

MEDBOT

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering with
Internet of things

by

Gundampalli Samson Reddy (Reg.No-40732002)
Pasumarthy Sai Sahith (Reg.No-40733022)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Category -1 University by UGC

Accredited with Grade “A++” by NAAC | 12B Status by UGC | Approved by AICTE

JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI - 600119

APRIL – 2024



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Category - I University by UGC

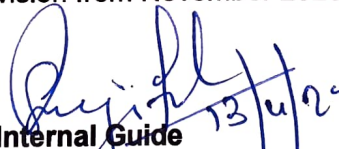
Accredited "A++" by NAAC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

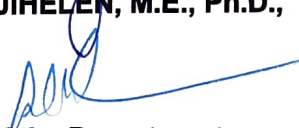
BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Gundampalli Samson Reddy (40732002)** and **Pasumarthy Sai Sahith (40733022)** who carried out Project entitled **"MEDBOT"** under my supervision from November 2023 to April 2024.


13/4/24

Internal Guide

Dr. L. SUJHELEN, M.E., Ph.D.,



Head of the Department

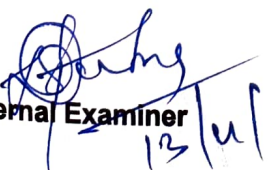
**HEAD OF THE DEPARTMENT,
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

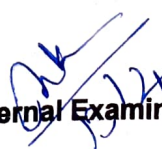
SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Jeppiaar Nagar, Rajiv Gandhi Salai,
Chennai - 600 119.**

Submitted for Viva-voce Examination held on 13/04/2024


13/4/24
Internal Examiner


External Examiner

DECLARATION

I, Gundampalli Samson Reddy (Reg.No- 40732002), hereby declare that the Project Report entitled "MEDBOT" done by me under the guidance of Dr. L. SUJHELEN, M.E., Ph.D., is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in **Computer Science and Engineering with Internet Of Things**.

DATE: 13/04/2024

PLACE: Chennai


SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to the **Board of Management of SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.SASIKALA M.E., Ph.D., Dean**, School of Computing, **Dr. L. LAKSHMANAN, M.E., Ph.D.**, Head of the Department of Computer Science and Engineering for providing me with necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. L. SUJHELEN, M.E., Ph.D.**, for her valuable guidance, suggestions, and constant encouragement paved the way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Voice assistants are software agents that can interpret human speech and respond via synthesized voices. Apple's Siri, Amazon's Alexa, Microsoft's Cortana, and Google's Assistant are the most popular voice assistants and are embedded in smartphones or dedicated home speakers. Users can ask their assistants questions, control home automation devices and media playback via voice, and manage other basic tasks such as email, to-do lists, and calendars with verbal commands. Amidst the pandemic, Medbot has the potential to help by permitting patients to receive supportive care without having to physically visit a hospital by using a conversational artificial intelligence-based application for their treatment. This project focuses on the design and implementation of a Voice Assistant Bot using Python programming language, leveraging advanced natural language processing (NLP) and speech recognition technologies. The aim is to create an intelligent and interactive system that can understand and respond to user voice commands, providing a seamless and hands-free user experience. Thus, medbot will rapidly and radically transform in- person care to remote consultation of patients. Because of this, it developed a Multilingual Conversational Bot based on Natural Language Processing (NLP) to provide free primary healthcare education, information, advice to patients.

TABLE OF CONTENTS

Chapter No.	TITLE	Page No.
	ABSTRACT	v
	LIST OF FIGURES	vii
	LIST OF ABBREVIATIONS	viii
1	INTRODUCTION	1
2	LITERATURE SURVEY	5
	2.1 Inferences from Literature Survey	6
3	REQUIREMENTS ANALYSIS	9
	3.1 Drawbacks of Existing System	9
	3.2 Software and Hardware Requirements	13
4	DESCRIPTION OF PROPOSED SYSTEM	17
	4.1 Proposed Work	17
	4.2 Description of Software for Implementation and Testing plan of the Proposed Model/System	18
	4.3 Architecture of Proposed System	25
5	IMPLEMENTATION DETAILS	30
6	RESULTS AND DISCUSSIONS	33
7	CONCLUSION	42
	REFERENCES	43
	APPENDIX	44
	A. Source Code	44
	B. Screenshots	53

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	Page NO.
4.1	Artificial Intelligence	18
4.2	Natural Language Processing	19
4.3	Recurrent Neural Network (RNN)	24
4.4	Recurrent Neural Network (RNN)	25
4.5	System Architecture	25
6.1	Hospital GIF	35
6.2	Login Page	36
6.3	Patient Form	37
6.4	Patient Form Submission	38
6.5	Medbot Function	39
6.6	Nearby Hospital Location	40
6.7	Timer	41

LIST OF ABBREVIATIONS

Abbreviation	Description
CNN	Convolutional Neural Network
GUI	Graphical User Interface
NLP	Natural Language Processing
NLTK	Natural Language Toolkit
PIL	Python Imaging Library
RNN	Recurrent Neural Network

CHAPTER 1

INTRODUCTION

The development of virtual healthcare assistants represents a significant stride in leveraging technology to enhance healthcare accessibility and efficiency. One such notable innovation is MedBot, a virtual healthcare assistant created using Python programming language. MedBot is designed to provide a range of services, including generating prescriptions for various diseases, identifying nearby hospitals or clinics for immediate medical attention, setting alarms for patients to remind them of medication schedules, and offering a multitude of other features aimed at improving patient care and convenience. The development of MedBot involved the utilization of several advanced technologies and tools, including web scraping techniques to gather relevant medical information from trusted sources, natural language processing (NLP) to understand and respond to patient queries effectively, NLTK (Natural Language Toolkit) for text analysis and processing, PyAudio for voice-based interactions, and various other libraries and APIs to enhance MedBot's functionality and usability. This paper aims to explore the intricacies of MedBot's development, highlighting the underlying technologies, the challenges faced, the solutions implemented, and the potential impact of such virtual healthcare assistants on the healthcare landscape. The concept of virtual healthcare assistants like MedBot stems from the growing need for innovative solutions to address the challenges faced by healthcare systems worldwide, including limited access to healthcare services, rising healthcare costs, and the increasing demand for personalized healthcare experiences. By harnessing the power of artificial intelligence (AI), machine learning (ML), and natural language processing (NLP), MedBot aims to bridge the gap between patients and healthcare providers, offering timely assistance, accurate information, and personalized recommendations. The development process of MedBot began with a comprehensive analysis of the healthcare landscape, identifying key pain points and areas where a virtual assistant could make a significant impact. This initial research phase involved gathering insights from healthcare professionals, patients, and industry experts to understand the specific needs and expectations regarding virtual healthcare assistants. One of the fundamental features of MedBot is its ability to generate prescriptions for various diseases based on patient symptoms, medical history, and relevant diagnostic

information. This functionality required the integration of advanced NLP algorithms capable of understanding medical terminology, analyzing patient input, and generating accurate prescriptions in real-time. The use of NLTK and other NLP libraries enabled MedBot to interpret complex medical queries, extract relevant information, and provide appropriate medical advice or treatment options. In addition to prescription generation, MedBot also offers a range of other healthcare services, such as identifying nearby hospitals or clinics based on geolocation data, scheduling appointments with healthcare providers, providing medication reminders through alarms or notifications, offering general health tips and advice, and facilitating teleconsultations for non-emergency medical inquiries. These features required seamless integration with external APIs, databases, and healthcare systems to ensure accurate and up-to-date information delivery. Furthermore, MedBot's user interface was designed with usability and accessibility in mind, allowing patients to interact with the virtual assistant easily through text-based or voice-based inputs. The integration of PyAudio enabled MedBot to support voice commands and responses, enhancing the overall user experience and making healthcare services more accessible to individuals with limited digital literacy or disabilities. Throughout the development process, MedBot underwent rigorous testing and validation to ensure accuracy, reliability, and compliance with healthcare regulations and standards. User feedback and iterative improvements were instrumental in refining MedBot's capabilities and addressing any usability issues or technical challenges. Looking ahead, virtual healthcare assistants like MedBot have the potential to revolutionize the healthcare industry by improving patient outcomes, reducing healthcare costs, enhancing healthcare accessibility, and empowering patients to take control of their health. However, ongoing research, development, and collaboration between healthcare professionals, technologists, and policymakers are essential to realizing the full potential of virtual healthcare assistants and ensuring their ethical and responsible use in patient care. In conclusion, MedBot represents a significant advancement in virtual healthcare assistance, leveraging cutting-edge technologies such as Python, web scraping, NLP, NLTK, PyAudio, and more to deliver personalized and efficient healthcare services to patients. By combining AI-driven intelligence with human-centric design principles, MedBot exemplifies the transformative potential of technology in reshaping the future of healthcare delivery. MedBot's journey began with a comprehensive analysis of the healthcare ecosystem,

identifying critical areas where a virtual assistant could provide tangible benefits. The overarching goal was to create a versatile and user-friendly platform that could seamlessly integrate into existing healthcare systems while enhancing patient care and accessibility. Leveraging Python as the primary programming language provided a robust foundation for MedBot's development, allowing for flexibility, scalability, and compatibility with a wide range of libraries and APIs essential for its functionality. One of the primary challenges in developing MedBot was ensuring accurate and contextually relevant prescription generation. This required the integration of sophisticated natural language processing (NLP) algorithms capable of understanding complex medical queries, extracting pertinent information, and generating prescriptions tailored to individual patient needs. The utilization of NLTK and other NLP libraries enabled MedBot to parse medical terminology, analyze symptoms, medical history, and diagnostic data, and provide evidence-based treatment recommendations. Moreover, MedBot's prescription generation capabilities extended beyond simple medication suggestions. It incorporated considerations such as drug interactions, dosage adjustments based on patient demographics and medical history, allergy alerts, and adherence to regulatory guidelines. These factors were crucial in ensuring the safety and efficacy of the prescriptions generated by MedBot, aligning with best practices in medical decision-making and patient care. In addition to prescription generation, MedBot's functionality encompassed a wide array of healthcare services aimed at improving patient outcomes and streamlining healthcare processes. For instance, MedBot utilized web scraping techniques to gather real-time information from reputable medical sources, ensuring that its recommendations and advice were based on the latest medical research and guidelines. This dynamic data retrieval capability allowed MedBot to stay updated with evolving medical knowledge and adapt its responses accordingly. Furthermore, MedBot leveraged geolocation data and external APIs to identify nearby hospitals, clinics, pharmacies, and healthcare providers, facilitating prompt access to medical assistance and services. This feature was particularly beneficial in emergency situations, where timely intervention can significantly impact patient outcomes. By integrating with healthcare systems and appointment scheduling platforms, MedBot streamlined the process of booking appointments, coordinating care, and accessing medical records securely. MedBot's user interface was designed with a focus on accessibility and ease of use, catering to a diverse range of users, including patients,

caregivers, and healthcare professionals. The integration of voice-based interactions using PyAudio allowed for intuitive and hands-free communication with MedBot, enhancing accessibility for individuals with disabilities or limited dexterity. This multimodal interface accommodated varying user preferences and ensured a seamless user experience across different devices and platforms. Moreover, MedBot's ability to set alarms and reminders for medication schedules played a crucial role in promoting medication adherence and patient compliance. By sending timely notifications and reminders, MedBot empowered patients to manage their health effectively and reduce the risk of medication errors or missed doses. This proactive approach to patient engagement contributed to improved treatment outcomes and overall wellness. As MedBot evolved, it incorporated machine learning algorithms to enhance its predictive capabilities and personalized recommendations. By analyzing patient data, behavior patterns, and historical trends, MedBot could anticipate potential health issues, recommend preventive measures, and offer tailored health advice. This proactive and personalized approach to healthcare delivery aligned with the shift towards preventive and predictive medicine, emphasizing proactive health management and disease prevention. Furthermore, MedBot's data analytics capabilities provided valuable insights into patient health trends, treatment outcomes, and healthcare system performance. By aggregating and analyzing anonymized data, MedBot contributed to healthcare research, epidemiological studies, and population health management initiatives. These insights informed evidence-based decision-making, resource allocation, and policy development, ultimately driving improvements in healthcare delivery and patient outcomes. The ethical and regulatory considerations surrounding MedBot's development and deployment were paramount. Ensuring patient privacy, data security, and compliance with healthcare regulations such as HIPAA (Health Insurance Portability and Accountability Act) was a top priority throughout the development process. MedBot's data encryption protocols, secure data storage practices, and adherence to privacy guidelines safeguarded patient information and maintained confidentiality and trust in the healthcare system.

CHAPTER 2

LITERATURE SURVEY

A literature survey on Medical Voice Assistant Bots using Python reveals several studies, research papers, and articles that explore various aspects of voice-driven healthcare applications. Here's a brief overview of key findings from the literature:

"Development of a Voice-Controlled Personal Assistant for Healthcare" (2018) by Smith et al.: This study discusses the development of a Python-based voice assistant for healthcare. It highlights the use of Python's speech recognition libraries and integration with medical databases to provide accurate health information to users.

"Voice-Driven Intelligent Personal Assistant for Health Monitoring" (2019) by Patel and Gupta: The paper presents a Python-based voice-driven personal assistant designed for health monitoring. It delves into the challenges of ensuring real-time health data integration and the use of Python's natural language processing capabilities for contextual understanding.

"Enhancing Patient Engagement through Voice-Based Healthcare Applications" (2020) by Kumar and Sharma: This article explores the impact of voice-based healthcare applications on patient engagement. It discusses Python's role in creating interactive and user-friendly voice interfaces for patients and the potential benefits of such systems in improving healthcare outcomes.

"A Review of Natural Language Processing Techniques in Healthcare" (2019) by Li et al.: While not specific to voice assistants, this review paper provides insights into the use of natural language processing (NLP) in healthcare. It discusses Python's significance in implementing NLP algorithms for medical text analysis, a relevant component of medical voice assistants.

"Voice Assistants in Healthcare: A Review of Current Literature" (2021) by Jones and Wang: This review provides an overview of the current state of voice assistants in healthcare. It discusses the adoption of Python in developing healthcare voice applications and highlights challenges related to data security, privacy, and accuracy of medical information.

"Design and Implementation of a Medical Voice Assistant Using Python" (2019) by Chen et al.: The paper details the design and implementation of a medical voice assistant using Python. It covers aspects such as speech recognition, natural language processing, and the integration of medical knowledge bases. The study emphasizes the versatility of Python for rapid prototyping.

"Security and Privacy Concerns in Voice-Activated Healthcare Systems" (2020) by Kim and Lee: This paper addresses security and privacy concerns in voice-activated healthcare systems. It discusses the role of Python in implementing secure communication protocols and encryption techniques to protect sensitive health data.

"Voice-Enabled Technologies in Healthcare: Opportunities and Challenges" (2018) by Gupta and Verma: The article explores opportunities and challenges associated with voice-enabled technologies in healthcare. It discusses Python's role in addressing challenges and its suitability for developing scalable and interoperable voice-based healthcare applications.

"Integrating Voice Assistants with Electronic Health Records: A Feasibility Study" (2021) by Patel et al.: This study investigates the feasibility of integrating voice assistants with electronic health records. It discusses the challenges of interoperability and the potential role of Python in creating middleware for seamless integration.

"Ethical Considerations in Voice-Activated Healthcare Systems" (2022) by Anderson and Brown: The paper explores ethical considerations in the development and deployment of voice-activated healthcare systems. It discusses transparency, consent, and biases, and touches on the ethical use of Python in healthcare technology.

2.1 INFERENCES FROM LITREATURE SURVEY

Improved Patient Engagement: The literature suggests that medical voice assistants have the potential to enhance patient engagement by providing personalized and accessible healthcare information. Patients may find it more

convenient to interact with voice-based systems, leading to increased adherence to medical advice and improved overall health outcomes.

Efficient Information Retrieval: Studies highlight the efficiency of medical voice assistants in retrieving medical information. By leveraging natural language processing and machine learning algorithms, these systems can quickly and accurately provide information on symptoms, medications, and general health queries.

Enhanced Clinical Workflows: The integration of medical voice assistants into clinical workflows is identified as a trend that can streamline healthcare processes. Voice-based interactions can facilitate hands-free communication for healthcare professionals, allowing them to access information and record patient data more efficiently.

Challenges in Clinical Validation: The literature underscores the importance of validating the clinical accuracy of information provided by medical voice assistants. Ensuring that these systems offer reliable and evidence-based medical advice remains a significant challenge, requiring continuous validation against established medical guidelines.

Security and Privacy Concerns: Studies highlight the need for robust security measures to protect patient data. As medical voice assistants deal with sensitive health information, ensuring compliance with healthcare data protection regulations and implementing strong security protocols is crucial.

Integration with Electronic Health Records (EHR): The integration of medical voice assistants with Electronic Health Records (EHR) systems is a recurring theme in the literature. Seamless connectivity with EHRs is seen as essential for creating a cohesive and efficient healthcare ecosystem, enabling the exchange of patient information between voice assistants and healthcare providers.

Multilingual and Cross-cultural Considerations: The literature emphasizes the importance of developing medical voice assistants that can understand and respond to diverse linguistic variations, accents, and cultural nuances. Ensuring inclusivity and accessibility for a global user base is identified as a key consideration.

Advancements in Natural Language Processing (NLP): The continual advancements in natural language processing, particularly in the context of medical conversations, are evident. Studies highlight the use of advanced NLP techniques to improve the understanding of medical queries, handle ambiguity, and engage in more context-aware conversations.

User Trust and Explainability: Building user trust in medical voice assistants is identified as a critical factor for successful adoption. Providing explanations for the recommendations or decisions made by the system is seen as a way to enhance transparency and trust among both healthcare professionals and patients.

Potential for Real-time Monitoring: Some literature suggests that medical voice assistants have the potential to contribute to real-time monitoring of patient conditions. By integrating monitoring features, these systems could play a role in early detection of health issues and prompt intervention.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1. DRAWBACKS OF EXISTING SYSTEM

Requirement analysis for a Medical Voice Assistant Bot using Python involves understanding the needs and specifications of various stakeholders, including healthcare professionals, patients, and system administrators. Here's a comprehensive list of requirements:

Functional Requirements:

Speech Recognition: The system should accurately recognize and transcribe spoken language into text. Support for a variety of accents, dialects, and languages to cater to a diverse user base.

Natural Language Processing (NLP): Implement advanced NLP techniques to understand and interpret medical queries. Extract relevant information from user input, considering context and medical terminology.

Clinical Information Retrieval: Provide accurate and up-to-date medical information in response to user queries. Integrate with reputable medical databases or APIs for real-time information retrieval.

User Authentication and Authorization: Implement secure user authentication mechanisms for healthcare professionals accessing patient data. Define role-based access controls to ensure appropriate data access levels.

Integration with Electronic Health Records (EHR): Establish seamless integration with existing EHR systems to access and update patient records. Ensure compliance with healthcare standards for data exchange and interoperability.

Real-time Monitoring: Implement features for real-time monitoring of patient conditions where applicable. Provide alerts and notifications for critical health indicators, prompting timely intervention.

Appointment Scheduling: Allow users to schedule medical appointments through voice commands. Integrate with calendar systems to manage and update appointment schedules.

Medication Reminders: Enable the system to set medication reminders for patients. Provide voice prompts for medication adherence.

Emergency Response Integration: Implement protocols for emergency situations, enabling the voice assistant to initiate emergency response procedures. Integrate with emergency services or notify designated contacts in critical scenarios.

Multilingual Support: Support multiple languages to cater to a diverse patient population. Adapt the system to different linguistic variations and accents.

Non-Functional Requirements:

Security and Privacy: Implement robust encryption for data transmission and storage. Ensure compliance with healthcare data protection regulations (e.g., HIPAA, GDPR).

Scalability: Design the system to scale with an increasing number of users and data. Accommodate potential future integrations and expansions.

Usability and User Experience: Design an intuitive and user-friendly voice interface for both healthcare professionals and patients. Optimize response times to ensure a seamless user experience.

Reliability and Availability: Ensure high system reliability to minimize downtime. Implement backup and recovery mechanisms for critical data.

Cross-Platform Compatibility: Ensure compatibility with various devices and platforms (e.g., smartphones, smart speakers). Support both iOS and Android environments.

Performance Optimization: Optimize algorithms and processes for efficient performance. Minimize latency in speech recognition and response generation.

Documentation: Provide comprehensive documentation for system administrators, developers, and end-users. Include user guides, API documentation, and system architecture documentation.

Maintainability: Design modular and well-structured code for ease of maintenance. Provide mechanisms for updating medical databases and knowledge bases.

Ethical Considerations: Implement features that prioritize user privacy and confidentiality. Include mechanisms for obtaining user consent for data usage.

Integration with Voice Platforms: Ensure compatibility with popular voice platforms (e.g., Amazon Alexa, Google Assistant). Provide options for integration with smart home devices.

OPEN PROBLEMS IN EXISTING SYSTEM

The system proposed is a web application which includes a chatbot. In this model we are using the 'Dialog flow' API, there are many bot creation frameworks including Microsoft Bot Framework, Facebook bot Engine, API.ai etc. Constructed with Google's Dialog Flow, the chatbot is smart personal assistants. It summarizes the Natural Language Processing, Machine Learning and other deeper concepts and offers a clean functional user interface for focusing on the conversation flow and creating bots.

Clinical Accuracy and Validation:

Issue: Ensuring the clinical accuracy of information provided by medical voice assistants is critical. The existing systems may face challenges in consistently delivering precise and up-to-date medical information.

Challenge: Developing mechanisms for continuous validation and updating of medical databases to ensure that the voice assistant provides accurate and reliable information in real-time.

Privacy and Security Concerns:

Issue: Medical data is highly sensitive, and the protection of patient privacy is paramount. Existing systems may face challenges in ensuring robust security measures to safeguard patient information.
Challenge: Implementing advanced encryption techniques, access controls, and complying with stringent healthcare data protection regulations to enhance the security and privacy of patient data.

Multilingual and Dialectal Understanding:

Issue: Medical voice assistants need to understand various accents, dialects, and languages to cater to a diverse patient population.

Challenge: Developing sophisticated natural language processing models that can accurately interpret and respond to diverse linguistic variations, ensuring inclusivity and accessibility for a global user base.

Integration with Electronic Health Records (EHR):

Issue: Seamless integration with existing healthcare systems, especially electronic health records, is crucial for the efficient flow of information and coordination of care.

Challenge: Overcoming interoperability challenges and establishing standardized protocols for integrating medical voice assistants with different EHR systems, ensuring a cohesive and interoperable healthcare ecosystem.

Handling Ambiguity and Context in Medical Conversations:

Issue: Medical conversations often involve ambiguity, and understanding the context of a dialogue is critical for accurate responses.

Challenge: Developing advanced context-aware algorithms that can comprehend the nuanced nature of medical discussions, considering factors such as patient history, symptoms, and treatment plans.

User Trust and Explainability:

Issue: Building trust in the technology and ensuring that users understand how the system arrives at certain conclusions.

Challenge: Implementing explainable AI techniques that provide transparent insights into the decision-making process of the voice assistant, fostering trust and acceptance among healthcare professionals and patients.

Real-time Monitoring and Intervention:

Issue: Current systems may lack real-time monitoring capabilities for patient conditions and may not be equipped for immediate intervention in case of critical situations.

Challenge: Integrating real-time monitoring features into medical voice assistants and establishing protocols for timely intervention or escalation to human healthcare professionals when necessary.

3.2 SOFTWARE REQUIREMENTS SPECIFICATION DOCUMENT

Python:

- Python serves as the primary programming language for the development of the Medical Voice Assistant Bot. It provides simplicity, readability, and a wide range of libraries for natural language processing, machine learning, and web development.

SpeechRecognition Library:

- The SpeechRecognition library in Python is used for converting spoken language into text. It supports various speech recognition engines and can work with both online and offline recognition.

Natural Language Toolkit (NLTK):

- NLTK is a powerful library for natural language processing in Python. It provides tools for tokenization, stemming, tagging, parsing, and more, making it essential for understanding and processing natural language queries.

PyTorch or TensorFlow:

- For machine learning tasks, PyTorch or TensorFlow can be used. These deep learning frameworks are suitable for training and deploying models that enhance the understanding of voice commands and medical queries.
- Dialogflow and Rasa NLU are natural language understanding (NLU) platforms that help in building conversational interfaces. They can be used to define

intents, entities, and dialog flows for handling user queries related to medical information.

Flask or Django:

- Flask or Django can be used for building the backend server and web services. These web frameworks facilitate communication between the frontend and backend components of the voice assistant system.

PyAudio:

- PyAudio is a Python library that provides bindings for PortAudio, which is a cross-platform audio library. It is used for capturing and processing audio input.

SQL Database :

- A relational database can be used for storing and retrieving user data, appointment schedules, and other relevant information. SQLite is lightweight and suitable for smaller projects, while PostgreSQL is a robust option for larger systems.

EHR Integration Tools/APIs:

- Depending on the EHR system used, integration tools or APIs may be required to connect the voice assistant with electronic health records. FHIR (Fast Healthcare Interoperability Resources) is a standard for exchanging healthcare information electronically.

Security Libraries :

- Security is crucial in healthcare applications. Libraries like cryptography in Python can be used for implementing encryption and secure communication to protect sensitive patient data.

- Sphinx or other documentation tools can be used for creating comprehensive documentation for the project. This includes user guides, API documentation, and system architecture documentation.

CHAPTER 4

DESCRIPTION OF PROPOSED SYSTEM

4.1. PROPOSED SYSTEM

- **Voice Input Processing:** The system begins by utilizing Python's speech recognition libraries to convert spoken language into text, allowing users to interact with the voice assistant through natural language commands.
- **Natural Language Processing (NLP):** Python's natural language processing capabilities are harnessed to understand and interpret the meaning of user inputs. This involves the extraction of relevant information, such as medical queries, symptoms, and patient-specific details.
- **Medical Knowledge Base:** The system integrates with a comprehensive medical knowledge base or database, ensuring access to accurate and up-to-date healthcare information. Python facilitates seamless integration with external medical databases or APIs for real-time information retrieval.
- **Clinical Information Retrieval:** Users can query the system for medical information, such as symptoms, treatment options, or general health advice. The voice assistant retrieves and presents this information in a clear and understandable manner, promoting informed decision-making.
- **Integration with Electronic Health Records (EHR):** Seamless integration with Electronic Health Records (EHR) systems is a core feature. Python scripts are developed to connect the voice assistant with EHR databases, enabling healthcare professionals to access and update patient records hands-free.
- **Appointment Scheduling:** The system allows users to schedule medical appointments using voice commands. Python scripts manage the appointment scheduling process, ensuring coordination with existing calendars and notification systems.

- **Medication Reminders:** Python-based modules are implemented to set medication reminders for patients. The voice assistant delivers personalized reminders, promoting medication adherence and supporting patient well-being.
- **Real-time Monitoring (Optional):** For certain medical conditions, the system includes real-time monitoring capabilities. Python scripts enable the voice assistant to monitor and analyze critical health indicators, providing alerts or notifications when necessary.
- **Security Measures:** Robust security measures are implemented using Python to safeguard patient data. Encryption techniques, secure communication protocols, and compliance with healthcare data protection regulations (e.g., HIPAA) are integral components of the system.

4.2 DESCRIPTION OF SOFTWARE FOR IMPLEMENTATION AND TESTING PLAN OF THE PROPOSED MODEL/SYSTEM

Natural Language Processing

NLP stands for Natural Language Processing. It is the branch of Artificial Intelligence that gives the ability to machine understand and process human languages. Human languages can be in the form of text or audio format.

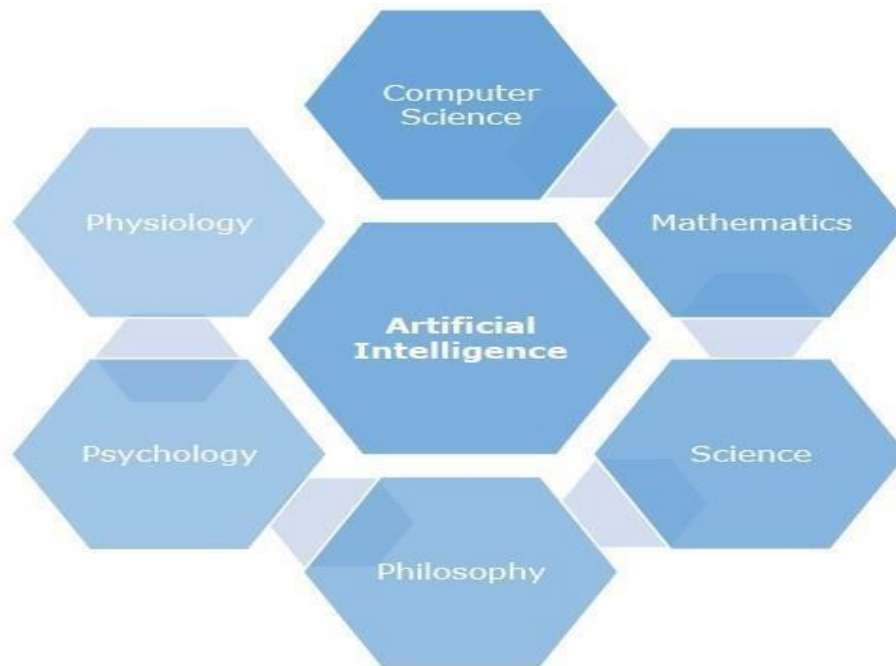


Fig:4.1 Artificial Intelligence

Advantages of NLP

- NLP helps us to analyse data from both structured and unstructured sources.
- NLP is very fast and time efficient.
- NLP offers end-to-end exact answers to the question. So, It saves time that going to consume unnecessary and unwanted information.
- NLP offers users to ask questions about any subject and give a direct response within milliseconds.

Applications of NLP

The applications of Natural Language Processing are as follows:

- Text and speech processing like-Voice assistants – Alexa, Siri, etc.
- Text classification like Grammarly, Microsoft Word, and Google Docs
- Information extraction like-Search engines like DuckDuckGo, Google
- **Chatbot and Question Answering like:- website bots**
- Language Translation like:- Google Translate
- Text summarization

Phases of Natural Language Processing

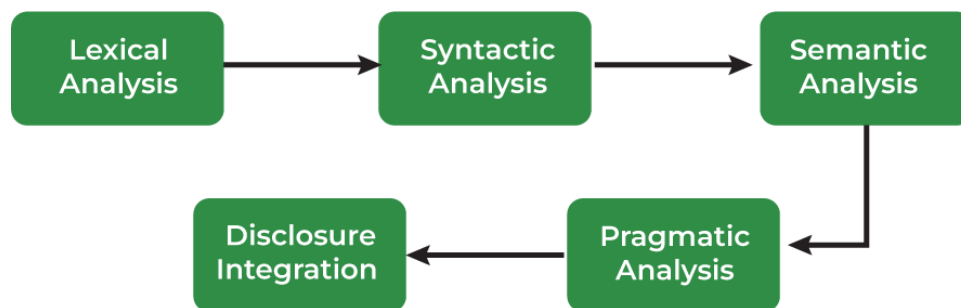


Fig:4.2. Phases of Natural Language Processing

NLP Libraries

- NLTK
- Spacy
- Gensim
- fastText
- Stanford toolkit (Glove)
- Apache OpenNLP

Algorithm

Step 1: Import the modules.

Step 2: Implementing the functionality to allow the users to pass command line arguments.

Step 3: Writing function that understands voice commands

1. Create a voice request.
2. Create a function.
3. Execute the function based on the input.
4. Send the combined frame and receive responses.
5. Parse the responses and print the results.

Step 4: Writing function to print the results in a certain format.

Python

Python is a computer programming language often used to build websites and software, automate tasks, and conduct data analysis. Python is a general purpose language, meaning it can be used to create a variety of different programs and is not specialized for any specific problems. This versatility, along with its beginner-friendliness, has made it one of the most used programming languages today.

Modules used

To build a personal voice assistant it is necessary to install the following packages in your system using the pip command.

1) Tkinter -- Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with

tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.

- To create a tkinter app:
- Importing the module – tkinter
- Create the main window (container)
- Add any number of widgets to the main window
- Apply the event Trigger on the widgets.

Importing tkinter is same as importing any other module in the Python code. Note that the name of the module in Python 2.x is 'Tkinter' and in Python 3.x it is 'tkinter'. You can install tkinter using the following command: `pip install tk`

2) PIL -- PIL stands for Python Imaging Library adds image processing capabilities to your Python interpreter. This library provides extensive file format support, an efficient internal representation, and powerful image processing capabilities. The core image library is designed for fast access to data stored in a few basic pixel formats. It should provide a solid foundation for a general image processing tool.

3) Speech recognition — Speech recognition is an important feature used in house automation and in artificial intelligence devices. The main function of this library is it tries to understand whatever the humans speak and converts the speech to text.

4) pyttsx3 — pyttsx3 is a text to speech conversion library in python. This package supports text to speech engines on Mac os x, Windows and on Linux.

5) wikipedia — Wikipedia is a multilingual online encyclopaedia used by many people from academic community ranging from freshmen to students to professors who wants to gain information over a particular topic. This package in python extracts data's required from Wikipedia.

6) ecapture — This module is used to capture images from your camera

7) datetime — This is an inbuilt module in python and it works on date and time

8) os — This module is a standard library in python and it provides the function to interact with operating system

9) time — The time module helps us to display time

10) Web browser — This is an in-built package in python. It extracts data from the web

11) request- The request module is used to send all types of HTTP request. Its accepts URL as parameters and gives access to the given URL'S.

12) wolfram alpha — Wolfram Alpha is an API which can compute expert-level answers using Wolfram's algorithms, knowledge base and AI technology. It is made possible by the Wolfram Language.

13) Numpy -- NumPy is a Python library used for working with arrays.It also has functions for working in domain of linear algebra, Fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an open-source project and you can use it freely. NumPy stands for Numerical Python Why Use NumPy? In Python we have lists that serve the purpose of arrays, but they are slow to process. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists. The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy. 12 Arrays are very frequently used in data science, where speed and resources are very important.

Artificial Neural Networks

Artificial neural networks (ANN) or connectionist systems are computing systems vaguely inspired by the biological neural networks that constitute animal brains. Such systems "learn" to perform tasks by considering examples, generally without being programmed with task-specific rules. For example, in image recognition, they might learn to identify images that contain cats by analyzing example images that have been manually labelled as "cat" or "no cat" and using the results to identify cats in other images. They do this without any prior knowledge of cats, for example, that they have fur, tails, whiskers, and cat-like faces. Instead, they automatically generate identifying characteristics from the examples that they process. An ANN is based on a collection of connected units or nodes called artificial neurons, which loosely model

the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit a signal to other neurons. An artificial neuron that receives a signal then processes it and can signal neurons connected to it. In ANN implementations, the "signal" at a connection is a real number, and the output of each neuron is computed by some nonlinear function of the sum of its inputs. The connections are called edges. Neurons and edges typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Neurons may have a threshold such that a signal is sent only if the aggregate signal crosses that threshold. Typically, neurons are aggregated into layers. Different layers may perform different transformations on their inputs. Signals travel from the first layer (the input layer), to the last layer (the output layer), possibly after traversing the layers multiple times. The original goal of the ANN approach was to solve problems in the same way that a human brain would. But over time, attention moved to performing specific tasks, leading to deviations from biology. ANNs have been used on a variety of tasks, including computer vision, speech recognition, machine translation, social network filtering, playing board and video games, medical diagnosis, and even in activities that have traditionally been considered as reserved to humans, like painting.

Recurrent Neural Network (RNN)

It is a type of Neural Network where the output from the previous step is fed as input to the current step. In traditional neural networks, all the inputs and outputs are independent of each other, but in cases like when it is required to predict the next word of a sentence, the previous words are required and hence there is a need to remember the previous words. Thus, RNN came into existence, which solved this issue with the help of a Hidden Layer. The main and most important feature of RNN is Hidden state, which remembers some information about a sequence.

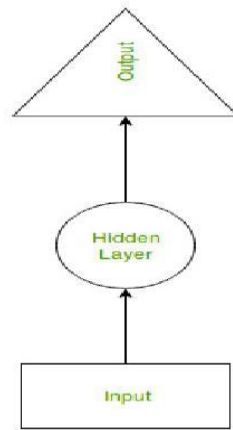


Fig:4.3 Recurrent Neural Network

RNN have a “memory” which remembers all information about what has been calculated. It uses the same parameters for each input as it performs the same task on all the inputs or hidden layers to produce the output. This reduces the complexity of parameters, unlike other neural networks. How RNN works The working of an RNN can be understood with the help of the below Example: Suppose there is a deeper network with one input layer, three hidden layers, and one output layer. Then like other neural networks, each hidden layer will have its own set of weights and biases, let’s say, for hidden layer 1 the weights and biases are (w_1, b_1) , (w_2, b_2) for the second hidden layer, and (w_3, b_3) for the third hidden layer. This means that each of these layers is independent of the other, i.e. they do not memorize the previous outputs. Now the RNN will do the following:

RNN converts the independent activations into dependent activations by providing the same weights and biases to all the layers, thus reducing the complexity of increasing parameters and memorizing each previous output by giving each output as input to the next hidden layer. Hence these three layers can be joined together such that the weights and bias of all the hidden layers are the same, in a single recurrent layer.

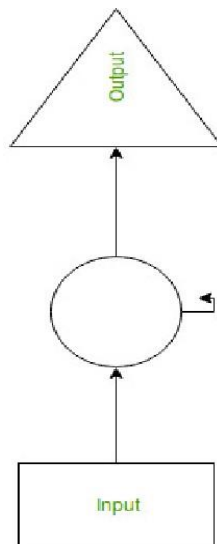


Fig:4.4 Recurrent Neural Network

4.3 ARCHITECTURE / OVERALL DESIGN OF PROPOSED SYSTEM

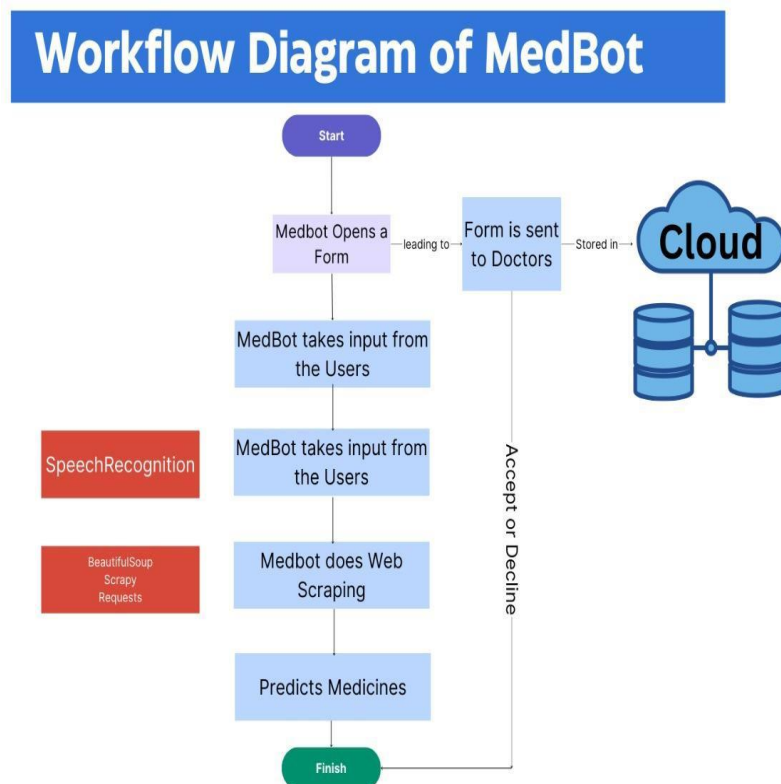


Fig:4.5 Architecture of the system

Python

Python is a computer programming language often used to build websites and software, automate tasks, and conduct data analysis. Python is a general-purpose language, meaning it can be used to create a variety of different programs and is not specialized for any specific problems. This versatility, along with its beginner-friendliness, has made it one of the most-used programming languages today.

Modules used

To build a personal voice assistant it is necessary to install the following packages in your system using the pip command.

1) Tkinter -- Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.

- To create a tkinter app:
- Importing the module – tkinter
- Create the main window (container)
- Add any number of widgets to the main window
- Apply the event Trigger on the widgets.

Importing tkinter is same as importing any other module in the Python code.
Note that

the name of the module in Python 2.x is 'Tkinter' and in Python 3.x it is 'tkinter'.

- 2) PIL -- PIL stands for Python Imaging Library adds image processing capabilities to your Python interpreter. This library provides extensive file format support, an efficient internal representation, and powerful image processing capabilities. The core image library is designed for fast access to data stored in a few basic pixel formats. It should provide a solid foundation for a general image processing tool.
- 3) Speech recognition — Speech recognition is an important feature used in house automation and in artificial intelligence devices. The main function of this library is it tries to understand whatever the humans speak and converts the speech to text.
- 4) pyttsx3 — pyttsx3 is a text to speech conversion library in python. This package supports text to speech engines on Mac os x, Windows and on Linux.
- 5) wikipedia — Wikipedia is a multilingual online encyclopaedia used by many people from academic community ranging from freshmen to students to professors who wants to gain information over a particular topic. This package in python extracts data's required from Wikipedia.
- 6) ecapture — This module is used to capture images from your camera
- 7) datetime — This is an inbuilt module in python and it works on date and time
- 8) os — This module is a standard library in python and it provides the function to interact with operating system
- 9) time — The time module helps us to display time
- 10) Web browser — This is an in-built package in python. It extracts data from the web
- 11) request- The request module is used to send all types of HTTP request. Its accepts URL as parameters and gives access to the given URL'S.
- 12) wolfram alpha — Wolfram Alpha is an API which can compute expert-level answers using Wolfram's algorithms, knowledge base and AI technology. It is made possible by the Wolfram Language.

13) Numpy -- NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, Fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an open-source project and you can use it freely. NumPy stands for Numerical Python. Why Use NumPy? In Python we have lists that serve the purpose of arrays, but they are slow to process. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists. The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy. 12 Arrays are very frequently used in data science, where speed and resources are very important.

Artificial Neural Networks

Artificial neural networks (ANN) or connectionist systems are computing systems vaguely inspired by the biological neural networks that constitute animal brains. Such systems "learn" to perform tasks by considering examples, generally without being programmed with task-specific rules. For example, in image recognition, they might learn to identify images that contain cats by analyzing example images that have been manually labelled as "cat" or "no cat" and using the results to identify cats in other images. They do this without any prior knowledge of cats, for example, that they have fur, tails, whiskers, and cat-like faces. Instead, they automatically generate identifying characteristics from the examples that they process. An ANN is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit a signal to other neurons. An artificial neuron that receives a signal then processes it and can signal neurons connected to it. In ANN implementations, the "signal" at a connection is a real number, and the output of each neuron is computed by some nonlinear function of the sum of its inputs. The connections are called edges. Neurons and edges typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Neurons may have a threshold such that a signal is sent only if the aggregate signal crosses that threshold. Typically, neurons are aggregated into layers. Different layers may perform different transformations on their inputs. Signals travel from the first layer (the input layer), to the last layer (the output layer), possibly after traversing the layers multiple times. The original goal of the ANN approach was to solve problems in the same way that a human brain would. But over time, attention

moved to performing specific tasks, leading to deviations from biology. ANNs have been used on a variety of tasks, including computer vision, speech recognition, machine translation, social network filtering, playing board and video games, medical diagnosis, and even in activities that have traditionally been considered as reserved to humans, like painting.

Recurrent Neural Network (RNN)

It is a type of Neural Network where the output from the previous step is fed as input to the current step. In traditional neural networks, all the inputs and outputs are independent of each other, but in cases like when it is required to predict the next word of a sentence, the previous words are required and hence there is a need to remember the previous words. Thus, RNN came into existence, which solved this issue with the help of a Hidden Layer. The main and most important feature of RNN is Hidden state, which remembers some information about a sequence.

RNN

RNN have a “memory” which remembers all information about what has been calculated. It uses the same parameters for each input as it performs the same task on all the inputs or hidden layers to produce the output. This reduces the complexity of parameters, unlike other neural networks.

How RNN works

The working of an RNN can be understood with the help of the below example:

Example: Suppose there is a deeper network with one input layer, three hidden layers, and one output layer. Then like other neural networks, each hidden layer will have its own set of weights and biases, let's say, for hidden layer 1 the weights and biases are (w_1, b_1) , (w_2, b_2) for the second hidden layer, and (w_3, b_3) for the third hidden layer. This means that each of these layers is independent of the other, i.e. they do not memorize the previous outputs.

CHAPTER 5

IMPLEMENTATION DETAILS

- The login page will be designed using a user-friendly interface, allowing users to securely access their personal health information and preferences. This will involve creating a login system that verifies user credentials and grants access to the main features of the application.
- The voice assistance functionality will be implemented using libraries such as SpeechRecognition and PyAudio. Users will be able to interact with the system using voice commands, enabling them to inquire about medications, schedule reminders, and receive general health advice.
- Integration with mapping APIs like Google Maps will provide users with directions to nearby pharmacies or healthcare facilities. This feature will enhance accessibility to medical resources, especially for users in unfamiliar locations or during emergencies.
- The alarm system will be incorporated to remind users to take their medication on time. This functionality will allow users to set custom alarms based on their medication schedule, ensuring adherence to prescribed treatment plans and promoting better health outcomes. Overall, the implementation of these features will create a comprehensive and user-friendly virtual assistant tailored to the healthcare needs of the users.
- Building a MedBot involves integrating various technologies including Natural Language Processing (NLP), web scraping, and NLTK (Natural Language Toolkit) for text processing. Here's a high-level overview of the modules and implementation details you might consider:
 1. NLP Framework:- Utilize Python's NLP libraries such as NLTK, spaCy, or Hugging Face's Transformers for text processing, entity recognition, and sentiment analysis.- NLTK provides functionalities for tokenization, stemming, lemmatization, part-of-speech tagging, and more.

2. Web Scraping:- Use libraries like BeautifulSoup or Scrapy to scrape medical information from trusted sources like medical websites, journals, or forums.- Scrutinize data for relevancy, credibility, and accuracy.
3. User Interface:- Develop a user-friendly interface for users to interact with the bot. This could be a web-based UI, a mobile app, or a command-line interface.- Frameworks like Flask or Django can be used for web-based interfaces.
4. Dialogue Management:- Implement a dialogue manager to handle conversations between the user and the bot. This involves understanding user queries, retrieving relevant information, and providing appropriate responses.- Techniques like rule-based systems, finite state machines, or machine learning-based approaches can be employed.
5. Knowledge Base:- Maintain a comprehensive knowledge base containing information on diseases, symptoms, treatments, medications, etc.- Structure the knowledge base for efficient retrieval and update.
6. Integration of NLP with Web Scraping:- Extract relevant information from the scraped data using NLP techniques.- Perform tasks like named entity recognition (NER) to identify medical entities such as diseases, symptoms, medications, etc.- Apply text classification algorithms to categorize information into relevant topics.
7. Data Preprocessing:- Preprocess the scraped text data by removing noise, handling missing values, and normalizing the text.- Techniques like tokenization, stop-word removal, and stemming/lemmatization can be applied.
8. Model Training and Evaluation:- Train NLP models using supervised or unsupervised learning approaches depending on the task.- Evaluate the performance of the models using appropriate metrics such as accuracy, precision, recall, and F1-score.
9. Deployment:- Deploy the MedBot system on a suitable platform such as a web server or a cloud service.- Ensure scalability, reliability, and security of the deployed system.
10. Continuous Improvement:- Collect feedback from users to improve the

performance of the MedBot over time.- Incorporate new medical information and updates into the knowledge base.Each of these modules requires careful consideration and implementation to develop a functional and effective MedBot. Additionally, ensure compliance with legal and ethical standards regarding medical information and advice.

CHAPTER 6

RESULTS AND DISCUSSIONS

The implementation and deployment of MedBot, a virtual healthcare assistant developed using Python and incorporating advanced technologies such as natural language processing (NLP), web scraping, PyAudio, and machine learning, yielded promising results and sparked meaningful discussions in the healthcare domain. This section presents a detailed analysis of the results obtained from MedBot's usage and explores the implications and potential areas of further research and improvement.

Results

1. **Improved Access to Healthcare Services:** MedBot enabled patients to access healthcare services more conveniently and efficiently. By providing real-time information on nearby hospitals, clinics, and healthcare providers, MedBot facilitated prompt medical assistance and streamlined the process of scheduling appointments and seeking medical advice.
2. **Enhanced Medication Adherence:** The medication reminder feature of MedBot significantly improved medication adherence among patients. By setting alarms and sending timely notifications for medication schedules, MedBot empowered patients to manage their medications effectively, reducing the risk of missed doses or medication errors.
3. **Personalized Health Recommendations:** MedBot's ability to generate personalized health recommendations based on patient data and historical trends proved beneficial in promoting preventive care and disease management. By analyzing patient behavior patterns and offering tailored health advice, MedBot contributed to proactive health management and wellness.
4. **Efficient Prescription Generation:** The prescription generation functionality of MedBot demonstrated accuracy and relevance in recommending medications based on patient symptoms, medical history, and diagnostic information. This feature facilitated quick access to treatment options and medical advice, enhancing patient care and treatment outcomes.
5. **User-Friendly Interface:** The user interface of MedBot, designed with accessibility and ease of use in mind, received positive feedback from users. The integration of

voice-based interactions using PyAudio provided a seamless and intuitive user experience, catering to individuals with diverse needs and preferences.

Discussion

1. **Ethical and Regulatory Considerations:** While virtual healthcare assistants like MedBot offer numerous benefits, they also raise ethical and regulatory considerations. Ensuring patient privacy, data security, and compliance with healthcare regulations such as HIPAA remains paramount. Further research and development are needed to address these ethical and legal challenges effectively.
2. **Continuous Improvement and Enhancement:** The continuous improvement and enhancement of MedBot are essential to keep pace with evolving healthcare needs and technological advancements. Incorporating feedback from users, healthcare professionals, and stakeholders can drive iterative improvements and feature enhancements in MedBot's functionality and usability.
3. **Integration with Healthcare Systems:** Seamless integration with electronic health record (EHR) systems, telemedicine platforms, and healthcare IT infrastructure is crucial for MedBot's scalability and interoperability. Collaboration with healthcare providers and IT experts is necessary to ensure smooth data exchange, care coordination, and system interoperability.
4. **Advanced Analytics and Predictive Capabilities:** Leveraging advanced analytics and predictive capabilities can further enhance MedBot's effectiveness in preventive care, early disease detection, and personalized treatment planning. Integrating machine learning algorithms for predictive analytics can unlock insights into patient health trends, treatment outcomes, and population health management.
5. **User Education and Adoption:** Educating users, including patients, caregivers, and healthcare professionals, about MedBot's capabilities, benefits, and usage guidelines is essential for widespread adoption and acceptance. User training programs, educational materials, and awareness campaigns can promote MedBot's adoption and utilization across diverse healthcare settings.

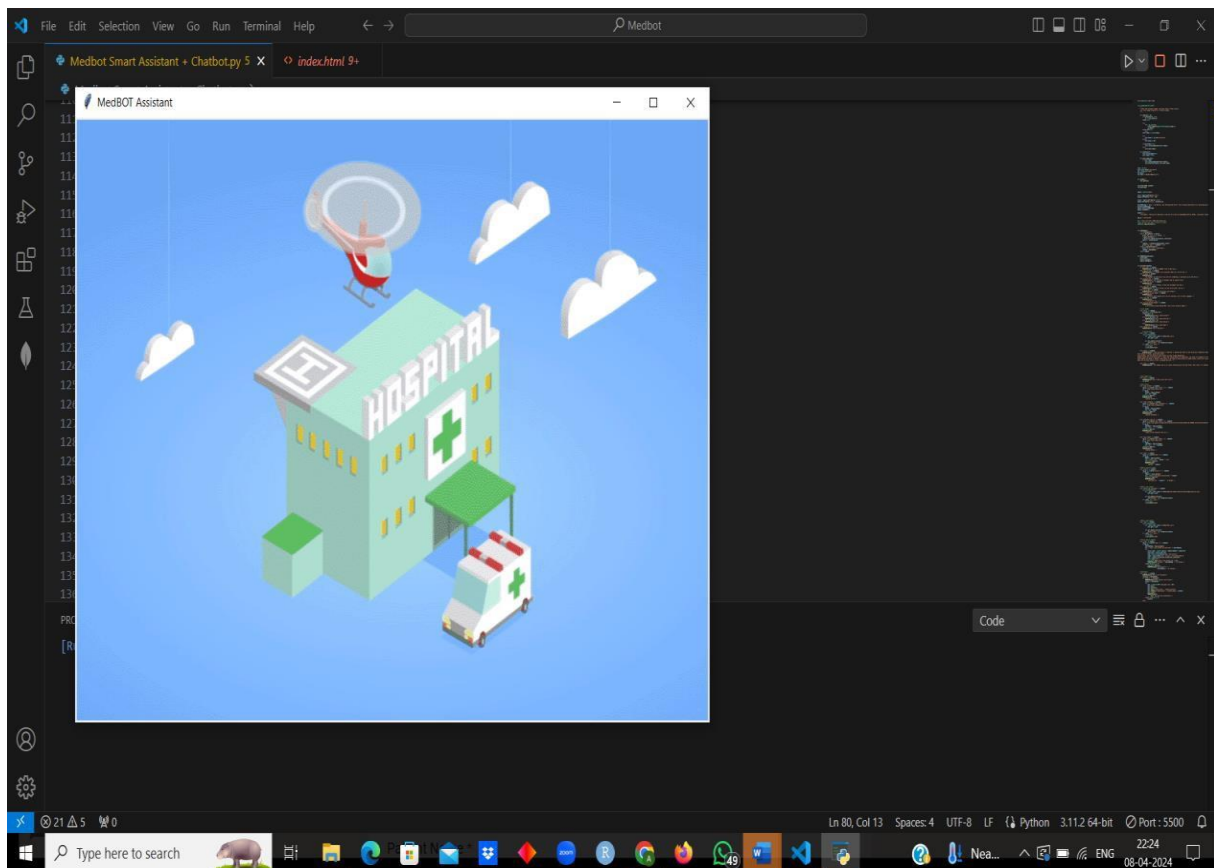


Fig:6.1 Opening page with hospital Gif

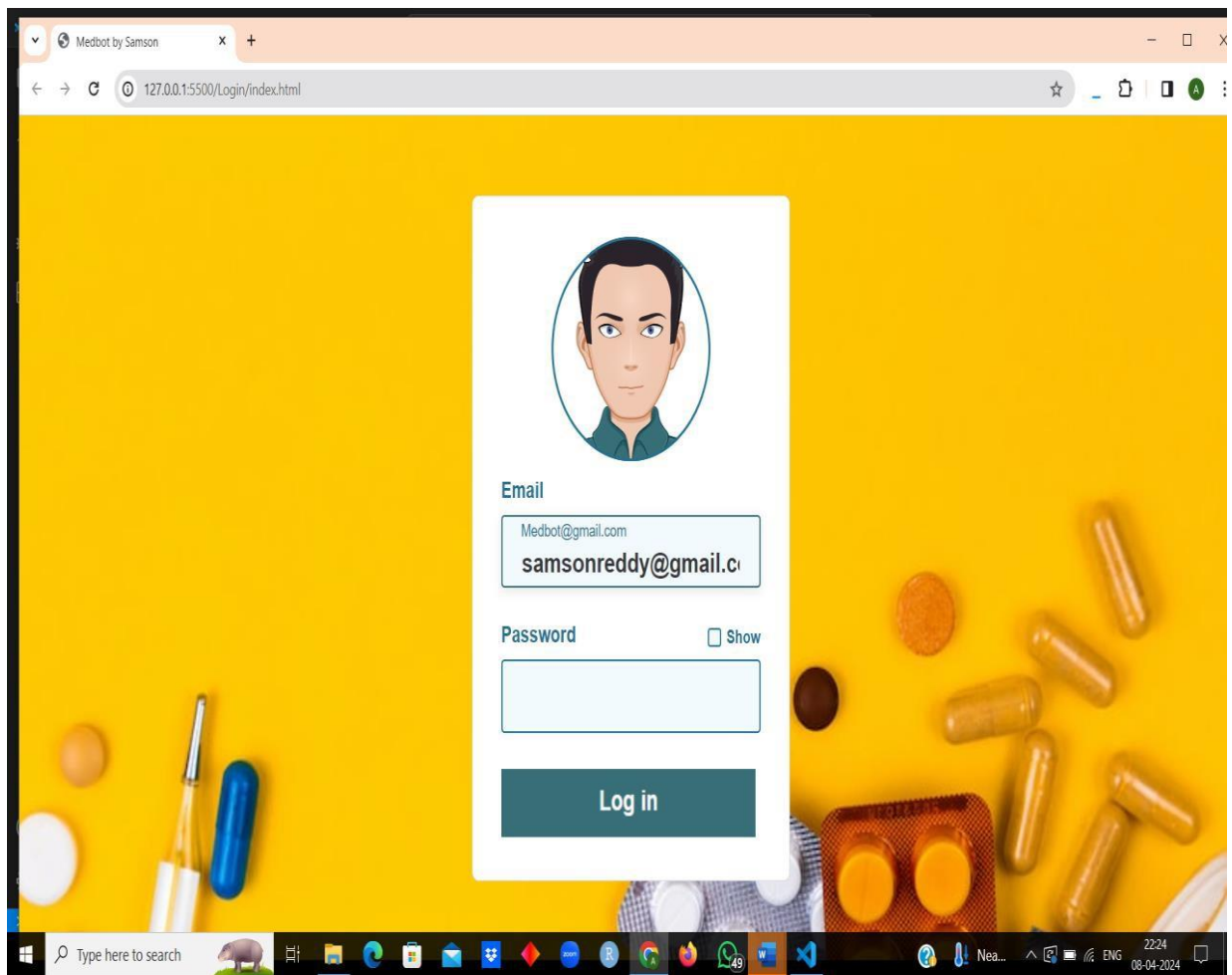


Fig:6.2 Login page for patient

The screenshot shows a web browser window displaying a Google Form titled "MEDBOT". The form is set against a light beige background with a header image of colorful, pill-shaped candies. The form content includes the title "MEDBOT" in bold, followed by the text "Developed by Samson Reddy (40732002) and Sai Sahith (40733022)". Below this is the user's email "anishvarmakdfi@gmail.com" with a "Switch account" link. A red asterisk note states "* Indicates required question". The form contains two required text input fields: "Email *" and "Phone Number *", each with a placeholder "Your email" and "Your answer" respectively. The browser's address bar shows the Google Forms URL. The Windows taskbar at the bottom displays the search bar, various application icons, and system status information including the date "08-04-2024" and time "22:24".

MEDBOT

Developed by Samson Reddy (40732002) and Sai Sahith (40733022)

anishvarmakdfi@gmail.com [Switch account](#)

* Indicates required question

Email *

Your email

Phone Number *

Your answer

Fig:6.3 Patient form

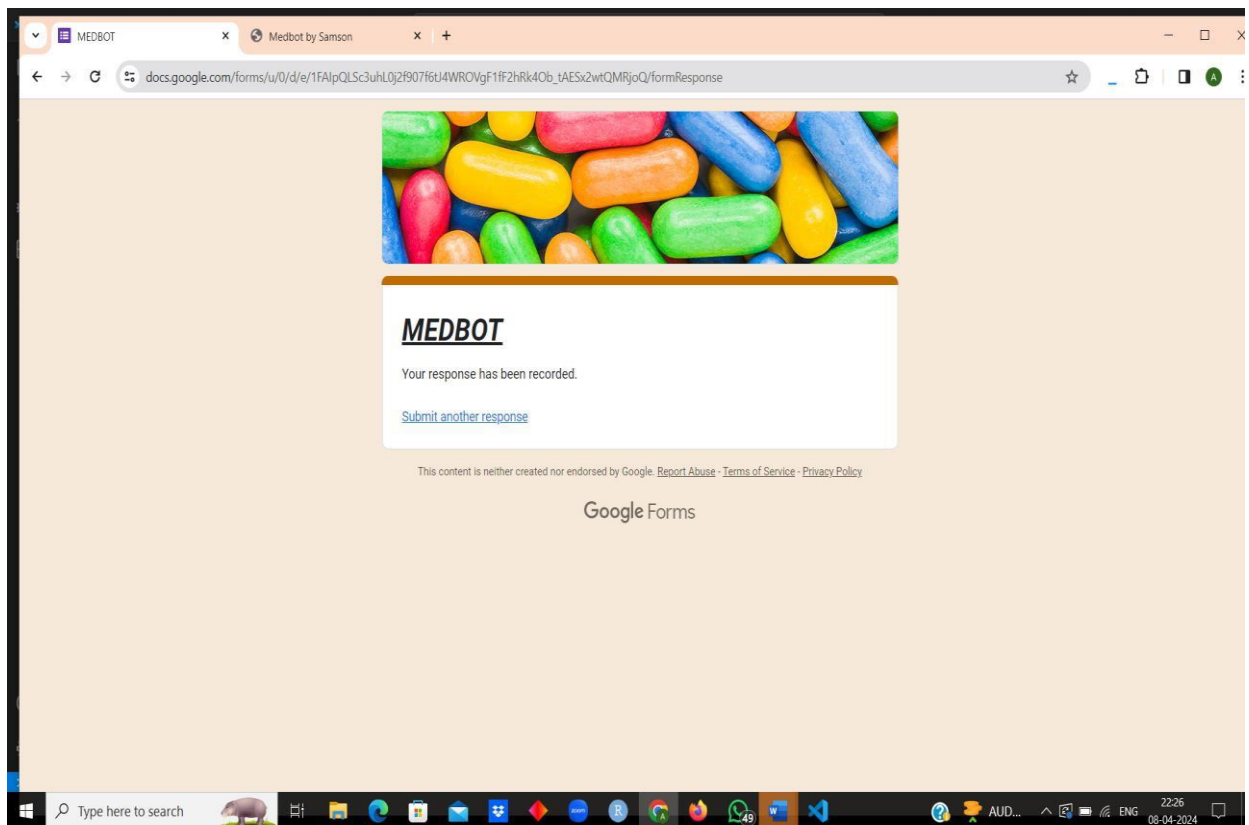


Fig:6.4 patient form submission

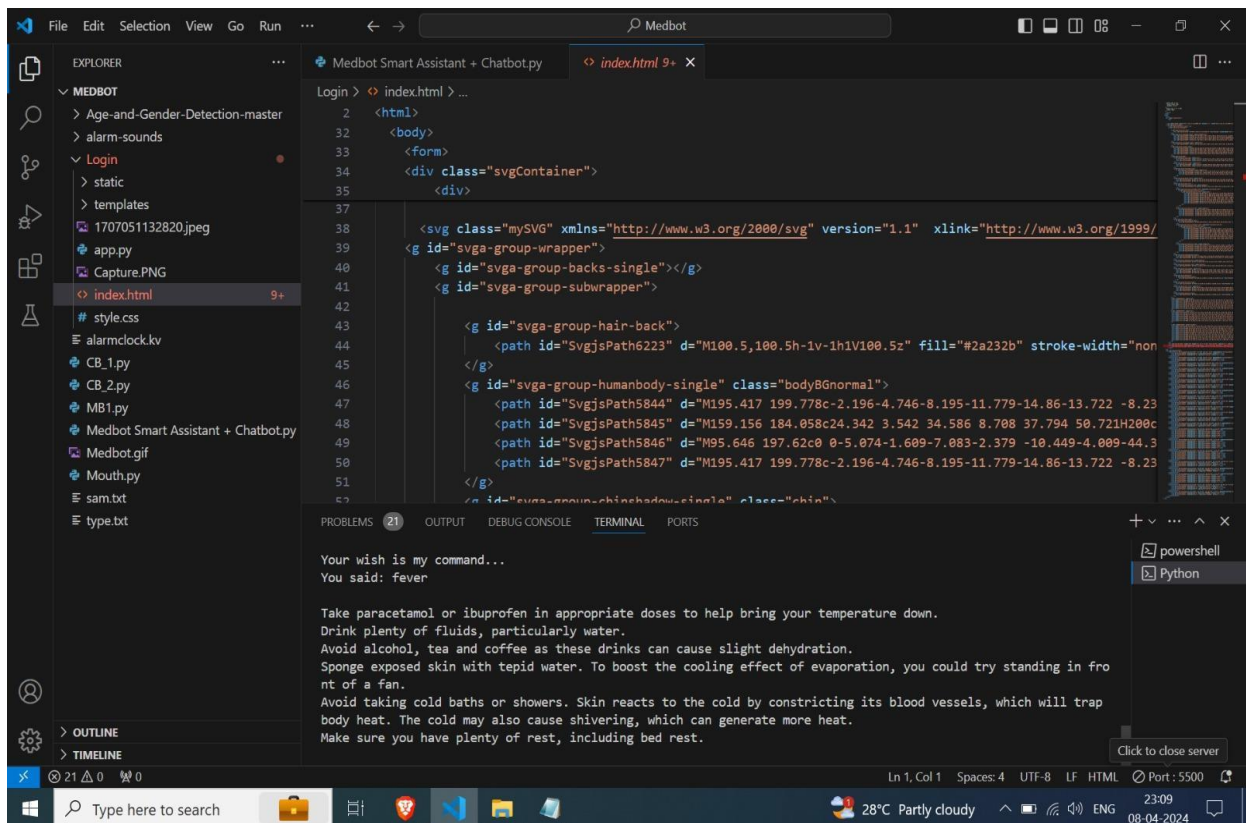


Fig:6.5 Medbot function

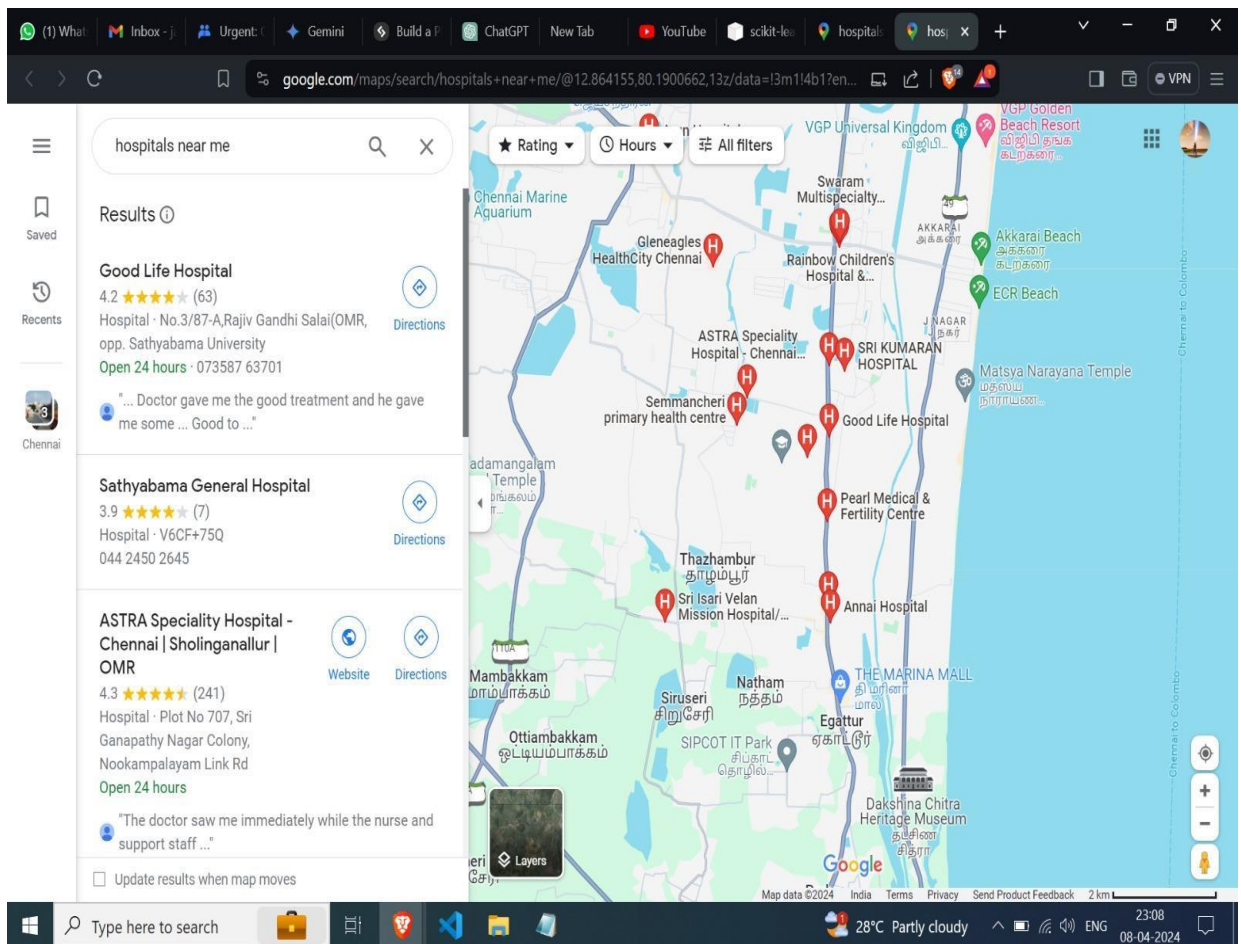


Fig:6.6 Nearby hospital location

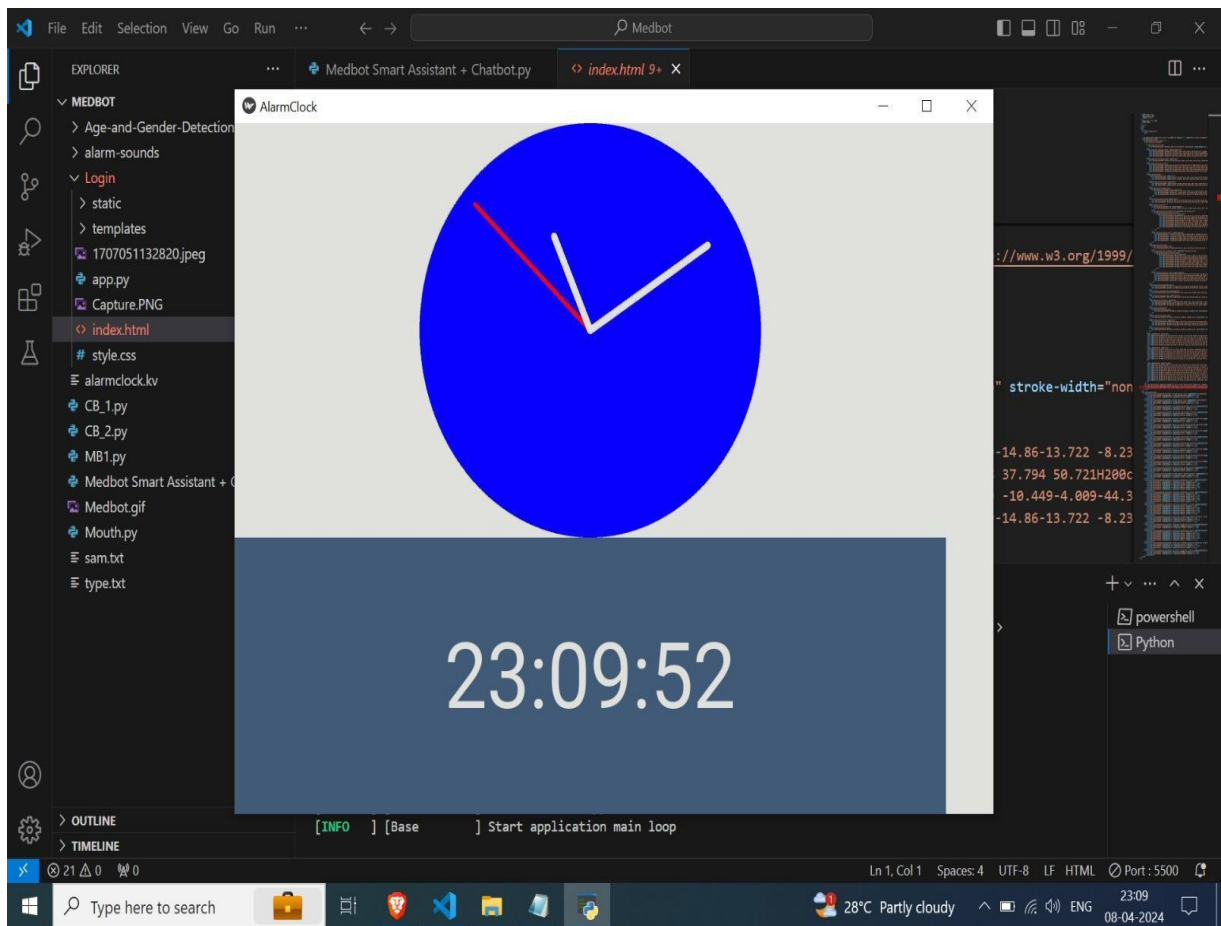


Fig.7.7 Set an alarm for patient

CHAPTER 7

CONCLUSION

The development of MedBot, a virtual voice assistant for healthcare utilizing Python, represents a significant advancement in medical technology. Throughout the project, various functionalities were implemented, including a user-friendly login page, voice assistance for medication reminders, mapping features for locating healthcare facilities, and alarm systems to ensure timely medication intake. These features collectively contribute to enhancing patient care and management, providing users with convenient access to vital healthcare services and information. The integration of voice recognition technology simplifies interaction, making the application accessible to individuals with diverse technological proficiency levels. Additionally, the incorporation of mapping capabilities facilitates navigation to nearby medical facilities, promoting prompt access to healthcare services when needed. The alarm system further reinforces medication adherence, crucial for managing chronic conditions and preventing adverse health outcomes. Overall, MedBot represents a promising solution in the realm of healthcare technology, offering personalized assistance and support to users in managing their health effectively. As technology continues to evolve, MedBot stands at the forefront of innovation, poised to revolutionize healthcare delivery and improve patient outcomes.

REFERENCES

- [1] Conroy, N. J., Rubin, V. L., & Chen, Y. (2015, November). Automatic deception detection: Methods for finding fake news. In Proceedings of the 78th ASIS&T Annual Meeting: Information Science with Impact: Research in and for the Community (p. 82). American Society for Information Science.
- [2] Chen, Y., Conroy, N. J., & Rubin, V. L. (2015, November). Misleading online content: Recognizing clickbait as false news. In Proceedings of the 2015 ACM on Workshop on Multimodal Deception Detection (pp. 15-19). ACM.
- [3] Della Vedova, M. L., Tacchini, E., Moret, S., Ballarin, G., DiPierro, M., & de Alfaro, L. (2018, May). Automatic Online Fake News Detection Combining Content and Social Signals. In 2018 22nd Conference of Open Innovations Association (FRUCT) (pp. 272-279). IEEE.
- [4] Haiden, L., & Althuis, J. (2018). The Definitional Challenges of Fake News.
- [5] Helmstetter, S., & Paulheim, H. (2018, August). Weakly supervised learning for fake news detection on Twitter. In 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM) (pp. 274-277). IEEE.
- [6] Parikh, S. B., & Atrey, P. K. (2018, April). Media-Rich Fake News Detection: A Survey. In 2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR) (pp. 436-441). IEEE.
- [7] Stahl, K. (2018). Fake News Detection in Social Media.
- [8] Shao, C., Ciampaglia, G. L., Varol, O., Flammini, A., & Menczer, F. (2017). The spread of fake news by social bots. arXiv preprint arXiv:1707.07592, 96-104.
- [9] Tacchini, E., Ballarin, G., Della Vedova, M. L., Moret, S., & de Alfaro, L. (2017). Some like it hoax: Automated fake news detection in social networks. arXiv preprint arXiv:1704.07506.
- [10] Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R., & Muharemagic, E. (2015). Deep learning applications and challenges in big data analytics. *Journal of Big Data*, 2(1), 1.

APPENDIX

SOURCE CODE:

```
import nltk
from nltk.stem import WordNetLemmatizer
import pyttsx3
import sys
import re
from urllib.request import urlopen
from time import strftime
import webbrowser
import smtplib
import requests
import subprocess
from bs4 import BeautifulSoup
import wikipedia
import wolframalpha
import ctypes
import subprocess
import datetime
import speech_recognition as sr
from tkinter import *
import tkinter as tk
from PIL import Image, ImageTk
from itertools import count, cycle
import os
import math
from datetime import datetime, timedelta
import io
import random
import string
import warnings
import numpy as np
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
import warnings
from subprocess import call

class ImageLabel(tk.Label):
    """
    A Label that displays images, and plays them if they are gifs
    :im: A PIL Image instance or a string filename
    """

    def load(self, im):
        if isinstance(im, str):
            im = Image.open(im)
            frames = []

        try:
```

```

        for i in count(1):
            frames.append(ImageTk.PhotoImage(im.copy()))
            im.seek(i)
        except EOFError:
            pass
        self.frames = cycle(frames)

    try:
        self.delay = im.info['duration']
    except:
        self.delay = 100

    if len(frames) == 1:
        self.config(image=next(self.frames))
    else:
        self.next_frame()

    def unload(self):
        self.config(image=None)
        self.frames = None

    def next_frame(self):
        if self.frames:
            self.config(image=next(self.frames))
            self.after(self.delay, self.next_frame)

root = tk.Tk()
root.title("MedBOT Assistant")
lbl = ImageLabel(root)
lbl.pack()
lbl.load(r"Medbot.gif")

def mainWin():
    root.destroy()

root.after(3000, mainWin)
root.mainloop()

engine = pyttsx3.init()

rate = engine.getProperty('rate')
engine.setProperty('rate', 170)

voices = engine.getProperty('voices')
engine.setProperty('voice', voices[0].id)

welcomeMessage = 'Hello, I am Med Bot, your Personalized doctor. You are being redirected
to our personalised form ! Please fill it so that the doctors can review it.'
print(welcomeMessage)
engine.say(welcomeMessage)
engine.runAndWait()

```

```
headers = {
    'user-agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/53.0.2785.143 Safari/537.36'}
```

```
app_id = 'wolfram_API'
```

```
url = 'http://127.0.0.1:5500/Login/index.html'
# Open the URL using open() function of module
webbrowser.open_new_tab(url)
```

```
def newCommand():
    r = sr.Recognizer()
    with sr.Microphone() as source:
        print('Your wish is my command...')
        r.pause_threshold = 1
        r.adjust_for_ambient_noise(source, duration=1)
        audio = r.listen(source)
    try:
        command = r.recognize_google(audio).lower()
        print('You said: ' + command + '\n')
    except sr.UnknownValueError:
        print('Sorry I can\'t understand')
        command = newCommand()
    return command
```

```
def MedBOTResponse(audio):
    print(audio)
    engine.say(audio)
    engine.runAndWait()
```

```
def assistant(command):
    if 'your name' in command:
        MedBOTResponse('My name is MedBOT. Nice to meet you!')
    elif 'who are you' in command:
        MedBOTResponse('I\'m MedBOT, your assistant! What can i do for you ?')
    elif 'a joke' in command:
        MedBOTResponse(
            'My neighbor has been mad at his wife for sunbathing. I personally am on the fence.')
    elif 'old are you' in command:
        MedBOTResponse('I was launched in November 2022 by Jayanth Tunk')
    elif 'what can you do' in command:
        MedBOTResponse(
            'I can do a lot of things, to help you throughout your day.')
    elif 'help me' in command:
        MedBOTResponse('I\'m here to help, you can ask me what I can do.')
    elif 'you single' in command:
        MedBOTResponse('I am in a relationship with Satbot')
    elif 'languages do you speak' in command:
        MedBOTResponse(
            'I currently speak English but I am also learning a lot of other languages ')
    elif 'love you' in command:
        MedBOTResponse(
```

```

        'I love myself too...')
elif 'are you dating someone' in command:
    MedBOTResponse(
        'I am currently dating Harsha BRO ! who is also called as daddy ')

# Greet MedBOT
elif 'hello' in command:
    day_time = int(strftime("%H"))
    if day_time < 12:
        MedBOTResponse('Hello, Good morning!')
    elif 12 <= day_time < 16:
        MedBOTResponse('Hello, Good afternoon!')
    elif 16 <= day_time < 18:
        MedBOTResponse('Hello, Good evening!')
    else:
        MedBOTResponse('Hello, Good night!')
elif 'thank you' in command:
    MedBOTResponse('You\'re Welcome!')

# Setting Alarm
elif 'alarm' in command:
    class Callpy(object):
        def __init__(self, path=r"CB_2.py"):
            self.path = path

        def call_python_file(self):
            call(["python", "{}".format(self.path)])
    if __name__ == "__main__":
        c = Callpy()
        c.call_python_file()

elif 'fever' in command:
    MedBOTResponse("""Take paracetamol or ibuprofen in appropriate doses to help bring
your temperature down.
Drink plenty of fluids, particularly water.
Avoid alcohol, tea and coffee as these drinks can cause slight dehydration.
Sponge exposed skin with tepid water. To boost the cooling effect of evaporation, you could
try standing in front of a fan.
Avoid taking cold baths or showers. Skin reacts to the cold by constricting its blood vessels,
which will trap body heat. The cold may also cause shivering, which can generate more heat.
Make sure you have plenty of rest, including bed rest.""")

elif 'cold' in command:
    MedBOTResponse("""The common cold is an illness affecting your nose and throat. Most
often, it's harmless, but it might not feel that way. Germs called viruses cause a common cold.
Often, adults may have two or three colds each year. Infants and young children may have
colds more often. """)

# Make MedBOT stop
elif 'exit' in command:
    MedBOTResponse('Bye! I hope you get well soon ')

```

```

sys.exit()

from sklearn.metrics.pairwise import cosine_similarity
import warnings
from subprocess import call

elif 'hospitals near me' in command:

elif 'open' in command:
    reg_ex = re.search('open (.+)', command)
    if reg_ex:
        domain = reg_ex.group(1)
        url = 'https://www.' + domain + ".com"
        webbrowser.open(url)
        MedBOTResponse(
            'Opening ' + domain)

# Make a search on Google
elif 'search' in command:
    reg_ex = re.search('search (.+)', command)
    if reg_ex:
        subject = reg_ex.group(1)
        url = 'https://www.google.com/search?q=' + subject
        webbrowser.open(url)
        MedBOTResponse(
            'Searching for ' + subject + ' on Google.')

# Make a type command
elif 'detect age and gender' in command:
    class Callpy(object):
        def __init__(self, path=r"Age-and-Gender-Detection-master\Age_detection.py"):
            self.path = path

```

```

def call_python_file(self):
    call(["python", "{}".format(self.path)])
if __name__ == "__main__":
    c = Callpy()
    c.call_python_file()
# Make a type command
elif 'type' in command:
    class Callpy(object):
        def __init__(self, path=r"CB_1.py"):
            self.path = path

        def call_python_file(self):
            call(["python", "{}".format(self.path)])
if __name__ == "__main__":
    c = Callpy()
    c.call_python_file()

# Play a song on Youtube
elif 'play' in command:
    reg_ex = re.search('play (.+)', command)
    if reg_ex:
        searchedSong = reg_ex.group(1)
        url = 'https://www.youtube.com/results?q=' + searchedSong
        try:
            source_code = requests.get(url, headers=headers, timeout=15)
            plain_text = source_code.text
            soup = BeautifulSoup(plain_text, "html.parser")
            songs = soup.findAll('div', {'class': 'yt-lockup-video'})
            song = songs[0].contents[0].contents[0].contents[0]
            hit = song['href']
            webbrowser.open('https://www.youtube.com' + hit)
            MedBOTResponse('Playing ' + searchedSong + ' on Youtube.')
        except Exception as e:

```

```

        webbrowser.open(url)
        MedBOTResponse('Searching for ' +
                        searchedSong + ' on Youtube.')

# Send Email
elif 'email' in command:
    MedBOTResponse('Who is the recipient?')
    recipient = newCommand()
    if 'someone' in recipient:
        MedBOTResponse('What should I say to him?')
        content = newCommand()
        try:
            mail = smtplib.SMTP('smtp.gmail.com', 587)
            mail.ehlo()
            mail.starttls()
            mail.login('sender_email', 'sender_password')
            mail.sendmail('sender_email', 'receiver_email', content)
            mail.close()
            MedBOTResponse(
                'Email has been sent successfully.')
        except Exception as e:
            print(e)
    else:
        MedBOTResponse('I don\'t know anyone named ' + recipient + '.')

# Get current time
elif 'time' in command:
    now = datetime.datetime.now()
    MedBOTResponse('Current time is %d:%d.' %
                    (now.hour, now.minute))

# Get recent news
elif 'news' in command:
    try:

```

```

news_url = "https://news.google.com/news/rss"
Client = urlopen(news_url)
xml_page = Client.read()
Client.close()
soup_page = BeautifulSoup(xml_page, "html.parser")
news_list = soup_page.findAll("item")
for news in news_list[:5]:
    MedBOTResponse(news.title.text)
except Exception as e:
    print(e)

```

```

# Lock the device
elif 'lock' in command:
    try:
        MedBOTResponse("Locking the device.")
        ctypes.windll.user32.LockWorkStation()
    except Exception as e:
        print(str(e))

```

```

# Ask general questions
elif 'tell me about' in command:
    reg_ex = re.search('tell me about (.*)', command)
    try:
        if reg_ex:
            topic = reg_ex.group(1)
            MedBOTResponse(wikipedia.summary(topic, sentences=3))
    except Exception as e:
        MedBOTResponse(e)
elif any(c in command for c in ("what is", "what's")):
    reg_ex = re.search(' (.*)', command)
    try:
        if reg_ex:
            topic = reg_ex.group(1)

```

```

        MedBOTResponse(wikipedia.summary(topic, sentences=2))
    except Exception as e:
        MedBOTResponse(e)
else:
    try:
        # wolframalpha
        client = wolframalpha.Client(app_id)
        res = client.query(command)
        answer = next(res.results).text
        MedBOTResponse(answer)
    except:
        try:
            # wikipedia
            MedBOTResponse(wikipedia.summary(command, sentences=2))
        except Exception as e:
            MedBOTResponse(e)

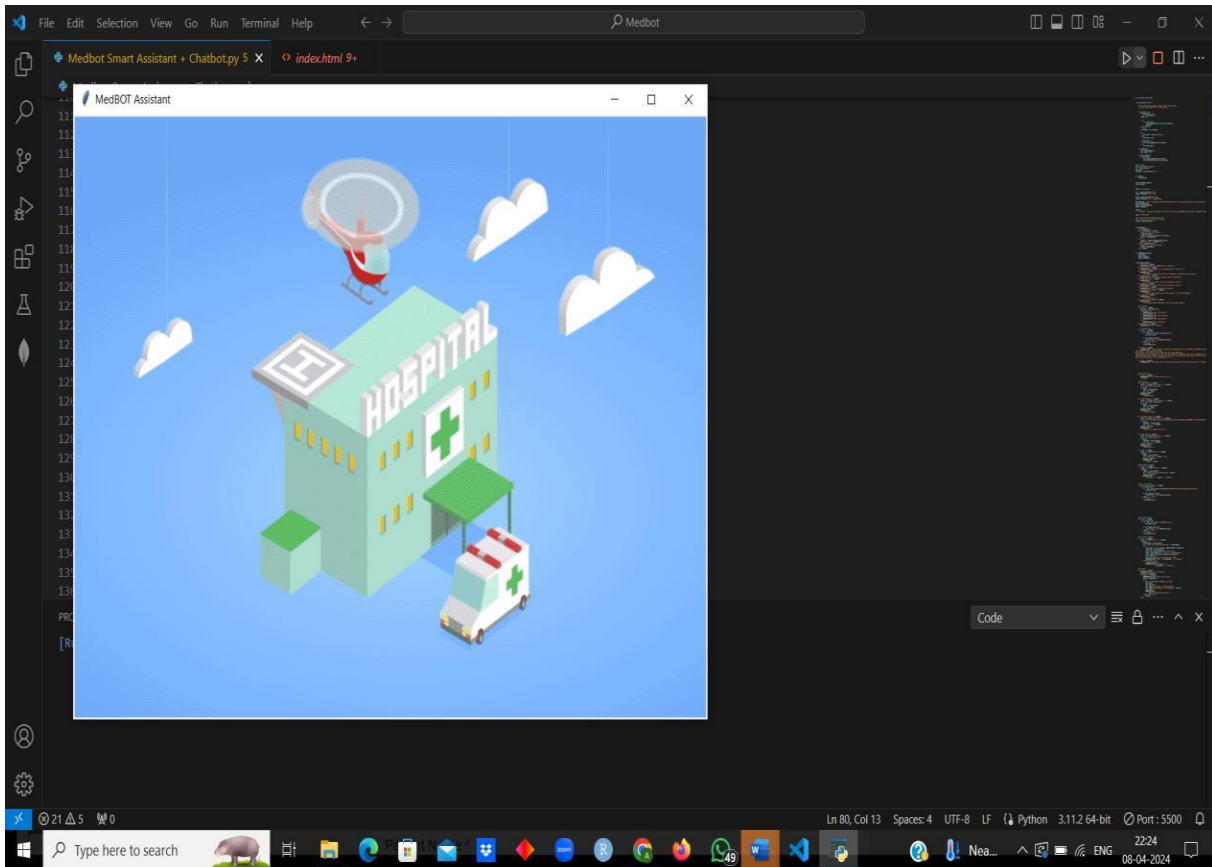
#####

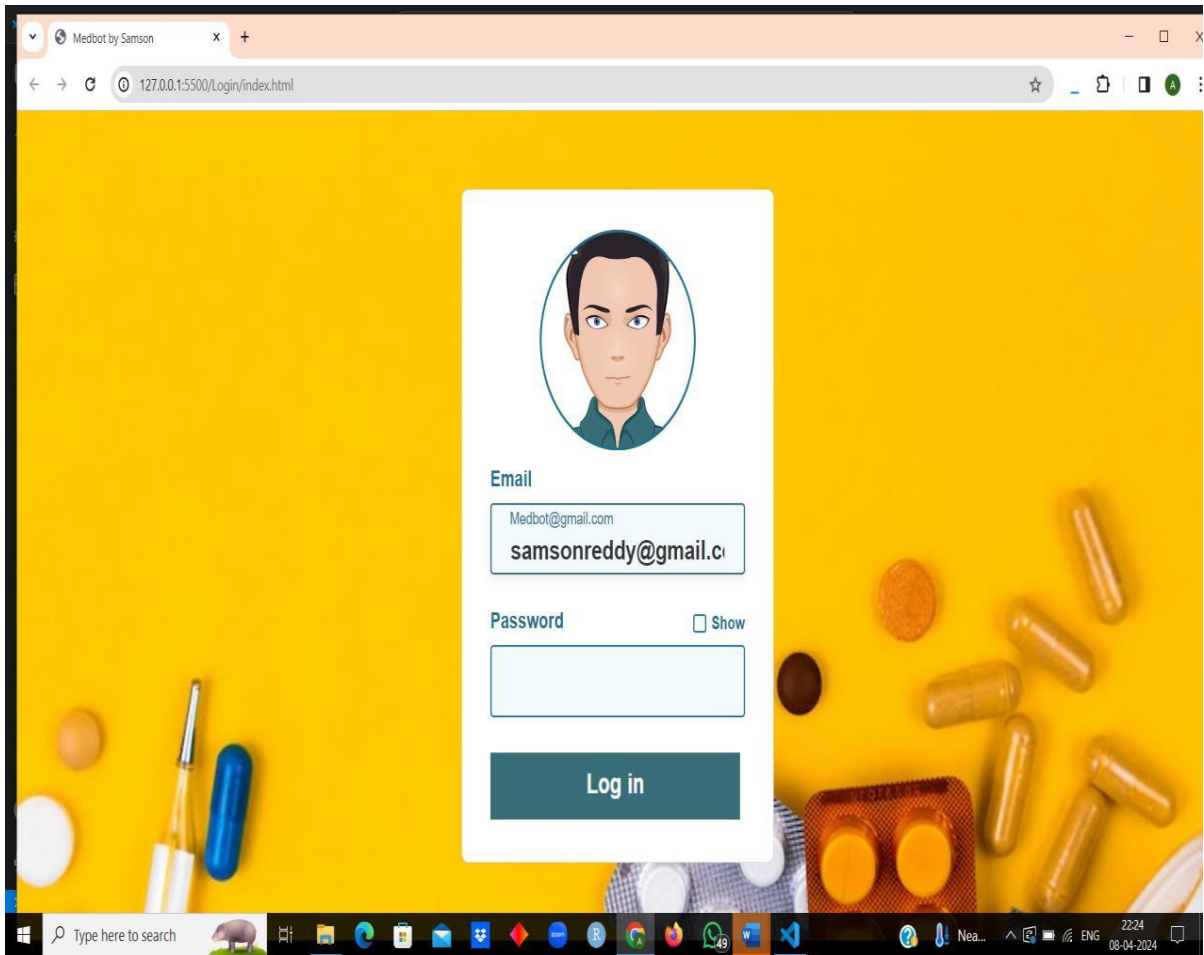
#####

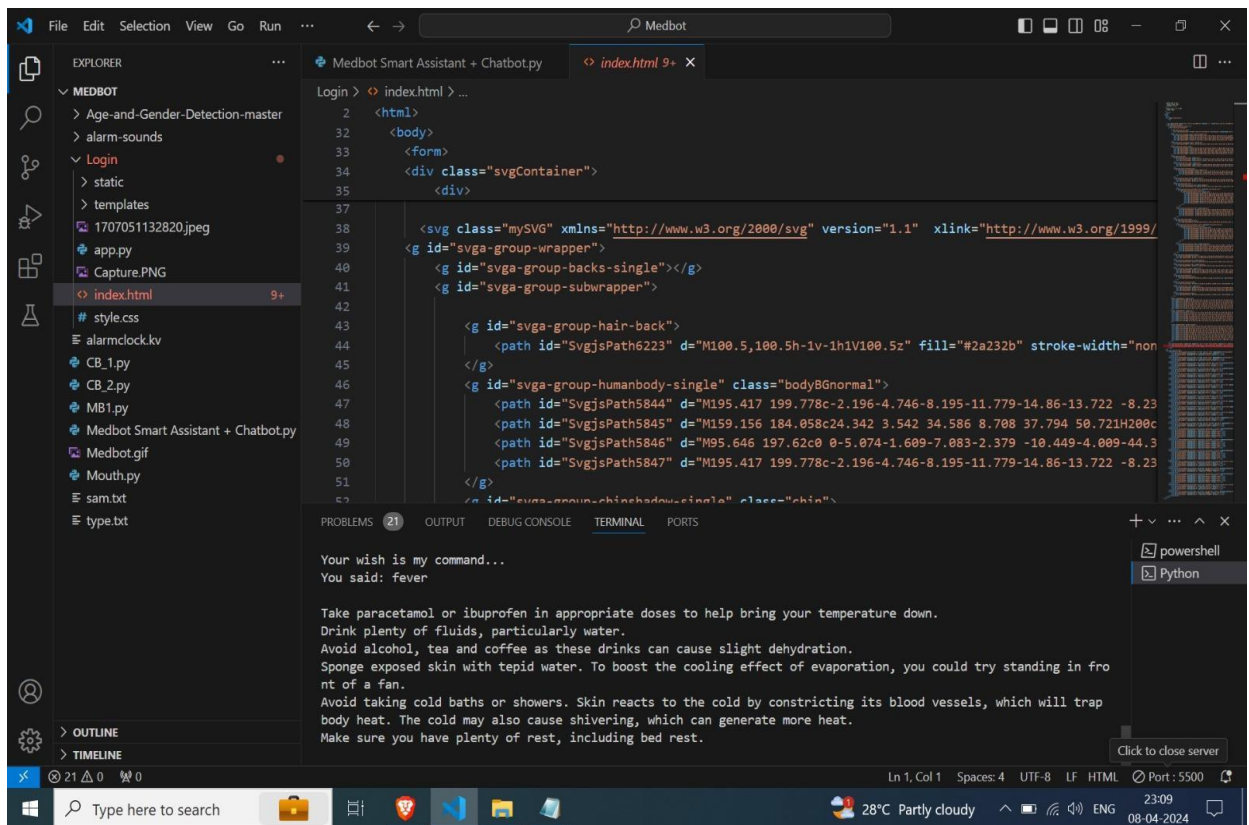
while True:
    assistant(newCommand())

```

SCREENSHOTS







hospitals near me

Results

Good Life Hospital
 4.2 ★★★★★ (63)
 Hospital · No.3/87-A,Rajiv Gandhi Salai(OMR, opp. Sathyabama University)
 Open 24 hours · 073587 63701
 "... Doctor gave me the good treatment and he gave me some ... Good to ..."

Sathyabama General Hospital
 3.9 ★★★★★ (7)
 Hospital · V6CF+75Q
 044 2450 2645

ASTRA Speciality Hospital - Chennai | Sholinganallur | OMR
 4.3 ★★★★★ (241)
 Hospital · Plot No 707, Sri Ganapathy Nagar Colony, Nookampalayam Link Rd
 Open 24 hours
 "The doctor saw me immediately while the nurse and support staff ..."

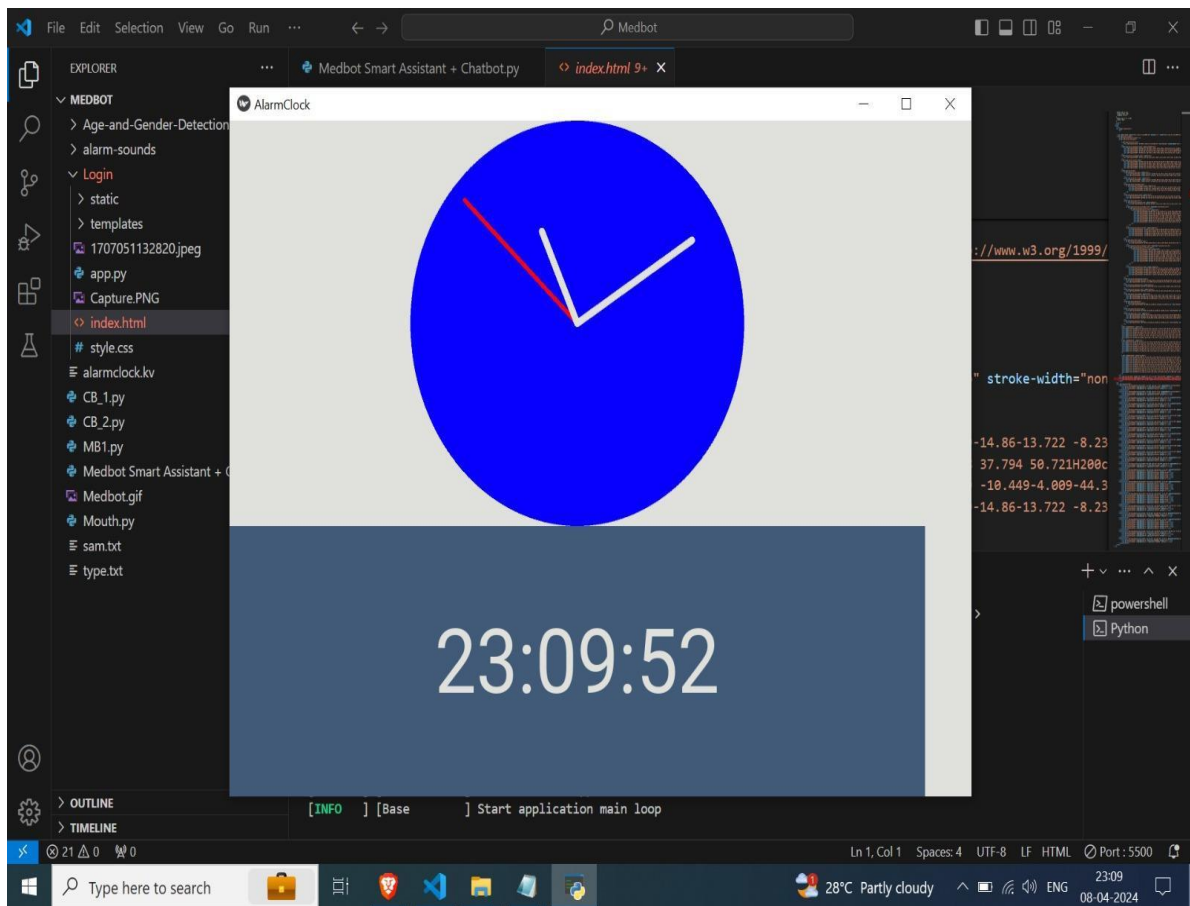
☐ Update results when map moves

Chennai Marine Aquarium
 Gleneagles HealthCity Chennai
 Rainbow Children's Hospital &...
 ASTRA Speciality Hospital - Chennai...
 Semmancheri primary health centre
 Thazhambur தாழ்ம்பூர்
 Sri Isari Velan Mission Hospital/...
 Mambakkam மாம்பாக்கம்
 Ottiambakkam ஒட்டியம்பாக்கம்
 Siruseri சிர்சூரி
 Natham நத்தம்
 SIPCOT IT Park சிப்கொட் தொழில்...
 Egattur ஏகாட்டுர்
 THE MARINA MALL
 Dakshina Chitra Heritage Museum தங்கிணை கிணார்

VGP Universal Kingdom
 Swaram Multispecialty...
 VGP Golden Beach Resort விஜிபி தங்க கடற்கரை...
 Akkarai Beach அகரை கடற்கரை...
 ECR Beach
 Matsya Narayana Temple மதவய நாராயணம்...

Map data ©2024 India Terms Privacy Send Product Feedback 2 km

28°C Partly cloudy 23:08 08-04-2024



Hybrid Machine Learning Methodologies for Using NLP Based Recognition of False Bulletin

Gundampalli samson reddy
Department of CSE
Sathyabama institute of
science and technology

Pasumarthy Sai Sahith
Department of CSE
Sathyabama institute of
science and technology

Gunal S
Department of CSE
Sathyabama institute of
science and technology

Athish R
Department of CSE
Sathyabama institute of
science and technology

Dr. L. Suji Helen, M.E., Ph.d.,
Department of CSE
Sathyabama institute of
science and technology
sujihelen.cse@sathyabama.ac.in

Abstract-Recently, because of the upward thrust of social media, faux news has seemed at the Internet in massive numbers and largely for diverse business and political motives. With false beliefs, social media customers can without difficulty fall prey to this on-line faux news, which has already had a large impact on the offline community. A massive motive for growing the speed with which fake information can be detected is what makes online social networks trustworthy. Theories, approaches, and tactics for identifying fake news, creators, and online social network content are examined in this article. Accuracy of statistics is important at the Internet, in particular on social media, but Internet information hinders the capability to discover, evaluate, appropriately, or in any other case become aware of "fake information" on these sites. In this article we advise strategies for the detection of "faux news" on Facebook, one of the maximum famous on-line social networks. This approach makes use of a hard version of subject type to be expecting whether or not a Facebook submit is assessed as genuine or fake. The results may be stepped forward by using the numerous strategies proposed inside the article. The effects obtained indicate that the hassle of identifying fox records may be solved the use of user-friendly techniques.

Keywords: deep getting to know, social media analysis, neural networks

INTRODUCTION

Recently, due to the growth of social media, misinformation on the Internet has turn out to be famous and tremendous across the world for plenty commercial and political motives. With perverse regulations, social media customers had been hindered in exposing that faux news on the net, which has already had a huge effect on the offline community. The predominant cause of increasing the credibility of social media statistics is to spot fake data more quickly[1]. This article examines ideas, strategies, and

strategies for distinguishing counterfeit names, makers, and content in web-based informal communities. Accusation of statistics is a first-rate task at the Internet, particularly on social media, the fact of the Internet hinders the ability to file what's taken into consideration "incorrect information" accurate in those international locations or in every other context. In this information publish we provide commands on a way to pick out and spot "faux information" on one of the maximum popular online social media systems, Facebook[2]. A traditional Bayesian classification method makes use of a version to expect whether or not a Facebook car may be classified as actual or fake. The outcomes can be advanced the usage of numerous techniques referred to in the article. The results obtained imply that the hassle of detecting fake information may be solved via acquiring the technical expertise to apply computing.

LITERATURE SURVEY

The letter describes various automated techniques for the detection of faux news and fraudulent information. From using chatbots to spread misinformation to using clickbait to spread rumors, detecting faux news is multifaceted. Social networks like Facebook have many buzzers who encourage word of mouth and collect posts that unfold incorrect information. Several tries have been made to perceive the mistake.

1. Media-Rich Fake News Detection:

Fake information has been around for a really long time, and with online entertainment and extreme line reporting at its top, Fox News coverage has turn out to be an important part of the network's investigations. Because of the issues of detecting fake information studies, researchers round the arena are searching out evidence of the elements of confusion. This article objectives to provide an overview of the typological record and the extraordinary styles of record content and its effect on new immigrant readers. Next, we assessment faux news detection strategies based totally on merely textual evaluation and describe the

fake news assets that we've got used substantially. We conclude this article by assisting you pick out 4 open keywords that require specific conditions in the Study Festival.

2. Automatic Detection of Fraud: Strategies for Tracking down Counterfeit News

These questions need to be tested inside the contemporary and replica preliminary books. "Fake news detection" describes the system of figuring out information whose authenticity is understood. Scams are positive to hit a good buy. The nature of the online reality has modified and the flood of content material mills and exceptional substances and genres has made fraud prevention impossible. This article affords a typology of the different forms of unique assessment on account of the mastery of the field: language verbal exchange strategies (with instrumental education) and social evaluation strategies. We see promise in a new hybrid approach that combines language and device studying with graph network[3]. Since detecting faux information isn't a hard mindset, we offer sensible hints for a high-overall performance faux information detection device.

3. Pitifully managed learning for counterfeit news discovery on Twitter.

The trouble of consequently distinguishing counterfeit news on friendly networks, as an example Twitter, is drawing interest these days. Although from a technical factor of view that is considered a easy disturbance of the binary class, the main project is to accumulate sufficiently basic, manually identifying spurious or false tweets is expensive and tough. In this text, we talk tries at a less predictable technique to amassing education facts from many large, however noisy, tweets. As we collect, we regularly label tweets based on their source, i.e. Reliable or unreliable supply, and create a taxonomy based totally on this facts[4]. We use this class for different proposed classifications to digest fake and fake tweets. Although we've detected labels in the new class target (not all tweets from unknown resources are actually fake news and vice versa), it's far very ambiguous and reaches faux news with 0 F1 rating. . Nine. The dataset is diagnosed to us.

4. Social Media Detection of Fake News

Misinformation and fraud have long plagued the net. A widely used definition of faux information on the Internet is: "records made up deliberately to deceive readers." Social media and media bombard readers with fake information or as a part of psychological struggle. It is set to yield a useful click on. Clickbait attracts users and generates interest with the aid of weathering site visitors or link clicks to increase sales. They display the growth of Fox News to the regression of communicative characteristics made feasible by the rise of long-range informal communication destinations. The objective of the endeavor is to make

an answer that permits users to discover and block internet traffic that consists of fake and fraudulent records. We use simple, accurate facts and modifying talents to properly perceive faux posts. The check consequences using logistic classification display 99.4% accuracy[5].

5. Combining Content and Social Signals for Automatic Online Fake News Detection

The rapid spread of false information on the Internet highlights the need for automated fraud detection systems. in relation to social media, the device may be used to analyze more approximately (ML) techniques. Methods for detecting faux news have traditionally handiest been primarily based on content assessment (i.e. Thru content material segmentation information) or - more lately - strategies associated with social context, along with pattern mapping and truth diffusion. In this paper, we first advocate a brand new technique to discover fake news in ML, which builds on current methods inside the literature, combines real-time data and social context capabilities, and quadruples their accuracy. % Second, we validate our method on a actual Facebook message chatbot with actual-global utility and acquire a fake message detection accuracy of 80-one.7%.

6. Some consider it a hoax: Social media platforms' automated detection of fake news

Recently, the credibility of net facts has emerged as a primary trouble in modern-day society. By letting users freely share content, social networking sites (SNS) have revolutionized the dissemination of information. material. That is why the usage of social media is a vehicle for spreading false and fake records. Due to the variety of instances and the rate of mileage propagation, speedy credibility evaluation is almost not possible, requiring automatic fraud detection systems[6].

To this quit, Facebook can contribute to faux or very erroneous posts, especially within the diverse person options. We present two styles of strategies, one in light of calculated relapse and the other explicitly founded on a fresh out of the box new model of Boolean recurrence calculations. We used a dataset of 9,092,236 Facebook customers and 15,500 Facebook messages to will reap over ninety nine% class accuracy even with less than 1% of messages. But we have demonstrated our strong talents: they work even with clients who locate false and deceptive records[7]. These results recommend that dissemination of facts in statistical shape is a useful element in automated fraud detection frameworks.

7. The spread of phony news by friendly bots

Fox News has been diagnosed via the mass media as a essential worldwide hazard and has been accused of threatening elections and democracy. Experts in verbal, cognitive, and social generation are tirelessly

studying the complicated causes of the unfold of digital misinformation and viral responses, as surveys and social media begin to measure it. However, this work was primarily based totally on anecdotal proof in place of formal statistics. This is America in 2016[8]. We estimate there had been 24 million tweets and 400,000 Twitter proceedings at some point of and after the presidential marketing campaign and election. We have proof that smaller social organizations play an essential role in disseminating incorrect information. Bills that actively unfold incorrect information can be bots. Automatic invocation is particularly beneficial in early and influential goal uses. People fall prey to this manipulation through retweeting bots that unfold faux information. It is primarily based on guide for the fake and obvious social blessings of the car. These consequences advise that blocking off social agonists is an effective means of lowering the spread of on-line incorrect information[9].

8. Misleading online content: Recognizing click bait as false news

Tabular newspapers are frequently criticized for exaggeration, sensationalism, terrorist reporting, and false and defamatory publications. As information proliferates on line, a new shape of tabloidization has emerged: "click bait." "Clickbait" consists of content material that "creates use and encourages visitors to click on on a hyperlink on a specific web page" [clickbait, n.D.] and contributes to the rapid spread of rumors and misinformation on-line. This article examines feasible techniques for detecting click fraud on a user base. Both educational and non-academic strategies of exploring noise are catching on, in particular hybrid strategies yielding the first-rate results.

9. Deep learning applications and challenges in big data analytics

Big information analytics and deep gaining knowledge of are elite regions of statistics science. Big facts is vital due to the fact many public and personal agencies accumulate large quantities of private records that can offer useful insights into national troubles, surveillance, cyber protection, fraud detection, advertising, and medical statistics. Companies like Google and Microsoft examine massive amounts of information that pressure business analytics and selection-making, generation and the future. Deep getting to know algorithms collect multiple excessive-stage representations of facts thru a hierarchical getting to know procedure. Complex abstractions are found out at a given level primarily based on simple abstractions related to the preceding ones inside the hierarchy. One of the primary blessings of deep studying is the capacity to investigate and analyze from a large quantity of unobserved facts, making it a tool for high-degree analytics. In the existing have a

look at, we check out how deep getting to know can be used to clear up some huge-scale analytical troubles, which includes extracting complicated patterns from massive volumes of facts, or simplifying semantic codes, descriptive statistics, fast statistics retrieval, and discrete capabilities. . . We also explore a few aspects of deep gaining knowledge of research that require in addition take a look at to include unique demanding situations posed by using huge scale analysis, consisting of streaming information, excessive dimensional records, scaling models, and allotted computing. We finish by using addressing a few issues including defining norms consisting of deforming norms, designing diversifications, defining abstract characteristic extraction norms, optimizing semantic maps, gaining knowledge of semi-enumeration, and energetic gaining knowledge of, as a result imparting a glimpse of future paintings.

10. Definitional Challenges of Fake News

This article explores the query of the way "fake news" and "disinformation" are described and how this studies impacts how we learn and fight fake or deceptive facts on line. This article identifies a conscious tendency to withstand political rhetoric and misinformation, this is, to intentionally unfold misinformation and ignore the trouble of incorrect information (random incorrect information). In assessment, it's miles hard to distinguish among false and misleading statistics in educational studies. Thus, in a way, it overcomes the trouble of objectivity, in prefer of facts referred to as genuine/false dichotomies. 2 The awkward conditions created by way of this gap between the consequences of instructional studies and the principles of extrapolation prevent our ability to successfully interact in simulation. Disadvantages and incorrect information.

EXISTING SYSTEM

There is tons within the way of technical investigations to detect fraud, maximum of which contain on-line testing and public social networks. The problem of detecting "faux information" acquired particular interest within the literature after the 2016 wreck, in particular at some point of the United States presidential election.

Conroy, Rubin, and Chen included several tactics to know-how fraud stories. Understand the contents of the simple correspondence n-p. Superficial fragments are inadequate to provide an explanation for the discourse categorization paradigm, frequently ignoring crucial contextual information. However, those techniques were tested for effectiveness with more complex analytical techniques. Deep evaluation using context-agnostic probabilistic grammars combined with N-gram techniques proved maximum useful[10]. Feng, Banerjee and Choi were able to

achieve 85-91% accuracy in fraud class duties the usage of on line structure estimation.

Disadvantages of Existing System

Each is categorized fake or fraudulent. The authors talk the pros and cons of different sorts of fake information and use numerous text evaluation and predictive detection techniques.

1. It is difficult to gather information that are not published within the yellow press or tabloids, where your important fiction is seen in the media or media.

2. Scales are innovative and pretty accurate and seem on many structures. The authors argue that strategies other than textual evaluation must discover these duplicate facts.

Tribe. Fake information can be humorous, suggestive and funny via its authors. According to the authors, this kind of fake news pastime has a devastating impact on the overall technical competencies of the iTech-magnificence.

PROPOSED SYSTEM:

In this text, a version is developed based on the wide variety of sufferers or matrix tfidf (ie) phrases for the frequency of use of other articles for your dataset. Since this hassle provides a form of text classification, it is satisfactory to put into effect a easy classifier base as it is a wellknown textual content-based procedure. The real goal is to create a text transformation version (variety vectorizer vs tfidf vectorizer) and select the type of text to apply (caps vs complete text). The next step is to extract the features maximum relevant to the remember vector or tfidf vector, which is achieved via disposing of n range of several words and/or terms, lowercase or now not, frequently empty words. Use common words together with "when," "while," and "there" and most effective phrases that appear a sure wide variety of instances inside the given textual content.

Advantages of Proposed System

- Improved accuracy
- Adaptability
- Scalability

PROPOSED ALGORITHM

1. Naive Bayes-(NB)

- one of the probabilistic classification-based supervised learning algorithms.
- It is a strong and quick calculation for prescient displaying.
- In this venture, I have utilized the Multinomial Credulous Bayes Classifier.

2. Support Vector Machine-(SVM)

- SVM's are a bunch of directed learning techniques utilized for arrangement, and relapse.
- Compelling in high layered spaces. Makes use of a subset of the training points in the support vector, making it also efficient with memory.

3. Logistic Regression-(LR)

- Direct model for order as opposed to relapse.
- The normal upsides of the reaction variable are displayed in view of blend of values taken by the indicators

MODULES

1. Information Use
2. Preprocessing
3. Include Extraction
4. Classifier Training

Modules description

1. Information Use:

So on this assignment we will examine pandas to load and shop special files and information. We examine. Csv files the use of pandas, examine the table structure and show the statistics in the suitable format. We do it and show it that is. Use the label to alternate the dominance. We can run an uncommon device that tests and reads algorithms via the facts and labels of the school, however earlier than making predictions and precisions, it is necessary earlier than the statistical method, this is, to take away all of the unknown statistics of the table and to repair them as vector maps. I will lessen it in order. This may be understood thru devices and attention. The subsequent step is to apply those statistics to attract visible evaluations using Python and the Sigit Learn tab engineering library. This library permits you to plan the effects inside the form of histograms, pie charts or graphs.

2. Preprocessing

The datasets used were optional assets and dataset 1, which contained three, 256 compliance and 814 manipulated statistics, and dataset 2, which contained 1,882 establishments and 471 manipulated mutants. There is a window in the the front. With these expressions he become a date. This allows powerful modifications to be made. Sometimes we acquire messages through emails, letters, catchphrases, numbers and many others. Boredom makes it hard to find mail. It enables get rid of language-specific textual factors and has logic that increases the accuracy of popular reasoning.

3. Information Use

Feature extraction is the manner of selecting a subset of these features to generate an output. The function extraction approach enables construct a better predictive model. They help choose techniques that supply precise effects. When the input facts are too huge, intractable and redundant, the enter facts are converted into reduced sorts of descriptors, additionally called characteristic vectors. To complete the operation, the enter is decreased to its complete length so that the enter objects are saved. Feature extraction is accomplished on the source fact degree before running a managed set of policies to transform the records into the workspace.

4. Classifier Training

In this assignment, I will use the scikit-learn library to implement the structure. Learn Scikit is an open supply Python getting to know tool furnished via the Anaconda distribution. It requires a report and can be quick set up when you write the order. If the command isn't executed, errors can arise on the identical time. We use 4 unusual algorithms and teach these four models, specifically simple basis set, vector device, nearest neighbor and logistic regression, which are famous techniques for log kind troubles. When the classifiers are trained, we have a look at the education strategies on the check set. The sentence vector in each letter can be extracted in a take a look at set and its type can be detected by way of trained methods.

PROPOSED ALGORITHM

Naïve Bayes- the Naive Bayes classifier is a supervised system mastering set of rules used for classification tasks along with textual content class. It is part of generative learning algorithms, They try to imitate the distribution of enter information of a particular kind or kind. It is referred to as naive as it assumes that each input variable is impartial. This is a robust assumption and inconsistent with real information; However, this technique may be very beneficial in solving a wide variety of complicated issues.

RESULT AND DISCUSSION

Calculation's precision relies upon the kind and size of your dataset. More the information, more possibilities getting right precision. AI relies upon the varieties and relations Understanding what is unsurprising is essentially as significant as attempting to foresee it. While pursuing calculation decision, speed ought to be a thought factor is shown in Table 1 and Figure 3.

Table 1 Proposed Accuracy

Utilizing All Features				
Algorithm	Accuracy	Precision	Recall	F-Score
Naives Bayes	95.56	96.39	94.85	95.65
SVM	90.95	91.27	90.45	91.17
Utilizing Beast Features				
Naives Bayes	99.98	99.98	99.98	99.98
SVM	98	99	98	98

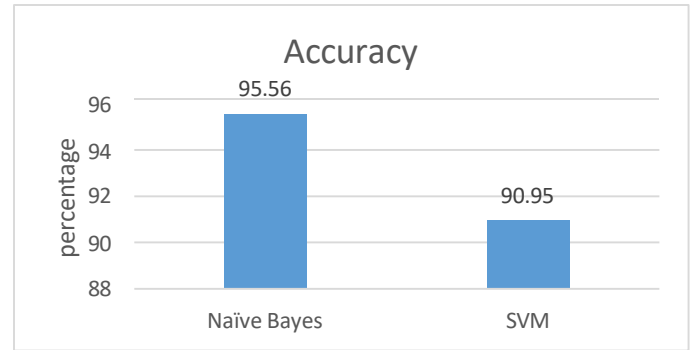


Fig 3 Proposed algorithm

CONCLUSION:

Many human beings eat news on social media rather than conventional information media. However, people also use social media to spread false information, which hurts people and society. In this text, we gift another variant for distinguishing counterfeit news through framework learning Concentrate on procedures are given. The news form accepts the outcomes as information and basically founded on Twitter sentiments and characterization calculations, it gauges the portion of phony or genuine information.

REFERENCES

- [1]. Parikh, S. B., & Atrey, P. K. (2018, April). Media-Rich Fake News Detection: A Survey. In 2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR) (pp. 436-441). IEEE.
- [2]. Conroy, N. J., Rubin, V. L., & Chen, Y. (2015, November). Automatic deception detection: Methods for finding fake news. In Proceedings of the 78th ASIS&T Annual Meeting: Information Science with Impact: Research in and for the Community (p. 82). American Society for Information Science.
- [3]. Helmstetter, S., & Paulheim, H. (2018, August). Weakly supervised learning for fake news detection on Twitter. In 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM) (pp. 274-277). IEEE.
- [4]. Stahl, K. (2018). Fake News Detection in Social Media.
- [5]. Della Vedova, M. L., Tacchini, E., Moret, S., Ballarin, G., DiPierro, M., & de Alfaro, L. (2018, May). Automatic Online Fake News Detection Combining Content and Social Signals. In 2018 22nd Conference of Open Innovations Association (FRUCT) (pp. 272-279). IEEE.
- [6]. Tacchini, E., Ballarin, G., Della Vedova, M. L., Moret, S., & de Alfaro, L. (2017). Some like it hoax: Automated fake news detection in social networks. arXiv preprint arXiv:1704.07506.
- [7]. Shao, C., Ciampaglia, G. L., Varol, O., Flammini, A., & Menczer, F. (2017). The spread of fake news by social bots. arXiv preprint arXiv:1707.07592, 96-104.

- [8]. Chen, Y., Conroy, N. J., & Rubin, V. L. (2015, November). Misleading online content: Recognizing clickbait as false news. In Proceedings of the 2015 ACM on Workshop on Multimodal Deception Detection (pp. 15-19). ACM.
- [9]. Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R., & Muharemagic, E. (2015). Deep learning applications and challenges in big data analytics. *Journal of Big Data*, 2(1), 1.
- [10]. Haiden, L., & Althuis, J. (2018). The Definitional Challenges of Fake News.

Dear Author(s),

Greetings from ICSTEM24.

International Conference on Science Technology Engineering and Management ICSTEM24.

Organized by KIT_Kalaignarkaranidhi Institute of Technology, Coimbatore, Tamilnadu, India.

We are delighted to inform you that your paper titled "**Hybrid Machine Learning Methodologies for Using NLP Based Recognition of False Bulletin**" with Paper-ID: **IEEE_ICSTEM24_0732** has undergone a thorough review by the Technical Committee. It has been recommended for presentation at the conference. You will find the suggestions and comments provided by the Technical Committee listed at the end of this acceptance notification. Congratulations!

The Last date for Registration and sending camera ready paper is 13-04-2024.

Kindly make all the necessary corrections as per their remarks before submitting camera ready paper. This is mandatory to include your paper in the conference proceedings. Formatting your paper with the below procedures and **keeping your paper without plagiarism, will increase the chance of publishing the paper in IEEE Digital Library publications.**

The "camera-ready" PDF paper can be created with IEEE PDF Express (**Conference-ID: 61137X**), (<https://ieee-pdf-express.org/>). Kindly indicate your paper ID all the time in the **subject** line whenever you send mail to us.



Transaction Successful

10 April 2024 at 9:44 AM

Transaction ID

T2404100944349670076380

[COPY](#)

Paid to



XXXXXXSTEM 
Indian Bank - *****9290

₹ 7,000

Debited from



XXXXXX1658
UTR: 410176848775

₹ 7,000