iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 11 **Issue:** VI **Month of publication:** June 2023

DOI: https://doi.org/10.22214/ijraset.2023.54481

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue VI Jun 2023- Available at www.ijraset.com

Light Control Design by Using Social Media

Lubna Yasmeen M[1], Biereddy Mridgala[2], Malgireddy Hrishith Raj Reddy[3], Maligireddy Pranav Reddy[4], Pegada Chandana[5]
Computer Science and Engineering, Sreyas Institute of Engineering and Technology,Hyderabad, Telangana, India

Abstract: ***Since from a decade many areas becoming trendy from them social media and internet of things are more in need for day today life. Here from social media chatting platform named telegram is been used parallel to make things work much easier than before with NodeMCU device sized as debit card which plays major role in implementing the control of multiple devices using internet of things. Telegram application which has a capability of having bot to communicate. In this project basic text created by telegram bot will be used as simple or major instructions and these instructions will be read by NodeMCU and control home appliances like bulbs, fans, air conditioners, televisions, etc as instructed by the telegram bot. All mentioned functions can be controlled in all modes of connectivity's like internet, local area network, Wi-Fi,.Etc. Which will leads us to save power and time majorly and also leads to enhance the life of the appliances. As of now most will be frequently checking social media. So people can also have tracking on appliances functioning. No matters on which user interface a particular user uses appliances can be controlled from entire world.***

Keywords: Local Area Network, Wi-Fi, Internet of Things, NodeMCU

I. INTRODUCTION

Home device market is growing every year due to development of smart devices Smart devices can be easily controlled using user friendly mobile applications installed in smart phones that are connected to the internet or a local network. It is the process of automatically performing everyday functions around the home to save the time, energy, money "Automation replaces human workers by machines". The system is about remotely managed automation and appliance like fan, lamps etc. using telegram and NodeMCU. Here users of this system has simple interface to control their home appliances from any where.

A. Problem Statement

The field of Automation has well advanced in Industries, as majority of automobile industry plants as well as bottling plants have Automated assembly lines. But automation has not yet implemented in the homes especially in India. If automation was to be used in homes than everyday life would be get eased. Simple example of use of automation in home can be seen in the transfer of water from the under-ground water tank to the over-head water tank, by sensing the level of water in both the tanks. This process eases every time effort the user has to put in for filling the tank and also helps in saving water. Also, people are getting more acquainted daily with the use of Smartphone and tablets which are capable of doing much of PC's work handy. So we have decided to make a low cost Embedded System in which the smart phones can be used to help automate entire home. In this system the user will have remote access and control them.

II. TECHNOLOGY DESCRIPTION

A. Lua Scripting Language

Lua originated in 1993 as a language for extending software applications to meet the increasing demand for customization at the time. It provided the basic facilities of most procedural programming languages, but more complicated or domain-specific features were not included; rather, it included mechanisms for extending the language, allowing programmers to implement such features. As Lua was intended to be a general embeddable extension language, the designers of Lua focused on improving its speed, portability, extensibility, and ease-of-use in development. Lua is a powerful, efficient, lightweight, embeddable scripting language. It supports procedural programming, object-oriented programming, functional programming, data-driven programming, and data description. Lua combines simple procedural syntax with powerful data description constructs based on associative arrays and extensible semantics. Lua is dynamically typed, runs by interpreting bytecode with a register-based virtual machine, and has automatic memory management with incremental garbage collection, making it ideal for configuration, scripting, and rapid prototyping. Lua has been used in many industrial applications (e.g., Adobe's Photoshop Lightroom), with an emphasis on embedded systems (e.g., the Ginga middleware for digital TV in Brazil) and games (e.g., World of Warcraft and Angry Birds).

Lua is currently the leading scripting language in games. Lua has a solid reference manual and there are several books about it. Several versions of Lua have been released and used in real applications since its creation in 1993. Lua featured in HOPL III, the Third ACM SIGPLAN History of Programming Languages Conference, in 2007. Lua won the Front Line Award 2011 from the Game Developers Magazine. Lua 1.0 was designed in such a way that its object constructors, being then slightly different from the current light and flexible style, incorporated the data-description syntax of SOL (hence the name Lua: Sol meaning "Sun" in Portuguese, and Lua meaning "Moon"). Lua syntax for control structures was mostly borrowed from Modula (if, while, repeat/until), but also had taken influence from CLU (multiple assignments and multiple returns from function calls, as a simpler alternative to reference parameters or explicit pointers), C++ ("neat idea of allowing a local variable to be declared only where we need it"), SNOBOL and AWK (associative arrays). 26 In an article published in Dr. Dobb's Journal, Lua's creators also state that LISP and Scheme with their single, ubiquitous data-structure mechanism (the list) were a major influence on their decision to develop the table as the primary data structure of Lua.

B. Features

Lua is commonly described as a "multi-paradigm" language, providing a small set of general features that can be extended to fit different problem types. Lua does not contain explicit support for inheritance, but allows it to be implemented with metatables. Similarly, Lua allows programmers to implement namespaces, classes, and other related features using its single table implementation; first-class functions allow the employment of many techniques from functional programming; and full lexical scoping allows fine-grained information hiding to enforce the principle of least privilege. In general, Lua strives to provide simple, flexible meta-features that can be extended as needed, rather than supply a feature-set specific to one programming paradigm. As a result, the base language is light—the full reference interpreter is only about 247 kB compiled—and easily adaptable to a broad range of applications. A dynamically typed language intended for use as an extension language or scripting language, Lua is compact enough to fit on a variety of host platforms. It supports only a small number of atomic data structures such as boolean values, numbers (double-precision floating point and 64- bit integers by default), and strings. Typical data structures such as arrays, sets, lists, and records can be represented using Lua's single native data structure, the table, which is essentially a heterogeneous associative array. Lua implements a small set of advanced features such as first-class functions, garbage collection, closures, proper tail calls, coercion (automatic conversion between string and number values at run time), coroutines (cooperative multitasking) and dynamic module loading. Lua is intended to be embedded into other applications, and provides a C API for this purpose. The API is divided into two parts: the Lua core and the Lua auxiliary library. The Lua API's design eliminates the 27 need for manual reference management in C code, unlike Python's API. The API, like the language, is minimalistic. Advanced functionality is provided by the auxiliary library, which consists largely of preprocessor macros which assist with complex table operations. The Lua C API is stack based. Lua provides functions to push and pop most simple C data types (integers, floats, etc.) to and from the stack, as well as functions for manipulating tables through the stack. The Lua stack is somewhat different from a traditional stack; the stack can be indexed directly, for example. Negative indices indicate offsets from the top of the stack. For example, −1 is the top (most recently pushed value), while positive indices indicate offsets from the bottom (oldest value). Marshalling data between C and Lua functions is also done using the stack. To call a Lua function, arguments are pushed onto the stack, and then the lua_call is used to call the actual function. When writing a C function to be directly called from Lua, the arguments are read from the stack. Besides standard library (core) modules it is possible to write extensions using the Lua API. Extension modules are shared objects which can be used to extend the functionality of the interpreter by providing native facilities to Lua scripts. Lua scripts may load extension modules using require, just like modules written in Lua itself, or with package.loadlib. When a C library is loaded via require('foo') Lua will look for the function luaopen_foo and call it, which acts as any C function callable from Lua and generally returns a table filled with methods . A growing collection of modules known as rocks are available through a package management system called LuaRocks, in the spirit of CPAN, RubyGems and Python eggs. Prewritten Lua bindings exist for most popular programming languages, including other scripting languages. For C++, there are a number of template-based approaches and some automatic binding generators.

C. Applications

In video game development, Lua is widely used as a scripting language by programmers, mainly due to its perceived easiness to embed, fast execution, and short learning curve.[24] Notable games which use Lua include Roblox,[25] Garry's Mod, World of Warcraft, Payday 2, Phantasy Star Online 2, Dota 2, Crysis,[26] and many others. Some games that do not natively support Lua programming or scripting, have this functionality added by mods, such as ComputerCraft does for Minecraft.

In addition, Lua is also used in non-video game software, such as Adobe 28 Lightroom, Moho, iClone, Aerospike and certain system software in FreeBSD and NetBSD, and is used as a template scripting language on MediaWiki using the Scribunto extension. In 2003, a poll conducted by GameDev.net showed Lua was the most popular scripting language for game programming.[28] On 12 January 2012, Lua was announced as a winner of the Front Line Award 2011 from the magazine Game Developer in the category Programming Tools. A large number of non-game applications also use Lua for extensibility, such as LuaTeX, an implementation of the TeX type-setting language, Redis, a key-value database, Neovim, a text editor, Nginx, a web server, and Wireshark, a network packet analyzer. Through the Scribunto extension, Lua is available as a server-side scripting language in the MediaWiki software that powers Wikipedia and other wikis. Among its uses are allowing the integration of data from Wikidata into articles, and powering the automated taxobox system.

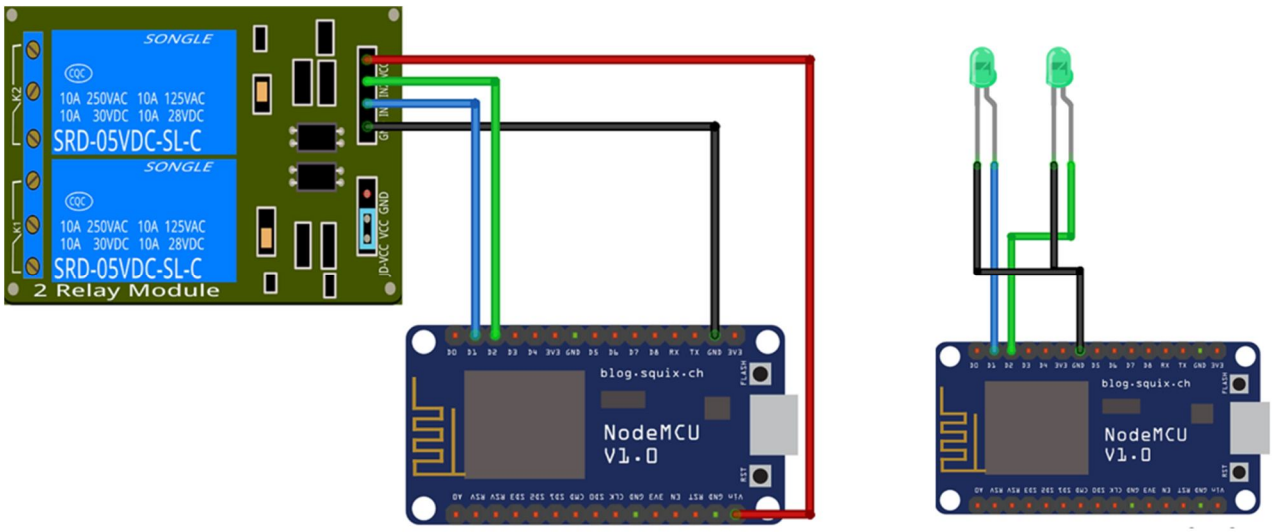
D. Advantages

1) Robust Language
2) Fast
3) Portable
4) Embeddable
5) Powerful (but simple)
6) Flexible
7) Easy app integration
8) Easy debugging
9) Cross-platform compatibility and support

III. MODULES:

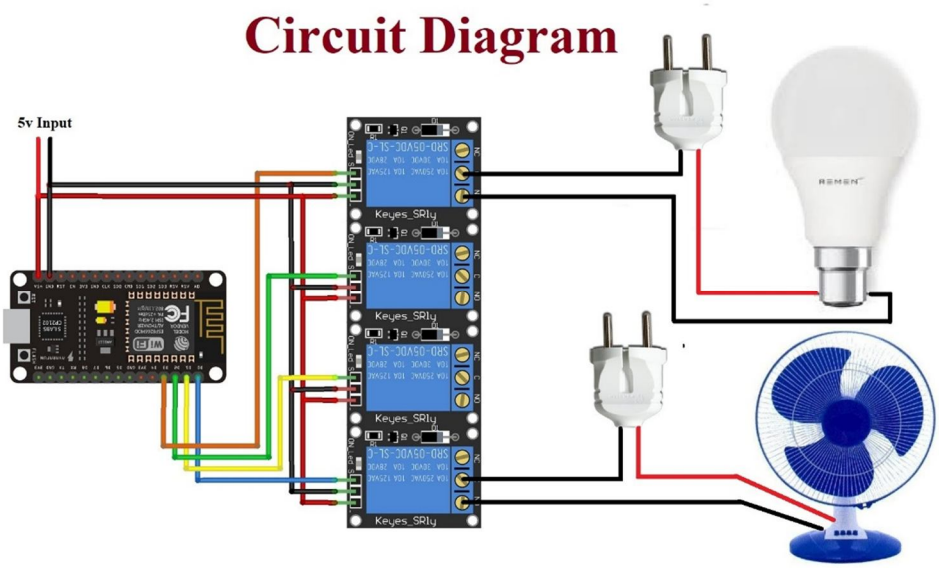
1) Control home appliances with Wi-Fi (Telegram app).
2) Control home appliances with Telegram dashboard.
3) Control home appliances with manual switches or push buttons.
4) Monitor real-time feedback in the Telegram App.
5) Control appliances without Wi-Fi from manual switches.

IV. ARCHITECTURE DIAGRAM



4.0.1 Architecture Diagram

V. CIRCUIT DIAGRAM



5.0.1 Circuit Diagram

VI. RESULTS

Figure 1: User's Telegram Interface

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue VI Jun 2023- Available at www.ijraset.com

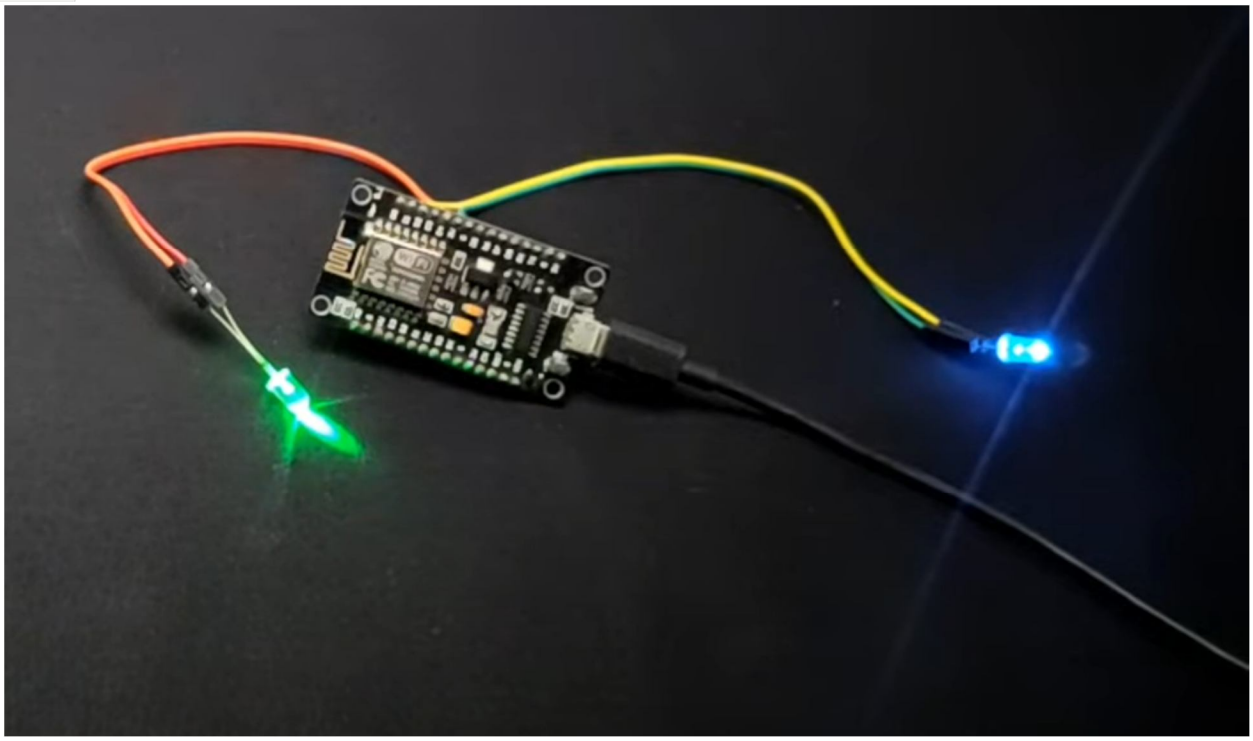


Figure 2: Lights has been turned on

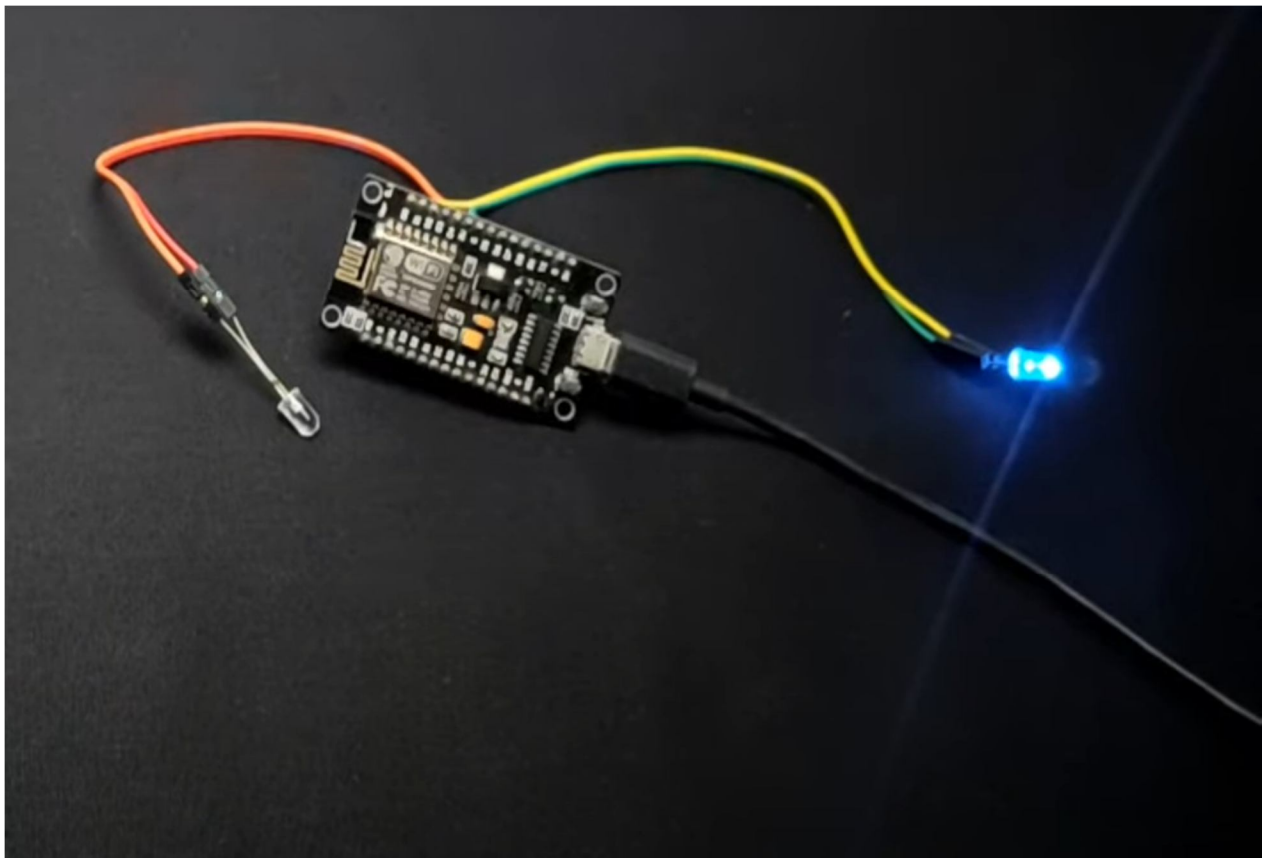


Figure 3: Light 1 has been turned on and 2 is turned off

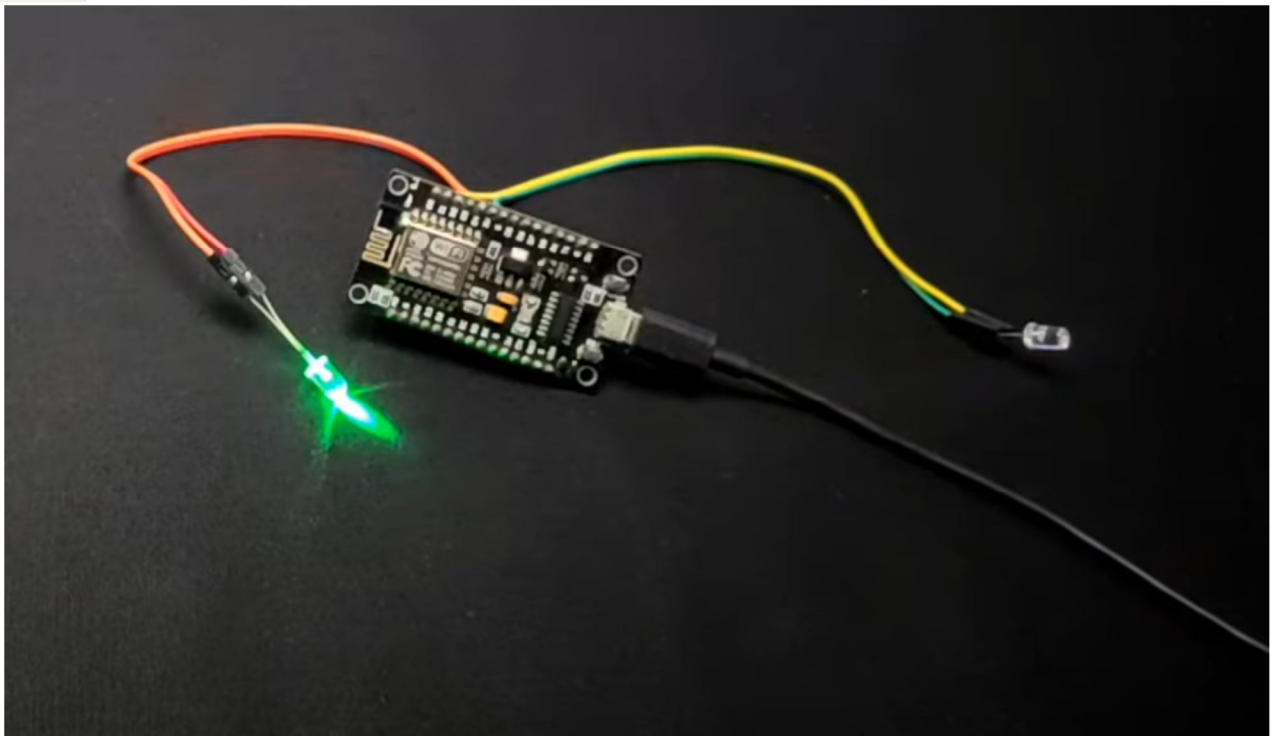


Figure 4: Light 1 has been turned off and 2 is turned on

VII. CONCLUSION

Today The ideas and benefits that are brought by the adaption of the internet of things (IoT) are likely to revolutionize how humans respond to their environment. The developments have made it possible for humans to operate the various functions within a home from one system. Even though there are security concerns on having all the house functions being managed by a single system, the fears are baseless. The technology customizes the features, and in case of intrusion in one area of the system, the chances of the homeowner noticing the change are high. More so, the IoT concept is beneficial to the environment where it embraces the responsible use of resources in the community.

VIII. FUTURE SCOPE

Today everything is reliant on calculation and data innovation, NodeMCU turns out to be a savvy, monetary and effective stage for executing the home automation. This paper gives an essential utilization of NodeMCU in home automation. The code used is very generic and adaptable in an easy to use way and can be stretched out for any future usage like power control mechanism, surveillance system, anti-theft system and so forth, effectively. Also, this strategy is superior to other home automation techniques is a few ways. Since Telegram is a cloud-based messaging application with focus on security and speed, it makes it easy to use by anyone since it is being offered in most of the platforms available in the market. What makes it unique is the secure end-to-end user protocol that it makes use of and the speed which it provides. On the other hand, it offers the unique property of being heavily encrypted and self-destruction of messages.

IX. ACKNOWLEDGEMENT

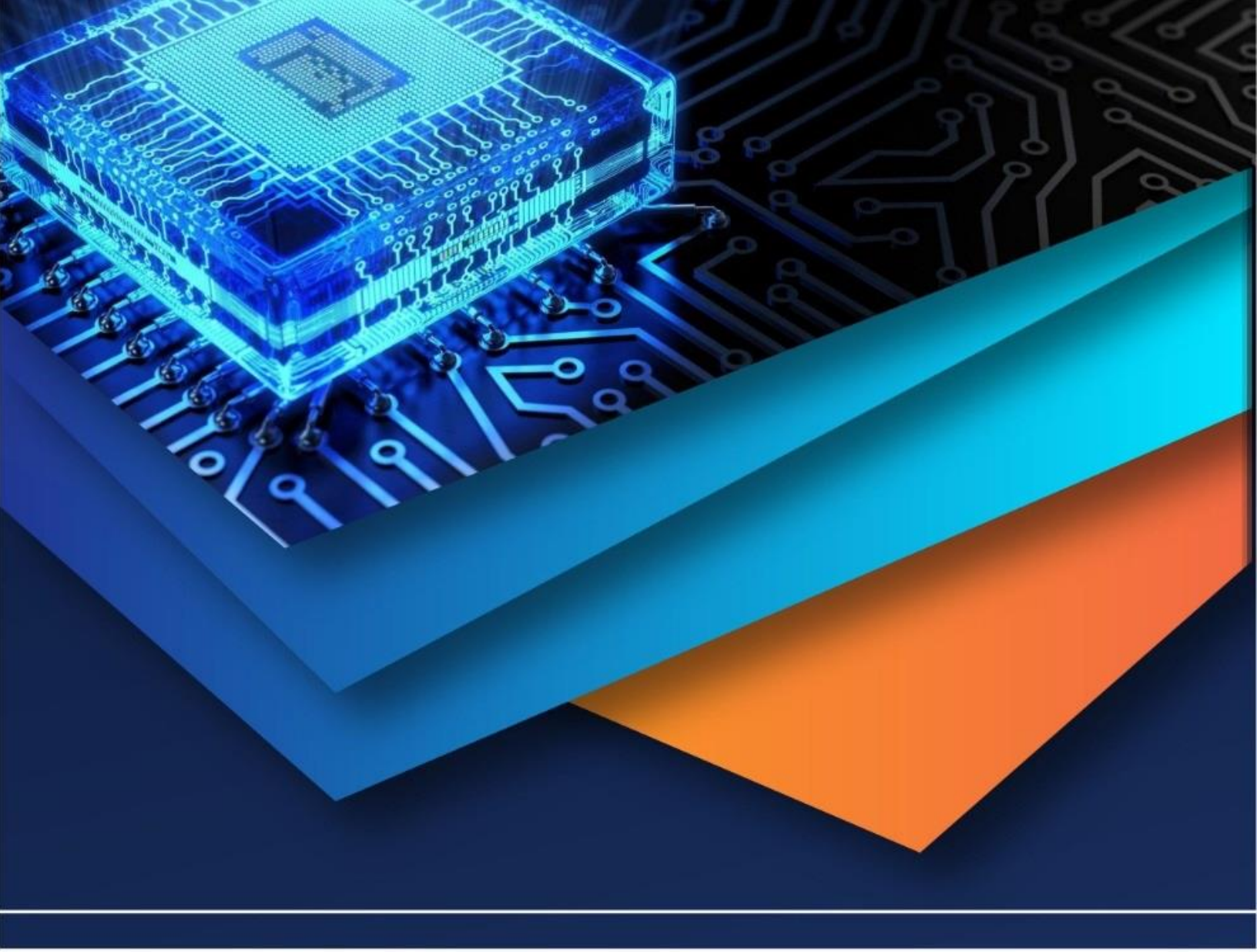
Thank you very much, Mrs. M Lubna Yasmeen, Assistant Professor, Department of Computer Science a person who helped us in every difficult situation we faced during the creation of the project. I would also like to thank Department Faculty, Principal and the Management of Sreyas Institute of Engineering and Technology for giving us an opportunity.

REFERENCES

[1] Yinghui H., Guanyu L., 2010. Descriptive Models for Internet of Things. IEEE International Conference on Intelligent Control and Information Processing, Dalian, China, Pages: 483- 486.

[2] S. Kousalya, G. Reddi Priya, R. Vasanthi, and B. Venkatesh, “IOT based smart security and smart home automation,” Int J Eng Res TechnolIJERT, vol. 7, no. 04, pp. 2278–0181, 2018.

[3] N. Hossain, M. A. Hossain, R. Sultana, and F. A. Lima, "A Security Framework for IOT based Smart Home Automation System," Glob. J. Comput. Sci. Technol., 2018.
[4] A. A. Salih and M. B. Abdulrazaq, "Combining best features selection using three classifiers in intrusion detection system," in 2019 International Conference on Advanced Science and Engineering (ICOASE), 2019, pp. 94–99.
[5] A. A. Salih and M. B. Abdulrazaq, "Combining best features selection using three classifiers in intrusion detection system," in 2019 International Conference on Advanced Science and Engineering (ICOASE), 2019, pp. 94–99.
[6] O. M. Ahmed and W. M. Abduallah, "A Review on Recent Steganography Techniques in Cloud Computing," Acad. J. Nawroz Univ., vol. 6, no. 3, pp. 106–111, 2017.
[7] M. A. Hoque and C. Davidson, "Design and Implementation of an IoT-Based Smart Home Security System," Int. J. Networked Distrib. Comput., vol. 7, no. 2, pp. 85–92, 2019. 55
[8] N. Harki, A. Ahmed, and L. Haji, "CPU Scheduling Techniques: A Review on Novel Approaches Strategy and Performance Assessment," J. Appl. Sci. Technol. Trends, vol. 1, no. 2, pp. 48–55, 2020.
[9] Third Generation Partnership Project, Technical Report 36.888 v12.0.0, Study on Provision of Low-cost Machine-Type Communications (MTC) User Equipment (UEs) Based on LTE, 2013.
[10] Third Generation Partnership Project, Technical Report 45.820 v13.0.0, Cellular System Support for Ultralow Complexity and Low Throughput Internet of Things, 2016.
[11] Third Generation Partnership Project, Technical Specification 23.060 v14.0.0, General Packet Radio Service (GPRS); Service Description; Stage 2, 2016.
[12] Third Generation Partnership Project, Technical Specification 24.008 v14.0.0, Mobile Radio Interface Layer 3 Specification; Core Network Protocols; Stage 3, 2016.
[13] Third Generation Partnership Project, Technical Specification 36.211 v14.0.0, Evolved Universal Terrestrial Radio Access (E-UTRA); Physical Channels and Modulation, 2016.
[14] Third Generation Partnership Project, Technical Specification 36.306 v14.0.0, Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment (UE) Radio Access Capabilities, 2016.
[15] Third Generation Partnership Project, Technical Report 38.913, v14.2.0, Technical Specification Group Radio Access Network; Study on Scenarios and Requirements for Next Generation Access Technologies; (Release 14), March 2017.
[16] Third Generation Partnership Project, Technical Specifications 36.331 v13.3.0, Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Radio Resource Control (RRC); Protocol 56 Specification, 2016.
[17] R. Howells, "The Business Case for IoT," SAP, 18 June 2015;
[18] D. Evans, "The Internet of Things: How the Next Evolution of the Internet Is Changing Everything," Cisco, Apr. 2011;
[19] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A Survey," Computer Networks, vol. 54, no. 15, 2010, pp. 2787–2805;
[20] Status Report: Reference Architecture Model Industrie 4.0 (RAMI4.0), VDI/VDE Society Measurement and Automatic Control, July 2015;
[21] F. John Dian, R. Vahidnia and A. Rahmati, "Wearables and the Internet of Things (IoT), Applications, Opportunities, and Challenges: A Survey," in IEEE Access, vol. 8, pp. 69200-69211, 2020, doi: 10.1109/ACCESS.2020.2986329.
[22] A. Yousefi, F. J. Dian, K. Somaratne "Analysis of time synchronization based on current measurement for Bluetooth Low Energy (BLE)," in IEEE IEMCON, pp. 602 – 607, Vancouver, OCT. 2017.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY