### **A SMART CHATBOT ARCHITECTURE BASED ON NLP AND MACHINE LEARNING FOR HEALTH CARE ASSISTANCE**

A project report submitted in the partial fulfillment of the requirements for the award of degree of

**BACHELOR OF TECHNOLOGY**

**in**

**COMPUTER SCIENCE AND ENGINEERING**

**by**

**KALLAM SAHITHI 18K11A0528**

**BARATAM VENU MADHAV 18K11A0507**

**KATTAMURI SRITHIJA 18K11A0535**

**KANDREGULA RAM TARUN 18K15A0529**

**under the guidance of**

#### **Mrs. D. HIMA BINDU**

**Assistant Professor**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

A picture containing text, clipart

Description automatically generated

**PYDAH COLLEGE OF ENGINEERING AND TECHNOLOGY**

**(Affiliated to JNTU, KAKINADA) (Recognized by A.I.C.T.E, NEW DELHI)**

**GAMBHEERAM, VISAKHAPATNAM-531163**

**(2018-2022)**

**PYDAH COLLEGE OF ENGINEERING AND TECHNOLOGY**

**(Affiliated to Jawaharlal Nehru Technological University, Kakinada, A.P)**

**GAMBHEERAM, VISHAKAPATNAM-531163**

Logo

Description automatically generated with medium confidence

**CERTIFICATE**

### This is to certify that the project work entitled **“A SMART CHATBOT ARCHITECTURE BASED ON NLP AND MACHINE LEARNING FOR HEALTH CARE ASSISTANCE”** has been carried out by the following students **KALLAM SAHITHI (18K11A0528), BARATAM VENU MADHAV (18K11A0507), KATTAMURI SRITHIJA (18K11A0535),KANDREGULA RAM TARUN (18K11A0529)** and submitted in the partial fulfillment of the requirements for the award of Bachelor of Technology in **COMPUTER SCIENCE AND ENGINEERING** during the year 2021-2022. The results embodied in this project report were not submitted in any other universities for the award of any degree or diploma.

PROJECT GUIDE HEAD OF THE DEPARTMENT

##### **Mrs.D. HIMA BINDU Mr. V. SRIKANTH**

Assistant Professor, Head of the department,

Department of CSE. Department of CSE.

##### 

**EXTERNAL EXAMINER**

**ACKNOWLEDGMENTS**

We express our sincere gratitude to our project guide **MRS. D. HIMA BINDU**, Assistant Professor of the Computer Science and Engineering Department at Pydah College of Engineering and Technology, for her valuable support and inspiration and constant encouragement throughout this dissertation.

We are grateful to **MR. V. SRIKANTH,** Head of the Department, Computer Science and Engineering for encouraging us to take up an innovative project and for the constant source of inspiration.

We express our gratitude and thanks to Principal **DR. M. ASHOK KUMAR**, Pydah College of Engineering and Technology, for his encouragement and inspiration at various points of time in the successful accomplishment of this project.

Our sincere thanks to both teaching and non-teaching, our parents, and our friends for giving out valuable suggestions throughout the project work.

**DECLARATION**

We hereby declare that the project entitled “**A SMART CHATBOT ARCHITECTURE BASED ON NLP AND MACHINE LEARNING FOR HEALTH CARE ASSISTANCE**” is original work by us for the award of the degree of **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE & ENGINEERING** from **JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY** **KAKINADA** and no part of it is earlier submitted to this or any other university.

**KALLAM SAHITHI (18K11A0528)**

**BARATAM VENU MADHAV (18K11A0507)**

**KATTAMURI SRITHIJA (18K11A0535)**

**KANDREGULA RAM TARUN (18K11A0529)**

## **ABSTRACT**

Artificial Intelligence has core branches like, Machine Learning which takes in data, searches patterns, improves itself using the data, and displays the outcome. To lead healthy lifestyle healthcare is very much important. In few unsocialized areas, it is quite hard to find a consultation with a doctor that easily regarding health issues. The main idea here is to make a healthcare chatbot based on Artificial Intelligence using NLP that can diagnose the disease and provide required details about the specific disease before consulting or visiting a doctor. Reduces the healthcare costs and improves accessibility to this medical chatbot. Specific chatbots act as virtual medical assistance, 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 kinds of diseases and provide the necessary information. A text-to-text and voice to voice medical chatbot involves patients in online conversation considering their health problems which provides a set of personalized diagnoses based on their provided symptoms. These bots connect with the potential patients visiting the site, helping them discover specialists and getting them access to correct treatment. This chatbot uses Natural language processing techniques to process and analyze the data and give the output inappropriate manner. It brings up the disease-related problems about whether the task mentioned above should be assigned to human staff. This healthcare chatbot system will provide patients healthcare support online at all times. It helps to generate health data and automatically delivers the information of reports to medical management. By asking the questions in series it helps the patient by guiding what exactly the user is looking for queries.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **LIST OF FIGURES** |  |
| **S.NO** | **FIG.NO** | **NAME OF THE FIGURES** | **PAGE.NO** |
| 1 | 3.1 | ALGORITHM | 12 |
| 2 | 6.1 | SYSTEM ARCHITECTURE | 39 |
| 3 | 6.2 | USECASE DIAGRAM | 40 |
| 4 | 6.3 | CLASS DIAGRAM | 41 |
| 5 | 6.4 | OBJECT DIAGRAM | 42 |
| 6 | 6.5 | SEQUENCE DIAGRAM | 43 |
| 7 | 6.6 | COLLABORATION DIAGRAM | 44 |
| 8 | 6.7 | STATE CHART DIAGRAM | 45 |
| 9 | 6.8 | ACTIVITY DIAGRAM | 46 |
| 10 | 6.9 | COMPONENT DIAGRAM | 47 |
| 11 | 6.10 | DEPLOYMENT DIAGRAM | 48 |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **LIST OF TABLES** |  |
| **S.NO** | **TABLE.NO** | **NAME OF THE TABLE** | **PAGE.NO** |
| 1 | 8.1 | TEST CASE | 67 |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **LIST OF SCREENS** |  |
| **S.NO** | **FIG.NO** | **NAME OF THE SCREENS** | **PAGE.NO** |
| 1 | 9.1.1 | ADMIN LOGIN | 69 |
| 2 | 9.1.2 | ADMIN HOMEPAGE | 69 |
| 3 | 9.1.3 | DATASET UPLOAD | 70 |
| 4 | 9.1.4 | USER INPUT SCREEN | 71 |

**TABLE OF CONTENTS**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **TITLE** | | | **PAGE NO** | | | | |
| **ABSTRACT** | | | **5** | | | | |
| **LIST OF FIGURES** | | | **6** | | | | |
| **LIST OF TABLES** | | | **7** | | | | |
| **LIST OF SCREENS** | | | **8** | | | | |
| **1.INTRODUCTION** | | | **13** | | | | |
| 1.1 Project Purpose | | | 15 | | | | |
| 1.2 Project Scope | | | 16 | | | | |
| 1.3 Project Overview | | | 16 | | | | |
| 1.4 Features | | | 16 | | | | |
| **2.LITERATURE SURVEY** | | | **17-18** | | | | |
| 2.1 Related work | | | 18-19 | | | | |
| **3.PROBLEM ANALYSIS** | | | **21** | | | | |
| 3.1 Existing System | | | 22 | | | | |
| 3.2 Proposed System | | | 23 | | | | |
| 3.3 Modules | | | 25 | | | | |
| 3.3.1 Admin  3.3.2 User | | | 25  25 | | | | |
| 3.4 Algorithm | | | 26 | | | | |
| 3.4.1 NLP | | | 26 | | | | |
| 3.4.2 Proter Stemmer algorithm | | | 26 | | | | |
| **4.SYSTEM SPECIFICATIONS** | | | **29** | | | | |
| 4.1 System Requirements | | | 30 | | | | |
| 4.2 Functional Requirements | | | | 31 | |
| 4.3 Non-Functional Requirements | | | | 32-33 | |
| **5.TECHNOLOGY DESCRIPTION** | | | | **34** | |
| 5.1 Software Environment | | | | 35 | |
| 5.1.1 Python Technology | | | | 36-44 | |
| **6.SYSTEM DESIGN** | | | |
| 6.1 Introduction to System Design | | | | 46-47 | |
| 6.1.1 System Architecture | | | | 48 | |
| 6.1.2 Unified Modelling Language | | | | 48 | |
| 6.1.2.1 Use Case Diagram | | | | 49 | |
| 6.1.2.2 Class Diagram | | | | 50 | |
| 6.1.2.3 Object Diagram | | | | 51 | |
| 6.1.2.4 Sequence Diagram | | | | 52 | |
| 6.1.2.5 Collaboration Diagram | | | | 53 | |
| 6.1.2.6 State Chart Diagram  6.1.2.7 Activity Diagram.  6.1.2.8 Component Diagram  6.1.2.9 Deployment Diagram  **7.SAMPLE CODE**  **8.TESTING**  8.1 Testing Methods  8.2 Testing Levels  8.2.1 Unit Testing  8.2.2 Integration Testing  8.2.3 User Acceptance Testing  8.3 Test Cases  **9.SCREENS**  9.1 Screen Shots  **10.CONCLUSION**  **11.BIBLIOGRAPHY**   |  |  | | --- | --- | |  | 78 73 | | | | | 52  53  54  55  **56-65**  **66**  67  67  67  70  70  71  **72**  73-75  **76-77**  **78-81** | |
|  | | | |  | |
|  | | | |  | |
|  | |
|  | |
|  | |
|  | |
|  | | | |  | |
|  | | | |  | |

|  |  |
| --- | --- |
|  |  |
|  |  |

## **Smart Chatbot System**

**INTRODUCTION**

**1. INTRODUCTION**

The Current artificial intelligence has developed to a point where programs can learn by the humans and effectively simplistic human conversations which is essential. One of the best-known examples of chatbots in recent history is Siri the AI assistant that is part of Apple's standard software for its products. Siri took chatbot mainstream in 2011. Since then, brands in every sector have started to use them, eventually developing a new trend conversational in user experience. This refers to an end-user experience in which your interaction with a firm or service is automated based on user prior behavior. If users are developing artificial intelligence applications like Alexa, which enables the use of voice to control devices. If you are a user, you can already interact with this Artificial Intelligence chatbot on popular messaging platforms like Facebook, Instagram and so on. Nowadays the use of chatbots has spread from user customer service to life and death risks. Chatbots are coming into the healthcare industry and can help to solve health problems. Health and fitness chatbots have begun to gain popularity in the market. Previous year Facebook has started allowing healthcare industries to create Messenger chatbots which would then communicate with users. A great example is Health Tap the first company to release a health bot on the Messenger app. It allows users to ask their medical-related queries and receive answers.

The chatbot is also known as chatter robots, are software agents that simulate human conversation via text or voice messages. One of the 1st and main goals of Chatbot had always been to resemble an intelligent human and make it hard for others to understand their real nature. With the development of more Chatbots of various architectures and capabilities, their usage has widely expanded. These conversational agents can go into a point of fooling the users and making them believe they are talking to a human but are very limited in improving their knowledge base at runtime. In order to understand the user input and provide a meaningful response, the chatbot uses artificial intelligence and deep learning methods. Moreover, they interact with humans, using natural language, different applications of Chat-bots such as medical chatbots, call centers, etc. A chatbot could help doctors, nurses, patients or their families. Better organization of patient information, medication management, helping in emergencies or with first aid, offering a solution for superficial medical issues: these are all possible situations for chatbots to step in and reduce the burden on medical professionals.

### **1.1 Project Purpose**

A medical chatbot facilitates the job of a healthcare provider and helps improve their performance by interacting with users in a human-like way. There are countless cases where intelligent medical chatbots could help physicians, nurses, therapists, patients, or their families. At the most basic level, a chatbot is a computer program that simulates and processes human conversation (either written or spoken), allowing humans to interact with digital devices as if they were communicating with a real person.

**1.2 Project Scope**

The future is not all voice interfaces, but instead it should focus on working within device capabilities to boost user accessibility and flexibility. Some chatbots can offer voice-to-text interfaces, voice-to-voice interfaces or text-to-voice interfaces, all depending on customer need or brand decision. The future of chatbots is that businesses will automate simple payments and allow users to pay directly over live chat or Facebook Messenger apps. The instant process makes the customer happy and improves customer satisfaction. MasterCard has also launched a chatbot, especially for customer payments

### **1.3 Project Overview**

Chatbots take the input from voice or text from the user, can extract medical details information using simple questions about current doctor, hospital details and diagnostic center details. Chatbots then store this information in the medical facility system to facilitate patient doctor-patient communication, and medical record keeping.

### **1.4 Features**

A medical chatbot is an AI-powered conversational solution that helps patients and healthcare companies connect with each other and provide a seamless customer experience. Various healthcare organizations have implemented medical chatbot to eliminate workload. As the employees will be busy taking care of patients, the Medical chatbot is here to rescue. Medical chatbot helps the patient to get all the information, fix an appointment and even receive feedback.

# 

# 

# **LITERATURE SURVEY**

## **2. LITERATURE SURVEY**

The literature review plays an important role in understanding the artificial intelligence domain and implementation of the working system. We have included three literature survey papers with proper explanations.

In this paper, Megha Manilal, Shobana AJ, Belfin RV have explained the usage of healthcare chatbot for cancer patients . This paper explains that the cancer can be detected at an early stage. Detection of cancer at an early stage helps to cure the disease and save the patient’s life. Most of the people detect the cancer at the last stage. Cancer is a disease which causes due to lasting growth and spread of abnormal cells. Cancer patients lose hope to live longer and healthier lives. Depression is expeditiously becoming one of the difficult phases in the health sector. In this paper, communication helps a lot to improve one’s mental health, this problem gets solved partially if the patient tries to open up to someone, but nobody is available at right time. This is the reason where chatbot comes into limelight . NLP is used in making of this chatbot which is an important component of artificial intelligence, so we can imbibe same thing in our chatbot for generation of accurate and responsive answers with respect to given queries. This project creates the lucidity in the public data distribution system as the workload becomes faster. The profit to an integrated GPU unit is that it is cheaper which in turn means a less costly computer device. Integrated graphic cards such as Intel, Ryzen also produce less heat and use drastically less power. However, there is no proper security system such as biometric system.

In this paper, Kalpana Devi S, Indumati V, Ishwariya S and Priya Shankar M had taken a survey on Medical Self Diagnosis . This Paper helps to understand the current health related issues and helps by providing necessary suggestion. Here this Chabot is implemented on mobile based applications to handle the user queries. Artificial intelligence is the key methodology used to build those intelligence bots. It is a field of computer engineering that highlights the development of smart machines that work and reacts like humans. Patients having more weight have a serious risk of developing dangerous diseases and health conditions. A rising trend of obesity is not only limited to developed countries, but to developing nations as well. As smartphones have rapidly gained for the sake of weight loss, mobile applications (apps) are used in public health as an intervention to keep track of diets, behavior, and weight, which is considered more effective than relying on consumer self-report measures. To facilitate objective data, a solution called "Smart Wireless Interactive Healthcare System" (Switches) is being developed. This Chabot system would provide a temporary detection, but patient will have to confirm it with the hospital . The dataset was used for cancer was not able to use for the learning model.

In this paper, Divya Madhu, Shinoy Shaji, Neeraj Jain had implemented medical assistance with the help of natural language processing and mobile development technology . The healthcare chatbot had used and tested on several android devices. Nowadays mobile conversations are becoming the trend in communication. The specialty of such conversations is, they are really simple and time saving mode of communication. So, chatbot can be really successful if it follows all the simplicity of an instant messaging application. Chabot’s are usually text driven, with images and unified widgets, which makes it easy to start interacting with a bot. Also, clarity is what helped the most successful brands win confidence users. These things are the core of a Chatbot concept that’s the uses gained the success are doomed for success. Basically, there are two types of Chabot’s are available unintelligent ones that act using predefined conversation flows written by people and intelligent AI Chabot’s that use machine learning. User can read about these two in more detail information in some of our other blog posts. User can fill it with your personality, user brand’s identity, make it speak to your users like you would, change its message depending on the input from the user . Nowadays medicine description portal is really stirring and unconventional in interaction and detailing of medicine.

### **2.1 Related Work**

**Healthcare Chatbot System**

Earlier, the artificial intelligence domain was not developed. After the invention of chatbot systems, the problems of users are solved in less time. In the field of healthcare, automated chatbot deployment in web applications is booming all over the world. Patients suffers from different types of diseases and visit to hospital for treatment purpose. Sometimes doctors are not available due to that, time required for nursing takes a lot [1]. To overcome this issue, medical chatbots were developed. These chatbots are trained and tested on live dataset also accuracy of the output is relevant. The AI based chatbot are fast, reliable and precise. Users provide the proper details and receive feedback according to their query. If any user makes minor mistake, the chatbot provides validation and autocorrection features. Nowadays, in all clinics and hospitals portal chatbots are performing multitasking work. A lot of time of patient is saved and tasks are completed in minimum effort.

**Natural Language Processing**

Natural Language Processing (NLP) is a domain of artificial intelligence that provides machines to learn, read and understand the meaning of human languages. Different fields such as Banking, Education, Finance and so on are using chat applications for solving their problems and marketing their products. Majority of the countries like Germany, Spain, China, India, Korea and so on have their own native language. While interacting with other user, language barrier is created. To reduce this type of complication NLP based autonomous provides variety of languages, which makes communication through verbal and nonverbal easier . In the healthcare chatbot, NLP is used for text processing. The characteristics of NLP in medical domain is useful: 1. Sentence Tokenization: In this method whole sentence is divided is into substrings and breaks into smaller words. When a patient enters the query in form of sentences, a whole set of words are converted into tokens, also split the sentences when there is punctuation mark. 2. Word Tokenization: There are varieties of words, which are present in the dictionaries. The segmentation of sentences is carried out. Words are assigned to tokens. The chatbot helps to classify the words according to the category present in medical records and gives the optimum feedback to a user. 3. Stemming and Lemmatization: Stemming is process, where words are chopped out from beginning and end. Whereas lemmatization is used in morphological analysis for extraction of words. When doctors entered the wrong data on their database, the responses of chatbot are not relevant. These above techniques are used to correct the information and give proper response to a patient in form of text.

# **PROBLEM ANALYSIS**

## **PROBLEM ANALYSIS**

### **3.1 EXISTING SYSTEM**

Many of the existing systems have chats through texts. Some limitations of such Chatbots are, there is no instant response given to the patient, they have to wait for expert’s acknowledgement for a long time. And also, there are a limited number of diseases in the dataset. Technical issues like voice messages are not accurate in the existing system. Some of the processes may charge amount to live chat or telephony communication. However, the issue of these technologies is cost effective in clinical practice remains a consideration for future research studies Now a days, health care is extremely necessary in our life. Today's people are busy with their works reception, workplace works and additional addicted to web. They are not involved regarding their health. So, they avoid to travel in hospitals for little issues.it may become a significant drawback.

By utilizing the field of Artificial Intelligence, one can develop numerous applications one of that is mentioned in this paper is a HealthCare chatbot system. In spite of the fact that chatbot can be deployed in various fields like marketing, education, banking, clinical and finance. Research is being done in making the regular rule based chatbots to be informative, responsive and complete the correspondence in a conversational human language

**Disadvantages**

* In Existing system, it takes more time to response to the user question
* .Pay some charges to perform live chat

### **3.2 PROPOSED SYSTEM**

The chatbots are conversational virtual assistants who automate interactions with the users. Chatbots are powered by artificial intelligence using machine learning techniques to understand natural language. The main motive of the paper is to help the users regarding minor health information. Initially when the user’s visits the website can ask the bot, queries. The system uses an expert system to answer the queries if the answer is not present in the database, then suggestion will be sent to the admin. Here the domain experts also should register themselves by giving various details. The data of the chatbot stored in the database in the form of pattern-template. Here SQL is used for handling the database

**Advantages**

* In proposed system it is reducing health care cost.
* It saves user’s time.

### **3.3 MODULES**

##### 

##### **3.3.1 Admin**

In this module, the admin login into the application using the admin credentials .user can upload the dataset into the dataset, admin can view the suggestion of the user and then resolve the issues.

**3.3.2 User**

User can send the query through voice or text, then algorithm is used, and result is viewed in the user page.

### **3.4 ALGORITHM**

**3.4.1 NLP Natural Language Processing**

acts as a fundamental pillar for recognition of language, which is used by Apple’s Siri and Google. It allows technology to recognize human natural language text and speech-based commands and include two major components natural language generation (NLG) and natural language understanding (NLU). Natural language understanding is more laborious than natural language generation, as the natural language has a remarkably rich structure and form. It maps the given input and analyzes multiple features of the language.

**NLU Natural Language**

Understanding is responsible for handling and converting formless data into a proper form that the system can easily understand NLP has further five main steps if we want that message should be easily understandable by a chatbot.

These steps are

· Lexical analysis

· Syntactic analysis (parsing)

· Semantic analysis

· Discourse integration

· Pragmatic analysis

**The lexical analysis**

Includes analysis and identification of words structure; it splits the text into the chapters, then into sentences, phrases, and words.

**Syntactic analyzer**

Parsing analyzes grammar and arrangement of words so that the relation among different words become more explicit. Sentences like “the hospital go to the doctor,” Are rejected by Syntactic analyzer.

**Semantic Analysis**

Check that either the text is completely meaningful or not, and it draws its correct meaning while mapping syntactic constructions. The semantic analysis will reject the phrase like “cold fire”.

**Pragmatic analysis and discourse**

Integration: analyze the concluding interpretation of the real message of the text. Such as the actual meaning of a phrase or a sentence relay on the overall context.

**NLG Natural Language Generation**

Involves text realization and text planning to generate an understandable response. In simple words, language generation is responsible for the formation of linguistically correct sentences and phrases. The key challenge faced by NLP is to understand the complications of natural human language The structure of language is itself very vague regarding syntax, lexis, and other components of speech such as similes and metaphors. A single word can be taken as a noun or a verb; a single sentence can be passed in many ways; moreover, a single input may have multiple meanings, etc.

**ASR Automatic Speech Recognition**

Comes under computational linguistics, which develops technologies and methodologies that enable the identification and translation of user speech into text with the help of computers. It is also called computer speech recognition, automatic speech recognition (ASR), or speech to text (STT). It includes research and knowledge in electrical engineering, linguistics, and computer science fields. Systems for speech recognition involve “training” which is also called enrollment, where individual speakers read isolated vocabulary or texts into the systems. Systems then analyze the specific voice of a user and use it to fine-tune the identification of that user’s speech, which results in enhanced accuracy. The system that does not require training is known as "speaker independent systems" on the other side systems that require the training are known as “speaker dependent”. The terms speaker identification or voice recognition refers to identify the speaker despite what they are trying to say. Identification of the speaker simplifies speech translation in the system, which is skilled in the voice of a particular person, or it can also be used to verify or authenticate the identity of the user for the security process. From the viewpoint of technology, the history of speech recognition is very vast, having remarkable innovations. Recently this field has progressed a lot by the advancement in big data and deep learning. The advancement is not only proved by the published academic papers but also by the variety of detailed learning methodologies adopted by industry in designing and deployment of speech recognition systems.

**Porter stemming algorithm**

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

For example: words such as “Likes”,” liked”,” likely” and” liking” will be reduced to “like” after stemming. In 1980, Porter presented a simple algorithm for stemming English language words.

How to use Porter Stemmer in nltk

Step 1 - Import the NLTK library and from NLTK import Porter Stemmer. import nltk from nltk. stem import Porter Stemmer.

Step 2 - Create a variable and store Porter Stemmer into it. ps = PorterStemmer()

Step 3 - let's see how to use PorterStemmer. print(ps.stem('bat')) print(ps.stem('batting'))

# **SYSTEM SPECIFICATIONS**

## **4. SYSTEM SPECIFICATIONS**

### **4.1 SYSTEM REQUIREMENTS**

**4.1.1 Hardware Requirements**

|  |  |  |
| --- | --- | --- |
| Processor | - | Pentium-IV |
| RAM | - | 4 GB RAM |
| Hard Disk | - | 512 GB |
| Keyboard | - | Standard Windows Keyboard |
| Mouse | - | Two or Three Button Mouse |
| Monitor | - | SVGA |

**4.1.2 Software Requirements**

|  |  |  |
| --- | --- | --- |
| Operating System | - | Windows 7 or higher |
| Coding Language | - | Python |
| Front End | - | Html, CSS |
| Back End  Server  Web Framework | -  -  - | My sql  Apache  Flask |

### **4.2 FUNCTIONAL REQUIREMENTS**

Functional requirements describe what the system should do, i.e., the services provided for the users and for other systems. The functional requirements can be further categorized as follows

**Input Requirements**

It describe what inputs the system should accept and under what conditions. The following are the input requirements for the present system.

* Reading the query from text or voice
* Loading the data from the database

**Output Requirements**

It describes what outputs the system should produce and under what conditions. Outputs can be to the screen or printed. The following are the output requirements for the present system.

* Display the prior and posterior probabilities**.**
* Assign the class labels to the multi-class text data set instances.

**Storage Requirements**

It describes the system should store those other systems might use. The following are the storage requirements for present system.

* The system stores the various benchmark text-classification data sets

**Computation Requirements**

It describes what computations the system should perform. The computations should be described at a level all the readers can understand. The following are the computation requirements for present system.

* Compute posterior probabilities by using Naïve Bayes classifier.
* Predict the class label of multi-class text using ECOC with Naïve Bayes

**4.3 Non-Functional Requirements**

Non-functional requirements as name suggests are those requirements which are not directly concerned with the specific functions delivered by the system. They may relate to emergent system properties such as reliability, response time, performance, and store occupancy. Many non-functional requirements relate to the system rather than to individual system features. This means that they are often more critical than individual functional requirements While failure to meet an individual functional requirement may degrade the system, failure to meet a non-functional system requirement may make the whole system unusable. These requirements are usually constraints that must be considered when designing the solutions.

**Performance**

* Response Time - Image Processing Pipeline should complete in under 20 seconds

**Scalability**

* System should scale up to 10000 simultaneous credit card transactions at any given moment

**Availability**

* System should be available 24/7; Any unavoidable shutdowns as part of maintenance must be informed 2 days prior

**Reliability**

* Online in-sync backup database and application servers should be available

**Recoverability**

* In case of unexpected or planned maintenance system shutdown, reboot and recover operation should be quick and automated

**Maintainability**

* Xml or Json or any text-based configuration facility must be provided

**Security**

* All data must be secured with privacy, data integrity, data confidentiality preserved at all costs.

**Regulatory**

* 2-layer authentication and data encryption must be provided.

**Environmental**

* The system should be available on multiple form-factors including but not limited to feature phones, smart phones, tablet pc, laptop, desktop, etc.
* The access point must be able to utilize the system with or without internet (utilize SMS where necessary)

**Data Integrity**

* All user master data must be provided data confidentiality and financial transaction data must be verified for data integrity

**Usability**

* The system should be very user friendly considering that ordinary citizens with or without technology know-how should be able to use it.

**Interoperability**

* The system should be platform independent, i.e., be accessible from any operating system, any electronics hardware manufacturer, any banking and other financial entity.

# 

# **TECHNOLOGY DESCRIPTION**

## **5. TECHNOLOGY DESCRIPTION**

### **5.1 SOFTWARE ENVIRONMENT**

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python’s elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

The Python interpreter and the extensive standard library are freely available in source or binary form for all major platforms from the Python Web site, <https://www.python.org/>, and may be freely distributed. The same site also contains distributions of and pointers to many free third-party Python modules, programs and tools, and additional documentation.

The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.

This tutorial introduces the reader informally to the basic concepts and features of the Python language and system. It helps to have a Python interpreter handy for hands-on experience, but all examples are self-contained, so the tutorial can be read off-line as well.

For a description of standard objects and modules, see [The Python Standard Library](#library-index). [The Python Language Reference](#reference-index) gives a more formal definition of the language. To write extensions in C or C++, read [Extending and Embedding the Python Interpreter](#extending-index) and [Python/C API Reference Manual](#c-api-index). There are also several books covering Python in depth.

This tutorial does not attempt to be comprehensive and cover every single feature, or even every commonly used feature. Instead, it introduces many of Python’s most noteworthy features, and will give you a good idea of the language’s flavor and style. After reading it, you will be able to read and write Python modules and programs, and you will be ready to learn more about the various Python library modules described in [The Python Standard Library](#library-index).

# **5.1.1 The Python Standard Library**

While [The Python Language Reference](#reference-index) describes the exact syntax and semantics of the Python language, this library reference manual describes the standard library that is distributed with Python. It also describes some of the optional components that are commonly included in Python distributions.

Python’s standard library is very extensive, offering a wide range of facilities as indicated by the long table of contents listed below. The library contains built-in modules (written in C) that provide access to system functionality such as file I/O that would otherwise be inaccessible to Python programmers, as well as modules written in Python that provide standardized solutions for many problems that occur in everyday programming. Some of these modules are explicitly designed to encourage and enhance the portability of Python programs by abstracting away platform-specifics into platform-neutral APIs.

The Python installers for the Windows platform usually include the entire standard library and often also include many additional components. For Unix-like operating systems Python is normally provided as a collection of packages, so it may be necessary to use the packaging tools provided with the operating system to obtain some or all the optional components

# **Dealing with Bugs**

Python is a mature programming language which has established a reputation for stability. In order to maintain this reputation, the developers would like to know of any deficiencies you find in Python.

It can be sometimes faster to fix bugs yourself and contribute patches to Python as it streamlines the process and involves less people. Learn how to [contribute](#contributing-to-python).

## **Documentation bugs**

If you find a bug in this documentation or would like to propose an improvement, please submit a bug report on the [tracker](#using-the-tracker). If you have a suggestion how to fix it, include that as well.

If you’re short on time, you can also email documentation bug reports to [docs@python.org](mailto:docs%40python.org) (behavioural bugs can be sent to [python-list@python.org](mailto:python-list%40python.org)). ‘docs@’ is a mailing list run by volunteers; your request will be noticed, though it may take a while to be processed.

See also

[Documentation bugs](https://bugs.python.org/issue?@filter=status&@filter=components&components=4&status=1&@columns=id,activity,title,status&@sort=-activity) on the Python issue tracker

## **Using the Python issue tracker**

Bug reports for Python itself should be submitted via the Python Bug Tracker (<https://bugs.python.org/>). The bug tracker offers a Web form which allows pertinent information to be entered and submitted to the developers.

The first step in filing a report is to determine whether the problem has already been reported. The advantage in doing so, aside from saving the developers time, is that you learn what has been done to fix it; it may be that the problem has already been fixed for the next release, or additional information is needed (in which case you are welcome to provide it if you can!). To do this, search the bug database using the search box on the top of the page.

If the problem you’re reporting is not already in the bug tracker, go back to the Python Bug Tracker and log in. If you don’t already have a tracker account, select the “Register” link or, if you use OpenID, one of the OpenID provider logos in the sidebar. It is not possible to submit a bug report anonymously.

Being now logged in, you can submit a bug. Select the “Create New” link in the sidebar to open the bug reporting form.

The submission form has a number of fields. For the “Title” field, enter a very short description of the problem; less than ten words is good. In the “Type” field, select the type of your problem; also select the “Component” and “Versions” to which the bug relates.

In the “Comment” field, describe the problem in detail, including what you expected to happen and what did happen. Be sure to include whether any extension modules were involved, and what hardware and software platform you were using (including version information as appropriate).

**Introduction to Data Mining**

Data mining integrates approaches and techniques from various disciplines such as machine learning, statistics, artificial intelligence, Decision Trees, database management, data warehousing, data visualization, spatial data analysis, probability graph theory etc. In short, data mining is a multi-disciplinary field.

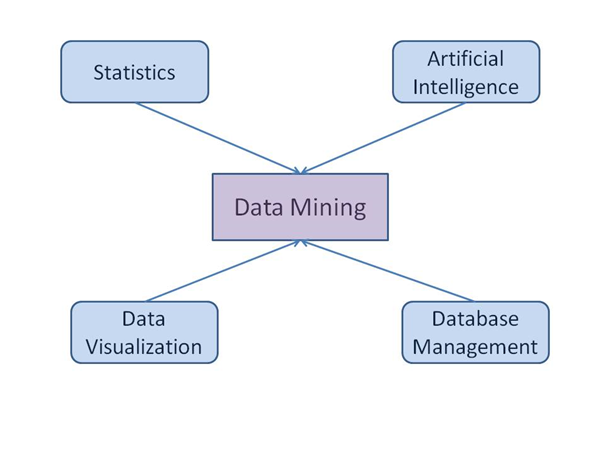


Figure 1

**Statistics**

Statistics includes a number of methods to analyze numerical data in large quantities. Different statistical tools used in data mining are regression analysis, cluster analysis, correlation analysis and Bayesian network. Statistical models are usually built from a training data set. Correlation analysis identifies the correlation of variables to each other. Bayesian network is a directed graph that represents causal relationship among data found out using the Bayesian probability theorem. Given below is a simple Bayesian network where the nodes represent variables whereas edges represent the relationship between the nodes.

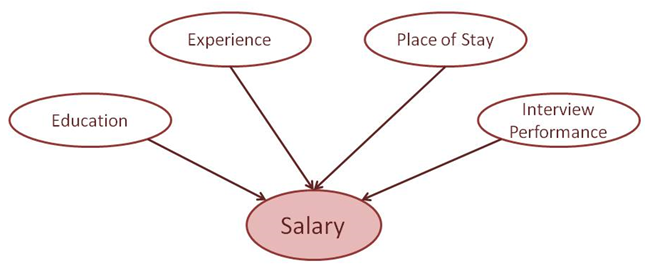


Figure 2

**Machine Learning**

Machine learning is the collection of methods, principles and algorithms that enables learning and prediction on the basis of past data. Machine learning is used to build new models and to search for a best model matching the test data. Machine learning methods normally use heuristics while searching for the model. Data mining uses a number of machine learning methods including inductive concept learning, conceptual clustering and decision tree induction. A decision tree is a classification tree that decides the class of an object by following the path from the root to a leaf node. Given below is a simple decision tree that is used for weather forecasting.

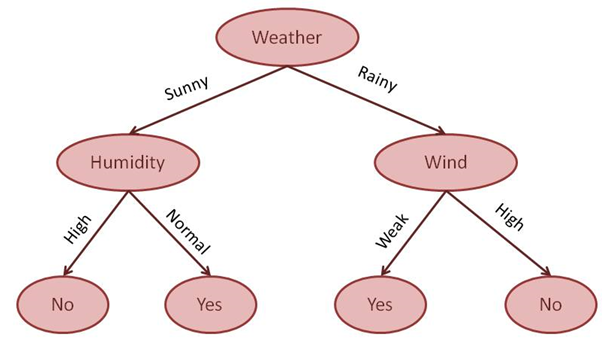


Figure 3

**Database Oriented Techniques**

Advancements in database and data warehouse implementation helps data mining in a number of ways. Database oriented techniques are used mainly to develop characteristics of the available data. Iterative database scanning for frequent item sets, attribute focusing, and attribute-oriented induction are some of the database-oriented techniques widely used in data mining. The iterative database scanning searches for frequent item sets in a database. Attribute oriented induction generalizes low level data into high level concepts using conceptual hierarchies.

**Decision Trees**

A Decision Tree is a set of connected nodes called neurons. A neuron is a computing device that computes some requirement of its inputs, and the inputs can even be the outputs of other neurons. A Decision Tree can be trained to find the relationship between input attributes and output attribute by adjusting the connections and the parameters of the nodes.

**Data Visualization**

The information extracted from large volumes of data should be presented well to the end user and data visualization techniques make this possible. Data is transformed into different visual objects such as dots, lines, shapes etc. and displayed in a two- or three-dimensional space. Data visualization is an effective way to identify trends, patterns, correlations and outliers from large amounts of data.

**Summary**

Data mining combines different techniques from various disciplines such as machine learning, statistics, database management, data visualization etc. These methods can be combined to deal with complex problems or to get alternative solutions. Normally data mining system employs one or more techniques to handle different kinds of data, different data mining tasks, different application areas and different data requirements.

**Patterns in Data Mining**

**1. Association**

* The items or objects in relational databases, transactional databases or any other information repositories are considered, while finding associations or correlations**.**

**2. Classification**

* The goal of classification is to construct a model with the help of historical data that can accurately predict the value.
* It maps the data into the predefined groups or classes and searches for the new patterns.
* For example: To predict weather on a particular day will be categorized into - sunny, rainy, or cloudy.

**3. Regression**

* Regression creates predictive models. Regression analysis is used to make predictions based on existing data by applying formulas.
* Regression is very useful for finding (or predicting) the information on the basis of previously known information.

**4. Cluster analysis**

* It is a process of portioning a set of data into a set of meaningful subclasses, called as cluster.
* It is used to place the data elements into the related groups without advanced knowledge of the group definitions.

**5. Forecasting**

* Forecasting is concerned with the discovery of knowledge or information patterns in data that can lead to reasonable predictions about the future.
* Technologies used in data mining
* Several techniques used in the development of data mining methods. Some of them are mentioned below

**1. Statistics**

It uses the mathematical analysis to express representations, model and summarize empirical data or real-world observations. Statistical analysis involves the collection of methods, applicable to large amount of data to conclude and report the trend.

**2. Machine learning**

Arthur Samuel defined machine learning as a field of study that gives computers the ability to learn without being programmed.

When the new data is entered in the computer, algorithms help the data to grow or change due to machine learning.

In machine learning, an algorithm is constructed to predict the data from the available database (Predictive analysis).

It is related to computational statistics.

The four types of machine learning are

* **Supervised learning**  
    
  It is based on the classification. It is also called as inductive learning. In this method, the desired outputs are included in the training dataset.
* **Unsupervised learning**

Learning is based on clustering. Clusters are formed on the basis of similarity measures and desired outputs are not included in the training dataset.  
Unsupervised

* **Semi-supervised learning**  
    
   Semi-supervised learning includes some desired outputs to the training dataset to generate the appropriate functions. This method generally avoids the large number of labeled examples (i.e., desired outputs).
* **Active learning**

Active learning is a powerful approach in analyzing the data efficiently.The algorithm is designed in such a way that, the desired output should be decided by the algorithm itself(the user plays important role in this type).

**3. Information retrieval**

Information deals with uncertain representations of the semantics of objects (text, images).   
For example: Finding relevant information from a large document.

**4. Database systems and data warehouse**

Databases are used for the purpose of recording the data as well as data warehousing.

Online Transactional Processing (OLTP) uses databases for day-to-day transaction   
purpose. To remove the redundant data and save the storage space, data is normalized and stored in the form of tables.

Entity-Relational modeling techniques are used for relational database management system design.

Data warehouses are used to store historical data which helps to take strategical decision for business. It is used for online analytical processing (OALP), which helps to analyze the data.

**5. Decision support system**

Decision support system is a category of information system. It is very useful in decision making for organizations.

It is an interactive software-based system which helps decision makers to extract useful information from the data, documents to make the decision.

**KDD and Data mining**

* The process of discovering knowledge in data and application of data mining techniques are referred to as Knowledge Discovery in Databases (KDD).
* KDD consists of various application domains such as artificial intelligence, pattern recognition, machine learning and data visualization.
* The main goal of KDD is to extract knowledge from large databases with the help of data mining methods.
* The different steps of KDD are as given below  
    
  **1. Data cleaning**

In this step, noise and irrelevant data are removed from the database.  
  
**2. Data integration**

In this step, the heterogeneous data sources are merged into a single data source.  
  
**3. Data selection**

In this step, the data which is relevant to the analysis process gets retrieved from the database.  
  
**4. Data transformation**

In this step, the selected data is transformed in such forms which are suitable for data mining.  
  
**5. Data mining**

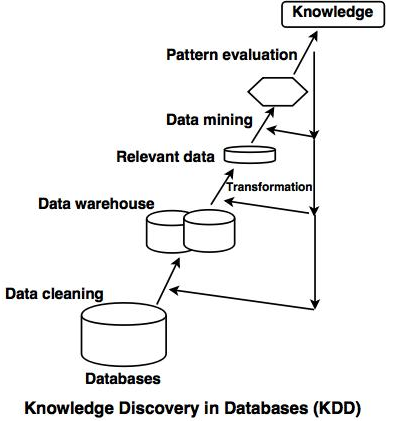
In this step, the various techniques are applied to extract the data patterns.

**6. Pattern evaluation**

In this step, the different data patterns are evaluated.

**7.Knowledge representation**

This is the final step of KDD, which represents the knowledge.



# **SYSTEM DESIGN**

## **6. SYSTEM DESIGN**

#### **6.1 INTRODUCTION TO SYSTEM DESIGN**

#### **Input design**

Input Design plays a vital role in the life cycle of software development, and it requires meticulous attention of developers. The input design is to feed data to the application as accurately as possible. So, inputs are supposed to be designed expertly so that the errors occurring while feeding is minimized. According to Software Engineering Concepts, the input forms or screens are designed to provide to have a validation control over the input limit, range, and other related validations.

This system has input screens in almost all the modules. Error messages are developed to alert the user whenever he commits some mistakes and guides him in the right way so that invalid entries are not made. Let us see deeply about this under module design.

Input design is the process of converting the user-created input into a computer-based format. The goal of the input design is to make the data entry logical and free from errors. The error is in the input is controlled by the input design. The application has been developed in a user-friendly manner. The forms have been designed in such a way during the processing; the cursor is placed in the position where must be entered. The user is also provided with an option to select an appropriate input from various alternatives related to the field in some instances.

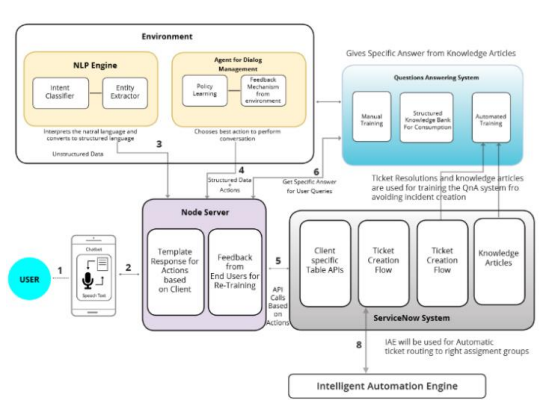
Validations are required for each data entered. Whenever a user enters an erroneous data, an error message is displayed, and the user can move on to the subsequent pages after completing all the entries in the current page.

#### **Output design**

The Output from the computer is required to mainly create an efficient method of communication within the company primarily among the project leader and his team members, in other words, the administrator and the clients. The output of VPN is the system which allows the project leader to manage his clients in terms of creating new clients and assigning new projects to them, maintaining a record of the project validity and providing folder level access to each client on the user side depending on the projects allotted to him. After completion of a project, a new project may be assigned to the client. User authentication procedures are maintained at the initial stages itself. A new user may be created by the administrator himself, or a user can see himself register as a new user, but the task of assigning projects and validating a new user rest with the administrator only.

The application starts running when it is executed for the first time. The server has to be started, and then the internet explorer in used as the browser. The project will run on the local area network so the server machine will serve as the administrator while the other connected systems can act as the clients. The developed system is highly user friendly and can be easily understood by anyone using it even for the first time.

**6.1.1 System Architecture**



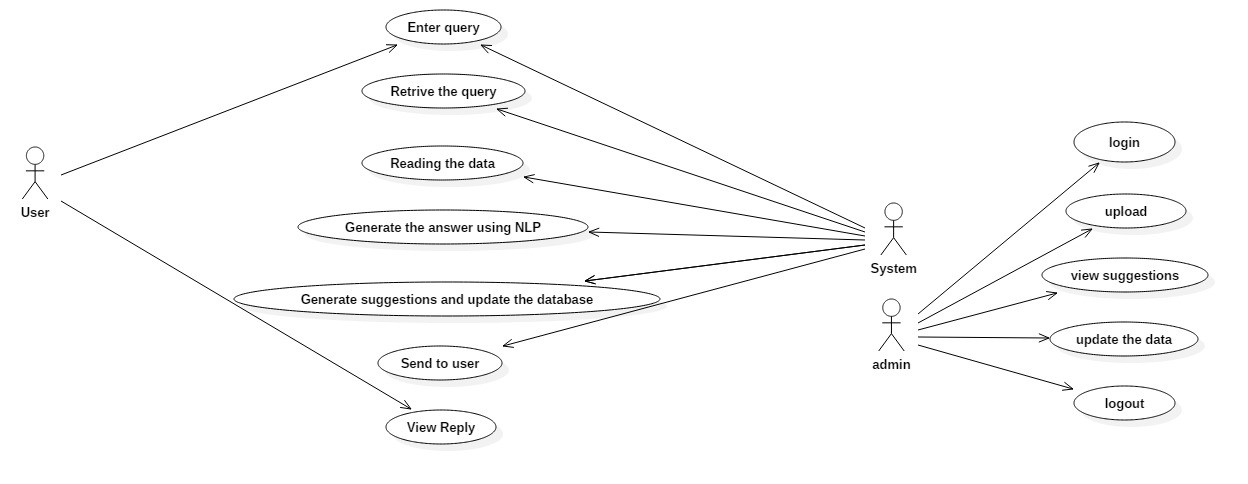
**Figure 6.1: System Architecture**

**6.1.2 Unified Modeling Language**

A UML diagram is a diagram based on the UML (Unified Modeling Language) with the purpose of **visually representing a system** along with its main actors, roles, actions, artifacts or classes, in order to understand better, alter, maintain, or document information about the system

**6.1.2.1 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 Specify the context of a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.



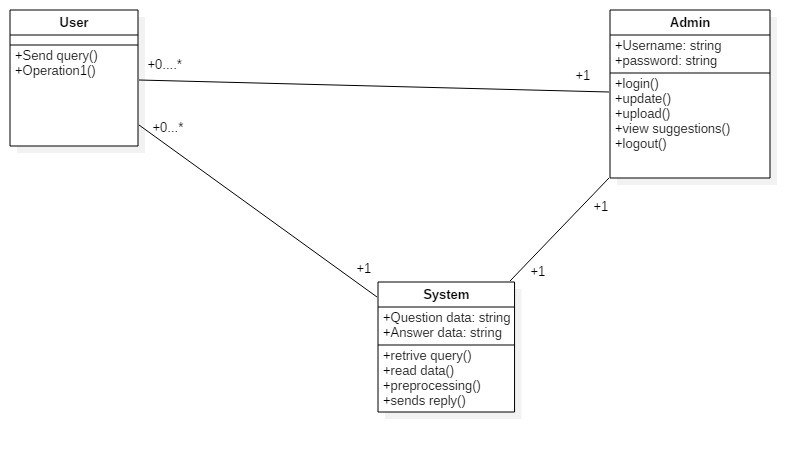
##### **Figure 6.2: Use case Diagram**

**6.1.2.2 The Class Diagram**

The class diagram is a static diagram. A class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

The class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

The class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.



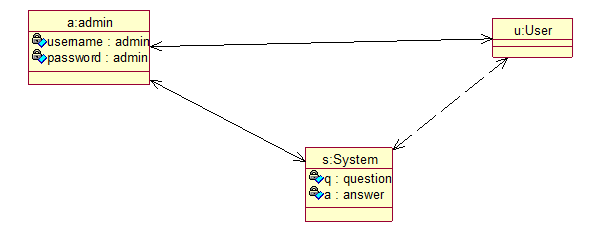
##### **Figure 6.3: Class Diagram**

**6.1.2.3 Object Diagram**

Object diagrams are derived from class diagrams, so object diagrams are dependent upon class diagrams.

Object diagrams represent an instance of a class diagram. The basic concepts are similar to class diagrams and object diagrams. Object diagrams also represent the static view of a system, but this static view is a snapshot of the system at a particular moment.

Object diagrams are used to render a set of objects and their relationships as an instance.

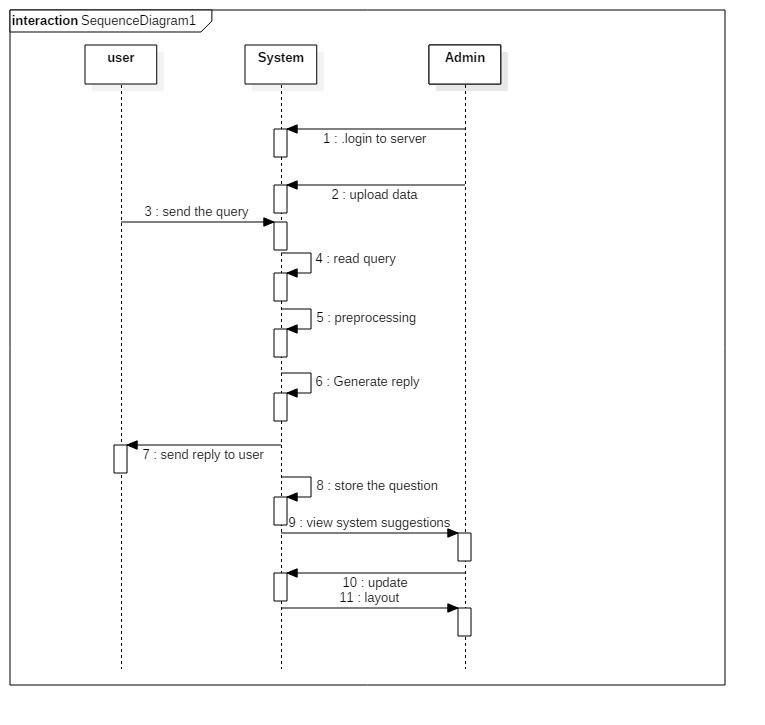


##### **Figure 6.4: Object Diagram**

**6.1.2.4 Sequence Diagram**

A sequence diagram simply depicts the interaction between objects in sequential order, i.e., the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by software developers to document and understand requirements for new and existing systems.

The sequence diagram consists of actors, objects, Lifelines, Messages (Self Messages, Synchronous messages, Asynchronous messages)

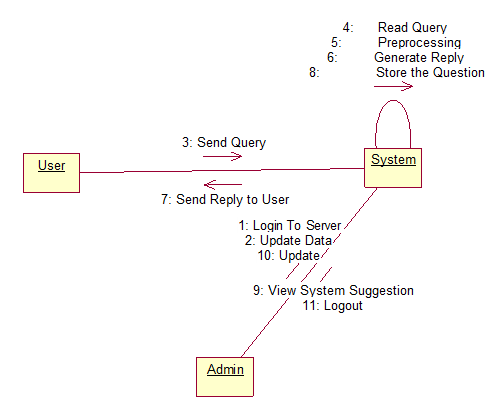


##### **Figure 6.5: Sequence Diagram**

**6.1.2.5 Collaboration Diagram**

The collaboration diagram represents the interaction of the objects to perform the behavior of a particular use case or a part of the use case. The designers use the Sequence diagram and Collaboration Diagrams to define and clarify the roles of the objects that perform a particular flow of events of a use case.

The sequence diagram is used to represent the sequence of messages that are flowing from one object to another. The collaboration diagram is used to represent the structural organization of the system and the message that is sent and received.

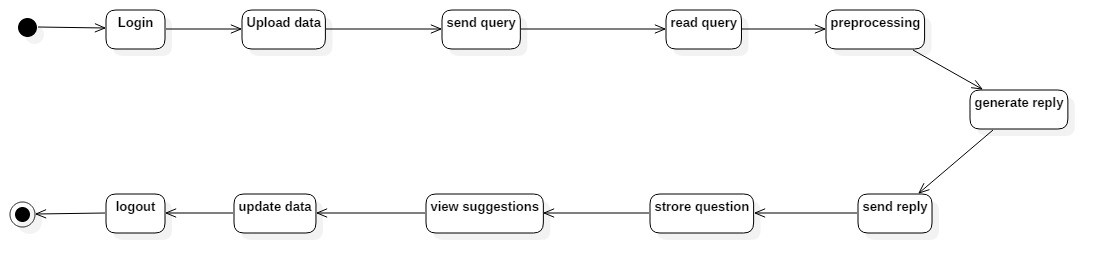


##### **Figure 6.6: Collaboration Diagram**

**6.1.2.6 State Chart Diagram**

It describes different states of a component in a system. The states are specific to a component/object of a system.

A State chart diagram describes a state machine. A state machine can be defined as a machine that defines different states of an object, and these states are controlled by external or internal events.



##### **Figure 6.7: State chart Diagram**

**6.1.2.7 Activity Diagram**

An activity diagram portrays the control flow from a start point to a finish point, showing the various decision paths that exist. In contrast, the activity is being executed; we can depict both sequential processing and concurrent processing of activities using an activity diagram.

Diagram

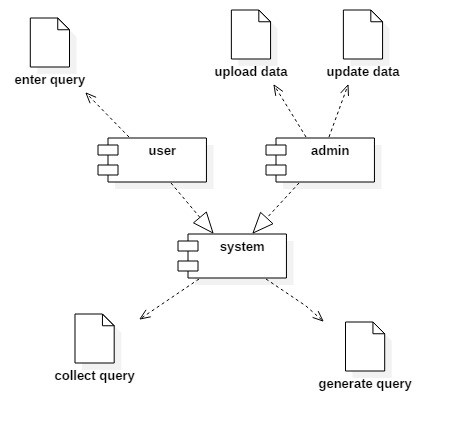
Description automatically generated

##### **Figure 6.8: Activity Diagram**

**6.1.2.8 Component Diagram**

Component diagrams are used to model the physical aspects of a system. Now the question is, what are these physical aspects? Physical aspects are elements such as executable, libraries, files, documents, etc. which reside in a node.

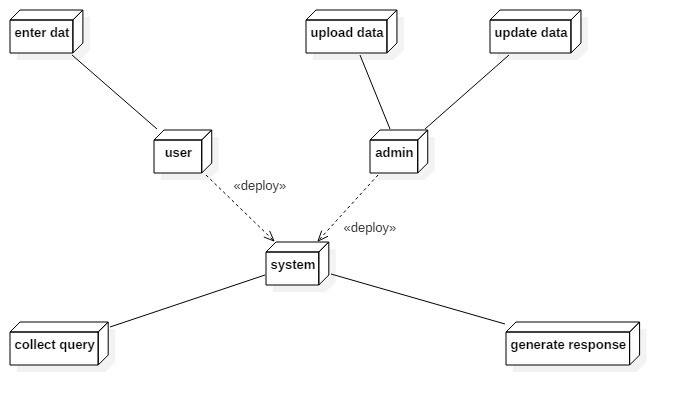
Component diagrams are used to visualize the organization and relationships among components in a system. These diagrams are also used to make executable systems. These are used for executable files, Library files, Tables, files.



##### **Figure 6.9: Component Diagram**

**6.1.2.9 Deployment Diagram**

Deployment Diagram is a type of diagram that specifies the physical hardware on which the software system will execute. It also determines how the software is deployed on the underlying hardware. It maps software pieces of a system to the device that is going to execute it.



**Figure 6.10: Deployment Diagram**

# **SAMPLE CODE**

**7. SAMPLE CODE**

#pip install -U spacy

#python -m spacy download en\_core\_web\_md

#pip install -U wheel

import pathlib

from flask import Flask

from flask import Flask, flash, redirect, render\_template, request, session, abort

import os

import pandas

#from werkzeug import secure\_filename

import mysql.connector

import sys

import glob

import math

import numpy as np

from gtts import gTTS

import speech\_recognition as sr

import spacy

nlp = spacy.load("en\_core\_web\_md")

noise\_test=["is","a","this","an","the","are","?","for","of","who","in","many","they",

"by","who","what","why","when","where","does","how","there","which","any",

"it","have","if","on","or","st","nd","/","and","has","can","you","was","i am",]

def remove\_noise(input\_text):

words=input\_text.split()

noise\_free\_words=[word for word in words if word not in noise\_test]

noise\_free\_text=" ".join(noise\_free\_words)

return noise\_free\_text

app = Flask(\_\_name\_\_)

db= mysql.connector.connect(user='root', database='chatbot1')

std\_embeddings\_index = {}

with open('numberbatch-en.txt', encoding="utf8") as f:

for line in f:

values = line.split(' ')

word = values[0]

embedding = np.asarray(values[1:], dtype='float32')

std\_embeddings\_index[word] = embedding

def cosineValue(v1,v2):

"compute cosine similarity of v1 to v2: (v1 dot v2)/{||v1||\*||v2||)"

sumxx, sumxy, sumyy = 0, 0, 0

for i in range(len(v1)):

x = v1[i];

y = v2[i];

sumxx += x\*x

sumyy += y\*y

sumxy += x\*y

return sumxy/math.sqrt(sumxx\*sumyy)

def get\_sentence\_vector(sentence, std\_embeddings\_index = std\_embeddings\_index ):

sent\_vector = 0

for word in sentence.lower().split():

if word not in std\_embeddings\_index :

word\_vector = np.array(np.random.uniform(-1.0, 1.0, 300))

std\_embeddings\_index[word] = word\_vector

else:

word\_vector = std\_embeddings\_index[word]

sent\_vector = sent\_vector + word\_vector

return sent\_vector

def cosine\_sim(sent1, sent2):

return cosineValue(get\_sentence\_vector(sent1), get\_sentence\_vector(sent2))

def read\_file(filename):

with open(filename, 'rb') as f:

photo = f.read()

sys.setdefaultencoding('utf-8')

return photo

@app.route('/')

def home():

return render\_template('index.html')

@app.route('/adminlogin')

def adminlogin():

return render\_template('adminlogin.html')

@app.route('/upload')

def Upload():

return render\_template('upload.html')

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

def do\_adminlogin():

flag=False

username=request.form['username']

password=request.form['password']

if username=='admin' and password=='admin':

session['logged\_in'] = True

flag=True

else:

flag=False

if flag:

return render\_template('adminhome.html')

else:

return render\_template('adminlogin.html')

@app.route('/Query')

def QueryDB():

file = pathlib.Path("welcome.mp3")

if file.exists ():

os.remove("welcome.mp3")

ques=[]

ans=[]

s3=" "

r = sr.Recognizer()

with sr.Microphone() as source:

print("Speak:")

audio = r.listen(source)

try:

s3=r.recognize\_google(audio)

except sr.UnknownValueError:

print("Could not understand audio")

except sr.RequestError as e:

print("Could not request results; {0}".format(e))

print ("Ur Spech is ",s3)

s2=remove\_noise(s3)

statement1 = nlp(s3)

cursor = db.cursor()

rows\_count=cursor.execute("select \* from quest")

data=cursor.fetchall()

ind=0

ind1=0

max1=0.0

for item in data:

ques.append(item[1])

ans.append(item[2])

ss1=str(item[1])

s1=remove\_noise(ss1)

statement2 = nlp(ss1)

cs=statement1.similarity(statement2)

print( item[1],' ',item[2],' ',cs)

if(cs>max1):

print("Hello")

max1=cs

ind1=ind

ind=ind+1

answer=ans[ind1]

if(max1>=.75):

mytext = answer

language = 'en'

myobj = gTTS(text=answer, lang=language, slow=False)

myobj.save("welcome1.mp3")

os.system("welcome1.mp3")

return render\_template('adminhome1.html',ques=s3,msg=answer)

else:

cursor1 = db.cursor()

print ("s is ",s3)

cursor1.execute('insert into suggestion(quest) values("%s")' % (s3))

db.commit()

mytext = "Answer not Available"

language = 'en'

myobj = gTTS(text=mytext, lang=language, slow=False)

myobj.save("welcome1.mp3")

os.system("welcome1.mp3")

return render\_template('adminhome1.html',ques=s3,msg="Answer not Available")

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

def textQueryDB():

file = pathlib.Path("welcome.mp3")

if file.exists ():

os.remove("welcome.mp3")

ques=[]

ans=[]

s3=request.form['t1']

print ("Ur Spech is ",s3)

s2=remove\_noise(s3)

statement1 = nlp(s3)

cursor = db.cursor()

rows\_count=cursor.execute("select \* from quest")

data=cursor.fetchall()

ind=0

ind1=0

max1=0.0

for item in data:

ques.append(item[1])

ans.append(item[2])

ss1=str(item[1])

s1=remove\_noise(ss1)

statement2 = nlp(ss1)

cs=statement1.similarity(statement2)

print( item[1],' ',item[2],' ',cs)

if(cs>max1):

print("Hello")

max1=cs

ind1=ind

ind=ind+1

answer=ans[ind1]

if(max1>=.75):

mytext = answer

language = 'en'

myobj = gTTS(text=answer, lang=language, slow=False)

myobj.save("welcome1.mp3")

os.system("welcome1.mp3")

return render\_template('adminhome1.html',ques=s3,msg=answer)

else:

cursor1 = db.cursor()

print ("s is ",s3)

cursor1.execute('insert into suggestion(quest) values("%s")' % (s3))

db.commit()

mytext = "Answer not Available"

language = 'en'

myobj = gTTS(text=mytext, lang=language, slow=False)

myobj.save("welcome1.mp3")

os.system("welcome1.mp3")

return render\_template('adminhome1.html',ques=s3,msg="Answer not Available")

@app.route('/suggestion')

def suggestionDB():

cursor = db.cursor()

rows\_count=cursor.execute("select \* from suggestion")

data=cursor.fetchall()

return render\_template('adminhome2.html',ddata=data)

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

def do\_uploadDB():

f = request.files['files']

df = pandas.read\_csv(f)

print(df)

# cursor1 = db.cursor()

# cursor1.execute("delete from quest")

# db.commit()

for index, row in df.iterrows():

cursor = db.cursor()

cursor.execute('insert into quest(question,answer) values("%s", "%s")' % \

(row['question'],row['answer']))

db.commit()

return render\_template('adminhome1.html',msg="File UploadedSuccessfully")

@app.route("/logout")

def logout():

session['logged\_in'] = False

return home()

if \_\_name\_\_ == "\_\_main\_\_":

app.secret\_key = os.urandom(12)

app.run(debug=True,host='0.0.0.0', port=8000)

# **TESTING**

## **8. 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 tests. Each test type addresses a specific testing requirement.

**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.

**Functional test**

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

Functional testing is 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.

**System Test**

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

**White Box Testing**

White Box Testing is a testing in which 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.

**Unit Testing**

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

**Test strategy and approach**

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

**Test objectives**

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

**Features to be tested**

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

**Integration Testing**

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

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

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

**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.

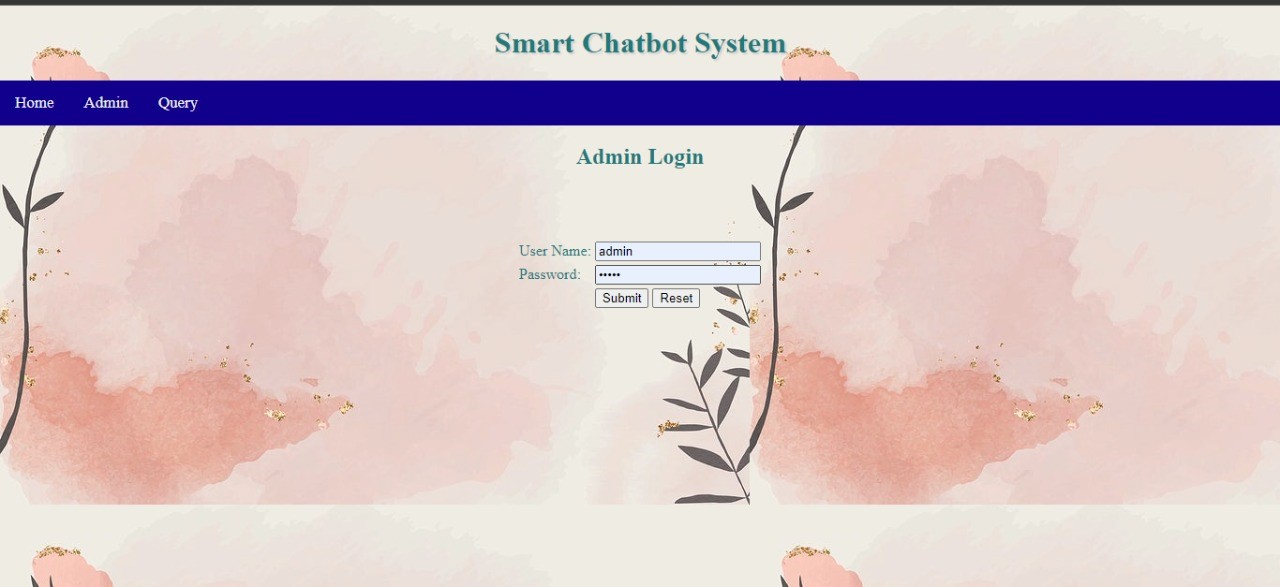
### **TEST CASES**

### **Admin login page**

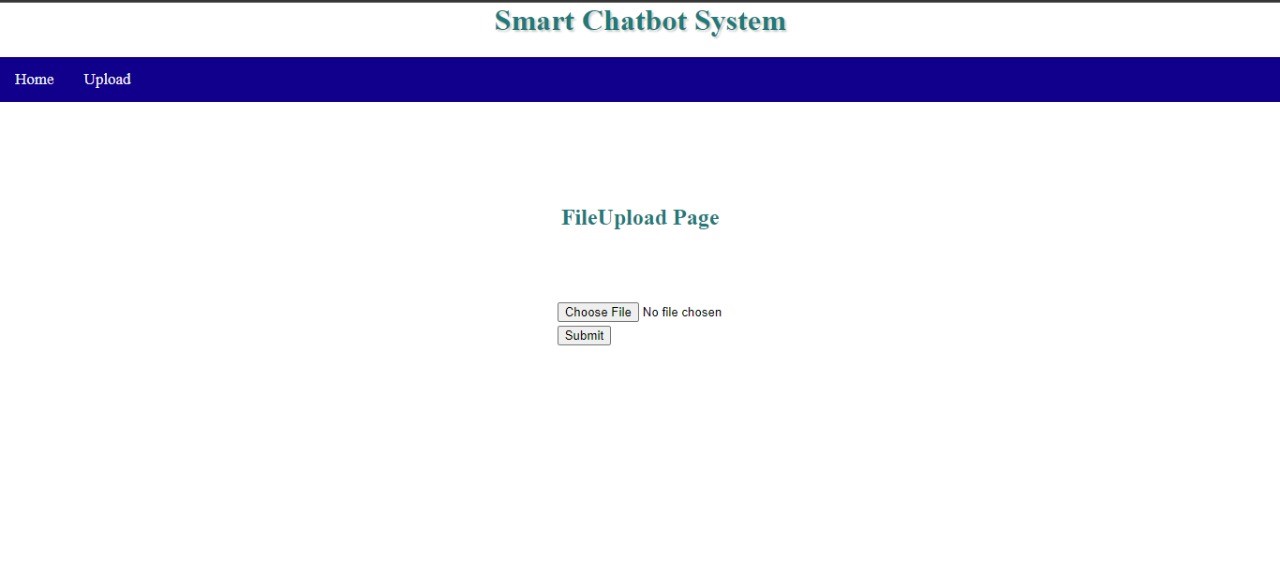
|  |  |  |  |
| --- | --- | --- | --- |
| **SL.No** | **Test Case** | **Excepted Result** | **Test Result** |
| 1 | Enter valid name and password & click on login button | Software should display main window | Successful |
| 2 | Enter invalid | Software should not display main window | successful |
| 3 | User Enters Query | Software should display the result if result is present | successful |
| 4 | User Enters Query | Software should send the suggestion to the admin if result is not available | successful |

# **SCREENS**

**9. SCREENS**







Graphical user interface

Description automatically generated

# **CONCLUSION**

## **10. CONCLUSION**

The main aim of the project AI Based Healthcare chatbot system using Natural Language Processing, which is easy to use and more secure than the current system it will cure the diseases and helps to maintain proper health in the current system. This system reduces the possibility of diseases. The information is processed and store in the database, then it is reverted to the user. Also, it provides an accurate information about the heath symptoms and medicines to the patients. The government will also keep the track of the medicines supplied to the medicals and hospitals. By using diagnosis software, the results are generated accurate and fast. For end users it became easy to gain access in healthcare website and explore different types of services. After using such web-based applications, the results of healthcare were affected in different countries and rate of mortality was steadily decreased. With the help of this natural language processing the proposed system can help the government organizations and hospitals also help in the development of the country. Thus, we successfully build up a system for hospitals and medical institute so that user can ask their queries with the medical assistant and book the doctor’s appointment by giving text messages.

**FUTURE WORK**

Future scope of the project could be AI Based Healthcare chatbot system using NLP can also include a mobile assistant in it which will be more functions will be added and can be accessed by many users. Which will also reduce the time and will also be accurate in the health details of patients given to the doctors. We can add biometric system for more secure authentication process.

# **BIBLIOGRAPHY**

## **BIBLIOGRAPHY**

* "Real World Smart Chatbot for Customer Care using a Software as a Service (SaaS) Architecture “Godson Michael D’silva1”, Sanket Thakare2, Shraddha More Available: <https://www.docme.ru/doc/2207164/i-smac.2017.8058261>
* Divya Madhu, Neeraj Jain C. J, ElmySebastain, Shinoy Shaji, Anandhu Ajaya kumar,” A Novel Approach for Medical Assistance Using Trained Chatbot”, International Conference 2016. Available: <https://www.semanticscholar.org/paper/Anovelapproachformedicalassi> stancusing Madhu Jain/b9117a428a35aca643f83e6986ae2 00807cba5c7
* YerlanJ Saurav Kumar Mishra, Dhirendra Bharti, Nidhi Mishra,” Dr.Vdoc: A Medical Chatbot that Acts as a Virtual Doctor”, Journal of Medical Science and Technology Volume: 6, Issue 3, 2016. Available: [http://medicaljournals.stmjournals.in/index.php/RRJoMST/article/view /30](http://medicaljournals.stmjournals.in/index.php/RRJoMST/article/view%20/30)
* PavlidouMeropi, Antonis S. Billis, Nicolas D.Hasanagas, CharalambosBratsas, Ioannis Antoniou, Panagiotis D. Bamidis, ”Conditional Entropy Based Retrieval Model in Patient-Care Conversational Cases”,2017 IEEE 30th International conference on Computer-Based Medical System. Available:- <https://ieeexplore.ieee.org/abstract/document/8104260>
* Abbas Saliimi Lokman, Jasni Mohamad Zain,FakultiSistemKomputer, Kejuruteraan Perisian,” Designing a Chatbot for Diabetic Patients”, ACM Transactions on Management Information Systems (TMIS), Volume 4, Issue 2, August 2018. Available: <http://ijircce.com/upload/2018/june/7_Pharmabot.pdf>
* Nicolas D .Hasanagas, CharalambosBratsas, Ioannis Antoniou, Panagiotis D. Bamidis, ”Conditional Entropy Based Retrieval Model in Patient-Care Conversational Cases”,2017 IEEE 30th International conference on Computer-Based Medical System. Available: <https://scholar.google.co.in/scholar?q=International+conference+on+C> omputerBased+Medical+System.&hl=en&as\_sdt=0&as\_vis=1&oi=sch olart
* Kowatsch, T., VollanMobile Coach. In: Designingthe Digital Transformation. DESRIST 2017. Lecture Notes in Computer Science, vol 10243, pp. 485-489. Springer, Berlin; Germany (2017). Available: <https://www.researchgate.net/publication/317181609_Design_and_Eva> luation\_of\_a\_Mobile\_Chat\_App\_for\_the\_Open\_Source\_Behavioral\_H ealth\_Intervention\_Platform\_MobileCoach
* Benilda Eleonor V. Comendador, Bien Michael B. Francisco, Jefferson S. Medenilla,Sharleen Mae T. Nacion, “Pharma Bot: A Pediatric Generic Medicine Consultant Chatbot “,Journal of Automation and Control Engineering Vol. 3, No. 2, April 2015. Available: <http://www.joace.org/uploadfile/2014/0801/20140801025000959.pdf>
* Ahmed Fadhil, “Beyond Patient Monitoring: Conversational Agents Role in Telemedicine & Healthcare Support for Home-Living Elderly Individuals”, a researcharticle, April 2017. Available: http://:www.researchgate.net/publication/323845985\_Beyond\_Patient \_Monitoring\_Conversational\_Agents\_Role\_in\_Telemedicine\_Healthcare\_Support\_For\_Home\_Living\_Elderly\_Individuals.pdf
* Fadhil, Ahmed. ” Can a Chatbot Determine My Diet? Addressing Challenges of Chatbot Application for Meal Recommendation.” arXiv preprint arXiv:1802.09100(2018). Available: http://:www.researchgate.net/publication/addressing\_challanges\_of\_ chatbot\_application\_for\_meal\_recommendation.pdf
* Chatterbot library for natural language processing purpose Available: - <https://chatterbot.readthedocs.io/en/stable/corpus.html>
* Healthcare chatbot system website interface template. Available: - <https://hellotars.com/chatbot-templates/healthcare/>
* Doctors and Hospital database connectivity program structure. Available: - <https://www.w3resource.com/sql-exercises/hospital-database-> sqlexercise-on-hospital-database.php
* Chatbot user interface guide by using front-end libraries. Available:- <https://www.userlike.com/en/blog/chatbot-ui>
* Admin and user query form authentication. Available:- <https://www.php.net/manual/en/features.http-auth.php>
* Flask framework tutorial roadmap for developing chatbot Available: - <https://beingdatum.com/build-a-chatbot-using-python-flask/>
* Healthcare dataset in form of yml extension Available:- <http://www.lib4dev.in/info/awesomedata/awesome-publicdatasets/26898879>
* Deployment of healthcare web application on localhost server tutorial Available:- <https://www.hostinger.in/tutorials/how-to-use-xampp-wordpress/>
* Web application testing for creating final test cases reports of user interface guidance. Available:- <https://www.perfecto.io/blog/ui-testing-comprehensive-guide>