CLOUD ENABLED ATTENDANCE SYSTEM USING FACE RECOGNITION

BATCH MEMBER

711221243048 : SWETHA C

PHASE 3 SUBMISSION DOCUMENT

TOPIC : START BUILDING THE CLOUD ENABLED ATTENDANCE SYSTEM FOR LOADING AND DATA PREPROCESSING.

INTRODUCTION :

- We are building a Smart Attendance System Using Face Recognition that can automatically take attendance using facial recognition technology.
- The system will use a camera to capture the face of each person and match it with the database to identify them.
- The system will store attendance records for each person in an Excel file and generates a report.

PRE-REQUISITES :

- To build this system, we need a basic understanding of programming languages such as Python and knowledge of facial recognition technology.
- Additionally, we require a computer with a webcam, internet connectivity, and a Python IDE.
- We also need to have libraries like OpenCV, face recognition, and NumPy pre-installed on our computers.

HOW ARE WE GOING TO BUILD THIS :

- We will build this face detection attendance system using Python programming language and facial recognition technology.
- We will use the OpenCV library to capture images from a webcam, detect faces, and extract facial features.
- We will then use the face_recognition library to recognize faces and compare them with the database to identify people.

- Finally, we will store the attendance records in a database and generate reports using NumPy.

REQUIREMENTS:

- **Hardware Requirements**:

  A computer with a webcam and internet connectivity is needed to run the system. The webcam should have a resolution of at least 720p to capture clear images.

- **Software Requirements**:

The system will be built using Python programming language. The required software includes a Python IDE like PyCharm or Spyder, OpenCV library, face_recognition library, and NumPy library.

- **Facial Recognition Algorithm**:

The system will use a facial recognition algorithm to recognize faces. The algorithm should be able to detect faces and extract facial features accurately.

- **User Interface:**

  The system should have a user-friendly interface to capture images, display results, and generate reports.

- **Attendance Management:**

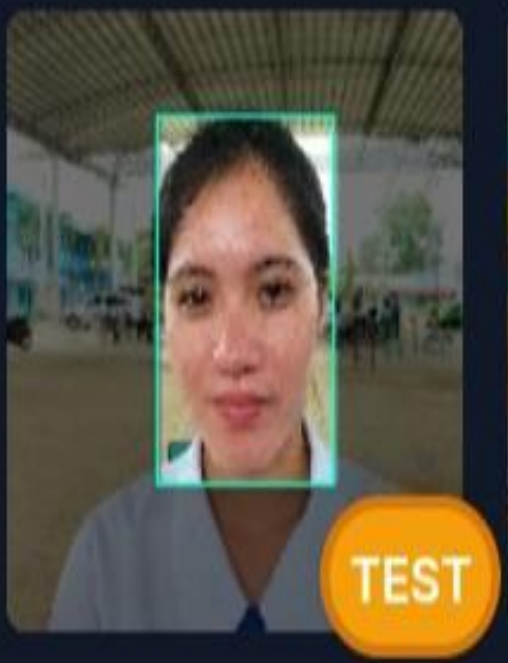
The system should be able to manage attendance records for each person, including the date and time of attendance.

GIVEN DATASET :

IMG20230...

IMG20230...

IMG20230...



IMG20230...

IMG20230...

IMG20230...

NECESSARY STEPS TO FOLLOW:

Step 1 : Data Acquisition

We will cover the process of acquiring data for building a Attendance Management System using Face Recognition.

```
# Import necessary libraries
Import cv2
# Initialize the camera
Cap = cv2.VideoCapture(0)
# Capture the image
While True:
    Ret, frame = cap.read()
    Cv2.imshow("frame", frame)
    If cv2.waitKey(1) == ord('q'):
        Break
Cap.release()
Cv2.destroyAllWindows()
```

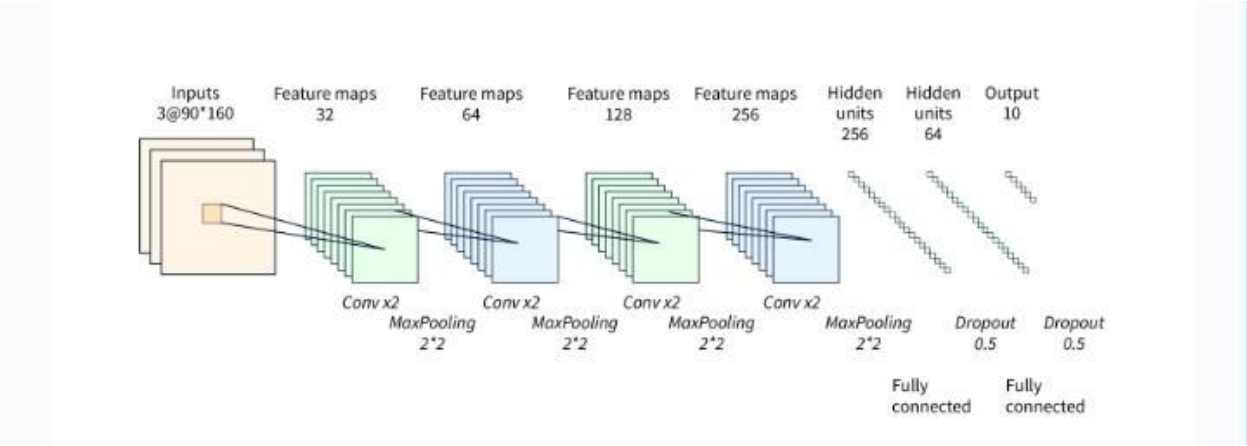
STEP 2 : DECIDE THEMETHODOLOGY

- The overall method combines several computer vision and deep learning techniques to perform real-time face recognition and attendance marking.

- It uses OpenCV's face detection algorithm and a pre-trained deep learning model for face recognition.
- The attendance log is stored in JSON format for easy access and manipulation.
- The code provides a user-friendly interface by displaying the video stream with the recognized names of individuals and a bounding box around their faces.

STEP 3 : DECIDE THE ALGORITHM

- The deep learning algorithm that is were using for the smart Attendance Management System is the face recognition model.
- This model uses a deep convolutional neural network (CNN) to extract features from facial images and learn to map these features to a unique embedding vector for each individual.
- The model is trained on a large dataset of facial images using a supervised learning approach, where it learns to minimize the difference between the predicted embedding vector and the true identity of the individual.
- The pre-trained face recognition model used in the above code is based on the ResNet architecture and has been trained on a large-scale face recognition dataset called VGGFace2.



STEP 4 : Image Acquisition

The first step is to acquire the images to recognize the faces. The below code represents the mechanism to perform this.

```
# Import necessary libraries

Import cv2

Import os

# Initialize the camera

Cap = cv2.VideoCapture(0)

# Create a directory for storing the dataset

If not os.path.exists(“dataset”):

    Os.makedirs(“dataset”)

# Create a dataset

Count = 0

While True:

    Ret, frame = cap.read()
```

```
    Cv2.imshow(“frame”, frame)
    If cv2.waitKey(1) == ord(‘q’):
        Break
    If cv2.waitKey(1) == ord(‘s’):
        Count += 1
        Filename = “dataset/user_” + str(count) + “.jpg”
        Cv2.imwrite(filename, frame)
Cap.release()
Cv2.destroyAllWindows()
```

STEP 5 : Storing the Face Embeddings

In this section, we will focus on how to store the face embeddings generated by the face recognition model.

```
# Import necessary libraries
Import face_recognition
Import os
Import numpy as np
# Create a directory for storing the embeddings
If not os.path.exists(“embeddings”):
    Os.makedirs(“embeddings”)
# Load the images
Image_paths = [os.path.join(“dataset”, f) for f in os.listdir(“dataset”)]
Images = []
```

```
For image_path in image_paths:
    Image = face_recognition.load_image_file(image_path)
    Images.append(image)
# Compute the face embeddings
Embeddings = []
For image in images:
    Face_locations = face_recognition.face_locations(image)
    Face_encodings = face_recognition.face_encodings(image,
face_locations)
    If len(face_encodings) == 1
   Embeddings.append(face_encodings[0])
# Save the embeddings
Np.savetxt("embeddings/embeddings.txt", embeddings)
```

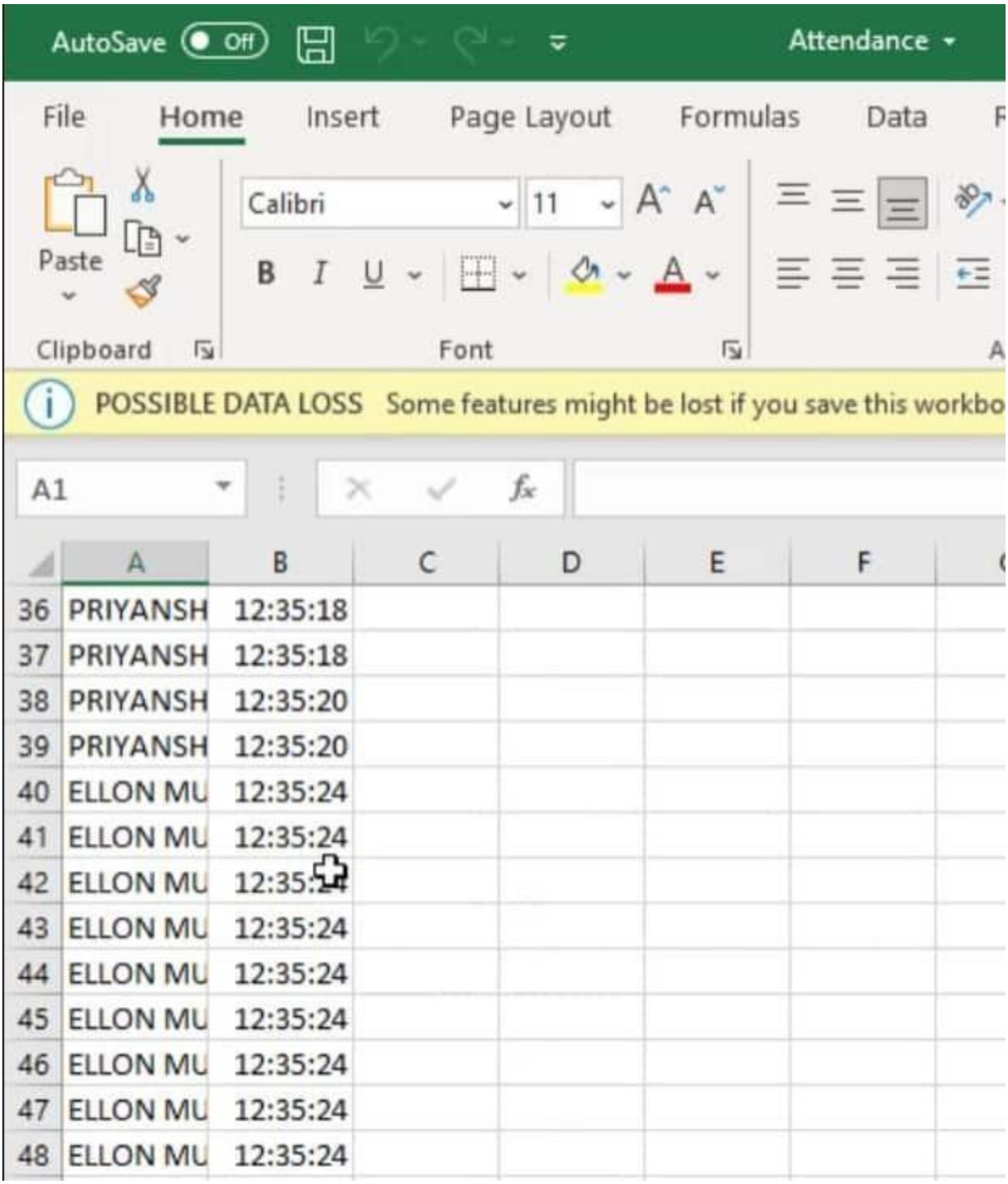
STEP 6 : Updating the Attendance to an Excel File

Here's an example code snippet to store the attendance data in an Excel file using the Pandas library:

```
Import pandas as pd
# Load the attendance log from the JSON file
With open('attendance.json', 'r') as f:
    Attendance_log = json.load(f)
# Convert the attendance log to a Pandas dataframe
Df = pd.DataFrame(attendance_log)
```

```
# Write the attendance data to an Excel file
Writer = pd.ExcelWriter('attendance_log.xlsx')
Df.to_excel(writer, index=False)
Writer.save()
```

STEP 7 : OUTPUT


AutoSave Off
Attendance
File Home Insert Page Layout Formulas Data
Paste
Calibri 11
Clipboard Font
POSSIBLE DATA LOSS Some features might be lost if you save this workbo
A1
A B C D E F
36 PRIYANSH 12:35:18
37 PRIYANSH 12:35:18
38 PRIYANSH 12:35:20
39 PRIYANSH 12:35:20
40 ELLON MU 12:35:24
41 ELLON MU 12:35:24
42 ELLON MU 12:35:24
43 ELLON MU 12:35:24
44 ELLON MU 12:35:24
45 ELLON MU 12:35:24
46 ELLON MU 12:35:24
47 ELLON MU 12:35:24
48 ELLON MU 12:35:24

CONCLUSION:

- In conclusion, the Smart Attendance Management System using Face Recognition is a highly innovative and efficient solution for attendance management in various institutions.
- The system uses state-of-the-art computer vision and deep learning algorithms to recognize individuals accurately and mark their attendance in real time.
- This eliminates the need for manual attendance management, which is prone to errors and can be time-consuming.
- The project also offers a user-friendly interface that displays live video streams and attendance logs, making it easy to use and understand.
- Overall, this project has great potential to revolutionize attendance management systems in various institutions and improve their efficiency and accuracy.