AI HEALTHCARE CHATBOT WEBSITE USING DJANGO

A Project Report Submitted in partial fulfillment of the requirement for the Award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

T.N.L. KRISHNA	**20T91A0595**
S. SUPRIYA RAVIKUMAR	**20T91A0577**
V. SHYAM	**20T91A0597**
K. PRADEEP	**21T95A0517**

Under the Esteemed Guidance of

Mr. P. SASI KUMAR, M.Tech., (Ph.D.),

Assistant Professor

Computer Science and Engineering

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GIET ENGINEERING COLLEGE

[Affiliated to JNTUK, Kakinada |Approved by AICTE| Accredited by NAAC A+]
NH-16, CHAITANYA KNOWLEDGE CITY, RAJAMAHENDRAVARAM – 533 296,
ANDHRA PRADESH

2020-2024

GIET ENGINEERING COLLEGE

[Affiliated to JNTUK, Kakinada |Approved by AICTE| Accredited by NAAC A+]
NH-16, CHAITANYA KNOWLEDGE CITY, RAJAMAHENDRAVARAM – 533 296,
ANDHRA PRADESH

2020-2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that **T.N.L. KRISHNA (20T91A0595), S. SUPRIYA RAVIKUMAR (20T91A0577), V. SHYAM (20T91A0597), K. PRADEEP (21T95A0517)** studying in IV B. Tech II Semester of Computer Science and Engineering have submitted their project “AI HEALTHCARE CHATBOT WEBSITE USING DJANGO”” during the academic year 2023-2024 in partial fulfillment of the requirements for the award of degree in Bachelor of Technology, JNTUK, Kakinada.

The result embodied in this project has not been submitted to any other University or Institute for the award of a degree.

Project Guide
Mr. P. SASI KUMAR, M.Tech., (Ph.D.),
Assistant Professor
CSE

Head of the Department
Dr. Sk. Meera Sharif
Professor and HoD
CSE

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

It is a privilege for us to have undertaken the project **"AI HEALTHCARE CHATBOT WEBSITE USING DJANGO"** in **GIET ENGINEERING COLLEGE, RAJAMAHENDRAVARAM**.

We avail this opportunity to express our deep sense of gratitude and heart full thanks to **Sri K. SASI KIRAN VARMA**, Vice Chairman of **GIET ENGINEERING COLLEGE, RAJAMAHENDRAVARAM.**

We are thankful to our Principal **Dr. M. VIJAY SEKHAR BABU** for encouraging us to do this project.

We are deeply indebted to our HoD **Dr. Sk. MEERA SHARIF**, Professor in Computer Science and Engineering, **GIET ENGINEERING COLLEGE**, whose motivation in the field of software development made us overcome all the hardships during the course of study.

We are heartily thankful to our project coordinator **Mr. P. SASI KUMAR,** Assistant Professor, **GIET ENGINEERING COLLEGE**, for his moral support which was always there to comfort and solace during tough times.

Finally, we would like to thank our **TEACHING AND NON-TEACHING STAFF** whose blessings and encouragement were always there as a source of strength and inspiration. Although the title "Acknowledgement" cannot represent our true feelings for all these persons,we feel very much thankful to all of them and also to our **PARENTS** and **FRIENDS** for encouraging and giving us all the moral support required for making this endeavor a reality.

T.N.L. KRISHNA	**20T91A0595**
S. SUPRIYA RAVIKUMAR	**20T91A0577**
V. SHYAM	**20T91A0597**
K. PRADEEP	**21T95A0517**

DECLARATION

We hereby declare that this project entitled "**AI HEALTHCARE CHATBOT WEBSITE USING DJANGO**" submitted to the Department of COMPUTER SCIENCE AND ENGINEERING, GIET ENGINEERING COLLEGE, affiliated to JNTUK, Kakinada, as partial fulfillment for the award of Bachelor of Technology degree is entirely the original work done by us and has not been submitted to any other organization.

Project Members	Pin Numbers	Signature
T.N.L. KRISHNA	**20T91A0595**	
S. SUPRIYA RAVIKUMAR	**20T91A0577**	
V. SHYAM	**20T91A0597**	
K. PRADEEP	**21T95A0517**	

ABSTRACT

With the increasing population of India and a rise in birth rates coupled with advancements in the medical field leading to a decrease in death rates, there is a concerning shortage of doctors to adequately serve the growing population. This issue becomes apparent when visiting government hospitals in cities, where the limited availability of doctors is a major cause of inadequate treatment and, in some cases, even resulting in patient deaths. Furthermore, doctors, being human, are prone to making mistakes in providing accurate treatments, which can also lead to patient fatalities. To address such situations, the development of an intelligent and smart chat bot that can offer advice to both doctors and patients becomes crucial, potentially saving the lives of hundreds of people. Virtual assistants, including chat bots, have the potential to assist patients and healthcare providers with various medical-related tasks. Chat bots are computer programs designed to engage in conversations with individuals, offering assistance through text messages, applications, or instant messaging. These bots can identify symptoms and provide diagnoses based on specific symptoms, as well as recommend appropriate doctors for prompt responses. While chat bots are already extensively employed in other industries such as retail to enhance processes, their integration into healthcare services can prove invaluable.

TABLE OF CONTENTS

CHAPTER – 1
INTRODUCTION

1.1. Project introduction

Welcome to our AI Healthcare Chatbot website built using Django! In today's fast-paced world, accessing healthcare services efficiently and conveniently is essential. Our platform aims to revolutionize the way individuals interact with healthcare information and services by incorporating artificial intelligence (AI) into a user-friendly chatbot interface. Our chatbot serves as a virtual assistant, capable of understanding and responding to user inquiries regarding various healthcare topics such as symptoms, conditions, treatments, and more. our chatbot provides accurate and personalized responses to address users' concerns effectively. Our platform offers a comprehensive repository of reliable medical information curated by healthcare professionals. Users can access articles, guides, and resources covering a wide range of health-related topics to educate themselves and make informed decisions about their well-being. By leveraging AI algorithms and medical databases, our platform assists users in assessing their health status and determining the appropriate course of action. Our website is designed to be mobile-friendly, allowing users to access healthcare information and services on-the-go using their smartphones or tablets. Whether at home, work, or on vacation, users can rely on our platform to meet their healthcare needs conveniently and efficiently. The objective of developing a Health-Care Chat Bot using Support Vector Machines (SVM) and Decision Tree algorithms is to provide an intelligent and interactive conversational system that can assist users in addressing their health-related queries and concerns. The chat bot will leverage the capabilities of both SVM and Decision Tree algorithms to enhance its performance and accuracy in understanding user inputs and providing relevant and reliable responses.

1.2. LITERATURE SURVEY

[1]E-Health is defined in as the practice of health-care supported by electronic processes and communications.

Everybody talks about e-health these days, but few people have come up with a clear definition of this comparatively new term. Barely in use before 1999, has this term now seemed to serve as a general "buzzword," used to characterize not only "Internet medicine", but also virtually everything related to computers and medicine? The term was apparently first used by industry leaders and marketing people rather than academics. They created and used this term in line with other "e-words" such as e-commerce, e-business, e-solutions, and so on, in an attempt to convey the promises, principles, excitement (and hype) around e-commerce (electronic commerce) to the health arena, and to give an account of the new possibilities the Internet is opening up to the area of health care. Intel, for example, referred to e-health as "a concerted effort undertaken by leaders in health care and hi-tech industries to fully harness the benefits available through convergence of the Internet and health care." Because the Internet created new opportunities and challenges to the traditional health care information technology industry, the use of a new term to address these issues seemed appropriate. These "new" challenges for the health care information technology industry were mainly (1) the capability of consumers to interact with their systems online (B2C = "business to consumer"); (2) improved possibilities for institution-to-institution transmissions of data (B2B = "business to business"); (3) new possibilities for peer-to-peer communication of consumers (C2C = "consumer to consumer").

[2] It shows that emerging technologies are revolutionizing the way of thinking about healthcare.

Existing technologies in healthcare, now commonplace, were once novel ideas, care models and devices, and new treatments. Today emerging technologies (ETs) are developed and implemented in healthcare organizations at a rapid rate. Nurses, nurse informaticists, and nurse educators should have a clear comprehension of the role of emerging technology in healthcare to optimize clinical practice. Innovation and innovators are essential to revolutionizing antiquated healthcare business models to offer new products, services, and models to modernize practice and serve the Quadruple Aim better. **Divergent collaborations** and **innovation centres** in healthcare organizations provide nurses the opportunity to be champions and early adopters and enforcers of ETs and responsible innovation, thereby improving safety and quality outcomes and promoting **health equity**. The knowledge of Nurses as innovators will further the impending need of ETs to serve the Quadruple Aim.

[3] People rely more and more frequently on health tracking devices, connected health devices, and personalized and proximity medicine

IoT technology keeps patients better connected to doctors via remote monitoring and virtual visits; it helps hospitals track staff and patients; IoT healthcare devices facilitate the care of chronic disease; IoT automates patient care workflow; it quickly culls, analyses data and disseminates data to keep everyone on the same page; it reduces inefficiency and errors; it optimizes the pharmaceutical manufacturing process, which can result in lower drug prices; it maintains quality control and manages sensitive items while they're in transit; it can even lower healthcare costs by streamlining the overall process.

[4] Chat bots can be programmed to respond the same way each time, to respond differently to messages containing certain keywords and even to use machine learning to adapt their responses to fit the situation

Through chat bots, one can communicate with text or voice interface and get reply through artificial intelligence. Typically, a chat bot will communicate with a real person. Chat bots are used in applications such as ecommerce customer service, call centres and Internet gaming. Chat bots are programs built to automatically engage with received messages. Chat bots can be programmed to respond the same way each time, to respond differently to messages containing certain keywords and even to use machine learning to adapt their responses to fit the situation. A developing number of hospitals, nursing homes, and even private centres, presently utilize online Chat bots for human services on their sites. These bots connect with potential patients visiting the site, helping them discover specialists, booking their appointments, and getting them access to the correct treatment. In any case, the utilization of artificial intelligence in an industry where individuals' lives could be in question, still starts misgivings in individuals. It brings up issues about whether the task mentioned above ought to be assigned to human staff.

[5]A developing number of hospitals, nursing homes, and even private centers, presently

utilize online Chat bots for human services on their sites.

Chatbots are programs built to automatically engage with received messages. Chatbots can be programmed to respond the same way each time, to respond differently to messages containing certain keywords and even to use machine learning to adapt their responses to fit the situation. A developing number of hospitals, nursing homes, and even private centres, presently utilize online Chatbots for human services on their sites. These bots connect with potential patients visiting the site, helping them discover specialists, booking their appointments, and getting them access to the correct treatment. In any case, the utilization of artificial intelligence in an industry where individuals' lives could be in question, still starts misgivings inindividuals. It brings up issues about whether the task mentioned above ought to be assigned to human staff. This healthcare chatbot system will help hospitals to provide healthcare support online 24 x 7, it answers deep as well as general questions. It also helps to generate leads and automatically delivers the information of leads to sales. By asking the questions in series it helps patients by guiding what exactly he/she is looking for.

CHAPTER – 2
SYSTEM ANALYSIS

2.1 EXISTING METHOD

The existing system for a healthcare chatbot incorporates blockchain technology and data analysis. This system aims to enhance the security, privacy, and integrity of healthcare data, while also leveraging data analysis techniques to provide valuable insights and improve the overall patient experience. This is more costly to use and time taking process. It can use by multiple authorizations.

2.2 DISADVANTAGES

- Complex user interface. Making user difficult to recognize the operations
- Late response to user due to highly dumped data set
- More processing time to process huge data

2.3 PROPOSED SYSTEM

The proposed system for a healthcare chatbot integrates Natural Language Processing (NLP) techniques, Decision Trees, and Support Vector Machines (SVM) to create an advanced and intelligent healthcare chatbot solution.

The chatbot utilizes NLP algorithms to understand and interpret user queries or messages in a natural language format. NLP allows the chat bot to extract relevant information, such as symptoms, medical history, or specific concerns from the user's input. By applying NLP techniques, the chatbot can comprehend and respond effectively to a wide range of user queries, facilitating a more interactive and personalized conversation.

2.4 ADVANTAGES

- More visually pleasing
- Less response time to user
- Simple data set
- Lessprocessingtimes

CHAPTER-3
REQUIREMENT SPECIFICATION

1.3. SOFTWARE REQUIREMENTS:

Operating System	Windows 7/8/10
Server-side Script	HTML, CSS, Bootstrap & JS
Programming Language	Python
Libraries	Flask, Pandas, MySQL. Connector, Os, Smtplib, NumPy
IDE/Workbench	PyCharm
Technology	Python 3.6+
Server Deployment	Xampp Server
Database	MySQL

3.2. HARDWARE REQUIREMENTS:

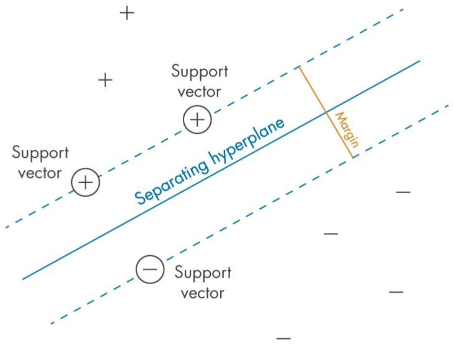
Processor	I3/Intel Processor
RAM	8GB (min)
Hard Disk	128 GB
Key Board	Standard Windows Keyboard
Mouse	Two or Three Button Mouse
Monitor	Any

CHAPTER-4

TECHNOLOGIES USED

4.1. SUPPORT VECTOR MACHINE (SVM):

Support Vector Machines (SVM) is a powerful machine-learning algorithm used for both classification and regression tasks. SVM works by finding the optimal hyperplane that separates different classes in a high-dimensional space. The algorithm aims to maximize the margin between classes, representing the distance between the hyperplane and the nearest data points of each class. SVM is particularly effective in handling complex decision boundaries and is robust against overfitting, making it suitable for various applications, such as image classification, text categorization, and bioinformatics. It utilizes a kernel trick, allowing it to implicitly map data into higher-dimensional spaces, enhancing its ability to classify nonlinear patterns.

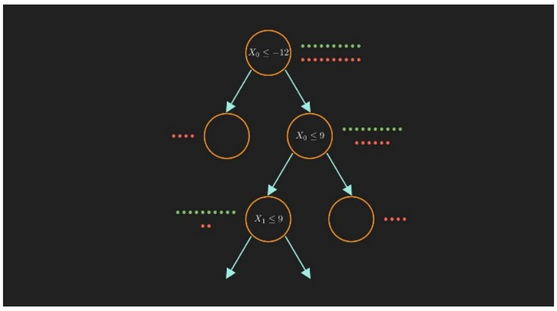


One key strength of SVM is its versatility, as it can handle both linear and nonlinear relationships in data. However, SVM's performance heavily depends on selecting appropriate hyperactive parameters and kernel functions. Additionally, SVM may face challenges with large datasets, as training time can increase significantly. In summary, SVM is a versatile algorithm known for its ability to handle complex decision boundaries and classify both linear and nonlinear patterns. Its effectiveness, coupled with the kernel trick, makes it a valuable tool in various domains where robust and accurate classification is essential.

4.2. DECISION TREE:

Decision Tree is a versatile and widely used algorithm in machine learning and data analysis. It operates by recursively partitioning the dataset based on feature values, creating a tree-like structure. At each node, the algorithm selects the feature that best splits the data, optimizing criteria such as Gini impurity or information gain. Decision Trees are intuitive and easy to interpret, making them valuable for both classification and regression tasks. They excel in handling non-linear relationships, capturing complex decision boundaries in the data.

Decision Trees offer transparency, enabling users to understand how predictions are made, making them valuable in fields such as finance, medicine, and industry. However, they are prone to overfitting, capturing noise in the data. Techniques like pruning and specifying minimum sample sizes help mitigate this. Ensemble methods like Random Forests further enhance performance by aggregating multiple decision trees.



Overall, Decision Trees are powerful tools for predictive modeling, providing a balance between interpretability and accuracy. Their adaptability to various data types and straightforward representation makes them a staple in the toolkit of machine learning practitioners for tasks ranging from credit scoring to medical diagnosis.

4.3. NATURAL LANGUAGE PROCESSING (NLP) :

Natural language processing (NLP) is a subfield of linguistics, computer science, and artificial intelligence concerned with the interactions between computers and human language, in particular how to program computers to process and analyse large amounts of natural language data. The result is a computer capable of "understanding" the contents of documents, including the contextual nuances of the language within them. The technology can then accurately extract information and insights contained in the documents as well as categorize and organize the documents themselves.

A chatbot is an NLP 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.

Why are chatbots important? A chatbot 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). Formulating responses to questions in natural language is one of the most typical Examples of Natural Language Processing applied in various enterprises' end-use applications.

CHAPTER – 5
SYSTEM DESIGN

5.1. INPUT DESIGN

Introduction of 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.

5.2. 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:

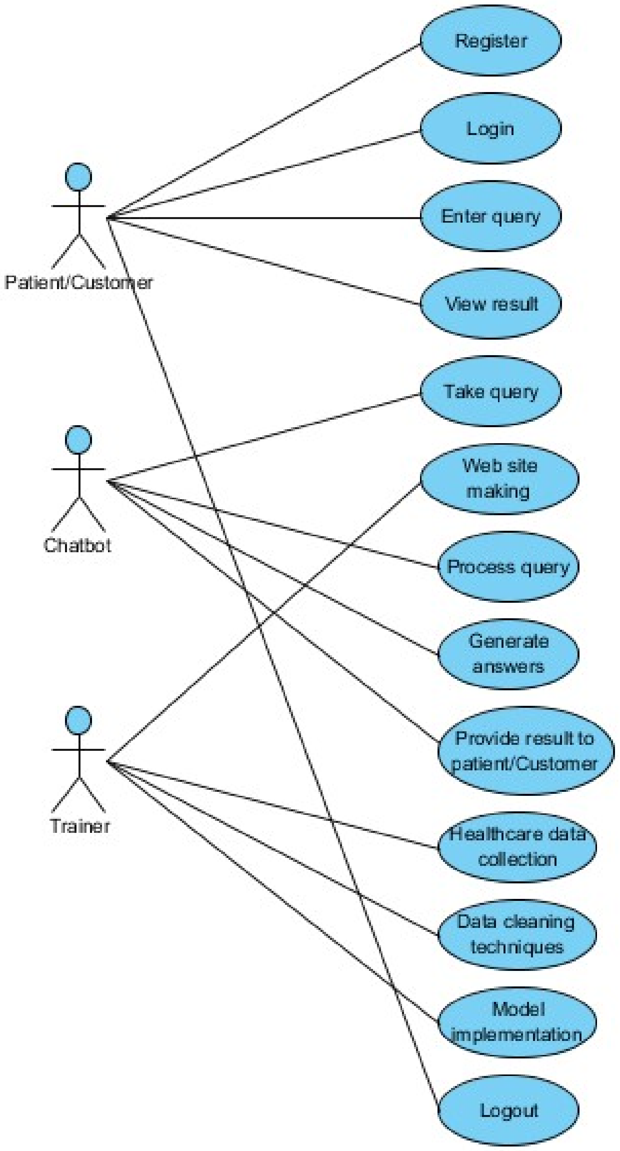
The objectives of input design are:

- 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.

5.3. UML Diagrams:

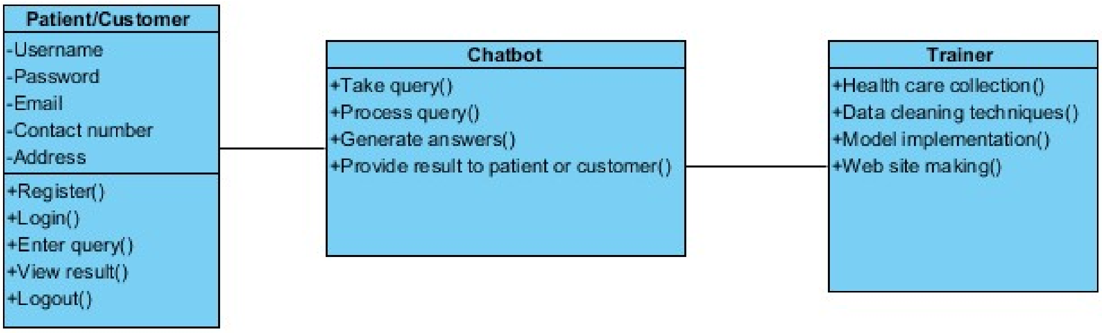
USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral 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.


Patient/Customer
Chatbot
Trainer
Register
Login
Enter query
View result
Take query
Web site making
Process query
Generate answers
Provide result to patient/Customer
Healthcare data collection
Data cleaning techniques
Model implementation
Logout

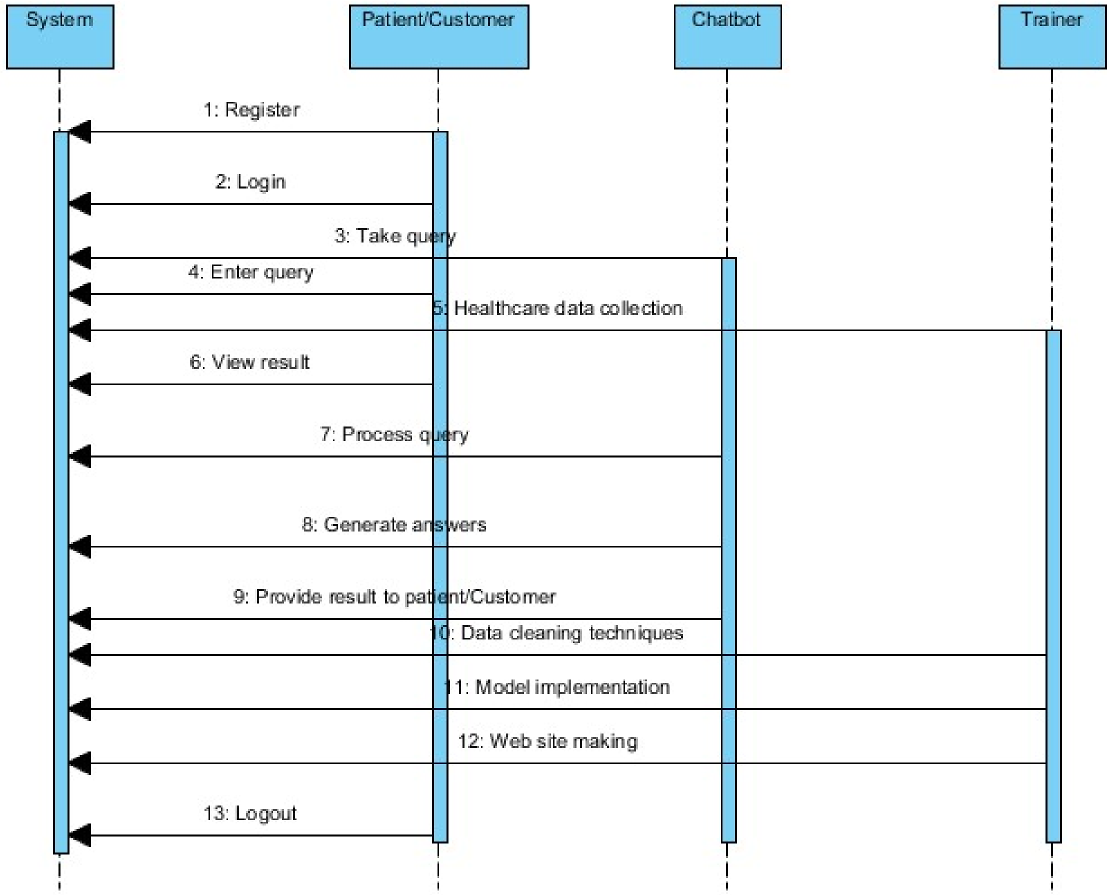
CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling 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.



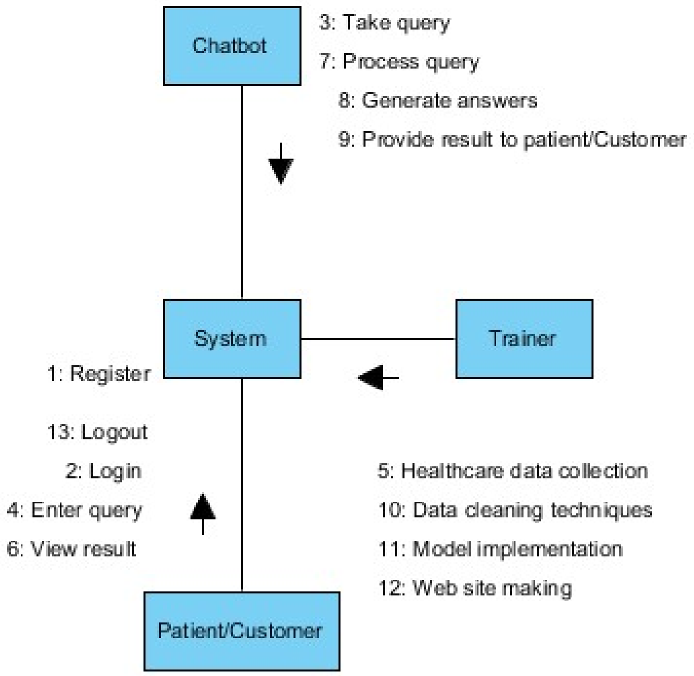
SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling 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.



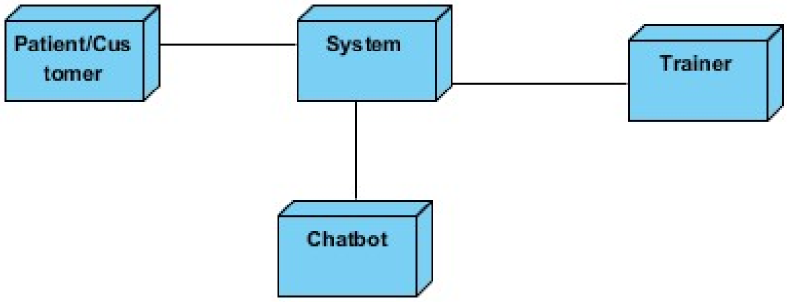
Collaboration Diagram:

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.



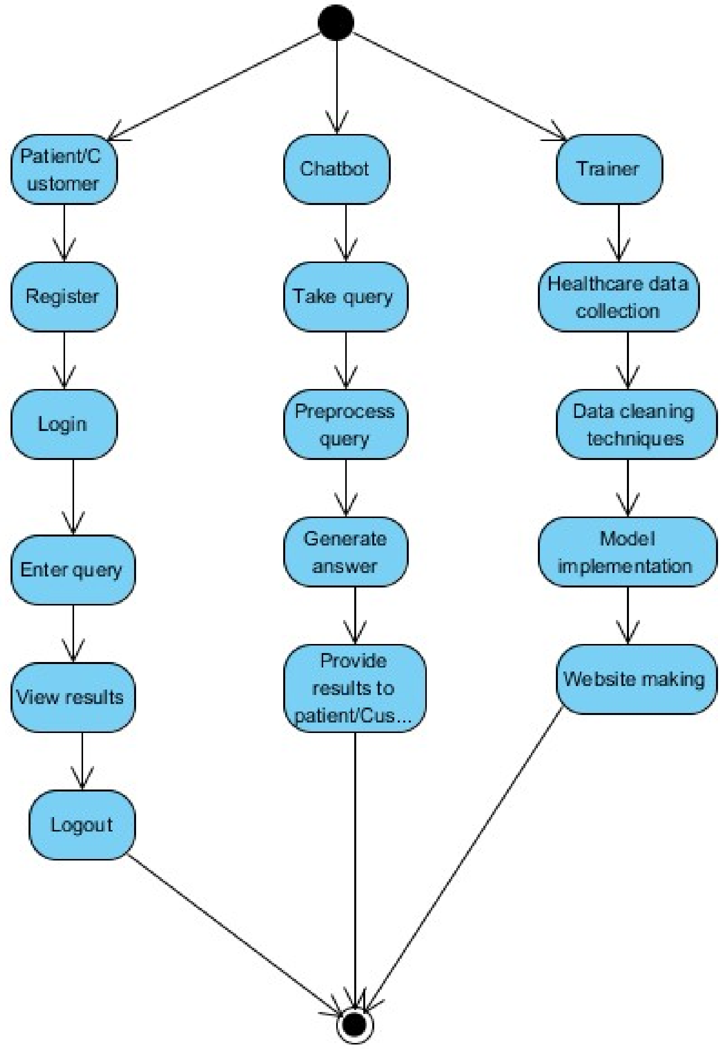
DEPLOYMENT DIAGRAM

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware's used to deploy the application.



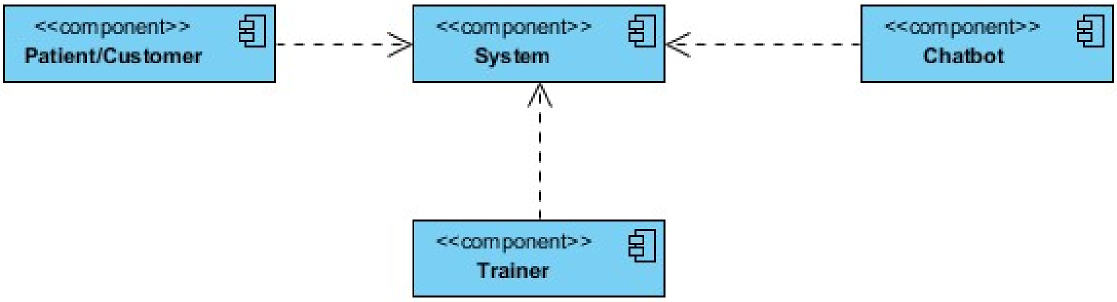
ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



Component diagram:

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical **c**omponents in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required functions is covered by

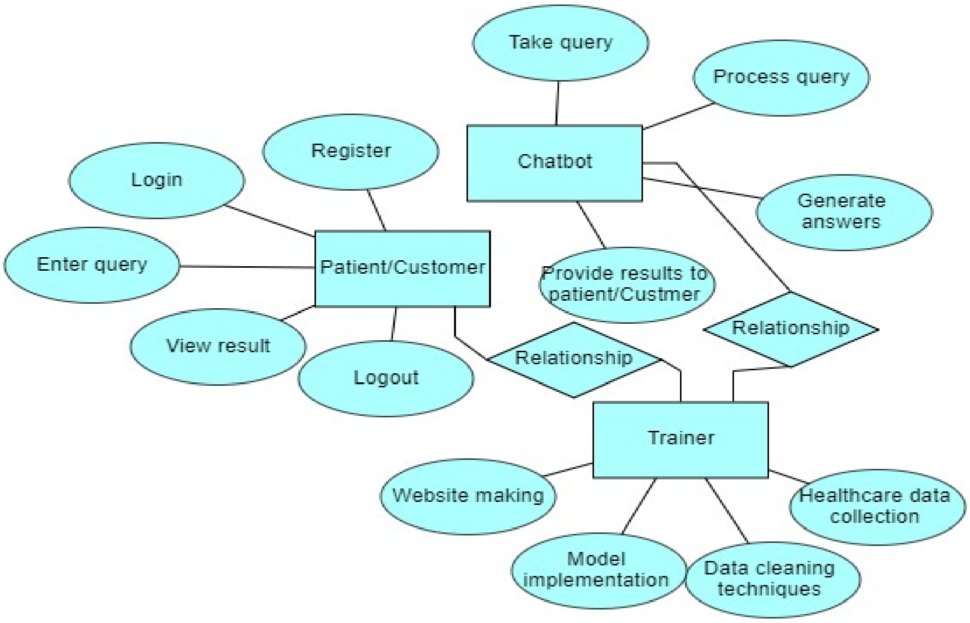


ER DIAGRAM

An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram).

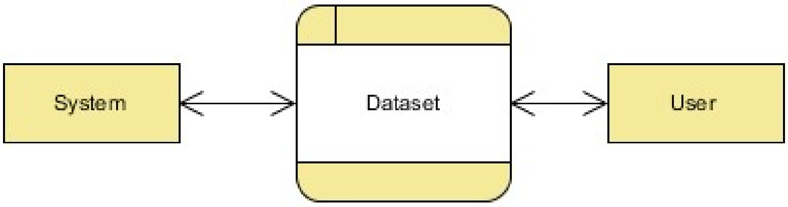
An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes.

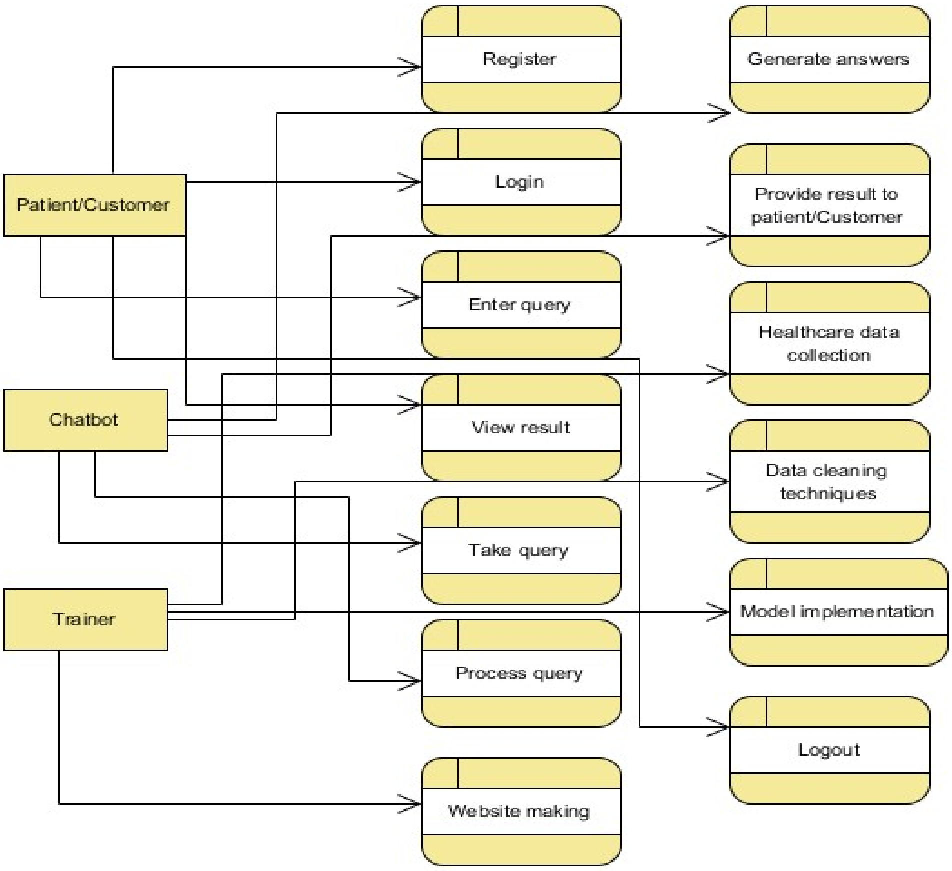
In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure .

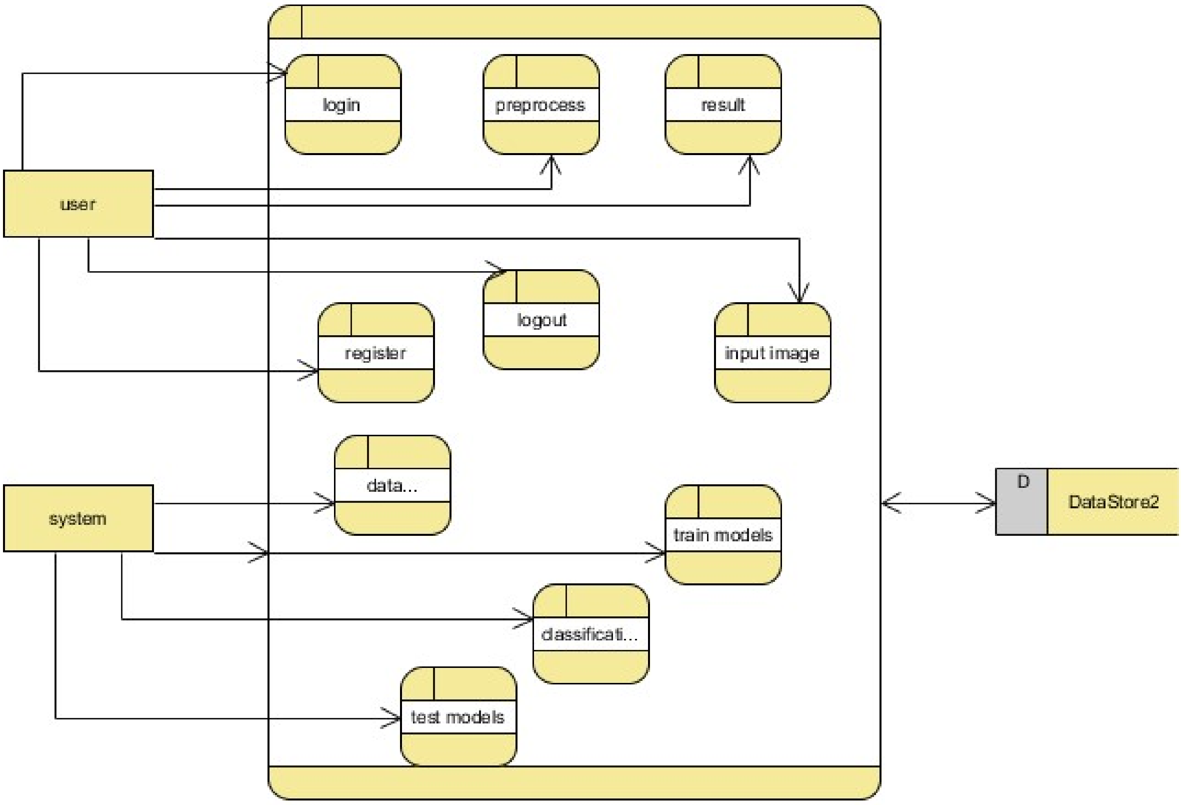


DFD DIAGRAM

A Data Flow Diagram (DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically. It can be manual, automated, or a combination of both. It shows how information enters and leaves the system, what changes the information and where information is stored. The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system.






login
preprocess
result
user
logout
register
input image
data...
system
train models
D
DataStore2
classificati...
test models

CHAPTER-6

CODING AND IMPLEMENTATION

Create chatbot.py:

```
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
from tkinter import *
warnings.filterwarnings("ignore", category=DeprecationWarning)


training = pd.read_csv('models/Training.csv')
testing= pd.read_csv('models/Testing.csv')
cols= training.columns
cols= cols[:-1]
x = training[cols]
y = training['prognosis']
y1= y


reduced_data = training.groupby(training['prognosis']).max()

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


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)


clf1  = DecisionTreeClassifier()
clf = clf1.fit(x_train,y_train)
# print(clf.score(x_train,y_train))
# print ("cross result========")


scores = cross_val_score(clf, x_test, y_test, cv=3)
# print (scores)
print (scores.mean())


model=SVC()
model.fit(x_train,y_train)
print("for svm: ")
print(model.score(x_test,y_test))

importances = clf.feature_importances_
indices = np.argsort(importances)[::-1]
features = cols

def readn(nstr):
    engine = pyttsx3.init()

    engine.setProperty('voice', "english+f5")
    engine.setProperty('rate', 130)

    engine.say(nstr)
    engine.runAndWait()
    engine.stop()


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

symptoms_dict = {}

for index, symptom in enumerate(x):
       symptoms_dict[symptom] = index
def calc_condition(exp,days):
    sum=0
    for item in exp:
         sum=sum+severityDictionary[item]
    if((sum*days)/(len(exp)+1)>13):
        res = "You should take the consultation from doctor. "
        ChatLog.insert(END, res + '\n\n', 'Bot')

    else:
        res = "It might not be that bad but you should take precautions."
        ChatLog.insert(END, res + '\n\n', 'Bot')


def getDescription():
    global description_list
    with open('models/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('models/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('models/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)


def getInfo():
    # name=input("Name:")
    stR = "Please Enter your Name"

    return stR

def check_pattern(dis_list,inp):
    import re
    pred_list=[]
    ptr=0
    patt = "^" + inp + "$"
    regexp = re.compile(inp)
    for item in dis_list:
        # print(f"comparing {inp} to {item}")
        if regexp.search(item):
            pred_list.append(item)



            # return 1,item
    if(len(pred_list)>0):
        return 1,pred_list
    else:
        return ptr,item

def sec_predict(symptoms_exp):
    df = pd.read_csv('models/Training.csv')
    X = df.iloc[:, :-1]
    y = df['prognosis']
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=20)
    rf_clf = DecisionTreeClassifier()
    rf_clf.fit(X_train, y_train)

    symptoms_dict = {}

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

    input_vector = np.zeros(len(symptoms_dict))
    for item in symptoms_exp:
      input_vector[[symptoms_dict[item]]] = 1


    return rf_clf.predict([input_vector])


def print_disease(node):
    #print(node)
    node = node[0]
    #print(len(node))
    val  = node.nonzero()
    # print(val)
    disease = le.inverse_transform(val[0])
    return disease


def get_node():
    global node, depth

    if tree_.feature[node] != _tree.TREE_UNDEFINED:
        name = feature_name[node]
        threshold = tree_.threshold[node]
        print(name, disease_input)
        if name == disease_input:
            val = 1
        else:
            val = 0
        if val <= threshold:
            print(node)



            node = tree_.children_left[node]
            depth = depth + 1
            get_node()
        else:
            symptoms_present.append(name)
            node = tree_.children_right[node]
            depth =depth + 1
            get_node()


def recurse():
    global flag_endloop, node, depth
    while True:
        num_days = int(ans)
        get_node()
        print("final_node", node)
        present_disease = print_disease(tree_.value[node])
        # print( "You may have " +  present_disease )
        red_cols = reduced_data.columns
        symptoms_given = red_cols[reduced_data.loc[present_disease].values[0].nonzero()]
        # dis_list=list(symptoms_present)
        # if len(dis_list)!=0:
        #     print("symptoms present  " + str(list(symptoms_present)))
        # print("symptoms given "  +  str(list(symptoms_given)) )

        res = "Are you experiencing any \n"
        ChatLog.insert(END, "Bot: " + res + '\n\n', 'Bot')
        symptoms_exp=[]
        for syms in list(symptoms_given):
            yield syms + "? \n"
            while True:
                inp=ans
                if(inp=="yes" or inp=="no"):
                    break
                else:
                    yield "provide proper answers i.e. (yes/no) : "
            if(inp=="yes"):
                symptoms_exp.append(syms)

        second_prediction=sec_predict(symptoms_exp)
        # print(second_prediction)
        calc_condition(symptoms_exp,num_days)
        if(present_disease[0]==second_prediction[0]):
            res = "You may have " + present_disease[0]
            ChatLog.insert(END, "Bot: " + res + '\n\n', 'Bot')

            res = description_list[present_disease[0]]
            ChatLog.insert(END, "Bot: " + res + '\n\n', 'Bot')

            # readn(f"You may have {present_disease[0]}")



            # readn(f"{description_list[present_disease[0]]}")

        else:
            res = "You may have " + present_disease[0] + " or " + second_prediction[0]
            ChatLog.insert(END, "Bot: " + str(res) + '\n\n', 'Bot')
            res = description_list[present_disease[0]]
            ChatLog.insert(END, "Bot: " + str(res) + '\n\n', 'Bot')
            res = description_list[second_prediction[0]]
            ChatLog.insert(END, "Bot: " + str(res) + '\n\n', 'Bot')

        # print(description_list[present_disease[0]])
        precution_list=precautionDictionary[present_disease[0]]
        res = "Take following measures : "
        ChatLog.insert(END, "Bot: " + res + '\n\n', 'Bot')
        for  i,j in enumerate(precution_list):
            res = str(i+1) + ")" + j
            ChatLog.insert(END, res + '\n\n', 'Bot')
        yield "Do you want to continue?"
        if ans == "yes":
            flag_endloop = False
            print("inside", flag_endloop)
            node, depth = 0, 1
            yield
            # res = tree_init_obj.__next__()
            # ChatLog.insert(END, res + '\n\n')
        else:
            quit()



def tree_to_code():
    global ans
    global feature_name, disease_input, num_days, flag_endloop, tree_, user_name

    yield "Hi, Please tell me your name."
    user_name = ans
    ChatLog.insert(END, "Bot: Hi," + user_name + '\n\n', 'Bot')

    while True:
        tree = clf
```

```
feature_names = cols
tree_ = tree.tree_

# print(tree_)
feature_name = [
    feature_names[i] if i != _tree.TREE_UNDEFINED else "undefined!"
    for i in tree_.feature
]

 chk_dis=",".join(feature_names).split(",")



# conf_inp=int()
while True:

    yield "Enter the symptom you are experiencing"
    disease_input = ans
    conf,cnf_dis=check_pattern(chk_dis,disease_input)
    if conf==1:
        res = "searches related to input: "
        ChatLog.insert(END, "Bot: " + res + '\n\n', 'Bot')
        for num, it in enumerate(cnf_dis):
            res =  str(num) + ")" + str(it)
            ChatLog.insert(END, res + '\n\n')
        if num!=0:
            yield "Select the one you meant (0 - {})".format(num)
            conf_inp = ans
        else:
            conf_inp=0
        disease_input=cnf_dis[int(conf_inp)]
        break
        # print("Did you mean: ",cnf_dis,"?(yes/no) :",end="")
        # conf_inp = input("")
        # if(conf_inp=="yes"):
        #     break
    else:
        ChatLog.insert(END, "Enter valid symptom." + "\n", 'Bot')


while True:
    try:
        flag_endloop = True
        yield "Okay. From how many days ? : "

        break
    except:
        yield "Enter number of days."
```

```
def send(event=None):
    msg = EntryBox.get("1.0",'end-1c').strip()
    EntryBox.delete("0.0",END)
    global ans

    if msg != '':
        ChatLog.config(state=NORMAL)
        ChatLog.insert(END, "\t\tYou: " + msg + '\n\n', 'You')
        ChatLog.config(foreground="#442265", font=("Verdana", 12 ))



        ans = msg
        print("main", flag_endloop)
        if not flag_endloop:
            res = tree_init_obj.__next__()
            ChatLog.insert(END, "Bot: " + res + '\n\n', 'Bot')
        else:
            res = tree_obj.__next__()
            if flag_endloop:
                ChatLog.insert(END, "Bot: " + res + '\n\n', 'Bot')
            else:
                res = tree_init_obj.__next__()
                ChatLog.insert(END, "Bot: " + res + '\n\n', 'Bot')



ans = None
flag_endloop = None
tree_init_obj = tree_to_code()
node, depth = 0, 1
tree_obj = recurse()
symptoms_present = []
getSeverityDict()
getDescription()
getprecautionDict()
base = Tk()
base.title("Chat Bot")
base.geometry("400x500")
base.resizable(width=FALSE, height=FALSE)
#Create Chat window
ChatLog = Text(base, bd=0, bg="white", height="8", width="50", font="Arial",)
ChatLog.config(state=DISABLED)
ChatLog.tag_config('You', background="grey", foreground="black")
ChatLog.tag_config('Bot', background="black", foreground="white")
#Bind scrollbar to Chat window
scrollbar = Scrollbar(base, command=ChatLog.yview, cursor="heart")
ChatLog['yscrollcommand'] = scrollbar.set
```

```
#Create Button to send message
SendButton = Button(base, font=("Verdana",12,'bold'), text="Send", width="12", height=5,
                    bd=0, bg="#32de97", activebackground="#3c9d9b",fg='#ffffff',
                    command= send )
base.bind('<Return>', send)
#Create the box to enter message
EntryBox = Text(base, bd=0, bg="white",width="29", height="5", font="Arial")
#EntryBox.bind("<Return>", send)
#Place all components on the screen
scrollbar.place(x=376,y=6, height=386)
ChatLog.place(x=6,y=6, height=386, width=370)
EntryBox.place(x=128, y=401, height=90, width=265)
SendButton.place(x=6, y=401, height=90
base.mainloop()
```

CHAPTER-7
TESTING AND VALIDATION

7.1. Feasibility Study

The feasibility of the project is analysed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- Economical feasibility
- Technical feasibility
- Social feasibility

Economical Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources.

Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

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.

7.2 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.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

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.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

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

Functional testing

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 centered 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.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is 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.

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

7.3. TEST CASES:

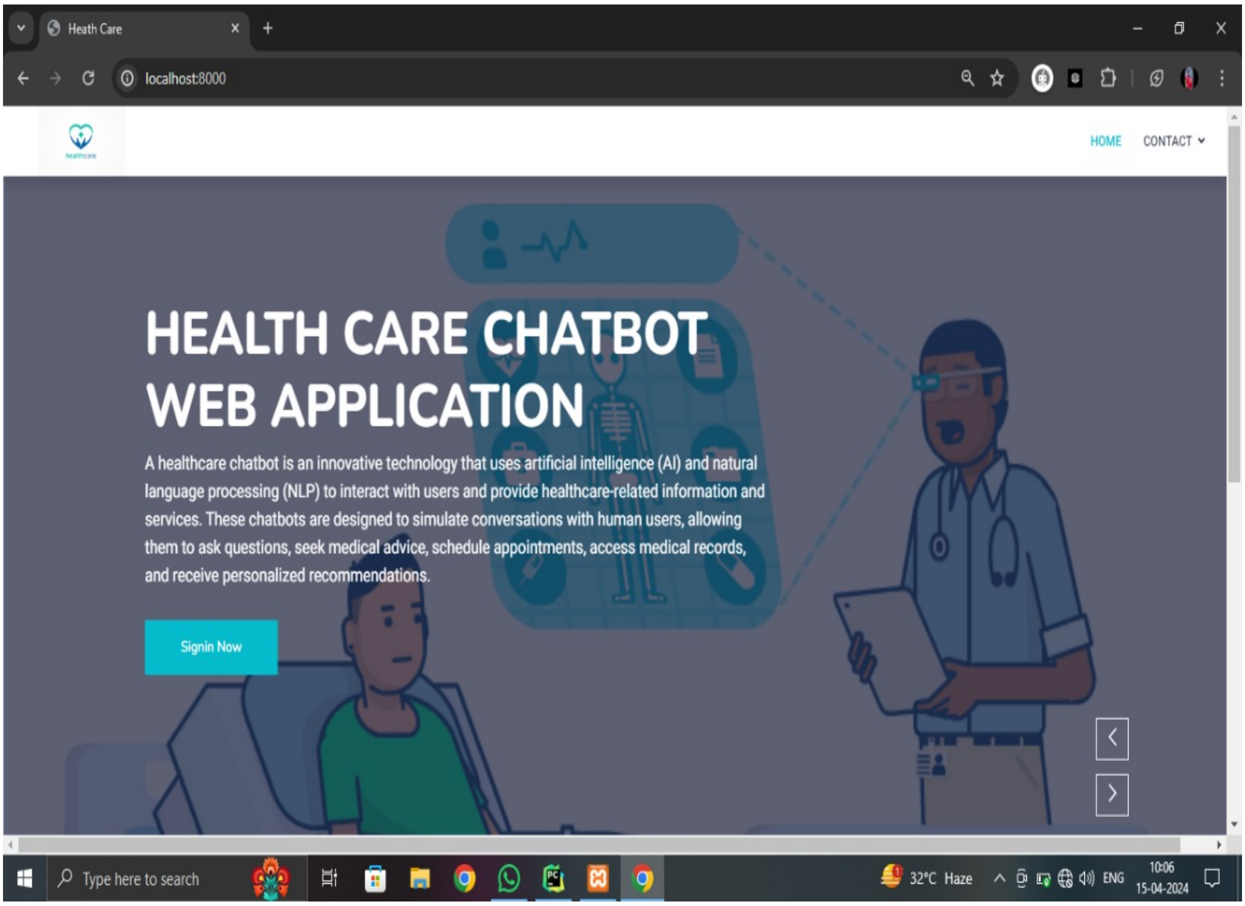
Input	Output	Result
Input image	Output be the disease prediction	Success

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	Classification	Input image provided.	Output should be there after getting the request, the disease will be predicted.	Model classified successfully	P

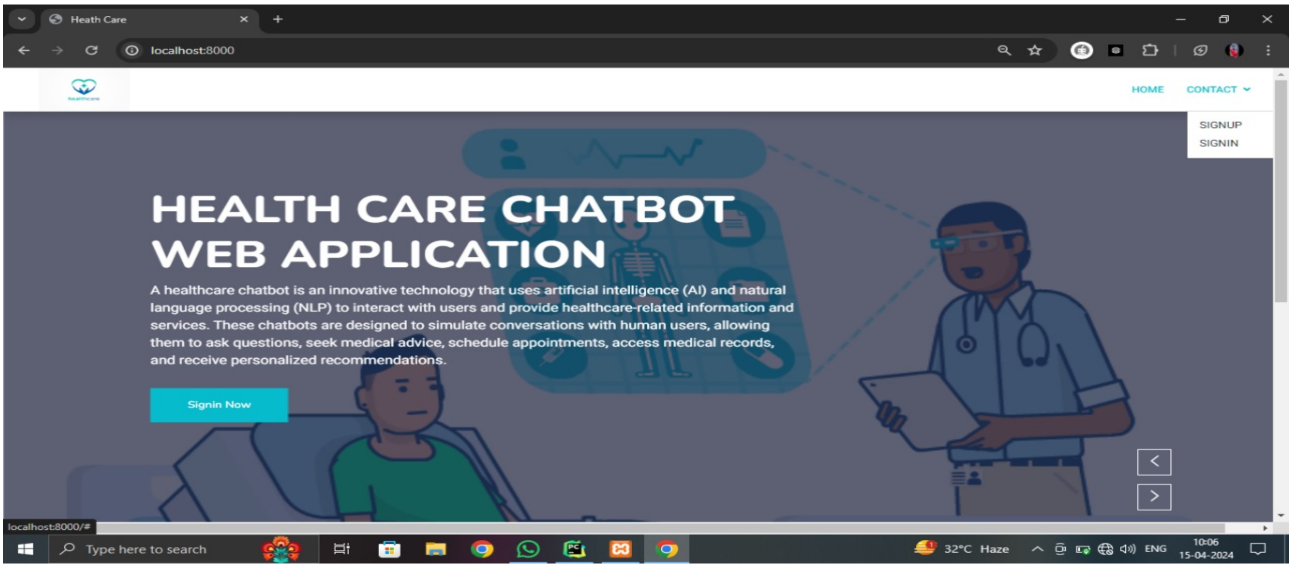
CHAPTER-8

OUTPUT SCREENS



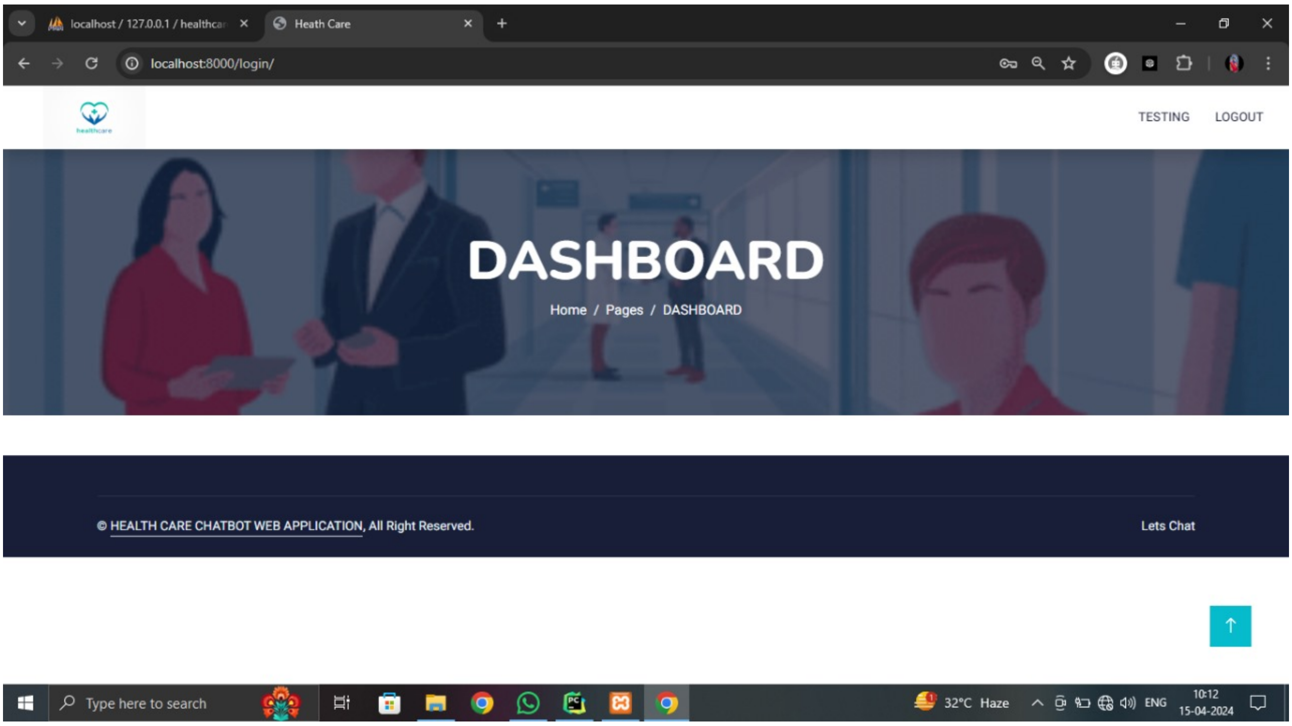
HEALTH CARE CHATBOT WEB APPLICATION

A healthcare chatbot is an innovative technology that uses artificial intelligence (AI) and natural language processing (NLP) to interact with users and provide healthcare-related information and services. These chatbots are designed to simulate conversations with human users, allowing them to ask questions, seek medical advice, schedule appointments, access medical records, and receive personalized recommendations.



HEALTH CARE CHATBOT WEB APPLICATION

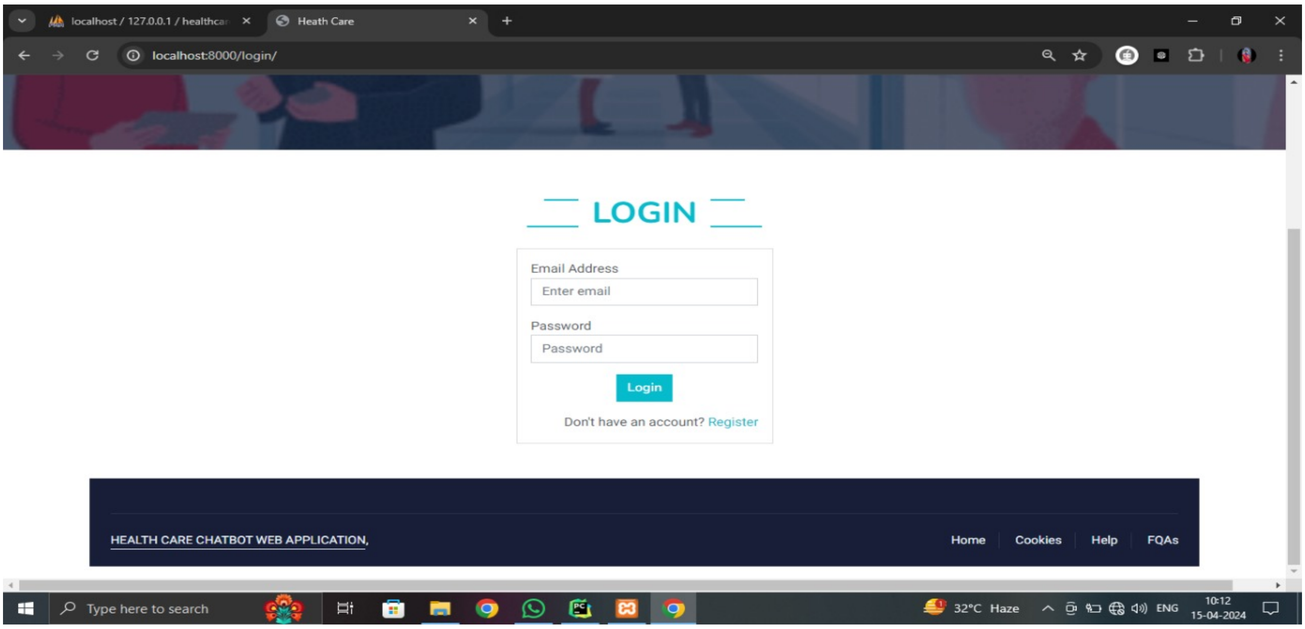
A healthcare chatbot is an innovative technology that uses artificial intelligence (AI) and natural language processing (NLP) to interact with users and provide healthcare-related information and services. These chatbots are designed to simulate conversations with human users, allowing them to ask questions, seek medical advice, schedule appointments, access medical records, and receive personalized recommendations.



DASHBOARD

Home / Pages / DASHBOARD

© HEALTH CARE CHATBOT WEB APPLICATION, All Right Reserved.



localhost:8000/login/

LOGIN

Email Address

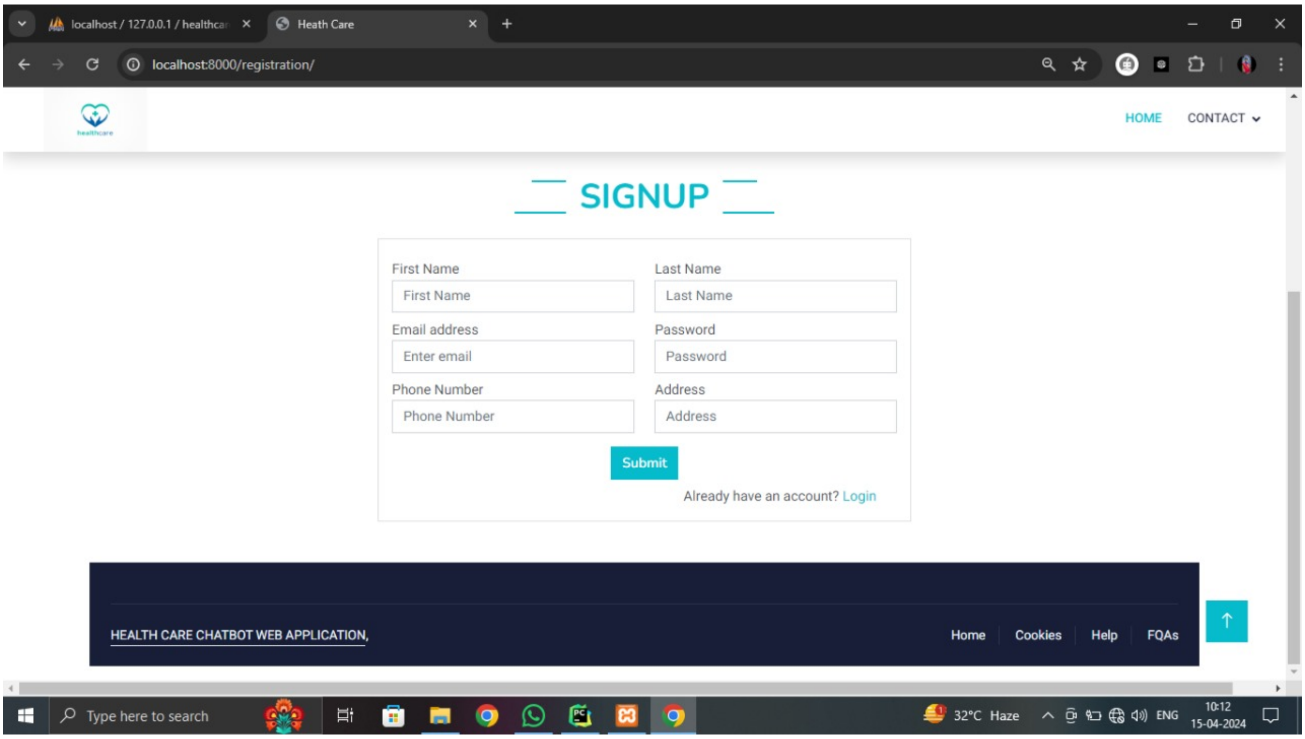
Enter email

Password

Password

Don't have an account? Register

HEALTH CARE CHATBOT WEB APPLICATION,



localhost:8000/registration/

SIGNUP

First Name

First Name

Last Name

Last Name

Email address

Enter email

Password

Password

Phone Number

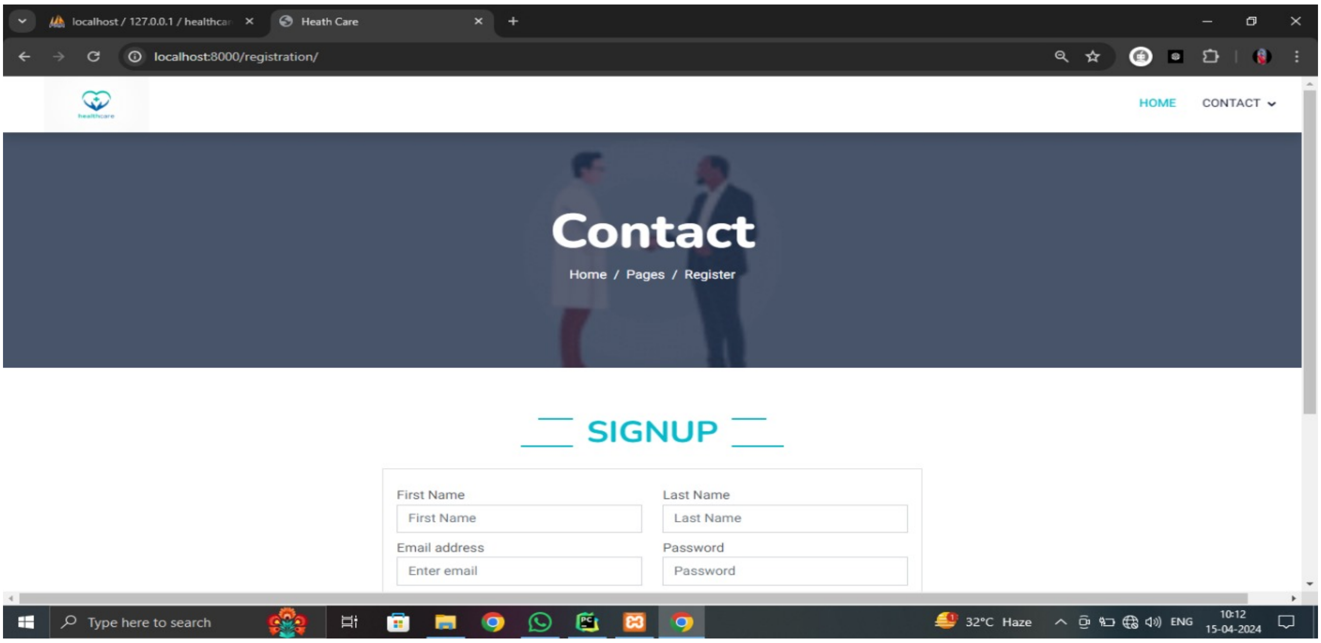
Phone Number

Address

Address

Already have an account? Login

HEALTH CARE CHATBOT WEB APPLICATION,



Contact

Home / Pages / Register

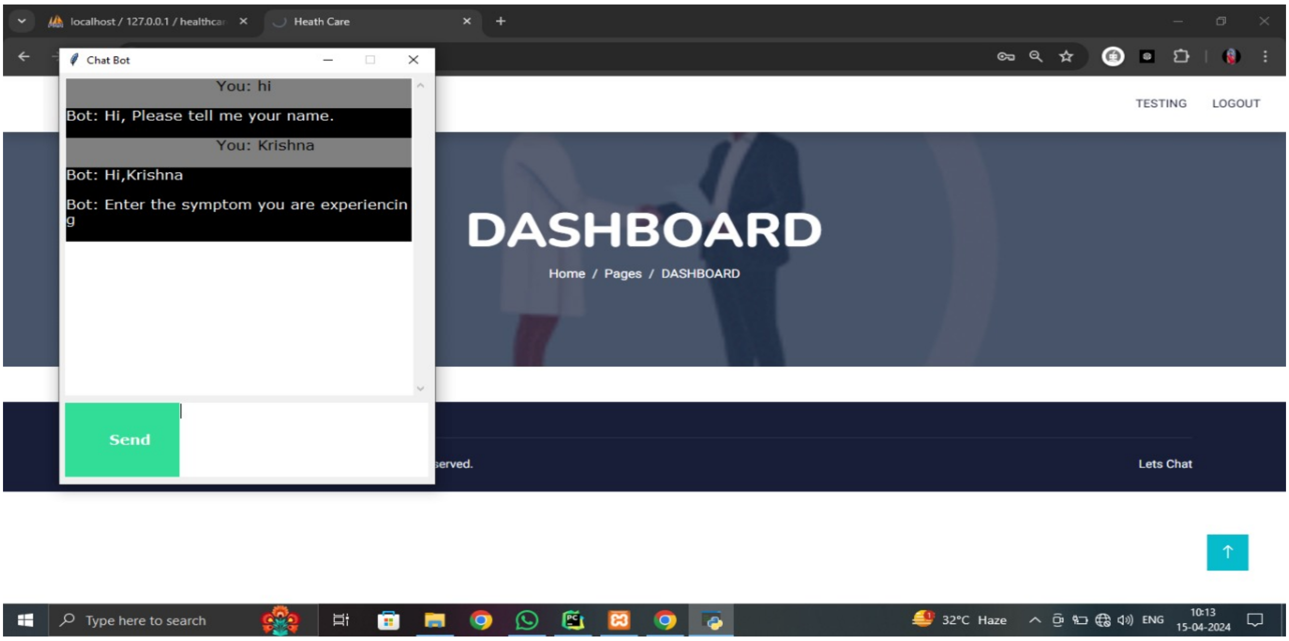
SIGNUP

First Name: First Name

Last Name: Last Name

Email address: Enter email

Password: Password



You: hi

Bot: Hi, Please tell me your name.

You: Krishna

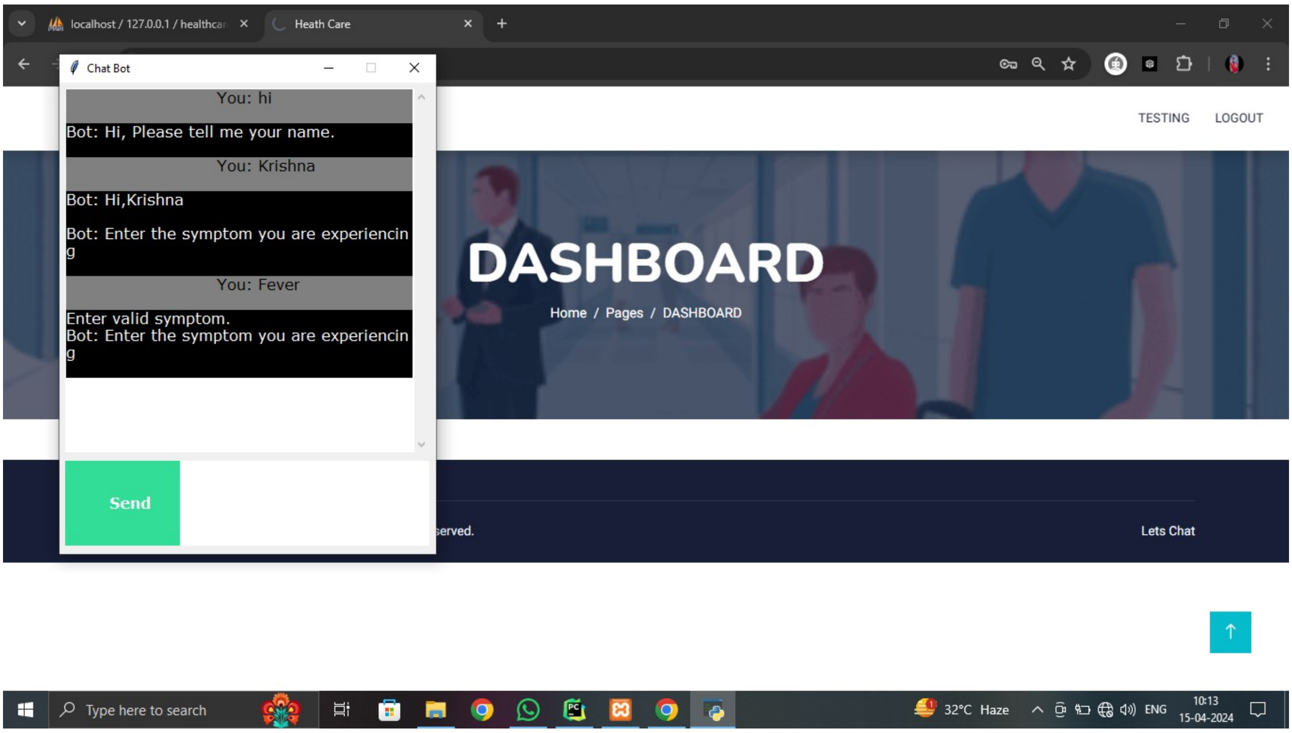
Bot: Hi,Krishna

Bot: Enter the symptom you are experiencin g

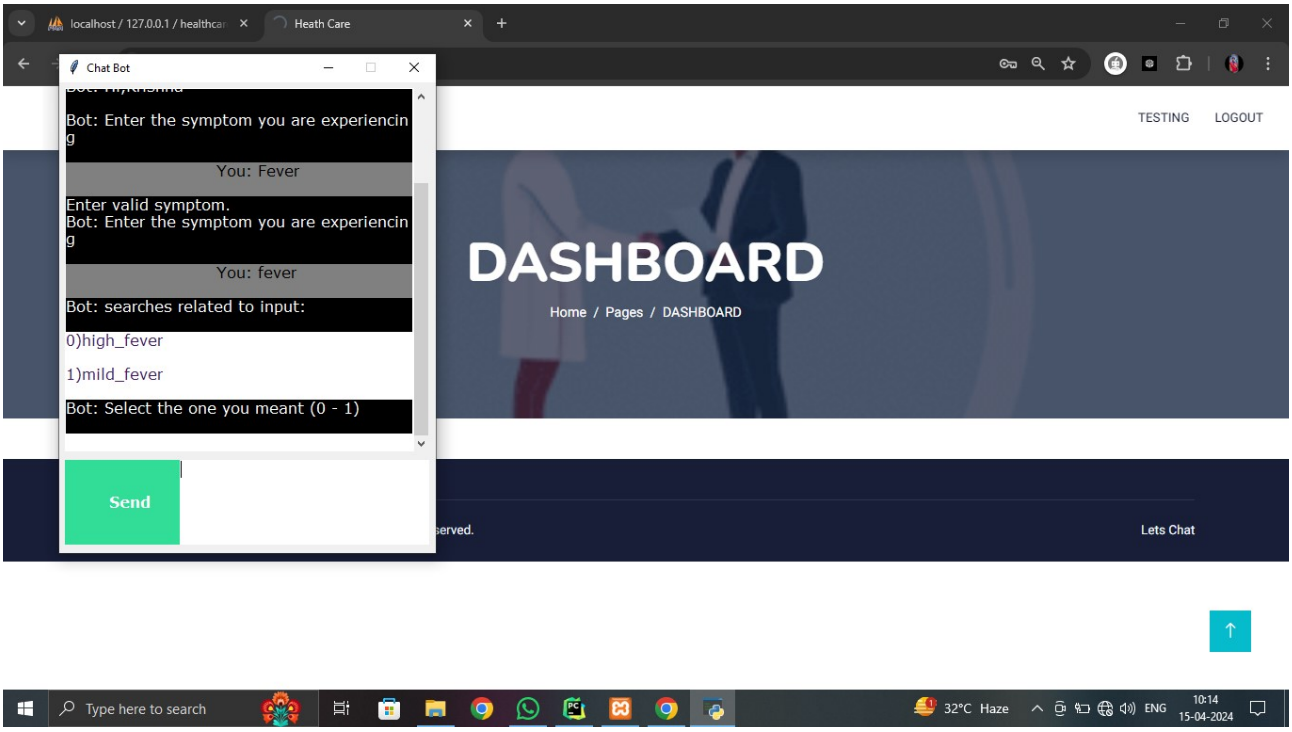
DASHBOARD

Home / Pages / DASHBOARD

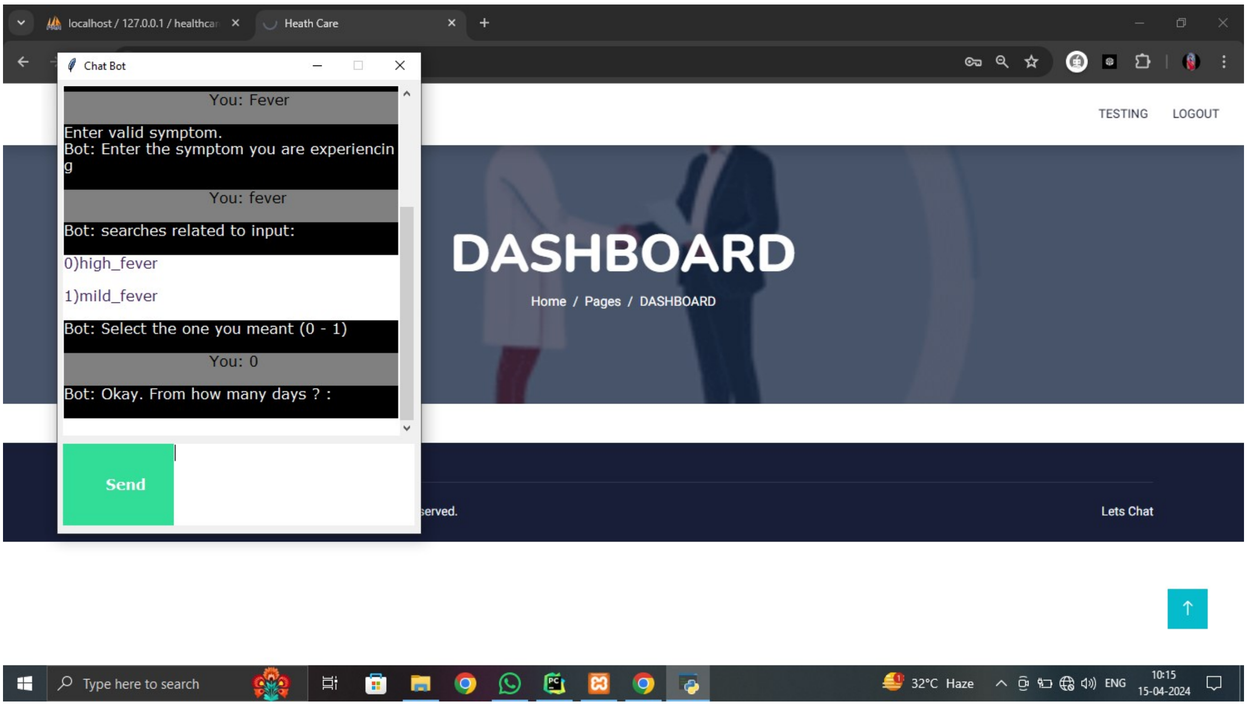
erved.



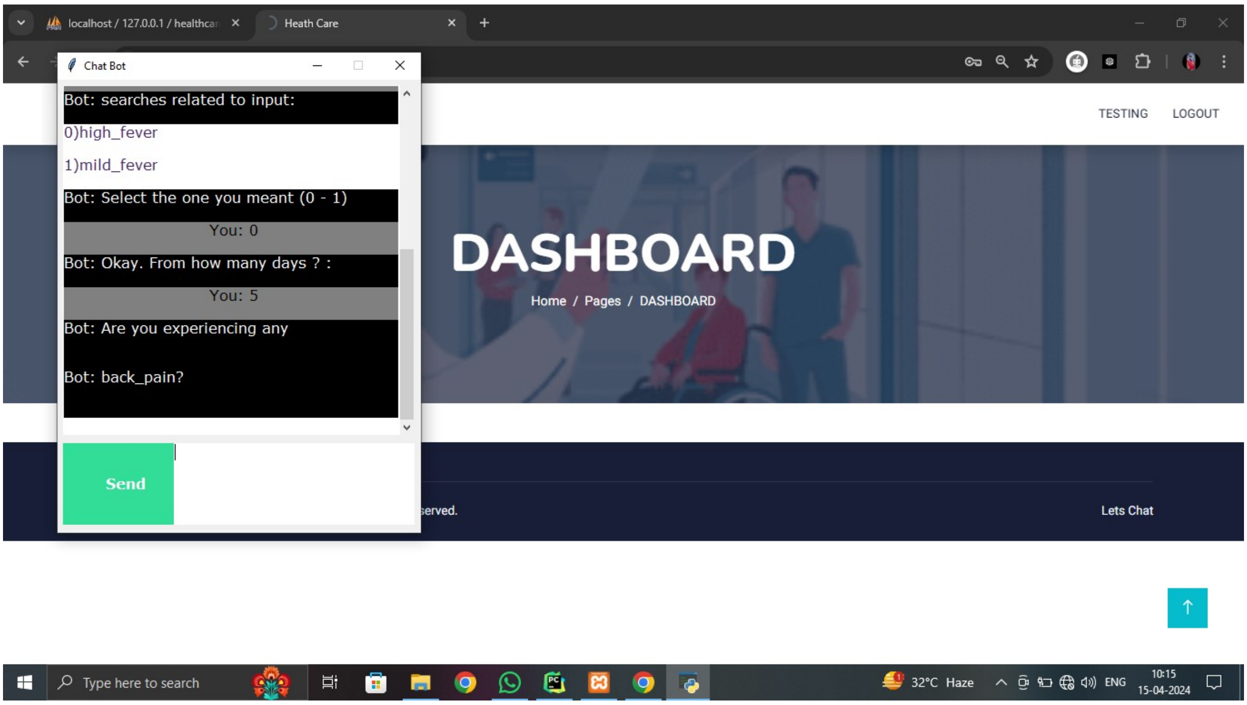
```
You: hi
Bot: Hi, Please tell me your name.
You: Krishna
Bot: Hi,Krishna
Bot: Enter the symptom you are experiencin
g
You: Fever
Enter valid symptom.
Bot: Enter the symptom you are experiencin
g
```



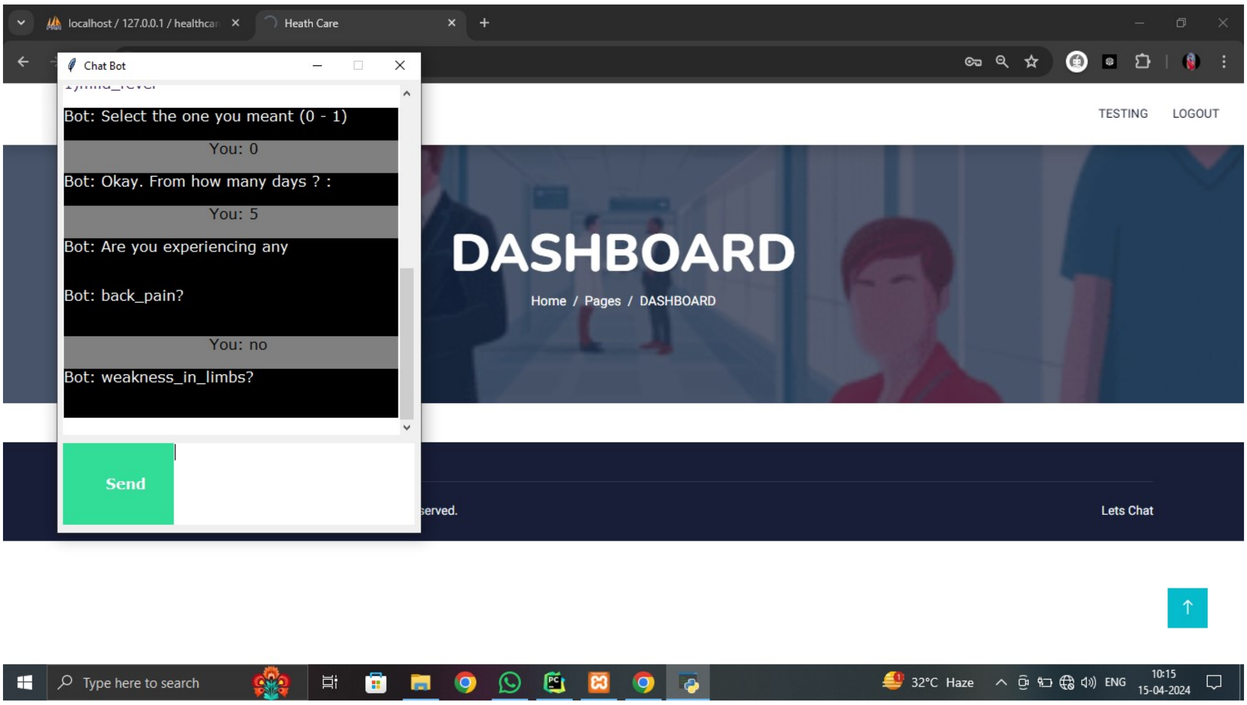
```
Bot: Hi,Krishna
Bot: Enter the symptom you are experiencin
g
You: Fever
Enter valid symptom.
Bot: Enter the symptom you are experiencin
g
You: fever
Bot: searches related to input:
0)high_fever
1)mild_fever
Bot: Select the one you meant (0 - 1)
```



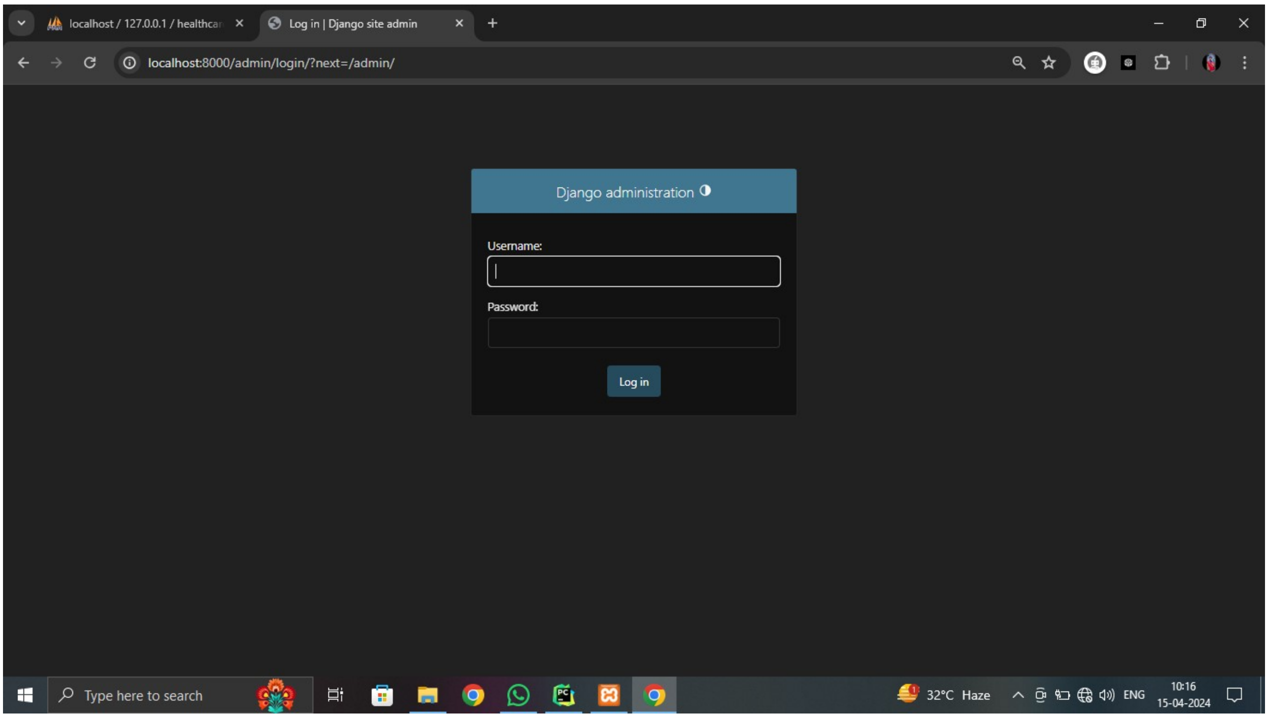
```
You: Fever
Enter valid symptom.
Bot: Enter the symptom you are experiencin
g
You: fever
Bot: searches related to input:
0)high_fever
1)mild_fever
Bot: Select the one you meant (0 - 1)
You: 0
Bot: Okay. From how many days ? :
```



```
Bot: searches related to input:
0)high_fever
1)mild_fever
Bot: Select the one you meant (0 - 1)
You: 0
Bot: Okay. From how many days ? :
You: 5
Bot: Are you experiencing any

Bot: back_pain?
```



```
Bot: Select the one you meant (0 - 1)
                    You: 0
Bot: Okay. From how many days ? :
                    You: 5
Bot: Are you experiencing any

Bot: back_pain?

                    You: no
Bot: weakness_in_limbs?
```

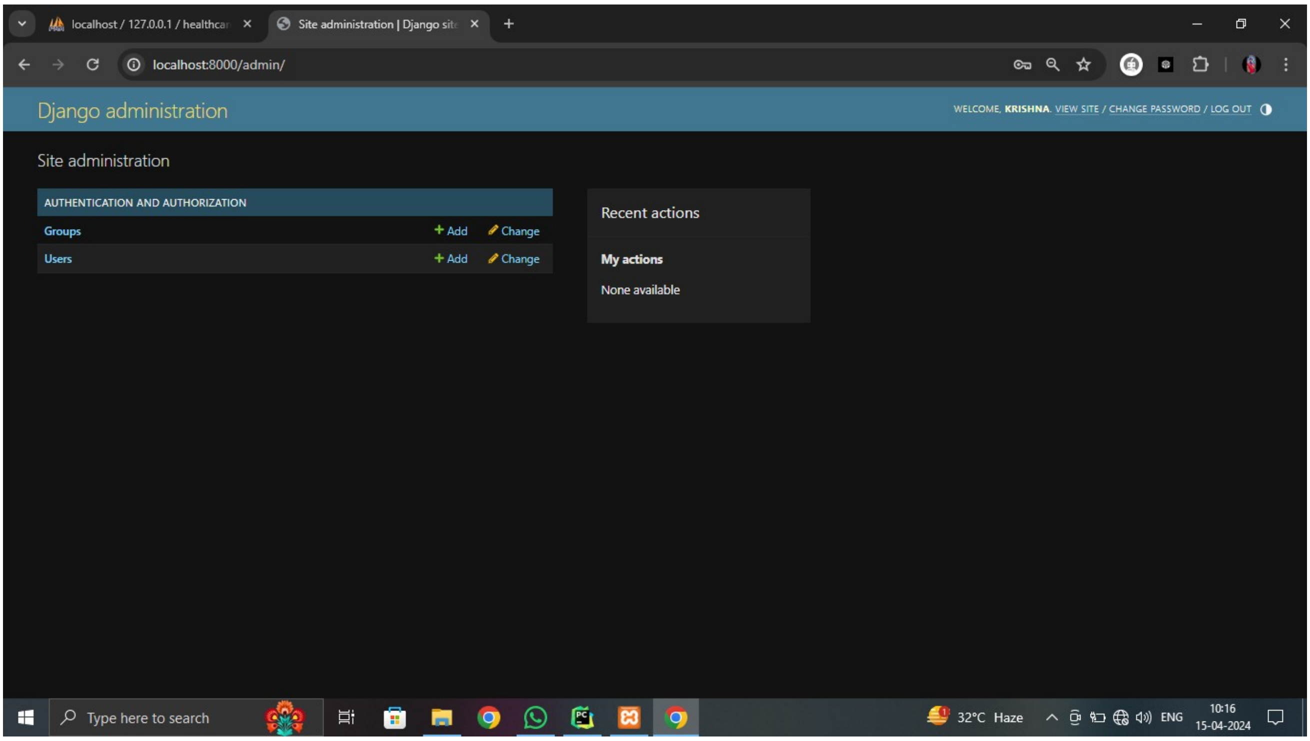


localhost:8000/admin/login/?next=/admin/

Django administration

Username:

Password:



localhost / 127.0.0.1 / healthcar
Site administration | Django site
localhost:8000/admin/
Django administration
WELCOME, KRISHNA. VIEW SITE / CHANGE PASSWORD / LOG OUT
Site administration
AUTHENTICATION AND AUTHORIZATION
Groups
+ Add
Change
Users
+ Add
Change
Recent actions
My actions
None available
Type here to search
32°C Haze
ENG
10:16
15-04-2024

CHAPTER-9

CONCLUSION AND FUTURE ENHANCEMENT

CONCLUSION:

Chat bots are an emerging technology that holds immense potential for the future. With their growing popularity among companies, they are expected to have a lasting impact. It is truly exciting to witness the development of this new domain in technology, as it surpasses previous limitations. We are developing this system in response to the increasing population of our country. Although such systems are already available in other countries, they are not yet widely accessible in our own. The shortage of doctors to meet the needs of patients is a well-known issue, particularly evident in the government hospitals throughout the city. By introducing a medical chat bot, we aim to provide medical assistance to patients when doctors are unavailable, ultimately enhancing the efficiency and performance of the medical industry and reducing the mortality rate

FUTURE ENHANCEMENT:

By incorporating these future enhancements, the Health-Care Chat Bot using SVM and Decision Tree can evolve into a more sophisticated and versatile tool, offering advanced capabilities, personalized support, and seamless integration with healthcare systems for the benefit of users' health and well-being.

CHAPTER-10

REFERENCES

TEXT BOOK REFERENCES:

[1] A. Allen, ‘‘Morphing telemedicine-telecare-telehealth-ehealth,’’ Telemed Today, Special, no. 2000, 2000.

[2] S. L. Murphy, J. Xu, K. D. Kochanek, and E. Arias, ‘‘Mortality in the United States, 2017,’’ *NCHS Data Brief*, no. 328, pp. 1–8, Nov. 2018.

[3] K. Myers, P. Berry, J. Blythe, K. Conley, M. Gervasio, D. L. McGuinness, D. Morley, A. Pfeffer, M. Pollack, and M. Tambe, ‘‘An intelligent personal assistant for task and time management,’’ *AI Mag.*, vol. 28, no. 2, p. 47, 2007.

[4] Flora Amato, Stefano Marrone, “Chatbots meet eHealth: automat zing healthcare”, proceeding of diet, May-2018.

[5] Benilda Eleonor V. Comendador, “Pharmabot: A pediatric generic Medicine consultant Chatbot”, proceeding of the JACE, April 2015.

[6] Divya, Indumathi, Ishwarya, Priyasankari, “A Self Diagnosis Medical Chatbot Using Artificial Intelligence”, proceeding MAT Journal, October-2017.

[7] Tobias Kowatsch,” Text-based Healthcare Chatbots Supporting Patient and Health”, 01 October 2017.

[8] Chin-Yuan Huang, Ming-Chin Yang, Chin-Yu Huang, “A Chatbot-supported Smart Wireless Interactive Healthcare System for Weight Control and Health Promotion”, proceeding of the IEEE, April-2018.

[9] Bouk richa, H., Wachsmuth, I.: Modeling Empathy for a Virtual Human: How, When and to What Extent. The 10th International Conference on Autonomous Agents and Multiagent

Systems-Volume 3. International Foundation for Autonomous Agents and Multiagent Systems, 2011., pp. 1135–1136

[10] Agarwal, R., Gao, G., DesRoches, C., et al.: The Digital Transformation of Healthcare: Current Status and the Road Ahead. Information Systems Research 21, 796-809 (2010).

[11] Aron, A., Aron, E.N., Smollan, D.: Inclusion of Other in the Self Scale and the structure of interpersonal closeness. Journal of Personality and Social Psychology 63, 596-612 (1992).

[12] Bick more, T., Cassell, J.: Social Dialogue with Embodied Conversational Agents. In: Kuppevelt, J.C.J., Bernsen, N.O., (eds.) Advances in Natural Multimodal Dialogue Systems, vol. 30, pp. 23–54. Springer, Dordrecht (2005).

[13] Bick more, T., Gruber, A., Picard, R.: Establishing the computer–patient working alliance in automated health behavior change interventions. Patient Education and Counselling 59, 21-30 (2005).