EMERGENCY CHAT BOT

REPORT

Submitted by

KARTHICK S (19IT046)

SURIYA B (19IT107)

RAAGHAVAN J S (19IT127)

in partial fulfillment for the completion of course Engineering Design Project

of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

THIAGARAJAR COLLEGE OFENGINEERING, MADURAI–15

(A Govt. Aided, Autonomous Institution, Affiliated to Anna University)

JUNE 2022

THIAGARAJAR COLLEGE OF ENGINEERING, MADURAI-15

(A Govt. Aided, Autonomous Institution, Affiliated to Anna University)

BONAFIDECERTIFICATE

Certified that this project report "**EMERGENCY CHAT BOT"** is the bonafide work of " **KARTHICK S(19IT046) , SURIYA B (19IT107) , RAAGAVAN JS (19IT127)** who carried out the project work under my supervision during the Academic Year 2021 - 2022.

SIGNATURE

Mrs.R.Parkavi,

ASSISTANT PROFESSOR,

INFORMATION TECHNOLOGY

THIAGARAJAR COLLEGE OF ENGINEERING,

MADURAI-15

TABLE OF CONTENTS

ABSTRACT

As access to various health information increases, the effectiveness and efficiency of such programs has become increasingly important. Although user interaction and workflow play a key role in the adoption of such programs, it allows users to ask the system for native text as input and provide them with relevant output such as data, help text, quotes from user manuals, filtering their queries further. ideas, becomes the next level of expertise needed in health information solutions. Chat-bots are a natural solution to this problem Chat bots easily brings 24X7 usage and service to the part that a human employee may need. This improves efficiency at a reduced cost. As argued with Petter Bae Brandtzaeg in his research people prefer chat bots because of production. Chat bots in health care offers a variety of solutions from diagnostic testing for basic health information (James and Vales, 2009). AI offers a limit to chat bot so in any other technology, because it can learn and grow over time. This saves you time and money on both patients and healthcare providers.

Thus, it creates a sense of smooth health care. However, most chat bot solutions work on fixed data or pre-defined applications that may meet certain requirements. Therefore, in this work, we propose a modular chat bot framework to re-align the health informatics solution. To demonstrate the effectiveness of the framework, we regard the “Blood Bank Management System” as a tool. In particular, the controlled blood bank control system has two modes of operation. First is uses a citizen-centered site where information about blood banks, blood stock, information, etc. is provided. The second is a set of users who use the application for daily use as blood bank employees. Therefore, we use the proposed framework to answer questions from citizens using the national public web site and users of the blood bank control system.

PROBLEM DESCRIPTION

The percentage of people who donate blood is growing day by day due to awareness about donating blood to those required. Recovered blood should be carefully controlled so that it does not adversely affect the blood recipient when they have received blood

From the observations and discussions that took place during the user needs phase, it happened found that there was no way to cooperate between HSNZ and the community to announce their blood donation system. The blood donation event program is usually advertised to the public so that you know about blood donation campaign time. At the blood bank, staff and nurses are notified only of a blood donation monthly program on the white board in the blood house. They therefore use the hand method to inform system. The problem arises when a given space is inadequate. A method used to inform employees about
The lunar system uses a white board and is written using a white board marker. So, writing is fun ambiguity. The public had no knowledge of blood donations. There are booklets distributed to donate but not to the public because they are only available at the blood donation house. Therefore, the community does not get it any information about blood donations other than going to the blood donation both.

We designed a chatbot application which is very easy to use .so in needy situation we simply give needed blood group to chatbot along with the city name and the chatbot reply with the name of the donor if he/she available.

So it greatly reduce the time and money during the needy situation . people who are unaware about the blood bank they need to search for the related blood type which is time and cost effective .it also require some third party agency and we are not sure we get the blood from the healthy donors.

To avoid these conflicts emergency chatbot came into effect which completely removes the conflicts.

BACKGROUND

A. Literature survey:

S.NO	**Paper / Article**	**Specification (Proposed Approach)**	**Remarks**
1.	Automated ChatBot Application for Blood Bank Management	The bot collects all the relevant information about the donor and acceptor. Whenever blood required for patients they need to ping the bot. Once entered in the bot, it gives information about the donor. • We use Uipath community edition to develop the bot application. We use the Google dialogflow for creating the conversation. We use chatbot.uipath.com to map the work done in Uipath Workflow and Google dialogflow.	By using of this application people who want to donate their blood can registered in this application by providing their details. When you ping the botitwill display the donors list and their details. The user can communicate with bot easily. This application can help their donor families to stay with their patients.

S.NO	**Paper / Article**	**Specification (Proposed Approach)**	**Remarks**
2.	A Hybrid Approach to Develop and Integrate Chatbot in Health Informatics Systems	we develop a chatbot that seeks free-form natural language queries by its users for blood and related services such as list of blood banks, live blood stock, blood donation camps etc. with one or more parameters as search criteria. - It answers for FAQ related blood like (why i should donate blood?) using NLP deep-learning algorithm.	The proposed system will bring the flexibility and ease in the blood search at the national level. With the use of Natural Language Processing, users can look for blood in query format rather than looking up on the web page. This saves time and effort which is vital in cases of emergencies.

S.NO	**Paper / Article**	**Specification (Proposed Approach)**	**Remarks**
3.	A self-diagnosis medical chatbot using artificial intelligence	The proposed idea is to create a medical chatbot using Artificial Intelligence that can diagnose the disease and provide basic details about the disease before consulting a doctor .To reduce the healthcare costs and improve accessibility to medical knowledge the medical chatbot is built. Certain chatbots acts as a medical reference books, which helps the patient know more about their disease and helps to improve their health.	The user can achieve the real benefit of a chatbot only when it can diagnose all kind of disease and provide necessary information. A text-to-text diagnosis bot engages patients in conversation about their medical issues and provides a personalized diagnosis based on their symptoms. Hence, people will have an idea about their health and have the right protection.

S.NO	**Paper / Article**	**Specification (Proposed Approach)**	**Remarks**
4	**HealthCare Medical ChatBot**	Users do not seem to remember all the medications either symptoms related to the actual disease. a little the lower user should go to the hospital in person a long-term test. In addition to manage the telephone needs of the Complaints Unit balancing is a form of turmoil. Such an encounter is also possible can be solved by introducing a medical chatbot by giving proper guidance regarding healthy living.	The user can get the clear medical advices during unwell period. It also predict the disease of the patients based on the input(symptoms) They are given.

B . IPR search:

S.NO	**Title of paper**	**Author**	**Methodology used**
1	A smart pervasive chatbot for Emergency case assistant based on cloud computing	Nourchène OUERHANI	**Image tagging:** Image tagging or automatic image annotation is the process of adding keywords to a digital image depending on the content of the information provided. Its goal is to anticipate multiple text labels that describe a particular invisible image. **Chatbot:** Chatbots and Virtual Assistants regroup multiple AI fields. This is where the bot understands a person's voice or text, extracts valuable knowledge from it, is given that knowledge base, performs a specific action, and feeds back in a human-understandable format (voice or text). **Speech to text:** The STT system posts a text stream from a voice input [34]. STT conversion begins with a process called automatic speech recognition (ASR). This enables speech recognition (LVCSR) of large vocabulary that is speaker-independent. **Natural language processing(NLP):**

			Natural Language Processing (NLP) is a field of machine learning that enables computers to understand using human language. Natural language is often ambiguous and violates possible formal explanations.
2	AI chatbot design during an epidemic like the novel coronavirus	Gopi batteni, Nalini chintalapudi, Francesco Amenta	**Chatbot:** Chatbots and Virtual Assistants regroup multiple AI fields. This is where the bot understands a person's voice or text, extracts valuable knowledge from it, is given that knowledge base, performs a specific action, and feeds back in a human-understandable format (voice or text). **Artificial intelligent markup language(AIML):** Designed bots can use artificial intelligence markup language (AIML) to process user requests and identify message patterns. AIML is an XML-based markup dialect for creating natural language software agents that provides users with a real human interactive experience. Depending on the user's response, AIML logic gets the symptomatic keywords and evaluates the user's existing medical condition. Ultimately, you need to make it feel like you're talking to a healthcare professional.

C . Scope of the problem / Objective:

- People are struggling to get blood donors within specified range during emergency time.
- Time Complexity-During emergency time people are roaming here and there for the blood. It increase the time and cost also increase,so using this chatbot we can reduce both time and cost.
- Some people don’t have awarness of blood bank nearby.so we can search the blood bank available nearby the his/her location and active donors with the help of chatbot application.
- Since searching for blood bank requires man power .so inorder to avoid such things we can use the chatbot.
- Unstable for user experience

D . Constraints / Limitations:

- The user may enter the wrong details during the registration.
- After the donating the blood the user may forgot the change his profile to inactive. It leads to problems such as the user may get many requests to donate blood.
- User may give incompatible input which is very difficult for processing.
- The person with disease also can enter his/her details.
- Here we are using python flask which need to handle multiple requests from the user.

DESIGN REQUIREMENTS

SCHEDULE – TIMELINE CHART :

Sl.No.	Milestones to be achieved	Estimated date	Remarks
1	Develop project ideas	8-04-2022	All ideas related to chat bot are identified and choosed the best among them.
2	Analyse requirements	15-04-2022	The requirements related to developed idea and the chat bot sources , also the necessary dependencies are analysed.
4	Prototype Development	22-04-2022	The prototype developed by sketching the different modules that are needed for blood bank chatbot
5	Module Development	30-04-2022	The modules are developed with the open sourse platform chat bot.
6	Coding Part	6-05-2022	The coding part has been done with suitable IDE .
7	Integration of modules	22-05-2022	The modules are integrated successfully in a suitable platform.
8	Testing Application	30-05-2022	The application is tested under unit and performance testing.
9	Deployment	9-06-2022	The code is ready to deploy to the stakeholders.

BUDGET:

- Since we are using free twilio account it has limited access only . So here only few user can accommodate at a time.
- The premium account has wide range of accessibility.
- It provides continuous access to user.
- So user can use it without any interrupt.
- MongoDB (Cloud Database) will permit a limited amount of insertion and updation in the database . To scale up to extend, we need to access its premium features.
- To make availability for 24x7 service, we need to host in a cloud platform so we in need of it.
- To get hosted in whats-app and verified by whats-app , we need to pay to whats-app platform .

RISK FACTORS :

- The Flask Application needs to handle multiple requests at a time.
- There may be a database ambiguity when the user inserts or deletes the record in the database.
- The server may crash due to technical failure.
- There may be a chance of accessing other details in the chatbot by using mobile numbers.
- If the third party platform (twilio) fails , then the entire system won't work. Because , it is a intermediate that communicates with whatsapp and the flask application (chat bot)

TEAM MEMBERS – ROLES AND RESPONSIBILITIES

S.NO	**Team Members**	**Roles**	**Responsibility**
1	Karthick S	Project Manager	Lead the team , Monitor all the necessary activities within the team , updating work and splitting the work to peer mates and combining all the modules together.
2	Suriya B	Project Archivist	Responsible for preparation of documents and updating and hosting the meetings. Also responsible for answering to the enquiries asked by the stakeholders.
3	Raghavan J S	Project Liaison Officer .	Responsible for arranging the meeting with the stakeholders. Responsible for acting as a liaison between multiple team members and stakeholders.

PROPOSED METODOLOGY

Data collection:

- Data is collected from the various users. It is stored and process by various techniques. The data's are stored in mangodb. We can retrieve and perform various operations on the data.

- The data is fed into the chatbot and based on the queries the result according to the queries is displayed.

Proposed Solution:

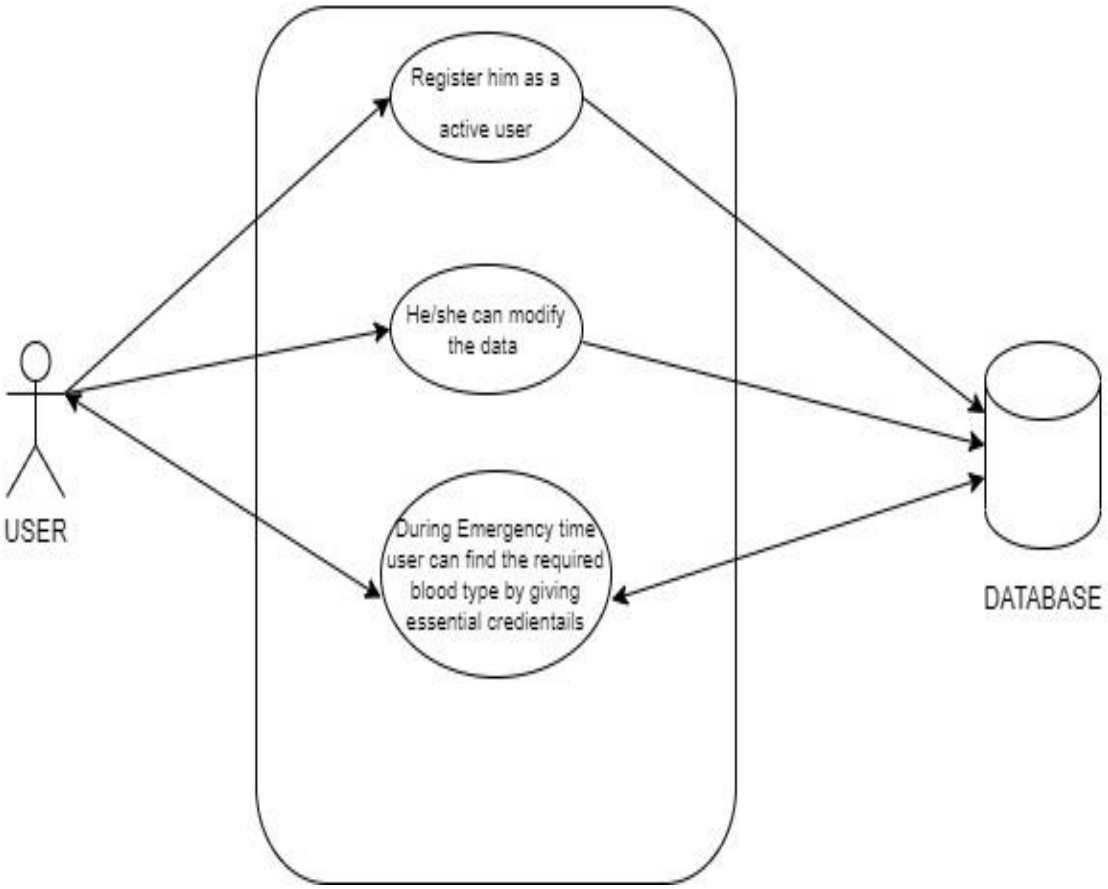
- Creation of a Chat Bot that the user can volunteer himself as a blood donor.
- The user can modify his/her details before and after donating the blood.
- The user can fetch the details of available blood donor by querying the city name.
- The user can get the details of the donor along with his mobile number and city.

Setting up the Flask and Ngrok:

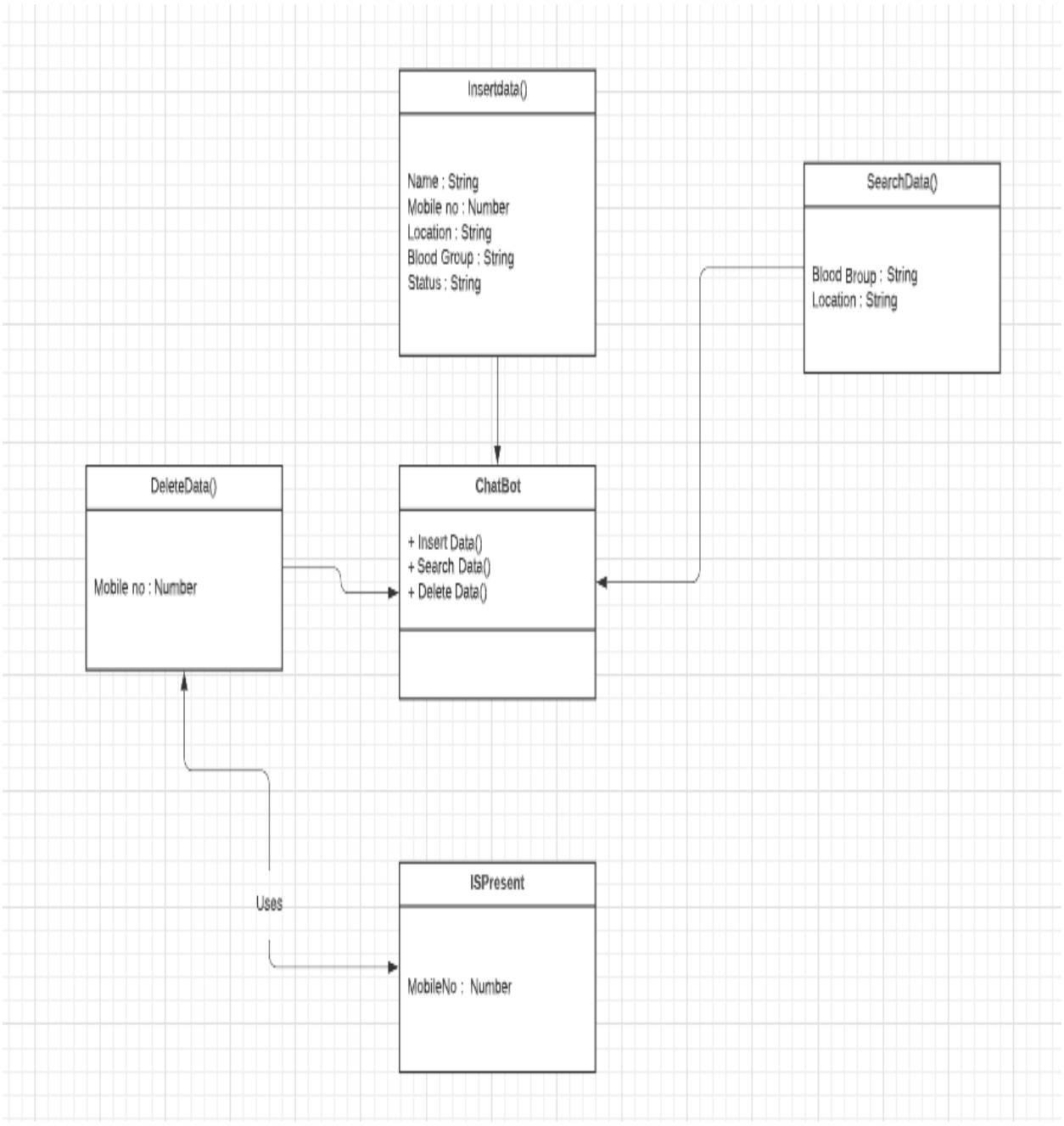
- Python flask is used to create a web application.
- The necessary libraries are imported such as
    1. Pymongo – Cloud database
    2. Requests –get the request from the user
    3. Twilio – connects with twilio API
- Ngrok is used to localhost url to public url.
- The url is pasted in the Twilio API to set up the connection between the flask and whatsapp.

DESIGN DIAGRAMS

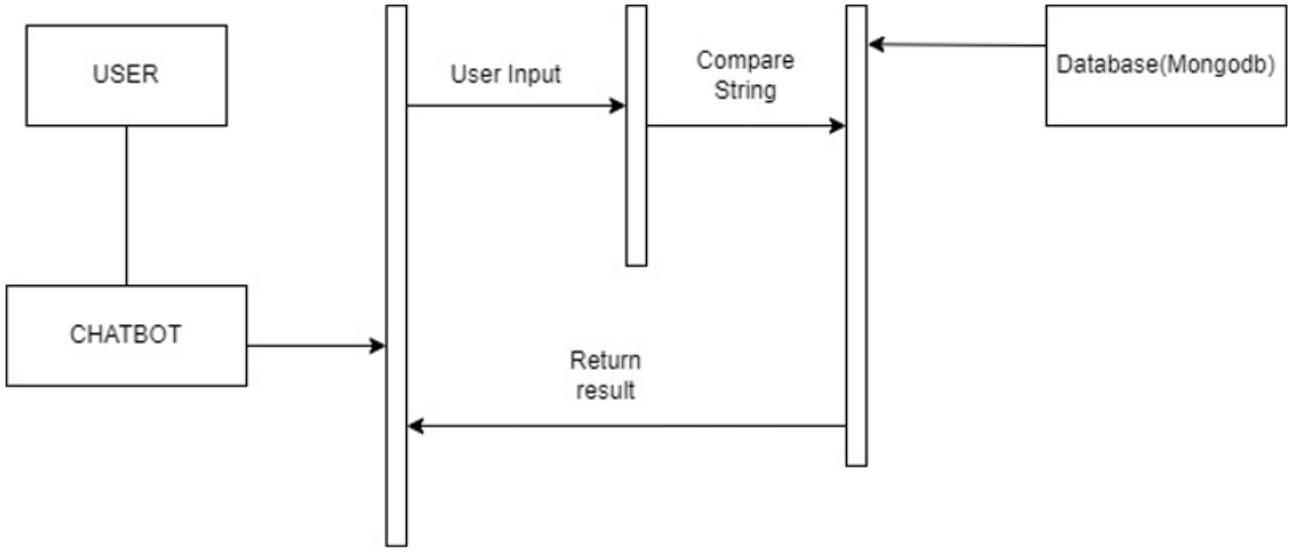
a. Use case diagrams:



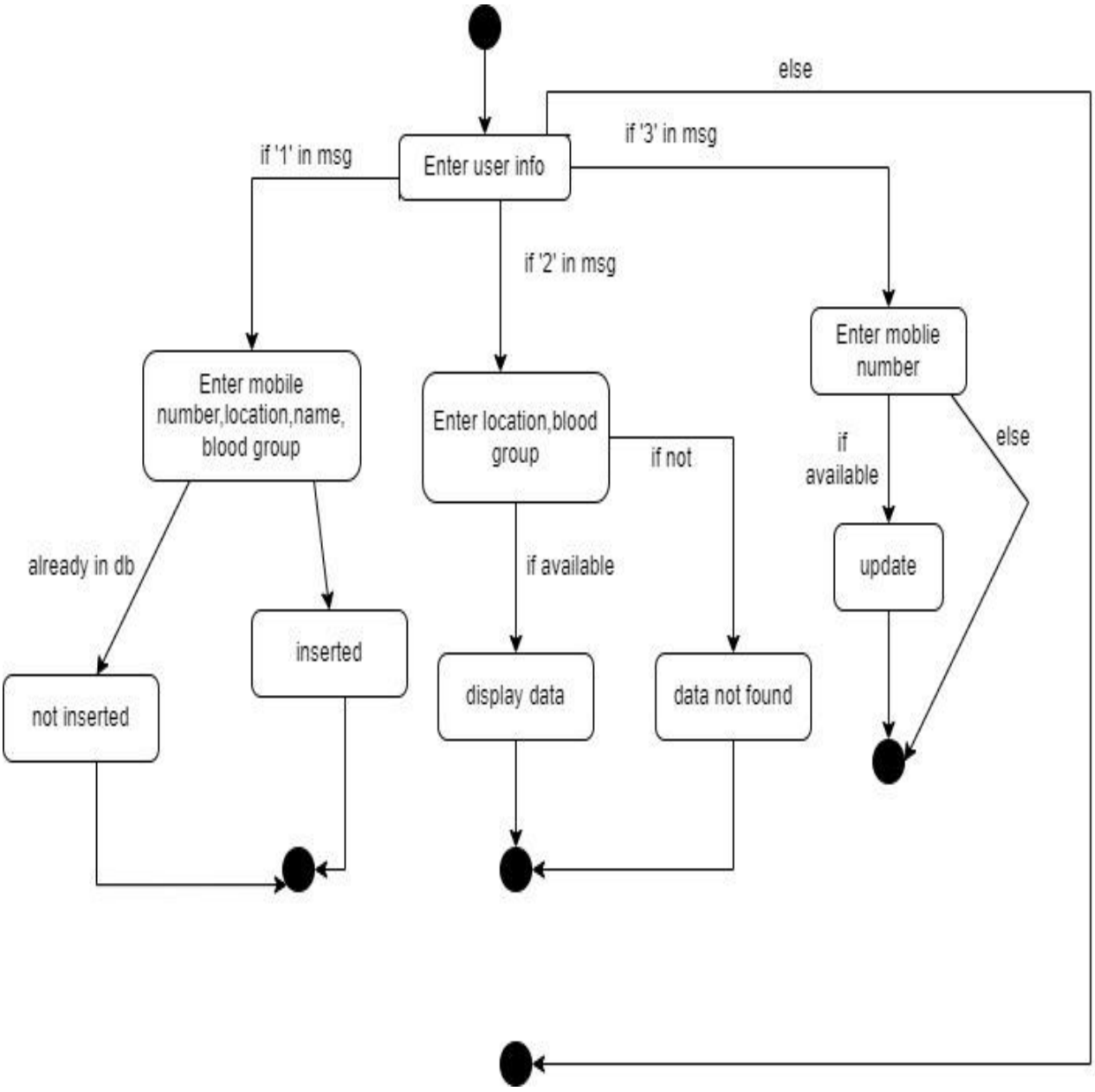
b. Class diagram:



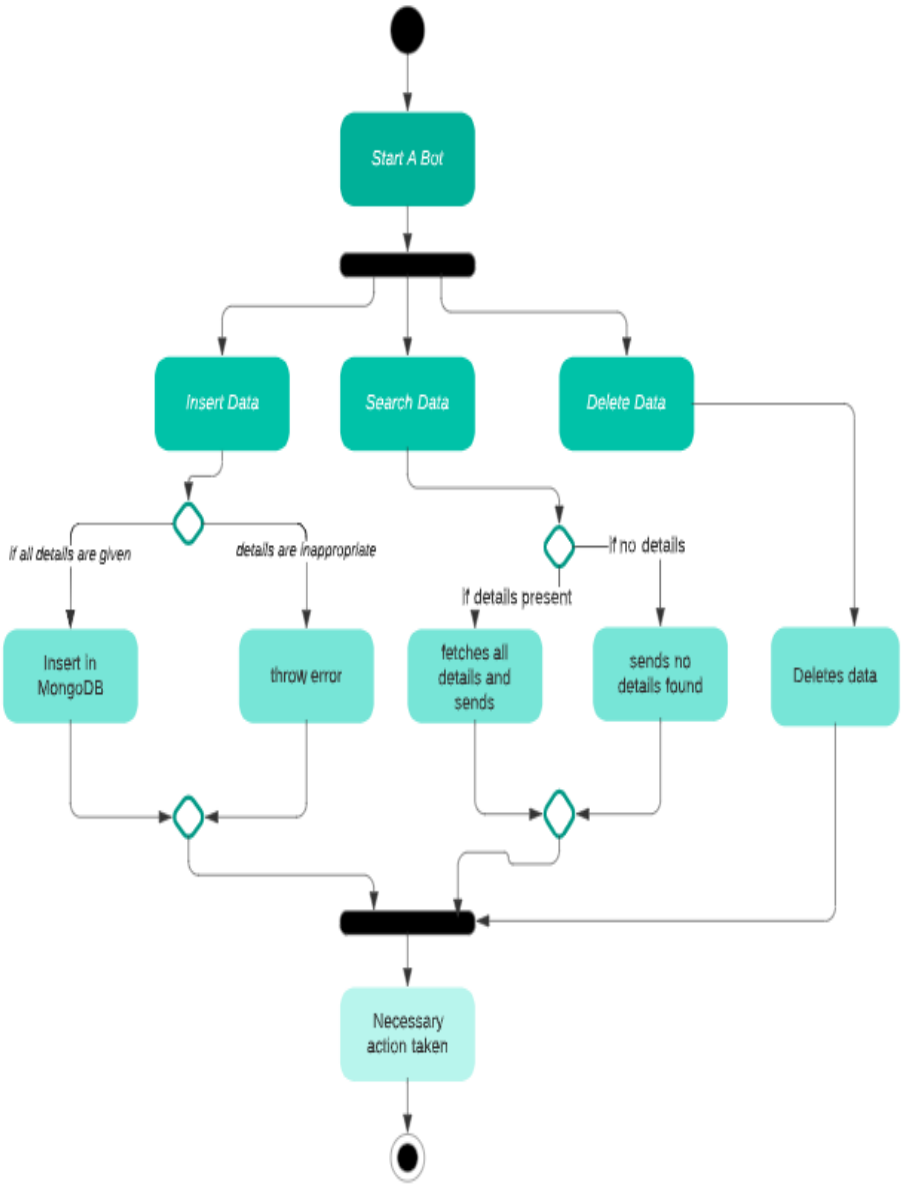
c.Sequence diagram:



d.State chart diagram:



e. Activity diagram:



DESIGN VERIFICATION MATRIX

Group	Inspection rules (criteria)	Completed / Satisfied		
		Low	Medium	High
Structure compliance	1. do the functional requirements defined?		YES	
	2. do the user interfaces and their features defined?			YES
Contents compliance	3. Is the user interface flow clear and stated properly?		YES	
	4. Do decision states, elections/ queries and calculations clearly defined?		YES	
	5. Do all requirements classified and enumerated?			YES
	6. Is the database design compatible with the database guide?			YES
	7. Do all user groups and their behaviors clearly defined?			YES
	8. Do all information security requirements clearly defined?	YES		
	9. DO all hardware requirements clearly defined?	YES		
	10. Do information/error messages, and in which case they are shown, defined?			YES
	11. Do there any requirement which is out of the scope?		YES	
	12. Do integration with other systems defined?		YES	
	13. Do there any conflicting requirements?	YES		
	14. Do error messages clearly indicate what action the user needs to take to correct the error?	YES		
	15. Are assumptions and limitations stated?		YES	
	16. Are the requirements clearly understood?			YES
	17. Can the requirements be tested? (Dependent on the test/hardware tools, the test methods, the test resources, the scalability and the observability.)			YES
	18. Are the requirements traceability satisfied via TFS?		YES	

BUSINESS ASPECTS

1.This type of Blood bank chatbot helps some **private hospitals** to use the volunteer details to get blood in difficult situation.

2. The data will be helpful in future aspects and can be accessed via API's using the number.

3.The data also helpful for other people to get and use the data for future scope details.

FINAL IMPLEMENTATION

FLASK-CODE :

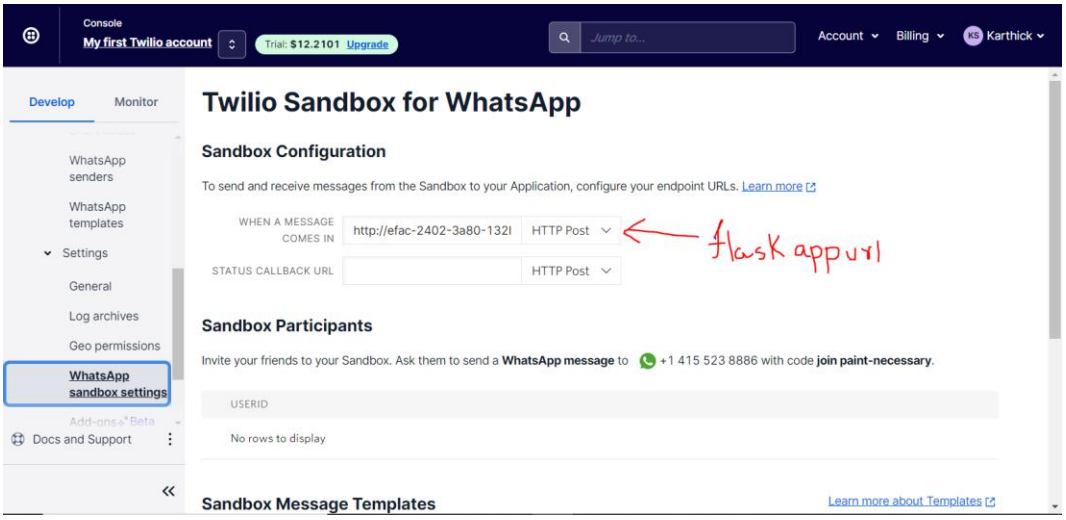
```
from turtle import Turtle
from flask import Flask, request
import requests
from twilio.twiml.messaging_response import MessagingResponse

app = Flask(_name_)


@app.route('/bot', methods=['POST'])

def bot():

    def getdatabase():
        from pymongo import MongoClient
        import pymongo
        connection_string =
"mongodb+srv://karthi:karthi1412@cluster0.grst1.mongodb.net/bloodvolunteers?r
etryWrites=true&w=majority"
        client = MongoClient(connection_string)
        return client['bloodvolunteers']

    def insert(details):
        dbname = getdatabase()
```

```
        collection_name = dbname["volunteer_details"]      # the collection name is
volunteer_details
        insertdata={
            "Name":details[1],
            "Mobile Number":details[2],
            "Location":details[3],
            "Blood Group":details[4],
            "Status":"Active"
        }
        collection_name.insert_one(insertdata)

    def searchdetails(det):
        dbname = getdatabase()
        collection_name = dbname["volunteer_details"]
        bloodgroup = det[1]
        location = det[2]
        item_details = collection_name.find()
        ans=""
        for item in item_details:
            if(item['Blood Group'].lower()==bloodgroup and
item['Location'].lower()==location and item['Status']=="Active"):
                #print("Person name = ",item['Name']," Mobile Number : ",item['Mobile
Number'])
                dett = "\n"+"Person name = "+item['Name']+"\n"+" Mobile Number :
"+item['Mobile Number']
                ans = ans+dett+"\n"
        return ans

    def delete_details(details):
        dbname = getdatabase()
        collection_name = dbname["volunteer_details"]      # the collection name is
volunteer_details
        myquery = { "Mobile Number": details[1] }
        collection_name.delete_one(myquery)

    def checkthenumberispresentornot(n):
        dbname = getdatabase()
        collection_name = dbname["volunteer_details"]
        item_details = collection_name.find()
        for item in item_details:
```

```
            if (item["Mobile Number"]==n):
                return True
        return False

    resp = MessagingResponse()
    msg = resp.message()

    responded = False
    incoming_msg = request.values.get('Body', '').lower()
    # 1->make volunteer 2->searching data
    if '1' in incoming_msg:         # he will provide 1,karthick,7418386415,theni,B+
        s = incoming_msg.split(",")
        if(len(s)==5):
            msg.body("Inserting details....\n")
            insert(s)
            msg.body("\nCongrats.. You are now a volunteer to donate blood .")
            responded = True
        else:
            msg.body("Error. for your reference please see below
example\n1,karthi,9999999999,theni,b+\n1,suriya,7777777777,madurai,B+")
            responded=True
    if '2' in incoming_msg:
        s = incoming_msg.split(",")      #he will give like 2,b+,theni
        if(len(s)==3):
            details_fetched = searchdetails(s)
            if(details_fetched!=""):
                msg.body(details_fetched)
                responded=True
            else:
                msg.body("Sorry No details Found.. Try with nearby location.")
                responded=True
        else:
            msg.body("Invalid input. for your reference please see below
example\n2,b+,madurai\n2,o+,theni")
            responded=True
    if '3' in incoming_msg:     #3,mobilenumber
        s=incoming_msg.split(",")
        if(len(s)==2):
            if(checkthenumberispresentornot(s[1])):
                #delete the record
```

```
            delete_details(s)
            msg.body("Deleted Sucessfully . Hope you will soon be a blood
volunteer. Take care !")
        else:
            msg.body("The number is not is in volunteer list . Cant able to delete.")
        responded = True
    else:
        msg.body("Invalid input. for your reference please see below
example\n3,9999999999\n3,7878786545")
        responded=True
  if 'menu' in incoming_msg:
    torespond="Hi Dear.The mode available are \n1-insert 2-search 3-delete
details ..\nExamples given below\nFor insert :
1,[yourname],[mobilenumber],[location],[bloodgroup]\nFor search :
2,[bloodgroup],[location]\nFor delete : 3,[mobilenumber]"
    msg.body(torespond)
    responded=True
  if not responded:
    msg.body("Please type menu to initialise chatbot. ")
  return str(resp)


if _name_ == '_main_':
  app.run()
```

TWILIO SETUP :



Flask app url can be generated using Ngrok (cloud flask app deployers are cost)

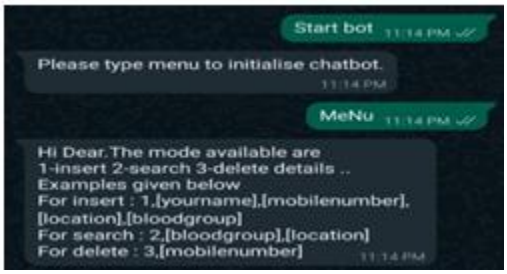
TESTING AND VALIDATION

a).Unit testing:

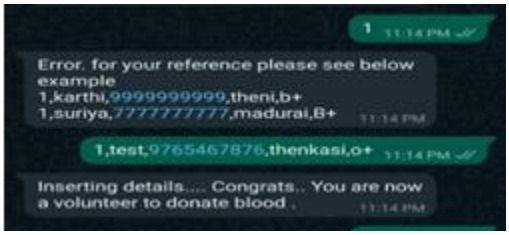
Unit testing is performed by the respective developers on the individual units of source code assigned areas. The developers use test data that is different from the test data of the quality assurance team.

The goal of unit testing is to isolate each part of the program and show that individual parts are correct in terms of requirements and functionality.

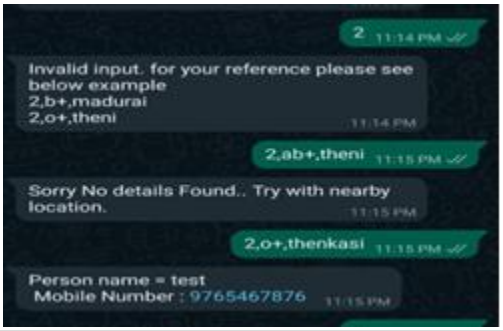
MENU :



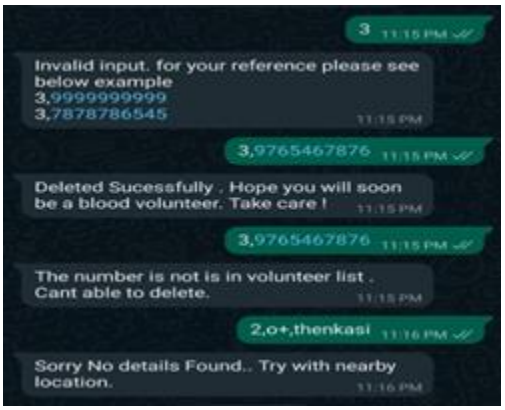
INSERTION :



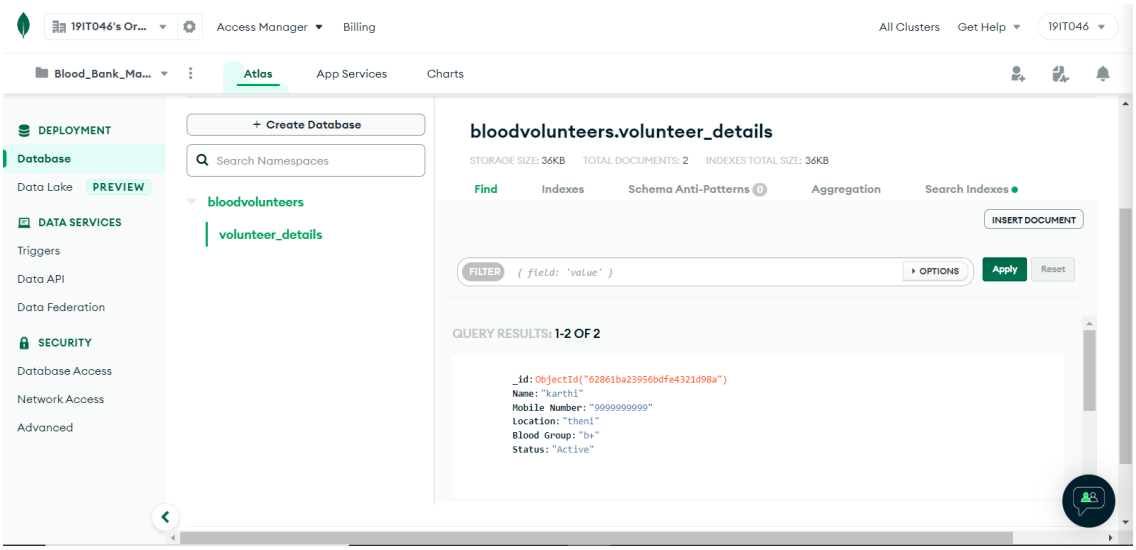
SEARCH :



DELETE :



CLOUD DATABASE (MongoDB)

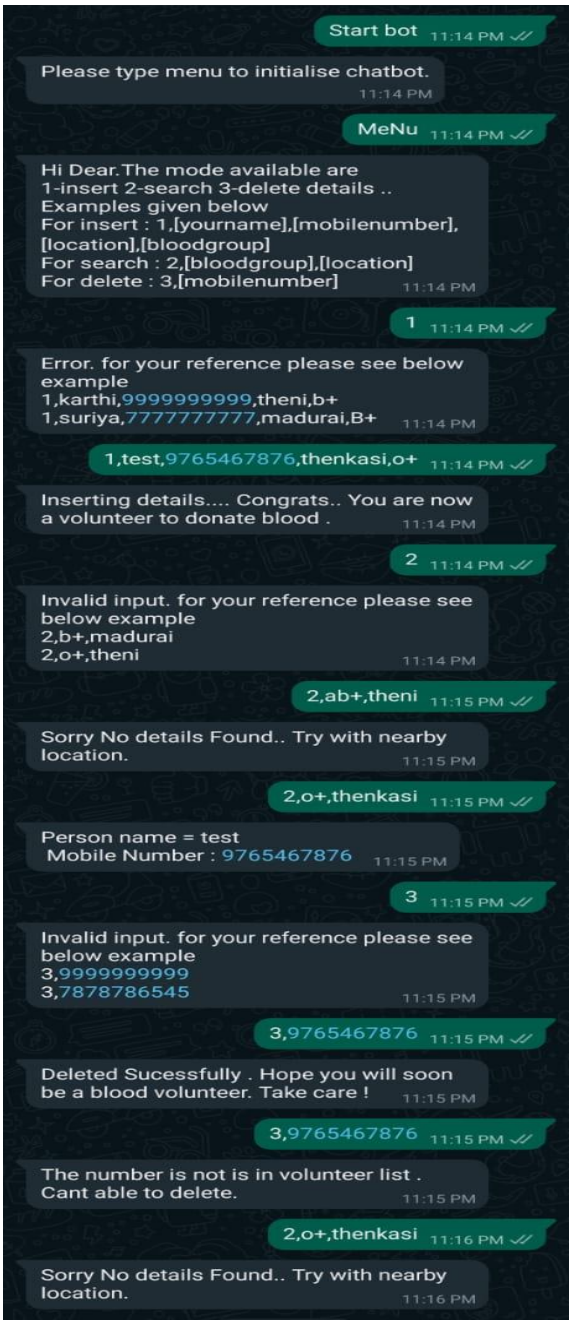


b)Integration testing:

Integration testing is defined as the testing of combined parts of an application to determine if they function correctly. Integration testing can be done in two ways: Bottom-up integration testing and Top-down integration testing.

Bottom-up integration - This testing begins with unit testing, followed by tests of progressively higher-level combinations of units called modules or builds.

Top-down integration - In this testing, the highest-level modules are tested first and progressively, lower-level modules are tested thereafter.



DELIVERABLES

Deliverables are the services that are to be provided upon the completion of a project.

1.This type of bot delivers people to get blood within the range in hospitals.

2.People can get the blood donors with mobile numbers , so that they can contact directly to the people without any intermediate person.

3.The blood bank also can use this data to get the required blood in a fraction of second.

4.The bot responses without any manpower also it is available for 24x7 service so , any time people won't need to depend on anyone else.

5. Data is collected from the various users. It is stored and process by various techniques. The data's are stored in mangodb. We can retrieve and perform various operations on the data.

6. We can use it from anywhere , since whatsapp is globally available , so chance of interruption is very less

7. It greatly reduce the time and cost of the user.

8. People with zero knowledge can use it.

9. Since whatsapp is highly available so we can use it anytime.

PERFORMANCE TEST

ATTRIBUTES	**REPORT**	**SUGGESTIONS**
Response time	Estimated performance 1-sec Performance achieved 1-sec Estimated performance achieved	Use HighSpeed Internet connection or any serverless database.
Data availability	High Estimated performance is achieved	-
Data accessibility	Data can be easily accessible from anywhere. Estimated performance is achieved	-
Scalability	Scalable up to the capacity of the database.	Using cloud service such as aws etc can be used.
Data transaction processing	Estimated performance is achieved.	-
Portability	Runs on android smart phones with lower to higher version. Estimated performance is achieved.	Can be made portable with IOS.
Compatibility	Compatible with any android version. Estimated performance is achieved.	-

FINANCIAL CONSIDERATIONS

This system is a completely software-oriented project and requires minimal attention to external physical financial costs. The product cost is considered to be:

1. Computational power/Energy (Includes Twilio premium account)

2.Storage

3.Memory

Category	Cost(INR)
Computational power/Energy	15000-25000
Memory(RAM)	8000-10000
Storage	5000-10000
TOTAL	28000-45000

DEPLOYMENT

1.Creation of a Chat Bot that the user can volunteer himself as a blood donor.

2. Can modify his own details after donating blood.

3.User can get the details providing blood group along with city name.

4.First , we type 1 it will insert the your data example name , address, phone number , city , address etc..

5.Second, suppose we mistake in data can we modify the data . we want to type It show modify data.. and enter the your correct phone number . Chat bot check your phone number. and phone number is correct allow to modify the data in you done mistake any address etc..

6. Third , delete suppose you want to delete name in the your data set.. you want to type 3 .it allow to delete the your you want to delete example address,name,phone number etc…