Coping with the “Post-Truth” Era. Applied Research in Memory and Cognition, 9(4), 434-440.

Shaina Raza and Cheng Ding. (2022). Fake news detection based on social contexts: a tranformer based approach . International Journal of Data Science and Analytics .

Rohit Kumar Kaliyar, A. G. (2021). FakeBERT: Fake news detection in social media with a BERT-based deep learning approach. Multimedia Tools and Applications, 80:11765–11788.

Pallabi Saitaki, K. G. (2022). Modelling Social Context for Fake News Detection: A Graph Neural Network Based Approach. Pandit Deendayal Energy University.

Shao, C., Ciampaglia, G. L., Varol, O., Yang, K. C., Flammini, A., & Menczer, F. (2018). The spread of fake news by social bots.

Martin, M. (2020). Prrototyping in Software Engineering: Methodolgogy Process. Retrieved from Ace Infoway: https://www.aceinfoway.com/blog/project-scope-and-prototype-model

Pressman, R. S. (2020). Software Engineering: A Practitioner's Approach. New York : McGraw-Hill Education.

Bahill, A. T. (2019). Tradeoff decisions in system design. Springer. Science+Business Media. .

Jeffrey A. Hoffer, J. G. (2021). Modern Systems Analysis and Design . Pearson Education .

Brahma Dathan, S. R. (2016). Object-Oriented Analysis, Design and Implementation: An Integrated Approach. Springer .

John W. Satzinger, R. B. (2015). Systems Analysis and Design in a Changing. Cengage Learning.

Dinesh Batra Joey F. George, J. S. (2019). Object-Oriented System Analysis and Design. Ingram.

Géron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition. O'Reilly Media, Inc.

Mudholkar, M. &. (2017). A Beginner’s Guide to Git and GitHub. Retrieved from ResearchGate: https://doi.org/10.13140/RG.2.2.20126.89927

Jupyter Team. (2021). JupyterLab: A web-based interactive development environment for Jupyter notebooks, JupyterLab 3.0. Retrieved from https://jupyterlab.readthedocs.io/en/stable/

# Appendices

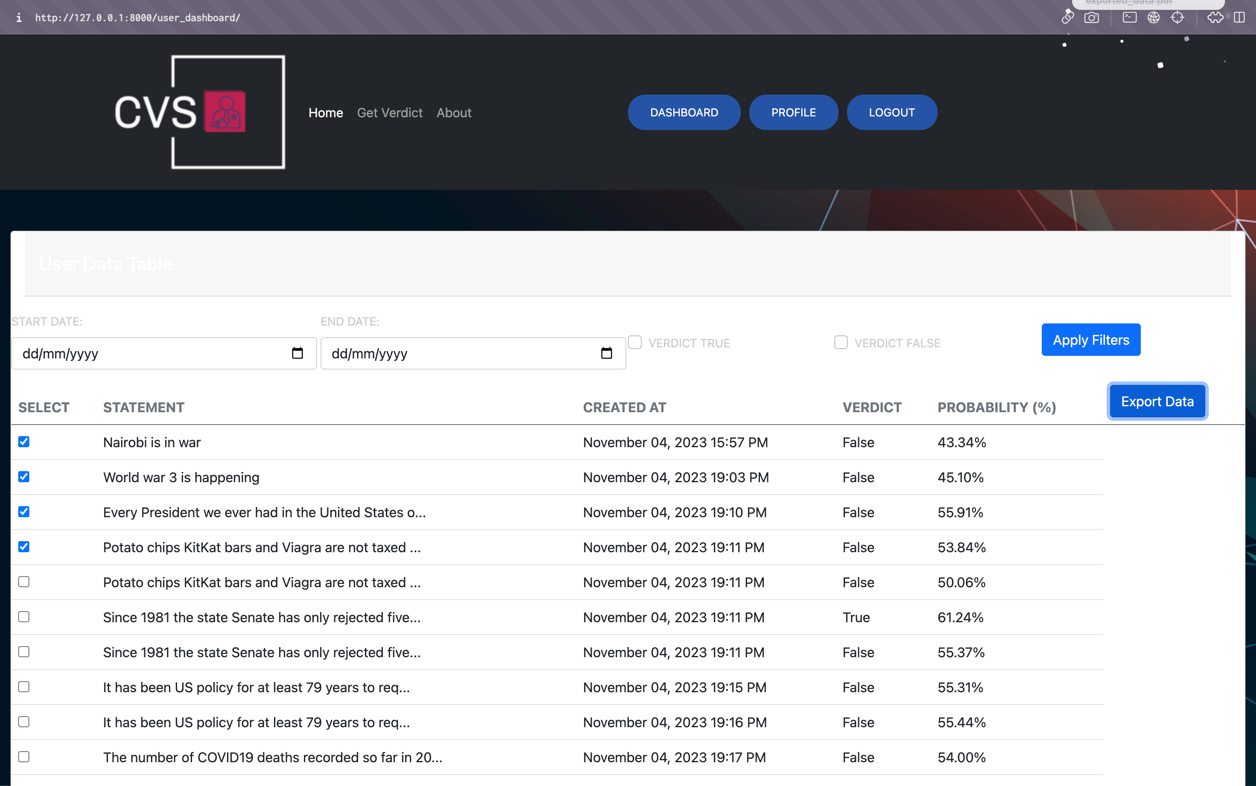
## Appendix 1: Timeline of Activities

A screenshot of a computer

Description automatically generated with medium confidence

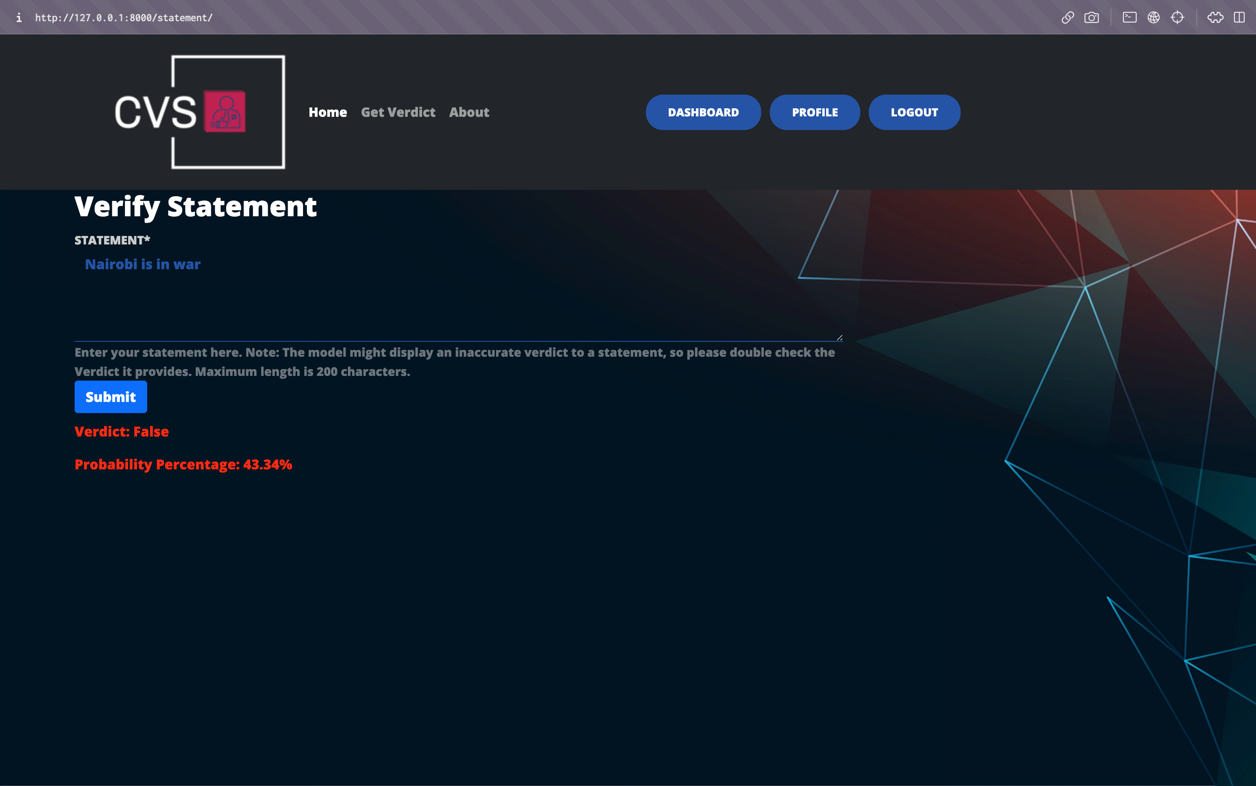
## Appendix 2: Test Cases

This shows user dashboard showing all the statements that are verified plus them being able to export those statements.



Appendix 2.1 user dashboard

This shows the page where a user inputs a statement for verification and receives a verdict



Appendix 2. 2 statement verification page

## Appendix 3: GitHub Analytics and Link

This shows the contributions to the project across the year.

A screenshot of a computer

Description automatically generated

Appendix 4. 1 Contributions analysis

This shows commits done on the project across the year.

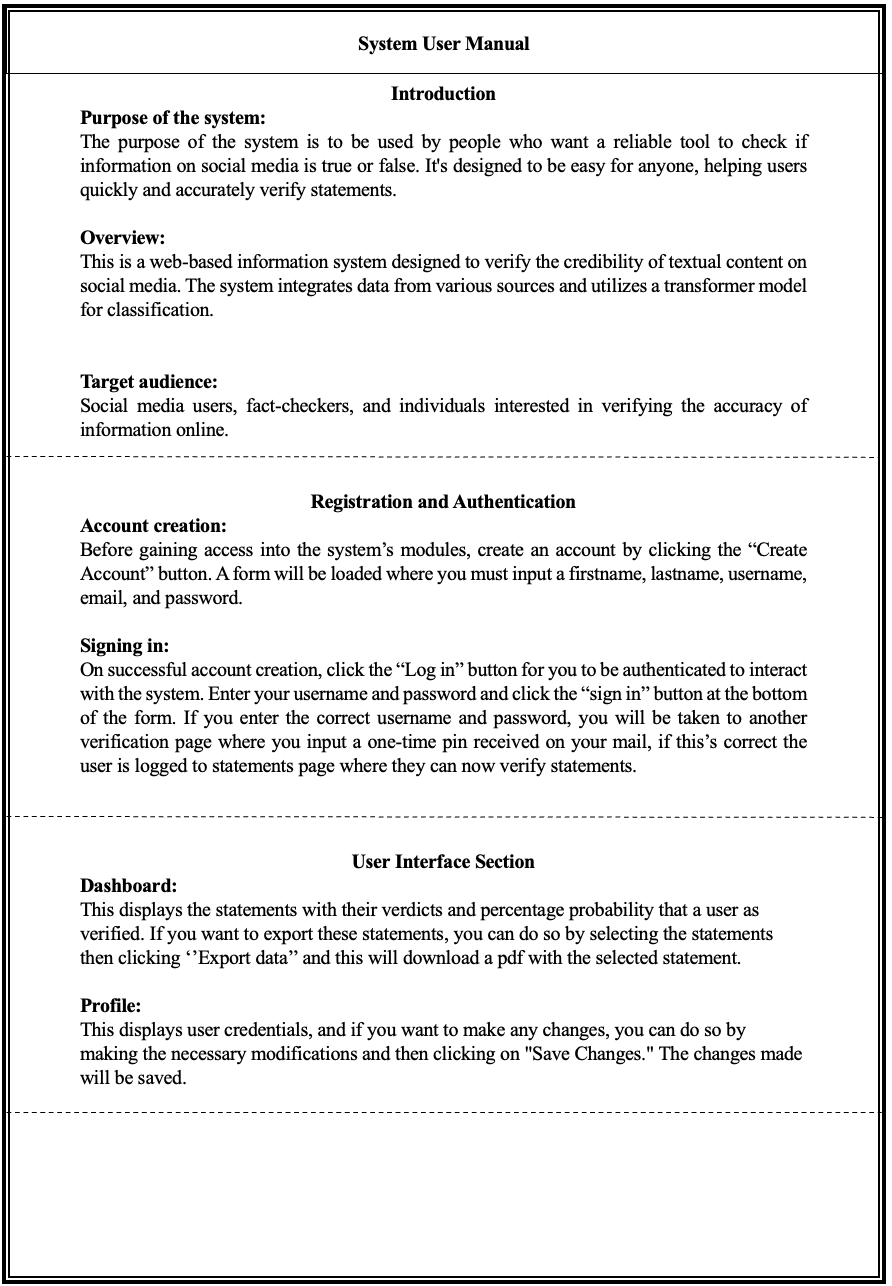
A screenshot of a computer

Description automatically generated

Appendix 4.2 commits analysis

Link to GitHub Repository: <https://github.com/derrickny/136788>

## Appendix 4: User Manual



A close-up of a sign

Description automatically generated

## Appendix 5: Plagiarism Report

A screenshot of a document

Description automatically generated

## Appendix 6: Marking Guide

**Strathmore University**

**School of Computing and Engineering Sciences**

**Information Systems Project Documentation Assessment Guide**

|  |  |
| --- | --- |
| Student Number | 136788 |
| Working Title | A Web-Based social media Textual Posts Credibility Verification System using Natural Language Processing |

|  |  |  |  |
| --- | --- | --- | --- |
| Evaluation Points | Weight | Score | Notes |
| Title | 1 |  |  |
| Abstract  Updated to include chapter 1-6 | 3 |  |  |
| Chapter 1-3  \*Checking previous proposal chapters for the correctness of title and problem statement, project scope as implemented and change of tenses |  |  |  |
| Problem Statement | 1 |  |  |
| Justification | 1 |  |  |
| Scope | 1 |  |  |
| Limitation | 1 |  |  |
| Literature Review | 2 |  |  |
| Methodology | 2 |  |  |
| Chapter 4  Correct functional requirements  Correct non-functional requirements  System Architecture and accompanying literature  4 Design diagrams and accompanying literature | 3  3  2  4 |  |  |
| Chapter 5  Setup Description: Hardware, software, support libraries, frameworks, versions and compatibility  Description of how the solution works to meet problem and business needs  Description of the test environment, data, test case   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Functional Requirement | Test Data | Expected Result | Actual Result | Pass/Fail | Evidence |   \*Check 3 core functional requirements and evidence of test available as appendix | 6  3  6 |  |  |
| Chapter 6  Valid Conclusion  Sound Recommendation | 2  2 |  |  |
| Presentation  Document Structure as per template provided and grammar  Citation and References  Document Numbering and Table of Contents/figures  Existence of required appendices | 2  2  2  1 |  |  |
| Total Marks | 50 |  |  |

Comments

Examiner Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_