Artificial intelligence applications

Submitted by

BADINENI HARSHITH

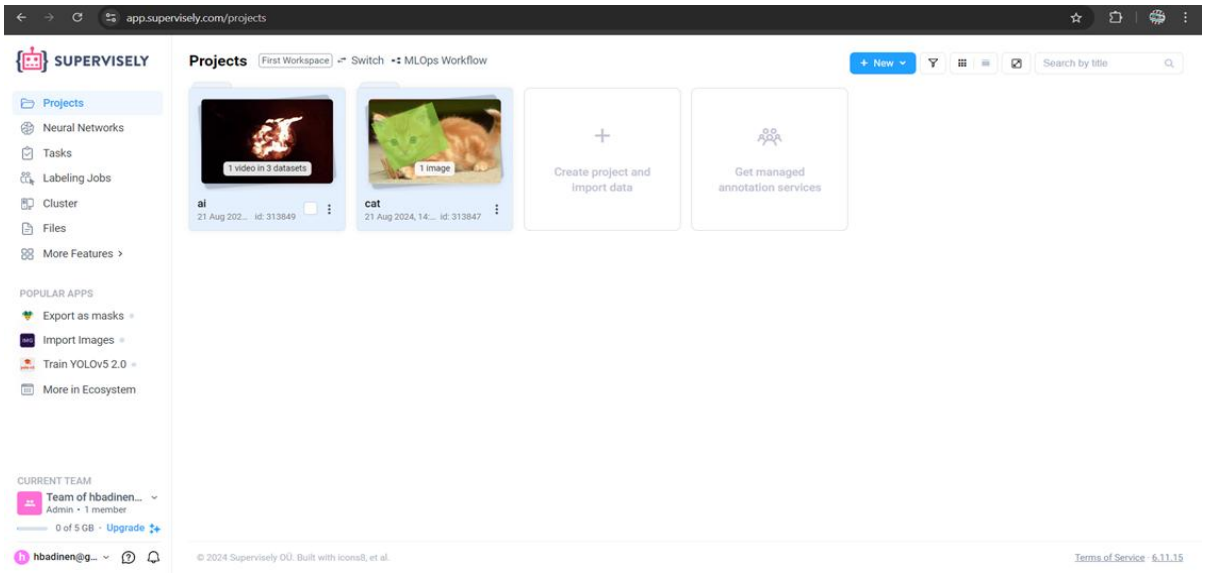
TABLE OF CONTENT

SL.NO	NAME OF EXPERIMENT
1	Implementation of Supervise.
2	Neural Network.
3	lobe.ai Image Processing.
4	Face And Eyes Recognition.
5	Sentiment Analysis And Polarity Detection.
5	Text to Speech recognition and Synthesis through APIs.
7	Building a Chatbot Using Pandora Bots

1) Implementation of Supervise.

Suprvise:perform data labeling for various images using object recognition

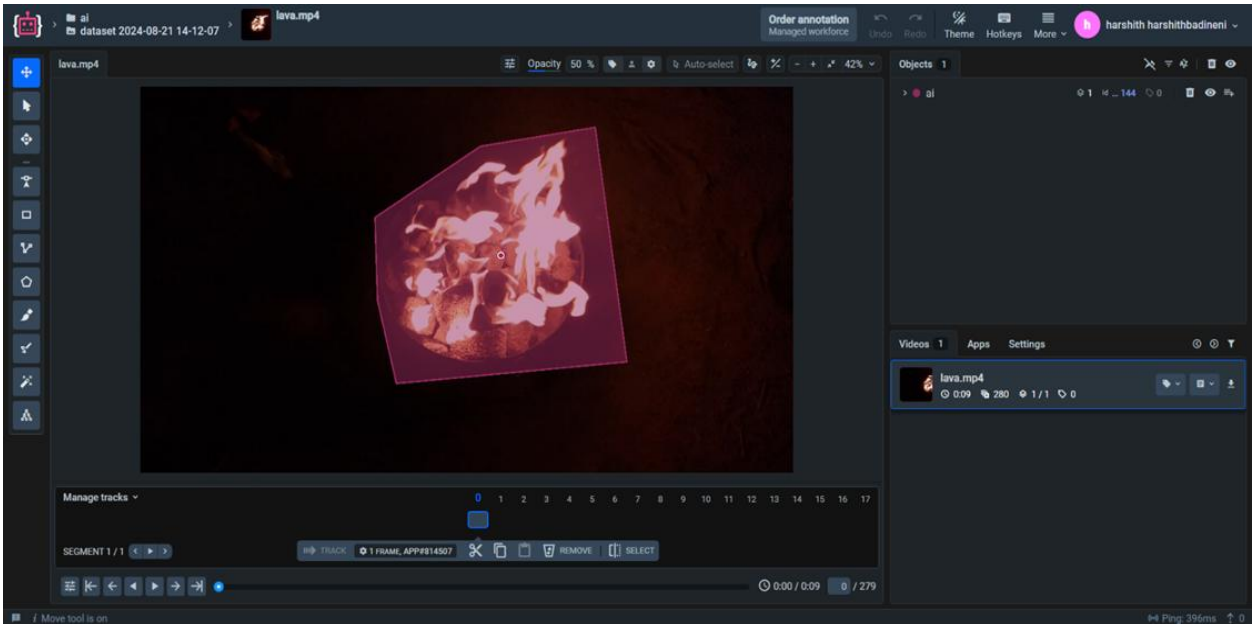
Using https://app.supervisely.com/ to perform object recognition (video/photo)



- **Supervisely** is a data-labeling platform designed for image and video annotation tasks, such as object recognition.
- It supports tasks like object detection, **segmentation** and **classification**
- Users can label **images or videos** by identifying and tagging specific objects, either manually or with AI assistance.
- Tools include **bounding boxes, polygons**, and **semantic segmentationto** mark objects.

- The platform integrates with **AI models** for **automated labeling,** improving speed and accuracy.
- Labels are **stored** for training machine-learning models.
- It is widely used in fields like **computer vision, autonomous driving**, and **robotics**.

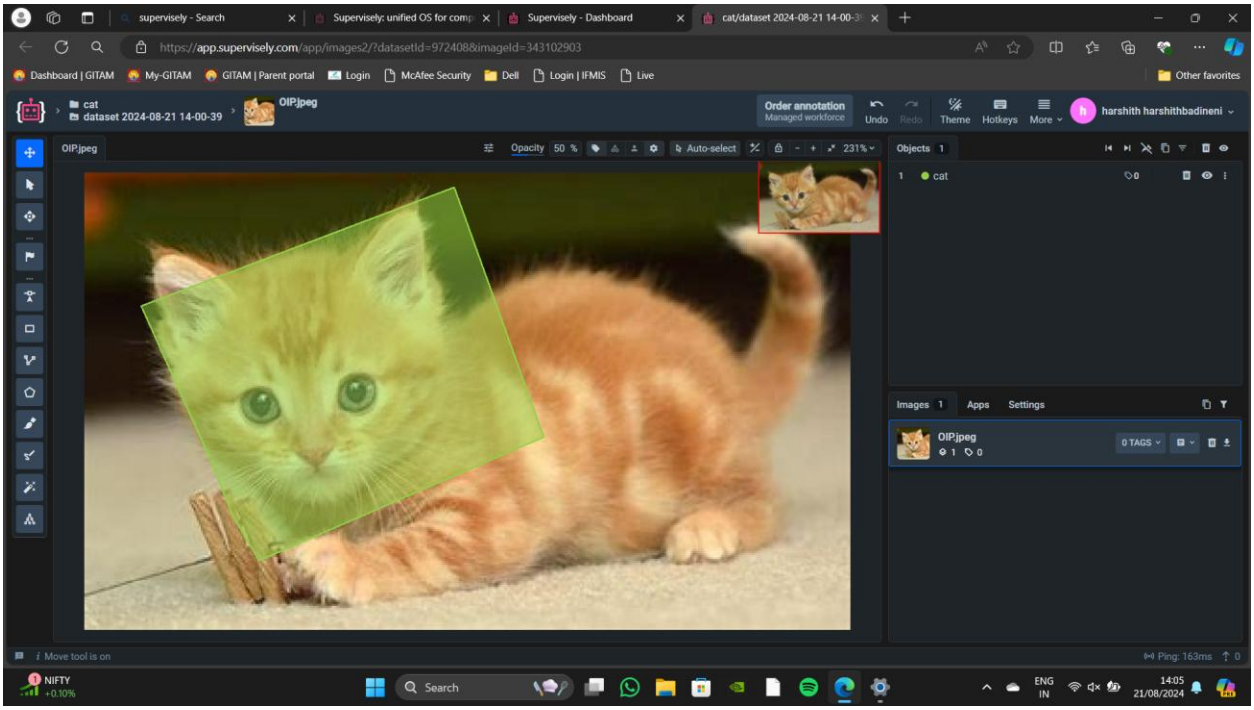
Video as input data:



code :

{"videoId":343103049,"datasetId":972426,"videoName":"lava.mp4","createdAt":"2024-08-21T08:42:30.718Z","updatedAt":"2024-08-21T08:43:15.449Z","description":"","tags":[],"objects":[{"id":1221384144,"classId":13296168,"datasetId":972426,"labelerLogin":"hbadinen@gitam.in","createdAt":"2024-08-21T08:42:59.636Z","updatedAt":"2024-08-21T08:42:59.636Z","tags":[],"entityId":343103049,"classTitle":"ai"}],"size":{"height":1080,"width":1920},"framesCount":280,"frames":[{"index":0,"figures":[{"id":1586561367,"classId":null,"objectId":1221384144,"description":"","geometryType":"polygon","trackId":null,"labelerLogin":"hbadinen@gitam.in","createdAt":"2024-08-21T08:43:04.822Z","updatedAt":"2024-08-21T08:43:15.429Z","geometry":{"points":{"exterior":[[697,598],[689,365],[1000,166],[1337,113],[1431,769],[753,830]],"interior":[]}}}]}]}

Image as input data:



code :

{"imageId":343102903,"imageName":"OIP.jpeg","createdAt":"2024-08-21T08:31:36.520Z","updatedAt":"2024-08-21T08:34:33.563Z","link":null,"annotation":{"description":"","tags":[],"size":{"height":244,"width":408},"objects":[{"id":1586507538,"classId":13296161,"objectId":null,"description":"","geometryType":"polygon","labelerLogin":"hbadinen@gitam.in","createdAt":"2024-08-21T08:33:53.786Z","updatedAt":"2024-08-21T08:34:33.547Z","tags":[],"classTitle":"cat","points":{"exterior":[[66,151],[30,71],[197,8],[244,141],[93,206],[79,184]],"interior":[]}}]}}

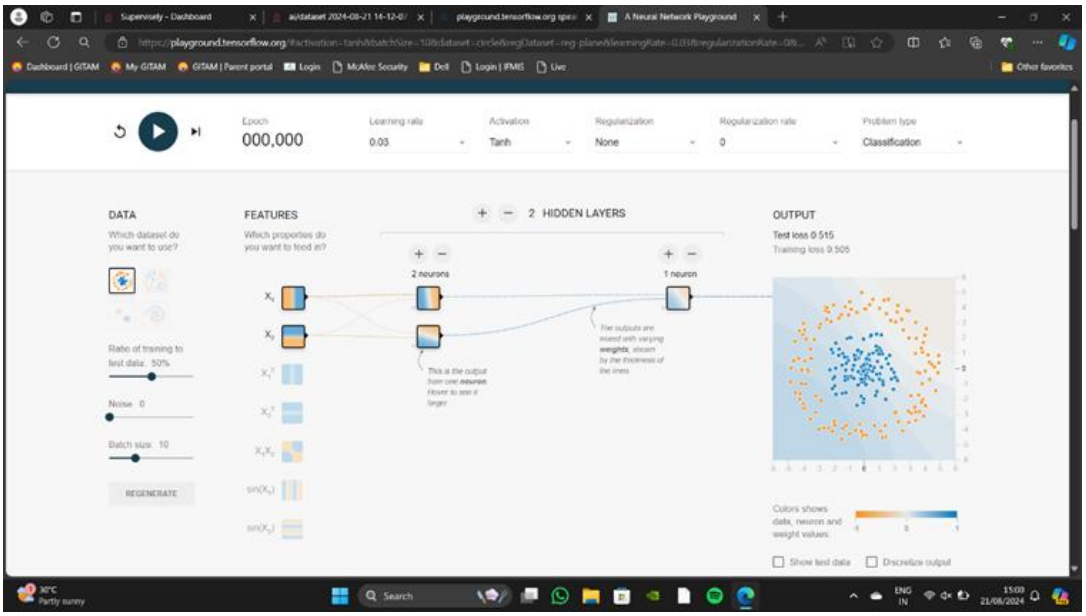
2) Neural Network.

Neural network: Reinforcement Learning, Introduction to Neural Networks, Deep Learning

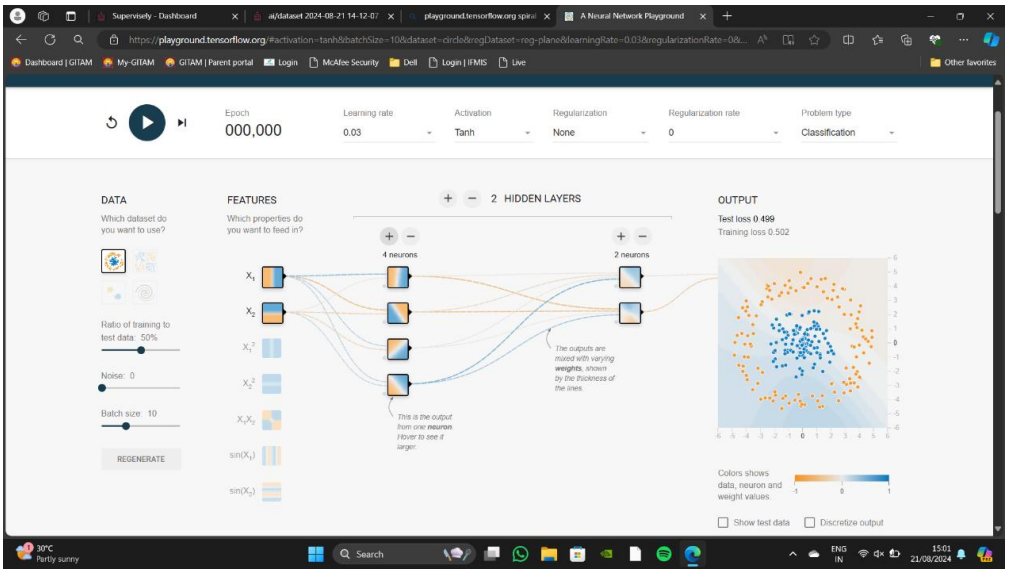
Explore the effect of different hyper parameters while implementing a Simple Fully Connected Neural Network.
https://playground.tensorflow.org/

- **Neural Networks**: Modeled after the human brain, they consist of layers of interconnected nodes (neurons) that process data and make predictions.
- **Reinforcement Learning**: A type of machine learning where agents learn by interacting with an environment to maximize cumulative rewards.
- **Deep Learning**: Involves neural networks with multiple layers (deep networks) to model complex patterns in data.
- **Hyperparameters**: Factors like learning rate, number of layers, number of neurons per layer, and activation functions that influence the performance of neural networks.
- **Effect of Hyperparameters**: Adjusting hyperparameters impacts the model's training speed, convergence, and accuracy, as explored through tools like TensorFlow Playground.

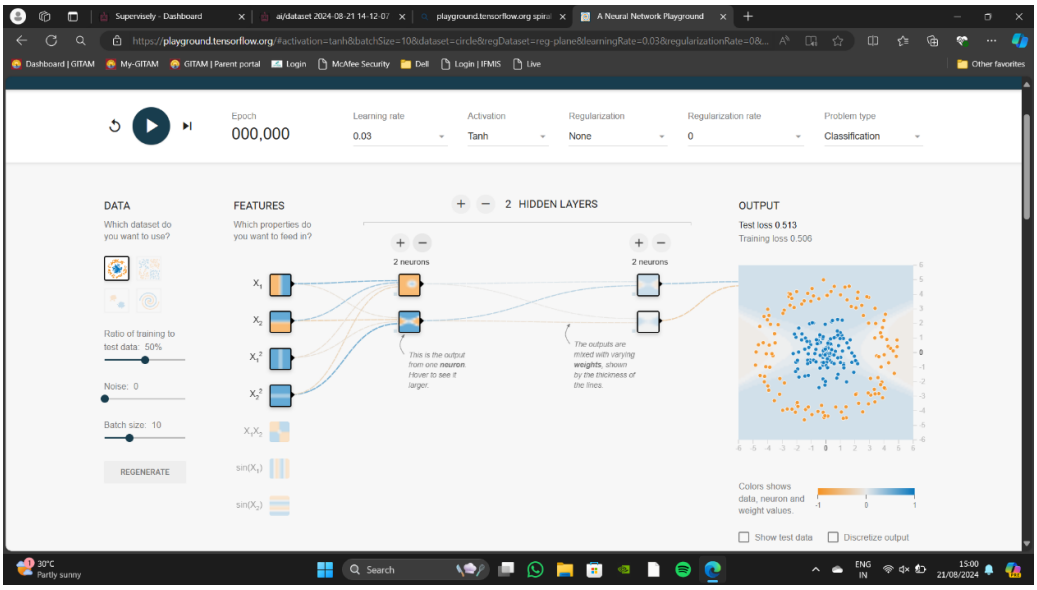
Scenario 1



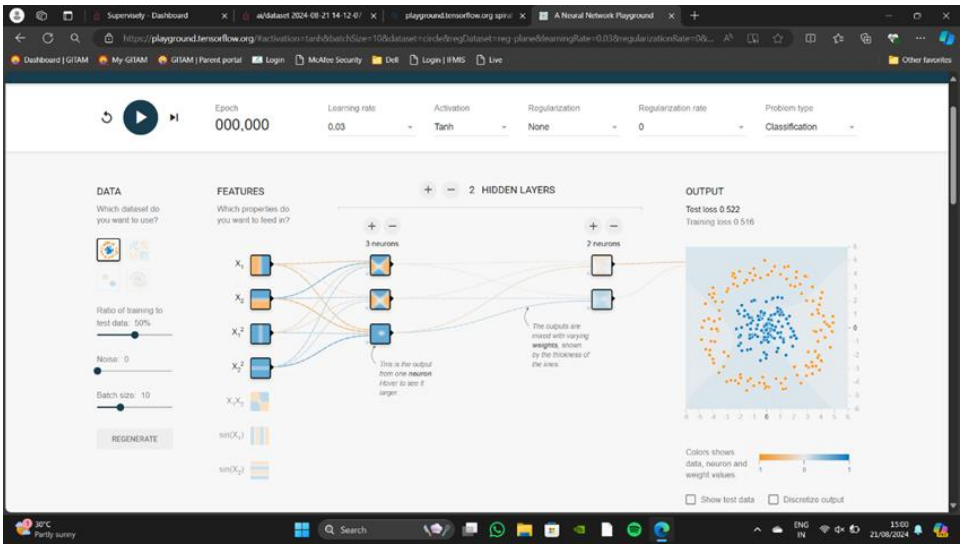
Scenario 2



Scenario 3



Scenario 4



3) lobe.ai Image Processing.

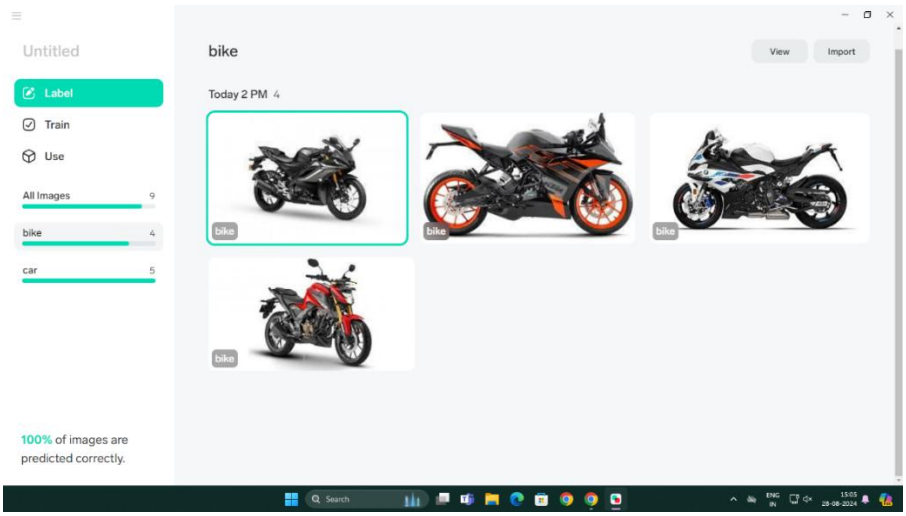
Image Processing & Computer Vision: Introduction to Image processing, Image Noise, Removal of Noise from Images, Color Enhancement, Edge Detection.

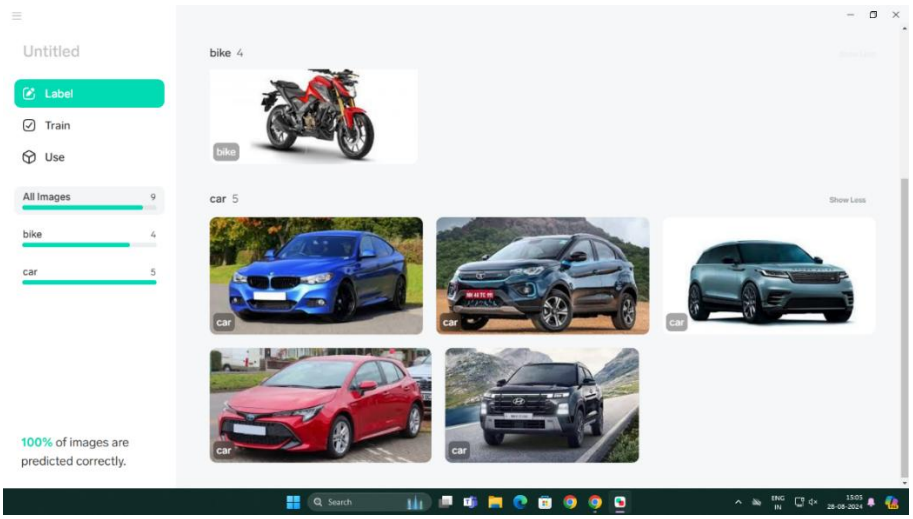
Lobe.ai: Build custom models using the visual tool for Object recognition and sentiment analysis that can convert facial expressions into emoticons
https://www.lobe.ai/

- Image Processing: Techniques used to enhance, manipulate, and analyze images.
- Image Noise: Random variations in pixel intensity that degrade image quality.
- -Noise Removal: Methods like filters (mean, median, Gaussian) to reduce noise.
- Color Enhancement: Adjusting image color balance, contrast, and brightness for visual improvement.
- Edge Detection: Techniques (e.g., Canny, Sobel) to identify object boundaries within images.

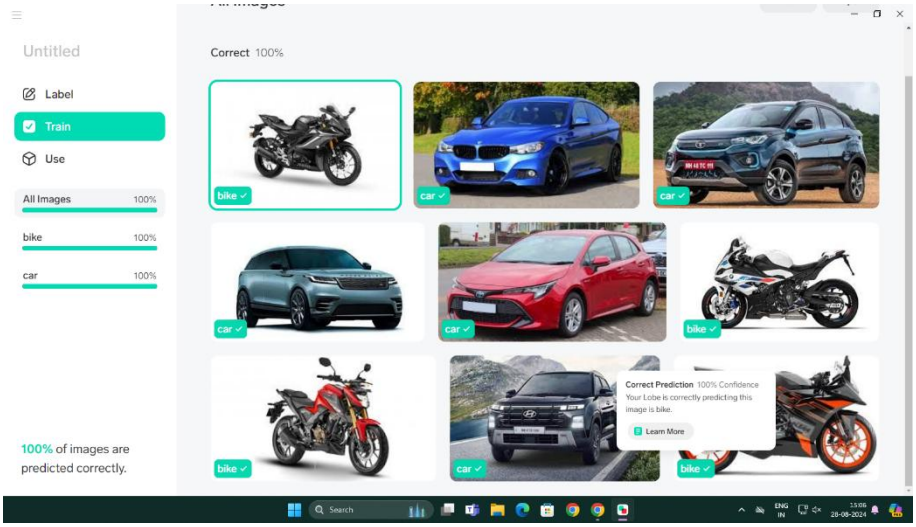
- Lobe.a: A visual tool for building custom models without coding.
- Object Recognition: Identify objects in images.
- Sentiment Analysis: Analyze facial expressions and convert them into emoticons.

a)Using images :

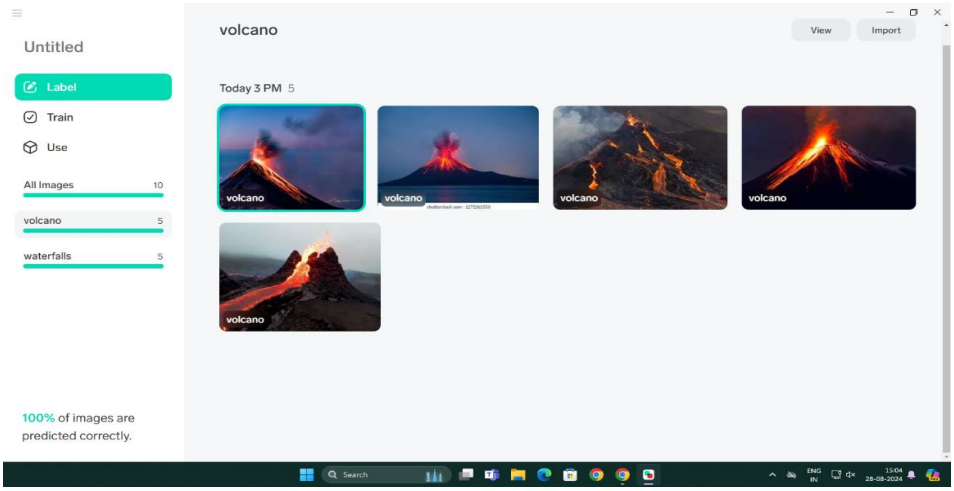


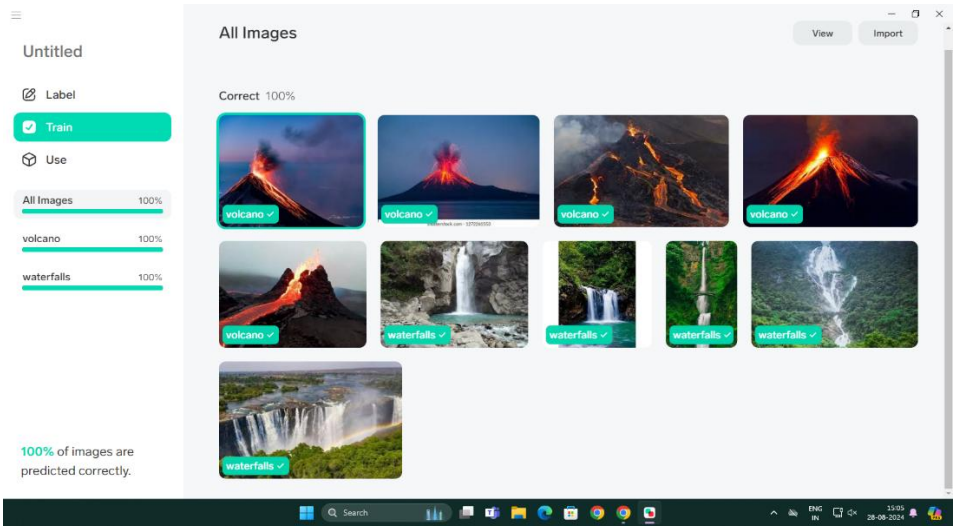


Output :

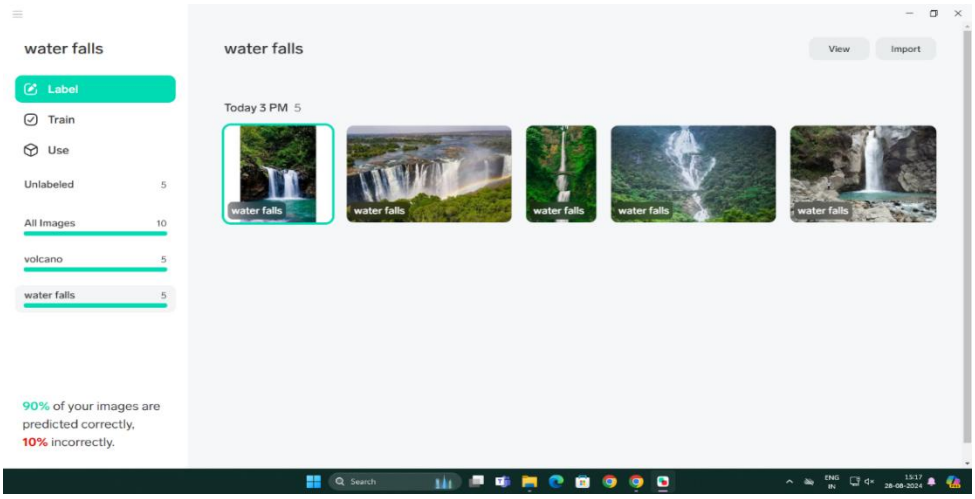


b)Using data set :

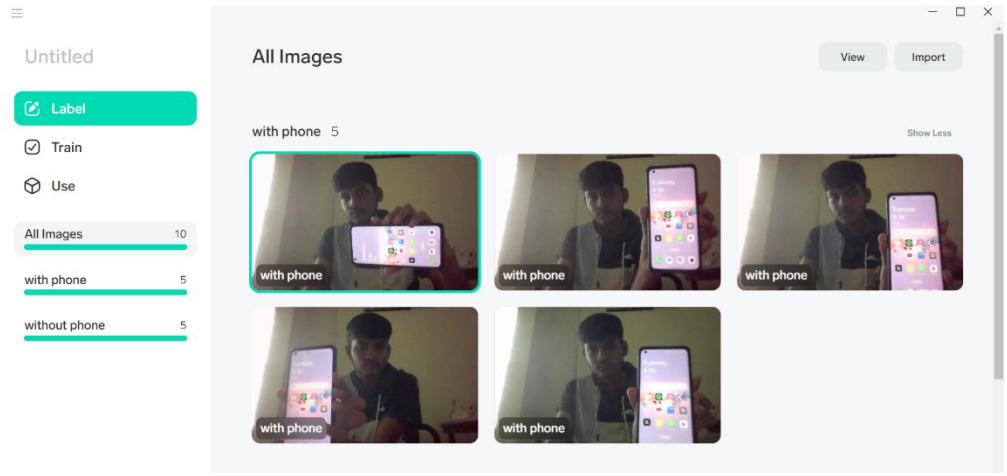


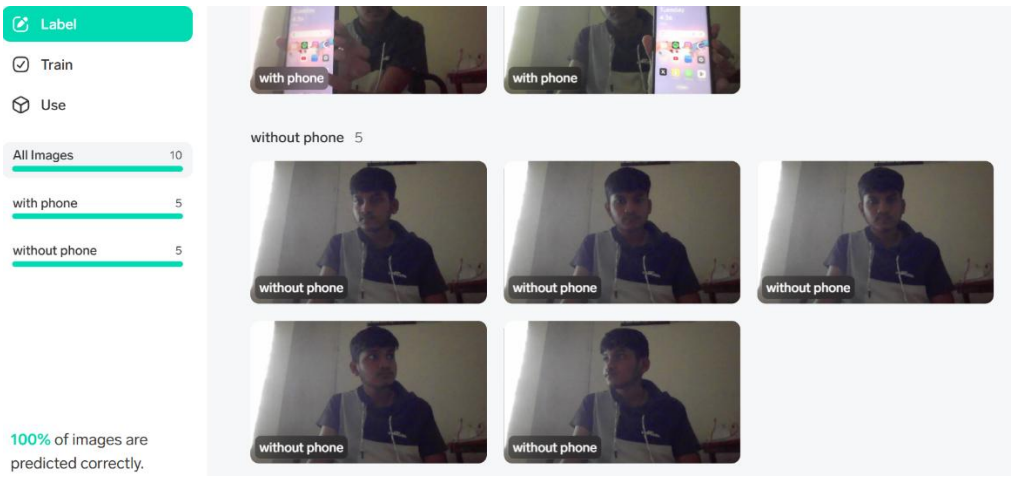


Output :

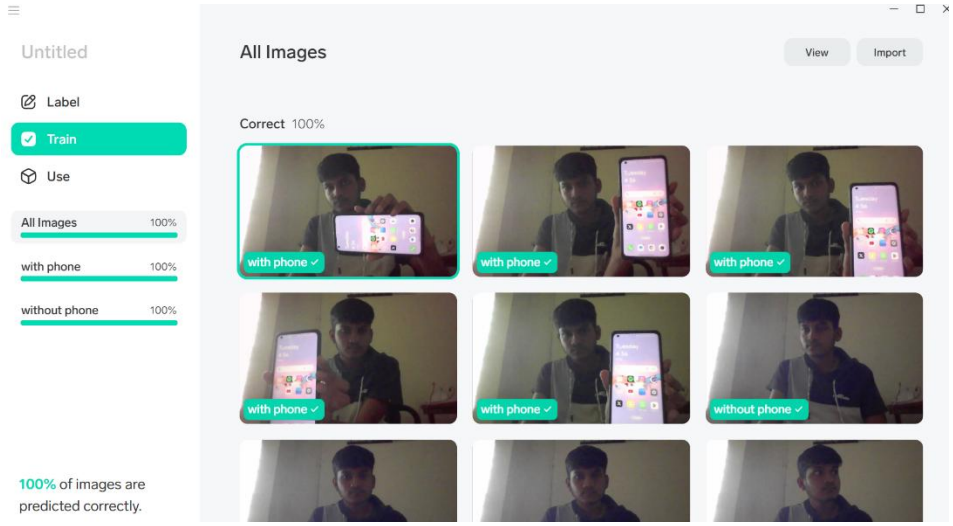


c)Using camera:





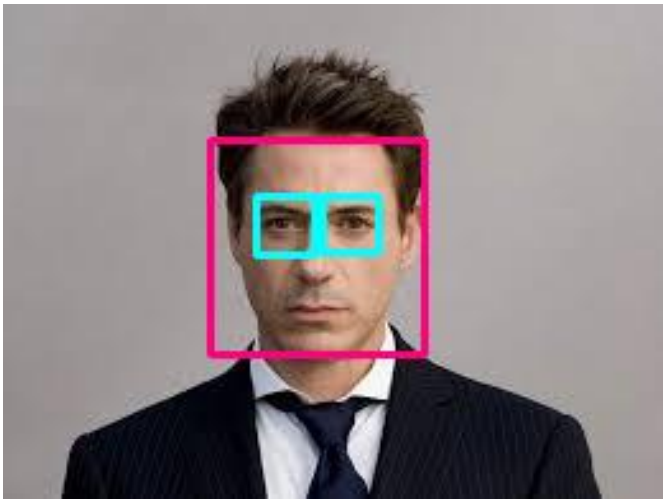
Output :



4)Face And Eyes Recognition.

Face and Eyes Recognition
Face and eyes recognition are specialized applications of computer vision and pattern recognition, focusing on detecting and identifying faces and eye regions in images or videos.

1. **Face Recognition**: This process involves detecting a face within an image and then matching it to a known database. It typically includes steps such as:
    - **Face Detection**: Using algorithms like Haar cascades or deep learning models (e.g., CNNs) to locate faces.
    - **Feature Extraction**: Extracting unique facial features such as distances between facial landmarks.
    - **Face Matching**: Comparing the extracted features to a database to identify or verify a person's identity.
2. **Eyes Recognition**: Eye recognition systems identify the eyes within the detected face and often focus on the iris or pupil for more detailed recognition. Eye recognition is crucial in applications like gaze tracking, biometric authentication, and emotion analysis.



Face and eye recognition using camera

ai 2023001778.py

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

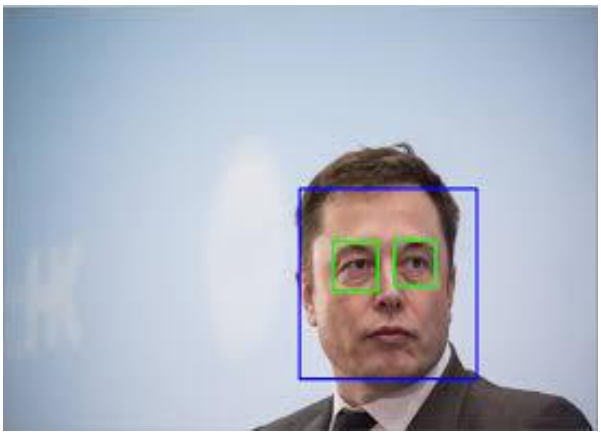
RAM Disk

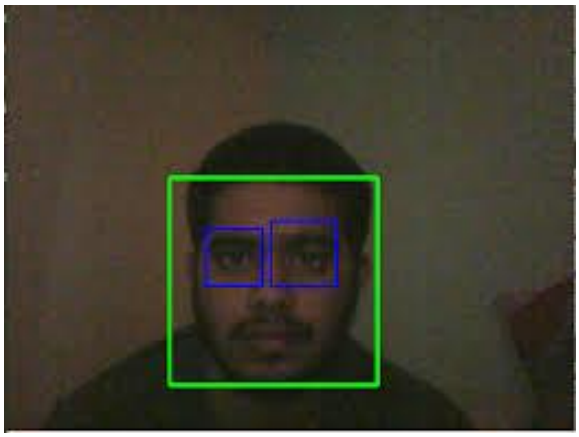
```
# Install OpenCV
!pip install opencv-python-headless
# Import necessary libraries
import numpy as np
import cv2
from google.colab.patches import cv2_imshow
# Download Haarcascade files
!wget -O haarcascade_frontalface_default.xml https://github.com/opencv/opencv/raw/master/data/haarcascades/haarcascade_frontalface_default.xml
!wget -O haarcascade_eye.xml https://github.com/opencv/opencv/raw/master/data/haarcascades/haarcascade_eye.xml
# Load the classifiers
face_classifier = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
eye_classifier = cv2.CascadeClassifier('haarcascade_eye.xml')
# Upload image
from google.colab import files
uploaded = files.upload()
# Read the uploaded image
for img_name in uploaded.keys():
    img = cv2.imread(img_name)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    # Detect faces
    faces = face_classifier.detectMultiScale(gray, 1.3, 5)
    # Draw rectangles around detected faces and eyes
    for (x, y, w, h) in faces:
        cv2.rectangle(img, (x, y), (x+w, y+h), (127, 0, 255), 2)
        roi_gray = gray[y:y+h, x:x+w]
        roi_color = img[y:y+h, x:x+w]
        eyes = eye_classifier.detectMultiScale(roi_gray)
        for (ex, ey, ew, eh) in eyes:
            cv2.rectangle(roi_color, (ex, ey), (ex+ew, ey+eh),(255, 255, 0), 2)

    # Display the image
    cv2_imshow(img)
```

✓ 17s completed at 10:16 PM

Output :





5)Sentiment Analysis And Polarity Detection.

Sentiment Analysis and Polarity Detection are core techniques in natural language processing (NLP) that help us understand the emotions, attitudes, and opinions expressed in text. These techniques are valuable in many fields, including customer feedback analysis, market research, and social media monitoring, to extract insights about how people feel about products, services, or topics.

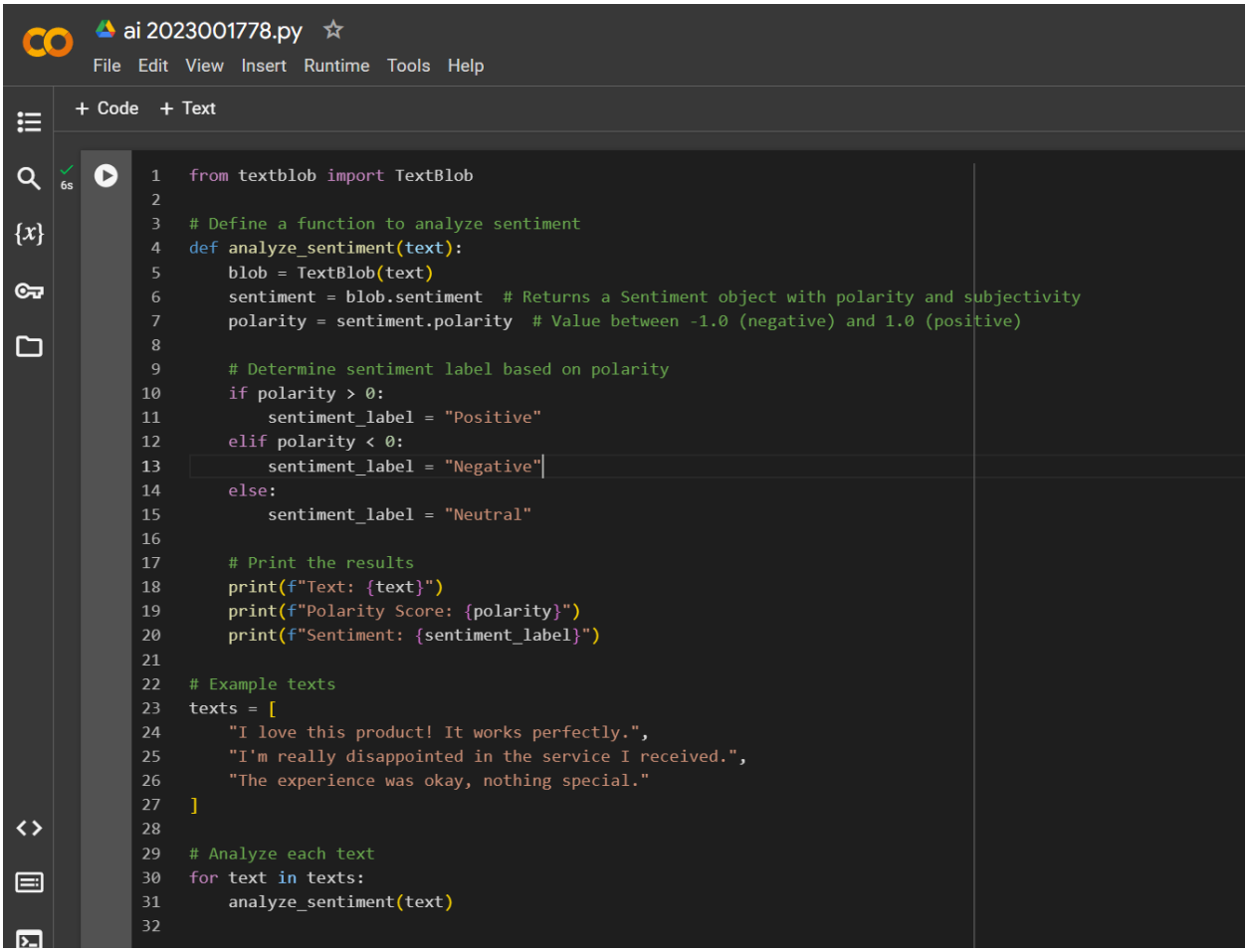
Key Concepts

1. **Sentiment Analysis**:
    - Sentiment analysis is the process of determining the emotional tone behind a piece of text. It generally classifies text as having a *positive*, *negative*, or *neutral* sentiment.
    - This process involves parsing the language and identifying key phrases, tone, and context to interpret the overall emotion. For instance, "I love this!" would yield a positive sentiment, while "This is disappointing" would be negative.
    - Sentiment analysis is used widely in industries to analyze large volumes of text data, especially for customer feedback and social media analysis, to measure public perception and brand sentiment.
2. **Polarity Detection**:
    - Polarity detection is a subset of sentiment analysis that quantifies the sentiment on a scale. Usually, polarity is represented as a score between `-1` and `1`, where `-1` represents a strongly negative sentiment, `0` represents neutrality, and `1` represents a strongly positive sentiment.
    - Polarity detection allows for more granular sentiment scoring, which can be used to understand the intensity of opinions and attitudes. For example, "It's okay" might score closer to 0, while "This is amazing!" would score closer to 1.

Applications

1. **Customer Feedback Analysis**: Analyzing reviews, surveys, or social media comments to understand customer satisfaction and identify areas for improvement.
2. **Brand Monitoring**: Tracking sentiment in online mentions, such as tweets or news articles, to gauge public perception of a brand.
3. **Market Research**: Understanding consumer attitudes toward products or services by analyzing sentiment trends over time.
4. **Recommendation Systems**: Providing better recommendations by gauging user sentiment towards various options, improving personalized suggestions.

Code:



Output:

```
Text: I love this product! It works perfectly.
Polarity Score: 0.8125
Sentiment: Positive
Text: I'm really disappointed in the service I received.
Polarity Score: -0.75
Sentiment: Negative
Text: The experience was okay, nothing special.
Polarity Score: 0.4285714285714286
Sentiment: Positive
```

6) Text to Speech recognition and Synthesis through APIs.

Text-to-Speech (TTS) and Speech Recognition (STT) Using APIs are key technologies in natural language processing that enable systems to interact with humans more naturally. These technologies are widely used in applications such as virtual assistants, accessibility tools, automated customer service, and IoT devices.

1. Speech Recognition (STT - Speech-to-Text)

- **Definition**: Speech recognition, or Speech-to-Text (STT), is the process of converting spoken language into written text. This is achieved through machine learning models that identify phonetic sounds and linguistic patterns in audio, transforming them into accurate textual representations.
- **How It Works**: Speech recognition models generally involve:

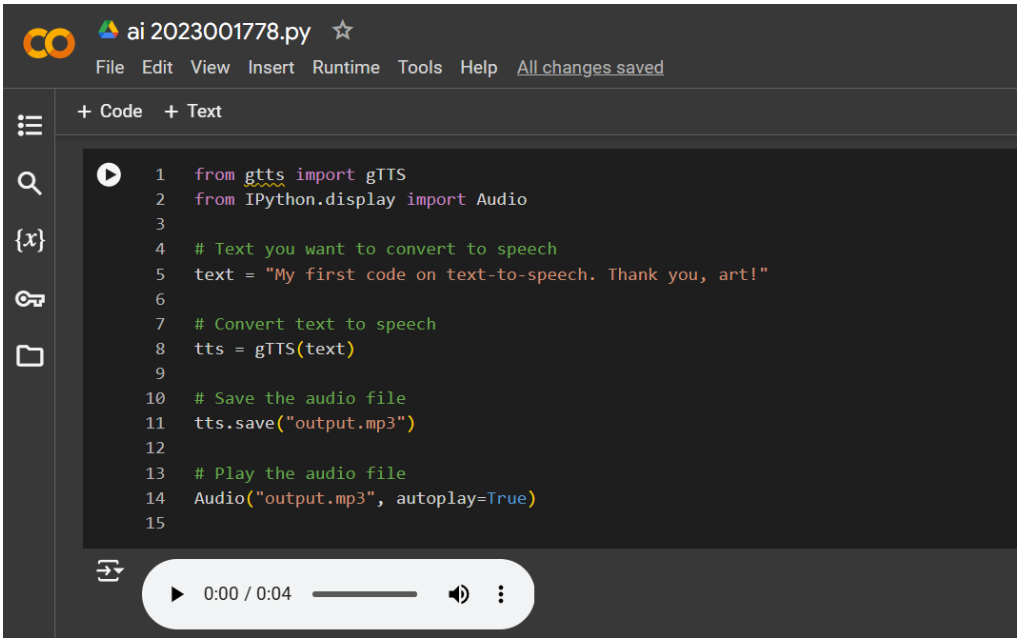
    - **Audio Preprocessing**: Audio signals are filtered and transformed to remove noise and improve clarity.
    - **Feature Extraction**: The model identifies key acoustic features in the speech, such as phonemes (basic units of sound).
    - **Language Modeling**: This stage interprets phonemes into words and applies contextual understanding to form coherent sentences.
    - **APIs for STT**: Common APIs include Google Speech-to-Text, IBM Watson, and Amazon Transcribe. These APIs can handle multiple languages, adapt to various accents, and are optimized for different environments (e.g., noisy settings).
- **Applications**:
    - **Virtual Assistants**: Voice-controlled interfaces like Google Assistant, Alexa, and Siri use STT to interpret user commands.
    - **Accessibility Tools**: STT helps the hearing impaired by transcribing conversations or enabling dictation.

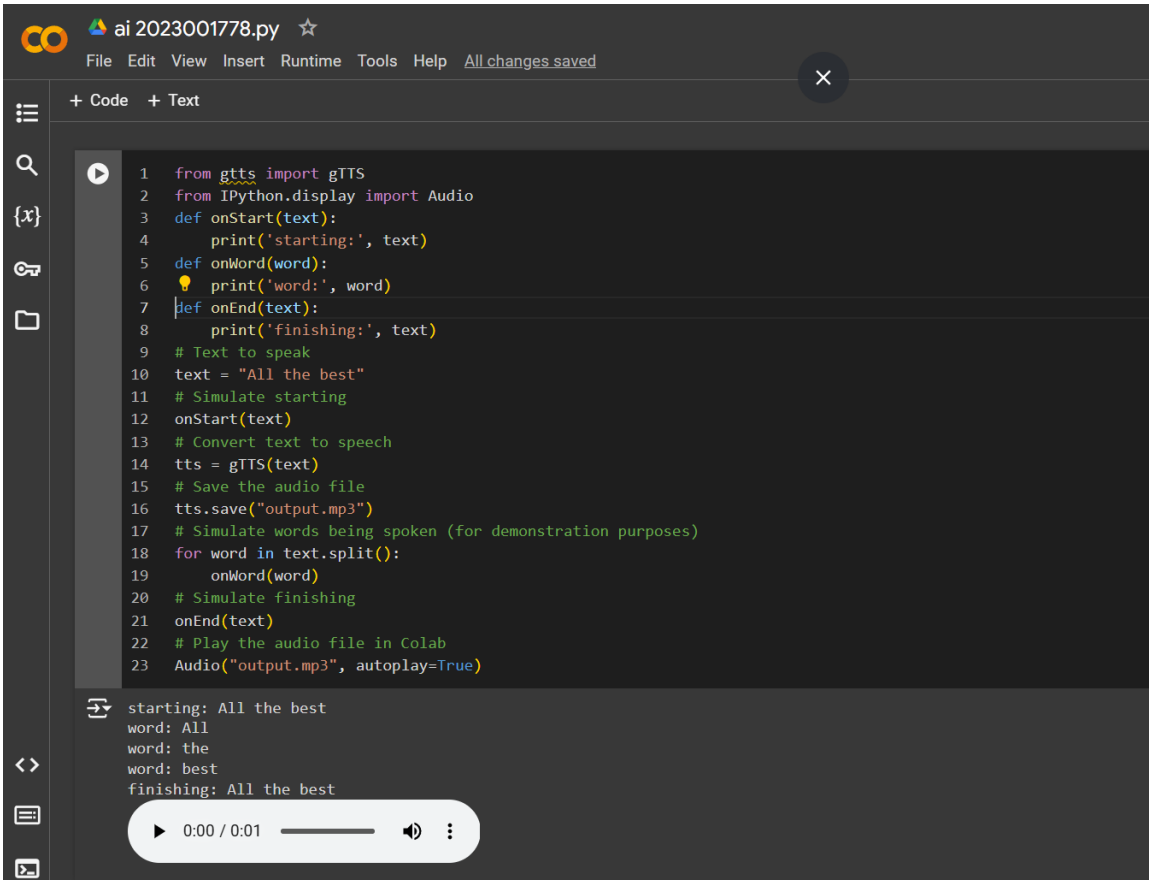
2. Text-to-Speech (TTS) Synthesis

- **Definition**: Text-to-Speech (TTS) converts written text into spoken audio. This technology uses AI models to analyze text and generate a natural-sounding voice output, enhancing user interaction with machines.
- **How It Works**:
    - **Text Analysis**: The TTS engine parses text and breaks it into phonetic units.

- **Prosody Modeling**: The model assigns tone, pitch, and rhythm to simulate natural speech patterns.
- **Voice Synthesis**: Using models like WaveNet or Tacotron, the text is converted into human-like audio output. Advanced TTS models can adjust for emotions, accents, or different voices.

Code and outputs:





7)Building a Chatbot Using Pandora Bots

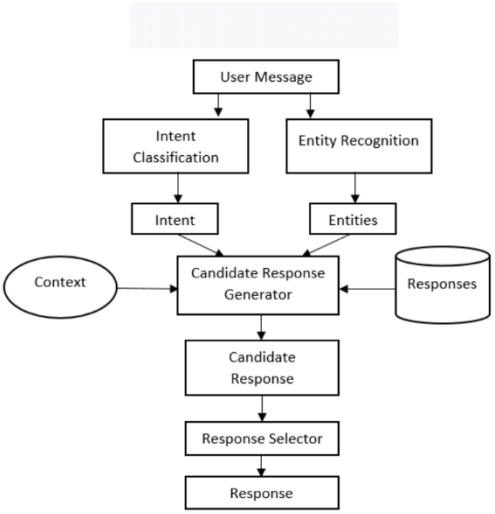
Introduction to ChatBot

- A chatbot is a software or computer program that simulates human conversation or "chatter" through text or voice interactions.
- Users in both business-to-consumer (B2C) and business-to-business (B2B) environments increasingly use chatbot virtual assistants to handle simple tasks.

Types of Chatbots

- Chatbots have varying levels of complexity, being either stateless or stateful. Stateless chatbots approach each conversation as if interacting with a new user. In contrast, stateful chatbots can review past interactions and frame new responses in context.

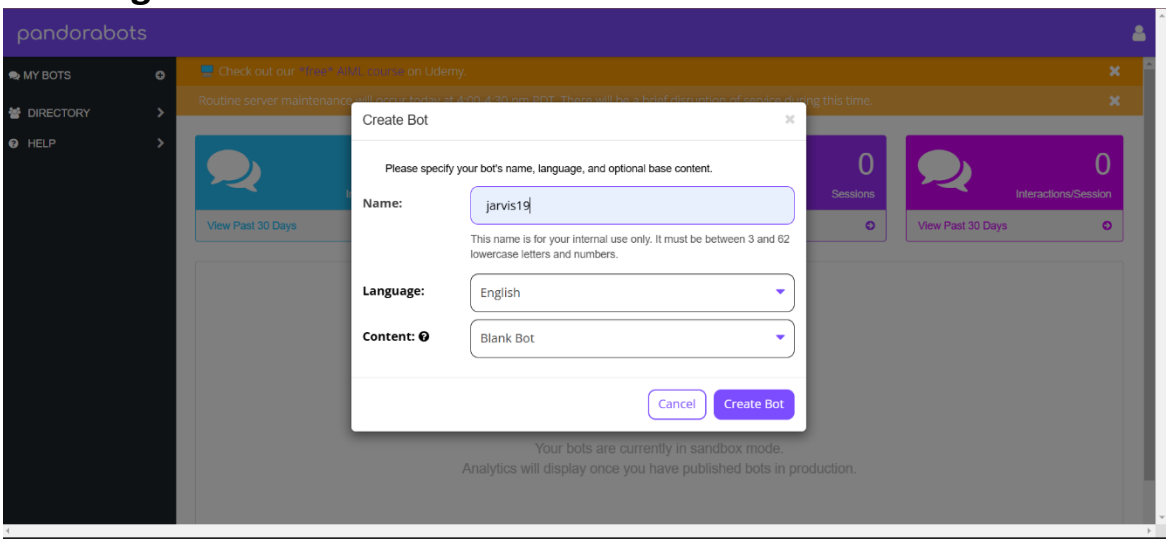
Architecture of Chatbot



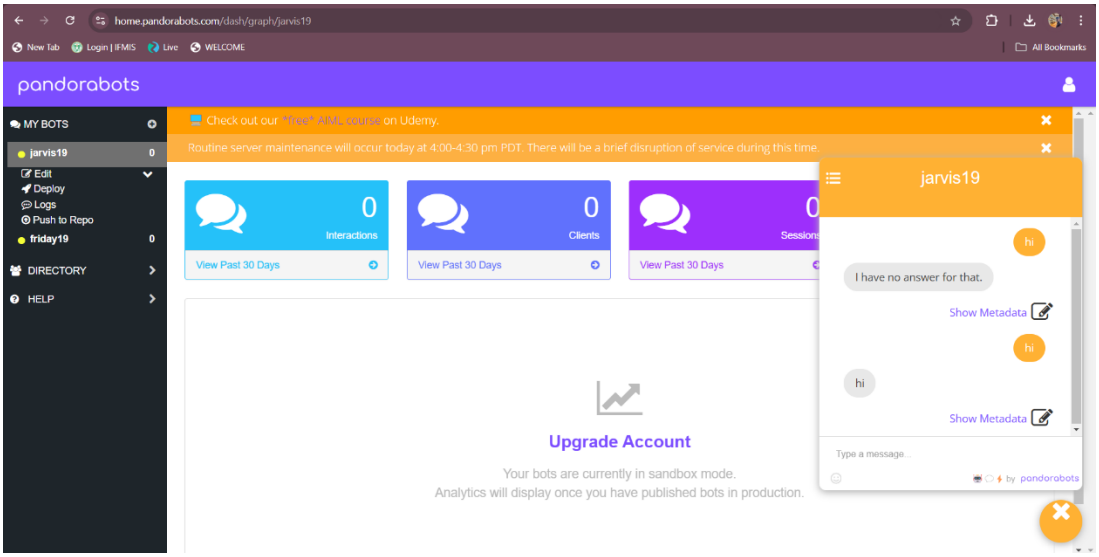
1. Blank Bot

- **Definition**: A Blank Bot is essentially a template or starter bot with minimal to no pre-set responses, allowing developers or users to customize it from scratch.
- **Purpose**: It's mainly used for testing or building customized chatbots from the ground up. Developers can program responses, intents, and flows according to their unique needs without the influence of any pre-configured logic.
- **Use Cases**: Blank Bots are often ideal when building a bot for a specific niche or highly specialized application where default responses might not apply.

Creating bot :



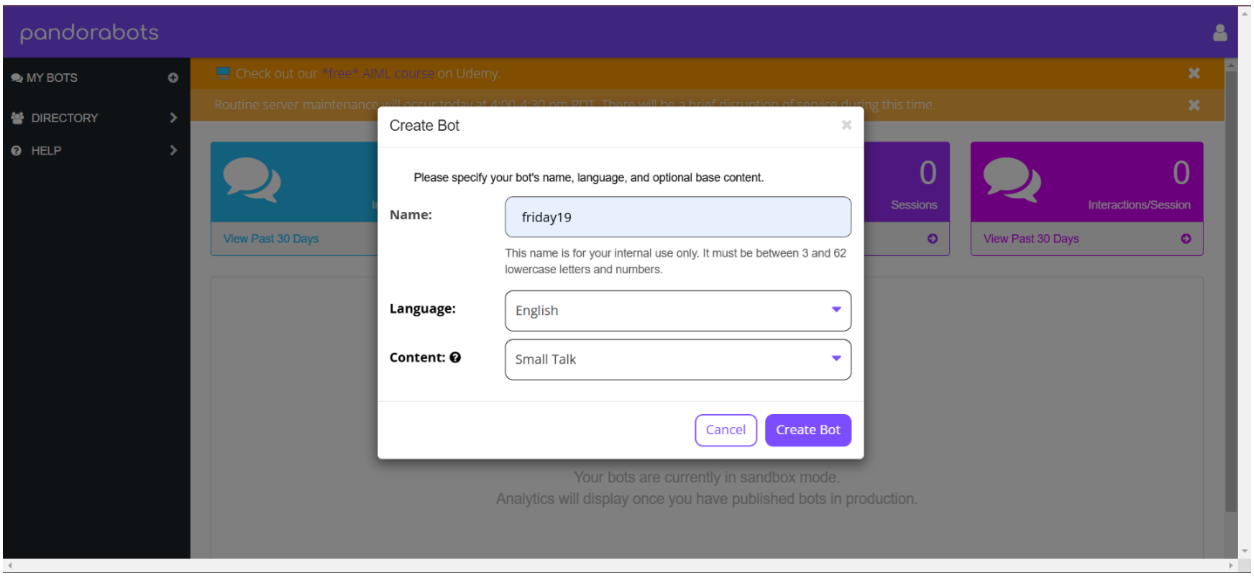
Using bot



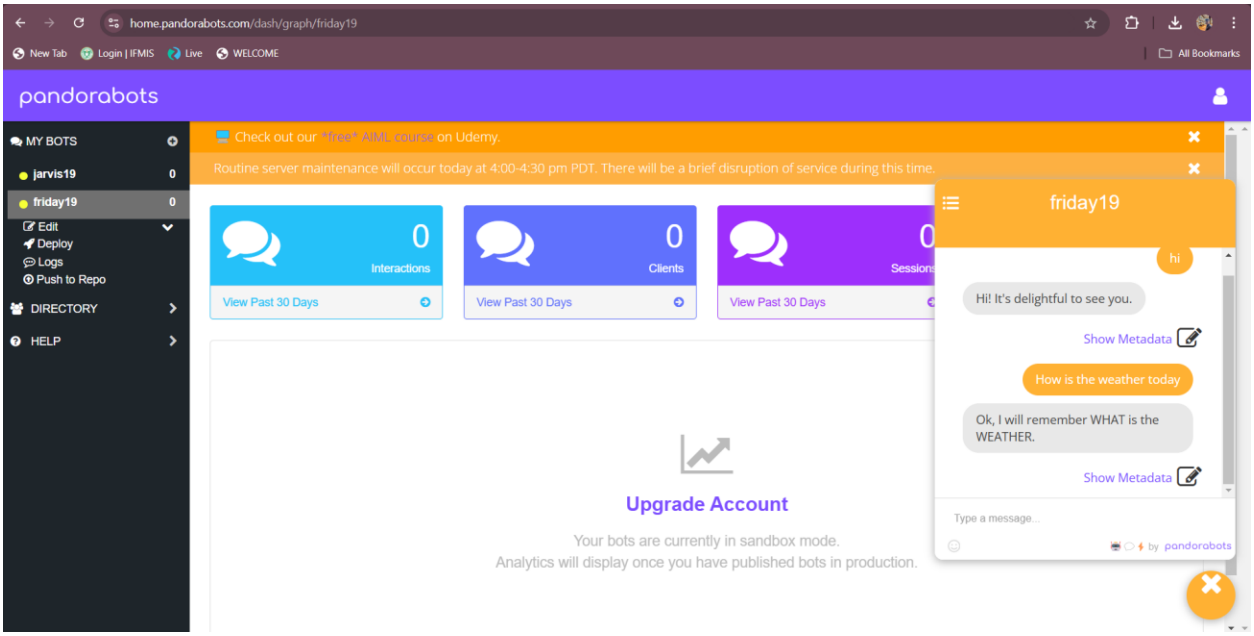
Small Bot

- **Definition**: A Small Bot typically has a limited set of features and responses, designed for simpler interactions or specific, narrow use cases.
- **Purpose**: Small Bots are useful for handling straightforward tasks or FAQs, where complex dialogue management or extensive capabilities are unnecessary. They are often lightweight and quicker to deploy due to their limited functionality.
- **Use Cases**: Small Bots are well-suited for customer support on common questions, simple information retrieval tasks, or handling single-function tasks like scheduling or basic inquiries.

Creating bot :



Using bot



Artificial_intelligence_applications_