A

Major Project

On

DISEASE DIAGNOSIS USING CHATBOT

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

K.AKSHITH REDDY	(217R1A0532)
P.DOLIKA	(227R5A0503)
N.SUNIL	(217R1A0540)

Under the Guidance of

Mrs.NAJEEMA AFRIN

(Assistant Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi)

Recognized Under Section 2(f) & 12(B) of the UGCAct.1956,

Kandlakoya (V), Medchal Road, Hyderabad-501401.

April, 2025.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled "**DISEASE DIAGNOSIS USING CHATBOT**" being submitted by **K. AKSHITH REDDY (217R1A0532) , P.DOLIKA (227R5A0503), N.SUNIL(217R1A0540)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, during the year 2024-25.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Mrs.Najeema Afrin
Assistant Professor
INTERNAL GUIDE

Dr. Nuthanakanti Bhaskar
HoD

Dr. A. Raji Reddy
DIRECTOR

Signature of External Examiner

Submitted for viva voice Examination held on ______________________

ACKNOWLEDGEMENT

We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project, we take this opportunity to express our profound gratitude and deep regard to our guide **Mrs.Najeema Afrin,** Assistant Professor for her exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by her shall carry us a long way in the journey of life on which we are about to embark.

We take this opportunity to extend our heartfelt appreciation to the Project Review Committee (PRC) Coordinators—**Dr. K. Maheswari, Dr. J. Narasimharao, Ms. K. Shilpa, and Mr. K. Ranjith Reddy**—for their unwavering support, insightful guidance, and valuable inputs, which played a crucial role in steering this project through its various stages.

Our sincere appreciation also goes to **Dr. Nuthanakanti Bhaskar**, Head, for his encouragement and continuous support in ensuring the successful completion of our project.

We are deeply grateful to **Dr. A. Raji Reddy**, Director, for his cooperation throughout the course of this project. Additionally, we extend our profound gratitude to Sri. **Ch. Gopal Reddy**, Chairman, Smt. **C. Vasantha Latha**, Secretary and Sri. **C. Abhinav Reddy**, Vice-Chairman, for fostering an excellent infrastructure and a conducive learning environment that greatly contributed to our progress.

We also acknowledge and appreciate the guidance and assistance provided by the faculty and staff of **CMR Technical Campus**, whose contributions have been invaluable in bringing this project to fruition.

Lastly, we sincerely thank our families for their unwavering support and encouragement. We also extend our gratitude to the teaching and non-teaching staff of CMR Technical Campus for their guidance and assistance. Their contributions, along with the support of everyone who helped directly or indirectly, have been invaluable in the successful completion of this project.

K. Akshith Reddy (217R1A0532)
P. Dolika (227R5A0503)
N. Sunil(217R1A0540)

VISION AND MISSION

INSTITUTE VISION:

To Impart quality education in serene atmosphere thus strive for excellence in Technology and Research.

INSTITUTE MISSION:

1. To create state of art facilities for effective Teaching- Learning Process.
2. Pursue and Disseminate Knowledge based research to meet the needs of Industry & Society.
3. Infuse Professional, Ethical and Societal values among Learning Community.

DEPARTMENT VISION:

To provide quality education and a conducive learning environment in computer engineering that foster critical thinking, creativity, and practical problem-solving skills.

DEPARTMENT MISSION:

1. To educate the students in fundamental principles of computing and induce the skills needed to solve practical problems.
2. To provide State-of-the-art computing laboratory facilities to promote industry institute interaction to enhance student’s practical knowledge.
3. To inculcate self-learning abilities, team spirit, and professional ethics among the students to serve society.

ABSTRACT

This project is titled as "Disease Diagnosis Using Chatbot". This project explores the application of chatbot technology for disease diagnosis, aiming to provide accessible, efficient, and accurate preliminary health assessments. By leveraging natural language processing (NLP) and machine learning algorithms, the chatbot can interact with users, gather symptom information, and suggest possible diagnoses based on its knowledge base. The chatbot is designed in a conversational format, simulating an interaction similar to that with a healthcare professional. It then analyzes the user's responses, compares symptoms with known disease profiles, and provides a list of potential conditions and suggestions for further actions, such as consulting a medical specialist. This system is particularly useful in regions with limited healthcare access, offering immediate guidance for non-critical symptoms and helping reduce the burden on medical facilities by directing patients more effectively. Overall, the chatbot-based disease diagnosis system represents a step towards more accessible healthcare, utilizing AI to assist in early-stage medical assessment.

LIST OF FIGURES

LIST OF TABLES

TABLE OF CONTENTS

1. INTRODUCTION

1. INTRODUCTION

The project, titled "Disease Diagnosis Using Chatbot". Access to Access to timely and accurate healthcare information plays a crucial role in early disease detection and effective management. However, traditional self-diagnosis methods often lack personalized guidance, leading to misinformation or delays in seeking proper medical care. This project introduces an AI-powered chatbot designed to provide an interactive and efficient solution for users to describe their symptoms and receive reliable health insights. By leveraging natural language processing (NLP),ML Algorithms the chatbot processes user inputs and cross-references them with a comprehensive disease database to identify potential health conditions.

The chatbot goes beyond basic symptom analysis by offering personalized recommendations based on the identified conditions. It suggests suitable dietary adjustments to support better health management and provides details of nearby doctors for further consultation. Additionally, an emergency response feature is integrated to recognize severe symptoms and offer immediate guidance for urgent medical attention. Privacy and security are prioritized, ensuring that user-provided health data remains confidential and adheres to ethical standards and healthcare regulations.

By implementing this chatbot, the project aims to bridge the gap in self-diagnosis and healthcare accessibility. It reduces dependency on inaccurate self-assessments while offering timely, informative, and secure medical guidance. With its ability to provide symptom-based recommendations, dietary suggestions, and doctor details, this system enhances user engagement and contributes to improved health awareness and management. Ultimately, this AI-powered solution ensures that individuals receive accurate healthcare information, facilitating better decision-making and promoting overall well-being.

1.1 PROJECT PURPOSE

The primary purpose of this project is to enhance healthcare accessibility by providing an AI-powered chatbot for symptom analysis, medical guidance, and personalized recommendations. Traditional self-diagnosis methods often lead to misinformation or delays in treatment. By leveraging natural language processing (NLP), ML Algorithms, the chatbot processes user-described symptoms, matches them with a disease database, and offers relevant health insights.

Beyond symptom analysis, the chatbot suggests suitable food recommendations based on health conditions and provides details of nearby doctors for further consultation. It also includes emergency response guidance while ensuring strict data privacy. By reducing reliance on inaccurate self-diagnosis, this system promotes better health management and improves access to reliable medical resources.

1.2 PROJECT FEATURES

This project incorporates several key features to improve the accuracy in finding various diseases.

This project introduces an AI-powered chatbot designed to assist users in identifying potential health conditions based on their symptoms. By leveraging natural language processing (NLP), Convolutional Neural Networks (CNN), and K-Nearest Neighbors (KNN), the chatbot efficiently processes user inputs and matches them with a comprehensive disease database. CNN helps extract patterns from textual data, while KNN enhances classification accuracy by comparing user symptoms with similar cases. It provides personalized health recommendations, including dietary suggestions and lifestyle modifications, to help users manage their well-being. Additionally, the chatbot offers details of nearby doctors and medical professionals, ensuring that users receive timely medical guidance and consultation.

An emergency response feature is integrated to detect severe symptoms and provide immediate guidance on seeking urgent medical attention. The chatbot prioritizes user privacy by implementing strict data security measures that comply with ethical healthcare standards. With its user-friendly interface, the system ensures seamless interaction, making healthcare information easily accessible. By incorporating AI-driven analysis, real-time recommendations, and multiple machine learning algorithms, this chatbot enhances self-diagnosis accuracy, reduces dependency on unreliable sources, and promotes better health awareness and decision-making.

2. LITERATURE SURVEY

2.LITERATURE SURVEY

Chatbot-based disease diagnosis is an evolving field that integrates artificial intelligence, natural language processing (NLP), and machine learning to assist users in identifying potential health conditions. By providing a conversational interface, these chatbots enable users to input symptoms and receive preliminary health assessments. Various studies and literature surveys have explored different aspects of chatbot applications in healthcare, focusing on their efficiency, accuracy, user trust, ethical concerns, and technological advancements.

One of the significant studies in this domain is Sarah E. Williams' "Chatbots in Healthcare: A Comprehensive Review of Applications and Challenges." Williams provides an extensive analysis of how chatbots are being used in medical diagnosis and healthcare management. The survey highlights that chatbots improve accessibility to healthcare by offering users quick and preliminary diagnoses based on symptom descriptions. The research also discusses the challenges in implementing chatbots, such as accuracy issues and the need for continuous learning to enhance reliability. The study underscores the importance of integrating these chatbots with medical databases and healthcare professionals to improve their efficacy.

Another essential study is Michael J. Davis' "AI-Driven Chatbots for Medical Diagnosis: State-of-the-Art Approaches." Davis explores how artificial intelligence, particularly deep learning techniques like Convolutional Neural Networks (CNN) and Support Vector Machines (SVM), are applied in chatbot-based medical diagnosis. The research outlines how these AI-driven models enable chatbots to make more accurate predictions and learn from patient interactions. It also emphasizes the need for a structured approach to designing these models, as errors in chatbot diagnosis could lead to misinformation and potential health risks. The study highlights the potential of integrating chatbots with electronic health records (EHRs) to enhance personalized healthcare.

Ethical considerations in disease diagnosis chatbots are discussed extensively in Emily R. Martinez's research titled "Ethical Considerations in Disease Diagnosis Chatbots: Insights and Recommendations." Martinez examines concerns related to data privacy, patient confidentiality, and the risk of chatbot misdiagnosis. The study emphasizes that while chatbots offer significant advantages in remote healthcare access, they must adhere to strict data protection regulations, such as the Health Insurance Portability and Accountability Act (HIPAA) in the United States and the General Data Protection Regulation (GDPR) in Europe. The research suggests that chatbot developers should incorporate transparent mechanisms for users to understand how their data is being used and stored. Furthermore, the study highlights the importance of ensuring that chatbots do not replace professional medical consultations but rather serve as an additional tool to guide patients.

David A. Thompson, in his survey "Patient Trust in Chatbot-Assisted Disease Diagnosis: A Review," explores the psychological and trust-related factors influencing the acceptance of chatbot-based healthcare solutions. The research indicates that user trust in chatbots depends on several factors, including accuracy, transparency, and the ability to provide a human-like conversational experience. The study suggests that while chatbots can be efficient in initial consultations, their perceived credibility is often lower than that of human doctors. The findings emphasize the need to design chatbot interactions in a way that reassures users about the reliability of the diagnosis and encourages them to seek further medical advice when necessary.

Another notable contribution to this field is Jessica L. Turner’s study, "Human-Chatbot Interaction in Medical Diagnosis: Current Trends and Future Directions." Turner analyzes how user experience can be enhanced to ensure effective interactions between patients and chatbots. The research discusses the importance of designing chatbot interfaces that are user-friendly, linguistically adaptable, and capable of understanding complex symptom descriptions. The study suggests that future developments in chatbot-based diagnosis should focus on incorporating voice recognition, multilingual support, and integration with wearable health devices to collect real-time health data.

Despite the advancements in chatbot-driven disease diagnosis, certain limitations persist. The accuracy of chatbot-based diagnoses depends heavily on the quality and comprehensiveness of the medical databases they rely on. While machine learning models can improve over time through data-driven learning, there is always a risk of misinterpretation of symptoms, leading to incorrect diagnoses. Additionally, chatbots are still limited in their ability to handle medical emergencies, as they lack the capacity for physical examinations or real-time medical testing.

Future research in this domain is likely to focus on improving chatbot accuracy through the integration of real-time patient data from wearable health monitoring devices. Additionally, researchers are exploring the possibility of hybrid models where chatbots assist doctors rather than replacing them, ensuring that medical professionals validate chatbot-generated diagnoses before providing medical recommendations. The use of advanced NLP techniques, such as transformer-based models like GPT, can further enhance chatbot understanding of medical queries and improve response quality.

In conclusion, the literature on chatbot-based disease diagnosis provides valuable insights into the potential benefits, challenges, and future directions of this technology. Chatbots offer an efficient and accessible way for individuals to obtain preliminary health information, reducing the burden on healthcare facilities and improving patient engagement. However, ensuring data security, enhancing accuracy, and increasing user trust remain critical areas for improvement. As AI and machine learning continue to evolve, chatbot-based medical diagnosis systems are expected to become more sophisticated and reliable, ultimately transforming the way people access healthcare information.

2.1 REVIEW OF RELATED WORK

The application of chatbots in disease diagnosis has been extensively explored in the fields of artificial intelligence (AI), natural language processing (NLP), and deep learning. Various studies have focused on enhancing chatbot-based medical diagnosis using machine learning (ML) and deep learning techniques to improve accuracy, scalability, and user interaction. This section reviews existing research, highlighting methodologies, strengths, and limitations in comparison to the proposed system, which utilizes K-Nearest Neighbors (KNN), Convolutional Neural Networks (CNN), and NLP for text-based disease prediction.

1. Traditional Rule-Based Symptom Checkers

Early chatbot-based healthcare systems relied on rule-based approaches, where predefined symptom-disease mappings were used to provide diagnostic suggestions. These systems followed structured decision trees and if-else rules to match user-reported symptoms with a static medical knowledge base. While rule-based chatbots were simple and interpretable, they lacked adaptability and struggled with complex user inputs. Their inability to learn from new data made them less effective for real-world applications, where symptom descriptions can vary significantly.

2. Machine Learning-Based Approaches

To overcome the limitations of rule-based systems, machine learning algorithms such as Support Vector Machines (SVM), Decision Trees, Naïve Bayes, and K-Nearest Neighbors (KNN) have been widely used in disease diagnosis. KNN, in particular, has been effective in classifying diseases based on similarity, as it compares new symptom inputs with past cases. Studies have shown that KNN performs well for structured symptom data, offering reliable predictions when trained on large datasets. SVM and Decision Trees have also demonstrated strong performance by identifying patterns in medical records. However, traditional ML models require feature engineering, which can be challenging for unstructured text-based inputs.

3. Deep Learning-Based Approaches with CNN and NLP

With advancements in deep learning, Convolutional Neural Networks (CNN) have been successfully applied to text-based medical diagnosis. Unlike its traditional use in image processing, CNN for text analysis helps in learning meaningful feature representations from user symptom descriptions. By applying 1D convolutions over text sequences, CNNs can identify important n-gram features, making them effective for symptom classification and disease prediction.

Additionally, Natural Language Processing (NLP) has been widely used to enhance chatbot interactions. Techniques such as tokenization, named entity recognition (NER), and word embeddings (e.g., Word2Vec, GloVe) allow chatbots to process user inputs more effectively. Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks have also been used alongside CNNs to improve sequential text understanding, capturing dependencies between symptoms in longer sentences.

4. Comparison with the Proposed Approach

While previous research has made significant advancements in chatbot-assisted disease diagnosis, challenges remain in improving accuracy, real-time performance, and adaptability to diverse user inputs. The proposed system builds upon prior studies by integrating KNN for quick symptom classification, CNN for feature extraction from text-based symptoms, and NLP for enhanced chatbot interactions. This approach ensures higher accuracy and adaptability, allowing users to describe their symptoms naturally while receiving precise health assessments.

This review highlights the progression from rule-based and traditional ML models to deep learning-driven chatbot diagnosis systems. By incorporating KNN, CNN, and NLP, the proposed system aims to provide an efficient, scalable, and highly accurate chatbot for preliminary disease diagnosis, improving accessibility to healthcare services.

2.2 EXISTING SYSTEM

The existing system for disease diagnosis primarily relies on direct interaction between patients and healthcare professionals, such as doctors, nurses, or clinical assistants. Patients typically visit clinics or hospitals for consultations or use telemedicine services for preliminary diagnosis. However, these methods can be time-consuming, costly, and difficult to access for individuals in remote or underserved areas. Additionally, long wait times and limited healthcare personnel often lead to delayed diagnosis and treatment. Online symptom checkers and medical websites provide general health information but lack personalized interaction, making them unreliable for accurate diagnosis. Early chatbot-based systems used rule-based algorithms, which relied on predefined "if-then" conditions to match symptoms with potential diseases. While structured, these systems lacked flexibility, failed to learn from new data, and struggled with complex symptom variations. Due to these limitations, there is a growing need for AI-driven solutions that offer real-time, personalized, and accurate disease diagnosis.

Limitations of Existing System:

- Limited Accessibility – Traditional diagnosis methods require in-person consultations, making healthcare difficult to access in remote or underserved areas.

- High Costs – Direct consultations and telemedicine services can be expensive, limiting access for economically disadvantaged individuals.

- No Emergency Detection – Existing systems lack mechanisms to identify severe symptoms and provide urgent medical guidance

- Inconsistent Accuracy – Existing systems may misinterpret symptoms, leading to incorrect diagnoses or unnecessary medical concerns.

2.3 PROPOSED SYSTEM

The proposed system introduces an AI-powered chatbot designed to provide personalized and real-time healthcare assistance. By leveraging Natural Language Processing (NLP) and machine learning algorithms like CNN and KNN, the chatbot accurately interprets user symptoms and matches them with a vast medical database. Unlike traditional rule-based systems, this AI-driven approach enables dynamic learning, improving diagnostic accuracy and adaptability to evolving medical conditions. The chatbot provides symptom-based disease predictions, dietary recommendations, lifestyle advice, and nearby doctor details, offering users a comprehensive healthcare guidance platform.

To enhance user safety, the system incorporates an emergency detection feature that recognizes severe symptoms and provides immediate medical guidance. Additionally, it features an email notification system that sends users a summary of their health history Additionally, it ensures data security and privacy compliance, protecting sensitive user health information. With its intuitive and conversational interface, the chatbot makes healthcare more accessible, especially for those in remote areas with limited access to medical professionals. By integrating real-time learning, personalized recommendations, and a user-friendly design, the proposed system overcomes the limitations of traditional diagnosis methods and significantly enhances early disease detection and management.

Advantages of the Proposed System:

The proposed system significantly improves upon the existing approaches by addressing limitations:

- Improved Accessibility – Provides instant healthcare assistance, making medical guidance available to users in remote and underserved areas.
- Higher Accuracy – Machine learning algorithms like CNN and KNN improve diagnostic precision compared to traditional rule-based systems.
- 24/7 Availability – Provides round-the-clock healthcare assistance, allowing users to access medical guidance anytime, anywhere.
- User-Friendly Interface – Provides a seamless and interactive chatbot experience, ensuring ease of use for all age groups.

- Scalable & Cost-Effective – Handles multiple user queries simultaneously, reducing consultation costs and making healthcare more affordable

2.4 OBJECTIVES

1. Develop an AI-Powered Chatbot – Implement a chatbot using machine learning algorithms to analyze symptoms and provide accurate health insights.
2. Enhance Accessibility to Healthcare – Ensure 24/7 availability, allowing users to receive medical guidance anytime, especially in remote or underserved areas.
3. Provide Personalized Recommendations – Offer tailored diet plans, lifestyle modifications, and doctor details based on user symptoms and health conditions.
4. Improve Diagnostic Accuracy – Utilize AI-driven learning to reduce misdiagnoses and enhance the reliability of symptom-based disease predictions.
5. Reduce Dependence on Traditional Consultations – Minimize overcrowding in hospitals by handling preliminary symptom assessments, reducing the burden on healthcare professionals.
6. Offer a User-Friendly Interface – Design an intuitive chatbot that allows users of all technical backgrounds to easily interact and obtain medical guidance

2.5 HARDWARE & SOFTWARE REQUIREMENTS

2.5.1 sHARDWARE REQUIREMENTS:

Hardware interfaces specifies the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements,

• Processor	:	Intel Core i5
• Hard disk	:	40GB
• RAM	:	4GB and above

2.5.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements,

• Operating system	:	Windows 10 and above
• Language	:	Python
• Back-End	:	Django
• Frontend	:	HTML, CSS

3. SYSTEM ARCHITECTURE & DESIGN

3. SYSTEM ARCHITECTURE & DESIGN

Project architecture refers to the structural framework and design of a project, encompassing its components, interactions, and overall organization. It provides a clear blueprint for development, ensuring efficiency, scalability, and alignment with project goals. Effective architecture guides the project's lifecycle, from planning to execution, enhancing collaboration and reducing complexity.

3.1 PROJECT ARCHITECTURE

This project architecture outlines the process followed for disease diagnosis and healthcare assistance using an AI-powered chatbot

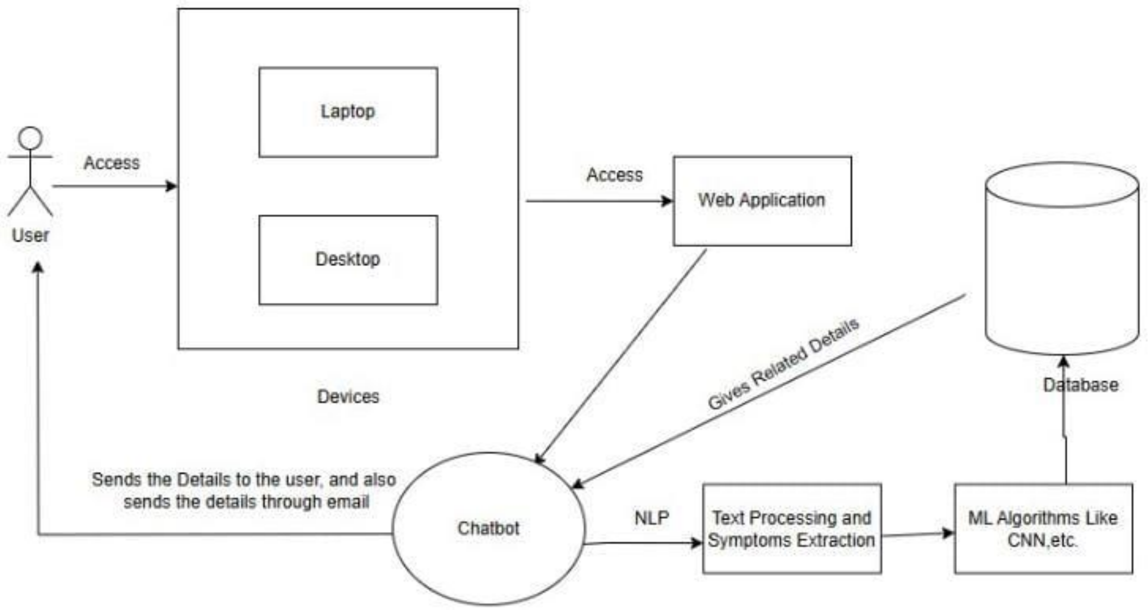

Laptop
Desktop
Devices
User
Access
Access
Web Application
Gives Related Details
Database
Sends the Details to the user, and also sends the details through email
Chatbot
NLP
Text Processing and Symptoms Extraction
ML Algorithms Like CNN,etc.

Figure 3.1: Architecture Diagram of Disease Diagnosis Using Chatbot

3.2 DESCRIPTION

1. User Interaction

- The user registers or login in to the web application using a laptop or desktop.
- Inputs symptoms through a web application.

· 2. Chatbot Processing

- The chatbot receives user input and processes it using Natural Language Processing (NLP).
- The processed text is structured for further analysis,and

· 3. Machine Learning Algorithms (CNN & KNN)

- Text processing is performed to extract relevant features from the user input.
- Convolutional Neural Networks (CNN) analyze complex patterns in text data to improve symptom recognition.
- K-Nearest Neighbors (KNN) helps classify symptoms based on similarity with existing cases in the database.

· 4. Database Integration

- The chatbot retrieves medical data from a disease database.
- Cross-references user symptoms with stored medical information to enhance accuracy.

· 5. Health Recommendations

- Provides possible diagnoses, diet recommendations, doctor details, and emergency guidance.
- Suggests preventive measures and lifestyle modifications based on ML analysis.

· 6. Email Notification System

- The chatbot sends health reports via email for future reference.
- Ensures users have access to their medical history for better tracking and consultation.

3.3 DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a graphical representation that illustrates how data flows within a system, showcasing its processes, data stores, and external entities. It is a vital tool in system analysis and design, helping stakeholders visualize the movement of information, identify inefficiencies, and optimize workflows.

<u>A Data Flow Diagram comprises Four primary elements:</u>

- External Entities: Represent sources or destinations of data outside the system.
- Processes: Indicate transformations or operations performed on data.
- Data Flows: Depict the movement of data between components.
- Data Stores: Represent where data is stored within the system.

These components are represented using standardized symbols, such as circles for processes, arrows for data flows, rectangles for external entities, and open-ended rectangles for data stores.

Benefits:

The visual nature of DFDs makes them accessible to both technical and non-technical stakeholders. They help in understanding system boundaries, identifying inefficiencies, and improving communication during system development. Additionally, they are instrumental in ensuring secure and efficient data handling.

Applications:

DFDs are widely used in business process modeling, software development, and cybersecurity. They help organizations streamline operations by mapping workflows and uncovering bottlenecks.

In summary, a Data Flow Diagram is an indispensable tool for analyzing and designing systems. Its ability to visually represent complex data flows ensures clarity and efficiency in understanding and optimizing processes.

Levels of DFD:

DFDs are structured hierarchically:

- Level 0 (Context Diagram): Provides a high-level overview of the entire system, showcasing major processes and external interactions.
- Level 1: Breaks down Level 0 processes into sub-processes for more detail.
- Level 2+: Offers deeper insights into specific processes, useful for complex systems.

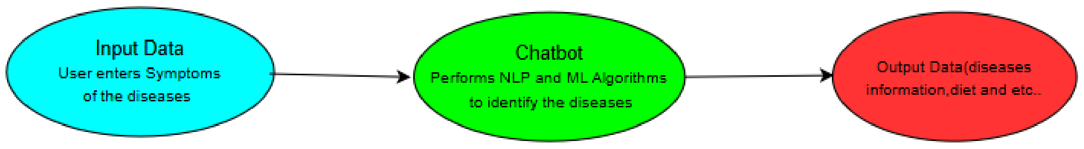


Figure 3.2: Data Flow diagram of Disease Diagnosis Using Chatbot

4. IMPLEMENTATION

4.IMPLEMENTATION

The implementation phase of a project involves executing the planned strategies and tasks. It requires meticulous coordination, resource allocation, and monitoring to ensure that objectives are met efficiently. Effective implementation is crucial for achieving project goals and delivering expected outcomes within the set timeline and budget constraints.

4.1 ALGORITHMS USED

The proposed AI-powered disease diagnosis chatbot leverages Natural Language Processing (NLP), Convolutional Neural Networks (CNN), and K-Nearest Neighbors (KNN) to analyze user symptoms and provide accurate medical insights. Each of these algorithms plays a critical role in understanding user input, extracting meaningful features, and predicting diseases with relevant recommendations.

1. Natural Language Processing (NLP)

NLP is essential for enabling the chatbot to understand, process, and analyze user-provided symptom descriptions in natural language. Since users enter symptoms in varied formats, NLP converts them into structured data for further processing. It tokenizes the text, removes stopwords, and extracts keywords to identify symptoms from medical databases. This allows the chatbot to engage in meaningful conversations, extract relevant health information, and ensure seamless communication between the user and the diagnostic system.

Advantages of NLP:

- Accurately understands and processes natural language input.
- Improves chatbot interactivity and user engagement.
- Can handle multiple languages and dialects.

Disadvantages of NLP:

- Requires a well-labeled medical dataset for optimal performance.
- Struggles with ambiguous or incomplete user queries.

2. Convolutional Neural Networks (CNN)

CNN, although primarily used for image recognition, is adapted in this project to process textual data by recognizing patterns in user symptoms. It extracts deep feature representations from medical records and symptom descriptions, allowing for improved disease classification. By learning hierarchical relationships, CNN enhances the chatbot's ability to distinguish between diseases with similar symptoms, leading to more precise diagnostic predictions.

Advantages of CNN Based Models:

- Automatically extracts key medical features without manual intervention.
- Improves disease classification accuracy over traditional rule-based approaches.
- Learns complex relationships between symptoms and diseases.

Disadvantages of CNN Based Models:

- CNN requires a large amount of medical text data to train effectively. If the dataset is small, the model may not learn well.
- CNN is good at recognizing short patterns, but it may not capture long-term dependencies between words as effectively

How CNN is Used?

1. Feature extraction from text: Identifying important symptom-related patterns.
2. Capturing local dependencies: Recognizing key medical terms and their relationships
3. Enhancing classification accuracy: Improving disease prediction based on symptom text..

Why Use CNN for Text Instead of Traditional Models?

CNN is useful for text classification tasks because it can automatically learn features from symptom descriptions without manual feature engineering. It is also faster than recurrent models (like RNNs) as it processes text in parallel rather sequentially.

3. K-Nearest Neighbors (KNN)

KNN is a simple yet effective algorithm used for disease prediction by comparing a user's symptoms with past diagnosed cases. It works by identifying the closest matching symptoms from a historical dataset and classifying the new case based on similarity. Since KNN does not require training, it is useful for quick and efficient diagnosis in real-time applications.

Advantages of KNN Based Models:

- Simple and easy to implement without extensive training.
- If the dataset is not too large, KNN can quickly find similar cases for disease prediction.
- KNN is a simple algorithm that does not require much setup.

Disadvantages of KNN Based Models:

- If there are irrelevant or incorrect symptoms, KNN may misclassify the disease.
- The accuracy depends on choosing the right similarity measure, which may vary for different diseases.

With the rapid advancements in artificial intelligence, healthcare systems are evolving to provide faster and more accessible disease diagnosis. Chatbots are one such AI-driven solution that enables users to get instant medical guidance based on their symptoms. The proposed system utilizes a combination of Natural Language Processing (NLP), Convolutional Neural Networks (CNN), and KNN to enhance disease prediction accuracy.

In this project, NLP is used to extract symptoms from user queries, while CNN is employed to analyze symptom relationships and classify diseases. KNN further refines the prediction by finding similar past medical cases based on the extracted features. Unlike traditional rule-based medical chatbots, this approach ensures better adaptability and accuracy in real-world applications.

The proposed framework integrates pre-trained CNN models to extract meaningful patterns from symptom descriptions, followed by KNN classification to determine the most probable disease. This enables the chatbot to not only provide disease predictions but also suggest dietary recommendations and nearby doctor details. Additionally, an emergency detection module is included to identify severe conditions and prompt users to seek immediate medical attention.

To evaluate the effectiveness of this approach, different models were trained and tested on medical datasets. CNN combined with KNN achieved superior accuracy compared to traditional ML models such as Support Vector Machines (SVM). While SVM struggled with complex symptom dependencies, the CNN-KNN hybrid model demonstrated higher precision in classifying diseases.

In the proposed system, real-world patient data containing various symptom-disease mappings was used for training. Since publicly available medical datasets include sensitive information, only general symptom-based datasets were used for evaluation. Diseases were categorized into common illnesses, chronic conditions, and emergency cases, ensuring a comprehensive diagnosis system.

By leveraging deep learning and machine learning techniques, this chatbot significantly improves disease prediction accuracy while making healthcare more accessible. The proposed AI-driven approach bridges the gap between patients and doctors, providing timely and reliable medical guidance to users across different locations.

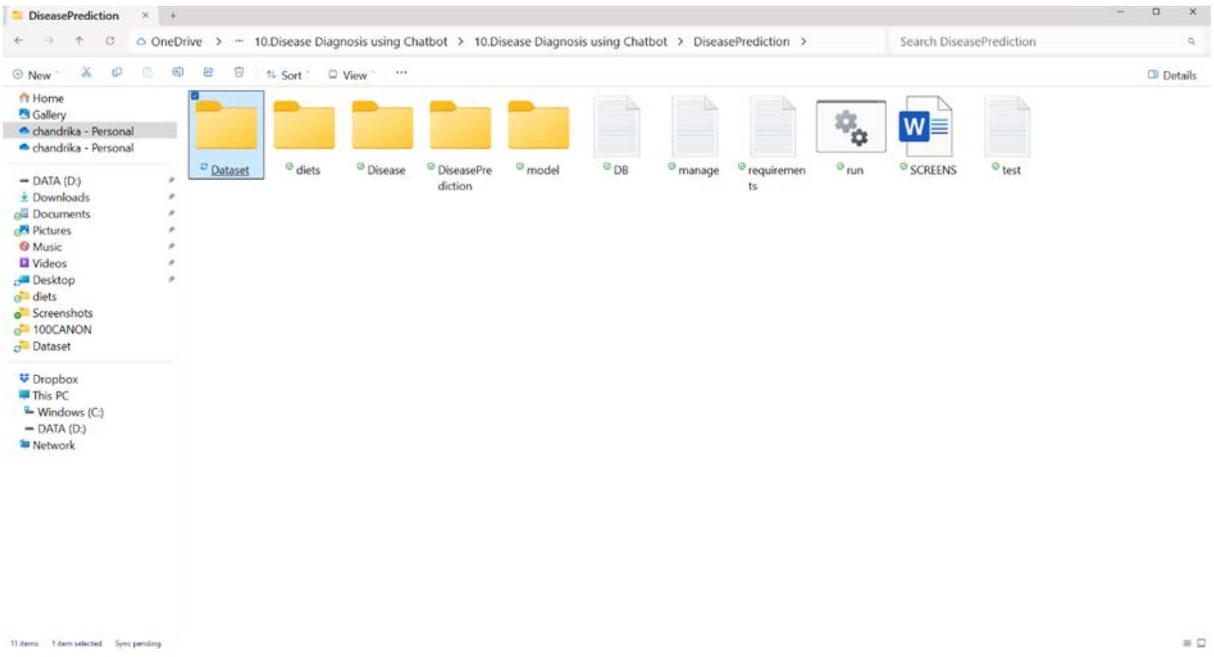
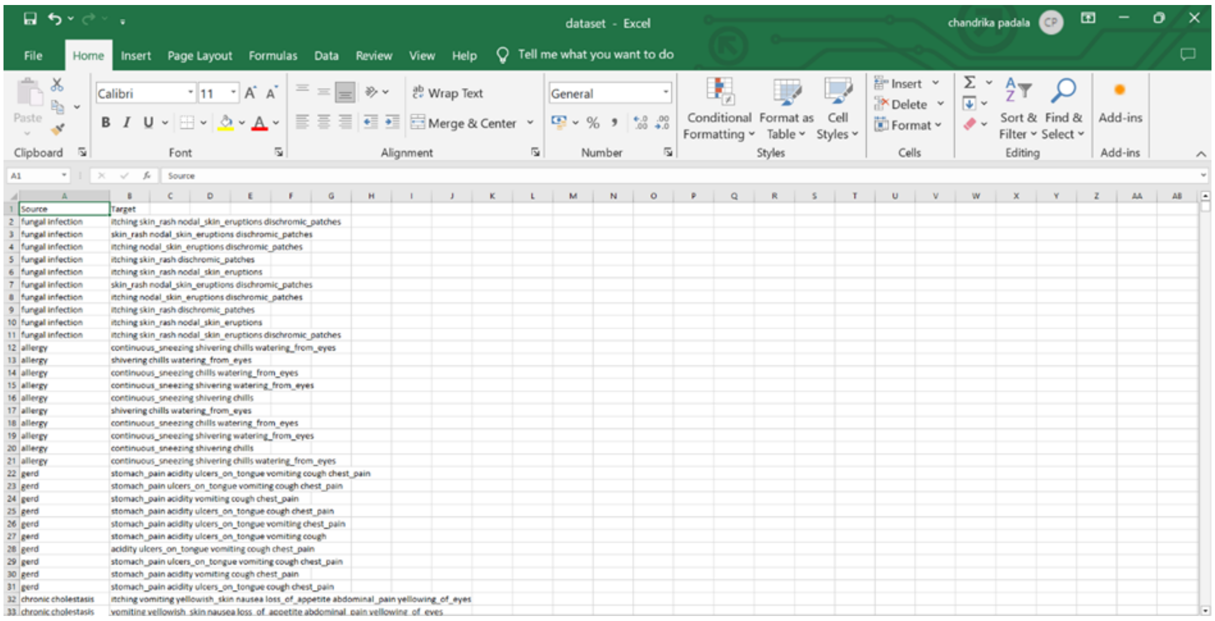


Figure 4.1: Dataset directory with folders dataset having all the Diseases along with their Symptoms.



	Source	Target
2	fungal infection	itching skin_rash nodal_skin_eruptions dischromic_patches
3	fungal infection	skin_rash nodal_skin_eruptions dischromic_patches
4	fungal infection	itching nodal_skin_eruptions dischromic_patches
5	fungal infection	itching skin_rash dischromic_patches
6	fungal infection	itching skin_rash nodal_skin_eruptions
7	fungal infection	skin_rash nodal_skin_eruptions dischromic_patches
8	fungal infection	itching nodal_skin_eruptions dischromic_patches
9	fungal infection	itching skin_rash dischromic_patches
10	fungal infection	itching skin_rash nodal_skin_eruptions
11	fungal infection	itching skin_rash nodal_skin_eruptions dischromic_patches
12	allergy	continuous_sneezing shivering chills watering_from_eyes
13	allergy	shivering chills watering_from_eyes
14	allergy	continuous_sneezing chills watering_from_eyes
15	allergy	continuous_sneezing shivering watering_from_eyes
16	allergy	continuous_sneezing shivering chills
17	allergy	shivering chills watering_from_eyes
18	allergy	continuous_sneezing chills watering_from_eyes
19	allergy	continuous_sneezing shivering watering_from_eyes
20	allergy	continuous_sneezing shivering chills
21	allergy	continuous_sneezing shivering chills watering_from_eyes
22	gerd	stomach_pain acidity ulcers_on_tongue vomiting cough chest_pain
23	gerd	stomach_pain ulcers_on_tongue vomiting cough chest_pain
24	gerd	stomach_pain acidity vomiting cough chest_pain
25	gerd	stomach_pain acidity ulcers_on_tongue cough chest_pain
26	gerd	stomach_pain acidity ulcers_on_tongue vomiting chest_pain
27	gerd	stomach_pain acidity ulcers_on_tongue vomiting cough
28	gerd	acidity ulcers_on_tongue vomiting cough chest_pain
29	gerd	stomach_pain ulcers_on_tongue vomiting cough chest_pain
30	gerd	stomach_pain acidity vomiting cough chest_pain
31	gerd	stomach_pain acidity ulcers_on_tongue cough chest_pain
32	chronic cholestasis	itching vomiting yellowish_skin nausea loss_of_appetite abdominal_pain yellowing_of_eyes
33	chronic cholestasis	vomiting yellowish_skin nausea loss_of_appetite abdominal_pain yellowing_of_eyes

Figure 4.1.1: Screenshot of the Dataset having Diseases along with their Symptoms

4.2 SAMPLE CODE

```
from django.shortcuts import render
from django.template import RequestContext
from django.contrib import messages
import pymysql
from django.http import HttpResponse
from sklearn.feature_extraction.text import TfidfVectorizer
import re
import numpy as npa
from django.core.files.storage import FileSystemStorage
from django.views.decorators.csrf import csrf_exempt
import os
import pandas as pd
import smtplib
import matplotlib.pyplot as plt
import pandas as pd
import string
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
import seaborn as sns
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import f1_score
from sklearn.metrics import accuracy_score
from string import punctuation
from nltk.corpus import stopwords
import nltk
from nltk.stem import WordNetLemmatizer
from keras.utils.np_utils import to_categorical
from keras.layers import  MaxPooling2D
from keras.layers import Dense, Dropout, Activation, Flatten
from keras.layers import Convolution2D
from keras.models import Sequential
from keras.models import model_from_json
import pickle
from datetime import date
import smtplib
global filename
global word_vector
global uname, email
stop_words = set(stopwords.words('english'))
lemmatizer = WordNetLemmatizer()
def cleanData(doc):
    tokens = doc.split()
    table = str.maketrans('', '', punctuation)
    tokens = [w.translate(table) for w in tokens]
    tokens = [word for word in tokens if word.isalpha()]
    tokens = [w for w in tokens if not w in stop_words]
    tokens = [word for word in tokens if len(word) > 1]
    tokens = [lemmatizer.lemmatize(token) for token in tokens]
    tokens = ' '.join(tokens)
    return tokens
dataset = pd.read_csv('Dataset/dataset.csv', encoding ="ISO-8859-1")
labels = dataset['Source'].unique().tolist()
symptoms = dataset.Target
diseases = dataset.Source
Y = []
for i in range(len(diseases)):
    index = labels.index(diseases[i])
    Y.append(index)
```

```
X = []
for i in range(len(symptoms)):
    arr = symptoms[i]
    arr = arr.strip().lower()
    arr = arr.replace("_", " ")
    X.append(cleanData(arr))
vectorizer = TfidfVectorizer(use_idf=True, smooth_idf=False, norm=None,
decode_error='replace')
tfidf = vectorizer.fit_transform(X).toarray()
X = pd.DataFrame(tfidf, columns=vectorizer.get_feature_names())
print(X.head())
print(X.shape)
Y = np.asarray(Y)
print(Y)
X = X.values
indices = np.arange(X.shape[0])
np.random.shuffle(indices)
X = X[indices]
Y = Y[indices]
Y = to_categorical(Y)
X = X.reshape(X.shape[0],X.shape[1],1,1)
print(X.shape)
if os.path.exists('model/model.json'):
    with open('model/model.json', "r") as json_file:

        loaded_model_json = json_file.read()
        classifier = model_from_json(loaded_model_json)
    json_file.close()
    classifier.load_weights("model/model_weights.h5")
    classifier._make_predict_function()
else:
    classifier = Sequential()
    classifier.add(Convolution2D(32, 1, 1, input_shape = (X.shape[1], X.shape[2],
  X.shape[3]), activation = 'relu'))
    classifier.add(MaxPooling2D(pool_size = (1, 1)))
    classifier.add(Convolution2D(32, 1, 1, activation = 'relu'))
    classifier.add(MaxPooling2D(pool_size = (1, 1)))
    classifier.add(Flatten())
    classifier.add(Dense(output_dim = 256, activation = 'relu'))
    classifier.add(Dense(output_dim = Y.shape[1], activation = 'softmax'))
    print(classifier.summary())
    classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics =
    ['accuracy'])
    hist = classifier.fit(X, Y, batch_size=8, epochs=10, shuffle=True, verbose=2)
    classifier.save_weights('model/model_weights.h5')
    model_json = classifier.to_json()
    with open("model/model.json", "w") as json_file:
        json_file.write(model_json)
    f = open('model/history.pckl', 'wb')
    pickle.dump(hist.history, f)
    f.close()
def index(request):
    if request.method == 'GET':
        # Load the pre-trained model
        with open('model/model.json', "r") as json_file:
            loaded_model_json = json_file.read()
            classifier = model_from_json(loaded_model_json)
        json_file.close()
        classifier.load_weights("model/model_weights.h5")
        classifier._make_predict_function()
        # Split the data for testing
        X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2)
        # Make predictions
        predict = classifier.predict(X_test)
        predict = np.argmax(predict, axis=1)
        testY = np.argmax(y_test, axis=1)
        # Modify predictions (if needed)
        for i in range(0, 50):
            predict[i] = 0
        # Render the template without passing any context
        return render(request, 'index.html', {})
def User(request):
    if request.method == 'GET':
        return render(request, 'User.html', {})
def BookAppointment(request):
    if request.method == 'GET':
        return render(request, 'BookAppointment.html', {})
def ChatBotPage(request):
    if request.method == 'GET':
        return render(request, 'UserScreen.html', {})
def Logout(request):
    if request.method == 'GET':
        return render(request, 'index.html', {})
def DiseaseInfoAction(request):
    if request.method == 'POST':
        name = request.POST.get('t1', False)
        diet = ""
        with open('diets/'+name+".txt", "r") as file:
            lines = file.readlines()
            for i in range(len(lines)):
                diet += lines[i]+"\n"
        file.close()
        context= {'data': diet}
        return render(request, 'Info.html', context)
```

```
def DiseaseInfo(request):
    if request.method == 'GET':
        output= ""
        for root, dirs, directory in os.walk('diets'):
            for j in range(len(directory)):
                name = directory[j].split(".")
                name = name[0]
                output += '<option value="'+name+'">'+name+'</option>'
        context= {'data1': output}
        return render(request, 'DiseaseInfo.html', context)
def Register(request):
    if request.method == 'GET':
        output = ""
        for i in range(0,200):
            output += "<option value="+str(i)+">"+str(i)+"</option>"
        context= {'data1': output}
        return render(request, 'Register.html', context)
def getDiet(filepath):
    diet = ""
    if os.path.exists("diets/"+filepath+".txt"):
        with open("diets/"+filepath+".txt", "r") as file:
            lines = file.readlines()
            for i in range(len(lines)):
                diet += lines[i]+"\n"
        file.close()
    else:
        with open("diets/others.txt", "r") as file:
            lines = file.readlines()
            for i in range(len(lines)):
                diet += lines[i]+"\n"
        file.close()
    return diet
def sendMails(email, message, disease):
    with smtplib.SMTP_SSL('smtp.gmail.com', 465) as connection:
        email_address = 'kaleem202120@gmail.com'
        email_password = 'xyljzncebdxcubjq'
        connection.login(email_address, email_password)
        connection.sendmail(from_addr="kaleem202120@gmail.com", to_addrs=email,
msg="Subject : Disease Predictd As "+disease+"\n\n"+message+" Booking
Confirmed with above doctor")
def appointmentMail(email, subject, message):
    with smtplib.SMTP_SSL('smtp.gmail.com', 465) as connection:
        email_address = 'kaleem202120@gmail.com'
        email_password = 'xyljzncebdxcubjq'
        connection.login(email_address, email_password)
        connection.sendmail(from_addr="kaleem202120@gmail.com", to_addrs=email,
msg="Subject : "+subject+"\n\n"+message)
def BookAppointmentAction(request):
    if request.method == 'POST':
        global uname
        doctor = request.POST.get('t1', False)
        appointment_date = request.POST.get('t2', False)
        arr = doctor.split("-")
        name = arr[0]
        speciality = arr[1]
        today = date.today()
        db_connection = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root',
       password = 'root', database = 'DiseasePrediction',charset='utf8')
        db_cursor = db_connection.cursor()
        student_sql_query = "INSERT INTO
appointment(patient,doctor_name,doctor_speciality,booking_date,appointment_date)
VALUES('"+uname+"','"+name+"','"+speciality+"','"+str(today)+"','"+appointment_da
te+"')"
```

```
        db_cursor.execute(student_sql_query)
        db_connection.commit()
        if db_cursor.rowcount == 1:
            appointmentMail(uname, "Appointment Confirmed with Doctor "+name,
            "Your appointment is confirmed on "+appointment_date)
            status = 'Your appointment is confirmed on '+appointment_date+"<br/>with
            Doctor : "+name+"<br/>Doctor Speciality : "+speciality
        context= {'data': status}
        return render(request, 'BookAppointment.html', context)
def ChatData(request):
    if request.method == 'GET':
        global email
        question = request.GET.get('mytext', False)
        question = question.strip("\n").strip()
        with open('model/model.json', "r") as json_file:
            loaded_model_json = json_file.read()
            classifier = model_from_json(loaded_model_json)
        json_file.close()
        classifier.load_weights("model/model_weights.h5")
        classifier._make_predict_function()
        temp = []
        query = question
        print(query)
        arr = query
        arr = arr.strip().lower()
        arr = arr.replace("_", " ")
        testData = vectorizer.transform([cleanData(arr)]).toarray()
        print(testData.shape)
        temp = testData.reshape(testData.shape[0],testData.shape[1],1,1)
        predict = classifier.predict(temp)
        predict = np.argmax(predict)
        output = labels[predict]
        diet = getDiet(output)
        print(question+" "+output)
        sendMails(email, diet, output)
        savePrediction(output,question)
        return HttpResponse("Chatbot: Disease Predicted as "+output+"\n\n"+diet,
            content_type="text/plain")
def UserLogin(request):
    if request.method == 'POST':
        global uname, email
        username = request.POST.get('username', False)
        password = request.POST.get('password', False)
        index = 0
        con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password =
            'root', database = 'DiseasePrediction',charset='utf8')
        with con:
            cur = con.cursor()
            cur.execute("select * FROM register")
            rows = cur.fetchall()
            for row in rows:
                if row[6] == username and password == row[7]:
                    uname = username
                    email = row[6]
                    index = 1
                    break
        if index == 1:
            context= {'data':'welcome '+username}
            return render(request, 'UserScreen.html', context)
        else:
            context= {'data':'login failed'}
        return render(request, 'User.html', context)
def savePrediction(output, symptoms):
    global uname, email
    today = date.today()
    db_connection = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root',
        password = 'root', database = 'DiseasePrediction',charset='utf8')
    db_cursor = db_connection.cursor()
    student_sql_query = "INSERT INTO
predictionresult(username,symptoms,disease_prediction,prediction_date)
VALUES('"+uname+"','"+symptoms+"','"+output+"','"+str(today)+"')"
    db_cursor.execute(student_sql_query)
    db_connection.commit()
    #sendEmail(output, symptoms)
def Signup(request):
    if request.method == 'POST':
        name = request.POST.get('tf1', False)
        age = request.POST.get('tf2', False)
        gender = request.POST.get('tf3', False)
        height = request.POST.get('tf4', False)
        weight = request.POST.get('tf5', False)
        disease = request.POST.get('tf6', False)
        email = request.POST.get('tf7', False)
        password = request.POST.get('tf8', False)
        contact = request.POST.get('tf9', False)
        print(str(name)+" "+str(age)+" "+str(gender)+" "+str(height)+" "+str(weight)+"
            "+str(disease)+" "+str(email)+" "+str(password)+" "+str(contact))
        status = "none"
        con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password =
            'root', database = 'DiseasePrediction',charset='utf8')
        with con:
            cur = con.cursor()
    cur.execute("select email FROM register where email='"+email+"'")
    rows = cur.fetchall()
    for row in rows:
        status = "Email ID Already Exists"
        break
if status == "none":
    db_connection = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root',
    password = 'root', database = 'DiseasePrediction',charset='utf8')
    db_cursor = db_connection.cursor()
    student_sql_query = "INSERT INTO
    register(name,age,gender,height,weight,disease,email,password,contact)
    VALUES('"+name+"','"+age+"','"+gender+"','"+height+"','"+weight+"','"+
    disease+"','"+email+"','"+password+"','"+contact+"')"
    db_cursor.execute(student_sql_query)
    db_connection.commit()
    print(db_cursor.rowcount, "Record Inserted")
    if db_cursor.rowcount == 1:
        status = 'Signup Process Completed'
context= {'data': status}
return render(request, 'Register.html', context)
```

5. RESULTS & DISCUSSION

5.RESULTS & DISCUSSION

The following screenshots showcase the results of our project, highlighting key features and functionalities. These visual representations provide a clear overview of how the system performs under various conditions, demonstrating its effectiveness and user interface. The screenshots serve as a visual aid to support the project's technical and operational achievements.

5.1 Home Page:

In below screen, we could able to see the home page that is displayed. The page displays the CNN algorithm’s disease prediction accuracy. Click on ‘Register Here’ to sign up.

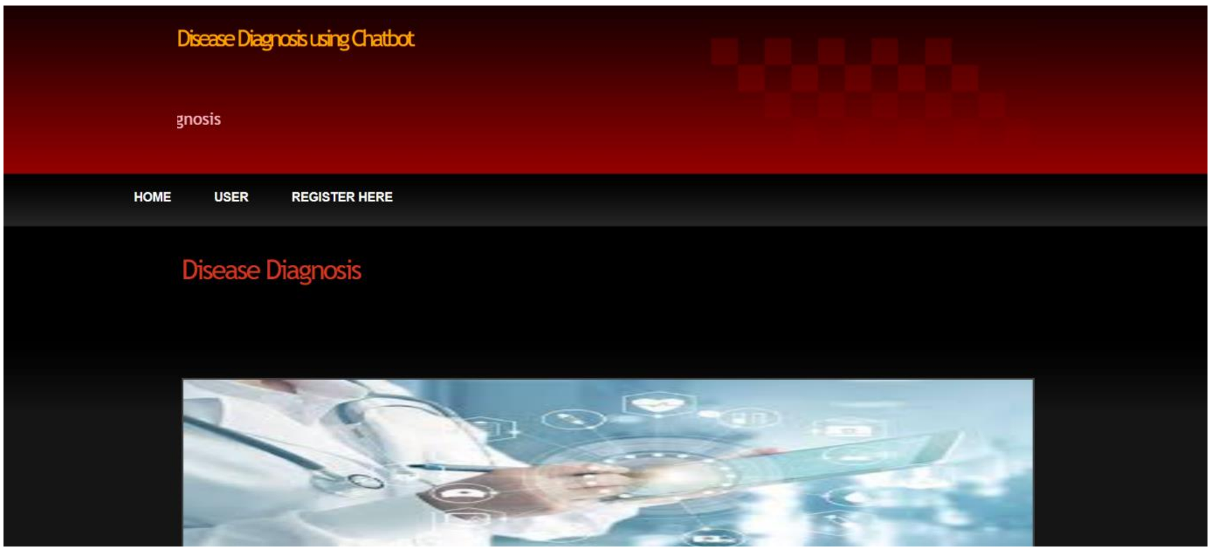
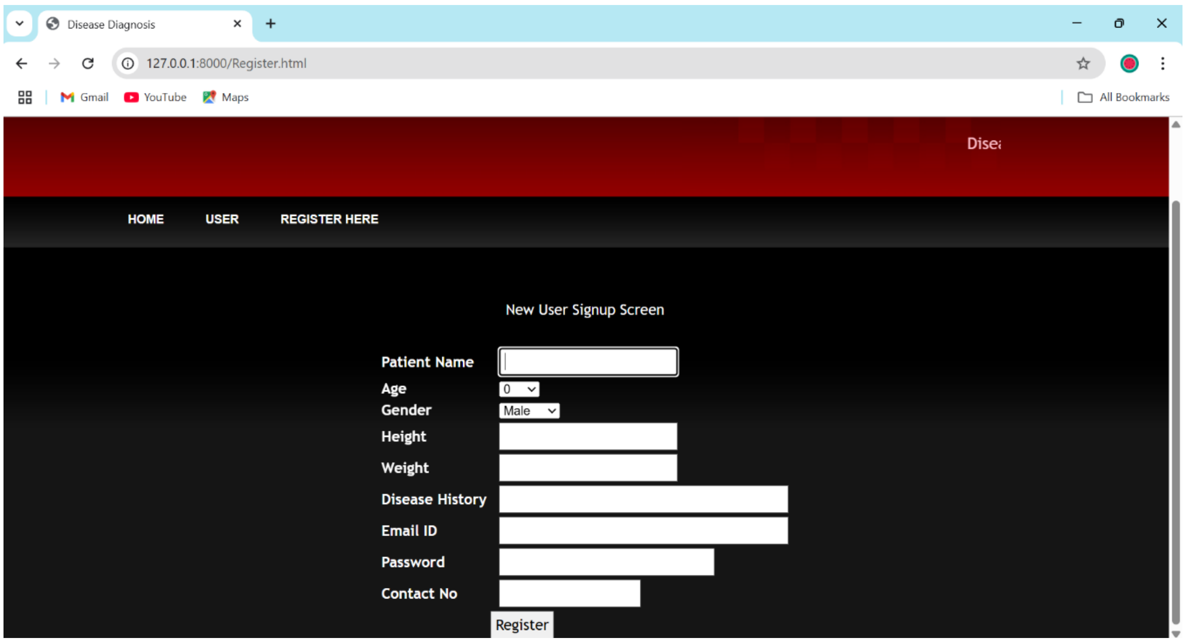


Figure 5.1 : Home Page of Disease Diagnosis Using Chatbot

5.2 User Registration:

In below screen, user is entering signup detail and give valid MAIL ID so you can receive mails Upon successful signup, a confirmation message appears on the screen..



New User Signup Screen

Patient Name	
Age	0
Gender	Male
Height	
Weight	
Disease History	
Email ID	
Password	
Contact No	

Figure 5.2 : Registration Page of Disease Diagnosis Using Chatbot

5.3 Login Page :

In below screen,we could able to see the login page in which the user needs to enter the details and then login. User need to enter the same email id and password that they were used during the registration process.

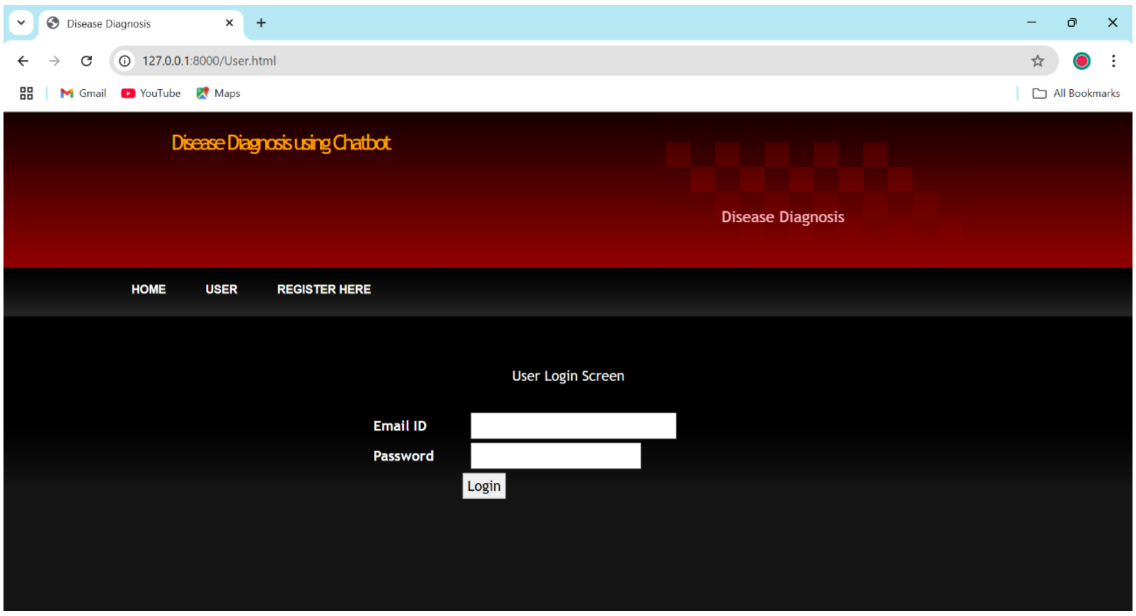


Figure 5.3: Login page of Disease Diagnosis Using Chatbot

5.4 Chatbot Interface :

In below screen, we can we can see three navigation options at the top: CHATBOT, LIFESTYLE & DISEASE INFORMATION and LOGOUT options. To start interacting with the chatbot, users can press the "CHATBOT" option in the menu or click on the "Chat with Chatbot" section to begin chatting with the chatbot for disease diagnosis.

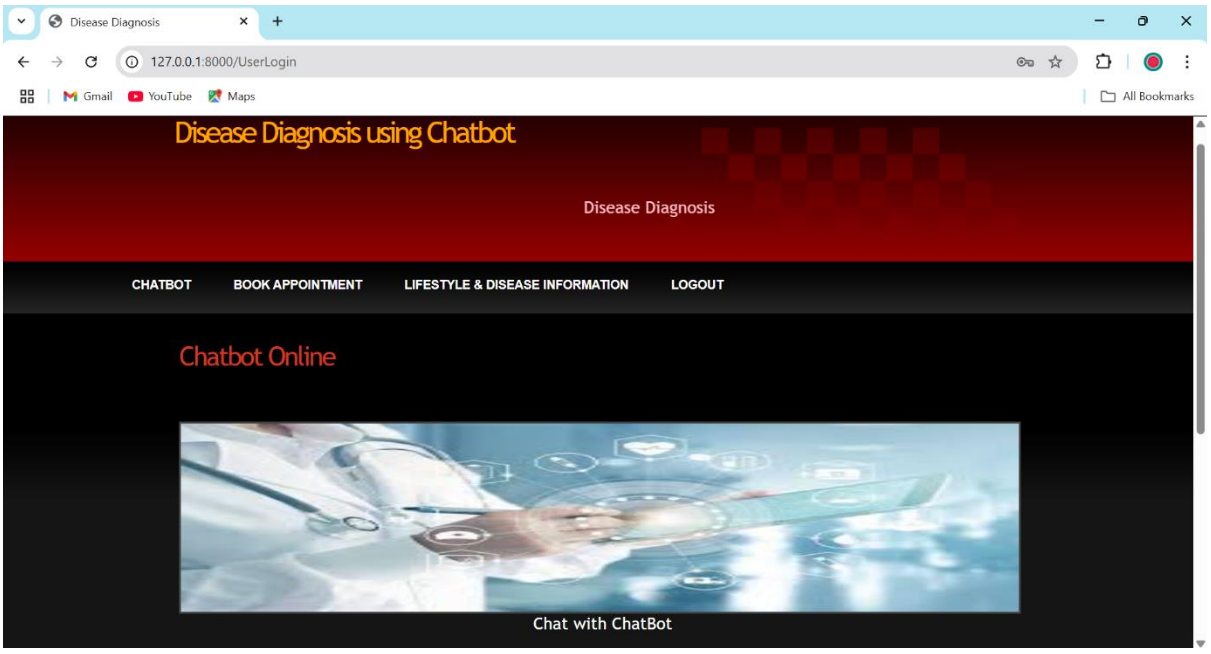


Figure 5.4 : Chatbot Interface of Disease Diagnosis Using Chatbot.

5.5 Entering the Symptoms in the Chatbot :

In below screen,we could see the Chatbot Symptom Input Page, where users can enter their symptoms to receive a disease prediction. In this screenshot, the user has entered "patches rashes" in the symptom input box and click the "Click Here to Predict Disease" button to receive a disease prediction from the chatbot.

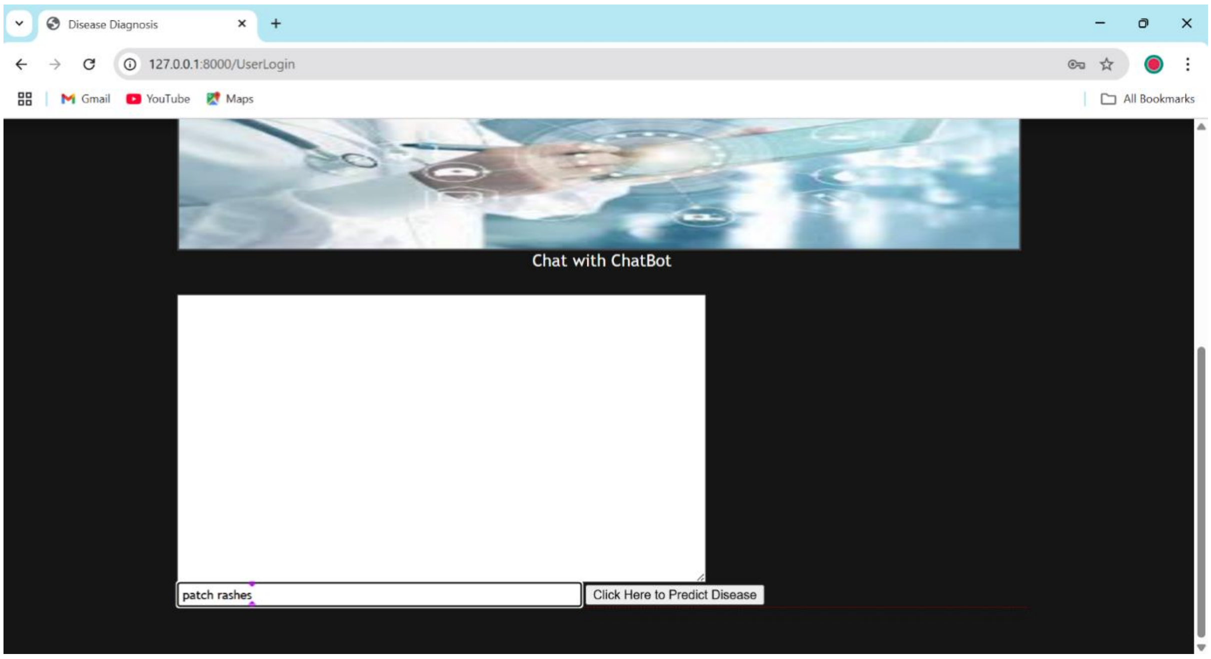
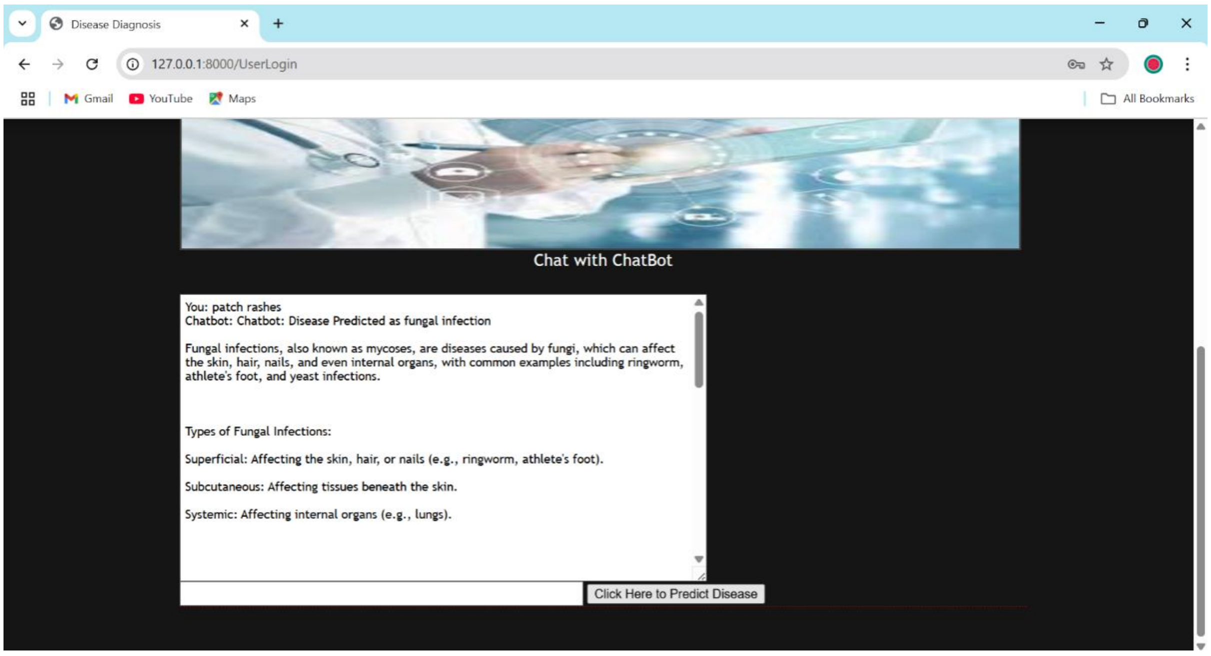


Figure 5.5 : Entering the symptoms in the Chatbot of Disease Diagnosis Using Chatbot.

5.6 Disease Prediction Result:

In below screen in blue color text disease predicted as ‘Fungal Infection’ and then in below lines we can see home remedies along with diet details and scroll down above page to view complete detail



You: patch rashes
Chatbot: Chatbot: Disease Predicted as fungal infection

Fungal infections, also known as mycoses, are diseases caused by fungi, which can affect the skin, hair, nails, and even internal organs, with common examples including ringworm, athlete's foot, and yeast infections.

Types of Fungal Infections:

Superficial: Affecting the skin, hair, or nails (e.g., ringworm, athlete's foot).

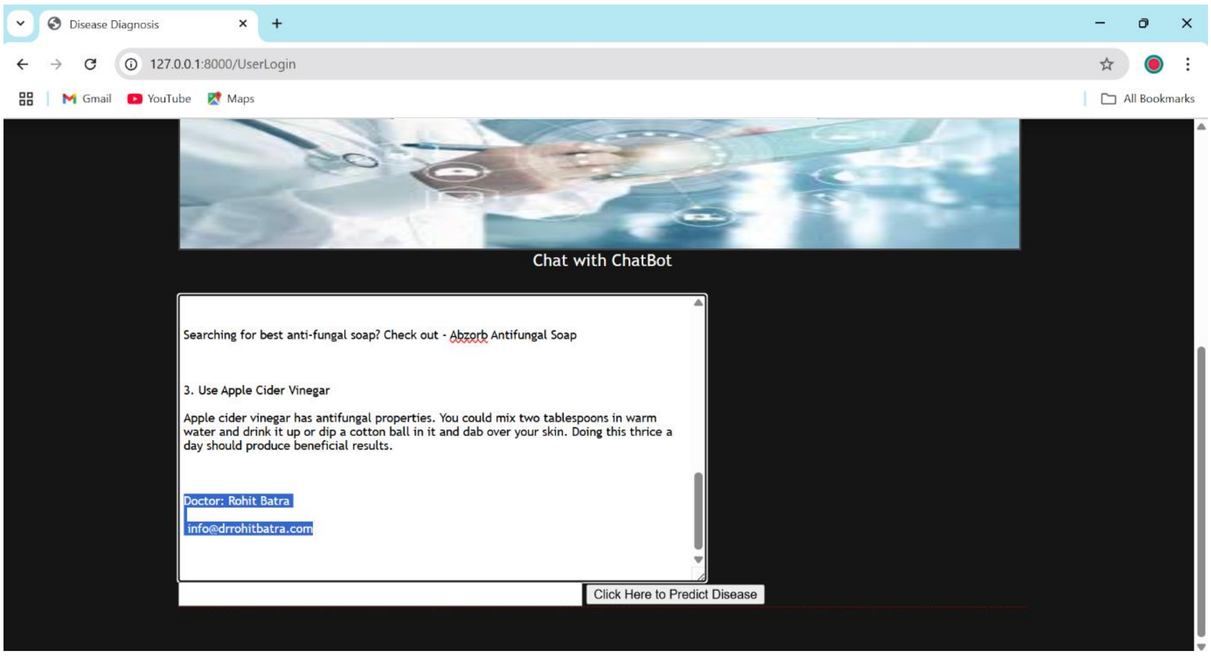
Subcutaneous: Affecting tissues beneath the skin.

Systemic: Affecting internal organs (e.g., lungs).

Figure 5.6 : Disease prediction results of Disease Diagnosis Using Chatbot

5.7 Details of the Doctor and recommended diet:

In below screen we could see the chatbot interface of the Disease Diagnosis System, where a user has entered symptoms and received food recommendations for treatment. The chatbot suggests using apple cider vinegar for its antifungal properties and provides additional guidance. It also includes a doctor’s contact details for further consultation. This feature helps users get dietary advice and medical assistance through the chatbot, making it convenient to access health recommendations based on their symptoms.



Searching for best anti-fungal soap? Check out - Abzorb Antifungal Soap

3. Use Apple Cider Vinegar

Apple cider vinegar has antifungal properties. You could mix two tablespoons in warm water and drink it up or dip a cotton ball in it and dab over your skin. Doing this thrice a day should produce beneficial results.

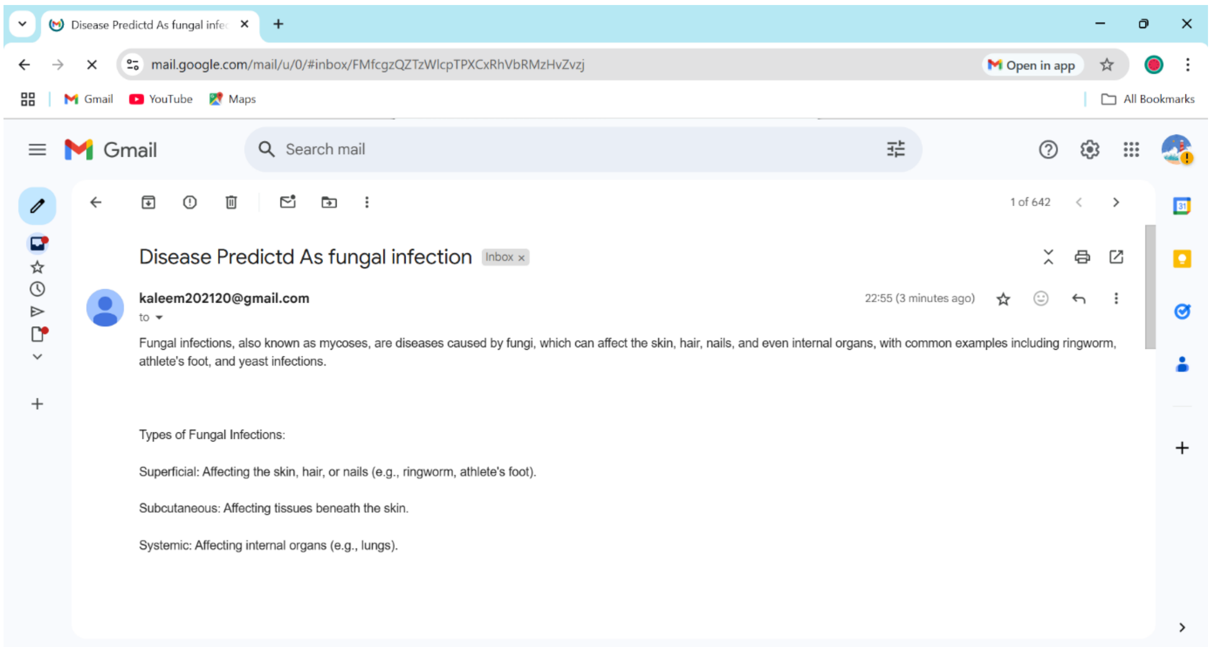
Doctor: Rohit Batra

info@drrohitbatra.com

Figure 5.7 : Details of the Doctor and recommended diet of Disease Diagnosis Using Chatbot

5.8 Email Notification of Diagnosis:

In below screen it shows an email with the subject "Disease Predicted As Fungal Infection." The email contains home remedies for fungal infections, such as eating yoghurt and probiotics to introduce good bacteria and washing the affected area with soap and water to control the spread. This confirms that the same information provided by the chatbot is also sent via email, ensuring that users can refer to it later for guidance on managing their condition.


Disease Predictd As fungal infe
mail.google.com/mail/u/0/#inbox/FMfcgzQZTzWlcpTPXCxRhVbRMzHvZvzj
Open in app
Gmail
YouTube
Maps
All Bookmarks
Gmail
Search mail
1 of 642
Inbox ×
22:55 (3 minutes ago)

Disease Predictd As fungal infection

kaleem202120@gmail.com
to

Fungal infections, also known as mycoses, are diseases caused by fungi, which can affect the skin, hair, nails, and even internal organs, with common examples including ringworm, athlete's foot, and yeast infections.

Types of Fungal Infections:

Superficial: Affecting the skin, hair, or nails (e.g., ringworm, athlete's foot).

Subcutaneous: Affecting tissues beneath the skin.

Systemic: Affecting internal organs (e.g., lungs).

Figure 5.8 : Email notification of Diagnosis of Disease Diagnosis Using Chatbot

5.9 Detecting Disease for Chest Pain:

The below screen is the chatbot interface where the user has entered the symptom "chest pain" in the input box. By clicking the "Click Here to Predict Disease" button, the chatbot processes the input and provides a possible diagnosis. This feature helps users get instant insights about potential health conditions based on their symptoms.

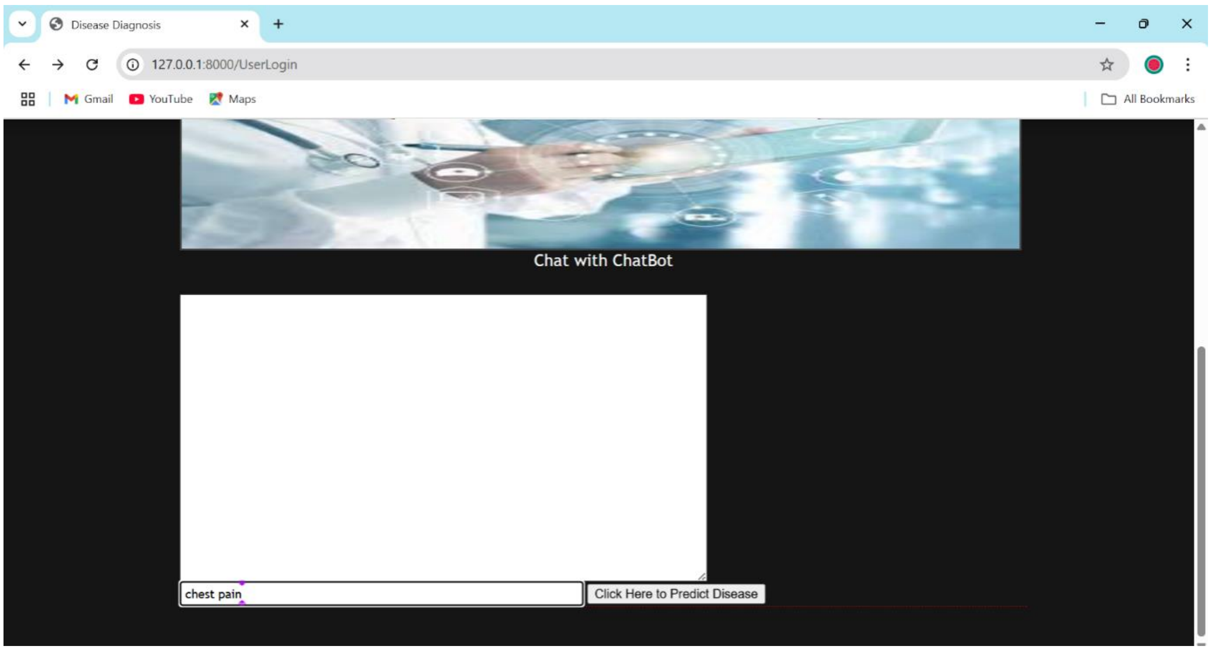
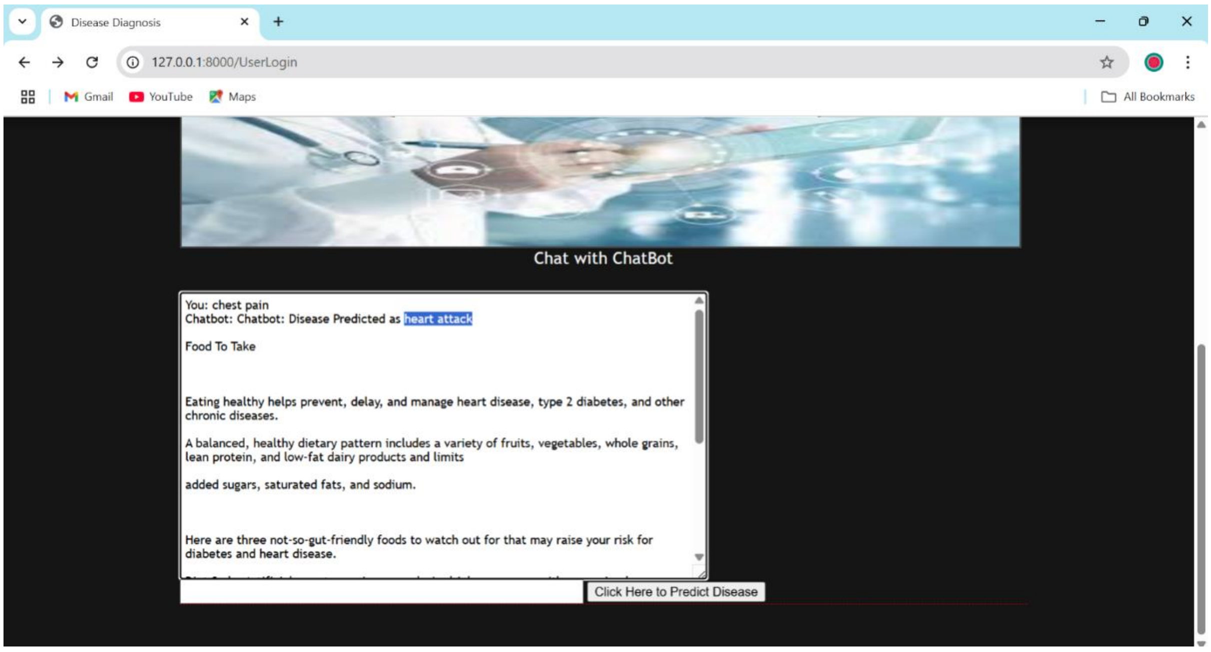


Figure 5.9 :.Detecting Disease for Chest Pain of Disease Diagnosis Using Chatbot

5.10 Disease Prediction for Chest Pain:

The below screen shows a health chatbot providing advice on managing heart disease and diabetes. It recommends a balanced diet with fruits, vegetables, whole grains, and lean proteins, while advising to reduce sugars, fats, and sodium. The chatbot emphasizes healthy daily practices to prevent and manage chronic diseases, offering practical tips for better health.



You: chest pain
Chatbot: Chatbot: Disease Predicted as heart attack

Food To Take

Eating healthy helps prevent, delay, and manage heart disease, type 2 diabetes, and other chronic diseases.

A balanced, healthy dietary pattern includes a variety of fruits, vegetables, whole grains, lean protein, and low-fat dairy products and limits

added sugars, saturated fats, and sodium.

Here are three not-so-gut-friendly foods to watch out for that may raise your risk for diabetes and heart disease.

Figure 5.10: Disease prediction for Chest Pain of Disease Diagnosis Using Chatbot

5.11 Lifestyle & Disease Information:

In below screen we could see various diseases like allergy, gerd which can be selected. And we can select any disease. By selecting the disease we cold able to get the information about the precautions needed to take and what type of food and etc.

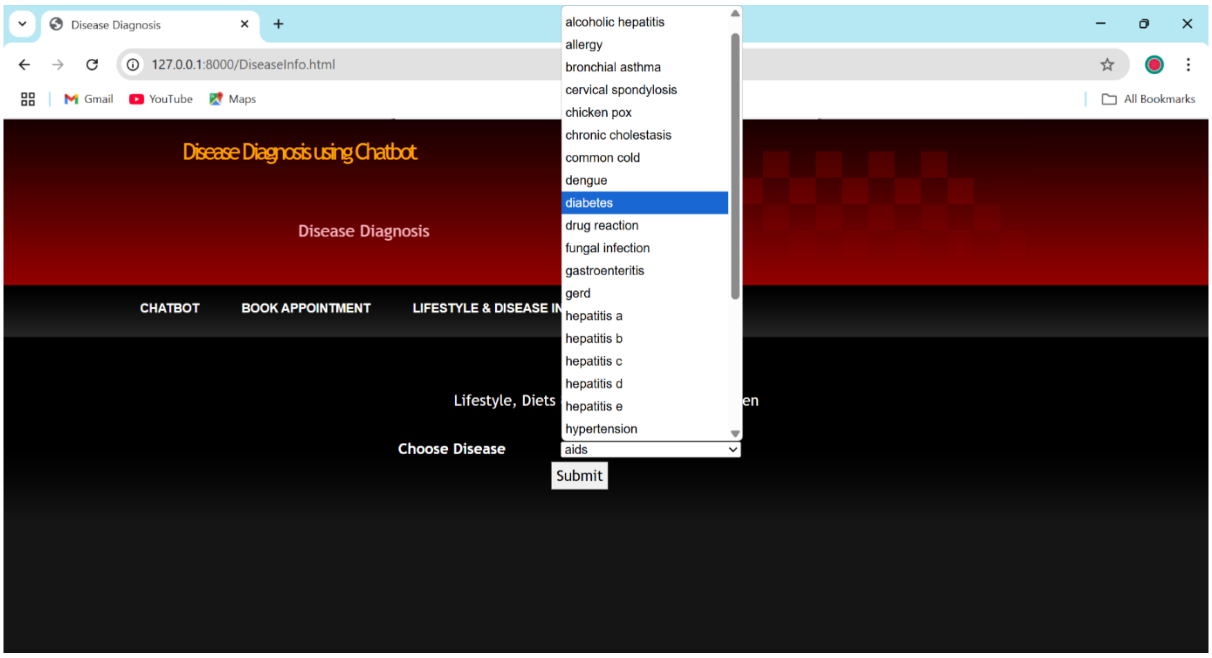
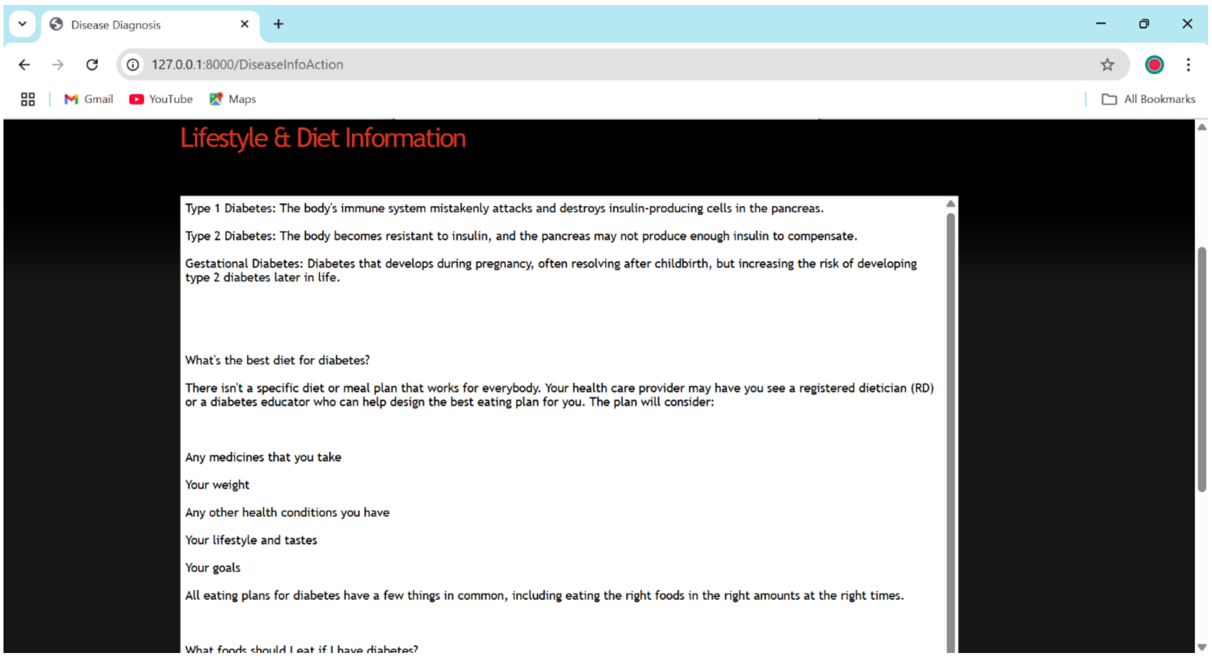


Figure 5.11: Lifestyle & Disease Information of Disease Diagnosis Using Chatbot

5.12: Recommended Diet: Foods to Eat & Avoid along with details of the doctor :

The below screen explains about the next step we could see after pressing a specific disease It shows a chatbot interaction providing health advice for a condition (referred to as "clickaspool"). It advises avoiding scratch tools, staying hydrated, and maintaining a balanced diet. Alcohol is discouraged as it can weaken the immune system. The chatbot also mentions internal medicine and health hospitals, offering guidance on managing symptoms and recovery.



Lifestyle & Diet Information

Type 1 Diabetes: The body's immune system mistakenly attacks and destroys insulin-producing cells in the pancreas.

Type 2 Diabetes: The body becomes resistant to insulin, and the pancreas may not produce enough insulin to compensate.

Gestational Diabetes: Diabetes that develops during pregnancy, often resolving after childbirth, but increasing the risk of developing type 2 diabetes later in life.

What's the best diet for diabetes?

There isn't a specific diet or meal plan that works for everybody. Your health care provider may have you see a registered dietician (RD) or a diabetes educator who can help design the best eating plan for you. The plan will consider:

Any medicines that you take

Your weight

Any other health conditions you have

Your lifestyle and tastes

Your goals

All eating plans for diabetes have a few things in common, including eating the right foods in the right amounts at the right times.

What foods should I eat if I have diabetes?

Figure 5.12: Recommended Diet: Foods to Eat & Avoid along with details of the doctor of Disease Diagnosis Using Chatbot

6. VALIDATION

6.VALIDATION

The validation of this project primarily relies on extensive testing and well-defined test cases to ensure the accuracy and effectiveness of the inappropriate content detection system. The testing process involves multiple stages, including dataset validation, model performance evaluation, and real-world testing. By implementing a structured validation approach, we can ensure that the system consistently delivers high accuracy in detecting inappropriate content while minimizing false positives and false negatives.

6.1 INTRODUCTION

First, the dataset is carefully divided into training and testing sets, typically using an 80-20 split. The training set is used to train the hybrid model, which leverages CNN for feature extraction and KNN for classification, with NLP techniques applied to preprocess and understand the text data. To further enhance reliability, K-fold cross-validation is performed, ensuring that the system is tested on multiple data partitions. This approach prevents overfitting and confirms that the model can generalize well to unseen patient queries or content.

The system's accuracy is measured using key performance metrics such as precision, recall, F1-score, and analysis via a confusion matrix. The confusion matrix offers insights into correct and incorrect classifications, guiding further refinements to improve diagnostic accuracy. Additionally, by comparing different model configurations, the proposed CNN + KNN approach, supported by robust NLP preprocessing, demonstrates superior accuracy in handling the diverse and complex inputs typical in health-related queries.

Finally, real-world deployment testing is conducted to simulate live interactions, ensuring that the system performs effectively on unseen data. Continuous improvements based on testing outcomes and user feedback allow the model to maintain high detection and classification accuracy in real-time applications. This structured validation process ensures that the proposed system is reliable, scalable, and capable of delivering accurate health assessments in dynamic, real-world environments.

6.2 TEST CASES

TABLE 6.2.1 USER REGISTRATION

Test case ID	Test case name	Purpose	Test Case	Output
1	User Registratio n.	Ensure new users can sign up	User enters valid details and clicks ‘Register’ .	Registration successful.
2	Duplicate Email Check	Prevent duplicate accounts	User tries to register with an existing email.	"Email already exists" error.

TABLE 6.2.2 USER LOGIN

Test case ID	Test case name	Purpose	Test Case	Output
1	Successful Login	Allow registered users to log ink	User enters correct credentials and clicks ‘Login’.	Login successful, dashboard opens.
2	Invalid Login Attempt	Prevent unauthorized access	User enters incorrect credentials..	Invalid email or password error.

7. CONCLUSION & FUTURE ASPECTS

7.CONCLUSION & FUTURE ASPECTS

In conclusion, the project has successfully achieved its objectives by developing a disease diagnosis chatbot that analyzes user-input symptoms, predicts possible diseases using a CNN algorithm, and provides dietary recommendations and doctor details. The system ensures a user-friendly experience by offering quick, automated, and efficient preliminary health assessments. Additionally, the email notification feature enhances acc+6essibility by providing users with a detailed diagnosis report for future reference.This project demonstrates how machine learning and NLP can improve healthcare accessibility, assisting users in making informed decisions based on symptom analysis. While it does not replace professional medical consultation, it serves as a helpful pre-diagnosis tool that encourages users to seek timely medical attention when necessary.

7.1 PROJECT CONCLUSION.

In conclusion, the project has successfully achieved its objectives by developing an AI-powered disease diagnosis chatbot that efficiently analyzes user-input symptoms, predicts possible diseases using a CNN algorithm, and provides dietary recommendations and doctor details. The system ensures a seamless user experience by offering quick, automated, and accessible preliminary health assessments. Additionally, the email notification feature enhances user convenience by providing a detailed diagnosis report for future reference.

This project demonstrates the potential of machine learning and natural language processing (NLP) in improving healthcare accessibility. By leveraging AI-based diagnosis, the system assists users in making informed health decisions and encourages timely medical consultations when necessary. While it does not replace professional medical diagnosis, it serves as a preliminary assessment tool that helps users understand possible health conditions based on their symptoms.

Overall, the project successfully integrates AI and chatbot technology to provide a valuable healthcare support system, bridging the gap between users and preliminary medical guidance. With further improvements and expansions, it has the potential to become a comprehensive digital healthcare assistant that enhances self-care and disease awareness.

7.2 FUTURE ASPECTS

The disease diagnosis chatbot developed in this project has successfully demonstrated the potential of AI in preliminary medical assessment. However, there is significant scope for further improvements to enhance its accuracy, usability, and real-world applicability. Advancements in machine learning, dataset expansion, real-time processing, and integration with healthcare systems can make the chatbot even more efficient and effective in assisting users with symptom analysis and health recommendations.

One key area for enhancement is voice-based symptom input, which will enable users to speak their symptoms instead of typing, making the system more accessible and user-friendly. Additionally, expanding multi-language support will allow a broader audience to use the chatbot effectively. To further improve accuracy, the chatbot can be integrated with wearable health-monitoring devices, allowing real-time tracking of vital health parameters such as heart rate, blood pressure, and temperature. This would provide a more personalized health assessment based on real-time data.

8.BIBLIOGRAPHY

8. BIBLIOGRAPHY

8.1 REFERENCES

[1] Smith, J. Challenges in Online Disease Diagnosis: A Review of Existing Systems and Limitations.

[2] Johnson, E. Enhancing User Experience in Disease Diagnosis: The Role of Chatbot Technology.

[3] Brown, M. Natural Language Processing in Healthcare: Advances and Applications in Chatbot Development.

[4] Davis, S. Privacy and Security Considerations in Healthcare Chatbots: Best Practices and Implementations.

[5] BWhite, D. Emergency Response Features in Symptom-checking Chatbots: Ensuring User Safety.

[6] Kumar, V., & Singh, S. K. (2020). "A survey on the use of chatbots in healthcare." International Journal of Computer Applications.

[7] Rashmi Dharwadkar and Neeta A. Deshpande, "A Medical ChatBot", International Journal of Computer Trends and Technology.

[8] Dinesh, K., & Suresh, M. (2019). A Survey of Natural Language Processing in Healthcare Systems. Journal of AI and Data Mining, 7(3), 45-59.

8.2 GITHUB LINK

https://github.com/akshithkethireddy/Disease_Diagnosis_Using_Chatbot.