Abstract—a chatbot’s aim is to make conversation between humans and machines and they are designed to depict how a human would react in a conversation. This paper shows the implementation of a chatbot which provides information related to the college such as directions to navigate on campus, history of the college, information on the management and administrative staff, as well as details such as the lecturers’ schedules, timetables, event notices, exam dates, frequently asked questions, etc. The chatbot is based on AIML (Artificial Intelligent Markup Language) and NLP (Natural language Processing) for parsing the sentences and training the bot to understand the user’s messages.

Keywords—AIML, NLP, Python, BeautifulSoup, chatbot, telegram

I. INTRODUCTION
The chatbot mimics the conversation flow by using natural language processing and AIML. They are usually used in dialog systems for several purposes which include customer service, acquiring information, virtual assistants. Chatbots are given the capability of small talk to engage the user in casual conversations in order to make them give the feel of being humane. The bot has been embedded knowledge to identify and understand the sentences and find the various conversational patterns. The principle is to parse the grammar in the inputs given by the user and match the patterns in order to return the responses.

Communication between students and the college office or department is through the staff who are available only at office hours. The chatbot seeks to tackle that issue. The bot will be used to have normal conversations along with providing college-related information. The bot uses pattern matching and NLP’s to be able to process the natural language inputs and understand the logic. It then formulates replies based on the matched sequences.

The chatbot resolves queries posed by students such as the location of buildings in the campus, teachers’ schedules, timetables, event notices, information about the staff, and frequently asked questions. The responses to the queries will be according to the data available in the knowledge base. Important keywords will be extracted from the sentences and their respective matches will be searched.

If a match is found, the relevant reply is returned to the user. In cases where the information is not available locally, the chatbot will scrape information from the internet using BeautifulSoup and extract the relevant information and return the corresponding results. The chatbot has a text-based user interface which simulates the feeling of messaging a person. The chatbot facilitates access to the information related to the college to the students from anywhere. This reduces the need for the staff to answer the same queries to every consecutive student and does not confine information access to office hours.

II. SOFTWARE REQUIREMENTS
The chatbot uses the following

A. AIML (Artificial Intelligent Markup Language)\(^1\)
AIML is an XML based markup language which is used to create artificially intelligent chatbots. AIML makes it possible to simulate human conversation while keeping the implementation simple, easy to understand and maintain.

B. Python\(^2\)
Python is a high-level, interpreted, interactive and object-oriented programming language. Python provides a dynamic type system with automatic memory management. It supports object-oriented, functional and procedural paradigms. Python also provides support through a large and comprehensive library.

C. Telegram\(^3\)
This API provides facility to create and connect bots to telegram. The telegram API allows building of customized Telegram clients. It is also open for developers who wish to create similar applications.

D. BeautifulSoup\(^4\)
BeautifulSoup is a Python library for parsing HTML and XML documents. It creates a parse tree for parsed pages that can be used to extract data from HTML, which is useful for web scraping.
III. DESIGN

A. Abstract Design

The first step in the development of any product or system is its design. It is a process of transforming the user requirements into a form which helps the programmer in the implementation of the product. Here the user is the one who interacts with the bot.

He can perform interactions such as chat with the bot and query the bot for information. The bot provides responses to the user. The response will vary according to the input. To reply to chat messages the bot will parse the user input message for keywords and match them to the patterns present in the brain file. Once the match is found, the response message will be generated. If the user requests for a web search, then the bot performs the look up and then responds to the user’s Android application with a link to the search result.

The architectural diagram is another representation of the abstract design in which the primary concern is to describe the top level structure and components of the system.

B. Functional Design

Functional Design is a paradigm that is used to simplify the design of a computer software. It can be represented using modular design diagrams, sequence diagrams, class diagrams and data flow diagrams.

Modular design is an approach that breaks down a system into smaller constituents called modules that can be independently created and used in various systems. A modular system is characterized by partitioning the system into discrete, scalable and reusable modules.
A sequence diagram shows interactions among objects arranged in a time sequence. It shows components like classes involved in the scenario and the sequence in which the messages are sent and received to perform the functionality. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development.

A class diagram describes the structure of a system by showcasing the classes of the system, their operations, attributes and the relationships among objects. The class diagram can be considered as the main building block of object-oriented modeling design.
A data flow diagram provides the flow of information for a system or process. It uses standard symbols like rectangles, arrows, and circles along with short text labels in order to show data inputs, outputs. Data flow diagrams can range from simple to in-depth, multi-level diagrams. The multi-level diagrams dig deeper into how the data is handled. These diagrams can be used to analyze an existing system or to model a new one.

C. Control Flow Design
Activity diagrams are graphical representations which describe the dynamic aspects of the program. Activity diagrams model both organizational and computational processes.

It represents the flow from one activity to another. It also shows the branched, concurrent and parallel flow of the system.

IV. IMPLEMENTATION

A. Pseudo Code
Pseudocode is an informal description that does not need any programming language syntax. It is used to create an outline or a draft of a program. Pseudocode is used to summarize a program's flow without the underlying details.

1.1 Pseudo Code for User Registration:

Begin
Input Name, Age, DOB, Gender, Email, Password, Mobile No, Sem, Dept
If username is empty
Display error message
Else check if username is valid
If username is invalid
Display error message
Else
Display registration successful
End

1.2 Pseudo Code for User Login:

Begin
Input Username, Password
If username or password is empty
Display error message
Else check if username and password is valid
If username and password is invalid
Display error message
Else
Display chat window
End

1.3 Pseudo Code for Server:

Begin
Check for updates from bot
If updates present
Retrieve user message and chatId
Else repeat
Check for bot commands
If bot command in message
If command == “/start”
Show welcome message
Else if command == “/load aiml”
Reload aiml files
Else
Repeat
1.4 Pseudo Code for App

Begin
Display login screen
Generate user chat Id
If internet connection not available
    Show error message
Else
    Show connected message
Prompt user for message
Read user chat message
Send Message to the bot with the chatId
Wait for reply
Read the reply object
Retrieve the chat message
Display the message
Repeat
End

CONCLUSION AND FUTURE SCOPE

In present day scenario, smartphones are used to perform more tasks than the people as well as computers. The chatbot application is developed by observing the need for a college assistant that is available at all times and accessible through an Android application from anywhere.

The flow of information between the administrator and the end user is necessary and this can be easily done through our application. Easy access to information regarding the college, the CSE dept., the professor’s schedules, contact details, course-related questions, class notes, events and student time-tables to the users at anytime and anywhere through our application.

The application can be made available to everyone and a reliable server can be chosen to accommodate heavy traffic and allow scalability in the future. New functionalities such as audio/video facilities can be provided to the user along with expanding the information data of other departments. It can have event notices which will have a time limit of the deadline. Once the deadline is passed the notice expires. It can have a 3D model of the college to navigate around the campus in 360 degrees.

REFERENCES
[2] https://docs.python.org/
[3] https://core.telegram.org/bots/api
[5] https://www.definitions.net/definition/Functional%20design