STM JOURNALS

International Journal of
Electrical and Communication
Engineering Technology

Volume 2, Issue 1, 2024
January–June
DOI (Journal): 10.37591/IJECET

https://journals.stmjournals.com/ijecet

Review IJECET

API Development for Cloud Parameter Curation

Shubham Salunkhe[1], Pruthviraj Zambare[2], Sakshi Shinde[3], S. K. Godase[4,*]

Abstract

API development for cloud parameter curation is pivotal in managing and optimizing various parameters within cloud environments. This project aims to develop a set of APIs that facilitate the efficient configuration, monitoring, and management of cloud parameters such as network settings, security policies, and resource allocation. Leveraging PHP for development and GoDaddy servers for hosting, the APIs provide a scalable and accessible solution for controlling cloud parameters remotely. Security measures, including SSL encryption and access controls, are implemented to ensure data integrity and protect against unauthorized access. Through this project, users can seamlessly adjust cloud parameters to optimize performance, scalability, and security, thus enhancing the overall efficiency and reliability of cloud-based systems.

Keywords: API, parameter curation, granular control, SSL encryption, PHP, python, motor,

INTRODUCTION

API development for cloud parameter curation is essential for effectively managing and fine-tuning various aspects of cloud infrastructure. APIs facilitate automation, granular control, and seamless integration with DevOps practices, empowering organizations to optimize performance, scalability, and security in the cloud environment. Developers leverage APIs to dynamically adjust parameters, enforce security policies, and monitor resources, fostering continuous improvement and innovation. Through APIs, organizations achieve greater efficiency, consistency, and agility in their cloud operations, enabling them to meet evolving business needs while ensuring compliance with regulatory requirements and industry best practices. For controlling motor parameters remotely, an API serves as the interface between the user/application and the motor device itself [1–9]. This API facilitates the exchange of commands and data, allowing users to adjust brightness and speed settings with ease, irrespective of their physical location. At the core of API development for motor control lies the concept of endpoints. These endpoints represent specific functionalities or actions that the API can perform. For our scenario, endpoints could include commands to set brightness levels, adjust motor speed, retrieve current settings, and more. To ensure security and accessibility, the API typically employs authentication mechanisms such as API keys, OAuth tokens, or other forms of credentials. This prevents unauthorized access and misuse of the motor control functionalities. Cloud integration is a crucial aspect of modern API development, enabling scalability, flexibility, and remote accessibility. By hosting the API on a cloud platform, users can interact with the motor from anywhere with an internet connection, leveraging the cloud's infrastructure for reliability and availability [10–18]. Implementation-wise, the API can be developed using various programming languages and frameworks, depending on factors

***Author for Correspondence**
S. K. Godase
E-mail: sonali.karche@sknscoe.ac.in

[1]Student, Department of Electronics & Telecommunication Engineering, Solapur University, Pandharpur, Maharashtra, India
[2]Assistant Professor, Department of Electronics & Telecommunication Engineering, Solapur University, Pandharpur, Maharashtra, India

Received Date: May 13, 2024
Accepted Date: June 11, 2024
Published Date: June 18, 2024

Citation: Shubham Salunkhe, Pruthviraj Zambare, Sakshi Shinde, S. K. Godase. API Development for Cloud Parameter Curation. International Journal of Electrical and Communication Engineering Technology. 2024;2(1): 1–8p.

like performance requirements, scalability goals, and developer preferences. Popular choices include Node.js, Python, Java, and frameworks like Express.js, Flask, or Spring Boot. For controlling brightness, the API endpoints might accept parameters such as intensity levels or colour values, translating these inputs into appropriate signals for the motor's control circuitry. Similarly, speed control endpoints could accept speed values or percentage adjustments, translating them into motor drive signals [19–24] Error handling and feedback mechanisms are critical components of the API design, ensuring that users receive informative responses in case of invalid inputs, technical issues, or other errors. Well-defined error codes and messages help diagnose and troubleshoot issues effectively. Documentation is indispensable for API usability and adoption. A comprehensive API documentation should detail each endpoint, its parameters, expected inputs, response formats, and example usage scenarios. Clear and concise documentation simplifies integration for developers and fosters a smooth user experience. In summary, API development for cloud-based motor control involves designing and implementing a set of endpoints that enable remote adjustment of brightness and speed parameters. Through secure authentication, cloud hosting, and robust error handling, the API provides a reliable and user-friendly interface for controlling motors in diverse applications. [25–32]

METHODOLOGY

Data Pre-Processing

In data processing for cloud parameter curation, a structured methodology is followed to handle the collection, pre-processing, analysis, and storage of data effectively. Firstly, data sources are identified, and raw data is cleansed and pre-processed to ensure consistency and accuracy. Suitable storage solutions are selected, and data schemas are designed to organize and manage the data efficiently. Data analysis is conducted using statistical methods or machine learning algorithms to derive meaningful insights. Real-time processing and batch processing workflows are implemented to handle streaming and large volumes of data, respectively. Data transformation is performed as needed to meet specific requirements, and data validation is carried out to ensure accuracy and compliance with standards [33–41] Security and compliance measures are implemented to protect sensitive data and adhere to regulations. Finally, monitoring and optimization processes are put in place to track processing pipelines' performance and enhance efficiency over time. Developing an API for cloud-based control of motor parameters, targeting brightness and speed adjustment, follows a systematic methodology to ensure efficiency and reliability. Initially, requirements are analysed, considering functionalities, security, and scalability needs. Design planning involves architecting the API, defining endpoints, data models, and authentication mechanisms. Endpoint definition focuses on specifying functionalities for brightness and speed control, alongside configuration retrieval and error handling. Security implementation includes robust authentication mechanisms like API keys or OAuth. Cloud integration [42–55]. leverages the chosen platform's infrastructure for scalability and reliability. Implementation involves developing endpoints and business logic with suitable frameworks and languages. A comprehensive testing strategy covers unit, integration, and end-to-end tests. Documentation is created detailing endpoints, request/response examples, authentication, and error codes. Deployment entails configuring resources for optimal performance and scalability. Monitoring and logging track usage, performance, and errors. Regular maintenance and updates address security [56], performance, and feature enhancements, ensuring continued reliability and relevance. This methodology ensures the effective design, implementation, and deployment of the API, enabling seamless remote management of motor parameters while prioritizing security, scalability, and reliability [57–63] Developing an API for cloud-based motor parameter control, specifically targeting brightness and speed adjustment, necessitates a methodical approach to ensure both efficacy and robustness. Initial stages involve meticulous requirement analysis to gauge the intricacies of functionalities required, alongside paramount considerations for security and scalability. Design planning forms the backbone of the process, entailing the architectural blueprinting of the API, delineating endpoints, intricately crafting data models, and fortifying access with stringent authentication mechanisms. Endpoint definition emerges as a pivotal phase, where granular functionalities for brightness and speed modulation are delineated, while considering auxiliary aspects like configuration retrieval and error handling. Security

implementation is paramount, requiring the integration of formidable authentication protocols such as API keys or OAuth to fortify against unauthorized access. Cloud integration stands as a linchpin in this process, capitalizing on the infrastructure prowess of chosen platforms to imbue the API with scalability and resilience. Implementation unfolds with precision, necessitating adept development of endpoints and business logic utilizing optimal frameworks and languages. A holistic testing strategy is indispensable, encompassing various tiers such as unit, integration, and end-to-end tests to ascertain functional robustness and mitigate potential vulnerabilities. Thorough documentation becomes imperative, elucidating endpoint nuances, furnishing illustrative request/response instances, and delineating authentication intricacies and error codes for seamless integration. Deployment marks a pivotal transition, demanding meticulous configuration of resources to optimize performance and scalability. Post-deployment, vigilant monitoring and logging are imperative to track usage patterns, assess performance metrics, and promptly identify and rectify anomalies. The process culminates in ongoing maintenance and updates, entailing iterative enhancements to bolster security, streamline performance, and accommodate evolving user requirements. Through this meticulously orchestrated methodology, the API seamlessly empowers remote management of motor parameters, while steadfastly upholding the tenets of security, scalability, and reliability.

Feature extraction

Feature extraction in API development for cloud parameter curation involves identifying and extracting relevant attributes from raw cloud data. Parameters such as network configurations, resource usage metrics, and security settings are identified as key aspects. Data is collected from various sources within the cloud infrastructure, including logs and monitoring tools. Relevant features are selected and transformed into numerical representations, ensuring they are suitable for analysis by machine learning models.

Dimensionality reduction techniques may be applied to handle large datasets or reduce redundancy. Feature engineering may involve creating new features derived from existing ones to capture additional information. Quality assurance checks are conducted to validate the accuracy and consistency of the extracted features.

Documentation of the extracted features is essential for transparency and understanding among users and stakeholders. Finally, the extracted features are integrated into analysis pipelines or machine learning workflows to support decision-making processes and optimize cloud parameter curation.

RESULT

For the API development project aimed at controlling motor speed and LED brightness in a cloud environment:

We've developed an API-driven solution to remotely manage motor speed and LED brightness parameters through cloud-based infrastructure. Leveraging PHP for API development, we ensure efficient communication between the cloud server and client devices. The APIs are designed to receive requests from client applications, process them, and adjust motor speed and LED brightness accordingly.

Using GoDaddy servers for hosting, we ensure reliable and scalable deployment of our APIs. We've configured the server environment to support PHP and provide necessary resources for smooth operation. Deploying the APIs onto the GoDaddy server, we ensure accessibility from anywhere with an internet connection.

Security measures, including SSL encryption and access controls, are implemented to protect data transmission and restrict unauthorized access to the APIs. Regular monitoring and maintenance of the server are conducted to ensure optimal performance and reliability.

With our API-driven solution, users can remotely control motor speed and LED brightness parameters through simple API calls, enabling seamless integration into various applications and IoT devices. This project demonstrates the power of API development for cloud parameter curation in enabling remote control and automation of physical devices for enhanced functionality and efficiency.

DISCUSSION

API development for cloud parameter curation has been a subject of significant research and discussion in recent years. Scholars have explored various dimensions of this topic, including optimization, security, automation, and interoperability. One notable area of focus is the optimization of cloud parameters through APIdriven approaches. Researchers have emphasized the importance of fine-tuning parameters such as network configurations and security policies to enhance performance and reduce costs in cloud environments. Security is another critical aspect addressed in the literature, with discussions on how APIs can enforce access controls, encryption standards, and compliance requirements to mitigate security risks. Automation has also been a key theme, with studies highlighting how APIs enable organizations to automate provisioning, deployment, and scaling of cloud resources, thereby improving operational efficiency and agility. Interoperability is essential for seamless integration across different cloud platforms, and researchers have emphasized the need for standardized APIs to facilitate interoperability. The API development for cloud-based parameter control, specifically targeting brightness and speed adjustment of motors, represents a significant advancement in the field of IoT and remote device management. This innovative solution offers a streamlined approach to controlling motorized devices, catering to a wide range of industrial, commercial, and consumer applications. One of the notable features of this API is its seamless integration with cloud platforms, enabling users to access and control motors from anywhere with internet connectivity. Leveraging the scalability and flexibility of cloud infrastructure, this API ensures reliable performance and accessibility across diverse environments. The design of the API prioritizes simplicity and efficiency, with carefully crafted endpoints that facilitate intuitive interaction with motor parameters. By providing endpoints for adjusting brightness and speed settings, retrieving current configurations, and handling error scenarios, the API offers comprehensive functionality while maintaining ease of use. Security is a paramount consideration in the development of this API, with robust authentication mechanisms in place to safeguard against unauthorized access and ensure data integrity. Through the implementation of secure communication protocols and access controls, users can trust the API to protect sensitive information and prevent unauthorized manipulation of motor settings. Documentation plays a crucial role in the adoption and usability of the API, providing developers with clear guidelines on how to integrate and utilize its functionalities effectively. Well-structured documentation, including detailed endpoint descriptions, example usage scenarios, and error handling guidelines, empowers developers to integrate the API seamlessly into their applications. Overall, the API development for cloud-based motor parameter control represents a significant step forward in enhancing the efficiency, accessibility, and security of remote device management. With its robust features, seamless cloud integration, and comprehensive documentation, this API is poised to revolutionize the way motors are controlled and managed in various domains.

CONCLUSION

API development for cloud parameter curation plays a critical role in optimizing the management of cloud resources, enabling organizations to achieve greater efficiency, flexibility, and security in their cloud environments. Through the development and utilization of APIs, organizations can automate routine tasks, finetune parameters according to specific requirements, and integrate cloud parameter management seamlessly into their DevOps practices. The review of literature highlights the importance of APIs in enabling granular control, scalability, and performance optimization of cloud resources, while also addressing security and compliance requirements. Additionally, emerging trends such as serverless computing, infrastructure as code, and AI-driven automation present exciting opportunities for further innovation in API development for cloud parameter curation. the development of an API tailored for cloud-based control of motor parameters, specifically focusing on brightness and speed adjustment, signifies a significant stride forward in the realm of IoT and remote device management.

This innovative solution encapsulates a comprehensive methodology designed to ensure efficiency, reliability, and scalability, catering to a diverse array of industrial, commercial, and consumer applications. Through meticulous requirement analysis, design planning, and implementation, the API embodies a sophisticated architecture, fortified with robust security mechanisms and seamless cloud integration. By offering a plethora of meticulously crafted endpoints, users can effortlessly interact with the API to manipulate motor parameters, regardless of their physical location. The emphasis on security is paramount, with stringent authentication protocols safeguarding against unauthorized access and ensuring data integrity. Moreover, the cloud integration aspect leverages the scalability and flexibility of cloud infrastructure, empowering users with unparalleled accessibility and reliability. Thorough testing, comprehensive documentation, and vigilant monitoring complement the development process, ensuring the API's functionality, usability, and performance meet the highest standards. Deployment and ongoing maintenance further solidify the API's position as a dependable and indispensable tool for remote motor parameter control. Ultimately, this API represents a culmination of cutting-edge technology and meticulous craftsmanship, poised to revolutionize the way motors are controlled and managed across various domains. With its seamless interface, robust security measures, and unparalleled scalability, the API paves the way for a future where remote motor control is not just a possibility but a reality. In addition to its technical prowess, the API's user-centric design fosters a seamless experience, enabling even non-technical users to interact with motor parameters effortlessly. Its intuitive endpoint structure and comprehensive documentation empower developers to integrate the API seamlessly into their applications, accelerating the adoption and proliferation of remote motor control solutions. Furthermore, the API's adaptability ensures compatibility with a wide range of motor types, from small consumer-grade devices to large industrial machinery, catering to diverse use cases across industries. Whether it's adjusting the brightness of LED lights in smart homes or fine-tuning the speed of conveyor belts in manufacturing plants, the API offers unparalleled versatility and flexibility. Moreover, the API's cloud-native architecture enables real-time updates and enhancements, ensuring it stays abreast of evolving industry standards and user requirements. Continuous optimization and refinement guarantee that the API remains at the forefront of innovation, driving efficiency gains and unlocking new possibilities in motor control technology. Beyond its technical merits, the API embodies a commitment to sustainability by enabling remote management of motors, thereby reducing the need for physical interventions and minimizing energy consumption. This eco-friendly approach aligns with the growing emphasis on sustainability and environmental stewardship across industries.

REFERENCES

1. Patel, S., & Borle, A. (2020). "API-driven Cloud Resource Management: A Review." International Journal of Computer Applications, 177(41), 23-28.
2. Singh, P., & Soni, S. (2019). "API-centric Approaches to Cloud Resource Optimization: A Comprehensive Survey." Journal of Cloud Computing: Advances, Systems and Applications, 8(1), 1-20.
3. Deshpande, H. S. and Karande, K. J. (2014, April). Efficient implementation of AES algorithm on FPGA. In 2014 International Conference on Communication and Signal Processing (pp. 1895-1899). IEEE.
4. Swami, S. S. (2017, August). An efficient FPGA implementation of discrete wavelet transform for image compression. In 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS) (pp. 3385-3389). IEEE.
5. Mane, P. B. (2018). High speed area efficient FPGA implementation of AES algorithm. International Journal of Reconfigurable and Embedded Systems, 7(3), 157-165.
6. Kulkarni, P. R. and Mane, P. B. (2017). Robust invisible watermarking for image authentication. In Emerging Trends in Electrical, Communications and Information Technologies: Proceedings of ICECIT-2015(pp. 193-200). Springer Singapore.
7. Mane, P. B. (2016). Area efficient high-speed FPGA based invisible watermarking for image authentication. Indian journal of Science and Technology.
8. Kashid, M. M., Karande, K. J. (2022, November). IoT-based environmental parameter monitoring using machine learning approach. In Proceedings of the International Conference on Cognitive and Intelligent Computing: ICCIC 2021, Volume 1 (pp. 43-51). Singapore: Springer Nature Singapore.

9. Mane, D. P. (2017). An Efficient implementation of DWT for image compression on reconfigurable platform. International Journal of Control Theory and Applications, 10(15), 1-7.
10. Mandwale, A. J. (2015, January). Different Approaches for Implementation of Viterbi decoder on reconfigurable platform. In 2015 International Conference on Pervasive Computing (ICPC) (pp. 1-4). IEEE.
11. Nagane, U. P. (2021). Moving object detection and tracking using Matlab. Journal of Science and Technology, 6, 86-89.
12. Jadhav, M. M. et al (2021). Machine learning based autonomous fire combat turret. Turkish Journal of Computer and Mathematics Education (TURCOMAT), 12(2), 2372-2381.
13. Mane, D. P. (2019). High throughput and area efficient FPGA implementation of AES algorithm. International Journal of Engineering and Advanced Technology, 8(4).
14. Shinde, G. N. (2021). An approach for robust digital imagewatermarking using DWT-PCA. Journal of Science and Technology, 6(1).
15. Shinde G. (2019). A robust digital image watermarking using DWT-PCA. International Journal of Innovations in Engineering Research and Technology, 6(4), 1-7.
16. Kalyankar, P. A., Thigale, S. P., Chavhan, P. G., and Jadhav, M. M. (2022). Scalable face image retrieval using AESC technique. Journal of Algebraic Statistics, 13(3), 173-176.
17. Kulkarni, P. (2015). Robust invisible digital image watermarking using discrete wavelet transform. International Journal of Engineering Research and Technology (IJERT), 4(01), 139-141.
18. Mane, D. P. (2018). Secure and area efficient implementation of digital image watermarking on reconfigurable platform. International Journal of Innovative Technology and Exploring Engineering, 8(2), 56-61.
19. Deshpande, H. S. and Karande, K. J. (2015, April). Area optimized implementation of AES algorithm on FPGA. In 2015 International Conference on Communications and Signal Processing (ICCSP) (pp. 0010-0014). IEEE.
20. Ghodake, R. G. (2016). Sensor based automatic drip irrigation system. Journal for Research, 2(02).
21. Mane, P. B. (2019). High-Speed area-efficient implementation of AES algorithm on reconfigurable platform. Computer and Network Security, 119.
22. Mane, P. B. (2014, October). Area optimization of cryptographic algorithm on less dense reconfigurable platform. In 2014 International Conference on Smart Structures and Systems (ICSSS) (pp. 86-89). IEEE.
23. Takale, S. (2022). DWT-PCA Based Video Watermarking. Journal of Electronics, Computer Networking and Applied Mathematics (JECNAM) ISSN, 2799-1156.
24. Patale, J. P., Jagadale, A. B., and Pise, A. (2023). A Systematic survey on Estimation of Electrical Vehicle. Journal of Electronics, Computer Networking and Applied Mathematics (JECNAM) ISSN, 2799-1156.
25. Jadhav, M. M., and Seth, M. (2022). Painless machine learning approach to estimate blood glucose level with non-invasive devices. In Artificial Intelligence, Internet of Things (IoT) and Smart Materials for Energy Applications (pp. 83-100). CRC Press.
26. Kondekar, R. P. (2017). Raspberry Pi based voice operated Robot. International Journal of Recent Engineering Research and Development, 2(12), 69-76.
27. Maske, Y., Jagadale, A. B., and Pise, A. C. (2023). Development of BIOBOT System to Assist COVID Patient and Caretakers. European Journal of Molecular and Clinical Medicine, 3472-3480.
28. Maske, Y., Jagadale, M. A., and Pise, M. A. (2021). Implementation of BIOBOT System for COVID Patient and Caretakers Assistant Using IOT. International Journal of Information Technology and;Amp, 30-43.
29. Jadhav, H. M., Mulani, A., and Jadhav, M. M. (2022). Design and development of chatbot based on reinforcement learning. Machine Learning Algorithms for Signal and Image Processing, 219-229.
30. Gadade, B. (2022). Automatic System for Car Health Monitoring. International Journal of Innovations in Engineering Research and Technology, 57-62.
31. Kamble, A., (2022). Google assistant-based device control. Int. J. of Aquatic Science, 13(1), 550-555.

32. Mandwale, A., and Mulani, A. O. (2015, January). Different Approaches forImplementation of Viterbi decoder. In IEEE International Conference on Pervasive Computing (ICPC).
33. Mulani, A. O., Jadhav, M. M., and Seth, M. (2022). Painless Non-invasive blood glucose concentration level estimation using PCA and machine learning. The CRC Book entitled Artificial Intelligence, Internet of Things (IoT) and Smart Materials for Energy Applications. Internet of Things (IoT) and Smart Materials for Energy Applications.
34. Boxey, A., Jadhav, A., Gade, P., Ghanti, P., and Mulani, A. O. (2022). Face Recognition using Raspberry Pi. Journal of Image Processing and Intelligent Remote Sensing (JIPIRS) ISSN 2815-0953.
35. Takale, S., and Mulani, D. A. Video Watermarking System. International Journal for Research in Applied Science and Engineering Technology (IJRASET), 10.
36. Shinde, M. R. S., and Mulani, A. O. (2015). Analysisof Biomedical Image Using Wavelet Transform. International Journal of Innovations in Engineering Research and Technology, 2(7), 1-7.
37. Mandwale, A., and Mulani, A. O. (2014, December). Implementation of Convolutional Encoder and Different Approaches for Viterbi Decoder. In IEEE International Conference on Communications, Signal Processing Computing and Information technologies.
38. Ghodake, R. G., and Mulani, A. O. (2018). Microcontroller Based Automatic Drip Irrigation System. In Techno-Societal 2016: Proceedings of the International Conference on Advanced Technologies for Societal Applications (pp. 109-115). Springer International Publishing.
39. Mulani, A. O., and Mane, P. B. (2016), "Fast and Efficient VLSI Implementation of DWT for Image Compression", International Journal of Control Theory and Applications, 9(41), pp.1006-1011.
40. Shinde, R., and Mulani, A. O. (2015). Analysis of Biomedical Image‖. International Journal on Recent and Innovative trends in technology (IJRITT).
41. Patale, J. P., Jagadale, A. B., Mulani, A. O., and Pise, A. (2022). Python Algorithm to Estimate Range of Electrical Vehicle. Telematique, 7046-7059.
42. Utpat, V. B., Karande, D. K., and Mulani, D. A. Grading of Pomegranate Using Quality Analysis‖. International Journal for Research in Applied Science and Engineering Technology (IJRASET), 10.
43. Mulani, A. O., Jadhav, M. M., and Seth, M. (2022). Painless Non-invasive blood glucose concentration level estimation using PCA and machine learning. The CRC Book entitled Artificial Intelligence, Internet of Things (IoT) and Smart Materials for Energy Applications.
44. Mandwale, A., and Mulani, A. O. (2016). Implementation of High-Speed Viterbi Decoder using FPGA. International Journal of Engineering Research and Technology‖ IJERT.
45. Kambale, A. (2023). HOME AUTOMATION USING GOOGLE ASSISTANT. UGC care approved journal, 32(1).
46. Sawant, R. A., and Mulani, A. O. Automatic PCB Track Design Machine. International Journal of Innovative Science and Research Technology, 7(9).
47. ABHANGRAO, M. R., JADHAV, M. S., GHODKE, M. P., and MULANI, A. Design and Implementation Of 8-bit Vedic Multiplier. JournalNX, 24-26.
48. Seth, M. (2022). Painless Machine learning approach to estimate blood glucose level of Non-Invasive device. Artificial Intelligence, Internet of Things (IoT) and Smart Materials for Energy Applications.
49. Korake, D. M., and Mulani, A. O. (2016). Design of Computer/Laptop Independent Data transfer system from one USB flash drive to another using ARM11 processor. International Journal of Science, Engineering and Technology Research.
50. Mulani, A. O., Birajadar, G., Ivković, N., Salah, B., and Darlis, A. R. (2023). Deep learning-based detection of dermatological diseases using convolutional neural networks and decision trees. Treatment du Signal, 40(6), 2819-2825.
51. Pathan, A. N., Shejal, S. A., Salgar, S. A., Harale, A. D., and Mulani, A. O. (2022). Hand Gesture Controlled Robotic System. Int. J. of Aquatic Science, 13(1), 487-493.
52. Dr. Altaf O. Mulani. (2024). A Comprehensive Survey on Semi-Automatic Solar-Powered Pesticide Sprayers for Farming. Journal of Energy Engineering and Thermodynamics (JEET) ISSN 2815-0945, 4(02), 21–28. https://doi.org/10.55529/jeet.42.21.28

53. Sandeep Kedar and A. O. Mulani (2024), IoT Based Soil, Water and Air Quality Monitoring System for Pomegranate Farming, NATURALISTA CAMPANO, Vol. 28, Issue 1.
54. Bhanudas Gadade, A O Mulani and A.D.Harale (2024). IOT Based Smart School Bus and Student Monitoring System. NATURALISTA CAMPANO, Vol. 28, Issue 1.
55. Anil Dhanawade, A. O Mulani and Anjali. C. Pise. (2024). Smart farming using IOT based Agri BOT. NATURALISTA CAMPANO, Vol. 28, Issue 1.
56. Dr. Shweta Sadanand Salunkhe and Dr. Altaf O. Mulani. (2024). Solar Mount Design Using High-Density Polyethylene. NATURALISTA CAMPANO, Vol. 28, Issue 1.
57. Sarda, M., Deshpande, B., Deo, S., and; Karanjkar, R. (2018). A comparative study on Maslow's theory and Indian Ashrama system.". International Journal of Innovative Technology and Exploring Engineering, 8(2), 48-50.
58. Deo, S., and Deo, S. (2019). Cybersquatting: Threat to domain name. International Journal of Innovative Technology and Exploring Engineering, 8(6), 1432-1434.
59. Shambhavee, H. M. (2019). Cyber-Stalking: Threat to People or Bane to Technology. International Journal on Trend in Scientific Research and Development, 3(2), 350-355.
60. Deo, S., and Deo, D. S. (2019). Domain name and its protection in India. International Journal of Recent Technology and Engineering.
61. Sarda, M., Deshpande, B., Deo, S., and Pathak, M. A. (2018). Intellectual Property and Mechanical Engineering-A Study Emphasizing the Importance of Knowledge Of Intellectual Property Rights Amongst Mechanical Engineers. International Journal of Social Science and Economic Research, 3(12), 6591-6596.
62. Garg, S., & Versteeg, S. (2018). "Automating Cloud Infrastructure Management through API Development: A Systematic Review." Journal of Cloud Computing: Advances, Systems and Applications, 7(1), 1-23.
63. Kumar, A., & Venugopal, K. R. (2017). "Integrating DevOps Practices with Cloud Parameter Curation: A Review of Literature." Journal of Cloud Computing: Advances, Systems and Applications, 10(1).

