

MEDICAL ALERT CHATBOT

A PROJECT REPORT

Submitted

In the partial fulfilment of the requirements for

The award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

M. DHAMODHARAN 19001A0513

B. VIJAYA LAKSHMI 19001A0514

Y. PRIYANKA 19001A0523

Under the guidance of

Mr. G. ESWAR M.Tech.,

Assistant Professor (Adhoc)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR
COLLEGE OF ENGINEERING**

(Autonomous)

ANANTHAPURAMU - 515002

ANDHRA PRADESH

2023

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

COLLEGE OF ENGINEERING (*AUTONOMOUS*)

ANANTHAPURAMU – 515002

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

This is to certify that the project entitled “**MEDIAL ALERT CHATBOT**” is a bonafide work of **M DHAMODHARAN**, bearing Admission No: **19001A0513** is submitted to the faculty of Computer Science & Engineering, in partial fulfilment of the requirements for the award of degree of **BACHELOR OF TECHNOLOGY** in COMPUTER SCIENCE AND ENGINEERING from Jawaharlal Nehru Technological University Anantapur, College of Engineering (Autonomous), Anantapuramu.

SIGNATURE OF GUIDE

Mr. G. Eswar M.Tech.

Assistant professor (Ad-hoc)

Department of CSE,

JNTUA College of Engineering

ANANTHAPURAMU-515002

SIGNATURE OF HOD

Dr. K.F. BHARATI, M.Tech, Ph.D

Head of CSE Dept.

Department of CSE,

JNTUA College of Engineering

ANANTHAPURAMU -515002

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned our efforts with success.

We are highly indebted to **G. Eswar** M.Tech., Assistant Professor (Adhoc), Computer Science and Engineering, JNTUA College of Engineering, Ananthapuramu for his guidance and constant supervision as well as for providing necessary information regarding the project. His support has helped us in the completion of the project.

We would like to express special thanks to **Dr. K.F. Bharati**, Head of the Department, Computer Science and Engineering, JNTUA College of Engineering, Ananthapuramu. Her wide support, knowledge and enthusiastic encouragement has inspired us to get better involved with our project and technical design.

We want to express our gratitude to **Prof. P. Sujatha**, Principal of JNTUA College of Engineering Ananthapuramu, for her co-operation and her timely help in the successful completion of the project.

We express our sincere thanks to the project committee members, faculty and staff of Computer Science and Engineering, JNTUA College of Engineering, Ananthapuramu, for their valuable guidance and technical support. Last but far from least, we also thank our family members and our friends for their moral support and constant encouragement, we are very thankful to one and all who helped us for the successful completion of the project.

With Gratitude,

M DHAMODHARAN (19001A0513)

B VIJAYA LAKSHMI (19001A0514)

Y PRIYANKA (19001A0523)

ABSTRACT

Healthcare is very important to lead a good life. However, it is very difficult to obtain the consultation with the doctor for every health problem. The idea is to create a medical chatbot using Machine Learning that can diagnose the disease and provide basic details about the disease before consulting a doctor. This will help to reduce healthcare costs and improve accessibility to medical knowledge through medical chatbot. The chatbots are computer programs that use natural language to interact with users.

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 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. Chat bots typically provide a text-based user interface, allowing the user to type commands, and it will not remind the user to take medicines on time.

A health care chatbot is a type of chatbot that is designed to provide information and assistance related to health and medical topics. These chatbots can be programmed to provide a wide range of information, including general health and wellness information, information about specific medical conditions and treatments, and reminding the patient to take the medicines on time. They can be a useful resource for individuals seeking information or guidance about their health, but it is important to keep in mind that they are not a substitute for professional medical care and should not be used to diagnose or treat medical conditions.

CONTENTS

ACKNOWLEDGEMENT

ABSTRACT

CHAPTER 1- 7-11

OBJECTIVE

1.1 INTRODUCTION

1.2 SCOPE

CHAPTER 2- 12-15

SYSTEM ANALYSIS & FEASABILITY STUDY

2.1 EXISTING SYSTEM

2.2 PROPOSED SYSTEM

2.3 WORKING OF AN AI CHATBOT

2.4 BLOCK DIAGRAM

2.5 ARCHITECTURE

CHAPTER 3- 16-23

ALGORITHMS USED

3.1 DECISION TREE CLASSIFIER

3.2 SUPPORT VECTOR MACHINE CLASSIFIER

3.3 NATURAL LANGUAGE PROCESSING

CHAPTER 4- 24-28

SOFTWARE REQUIREMENTS SPECIFICATION

4.1 FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS

4.2 SYSTEM SPECIFICATIONS (HARDWARE & SOFTWARE)

4.3 SYSTEM DESIGN (INPUT DESIGN & OUTPUT DESIGN)

CHAPTER 5-	29
MODULES	
5.1 DATA PRE-PROCESSING	
5.2 LABEL ENCODING	
5.3 SPLITTING THE DATA	
5.4 MODEL FITTING	
CHAPTER 6-	30-33
UML DIAGRAMS	
6.1 USECASE DIAGRAM	
6.2 SEQUENCE DIAGRAM	
6.3 ACTIVITY DIAGRAM	
CHAPTER 7-	34-38
SYSTEM IMPLEMENTATION	
CHAPTER 8-	39-41
USER INTERFACE	
CHAPTER 9-	42-45
SYSTEM TESTING	
9.1 SYSTEM TEST	
9.2 TYPES OF TESTS	
9.3 TEST CASES	
CHAPTER 10-	46
CONCLUSION	
CHAPTER 11-	47
REFERENCES	

LIST OF FIGURES

FIG NO.	FIGURE NAME	PAGE NO.
2.1	WORKING OF AN AI CHATBOT	14
2.2	WORKFLOW OF THE SYSTEM	14
2.3	RECOGNIZING USER INPUT	15
3.1	SVM	18
3.2	HYPER PLANES AND SUPPORT VECTORS	19
3.3	WATERFALL MODEL	22
6.1	USECASE DIAGRAM	31
6.2	SEQUENCE DIAGRAM	32
6.3	ACTIVITY DIAGRAM	33
7.1	IMPORTING PACKAGES	34
7.2	DATA PRE-PROCESSING	36
7.3	LABEL ENCODING	37
7.4	SPLITTING THE DATA	37
7.5	MODEL FITTING	38
8.1	USER INTERFACE	39
8.2	WORKING OF CHATBOT	40
8.3	PROCESSING THE INPUT	40
8.4	RESPONSE GENERATION	41

LIST OF TABLES

TABLE NO.	TABLE NAME	PAGE NO.
9.1	TEST CASES	45
9.2	TEST CASES MODEL BUILDING	45

CHAPTER 1

OBJECTIVE

The objective of a medical alert chatbot project is to develop a conversational artificial intelligence (AI) system that provides timely and accurate medical information, guidance, and assistance to users in emergency situations or when they require immediate medical attention.

INTRODUCTION

Medical alert chatbots are virtual assistants designed to provide assistance and support for individuals who may require medical attention or have medical concerns. These chatbots are powered by artificial intelligence (AI) and are programmed to provide accurate and reliable information about various health-related topics, offer advice on medical emergencies, and help users navigate through medical situations with guidance and recommendations.

The primary goal of a medical alert chatbot is to provide prompt and accurate information to help users make informed decisions about their health. They can provide information on common medical conditions, symptoms, first aid measures, and emergency response procedures. They can also offer reminders for taking medications, scheduling appointments, and maintaining a healthy lifestyle.

However, it's important to note that medical alert chatbots are not a substitute for professional medical advice. They are designed to provide information and support, but users should always consult a qualified healthcare provider for any medical concerns. Medical alert chatbots do not provide diagnosis or treatment recommendations, but they can help users find relevant information and resources to better understand their health conditions and make informed decisions about their healthcare.

The purpose of a medical alert chatbot is to provide reliable, convenient, and accessible information to assist users in making informed decisions about their health and well-being. They are available 24/7 to answer questions, provide support, and offer guidance on medical matters. However, it's important to keep in mind that they are AI-

powered chatbots and not human medical professionals, and their responses are based on the information available to them at the time of the conversation.

Now a days, health care is extremely necessary in our life. Today's people are busy with their works reception, workplace works and additional addicted to web. They are not involved regarding their health. So they avoid to travel in hospitals for little issues. it may become a significant drawback. So, we will offer a thought is to make a health care chatbot system using AI that may identification the illness and supply basic information regarding the illness before consulting a doctor. Which helps the patients apprehend additional regarding their illness and improves their health. User can do the all reasonably illness information.

The system application uses question and answer protocol within the style of chatbot to answer user queries. The response to the question is replied supported the user question. The significant keywords are fetched from the sentence and answer to those sentences. If match is discovered or vital answer are given, or similar answers are displayed can identification which sort of illness you have got supported user symptoms and additionally offers doctor details of explicit illness. It may cut back their health problems by victimization this application system.

The system is developed to scale back the tending price and time of the users because it isn't potential for the users to go to the doctors or consultants once in real time required.

If a patient said, "my head hurts", Eliza would respond, "why do you say your head hurts?". But Eliza was not a doctor, or even a person. It was a program – 200 lines of experimental code written 50 years ago at MIT. This program was proof that it's possible to simulate human conversation using so Eliza was the first chatbot ever created. Chatbots are a new type of user interface which takes advantage of artificial intelligence software that can answer questions, converse, and assist us in a way that mimics human conversation.

The first chatbot simulated a psychotherapist, but it wasn't very human. In 2017, however, you can create a doctor-like chatbot that can pre-diagnose your symptoms before you visit a real doctor.

In conclusion, medical alert chatbots are valuable tools that can provide helpful information and support for individuals with medical concerns. They can assist users in finding accurate and reliable information about their health, but they should not be relied upon as a substitute for professional medical advice.

SCOPE

Healthcare and chatbots:

The use of chatbots has spread from consumer customer service to matters of life and death. Chatbots are entering the healthcare industry and can help solve many of its problems.

Information and Education:

The medical alert chatbot can provide accurate and reliable information about various health topics, such as common medical conditions, symptoms, first aid measures, and emergency response procedures. It can educate users about healthy lifestyle practices, medication management, and preventive care.

Emergency Assistance:

The chatbot can offer guidance on what to do in case of a medical emergency, such as providing step-by-step instructions for performing CPR, managing choking incidents, or handling allergic reactions. It can also provide information on the nearest hospitals, urgent care centres, or emergency services.

Medication Reminders:

The chatbot can send reminders to users to take their medications on time, based on their medication schedule and dosage instructions. It can help users stay compliant with their prescribed medications and reduce the risk of medication-related issues.

Symptom Assessment:

The chatbot can assist users in evaluating their symptoms and provide general information on potential causes, severity, and recommended next steps. It can help users decide whether they need to seek immediate medical attention or take self-care measures.

Referral to Healthcare Providers:

The chatbot can provide users with recommendations for healthcare providers based on their symptoms, location, and preferences. It can help users find appropriate medical care based on their needs and facilitate referrals to healthcare providers.

CHAPTER 2

SYSTEM ANALYSIS & FEASIBILITY STUDY

Existing system:

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 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. Chat bots typically provide a text-based user interface, allowing the user to type commands, and it will not remind the user to take medicines on time.

Proposed system:

These chatbots can be programmed to provide a wide range of information, including general health and wellness information, information about specific medical conditions and treatments, and reminding the patient to take the medicines on time. They can be a useful resource for individuals seeking information or guidance about their health, but it is important to keep in mind that they are not a substitute for professional medical care and should not be used to diagnose or treat medical conditions.

24 Hours Service

Getting Instant Response

Easy Communication

Detailed/Expert Answers

Answers To Complex Questions

Friendliness And Approachability

Improved Access To Care

Increased Convenience For Patients

Cost Savings

Working of an AI chatbot:

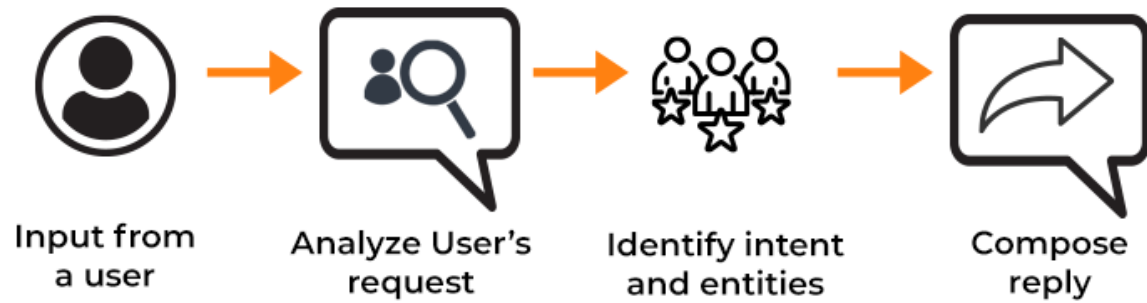


Fig 2.1: Working of an AI chatbot

Block diagram:

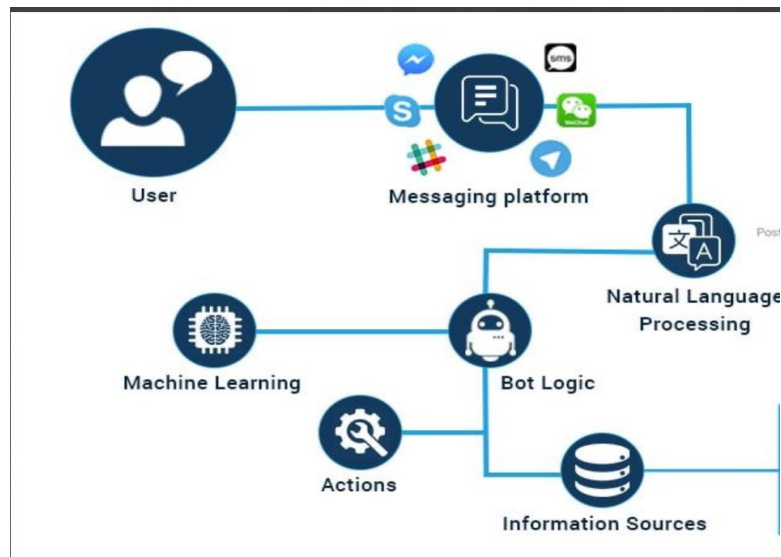


Fig 2.2: Workflow of the system

Architecture:

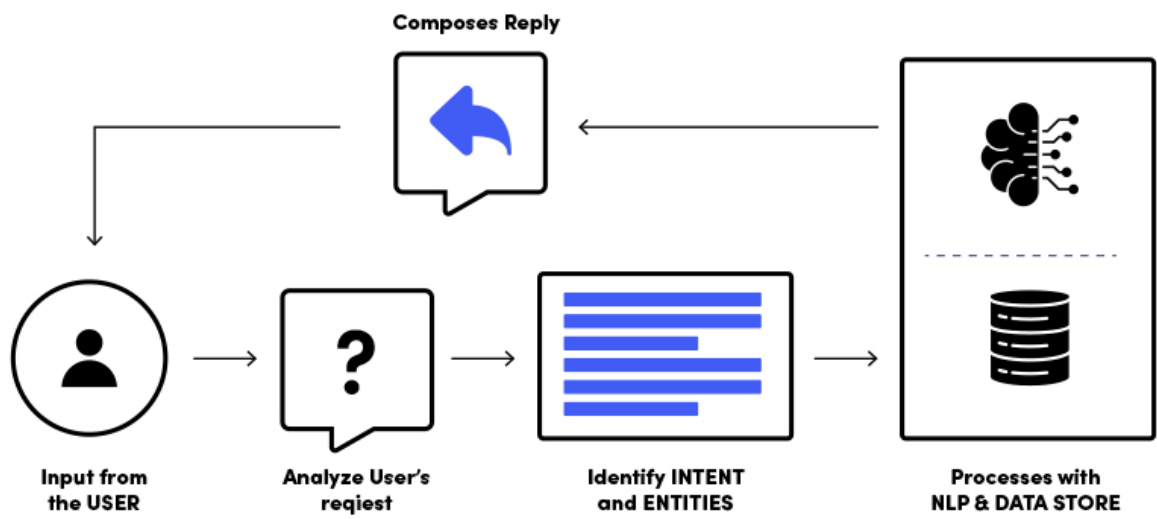


Fig 2.3: Recognizing user input

CHAPTER 3

ALGORITHMS USED:

1. DECISION TREE CLASSIFIER:

A decision tree classifier is a type of supervised machine learning algorithm that can be used for classification tasks. In the context of a medical alert chatbot project, the decision tree classifier can be used to predict the urgency or severity of a medical condition based on user input or symptoms.

Here's a high-level overview of how we could integrate a decision tree classifier algorithm in this project:

Data Collection and Preparation:

Gather a labeled dataset of medical cases, where each case includes relevant features such as symptoms, age, gender, medical history, and corresponding labels indicating the urgency or severity of the condition (e.g., low, medium, high urgency). This dataset will be used to train and evaluate the decision tree classifier.

Feature Engineering:

Analyse the collected data and identify relevant features that could be used as input for the decision tree classifier. This may involve converting categorical variables into numerical representations or normalizing numeric features.

Model Training:

Split the dataset into training and validation sets. Use the training set to train the decision tree classifier. During training, the algorithm will learn to identify patterns in the input features that correlate with the urgency or severity of the medical condition.

Model Evaluation:

Evaluate the trained decision tree classifier on the validation set to assess its performance. Common evaluation metrics for classification tasks include accuracy,

precision, recall, and F1 score. If the performance is not satisfactory, you may need to fine-tune the model or collect more data.

Model Integration:

Once the decision tree classifier is trained and evaluated, integrate it into the chatbot system. The chatbot can prompt users for relevant information, such as symptoms or medical history, and use the trained decision tree classifier to predict the urgency or severity of the condition based on the input provided.

Deployment and Testing:

Deploy the chatbot system with the integrated decision tree classifier in a controlled environment and conduct thorough testing to ensure its functionality and accuracy. Monitor the system during real-world usage and collect feedback to further improve its performance.

Regular Updates:

Keep the decision tree classifier updated with the latest medical data and continuously monitor and evaluate its performance. Update the chatbot system as needed to ensure it remains accurate and effective in providing medical alerts.

2. SUPPORT VECTOR MACHINE CLASSIFIER

The objective of the support vector machine algorithm is to find a hyper plane in an N-dimensional space (N — the number of features) that distinctly classifies the data points.

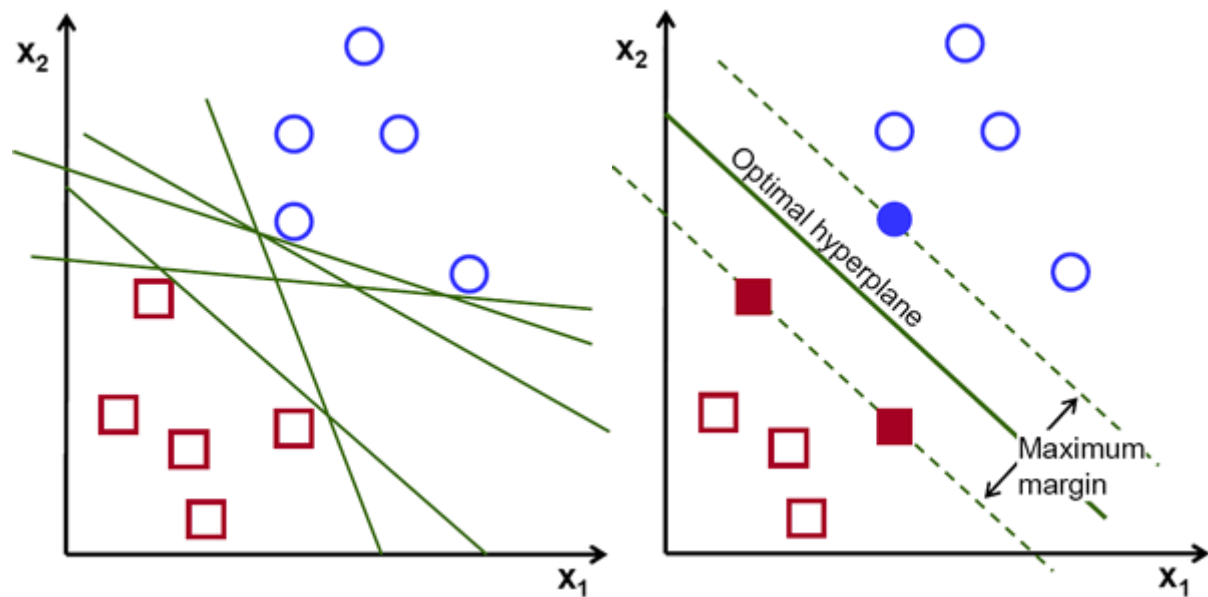
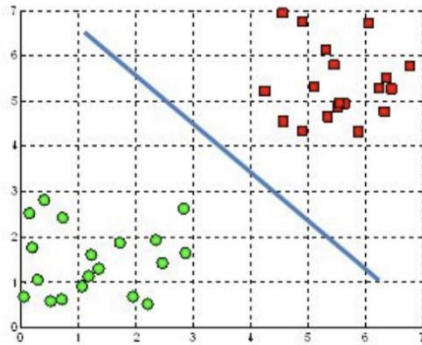


Fig 3.1: SVM

Possible hyper planes:

To separate the two classes of data points, there are many possible hyper planes that could be chosen. Our objective is to find a plane that has the maximum margin, i.e. the maximum distance between data points of both classes. Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence.

A hyperplane in \mathbb{R}^2 is a line



A hyperplane in \mathbb{R}^3 is a plane

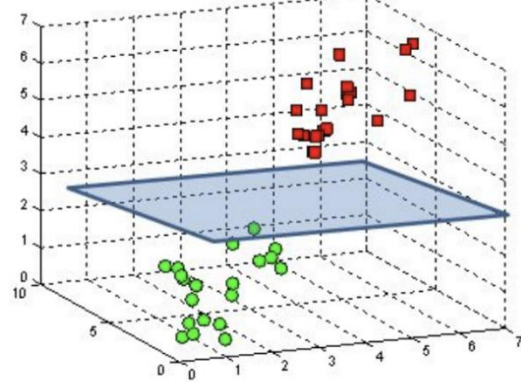


Fig 3.2: Hyper planes and Support Vectors

Hyper planes in 2D and 3D feature space:

Hyper planes are decision boundaries that help classify the data points. Data points falling on either side of the hyper plane can be attributed to different classes. Also, the dimension of the hyper plane depends upon the number of features. If the number of input features is 2, then the hyper plane is just a line. If the number of input features is 3, then the hyper plane becomes a two-dimensional plane. It becomes difficult to imagine when the number of features exceeds 3.

3. NATURAL LANGUAGE PROCESSING

A medical alert chatbot is a virtual assistant that uses natural language processing (NLP) techniques to provide timely and accurate health-related information, offer support, and generate alerts in case of medical emergencies. NLP is a critical component of such a project, as it enables the chatbot to understand and respond to user inputs in a human-like manner. Here are some key aspects of NLP in a medical alert chatbot project:

Intent recognition:

NLP algorithms are used to identify the intent of user queries or messages. For example, the chatbot needs to understand if the user is asking for information about a specific medical condition, requesting assistance for a medical emergency, or seeking general health advice. Intent recognition is crucial for the chatbot to provide relevant and accurate responses.

Entity extraction:

NLP algorithms are used to extract relevant entities from user queries or messages. Entities are specific pieces of information, such as medical conditions, symptoms, medications, or locations, that are important for understanding the context and providing appropriate responses. For example, if a user mentions chest pain as a symptom, the chatbot needs to extract "chest pain" as an entity to provide relevant information and generate appropriate alerts.

Contextual understanding:

NLP techniques are used to analyze the context of user queries or messages to provide accurate responses. For example, the chatbot needs to understand the severity of a symptom, the medical history of the user, and any other relevant information to offer appropriate advice or generate alerts. Contextual understanding is crucial for tailoring the responses of the chatbot to the specific needs and situation of the user.

Language generation:

NLP algorithms are used to generate human-like responses to user queries or messages. The chatbot needs to provide information in a clear and understandable manner, using appropriate medical terminology and language that is suitable for the user's level of health literacy. Language generation techniques, such as template-based responses, rule-based responses, or machine learning-based responses, can be used to generate responses that are relevant, accurate, and easy to understand.

Emergency alert generation:

In case of a medical emergency, the chatbot needs to generate alerts and notify appropriate parties, such as emergency services or designated contacts of the user. NLP algorithms can be used to analyze the user's messages or queries for keywords or patterns that indicate a medical emergency, and trigger an alert generation process accordingly. This can include sending notifications through various communication channels, such as SMS, email, or phone calls, to ensure timely and effective response.

Privacy and security:

NLP techniques need to be used in compliance with privacy and security regulations, such as HIPAA (Health Insurance Portability and Accountability Act) in the United States, to ensure the confidentiality and integrity of user data. This includes data encryption, access controls, and proper handling of sensitive health information.

Overall, NLP plays a critical role in the development and operation of a medical alert chatbot project, enabling the chatbot to understand user queries, provide relevant information, generate alerts in case of emergencies, and ensure a user-friendly and effective interaction experience. Proper implementation of NLP techniques can greatly enhance the accuracy, efficiency, and usability of a medical alert chatbot, leading to improved health outcomes for users.

SOFTWARE DEVELOPMENT LIFE CYCLE – SDLC:

In our project we use waterfall model as our software development cycle because of its step-by-step procedure while implementing.

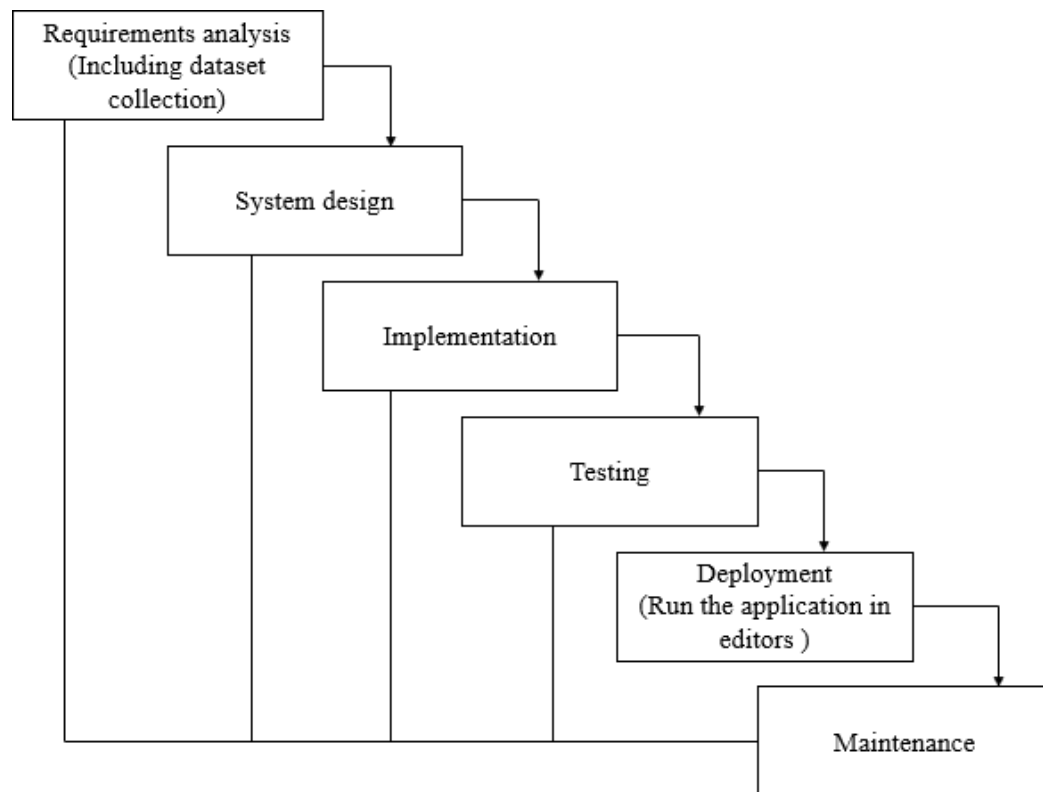


Fig 3.3: Waterfall Model

❖ **Requirement Gathering and analysis:**

All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.

❖ **System Design:**

The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.

❖ **Implementation:**

With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.

❖ **Integration and Testing:**

All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

❖ **Deployment of system:**

Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.

❖ **Maintenance:**

There are some issues which come up in the client environment. To fix those issues, patches are released. Also, to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

CHAPTER 4

SYSTEM REQUIREMENTS SPECIFICATION

Functional and non-functional requirements:

Requirement's analysis is very critical process that enables the success of a system or software project to be assessed. Requirements are generally split into two types: Functional and non-functional requirements.

Functional Requirements:

Functional requirements are product features or functions that developers must implement to enable users to accomplish their tasks. So, it's important to make them clear both for the development team and the stakeholders. Generally, functional requirements describe system behaviour under specific conditions.

Functional requirements for a medical alert chatbot project refer to the features, capabilities, and behaviours that the chatbot should possess in order to fulfil its intended purpose of providing medical alerts. Here are some potential functional requirements for a medical alert chatbot:

- Alert generation
- Natural language understanding
- Medical information retrieval
- Decision support
- User profiling
- Multichannel support
- Context awareness
- User feedback and learning
- Privacy and security

Non-functional requirements:

These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non-behavioural requirements.

They basically deal with issues like:

- Portability
- Security
- Maintainability
- Reliability
- Scalability
- Performance
- Reusability
- Flexibility

SYSTEM SPECIFICATIONS:

HARDWARE SPECIFICATIONS:

- Processor - I3/Intel Processor
- RAM - 8GB (min)
- Hard Disk - 128 GB
- Keyboard - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - Any

SOFTWARE SPECIFICATIONS:

- Operating System : Windows 10/11
- Server-side Script : Python, HTML, CSS
- IDE : Visual Studio Code
- Libraries Used : Pandas, NumPy, Sklearn, re, time

SYSTEM DESIGN:

Input Design:

In an information system, input is the raw data that is processed to produce output. During the input design, the developers must consider the input devices such as PC, MICR, OMR, etc.

Therefore, the quality of system input determines the quality of system output. Well-designed input forms and screens have following properties –

- It should serve specific purpose effectively such as storing, recording, and retrieving the information.
- It ensures proper completion with accuracy.
- It should be easy to fill and straightforward.
- It should focus on user's attention, consistency, and simplicity.
- All these objectives are obtained using the knowledge of basic design principles regarding –
 - What are the inputs needed for the system?
 - How end users respond to different elements of forms and screens.

Objectives for Input Design:

The objectives of input design are –

- To design data entry and input procedures
- To reduce input volume
- To design source documents for data capture or devise other data capture methods
- To design input data records, data entry screens, user interface screens, etc.
- To use validation checks and develop effective input controls.

Output Design:

The design of output is the most important task of any system. During output design, developers identify the type of outputs needed, and consider the necessary output controls and prototype report layouts.

Objectives of Output Design:

The objectives of input design are:

- To develop output design that serves the intended purpose and eliminates the production of unwanted output.
- To develop the output design that meets the end user's requirements.
- To deliver the appropriate quantity of output.
- To form the output in appropriate format and direct it to the right person.
- To make the output available on time for making good decisions.

CHAPTER 5

MODULES:

The modules the system contains are:

1. Data pre-processing
2. Label encoding
3. Splitting the data
4. Model fitting

Data pre-processing:

Data pre-processing is a data mining technique which is used to transform the raw data in a useful and efficient format.

Label encoding:

Label Encoding is a popular encoding technique for handling categorical variables. In this technique, each label is assigned a unique integer based on alphabetical ordering.

Splitting the data:

The train-test split procedure is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model.

Model fitting:

Model fitting is a measure of how well a machine learning model generalizes to similar data to that on which it was trained. A model that is well-fitted produces more accurate outcomes. A model that is overfitted matches the data too closely. A model that is underfitted doesn't match closely enough.

CHAPTER 6

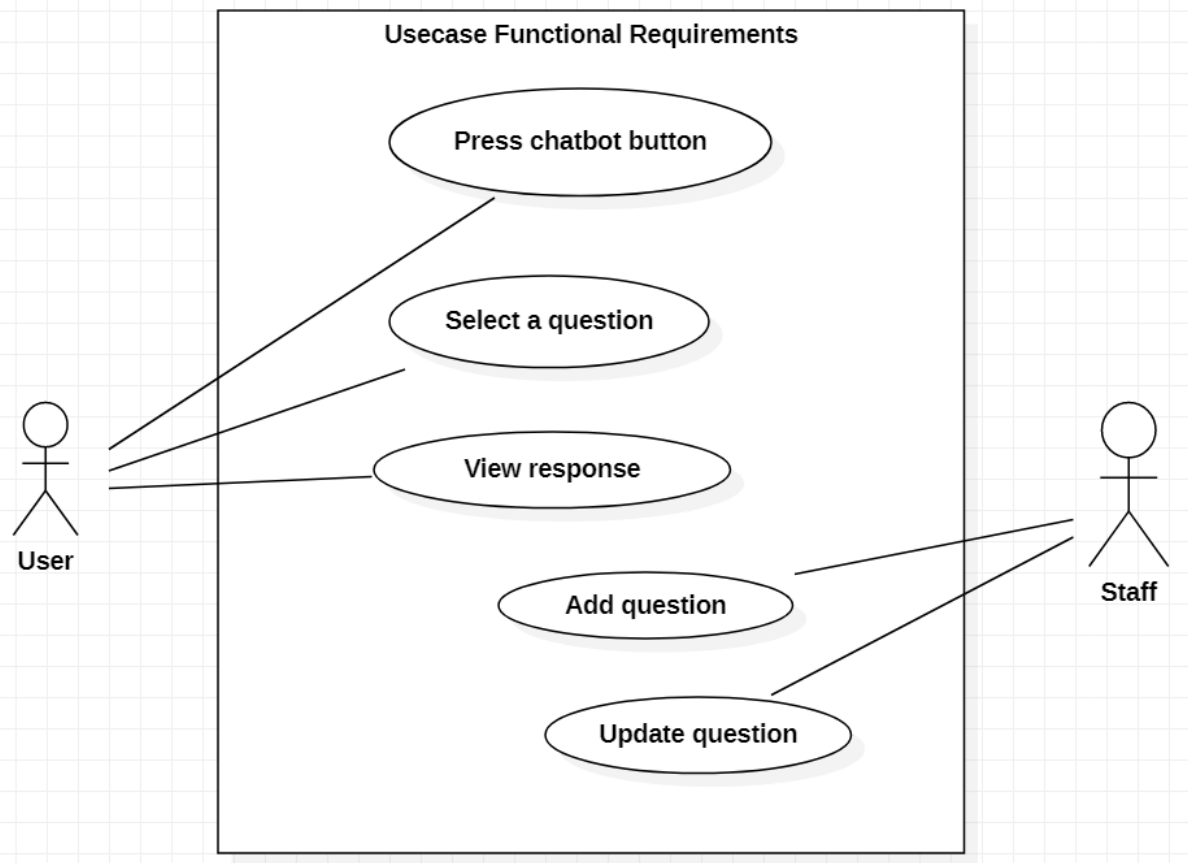
UML DIAGRAMS

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 goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

6.1 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.



6.1 Usecase Diagram

6.2 Sequence diagram:

The sequence diagram represents the flow of messages in the system and is also termed as an event diagram. It helps in envisioning several dynamic scenarios. It portrays the communication between any two lifelines as a time-ordered sequence of events, such that these lifelines took part at the run time.

In UML, the lifeline is represented by a vertical bar, whereas the message flow is represented by a vertical dotted line that extends across the bottom of the page. It incorporates the iterations as well as branching.

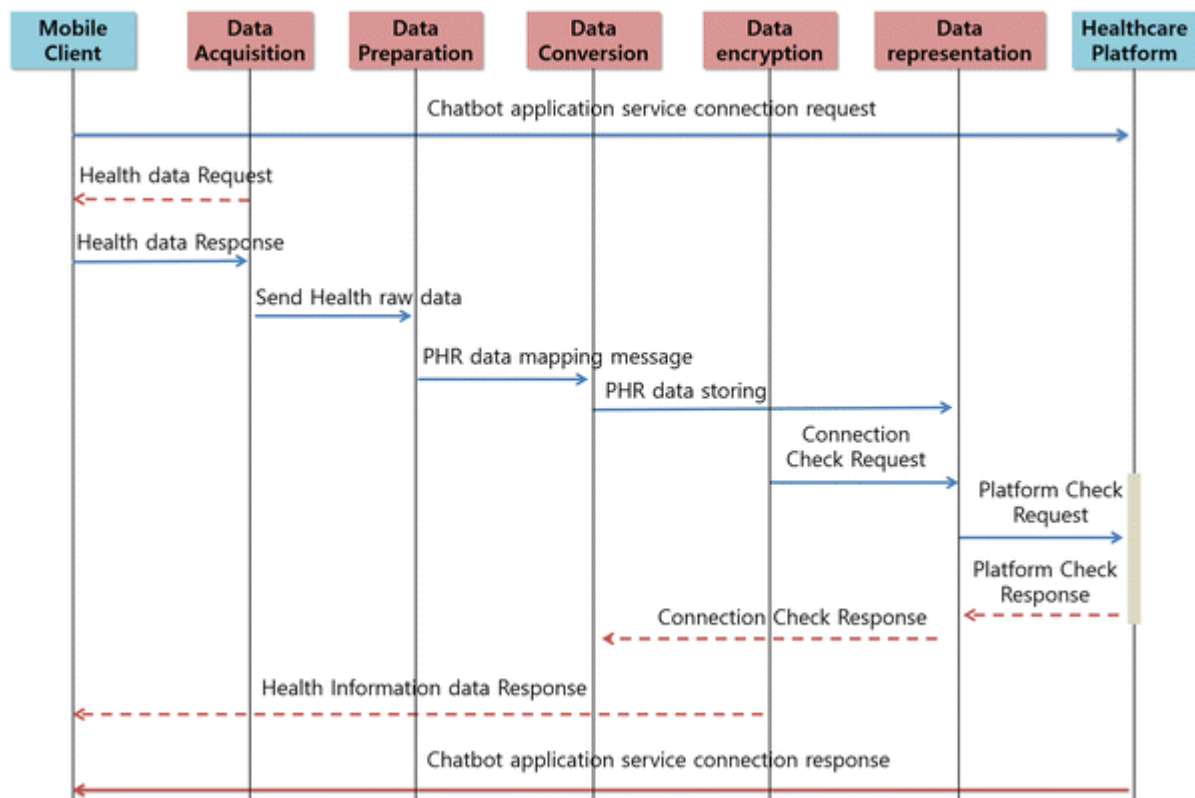


Fig 6.2 Sequence Diagram

6.3 Activity diagram:

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system.

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc.,

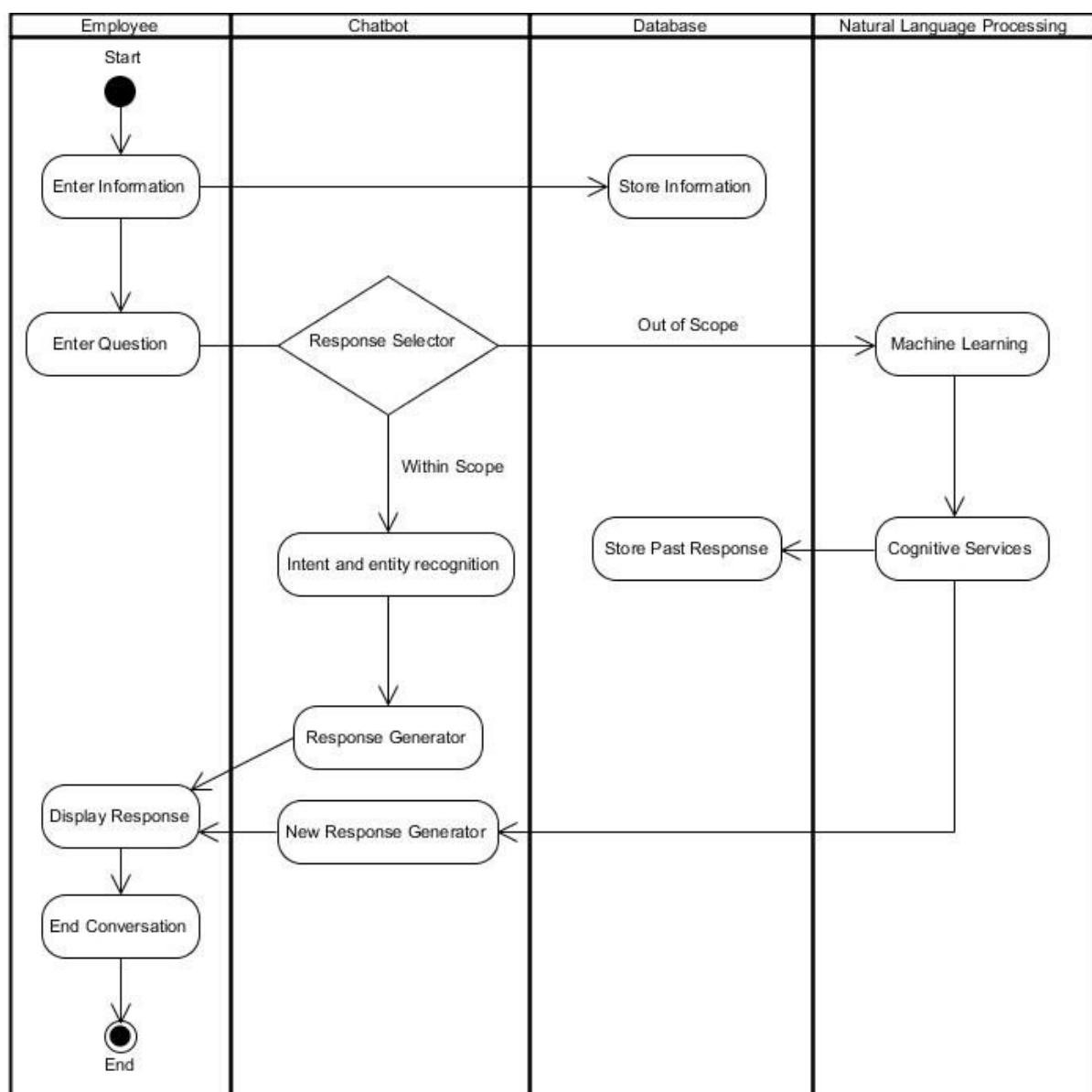


Fig 6.3 Activity Diagram

CHAPTER 7

SYSTEM IMPLEMENTATION

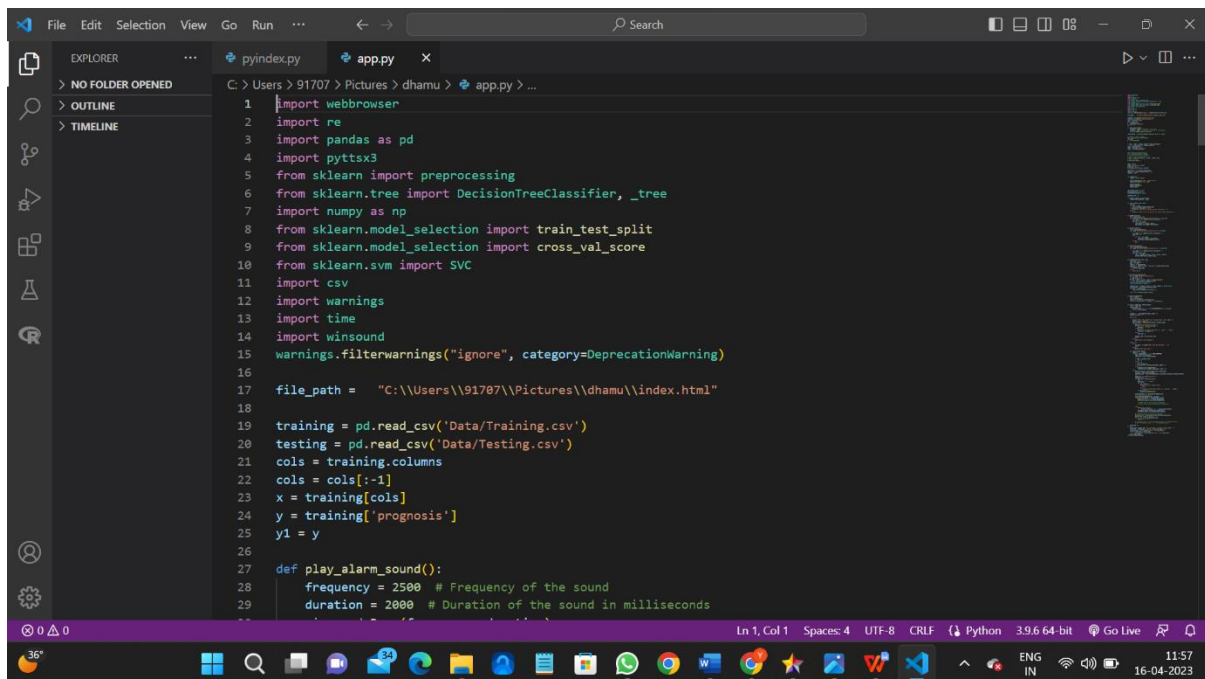
Packages used:

1. pandas
2. nlkt
3. skleran
4. numpy
5. time

IDE-used:

Visual Studio Code

*importing the packages into the library:



```
1 import webbrowser
2 import re
3 import pandas as pd
4 import pyttsx3
5 from sklearn import preprocessing
6 from sklearn.tree import DecisionTreeClassifier, _tree
7 import numpy as np
8 from sklearn.model_selection import train_test_split
9 from sklearn.model_selection import cross_val_score
10 from sklearn.svm import SVC
11 import csv
12 import warnings
13 import time
14 import winsound
15 warnings.filterwarnings("ignore", category=DeprecationWarning)
16
17 file_path = "C:\\Users\\91707\\Pictures\\dhamu\\index.html"
18
19 training = pd.read_csv('Data/Training.csv')
20 testing = pd.read_csv('Data/Testing.csv')
21 cols = training.columns
22 cols = cols[:-1]
23 x = training[cols]
24 y = training['prognosis']
25 y1 = y
26
27 def play_alarm_sound():
28     frequency = 2500 # Frequency of the sound
29     duration = 2000 # Duration of the sound in milliseconds
```

Fig 7.1: Importing packages

```
1 import webbrowser
2 import re
3 import pandas as pd
4 import pytsx3
5 from sklearn import preprocessing
6 from sklearn.tree import DecisionTreeClassifier, _tree
7 import numpy as np
8 from sklearn.model_selection import train_test_split
9 from sklearn.model_selection import cross_val_score
10 from sklearn.svm import SVC
11 import csv
12 import warnings
13 import time
14 import winsound
```

command prompt, with specified path, type “pip install package name” which you want to install (like numpy, pandas, seaborn, scikit-learn, matplotlib.pyplot)

Pre-processing the data:

Data pre-processing is a data mining technique which is used to transform the raw data in a useful and efficient format.

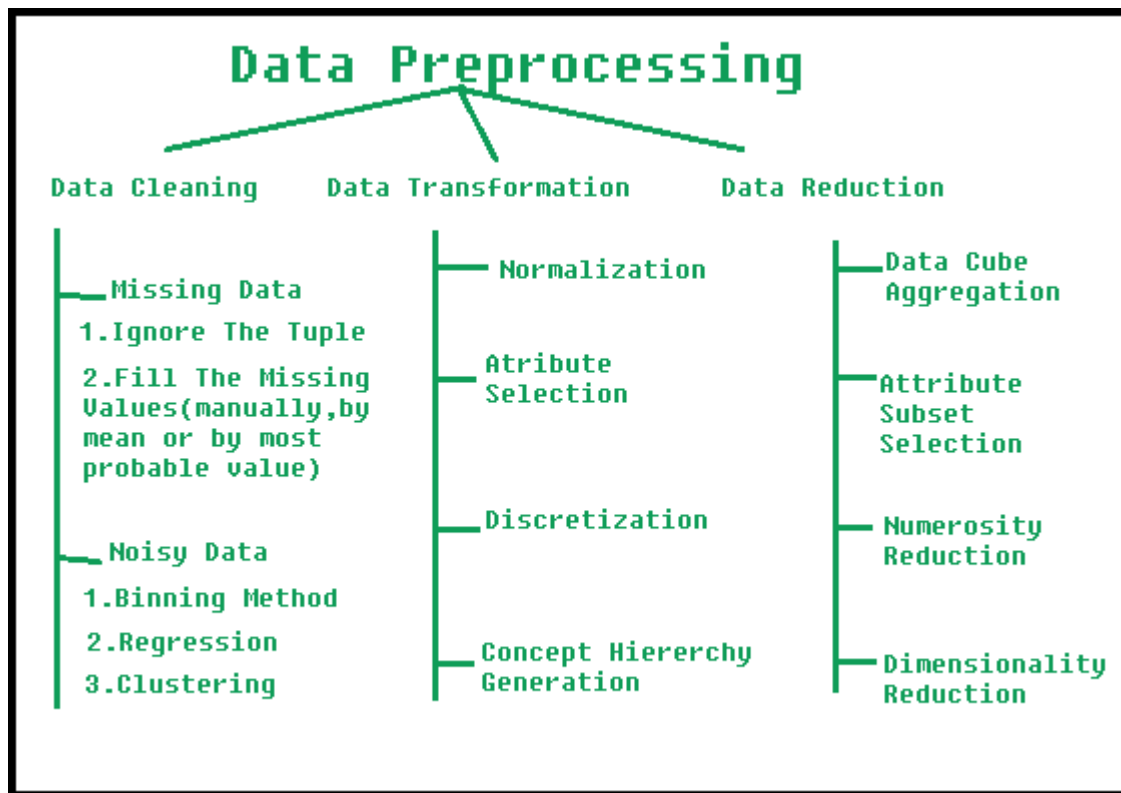


Fig 7.2: Data pre-processing diagram

The Porter stemming algorithm (or ‘Porter stemmer’) is a process for removing the commoner morphological and inflexional endings from words in English. Its main use is as part of a term normalisation process that is usually done when setting up Information Retrieval systems.

Label Encoding:

Label Encoding is a popular encoding technique for handling categorical variables. In this technique, each label is assigned a unique integer based on alphabetical ordering.

```
# mapping strings to numbers
le = preprocessing.LabelEncoder()
le.fit(y)
y = le.transform(y)
```

Fig 7.3: Label encoding

Splitting the data:

The train-test split procedure is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model.

```
x_train, x_test, y_train, y_test = train_test_split(
    x, y, test_size=0.33, random_state=42)
testx = testing[cols]
testy = testing['prognosis']
testy = le.transform(testy)
```

Fig 7.4: Splitting the data

Model Fitting:

Model fitting is a measure of how well a machine learning model generalizes to similar data to that on which it was trained. A model that is well-fitted produces more accurate outcomes. A model that is overfitted matches the data too closely. A model that is underfitted doesn't match closely enough.

```
44
45     clf1 = DecisionTreeClassifier()
46     clf = clf1.fit(x_train, y_train)
47     # print(clf.score(x_train,y_train))
48     # print ("cross result=====")
49     scores = cross_val_score(clf, x_test, y_test, cv=3)
50     # print (scores)
51     print(scores.mean())
52
53
54     model = SVC()
55     model.fit(x_train, y_train)
56     print("for svm: ")
57     print(model.score(x_test, y_test))
58
59     importances = clf.feature_importances_
60     indices = np.argsort(importances)[::-1]
61     features = cols
62
```

Fig 7.5: Model fitting

CHAPTER 8

USER INTERFACE

we have implemented all the frontend related HTML, CSS and in app.py we have used all those files and routed to the different paths provided by the user based on the selection like/getdata,/uploaddata,/viewdata,/preprocess,/trainmodel,/predict.



Fig 8.1: User interface

Working of chatbot:

The chatbot interacts with users through a chat interface, which can be implemented in various platforms such as a website, messaging app, or mobile app. Users input their medical symptoms, conditions, or concerns through text, or other input methods.



Fig 8.2: Working of chatbot

Processing the input:

The chatbot is typically integrated with a medical knowledge base, which contains a repository of medical information such as symptoms, conditions, treatments, and emergency procedures. This knowledge base is used by the chatbot to provide accurate and relevant responses to user queries.

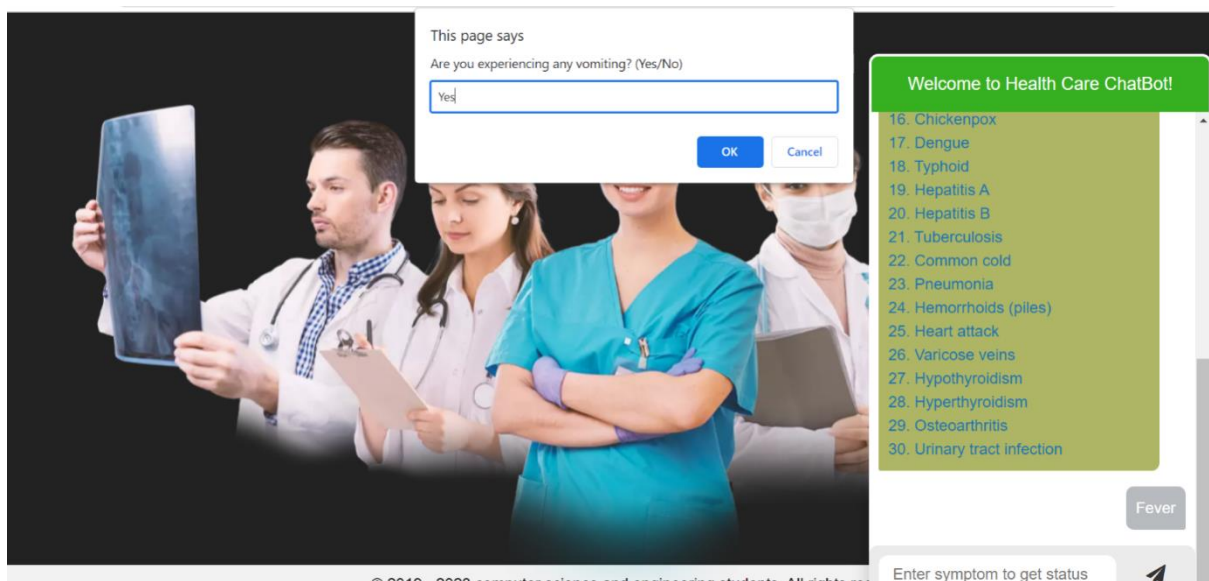


Fig 8.3: Processing the input

Decision-Making:

Based on the user input and the medical knowledge base, the chatbot employs decision-making algorithms or rules to determine the appropriate course of action. This can involve identifying potential medical emergencies, providing first aid instructions, recommending immediate medical attention, or referring the user to relevant healthcare resources.

Response Generation:

The chatbot generates responses based on the decision-making process. Responses can be in the form of text, images, links, or other formats, depending on the platform and requirements of the project. The chatbot may also ask clarifying questions to gather additional information from the user in order to provide more accurate and personalized responses.

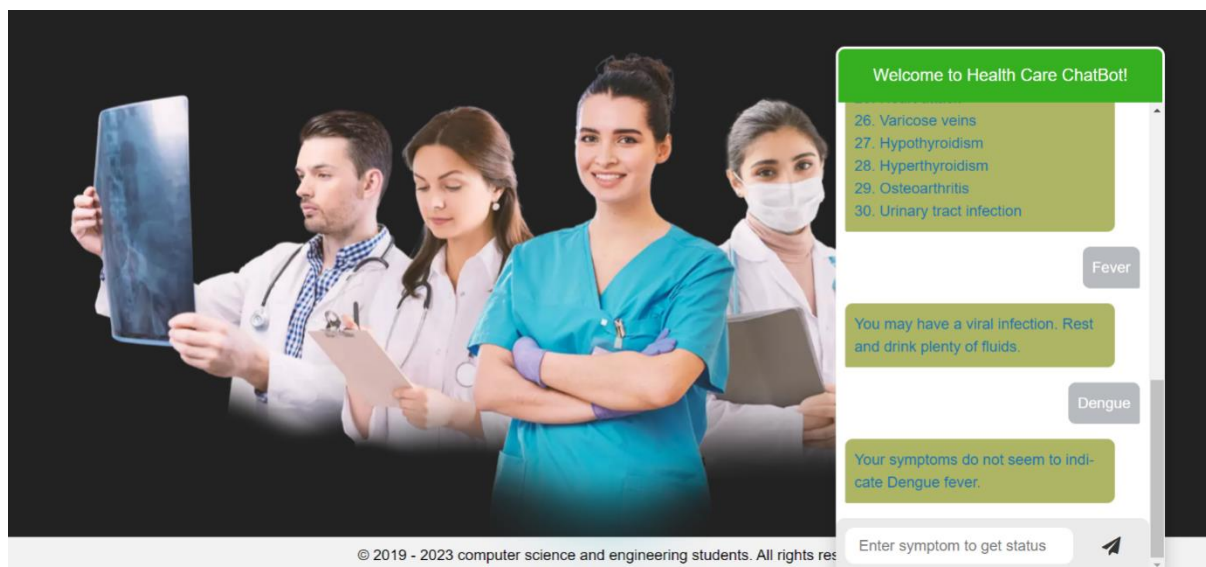


Fig 8.4: Response generation

CHAPTER 9

SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

9.1 SYSTEM TEST:

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing:

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing:

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

9.2 TYPES OF TESTS

Unit testing:

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration.

This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format.
- No duplicate entries should be allowed.
- All links should take the user to the correct page.

Integration testing:

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g., components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Functional test:

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centred on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

Acceptance Testing:

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user

Test Results:

All the test cases mentioned above passed successfully. No defects encountered.

TEST CASES:

Input	Output	Result
Input text	Tested for whether the text consists of news data or not	Success

Test cases Model building:

S.NO	Test cases	I/O	Expected O/T	Actual O/T	P/F
1	Read the dataset.	Dataset path.	Dataset need to read successfully.	Dataset fetched successfully.	P
2	Performing pre-processing on the dataset	Pre-processing part takes place	Pre-processing should be performed on dataset	Pre-processing successfully completed.	P
3	Model Building	Model Building for the clean data	Need to create model using required algorithms	Model Created Successfully.	P
4	Scores Generated and comparison	Input provided.	Output should be resulted comparison plot	Resulted successfully	P

CHAPTER 10

CONCLUSION

It is a valuable tool for providing timely medical information and assistance in case of emergencies. However, it's important to note that a medical alert chatbot is not a replacement for professional medical advice or emergency services, and it should always be used in conjunction with qualified healthcare providers for accurate diagnosis and treatment. With proper design, implementation, and ongoing updates, a medical alert chatbot project has the potential to improve healthcare outcomes and enhance emergency response capabilities.

CHAPTER 11

REFERENCES:

1. "The Use of Chatbots in Healthcare: A Systematic Review" - This article published in the Journal of Medical Internet Research (JMIR) provides an overview of the use of chatbots in healthcare, including their applications in medical alert and emergency response scenarios.

2. "Artificial Intelligence Chatbot for Medical Diagnosis: An Experimental Study" - This research paper published in the Journal of Medical Systems presents a study on an AI chatbot for medical diagnosis, including its ability to identify medical emergencies and provide appropriate responses.

3. "Building a Medical Chatbot: Lessons Learned from Developing, Deploying, and Evaluating a Chatbot for COVID-19 Health Assessment" - This paper presented at the Association for Computational Linguistics (ACL) 2020 conference discusses the development and evaluation of a chatbot for COVID-19 health assessment, including its potential use as a medical alert tool.