

**Full name**: Khushi Singh

**Email: khushisingh82072@gmail.com**

**GitHub ID**: <https://github.com/kalamkaar9404/gittut.git>

**Location (Country and city)**: Gwalior, India

PROBLEM STATEMENT - **Digital trust and information integrity validation solution**

Objective - **To offer businesses a reliable platform/tool for verifying trustworthy information.**

**SOLUTION -**

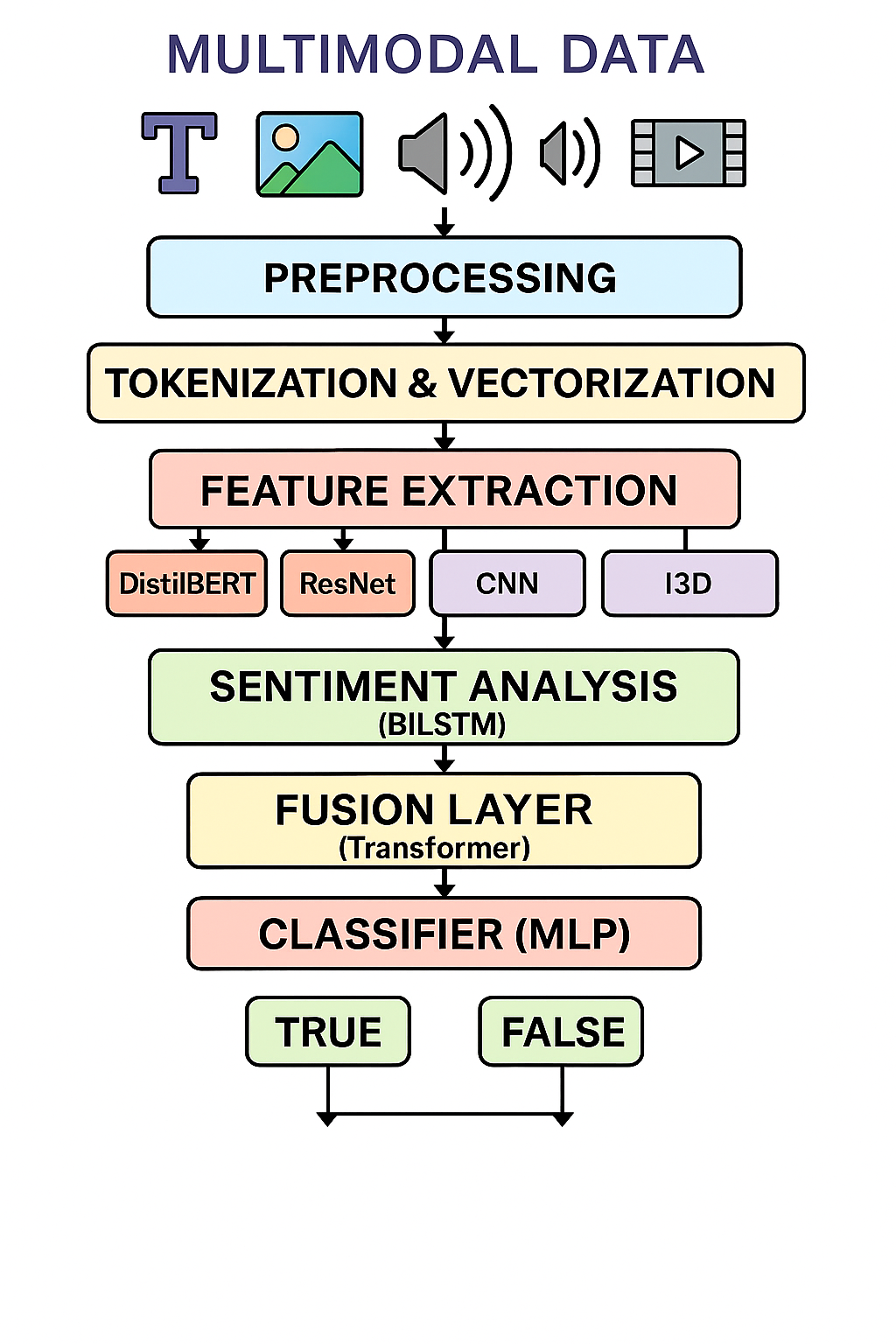
**InfoCrucible -** The Multimodal Fake Information Detector

The proposed solution implements multimodal approach for various input data such as text, images, videos and audio for detecting misinformation (unintentional spread of wrong information) and disinformation (intentional spread of false information).

Proposed Architecture-

We will combine embeddings from the three layers and pass them to a fusion layer for the final classification.

Step 1: Multimodal Data is preprocessed to remove noise. Each modality (text, image, audio, video) should be preprocessed individually.

* Text: remove stopwords, lowercase, lemmatize.
* Image: resize, normalize.
* Audio: noise reduction, convert to spectrogram.
* Video: frame sampling, resizing, extract audio if needed.

Step 2: This includes tokenization (dividing into smaller chunks) of preprocessed multimodal data and then converting tokenized data to embeddings (vectors). Each modality is tokenized/vectorized/encoded separately.

Step 3: These embeddings are then passed to encoders for feature extraction (like DistilBERT for text, EfficientNet B3 or ResNet for image).

Step 4: Sentiment analysis layer for textual or audio data to give sentiment scores.

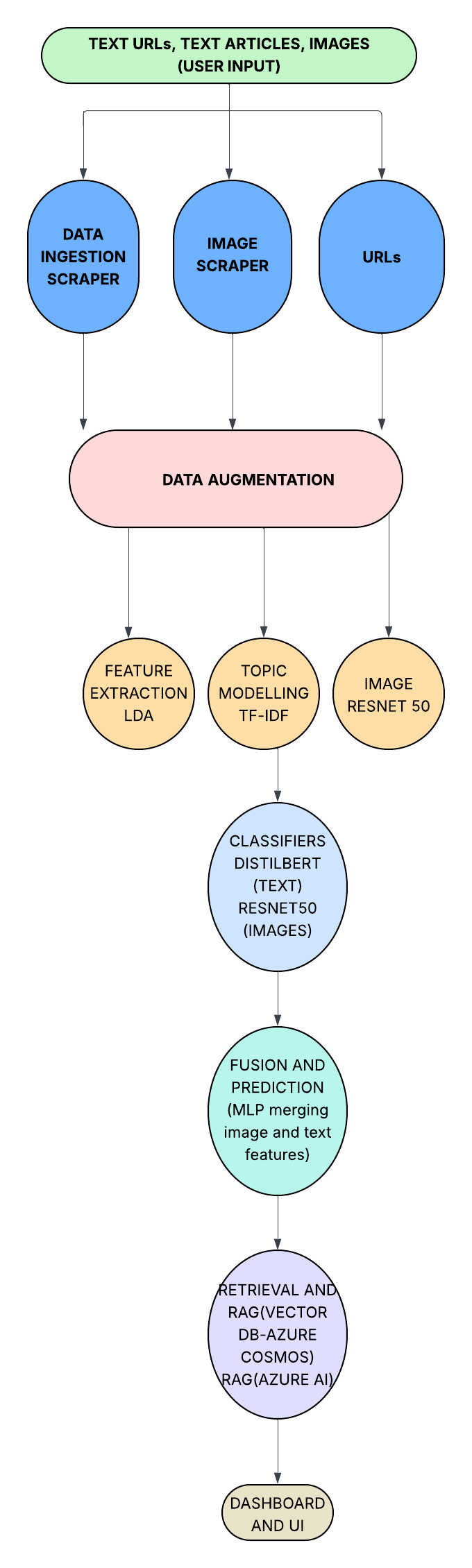
**Step 5: Attention Layer**

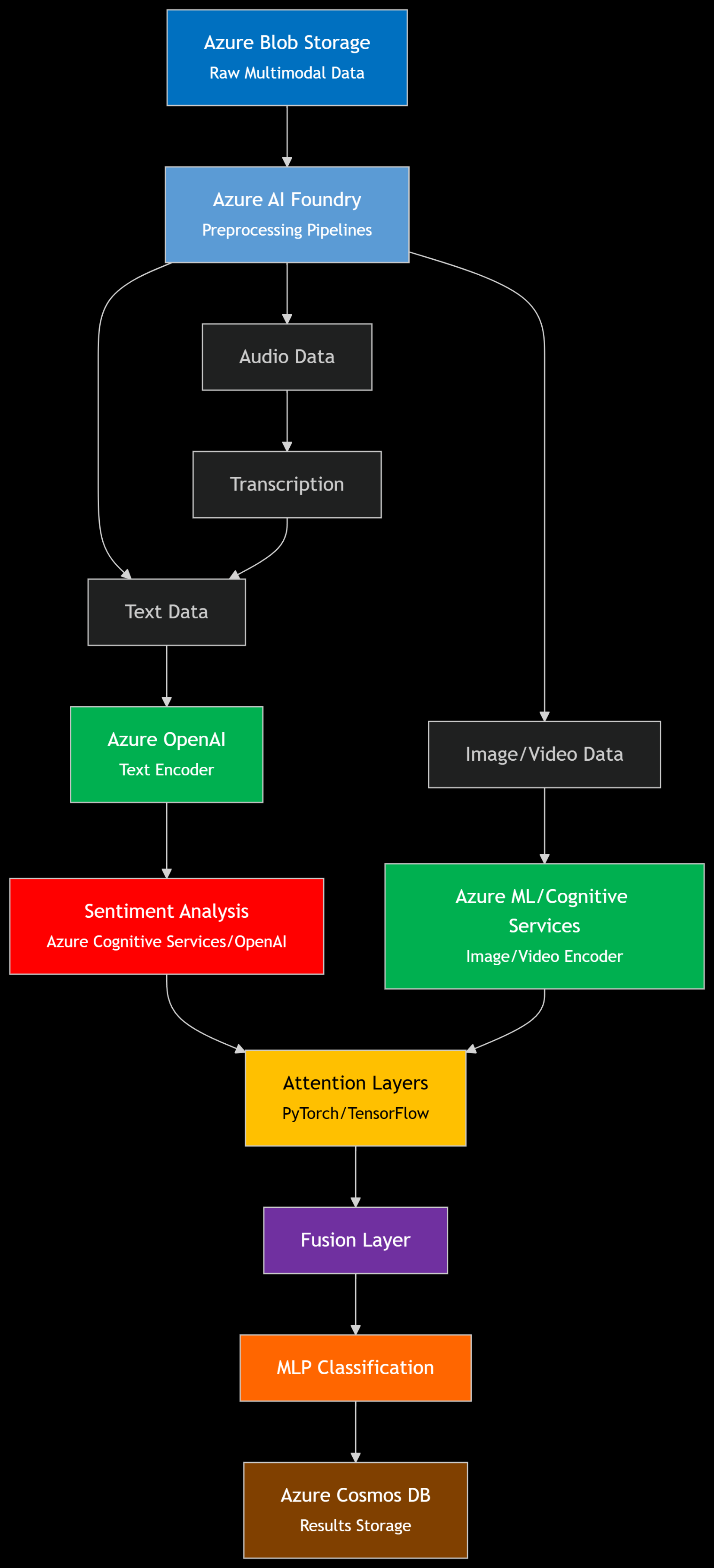
Apply **cross-modal attention** or **self-attention**:

* Helps the model **focus on important parts** of each modality (e.g., image caption mismatch)
* Especially useful in **video (temporal attention)** and for **text-image contradiction**, e.g., in deepfakes.

Step 6: a fusion layer which merges the feature vector from all the individual modalities.

Step 7: MLP for final classification as true or false information.





**Integration of The Architecture with Azure Services:**

**Step 1 & 2 (Preprocessing and Embedding):**

* **Azure Storage** (Blob Storage) can be used to store raw and preprocessed multimodal data (text, images, audio, video). Separate containers can be created for each modality for organized storage.
* **Azure AI Foundry** for scalable and efficient preprocessing pipelines. Foundry provides AI services and orchestration to apply transformations like stopword removal, resizing, noise reduction, spectrogram conversion, and tokenization in an automated, cloud-native way.

**Step 3 (Encoders for Feature Extraction):**

* **Azure OpenAI Service** to leverage models like DistilBERT or other transformer-based models.
* For image/video encoders (like EfficientNetB3 or ResNet), use of **Azure Machine Learning** (which can be integrated via Azure AI Foundry) or Azure’s **Cognitive Services (Vision)** for feature extraction if prebuilt models suffice.

**Step 4 (Sentiment Analysis Layer):**

* Utilization of **Azure Cognitive Services Text Analytics** or Azure OpenAI to perform sentiment analysis on text and audio transcriptions.

**Step 5 (Attention Layers & Cross-Modal Attention):**

* Attention mechanisms using frameworks like PyTorch or TensorFlow, and deployment of the model on **Azure Machine Learning** or within Azure AI Foundry’s AI orchestration.
* Azure AI Foundry can help manage compute resources and workflow orchestration.

**Step 6 (Fusion Layer):**

Fusion of modality features can be done within the model architecture running on **Azure Machine Learning** or Azure AI Foundry. Orchestration of the flow of data from individual modality encoders to the fusion layer.

**Step 7 (MLP for Classification):**

* Deployment of the final MLP classifier as part of your model pipeline on **Azure Machine Learning** or within Azure AI Foundry.

**Additional Storage & Data Management:**

* Store intermediate embeddings, metadata, and inference results in **Azure Cosmos DB**, a globally distributed NoSQL database service, to enable fast querying and low-latency access for your application. Cosmos DB is great for storing multimodal metadata and classification outcomes with scalability.

COMPLETE SOLUTION-

Data Ingestion covers both URL scraping (Phase 1) and direct article downloads (Phase 2).

Data Augmentation implements the Bag‑of‑Words pseudo‑fake augmentation using cosine similarity (Phase 2 & 3).

Topic Modeling (Phase 4) uses LDA to assign a topic category field.

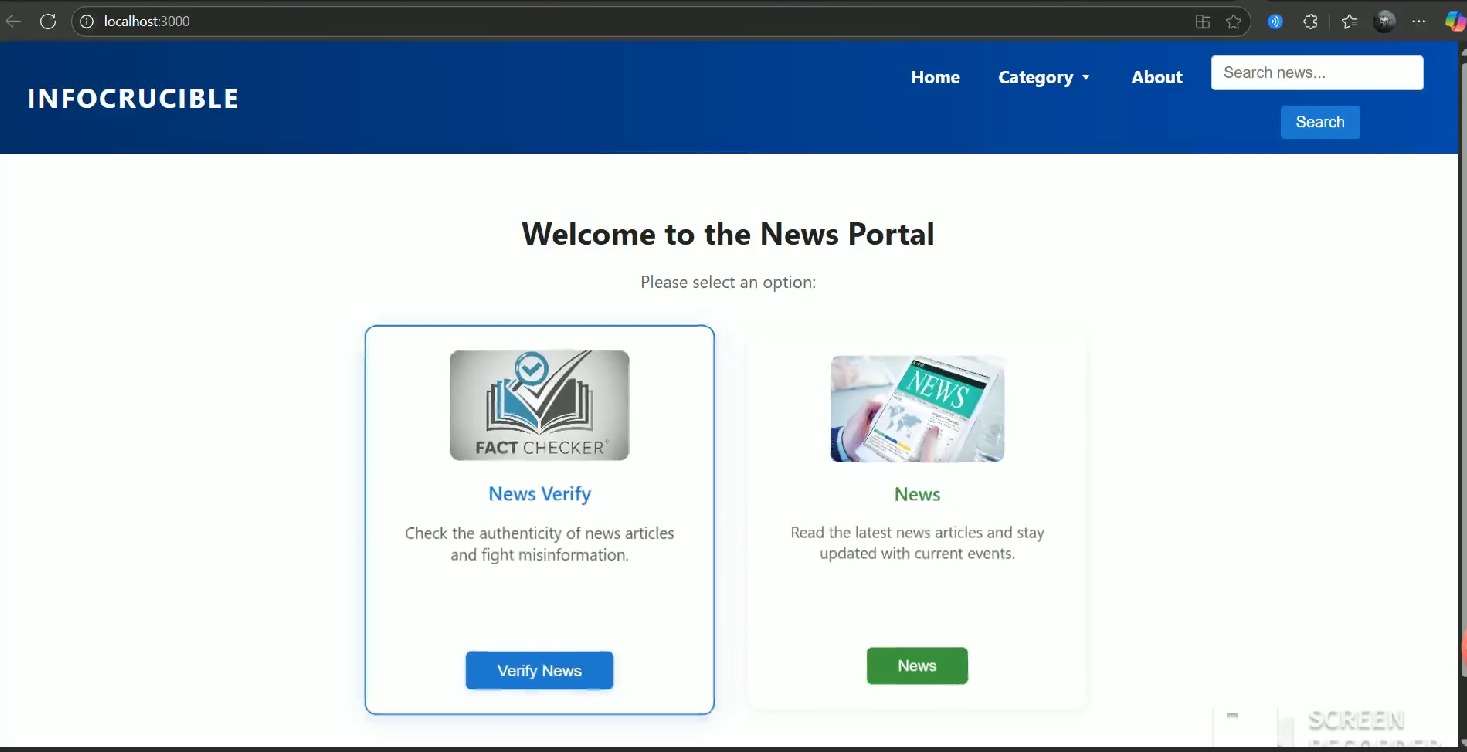
Feature Extraction splits into text (TF‑IDF, embeddings for DistilBERT) and image (deepfake/detection features via ResNet‑50).

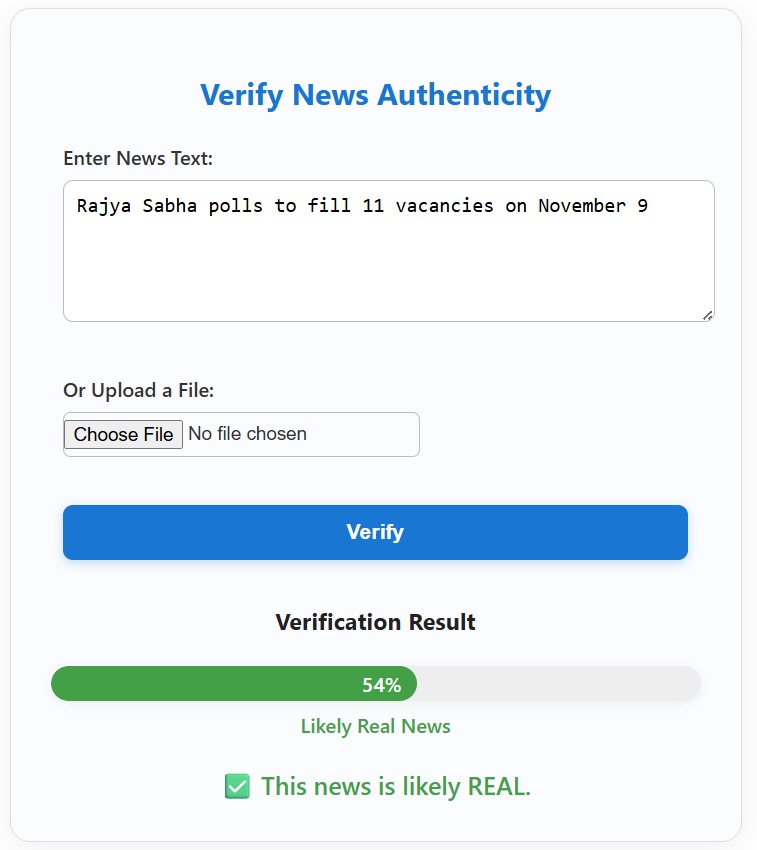
Classifiers (Phase 5–7) include both classical ML (DT, NB, LR, KNN) and deep‑learning models.

Retrieval & RAG uses Cosmos DB’s vector search (thread detection) and Azure OpenAI for RAG-based real-time fact checking.

Dashboard & API is your React frontend + serverless Azure Functions (for real-time link/file checks).

WORKING PROTOTYPE:





ADDITIONAL USP-

1.The major challenge in detection of false information lies in-

* There is a blurred boundary in between real and fake information since there is no clear set of rules for the distinction.
* Sometimes some amount of real news is mixed with fake news to threaten integrity

Thus, we can use Fuzzy logic for sentiment analysis which handles entities which cannot be classified as either but the logic can perform approximate reasoning to improve performance significantly.