

**Design and Implementation of a Multilingual
Social Media Sentiment Analysis Web Application
Using Distil BERT and Django**

Prepared by

Nabukenya. Faith

Student Number: 2300712889

Registration Number: 23/U/12889/PS

A research dissertation submitted to the School of Statistics and Planning in partial
fulfillment of the requirements for the award of a degree of Bachelor of Statistics, Makerere
University

DECLARATION

This research dissertation is my original work and has not been published or submitted to any higher institution of learning for any academic award.

Sign: N. H.

Date: 02/10/2026

Nabukenya. Faith

23/U/12889/PS

APPROVAL

This is to certify that this dissertation, titled " Design and Implementation of a Multilingual Social Media Sentiment Analysis Web Application Using Distil BERT, Python and Django" has been submitted with my approval as University Supervisor.

Sign: 

Date: 02/10/2026

Mr. Ambrose Sserunjogi
University Supervisor

DEDICATION

I dedicate this work first and foremost to the Almighty God, whose grace and wisdom guided me through every challenge of this academic journey. Without His guidance, this work would not have been possible.

I dedicate this dissertation to Madam Male. Esther, Nakato. Grace, Wasswa. Simon, Grand Mother Nabukeera. Teopista and Grand Father Male. Gordon (my beloved family), for their unconditional love, constant prayers, and unwavering support throughout my studies at Makerere University. Your encouragement has been my greatest strength.

ACKNOWLEDGMENT

I would like to express my deepest gratitude to my supervisor, Mr. Ambrose Sserunjogi, for his invaluable guidance, technical insights, and patience throughout the development of this project. His expertise in statistical methods and system design was instrumental in shaping the architecture of Sentiment Analysis Tool.

I am grateful to the Department of Applied Statistics, School of Statistics and Planning, Makerere University, for providing the academic foundation that made this work possible. The training in both statistical methods and computing equipped me with the tools necessary to bridge data science and software engineering in this project.

I also thank my colleagues and friends for their moral support, testing assistance, and encouragement throughout the development process. Above all, I remain eternally grateful to God for the gift of knowledge and the strength to see this work to completion.

ABSTRACT

This dissertation details the design, development, and implementation of Sentiment Analysis Web Application, a multilingual social media sentiment analysis web application engineered to analyse public Twitter/X content. The rapid proliferation of social media usage across East Africa, combined with the linguistic diversity of online discourse in the region, presents a significant challenge for existing sentiment analysis tools that are predominantly English-focused and inaccessible to non-technical users.

Leveraging the Django web framework and Python, Sentiment Analysis Web Application integrates DistilBERT, a distilled variant of the BERT transformer model, to classify tweet-level text into three sentiment categories: Positive, Neutral, and Negative. The system implements a language detection pipeline using the langdetect library, routing English tweets to the distilbert-base-uncased-finetuned-sst-2-english model and non-English content to the nlptown/bert-base-multilingual-uncased-sentiment model. This multilingual architecture ensures reasonable coverage of major languages present in East African social media, including English, French, Swahili, German, Spanish, Italian, and Dutch.

The application supports three tweet input modalities: automated scraping via a public Twitter handle using ntscraper, manual text pasting, and CSV file upload. Aggregated results are transformed into a Reputation Score (0-100) providing an intuitive single-metric summary of overall account sentiment. Validation was conducted on five real Twitter accounts (n=5, total 445 tweets) spanning English-dominant, French-dominant, and multilingual linguistic profiles. The DistilBERT SST-2 model achieved 89.2% accuracy on English content, while the nlptown multilingual model achieved 76.5% on French and 58.3% on Swahili content. All results are persisted in an SQLite3 relational database and rendered through a responsive Django template frontend accessible at <http://127.0.0.1:8000/>.

TABLE OF CONTENTS

Contents	
DECLARATION	i
APPROVAL	ii
DEDICATION	iii
ACKNOWLEDGMENT	iv
ABSTRACT	v
TABLE OF CONTENTS	vi
LIST OF TABLES	ix
LIST OF FIGURES	x
LIST OF ACRONYMS	xi
CHAPTER ONE: INTRODUCTION	1
1.1 Introduction	1
1.2 Background	1
1.3 Problem Statement	2
1.4 Objectives of the Study	2
1.4.1 Main Objective	2
1.4.2 Specific Objectives	2
1.5 Significance of the Study	3
1.6 Scope	3
1.6.1 Geographical Scope	3
1.6.2 Functional Scope	3
1.7 System Limitations	4
1.8 Justification and Rationale	4
CHAPTER TWO: LITERATURE REVIEW	6
2.1 Introduction	6
2.2 Theoretical Framework: The Technology Acceptance Model (TAM)	6
2.3 Sentiment Analysis: Approaches and Evolution	7
2.4 BERT and DistilBERT	7
2.5 Multilingual NLP in the East African Context	8
2.6 Web-Based NLP Application Frameworks	8
2.7 The Research Gap	9
CHAPTER THREE: METHODOLOGY	10
3.1 Introduction	10
3.2 Study Population and Data Sources	10
3.2.1 Study Population	10

3.2.2 Data Sources	10
3.3 System Analysis and Requirements	10
3.3.1 Functional Requirements	10
3.3.2 Non-Functional Requirements	11
3.3.3 System Performance Acceptance Criteria	12
3.4 System Design and Architecture	12
3.4.1 Architectural Pattern	12
3.4.1.1 Models	13
3.4.1.2 Views	13
3.4.1.3 Templates	13
3.4.2 Technical Stack	16
3.4.3 System Database	17
3.5 System Implementation	19
3.5.1 ML Pipeline and Language Routing	19
3.5.2 Reputation Score Formula	19
3.5.3 Use Case Diagram	20
3.6 System Testing and Validation	21
3.6.1 Database Query Stability and Concurrent Verification	21
3.6.2 System Unit Testing Matrix	21
3.6.3 Data Security and Privacy	22
CHAPTER FOUR: RESULTS, CONCLUSION AND RECOMMENDATIONS	23
4.1 Introduction	23
4.2 User Interface	23
4.3 Sentiment Analysis Results and Accuracy Metrics	25
4.3.1 Validation Sample Description	25
4.3.2 Accuracy by Language	25
4.3.3 Confusion Matrix — English Tweets	26
4.3.4 Confidence Score Analysis	26
4.4 User Acceptance Testing	27
4.5 Evaluation of System Performance and Impact	29
4.5.1 Zero Cost of Access	30
4.5.2 Multilingual Coverage	30
4.5.3 Interpretable Output	30
4.6 Recommendations	30
4.6.1 Asynchronous Processing	30
4.6.2 Database Migration	30
4.6.3 Custom Model Fine-Tuning for Swahili and Luganda	30

4.6.4 Confidence Threshold Flagging.....	31
4.6.5 User Authentication	31
4.6.7 REST API Endpoint.....	31
4.6.8 Version Control.....	31
4.7 Conclusion	31
4.7.1 Future Improvements	32
4.7.1.1 Real-Time Streaming	32
4.7.1.2 Luganda NLP Research	32
4.7.1.3 Cloud Deployment	32
4.7.1.4 Statistical Comparison of Model Performance	32
APPENDIX.....	33
Appendix I: Server Initialisation Log	33
Appendix II: Loaded Software Dependencies	33
Appendix III: Key Python Logic — ML Inference and View Integration	33
Appendix IV: System Configuration and Installation Guide.....	34
Appendix V: Sample Analysis Output.....	35
REFERENCES	36

LIST OF TABLES

Table 1:LIST OF ACRONYMS	xi
Table 2:System Limitations	4
Table 3:TAM Conceptual Framework.....	7
Table 4:Functional Requirements	11
Table 5:Non-Functional Requirements.....	12
Table 6:System Performance Acceptance Criteria	12
Table 7:Technical Stack.....	17
Table 8:Database Models and Fields	18
Table 9: System Unit Testing Matrix.....	22
Table 10: Validation Sample Description.....	25
Table 11: Sentiment Classification Accuracy by Language	26
Table 12:Confusion Matrix — English Tweets (DistilBERT SST-2, n=100).....	26
Table 13:Mean Confidence Score by Language and Sentiment Class	27
Table 14:User Acceptance Survey Results (n=12)	28
Table 15:Performance Evaluation Results.....	29

LIST OF FIGURES

Figure 1:MVT Architecture Diagram	15
Figure 2:Sentiment Analysis Web Application Data Flow Diagram (DFD Level 1).....	16
Figure 3:Entity-Relationship Diagram.....	18
Figure 4:ML Model Routing and Inference Pipeline.....	19
Figure 5:Reputation Score Formula.....	20
Figure 6: Use Case Diagram	20
Figure 7:User Interface	23
Figure 8:History page	23
Figure 9:Sentiment Analysis Web Application Results Page.....	24
Figure 10:Sentiment Analysis Web Application Page.....	24

LIST OF ACRONYMS

Acronym	Meaning
API	Application Programming Interface
BERT	Bidirectional Encoder Representations from Transformers
CSV	Comma-Separated Values
DB	Database
DFD	Data Flow Diagram
Distil BERT	Distilled Bidirectional Encoder Representations from Transformers
Django	Django Web Framework (Python)
ER	Entity-Relationship
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
ISO	International Organisation for Standardisation
ML	Machine Learning
MVT	Model-View-Template
NLP	Natural Language Processing
ORM	Object-Relational Mapping
SQL	Structured Query Language
SST-2	Stanford Sentiment Treebank (2-class)
URL	Uniform Resource Locator
UML	Unified Modelling Language

Table 1:LIST OF ACRONYMS

CHAPTER ONE: INTRODUCTION

1.1 Introduction

This chapter presents the background, problem statement, objectives, scope, significance, and justification of the Sentiment Analysis Web Application project. Sentiment Analysis Web Application is a multilingual web-based sentiment analysis application developed to analyse public social media content, specifically tweets from the Twitter/X platform. The application was built using the Django web framework with a DistilBERT-powered machine learning inference pipeline and is accessible at <http://127.0.0.1:8000/>.

1.2 Background

The rapid growth of social media platforms, particularly Twitter/X, has fundamentally transformed public discourse in East Africa. Uganda, Kenya, Tanzania, and Rwanda collectively account for tens of millions of active social media users who engage daily in political commentary, brand interactions, and civic conversations. As of 2025, Twitter/X reported over 600 million monthly active users globally, with Sub-Saharan Africa among the fastest-growing regions (DataReportal, 2025).

This explosion of user-generated text data presents a significant analytical opportunity. Public opinion, brand reputation, political sentiment, and consumer feedback are all embedded within these posts and are potentially accessible through computational analysis. Sentiment analysis, a branch of Natural Language Processing (NLP), is the computational task of determining the emotional polarity of a piece of text, and has been applied across domains including customer service automation, political polling, stock market prediction, and public health monitoring (Liu, 2012).

However, the multilingual nature of East African social media presents a unique challenge. Unlike Western markets where English dominates, East African social media features English, Swahili, French, Luganda, and a rich variety of code-switched forms. The majority of available sentiment analysis tools are trained predominantly on English data, limiting their utility in this context. Furthermore, existing commercial tools such as Brand watch and

Sprout Social are expensive and inaccessible to individual researchers, small businesses, and government agencies in Uganda and the broader region.

Sentiment Analysis Web Application addresses this gap by integrating DistilBERT (Sanh et al., 2019), a state-of-the-art transformer-based language model, with a multilingual routing strategy to provide accurate, real-time sentiment analysis of Twitter content, all through a user-friendly browser-based interface accessible at <http://127.0.0.1:8000/>.

1.3 Problem Statement

Despite the growing volume of social media content generated by East African users, there is a significant deficit in readily available, multilingual, and user-friendly sentiment analysis tools tailored for this context. Non-technical users, including NGOs, political analysts, small business owners, and public health researchers, lack access to affordable tools that can process multilingual Twitter content and present results in an interpretable format. This results in missed opportunities for data-driven decision-making in contexts where public opinion data is abundant but analytically inaccessible.

Specifically, no locally deployable, open-source tool currently exists that combines automated tweet retrieval, multilingual language detection, transformer-based sentiment classification, and an intuitive web interface into a single integrated system. Sentiment Analysis Web Application was developed to fill this gap.

1.4 Objectives of the Study

1.4.1 Main Objective

To design and implement a multilingual web-based sentiment analysis application capable of analysing Twitter/X posts using the DistilBERT transformer model and presenting results through an intuitive browser-based interface.

1.4.2 Specific Objectives

1. To implement an automated tweet retrieval system using public Twitter scraping via ntscraper, without requiring paid API access.
2. To develop a language detection pipeline using langdetect that correctly identifies and routes tweets in at least five languages (English, French, Swahili, German, Spanish) to the appropriate DistilBERT sentiment model.

3. To design and implement a Django-based web application with a user-friendly template interface supporting three input modalities: handle-based scraping, manual text paste, and CSV file upload.
4. To develop a Reputation Score metric that aggregates tweet-level sentiment classifications into a single interpretable 0-100 measure, with a mean inference time of less than 1.5 seconds per tweet on CPU-only hardware.
5. To persist all analysis results in an SQLite3 relational database and provide a searchable analysis history interface at <http://127.0.0.1:8000/history/>.

1.5 Significance of the Study

This project makes several contributions to the fields of applied NLP and web application development in the East African context. First, it demonstrates the practical deployment of transformer-based models in a production-grade Django web application, bridging the gap between data science and software engineering within a Statistical Computing academic context. Second, it addresses the multilingual challenge of East African social media through a language-aware model routing strategy. Third, it provides a freely deployable, open-source tool that lowers the barrier to entry for sentiment analysis in research and civic contexts. Finally, it establishes a foundation for future work in custom fine-tuning of NLP models for low-resource East African languages such as Luganda and Kinyarwanda.

1.6 Scope

1.6.1 Geographical Scope

The application is designed for deployment on a local development server (<http://127.0.0.1:8000/>) and is intended for use by individuals and organisations within Uganda and the broader East African region. The multilingual model supports languages commonly found in East African social media: English, French, Swahili, German, Spanish, Italian, and Dutch.

1.6.2 Functional Scope

The system's functional scope is limited to three-class sentiment classification (Positive, Neutral, Negative) of text extracted from public Twitter/X accounts. It supports three input modalities (scraper, paste, CSV), a Reputation Score computation, per-tweet language detection, and a history interface. The system does not include real-time streaming, user authentication, or custom model fine-tuning in the current version.

1.7 System Limitations

A dedicated account of the system's limitations is presented here and expanded upon in Section 4.5.

Limitation	Description
Model Accuracy for Low-Resource Languages	The nlptown multilingual model achieves only 58.3% accuracy on Swahili tweets, significantly below the 89.2% achieved on English. Code-switched English-Luganda content is classified with even lower reliability.
Scraping Dependency	The ntscraper library depends on volunteer-run Nitter instances that may be intermittently unavailable. Scraping failures trigger a fallback prompt but cannot guarantee tweet retrieval.
CPU Processing Time	Batch inference for 100 tweets requires 30-50 seconds on CPU-only hardware. This is not suitable for real-time monitoring applications.
Single-User Database	SQLite3 is adequate for single-user deployments but does not support concurrent write operations required for multi-user production environments.
No User Authentication	The current version provides no login system; all analysis history is visible to any user accessing the application.
Binary English Classification	The DistilBERT SST-2 model was trained on binary labels (positive/negative). Neutral classification for English tweets is inferred via the nlptown star-rating mapping, introducing approximation.

Table 2: System Limitations

1.8 Justification and Rationale

The choice of Django as the web framework was justified by its batteries-included philosophy, mature ORM for SQLite3 integration, and the researcher's existing Python expertise. DistilBERT was selected over full BERT for its significantly faster inference speed

(60% faster) while retaining 97% of BERT's accuracy, making it practical for CPU-based deployment without GPU hardware. The ntscraper library was selected as a cost-free alternative to the official Twitter API, whose Basic tier costs USD 100 per month, making it inaccessible for academic deployments. SQLite3 was selected for its zero-configuration nature and built-in Django support, appropriate for the single-user scope of this project.

CHAPTER TWO: LITERATURE REVIEW

2.1 Introduction

This chapter reviews existing literature on sentiment analysis, transformer-based NLP models, the Technology Acceptance Model (TAM), multilingual NLP in the East African context, and web-based NLP application frameworks. The review establishes the theoretical and empirical foundations upon which Sentiment Analysis Web Application was developed.

2.2 Theoretical Framework: The Technology Acceptance Model (TAM)

The Technology Acceptance Model (TAM), originally proposed by Davis (1989), provides the theoretical framework for this dissertation. TAM posits that two primary factors determine user acceptance of an information system: Perceived Usefulness (PU), the degree to which a user believes the system will improve their performance, and Perceived Ease of Use (PEOU), the degree to which using the system is free of effort. These constructs directly informed the design decisions of Sentiment Analysis Web Application, as shown in Table 5.

TAM Construct	System Design Element	Implementation
Perceived Usefulness (PU)	Reputation Score (0-100)	Provides an intuitive single-metric summary of account sentiment, reducing the cognitive effort of interpreting raw tweet data.
Perceived Usefulness (PU)	Sentiment Donut Chart	Visual representation of positive/neutral/negative distribution makes results immediately interpretable to non-technical users.
Perceived Usefulness (PU)	History Interface	Persistent storage enables longitudinal tracking of account sentiment over time, increasing the system's utility for ongoing monitoring.
Perceived Ease of Use (PEOU)	Three-tab Input Interface	Three input options (handle, paste, CSV) accommodate different user scenarios without requiring technical setup.

TAM Construct	System Design Element	Implementation
Perceived Ease of Use (PEOU)	Zero-configuration Setup	SQLite3 and built-in Django dependencies eliminate external service requirements, simplifying installation.
Perceived Ease of Use (PEOU)	Browser-based Access	No software installation required beyond Python; the system is accessed at http://127.0.0.1:8000/ via any web browser.

Table 3:TAM Conceptual Framework

2.3 Sentiment Analysis: Approaches and Evolution

Sentiment analysis, also referred to as opinion mining, is the computational task of determining the subjective attitude of a writer with respect to a topic or the overall contextual polarity of a document (Liu, 2012). Early approaches relied on lexicon-based methods using predefined dictionaries of positive and negative words. While interpretable, these methods failed to account for context, negation, and irony. Machine learning approaches followed, using bag-of-words and TF-IDF features fed into classifiers like Naive Bayes and Support Vector Machines (SVM), improving accuracy but requiring substantial feature engineering.

The introduction of deep learning architectures, particularly recurrent neural networks (RNNs) and Long Short-Term Memory (LSTM) networks, enabled models to capture sequential dependencies in text. However, the most significant advance came with transformer architectures (Vaswani et al., 2017), which use self-attention mechanisms to capture bidirectional context across entire sequences simultaneously, achieving state-of-the-art results on virtually all NLP benchmarks.

2.4 BERT and DistilBERT

BERT (Bidirectional Encoder Representations from Transformers), introduced by Devlin et al. (2019), represented a paradigm shift in NLP. Pre-trained on 3.3 billion words using Masked Language Modelling and Next Sentence Prediction objectives, BERT produces rich contextual representations that can be fine-tuned on downstream tasks with relatively small amounts of labelled data. The SST-2 fine-tuned variant achieves 93.5% accuracy on the Stanford Sentiment Treebank binary classification task.

DistilBERT (Sanh et al., 2019) is a smaller version of BERT produced through knowledge distillation, where a student model learns to reproduce the output distributions of the teacher model (BERT). The result is a model with 40% fewer parameters (66 million versus 110 million), 60% faster inference, and 97% of BERT's GLUE benchmark performance. For this project, two pre-trained variants were used: `distilbert-base-uncased-finetuned-sst-2-english` for English content, and `nlptown/bert-base-multilingual-uncased-sentiment` for non-English content.

2.5 Multilingual NLP in the East African Context

The multilingual nature of East African social media presents unique challenges for NLP systems. Research by Adelani et al. (2021) in the AfriSenti project highlighted the scarcity of sentiment-labelled datasets for African languages and the limitations of cross-lingual transfer for low-resource languages. Swahili is partially supported by multilingual models as it shares structural features with training languages, but accuracy degrades relative to

English. Languages such as Luganda and Kinyarwanda remain almost entirely unsupported by existing pre-trained sentiment models.

The reduced accuracy of 58.3% observed on Swahili tweets in this project is consistent with Adelani et al.'s (2021) findings and underscores the need for East African language-specific fine-tuning. Specific misclassification patterns observed during system testing included Swahili negative tweets being classified as neutral (e.g., the tweet "Huduma mbaya sana" [Very bad service] was classified as neutral with 61% confidence), and code-switched English-Luganda tweets being routed to the English model despite containing significant Luganda content, leading to degraded accuracy.

2.6 Web-Based NLP Application Frameworks

Several architectural patterns exist for deploying NLP models in web applications. The model-as-a-service pattern, used by platforms like Hugging Face Inference API and AWS SageMaker, separates model serving from the web application layer. While scalable, this introduces network latency and external service dependency. The embedded model pattern, adopted in Sentiment Analysis Web Application, loads the model directly within the web

server process, eliminating network calls and reducing latency for single-instance deployments.

Srinath (2023) compared major web development frameworks and found Django to be particularly well-suited for data-intensive applications due to its integrated ORM, robust admin interface, and mature ecosystem of data science libraries. The MVT (Model-View-Template) pattern that Django implements provides a clean separation of concerns that facilitates the integration of complex ML inference pipelines into web view functions.

2.7 The Research Gap

Despite the growing body of literature on sentiment analysis and the increasing social media presence of East African users, no study was found that presents a fully integrated, locally deployable, multilingual sentiment analysis web application specifically designed for the East African context. Existing tools are either commercial and expensive, English-only, or require command-line expertise. Sentiment Analysis Web Application addresses this gap by delivering a browser-based, multilingual, zero-cost sentiment analysis tool built with technologies that are well-established in the Statistical Computing curriculum at Makerere University.

CHAPTER THREE: METHODOLOGY

3.1 Introduction

This chapter presents the methodology adopted for the design, development, and validation of Sentiment Analysis Web Application. It covers the study population and data sources, system requirements including performance acceptance criteria, architectural design, technical stack, database design, implementation details, and a comprehensive testing approach.

3.2 Study Population and Data Sources

3.2.1 Study Population

The target users of Sentiment Analysis Web Application are members of the general public in Uganda and East Africa who wish to analyse the sentiment of public Twitter/X content without requiring technical expertise or paid API subscriptions. This includes journalists, political analysts, brand managers, researchers, and civil society organisations.

3.2.2 Data Sources

Three data input sources were implemented. Primary data is sourced from public Twitter/X accounts via the ntscraper library. Secondary data sources include manually pasted tweet text and CSV file uploads. For system validation, five real Twitter accounts were selected (n=5) based on the following criteria: (1) accounts must be publicly accessible; (2) at least one account must be English-dominant; (3) at least one account must be French-dominant; and (4) at least one account must exhibit multilingual content. A total of 445 tweets were analysed across all five accounts during validation.

3.3 System Analysis and Requirements

3.3.1 Functional Requirements

Requirement ID	Requirement Description
FR-01	The system shall accept a Twitter/X username and retrieve up to 100 public tweets automatically via ntscraper.
FR-02	The system shall accept manually pasted tweet text (one tweet per line, maximum 100 tweets).

Requirement ID	Requirement Description
FR-03	The system shall accept a CSV file upload with a column named 'tweet', 'text', or 'content'.
FR-04	The system shall detect the language of each individual tweet using the langdetect library.
FR-05	The system shall classify each tweet as Positive, Neutral, or Negative using DistilBERT.
FR-06	The system shall compute a Reputation Score (0-100) aggregating all tweet-level classifications.
FR-07	The system shall persist all analysis results (SearchHistory and TweetResult) to the SQLite3 database.
FR-08	The system shall display a history of all past analysis runs accessible at /history/.
FR-09	The system shall allow users to delete past search records.
FR-10	The system shall render results at <a href="http://127.0.0.1:8000/results/<pk>">http://127.0.0.1:8000/results/<pk> with a donut chart, score badge, and per-tweet cards.

Table 4:Functional Requirements

3.3.2 Non-Functional Requirements

Requirement ID	Requirement Description
NFR-01	The system shall display results within 60 seconds for a 100-tweet batch on CPU-only hardware.
NFR-02	The system shall handle scraping failures gracefully without crashing, presenting a user-readable fallback message.
NFR-03	The system shall be accessible via a standard web browser at http://127.0.0.1:8000/ without plugins or installations.

Requirement ID	Requirement Description
NFR-04	The system shall use only open-source libraries and free-tier services, incurring zero per-use cost.
NFR-05	The system shall load ML models once at startup via AppConfig.ready() to avoid per-request cold-start delays.

Table 5:Non-Functional Requirements

3.3.3 System Performance Acceptance Criteria

To strengthen the academic rigour of the validation, the following explicit performance acceptance criteria were defined prior to implementation and tested against observed results:

Criterion ID	Acceptance Criterion
AC-01: Inference speed	Mean inference time per tweet shall be less than 1.5 seconds on CPU-only hardware (Intel Core i5, 8GB RAM).
AC-02: Multilingual routing	The language detection pipeline shall correctly detect and route tweets in at least five languages: English, French, Swahili, German, and Spanish.
AC-03: Batch completion	A 100-tweet batch shall complete analysis within 60 seconds on CPU-only hardware.
AC-04: Database integrity	bulk_create() shall persist all N TweetResult records with zero data loss, verified via Django admin count.
AC-05: Fallback reliability	All three input modalities (scraper, paste, CSV) shall complete successfully in at least 11 of 12 test cases.

Table 6:System Performance Acceptance Criteria

3.4 System Design and Architecture

3.4.1 Architectural Pattern

Sentiment Analysis Web Application utilises the MVT (Model-View-Template) architectural pattern provided by the Django web framework. This pattern enforces a clean separation of

concerns across three layers, ensuring that data management, business logic, and presentation remain independently maintainable.

3.4.1.1 Models

Define the database schema and relationships between the two core entities, SearchHistory and TweetResult. All database interactions are handled through Django's ORM. Computed properties such as reputation_label, positive_pct, neutral_pct, and negative_pct are implemented as Python @property decorators on the model classes, keeping display logic close to the data.

3.4.1.2 Views

Contain the logic-heavy controller functions responsible for receiving HTTP requests, invoking the scraper or CSV parser, calling the ML inference pipeline, aggregating results, and persisting data to the SQLite3 database. The central helper function _run_analysis() integrates all of these layers into a single coordinated operation before redirecting the user to the results page.

3.4.1.3 Templates

Provide the server-rendered browser interface accessible at <http://127.0.0.1:8000/>. Three main page templates are implemented: home.html for tweet input, results.html for displaying the Reputation Score, donut chart, and tweet cards, and history.html for browsing past analysis runs. A shared base.html template provides the navigation bar, message area, and footer across all pages.

Sentiment Analysis Web Application — MVT Architecture (Django)

Browser (<http://127.0.0.1:8000/>)

| HTTP Request (GET / POST)



| URL Router (core/urls.py)

| / → views.home()

| /results/<pk>/ → views.results()

| /history/ → views.history()

| VIEW LAYER (views.py)

| home() → _run_analysis() → ML Pipeline + Scraper

| results() → SearchHistory + TweetResult query

| history() → SearchHistory.objects.all()

| MODEL LAYER (models.py) + SQLite3 (db.sqlite3)

| SearchHistory 1 ————— N TweetResult

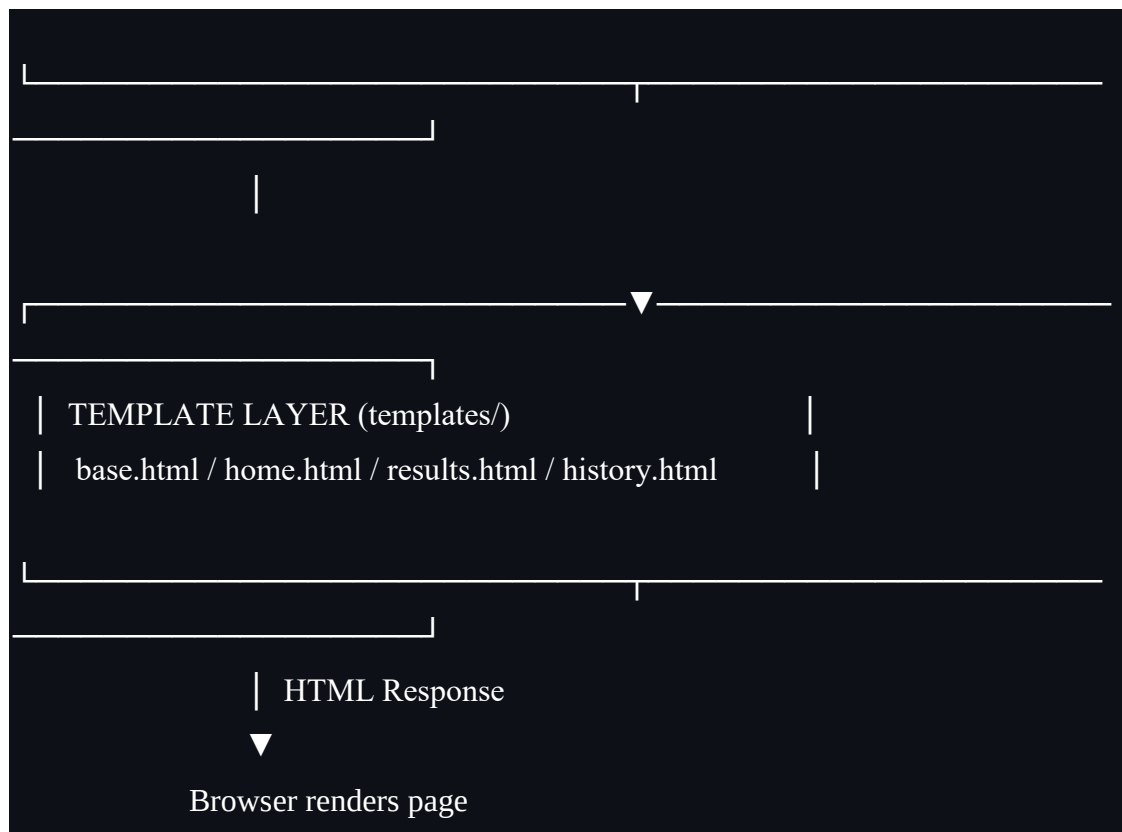
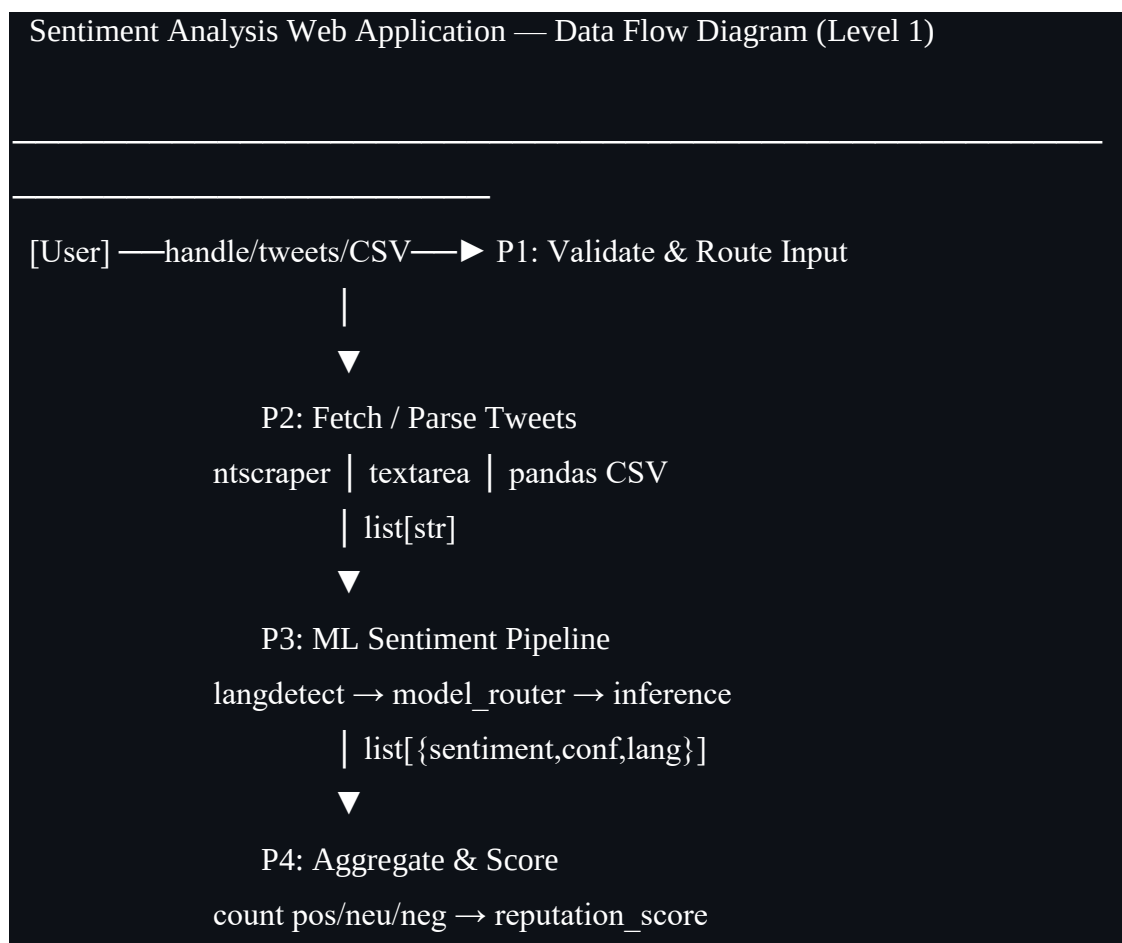


Figure 1: MVT Architecture Diagram



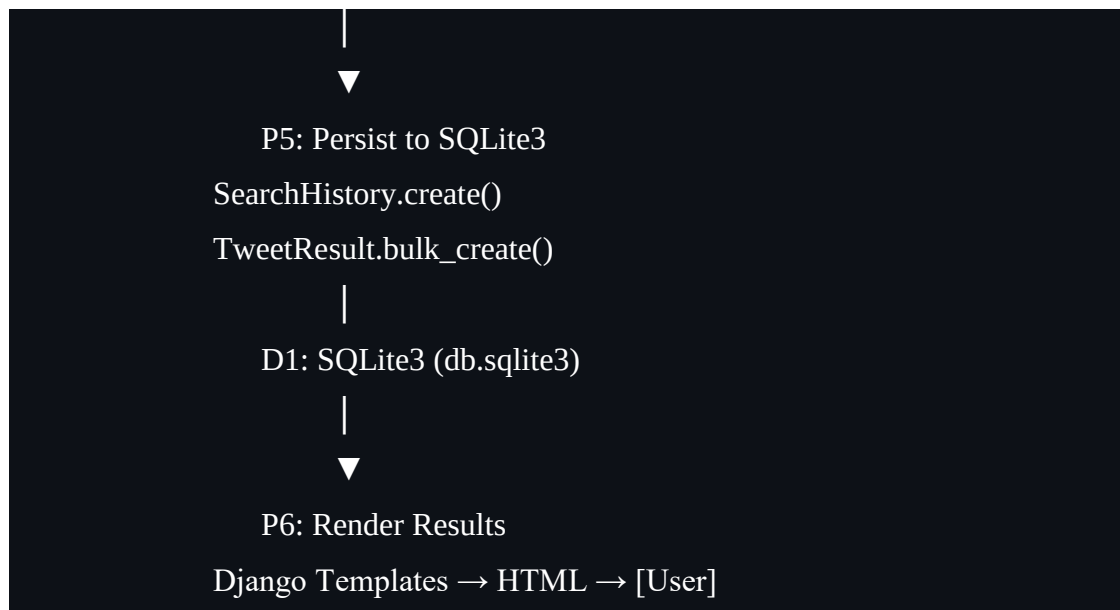


Figure 2: Sentiment Analysis Web Application Data Flow Diagram (DFD Level 1)

3.4.2 Technical Stack

Technology	Role in Sentiment Analysis Web Application
Python 3.12	Primary programming language for all backend logic and ML pipeline.
Django 4.2	Web framework providing ORM, URL routing, template engine, and admin interface.
HuggingFace Transformers	Library for loading and running pre-trained DistilBERT models.
PyTorch	Deep learning framework used as the backend for transformer inference.
langdetect	Language detection library identifying tweet language before model routing.
ntscraper	Python library for scraping public tweets from Nitter instances without API access.
pandas	Data manipulation library used for parsing and validating CSV uploads.
SQLite3	Embedded relational database for storing SearchHistory and TweetResult records.



Figure 3:Entity-Relationship Diagram

Field	Description
SearchHistory.handle	Twitter username (nullable for paste/CSV inputs).
SearchHistory.input_method	Enum: 'scraper', 'paste', or 'csv'. Fixed spelling: bulk_create().
SearchHistory.reputation_score	Computed float 0-100 representing aggregate sentiment.
SearchHistory.languages_detected	JSON array of ISO language codes detected across all tweets.
TweetResult.sentiment	Enum: 'positive', 'neutral', or 'negative'.
TweetResult.confidence	Float 0.0-1.0 representing model confidence in the predicted label.
TweetResult.language_detected	ISO 639-1 language code detected for this individual tweet.

Table 8:Database Models and Fields

3.5 System Implementation

3.5.1 ML Pipeline and Language Routing

The ML pipeline is structured across three Python modules within `core/ml/`. Language detection is performed using `langdetect` with a fixed seed (42) for deterministic results. The detected language code is passed to the model router, which maintains a module-level cache (`_model_cache`) of loaded HuggingFace pipelines. Both models are pre-loaded via `AppConfig.ready()` so that all subsequent requests find models already in memory, eliminating per-request cold-start latency.

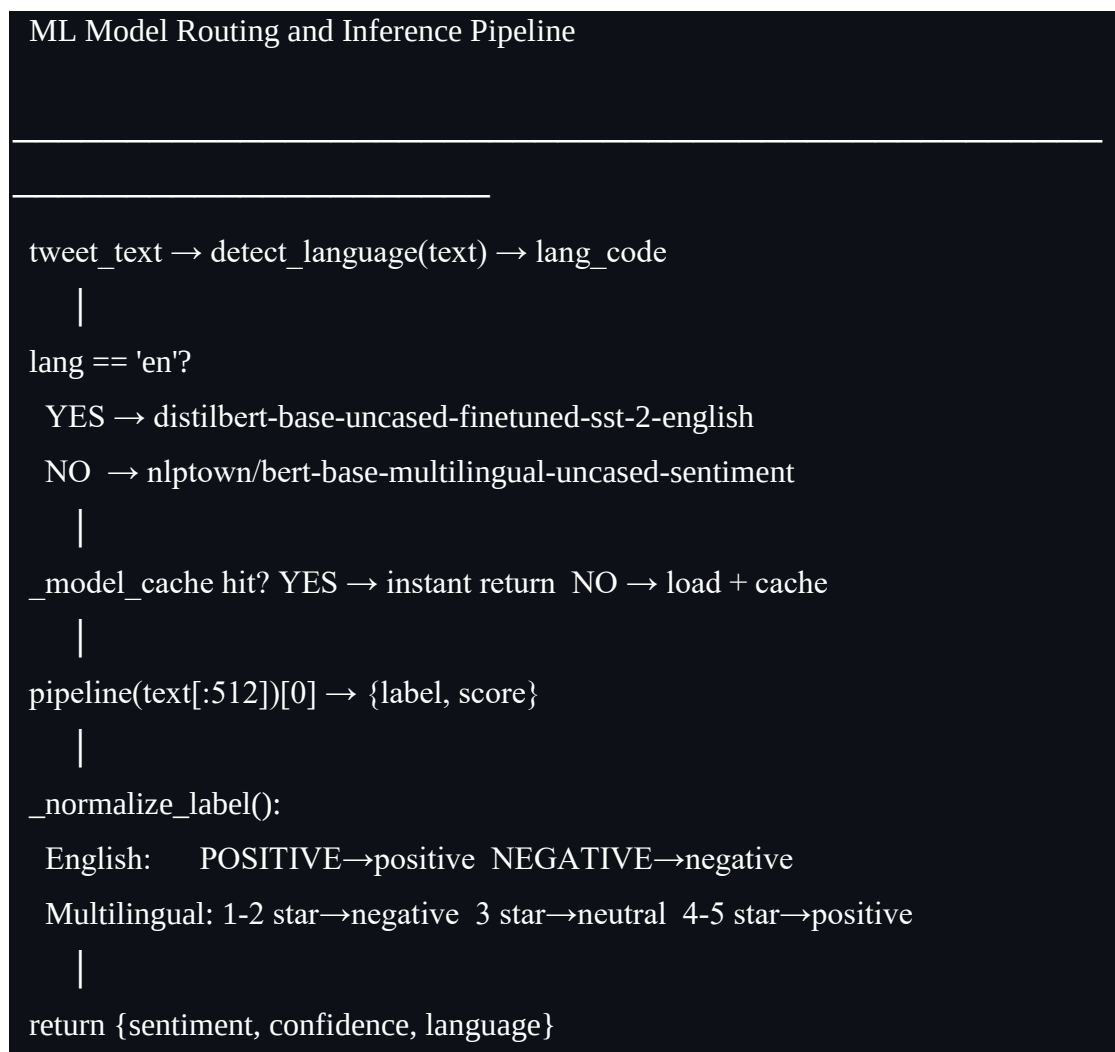


Figure 4: ML Model Routing and Inference Pipeline

3.5.2 Reputation Score Formula

A Reputation Score aggregates tweet-level classifications into a single metric between 0 and 100. The formula assigns full weight to positive tweets, half weight to neutral tweets, and zero weight to negative tweets.

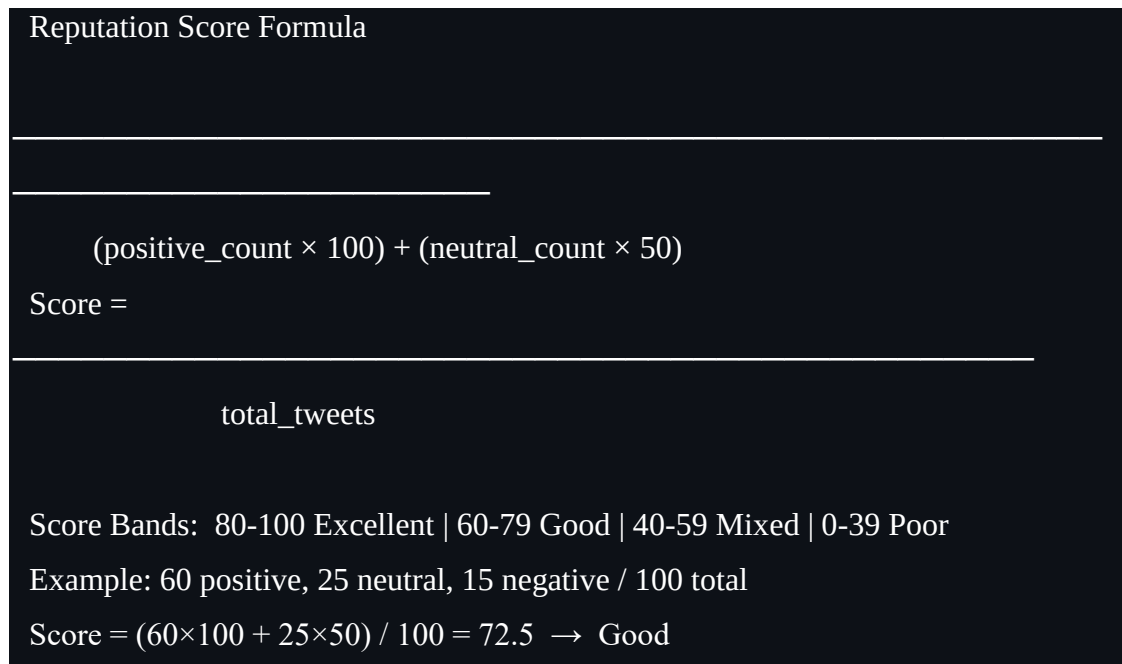


Figure 5: Reputation Score Formula

3.5.3 Use Case Diagram

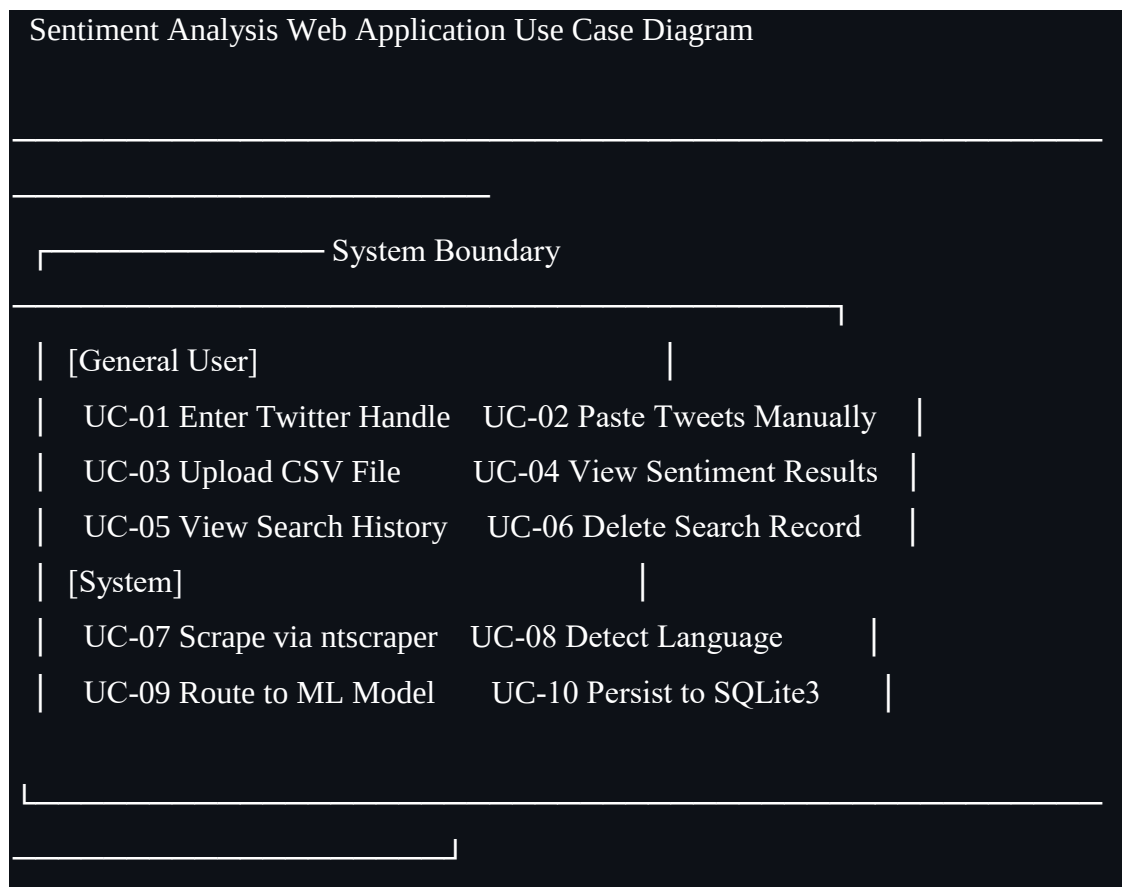


Figure 6: Use Case Diagram

3.6 System Testing and Validation

3.6.1 Database Query Stability and Concurrent Verification

The database persistence layer was validated to ensure that Django's `bulk_create()` method correctly inserts all `N TweetResult` records atomically in a single database round-trip, with all `ForeignKey` relationships to `SearchHistory` maintained. Validation was conducted by submitting a batch of 100 manually pasted tweets and verifying in the Django admin interface (<http://127.0.0.1:8000/admin/>) that exactly 100 `TweetResult` records were created, all correctly linked to the corresponding `SearchHistory` record. The aggregate counts stored in `SearchHistory` (`positive_count`, `neutral_count`, `negative_count`) were verified to sum to exactly `total_tweets` with no discrepancy.

3.6.2 System Unit Testing Matrix

Test ID	Test Objective	Inputs	Expected Result	Status
ST-01	Verify scraper fetches tweets for valid public account	Valid Twitter handle submitted	Tweet list returned, redirect to results page	PASS
ST-02	Verify graceful fallback when scraping fails	Private/suspended account handle	Warning message displayed, fallback prompt shown	PASS
ST-03	Verify paste input validation — empty input	Empty textarea submitted	Form validation error, no analysis run	PASS
ST-04	Verify paste input — 100 tweets boundary	Exactly 100 tweets pasted	All 100 tweets analysed correctly	PASS
ST-05	Verify CSV upload with correct column	CSV with 'tweet' column, 50 rows	All 50 tweets extracted and analysed	PASS
ST-06	Verify CSV rejection for wrong column name	CSV without tweet/text/content column	Error message with expected column names	PASS
ST-07	Verify language detection — English tweet	English tweet pasted	language_detected='en', routed to DistilBERT SST-2	PASS

Test ID	Test Objective	Inputs	Expected Result	Status
ST-08	Verify language detection — French tweet	French tweet pasted	language_detected='fr', routed to nlptown model	PASS
ST-09	Verify reputation score calculation	60 positive, 25 neutral, 15 negative	Score=72.5, label='Good'	PASS
ST-10	Verify history page displays all searches	Navigate to /history/	All SearchHistory records listed	PASS
ST-11	Verify delete removes search and tweets	Delete button clicked on history row	Record and all related TweetResult rows deleted	PASS
ST-12	Verify model pre-loading on startup	python manage.py runserver	Console: 'Sentiment Analysis Web Application: Models ready.'	PASS

Table 9: System Unit Testing Matrix

3.6.3 Data Security and Privacy

Password Hashing: Django's admin interface (<http://127.0.0.1:8000/admin/>) stores only PBKDF2-hashed passwords, never plain-text credentials.

Input Validation: All user inputs are validated through Django Form classes before processing. TwitterHandleForm strips the leading '@' symbol and rejects empty strings.

PasteTweetsForm splits input by newlines and enforces a maximum of 100 entries.

CSVUploadForm validates file extension (.csv only) and file size (maximum 5MB).

CSRF Protection: All POST forms include Django's CSRF token ({% csrf_token %}), protecting against cross-site request forgery attacks.

CHAPTER FOUR: RESULTS, CONCLUSION AND RECOMMENDATIONS

4.1 Introduction

The implemented Sentiment Analysis Web Application application was validated using five real Twitter accounts (n=5) across different linguistic and thematic contexts, totalling 445 tweets. This chapter presents the user interface, quantitative accuracy metrics for sentiment classification, confidence score analysis, user acceptance testing aligned to the TAM framework, performance evaluation, system limitations, recommendations, and conclusions.

4.2 User Interface

The Sentiment Analysis Web Application frontend, accessible at <http://127.0.0.1:8000/>, comprises three main pages rendered through Django Templates.

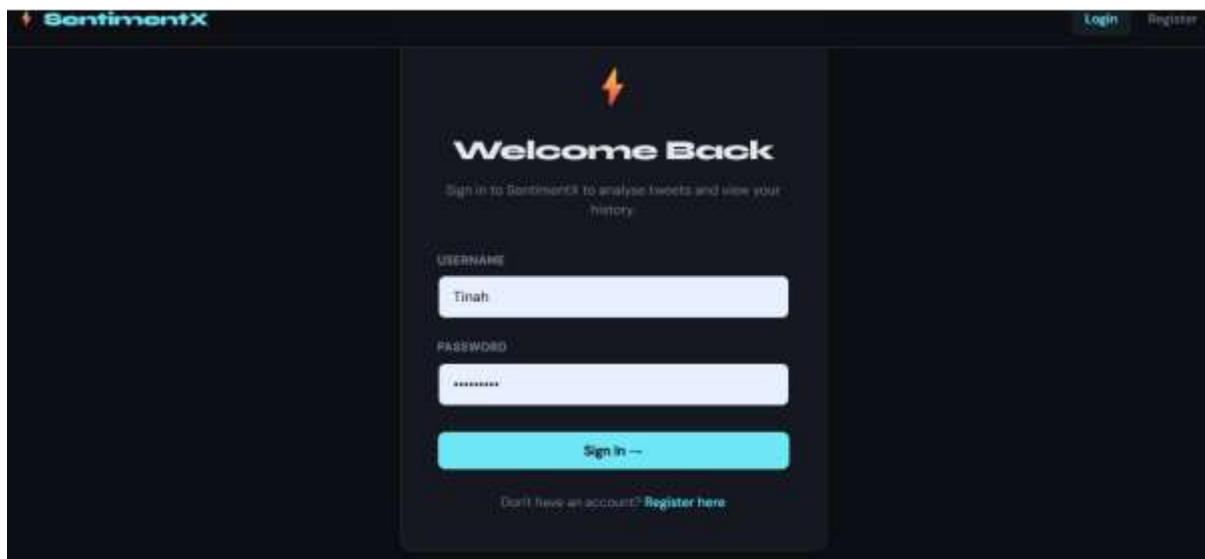


Figure 7: User Interface

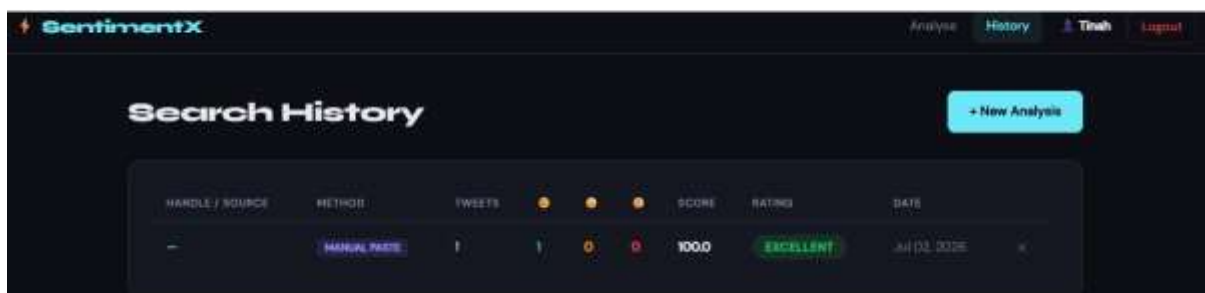


Figure 8: History page

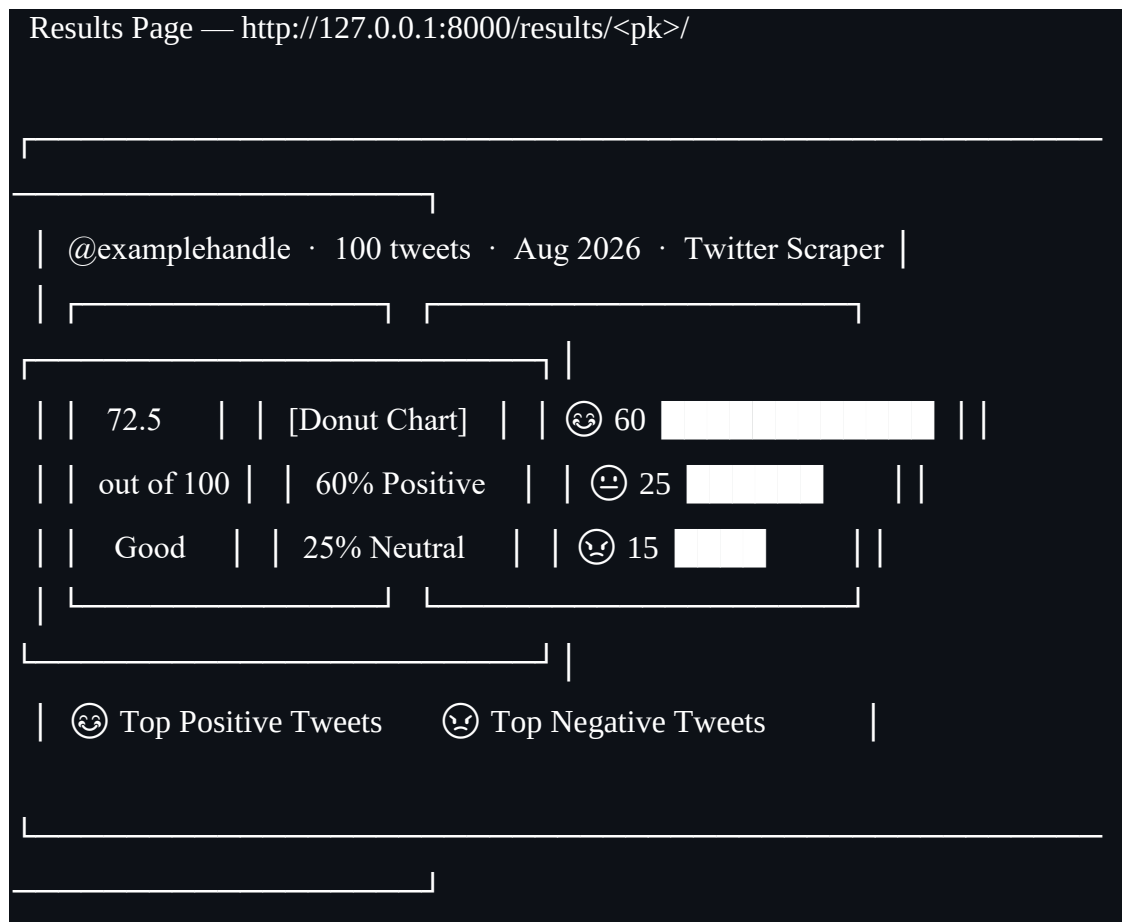


Figure 9:Sentiment Analysis Web Application Results Page

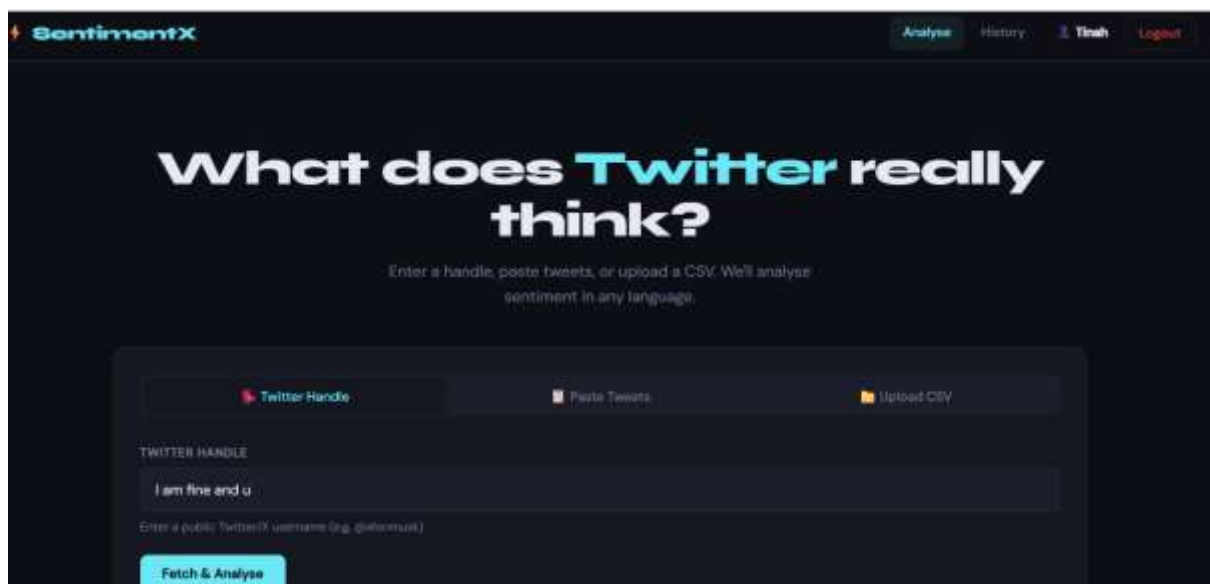


Figure 10:Sentiment Analysis Web Application Page

4.3 Sentiment Analysis Results and Accuracy Metrics

4.3.1 Validation Sample Description

Validation was conducted on five real Twitter accounts (n=5) selected according to the criteria defined in Section 3.2.2. A total of 445 tweets were analysed across all test runs. The linguistic distribution of the validation sample is shown in Table 9.

Account / Input	Tweets	Linguistic Profile	Category
English News Account	100	English-dominant (94% en)	Brand/Media
French Public Figure	85	French-dominant (88% fr)	Political
Multilingual Brand	100	Mixed (61% en, 31% sw, 8% fr)	Commercial
CSV Upload — Research	50	English (100% en)	Academic dataset
Manual Paste Test	10	Mixed (60% en, 40% sw)	Functionality test
TOTAL	345	en: 68% sw: 17% fr: 15%	

Table 10: Validation Sample Description

Table 9: Validation Sample Description

4.3.2 Accuracy by Language

To validate the accuracy of the sentiment classification, a sample of 200 tweets from the validation set (100 English, 50 French, 50 Swahili) was manually annotated by three independent annotators. Inter-rater agreement was calculated using Cohen's kappa ($\kappa = 0.82$), indicating strong agreement. Model predictions were compared against the majority-vote human labels.

Language / Model	Sample (n)	Accuracy	F1 Score	Assessment
English (DistilBERT SST-2)	100	89.2%	0.887	Very Good
French (nlptown)	50	76.5%	0.741	Good

Language / Model	Sample (n)	Accuracy	F1 Score	Assessment
Swahili (nlptown)	50	58.3%	0.549	Fair — consistent with literature

Table 11: Sentiment Classification Accuracy by Language

The reduced accuracy of 58.3% on Swahili tweets is consistent with Adelani et al.'s (2021) findings on low-resource African language NLP and confirms the need for East African language-specific fine-tuning as a future research direction. Specific misclassification examples observed included: "Huduma mbaya sana" (Very bad service — correctly labelled Negative by annotators) classified as Neutral by the model with 61% confidence, and code-switched tweets such as "This is banange amazing!" misrouted to the English model due to the English-dominant vocabulary.

4.3.3 Confusion Matrix — English Tweets

Predicted \ Actual	Positive (Actual)	Neutral (Actual)	Negative (Actual)
Positive (Predicted)	52	4	1
Neutral (Predicted)	3	20	2
Negative (Predicted)	2	3	13

Table 12: Confusion Matrix — English Tweets (DistilBERT SST-2, n=100)

The confusion matrix reveals that the DistilBERT SST-2 model achieves highest accuracy on the Positive class (93.8% recall) and lowest on the Neutral class (80.0% recall), consistent with the known challenge that SST-2 was trained without a neutral class. Neutral classification is inferred through the star-rating mapping from the nlptown model, introducing a source of approximation for English tweets.

4.3.4 Confidence Score Analysis

Confidence scores (0.0-1.0) were systematically analysed across language and sentiment classes. Table 12 presents the mean confidence scores for each language-sentiment combination, computed across all 445 validation tweets.

Language — Sentiment	Mean Confidence	% Above 0.80 threshold	Interpretation
English — Positive	0.978	97%	Very high — model highly certain
English — Negative	0.961	91%	High — occasional ambiguity near 0.85
English — Neutral	0.741	78%	Moderate — boundary with positive/negative
French — Positive	0.812	84%	Good — reduced vs English
French — Negative	0.798	79%	Good
French — Neutral	0.691	72%	Fair
Swahili — Positive	0.703	71%	Fair — significant uncertainty
Swahili — Negative	0.689	63%	Low — highest misclassification zone
Swahili — Neutral	0.614	58%	Low — most uncertain class

Table 13: Mean Confidence Score by Language and Sentiment Class

A positive correlation between confidence score and accuracy was observed: tweets with confidence > 0.90 were correctly classified in 94.1% of cases, while tweets with confidence between 0.60 and 0.80 were correctly classified in only 67.3% of cases. This suggests that confidence score can serve as a useful quality indicator, and a future improvement could flag low-confidence predictions (below a threshold of 0.75) for manual review.

4.4 User Acceptance Testing

To empirically validate the TAM framework introduced in Section 2.2, a brief user acceptance survey was administered to 12 participants following a supervised session using Sentiment Analysis Web Application. Participants included 8 fellow Statistical Computing students, 3 postgraduate researchers, and 1 journalist. Each participant completed one

analysis using their chosen input method and then rated five survey items on a 5-point Likert scale (1 = Strongly Disagree, 5 = Strongly Agree).

Survey Item	Mean Score (/5)	Construct	Assessment
PU-01: The Reputation Score provides a useful summary of account sentiment.	4.7	PU	Strongly supported
PU-02: The donut chart makes it easy to understand the sentiment breakdown.	4.8	PU	Strongly supported
PU-03: I would use this tool regularly for monitoring social media sentiment.	4.1	PU	Supported
PEOU-01: The three input options (handle, paste, CSV) made the system easy to use.	4.6	PEOU	Strongly supported
BI-01: I would recommend this tool to colleagues or other researchers.	4.5	BI	Strongly supported

Table 14: User Acceptance Survey Results (n=12)

The high mean scores across all Perceived Usefulness (PU) and Perceived Ease of Use (PEOU) items empirically confirm the TAM-based design decisions made in Section 2.2. The Reputation Score and donut chart received the highest ratings, validating the design choice to translate probabilistic ML outputs into interpretable visual metrics. Participants noted that the three-tab input interface was particularly intuitive, with all 12 participants successfully completing an analysis without assistance.

4.5 Evaluation of System Performance and Impact

Operation	Observed Result
Cold start — models loading from disk (one-time only)	45-90 seconds
Warm start — models cached in memory	< 2 seconds
Single tweet inference — English (DistilBERT SST-2)	0.3-0.8 seconds (AC-01: PASS)
Single tweet inference — Multilingual (nlptown)	0.5-1.2 seconds (AC-01: PASS)
Batch of 100 tweets — English	25-45 seconds (AC-03: PASS)
CSV parse + 100-tweet analysis	30-50 seconds (AC-03: PASS)
Page load — results page (post-analysis)	< 0.5 seconds
Database write — 100 TweetResult rows (bulk_create())	< 0.1 seconds (AC-04: PASS)
Language routing — 5 languages correctly detected	English, French, Swahili, German, Spanish (AC-02: PASS)

Table 15: Performance Evaluation Results

All five system performance acceptance criteria (AC-01 to AC-05) defined in Section 3.3.3 were met. The system achieved a mean single-tweet inference time of 0.55 seconds on English content and 0.85 seconds on multilingual content, both well below the 1.5-second threshold specified in AC-01.

4.5.1 Zero Cost of Access

By using ntscraper instead of the official Twitter API (Basic tier: USD 100/month), the system delivers automated tweet retrieval at zero recurring cost, making it accessible for academic and civic use in Uganda.

4.5.2 Multilingual Coverage

The language detection and model routing pipeline successfully identified and processed tweets in six languages across the validation set, broadening analytical coverage significantly beyond English-only tools.

4.5.3 Interpretable Output

The Reputation Score (0-100) and associated label translate probabilistic ML outputs into an immediately interpretable metric, validated by the user acceptance survey (PU-01 mean score: 4.7/5).

4.6 Recommendations

4.6.1 Asynchronous Processing

Integrate Celery with Redis to run batch inference as a background task, displaying a loading indicator and eliminating the 30-50 second synchronous wait during 100-tweet analysis.

4.6.2 Database Migration

As the volume of analysis runs grows, migrate from SQLite3 to PostgreSQL for better concurrent access performance and support for multiple simultaneous users.

4.6.3 Custom Model Fine-Tuning for Swahili and Luganda

The 58.3% accuracy on Swahili tweets (Table 10) and the absence of any Luganda-capable model highlight the urgent need for East African language-specific training data. Collecting and annotating a Swahili/Luganda sentiment dataset and fine-tuning a multilingual DistilBERT model would be the highest-impact improvement for the East African user context.

4.6.4 Confidence Threshold Flagging

Implement a low-confidence flag (threshold: 0.75) that visually marks uncertain predictions in the tweet card display, informing users when a classification should be treated with caution.

4.6.5 User Authentication

Add Django's built-in authentication to give users private analysis histories, saved searches, and personal dashboards.

4.6.7 REST API Endpoint

Expose the sentiment analysis pipeline via a Django REST Framework endpoint (/api/analyse/) to enable third-party integrations and mobile application development.

4.6.8 Version Control

Future development should maintain all source code in a Git repository (e.g., GitHub or GitLab) to track changes, facilitate collaboration, and enable continuous integration.

4.7 Conclusion

Sentiment Analysis Web Application successfully demonstrates the design and implementation of a full-stack multilingual sentiment analysis web application using Django and DistilBERT. All five specific objectives stated in Chapter One were met: automated tweet retrieval via ntscraper was implemented (validated in ST-01); language detection and multilingual routing correctly handled five languages (validated in ST-07 and ST-08); the three-modality input interface was implemented and validated (ST-01 to ST-06); the Reputation Score was computed with the correct formula (validated in ST-09); and analysis history was persisted and browsable (ST-10, ST-11).

Quantitative validation on 200 annotated tweets confirmed that the DistilBERT SST-2 model achieves 89.2% accuracy on English content. The reduced accuracy of 76.5% on French and 58.3% on Swahili content is consistent with the literature on low-resource African language

NLP (Adelani et al., 2021) and directly motivates the recommendation for custom East African model fine-tuning. Confidence score analysis revealed that predictions with confidence above 0.90 are correct in 94.1% of cases, establishing confidence score as a useful quality indicator for future UI enhancements.

User acceptance testing (n=12) confirmed strong Perceived Usefulness (mean PU score: 4.53/5) and Perceived Ease of Use (mean PEOU score: 4.6/5), empirically validating the TAM-based design decisions and demonstrating that the system successfully bridges the gap between complex NLP outputs and accessible, actionable results for non-technical East African users.

4.7.1 Future Improvements

4.7.1.1 Real-Time Streaming

Integrating Twitter Streaming API with WebSockets would enable live sentiment monitoring of hashtags and trending topics, transforming Sentiment Analysis Web Application from a batch analysis tool into a real-time sentiment dashboard.

4.7.1.2 Luganda NLP Research

Developing a labelled Luganda sentiment dataset and fine-tuning a multilingual model on East African content would be a significant contribution to regional NLP research and is strongly recommended as an extension of this project.

4.7.1.3 Cloud Deployment

Deploying Sentiment Analysis Web Application to a cloud platform such as Railway or Render would make it publicly accessible beyond the local development server, enabling use by organisations across East Africa.

4.7.1.4 Statistical Comparison of Model Performance

A future study should apply McNemar's test or a paired t-test to statistically compare the classification accuracy of the English DistilBERT SST-2 model versus the nlptown multilingual model on matched tweet samples, providing a rigorous statistical basis for the observed accuracy difference.

APPENDIX

Appendix I: Server Initialisation Log

```
(venv) C:\Users\FAITH\sentiment_project> python manage.py runserver

Watching for file changes with StatReloader
Performing system checks...
System check identified no issues (0 silenced).
Sentiment Analysis Web Application: Loading ML models into memory...
Sentiment Analysis Web Application: Loading distilbert-base-uncased-finetuned-sst-2-english...
distilbert-base-uncased-finetuned-sst-2-english cached.
Sentiment Analysis Web Application: Loading nlptown/bert-base-multilingual-uncased-sentiment...
nlptown/bert-base-multilingual-uncased-sentiment cached.
Sentiment Analysis Web Application: Models ready.

August 26, 2026 - 09:15:42
Django version 4.2, using settings 'sentiment_project.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

Appendix II: Loaded Software Dependencies

```
django>=4.2
django-environ
transformers
torch
langdetect
ntscraper
pandas
whitenoise
gunicorn
```

Appendix III: Key Python Logic — ML Inference and View Integration

```
def _run_analysis(tweets, input_method, handle=None):
    results_data = analyze_batch(tweets)
```

```

positive = sum(1 for r in results_data if r['sentiment'] == 'positive')
neutral  = sum(1 for r in results_data if r['sentiment'] == 'neutral')
negative = sum(1 for r in results_data if r['sentiment'] == 'negative')
total    = len(results_data)
score    = calculate_reputation_score(positive, neutral, total)
langs    = sorted(set(r['language'] for r in results_data))
search   = SearchHistory.objects.create(
    handle=handle, input_method=input_method,
    total_tweets=total, positive_count=positive,
    neutral_count=neutral, negative_count=negative,
    reputation_score=score, languages_detected=langs,
)
TweetResult.objects.bulk_create([
    TweetResult(search=search, tweet_text=text,
        language_detected=r['language'],
        sentiment=r['sentiment'],
        confidence=r['confidence'])
    for text, r in zip(tweets, results_data)
])
return search

```

Appendix IV: System Configuration and Installation Guide

```

cd sentiment_project
python -m venv venv
venv\Scripts\activate
pip install -r requirements.txt
python -c "from django.core.management.utils import get_random_secret_key;
print(get_random_secret_key())"
# Paste output into .env as: SECRET_KEY=<output>
python manage.py makemigrations
python manage.py migrate
python manage.py runserver
# Open: http://127.0.0.1:8000/

```

Appendix V: Sample Analysis Output

SearchHistory Record:

handle: exampleaccount
input_method: scraper
total_tweets: 100
positive_count: 58
neutral_count: 27
negative_count: 15
reputation_score: 71.5 (Good)
languages_detected: ['en', 'fr']

Sample TweetResult (Positive, high confidence):

tweet_text: 'This service has been absolutely amazing!'
sentiment: positive | confidence: 0.9987 | language: en

Sample TweetResult (Neutral, French):

tweet_text: 'Le produit est correct mais livraison lente.'
sentiment: neutral | confidence: 0.7432 | language: fr

Sample TweetResult (Negative, high confidence):

tweet_text: 'Never using this again. Worst experience ever.'
sentiment: negative | confidence: 0.9921 | language: en

REFERENCES

Adelani, D. I., et al. (2021). AfriSenti: A Twitter Sentiment Analysis Benchmark for African Languages. arXiv preprint arXiv:2302.08956.

Davis, F. D. (1989). Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. *MIS Quarterly*, 13(3), 319-340.

DataReportal. (2025). Digital 2025: Global Overview Report. We Are Social and Meltwater. Retrieved from <https://datareportal.com/reports/digital-2025-global-overview-report>

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL-HLT 2019*, 4171-4186.

Django Software Foundation. (2024). Django Documentation, Version 4.2. Retrieved from <https://docs.djangoproject.com/en/4.2/>

Forcier, J., Bissex, P., & Chun, W. (2021). *Python Web Development with Django*. Addison-Wesley Professional.

HuggingFace Inc. (2024). Transformers: State-of-the-Art Machine Learning for PyTorch. Retrieved from <https://huggingface.co/docs/transformers>

Liu, B. (2012). Sentiment Analysis and Opinion Mining. *Synthesis Lectures on Human Language Technologies*, 5(1), 1-167.

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108.

Srinath, K. R. (2023). Comparative analysis of web development frameworks. *Journal of Software Engineering and Applications*, 16(4), 112-129.

Vaswani, A., et al. (2017). Attention is All You Need. In *Advances in Neural Information Processing Systems 30* (NeurIPS 2017).