EDUCATIONAL CHATBOT

Dr P. Sreenivasulu[1], B. Reddaiah Reddy[2]

Professor[1], Student[2]Electronics and Communications Engineering

Audisankara College of Engineering &Technology, Gudur, Tirupati, A.P, India.

ABSTRACT

Educational sectors are being transformed into digitalization to prove their significance in their domain. Chatbots play a major role in the educational sector in terms of saving time and effort. Chatbots are being used in nearly all manufacturing areas to provide assistance. They are becoming increasingly popular in industries such as telecommunications, healthcare, and education. Our purpose in creating this chatbot is to reduce the effort for the people working in the educational domain and We found that educational chatbots vary from the basic level of bots which are useful for personalized messages to answer the learning content desired by the users. Results show that chatbots which are part of these applications are working rapidly to become artificial intelligence chatbot assistants. In this work, we develop a chatbot for the Educational sector which can be able to answer outcomes needful to the user.

1.INTRODUCTION

Chatbots are trending and they can now be found in almost every industry from e-commerce to travel. The increased use of bots may be due to improved language processing or the more accessible development tools for non-developers. It may also be that many chatbots are made available through mainstream messaging applications, thus not forcing the user to download yet another application and allowing them to keep using an application they are already comfortable with. Chatbots were initially programmed to be entertaining and to mimic human interaction. This is still a common explanation for designing chatbots, but as the technology';s adoption has grown, so have the variety of applications. The chatbot platform has also been used for a variety of purposes, things like retrieving information, answering questions, and assisting in the making of fact-based decisions. Among other items, you might work as a shopping assistant, a museum guide, a language partner, or in education. A chatbot can be used to retrieve information and answer a user's questions about a given subject. This kind of chatbot can be used for a variety of purposes, including immersive FAQs and assisting customers in making decisions.

2.LITERATURE SURVEY

A literature survey or literature review means study of references papers and old algorithms that we have read for designing the proposed methods. It also helps in reporting summarization of all the old references papers, their drawbacks.The detailed literature survey for the project helps in comparing and contrasting various methods, algorithms in various ways that have implemented in the research.

[1] M. J. Pereira, L. Coheur, P. Fialho, and R. Ribeiro, "Chatbots' greetings to human-computer communication," arXiv preprint arXiv:1609.06479, 2016.

Both dialogue systems and chatbots aim at putting into action communication between humans and computers. However, instead of focusing on sophisticated techniques to perform natural language understanding, as the former usually do, chatbots seek to mimic conversation. Since Eliza, the first chatbot ever, developed in 1966, there were many interesting ideas explored by the chatbots' community. Actually, more than just ideas, some chatbots' developers also provide free resources, including tools and large-scale corpora. It is our opinion that this know-how and materials should not be neglected, as they might be put to use in the human-computer communication field (and some authors already do it). Thus, in this paper we present a historical overview of the chatbots' developments, we review what we consider to be the main contributions of this community,

and we point to some possible ways of coupling these with current work in the human-computer communication research line.

[2] O. Deryugina, "Chatterbots," Scientific and Technical Information Processing, vol. 37, no. 2, pp. 143–147, 2010.

A survey of chatterbots is presented along with their brief background including their emergence, description of the first programs of this type and some of their modern commercial and educational applications. A Chatterbot is a computer program that is designed to simulate an intelligent conversation with one or more human users. To put it more simply, this is a program that, ideally, "communicates" with a person on an equal footing; it somehow analyzes human cue words or phrases and gives appropriate reasonable current consideration; further it will be assumed that the programs we deal with have a character interface.

[3] J. S. Malik, P. Goyal, and A. K. Sharma, "A comprehensive approach towards data preprocessing techniques & association rules," in Proceedings of The4th National Conference, 2010.

Data preprocessing is an important and critical step in the data mining process and it has a huge impact on the success of a data mining project.[1](3) Data preprocessing is a step of the Knowledge discovery in databases (KDD) process that reduces the complexity of the data and offers better conditions to subsequent analysis. Through this the nature of the data is better understood and the data analysis is performed more accurately and efficiently. Data pre-processing is challenging as it involves extensive manual effort and time in developing the data operation scripts. There are a number of different tools and methods used for preprocessing, including: sampling, which selects a representative subset from a large population of data; transformation, which manipulates raw data to produce a single input; denoising, which removes noise from data; normalization, which organizes data for more efficient access; and feature extraction, which pulls out specific data that is significant in some particular context. Pre-processing technique is also useful for association rules. Like

A prior, Partitioned, Pincer -search algo. and many more algos.

[4] B. A. Shawar and E. Atwell, "Chatbots: are they really useful?" in LDVForum, vol. 22, no. 1, 2007, pp. 29–49.

Chatbots are computer programs that interact with users using natural languages. This technology started in the 1960's; the aim was to see if chatbot systems could fool users that they were real humans. However, chatbot systems are not only built to mimic human conversation, and entertain users. In this paper, we investigate other applications where chatbots could be useful such as education, information retrieval, business, and e- commerce. A range of chatbots with useful applications, including several based on the ALICE/AIML architecture, are presented in this paper.

3.SYSTEMS ANALYSIS

3.1. EXISTING SYSTEM

Traditional interaction basically uses high resources to estimate metrics like service requirements, budgets, and performance. Traditional practices fail because it needs resources like people to be at the office to answer the questions. While in the case of bots, time is not a matter for the users who want to make an inquiry. This creates a lot of constraints regarding educational things, universities, and learning courses offered by them.

3.1.1 DISADVANTAGES

- ➢ Low efficiency.
- ➢ Time consuming.
- ➢ High complexities.
- ➢ Resources consuming

3.2 PROPOSED SYSTEM

We propose this application that can be considered a useful system since it helps to reduce the limitations obtained by traditional methods. By providing support through the chat bots, it can be able to generate best results for attributes without any overlap. The system is able to reply to us very efficiently. The system is developed in a Flask based Python environment which uses NLP. MySQL is used for database management and the models involved.

3.2.1 ADVANTAGES

- High efficiency.
- Time Saving.
- Low complexities.

3.3 HARDWARE & SOFTWARE REQUIREMENTS

This guide outlines minimum software and hardware requirements for deploying Mattermost. Requirements may

vary based on utilization and observing performance of pilot projects is recommended prior to scale out.

3.3.1 HARDWARE CONFIGURATION

- Processor - I3/Intel Processor
- RAM - 4GB (min)
- Hard Disk - 128 GB

3.3.2 SOFTWARE CONFIGURATION

- Operating System : Windows 7+
- Server-side Script : Python 3.6+
- IDE : PyCharm
- Libraries Used : Pandas, Numpy, Sklearn
- Framework : Flask
- Database : MySQL

4.SYSTEM DESIGN

4.1 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML comprises two major components: A Meta- model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software systems, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development processes.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of the OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

USE CASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

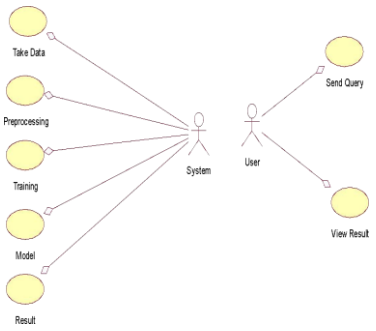


Fig : 4.1 Use Case Diagram

CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods),

and the relationships among the classes. It explains which class contains information.

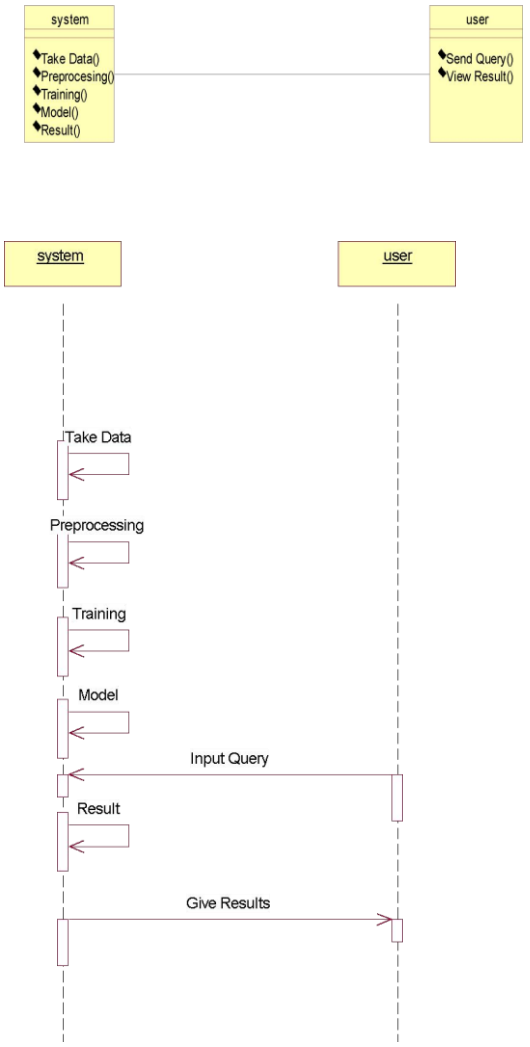


Fig : 4.3 Sequence Diagram

COLLABORATION DIAGRAM

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

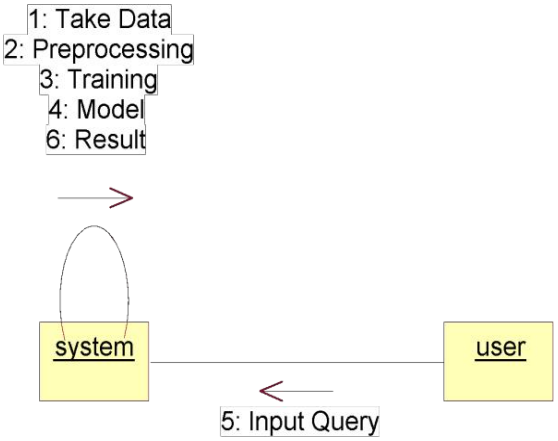


Fig : 4.4 Collaboration Diagram

DEPLOYMENT DIAGRAM

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

Fig : 4.5 Deployment Diagram

ACTIVITY DIAGRAM

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

Fig : 4.6 Activity Diagram

COMPONENT DIAGRAM

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

Fig : 4.7 Component Diagram

ER DIAGRAM

An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database.

The main components of the E-R model are: entity set and relationship set.

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in a database, so by showing relationships among tables and their attributes, ER diagrams show the complete logical structure of a database. Let's have a look at a simple ER diagram to understand this concept.

DFD DIAGRAM

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

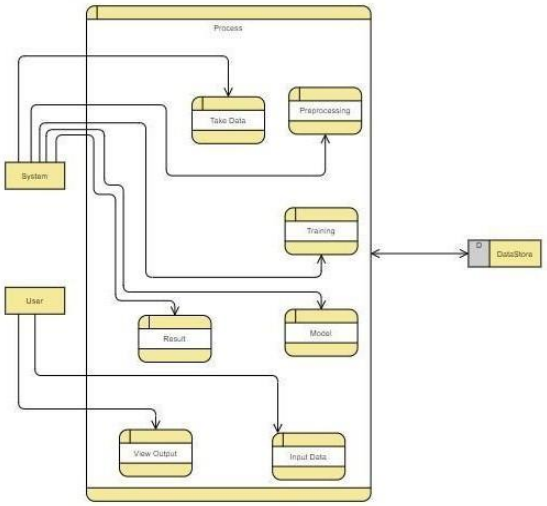


Fig : 4.8 DFD Diagram

5. IMPLEMENTATION

5.1 MODULES

System :

Take Data

System will receive data from the user to collect the information and process it to show according to the user search in the database of the exported links through meta data.

Preprocessing

The system will undergo preprocessing state in which the searched data will be verified through the database in the mentioned informatory where it can be later used to refer.

Training

The System will get trained though the meticulous data in the form of searches given by the user to extract data from the database though chatbot in the cloud.

Model

The system will work based on the model that are included in the main stream of the application that is provided in chatbot for answering the query of user defined.

Results

The system will deliver the output to the user for what the user is trying to find out using this type of service in the derived way of format that will show the information to user.

User :

Send Query

User will send a Query to the system in order to find the solution according to his need and information gathering through the portal using the services of chatbot on the platform.

View Query Result

User will view query results in the output screen of the website using any kind of device.

6 .TESTING

6.1 SOFTWARE TESTING

Software testing is an investigation conducted to provide stakeholders with information about the quality of the software product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include the process of executing a program or application with the intent of finding failures, and verifying

that the software product is fit for use.

Software testing involves the execution of a software component or system component to evaluate one or more properties of interest. In general, these properties indicate the extent to which the component or system under test:

- meets the requirements that guided its design and development,
- responds correctly to all kinds of inputs,
- performs its functions within an acceptable time,
- is sufficiently usable,
- can be installed and run in its intended environments
- achieves the general result its stakeholders desire.

6.2 TESTING TECHNIQUES

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

TESTING METHODOLOGIES

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 satisfactory, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

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 identifying 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.

6.3 TESTING STRATEGIES

Unit Testing 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 test 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 is usually conducted as part of a combined code and unit test phase of the software life cycle, 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.

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

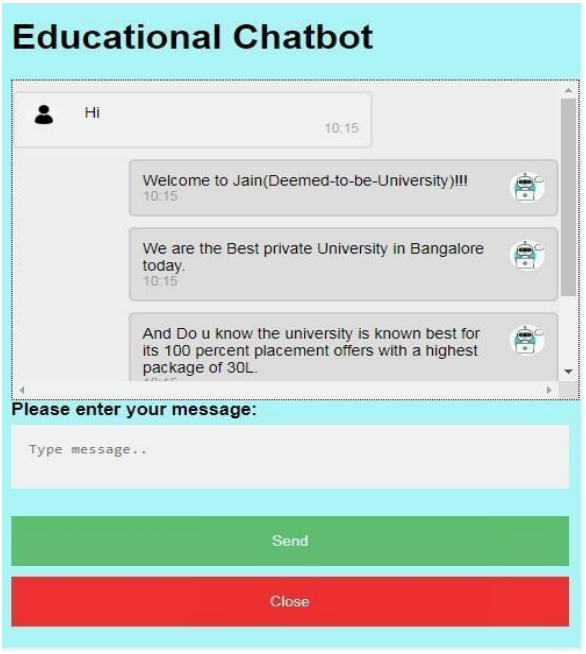
User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results

ll the test cases mentioned above passed successfully. No defects encountered.

7.RESULTS

This system works fine and the bot answers the queries of the users. some humour has been added into the bot database to make it less boring to interact with chat bot as it is designed conceptually and offers a seamless omnipresent counselling experience. Naturally, a chat bot is faster than humans as it does not require time to think and then type. Everything else is working properly. As an expert multi-tasker, it can engage with a number of prospects at the same time. Its proactive approach encourages greater interaction, providing more chances of conversion as compared to live chat, that requires a human to actively manage it.



8.REFERENCES

[1] M. J. Pereira, L. Coheur, P. Fialho, and R. Ribeiro, “Chatbots’ greetings to human- computer communication,” arXiv preprint arXiv:1609.06479, 2016.

[2] O. Deryugina, “Chatterbots,” Scientific and Technical Information Processing, vol. 37, no. 2, pp. 143–147, 2010

J. S. Malik, P. Goyal, and A. K. Sharma, “A comprehensive approach towards data preprocessing techniques & association rules,” in Proceedings of The4th National Conference, 2010

[3] B. A. Shawar and E. Atwell, “Chatbots: are they really useful?” in LDV Forum, vol. 22, no. 1, 2007, pp. 29–49.

[4] A. Bordes, S. Chopra, and J. Weston, “Question answering with subgraph embeddings,” arXivpreprint arXiv:1406.3676, 2014.

[5] B. Setiaji and F. W. Wibowo, “Chatbot using a knowledge in database: Human-to-machine conversation modeling,” in Intelligent Systems, Modelling and Simulation (ISMS), 2016 7th International Conference on. IEEE, 2016, pp. 72–77.

[6] H. Wang, Z. Lu, H. Li, and E. Chen, “A dataset for research on short-text conversations.” inEMNLP, 2013, pp. 935–945.

[7] D. Britz, “Deep learning for chatbots, part 1–introduction,” 2017.