AUTO SOURCE CODE GENERATOR

[1]Sagar Kesarkar, [2]Prakash Karpe, [3]Rahul Jadhav, [4]Prof.Diksha Bhave
[1]B.E. Student, [2]B.E. Student, [3]B.E. Student, [4]Professor
Department of Computer Engineering
Shivajirao S. Jondhale College of Engineering
Dombivli East, Thane, Maharashtra 421204

Abstract: *Pseudo code is a man-made and informal language that helps developers to make algorithms. In this paper a software package tool is represented, for translating the pseudo code into a specific supply artificial language. This tool compiles the pseudo code given by the user and interprets it to a supply artificial language. The scope of the tool is very much wide as we can extend it to a universal programming tool which produces any of the specified programming language from a given pseudo code. Here we tend to gift the answer for translating the pseudo code to a artificial language by victimization the various stages of a compiler.*

Keyword*– Compiler, Pseudo code, Source code, Pseudo code Compiler, C, C++, HTML*

I. INTRODUCTION

Generally a compiler is treated as one unit that maps a ASCII text file into a semantically equivalent object program. If we are analyzing a little, we see that there are mainly two stages in this mapping: analysis and synthesis. The analysis phase splits up the source code into sub parts and imposes a grammatical structure on them. It then uses this grammatical structure to make associate intermediate illustration of the ASCII text file. If the analysis section detects that the ASCII text file is either semantically unsound or syntactically weak, then it must provide informative messages to the user. The analysis section collects data concerning the ASCII text file and stores it within the system known as an emblem table, which is passed along with the intermediate representation to the synthesis phase. The synthesis section constructs the object program from the intermediate code and also the data from the image table. The synthesis section is usually known as the rear finish and also the analysis section is that the face of the compiler. Compilation method consists of a sequence of phases, each of which transforms one representation of the source code to another.

II. LITERATURE REVIEW

The following research articles are selected for review, keeping in mind the traditional and conventional approaches of Auto Source Code Generator. This given below information is the observed, segregated and highlighted points from all the base systems whose paper we have used as reference paper for our system.

Our base system was a project by Vishal Parekh, Dwivedi Nilesh in L. D. College of Engineering, Ahmedabad, India (JETIR) and November 2016. They propose a translation process that allows the people to write the algorithmic solutions of the problem in the form of Pseudo code which can further be translated into a programming language like C, C++, and XML. But it wasn't generating proper output in XML. **[1]**

Project by Dipanjan Das, Ryan McDonald in Google Research, New York, USA and Carnegie Mellon University, Pittsburgh, PA, USA, in 2014. Their system was to convert multiple languages into a common intermediate language. But the system was too complex and hard to maintain the big dataset **[2]**

Pseudo Code to Source Programming Language Translator Project by Amal M R, Jamsheed C V, International Journal of Computational Science and Information Technology (IJCSITY), May 2016. They use Pseudo Code compilation technique in their project for effective for simple pseudo code. But system was not working for the complex algorithms and function n, array and structure**. [3]**

Project by Priyanka, Priyanka HL, Priyanka P, Naveen Chandra Gowda, in Asian Journal of Engineering and Technology Innovation (AJETI), 2017. The project is converting the pseudo code-algorithm to a C equivalent program is by inspecting each and every keyword present in the individual lines of an algorithm. But this project generates output in only C language. **[4]**

III. PROPOSED SYSTEM

As we have observed other system they are ready to implement their restricted action as per their specifications they provide only C, C ++ languages code for simple program implementation but our system provides C, C ++ and HTML languages with complex program like all looping techniques and other functions. The Proposed system architecture consists of several modules interacting with each other to accept an algorithm in natural language as pseudo code and interpret it in Programming language. The system accepts pseudo code as an input from the user, via a desktop application which will be stored in the file.

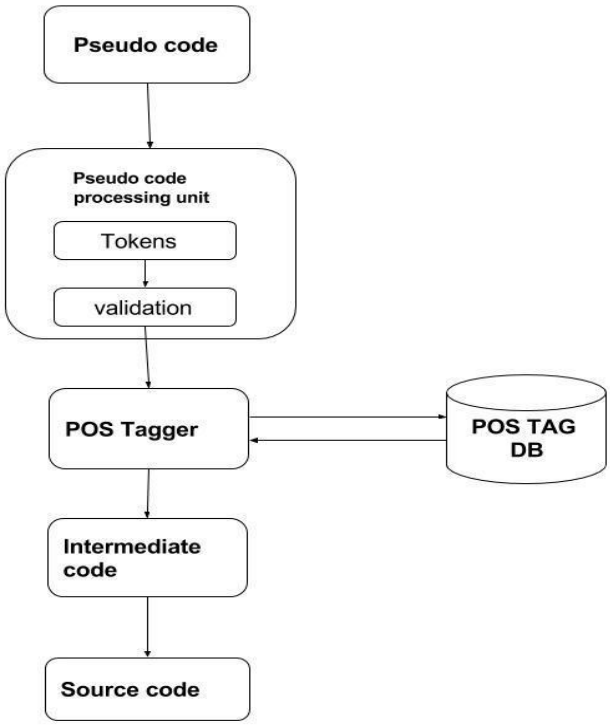


Fig 1- Architecture diagram of the system

After accepting the pseudo code from the user, the system will validate the data in the file and then by applying some basic natural language processing algorithms, it will generate the tokens. Each token will be mapped by the dataset stored into the Part-of-speech [POS] Database and an intermediate file will be generated then system will check whether the code in the intermediate file is semantically correct and will generate the final source code.

System flow

a) SELECTING FORMAT:-

In this part of process the user using the system decides either of the three formats from C, C ++ or HTML in which he needs the final output of his generated code, so by selecting this at the very start the system know what sort of output he is supposed to generate and starts working accordingly from the very start.

b) FEEDING INPUT:-

After selecting the format the next part is to feed the input to the system. User will provide the input as pseudo code. Pseudo code is notation resembling a simplified programming language, used in program design

c) GENERATING OUTPUT

After receiving input from the user system will start to generate output. Depending on the format of language (either C, C++ or HTML) which user selected at the start of the process, system will generate output. After output is generated it will be displayed on the user interface.

IV. COMPONENT

The front end of the application will be constructed using PYTHON. It is mostly used to develop high end software and embedded system which can be used in most of the known environment. To maintain that large amount of data to process the application from Backend needs database, which will be provide by MySQL.

SYSTEM REQUIREMENT

An application is designed to be compatible with all android OS based devices and IOS devices.

a) Software requirements:

- Windows XP, 7, 8.1 or higher
- Python
- Database
- Framework Tkinter from Python

b) Hardware Requirements:

- Processor: Intel Dual Core or higher
- Hard disc space: 5Gb minimum
- MEMORY: 1Gb RAM
- IOS device with 10 or higher

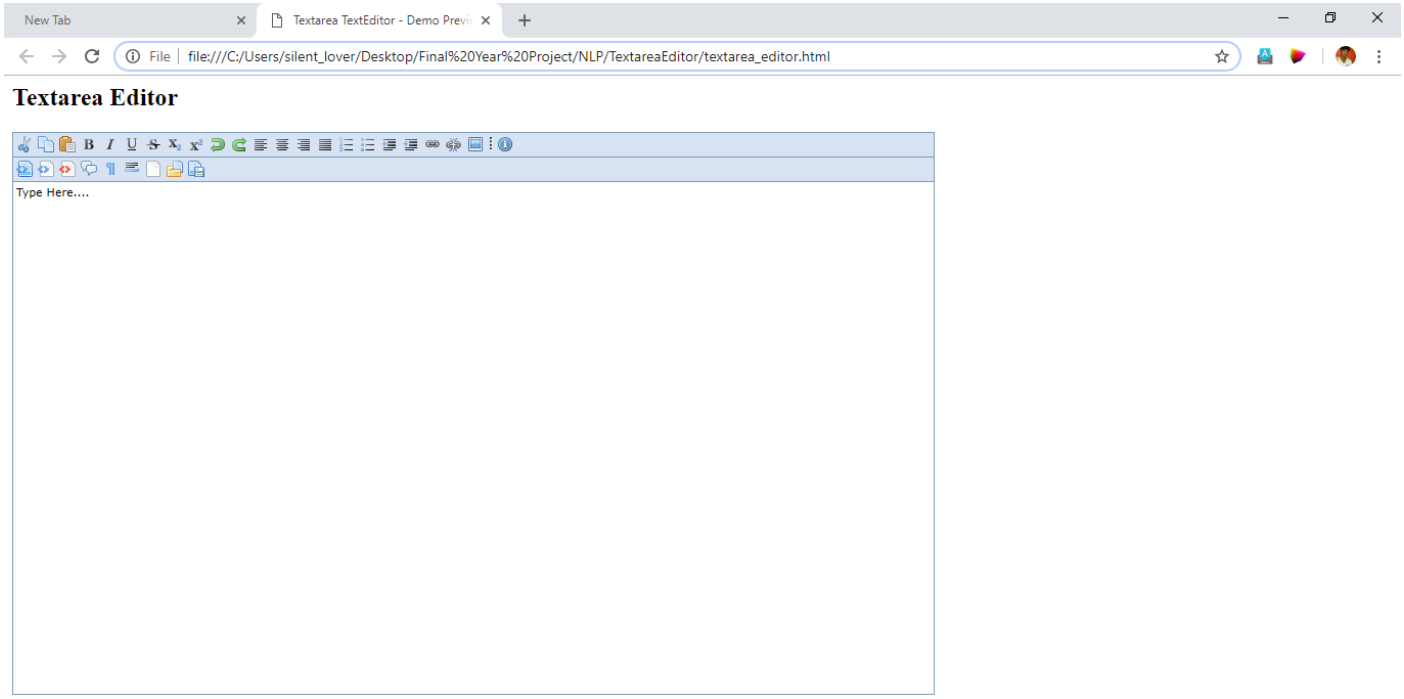
V. MAINTENANCE

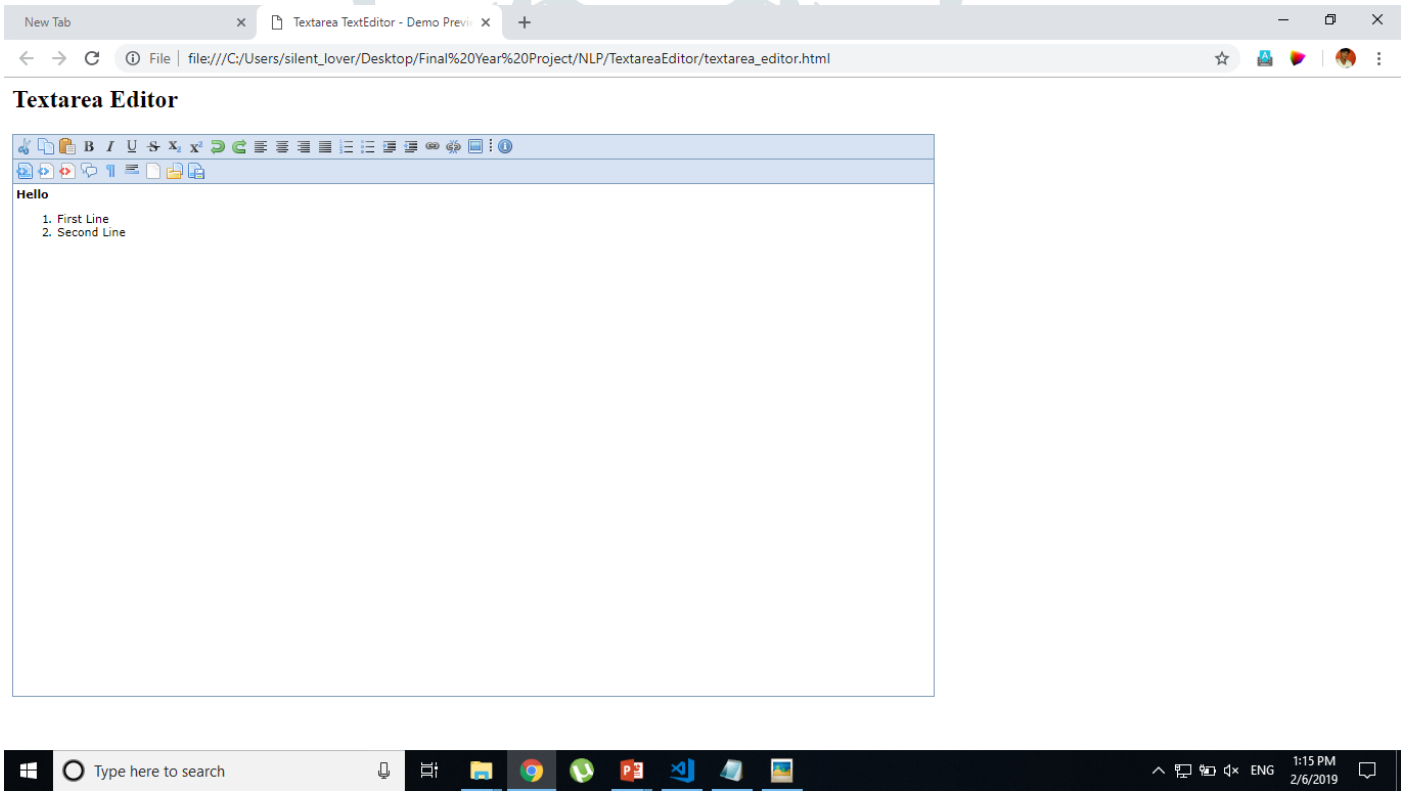
Maintenance is the phase that brings more revenue to the system developers then the development phase. As time goes by or based on the level or size of the society there might occur a situation where the system has much more amount of user then it can usually handle and might lead to a crash down of the system. So for such thing to be avoided the systems will be in a need of maintainability after a certain finite amount of time. The task of maintenance in generally divided on basis of the module that might have a certain issue, but basically it is divided into hardware and software.

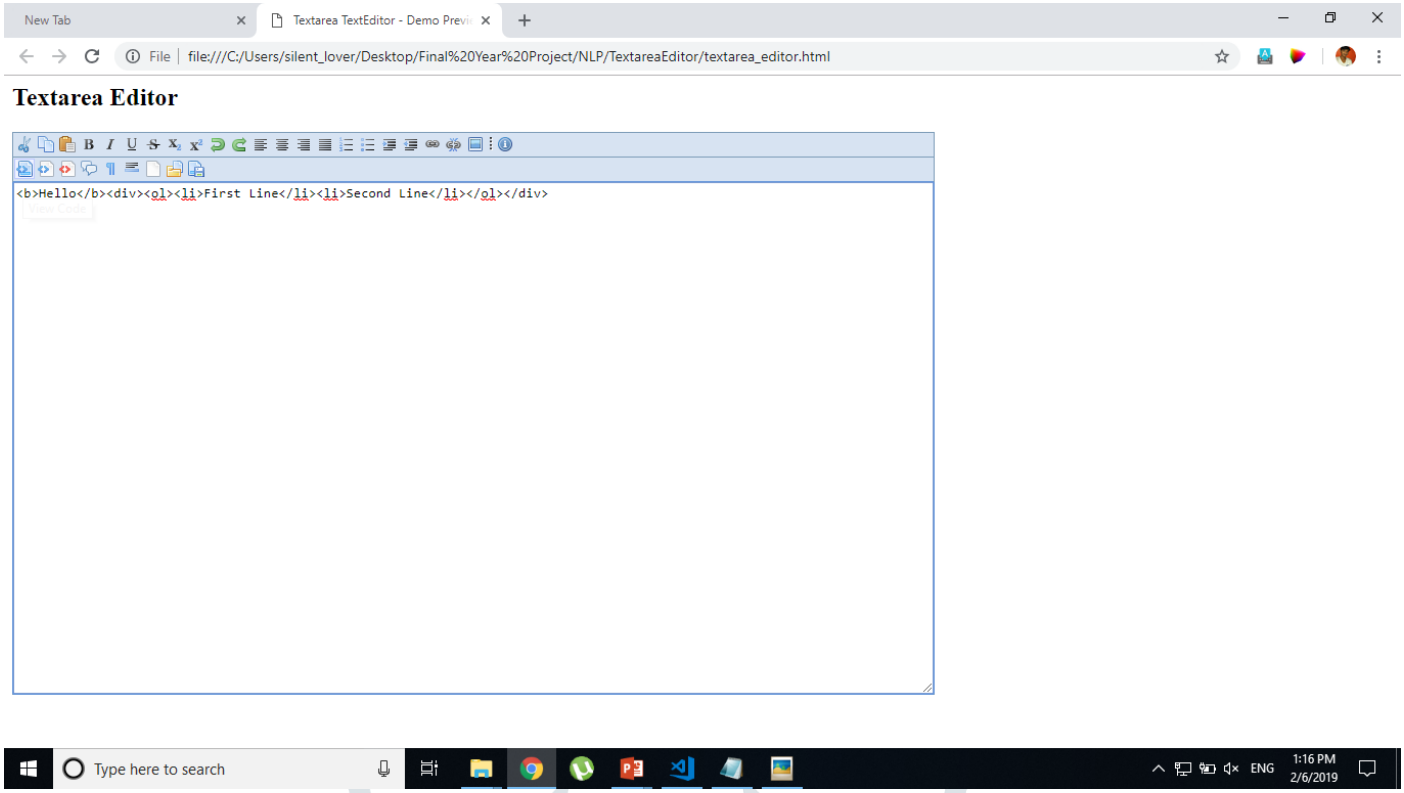
VI. FEATURES

- It is much more robust than the previous system.
- It is easy and efficient user interface.
- This application consumes less memory than the previous system.
- Free of cost.
- It can generate outputs in multiple form.

VII. RESULTS


New Tab
Textarea TextEditor - Demo Previ
File | file:///C:/Users/silent_lover/Desktop/Final%20Year%20Project/NLP/TextareaEditor/textarea_editor.html
Textarea Editor
Type Here....
Type here to search
ENG
1:13 PM
2/6/2019


New Tab
Textarea TextEditor - Demo Previ
File | file:///C:/Users/silent_lover/Desktop/Final%20Year%20Project/NLP/TextareaEditor/textarea_editor.html
Textarea Editor
Hello
1. First Line
2. Second Line
Type here to search
ENG
1:15 PM
2/6/2019



VIII. CONCLUSION

The developed software exterminates the thinking that a non-technical person cannot program a software for his own by providing a simple, acoustic and flexible interface which matches up to the users thinking and provides him the desired output at every step so that he can easily code any type of software whenever needed.

IX. REFERENCE

[1] Vishal Parekh, Dwivedi Nilesh, "Pseudo Code to Source Code Translation", in Journal of Emerging Technologies and Innovative Research (JETIR), Volume 3, Issue 11, November 2016

[2] Dipanjan Das, Ryan McDonald in Google Research, New York, USA and Carnegie Mellon University, Pittsburgh, PA, USA, in 2014

[3] Amal M R, Jamsheed C V, International Journal of Computational Science and Information Technology (IJCSITY), May 2016

[4] Priyanka, Priyanka HL, Priyanka P, Naveen Chandra Gowda, in Asian Journal of Engineering and Technology Innovation (AJETI), 2017.