**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   
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  
Sayan Maity Enrollment No. 12021002029198

**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) Kuntaleeka 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 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...............................................................................................5**

**CHAPTER – 1: INTRODUCTION..........................................................................6**

**CHAPTER – 2: LITERATURE SURVEY.............................................................7**

**CHAPTER – 3: PROBLEM STATEMENT ..........................................................8**

**CHAPTER – 4: PROPOSED SOLUTION .............................................................9**

**CHAPTER – 5: EXPERIMENTAL SETUP AND RESULT ANALYSIS...........10**

**CHAPTER – 6: CONCLUSION & FUTURE SCOPE..........................................15**

**BIBLIOGRAPHY .....................................................................................................16**

# ABSTRACT

The project centres around the development of a conversational chatbot designed to process multiple PDF documents efficiently. Leveraging technologies such as Streamlit, PyPDF2, and advanced natural language processing libraries, the system undertakes the extraction of textual information from uploaded PDFs. These documents are subsequently segmented into manageable chunks, and sophisticated embeddings are employed to enhance contextual understanding. The chatbot engages users in dynamic conversations, providing insightful responses based on learned patterns and contextual cues.

In the realm of conversational AI, the integration of Streamlit, PyPDF2, and language models signifies a novel approach to address the challenges associated with PDF text processing. The system's ability to seamlessly extract information from diverse PDFs aligns with the increasing demand for intelligent chatbots capable of handling various user queries. By incorporating state-of-the-art technologies, the project exemplifies a cutting-edge solution that pushes the boundaries of traditional information retrieval systems.

Throughout the development process, the project confronts and overcomes challenges related to PDF processing, user interaction, and conversational context preservation. This report provides a comprehensive exploration of the methodology, detailing the intricacies of the implemented solution. The code snippets and algorithms showcased offer a transparent view of the system's architecture and design, enhancing the reader's understanding of the proposed solution's intricacies and efficacy.

# INTRODUCTION

In the rapidly evolving landscape of artificial intelligence (AI), the emergence of a PDF Chatbot addresses a critical need in information retrieval systems. PDFs, ubiquitous in modern document management, present a unique challenge due to their diverse structures and varying content types. The PDF Chatbot stands as a tailored solution, equipped with advanced natural language processing and document processing capabilities to decipher the intricacies of PDFs, making them more accessible and user-friendly.

One of the compelling advantages of the PDF Chatbot lies in its ability to enhance user interaction. Traditional search engines may struggle to retrieve specific information within PDFs, but the chatbot, armed with conversational abilities, offers a more intuitive and interactive experience. Users can engage in natural language queries, and the chatbot, leveraging contextual understanding, provides relevant and nuanced responses. This aligns with the current trend in AI development, emphasizing user-centric solutions that prioritize ease of use and accessibility.

Efficient retrieval of contextual information is another key aspect where the PDF Chatbot excels. Beyond merely extracting data, the chatbot comprehends the context in which the information is presented, contributing to more meaningful responses. In a landscape prioritizing contextual awareness, the chatbot aligns with the broader goals of AI development by providing a more sophisticated understanding of user queries.

Furthermore, the PDF Chatbot streamlines information processing workflows, particularly beneficial for professionals dealing with extensive documentation. It acts as a valuable assistant, swiftly retrieving pertinent information from PDFs and contributing to increased productivity. Its modular nature ensures seamless integration into existing AI ecosystems, adapting to diverse workflows and complementing other AI solutions in use.

As the AI landscape continues to evolve, the PDF Chatbot stands out as a future-proof tool. Its adaptability, capacity to learn from user interactions, and ability to integrate new language models position it as a versatile asset in the ongoing development of AI-driven information systems. In essence, the PDF Chatbot not only addresses current challenges in PDF document management but also aligns with the overarching trends in AI development, ensuring its relevance in the dynamic AI landscape.

# LITERATURE SURVEY

In the evolving landscape of information and document management, the need for a PDF Chatbot becomes increasingly pronounced, presenting a valuable solution to address the challenges posed by the vast and varied realm of PDF documents. As we delve into the intricacies of this specialized chatbot, its potential benefits to current AI-based developments become evident, reflecting a transformative impact on the efficiency and effectiveness of information retrieval systems.

Addressing the PDF Complexity:

PDFs, with their ubiquitous presence in modern document sharing and storage, often present a unique challenge for traditional information retrieval systems. The inherent complexity lies in the diverse structures, varying text placements, and the potential mix of textual and graphical elements within a single document. A PDF Chatbot is tailored to navigate this complexity, providing a streamlined means of extracting and interpreting information from PDFs. Through advanced natural language processing and document processing techniques, the chatbot can decipher the intricacies of PDFs, making them more accessible and user-friendly.

Enhanced User Interaction:

Traditional search engines or document management systems may fall short when users seek specific information within a PDF. A PDF Chatbot, armed with conversational abilities, bridges this gap by offering a more intuitive and interactive experience. Users can pose natural language queries, and the chatbot, leveraging its contextual understanding, can retrieve relevant information from PDFs. This level of interaction aligns with the growing trend in AI development towards user-centric solutions that prioritize ease of use and accessibility.

Efficient Retrieval of Contextual Information:

Current AI-based developments emphasize the importance of contextual understanding and personalized interactions. A PDF Chatbot contributes to this narrative by not only extracting information but also comprehending the context in which it is presented. This contextual awareness enables the chatbot to provide more relevant and nuanced responses, catering to the specific needs and queries of users. In a landscape where user experience is paramount, the ability to retrieve contextually rich information from PDFs aligns with the broader goals of AI development.

Streamlining Information Processing Workflows:

For professionals across various industries, dealing with extensive documentation is a common challenge. A PDF Chatbot acts as a valuable assistant in streamlining information processing workflows. Whether in legal, medical, or business contexts, the chatbot can swiftly retrieve pertinent information from PDF documents, saving time and effort. This efficiency aligns with the overarching trend in AI towards automation and optimization of routine tasks, contributing to increased productivity.

Adaptability and Integration:

The modular nature of a PDF Chatbot allows for seamless integration into existing AI ecosystems and workflows. As organizations continue to adopt diverse AI solutions, the ability of the PDF Chatbot to complement and enhance these systems is a noteworthy advantage. Whether integrated into customer support platforms, knowledge management systems, or collaborative tools, the chatbot's adaptability ensures its relevance in various AI-based developments.

Future-Proofing Information Retrieval:

The dynamic nature of AI development necessitates solutions that can adapt to emerging challenges and technologies. A PDF Chatbot, equipped with advanced document processing capabilities, positions itself as a future-proof tool in the realm of information retrieval. Its ability to learn from user interactions, adapt to evolving document formats, and integrate new language models ensures its relevance in an ever-changing AI landscape.

In conclusion, the need for a PDF Chatbot arises from the intricate challenges posed by PDF documents, and its potential benefits to current AI-based developments are multifaceted. By addressing the complexities of PDFs, enhancing user interaction, providing efficient contextual information retrieval, streamlining workflows, and ensuring adaptability, the PDF Chatbot stands as a valuable asset in the evolving landscape of AI-driven information systems.

# PROBLEM STATEMENT

Problem Overview:

PDFs, as a widely adopted format for document storage and sharing, present a multifaceted challenge for users seeking efficient information retrieval. Traditional methods of document processing and search engines often struggle to navigate the intricate structures of PDFs, hindering the seamless extraction of relevant information. The lack of a dedicated system capable of effectively handling the diverse content types, structures, and graphical elements within PDFs results in a significant gap in information accessibility.

Inefficiencies in Traditional Systems:

Existing document management systems and search engines are predominantly optimized for plain text and may not adequately interpret the varied content within PDF documents. This limitation leads to suboptimal user experiences, prolonged search times, and, in some cases, the outright failure to extract essential information from PDFs. Consequently, professionals across various domains, including legal, medical, and business, face impediments in their workflows due to the inefficiencies of these systems.

Need for Contextual Understanding:

Moreover, the current state of information retrieval systems lacks a conversational approach, restricting users to rigid search queries and predefined commands. The absence of a chatbot capable of engaging in dynamic conversations and understanding contextual nuances further hinders the user's ability to interact naturally with the system.

Proposal for a Solution:

The identified problems underscore the necessity for a specialized chatbot that can navigate the complexities of PDFs, offering an interactive and efficient means of information retrieval. The proposed solution involves the development of a chatbot empowered with advanced natural language processing and document processing techniques, enabling it to comprehend the intricacies of PDF documents and provide users with a seamless, conversational experience for effective information extraction.

In summary, the problem statement revolves around the inefficiencies of existing systems in handling the complexities of PDFs and the absence of a dedicated conversational chatbot to address these challenges, emphasizing the need for an innovative solution to enhance information retrieval from PDF documents.

# PROPOSED SOLUTION

Conversational Chatbot Enhancement:

Further enhance the chatbot's conversational capabilities to handle diverse user queries seamlessly.

Integrate Advanced Document Processing Techniques:

Explore and integrate advanced document processing techniques to refine the extraction of information from PDFs.

Optimize NLP Models:

Continuously optimize and fine-tune the NLP models to improve the chatbot's contextual understanding and response accuracy.

Explore Alternative Embeddings:

Investigate alternative embeddings, such as HuggingFaceInstructEmbeddings, to potentially enhance the system's understanding of complex document structures.

Enhance User Interface (UI):

Continue refining the UI using Streamlit, incorporating user-friendly features for a more intuitive interaction experience.

Integrate Dynamic Learning Mechanisms:

Explore the integration of dynamic learning mechanisms to enable the chatbot to adapt and improve based on ongoing user interactions (using OpenAI).

Scalability and Robustness Improvements:

Focus on enhancing the system's scalability and robustness to accommodate an increasing volume of data and ensure consistent performance.

Explore Model Repositories:

Investigate the utilization of model repositories, such as HuggingFaceHub, to explore alternative language models and stay updated with the latest advancements.

Continuous Monitoring and Maintenance:

Implement continuous monitoring and maintenance procedures to identify and address potential issues promptly, ensuring the system's reliability.

# 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()

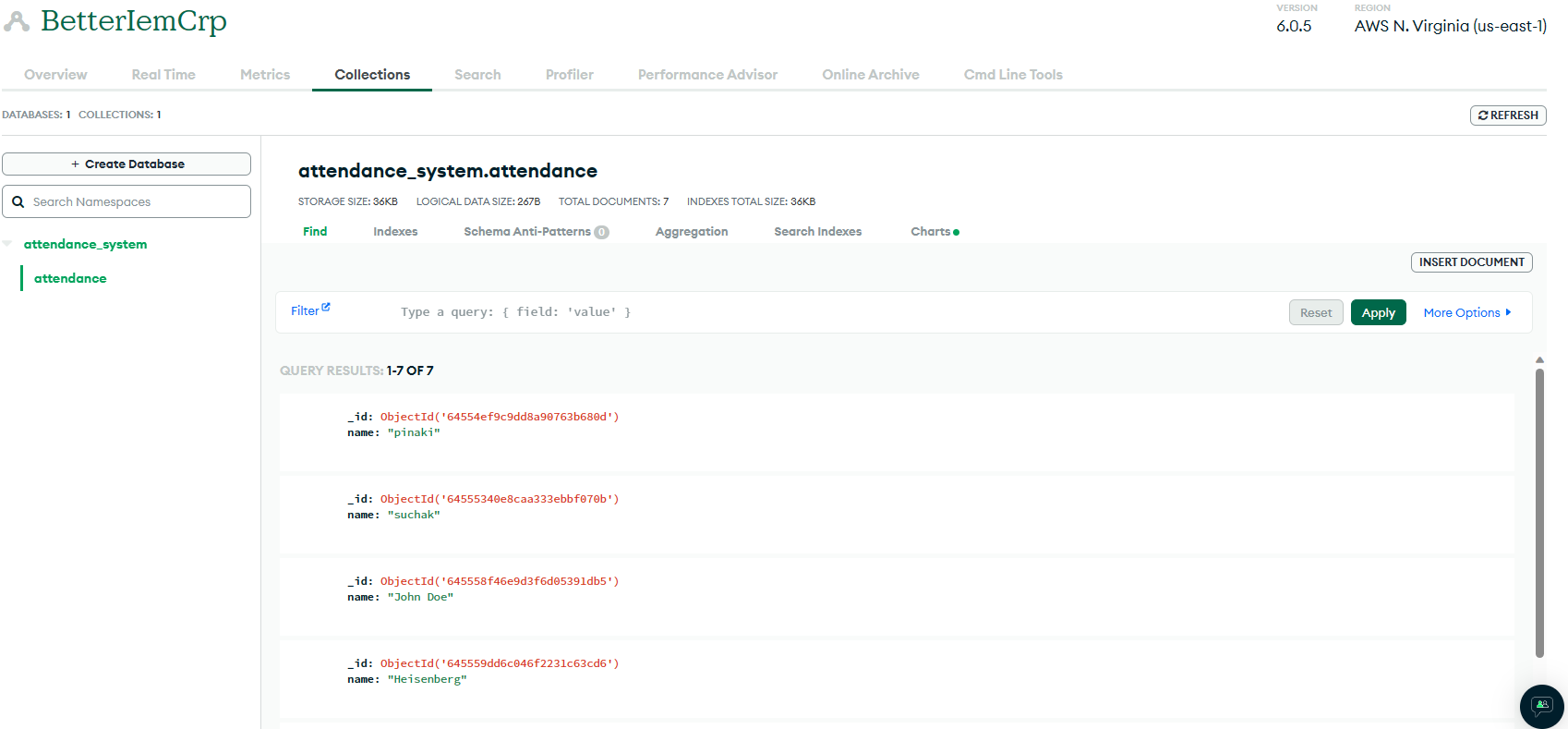
# create a connection with the MongoDB Atlas cluster

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]. |