A Mini Project Report

ON

REAL-TIME FACE ATTENDANCE SYSTEM

Submitted to

DEAPRTMENT
of

CSE (AI&ML)
By

Adithya Pisupati

245321748111

Under the guidance

Of

Mr. M. Manohar Rao

Assistant Professor

NEIL GOGTE INSTITUTE OF TECHNOLOGY
Kachavanisingaram Village, Hyderabad, Telangana 500058.

FEBRUARY 2024

NEIL GOGTE INSTITUTE OF TECHNOLOGY

A Unit of Keshav Memorial Technical Education (KMTES)
Approved by AICTE, New Delhi & Affiliated to Osmania University, Hyderabad

CERTIFICATE

This is to certify that the Mini project work entitled "**REAL-TIME FACE ATTENDANCE SYSTEM"** *is a bonafide work carried out by* **P.Adithya (245321748111)** of III-year V semester **Bachelor of Engineering** *in CSE(***AIML)** *by Osmania University, Hyderabad during the academic year* **2023-2024** *is a record of bonafide work carried out by him. The results embodied in this report have not been submitted to any other University or Institution for the award of any degree.*

Internal Guide
Mr. M. Manohar Rao
Assistant Professor

Head of Department
Dr. T. Prem Chander
Associate Professor

External

NEIL GOGTE INSTITUTE OF TECHNOLOGY
A Unit of Keshav Memorial Technical Education (KMTES)
Approved by AICTE, New Delhi & Affiliated to Osmania University, Hyderabad

DECLARATION

I hereby declare that the Mini Project Report entitled, "**REAL-TIME FACE ATTENDANCE SYSTEM**" submitted for the B.E degree is entirely my work and all ideas and references have been duly acknowledged. It does not contain any work for the award of any other degree.

Date:

P.Adithya 245321748111

ACKNOWLEDGEMENT

I am happy to express my deep sense of gratitude to the principal of the college **Dr. R. Shyam Sunder, Professor**, Neil Gogte Institute of Technology, for having provided me with adequate facilities to pursue our project.

I would like to thank**, Dr. T. Prem Chander**, **Head of the Department,** CSE(AIML), Neil Gogte Institute of Technology, for having provided the freedom to use all the facilities available in the department, especially the laboratories and the library.

I would also like to thank my internal guide **Mr. M. Manohar Rao, Assistant Professor** for his technical guidance & constant encouragement.

I sincerely thank my seniors and all the teaching and non-teaching staff of the Department of Computer Science & Engineering for their timely suggestions, healthy criticism and motivation during this work.

Finally, I express our immense gratitude with pleasure to the other individuals who have either directly or indirectly contributed to my need at the right time for the development and success of this work.

ABSTRACT

The "Real-Time Face Attendance System" introduces a transformative solution to traditional attendance management, seamlessly merging advanced facial recognition technology with machine learning algorithms. By employing the FaceNet model for face detection and the Haar cascade classifier for face extraction, the system ensures accurate identification while overcoming challenges associated with conventional methods. The project's contactless and realtime attendance recording capabilities not only enhance efficiency but also address concerns related to authentication errors and time-consuming operations. The intuitive user interface further contributes to its user-friendly design, making it applicable across educational institutions, corporate settings, and public facilities. Through rigorous testing, the system demonstrates robustness and precision, underscoring its potential to revolutionize attendance management practices in diverse environments.

TABLE OF CONTENTS

List of Figures

V

CHAPTER – 1 INTRODUCTION

1.1 PROBLEM STATEMENT

Traditional attendance systems encounter challenges, including authentication errors, timeconsuming processes, and hygiene concerns associated with touch-based methods. In response to these limitations, this project proposes a Real-Time Face Attendance System that harnesses facial recognition technology to provide a contactless, efficient, and precise solution for attendance recording. The system aims to streamline processes in educational, corporate, and public settings, offering an innovative approach to address existing issues and improve overall attendance management practices.

1.2 MOTIVATION

The driving force behind undertaking the "Real-Time Face Attendance System" project emanates from the critical role attendance plays in the contemporary educational landscape. With colleges enforcing stringent attendance policies and students facing repercussions for falling below attendance thresholds, the need for an accurate, efficient, and fair attendance management system becomes paramount. Personal observations of instances where students are erroneously marked absent despite attending classes underscore the limitations of traditional methods. This project is a proactive response to bridge the gap in attendance tracking, ensuring precision and fairness in recording students' participation. By harnessing advanced facial recognition technology, the system aims not only to streamline attendance procedures but also to contribute to the ongoing discourse on attendance management in educational institutions, addressing challenges and fostering a more equitable educational environment.

1.3 SCOPE

Embracing the challenge of modernizing attendance management, the "Real-Time Face Attendance System" project seeks to introduce an innovative solution harnessing facial recognition technology. With a focus on efficiency and precision, this project aims to redefine attendance recording practices, offering a contactless alternative to conventional methods. Beyond the confines

of educational institutions, the system's adaptability extends to corporate settings and public facilities, promising a versatile tool for attendance tracking. By incorporating FaceNet and HaarCascadeClassifier technologies, the project explores applications beyond attendance, envisioning contributions to security systems and personalized user experiences.

1.4 OUTLINE

The "Real-Time Face Attendance" project integrates advanced FaceNet and HaarCascadeClassifier models for efficient real-time face detection, prioritizing convenience in the training phase. Leveraging one-shot learning, the model requires just a single photo for accurate face detection, enabling the use of passport-size photos for streamlined training of students and faculty. This convenient approach enhances the project's adaptability, ensuring precise attendance recording in various environments, thanks to the model's one-shot learning capabilities.

CHAPTER – 2
LITERATURE SURVEY

2.1 EXISTING SYSTEM:

The existing attendance systems primarily rely on manual processes or conventional biometric methods, often encountering challenges such as authentication errors, time-consuming operations, and limited scalability. Moreover, the dependence on physical touch-based systems has raised concerns regarding hygiene and adaptability in various environments.

2.2 PROPOSED SYSTEM:

Incorporating cutting-edge facial recognition technologies, the proposed "Real-Time Face Attendance" system stands as a significant leap in attendance management. By harnessing the capabilities of FaceNet and HaarCascadeClassifier models, the system ensures swift and precise attendance recording through real-time face detection. A standout feature is the one-shot learning ability, streamlining training with just a single photo per individual—perfect for utilizing passport-size photos. This approach enhances adaptability across diverse environments, offering a contactless and efficient attendance management solution. The proposed system seeks to revolutionize traditional attendance practices, addressing manual tracking challenges and providing a user-friendly and accurate alternative for educational institutions, corporate settings, and public facilities.

CHAPTER - 3

SOFTWARE REQUIREMENTS SPECIFICATION

3.1 OVERALL DESCRIPTION:

This SRS is an overview of the whole project scenario. This document is to present a detailed description of the course management system. It will explain the purpose and features of the system, the interfaces of the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for both stakeholders and developers of the system.

3.2. OPERATING ENVIRONMENT:

3.2.1 Software Requirements:

Operating System	:	Windows 7 (Min)
Front End	:	Python
Back End	:	Python
Database	:	Microsoft Excel

3.2.2 Hardware Requirements:

Processor	:	Intel® Pentium® G4560 (Min)
Speed	:	3.5 GHz (Min)
RAM	:	8 GB (Min)
Hard Disk	:	50 GB (Min)
Graphics Card	:	Integrated (Intel UHD Graphics 630)

3.3. FUNCTIONAL REQUIREMENTS:

The operating environment for the Real-Time Face Attendance System project encompasses a set of technologies and tools crucial for its development, deployment, and utilization. The system operates primarily within a Python environment, utilizing Jupyter Notebook for code development and experimentation. The following libraries were installed to run the code successfully:

- Pandas: Used for data manipulation and analysis, including handling Excel files for attendance recording.
- OpenCV-Python: Employed for image processing tasks such as face detection and extraction using the Haar Cascade classifier.
- Keras: Utilized for building and training neural networks, specifically for integrating the FaceNet model for face recognition.
- TensorFlow: Integrated with Keras for efficient computation during model training and inference.
- NumPy: Utilized for numerical computing tasks, including array manipulation and mathematical operations.
- PIL (Python Imaging Library): Used for image processing tasks such as resizing and conversion between different image formats.

Additionally, the Haar Cascade classifier XML file was downloaded from the internet and utilized for face extraction during the preprocessing stage. The FaceNet library was employed for face detection, providing accurate embeddings for face recognition tasks.

The system's functionality includes real-time face detection and recognition, with the ability to capture attendance data efficiently. Upon detecting a face, the system matches it with preregistered faces using the FaceNet model, recording attendance along with timestamps in an Excel spreadsheet. The system ensures accuracy, efficiency, and user-friendliness, contributing to streamlined attendance management practices in various settings.

Overall, the Real-Time Face Attendance System is designed to be versatile, scalable, and optimized for seamless interaction and utilization in educational, corporate, and public environments.

3.4 NON - FUNCTIONAL REQUIREMENTS:

Performance:

- The system aims to achieve real-time face detection and recognition, ensuring prompt responses to user interactions.

- The prediction process for attendance recording should ideally take place within a few seconds to maintain a seamless user experience during face recognition.

Scalability:

- The system is designed to efficiently handle increasing demands, such as a growing number of users or concurrent face recognition requests, without compromising performance.

- Flexible resource management enables the system to scale up or down based on varying loads, ensuring smooth operation even under high traffic conditions.

Usability:

- The user interface of the attendance system is crafted to be intuitive and user-friendly, catering to users of different technical backgrounds.

- Effortless navigation and clear instructions allow users to easily input data and interpret attendance records without requiring extensive training or technical knowledge.

Reliability:

- High availability is crucial for the system to ensure minimal downtime and consistent performance.

- The system's reliability ensures dependable face recognition outcomes based on provided inputs, fostering trust among users and stakeholders.

Maintainability:

- The system's architecture is designed for ease of maintenance, with clear documentation and adherence to coding best practices.

CHAPTER-4

SYSTEM DESIGN

4.1 Use case Diagram:

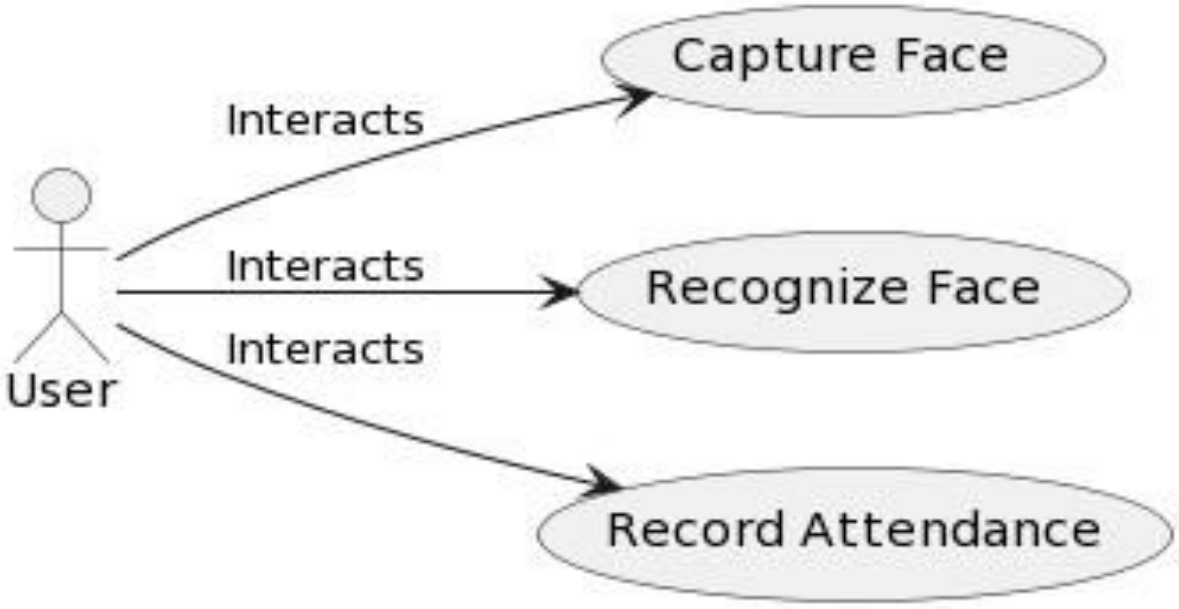

Fig 4.1: Use case diagram

The "User" actor initiates interactions with the system through three key use cases: "Capture Face," "Recognize Face," and "Record Attendance." The "Capture Face" use case involves users capturing their facial image for recognition, while "Recognize Face" encompasses the system's ability to identify and authenticate the captured faces using facial recognition algorithms. Subsequently, the "Record Attendance" use case illustrates the system's capability to log attendance records based on the recognized faces. The diagram provides a concise visualization of the primary user actions and system responses in the context of face recognition and attendance.

4.2 Class Diagram:

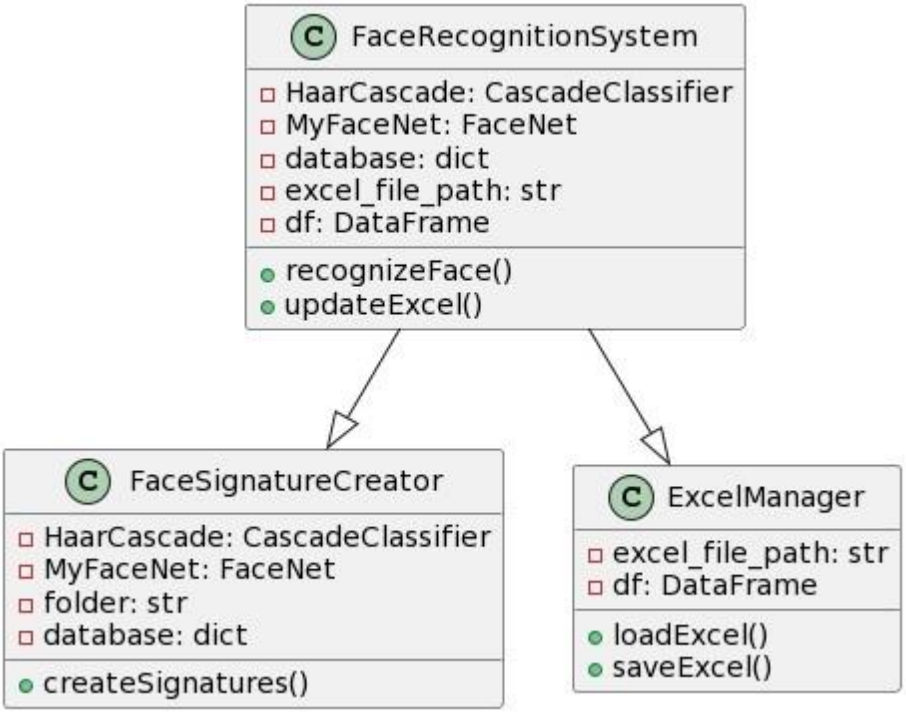

Fig.4.2 : Class diagram

The Class Diagram provides an overview of the key classes and their relationships within the Real-Time Face Attendance System. The central class, "FaceRecognitionSystem," encapsulates the HaarCascade, MyFaceNet, database, and Excel-related functionalities. It collaborates with two specialized classes: "FaceSignatureCreator" responsible for creating face signatures, and "ExcelManager" handling Excel file operations. The relationships depict how these classes are interconnected, highlighting the system's modular and organized structure. This diagram aids in understanding the system's composition and the responsibilities assigned to each class.

4. 3 Sequence Diagram:

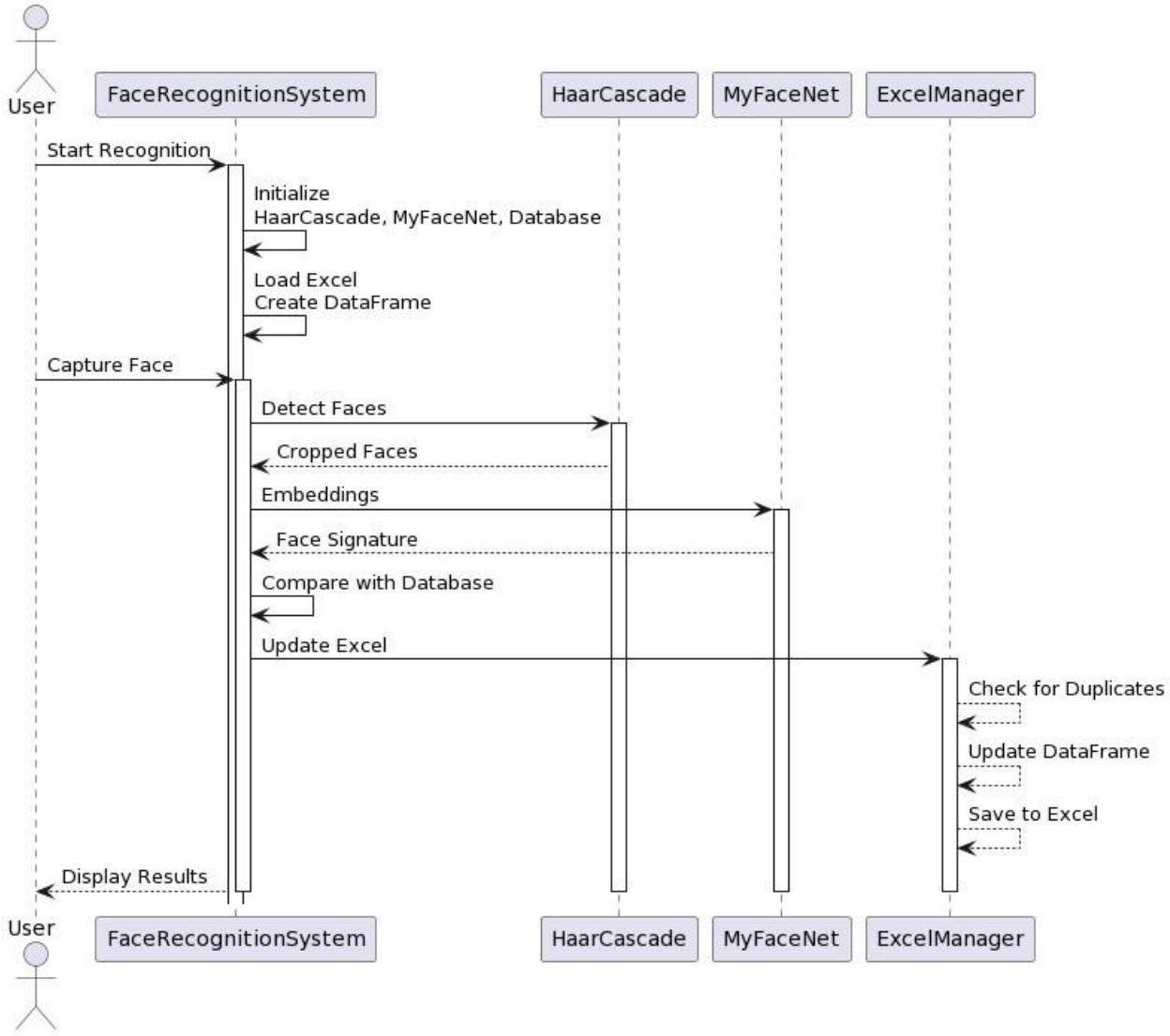

Fig4.3 : Sequence Diagam

The Sequence Diagram illustrates the dynamic interactions and sequence of events during the recognition process in the Real-Time Face Attendance System. It starts with the "User" initiating face recognition. The system activates various components, such as HaarCascade for face detection and MyFaceNet for generating face embeddings. The system then compares the face signature with the database, updates the Excel file, and finally displays the results to the user.

CHAPTER - 5 IMPLEMENTATION

5.1 SAMPLE CODE

5.1.1. Create_Signature_Haar.py

```
import os from os import listdir
from PIL import Image from numpy
import asarray from numpy import
expand_dims from matplotlib import
pyplot from keras.models import
load_model from keras_facenet
import FaceNet import numpy as np
import pickle import cv2
HaarCascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml') MyFaceNet
= FaceNet()
folder='faces/' database = {} for filename in listdir(folder):
path = folder + filename gbr1 = cv2.imread(folder +
filename) croppedFace =
HaarCascade.detectMultiScale(gbr1,1.1,4) if
len(croppedFace)>0:
        x1, y1, width, height = croppedFace[0]

  else:
       x1, y1, width, height = 1, 1, 10, 10
x1, y1 = abs(x1), abs(y1) x2, y2 =
x1 + width, y1 + height
    gbr = cv2.cvtColor(gbr1, cv2.COLOR_BGR2RGB)
    gbr = Image.fromarray(gbr)                gbr_array
= asarray(gbr)
```

```
    face = gbr_array[y1:y2, x1:x2]                face
= Image.fromarray(face)                face =
face.resize((160,160))    face = asarray(face)    face
=  expand_dims(face,  axis=0)          signature  =
MyFaceNet.embeddings(face)
database[os.path.splitext(filename)[0]]=signature
with      open("data.pkl",      "wb")      as      f:
pickle.dump(database, f) print(database)
```

5.1.2 Recognize_Face_Haar.py:

```
from PIL import Image from
keras.models import load_model from
keras_facenet import FaceNet import
numpy as np from numpy import
asarray from numpy import
expand_dims import pandas as pd
import datetime

import pickle import
cv2

HaarCascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml') MyFaceNet
= FaceNet()
with open("data.pkl", "rb") as myFile:
database = pickle.load(myFile) #
Load or create an Excel file
excel_file_path = 'face_data.xlsx'
try:
```

```
    df = pd.read_excel(excel_file_path) except
FileNotFoundError:
    # If the file doesn't exist, create a new DataFrame with columns        df =
pd.DataFrame(columns=['Name', 'Time', 'Date'])        df.to_excel(excel_file_path,
index=False)    # Save the DataFrame to the Excel file              df =
pd.read_excel(excel_file_path)   # Reload the DataFrame after saving cap =
cv2.VideoCapture(0)
cv2.namedWindow('res', cv2.WINDOW_NORMAL)
# Display window to ensure it's not minimized
_, gbr1 = cap.read()
cv2.imshow('res',
gbr1) cv2.waitKey(1)
while True:
    _, gbr1 = cap.read()     croppedFace =
HaarCascade.detectMultiScale(gbr1, 1.1, 4)     if
len(croppedFace) > 0:

        x1, y1, width, height = croppedFace[0]
    else:
        x1, y1, width, height = 1, 1, 10, 10
x1, y1 = abs(x1), abs(y1)     x2, y2 =
x1 + width, y1 + height
    gbr = cv2.cvtColor(gbr1, cv2.COLOR_BGR2RGB)
    gbr = Image.fromarray(gbr)     gbr_array
= asarray(gbr)     face = gbr_array[y1:y2,
x1:x2]     face = Image.fromarray(face)
face = face.resize((160, 160))     face =
```

```
asarray(face)     face = expand_dims(face,
axis=0)     signature =
MyFaceNet.embeddings(face)     min_dist =
100
   identity = ' '
   for key, value in database.items():
      dist = np.linalg.norm(value - signature)
if dist < min_dist:                min_dist = dist
identity = key
   # Get current time and date     current_time =
datetime.datetime.now().strftime("%H:%M:%S")     current_date =
datetime.datetime.now().strftime("%Y-%m-%d")     # Check if the
DataFrame is empty or if the person's name already exists for the
current date

   if df.empty or not df[(df['Name'] == identity) & (df['Date'] == current_date)].empty:
      # Check for duplicates within the DataFrame for the same date and name
      duplicate_index = df[(df['Name'] == identity) & (df['Date'] ==
current_date)].index        if not duplicate_index.empty:
         # Remove duplicate entries for the same date and name
df = df.drop(duplicate_index)
      # Create a new DataFrame for the current entry
      new_entry = pd.DataFrame({'Name': [identity], 'Time': [current_time], 'Date':
[current_date]})
      # Concatenate the new entry with the existing DataFrame
df = pd.concat([df, new_entry], ignore_index=True)
      #              Update              Excel              file
df.to_excel(excel_file_path, index=False)
```

```
    cv2.putText(gbr1, identity, (100, 100), cv2.FONT_HERSHEY_SIMPLEX, 1,
(255, 255, 0), 2, cv2.LINE_AA)

    cv2.rectangle(gbr1, (x1, y1), (x2, y2), (0, 255, 0), 2)     cv2.imshow('res',

gbr1)     k = cv2.waitKey(5) & 0xFF     if k == 27 or

cv2.getWindowProperty('res', cv2.WND_PROP_VISIBLE) < 1:

        break

cv2.destroyAllWindows()

cap.release()
```

CHAPTER - 6
SCREENSHOTS

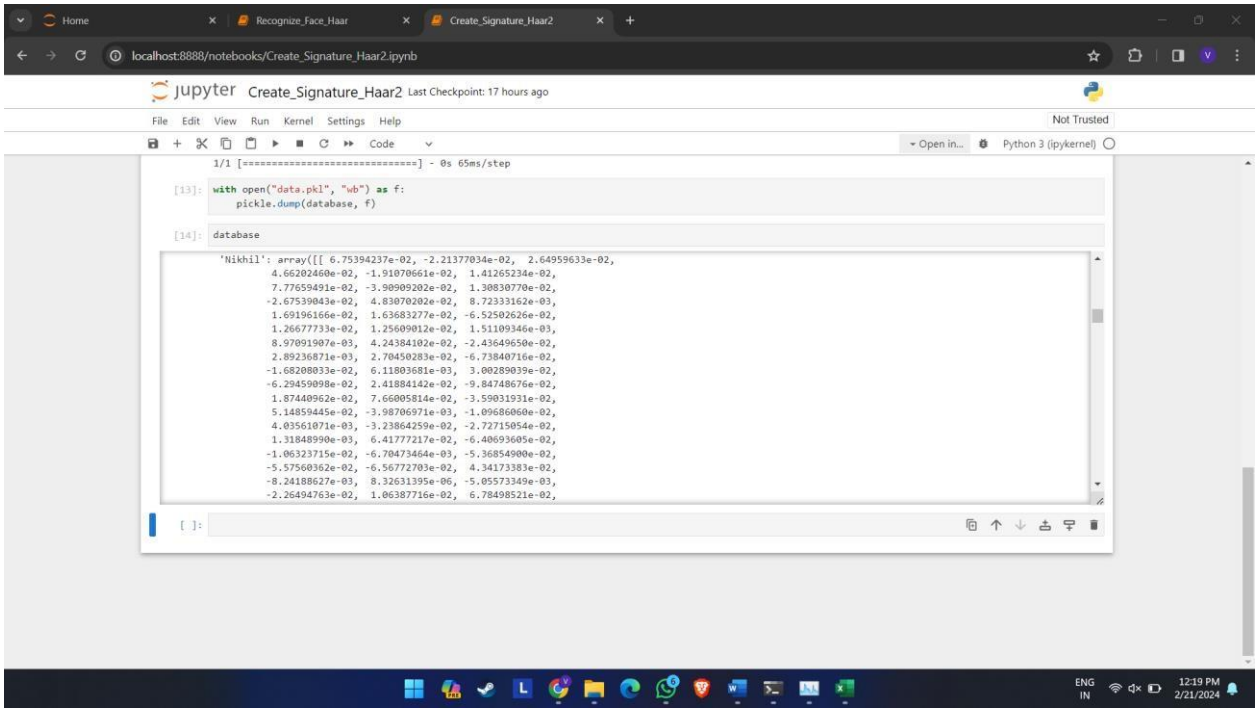

Figure 7.1: Generation of face signatures

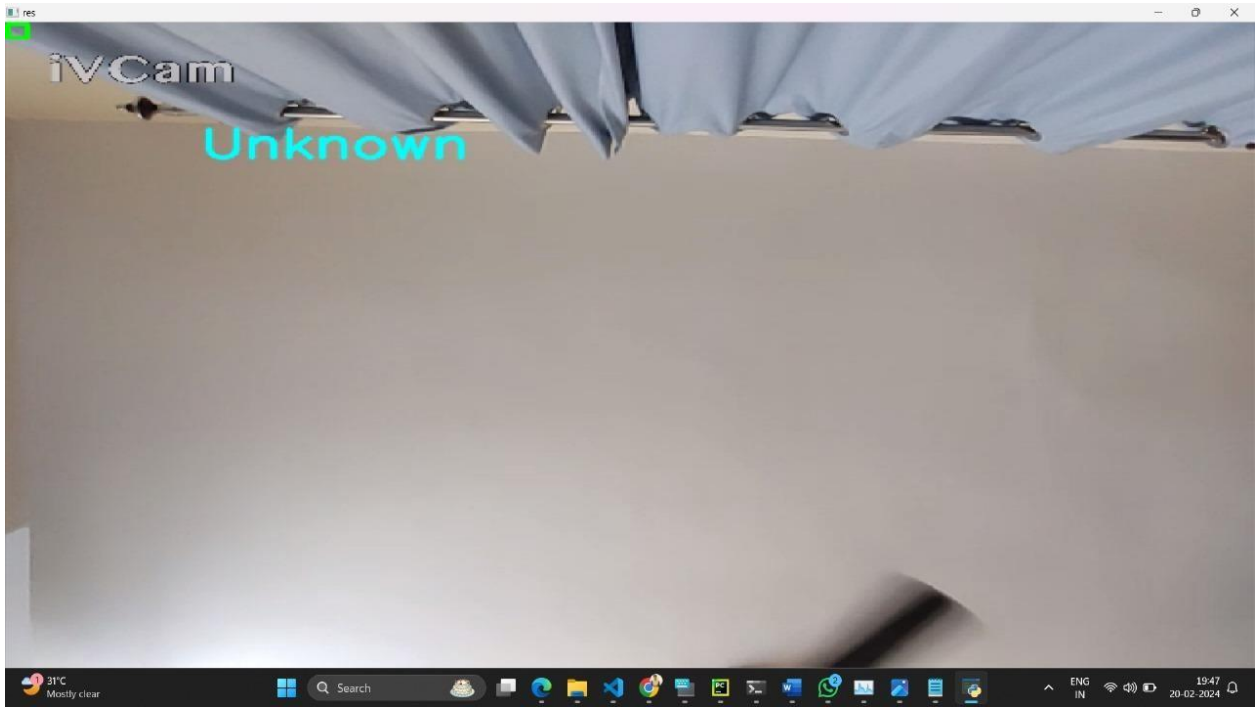

Figure 7.2: No faces detected

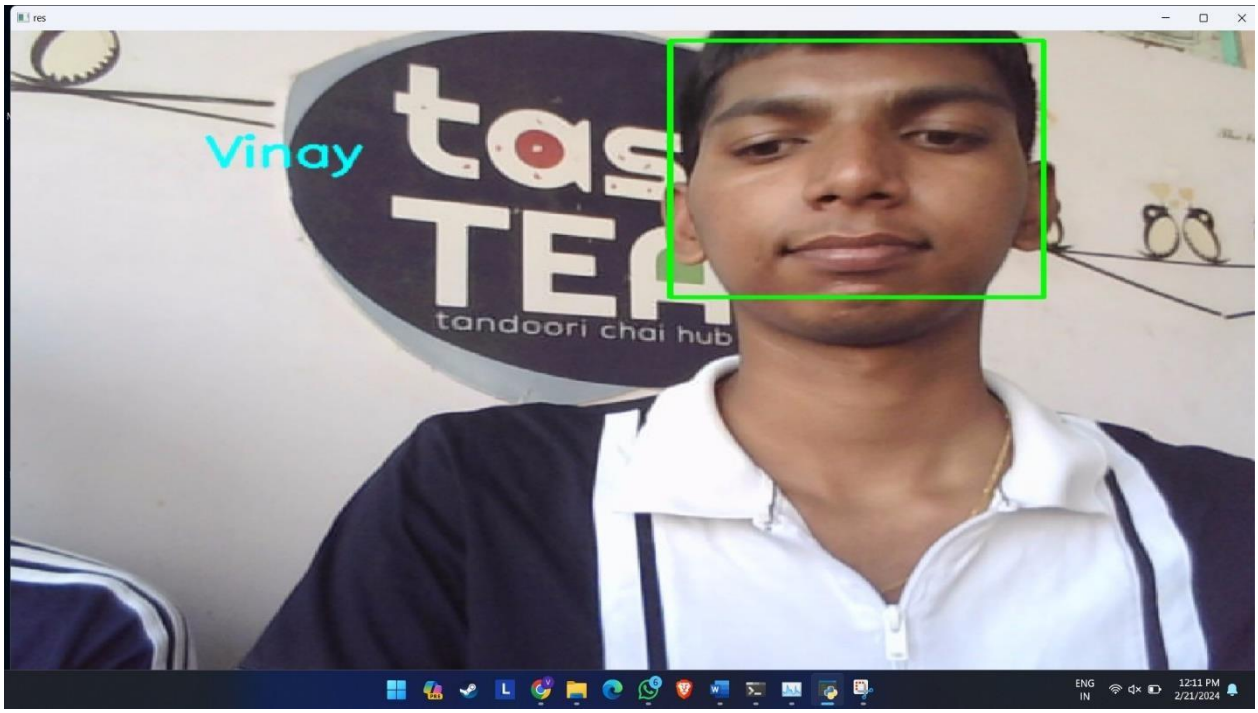

Figure 7.2: Face detected

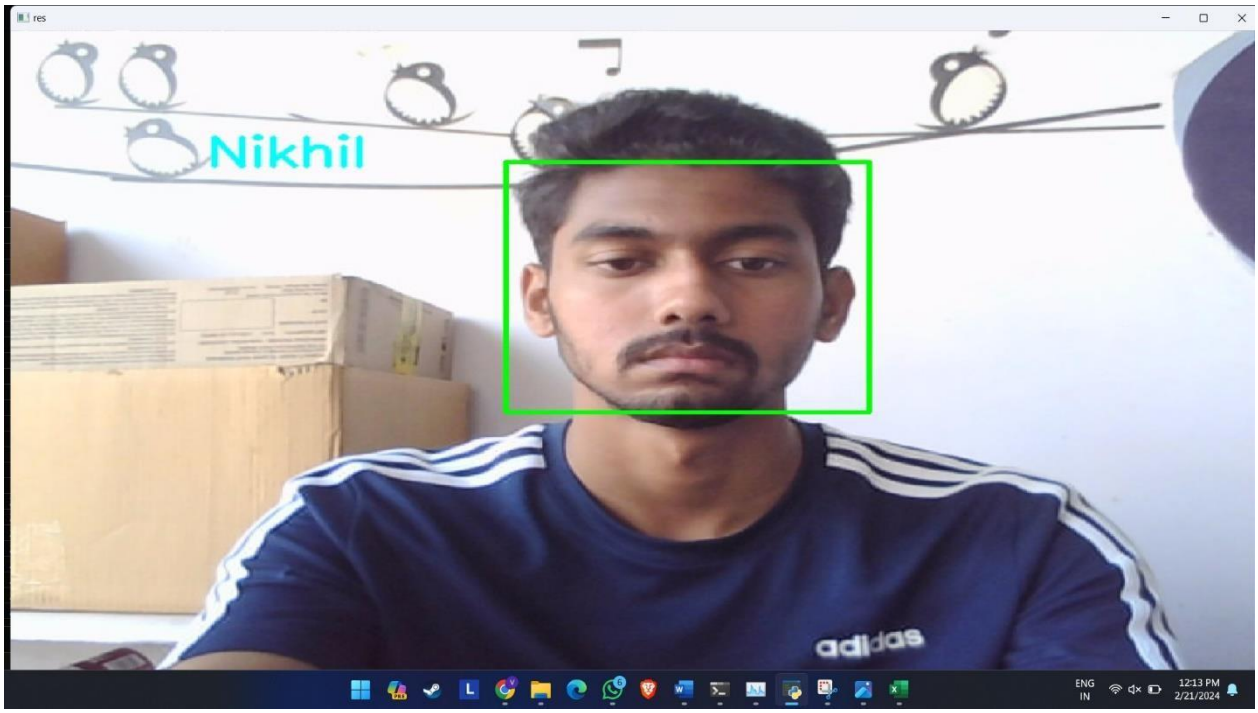

Figure 7.3: Face detected

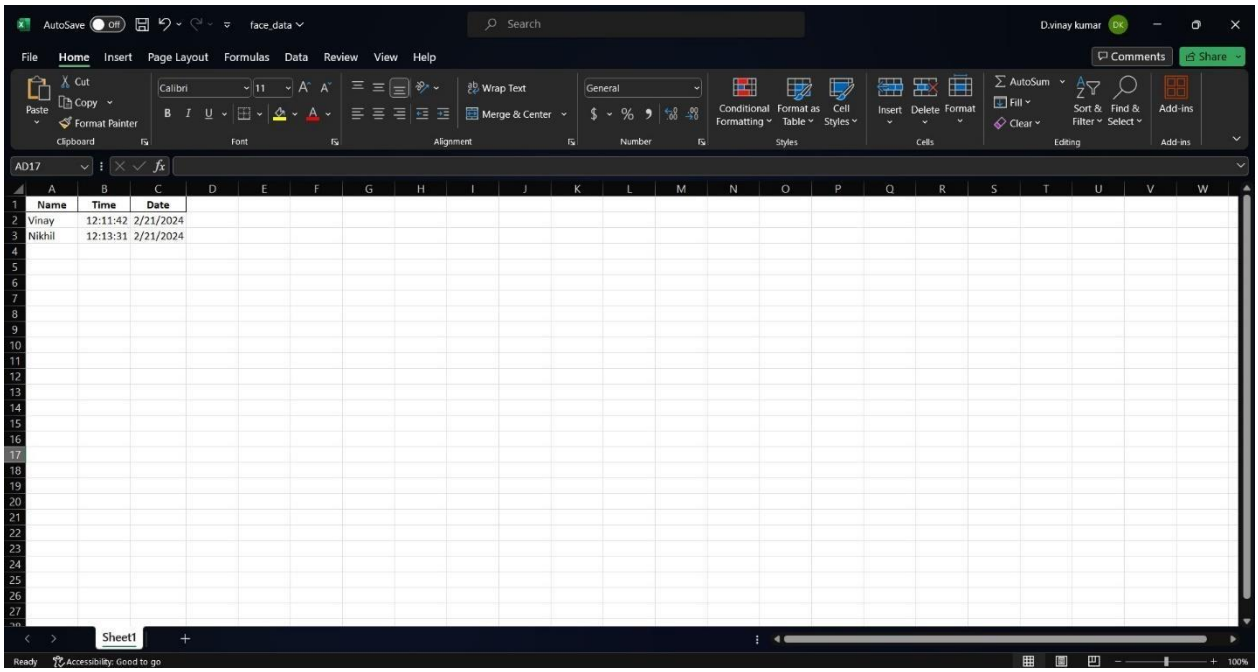

Figure 7.4: Updated excel sheet for attendance

CHAPTER - 7
CONCLUSION AND FUTURE SCOPE

In conclusion, the Real-Time Face Attendance System represents a significant advancement in attendance management, addressing critical challenges inherent in traditional methods. By harnessing state-of-the-art facial recognition technologies such as FaceNet and HaarCascadeClassifier, the system offers a contactless, efficient, and precise solution for recording attendance across various sectors. Its successful implementation underscores its potential to streamline processes and enhance fairness and accuracy in attendance tracking. Moreover, the adaptability of the system makes it well-suited for widespread adoption, presenting a user-friendly alternative that aligns with the evolving needs of educational institutions, corporate environments, and public facilities.

Looking forward, the future scope of the Real-Time Face Attendance System encompasses several avenues for further enhancement and application. Firstly, ongoing research and development efforts could focus on refining the system's accuracy and robustness through continuous optimization and validation. Additionally, exploration into advanced machine learning techniques and cloud-based infrastructure could enable real-time data analysis and predictive analytics, facilitating proactive attendance management strategies. Furthermore, the system's versatility opens opportunities for integration with complementary technologies, such as biometric authentication solutions and personalized user experiences, paving the way for comprehensive attendance management ecosystems that cater to diverse needs and use cases. Collaboration with industry partners and stakeholders will be instrumental in driving innovation and fostering the adoption of cutting-edge solutions to address evolving challenges in attendance tracking and beyond.

BIBLIOGRAPHY

[1] Nemuelpah. "Face Recognition with FaceNET." GitHub, https://github.com/nemuelpah/FaceRecognition-with-FaceNET.

[2] Ageitgey. "Face Recognition." GitHub, https://github.com/ageitgey/face_recognition.

[3] David Sandberg. "FaceNet." GitHub, https://github.com/davidsandberg/facenet.

[4] OpenCV. "Cascade Classifier." OpenCV Documentation, https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html.

[5] Analytics Vidhya. "Haar Cascades Explained." Medium, https://medium.com/analyticsvidhya/haar-cascades-explained-38210e57970d.

[6] Prabhu, Raghav. "Understanding of Convolutional Neural Network (CNN) - Deep Learning." Medium, https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neuralnetwork-cnn-deep-learning-99760835f148.

[7] ComputerVisionEng. "Face Attendance System." GitHub, https://github.com/computervisioneng/face-attendance-system.

[8] MiniVision AI. "Silent Face Anti-Spoofing." GitHub, https://github.com/minivision-ai/SilentFace-Anti-Spoofing.

[9] Krish Naik. “Deep Learning Indepth Tutorials In 5 Hours With Krish Naik” YouTube, 22 May 2022, https://www.youtube.com/watch?v=d2kxUVwWWwU

APPENDIX A: TOOLS AND TECHNOLOGIES

- PYTHON: Python served as the primary programming language for the development of the Real-Time Face Attendance System. Its extensive libraries and frameworks streamlined coding processes and contributed to the overall efficiency of the project.

- JUPYTER NOTEBOOK: Jupyter Notebook played a crucial role in the development and documentation phases of the project. This open-source web application facilitated the creation of interactive notebooks containing live code, visualizations, and explanatory text, enhancing collaboration and reproducibility.

- FACENET: FaceNet, an open-source deep learning framework, was instrumental in implementing facial recognition capabilities within the Real-Time Face Attendance System. Leveraging FaceNet's pre-trained models and APIs, the system achieved accurate and real-time face detection and recognition.

- OPENCV: OpenCV (Open-Source Computer Vision Library) provided essential functionalities for image processing and computer vision tasks. Its comprehensive set of tools and algorithms, including the Haar Cascade classifier, facilitated face detection and feature extraction in the project.

- PANDAS: The Pandas library in Python played a vital role in data manipulation and analysis tasks within the Real-Time Face Attendance System. Its powerful data structures and functions enabled efficient handling of attendance records and integration with external data sources.

- NUMPY: NumPy, a fundamental package for scientific computing in Python, provided support for numerical operations and array manipulation. Its array-based computing capabilities enhanced the performance and efficiency of mathematical operations within the project.

- KERAS: Keras, an open-source software library, provided a user-friendly interface for building and training artificial neural networks. Integrated with TensorFlow, Keras facilitated the implementation of deep learning models for face recognition tasks in the project.

- HAAR CASCADE CLASSIFIER: The Haar Cascade Classifier, integrated into the OpenCV framework, facilitated efficient and reliable face detection by analyzing predefined features and classifiers trained to recognize specific patterns resembling facial features.

- OPENPYXL & EXCEL: Leveraged openpyxl library in Python for seamless Excel integration, facilitating efficient attendance tracking alongside the Haar Cascade Classifier for face detection.