A Project Report on

Healthcare Chatbot Using Machine Learning

Submitted in partial fulfillment for award of

Bachelor of Technology

Degree

in

Data Science

By

N. Chandra Vamsi (L21ADS402)

Under the guidance of
M. Rishitha, **M. Tech**
Assistant Professor

Department of Cyber Security, Data Science & AIML
Bapatla Engineering College
(Autonomous)
(Affiliated to Acharya Nagarjuna University)
BAPATLA – 522 102, Andhra Pradesh, INDIA
2023-2024

Department of
Cyber Security, Data Science & AIML

CERTIFICATE

This is to certify that the project report entitled **Healthcare Chatbot using Machine Learning** that is being submitted by N. Chandra Vamsi (L21ADS402) in partial fulfillment for the award of the Degree of Bachelor of Technology in Cyber Security to the Acharya Nagarjuna University is a record of bonafide work carried out by them under our guidance and supervision.

Signature of the Guide
M. Rishitha, M. Tech
Asst. Professor

Signature of the HOD
Prof. V. Chakradhar, M.Tech
HOD of CB, DS & AIML

Signature of External Examiner

ACKNOWLEDGEMENT

We sincerely thank the following distinguished personalities who have given their advice and support for successful completion of the work.

We are deeply indebted to our most respected guide **M. RISHITHA,** Asst. Professor**,** Department of CB, DS & AIML, for her valuable and inspiring guidance, comments, suggestions, and encouragement.

We extend our sincere thanks to Prof**. V. CHAKRADHAR**, Head of the Department of CB, DS & AIML, for extending his cooperation and providing the required resources.

We would like to thank our beloved **Principal**, **Dr. NAZEER SHAIK,** for providing the online resources and other facilities to carry out this work.

We would like to express our sincere thanks to our project coordinator **G. V. LEELA KUMARI** Asst. Prof., Department of CB, DS & AIML for her helpful suggestions in presenting this document.

We extend our sincere thanks to all other teaching faculty and non-teaching staff of the department, who helped directly or indirectly for their cooperation and encouragement.

R. Chandra Vamsi (L21ADS402)

1 TABLE OF CONTENTS

Contents **Page No**

LIST OF FIGURES

Figure **Page No**

LIST OF EQUATIONS

Equation **Page No**

ABSTRACT

Hospitals are the most widely used means by which a sick person gets medical check-ups, disease diagnosis and treatment recommendation. This has been a practice by almost all the people over the world. People consider it as the most reliable means to check their health status. The proposed system is to create an alternative to this conventional method of visiting a hospital and making an appointment with a doctor to get diagnosis. This research intends to apply the concepts of natural language processing and machine learning to create a chatbot application. People can interact with the chatbot just like they do with another human and through a series of queries, chatbot will identify the symptoms of the user and thereby, predicts the disease and recommends treatment. This system can be of great use to people in conducting daily check-ups, makes people aware of their health status and encourages people to make proper measures to remain healthy. According to this research, such a system is not widely used and people are less aware of it. Executing this proposed framework can help people avoid the time-consuming method of visiting hospitals by using this free of cost application, wherever they are.

KEY WORDS: Medical chatbot, Machine Learning, Disease Prediction, Treatment, KNN

CHAPTER – 1

1. INTRODUCTION

1.1 Background

Computers give us information; they engage us and help us in a lot of manners. A chatbot is a software or computer program that simulates human conversation or "chatter" through text or voice interactions. Yet, this paper concentrates only on text. These systems can learn themselves and restore their knowledge using human assistance or using web resources. This application is incredibly fundamental since knowledge is stored in advance. The system application uses the question-and-answer protocol in the form of a chatbot to answer user queries. This system is developed to reduce the healthcare cost and time of the users, as it is not possible for the users to visit the doctors or experts when immediately needed to diagnose a disease. The response to the question will be replied based on the user query and knowledge base. The significant keywords are fetched from the sentence and answer to those sentences, if the match is discovered or the significant, answer will be given, or similar answers will be displayed. Here the users can type in the symptoms they are facing and the chatbot will fetch the dataset with correct diagnose of disease/illness. It will also provide you the doctors details such as name, prognosis, website, etc if asked. The chatbot is made using python programming language.

1.2 Chatbots

A chatbot is artificial intelligence (AI) software that can simulate a conversation (or a chat) with a user in natural language through messaging applications, websites, mobile apps or through the telephone. A conversational agent is often described as one of the most advanced and promising expressions of interaction between humans and

machines. However, from a technological point of view, a chatbot only represents the natural evolution of a Question Answering system leveraging Natural Language Processing (NLP). They are often used for user-friendly customer-service triaging. Instead of having a conversation with another person, the user talks with a bot that is powered by basic rules or Machine Learning (ML). Every chatbot serves a specific purpose; health chatbots are designed to help with health-related issues.

Health chatbots could potentially provide many different services. They might give the user health-related information. They can help set up appointments and later send reminders for them. While they cannot make official diagnoses, if you tell them your symptoms, they can give you a likely diagnosis. In some cases, health chatbots are also able to connect patients with clinicians for diagnosis or treatment. The general idea is that in the future, these talking or texting smart algorithms might become the first contact point for primary care. Patients will not get in touch with physicians or nurses or any medical professional with every one of their health questions but will turn to chatbots first. If the little medical helper cannot comfortably respond to the raised issues, it will transfer the case to a real-life doctor.

1.3 Healthcare chatbots concerns

Chatbots provide instant conversational responses and make connecting simple for patients. And when implemented properly, they can help care providers to surpass patient expectations and improve patient outcomes. However, AI solutions some- times lack the most important quality to good care delivery: a human touch. Digital health platforms, especially platforms targeting mental health issues have seen an important growth nowadays, as in-person appointments have been relegated to the digital sphere. For those who cannot access therapy from a human clinician, mental health chatbot

platforms are an increasingly popular digital alternative. Critics of these platforms have repeatedly questioned their efficacy, due to the lack of face-to-face connection and empathy between patient and clinician. Due to technologies like AI, ML, and NLP, it is said that chatbots have reached a level where they can gauge human sentiments. The uniqueness in every individual's behaviour can confuse chatbots. Some people may prefer a casual talk, others may like the conversation to be formal. Lack of empathy and unawareness of context can prove to be a very important obstacle in the healthcare space. In the following section, we have performed a literature survey on existing Healthcare Chatbot applications and have tried to address the aforementioned, as well as other possible issues.

Health care institutions are essential as it provides to every single people in the world a proper health care. Its main purpose is to improve the current health of the community that we have shared and created. A health care institution such as hospitals or medical centres would essentially consist of numerous of doctors that were qualified and specialize on treating patients of their current illness that they endured and to restore them to proper health. There are currently a lot of health institutions that has been developed such as hospitals and medical centres which are crucial to maintain and improve the health of the community around us. It is a prime establishment of giving proper health care especially for every one of us who have ever lived. For every illness and diseases that people may face today and sometime in the future, it is because of these medical institutions and all the doctors who worked at these places that have made our lives physically better and healthy. Although hospitals now are well-equipped with their staffs working, there are still known issues that still persists that cause the staffs to make poor clinical decision that affects a patient's health such as the lack of qualified

doctors, unorganized health information and poor communications between doctors and patients.

Throughout this day and age, new technologies have been created and developed to improve people's daily life and routine especially for health care. Doctors and nurses were now guided by smart health prediction system on the purpose of storing medical information that may be used for research and diagnosis. Few years ago, doctors were expected to use their own intuition and experience to handle every medical situation that different patients are facing every day. Although their current approach may have saved people's lives back then, they are still prone to errors and wrongdoings that have endangered the human life. It is without a doubt a heavy burden for everyone especially the medical staffs to understand that a number of decisions could heavily affect other people's lives and health, it is also why such system itself proves to be vital on guiding medical staff to make a proper clinical decision to cure and restore the human health.

1.3.1 AI in healthcare

AI in healthcare is a transformative technology that leverages artificial intelligence and machine learning to revolutionize various aspects of the healthcare industry. It encompasses a wide range of applications, from diagnostics and treatment to patient management and drug discovery. One of its primary benefits is the ability to analyse vast amounts of medical data, including medical images, patient records, and genetic information, to assist healthcare professionals in making more accurate and timely decisions. AI-powered diagnostic tools can detect diseases earlier and with higher accuracy, leading to improved patient outcomes. Additionally, AI-driven predictive analytics can help hospitals and clinics optimize resource allocation and reduce costs. Furthermore, AI is aiding in drug discovery by rapidly identifying

potential compounds and accelerating research processes. While AI in healthcare holds enormous promise, it also comes with challenges related to data privacy, regulatory compliance, and ethical considerations. Nevertheless, it is poised to play an increasingly integral role in shaping the future of healthcare by enhancing patient care, reducing medical errors, and advancing medical research and development.

1.3.2 Convolutional Neural Network

Convolutional neural networks (CNNs) represent an interesting method for adaptive image processing, and form a link between general feedforward neural networks and adaptive filters. Two dimensional CNNs are formed by one or more layers of two-dimensional filters, with possible non-linear activation functions and/or down-sampling. Conventional neural network error minimization methods may be used to optimize convolutional networks in order to implement quite powerful image transformations. CNNs possess key properties of translation invariance and spatially local connections (receptive fields). CNNs are an interesting alternative when the input is spatially or temporally distributed, and the desired output of a system may be specified.

1.4 Objectives of the project

- The objective of developing a healthcare chatbot using a 1D-CNN algorithm is to create an intelligent, automated system capable of assisting individuals in addressing health-related inquiries, particularly focusing on healthcare.
- By leveraging machine learning techniques, specifically the 1D-CNN architecture, the project aims to analyse and interpret sequential health data.

- The primary goal is to enable the chatbot to recognize patterns, extract significant features, and provide informed responses or recommendations based on the input received.

CHAPTER -2

2 LITERATURE SURVEY

The literature survey provides valuable context and foundational knowledge for informing the design, implementation, and evaluation of the proposed hybrid approach, which integrates blacklist and machine learning methodologies to bolster cybersecurity defences against the proliferation of malicious URLs.

2.1 Cure Bot -An Artificially Intelligent Interactive Bot for Medical Diagnostics

An Automated System which is intended to interact with humans is termed as a Chatbot. The agent interacts with the users and supply responses to the queries. The Chatbot has the self-learning capacity which may easily understand the input and provides the specified output to the user. The bot creates its own Database within the run time while the training face from which it recognizes various patterns and by that it gives specific prediction to the specified query. The bot communicates with the user for the input & gives the output by making use of its algorithms and Prediction Analysis. The primary level of processing in our architecture deals with audio I/O. When a user makes a question, the user query is converted from audio input into Text, and this is often said as Speech-to-Text. Within the second level of processing, the extracted text is employed as a basis for performing language Understanding on the generated text to decode the semantic meaning of the user input and recognize morphemes. Within the case of a talking interface, this is often considered the primary level of processing because of the absence of the requirement for audio-to-text conversion. The Cure bot can be used by medical practitioners to connect with their patients during any emergency help. This bot has the potential to help by permitting patients to receive

supportive care without having to physically visit a hospital by using conversational Artificial intelligence-based applications for their treatment. The study helps as a computer application acting as a personal virtual doctor that has been designed and trained to interact with patients like human beings. This application provides treatment based on symptoms conveyed as per the patient's need and illness.

2.1.1 Advantages

- Reduces healthcare costs.
- Rapid processing of medical data

2.1.2 Disadvantages

- Data Privacy Concern
- Limited ability to provide personalized care

2.2 Deep Learning Techniques for Implementation of Chatbots

Chatbots are software programs that interact with clients using natural languages. The motto of the researchers was to know if chatbots can able to fool the clients that they were real humans. To develop a chatbot that can pass the Turing test, plenty of effort done with the introduction of the ELIZA chatbot in the year 1966. Various approaches for the development of chatbots and different technologies in the creation of chatbots developed because of those efforts. NLTK is a module in python which can able to perform Natural Language Processing. It is used to analyses the input in the form of speech and generate responses that are similar to humans. Nowadays there is a lot of demand for virtual assistants such as Siri, Cortana, Google Assistant and Alexa, and speech-based search engines. Nowadays Chatbots are gaining massive

demand mainly in the business sector for automating client service and also for reducing efforts of humans. Chatbots typically used for information acquisition in dialogue systems. To perfectly imitate a human response, a chatbot should examine the query asked by a client correctly and design an appropriate response. In this study we compare and discuss the different technologies used in the chatbots and also address the design and implementation of a chatbot system. After successful execution of chatbot in our college, we will implement it in other fields like medical, sports, forensic, etc. It will be very beneficial in all the areas as without spending much time, and we can access the relevant information and that too without any sorting. On success, we will make this chatbot available to all the users as an android app.

2.2.1 Advantages

- Better to Understand and Respond to Natural Language.
- Handle a high volume of requests
- Cost-Effective

2.2.2 Disadvantages

- Difficult to understand
- Maintaining deep learning models can be complex and require continuous effort
- Handle a large number of users

2.3 Contextual Chatbot for Healthcare Purposes (using Deep Learning)

As the demand in Machine Learning & AI keeps growing, new technologies will keep coming in the market which will impact our day-to-day activities, and one such technology is Virtual Assistant Bots or simply Chatbots. Chatbots have evolved from being Menu/Button based, to Keywords based and now Contextual based. The most advanced among all of the above is contextual based because it uses Machine Learning and Artificial Intelligence techniques to store and process the training models which help the chatbot to give better and appropriate response when user asks domain specific questions to the bot. In this paper we will be not only discussing about the working of our model but also the applications and relevant work conducted in this domain, also there will be discussion about the challenges and future scope of this technology. For this work, neural networks have been used to train data and various packages which help us in giving better results. In this chatbot we will be integrating the concepts of Natural Language Processing with Deep Learning for getting better results. Healthcare plays a wide role in our daily lives, whenever a person feels sick he/she visits their family doctor or any nearby clinic just to get to know what issues they are facing, in the recent years many companies & institutions have collaborated with hospitals to provide support which can help doctors and medical staff to deal with patients in better way and reduce their efforts with the help of technology. Chatbots can play a major role in reshaping the healthcare industry by providing either predictive diagnosis or any other assistantships like booking an appointment

2.3.1 Advantages

- Rapidly processes and retrieves medical information, improving response times and healthcare decision-making.

- Analyses vast healthcare datasets to identify trends

2.3.2 Disadvantages

- It may lack a true understanding of complex medical concepts
- Handling sensitive patient data raises privacy and security concerns

2.4 Developing A Medical Chatbot Using AI

The proposed idea is to develop a medical Chatbot powered by Artificial Intelligence (AI) to diagnose diseases and provide basic information about them before patients consult a doctor. The primary goals are to reduce healthcare costs and enhance access to medical knowledge. This Chatbot would serve as a medical reference tool, offering patients insights into their conditions and aiding in their healthcare journey. Chatbots, in this context, are computer programs that simulate human conversations using Machine Learning algorithms. They are designed to be versatile virtual assistants capable of tasks such as answering health-related questions. Chatbot technology is currently in high demand and widely used. Proposed the medical Chatbot functioning depends on Natural Language processing that helps users to submit their problems about their health. The User can ask any personal query related to health care through the Chatbot without being physically available to the hospital. By using the Hidden Markov Model for text voice conversion at the time of answer retrieval in the medical chatbot. The query is sent to the chatbot and gets a related answer and display the answer on the application The challenge lies in making Chatbots efficient in the medical field, which is the focus of this project. The proposed platform allows users to interact with a Chatbot that has been extensively trained on medical datasets using Machine Learning algorithms. Unlike traditional programming, where logic dictates the outcome, Machine Learning algorithms take a more natural approach, with outputs

based on the data they are trained on. The project aims to create a real-time medical system that can also convert text-based results into voice format, making it accessible and user-friendly. In summary, the project aims to harness AI and Machine Learning to develop a medical Chatbot capable of diagnosing diseases, providing medical information, and improving the healthcare experience for users.

2.4.1 Advantages

- Provides accessible and immediate medical information and guidance 24/7
- Helps reduce healthcare costs

2.4.2 Disadvantages

- Difficult to extract knowledge from the medical crowd-sourced Q&A website
- Irrelevant question-answer pairs may be extracted

CHAPTER-3

3. PROBLEM STATEMENT

Healthcare chatbot using machine learning involves developing a system that can accurately understand and respond to user queries related to health, provide relevant information, and potentially assist in symptom analysis and appointment scheduling.The most common place for sick people to receive medical examinations, disease diagnosis, and treatment recommendations is at hospitals. Almost everyone in the world has been doing this for a long time. It is regarded as the most trustworthy method for determining one's health status. Most of the time, consumers are unaware of all the available treatments or symptoms for a given ailment. Users must go visit the hospital for a checkup for minor issues, which takes additional time. As a result, there has recently been a substantial effort to reduce the workload of doctors and increase the general competency of the health care approach using machine learning

CHAPTER-4

4 SYSTEM ANALYSIS

4.1 Existing System

This existing system investigates the creation and assessment of a health chatbot system that aims to enhance healthcare services using decision trees and support vector machines (SVM). The primary goal is to develop a chatbot that effectively aids users in analysing symptoms and provides personalized healthcare recommendations. To achieve this, a comprehensive dataset is gathered, consisting of patient symptoms and corresponding diagnoses. Rigorous preprocessing techniques are applied to ensure the quality and usability of the dataset. The study employs decision trees and SVM as the machine learning algorithms. Decision trees construct a tree-like structure that enables the chatbot to analyse symptoms. By traversing the decision tree based on user inputs, the chatbot can identify the most likely diagnosis, facilitating accurate and efficient symptom analysis. On the other hand, SVM is utilized to generate tailored treatment recommendations. By examining patterns and relationships within the dataset, the SVM algorithm can provide personalized healthcare guidance to users.

4.1.1 Disadvantages

- Hyperparameters can be complex
- Difficulty in Handling Imbalanced Data
- Limited for Text Classification
- Time-consuming
- Poor performance

4.2 Proposed System

The objective of the system is to build an artificial intelligence based chatbot for healthcare using python programming language. There are numerous chatbots being used today however this particular chatbot is for making healthcare and healthcare industry more flexible, by making patients easily connect with the healthcare provider. In this chatbot we will be using a dataset containing various symptoms along with the disease related to those symptoms. Whenever the user will type in the symptoms, he/she is facing, the chatbot will fetch the dataset for those symptoms and answer the user about what type of disease it could be. We will also be using a dataset containing a list of doctors belonging to different areas of expertise, for example dermatologists, gynaecologist, orthopaedist, etc from different locations along with their details. If user wants to know the nearby doctors or have communication with a doctor curing that particular disease the chatbot will provide the user, with the details of the same. This chatbot will have a user-friendly interface. This chatbot will be very useful for patients wanting an immediate response to a particular symptom as it will be working 24x7.

4.2.1 Advantages of the proposed system

- Reduced overfitting
- Process data quickly
- Enhancing their performance over time
- Cost savings
- Scalability and Adaptability

CHAPTER – 5

5 SYSTEM REQUIREMENTS

5.1 Technologies Used

- The versatile programming language well known for its simplicity and readability, nothing but PYTHON was used.
- The standard markup language HTML, CSS, and JavaScript was used for documents designing and to be displayed in a web browser.

5.2 Hardware Requirements

- A laptop/desktop that having descent specifications and in good running condition.
- That laptop must contain a minimum of 4GB RAM and a minimum of 128GB HDD/SDD (storage) capability.

5.3 Software Requirements

- To execute our project, you need visual studio code
- You will get better experience with the OS on or above windows 10.

5.4 Libraries Used

1. Scikit Learn: A powerful machine learning library in Python, to implement various algorithms for tasks such as classification, regression, clustering, and dimensionality reduction.
2. Pandas: A popular Python library, offering powerful tools for handling structured data through its Data Frame and Series data structures.
3. Numpy: A fundamental library for scientific computing in Python, provides support for powerful array operations, mathematical functions, linear algebra,

and random number generation, serving as a backbone for many numerical computing tasks.

4. Tensorflow: A powerful open-source library for numerical computation and machine learning developed by Google Brain. It's widely used for building and training deep learning models, offering a flexible framework for tasks like classification, regression, and more. TensorFlow's key feature is its computational graph abstraction, enabling efficient execution across CPUs, GPUs, and even TPUs for accelerated performance. With its extensive documentation and large community, TensorFlow remains a top choice for both beginners and experts in the field of machine learning.
5. Flask: A lightweight and flexible web framework in Python, empowers developers to build web applications quickly and efficiently by providing essential tools and libraries for routing, templating, and interacting with databases.
6. Keras: A high-level neural networks API, written in Python, that allows for easy and fast experimentation with deep learning models. It's designed to be user-friendly, modular, and extensible, enabling both beginners and experts to quickly build and train neural networks. With Keras, you can create various types of models, from simple feedforward networks to complex convolutional and recurrent neural networks, with minimal code. It's built on top of TensorFlow, Theano, or Microsoft Cognitive Toolkit (CNTK), providing a seamless interface for building and deploying deep learning models.
7. Pyttx3: A Python library for text-to-speech conversion. It provides a simple interface for synthesizing speech from text. With `pyttsx3`, you can easily incorporate speech output into your Python projects

CHAPTER-6

6 SYSTEM DESIGN

6.1 UML Diagrams

UML stands for Unified Modelling Language. UML is a standardized general – purpose Modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The Unified Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business Modelling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the Modelling of large and complex systems. The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual Modelling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Provide a formal basis for understanding the Modelling language.
4. Encourage the growth of OO tools market.
5. Support higher level development concepts such as collaborations, frameworks, patterns, and components.

6.1.1 Use Case Diagram

A use case diagram in the Unified Modelling Language (UML) is a type of behavioural diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

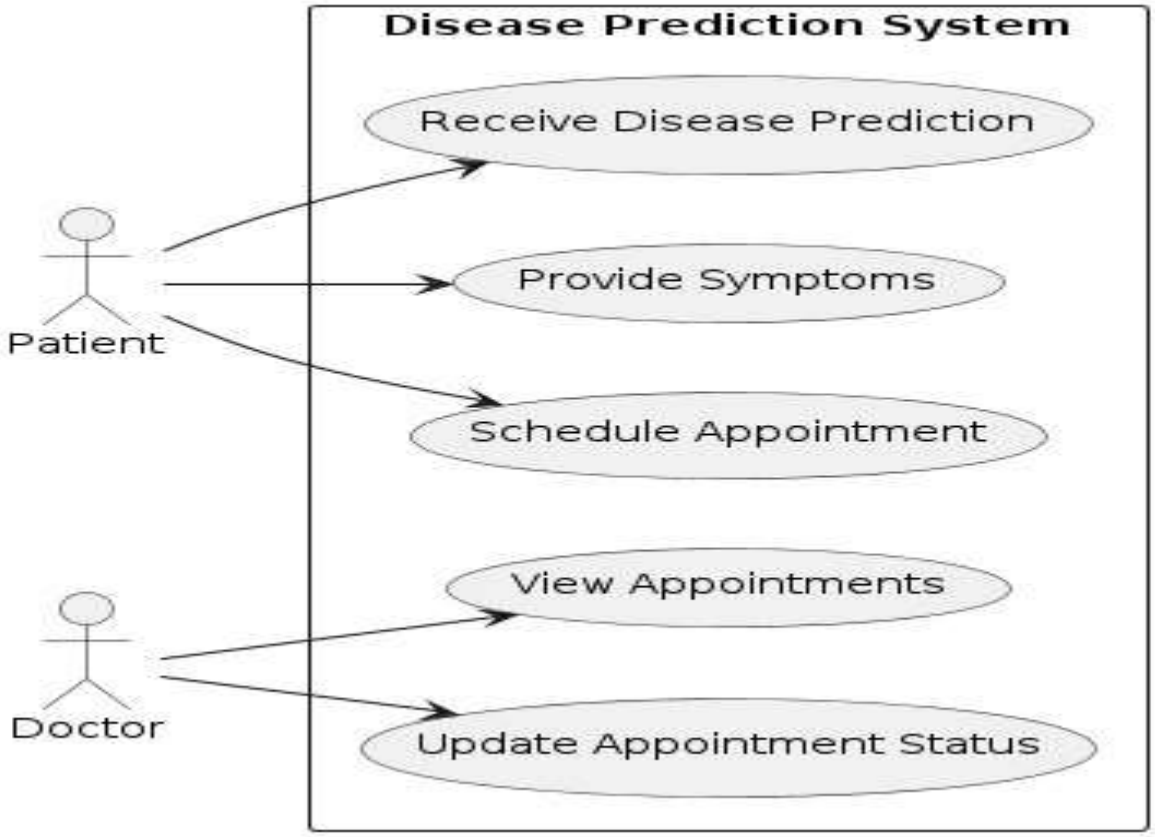


Figure 6.1 Use case Diagram

6.1.2 Class Diagram

In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing

the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

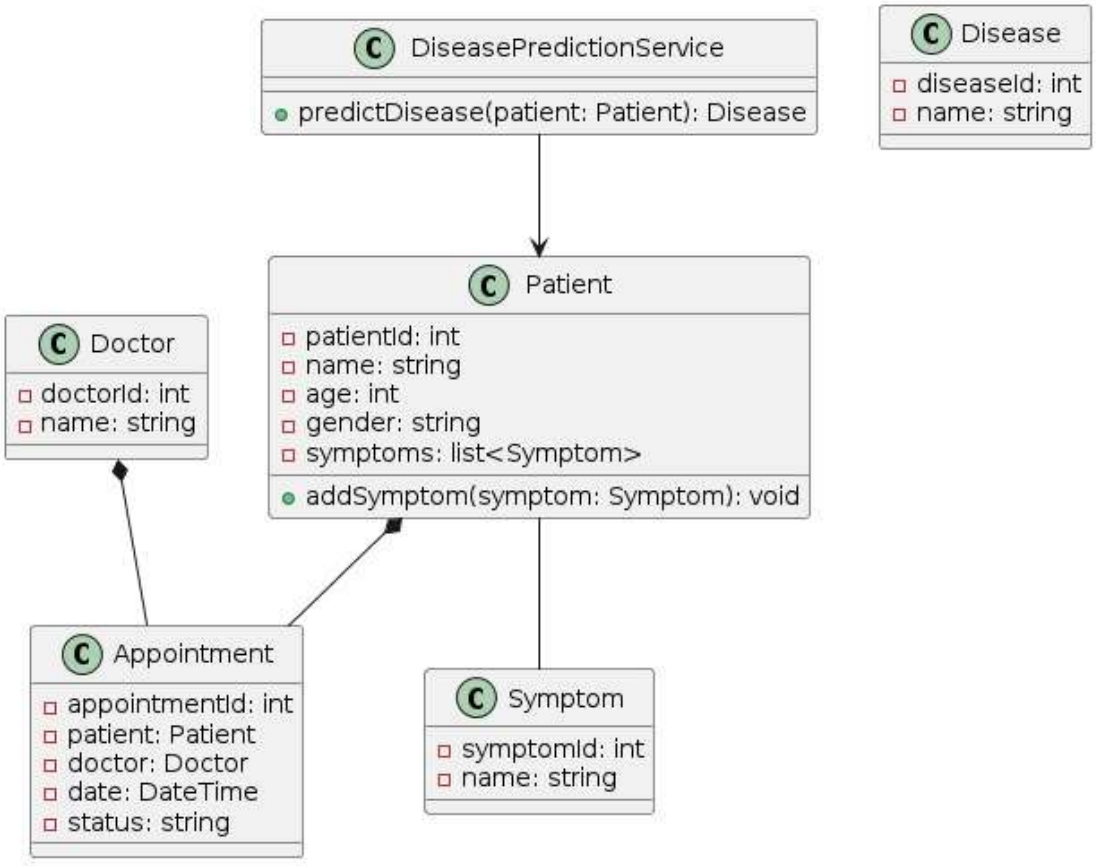


Figure 6.2 Class Diagram

6.1.3 Sequence Diagram

A sequence diagram in Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

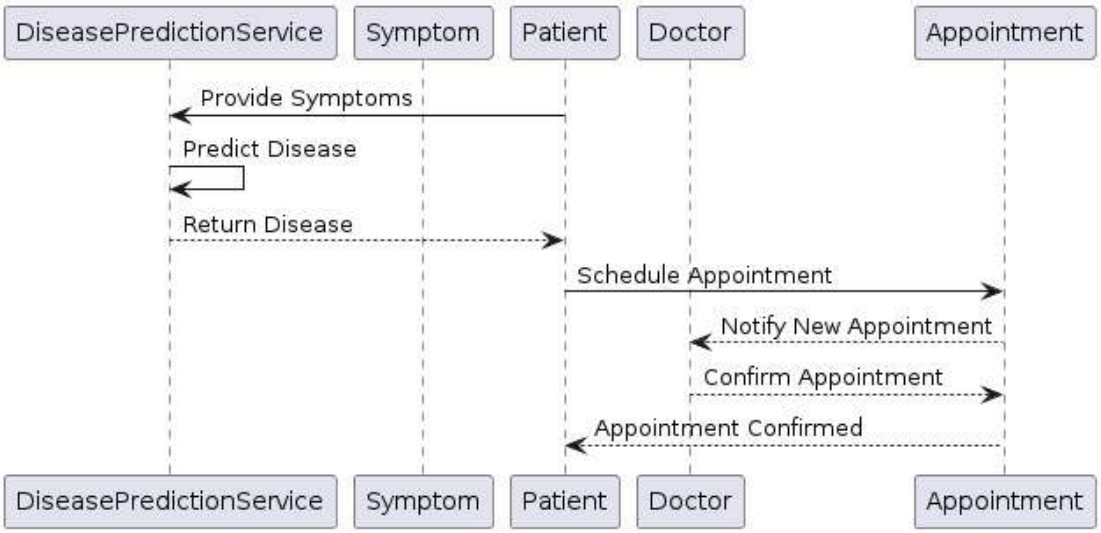


Figure 6.3 Sequence diagram

6.2 Architecture of the Proposed System

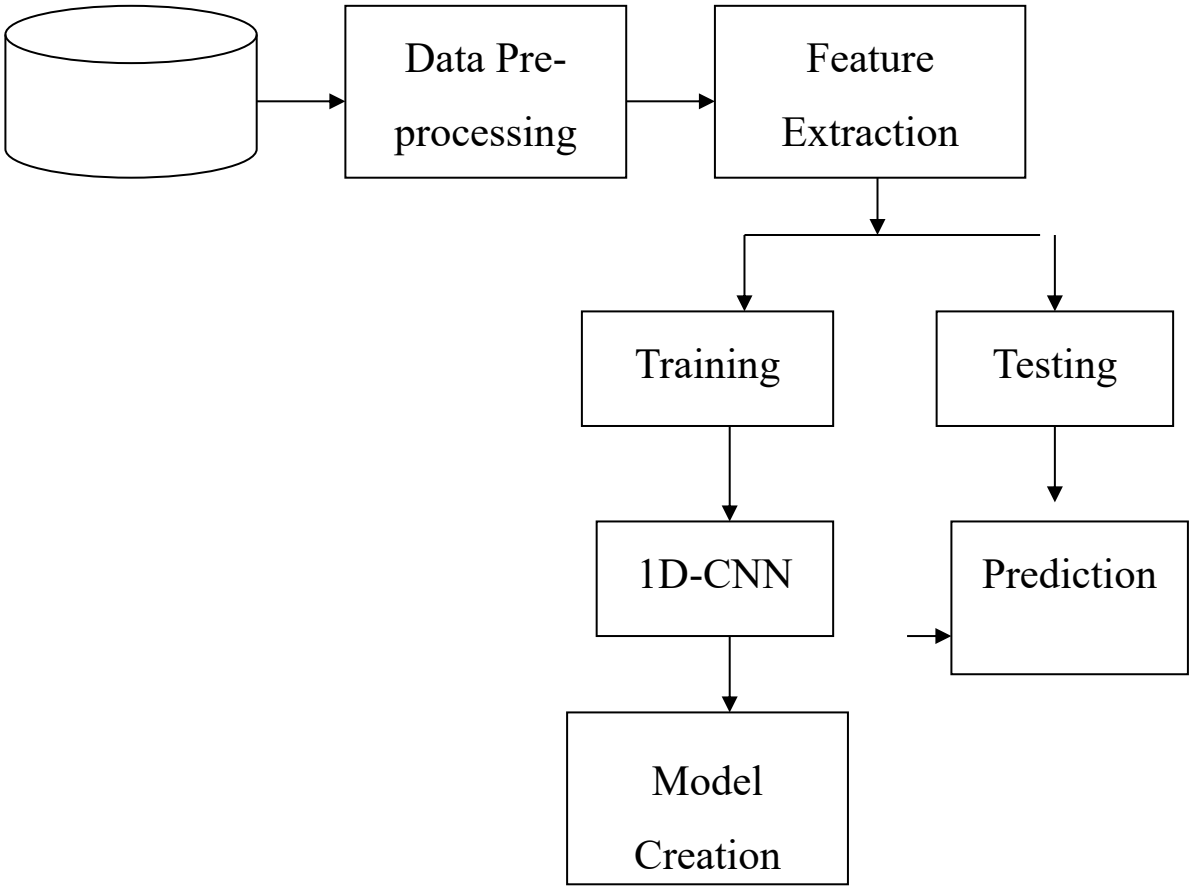


Figure 6.4 System Architecture

6.3 Deep Learning

In this work, CNN is used as the deep learning architecture which consists of stack of the convolution layer (CL), rectified linear unit (ReLU unit) layer, maximum-pooling layer (MP), fully connected layer (FC) and classifier layer. In the convolute ion layer the input gastric image is convolve with the filter bank having six filter kernels. In the RelU layer, negative values are rounded to the zero to increase the non-linear nature of the data. The max-pooling layer removes the unimportant features and decreases the size of the feature vector. In the fully connected layer, the multidimensional data vector is transformed in to the single dimensional data vector which is further provided to the classifier layer. For the classification we are suing KNN classifier which is simple to implement and takes very less time for the training. For the matching in the KNN, Euclidean distance is used. We have used three stacks of the CNN as the deep architecture. The structural design of the CNN is shown in the figure.

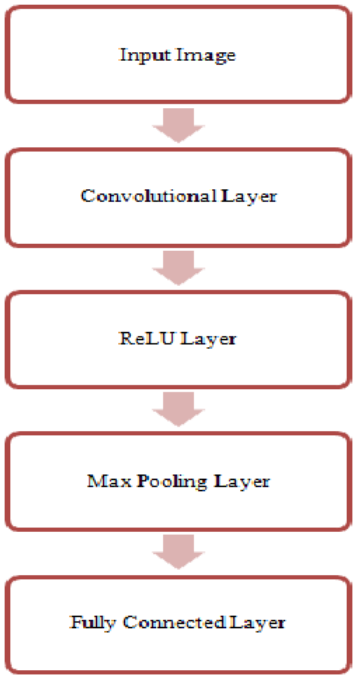


Figure 6.5 CNN 1 layer architecture

6.3.1 Convolutional Neural Networks

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm that can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image, and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics. The architecture of a ConvNet is similar to that of the connectivity pattern of Neurons in the human brain.

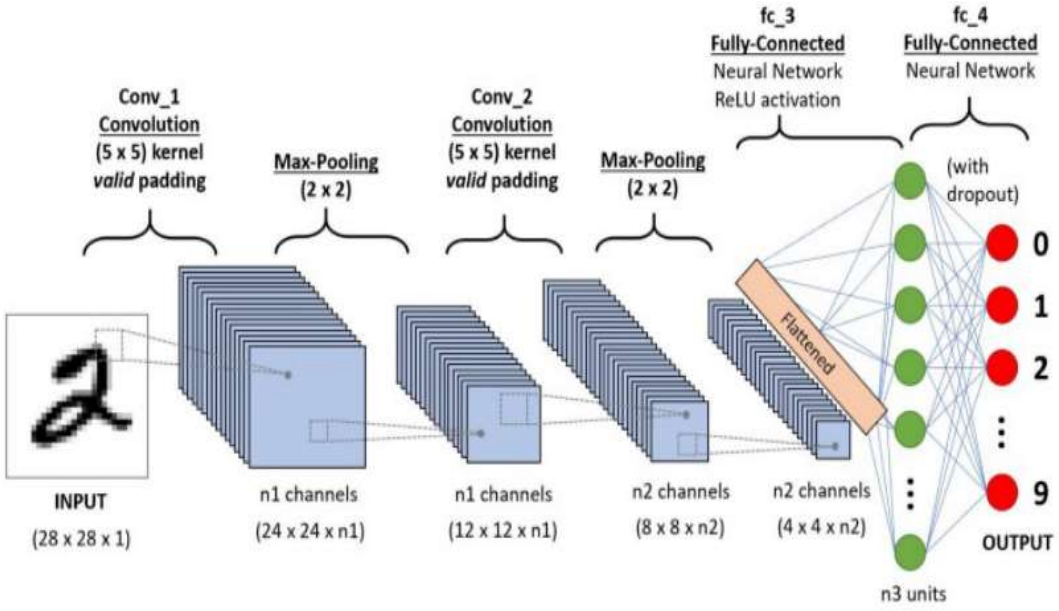


Figure 6.6 Convolutional neural network

A ConvNet is able to capture the spatial and temporal dependencies in an image through the application of relevant filters. The architecture performs a better fitting to the image dataset due to the reduction in the number of parameters involved and the reusability of weights. In other words, the network can be trained to understand the

sophistication of the image better. The objective of the convolution operation is to extract high-level features such as edges from the input image. ConvNets need not be limited to only one convolutional layer. Conventionally, the first ConvLayer is responsible for capturing the Low-Level features such as edges, color, gradient orientation, etc. With added layers, the architecture adapts to the high-level features as well, giving us a network that has a wholesome understanding of images in the dataset, similar to how we would. There are two types of results to the operation — one in which the convolved feature is reduced in dimensionality as compared to the input, and the other in which the dimensionality is either increased or remains the same. This is done by applying Valid Padding in case of the former, or the Same Padding in the case of the latter.

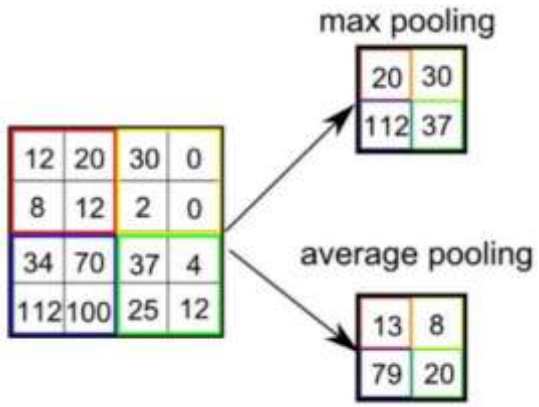


Figure 6.7 Pooling

Similar to the convolutional layer, the pooling layer is responsible for reducing the spatial size of the convolved feature. This is to decrease the computational power required to process the data through dimensionality reduction. Furthermore, it is useful for extracting dominant features which are rotational and positional invariant, thus maintaining the process of effectively training of the model. There are two types of

Pooling: Max Pooling and Average Pooling 1.3. Max Pooling returns the maximum value from the portion of the image covered by the Kernel.

On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel. Max Pooling also performs as a Noise Suppressant. It discards the noisy activations altogether and also performs de-noising along with dimensionality reduction. On the other hand, Average Pooling simply performs dimensionality reduction as a noise suppressing mechanism. Hence, we can say that Max Pooling performs a lot better than Average Pooling. The Convolutional Layer and the Pooling Layer, together form the i-th layer of a Convolutional Neural Network. Depending on the complexities in the images, the number of such layers may be increased for capturing low-levels details even further, but at the cost of more computational power.

The final output is flattened and feed it to a regular Neural Network for classification purposes. Adding a Fully-Connected layer 1.4 is a usual way of learning non-linear combinations of the high-level features as represented by the output of the convolutional layer. The Fully-Connected layer is learning a possibly non-linear function in that space. Now that we have converted our input image into a suitable form, we shall flatten the image into a column vector. The flattened output is fed to a feed-forward neural network and backpropagation applied to every iteration of training. Over a series of epochs, the model is able to distinguish between dominating and certain low-level features in images and classify them using the SoftMax classification technique.

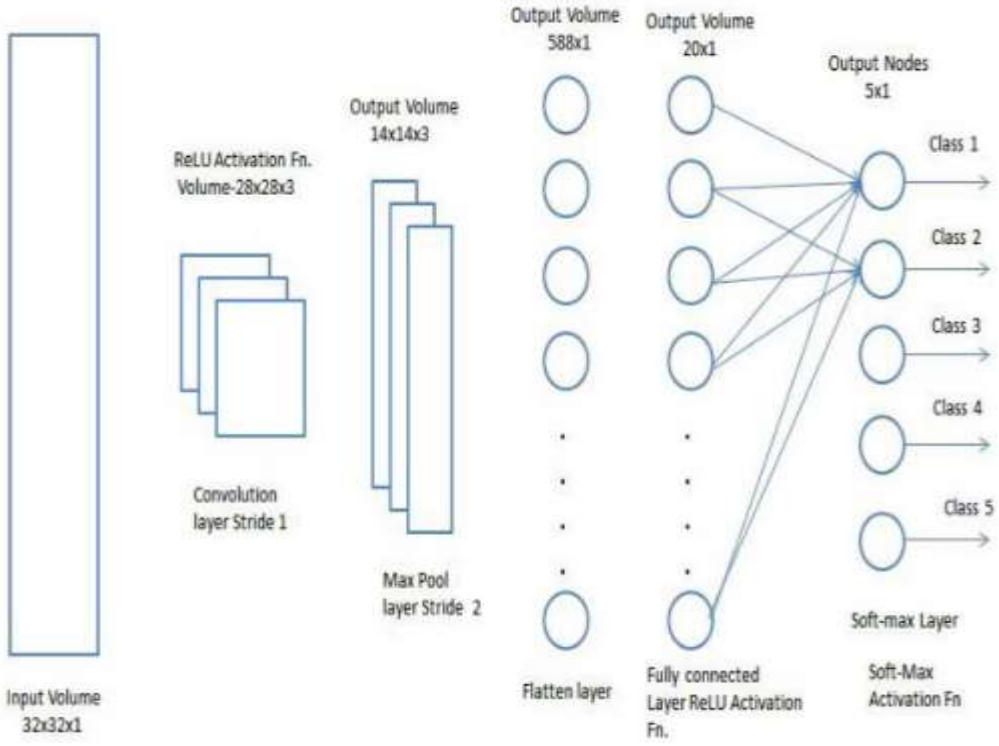


Figure 6.8 CNN classification - fully connected layer

6.3.2 D Convolutional Neural Network

This section describes the proposed architecture and all its constituent layers in detail along with the techniques used to optimize the architecture. It also gives some theoretical background about the 1-D convolutional neural network (CNN) which is central to the proposed architecture. Conventional 2D CNN has become very popular in pattern recognition problems like Image classification and object detection. CNNs are similar to ANN in which they consist of self-optimizing neurons which are trained to perform a certain task. This has led to the development of 1- D CNN which can operate on one-dimensional datasets or Time series data. The proposed architecture using this concept of 1D CNN is shown in Fig. 3.5 below.

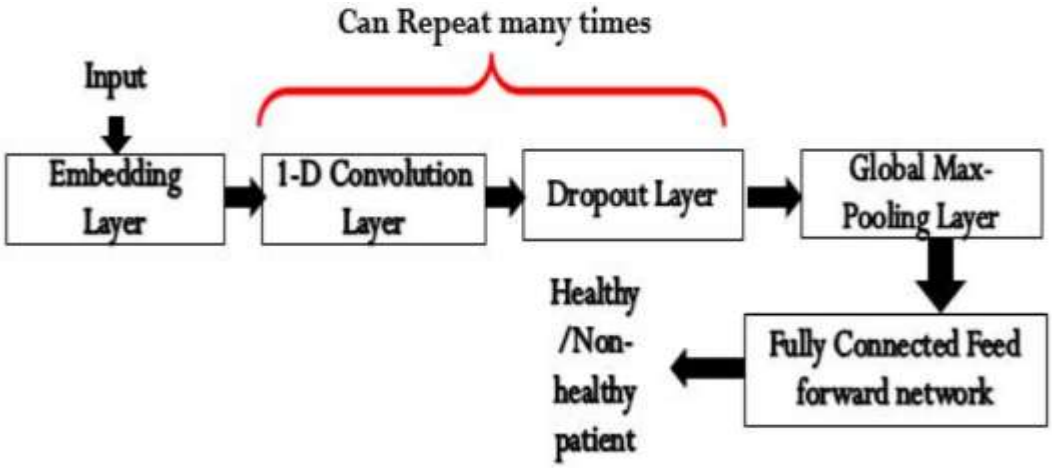


Figure 6.9 Proposed 1-D CNN Architecture

The input to the architecture will be the 13 characteristics that are crucial in the prediction of heart disease. These features are converted to a new representation called word embedding by the layer called Embedding Layer. It is similar to the Bag of Words concept used for Text data. It helps in a better representation of the dataset according to unique values present in each of the features. The Embedding layer's output is given to the 1D CNN layer for feature extraction. 1D CNN is very similar to conventional 2D CNN but the convolution operation is only applied to the one dimension which results in shallow architecture which can be easily trained on normal CPU or even embedded development boards. The convolution operation helps in finding useful hierarchical features from the dataset which are useful in classification. The dimensions of the output features after 1D CNN can be calculated using the equation given below:

$$x = \frac{w+2p-f}{s} + 1$$

Equation 1 Dimensions of output features after 1D CNN

Where x is the dimension of output features and w is the size of input features. f indicates the size of the filter used for convolutions. 'p' indicates padding which are values added on the boundary before applying convolution. 's' indicates stride which is the value travelled after applying convolution operation. The 1D convolution operation is a linear operation that is not useful in classifying nonlinear data. Most of the real-world dataset is nonlinear which requires some nonlinear operation after convolution. This nonlinear function is called an activation function. Some of the most common activation functions are the sigmoid, hyperbolic tangent, and rectified linear unit (RelU). The proposed architecture uses the RelU activation function which is easy to compute and allows faster computation. It also does not suffer from vanishing or exploding gradient problems. There can be multiple convolution layers in the architecture followed by an activation function. The proposed architecture uses two 1-D convolution layers with 128 filters and filter sizes of 3. The output of the final convolution layer is passed through the global max-pooling layer which pools the maximum value from all the channels and reduces the dimension of output. The output of pooling is given to the fully connected layer with 256 neurons which extracts the useful features for classification. This layer is similar to the hidden layer is ANN. The final layer contains a single neuron which gives the classification probability. The final layer uses the sigmoid activation function as it directly gives the probability for binary classification.

The proposed 1D CNN architecture contains around 0.13 million trainable parameters which will get adapted during the training of the network. It was observed that general CNN architecture overfitted the training data meaning that training accuracy was very high and validation accuracy was low. The dropout technique was introduced to remove overfitting. It removes random neurons with a certain probability

during training which allows the different networks to be trained at every iteration. This will help in the network not being too dependent on any single neuron of the network. The dropout layer has been introduced after each trainable layer in the proposed architecture. The addition of the dropout layer helped the training and test accuracy to be very similar which points to the network adapting well to data that it has not seen.

6.3.3 1D CNN algorithm steps

A 1D CNN (Convolutional Neural Network) is typically used for processing one-dimensional sequential data, such as time series data, signal processing, and more. Here are the general steps involved in building and training a 1D CNN:

1. **Data Preprocessing:**

- Data Collection: Gather the 1D sequential data you want to work with.
- Data Preparation: This involves formatting your data appropriately, splitting it into training, validation, and test sets.

2. **Import Necessary Libraries:**

- Libraries like TensorFlow, Keras, or PyTorch can be used for implementing CNNs.

3. **Model Building:**

- Input Layer: Define the input shape according to the sequence length.
- Convolutional Layers: Stack one or more convolutional layers to extract features from the input data.
- Activation Function: Usually ReLU is used after each convolutional layer.
- Pooling Layers: Pooling layers (like MaxPooling) help in reducing the dimensionality and extracting dominant features.
- Flattening: Flatten the output to prepare it for the fully connected layers.

- Fully Connected (Dense) Layers: Add one or more dense layers for classification/regression.
- Output Layer: Define the output layer based on the task (e.g., SoftMax for classification or linear for regression).

4. Model Compilation:

- Choose an optimizer (Adam, RMSprop, etc.).
- Define a loss function based on the task (e.g., categorical cross-entropy for classification).
- Specify metrics to measure during training (accuracy, precision, recall, etc.).

5. Training the Model:

- Feed the training data to the model.
- Adjust the model parameters (weights) using backpropagation and the selected optimizer.
- Monitor the model's performance on the validation set to avoid overfitting.

6. Model Evaluation:

- Evaluate the model using the test set to get an unbiased estimate of its performance.

7. Fine-tuning and Optimization:

- Adjust model hyperparameters, architecture, or incorporate techniques like regularization, dropout, etc., to improve performance.

8. Prediction and Inference:

- Once the model is trained, you can use it to make predictions on new, unseen data.

CHAPTER – 7

7 SYSTEM IMPLIMENTATION

7.1 Modules For The Proposed System

This project consists of three modules. They are

- Dataset Collection
- Pre-Processing
- Classification

7.1.1 Data Collection

In an AI healthcare chatbot, the Data Collection module plays a pivotal role in gathering crucial patient information and medical history. This module is designed to interact with users in a conversational manner, eliciting relevant details about their symptoms, medical conditions, and personal health background. Through a combination of natural language processing and structured questioning, the chatbot collects comprehensive data while maintaining a user-friendly and empathetic tone. It can inquire about symptoms, medication history, allergies, lifestyle factors, and more, ensuring a holistic understanding of the patient's health profile. This data is subsequently processed and made available to healthcare professionals, aiding in more accurate diagnoses and personalized treatment recommendations. Overall, the Data Collection module in an AI healthcare chatbot optimizes the initial patient assessment process, contributing to more informed and efficient healthcare interactions.

7.1.2 Data Preprocessing

Preprocessing consists of several steps, namely cleaning the data then selecting the attributes to be used. Attributes that are deemed insignificant will be removed. The

original attribute consisted of 14 attributes. After being selected and deleted, the attributes used are 11 attributes. Input attributes are cough, Tiredness, Nasal-Congestion, Runny-Nose, sore throat, fever, breath shortness, Diarrhea, gender, test indication, and age. Meanwhile, the target attribute is severity. The severity attribute has 4 values, namely none, mild, moderate, and severe.

The min–max method is used for data pretreatment. Feature normalization is to preprocess the sample data and keep it at the same order of magnitude so that comprehensive comparative evaluation can be carried out on the pre-processed data. Formula is adopted for data preprocessing that is, mapping the data to 0∼1 after linear transformation.

$$X^* = (X - \text{Min})/(\text{Max} - \text{Min}).$$

Equation 2 Normalization

In Formula, X* is the pre-processed sample data, X is the original sample data, Max is the maximum value of the sample data, and Min is the minimum value of the sample data

7.1.3 Classification

1D-CNN

- Conventional 1D CNN has become very popular in pattern recognition problems like Image classification and object detection.
- CNNs are similar to ANN in which they consist of self-optimizing neurons which are trained to perform a certain task.

- This has led to the development of 1-D CNN which can operate on one-dimensional dataset or Time series data
- The disease detection function involves leveraging the power of deep learning to sift through and interpret the provided health information.

7.1.4 Collect the dataset

In the initial phase, our primary focus is on acquiring a comprehensive dataset containing a diverse range of URLs. Leveraging platforms like Kaggle, we source datasets that encompass both benign and malicious URLs, ensuring a well-rounded representation of internet traffic. Through meticulous curation and validation processes, we strive to assemble a high-quality dataset that serves as the foundation for our subsequent analyses and model development.

7.1.5 Explore the dataset

Once the dataset is obtained, we embark on a thorough exploration process to gain insights into its structure, content, and potential biases. Utilizing data visualization techniques and statistical analysis, we delve into various aspects of the dataset, such as the distribution of URLs, the prevalence of malicious entries, and the correlation between different features. This exploratory phase equips us with valuable insights that inform subsequent steps in the workflow.

7.2 Source Code:

Main.py:

```
from flask import Flask
from flask import Flask, flash, redirect, render_template, request,
session, abort, url_for
```

```
from datetime import datetime
from datetime import date
import random
import threading
import os
import time
import urllib.request
import urllib.parse
import re
import pandas as pd
import pyttsx3
from sklearn import preprocessing
from sklearn.tree import DecisionTreeClassifier,_tree
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.svm import SVC
import csv
import warnings
import tensorflow as tf
from keras.layers import Flatten, Dense, Conv1D, MaxPool1D,
Dropout,LeakyReLU,LSTM,GRU,MaxPooling1D,Bidirectional,LSTM,Input,BatchN
ormalization
from keras.models import Sequential
warnings.filterwarnings("ignore", category=DeprecationWarning)
import pandas as pd
from sklearn.preprocessing import LabelEncoder
modeldata = tf.keras.models.load_model('Train.h5')
training = pd.read_csv('Data/Training.csv')
testing= pd.read_csv('Data/Testing.csv')

df_train = pd.read_csv('Data/Training.csv')
df_test = pd.read_csv('Data/Testing.csv')
print('=====Feature Extracted Data=========')
print(df_train)
print('Size of Data')
print('==================')
print(df_train.shape)
print('=====Label Data=========')
print(df_train.prognosis.value_counts())
import numpy as np
labelencoder = LabelEncoder()
df_train['prognosis'] =
labelencoder.fit_transform(df_train['prognosis'])


Y_train = df_train['prognosis'].values.reshape(-1,1)
Y_train=np.ravel(Y_train)
df_test['prognosis'] = labelencoder.fit_transform(df_test['prognosis'])
```

```
Y_test = df_test['prognosis'].values.reshape(-1,1)
Y_test=np.ravel(Y_test)

X_train= df_train.drop(['prognosis'],axis=1).values
X_test= df_test.drop(['prognosis'],axis=1).values

from keras.layers import Flatten, Dense, Conv1D, MaxPool1D,
Dropout,LeakyReLU,LSTM,GRU,MaxPooling1D,Bidirectional,LSTM,Input,BatchN
ormalization
from keras.models import Sequential
# Create sequential model
import numpy as np
X_train1 = np.array(X_train).reshape(X_train.shape[0],
X_train.shape[1], 1)
X_test1 = np.array(X_test).reshape(X_test.shape[0], X_test.shape[1], 1)
cnn_model1 = Sequential()
#First CNN layer  with 32 filters, conv window 3, relu activation and
same padding
cnn_model1.add(LSTM(16,return_sequences=True,input_shape=(X_train.shape
[1],1)))
cnn_model1.add(Conv1D(filters=32, kernel_size=(3,), padding='same',
activation=LeakyReLU(alpha=0.001)))
cnn_model1.add(BatchNormalization())
#Second CNN layer  with 64 filters, conv window 3, relu activation and
same padding
cnn_model1.add(Conv1D(filters=64, kernel_size=(3,), padding='same',
activation=LeakyReLU(alpha=0.001)))
cnn_model1.add(BatchNormalization())
#Fourth CNN layer with Max pooling
cnn_model1.add(MaxPooling1D(pool_size=(3,), strides=2, padding='same'))
cnn_model1.add(Dropout(0.2))
#Flatten the output
cnn_model1.add(Flatten())
#Add a dense layer with 256 neurons
cnn_model1.add(Dense(units = 128, activation=LeakyReLU(alpha=0.001)))
#Add a dense layer with 512 neurons
cnn_model1.add(Dense(units = 256, activation=LeakyReLU(alpha=0.001)))
#Softmax as last layer with five outputs
cnn_model1.add(Dense(units = 41, activation='softmax'))
cnn_model1.compile(optimizer='adam', loss =
'sparse_categorical_crossentropy', metrics=['accuracy'])
cnn_model1.summary()
##history = cnn_model1.fit(X_train1, Y_train, epochs=5, batch_size =
32, validation_data = (X_test1, Y_test))
### summarize history for accuracy
##tacc=history.history['accuracy']
##plt.plot(tacc)
##plt.plot(history.history['val_accuracy'])
```

```
##plt.title('model accuracy')
##plt.ylabel('accuracy')
##plt.xlabel('epoch')
##plt.legend(['train', 'test'], loc='upper left')
##plt.show()
### summarize history for loss
##plt.plot(history.history['loss'])
##plt.plot(history.history['val_loss'])
##plt.title('model loss')
##plt.ylabel('loss')
##plt.xlabel('epoch')
##plt.legend(['train', 'test'], loc='upper left')
##plt.show()
##cnn_model1.save('/content/gdrive/MyDrive/Train.h5')


cols= training.columns
cols= cols[:-1]
x = training[cols]
y = training['prognosis']
y1= y

reduced_data = training.groupby(training['prognosis']).max()
y_predict=modeldata.predict(X_test)
y_predict=np.argmax(y_predict,axis=1)
y_true=Y_test
from sklearn.metrics import
accuracy_score,precision_recall_fscore_support
test_accuracy=accuracy_score(y_true, y_predict)

print("Accuracy: ")
print(test_accuracy)

severityDictionary=dict()
description_list = dict()
precautionDictionary=dict()

symptoms_dict = {}

for index, symptom in enumerate(x):
       symptoms_dict[symptom] = index

def getDescription():
    global description_list
    with open('MasterData/symptom_Description.csv') as csv_file:
        csv_reader = csv.reader(csv_file, delimiter=',')
        line_count = 0
        for row in csv_reader:
            _description={row[0]:row[1]}
```

```
            description_list.update(_description)

def getSeverityDict():
    global severityDictionary
    with open('MasterData/symptom_severity.csv') as csv_file:

        csv_reader = csv.reader(csv_file, delimiter=',')
        line_count = 0
        try:
            for row in csv_reader:
                _diction={row[0]:int(row[1])}
                severityDictionary.update(_diction)
        except:
            pass

def getprecautionDict():
    global precautionDictionary
    with open('MasterData/symptom_precaution.csv') as csv_file:

        csv_reader = csv.reader(csv_file, delimiter=',')
        line_count = 0
        for row in csv_reader:
            _prec={row[0]:[row[1],row[2],row[3],row[4]]}
            precautionDictionary.update(_prec)

severityDictionary=dict()
precautionDictionary=dict()

app = Flask(__name__)

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/Symres')
def Symres():
    return render_template('Symres.html')
#Adding translator
from googletrans import Translator

translator = Translator()
@app.route('/translate', methods=['POST'])
def translate_text():

    text = request.form['text']
    target = request.form['target']
    source = request.form['source']
    translated = translator.translate(text, dest=target, src=source)
```

```
    return translated.text

@app.route('/Symptoms',methods=['POST','GET'])
def Symptoms():
    second_prediction=''
    _description=''
    FData1=''
    FData2=''
    FData3=''
    FData4=''
    Ddata1=''
    Ddata2=''
    if request.method=='POST':
        sym1=request.form['sym1']
        sym2=request.form['sym2']
        sym3=request.form['sym3']
        sym4=request.form['sym4']
        sym5=request.form['sym5']
        sym6=request.form['sym6']
        sym7=request.form['sym7']
        sym8=request.form['sym8']
        sym9=request.form['sym9']
        sym10=request.form['sym10']
        symptoms_exp=[]
        if sym1=='yes':
            symptoms_exp.append('continuous_sneezing')
        if sym2=='yes':
            symptoms_exp.append('fatigue')
        if sym3=='yes':
            symptoms_exp.append('high_fever')
        if sym4=='yes':
            symptoms_exp.append('muscle_pain')
        if sym5=='yes':
            symptoms_exp.append('runny_nose')
        if sym6=='yes':
            symptoms_exp.append('skin_rash')
        if sym7=='yes':
            symptoms_exp.append('joint_pain')
        if sym8=='yes':
            symptoms_exp.append('vomiting')
        if sym9=='yes':
            symptoms_exp.append('back_pain')
        if sym10=='yes':
            symptoms_exp.append('burning_micturition')
        print(symptoms_exp)
        df = pd.read_csv('Data/Training.csv')
        X = df.iloc[:, :-1]
```

```
        symptoms_dict = {symptom: index for index, symptom in
enumerate(X)}
        input_vector = np.zeros(len(symptoms_dict))
        for item in symptoms_exp:
            input_vector[[symptoms_dict[item]]] = 1
        X_Res = np.array([input_vector]).reshape(1,-1,1)
##          print(X_Res)
        y_predict=modeldata.predict(X_Res)
        predict=np.argmax(y_predict,axis=1)
        if(predict==0):
            fpredict="(vertigo) Paroymsal  Positional Vertigo"
        elif(predict==1):
            fpredict="AIDS"
        elif(predict==2):
            fpredict="Acne"
        elif(predict==3):
            fpredict="Alcoholic hepatitis"
        elif(predict==4):
            fpredict="Allergy"
        elif(predict==5):
            fpredict="Arthritis"
        elif(predict==6):
            fpredict="Bronchial Asthma"
        elif(predict==7):
            fpredict="Cervical spondylosis"
        elif(predict==8):
            fpredict="Chicken pox"
        elif(predict==9):
            fpredict="Chronic cholestasis"
        elif(predict==10):
            fpredict="Common Cold"
        elif(predict==11):
            fpredict="Dengue"
        elif(predict==12):
            fpredict="Diabetes"
        elif(predict==13):
            fpredict="Dimorphic hemmorhoids(piles)"
        elif(predict==14):
            fpredict="Drug Reaction"
        elif(predict==15):
            fpredict="Fungal infection"
        elif(predict==16):
            fpredict="GERD"
        elif(predict==17):
            fpredict="Gastroenteritis"
        elif(predict==18):
            fpredict="Heart attack"
        elif(predict==19):
```

```
            fpredict="Hepatitis B"
        elif(predict==20):
            fpredict="Hepatitis C"
        elif(predict==21):
            fpredict="Hepatitis D"
        elif(predict==22):
            fpredict="Hepatitis E"
        elif(predict==23):
            fpredict="Hypertension"
        elif(predict==24):
            fpredict="Hyperthyroidism"
        elif(predict==25):
            fpredict="Hypoglycemia"
        elif(predict==26):
            fpredict="Hypothyroidism"
        elif(predict==27):
            fpredict="Impetigo"
        elif(predict==28):
            fpredict="Jaundice"
        elif(predict==29):
            fpredict="Malaria"
        elif(predict==30):
            fpredict="Migraine"
        elif(predict==31):
            fpredict="Osteoarthristis"
        elif(predict==32):
            fpredict="Paralysis (brain hemorrhage)"
        elif(predict==33):
            fpredict="Peptic ulcer diseae"
        elif(predict==34):
            fpredict="Pneumonia"
        elif(predict==35):
            fpredict="Psoriasis"
        elif(predict==36):
            fpredict="Tuberculosis"
        elif(predict==37):
            fpredict="Typhoid"
        elif(predict==38):
            fpredict="Urinary tract infection"
        elif(predict==39):
            fpredict="Varicose veins"
        elif(predict==40):
            fpredict="hepatitis A"
        second_prediction=fpredict
        print(second_prediction)
```

```
        symptom_dataset =
pd.read_csv('MasterData/symptom_Description.csv', names = ['Name',
'Description'])
        symptom_Description = pd.DataFrame()
        symptom_Description['name'] = symptom_dataset['Name']
        symptom_Description['descp'] = symptom_dataset['Description']
        symptom_record =
symptom_Description[symptom_Description['name'] == second_prediction]
        print(symptom_record['descp'])
        _description=symptom_record['descp'].tolist()
        _description=_description[0]
        print(_description)
        precautionDictionary=dict()
        with open('MasterData/symptom_precaution.csv') as csv_file:
            csv_reader = csv.reader(csv_file, delimiter=',')
            line_count = 0
            for row in csv_reader:
                _prec={row[0]:[row[1],row[2],row[3],row[4]]}
                precautionDictionary.update(_prec)
        precution_list=precautionDictionary[second_prediction]
        print("Take following measures : ")
        FData=[]
        for  i,j in enumerate(precution_list):
            print(i+1,")",j)
            strdata=j
            FData.append(strdata)
        FData1=FData[0]
        FData2=FData[1]
        FData3=FData[2]
        FData4=FData[3]
        dimensionality_reduction =
training.groupby(training['prognosis']).max()
        doc_dataset = pd.read_csv('Data/doctors_dataset.csv', names =
['Name', 'Description'])
        diseases = dimensionality_reduction.index
        diseases = pd.DataFrame(diseases)
        doctors = pd.DataFrame()
        doctors['name'] = np.nan
        doctors['link'] = np.nan
        doctors['disease'] = np.nan
        doctors['disease'] = diseases['prognosis']
        doctors['name'] = doc_dataset['Name']
        doctors['link'] = doc_dataset['Description']
        record = doctors[doctors['disease'] == second_prediction]
        Ddata1=record['name'].tolist()
        Ddata1=Ddata1[0]
        Ddata2=record['link'].tolist()
        Ddata2=Ddata2[0]
```

```
        second_prediction=second_prediction
        print(record['name'])
        print(record['link'].tolist())
##        myresult=1
##        if myresult==1:
##            return
redirect(url_for('Symres',second_prediction=second_prediction,descripti
on=_description,Ddata1=Ddata1,Ddata2=Ddata2,FData1=FData1,FData2=FData2
,FData3=FData3,FData4=FData4))
    return
render_template('Symptoms.html',second_prediction=second_prediction,des
cription=_description,Ddata1=Ddata1,Ddata2=Ddata2,FData1=FData1,FData2=
FData2,FData3=FData3,FData4=FData4)
##################################>>>>>>>>>>>>>>>>>>USER<<<<<<<<<<<<<<<<
####################################################################

if __name__ == "__main__":
    app.secret_key = os.urandom(12)
    app.run(debug=True,host='0.0.0.0', port=5000)
```

Symptoms.html:

```
<html>

<head>
  <title>HEALTHCARE CHATBOT</title>
  <link rel="shortcut icon" href="img/icon.ico">

  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">

  <link rel="shortcut icon" href="../static/favicon.ico">

  <!-- Animate.css -->
  <link rel="stylesheet" href="../static/css/animate.css">
  <!-- Icomoon Icon Fonts-->
  <link rel="stylesheet" href="../static/css/icomoon.css">
  <!-- Simple Line Icons -->
  <link rel="stylesheet" href="../static/css/simple-line-icons.css">
  <!-- Bootstrap  -->
  <link rel="stylesheet" href="../static/css/bootstrap.css">
  <!-- Owl Carousel  -->
  <link rel="stylesheet" href="../static/css/owl.carousel.min.css">
```

```
  <link rel="stylesheet"
href="../static/css/owl.theme.default.min.css">
  <!-- Style -->
  <link rel="stylesheet" href="../static/css/style.css">


  <!-- Modernizr JS -->
  <script src="../static/js/modernizr-2.6.2.min.js"></script>
  <link rel="stylesheet" href="../static/style.css">
  <!-- FOR IE9 below -->
  <!--[if lt IE 9]>
  <script src="js/respond.min.js"></script>
  <![endif]-->
</head>

<body>
  <div class="panel panel-default">
    <div class="t1" align="center">
      <spa>HEALTHCARE CHATBOT</span>
    </div>

  </div>

  <!--start content area-->

  <div class="row">
    <div class="col-lg-3">

      <!-- A grey horizontal navbar that becomes vertical on small
screens -->
    </div>



    <div class="col-lg-6">
      <div class="card">
        <div class="card-header d-flex align-items-center">
          <h2 class="h5 display display">
            <h2>Symptoms Information</h2>
          </h2>
        </div>
        <div class="card-block">
          <p></p>
          <form name="form1" method="post" action="/Symptoms">
<label>Are you experiencing on</label>
            <div class="row">
              <div class="col-md-6">
                <div class="form-group">
                  <label>continuous_sneezing ? </label>
```

```
            <select name="sym1" id="sym1" class="form-control">

              <option value='yes'>yes</option>
              <option value='no'>no</option>

            </select>



          </div>
        </div>


        <div class="col-md-6">
          <div class="form-group">
            <label>fatigue ? </label>
            <select name="sym2" id="sym2" class="form-control">

              <option value='yes'>yes</option>
              <option value='no'>no</option>
            </select>

          </div>
        </div>
      </div>



      <div class="row">
        <div class="col-md-6">
          <div class="form-group">
            <label>high_fever ? </label>
            <select name="sym3" id="sym3" class="form-control">

              <option value='yes'>yes</option>
              <option value='no'>no</option>
            </select>

            </select>



          </div>
        </div>
```

```
            <div class="col-md-6">
              <div class="form-group">
                <label>muscle_pain ? </label>
                <select name="sym4" id="sym4" class="form-control">

                  <option value='yes'>yes</option>
                  <option value='no'>no</option>
                </select>
              </div>
            </div>
          </div>



          <div class="row">
            <div class="col-md-6">
              <div class="form-group">
                <label>runny_nose ? </label>
                <select name="sym5" id="sym5" class="form-control">

                  <option value='yes'>yes</option>
                  <option value='no'>no</option>
                </select>

                </select>



              </div>
            </div>


            <div class="col-md-6">
              <div class="form-group">
                <label> skin_rash ? </label>
                <select name="sym6" id="sym6" class="form-control">

                  <option value='yes'>yes</option>
                  <option value='no'>no</option>
                </select>
              </div>
            </div>
          </div>


          <div class="row">
            <div class="col-md-6">
              <div class="form-group">
                <label> joint_pain ? </label>
```

```
                <select name="sym7" id="sym7" class="form-control">

                  <option value='yes'>yes</option>
                  <option value='no'>no</option>
                </select>

                </select>



              </div>
            </div>


            <div class="col-md-6">
              <div class="form-group">
                <label> vomiting ? </label>
                <select name="sym8" id="sym8" class="form-control">

                  <option value='yes'>yes</option>
                  <option value='no'>no</option>
                </select>
              </div>
            </div>
          </div>

          <div class="row">
            <div class="col-md-6">
              <div class="form-group">
                <label>back_pain ? </label>
                <select name="sym9" id="sym9" class="form-control">

                  <option value='yes'>yes</option>
                  <option value='no'>no</option>
                </select>

                </select>



              </div>
            </div>


            <div class="col-md-6">
              <div class="form-group">
                <label> burning_micturition ? </label>
                <select name="sym10" id="sym10" class="form-control">
```

```
                    <option value='yes'>yes</option>
                    <option value='no'>no</option>
                  </select>
                </div>
              </div>
            </div>

            <div class="form-group">
              <input type="submit" name="btn" value="Submit" class="btn
btn-primary">
            </div>
            {%if second_prediction%}



            <div class="form-group">
              <p align="left" class="msg">
              <h4>You may have <b style="color:red"> {{
second_prediction }}</b></h4>
              </p>

              <p align="left" class="msg">
              <h6>{{ description }}</h6>
              </p>
              <p align="left" class="msg">
              <h6>Take following measures :</h6>
              </p>
              <p align="left" class="msg">1.) {{ FData1 }}</p>
              <p align="left" class="msg">2.) {{ FData2 }}</p>
              <p align="left" class="msg">3.) {{ FData3 }}</p>
              <p align="left" class="msg">4.) {{ FData4 }}</p>
              <p align="left" class="msg">
              <h6>Doctor Name: <b style="color:red"> {{ Ddata1
}}</b></h6>
              </p>

              <a href="{{ Ddata2 }}">{{ Ddata2 }}</a>
            </div>
            {%endif%}
          </form>
        </div>
      </div>

    </div>
    <div id="google_translate_element"></div>

    <script type="text/javascript">
      function googleTranslateElementInit() {
        new google.translate.TranslateElement(
```

```
          { pageLanguage: 'en' },
          'google_translate_element'
        );
      }
    </script>

    <script type="text/javascript"
src="https://translate.google.com/translate_a/element.js?
cb=googleTranslateElementInit">
</script>


</body>

</html>
```

7.3 System Testing

System Testing is the process of executing software in a controlled manner. Software testing is often used in association with the terms verification and validation. Validation is the checking or testing of items, includes software, for conformance and consistency with an associated specification. Software testing is just one kind of verification, which also uses techniques such as reviews, analysis, inspections, and walkthroughs. Validation is the process of checking that what has been specified is what the user actually wanted.

Software testing should not be confused with debugging. Debugging is the process of analyzing and localizing bugs when software does not behave as expected. Although the identification of some bugs will be obvious from playing with the software, a methodical approach to software testing is a much more thorough means for identifying bugs. Debugging is therefore an activity which supports testing, but cannot replace testing. Other activities which are often associated with software testing are static analysis and dynamic analysis. Static analysis investigates the source code of

software, looking for problems and gathering metrics without actually executing the code.

7.3.1 Types of Testing

Unit Testing

Unit testing focuses verification effort on the smallest unit of software design – the software component or module. Using the component level design description as a guide, important control paths are tested to uncover errors within the boundary of the module. The relative complexity of tests and uncovered scope established for unit testing. The unit testing is white-box oriented, and step can be conducted in parallel for multiple components. The modular interface is tested to ensure that information properly flows into and out of the program unit under test. The local data structure is examined to ensure that data stored temporarily maintains its integrity during all steps in an algorithm's execution. Boundary conditions are tested to ensure that all statements in a module have been executed at least once. Finally, all error handling paths are tested. Tests of data flow across a module interface are required before any other test is initiated. If data do not enter and exit properly, all other tests are moot. Selective testing of execution paths is an essential task during the unit test. Good design dictates that error conditions be anticipated and error handling paths set up to reroute or cleanly terminate processing when an error does occur. Boundary testing is the last task of unit testing step. Software often fails at its boundaries.

Integration Testing

Integration testing is systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. The objective is to take unit tested components and build a program structure that has been dictated by design. The entire program is tested as whole.

Correction is difficult because isolation of causes is complicated by vast expanse of entire program. Once these errors are corrected, new ones appear and the process continues in a seemingly endless loop. After unit testing in Sell-Soft System all the modules were integrated to test for any inconsistencies in the interfaces. Moreover differences in program structures were removed and a unique program structure was evolved.

Acceptance Testing

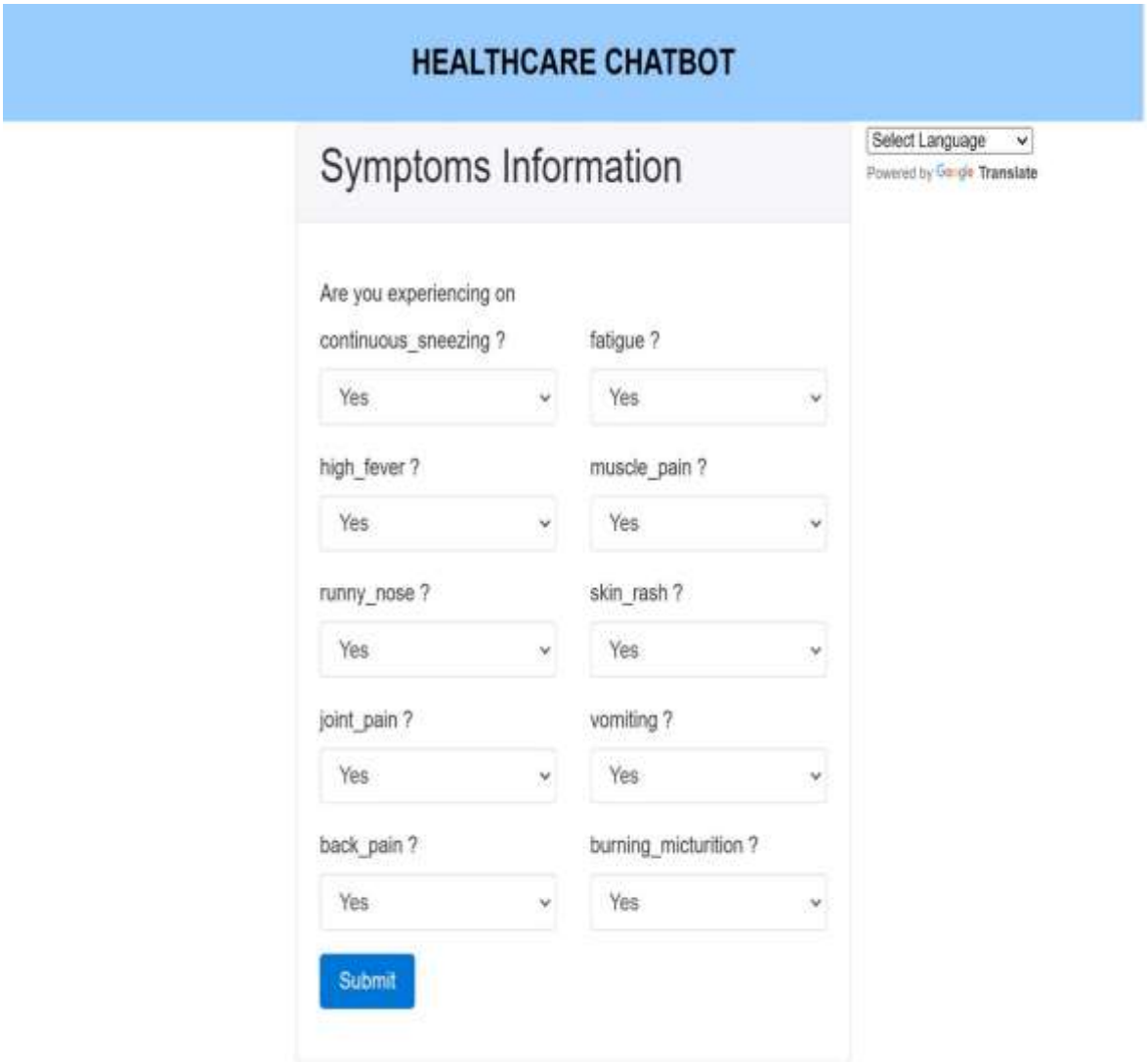
Testing generally involves running a suite of tests on the completed system. Each individual test, known as a case, exercises a particular operating condition of the user's environment or feature of the system, and will result in a pass or fail, or outcome. There is generally no degree of success or failure. The test environment is usually designed to be identical, or as close as possible, to the anticipated user's environment, including extremes of such. These test cases must each be accompanied by test case input data or a formal description of the operational activities (or both) to be performed—intended to thoroughly exercise the specific case—and a formal description of the expected results.

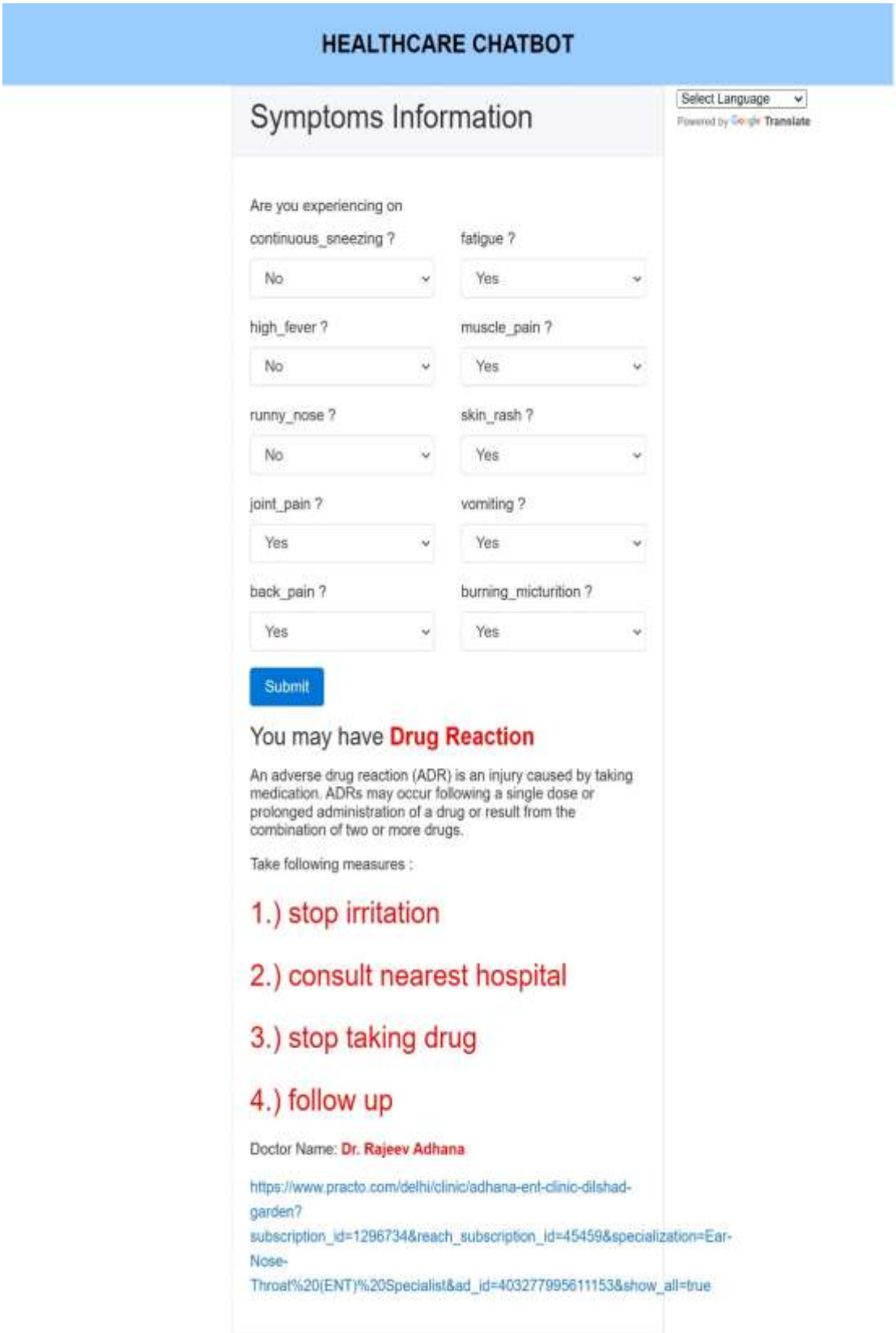
Validation Testing

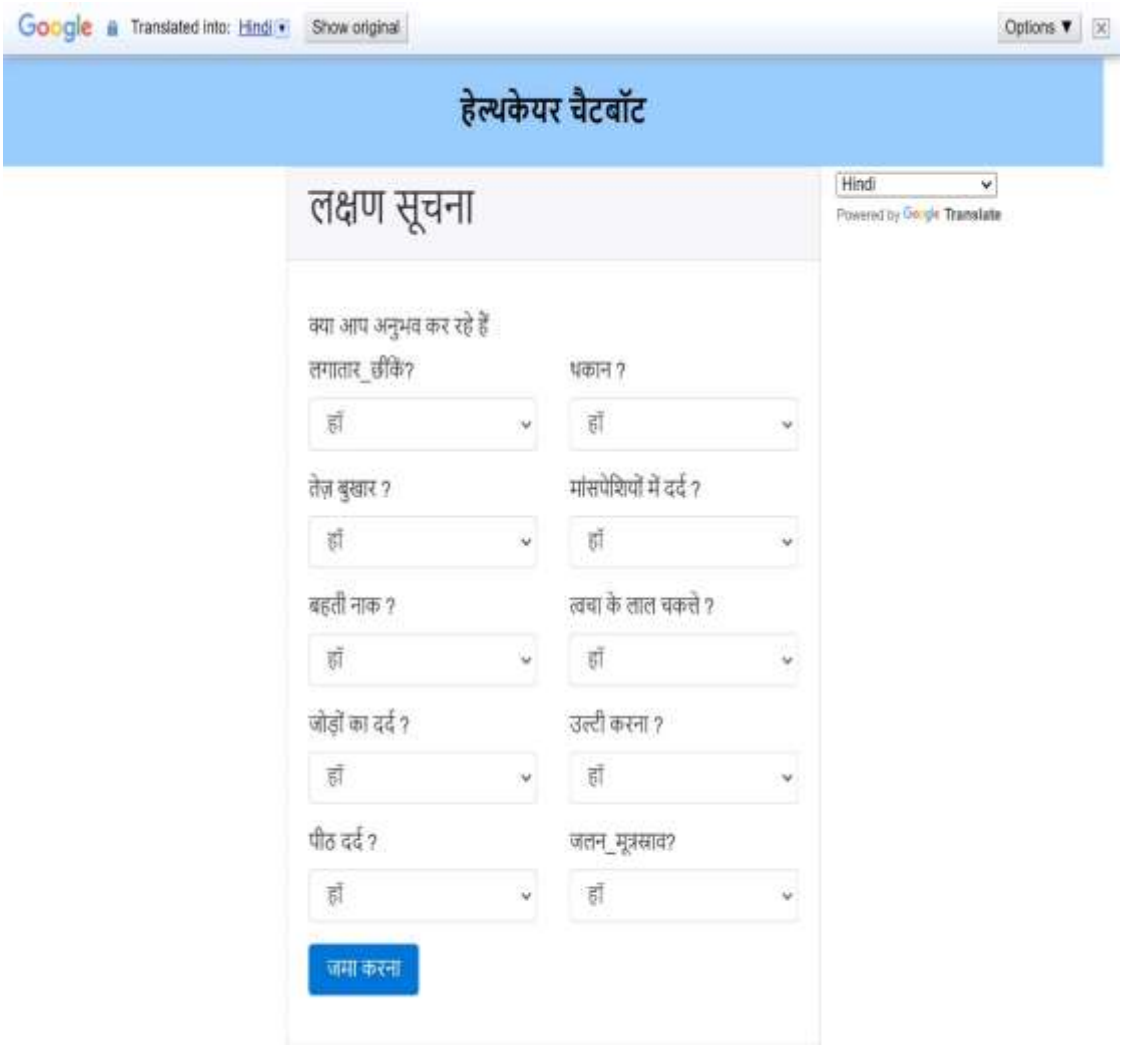
This is the final step in testing. In this the entire system was tested as a whole with all forms, code, modules and class modules. This form of testing is popularly known as Black Box testing or System testing. Black Box testing method focuses on the functional requirements of the software. That is, Black Box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program. Black Box testing attempts to find errors in the following categories; incorrect or missing functions, interface errors, errors in data structures or external data access, performance errors and initialization errors and termination errors.

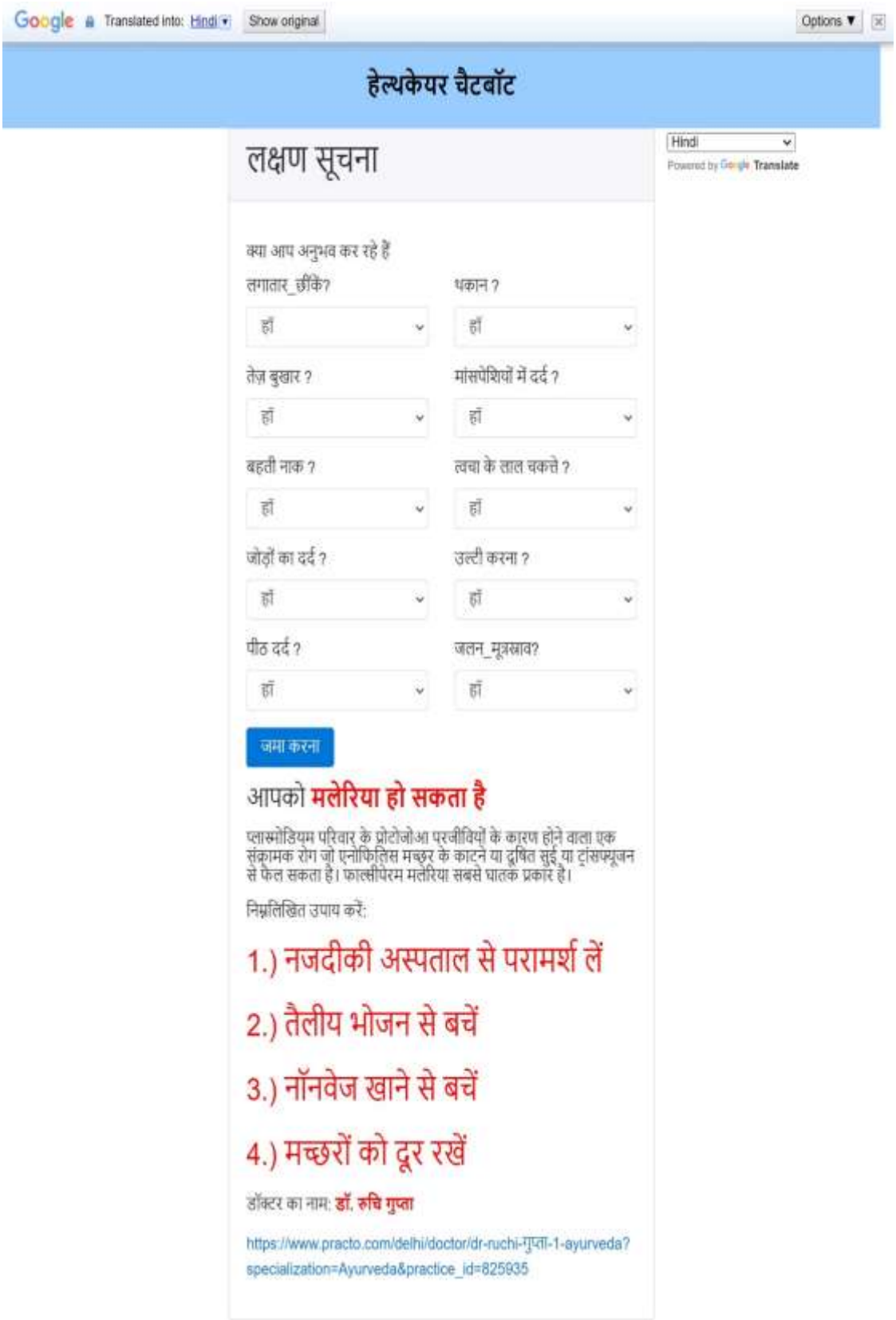
CHAPTER-8

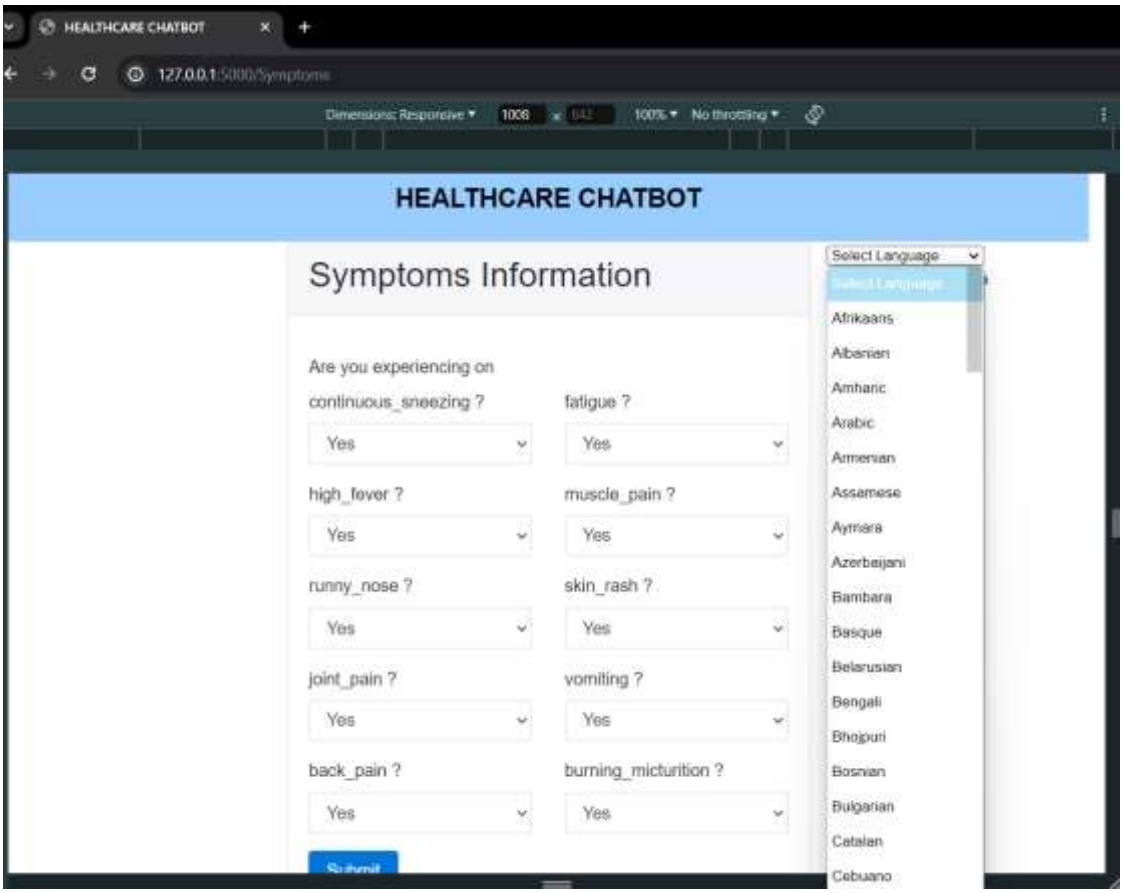
8 RESULT




HEALTHCARE CHATBOT
Symptoms Information
Select Language
Powered by Google Translate
Are you experiencing on
continuous_sneezing ?
No
fatigue ?
Yes
high_fever ?
No
muscle_pain ?
Yes
runny_nose ?
No
skin_rash ?
Yes
joint_pain ?
Yes
vomiting ?
Yes
back_pain ?
Yes
burning_micturition ?
Yes
Submit
You may have Drug Reaction
An adverse drug reaction (ADR) is an injury caused by taking medication. ADRs may occur following a single dose or prolonged administration of a drug or result from the combination of two or more drugs.
Take following measures :
1.) stop irritation
2.) consult nearest hospital
3.) stop taking drug
4.) follow up
Doctor Name: Dr. Rajeev Adhana
https://www.practo.com/delhi/clinic/adhana-ent-clinic-dilshad-garden?subscription_id=1296734&reach_subscription_id=45459&specialization=Ear-Nose-Throat%20(ENT)%20Specialist&ad_id=4032779956111153&show_all=true


Google Translated into: Hindi Show original Options
हेल्थकेयर चैटबॉट
लक्षण सूचना
Hindi
Powered by Google Translate
क्या आप अनुभव कर रहे हैं
लगातार_छींकें?
धकान ?
हाँ
तेज़ बुखार ?
मांसपेशियों में दर्द ?
बहती नाक ?
त्वचा के लाल चकत्ते ?
जोड़ों का दर्द ?
उल्टी करना ?
पीठ दर्द ?
जलन_मूत्रस्राव?
जमा करना


Google Translated into: Hindi Show original Options
हेल्थकेयर चैटबॉट
लक्षण सूचना
Hindi
Powered by Google Translate
क्या आप अनुभव कर रहे हैं
लगातार_छींकें?
थकान ?
तेज़ बुखार ?
मांसपेशियों में दर्द ?
बहती नाक ?
त्वचा के लाल चकत्ते ?
जोड़ों का दर्द ?
उल्टी करना ?
पीठ दर्द ?
जलन_मूत्रस्राव?
हाँ
जमा करना
आपको मलेरिया हो सकता है
प्लास्मोडियम परिवार के प्रोटोजोआ परजीवियों के कारण होने वाला एक संक्रामक रोग जो एनोफिलिस मच्छर के काटने या दूषित सुई या ट्रांसफ्यूजन से फैल सकता है। फाल्सीपेरम मलेरिया सबसे घातक प्रकार है।
निम्नलिखित उपाय करें:
1.) नजदीकी अस्पताल से परामर्श लें
2.) तैलीय भोजन से बचें
3.) नॉनवेज खाने से बचें
4.) मच्छरों को दूर रखें
डॉक्टर का नाम: डॉ. रुचि गुप्ता
https://www.practo.com/delhi/doctor/dr-ruchi-gupta-1-ayurveda?specialization=Ayurveda&practice_id=825935


HEALTHCARE CHATBOT
127.0.0.1:5000/Symptoms
HEALTHCARE CHATBOT
Symptoms Information
Select Language
Are you experiencing on
continuous_sneezing ?
fatigue ?
high_fever ?
muscle_pain ?
runny_nose ?
skin_rash ?
joint_pain ?
vomiting ?
back_pain ?
burning_micturition ?
Yes
Afrikaans
Albanian
Amharic
Arabic
Armenian
Assamese
Aymara
Azerbaijani
Bambara
Basque
Belarusian
Bengali
Bhojpuri
Bosnian
Bulgarian
Catalan
Cebuano

CONCLUSION

A Chatbot is a great tool for conversation. Here the application is developed to provide quality of answers in a short period of time. It removes the burden from the answer provider by directly delivering the answer to the user using an expert system. The project is developed for the user to save the user their time in consulting the doctors or experts for the healthcare solution. Here we developed the application using machine learning. Future scope of this chatbot is very vast. The smartness and intelligence of this chatbot can be increased by conducting more study and increasing the database so that Chabot could answer all type of question about every type of disease. Audio system can also be included in this system to make this Chabot more interactive.

FEATURE SCOPE

The future of AI healthcare chatbots holds promising possibilities for numerous enhancements. First and foremost, we can anticipate these chatbots evolving into more sophisticated virtual healthcare assistants, capable of providing personalized medical advice and treatment recommendations based on a patient's medical history, real-time health data, and even genetic information. Overall, the future of AI healthcare chatbots promises a revolution in healthcare delivery, offering more accessible, personalized, and efficient healthcare services to individuals while supporting healthcare professionals in their decision-making processes.

REFERENCES

[1] Hiba Hussain1, Komal Aswani2, Mahima Gupta3, Dr. G.T.Thampi4,"Implementation of Disease Prediction Chatbot and Report Analyzer using the Concepts of NLP, Machine Learning and OCR,"IRJET,Apr 2020.

[2] Oh, K.-J., D. Lee, B. Ko, and H.-J. Choi, A chatbot for psychiatric counseling in mental healthcare service based on emotional dialogue analysis and sentence generation. In 2017 18th IEEE International Conference on Mobile Data Management (MDM). IEEE, 2017.

[3] Kowatsch, T., M. Nißen, C.-H. I. Shih, D. Rüegger, D. Volland, A. Filler, F. Künzler, F. Barata, D. Büchter, B. Brogle, et al. (2017). Text-based healthcare chatbots supporting patient and health professional teams: preliminary results of a randomized controlled trial on childhood obesity.

[4] Lin Ni(B), Chenhao Lu, Niu Liu, and Jiamou Liu," MANDY: Towards a Smart Primary Care Chatbot Application", SPRINGER,2017. [5] Divya, S., V. Indumathi, S. Ishwarya, M. Priyasankari, and S. K. Devi (2018). A self-diagnosis medical chatbot using artificial intelligence. Journal of Web Development and Web Designing, 3(1), 1–7.

[6] Chung, K. and R. C. Park (2019). Chatbot-based heathcare service with a knowledge base for cloud computing. Cluster Computing, 22(1), 1925–1937.

[7] Ahmed Fadil, Gianluca Schiavo,"Design for healthcare chatbot"Arxiv, 2019.

[8] Beaudry, J., A. Consigli, C. Clark, and K. J. Robinson (2019). Getting ready for adult healthcare: Designing a chatbot to coach adolescents with special health needs through the transitions of care. Journal of pediatric nursing, 49, 85–91.

[9] Kavitha, B. and C. R. Murthy (2019). Chatbot for healthcare system using artificial intelligence.

[10] Kandpal, P., K. Jasnani, R. Raut, and S. Bhorge, Contextual chatbot for healthcare purposes (using deep learning). In 2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4). IEEE, 2020.

[11] Vaibhav Tode, Himanshu Gadge, Prateek Kachare and Sudarshan Madane," CureBot -An Artificially Intelligent Interactive Bot for Medical Diagnostics" International Research journal of Engineering and Technology (IRJET)., Vol.7, no.12 (Dec 2020).

[12] Satyendra Praneel Reddy Karri and Dr Santosh kumar," Deep Learning Techniques for Implementation of Chatbots", 2020 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, INDIA, (2020) January 22-24.

[13] Dipesh Kadariya, Revathy Venkataramanan, Hong Yung Yip, Maninder Kalra, Krishna Prasad Thirunarayanan and Amit Sheth, "kBot: Knowledge-Enabled personalized Chatbot for Asthma Self-Management", 2019 IEEE International Conference on Smart Computing (SMARTCOMP), Washington, DC, USA,(2019) June 12-15

[14] Katlego Mabunda and Abejide Ade-Ibijola," PathBot: An Intelligent Chatbot for Guiding Visitors and Locating Venues", 6th International Conference on Soft

Computing & Machine Intelligence (ISCMI), Johannesburg Parktonian All-Suite, South Africa,(2019) November 19-20.

[15] Ali Bou Nassif, Ismail Shahin, Imtinan Attili and Mohammad Azzeh," Speech Recognition Using Deep Neural Networks: A Systematic Review"IEEE-Access, Vol. 7, (2019), pp(99).1-1.

[16] Connor Shorten, Taghi M. Khoshgoftaar and Borko Furht," Deep Learning applications for COVID-19", SpringerOpen J Big Data 8, Article no8, (Jan 2021).

[17] A. F. Ur Rahman Khilji, S. R. Laskar, P. Pakray, R. A. Kadir, M. S. Lydia and S. Bandy-opadhyay, "HealFavor: Dataset and A Prototype System for Healthcare ChatBot," 2020 International Conference on Data Science, Artificial Intelligence, and Business Analytics (DATABIA), 2020.