Table of Contents

1. Project overview

A chatterbot or chat bot is a computer program designed to simulate an intelligent conversation with one or more human users via auditory or textual methods. Chatbots can be programmed for small talk, or can also serve as a medium of interaction with users, providing them with answers based on regular questions. The chatbot understands context and delivers a response based on the message given to it. Chatbot is one of many examples of AI. Chatbots were initially designed as means of entertainment and some of them have been designed to pass the Turing Test.

This project of chatbot mainly deals with Websites with a large amount of content and poorly structured navigation can make it difficult for user to find the information easily and quickly. In this case, a chatbot to make it easier for the user to find information. The user has an option to chat with the bot and ask it normal questions to get responses.

2. Literature Review

ELIZA was one of the first chatbot and the brain behind it was Joseph Weizenbaum. ELIZA's key method of operation involves the recognition of cue words or phrases in the input, and the output of corresponding pre-prepared or pre-programmed responses that can move the conversation forward in an apparently meaningful way emulate a psychotherapist. More recent notable programs include A.L.I.C.E., Jabberwocky and D.U.D.E. AskJevees is a web-based search engine While ELIZA and PARRY were used exclusively to simulate typed conversation, many chatterbots now include functional features such as games and web searching abilities. Most of the existing virtual agents, also known as the chatbots, are mainly for entertainment and research purpose. Successful and Award winning chatbots like A.L.I.C.E and CleverBot focus on generic responses to entertain the end user.

3. Methodology:

Python language will be used for this project. We use Django framework to develop chatbot for web, Chatterbot package and Natural language tool kit library will be used to take input from user, translate it to machine readable form, process it, look suitable response from datasets and give response to user.

Plan of Action:

1. Using Django python framework for website development.
2. NLTK to process natural language.
3. Implement AI in chatbot.

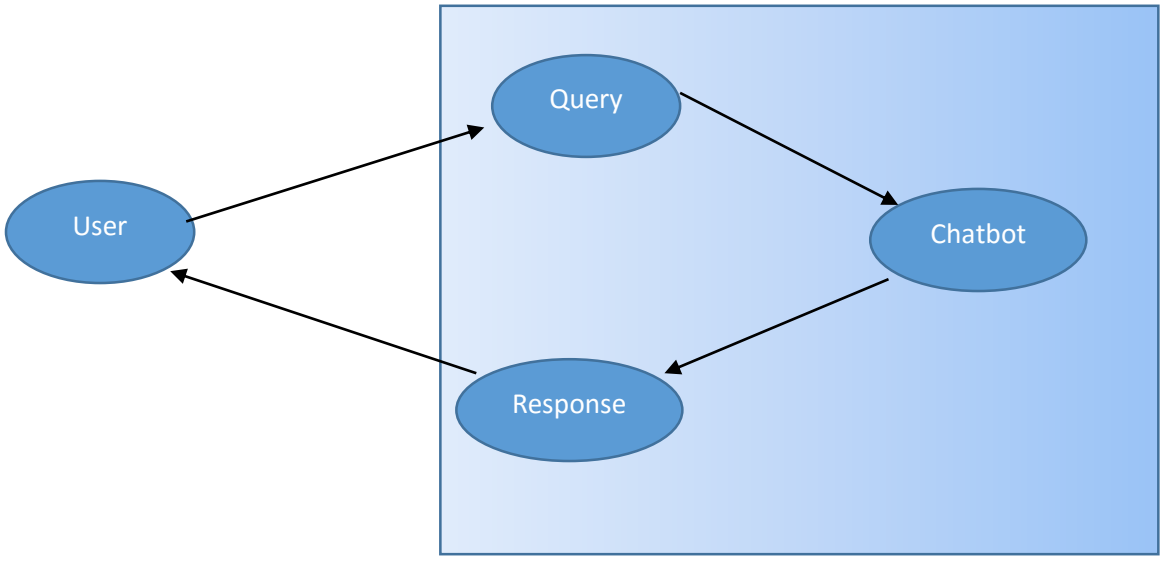


Figure 1 Project Overview

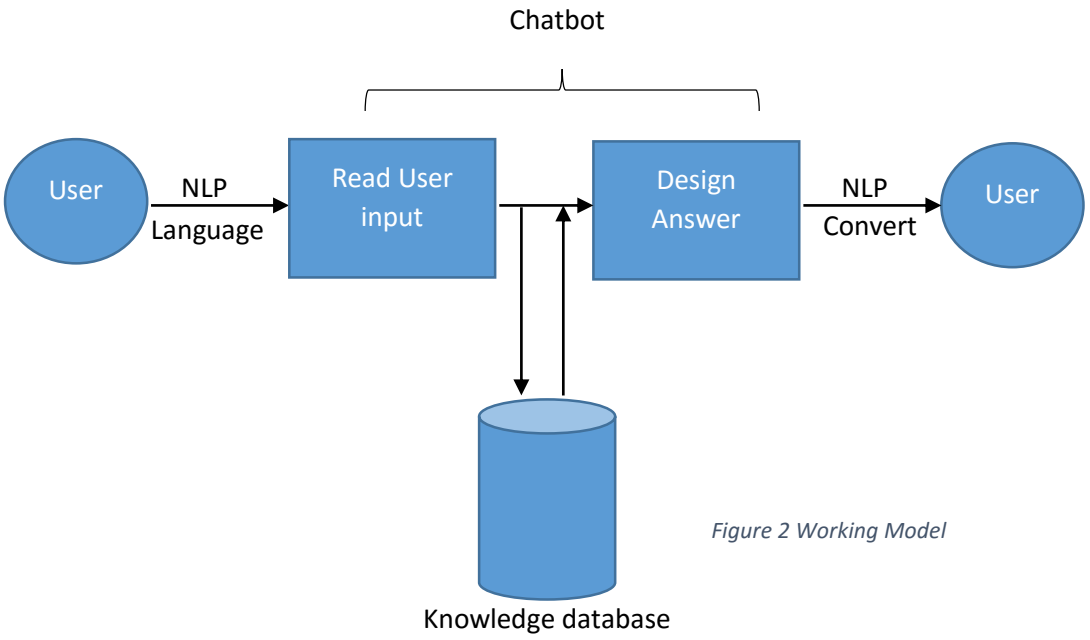


Figure 2 Working Model

2. Implementation

2.1 The User Business or Background of the Project Effort

The purpose of a chatbot program is generally to simulate conversation and entertain the user. More specialized chatbots have been created to assist with particular tasks, such as shopping. The golden standard that the general chatbot tries to achieve is to pass the Turing test, which means to generate conversation which is indistinguishable from that of a real person. State of the art chatbots have not yet reached this goal, which makes this field so interesting to work in.

For this project, the bot has been implemented for 'Paschimanchal Campus website' where the bot has been well trained to give all the possible answers to the related queries that may be arise by any site visitors. The problem of page navigation and not getting the proper answers are solved by this project.

2.2 Goals of the Project

1. To make the web based bot that make it easier for the user to find information easily and fast.
2. To attract the clients on sites by making them feel easy to use the sites.
3. To make the client easy as they will have the guide anytime they make access to the web.
4. Owner don't have to invest extra on hiring the separate employee just to provide the customer service chat 24 hrs.

3. Method for Software Approach:

We used SQLite3 to build our database and used Python and SQL programming languages to access this database. Most of the part of the code is written in Python for backend, only the commands used to interact with the database are in SQL. We installed Django server in Pycharm IDE. Django is a free and open-source web framework, written in Python, which follows the model–view–controller (MVC) architectural pattern. It is maintained by the Django Software Foundation (DSF), an independent organization established as a 501(c) (3) non-profit.

Simple pattern matching technique using regular expressions has been used. We have also used NLTK (Natural Language Tool Kit) for separating parts of speech.

Natural language toolkit:

The NLTK module is a massive tool kit, aimed at helping you with the entire Natural Language Processing (NLP) methodology. It was developed by Steven Bird and Edward Loper in the Department of Computer and Information Science at the University of Pennsylvania. NLTK will aid you with everything from splitting sentences from paragraphs, splitting up words, recognizing the part of speech of those words, highlighting the main subjects, and then even with helping your machine to understand what the text is all about. NLTK is suitable for linguists, engineers, students, educators, researchers, and industry users alike. NLTK is available for Windows, Mac OS X, and Linux. Best of all, NLTK is a free, open source, community-driven project. NLTK has been called “a wonderful tool for teaching, and working in, computational linguistics using Python,” and “an amazing library to play with natural language.”

Django

Django is a free and open source web development framework, written in python, which follows the model-view-controller (MVC) architectural pattern. It is maintained by the Django software foundation (DSF), an independent organization established as a 501(c)(3) non-profit. Django was designed to help developers take applications from concept to completion as quickly as possible. Django includes dozens of extras you can use to handle common Web development tasks. Django takes care of user authentication, content administration, site maps, RSS feeds, and many more tasks right out of the box. Django takes security seriously and helps developers avoid many common security mistakes, such as SQL injection, cross-site scripting, cross-site request forgery and click jacking. Its user authentication system provides a secure

way to manage user accounts and passwords. Some of the busiest sites on the planet use Django ability to quickly and flexibly scale to meet the heaviest traffic demands. Companies, organizations and governments have used Django to build all sorts of things — from content management systems to social networks to scientific computing platforms. Some well-known sites that use Django include Pinterest, Instagram, Mozila, The Washington Times, the Public Broadcasting Service, Disqus, Bitbucket and Nextdoor.

SQLite3:

SQLite3 is a compact free database you can use easily create and use a database. Though SQLite3 is not a full-featured database, it supports a surprisingly large set of the SQL standard, and is ideal for those just starting to learn *SQL* as well for developers that need a simple database engine to plug into their applications. SQLite version 3.0 introduces important changes to the library, including:

- A more compact format for database files.
- Manifest typing and BLOB support.
- Support for both UTF-8 and UTF-16 text.
- User-defined text collating sequences.
- 64-bit ROWIDs.
- Improved Concurrency.

Some benefits of SQLite3 includes:

- Small and self-contained: No additional program or components are required for it to run.
- Portable: It is really to share databases. Just copy one file to do so.
- Standards aware: SQLite is based on standard SQL language. Some features are omitted but those implemented closely adhere to standards. This means what you learn will easily translate to other database engines.

We are concerned with things like maximum number of bytes in a BLOB and maximum number of columns in a table. SQLite3 exerts limitations on databases created by untrusted code to prevent service attack. Macro SQLITE_MAX_LENGTH defines the max number of BLOB or strings in SQLite. The default value of this macro is 1,000,000,000.However, you can change the value by using following command. Max value of a string length can be 2^31-1, it is recommended not to increase the max string length in security sensitive apps. While

processing the INSERT or SELECT command all contents of a row in a table are encoded as a single BLOB. So the max number of bytes in a row can also determine by the above mentioned command. You can lower the BLOB length by using the following command. SQLITE_MAX_COLUMN parameter is used to set upper limit on

- The maximum number of columns in Table, Index or View.
- The maximum numbers of columns by using SELECTS or INSERT statement.
- Number of terms in ORDER BY, GROUP BY, or SET clause.

Default value of a SQLite max column is 2000. However, you can redefine this limit during the compile time. The default value of the maximum number of Bytes in the text of the SQL statement is 1000000, but you can change this limit. It is recommended to prepare short SQL statement and then use SQLite3 bind ABCD () functions to bind the large string values. The maximum numbers of tables to be Join in SQLite is 64. The SQLite uses bitmaps with one bit per join table and there is no way to change this limit. The default of max numbers of arguments in a function is 100. However, you can change this limit at run time by using following interface. The default value of page count limit is 1073741823. When you attempt to insert data after this limit, it will return SQLite _full. The maximum value of SQLITEPAGE_COUNT is 2147483646.The max limit of number of rows in a table is 2^64.

Word-List:

Word-List is a list of word. It is like a string. Word-List has number of elements and each element is varied according to the number or index from zero to number-1. We can add value to the list or can delete from the end of the list. There are several ways to create list, one way is to enclose the element in the square bracket.

4. System requirements:

4.1 Hardware Requirements

Processor	:	Intel Pentium III 1.2 GHz and above
RAM	:	128 MB and above
Hard disk	:	40GB and above
Monitor	:	CRT or LCD monitor
Keyboard	:	Normal or Multimedia
Mouse	:	Compatible mouse

4.2 Software Requirements

Platform	:	Pycharm
Front End	:	HTML, CSS
Language	:	Python
Back End	:	Python, SQLite3
Operation System	:	Windows XP or above, Linux
Browser	:	Any latest browser

5. Requirement Specifications

5.1 Purpose

This system provides users to communicate to the site with the help of a chat bot. Users can just go to the site and ask any queries related to the site which solves the irritating problem of navigating through pages to retrieve the information. The bot can be trained with necessary queries that may be arose by the users whoever visit that site.

5.2 Reference

a. Previously developed bots (ELIZA, ALICE).
b. Survey in communities for the possible queries to train the bot.

6. Specific requirement:

6.1 Functional Specifications

- **Users**

Users just can ask bot the queries related to the site. The problem of page navigation and not getting the required results are all eliminated by this chatbot that is embedded within the website. This helps user to use the site quite comfortably and easily. The queries that are not answered or whose answers are not in the database are forwarded to admin panel and users are made aware through the special message.

- **Admin**

All the answers related to possible questionnaires will be entered into the database through training by the help of admin. The answers related to the queries in the sites are prepared. The answers that have been left out or not being noticed during training will be trained afterword when user asks them. If unknown query then users will be given special message and later on the query will be answered and admin will update database accordingly.

6.2 Non-Functional Specification

- The database is secure and cannot be accessed by unauthorized user. Only admins can have the authority to train the bots and make changes to the information.
- 24 X 7 availability
- Better component design to get better performance at peak time.
- The database used here is robust, reliable & fast. So users will doesn't have to wait for the output very long time.
- This application can be accessed from any type of platform.
- There is no case of redundancy in the database so it will not take extra memory space.

6.3 External Interface Requirement

A. User Interface

After opening the site, the bot is easily noticeable and user can communicate with bot to ask the queries related to the site. Communication is easy and user friendly. User need not visit through the whole site to get the information and just can get the info with the help of chat bot. So chat bot has provided user easy and interactive environment of communicating through the site.

B. Hardware Interface

Since, it is embedded within any web site, it doesn't require any special hardware. So, the hardware that support any sort of latest web browsers can support the application.

C. Software Interface

Any latest web browsers as chrome, internet explorer, Mozilla Firefox etc. can be used for running this application.

D. Communication Protocols

HTTP functions as a request-response in the client-server computing model. A web-browser, for example, may be the client and an application running on a computer hosting a web-site may be the server. The client submits an HTTP request message to the server. The server, which provides resources such as HTML files and other content, or performs other functions on behalf of the client, returns a response message to the client.

Internet Relay Chat Protocol (IRCP) is an application layer protocol that facilitates communication in the form of text. The chat process works on a client/server networking model. IRC clients are computer programs that a user can install on their system. These clients communicate with chat servers to transfer messages to other clients. IRC is mainly designed for group communication in discussion forums, called channels, but also allows one-on-one communication via private messages as well as chat and data transfer, including file sharing.

6.4 Main Attributes

This application includes its user friendliness, fast speed, small size and portability. Since internal changes are not required for updating of basic information in the app, it is highly maintainable. Since can be embedded within web application, it can easily be applied in any platform if they have web browsers.

6.5 Maintainability

The bot embedded within web application allows user to, make it easier for the user to find information. The user has an option to chat with the bot and ask it normal questions to get responses.

The application is developed in such a manner that other features and algorithms can also be added easily to make it more powerful and understandable. The features such as understanding natural language, communication with the bot by speech recognition technique etc. can be added easily and efficiently. Since we have implemented this only in the college website. However, it can be implemented easily in any sites. For example in E-commerce sites which can be very much effective for informing customers about the products which can be very much effective in buying and selling products.

6.6 Transferability Conversion

Since, it is a web based chat application it can easily be transferred from one platform to other. No matter what the platform is, it must have installed any latest web browsers as internet explorer, goggle chrome, Mozilla Firefox etc.

7. Algorithms Implemented

7.1 Testing Algorithm

1. It stores the user's query in a wordlists test word by word, does the required filtering of prepositions, conjunctions, auxiliary verbs, interjections etc.

2. It finds out all the nouns, main verbs, wh question, determiner store them in an array resultwords. If no words is found it adds word ‘bemorespecific’ automatically

3. Every word in resultwords is a keyword. And weight is calculated of every words. (a=1, b=2…. z=26 weight = (log(s,5)), abs (s=s+alphabet * a), a=a*10.

4. It goes to the table containing associations, find out the matching weight, match word_id,sentence_id Return Sentence from sentences table.

5. If no match if found, return default answer and new query will be updated in newquestions table.

7.2 Training Algorithm

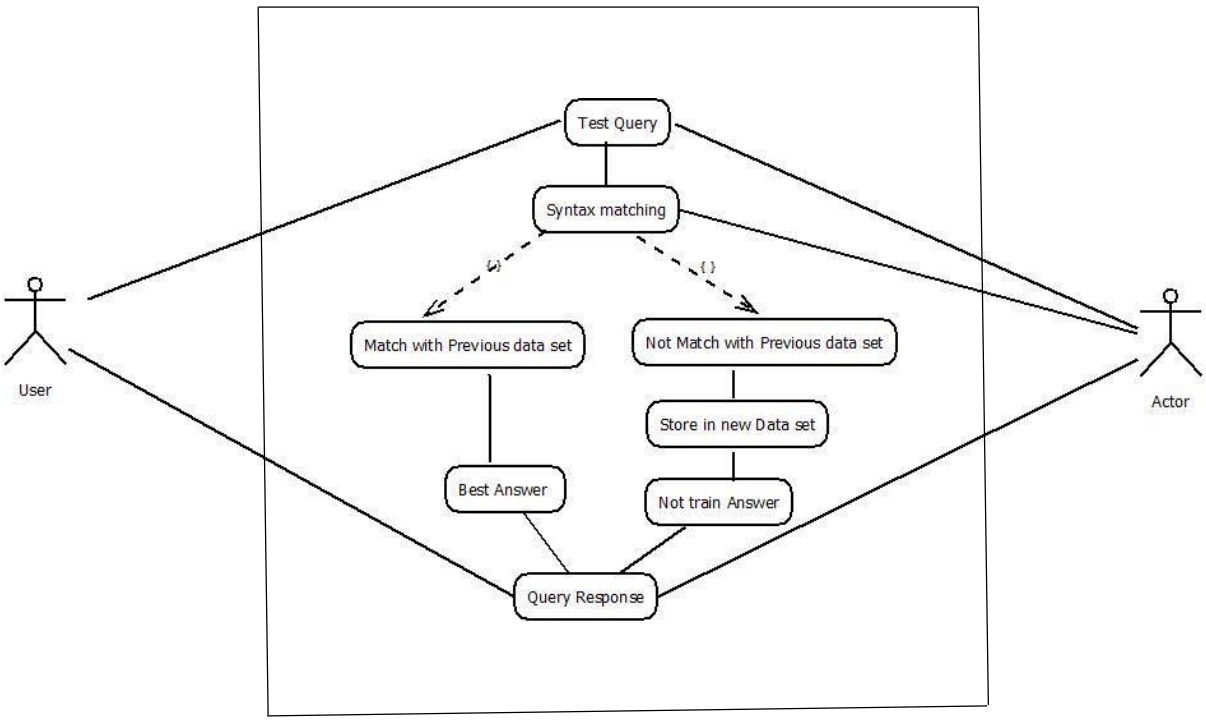
1. Bot starts conversation.

2. Users input is stored in H, sentences are break down into words using regex, words are inserted into table words along with whole sentence.

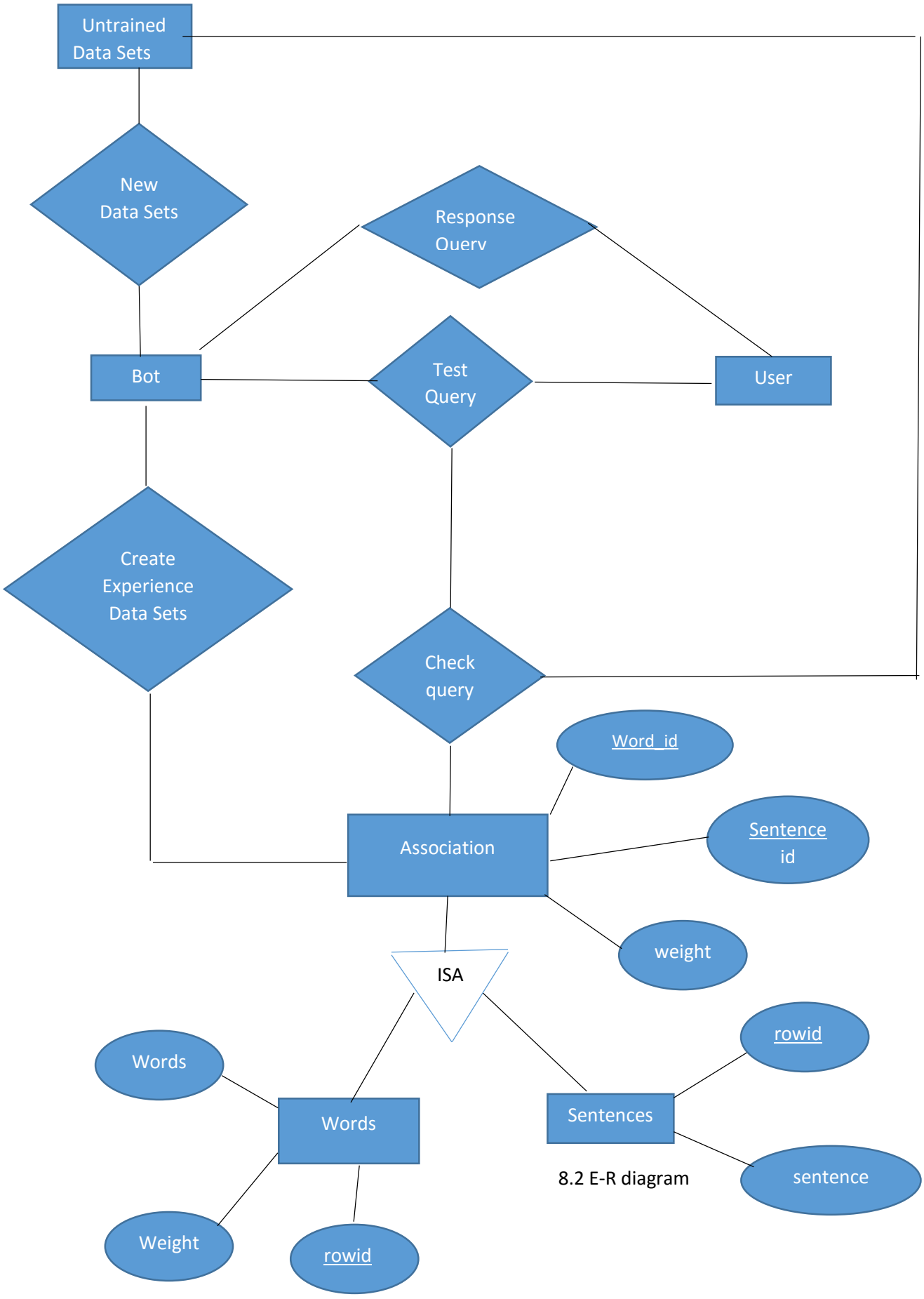
3. With the next input, word_id of previous input and sentence_id of next input are inserted into table association along with weight of word.

8. Diagrams

8.1 Use case diagram

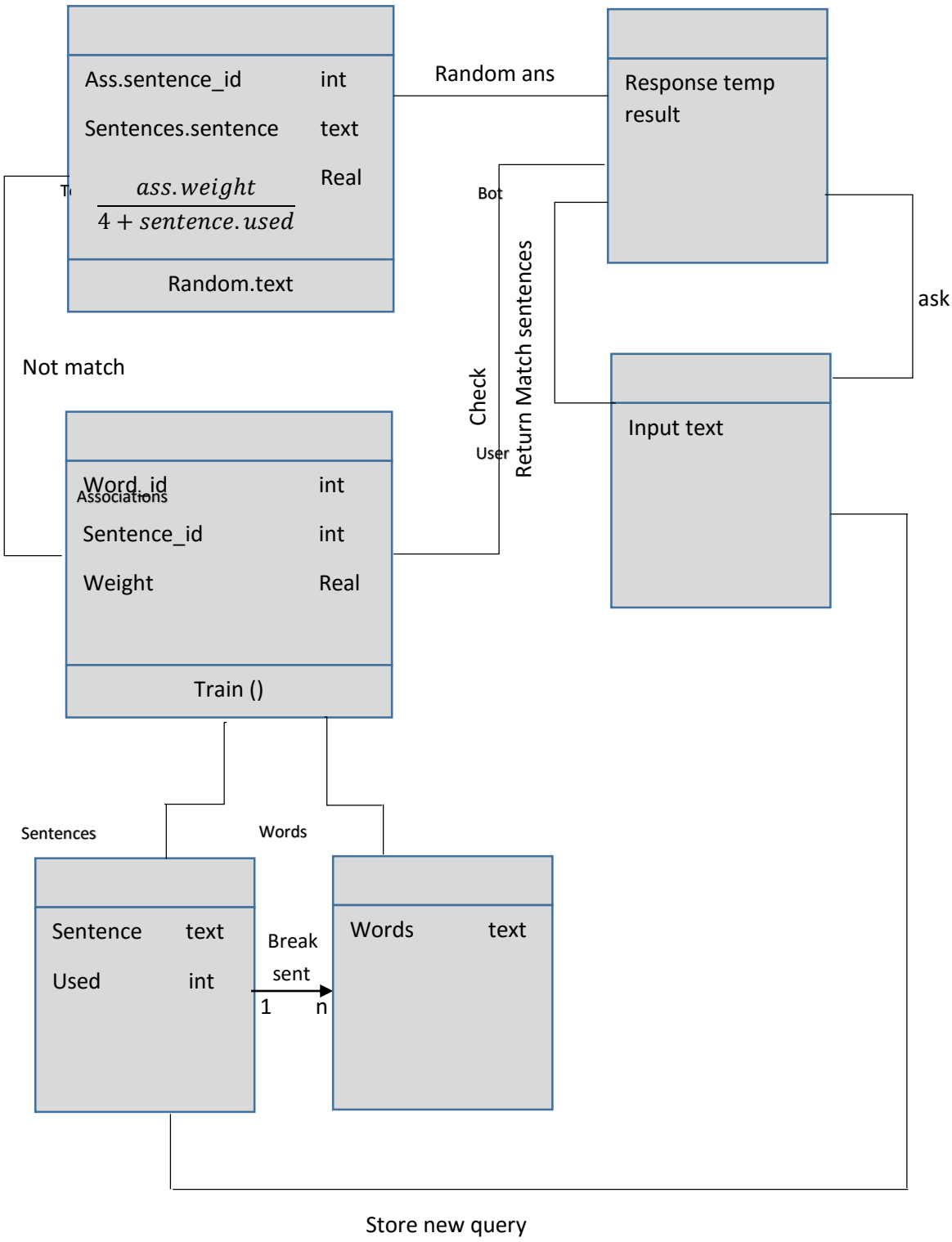


8.2 ER Diagram

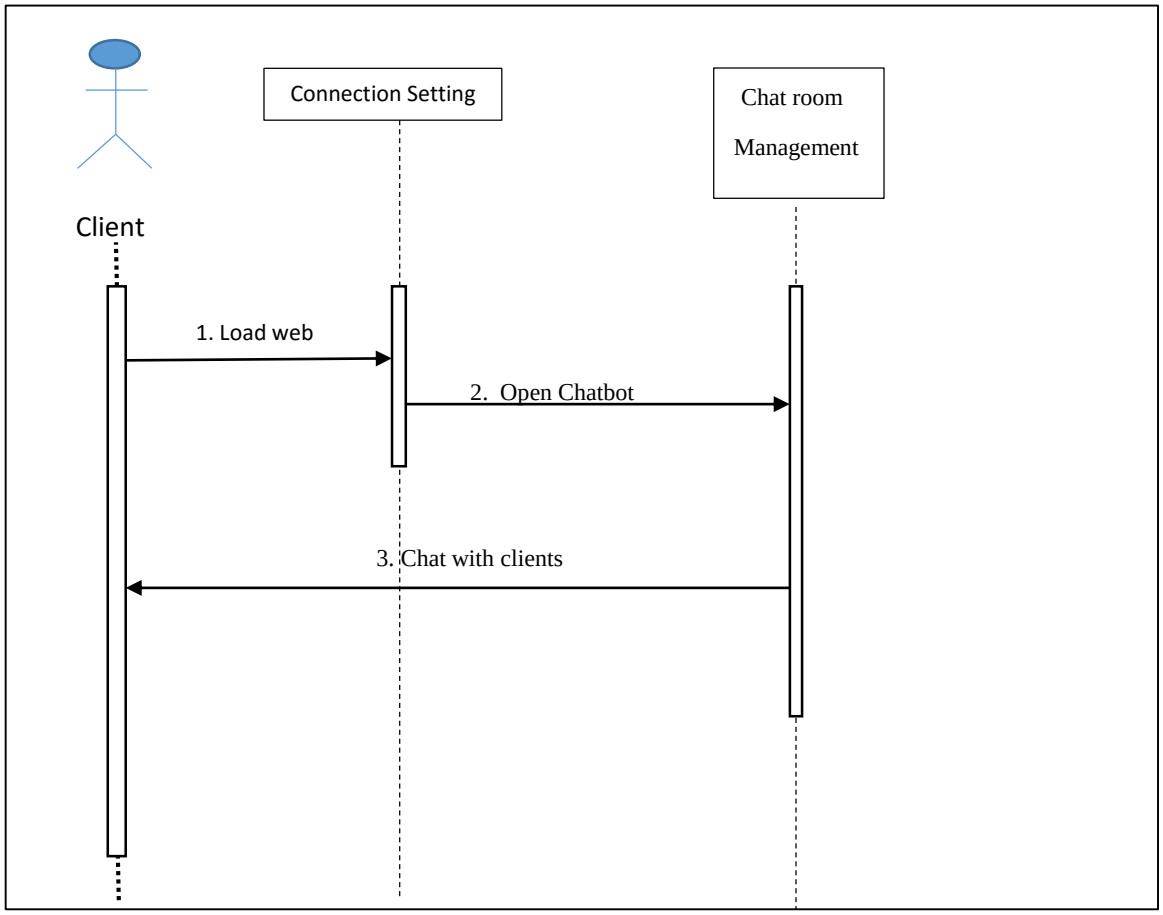


8.2 E-R diagram

8.3 Domain Model



8.4 Sequence Diagram



9.Output

Fig E-BOT

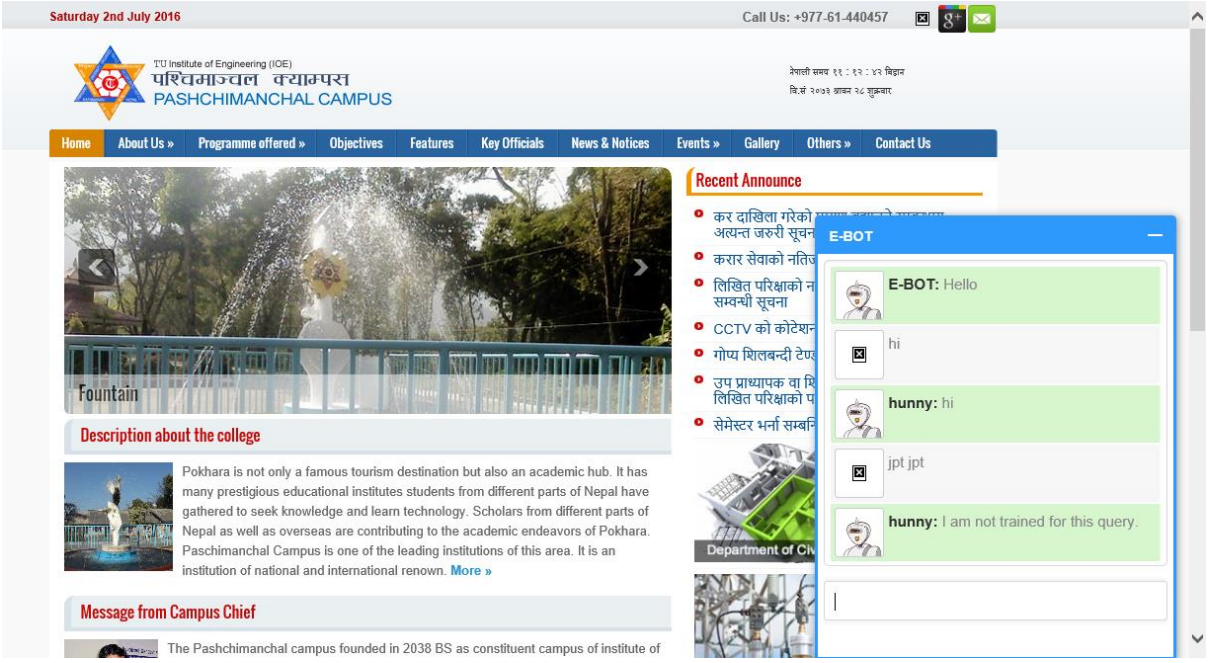


Fig. E-BOT on action

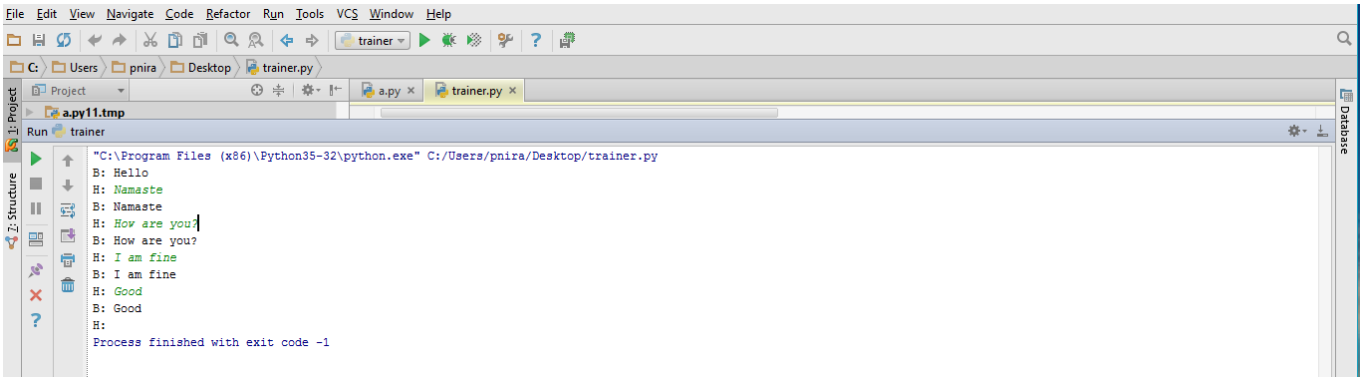


Fig: Training

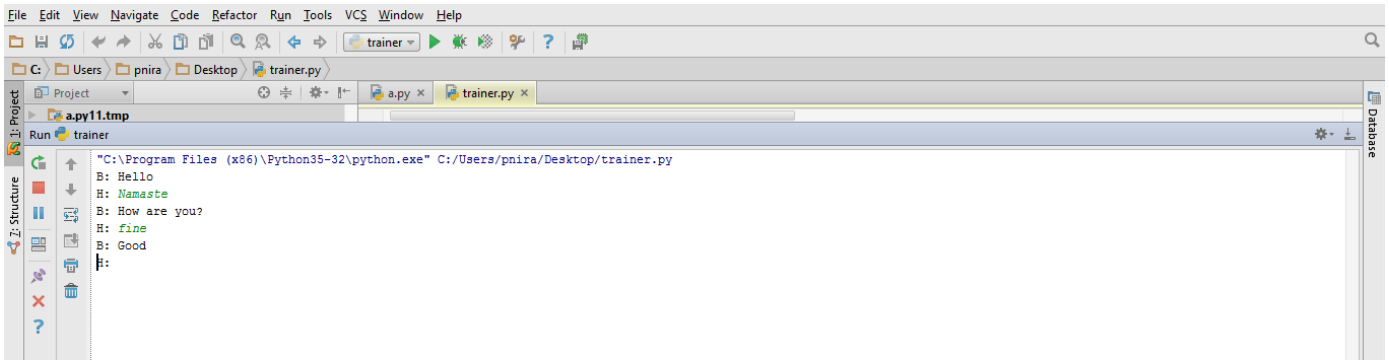


Fig: Result

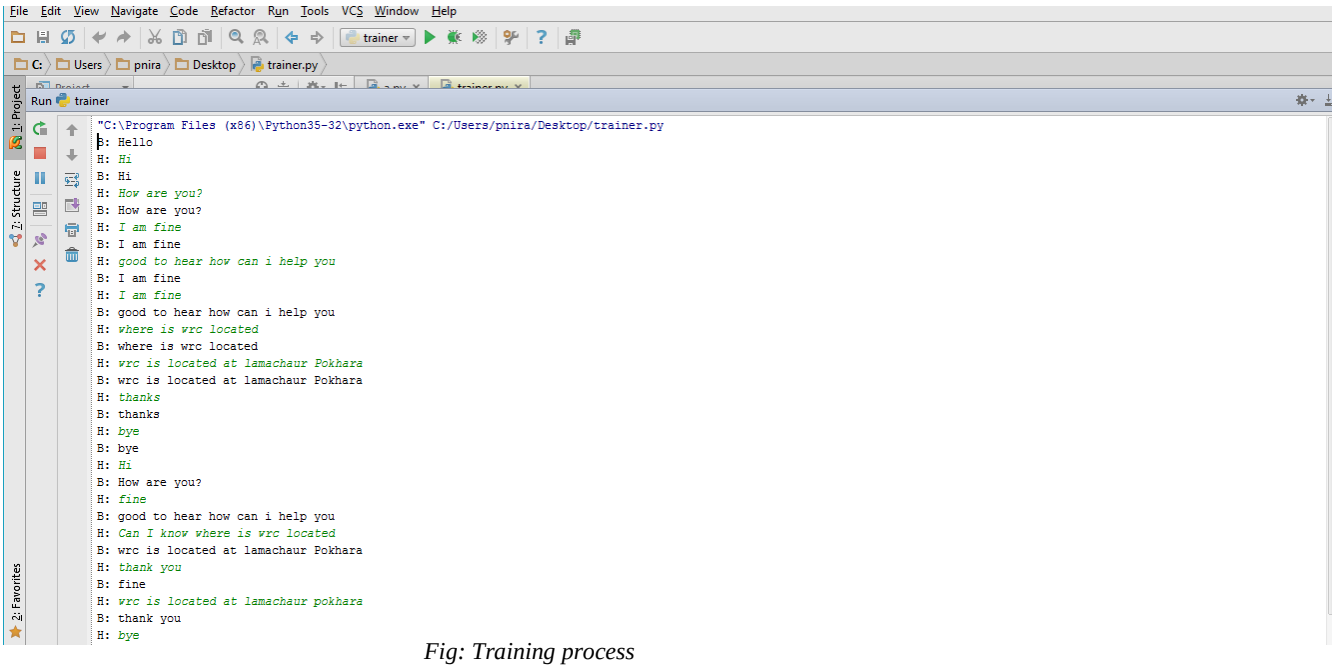


Fig: Training process

10. Result and Evaluation

Quality of any software depends upon its acceptability or sufficiency. The quality of the app includes easy and quick update and extraordinary user interface. We have evaluated that most of the apps though they are very code effective and conceptual have failed for simple reason. Mainly the reason is failing to provide good user interface. Our result has been built keeping in mind that user friendliness and interface attracts the user and makes app successful. We designed this app not only for implementation on the local field but for making the app go international through good architectural design and easy addition of new features possible.

This project of chatbot mainly deals with Websites with a large amount of content and poorly structured navigation can make it difficult for user to find the information easily and quickly. In this case, a chatbot to make it easier for the user to find information. The user has an option to chat with the bot and ask it normal questions to get responses.

The app is productive, effective and efficient for the user as well as developer who is willing to test his/her own capability and creativity. As mentioned above, this app is made in such a way that any new ideas are easy to be added through regression testing mechanism so it is productive and crucial.

10. Testing

- **System Testing**

System testing is a critical element of quality assurance and represents the ultimate review of analysis, design and coding. Test case design focuses on a set of techniques for the creation of test because that meet overall testing objective. When a system is developed it is hoped that it performs properly. The main purpose of testing an information system is to find the errors and correct them. The scope of system testing should include both manual and computerized operations. System testing is comprehensive evaluation of the programs, manual procedures, computer operations and controls.

System testing is the process of checking whether the developed system is working according to the objective and requirement. All testing is to be conducted in accordance to the test conditions specified earlier. This will ensure that the test coverage meets the requirements and that testing is done in a systematic manner.

The process of analyzing the software item to detect the differences between existing or required condition and evaluate the features of the software items. The thorough testing of the system before release of the software needs to be done vide the various test cases and modes so that the software becomes devoid of bugs and uses minimum space requirements as well as minimum time to perform. The test cases were selected beforehand with expected results defined and actual results recorded for comparison. The selection of test cases is done vide "White Box Testing" technique to check the internal programming logic and efficiency and vide "Black Box Testing" technique to check software requirement fulfillment with intension of finding maximum number of errors with minimum effort and time. Although test cases are a design by considering the cyclomatic complexity, conditional test, still the software code is not in its optional form, as all other possible alternative parts in the software are not considered. At the integration level, the software will be passing to the third party tests which would further enhance the software optimality and efficiency.

The quality and standardization of the software / application package depends truly on the various predefined testing norms and on the performances of the software over those norms. There are various standards existing in the software industry the engineered end product strives to achieve viz. ISO 9002 SEI CMM Level5 etc. These standards are achieved only when the concerned software fulfils the tests as per the respective testing norms predefined in them vide the various test cases and parameters using the CASE topologies. Generally, software is tested

both on a stand-alone mode as well after integrating all the modules in the system vide deferent available testing methods/norms.

TEST CHARACTERS:

1. A good test has a high probability of finding an error.
2. A good test is not redundant.
3. A good test should be “best of breed”.
4. A good test should be neither too simple nor too complex.

BLACK BOX TESTING:

The method of Black Box Testing is used by the software engineer to derive the required results of the test cases:

1. Black Box Testing alludes to test that are conducted at the software interface.
2. A Black Box Test examines some fundamental aspect of a system with little regard for the internal logic structure of the software.
3. A limited number of important logical paths can be selected and exercised.
4. Important data structure can be probed for validity.

Black box testing was performed to find errors in the following categories:-

1. Incorrect or missing functions
2. Errors in data in binary format.
3. Error in data in integer format.
4. File error.
5. Memory access error.
6. Variable error .
7. Performance error

WHITE BOX TESTING:

White Box Testing is sometimes called Glass Box Testing. Using White Box Testing methods the software engineer can derive the following test cases:

1. Guarantee that all independent paths within a module have been exercised at least once.
2. Exercise all logical decisions on their true and false sides.

3. Execute all loops at their boundaries and within their operational bounds.
4. Exercise internal data structures to ensure the validity.

In White Box Testing efforts were made to handle the following:-

1. Number of input parameters equal to number of arguments.
2. Parameters and arguments attributes match.
3. Number of arguments transmitted is called modules equal to attributes of parameters.
4. Unit system of argument transmitted is called modules equal unit system of parameter.
5. Number of attributes and order of arguments to build in functions correct.
6. Any references to parameters not associated to build in functions correct.
7. Input only arguments altered.
8. Global variable definition consistent across module.
9. Files attributes correct.
10. Format specifications matches I/O specification.
11. Files opened before use.
12. File closed while working is going on.
13. I/O errors handled.
14. Any textual errors in output information.

UNIT TESTING:

The unit testing is performed to test the validity of the individual units. This is done in the coding phase with the interactive testing. Thus it itself constitutes a majority of functionality test for each logical unit.

INTEGRITY TESTING:

When all the development of all the units or modules is completed and integrated the integrity test phase is started. In this phase the interface between the modules are tested. This phase basically verifies whether inter module exchange of information and events are as per required system behavior.

VALIDATION:

Tests were performed to find conformity with the requirements. Plans and procedures were designed to ensure that all functional requirements are satisfied. The software was alpha-tested. There are two goals in preparing test plans. Firstly, a properly detailed test plan demonstrates that the program specifications are understood completely. Secondly, the test plan is used during program testing to prove the correctness of the program.

11. Quality assurance

Quality assurance is important because poor quality software may be annoying and affect goodwill of the organization. We plan to release the software with description for quality assurance. Certain standards are maintained for developing this web application. Model used and software requirements will be provided to the user with the release of software.

Feedback from any website are most welcome and if user's demands seem feasible and helps to improve the apps, we will work on it as soon as possible. Also any problems while using the app will be greatly appreciated.

12. Conclusion and further work

This application is platform and user friendly. So, user can access this application from anywhere around the world and any person having the basic knowledge of computer can go through the site and make the proper use of bot to retrieve the information. Hence, this project of chatbot mainly deals with Websites with a large amount of content and poorly structured navigation can make it difficult for user to find the information easily and quickly. So, a chatbot could make it easier for the user to find information. The user has an option to chat with the bot and ask it normal questions to get responses.

However, further works yet to be done to make it more dynamic and complete. The features such as understanding natural language, communication with the bot by speech recognition technique etc. can be added easily and efficiently. Since we have implemented this only in the college website. However, it can be implemented easily in any sites. For example in E-commerce sites which can be very much effective for informing customers about the products which can be very much effective in buying and selling products.

Bibliography

nltk. (n.d.). *Natual Language tool kit*. Retrieved from NLTK: http://www.nltk.org

Alice (2002). A.L.I.C.E AI Foundation , http://www.Alicebot.org/

Abu Shawar, B., Atwell, E. (2003a). Using dialogue corpora to train a chatbot in: Archer, D, Rayson, P, Wilson, A & McEnery, T (editors) Proceedings of CL200, pp.681-690

Abu Shawar., B., Atwell, E. (2003b). Machine learning from dialogue corpora to generate chatbots. Expert Update, vol. 6, pp. 25-30.

Chatbot for diabetic patient. http://www.slideshare.net/harshitg3/ chatbot-for-diabetic-patient#. Accessed March 4, 2014

http://www.cleverbot.com/. Accessed March 4, 2014

Mitkov, R., Orasan, C., and Evans, R. The importance of annotated corpora for nlp: the cases of anaphora resolution and clause splitting, 1999.

Wallace, R. The anatomy of a.l.i.c.e. In Parsing the Turing Test, R. Epstein, G. Roberts, and G. Beber, Eds. Springer Netherlands, 2009, pp. 181– 210.

Abney, S. P. (1991). Parsing by chunks. In Berwick, R. C., Abney, S. P., and Tenny, C. (Eds.), Principle-Based Parsing: Computation and Psycholinguistics, pp. 257–278. Kluwer, Dordrecht.

Abney, S. P. (1997). Stochastic attribute-value grammars. Computational Linguistics, 23(4), 597–618.

Abney, S. P., Schapire, R. E., and Singer, Y. (1999). Boosting applied to tagging and PP attachment. In Proceedingsof the 1999 JointSIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC99), College Park, MD, pp. 38–45.

Ades, A. E. and Steedman, M. J. (1982). On the order of words. Linguistics and Philosophy, 4, 517–558.

Adjukiewicz, K. (1935). Die syntaktische Konnexita¨t. Studia Philosophica, 1, 1– 27. English translation “Syntactic Connexion” by H. Weber in McCall, S. (Ed.) Polish Logic, pp. 207–231, Oxford University Press, Oxford, 1967.

Aha, D. W., Kibler, D., and Albert, M. K. (1991). Instance-based learning algorithms. Machine Learning, 6, 37–66.

Aho, A. V., Sethi, R., and Ullman, J. D. (1986). Compilers: Principles, Techniques, and Tools. Addison-Wesley, Reading, MA.

Aho, A. V. and Ullman, J. D. (1972). The Theory of Parsing, Translation, and Compiling, Vol. 1. Prentice-Hall, Englewood Cliffs, NJ.

Algoet, P. H. and Cover, T. M. (1988). A sandwich proof of the Shannon-McMillanBreiman theorem. The Annals of Probability, 16(2), 899–909.

Bobrow, D. G., Kaplan, R. M., Kay, M., Norman, D. A., Thompson, H., and Winograd, T. (1977). Gus, a frame driven dialog system. Artificial Intelligence, 8, 155–173.

Bybee, J. L. (1996). The phonology of the lexicon: evidence from lexical diffusion. In Barlow, M. and Kemmer, S. (Eds.), Usage-based Models of Language.

Bybee, J. L. and Slobin, D. I. (1982). Rules and schemas in the development and use of English past tense. Language, 58, 265–289.

Gazdar, G. and Mellish, C. (1989). Natural Language Processing in LISP. Addison Wesley.

Nida, E. A. (1975). Componential Analysis of Meaning: An Introduction to Semantic Structures. Mouton, The Hague.