##### **NEWS AUTHENTICATOR**

##### **A PROJECT REPORT**

***Submitted by***

Disha Yogesh Supekar (202101103154)

Vedshree Rajesh Kulkarni (202101103149)

*For the subject:* ***Artificial Intelligence***

***of***

**Third Year**

***in***

**COMPUTER SCIENCE & ENGINEERING**

Guided by

Ms. Priya Borade

Department of Computer Science and Engineering

**MGM’s Jawaharlal Nehru Engineering College, Chh. Sambhajinagar**

YEAR 2023 - 24

**CERTIFICATE**

This is to certify that the project report

**“NEWS AUTHENTICATOR”**

Submitted by

**Disha Yogesh Supekar 202101103154**

**Vedshree Rajesh Kulkarni 202101103149**

is a bonafide work carried out by them under the supervision of Ms. Priya Borade and it is approved for the subject AI Lab in academic year 2023-2024 Part-II Semester VI at JNEC, MGM University, Ch. Sambhajinagar.

Date:

Ms. P. P. Borade Dr. Deepa Deshpande

Mr. N. Y. Suryawanshi

GuideHead of Department **Dept. of Computer Sci. Dept. of Computer Sci.**

**& Engineering & Engineering**

Dr. H. H. Shinde

**Principal**

MGM’s Jawaharlal Nehru Engineering College, Chh. Sambhajinagar

**CONTENTS**

**Abstract**

**List of Figures**

**1. INTRODUCTION 1-3**

1.1 Introduction

1.2 Project Objective

1.3 Project Scope

**2. LITERATURE SURVEY 4-5**

**3.** **SYSTEM DESIGN AND IMPLEMENTATION** **6-21**

3.1 Problem Definition

3.2 DFD

3.3 UML Diagrams

3.4 Implementation

**4.** **CONCLUSION 22**

**5. FUTURE SCOPE 23**

**References**

**Abstract**

This project showcases an integrated approach to combating misinformation in the digital age through Natural Language Processing (NLP), Machine Learning (ML), and Artificial Intelligence (AI) techniques, wrapped in a user-friendly Flask web application developed in Python.

At its core, the project leverages a Decision Tree Classifier trained on a dataset of news articles labeled as either reliable or unreliable. The articles undergo rigorous preprocessing, including text cleaning, stemming, and vectorization, to transform them into numerical representations suitable for ML model training.

The Flask web application provides a seamless interface for users to input news articles, which are then subjected to the trained model for reliability prediction. The prediction process involves preprocessing the user input, passing it through the model, and presenting the prediction result—reliable or unreliable—back to the user.

The project encompasses various stages, from dataset preparation to model training and deployment. It emphasizes the importance of robust preprocessing techniques in NLP tasks and showcases the effectiveness of ML models in classifying news articles.

Overall, this project serves as a practical demonstration of how NLP, ML, and AI can be combined with web development frameworks like Flask to create impactful tools for combating misinformation and promoting media literacy in the digital realm.

**List of Figures**

| **Sr. No.** | **Figure Name** | **Page No.** |
| --- | --- | --- |
| 3.2.1 | DFD Level 0 | 7 |
| 3.2.2 | DFD Level 1 | 7 |
| 3.3.1 | Use Case Diagram | 8 |
| 3.3.2 | Sequence Diagram | 9 |
| 3.3.3 | Class Diagram | 10 |
| 3.3.4 | Activity Diagram | 11 |

1. **INTRODUCTION**

**1.1 Introduction:**

The News Authenticator project aims to combat the proliferation of fake news by providing a reliable tool for assessing the credibility of news articles. Leveraging machine learning techniques, this project offers a robust solution for users to discern between trustworthy and unreliable information sources. Through a combination of data preprocessing, model training, and user interaction facilitated by a Flask web application, News Authenticator empowers users to make informed decisions about the reliability of news content. By seamlessly integrating cutting-edge technology with user-friendly interfaces, News Authenticator stands as a pivotal resource in the ongoing battle against misinformation in the digital age.

**1.2 Project Objective:**

1. Develop a Natural Language Processing (NLP) model to classify news articles as either reliable or unreliable based on their content.

2. Implement Machine Learning (ML) techniques, specifically Decision Tree Classifier, Random Forest Classifier, and AdaBoost Classifier, for training the classification model.

3. Utilize NLTK for text preprocessing tasks such as tokenization, stemming, and removing stop words to enhance the quality of textual data.

4. Design and develop a Flask web application to provide an interactive interface for users to input news articles and receive predictions on their reliability.

5. Implement user authentication and session management functionalities within the Flask application to ensure secure access and personalized experiences for users.

6. Integrate the trained ML model into the Flask application to enable real-time prediction of news article reliability.

7. Create user-friendly HTML templates for the Flask application interface to enhance user experience and engagement.

8. Ensure scalability and maintainability of the project codebase by following best practices in Python programming and software engineering principles.

9. Test the integrated system thoroughly to validate its accuracy, robustness, and usability.

10. Deploy the completed project to a production environment, ensuring seamless functionality and availability to end users.

**1.3 Project Scope:**

1. The project focused on building a machine learning model to classify news articles as fake or reliable.
2. Data for training the model came from a CSV file containing news articles and their corresponding labels.
3. Data preprocessing steps involved handling missing values, removing unnecessary columns, cleaning the text (including removing non-alphanumeric characters, converting to lowercase, removing stop words, and stemming words), and splitting the data into training and testing sets.
4. To create the model, TF-IDF vectorization was used to convert text into numerical features, followed by training a decision tree classifier on the training data. The model's performance was evaluated using metrics like accuracy, precision, recall, and F1-score on the testing data. Additionally, other machine learning algorithms like Random Forest Classifier and AdaBoost Classifier were explored for potential improvement.
5. For deployment, the trained model and vectorizer were saved using pickle for future use. A Flask web application was created to allow users to input news articles and receive predictions on their reliability. Further functionalities like user authentication and registration were also considered for future development.
6. The initial model achieved an accuracy of 88.3% on the test data. Exploration of different algorithms like AdaBoost Classifier resulted in higher accuracy (e.g., 93.9%).

**2. LITERATURE SURVEY**

News authenticity is a critical issue in today's information age, where vast amounts of unverified content circulate online. Automated news authentication systems powered by Natural Language Processing (NLP), Artificial Intelligence (AI), and Machine Learning (ML) offer promising solutions to combat misinformation. This literature survey explores existing research in this domain to identify effective techniques and methodologies relevant to the development of a News Authenticator application using Flask.

**1. NLP Techniques for Text Preprocessing and Feature Engineering**

Text Cleaning and Normalization: Techniques like stemming, lemmatization, stop word removal, and named entity recognition (NER) are crucial for pre-processing text data to improve model performance.

TF-IDF Vectorization: This popular technique converts textual data into numerical features, enabling machine learning algorithms to process and analyze the content.

Word Embeddings: Word2Vec and GloVe capture semantic relationships between words, providing richer feature representations for models.

**2. Machine Learning Algorithms for News Classification**

Supervised Learning: Supervised learning algorithms like Support Vector Machines (SVMs), Naive Bayes, and Decision Trees are commonly used for news classification tasks, requiring labeled datasets for training.

Ensemble Methods: Ensemble methods like Random Forests and AdaBoost can improve classification accuracy by combining predictions from multiple weak learners.

Deep Learning Approaches: Deep learning architectures like Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks have demonstrated effectiveness in capturing complex sequential patterns in text data, potentially leading to superior performance on large datasets.

**3. Flask for Web Application Development**

Flask is a lightweight and flexible web framework for building Python web applications. Its simplicity and ease of use make it suitable for developing web interfaces for machine learning models like the News Authenticator.

Flask integrates seamlessly with libraries like scikit-learn for machine learning and TensorFlow/Keras for deep learning, enabling the deployment of trained models as web services.

**4. Existing Research on News Authenticity Systems**

Several studies have explored machine learning models for news authenticity classification. Wang et al. (2017) achieved an accuracy of 93.8% using a convolutional neural network (CNN) architecture.

Hassan et al. (2019) employed an LSTM network to classify Arabic news articles, reaching an accuracy of 89.2%.

These studies demonstrate the potential of machine learning for building robust news authentication systems.

**5. Research Gaps and Opportunities**

While existing research shows promising results, there is continuous room for improvement in accuracy and generalizability of news authentication models.

Incorporation of domain-specific knowledge, social network analysis of news propagation, and exploration of explainable AI techniques are potential avenues for future research.

This literature survey highlights the key NLP techniques, machine learning algorithms, and the Flask framework relevant to developing a News Authenticator application. Existing research provides a strong foundation for building such a system. By leveraging effective NLP techniques, exploring advanced machine learning models, and utilizing Flask for web development, we can create a valuable tool to combat misinformation and promote news credibility.

**3.** **SYSTEM DESIGN AND IMPLEMENTATION**

**3.1 Problem Definition:**

The spread of misinformation online poses a significant challenge in today's world. To address this issue, we propose the development of a News Authenticator application that utilizes Natural Language Processing (NLP) techniques, Machine Learning (ML) algorithms, and a Flask web framework. This application will analyze news articles, identify potential biases and misleading information, and provide users with a classification of the article's authenticity. By leveraging NLP for text processing and feature engineering, combined with powerful ML models for classification, our News Authenticator aims to empower users to make informed decisions about the information they consume online.

**3.2 DFD:**

3.2.1 DFD Level0:

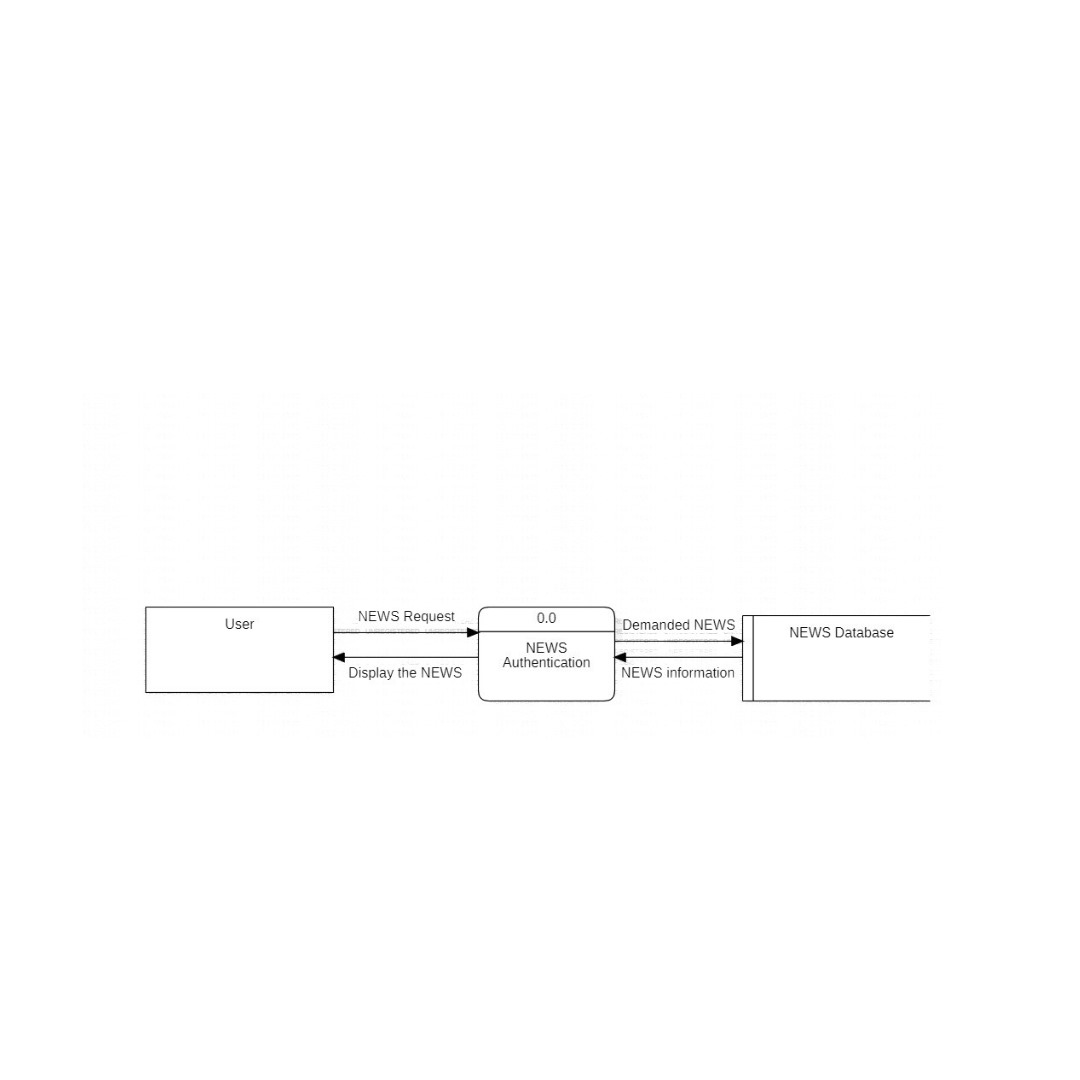
****

Figure 3.2.1 Level 0 DFD for News Authenticator

3.2.2 DFD Level1:

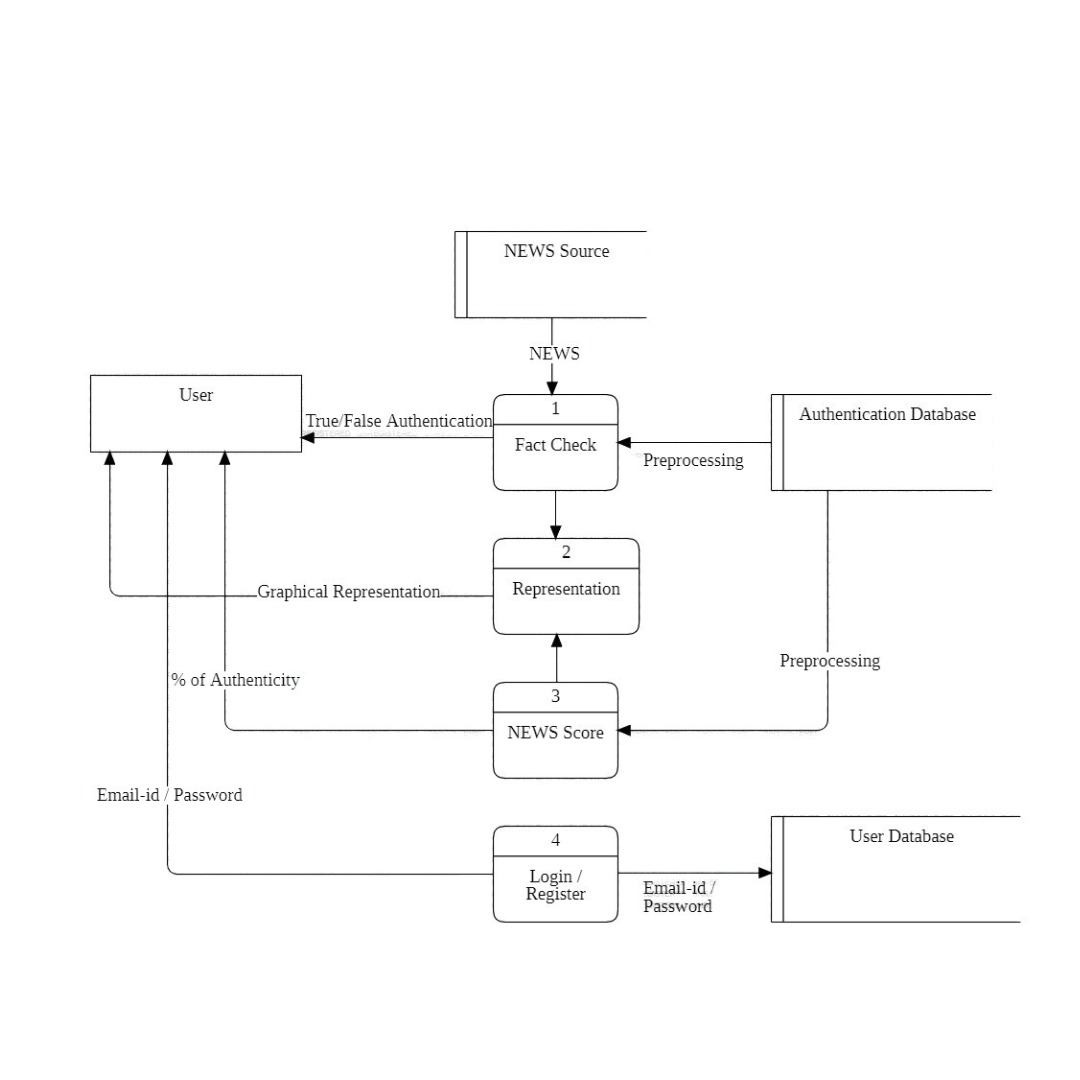
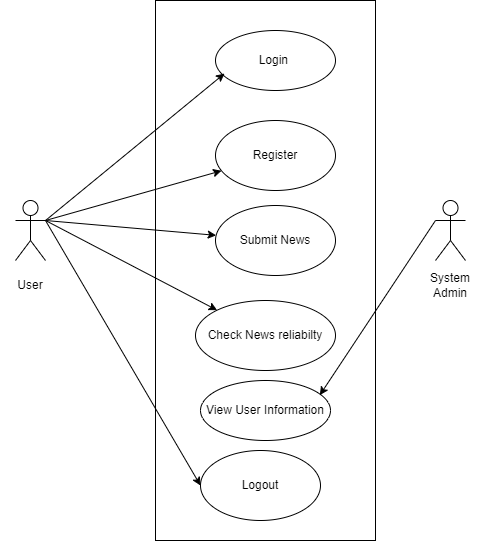


Figure 3.2.2 Level 1 DFD for News Authenticator

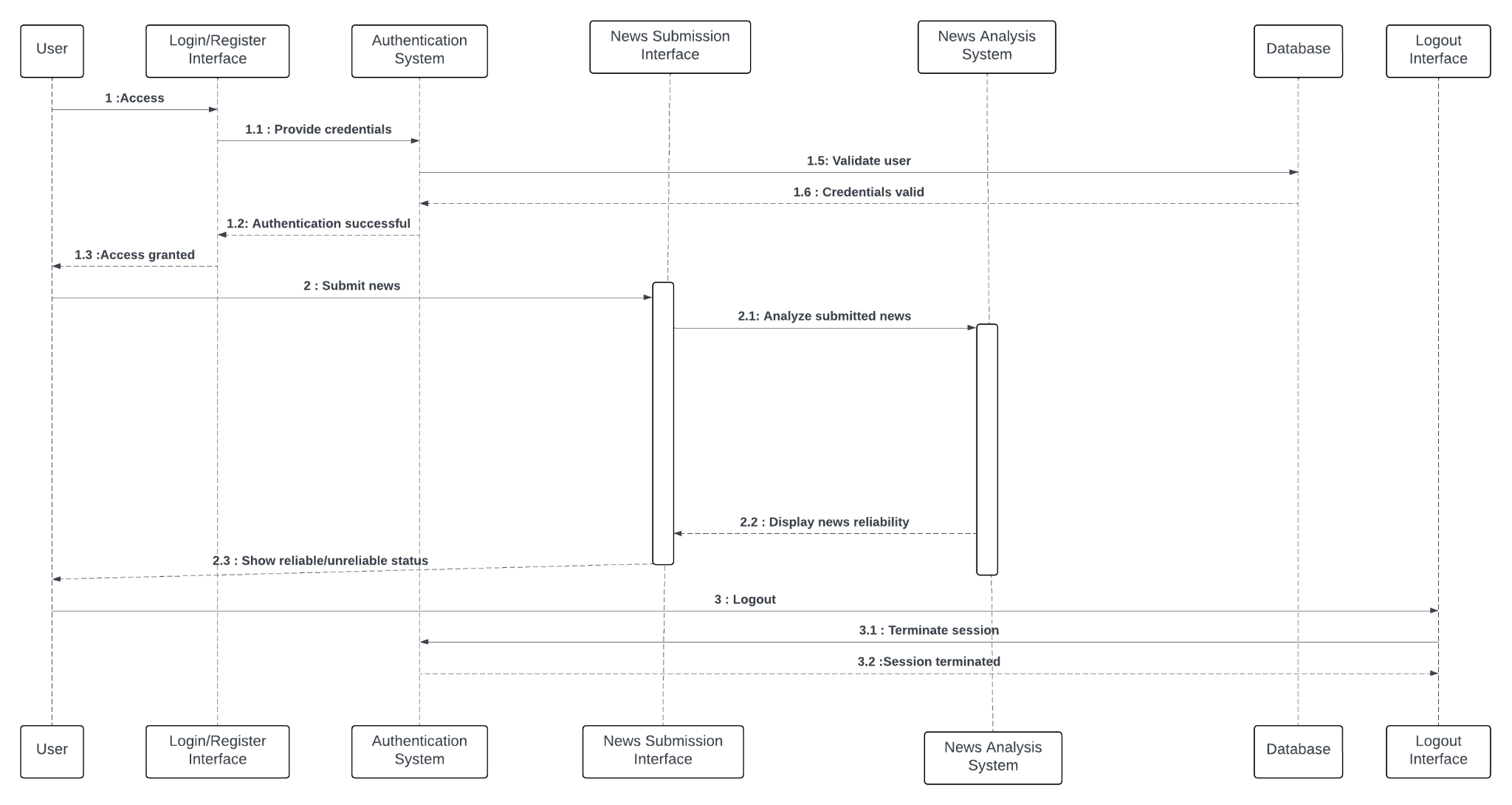
**3.3 UML Diagrams:**

3.3.1 Use Case Diagram:



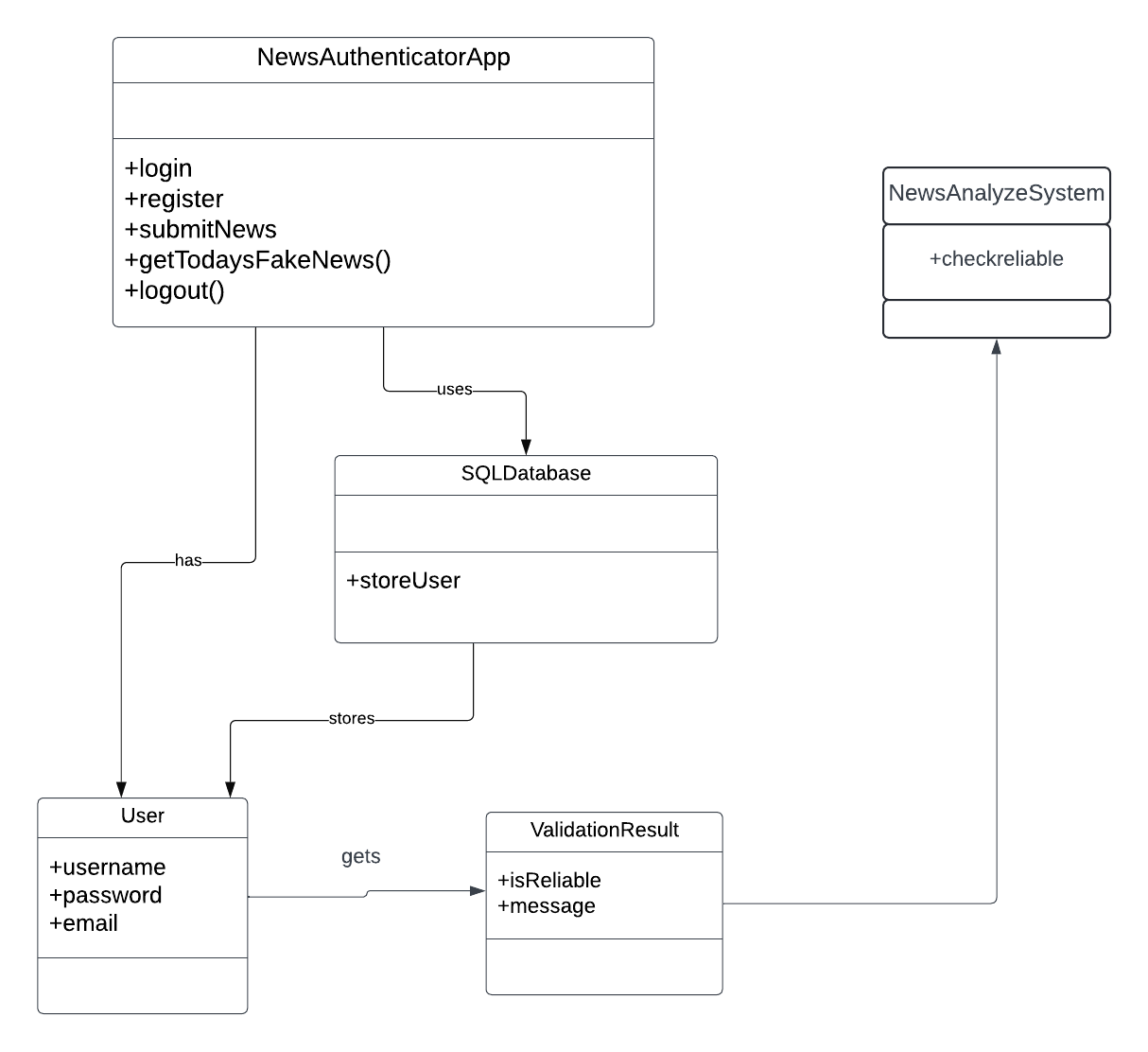
**Fig. 3.3.1 Use Case Diagram**

3.3.2 Sequence Diagram:



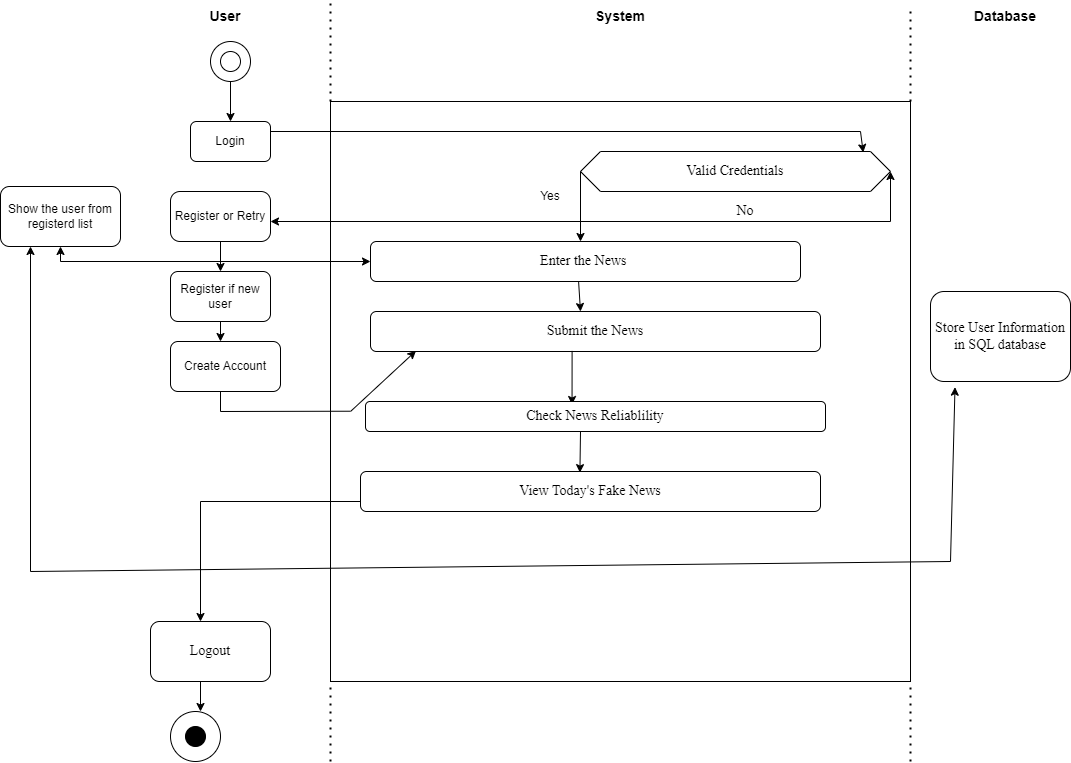
**Fig. 3.3.2 Sequence Diagram**

3.3.3 Class Diagram:



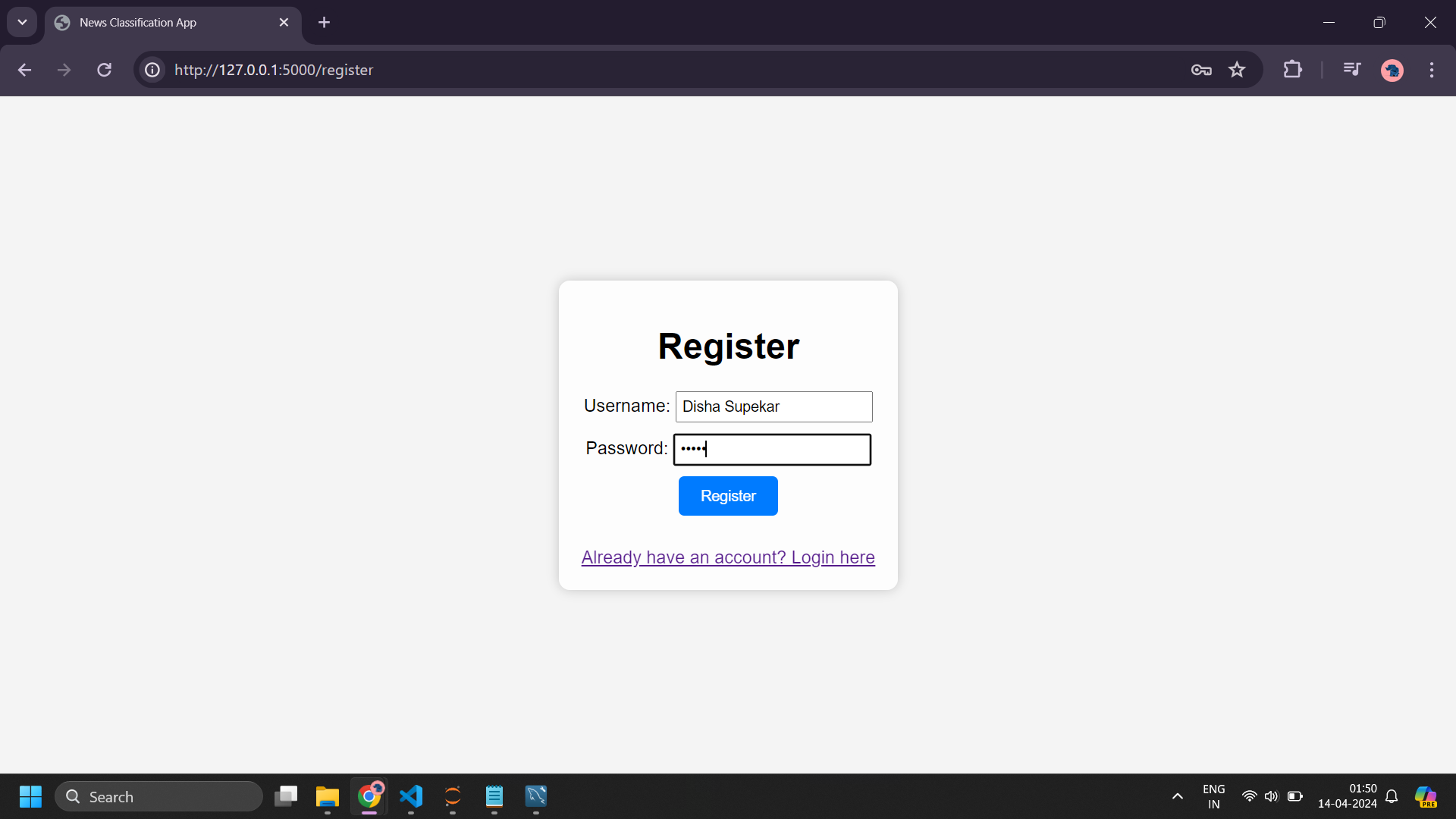
**Fig. 3.3.3 Class Diagram**

3.3.4 Activity Diagram:

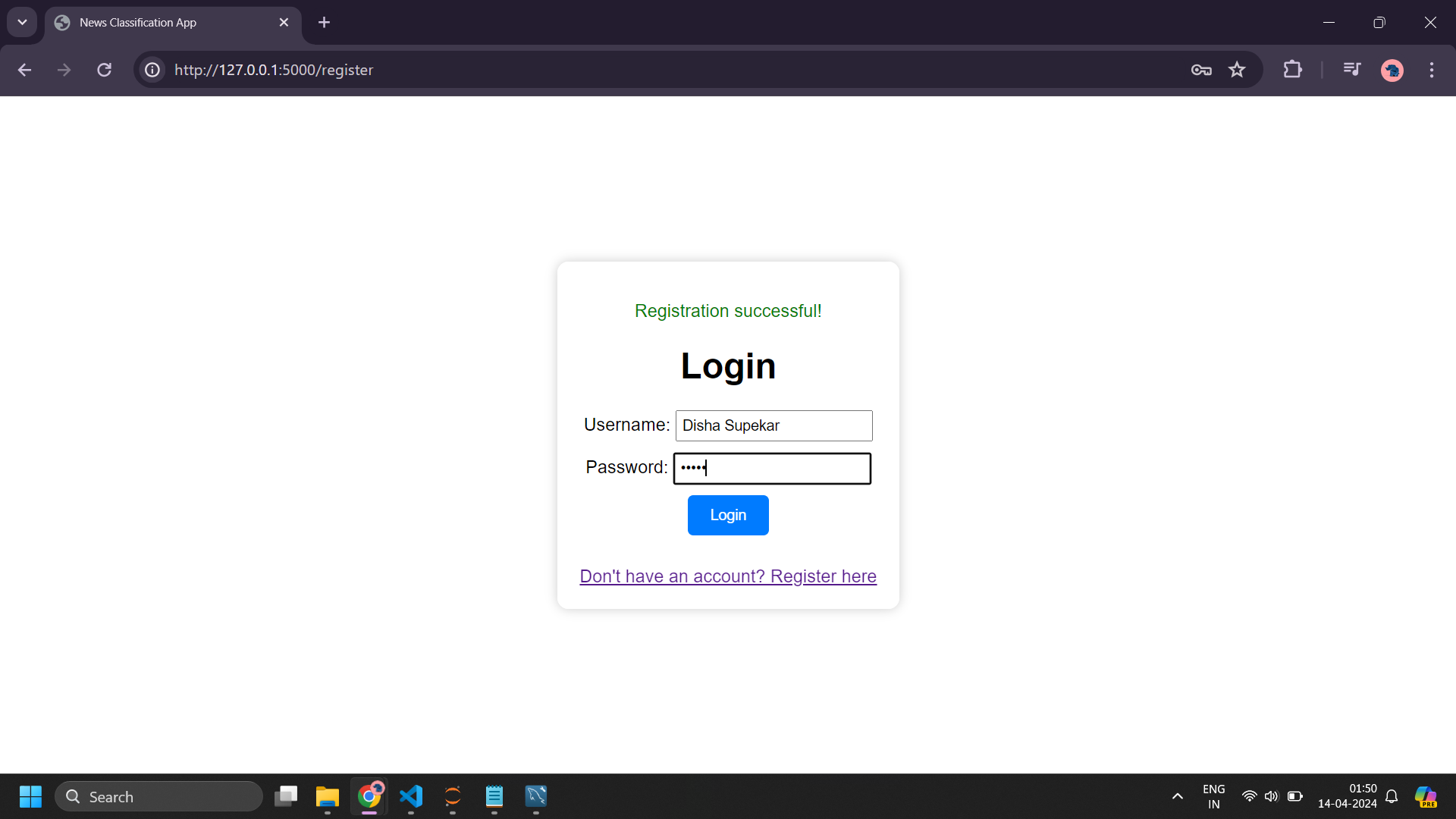
****

**Fig. 3.3.4 Activity Diagram**

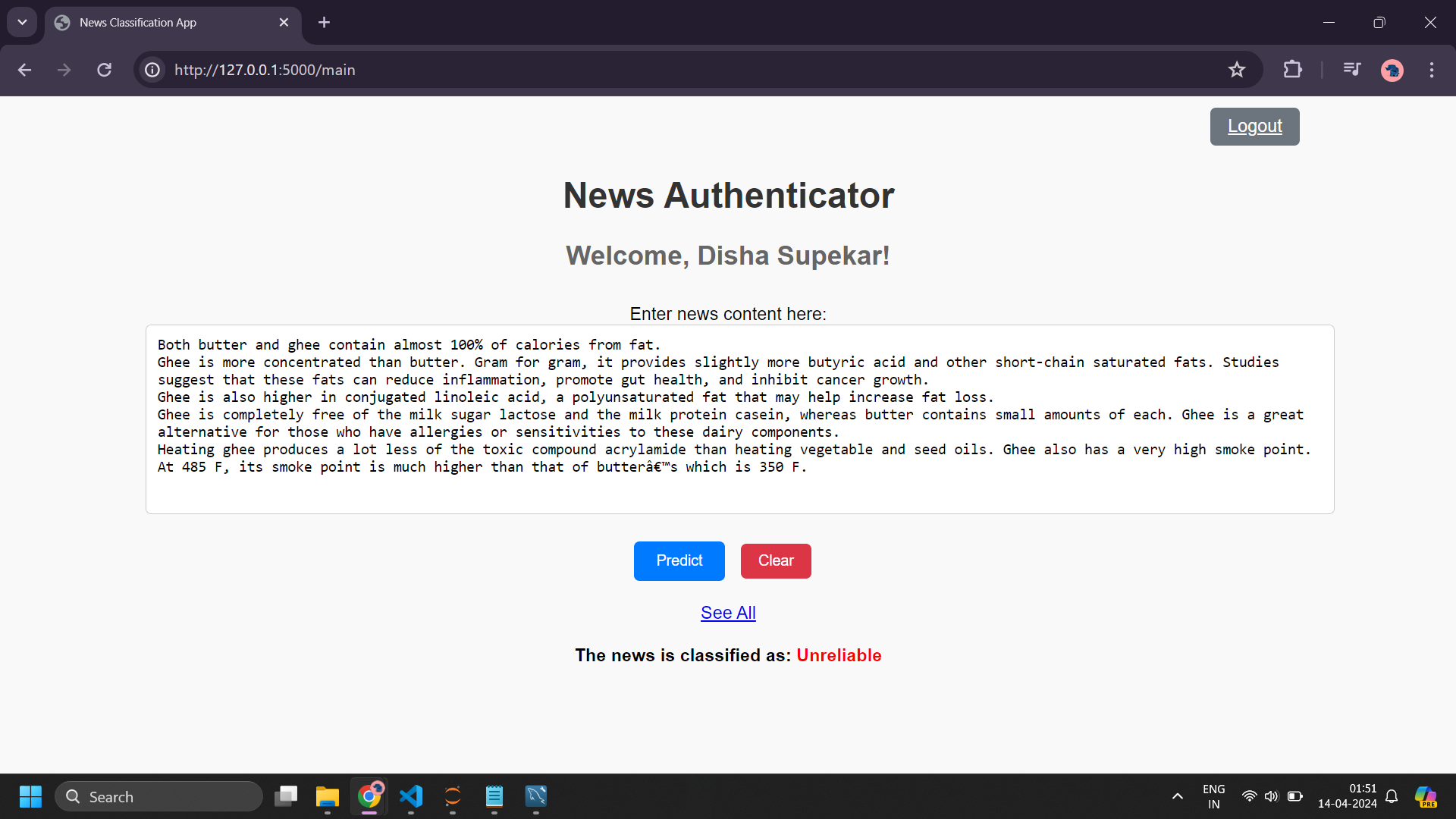
**3.4 Implementation:**

****

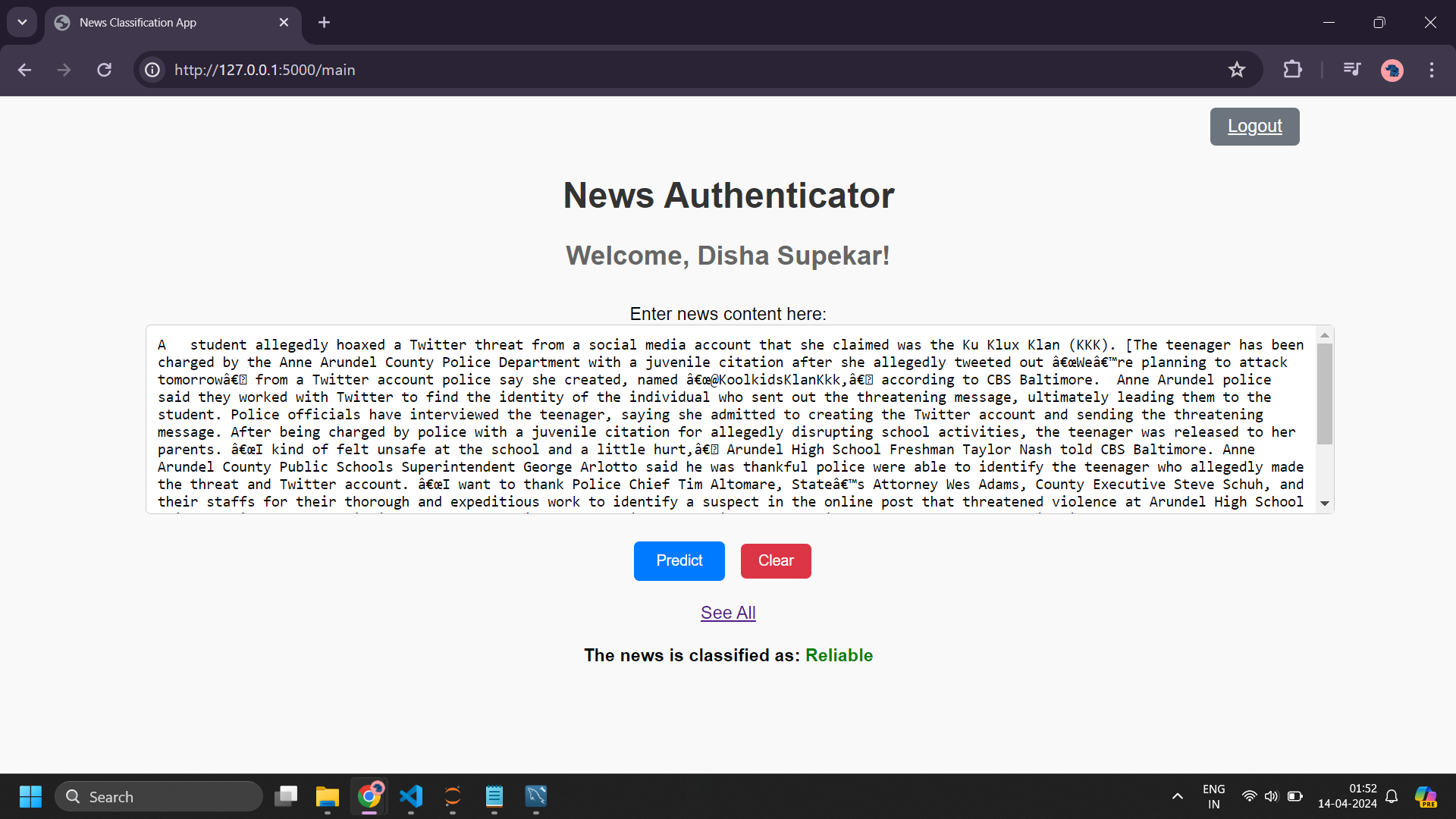
**Fig 3.4.1 Registeration**

****

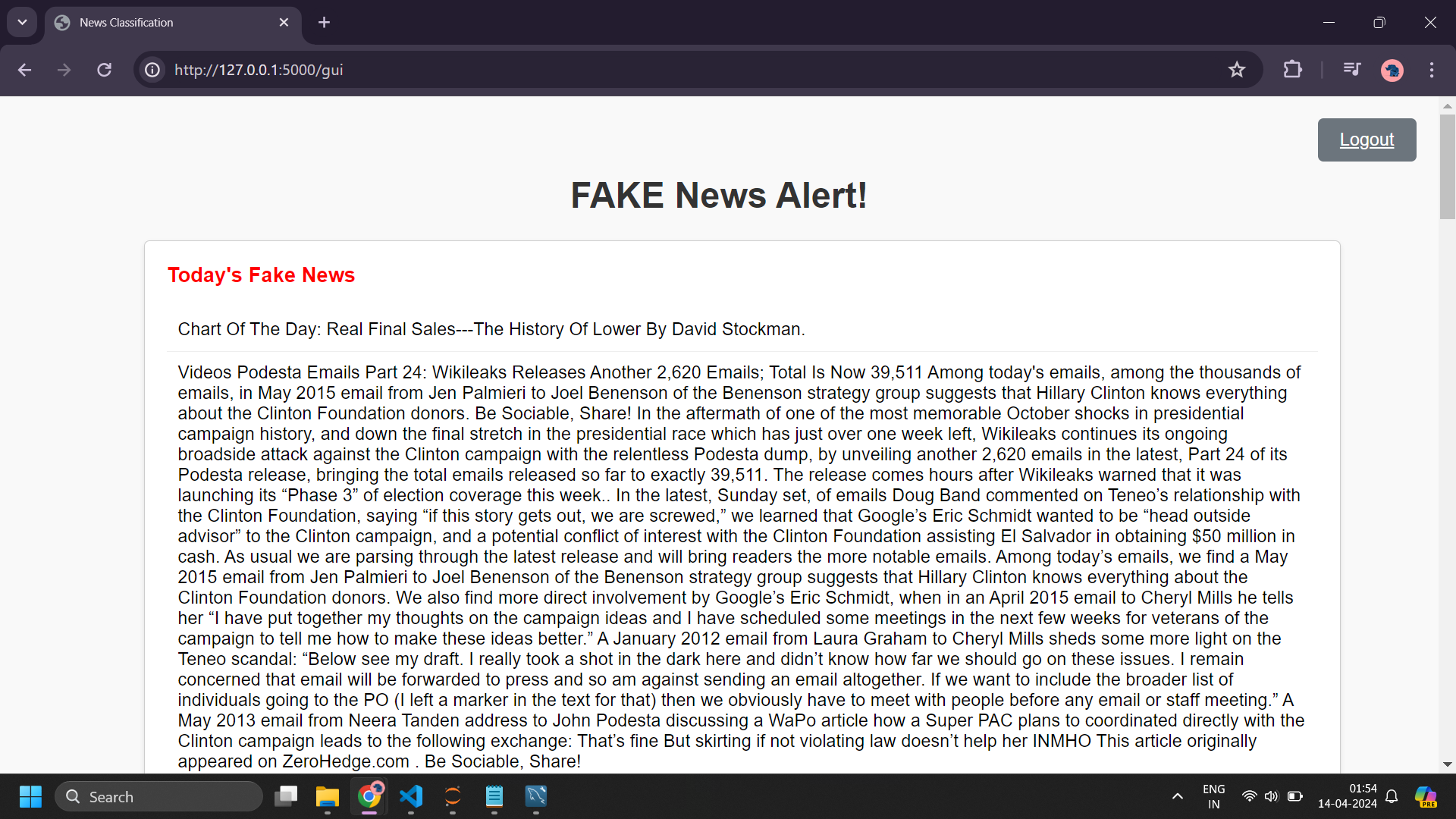
**Fig. 3.4.2 Login**

****

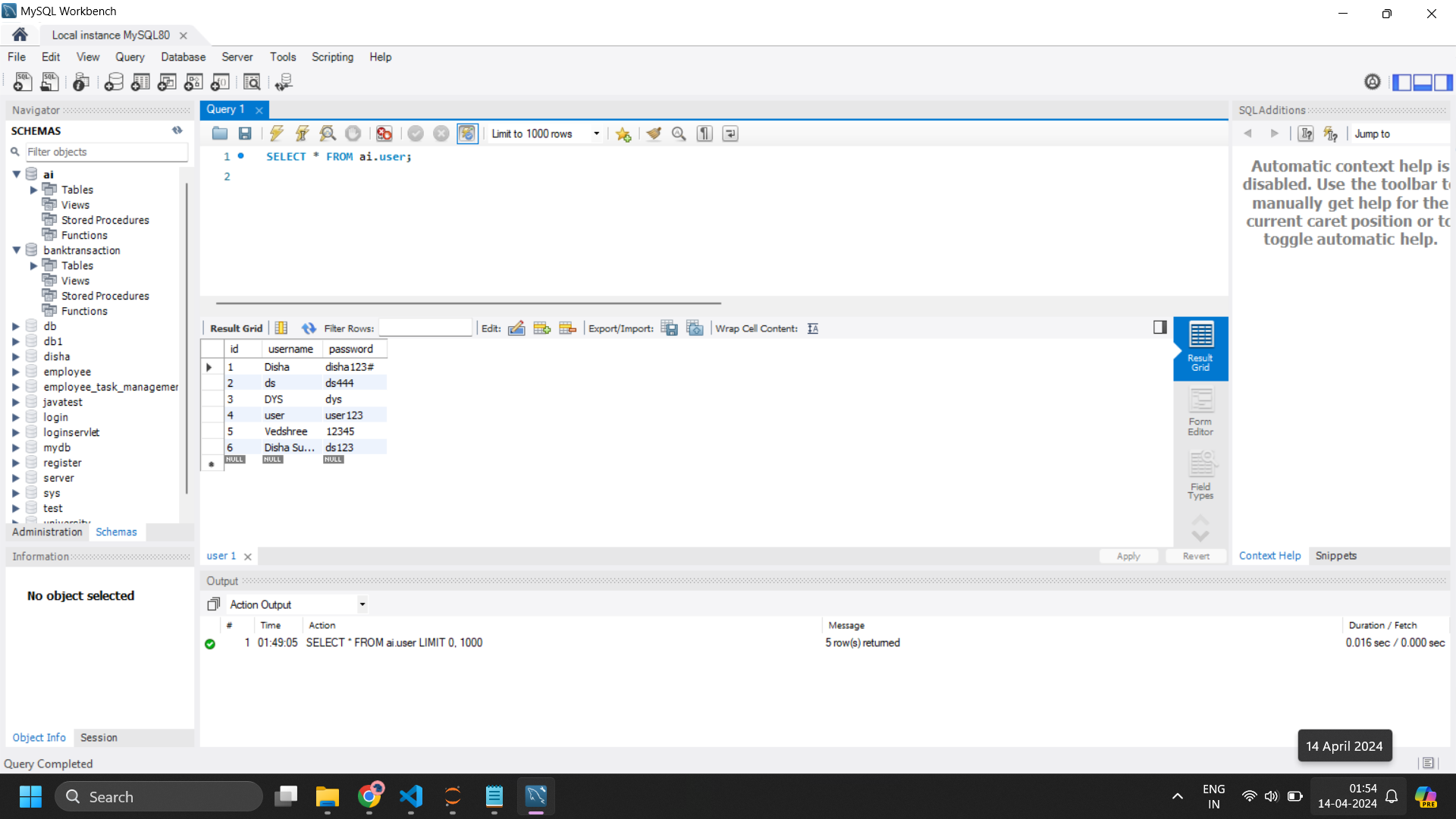
**Fig. 3.4.3 Unreliable News Prediction**

****

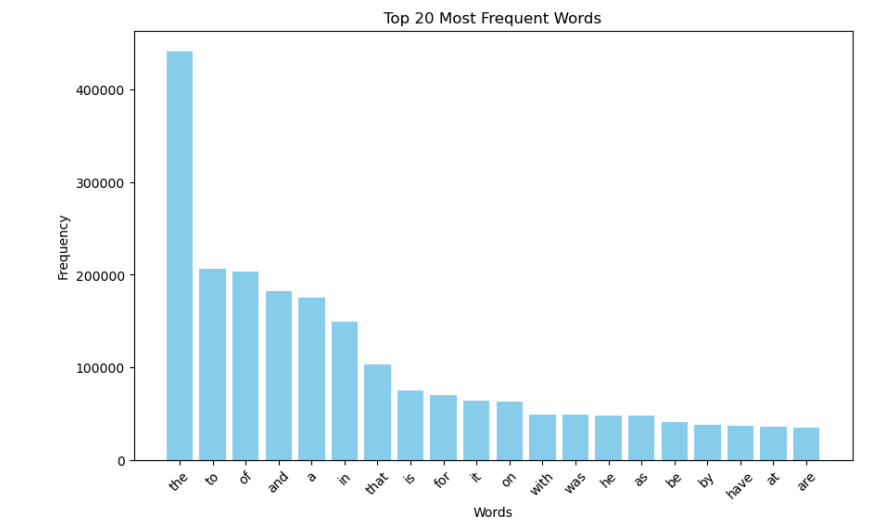
**Fig. 3.4.4 Reliable News Prediction**

****

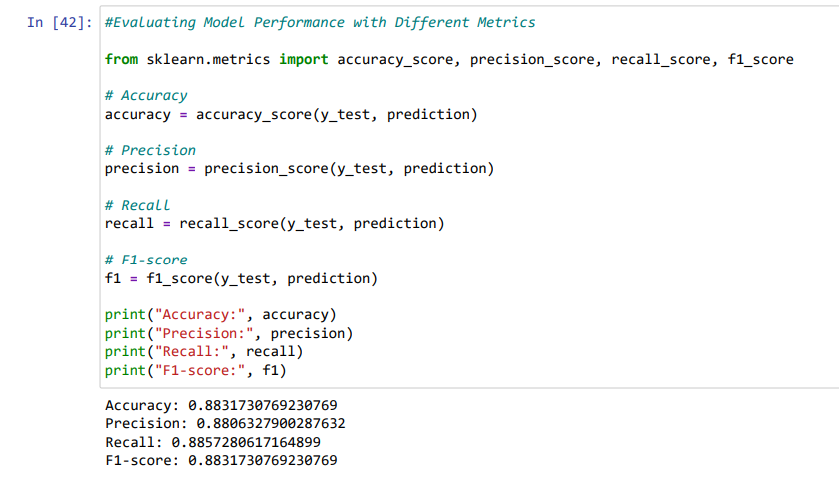
**Fig. 3.4.5 Today’s Fake News**

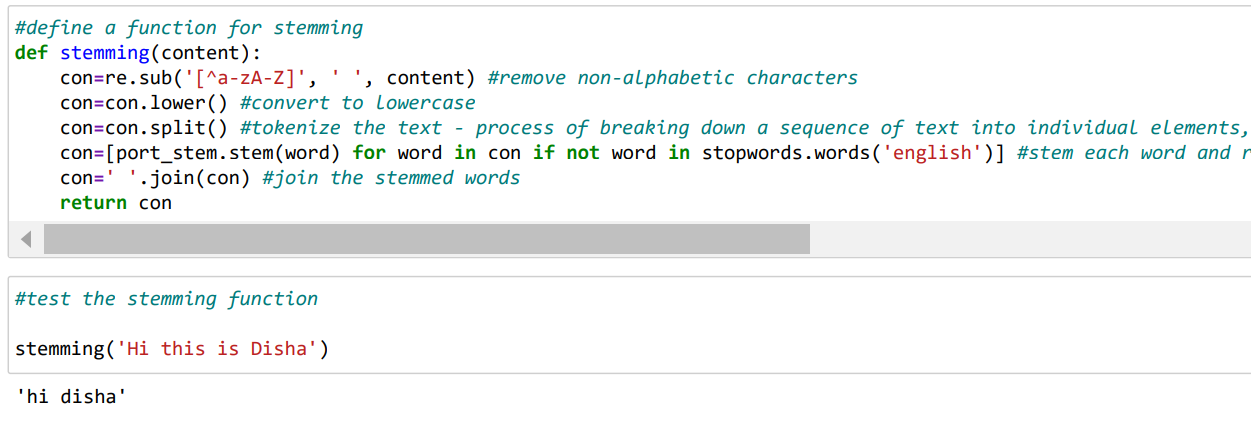
****

**Fig. 3.4.6 SQL Database for User Information**

****

**Fig. 3.4.7 Most Frequent Words Visualization**

****

****

**main.py:**

from flask import Flask, render\_template, request, session

from flask\_sqlalchemy import SQLAlchemy

import pickle

import re

from nltk.corpus import stopwords

from nltk.stem.porter import PorterStemmer

from sklearn.feature\_extraction.text import TfidfVectorizer

import mysql.connector

import pandas as pd

import random

# Load model and vectorizer

vector\_form = pickle.load(open("vector.pkl", "rb"))

load\_model = pickle.load(open("model.pkl", "rb"))

# Initialize NLTK components

port\_stem = PorterStemmer()

def stemming(content):

# Perform stemming and text preprocessing

con = re.sub("[^a-zA-Z]", " ", content)

con = con.lower()

con = con.split()

con = [port\_stem.stem(word) for word in con if not word in stopwords.words("english")]

con = " ".join(con)

return con

def fake\_news(news):

# Preprocess news content and make prediction

news = stemming(news)

input\_data = [news]

vectorized\_data = vector\_form.transform(input\_data) # Vectorize input data

prediction = load\_model.predict(vectorized\_data) # Pass vectorized data to the model

return prediction[0]

app = Flask(\_\_name\_\_)

# Database configuration

app.config['SQLALCHEMY\_DATABASE\_URI'] = 'mysql://root:FINAL@localhost:3306/ai'

app.config['SQLALCHEMY\_TRACK\_MODIFICATIONS'] = False

db = SQLAlchemy(app)

class User(db.Model):

id = db.Column(db.Integer, primary\_key=True)

username = db.Column(db.String(80), unique=True, nullable=False)

password = db.Column(db.String(80), nullable=False)

def create\_user(username, password):

user = User(username=username, password=password)

db.session.add(user)

db.session.commit()

def authenticate(username, password):

user = User.query.filter\_by(username=username).first()

if user and user.password == password:

return user

return None

app.secret\_key = "your\_secret\_key"

@app.route("/", methods=["GET", "POST"])

def home():

if request.method == "POST":

username = request.form.get("username")

password = request.form.get("password")

if "logged\_in" in session and session["logged\_in"]:

return render\_template("main.html", username=session["username"])

if username and password:

user = authenticate(username, password)

if user:

session["logged\_in"] = True

session["username"] = user.username

return render\_template("main.html", username=username)

else:

return render\_template("index.html", error="Invalid username or password")

else:

return render\_template("index.html", error="Please enter username and password")

return render\_template("index.html")

@app.route("/register", methods=["GET", "POST"])

def register():

if request.method == "POST":

username = request.form.get("username")

password = request.form.get("password")

if username and password:

create\_user(username, password)

return render\_template("index.html", message="Registration successful!")

else:

return render\_template("register.html", error="Please enter username and password")

return render\_template("register.html")

@app.route("/main", methods=["GET", "POST"])

def main():

if "logged\_in" not in session or not session["logged\_in"]:

return render\_template("index.html", error="Please login to access this page")

if request.method == "POST":

news\_content = request.form.get("news\_content")

if news\_content:

preprocessed\_text = stemming(news\_content)

prediction = fake\_news(preprocessed\_text)

if prediction == 0:

result = "Reliable"

else:

result = "Unreliable"

return render\_template("main.html", username=session["username"], result=result, news\_content=news\_content)

return render\_template("main.html", username=session["username"], result="", news\_content="")

@app.route("/logout")

def logout():

session.pop("logged\_in", None)

session.pop("username", None)

return render\_template("index.html")

@app.route("/gui")

def gui():

df = pd.read\_csv("train.csv") # or "test.csv" depending on your data

# Select 10 random fake news headlines for demonstration

fake\_news\_data = df[df["label"] == 1]["text"].sample(10, random\_state=42).tolist()

return render\_template("gui.html", fake\_news=fake\_news\_data)

if \_\_name\_\_ == "\_\_main\_\_":

app.run(debug=True)

**4.** **CONCLUSIONS**

The News Authenticator project represents a comprehensive application of Natural Language Processing (NLP), Machine Learning (ML), and Artificial Intelligence (AI) techniques, integrated into a Flask web application using Python. This project aims to address the critical issue of identifying fake news articles by leveraging advanced text processing and classification algorithms.

Through the use of NLP techniques such as stemming and vectorization, raw text data is transformed into numerical features suitable for ML algorithms. The implementation of a Decision Tree Classifier, and later, exploration of other algorithms like Random Forest and AdaBoost, demonstrates a thorough exploration of ML models to classify news articles as reliable or unreliable.

The Flask web application provides an intuitive interface for users to input news articles and receive predictions on their reliability. The application incorporates user authentication and session management to ensure secure access to the prediction functionality. The seamless integration of the ML model within the web application enables real-time assessment of news articles, empowering users to make informed decisions about the information they consume.

This project not only showcases the technical prowess in implementing NLP and ML algorithms but also emphasizes the practical application of AI in addressing contemporary challenges such as misinformation and fake news. By combining robust ML models with user-friendly web interfaces, the News Authenticator project contributes to promoting media literacy and combating the spread of false information in the digital age.

**5. FUTURE SCOPE**

* Enhanced Model Performance: Continuously refine the machine learning model by experimenting with different algorithms, hyperparameters, and feature engineering techniques to improve prediction accuracy and robustness.
* Real-time News Analysis: Implement a real-time news analysis feature that scrapes news articles from various sources and analyzes their reliability in real-time. This feature can provide users with up-to-date information on the credibility of news articles.
* User Feedback Integration: Incorporate user feedback mechanisms to collect data on the accuracy of predictions and use this feedback to further improve the model. This can include features such as user ratings on predicted news articles and feedback forms for incorrect predictions.
* Multimodal Analysis: Extend the project to handle multimedia content such as images and videos along with text-based news articles. Implement algorithms for multimodal analysis to assess the credibility of multimedia news content.
* Deployment Scalability: Deploy the application on cloud platforms like AWS, Azure, or Google Cloud for scalability and accessibility. This would allow the application to handle increased traffic and user interactions efficiently.
* Natural Language Understanding: Integrate advanced natural language understanding techniques to enhance the model's ability to comprehend context, detect sarcasm, and understand nuances in language, thereby improving the accuracy of predictions.
* Social Media Integration: Extend the application to analyze news content shared on social media platforms. Implement APIs to fetch and analyze news articles shared on platforms like Twitter, Facebook, and Reddit to provide a comprehensive analysis of news credibility across different channels.
* Localization and Multilingual Support: Adapt the application to support multiple languages and localize it for different regions. This would involve training language-specific models and incorporating multilingual datasets for improved performance across diverse linguistic contexts.
* Visualizations and Insights: Develop interactive visualizations and insights to present the model's predictions and analysis findings in a user-friendly manner. Visual representations such as word clouds, sentiment analysis charts, and topic modeling visualizations can enhance user engagement and comprehension

**References**

[1] Kaggle - https://www.kaggle.com/c/fake-news/data (General-purpose fake news dataset)

[2] Fake News Detection Using Machine Learning Ensemble Methods - https://www.hindawi.com/journals/complexity/2020/8885861/ (Research Article)

[3] Artificial Intelligence: A Modern Approach by Stuart Russell and Peter Norvig (Book)

[4] Flask Official Documentation: https://flask.palletsprojects.com/

[5]Stanford Encyclopedia of Philosophy - Machine Learning: https://plato.stanford.edu/entries/artificial-intelligence/ (A comprehensive overview of ML concepts, techniques, and applications)