Attendance-System through Real-time Database

Project report in partial fulfilment of the requirement for the award of the degree of

Bachelor of Technology

In

CSE (IOT, BCT, CYS)

Submitted By

Pinaki Pritam Singha	Enrollment No. 12021002029065
Trideep Saha	Enrollment No. 12021002029040
Ratul Das	Enrollment No. 12021002029046
Suchak Sarkar	Enrollment No. 12021002029071
Sayan Maity	Enrollment No. 12021002029198
Kuntaleeka Kundu	Enrollment No. 12021002029016
Tathagata Ghosh	Enrollment No. 12021002029039
Yagnasree Chakraborty	Enrollment No. 12021002029062
Aniket Marik	Enrollment No. 12021002001073
Nilarghya Nandan	Enrollment No. 12021002029036

Under the guidance of

Prof. Sandip Mondal

Department of CSE (IOT, BCT, CYS)

UNIVERSITY OF ENGINEERING & MANAGEMENT, KOLKATA

University Area, Plot No. III – B/5, New Town, Action Area – III, Kolkata – 700160.

CERTIFICATE

This is to certify that the project titled **Attendance-System through Real-time Database** submitted by **Pinaki Pritam Singha (University Roll No. 12021002029065), Trideep Saha (University Roll No. 12021002029040)**, Ratul Das **(University Roll No. 12021002029046),** Suchak Sarkar **(University Roll No. 12021002029071)**, Sayan Maity **(University Roll No. 12021002029198)** Kumtaleeka Kundu **(University Roll No. 12021002029016),** Tathagata Ghosh **(University Roll No. 12021002029039),** Yagnasree Chakraborty **(University Roll No. 12021002029062),** Aniket Marik **(University Roll No. 12021002001073) and** Nilarghya Nandan **(University Roll No. 12021002029036)** students of UNIVERSITY OF ENGINEERING & MANAGEMENT, KOLKATA, in partial fulfilment of requirement for the degree of Bachelor of Computer Science (IOT, BCT, CYS), is a Bonafede work carried out by them under the supervision and guidance of Prof. Sandip Mondal during 4th Semester of academic session of 2022 - 2023. The content of this report has not been submitted to any other university or institute. I am glad to inform that the work is entirely original and its performance is found to be quite satisfactory.

_____

Signature of Guide

_____

Signature of Head of the Department

ACKNOWLEDGEMENT

We would like to take this opportunity to thank everyone whose cooperation and encouragement throughout the ongoing course of this project remains invaluable to us.

We are sincerely grateful to our guide Prof. Sandip Mondal of the Department of CSE (IOT, BCT, CYS), UEM, Kolkata, for his wisdom, guidance and inspiration that helped us to go through with this project and take it to where it stands now.

Last but not the least, we would like to extend our warm regards to our families and peers who have kept supporting us and always had faith in our work.

Pinaki Pritam Singha

Trideep Saha

Ratul Das

Suchak Sarkar

Sayan Maity

Kuntaleeka Kundu

Tathagata Ghosh

Yagnasree Chakraborty

Aniket Marik

Nilarghya Nandan

TABLE OF CONTENTS

ABSTRACT

A NoSQL document-based database system is a type of database management system that stores and retrieves data in a document-oriented way. Instead of storing data in tables and rows like a traditional relational database, a NoSQL database stores data as documents, typically in JSON or BSON format. This allows for more flexibility in storing and querying data, as well as the ability to handle unstructured and semi-structured data. [1]

One of the main advantages of a NoSQL document-based database system is its ability to scale horizontally, meaning it can handle increasing amounts of data by adding more nodes to the cluster. Additionally, since documents can be self-contained and hierarchical, they can be retrieved in a single query, reducing the need for complex join operations. NoSQL databases also allow for easier schema evolution and can handle data with varying structures and types. [2]

Our main objective is to use this technology to build a working attendance system that would first take visual image face data followed by maintaining a database system.

INTRODUCTION

NoSQL is a type of database management system that is used to store and manage data in a non-tabular form, unlike traditional relational database management systems (RDBMS). In NoSQL databases, the data is stored in a non-tabular format, such as key-value pairs, document-oriented, graph-based or other formats. One of the most popular types of NoSQL databases is document-based databases, which are used to store and manage data in the form of documents.

MongoDB is a widely used document-based NoSQL database system. It is an open-source database that is designed to store and manage large volumes of unstructured and semi-structured data. MongoDB is written in C++ and provides a flexible document model that allows users to store data in JSON-like documents with dynamic schemas. This means that data can be stored without a predefined structure, which makes MongoDB highly scalable and adaptable to different types of data. [1]

One of the most significant advantages of MongoDB is its flexibility. MongoDB allows users to store and manage data in a variety of formats, including JSON, BSON, XML, and others. This flexibility makes it easier for developers to store and manage data in a way that is best suited to their needs.

Another advantage of MongoDB is its scalability. MongoDB is designed to scale horizontally, meaning that new servers can be added to the cluster to handle additional data and requests. This makes it easier for organizations to handle large volumes of data and ensures that the database can handle the load as the organization grows. [2]

HaarCascade is a machine learning-based object detection algorithm. It is used to detect objects in images and videos. HaarCascade is an open-source project that was developed by Viola and Jones in 2001. It is based on a machine learning technique called the Haar wavelet transform. [3]

One of the advantages of HaarCascade is its speed. HaarCascade is designed to detect objects quickly and accurately, making it ideal for real-time applications such as face recognition, object detection, and others.

Another advantage of HaarCascade is its accuracy. HaarCascade is a highly accurate algorithm that can detect objects with high precision and recall. This makes it ideal for applications that require high levels of accuracy, such as medical imaging and biometric identification. [4]

This project is an implementation of an attendance system using the above technologies explained. It uses OpenCV for face detection, MongoDB for database management, and the pyttsx3 library for text-to-speech output. The system captures images from a camera, detects faces, and prompts the user to provide their name. It then adds the user's name to the attendance database if it is not already present.

LITERATURE SURVEY

NoSQL databases have become increasingly popular in recent years due to their ability to store and manage large volumes of unstructured data. A NoSQL database is designed to handle unstructured data, such as text, images, videos, and other multimedia files, which makes them an ideal choice for many modern applications. One of the most popular NoSQL databases is MongoDB, a document-based database that allows users to store data in flexible, JSON-like documents. [5]

MongoDB is designed to be highly scalable and flexible, making it a popular choice for web applications, mobile applications, and IoT devices. It is built on top of a distributed architecture, which means that data can be easily spread across multiple servers and scaled horizontally as needed. This makes MongoDB an ideal choice for applications that require high availability and low latency. [6]

One of the key advantages of MongoDB is its support for complex queries and indexing. Unlike traditional relational databases, MongoDB does not use a schema to define the structure of the data. Instead, it uses a flexible document model that allows developers to store and retrieve data in any format. This means that users can perform complex queries and search for data using a variety of different criteria. Another advantage of MongoDB is its support for geospatial data. This allows users to store and search for data based on its location, which makes it ideal for applications that require location-based services. MongoDB also supports a wide range of programming languages, including Java, Python, C++, and Ruby, which makes it easy to integrate with a variety of different systems.

In addition to its support for NoSQL databases, MongoDB also has several other features that make it an attractive choice for many developers. For example, it supports atomic transactions, which ensure that data is always consistent and reliable. It also has built-in support for data encryption, which helps to ensure that data is always secure and protected. [2]

Otherwise face detection is a key application of computer vision that has many practical applications. It involves detecting human faces in images or video feeds and is used in a wide range of applications, from security systems to social media platforms. Haar cascades are one of the most commonly used techniques for face detection. They work by applying a set of pre-trained classifiers to an image or video feed and using these classifiers to identify regions of the image that are likely to contain a face. [3]

Moving forward, Pyttsx3 is a Python library that provides a simple way to convert text into speech. It uses the native text-to-speech engine provided by the operating system and allows users to adjust the speed and pitch of the speech output. This makes it an ideal choice for applications that require text-to-speech functionality, such as assistive technology applications. [7]

In summary, NoSQL document-based databases, such as MongoDB, provide a flexible and scalable way to store and manage large volumes of unstructured data. They offer several advantages over traditional relational databases, including support for complex queries, geospatial data, and a wide range of programming languages. Face detection using Haar cascades is a widely used computer vision technique that has many practical applications. Finally, Pyttsx3 provides a simple way to convert text into speech, making it an ideal choice for text-to-speech applications.

PROBLEM STATEMENT

Manual attendance tracking has been a prevalent practice for decades, and it remains a critical aspect of various organizational settings. However, it is prone to several limitations, such as errors in recording, time-consuming processes, and possible tampering. As organizations continue to grow and the demand for efficiency and accuracy increases, there is a need for a more streamlined and secure attendance tracking system. Automating the attendance system can address these limitations and help organizations save time, effort, and resources while increasing accuracy and security.

The proposed system leverages face detection to provide a more reliable and better attendance tracking process. The system can detect and recognize individuals' faces and capture their attendance with a considerable level of accuracy. This dual-technology approach enhances the system's security, as it reduces the likelihood of fraud or impersonation.

The proposed system is highly versatile and can be used in various settings, including classrooms, offices, and events. For example, in a classroom setting, the system can help teachers accurately track attendance and reduce the time spent on manual recording. In an office setting, the system can simplify the attendance tracking process, thereby allowing employees to focus on their work. In an event setting, the system can quickly and efficiently record attendance, providing organizers with real-time insights into attendance numbers.

Overall, the proposed system provides a reliable, efficient, and secure attendance tracking solution that can address the limitations of traditional manual tracking methods.

PROPOSED SOLUTION

1. Use a camera for face detection and recognition to automate attendance tracking
2. Integrate prompt window input for more secure recording process
3. Use a NoSQL document-based database system such as MongoDB to store attendance records securely and efficiently
4. Ensure data privacy and security by implementing appropriate access controls and encryption measures.

EXPRERIMENTAL SETUP AND RESULT ANALYSIS

Libraries/Plug-ins Required:

PyMongo

Python library used for working with MongoDB, a popular NoSQL document-based database system. It allows Python developers to interact with MongoDB databases, collections, and documents in an easy and efficient manner. PyMongo supports all major features of MongoDB, including indexing, sharding, and replication.

speech_recognition

Python library that allows developers to easily integrate speech recognition capabilities into their applications. It uses the Google Speech Recognition API to recognize spoken words and convert them to text.

cv2

cv2 is a popular computer vision library in Python used for various image and video processing applications.

Pyttsx3

Pyttsx3 is a Python package that provides a simple and easy way to convert text to speech. It uses the Microsoft Text-to-Speech engine to convert written text into audible speech, and supports several languages and voices.

PyAudio

Python library that provides a simple and easy-to-use interface for working with audio data in real-time.

EasyGUI

Python library for creating simple GUI (Graphical User Interface) applications.

Camera (USB)

A working desktop/laptop compatible or integrated camera. The better the resolution, the better the accuracy of the model.

Python

Any device capable of execution and using python libraries is sufficient.

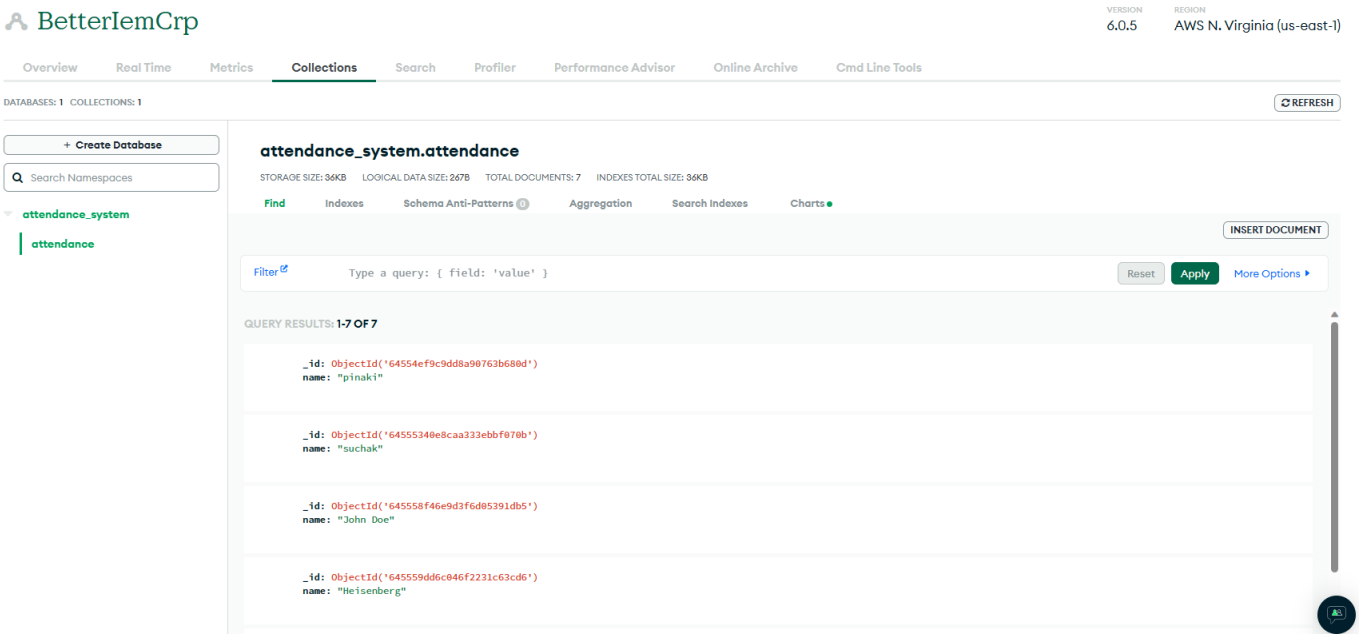
Code (Python):

```
import cv2
from pymongo import MongoClient
import pymongo as pymongo
import speech_recognition as sr
import pyttsx3
import pyaudio as pyaudio
import easygui

# define the connection string
MONGO_CONNECTION_STRING = "mongodb+srv"
# create a MongoClient object and pass the connection string
client = MongoClient(MONGO_CONNECTION_STRING)

# create a database named "attendance_system"
db = client["attendance_system"]

# create a collection named "attendance"
attendance_collection = db["attendance"]

# function to insert a document in the "attendance" collection
def add_attendance(name):
    engine= pyttsx3.init()
    attendance = {"name": name}
    if name == "" or name is None:
        engine.say("Program Terminated.")
        engine.runAndWait()
        exit(0)
    existing_attendance = attendance_collection.find_one({"name": name})
    if existing_attendance:
        with sr.Microphone() as source:
            engine.say("Attendence already added for")
            engine.say(name)
            engine.runAndWait()
            easygui.msgbox("Attendance already added for " + name,
title="Attendance")
    else:
        attendance_collection.insert_one(attendance)
        with sr.Microphone() as source:
            engine.say("Attendence Added for")
            engine.say(name)
            engine.runAndWait()
            easygui.msgbox("Attendance added for " + name, title="Attendance")

# function to create a connection with the MongoDB cluster
def create_connection():
    try:
        client.admin.command('ping')
        print("Connected to MongoDB Atlas")
```

```
    except pymongo.errors.ConnectionFailure:
        print("Failed to connect to MongoDB Atlas")

# function to capture image and detect face
def detect_face():
    # capture frames from a camera
    cap = cv2.VideoCapture(0)
    face_count = 0

    # load the required trained XML classifiers
    face_cascade =
cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
    eye_cascade = cv2.CascadeClassifier('haarcascade_eye.xml')

    #Intializing the Speech Recognizer and Text to speech Engine
    r= sr.Recognizer()
    engine= pyttsx3.init()

    # initialize a variable to keep track of the number of faces detected
    num_faces_detected = 0

    # loop runs if capturing has been initialized.
    while True:
        # check if the required number of faces have been detected
        if num_faces_detected == 1:
            break
        ret, img = cap.read()

        # convert to gray scale of each frames
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

        # Detects faces of different sizes in the input image
        faces = face_cascade.detectMultiScale(gray, 1.3, 5)

        # loop through each face detected
        for (x, y, w, h) in faces:
            # To draw a rectangle in a face
            cv2.rectangle(img, (x, y), (x+w, y+h), (255, 255, 0), 2)
            roi_gray = gray[y:y+h, x:x+w]
            roi_color = img[y:y+h, x:x+w]
            face_count += 1

            # Detects eyes of different sizes in the input image
            eyes = eye_cascade.detectMultiScale(roi_gray, scaleFactor=1.1,
minNeighbors=5)

            #To draw a rectangle in eyes
            for (ex, ey, ew, eh) in eyes:
                cv2.rectangle(roi_color, (ex, ey), (ex+ew, ey+eh), (0, 127,
255), 2)
```

```
            if face_count > 100:
                with sr.Microphone() as source:
                    engine.say("Please enter your name")
                    engine.runAndWait()

                # get the name of the person from the user via voice input
                name = easygui.enterbox("Please enter your name:",
title="Enter Name")

                # add the name to the attendance database
                add_attendance(name)
                num_faces_detected += 1

        # Display an image in a window
        cv2.imshow('img', img)

        # Wait for Esc key to stop
        k = cv2.waitKey(30) & 0xff
        if k == 27:
            break

    # Close the window
    cap.release()

    # De-allocate any associated memory usage
    cv2.destroyAllWindows()

# create a connection with the MongoDB Atlas cluster
create_connection()

# detect face and take attendance
detect_face()
```

Database Configuration:



CONCLUSION & FUTURE SCOPE

In conclusion, the above Python code is a basic attendance system that uses face detection to take attendance and store it in a MongoDB database. The system can detect and recognize faces, capture the name of the person and add their name to the attendance database. The system also uses text-to-speech to provide feedback to the user about the success of their attendance.

In terms of future scope, this system can be further enhanced to include more features such as automatic email notifications to teachers or parents about the attendance of their students, integration with other biometric technologies like fingerprint or iris recognition, and the ability to generate reports on attendance for specific time periods or individual students. Additionally, the system can be made more secure by implementing authentication mechanisms for access to the attendance database.

BIBLIOGRAPHY

[1] M. F. a. P. Sadalage, NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence.

[2] J. W. a. E. R. Luc Perkins, Seven Databases in Seven Weeks: A Guide to Modern Databases and the NoSQL Movement.

[3] V. a. Jones, “Rapid Object Detection using a Boosted Cascade of Simple Features,” [Online]. Available: https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-cvpr-01.pdf. [Accessed 29 April 2023].

[4] “Real-Time Object Detection with Haar Cascades,” OpenCV, [Online]. Available: https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html. [Accessed 1 May 2023].

[5] “NoSQL,” Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/NoSQL. [Accessed 1 May 2023].

[6] “MongoDB Documentation,” MongoDB, [Online]. Available: https://docs.mongodb.com/manual/introduction/. [Accessed 2 May 2023].

[7] “Pyttsx3 Documentation,” [Online]. Available: https://pyttsx3.readthedocs.io/en/latest/. [Accessed 2 May 2023].