****

**Name:**

**Zeeshan Haider**

**Class:**

**065**

**PAI**

**PROJECT**

# ****Fake News Detection Web App****

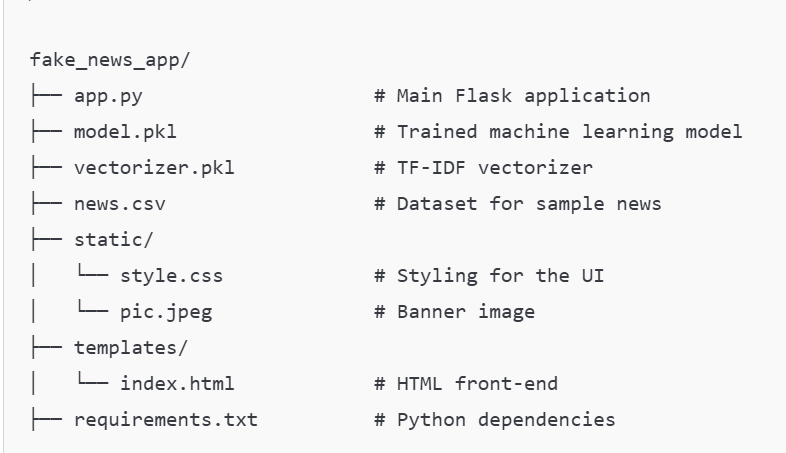
## ****1. Introduction****

The **Fake News Detection** project is a machine learning-based web application designed to classify news articles as real or fake. Built using Python, Flask, and a machine learning pipeline, the app enables users to input news text or load sample headlines and receive instant predictions. The primary goal is to address misinformation by identifying potentially fabricated news using natural language processing (NLP) techniques.

**2. Tools & Technologies**

* **Language:** Python 3.x
* **Web Framework:** Flask
* **Machine Learning:** Scikit-learn
* **Data Handling:** Pandas
* **Vectorization:** TfidfVectorizer
* **Front-End:** HTML, CSS, JavaScript
* **Model Serialization:** Pickle
* **Browser Interaction:** JavaScript (Fetch API)

**3. Project Structure & File Overview**



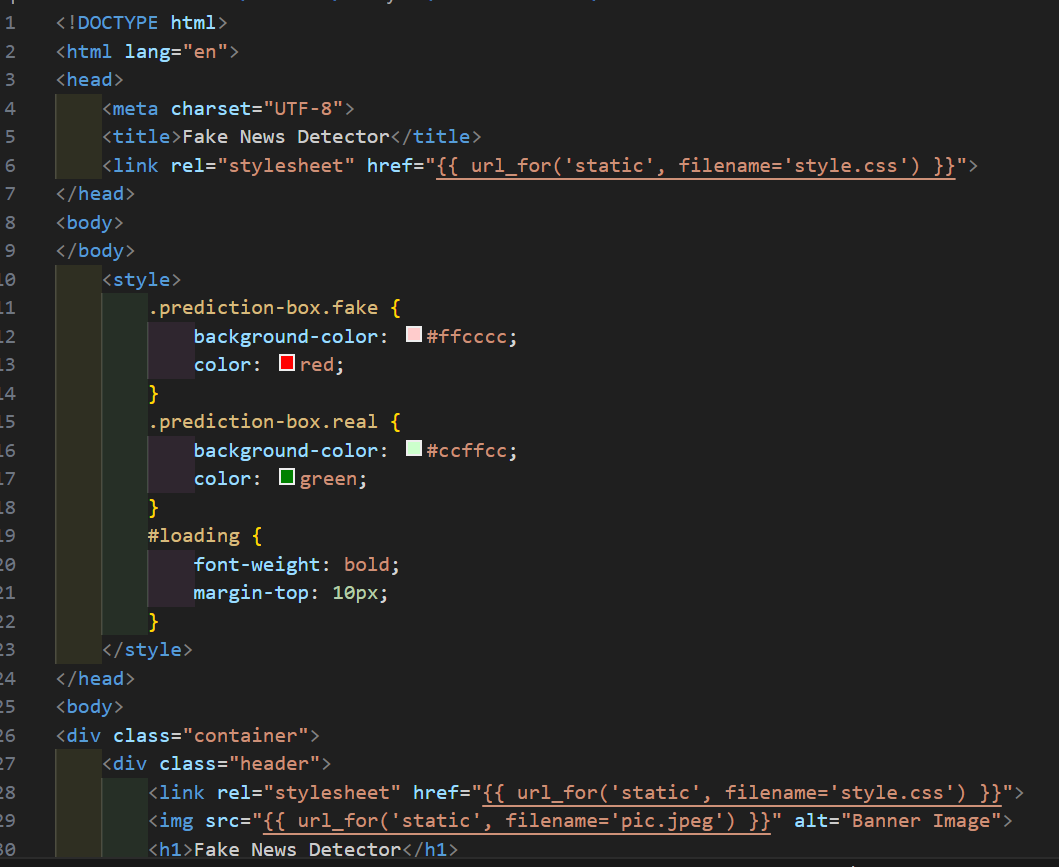
### **3.1 File Descriptions**

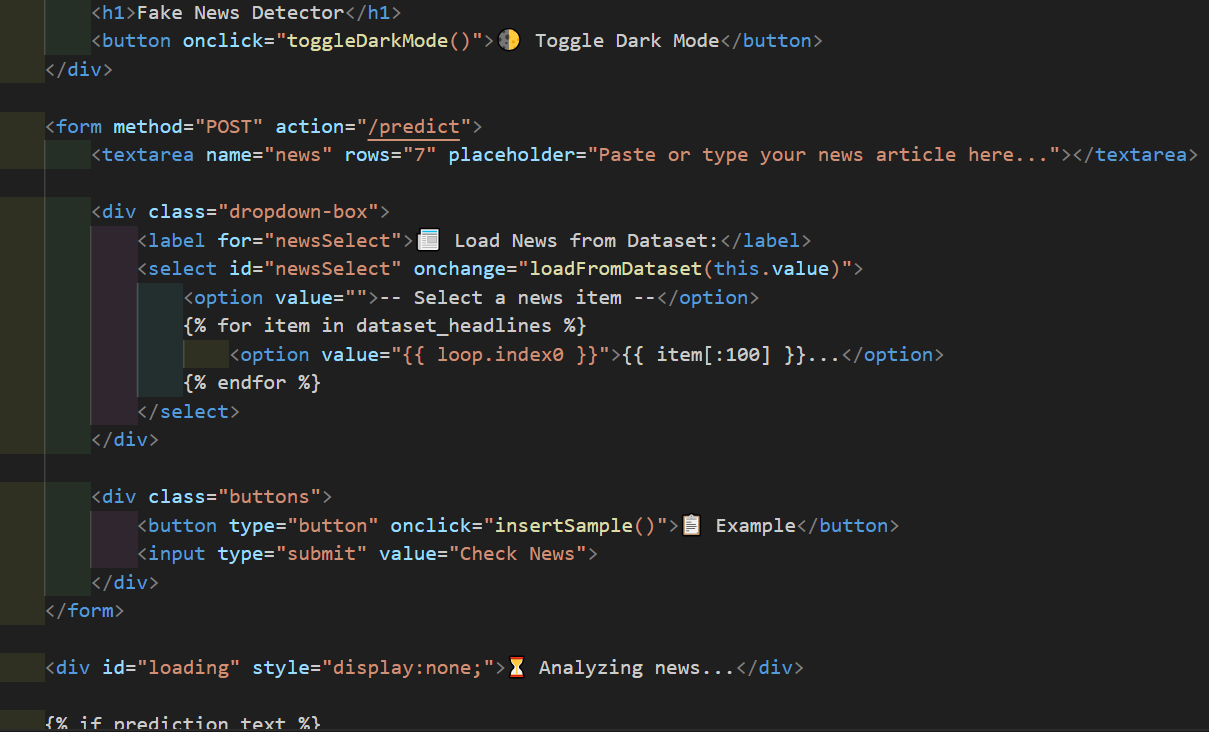
* **app.py:**   
  The main backend application built with Flask. It handles routes, user input, model prediction, and rendering HTML templates.

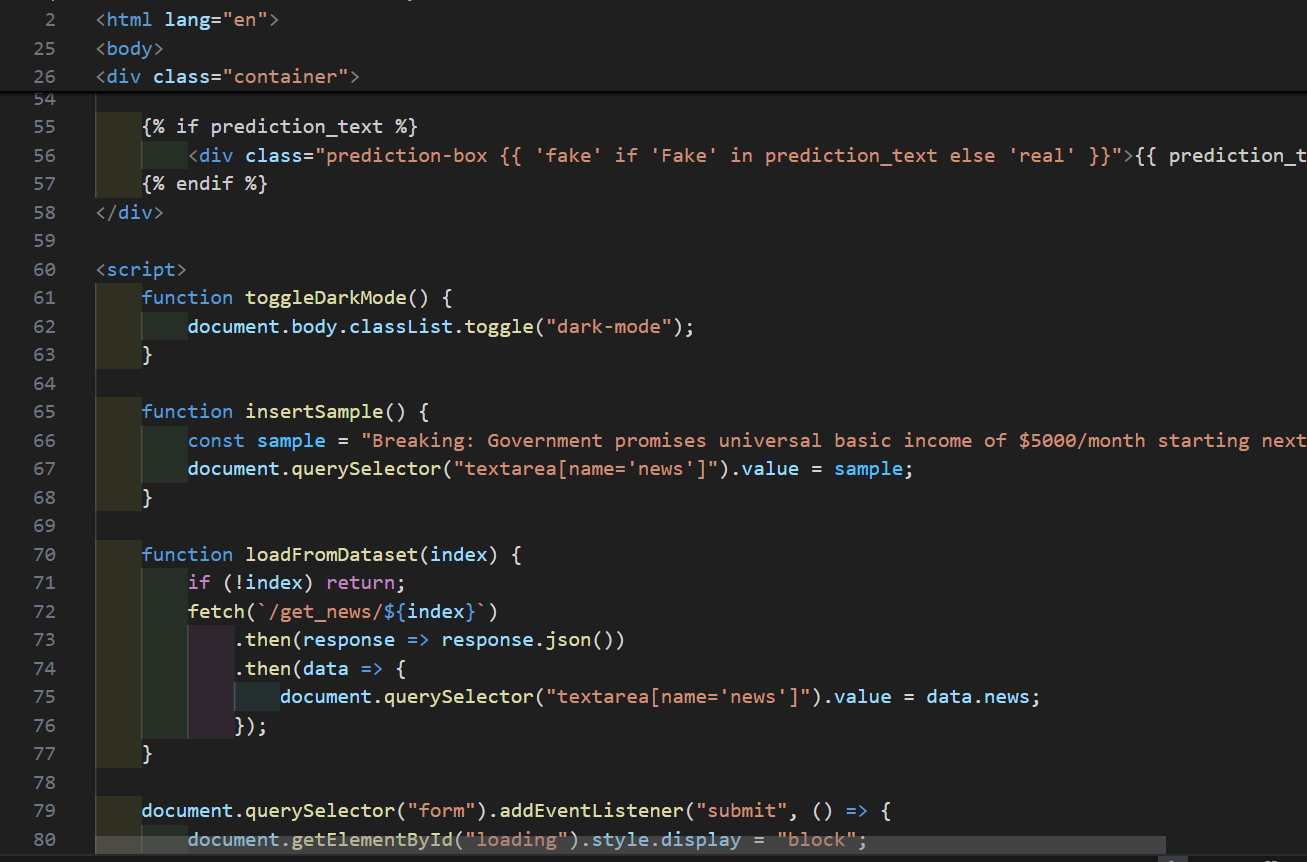




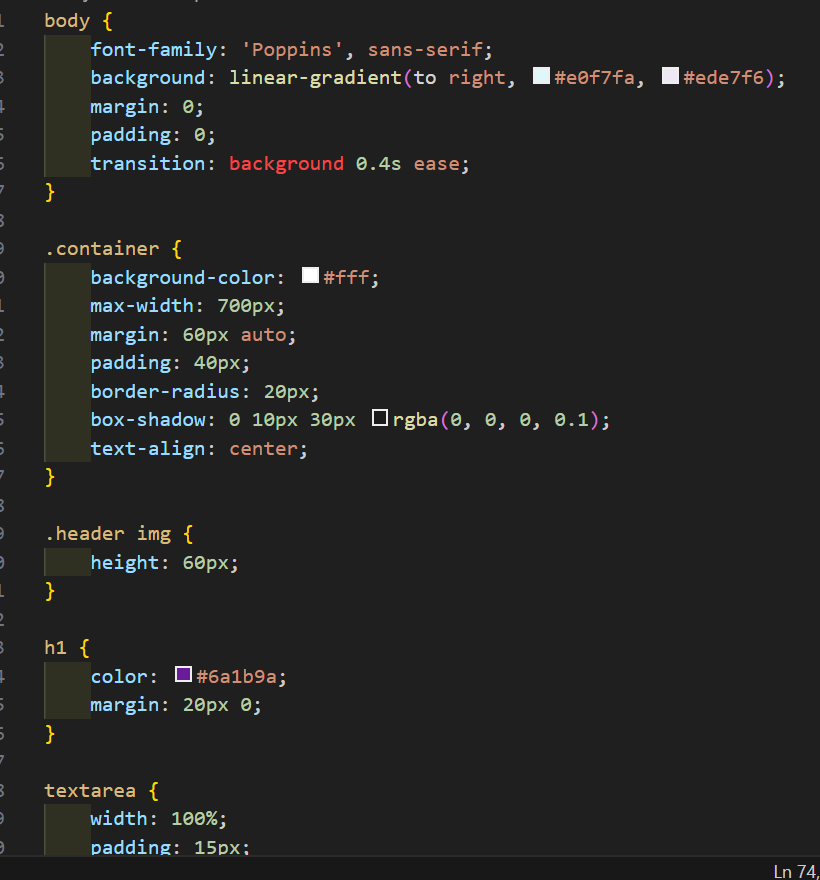
* **model.pkl:**  
  Serialized scikit-learn classifier trained to detect fake news.
* **vectorizer.pkl:**  
  TF-IDF vectorizer used to convert text into numerical features.
* **news.csv:**  
  Dataset used to display sample news headlines and provide testing data.
* **index.html:**  
  Frontend layout of the web application. Allows text input, dataset selection, and displays prediction.

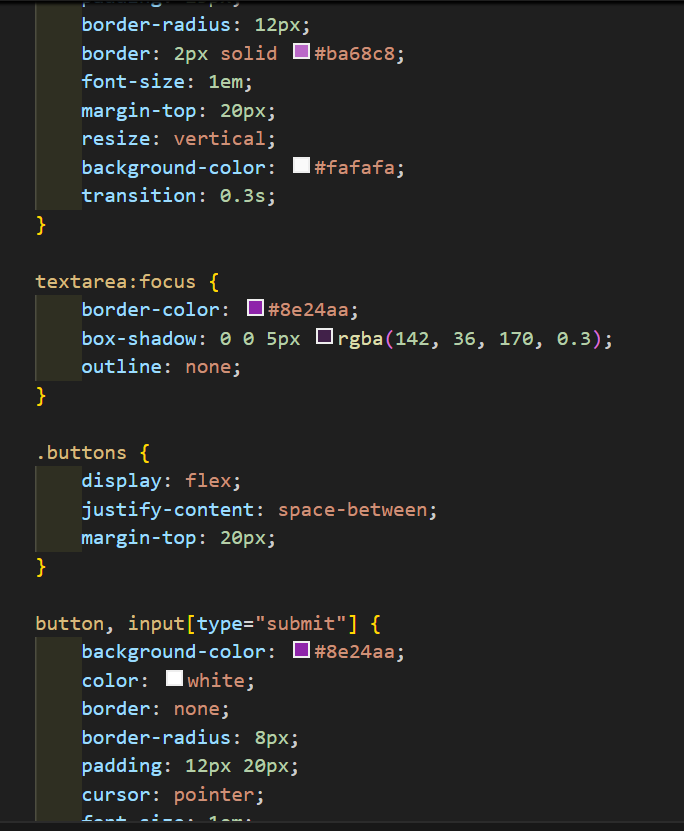


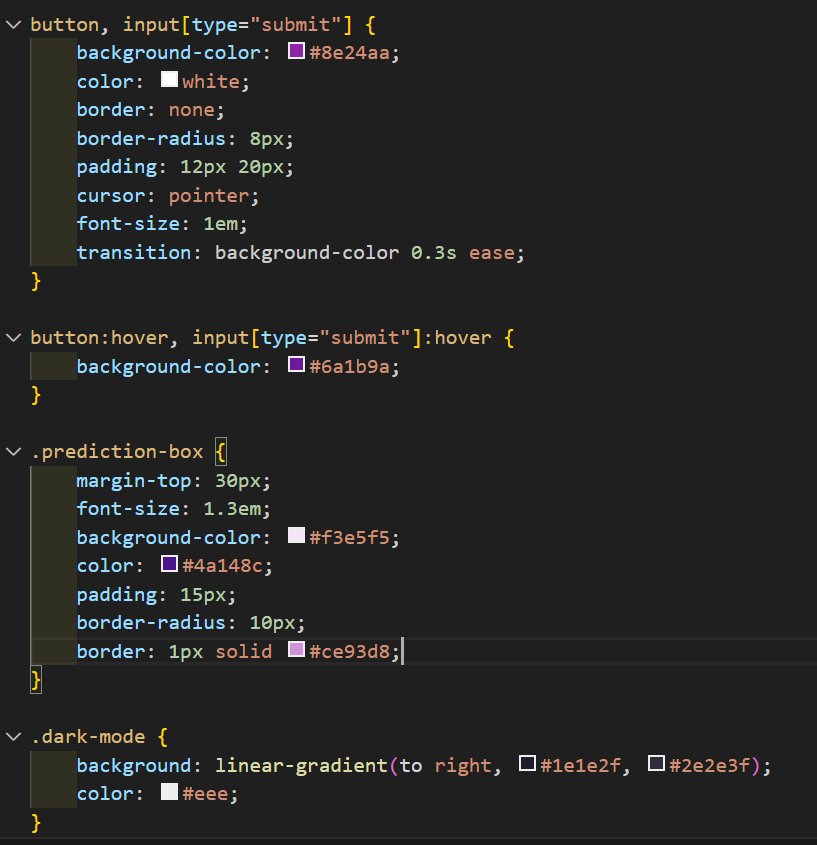


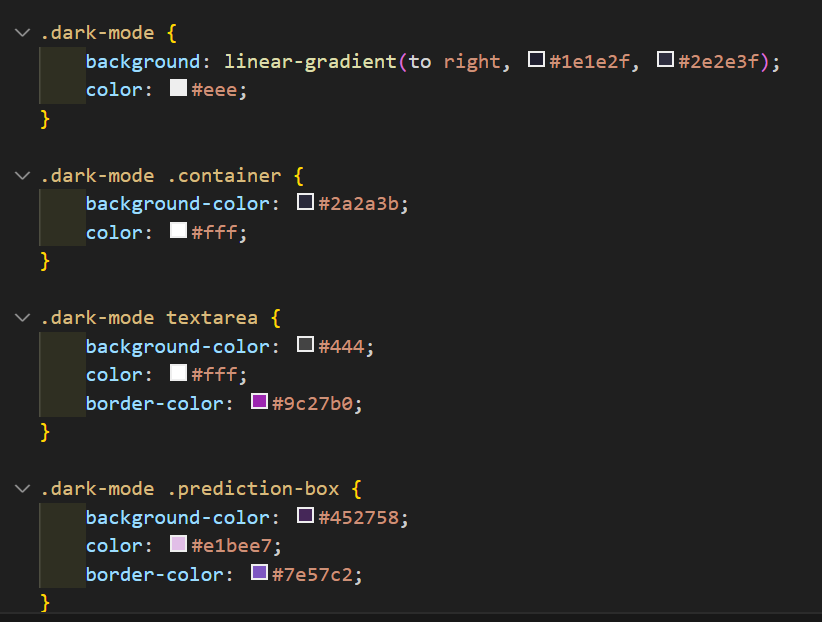


* **style.css:**  
  Provides modern styling, light/dark modes, and layout enhancements.

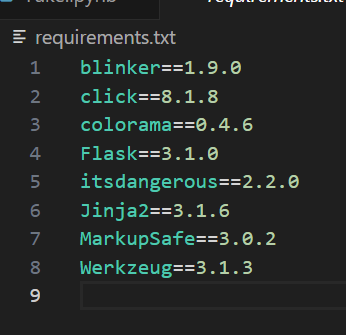








* **pic.jpeg:**  
  A banner image used to decorate the web interface.
* **requirements.txt:**  
  List of all Python packages required to run the application.



**3.2 File Connection & Execution Order**

1. **Startup:** Run python app.py
2. **Process Flow:**
   * The Flask app loads the model and vectorizer.
   * Loads the dataset and displays sample headlines on the homepage.
   * When the user submits news text:
     + If no label is selected, the model predicts "Fake" or "Real".
     + If the user selects a manual label, it is echoed.
   * The result is returned to the user in a styled prediction box.

**4. How It Works**

1. **Homepage Load:**
   * Loads sample news from news.csv.
   * Displays a styled UI with an input form and dropdown menu.
2. **User Interaction:**
   * Users can enter custom news or select a sample.
   * On form submission, the text is sent to /predict.
3. **Prediction Logic:**
   * Text is vectorized using vectorizer.pkl.
   * Passed to model.pkl for classification.
   * Output is displayed with color-coded results (green for Real, red for Fake).
4. **JavaScript Features:**
   * Toggle dark mode.
   * Insert sample news.
   * Dynamically load headlines from dataset using AJAX (Fetch API).

## ****5. Results****

### Example Output:

* Input:  
  "Breaking: Government promises universal basic income of $5000/month starting next week."
* Output:  
  Prediction: Fake

### UI Elements:

* Smooth modern design with Toggl able dark mode.
* Easy input and interaction with dynamic feedback.
* Instant prediction using an ML pipeline.

You can insert screenshots here showing:

* Homepage
* Prediction result
* Dark mode
* Dropdown interaction

**6. Conclusion**

This project demonstrates a practical application of machine learning in identifying fake news. By integrating a trained model with an interactive web interface, it provides a user-friendly tool for public awareness. The use of TF-IDF for feature extraction and Flask for deployment makes it both accessible and effective.

Possible future improvements include:

* Adding support for multilingual text.
* Using more advanced NLP models like BERT.
* Allowing user feedback for model retraining.