**MINOR PROJECT REPORT**

**JORHAT INSTITUTE OF SCIENCE AND TECHNOLOGY**

**DEPARTMENT OF CS & IT**

****

**Title of the Project**

**“Face Recognition Based Attendance System”**

**Submitted by:**

Name of the Candidate: Ruhon Borah

ASTU Roll No: 210810004037

Registration No:

Name of the Candidate: Syed Marjukur Rohman

ASTU Roll No: 210810004046

Registration No:018308121

**Supervised by:**

Name of the Supervisor: Mr. Siddhartha Adhyapok

**CERTIFICATE**

This project entitled "Face Recognition based Attendance System" submitted by RUHON BORAH (ROLL NO. 210810004037) and SYED MARJUKUR ROHMAN (ROLL NO. 210810004046) in partial fulfilment of requirements for the degree of Bachelor of Science in Information Technology of JIST, under Assam Science and Technology University has been examined.

Internal Examiner External Examiner

Date - Date –

Place -Jorhat Place – Jorhat

**SUPERVISOR's CERTIFICATE**

This is to certify that Ruhon Borah and Syed Marjukur Rohman are the student of B.ScIT of Jorhat Institute of Science And Technology (Name of the College) under Assam Science and Technology University, Assam has worked under my supervision and guidance for his/her Project Work and prepared a Project Report with the title FACE RECOGNITION BASED ATTENDANCE SYSTEM which he is submitting, is his genuine and original work to the best of my knowledge.

Place: Jorhat Signature

Date Mr. Siddhartha Adhyapok

Assistant Professor at JIST

**STUDENT's DECLARATION**

We, Ruhon Borah and Syed Marjukur Rohman hereby declare that the Project Work with the title FACE RECOGNIZE BASED ATTENDANCE SYSTEM submitted by me for the partial fulfilment of the degree of B.Sc(IT) under Assam Science and Technology University, Assam is my original work and has not been submitted earlier to any other University /Institution for the fulfilment of the requirement for any course of study. We also declare that no chapter of this manuscript in whole or in part has been incorporated in this report from any earlier work done by others or by use. However, extracts of any literature which has been used for this report has been duly acknowledged providing details of such literature in the references.

Place: JIST, Jorhat Signature

Date: Name: Ruhon Borah

Roll No:210810004037

Registration No:

Signature

Name: Syed Marjukur Rohman

Roll No:210810004046

Registration No:

**ACKNOWLEDGEMENT**

The ability to help and have patience to exercise diligence and provide support is a quality admonished by very few. Any job in this world, however trivial or tough cannot be accomplished without the assistance of the others. We would hereby take the opportunity to express my indebtedness to people who have helped us to accomplish this task. The present line of accomplishment is not a formality but an honest word of appreciation that has exactly been felt by me during my project.

Our deepest thanks to the project guide Mr. Siddhartha Adhyapok (Assistant Professor at JIST) for his help and his deep interest in the development of the project and constant reminder for updates. I appreciate his involvement in the major project with deepest gratitude.

We also take this opportunity to express my indebtedness to my parents, all our respected professors of JIST and friends for their kind consent guidance, valuable suggestion and affectionate encouragement.

Ruhon Borah (Roll No.:210810004037)

Syed Marjukur Rohman (Roll No.:210810004037)

5th Semester, BSc-(IT)

Jorhat Institute of Science & Technology,

Sotai-Jorhat, Assam, India, 785010.

**Content page**

**ABSTRACT**

In colleges, universities, organizations, schools, and offices, taking attendance is one of the most important tasks that must be done on a daily basis. The majority of the time, it is done manually, such as by calling by name or by roll number. The main goal of this project is to create a Face Recognition-based attendance system that will turn this manual process into an automated one. This project meets the requirements for bringing modernization to the way attendance is handled, as well as the criteria for time management. This device is installed in the classroom, where and student's information, such as name, roll number, class, sec, and photographs, is trained. The images are extracted using Open CV. Before the start of the corresponding class, the student can approach the machine, which will begin taking pictures and comparing them to the qualified dataset. Logitech C270 web camera and NVIDIA Jetson Nano Developer kit were used in this project as the camera and processing board. The image is processed as follows: first, faces are identified using a Haarcascade classifier, then faces are recognized using the LBPH (Local Binary Pattern Histogram) Algorithm, histogram data is checked against an established dataset, and the device automatically labels attendance. An Excel sheet is developed, and it is updated every hour with the information from the respective class instructor.

**PROJECT INTRODUCTION**

* 1. **Project title:**

The project titled as “FACE RECOGNISE BASED ATTENDENANCE SYETEM”.

**1.2project Objective:**

Attendance is prime important for both the teacher and student of an educational organization. So it is very important to keep record of the attendance. The problem arises when we think about the traditional process of taking attendance in class room.

Calling name or roll number of the student for attendance is not only a problem of time consumption but also it needs energy. So an automatic attendance system can solve all above problems.

There are some automatic attendances making system which are currently used by much institution. One of such system is biometric technique and RFID system. Although it is automatic and a step ahead of traditional method it fails to meet the time constraint. The student has to wait in queue for giving attendance, which is time taking.

This project introduces an involuntary attendance marking system, devoid of any kind of interference with the normal teaching procedure. The system can be also implemented during exam sessions or in other teaching activities where attendance is highly essential. This system eliminates classical student identification such as calling name of the student, or checking respective identification cards of the student, which can not only interfere with the ongoing teaching process, but also can be stressful for students during examination sessions. In addition, the students have to register to be recognized. The enrolment can be done on the spot through the userfriendly interface.

**1.2 Background:**

Face recognition is crucial in daily life in order to identify family, friends or someone we are familiar with. We might not perceive that several steps have actually taken in order to identify human faces. Human intelligence allows us to receive information and interpret the information in the recognition process. We receive information through the image projected into our eyes, by specifically retina in the form of light. Light is a form of electromagnetic waves which are radiated from a source onto an object and projected to human vision. Robinson-Riegler, G., & Robinson-Riegler, B. (2008) mentioned that after visual processing done by the human visual system, we actually classify shape, size, contour and the texture of the object in order to analyze the information. The analyzed information will be compared to other representations of objects or face that exist in our memory to recognize. In fact, it is a hard challenge to build an automated system to have the same capability as a human to recognize faces. However, we need large memory to recognize different faces, for example, in the Universities, there are a lot of students with different race and gender, it is impossible to remember every face of the individual without making mistakes. In order to overcome human limitations, computers with almost limitless memory, high processing speed and power are used in face recognition systems.

The human face is a unique representation of individual identity. Thus, face recognition is defined as a biometric method in which identification of an individual is performed by comparing real-time capture image with stored images in the database of that person (Margaret Rouse, 2012).

Nowadays, face recognition system is prevalent due to its simplicity and awesome performance. For instance, airport protection systems and FBI use face recognition for criminal investigations by tracking suspects, missing children and drug activities (Robert Silk, 2017). Apart from that, Facebook which is a popular social networking website implement face recognition to allow the users to tag their friends in the photo for entertainment purposes (Sidney Fussell, 2018). Furthermore, Intel Company allows the users to use face recognition to get access to their online account (Reichert, C., 2017). Apple allows the users to unlock their mobile phone, iPhone X by using face recognition (deAgonia, M., 2017).

The work on face recognition began in 1960. Woody Bledsoe, Helen Chan Wolf and Charles Bisson had introduced a system which required the administrator to locate eyes, ears, nose and mouth from images. The distance and ratios between the located features and the common reference points are then calculated and compared. The studies are further enhanced by Goldstein, Harmon, and Lesk in 1970 by using other features such as hair colour and lip thickness to automate the recognition. In 1988, Kirby and Sirovich first suggested principle component analysis (PCA) to solve face recognition problem. Many studies on face recognition were then conducted continuously until today (Ashley DuVal, 2012).

* 1. **Problem Statement:**

Traditional student attendance marking technique is often facing a lot of trouble. The face recognition student attendance system emphasizes its simplicity by eliminating classical student attendance marking technique such as 5 calling student names or checking respective identification cards. There are not only disturbing the teaching process but also causes distraction for students during exam sessions. Apart from calling names, attendance sheet is passed around the classroom during the lecture sessions. The lecture class especially the class with a large number of students might find it difficult to have the attendance sheet being passed around the class. Thus, face recognition attendance system is proposed in order to replace the manual signing of the presence of students which are burdensome and causes students get distracted in order to sign for their attendance. Furthermore, the face recognition based automated student attendance system able to overcome the problem of fraudulent approach and lecturers does not have to count the number of students several times to ensure the presence of the students.

The paper proposed by Zhao, W et al. (2003) has listed the difficulties of facial identification. One of the difficulties of facial identification is the identification between known and unknown images. In addition, paper proposed by Pooja G.R et al. (2010) found out that the training process for face recognition student attendance system is slow and time-consuming. In addition, the paper proposed by Priyanka Wagh et al. (2015) mentioned that different lighting and head poses are often the problems that could degrade the performance of face recognition based student attendance system.

Hence, there is a need to develop a real time operating student attendance system which means the identification process must be done within defined time constraints to prevent omission. The extracted features from facial images which represent the identity of the students have to be consistent towards a change in background, illumination, pose and expression. High accuracy and fast computation time will be 6 the evaluation points of the performance.

* 1. **Aims and Objectives**:

The objective of this project is to develop face recognition attendance system. Expected achievements in order to fulfill the objectives are:

● To detect the face segment from the video frame.

● To extract the useful features from the face detected.

● To classify the features in order to recognize the face detected.

● To record the attendance of the identified student.

**REQUIREMENT ANALYSIS AND GATHERING**

**2.1 INTRODUCTION**:

Requirement analysis, also called requirement engineering is the process of determining user expectations for a new or modified product. These features called requirement must be quantifiable, relevant and detailed. Requirement analysis is an important aspect of project management. In systems engineering and software engineering, requirement analysis encompasses those tasks that go into determining the conditions to meet for a new or modified product, taking into account of the possibly conflicting requirements of the various stakeholders, analyzing, documenting, validating and managing software or system requirements. System analysis is a detailed study of various operations performed by a system and their relationships within and outside the system. It is a systematic technique that define its purpose and objectives. The goal of the system development is to deliver the system in line with the user's requirement.

In business, system analysis and design refer to the process of examining a business situation with the intent of improving it with a better procedure and methods. It is the process of gathering and interpreting the facts, diagnosing problems and using the information to recommend improvements to the system.

Requirement analysis is critical for success or failure of a system or software project. The requirements should be documented, actionable, measurable, testable, traceable, related to identified business needs or opportunities and defined to a level of detail sufficient for system design.

Requirement analysis can be a long and tiring process during which many delicate psychological skills are involved. Large systems may confront analysis with hundreds or thousands of system requirements. New systems change the environment and relationships between people. So, it is important to identify all the stakeholders, take into account all their needs and ensure they understand the implications of the new systems. Analysts can employ several techniques to elicit the requirements from the customer. These may include the development of scenarios (represented as user stories in agile methods). The identification of use cases, the use of workplace observation or ethnography, holding interviews, or focus groups (more aptly named in this context as requirements workshops, or requirements review sessions) and creating requirements lists. Prototyping may be used to develop an example system that can be demonstrated to stakeholders. Where necessary, the analyst will employ a combination of these methods to establish the exact requirements of the stakeholders, so that a system that meets the business needs is produced. Requirement qualities can be improved through these and other methods are such as visualization and simulation.

**2.2 HARDWARE REQUIREMENTS:**

AMD Ryzen 7 is used as a processor because it is fast than other processors an provide reliable and stable and we can run our pc for longtime.

By using this processor, we can keep on developing our project without any worries.

Ram 8 gb is used as it will provide fast reading and writing capabilities and will in turn support in processing.

720p Webcam is used for capturing image or face.

**2.4 SOFTWARE REQUIREMENTS**

**2.4.1 SOFTWARE TOOLS:**

**Client-Server:** Google Chrome or any web browser

**Development Tools:** Visual Studio Code

**Operating system:** Windows 10 is used as the operating system as it is stable and supports more features and is more user friendly

**Development tools and Programming language:** HTML is used to write the whole code for the interface and style the webpage with CSS , python is used for detection and function the whole process.

**2.4.2 Front-End:**

**HTML**

HTML (HyperText Markup Language) stands as the cornerstone of web development, serving as the backbone for creating and structuring content on the World Wide Web. It plays a pivotal role in defining the layout and presentation of web pages, providing a standardized language that browsers interpret to render the visual elements we see on websites.

At its core, HTML is a markup language, and its syntax revolves around the use of tags – enclosed in angle brackets (< >) – to define various elements within a document. These elements range from headings and paragraphs to images and hyperlinks. The structure of HTML documents typically consists of two main sections: the head and the body. The head contains meta-information about the document, such as the title and links to external resources, while the body holds the actual content of the page.

HTML's power lies in its ability to provide a semantic structure to content. With the introduction of HTML5, a set of semantic elements was introduced, allowing developers to convey the meaning and role of each section more clearly. Elements such as `<header>`, `<div>`, `<th>`, and `<td>` contribute to a better understanding of a page's structure, both for developers and search engines.

HTML embraces the principle of accessibility, allowing developers to create websites that cater to a diverse audience, including individuals with disabilities. Accessibility features can be integrated using attributes such as ARIA (Accessible Rich Internet Applications) attributes and semantic HTML elements. These enhancements contribute to a more inclusive and user-friendly web. HTML is essential for building websites that are not only visually appealing but also accessible and user-friendly.

**CSS**

Cascading Style Sheets, commonly known as CSS, is a cornerstone technology in web development that empowers developers to control the visual presentation and layout of HTML documents. Created to enhance the separation of concerns in web design, CSS serves as the styling language that complements HTML's structural and content-oriented role. This article delves into the fundamental aspects of CSS, its features, and its evolving role in modern web development.

At its core, CSS operates through a set of rules that define how elements in an HTML document should be styled. These rules consist of selectors, properties, and values. Selectors target HTML elements, while properties and values dictate the appearance of those elements. This separation of structure and style allows for modularity and easier maintenance of code.

CSS offers a wide array of selectors, ranging from simple element selectors to more complex class and ID selectors. This flexibility enables developers to apply styles globally, to specific elements, or based on their context within the document. This ability to target elements precisely contributes to the fine-grained control that CSS provides over the visual presentation of a webpage.

In the era of diverse devices and screen sizes, CSS plays a crucial role in achieving responsive design. Media queries allow developers to apply different styles based on the characteristics of the device, ensuring a seamless and visually pleasing experience across various platforms.

CSS introduces powerful layout models like Flexbox and Grid, revolutionizing the way developers structure and align content within a webpage. Flexbox is particularly useful for one-dimensional layouts, offering a flexible and efficient way to distribute space among items in a container. On the other hand, Grid excels in two-dimensional layouts, providing precise control over rows and columns.

CSS enables the creation of dynamic and interactive web experiences through animations and transitions. Keyframes define animation sequences, allowing developers to bring elements to life. Transitions, on the other hand, smoothly change property values over a specified duration, enhancing user interactions.

**2.4.3 Back-End:**

**PYTHON**

Python, a versatile and high-level programming language, has emerged as a cornerstone in the world of software development. Guido van Rossum created Python in the late 1980s, and since then, it has gained immense popularity due to its simplicity, readability, and extensive community support. In this exploration of Python, we'll delve into its key features, use cases, and the reasons behind its widespread adoption.

One of Python's defining characteristics is its emphasis on readability. The language is designed with a clean and straightforward syntax, making it accessible to both beginners and experienced developers. Python's use of indentation for code blocks, rather than relying on explicit braces or keywords, contributes to its clarity and simplicity.

Python's versatility is showcased by its ability to run on various platforms, including Windows, macOS, and Linux. Its cross-platform compatibility allows developers to write code on one system and seamlessly execute it on another, minimizing the need for platform-specific modifications.

Python boasts an extensive standard library that provides a wide range of modules and packages, offering solutions for tasks ranging from working with databases and handling regular expressions to implementing web development frameworks. This comprehensive library reduces the need for external dependencies and accelerates the development process.

Python's thriving community is a driving force behind its success. Developers worldwide contribute to an expansive ecosystem of libraries and frameworks that cater to various domains such as web development (Django, Flask), data science (NumPy, Pandas), machine learning (TensorFlow, PyTorch), and more. This rich ecosystem empowers developers to leverage existing solutions and accelerates the development of diverse applications.

Python has become the language of choice for data scientists and machine learning practitioners. Libraries like NumPy and Pandas facilitate data manipulation and analysis, while machine learning frameworks like TensorFlow and PyTorch provide robust tools for building and training machine learning models. Python's simplicity is particularly advantageous in data science, allowing practitioners to focus on the problem at hand rather than grappling with complex syntax.

Python's scripting capabilities make it well-suited for automation tasks and system administration. From writing simple scripts to automate repetitive tasks to creating complex automation workflows, Python's versatility shines in scenarios where efficiency and readability are paramount.

In conclusion, Python has rightfully earned its place as a leading programming language in the software development landscape. Its readability, versatility, extensive ecosystem, and community support contribute to its widespread adoption across diverse domains. Whether you're a beginner learning to code, a data scientist analyzing complex datasets, or a web developer building scalable applications, Python's strengths make it a reliable and powerful choice for a wide range of applications. As the technological landscape continues to evolve, Python is likely to remain a prominent and influential force in the world of programming.

**Modules Used:**

1. **Python Module OpenCV:**

OpenCV (Open Source Computer Vision) is a powerful open-source computer vision and machine learning library that plays a pivotal role in numerous applications, ranging from image and video processing to robotics and artificial intelligence. This report provides a comprehensive overview of the OpenCV Python module, highlighting its key features, capabilities, and common use cases.

**1.1. Reading and Displaying Images:**

OpenCV facilitates the reading and manipulation of images. Images can be loaded into Python scripts using the `cv2.imread()` function, and the results can be displayed using `cv2.imshow()`. The following snippet demonstrates these basic operations:

img = cv2.imread('image.jpg')

cv2.imshow('Image', img)

cv2.waitKey(0)

**cv2.destroyAllWindows()**

**1.2. Reading and Displaying Videos:**

For video processing, OpenCV provides tools to capture video from sources such as webcams. The following example captures video from the default webcam and displays it:

cap = cv2.VideoCapture(0)

while True:

    ret, frame = cap.read()

    cv2.imshow('Video', frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):

        break

cap.release()

**cv2.destroyAllWindows()**

**1. 3. Image Operations and Processing:**

OpenCV supports a myriad of image processing operations, including resizing, rotating, cropping, and color space conversions. These operations are essential for various computer vision tasks and are easily performed using built-in functions.

**1.4. Object Detection:**

Object detection is a crucial aspect of computer vision, and OpenCV provides tools for both classical and deep learning-based approaches. Techniques such as Haarcascades, YOLO (You Only Look Once), and SSD (Single Shot Multibox Detector) are supported.

**1.5. Feature Detection and Matching:**

OpenCV incorporates algorithms for feature detection and matching, enabling tasks such as image recognition and tracking. Algorithms like SIFT (Scale-Invariant Feature Transform), SURF (Speeded-Up Robust Features), and ORB (Oriented FAST and Rotated BRIEF) are readily available.

The OpenCV Python module stands as a versatile and indispensable tool for a wide array of computer vision applications. Its rich set of functionalities, ease of use, and active community support make it a go-to choice for researchers, developers, and hobbyists in the field of computer vision. This report has provided a glimpse into the capabilities of OpenCV, but the library's depth and breadth invite further exploration, especially through the extensive official documentation and community resources.

1. **Python Module Face\_Recognition:**

Facial recognition technology has become an integral part of various applications, from security systems to social media platforms. The Python module `face\_recognition` simplifies the process of working with facial recognition by providing a high-level and user-friendly interface. This report aims to provide an overview of the `face\_recognition` module, its key features, and practical applications.

**2.1. Face Detection:**

One of the primary features of the `face\_recognition` module is its ability to detect faces in images. It utilizes the popular face detection library, Dlib, under the hood. The following example demonstrates basic face detection:

import face\_recognition

import cv2

# Load an image

image\_path = 'path/to/image.jpg'

image = face\_recognition.load\_image\_file(image\_path)

# Find all face locations in the image

face\_locations = face\_recognition.face\_locations(image)

# Draw rectangles around the faces

for face\_location in face\_locations:

    top, right, bottom, left = face\_location

    cv2.rectangle(image, (left, top), (right, bottom), (0, 255, 0), 2)

# Display the image with faces highlighted

cv2.imshow('Faces', image)

cv2.waitKey(0)

cv2.destroyAllWindows()

**2.2. Facial Recognition:**

The module also supports facial recognition by comparing face encodings. Face encodings are numerical representations of facial features that can be used for identification. The example below illustrates how to compare faces:

import face\_recognition

# Load images with known and unknown faces

known\_image = face\_recognition.load\_image\_file('known\_person.jpg')

unknown\_image = face\_recognition.load\_image\_file('unknown\_person.jpg')

# Encode the faces

known\_face\_encoding = face\_recognition.face\_encodings(known\_image)[0]

unknown\_face\_encoding = face\_recognition.face\_encodings(unknown\_image)[0]

# Compare the faces

results = face\_recognition.compare\_faces([known\_face\_encoding], unknown\_face\_encoding)

if results[0]:

    print("It's the known person!")

else:

    print("Unknown person.")

**2.3. Real-time Face Recognition**

Using the module along with a video stream, real-time face recognition can be implemented. This is particularly useful for security applications or access control systems.

The `face\_recognition` module simplifies facial recognition tasks in Python, making it accessible for developers to implement solutions involving face detection, recognition, and landmark identification. Its integration with Dlib for face detection and encoding contributes to its accuracy and reliability. While this report covers fundamental aspects of the module, users are encouraged to explore its documentation and community resources for in-depth understanding and advanced usage.

1. **Python Module NumPy:**

NumPy, short for Numerical Python, is a fundamental package for scientific computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays. This report explores the key features, capabilities, and applications of the NumPy module in Python.

**3.1. N-Dimensional Arrays**:

At the core of NumPy is the `numpy.ndarray` class, which enables the creation of arrays with multiple dimensions. These arrays are efficient for storing and manipulating large sets of numerical data. Creating a simple array looks like this:

import numpy as np

# Create a 1D array

arr\_1d = np.array([1, 2, 3])

# Create a 2D array

arr\_2d = np.array([[1, 2, 3], [4, 5, 6]])

**3.2. Array Operations:**

NumPy provides a wide range of mathematical functions and operations that can be performed on arrays. These include element-wise operations, linear algebra operations, statistical operations, and more. For example:

import numpy as np

# Element-wise addition

result = np.array([1, 2, 3]) + np.array([4, 5, 6])

# Matrix multiplication

matrix\_product = np.dot(matrix\_a, matrix\_b)

# Statistical operations

mean\_value = np.mean(data)

**3.3. Broadcasting:**

NumPy allows operations between arrays of different shapes and sizes through a feature called broadcasting. This simplifies tasks where arrays with different dimensions need to be combined or operated upon.

**3.4. Universal Functions (ufuncs):**

NumPy's universal functions perform element-wise operations on arrays, and they are vectorized, meaning they operate on entire arrays without the need for explicit looping. This enhances performance and readability of code.

**3.5. Indexing and Slicing:**

NumPy supports powerful indexing and slicing operations, enabling the extraction and manipulation of specific elements or subarrays from an array.

**3.6. Integration with Other Libraries:**

NumPy serves as the foundation for many other scientific computing libraries in Python, including SciPy, Matplotlib, and scikit-learn. This integration creates a powerful ecosystem for data analysis, machine learning, and scientific research.

NumPy stands as an indispensable tool in the Python scientific computing landscape. Its versatile array operations, powerful mathematical functions, and seamless integration with other libraries make it a go-to choice for researchers, data scientists, and engineers. As we've only scratched the surface in this report, the rich functionality of NumPy invites further exploration through the extensive official documentation and community resources.

1. **Python Module: Pandas:**

Pandas, a versatile and powerful data manipulation and analysis library in Python, provides high-level data structures and functions for efficiently manipulating large datasets. This report aims to provide an overview of the key features, functionalities, and applications of the pandas module.

* 1. **Data Structures:**

Pandas introduces two primary data structures - Series and DataFrame.

* + 1. **Series:**

A one-dimensional labeled array capable of holding any data type.

s = pd.Series([1, 3, 5, np.nan, 6, 8])

* + 1. **DataFrame**:

A two-dimensional labeled data structure with columns that can be of different types.

df = pd.DataFrame({'A': [1, 2, 3], 'B': [4, 5, 6]})

* 1. **Data Cleaning and Manipulation:**

Pandas provides a plethora of functions for cleaning and manipulating data. This includes handling missing values, reshaping data, merging and joining datasets, and transforming variables.

# Removing missing values

df.dropna()

# Filling missing values

df.fillna(value)

# Grouping and aggregating data

df.groupby('column\_name').mean()

* 1. **Indexing and Selection:**

Pandas allows for efficient indexing and selection of data using labels, integer-based location, or boolean indexing.

# Selecting columns

df['column\_name']

# Slicing rows

df.iloc[0:3]

# Boolean indexing

df[df['column\_name'] > value]

* 1. **Input and Output:**

Pandas supports various file formats for reading and writing data, including CSV, Excel, SQL databases, and more.

# Reading from CSV

df = pd.read\_csv('data.csv')

# Writing to Excel

df.to\_excel('output.xlsx', index=False)

* 1. **Integration with Other Libraries:**

Pandas integrates seamlessly with other data science and machine learning libraries in the Python ecosystem, such as NumPy, scikit-learn, and TensorFlow.

Pandas plays a crucial role in the Python data science ecosystem, providing an intuitive and efficient framework for data manipulation and analysis. Its user-friendly interface, extensive functionality, and compatibility with other libraries make it a cornerstone for data scientists, analysts, and researchers. As this report has only scratched the surface of pandas capabilities, further exploration through the official documentation and community resources is encouraged.

1. **Web Development with Flask module in Python:**

Flask, a lightweight and flexible web framework for Python, has emerged as a popular choice for building web applications due to its simplicity, ease of use, and extensibility. This report provides an overview of Flask, highlighting its key features, components, and applications in web development.

* 1. **Setting up:**

from flask import Flask

app = Flask(\_\_name\_\_)

@app.route('/')

def hello\_world():

    return 'Hello, World!'

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

* 1. **Routing:**

Flask uses decorators to define routes. The `@app.route()` decorator associates a function with a URL, allowing developers to define the behaviour of the application for different routes.

@app.route('/about')

def about():

    return 'About Us'

* 1. **Request Handling:**

Flask simplifies request handling with the `request` object, allowing easy access to form data, request parameters, and other request-related information.

from flask import request

@app.route('/submit', methods=['POST'])

def submit\_form():

    user\_input = request.form['user\_input']

    # Process the input

    return 'Form submitted successfully'

Flask's simplicity and flexibility make it an ideal choice for developers seeking a lightweight yet powerful web framework. Its intuitive design, extensive documentation, and vibrant community support contribute to its popularity in both small projects and large-scale web applications. Developers are encouraged to explore Flask's capabilities further through official documentation and community resources.

**DESIGNING**

**3.1 Block Diagram:**

Camera

Pre-Processing

Face-Detection

Face-Recognition

Excel Sheet

CSV

Update

Face/Image

Folder

**3.2Algorithm Used:**

**Step 1:** **Import the necessary libraries.**

import face\_recognition

import cv2

import numpy as np

import csv

import os

from playsound import playsound

from datetime import datetime

import pandas as pd

from subprocess import call

**Step 2:  Define a folder path where your training image dataset will be stored.**

path = ‘C:\Users\LENOVO\Desktop\Minor Project\images’

and

path = ‘C:\Users\Ruhon Borah\Desktop\Minor Project\images’

Note: We are storing images of people by “person\_name.jpg/jpeg” this format only.

**Step 3: Create a list to store person\_name and image array.**

images =[]

known\_faces\_names=[]

myList = os.listdir(path)

print(myList)

for cl in myList:

    curImg = cv2.imread(f'{path}/{cl}')

    images.append(curImg)

    known\_faces\_names.append(os.path.splitext(cl)[0]) #removes file extension from the image and takes the name of the image

    print(known\_faces\_names)

**Step 4: Create a function to encode all the train images and store them in an empty list known\_faces\_encodings=[ ].**

known\_faces\_encodings=[]

def findEncodings(images):

    for img in images:

        img = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)

        encode = face\_recognition.face\_encodings(img)[0]

        known\_faces\_encodings.append(encode)

    return known\_faces\_encodings

known\_faces\_encodings = findEncodings(images)

student=known\_faces\_names.copy()#create another list that stores the known students names.

**Step 5:** Now create a csv file as ‘today’s\_date.csv’

nowdate = datetime.now()

current\_date=nowdate.strftime("%d-%m-%Y")

#creating csv file

f = open(current\_date+".csv","w+",newline='')

lnwriter=csv.writer(f)

lnwriter.writerow(["Name","Roll no","Time"])

## Step 6: Read Webcam for Real-Time Recognition and mark the attendance in the csv file and remove the names of students from the list that are being marked present.

video\_capture=cv2.VideoCapture(0)

face\_loc=[]

face\_encodings=[]

face\_name=[]

s=True

while True:

    ret, frame=video\_capture.read()

    small\_frame=cv2.resize(frame,(0,0),fx=0.25,fy=0.25)

    small\_frame=cv2.cvtColor(small\_frame, cv2.COLOR\_BGR2RGB)

    if s:

        face\_locations= face\_recognition.face\_locations(small\_frame)

        face\_encodings=face\_recognition.face\_encodings(small\_frame, face\_locations)

        face\_names=[]

        for face\_encoding in face\_encodings:

            matches=face\_recognition.compare\_faces(known\_faces\_encodings, face\_encoding)

            name=""

            face\_distance=face\_recognition.face\_distance(known\_faces\_encodings,face\_encoding)

            best\_match\_index= np.argmin(face\_distance)

            if matches[best\_match\_index]:

                name= known\_faces\_names[best\_match\_index]

            face\_names.append(name)

            if name in known\_faces\_names:

                if name in student:

                    roll\_no=known\_faces\_names.index(name)+1

                    student.remove(name)

                    print(student)

                    nowtime = datetime.now()

                    current\_time = nowtime.strftime("%H:%M:%S")

                    playsound("attendance.wav")

                    lnwriter.writerow([name.capitalize(),roll\_no,current\_time])

    cv2.imshow("attendence sys",frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):

        break

**3.3 Flowchart:**

**Image stored in system**

**Recognition process start**

**Camera captures the user image**

**Compare with system image**

**Ignore the unknown person**

**No Data record**

**Attendance uploaded in Web server automatically**

**A csv file generated with student details**

**Present to the user**

**Check Continuously**

**Don’t match**

**Match**

**3.4 Webserver Implementation**

**Flask Web Server workings:**

**Step 1: Import Modules :**

*import* pandas as pd

*from* flask *import* Flask,request,render\_template

*import* webbrowser

**Step 2: Initiate Flask**

app = Flask(\_\_name\_\_)

**Step 3: Create a function that extracts the attendance values from the CSV file:**

def extract\_attendance():

    df=pd.read\_csv(f"{current\_date}.csv")

    names = df['Name']

    rolls = df['Roll no']

    times = df['Time']

    l = len(df)

*return* names,rolls,times,l

**Step 4: Post the attendance on the web page:**

@app.route('/')

def home():

    names,rolls,times,l = extract\_attendance()

*return* render\_template('home2.html',*names*=names,*rolls*=rolls,*times*=times,*l*=l)

**Step 5: Run the server:**

webbrowser.open('http://127.0.0.1:5000')

*if* \_\_name\_\_=="\_\_main\_\_":

    app.run()

Run the server with hostname.

**LIMITATION OF THE EXISTING SYSTEM**

One of the main challenges of face detection and recognition is the high variability of human faces in terms of shape, size, pose, expression, illumination, occlusion, and makeup. These factors can make it difficult for the algorithms to generalize and cope with different scenarios and conditions. For example, face detection can fail when the face is partially hidden by a mask, a hat, or a hand, or when the face is tilted, rotated, or distorted. Face recognition can also suffer from low performance when the face is not aligned, cropped, or normalized, or when the face changes over time due to aging, facial hair, or cosmetics.

Face detection and recognition is the diversity and quality of the datasets that are used to train and evaluate the algorithms. Different datasets may have different characteristics, such as resolution, format, annotation, distribution, and bias. For instance, some datasets may have more images of certain ethnicities, genders, ages, or backgrounds than others, which can affect the fairness and representativeness of the algorithms. Moreover, some datasets may have noisy, incomplete, or inaccurate labels, which can degrade the reliability and validity of the algorithms.

**CONCLUSION**

Face recognition systems are part of facial image processing applications and their significance as a research area are increasing recently. Implementations of system are crime prevention, video surveillance, person verification, and similar security activities. The face recognition system implementation can be part of universities. Face Recognition Based Attendance System has been envisioned for the purpose of reducing the errors that occur in the traditional (manual) attendance taking system. The aim is to automate and make a system that is useful to the organization such as an institute. The efficient and accurate method of attendance in the office environment that can replace the old manual methods. This method is secure enough, reliable and available for use. Proposed algorithm is capable of detect multiple faces, and performance of system has acceptable good results.

**BIBLOGRAPHY**

1. <https://www.python.org/>
2. <https://pypi.org/>
3. <https://getbootstrap.com/>
4. <https://www.geeksforgeeks.org/>