ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 7, Issue 7, July 2019

Artificial Intelligence Based Automatic Application Source Code Generator Using Interactive Chatbot and Natural Language Processing Approach in both Standalone and Web Application

Yuvaraj Jonna [1], Nuthan Kanth Jonna[2], S S R C Kashyap[3]

Full Stack Software Developer, Object Frontier Software India Pvt, Ltd, Chennai, Tamilnadu, India[1]

Full Stack Software Developer, Accenture, Hyderabad, Telangana, India[2]

Full Stack Software Developer, Object Frontier Software India Pvt, Ltd, Chennai, Tamilnadu, India[3]

ABSTRACT: The idea is to create a tool which can generate the source code of a web application on its own. It is noted that the development of applications with the help of manual systems are highly complex and labor-intensive, requiring a high level of engineering training and skill. An automated application generator tool has been developed to automate this process, it follows the model-based development approach which is vital to the cost-effectiveness for development, it eliminates the manual efforts from coding to design, therefore accelerating the process while decreasing the chance of errors when compared to manual coding. A workstation system has been modeled that interactively aids the application generation procedure by writing the complete source code, with the complete build process, database creation, and connection handling along with a responsive user interface, using the composite techniques of artificial intelligence, fifth-generation languages, UI template, API contract, and Documentation thus adhering the coding best practices, semantics and syntactical constraints vividly. The model understands the user's requirements by interacting with the user thereby developing a complete full-stack application by undergoing a lot of refactoring internally to modulate the source code to produce a full-fledged working Application. The user may also include specific hints or annotations that will guide the automatic app generator tool in choosing a correct programming construct.

KEYWORDS: Fifth Generation Languages; Automatic Programming; Application software Generator; Code Generators; target code generators; AI

I. INTRODUCTION

Nowadays automation and the growth of computer engineering has drastically increased. As many people working in the it industry know that, the rate of development in the software industry is affecting the jobs in the technology-dependent market, in which the most work is being done by the software engineers could be now completed with the help of the relatively inexpensive robots and the automated computerized system by implementing the methods of the artificial intelligence [3]. Most of the futurists, researchers, influencers are arguing the long-term impact and the immediate consequences of this automation in recent years. Many industries faced a hard time to survive in the upcoming software market, whereas the core job tasks intend in following well-defined, repetitive procedures, which can be invariably coded through properly designed software. While analysing the IT industry's major fallout, it would be a better option in considering out the benefits of using automated tools. It also provides a new set of valuable opportunities in relevant fields. It just defines the humans work labour can be utilized in much more advanced jobs.

A. TRAJECTORY OF AUTOMATION

Generally, computerization, automation, and different types of non-human work have been received inside labor-intensive, very tedious occupations. Similarly, people are traded for proficient field-work and better results. Modern progressions have kept up much a similar direction, where human work executing basic errands is frequently supplanted by machines (and later robots) that can play out similar undertakings, a lot quicker and with a little vacation. However, while automation has normally been consigned to mechanical assignments and upgrades in the workforce, with the headway of computerization [6], these mechanized procedures are currently starting to infringe on the more inventive occupations too, where unobtrusive human decisions were customarily a need previously. PCs are currently every now and again utilized for errands extending from performing high-volume monetary exchanging to observing medical clinic serious consideration units.

This move comes to a great extent from the intensity of programming, algorithms, and AI/ML advancements. Automation has even begun to infringe on software engineering and programming, where advancements in algorithm design allow computers to make complex parameter and design choices within the software. Coding, specifically, as of observed has a spike in the immediate utilization of automation as a method for making better software. The ascent of agile programming advancement procedures has put overwhelming importance on automated testing and different DevOps practices in the course of the past few decades.

B. PROGRAMMING IS FOR EVERYONE

Since automation, computerization, robotics, and many different professions are all combined within these modern corporate sayings, simply having the title of "software engineer" isn’t generally the zenith of a well-suited employee for many of these organizations. Rather, what is important most to businesses is the ability to automate, to examine and assess issues intelligently to provide an out of the box solution at the workplace, with such a sensational move in job openings, gifted workers in the advanced working environment may need to be talented in zones that were recently thought to be out of the domain of probability, for example, coding. However, coding no longer a profession that ought to be considered elite for the top few. It feels good to see that coding and computer science became K-12 education. Rather, there’s an influx of both free and inexpensive education and training, millions of existing developers often able to provide guidance, and most importantly as we examined above, a huge demand for programmers and logical thinkers in technical positions.

II. RELATED WORK

In the last few years, a considerable amount of research has taken place to apply ML techniques to industrial problems giving the essential role to an AI framework in the business space, there is a need to address the difficulties that AI parts understand in the field of software systems. The essential focal point of this exploration is to provide a total application which is built inbuilt in a well oriented, systematic structure for developing the whole application source code, by assessing the situations given by the client, and deploying the complete application to live environment or giving it to the user in the user required format thereby solving the complex scenarios. In this way, we overview past work that takes a glance at the crossing point between AI and software engineering. We order the related work into three classifications. The first classification considers the use of AI in programming designing.
The applications incorporate using AI strategies in anticipating programming issues and deformities suggesting procedure model, and [1]estimating development effort. Although this classification goes for utilizing AI procedures to upgrade the way towards making software systems, none of these endeavors takes a gander at the difficulties of utilizing AI segments as a piece of the software systems.

As for the second classification, we present predictive models as a new class of necessities for engineering the problems, we study existing work that consolidates the domains of AI, re-engineering and prerequisites designing. We found that, in the course of the most recent decades, most of the researchers have utilized AI models in analyzing the prerequisites for various programming frameworks [1, 7, 8]. Research [7] utilizes supervised learning approaches to classify requirements as functional and non-functional requirements. Authors in [7] used AI to sort programming prerequisites by combining the user’s preferences.

DOI: 10.15680/IJIRCCE.2019. 0707006

Additionally, authors in [8] present an automated ranking system for overseeing potential risks. Nonetheless, in contrast to our work, none of the current examinations have concentrated on inspiring the necessities of AI applications themselves. The third classification aims at addressing the difficulties in creating AI systems [6, 1, 8, 5]. Past work [6, 8] states that distributed systems frameworks are required for a node-to- node AI pipeline. Authors in [1] propose an open-source distributed AI library for versatile usage of standard AI strategies. Specialists in [5] displayed a framework to optimize node-to-node AI frameworks, while authors in [9] presented a framework to oversee AI models. Even though these investigations center around offering answers for facilitating the way toward making AI models, none of these arrangements attempted to think about the point of view of generating a complete application by writing the complete source code considering the user requirements. These endeavors did not address the difficulties of distinguishing chances to improve the application development process utilizing AI. Besides, a large portion of these arrangements did not consider the prerequisites upheld by the software development domain.

In contrast to the past work, we offer an application dependent on the integrated AI framework that aides in developing, evaluating, and maintaining up the application development process by considering the client inputs. It is noted that the development of applications with the help of manual systems is highly complex and labor-intensive, requiring a high level of engineering training and skill. An automated application generator tool has been developed to automate these processes, it follows the model-based development approach which is vital to the cost-effectiveness for development, it eliminates the manual efforts from coding to design, therefore accelerating the process while decreasing the chance of errors when compared to manual coding. A workstation system has been modeled that interactively aids the application generation procedure by writing the complete source code, with the complete build process, database creation, and connection handling, along with a responsive user interface, using the composite techniques of artificial intelligence and fifth-generation languages. The model understands the user's requirements by interacting with the user thereby developing a complete full-stack application by undergoing a lot of refactoring internally to modulate the source code to produce a full-fledged working Application. The user may also include specific hints or annotations that will guide the automatic app generator tool in choosing a correct programming construct.

The main advantages of this application generator are that it is platform-independent i.e., it can generate applications which can be built and run on any platform and the whole application building process along with its dependencies are being containerized in the modules and the sub-modules of the code generators which will be running on a server such that the generated native applications can be run in any operating systems.

Depending on the target programming paradigm and requirements provided by the user for which the application to be built-on the application generator will enhance and make understand the user of the possible potential barriers while building the application and also to avoid modelling constructs the device will provide the possible errors to the user while generating the applications or with the help of reverse engineering it can make the changes immediately to the code to check the further consequences, suggests possible ways to build a solution using threat handling which is a key aspect of artificial intelligence such that the user can rethink of his considerations while the generator is doing the work.

III. PROJECT FORMULATION

This article idea is to create a tool where it can generate the source code of a web application on its own with user-specified programming stack by considering the user requirements such as Documentation, UI template, API contract, Database creation, etc., thus adhering the coding best practices, semantics and syntactical constraints vividly.

It is noted that the development of applications with the help of manual systems is highly complex and labor- intensive, requiring a high level of engineering training and skill. An automated application generator tool [2] has been developed to automate these processes, it follows the model-based development approach which is vital to the cost- effectiveness for development, it eliminates the manual efforts from coding to design, therefore accelerating the process while decreasing the chance of errors when compared to manual coding.

A workstation system has been modeled that interactively aids the application generation procedure by writing the complete source code, with the complete build process, database creation, and connection handling, along with a responsive user interface, using the composite techniques of artificial intelligence and fifth-generation languages. The model understands the user's requirements by interacting with the user thereby developing a complete full- stack application by undergoing a lot of refactoring internally to modulate the source code to produce a full-fledged working Application [4]. The user may also include specific hints or annotations that will guide the automatic app generator tool in choosing a correct programming construct.

The device architecture utilizes a set of rules on how to design the application and processing probabilities of a target test station and provides a corresponding set of target code generators. The kernel processor displays a sequential set of menus, showing the application writer the range of test, stimuli, and measurement possibilities, and allows appropriate selections to be made. The corresponding code generators are then invoked to create the application program described by the menu selections. Successful results can be obtained with a subset of a full-scale automatic application generator.

The whole idea is to create a tool which can generate the source code of a web application by gathering user inputs thereby helping the user in saving time. It is noted that the development of applications with the help of manual systems is highly complex and labor-intensive, requiring a high level of engineering training and skill. An automated application generator tool has been developed to automate these processes, it follows the model-based development approach which is vital to the cost-effectiveness for development, it eliminates the manual efforts from coding to design, therefore accelerating the process while decreasing the chance of errors when compared to manual coding [2].

A workstation system has been modeled that interactively aids the application generation procedure by writing the complete source code, with the complete build process, database creation and connection handling along with a responsive user interface, using the composite techniques of artificial intelligence, fifth-generation languages, UI template, API contract, and Documentation thus adhering the coding best practices, semantics and syntactical constraints vividly.

IV. PROPOSED MODEL

The AppGen software will be available in two modes the first will be a web application and the second would be a Standalone application in which these two versions contains 5 major parts in the generation process of the source code of the Application based on the user requirements and specified Tech Stack by the user [4].

The 5 major parts are stated below:
1.App Generator
2.Code Generator
3.Sub-Code Generator
4.Language module
5.Code Integrator

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 7, Issue 7, July 2019

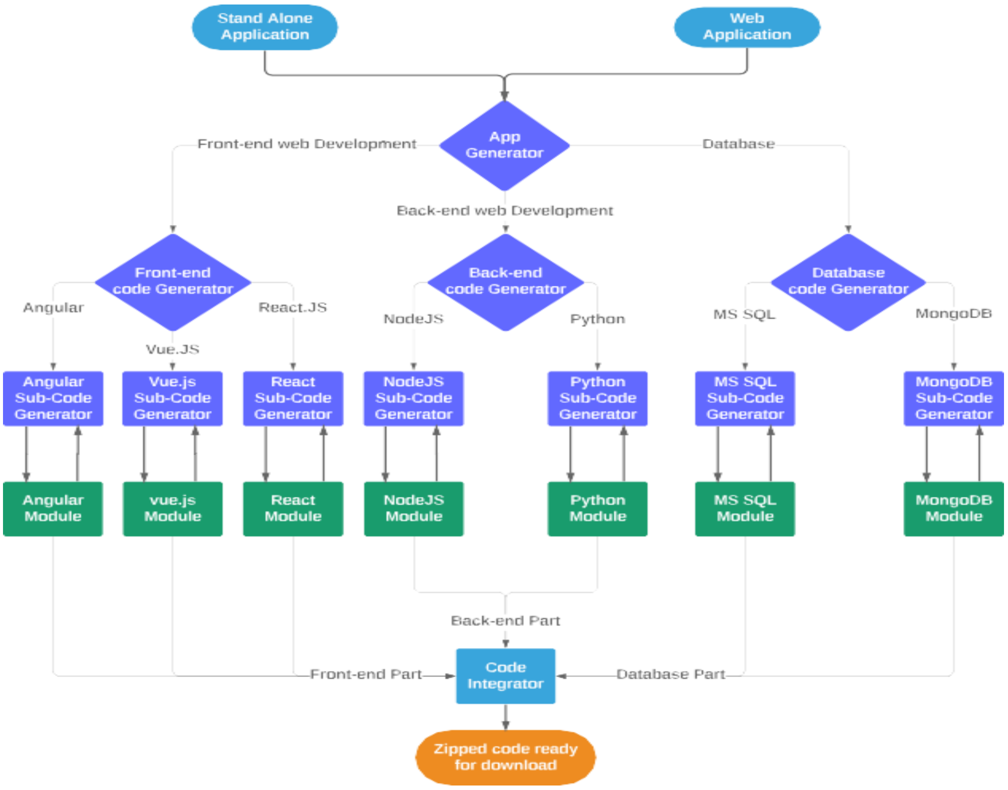


Fig:1 The Schematic block diagram of the App Gen

The Fig 1 defines the schematic block view of the application generator by clearly picturing the block-wise diagram for depicting the process of generating an application by considering the user inputs using application generator, though each block comprises of many packages and libraries, they are grouped logically to form a module to analyze the user inputs and generate the source code using the user desired tech stack to form the application.

1) *App Generator*

The App Generator is a part of AppGen which receives the data from the user (i.e: from both web and standalone application), analyze and frames into a definite structure by processing the user given keywords. It further identifies the phase of development (i.e. Back-end, Front-end, Database) and sends the data to the corresponding Code Generators for further process.

2) *Code Generator*

Each Code Generator is designed for specific development phases such as database creation, back- end development, and front-end design. It gets the data from the App Generator and analyzes in such a way that it identifies the tech stack to develop the source code based on the user logic and specified requirements. further, it navigates to the corresponding sub-code generators for generating the source code.

3) *Sub-Code Generator*

The Sub-Code Generators play a vital role in App-Gen. It is unique for every Tech Stack. It analyzes, develops and writes the source code of the Application based on the user requirements and logic by interacting with the code modules for syntax and best practices. These crumbs of the generated code will be sent to the Code Integrator for Developing an End-To-End Application.

4) *Code Module*

Every Tech stack contains their code modules. Each Code Module contains syntax and best practices of corresponding Tech stack which interact with the Sub-code Generator and provides the appropriate syntax for the development of the application.

5) *Code Integrator*

This is the final stage of the AppGen where it collects all the crumbs of code from the Sub-Code Generator of each Code generator (i.e. from Database, Back-end, Front-end) and it integrates all the crumbs of code to form a full-stack application which is developed based on the User Requirement and specified Tech stack. This Developed application is served to the user in two different ways:

1. Firstly, AppGen will package the developed application and make the application available for download.
2. Secondly, AppGen will ask for the GitHub access from the user then creates a new repository and push the developed code into the new repository of the user account.

V. DESCRIPTION

The Automatic Application Generator as we call it the AppGen is a multifaceted platform application is used to understand the given human requirements and help in generating an application in the human desired programming languages/frameworks by following proper syntax, semantics, and standards using Natural language processing toolkit. One advantage of letting an AI loose in this way is that it can search more thoroughly and widely than a human developer.

Thus, the **AppGen** could piece together source code in a way human may not have thought of. This version of AppGen comes as both web Application and Standalone Application where it can generate the source code of base UI Application based on user inputs and specified programming stack. This AppGen internally integrated with a Chatbot for the user interaction and it has two modes Firstly, Friendly mode where user can chat with the bot and can find some interesting stuff. Secondly, Development mode where the AppGen starts generating the Source Code of Application based on user requirements and specified Programming Stack.

A. HOW TO USE?

The chatbot is being served in the front-end to receive the user inputs, the chatbot is build using the existing python library known as ChatterBot V 1.0.2, which contains logical adapters for making the simple arithmetic calculation, time adapters for displaying current time and threshold adapters for matching the trained data of the chatbot with respect to the user input and in order to generate accurate result the threshold value is set to 90%.

Thus, would make the job of generating the automated responses to an analyzed user's input easily which has been integrated using Tkinter for developing GUI for standalone application. Also, we have integrated the AppGen with Azure's cognitive services for text-to-speech and vice-versa. In order to get out the correct analysis of what user is asking as a requirement and what appropriate responses the chatbot is giving to the user, the data in which the chatbot has been trained is given in yml format and to be trained with more valid inputs to ensure it would start off having knowledge about specific responses and to distinguish the mood of the user to generate the responses in favor for the user by analyzing his conversation pattern using the sentimental analysis.

The current training method for the chatbot is by using its corpus data and creating a list of statements which would represent a conversation using a list of statements that represent a conversation. The application can be used in two interactive modes. Firstly, Friendly Chat Mode, in which the user can interact with the chatbot to give his requirements and to clarify doubts. The program selects the closest matching response by searching for the closest matching known statement that matches the input, it then chooses a response from the selection of known responses to that statement.

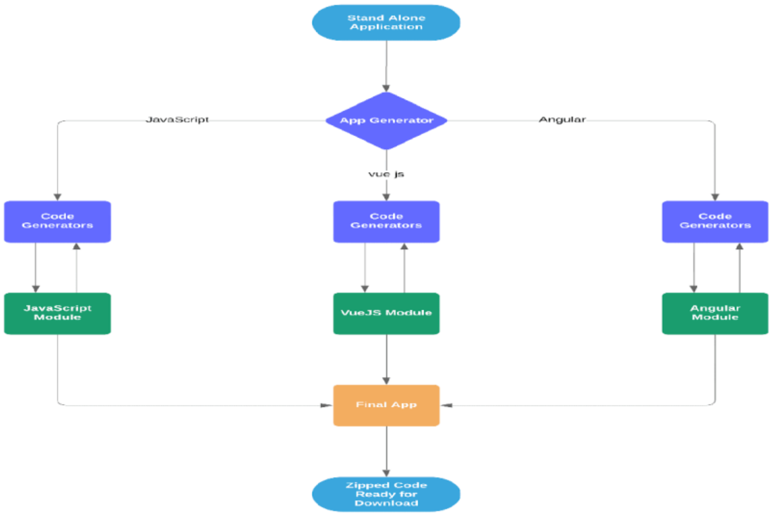
The other is Developer Mode in which the App generation based on the user requirements takes place that can be initiated as follows:

use 'create' keyword to start Generating Application and also specify the language in which app should be developed. example: 'create the addition of 2 numbers in vuejs'
In addition to this, We have Worked on NLP for enabling "Text-To-Speech " and " Speech-To-Text " conversion. This will help the user for better interaction with the AppGen. There is an option for Enabling/Disabling the conversion based on user preferences.

B. APPGEN AS STANDALONE APPLICATION

This Appgen Standalone application is built for Generating the Source code of application within the local machine, It uses the Tkinter for the GUI and this is completely built on python. This AppGen also supports Text-To- Speech and Speech-To-Text conversion, this comes with both friendly mode and Development mode as stated above.

Fig:2 The Schematic block diagram of the App Gen as. standalone application



This has some commands where we can find some help on how to use this application.

6) Working of Development Mode in standalone Application

Once the chatbot gets the input it will first look for the keywords whether the starting word of the string is " create " or not, if it is "create" then it starts development mode where the AppGen will split the string into a list of words and tries to find the required keywords.

After finding the keywords it will analyze the language or framework in which the code has to be generated and sends the information/keywords to the appropriate code generators. These Code Generators will analyze the keywords given by the user and interact with their associated code modules for the pre-programmed code templates and syntax for Generating the Application source code.

The AppGen will create a "final App" folder where the code Generator generates the source code for the user-specified functionalities in the preferred language/framework. Once after creating the "Final App" folder, it will be packaged/zipped and make available for downloading the Generated Application.

C. APPGEN AS WEB APPLICATION

This AppGen Web application is built for Generating the Source code of application globally through the internet, it uses the WebSocket for communication and this is completely built on python. This AppGen also likely to support Text-To- Speech and Speech-To-Text conversion.

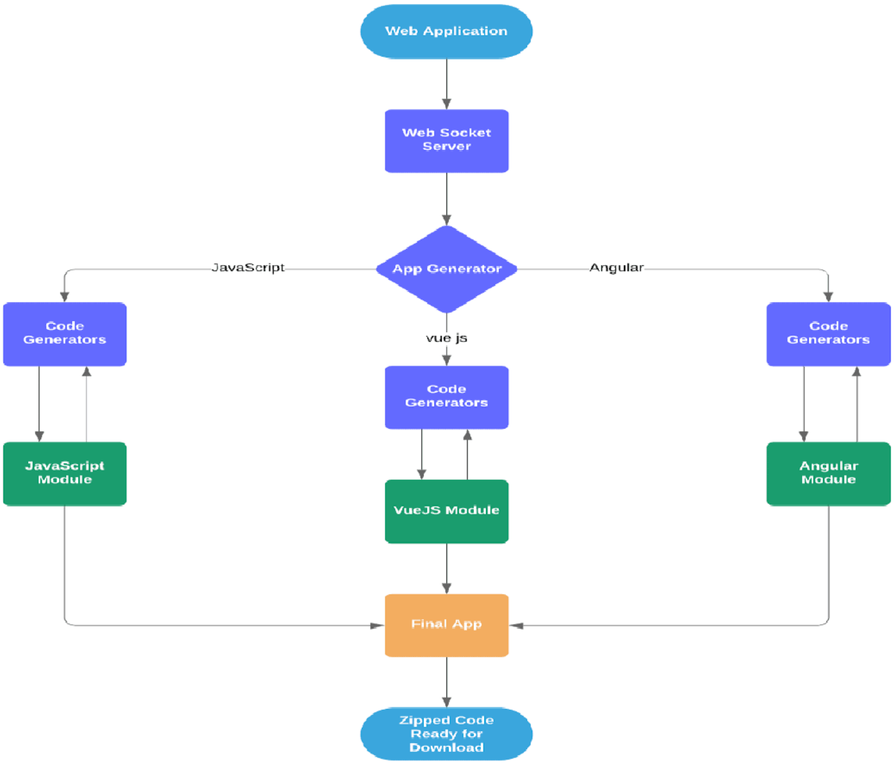


Fig:3 The Schematic block diagram of the App Gen as a web application

This comes with both friendly mode and Development mode as stated above. This has some commands where we can find some help on how to use this application. This Application is built using python and deployed using Azure cloud, thus anyone could use it to generate the source code of the application based on their specifications.

7) Working Of Development Mode in Web Application

In this Web Application, we use Simple Web Socket Server for running the WebSocket Application in server and this will help to go Online. Once the user connects with the AppGen then it will maintain the connection until the user closes the connection.

As we use WebSocket for communication, the establishment of connection takes place only once rather than connecting for every time. The WebSocket uses a WS/ WSS protocol for the connection establishment in which we will create an instance from the UI for changing the protocol from HTTP to WS/WSS protocol for connection establishment where the service success status code will be 101, which maintains a session for a particular user, the session will not close until the user quits the application.

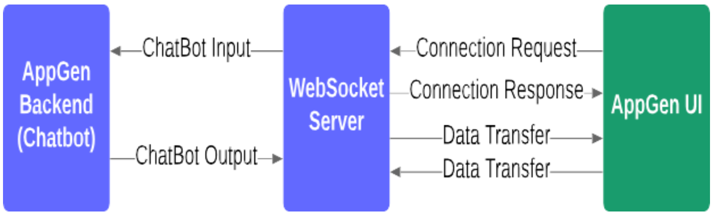


Fig:4 The block diagram of the integration of the App Gen backend and UI with the web socket

Once after the establishment of connection, the Appgen is ready to use. It will have a chatbot in the front-end to serve the Web Application. Once the chatbot gets the input from the user, it will first look for the keywords whether if the starting word of the string is " create " or not, if it is " create " then it starts development mode where the AppGen will split the string into a list of words and tries to find the required keywords.

After finding the keywords it will analyze the language or framework in which the code has to be generated and sends the information/keywords to the appropriate code generators. These Code Generators will analyze the keywords and interact with the associated code modules which contain the syntax and semantics for the code templates in generating the Application source code.

The AppGen will create a folder titled "final App" which contains the source code of the application which is generated by the code generator module, by adhering the practices and language/framework specified by the user. Once after creating the "Final App" folder, it will be packaged/zipped and make available for downloading the Generated Application.

Commands for AppGen
help = Guides on how to start using App Generator Bot list = It will List the features
list -O = It will List the operations that it can perform.
list -L = It will Lists the languages/framework in which it can develop the App.
The above listed are the commands through which the user can comfortably access the **AppGen**.

VI. Result Analysis & Benefits

A. EXPECTED OUTPUT

The proposed version of AppGen Will creates a base web application based on the user requirements along with the required configurations and environmental set-up in 3 different Tech-stacks namely VueJS, Angular, JavaScript. The current version is built for demonstrating the idea behind AppGen. This version of AppGen is made available in both web application and stand-alone application which are integrated with a chatbot for interacting with the user to get the required data for AppGen. The chatbot in AppGen is also configured to perform arithmetic operations and chat friendly with the user.
Version 2.0 will be developed to satisfy real-time scenarios in developing a complete full-stack application by reducing the effort of the developer in the application development process and to utilize their efforts in solving complex tasks.
The result of this tool is to generate the application source code based on the user requirements and user specified programming language in different programming stacks such as Back-end, Front-end, Database. The generated code is approximately 60% - 70% accurate for the user given requirements.

B. Benefits

1. This application reduces the human effort to build the application from scratch.
2. It saves time and money by creating a developed application based on user requirements.
3. The Application generator will create well documented, high-quality source code which is easily customizable.
4. This application will be much more helpful for the developers to invest their time on the complex/performance-related tasks instead of the same repetitive code building part.
5. This application can be used by start-up companies for releasing the alpha version of the products instantly, thereby helping to generate revenue and that can be used for the upbringing of the start-up companies.
6. This application reduces the risk of human errors in the source code
7. This application can be used by start-up companies for releasing the alpha version of the products rapidly, thereby helping generate revenue.

VII. Conclusion & Future Scope

In this paper, we presented the idea of an automatic application generator which develops an end-to-end web application completely by writing the source code along with the build files based on the user requirements, thereby

DOI: 10.15680/IJIRCCE.2019. 0707006

providing the fruitful results in the technical stack provided by the user. We designed the AppGen following the lean startup methodology to accommodate the business domain and the end-user perspectives. The proposed application consists of a set of phases during which the AppGen users can define possible opportunities to develop, evaluate, and deploy the generated application in their defined terms. The Application provides the user to choose their tech-stack from the available range of the programming paradigms from the scope of AppGen. To evaluate the application, we have undertaken a case study by testing the application with random inputs and undergoing all kinds of testing approaches provided the best results. The application will identify the initial requirements from the user to generate the source code of the desired application based on user requirements. The model is currently undergoing additional field testing and being updated to the fullest using machine learning algorithms. The results of the case study prove that the use of the proposed application (AppGen) can help organizations to deliver the software product quickly and initially it will help for the fund-raising.

REFERENCES

1. Anjum Avesani, P., Perini, A., Siena, A., Susi, A., 2015. Goals at risk? machine learning at support of early assessment, in: 2015 IEEE 23rd International Requirements Engineering Conference (RE), IEEE. pp. 252–255.
2. Budinsky, F.J., Finnie, M.A., Vlissides, J.M., Yu, P.S., 1996. Automatic code generation from design patterns. IBM systems Journal 35, 151–171.
3. Charntaweekhun, K., Wangsiripitak, S., 2006. Visual programming using flowchart, in: 2006 International Symposium on Communications and Information Technologies, IEEE. pp. 1062–1065.
4. Kurtanovis, Z., Maalej, W., 2017. Automatically classifying functional and non-functional requirements using supervised machine learning, in: 2017 IEEE 25th International Requirements Engineering Conference (RE), IEEE. pp. 490–495.
5. Meng, X., Bradley, J., Yavuz, B., Sparks, E., Venkataraman, S., Liu, D., Freeman, J., Tsai, D., Amde, M., Owen, S., et al., 2016. Mllib: Machine learning in apache spark. The Journal of Machine Learning Research 17, 1235–1241.
6. Perini, A., Susi, A., Avesani, P., 2012. A machine learning approach to software requirements prioritization. IEEE Transactions on Software Engineering 39, 445–461.
7. Sparks, E.R., Venkataraman, S., Kaftan, T., Franklin, M.J., Recht, B., 2017. Keystoneml: Optimizing pipelines for large-scale advanced analytics, in: 2017 IEEE 33rd international conference on data engineering (ICDE), IEEE. pp. 535–546.
8. Supaartagorn, C., 2017. Web application for automatic code generator using a structured flowchart, in: 2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS), IEEE. pp. 114– 117.
9. Vartak, M., Subramanyam, H., Lee, W.E., Viswanathan, S., Husnoo, S., Madden, S., Zaharia, M., 2016. M odel db: a system for machine learning model management, in: Proceedings of the Workshop on Human-In-the-Loop Data Analytics, ACM. p. 14.

BIOGRAPHY

Jonna Yuvaraj *(primary author)*He received Bachelor's degree (2017) in Electronics and Communication Engineering from Anna University, Tamilnadu. During his college days, he achieved a world record of building a mobile app using Microsoft tools organized by Microsoft Research Community, Chennai and GRRIL Pty Ltd, London, UK. He designed and developed websites for the National level symposiums for three consecutive years at Kingston college of engineering, also organized National level symposium and conducted many technical events. He secured first prize in project presentation at SAEINDIA South India level student convention for two consecutive years. He also developed a project to help the visually impaired people using the magnetic tensor approach based on geomagnetic field effects. He is a certified ZOHO creator developer. Since 2017 July, he has been a Full Stack Software Engineer at Delivery Unit, Object Frontier Software Pvt. Ltd. He continued his passion for research regarding the cognitive computing approaches, ML, Cloud computing. He contributed to the companies RD team for the growth of the organization, in route to achieving that he completed his Azure developer cloud certification. His main area of interest is in web development, data analysis, ML exploring cloud computing and providing an optimized solution for the hypervisor-based cloud architecture. He authored a book on C programming which is indexed in most of the indexing platforms.

DOI: 10.15680/IJIRCCE.2019. 0707006

Jonna Nuthan *(co-author)*He is graduate from the University of Houston-Clear Lake in the field of Management Information Systems and received a bachelor's degree from the Anna University in the field of Electronics and Communication Engineering. Right from schooling he actively took part in science fairs, during bachelors he presented 12 plus papers at a national level technical symposium and Conferences. He worked as an intern at one of India's renowned research organization National Atmospheric and Research Laboratory (NARL), where he involved in designing and development of Automated Tracking software for detecting Abnormal Radar Responses along with Senior Scientists. While pursuing master's degree he has taken up multiple roles, He filled in as Tutor for Management Information Systems by assisting his peers in understanding concepts of various programming languages, also served as Graduate Teaching Assistant under the Faculty Chair of MIS UHCL besides these he is also involved in various community services. He worked with Accenture as a Software Engineer for a couple of years, His work at Accenture involved in getting hands-on with many cutting-edge technologies along which he has overtaken many projects and assisted clients. Currently, he is working as a Software Developer in Association Health Care Management Inc (AHCM).

S S R C Kashyap *(co-author)* He received B.E. (2018) in Electronics and Communication Engineering from Sreenivasa Institute of Technology and Management Studies, Chittoor, Andhra Pradesh. While in engineering developed an interest towards machine learning, he worked on a prototype regarding the computer vision approach to help physically challenged people using image processing techniques and also pursued a certification on satellite imaging and radar sensing. Since 2018 June he has been started working for the Object Frontier Software Pvt, Ltd. As a Full Stack Software Engineer. He was a reviewer and publishing data curator for a book on C programming. His main area of interest is regarding Genetic programming, Cryptography, Game Theory, Supervised Learning.