

A
Mini Project
On
**HOSPITAL MANAGEMENT SYSTEM USING
CHATBOT**

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

BACHELOR OF TECHNOLOGY

In
COMPUTER SCIENCE AND ENGINEERING

By
BHAVANA VANGA (207R1A05P5)
JEERLA PRANAY (207R1A05L4)
DARIPELLI AKSHAY (207R1A05K6)

Under the Guidance of

Mrs.S.APARNA

(Assistant Professor)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi) Recognized Under Section 2(f) & 12(B) of the UGC Act, 1956, Kandlakoya (V), Medchal Road, Hyderabad-501401.

2020-2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

This is to certify that the project entitled **“HOSPITAL MANAGEMENT SYSTEM USING CHATBOT”** being submitted by **BHAVANA VANGA (207R1A05P5), PRANAY JEERLA (207R1A05L4) & DARIPELLI AKSHAY (207R1A05K6)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2023-24.

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

Mrs.S.APARNA

(Assistant Professor)
INTERNAL GUIDE

Dr. A. Raji Reddy

DIRECTOR

Dr. K. Srujan Raju
HoD CSE

EXTERNAL EXAMINAR

Submitted for viva voice Examination held on _____

ACKNOWLEDGEMENT

Apart from the efforts of me, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard to my guide **Mrs.S.Aparna**, Assistant Professor for her exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by her shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to the Project Committee Review (PRC) **G. Vinesh Shanker, Dr.J. Narasimha Rao, Ms. Shilpa &Dr. K. Maheswari** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju**, Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A . Raji Reddy** , Director for being cooperative throughout the course of this project. We also express our sincere gratitude to **Sri. Ch. Gopal Reddy**, Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

V.BHAVANA (207R1A05P5)

J.PRANAY (207R1A05L4)

D.AKSHAY (207R1A05K6)

ABSTRACT

Through chatbots 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. 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 in individuals. 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 x7, 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.

In this project we are designing hospital systems where chatbot will accept symptoms from patient and then suggest doctor availability date and time for that symptoms. To send SMS to doctor we need to have mobile service provider without that this service will not work and you are asking to generate prescription by chatbot but we don't have diseases and related medicines to generate prescription so we are not doing this but chatbot will suggest doctor by taking symptoms from patients.

LIST OF FIGURES/TABLES

FIGURE NO	FIGURE NAME	PAGE NO
Figure 3.1	Project Architecture of Hospital Management System using Chatbot	6
Figure 3.2	UseCase Diagram of Hospital Management System using Chatbot	7
Figure 3.3	Class Diagram of Hospital Management System using Chatbot	8
Figure 3.4	Sequence diagram of Hospital Management System using Chatbot	9
Figure 3.5	Activity diagram of Hospital Management System using Chatbot	10

LIST OF SCREENSHOTS

SCREENSHOT NO.	SCREENSHOT NAME	PAGE NO.
Screenshot 5.1	Home Page	20
Screenshot 5.2	Admin Login	20
Screenshot 5.3	Doctor SignUp	21
Screenshot 5.4	Doctor Details	21
Screenshot 5.5	Patient SignUp	22
Screenshot 5.6	Patient Details	22
Screenshot 5.7	Time Table Fixation	23
Screenshot 5.8	Prescription Screen	23
Screenshot 5.9	Patient Interaction	24

TABLE OF CONTENTS

ABSTRACT	i
LIST OF FIGURES	ii
LIST OF SCREENSHOTS	iii
1.INTRODUCTION	
1.1 PROJECT SCOPE	1
1.2 PROJECT PURPOSE	1
1.3 PROJECT FEATURES	1
2.SYSTEM ANALYSIS	
2.1 PROBLEM DEFINITION	2
2.2 EXISTING SYSTEM	2
2.2.1 LIMITATIONS OF THE EXISTING SYSTEM	3
2.3 PROPOSED SYSTEM	3
2.3.1 ADVANTAGES OF PROPOSED SYSTEM	3
2.4 FEASIBILITY STUDY	4
2.4.1 ECONOMIC FEASIBILITY	4
2.4.2 TECHNICAL FEASIBILITY	5
2.4.3 SOCIAL FEASIBILITY	5
2.5 HARDWARE & SOFTWARE REQUIREMENTS	5
2.5.1 HARDWARE REQUIREMENTS	5
2.5.2 SOFTWARE REQUIREMENTS	5
3.ARCHITECTURE	
3.1 PROJECT ARCHITECTURE	6
3.2 DESCRIPTION	6
3.3 USE CASE DIAGRAM	7
3.4 CLASS DIAGRAM	8
3.5 SEQUENCE DIAGRAM	9
3.6 ACTIVITY DIAGRAM	10
4.IMPLEMENTATION	
4.1 SAMPLE CODE	11

TABLE OF CONTENTS

5.SCREENSHOTS	20
6.TESTING	
6.1 INTRODUCTION TO TESTING	25
6.2 TYPES OF TESTING	25
6.2.1 UNIT TESTING	25
6.2.2 INTEGRATION TESTING	25
6.2.3 FUNCTIONAL TESTING	26
6.3 TEST CASES	27
6.3.1 CLASSIFICATION	27
7.CONCLUSION & FUTURE SCOPE	
7.1 PROJECT CONCLUSION	28
7.2 FUTURE SCOPE	28
8.REFERENCES	
8.1 REFERENCES	29
8.2 GITHUB LINK	29

1.INTRODUCTION

1. INTRODUCTION

1.1 PROJECT SCOPE:

This project entails the development of a healthcare chatbot system focusing on text based interactions. It serves three primary user roles: Users, Admins, and Doctors. Users can register, log in, view hospital and doctor details, and interact with the chatbot for symptom-based doctor availability recommendations. Admins manage the chatbot's knowledge base, user profiles, and hospital/doctor information. Doctors access patient details, generate prescriptions, and manage schedules. The system utilizes Python, MySQL, and Django. Future enhancements may involve SMS integration and expanding the chatbot's medical knowledge base.

1.2 PROJECT PURPOSE:

The primary objective of this project is to develop a healthcare system with a chatbot interface, designed to assist patients in swiftly identifying doctor availability based on their presented symptoms. While it aims to enhance healthcare accessibility and streamline the appointment process, certain limitations must be acknowledged. Nevertheless, the core mission remains to provide users with a user-friendly platform that employs symptom-based recommendations to connect patients with available doctors, ultimately simplifying the initial healthcare consultation process.

1.3 PROJECT FEATURES:

The project will create a system that uses advanced technology (like 5G and smart devices) to help people with diabetes. It will let them securely share their health data, like blood sugar levels, with their doctors. Patients can even have virtual meetings with their doctors. It's like having a smart helper for diabetes that keeps everything private and secure. The project will make sure it follows all the rules for keeping health information safe, and it will keep improving over time to help people even more.

2.SYSTEM ANALYSIS

2.SYSTEM ANALYSIS

System analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, “what must be done to solve the problem?”

2.1 PROBLEM DEFINITION

This project addresses the problem of inefficient healthcare access and delayed medical consultations. Individuals often struggle to find the right doctors promptly based on their symptoms, resulting in delayed treatment. Additionally, the absence of user-friendly appointment scheduling platforms compounds the issue. While SMS notifications to doctors and automated prescription generation are desirable features, their absence due to technical and data limitations hinders effective healthcare delivery. The project aims to resolve these challenges by creating a healthcare chatbot system that simplifies the process of connecting patients with available doctors based on their symptoms.

2.2 EXISTING SYSTEM

Currently, several hospitals have adopted chatbots for their management systems to enhance efficiency, support, engagement with patients, and healthcare professionals in different ways.

- **Patient appointment scheduling:** Hospitals are using chatbots to simplify the appointment scheduling process for patients. Patients can easily schedule appointments, check availability, and receive reminders before their scheduled visit.
- **Data collection and analysis:** Hospitals has database to collect and analyze data from patients. This information includes personal and medical history, prescription information, and test results.

2.2.1 DISADVANTAGES OF EXISTING SYSTEM

Following are disadvantages of existing system:

- **Lack of Personalization:** Chatbots cannot provide personalized recommendations or suggestions for patients as they do not have access to the patient's medical history or records.
- **Security Concerns:** Chatbots may pose a security risk if they are not programmed with adequate security measures. They may inadvertently reveal private patient information to unauthorized personnel, which can lead to serious consequences.

2.3 PROPOSED SYSTEM

This is an AI-based tool developed to manage hospital operations using a chatbot interface. This system aims to make hospital management more efficient, improve patient satisfaction and reduce human errors.

We are using the latest technology Django for faster retrieval of answers for the queries by the chatbot. For the website creation we are using html, css and javascript.

- **Chat Support:** Chatbots resolve the query of user regarding any health issues, appointments, etc.
- **Report Retrieval:** Patients can access their medical reports and test results through the chatbot interface. The chatbot will provide a secure link to access the report.

2.3.1 ADVANTAGES OF PROPOSED SYSTEM

Following are advantages of proposed system:

- **Improved Patient Satisfaction:** The chatbot interface will provide patients with quick and convenient access to hospital services. This will improve patient satisfaction by reducing wait times and improving communication.
- **Reduced Human Errors:** The chatbot interface will reduce the likelihood of human errors by automating repetitive tasks, such as inventory monitoring and appointment scheduling.

2.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and a 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. Three key considerations involved in the feasibility analysis:

- Economic Feasibility
- Technical Feasibility
- Social Feasibility

2.4.1 ECONOMICAL FEASIBILITY

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on a project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.

The following are some of the important financial questions asked during preliminary investigation:

- The costs conduct a full system investigation.
- The cost of the hardware and software.
- The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also, all the resources are already available, it gives an indication that the system is economically possible for development.

2.4.2 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. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.4.3 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.

2.5 HARDWARE AND SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS:

Hardware interfaces specify the logical characteristics of each interface between the software product and the hardware components of the system.

The following are some hardware requirements.

- i3 Processor Based Computer or higher
- Memory: 1 GB
- Hard Drive: 50 GB
- Monitor
- Internet Connection

2.5.2 SOFTWARE REQUIREMENTS

Software Requirements specifies the logical characteristics of each interface and software components of the system.

The following are some software requirements.

- Windows 7 or higher
- WAMP Server
- Notepad++
- My SQL 5.5
- Google Chrome Browser

3.ARCHITECTURE

3.1 PROJECT ARCHITECTURE

This project architecture shows the procedure followed for classification, starting from input to final prediction.

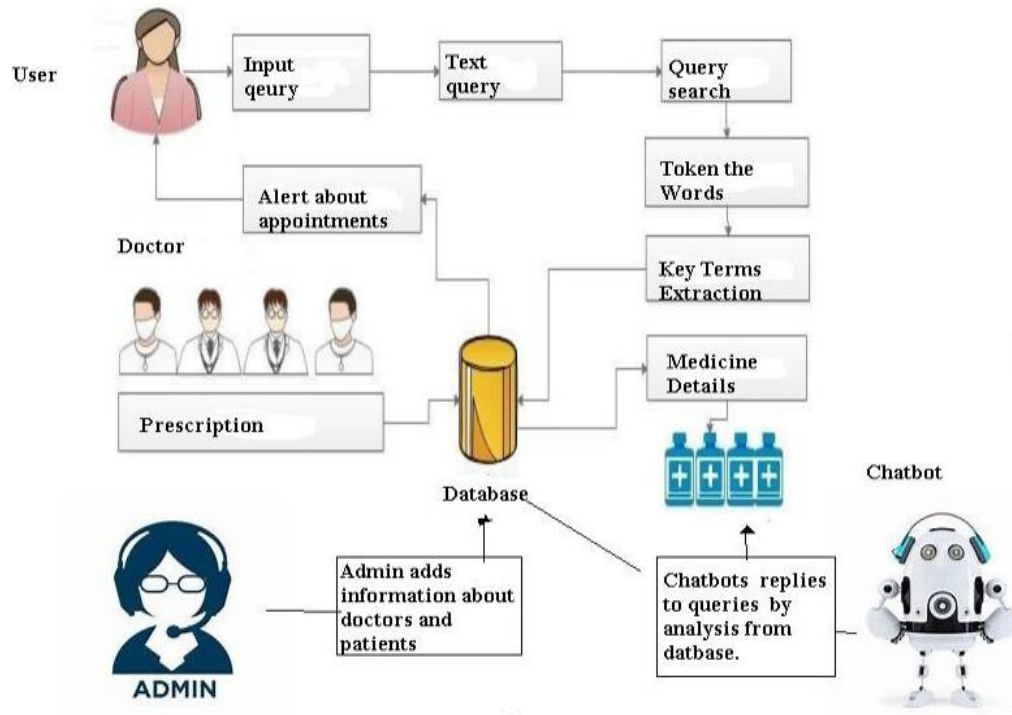


Figure 3.1: Project Architecture Of Hospital Management System Using Chatbot

3.2 DESCRIPTION

We are designing hospital systems where chatbot will accept symptoms from patient and then suggest doctor availability date and time for that symptoms. To send SMS to doctor we need to have mobile service provider without that this service will not work and you are asking to generate prescription by chatbot but we don't have diseases and related medicines to generate prescription so we are not doing this but chatbot will suggest doctor by taking symptoms from patients.

3.3 USE CASE DIAGRAM

In the use case diagram, we have basically one actor who is the user in the trained model. A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has. The use cases are represented by either circles or ellipses. The actors are often shown as stick figures.

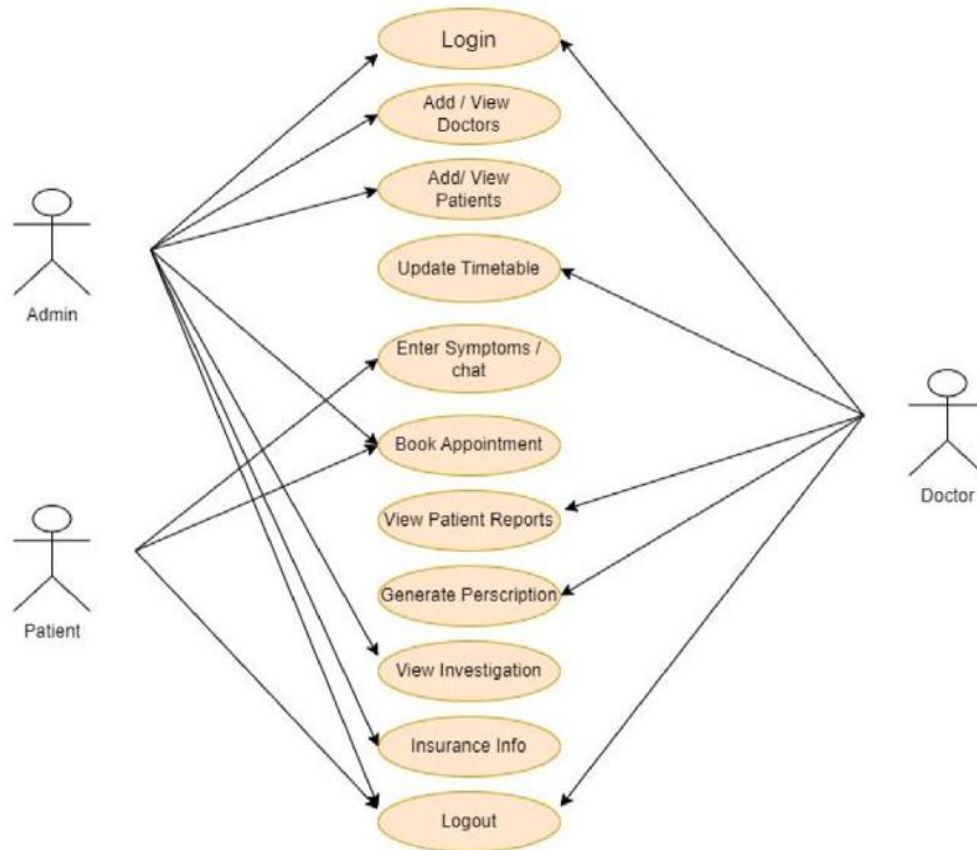


Figure 3.2: Use Case Diagram of Hospital Management System Using Chatbot

3.4 CLASS DIAGRAM

Class diagram 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 objects.

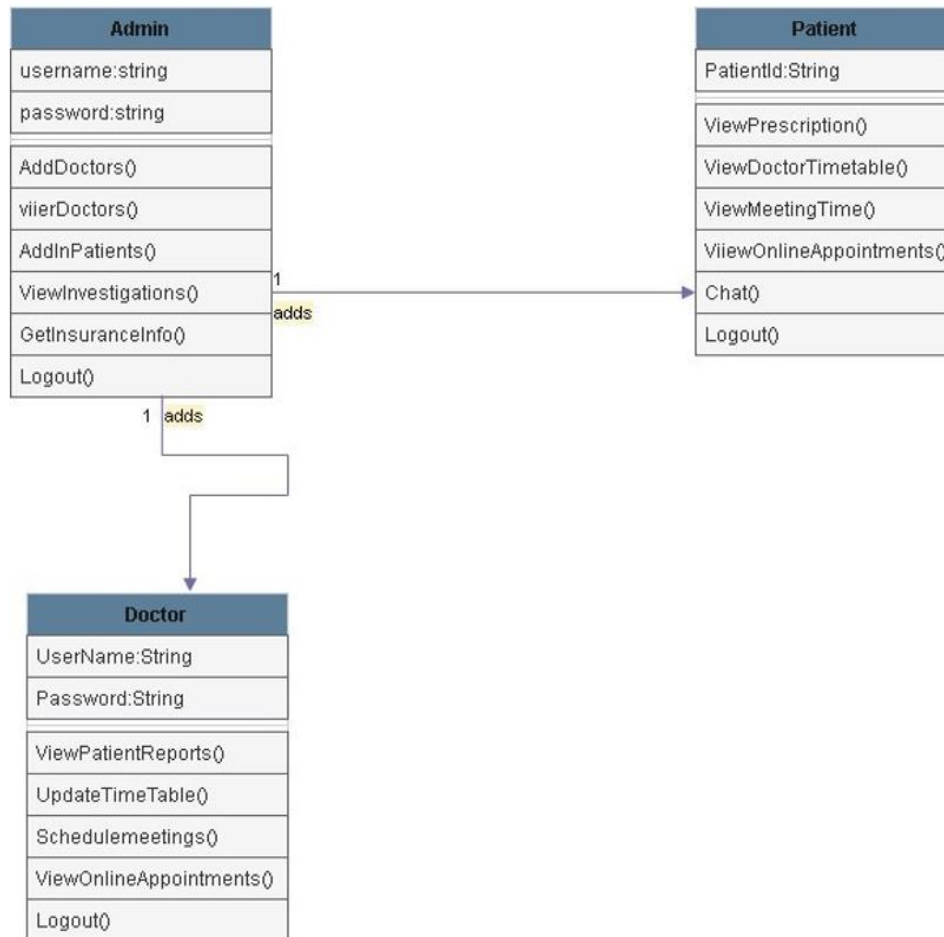


Figure 3.3: Class Diagram of Hospital Management System Using Chatbot

3.5 SEQUENCE DIAGRAM

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the logical view of the system under development.

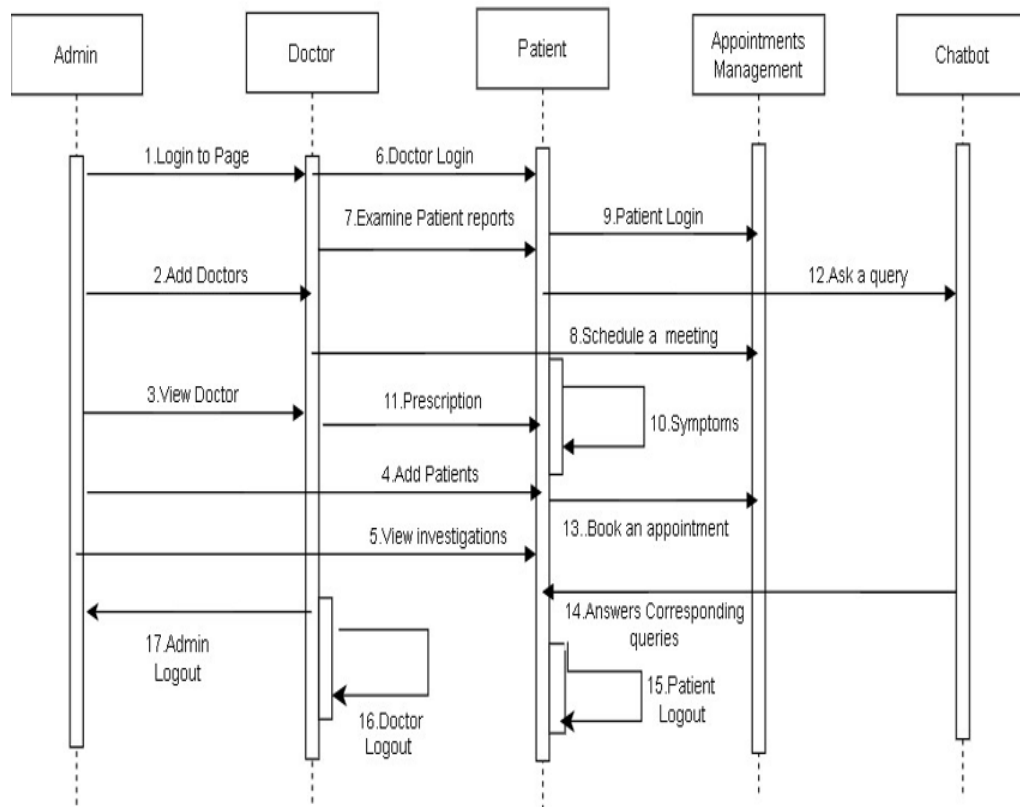


Figure 3.4: Sequence Diagram of Hospital Management System Using Chatbot

3.6 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. They can also include elements showing the flow of data between activities through one or more data stores.



Figure 3.5: Activity Diagram of Hospital Management System Using Chatbot

4.IMPLEMENTATAION

4.1 SAMPLE CODE

```

from django.shortcuts import render
from django.template import RequestContext
from django.contrib import messages
import pymysql
from django.http import HttpResponse
from django.core.files.storage import FileSystemStorage
import os
from datetime import date
import os
global userid, pnameValue, pdateValue
def Chat(request):
    if request.method == 'GET':
        return render(request, 'Chat.html', {})
def searchQuery(query, data):
    arr = data.split(" ")
    qry = query.split(" ")
    count =0
    for i in range(len(qry)):
        for j in range(len(arr)):
            if qry[i] == arr[j]:
                count = count + 1
    return count
def getTime(doctor):
    available = "Not Available, Not Available, Not Available"
    con=pymysql.connect(host='127.0.0.1',port=3306,user= root,password = 'root',
    database = 'HospitalDB',charset='utf8')
    with con:
        for row in rows:
            cur = con.cursor()

```

```

    if row[0] == doctor:
        available = row[0]+","+row[1]+","+row[2]
        break
    return available

def ChatData(request):
    if request.method == 'GET':
        query = request.GET.get('mytext', False)
        desc = "Unable to recognize your query"
        score = 0
        doctor = "none"
        con=pymysql.connect(host='127.0.0.1',port =3306,user ='root',password =
        'root', data= 'HospitalDB',charset='utf8')
        with con:
            cur = con.cursor()
            cur.execute("select username,speciality FROM adddoctor")
            rows = cur.fetchall()
            for row in rows:
                description = row[1]
                count = searchQuery(query.lower(), description.lower())
                if count > score:
                    score = count
                    desc = description
                    doctor = row[0]
                    details = getTime(doctor).split(",")
                    output="DoctorName:"+details[0]+"r\nAvailabilityDate
                    Availability Slot Timing: "+details[2]
                return HttpResponse(output, content_type="text/plain")

def Prescription(request):
    if request.method == 'GET':
        return render(request, 'ViewPatientPrescription.html', context)

```



```

        global pnameValue, pdateValue
        global userid
        pnameValue = request.GET['pname']
        pdateValue = request.GET['pdate']
        context= {'data':"Patient Name: "+pnameValue}
        return render(request, 'Prescription.html', context)
def PrescriptionAction(request):
    global pnameValue, pdateValue, userid
    prescription = request.POST.get('t1', False)
    db_connection = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root',
    password = 'root', database = 'HospitalDB',charset='utf8')
    db_cursor = db_connection.cursor()
    student_sql_query = "update addpatient set prescription='"+prescription+"
    where visit_date='"+pdateValue+" and patient_id='"+pnameValue+"
    and prescription='Pending'"
    db_cursor.execute(student_sql_query)
    db_connection.commit()
    print(db_cursor.rowcount, "Record Inserted")
    if db_cursor.rowcount == 1:
        context= {'data':Prescription generated successfully for patient '+pnameValue}
        return render(request, 'Doctorscreen.html', context)
    else:
        context= {'data':'Error in generating prescription'}
        return render(request, 'DoctorScreen.html', context)
def ViewPatientReport(request):
    if request.method == 'GET':
        global usinging
        output = ""
        con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password =
        'root', database = 'HospitalDB',charset='utf8')

```

with con:

```

cur = con.cursor()
cur.execute("select * FROM addpatient where consult_doctor='"+userid+"'")
rows = cur.fetchall()
for row in rows:
    output+='<tr><td><font size="" color="black">'+str(row[0])+'</td>'
    output+='<td><font size="" color="black">'+str(row[1])+'</td>'
    output+='<td><font size="" color="black">'+str(row[2])+'</td>'
    output+='<td><font size="" color="black">'+str(row[3])+'</td>'
    output+='<td><font size="" color="black">'+str(row[4])+'</td>'
    output+='<td><font size="" color="black">'+str(row[5])+'</td>'
    if row[6] != 'Pending':
        output+='<td><font size="" color="black">'+str(row[6])+'</td>'
    else:
        output+=
        '<td><a href=\Prescription?pname='+str(row[0])+'&pdate='+str(row[7])
        +'\'><font size=3 color=black>Click Here</font></a></td>'
        output+='<td><font size="" color="black">'+str(row[7])+'</td>'
        context= {'data':output}
    return render(request, 'ViewPatientReport.html', context)

```

def ViewPrescription(request):

```

if request.method == 'GET':
    output = ""
    con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = 'root',
    database = 'HospitalDB',charset='utf8'
    with con:
        cur.execute("select * FROM addpatient")
        rows = cur.fetchall()
        for row in rows:
            cur = con.cursor()

```

```

output+='<tr><td><font size="" color="black">'+str(row[0])+</td>'
output+='<td><font size="" color="black">'+str(row[1])+</td>'
output+='<td><font size="" color="black">'+str(row[2])+</td>'
output+='<td><font size="" color="black">'+str(row[3])+</td>'
output+='<td><font size="" color="black">'+str(row[4])+</td>'
output+='<td><font size="" color="black">'+str(row[5])+</td>'
output+='<td><font size="" color="black">'+str(row[6])+</td>'
output+='<td><font size="" color="black">'+str(row[7])+</td>'
context= {'data':output}

return render(request, 'ViewPrescription.html', context)

def ViewHospitalDetails(request):
    if request.method == 'GET':
        output = ""
        con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root',
        password = 'root', database = 'HospitalDB',charset='utf8')
        with con:
            cur = con.cursor()
            cur.execute("select * FROM adddoctor")
            rows = cur.fetchall()

        for row in rows:
            output+='<tr><td><font size="" color="black">'+str(row[0])+</td>'
            output+='<td><font size="" color="black">'+str(row[1])+</td>'
            output+='<td><font size="" color="black">'+str(row[2])+</td>'
            output+='<td><font size="" color="black">'+str(row[3])+</td>'
            output+='<td><font size="" color="black">'+str(row[4])+</td>'
            output+='<td><font size="" color="black">'+str(row[5])+</td>'
            output+='<td><font size="" color="black">'+str(row[7])+</td>'
            output+='<td><font size="" color="black">'+str(row[8])+</td>'
            context= {'data':output}

        return render(request, 'ViewHospitalDetails.html', context)

```

```

def OnlineAppointmentsAction(request):
    if request.method == 'POST':
        global userid
        available_date = request.POST.get('t1', False)
        slot = request.POST.get('t2', False)
        db_connection = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root'
        ,password = 'root', database = 'HospitalDB',charset='utf8')
        db_cursor = db_connection.cursor()
        student_sql_query="INSERT INTO
        db_cursor.execute(student_sql_query)
        db_connection.commit()
        print(db_cursor.rowcount, "Record Inserted")
        if db_cursor.rowcount == 1:
            context= {'data':'Online Appointments time updated'}
            return render(request, 'OnlineAppointments.html', context)
        else:
            context= {'data':'Error in updating Online Appointments details'}
            return render(request, 'OnlineAppointments.html', context)

def OnlineAppointments(request):
    if request.method == 'GET':
        return render(request, 'OnlineAppointments.html', {})

def ScheduleMeetingsAction(request):
    if request.method == 'POST':
        global userid
        available_date = request.POST.get('t1', False)
        slot = request.POST.get('t2', False)
        db_connection = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root',
        password = 'root', database = 'HospitalDB',charset='utf8')
        db_cursor = db_connection.cursor()
        student_sql_query="INSERT NTO meetings(username,meeting_date,available
        slots)  VALUES('"+str(userid)+"','"+available_date+"','"+slot+"')"

```

```

        db_cursor.execute(student_sql_query)
        db_connection.commit()
        print(db_cursor.rowcount, "Record Inserted")
        if db_cursor.rowcount == 1:
            context= {'data':'Meetings time updated'}
            return render(request, 'ScheduleMeetings.html', context)
        else:
            context= {'data':'Error in updating meeting details'}
            return render(request, 'ScheduleMeetings.html', context)
def ScheduleMeetings(request):
    if request.method == 'GET':
        return render(request, 'ScheduleMeetings.html', {})
def UpdateTimeTableAction(request):
    if request.method == 'POST':
        global userid
        available_date = request.POST.get('t1', False)
        slot = request.POST.get('t2', False)
        db_connection = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root',
        password = 'root', database = 'HospitalDB',charset='utf8')
        db_cursor = db_connection.cursor()
        student_sql_query = "INSERT INTO timetable1(username,time_table,available
        _slots)  VALUES('"+str(userid)+"','"+available_date+"','"+slot+"')"
        db_cursor.execute(student_sql_query)
        db_connection.commit()
        output += '<option value="'+row[0]+'>'+row[0]+'</option>'
        output += "</select></td></tr>"
        context= {'data':output}
        render(request, 'PatientSignup.html', context)
def AddDoctor(request):
    if request.method == 'GET':
        return render(request, 'AddDoctor.html', {})

```

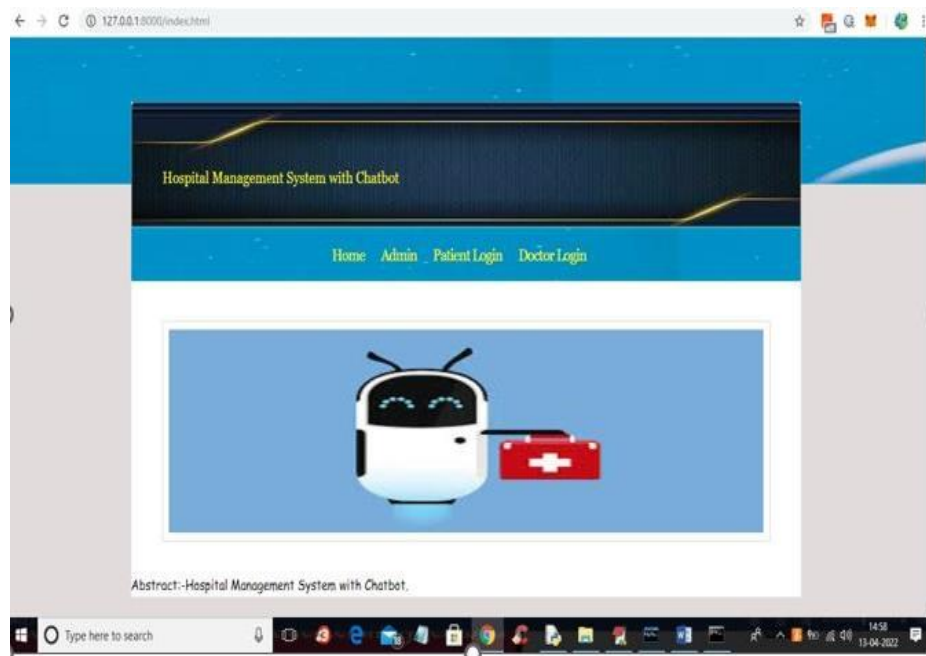
```

def AdminLogin(request):
    if request.method == 'GET':
        return render(request, 'AdminLogin.html', {})
def AdminLoginAction(request):
    if request.method == 'POST':
        global userid
        user = request.POST.get('t1', False)
        password = request.POST.get('t2', False)
        if user == "admin" and password == "admin":
            context= {'data':'Welcome '+user}
            return render(request, 'AdminScreen.html', context)
        else:
            context= {'data':'Invalid Login'}
            return render(request, 'AdminLogin.html', context)
def PatientSignupAction(request):
    if request.method == 'POST':
        today = date.today()
        disease = request.POST.get('t1', False)
        age = request.POST.get('t2', False)
        contact = request.POST.get('t3', False)
        fee = request.POST.get('t4', False)
        doctor = request.POST.get('t5', False)
        pid = request.POST.get('t6', False)
        if pid == "0":
            username = request.POST.get('t1', False)
            password = request.POST.get('t2', False)
            status = 'none'
            con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password =
            'root', database = 'HospitalDB',charset='utf8')
            with con:
                cur = con.cursor()

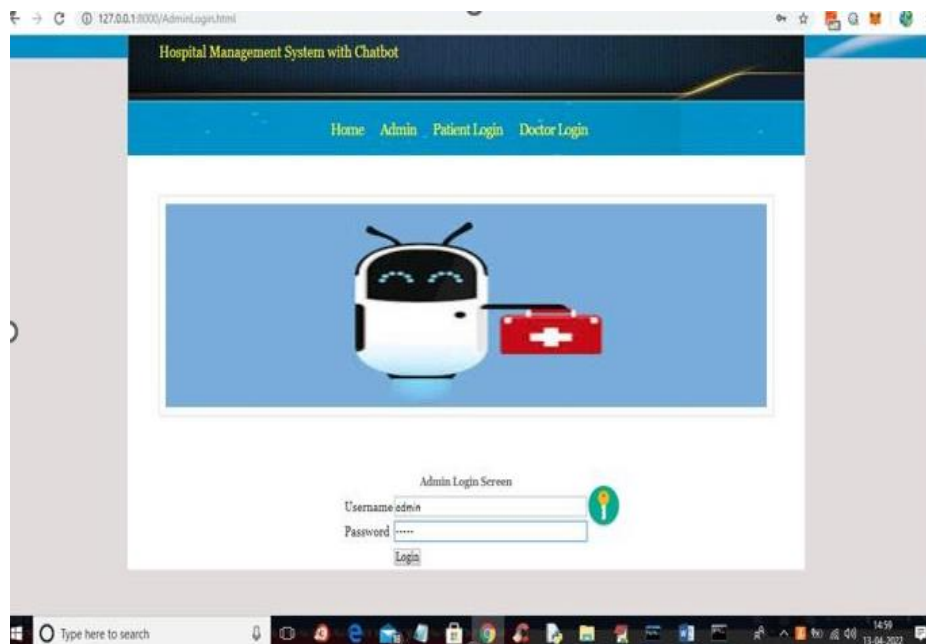
```

```
cur.execute("select username,password FROM adddoctor")
rows = cur.fetchall()
for row in rows:
    if row[0] == username and row[1] == password:
        userid = username
        status = 'success'
        break
if status == 'success':
    output = 'Welcome '+username
    context= {'data':output}
    return render(request, 'DoctorScreen.html', context)
else:
    context= {'data':'Invalid username'}
    return render(request, 'DoctorLogin.html', context)
```

5.SCREENSHOTS



Screenshot 5.1 Home Page



Screenshot 5.2 Admin Page

New Doctor/Hospital Signup Screen

Username:

Password:

Email ID:

Contact No:

Qualification:

Experience Details:

Hospital Name:

Address:

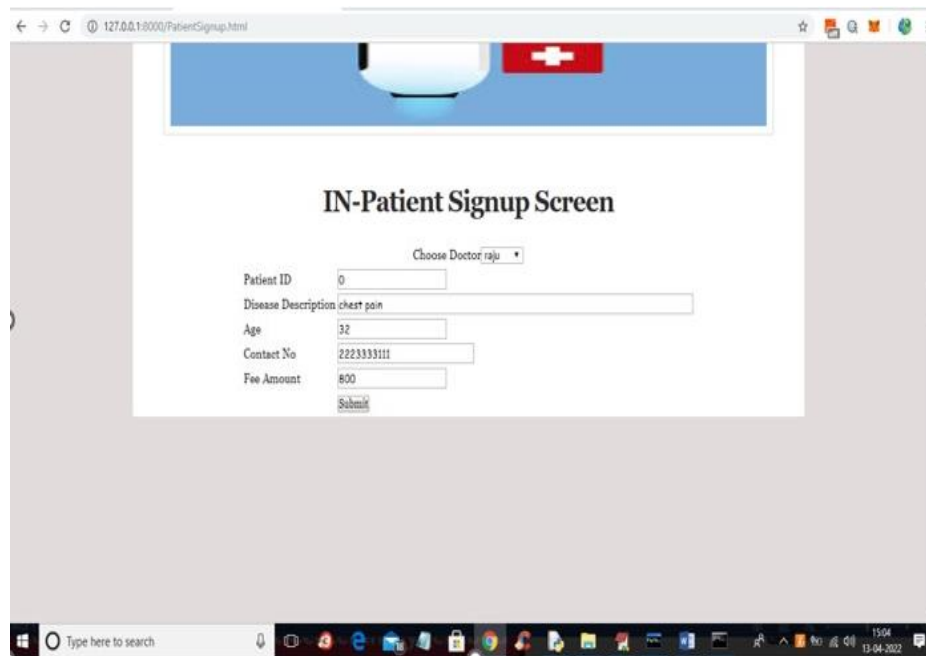
Symptoms Speciality:

Screenshot 5.3 Doctor SignUp

View Doctor/Hospital Details Screen

Doctor Name	Password	Email ID	Contact No	Qualification	Experience Details	Hospital Details	Address	Speciality
kumar	kumar	kumar@gmail.com	8885556667	MBBS	10 years in general medicines	Appollo Hospital	hyd	cold cough pains fever
raju	raju	raju@gmail.com	6667775566	MBBS	10 years experience in heart disease	Appollo Hospital	hyd	chest pain feeling conjuction shortness of breathe

Screenshot 5.4 Doctor Details



IN-Patient Signup Screen

Choose Doctor: raju

Patient ID: 0

Disease Description: chest pain

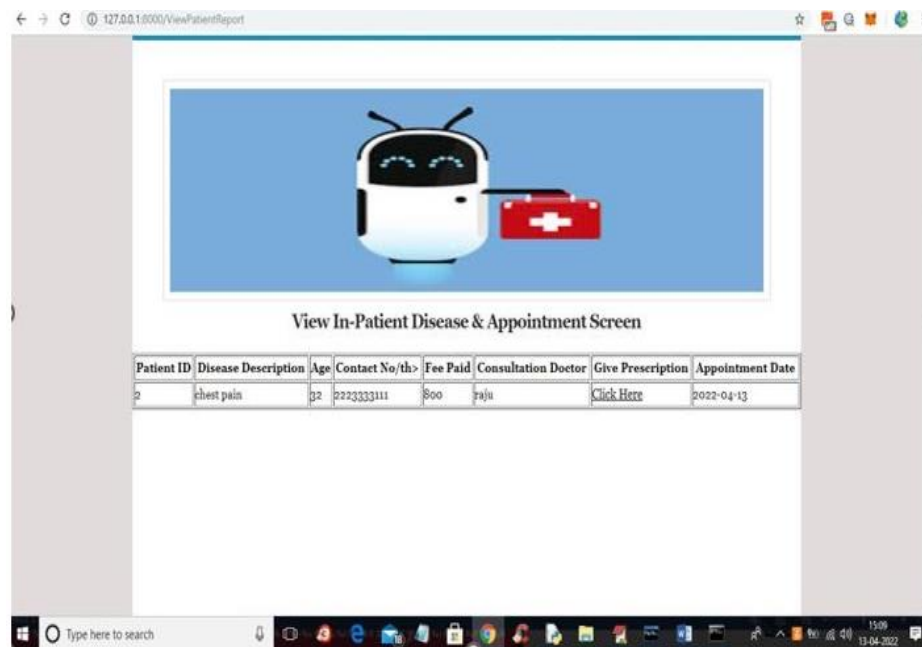
Age: 32

Contact No: 2223333111

Fee Amount: 800

Submit

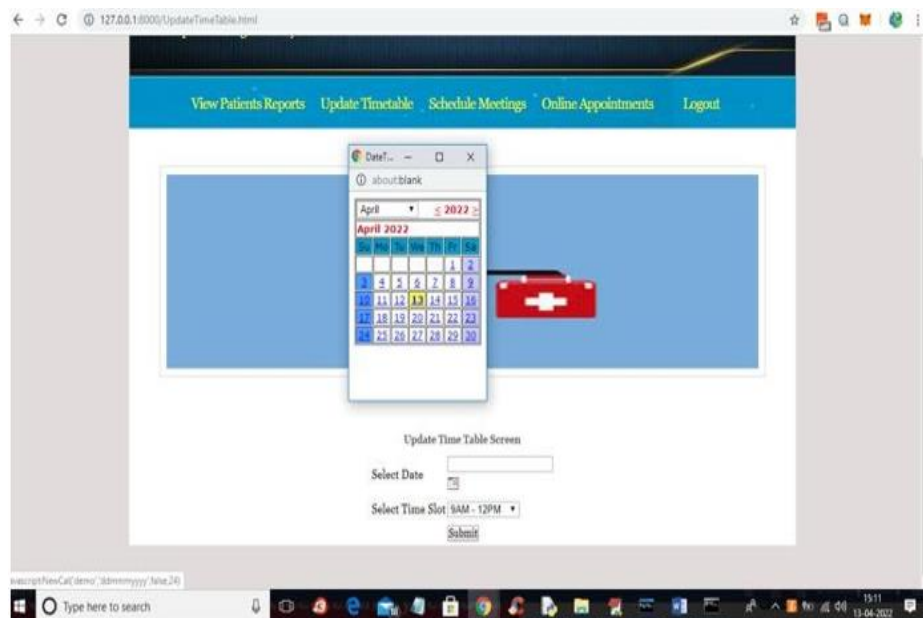
Screenshot 5.5 Patient SigmUp



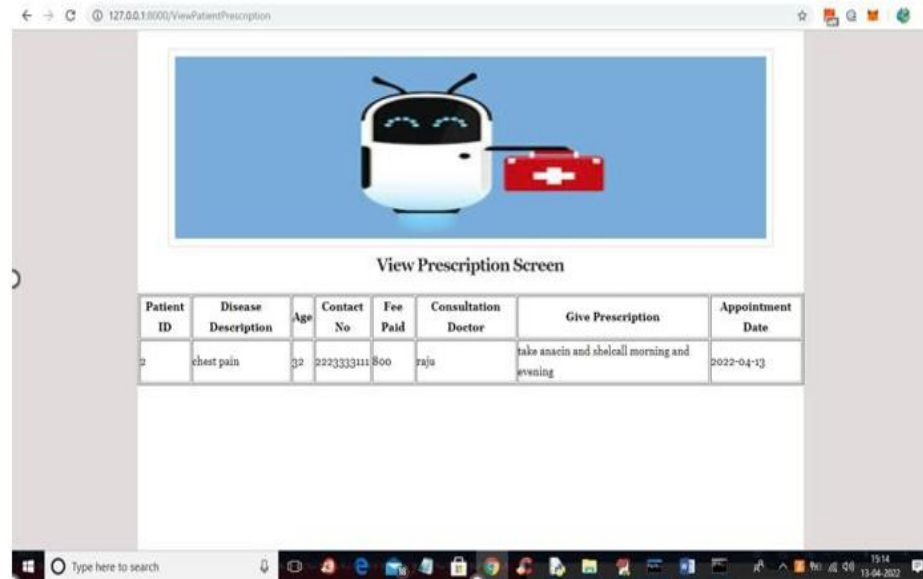
View In-Patient Disease & Appointment Screen

Patient ID	Disease Description	Age	Contact No/th>	Fee Paid	Consultation Doctor	Give Prescription	Appointment Date
2	chest pain	32	2223333111	800	raju	Click Here	2022-04-13

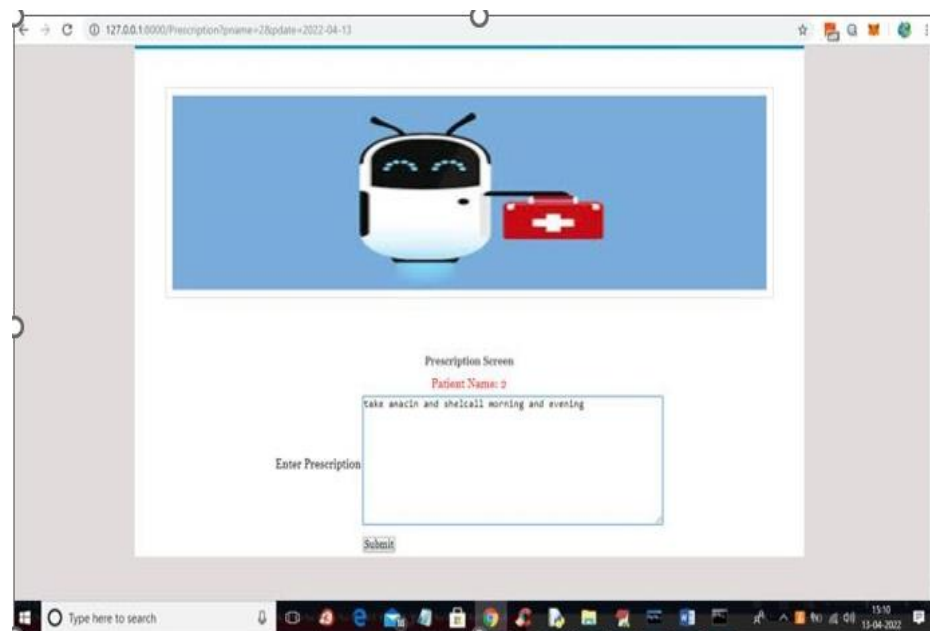
Screenshot 5.6 Patient Details



Screenshot 5.7 Timetable Fixation



Screenshot 5.8 Prescription Screen



Screenshot 5.9 Patient Interaction With Chatbot

6.TESTING

6. TESTING

6.1 INTRODUCTION TO 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.

6.2 TYPES OF TESTING

6.2.1 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.

6.2.2 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 satisfactory, 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.

6.2.3 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.

6.3 TEST CASES

6.3.1 CLASSIFICATION

S.NO	Test Case	Excepted Result	Result	Remarks(IF Fails)
1.	User Register	If User registration successfully.	Pass	If already user email <u>exist</u> then it fails.
2.	User Login	If Username and password is correct then it will getting valid page.	Pass	Un Register Users will not <u>logged in</u> .
3.	User View User	Show our dataset	Pass	If Data set Not Available fail.
4.	View Fast History Results	The Four Alarm Score Should be Displayed.	Pass	The Four Alarm Score Not Displaying fail
5.	User Prediction	Display Review with true results	Pass	Results not True Fail
6.	Show Detection process	Display Detection process	Pass	Results Not True Fail
7.	Show Eye Blink Process	Display Eye Blink Process	Pass	If Results not Displayed Fail.
8.	Admin login	Admin can login with his login credential. If success he <u>get</u> his home page	Pass	Invalid login details will not <u>allowed here</u>
9.	Admin can activate the register users	Admin can activate the register user id	Pass	If user id not found then it won't login
10.	Results	For our Four models the accuracy and F1 Score	Pass	If Accuracy <u>And</u> F1 Score Not Displayed fail

7.CONCLUSION

7.CONCLUSION & FUTURE SCOPE

7.1 PROJECT CONCLUSION

A Chatbot is a great tool for conversation. Here the application is developed to provide quality of answers in a short period of time. It removes the burden from the answer provider by directly delivering the answer to the user using an expert system. The project is developed for the user to save the user their time in consulting the doctors or experts for the healthcare solution. The application is improved with the security and effectiveness upgrades by ensuring user protection and characters and retrieving answers consequently for the questions.

7.2 FUTURE SCOPE

This project can be enhanced in few ways such that this project can be used to its fullest:

- **Telemedicine Integration:** Seamless integration with telemedicine platforms, allowing patients to initiate video consultations directly through the chatbot interface.
- **Integration with Wearables and Health Apps:** Utilizing data from wearables and health applications to provide real-time health insights and recommendations.
- **AI-aided Triage:** Using AI algorithms to assist in patient triage, helping determine the urgency of cases and suggesting appropriate actions.

8.BIBILOGRAPHY

8. BIBILOGRAPHY

8.1 REFERENCES

- [1] M. Mittal, B. Gopi, D. Singh, T. Nagarwal and P. Yadav, "Web-based chatbot for frequently asked queries (FAQ) in hospitals", Journal of Taibah University Medical Sciences, vol. 16, no. 5, pp. 740-746, 2021.
- [2] R. B. Mathew, S. Varghese, S. E. Joy and S. S. Alex, "Chatbot for disease prediction and treatment recommendation using machine learning", 3rd Int. Conference on Trends in Electronics and Informatics, pp. 851-856, 2019.
- [3] L. Athota, V. K. Shukla, N. Pandey and A. Rana, "Chatbot for healthcare system using artificial intelligence", 8th Int. Conference on Reliability Infocom Technologies and Optimization, pp. 619-622, 2020.

8.2 GITHUB LINK

<https://github.com/BhavanaVanga/hospital-management-system-using-chatbot>

