

Intelligent Iot Wind Emulation System Based On Real – Time Data Fetching Approach

S.Geetha¹, P.Latha², Aruna J³, Jayaraj V⁴, Pranav G⁵, Janarthanan S⁶

^{1,2,3}Asst. Prof / Department of CSE,V.S.B. Engineering College,Karur, Tamil Nadu

^{4,5,6}Department of CSE,V.S.B. Engineering College,Karur, Tamil Nadu

geethamazhilaran@gmail.com, janarthananjana595@gmail.com, jayaraj.veluchamy@gmail.com,
g.pranavbalaji@gmail.com, janarthananjana595@gmail.com

ABSTRACT: At the heart of the current debate is the fact that wind energy's intrinsic instability is the main barrier to its further growth as a conversion technology. The IoT is the new standard for efficient and intelligent control of renewable energy installations. The Internet of Things, on the other hand, is a fascinating field that offers a lot of room for growth and heavy lifting. By simulating every conceivable operational configuration of wind turbines and configuring the control algorithms and power electronics topologies appropriately, researchers may be able to make wind energy conversion systems more reliable. The proposed wind simulation platform is significantly more efficient than current solutions efficient and has much better traceability, with an emphasis on wind turbines. The technology that simulates wind uses real-time data on wind speeds. By means of an Internet of Things (IoT) cloud application programming interface, the worldwide nodes are able to access both historical and future wind data. The VEE Pro environment is used to mimic the wind emulation system, which is subsequently connected to the FPGA controller and the IoT cloud API. To investigate and enhance wind turbine efficiency, data collected from the Internet of Things is transformed into useful information. In addition, the ESP8266's connection with the host computer is protected by robust security protocols. Then, in a real-time setting, we will analyze the power generating capacity and the status of the system for converting wind energy utilizing the suggested Internet of Things algorithm and its associated hardware prototype.

KEYWORDS: Python, application programming interface, web services, FPGA controller, wind emulator, DC machine, and the Internet of Things.

INTRODUCTION

To satisfy the ever-increasing need for energy, renewable sources are a perfect substitute for fossil fuels. Wind power is booming in popularity throughout the globe as a result of financial incentives and technical developments in recent years. Consequently, producing electricity from wind resources aids in sustainability in the long run [1]. About 2.5% of the world's energy comes from wind power at the moment [2]. Turbines, generators, power electronics circuits, and mechanical structures are all part of the wind energy filament that converts wind into electricity [3]. Researchers are highly motivated to build a plethora of tools, models, and algorithms in the target area due to the significant obstacles in terms of development and innovation [4]. To evaluate the new models and to examine the distinctive performance of the existing models using different variables[5]. An emulator of a wind turbine is a necessary piece of hardware to assess possible innovative designs for wind energy conversion systems. While this simulator does not mimic real wind turbines or natural wind resources, it does recreate both static and dynamic properties in a controlled environment [6, 7]. There are a number of ways to simulate wind turbines in a lab, but the most typical one involves connecting an electric motor to the shaft of the wind generator. The DC motor is the main choice for simulating a wind turbine, which is basic yet has great dynamic features. Its speed and torque control are very straightforward [8], [9]. To acquire the speed and torque that a DC motor needs, a DC chopper is used to control the current flowing through the motor's armature. Since wind speed data is the main input for the wind simulation system, data on wind speeds at various locations across the world is retrieved using an Internet of Things (IoT) cloud API. The data gathered by the DC chopper is used by the controller to generate control pulses. A smart Internet of Things (IoT) system built around a data fetching technique is the main focus of this work's wind simulation system. With the help of the internet of things, we can also remotely monitor the wind emulator system's output parameters.

RELATED WORK

The study [9] details the operation of a prototype wind emulator that is both adaptable and built using a DSPACE controller. There are more publications that have made significant advancements that are covered in the literature [10], [11]. This whole mess is known as the Internet of things. Without the need for a human intermediary, the Internet of Things (IoT) may transmit data using unique identifiers (UIDs). Improved system dependability, efficiency, and sustainability may result from increased information availability made possible by this degree of linkage [12]. [13]. Secure and robust artificial IoT with honeynet-based threat detection and situational awareness was recently shown in a paper [14]. There are a lot of suggestions for enhancements to the protected data transmission that might be made once the Internet of Things (IoT) starts to transmit data [15], [16]. As part of ongoing efforts to reduce electromagnetic pollution, increase energy efficiency, and cut power consumption, the features of the Internet of Things (IoT) communications enable the widespread adoption of inexpensive and easily integrated devices. Using the Internet of Things (IoT), the methods for real-time control and condition monitoring are detailed i [17]. Internet of things devices are useless without application programming interfaces, which are the application enablers. The term "application programming interface" (API) refers to a collection of protocols that enable data transfer between applications or computers [18]. Originally, API was used for data transmission between software applications. Web application programming interface development started in the early 2000s, and since then, the technology has been consistently improved [19]. In order to construct their software, almost all companies use web applications or web services API. By providing access to data that enables objects to be connected, API creates a gateway between the internet and things, revealing possibilities that were previously hidden. IoT cloud API are the points of interaction between an IoT device and the internet and other elements within the network [20]. You have to utilize a service if you want to know the wind speed at a specific location in real time. It offers data that is both readily accessible and updated in response to requests. Unauthorized devices cannot use the API unless the device is authenticated using the API connection string. The gadget makes requests to ascertain the continuous data from the server after connecting to the API. [21].

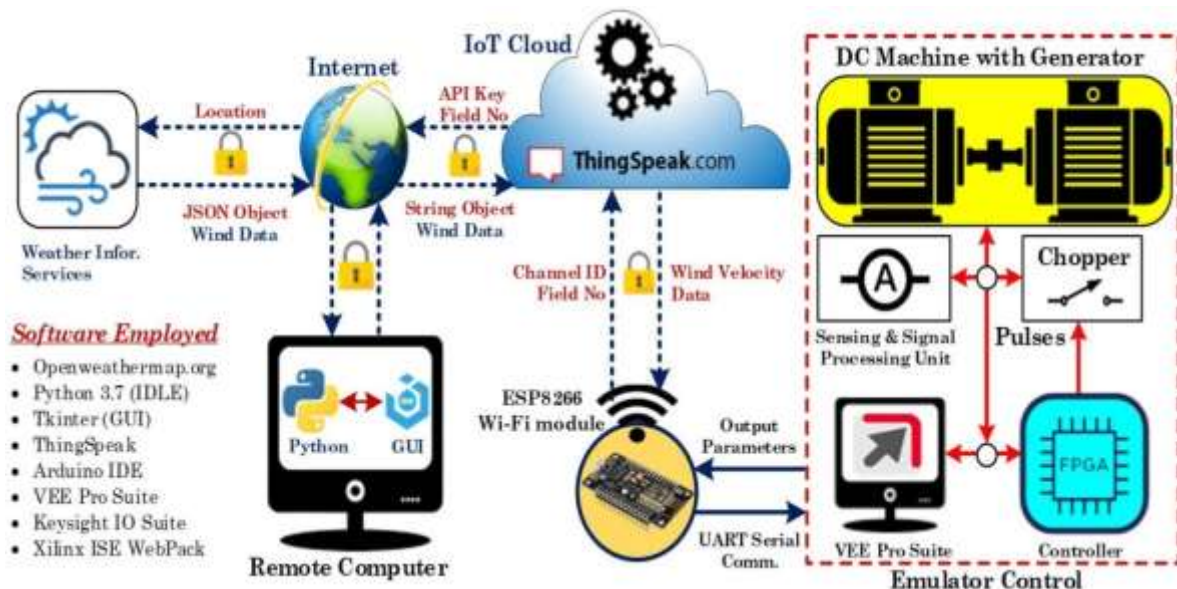


Fig 1. Interpretation of IoT wind emulation system

Internet of Things (IoT) security is the primary emphasis of the literature [13-21], with domain-specific information provided mostly for the power production and automotive industries [18 - 20]. The wind emulation system is currently greatly impacted by the impending difficulty of researchers identifying basic IoT security needs. A separate issue is the interaction between the right Internet of Things (IoT) technology and the protocols used by security services. Researchers in the area of real-time data transmission may use the work's security suggestions, which are not dependent on protocol modifications, to better protect their data from cyberattacks.

A free meteorological data provider called "OpenWeatherMap" is cited in an article as the source for real-time wind data for a specific destination. Both External Markup Language and JavaScript Object Notation are used to display the data, which may be accessed using the OpenWeatherMap API. The ThingSpeak platform receives the stored wind data using the public ThingSpeak API. Tkinter, a Python GUI building package, was used to design the user interface.

An overview of the primary components and their respective communication interfaces that allow the wind emulator to gather real-time wind data for any particular location.

SIGNIFICANCE OF THE ARTICLE

The built-in wind turbine has an Internet of Things (IoT) gadget that shows the electric and mechanical output characteristics. The created system may also show how to analyze the power consumption of duty-cycled Internet of Things nodes and how to synchronize their timing. Here are the key points that the work brings to the table:

- 1) A thorough development of a VEE Pro platform-based wind emulation model.
 - 2) A smart method for retrieving data (used by the wind simulation system) that is built using the IoT Cloud API.
 - 3) A platform that simulates winds in real-time, allowing researchers to test out several downwind versions with controllable pitch angles. In order to extract the various electrical and mechanical metrics (4), a user-interactive GUI is required to obtain wind data from specified places. This data will then be interfaced with the simulator. To simplify the user's prediction power need, a predicted wind speed is read using the OpenWeatherMap service and API URL.
 - 6) The methods for the ESP8266 to communicate with the host computer have been implemented well.
- This paper is fundamentally organized as follows: A thorough mathematical model of the wind emulation system, comprising the wind turbine mode, DC motor model, and PI control, is presented in Section II. In Section III, we can see how the wind emulation system, which is built on the Internet of Things (IoT) cloud API, gets data, combines it, and then simulates the wind. The projected wind simulating system based on the Internet of Things (IoT) is currently presented step-by-step in Section IV. Section V presents the experimental findings of the wind emulator system that is planned to be based on the IoT cloud API. and lastly, section VI finished with future scope.

MODELLING OF WIND EMULATION SYSTEM

The wind turbine and the DC motor with speed control must be meticulously modeled mathematically in order to replicate the wind turbine system. ThinkSpeak is used to import the dynamic-wind-velocity data from the Openweather map into the VEE Pro environment after retrieving it via the IoT cloud API. In a similar manner, a quadrature encoder pulse sensor via the controller's ADC channel provides the real-time rotor speed data. The wind turbine model calculates the turbine reference torque (τ_{ref}) by using the rotor speed and wind velocity as inputs [8]. In order to give the PI controller with error indicators, the obtained τ_{ref} is compared with the actual torque. The DC machine's speed and torque are both managed by the PI controller's output signals, even though Elite helicopter.

A. MODELLING OF WIND EMULATION SYSTEM

The turbine's steady-state properties are taken for granted in the model. Due to the infinite stiffness of the power train, the turbine's momentum and friction coefficient should be integrated.

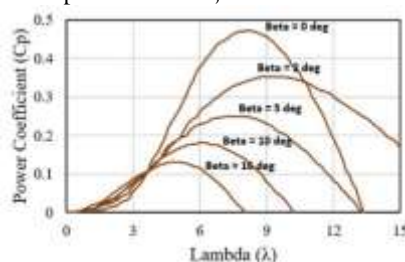


Fig 2. Characteristic curve between C_p and λ

of the generator-connected turbine. The electricity produced by the turbine is supplied as,

$$P_m = c_p(\lambda, \beta) \frac{\rho A}{2} v_{wind}^3 \quad (1)$$

Here, c_p is the power coefficient, V_{wind} is the wind velocity in p.u. of the basis wind speed, and P_m is the mechanical output power/capacity of nominal power. While k_p is the power gain in p.u., the expected wind speed in m/s is assumed to be the baseline wind speed. Power increase is either less than or equal to one. For the expression $c_p(\lambda, \beta)$, a generic expression is used, and it depends on turbine parameters.

$$P_{m_pu} = k_p c_{p_pu} v_{wind_pu}^3 \quad (2)$$

$$c_p(\lambda, \beta) = c_1(c_2/\lambda_i - c_3\beta - c_4)e^{-c_5/\lambda_i} + c_6\lambda \quad (3)$$

With,

$$\frac{1}{\lambda_i} = \frac{1}{\lambda + 0.08\beta} - \frac{0.035}{\beta^3 + 1} \quad (4)$$

Because they are dependent on the kind of wind turbine rotor, the coefficients (3) c_1 through c_6 can change for any turbine design. Furthermore, wind speed and turbine angular velocity also affect parameters c_2 and c_6 . After [22], [23], $c_1=0.5176$, $c_2=116$, $c_3=0.4$, $c_4=5$, $c_5=21$, $c_6=0.0068$, the coefficient between c_1 and c_6 is calculated. Fig. 2 shows the characteristics curve utilizing c_p and λ for various pitch angle β ranges. At $\beta = 0^\circ$, $\lambda = 8.1$, the greatest power coefficient value is 0.48. A nominal value, λ_{nom} , provides the value of λ).

B. MATHEMATICAL MODELLING OF WIND TURBINE

As far as machines that need choppers or rectifiers go, DC motors are the most user-friendly. Since the model used is a DC motor unique excitation model, the value of the field winding flux, ϕ_f , remains constant.

$$\phi_f = k\phi i_f \quad (5)$$

At the terminals of the armature;

$$U_a = R_a i_a + L_a \frac{di_a}{dt} + E \quad (6)$$

E stands for the back emf, $K\phi$ is the torque, and $E = K\phi\Omega_m$ in this instance. The armature current value, however, is given as,

$$\frac{di_a}{dt} = \frac{1}{L_a} (U_a - R_a i_a - K\phi\Omega_m) \quad (7)$$

Since the voltages U_f and R_f on the excitation inductor are fixed, the excitation current I_f is determined to be.,

$$I_f = \frac{U_f}{R_f} \quad (8)$$

This equation controls the DC motor's kinematics.,

$$J \frac{d\Omega_m}{dt} + f\Omega_m = \tau_m - \tau_r \quad (9)$$

The variable f stands for friction, and the variable τ_r for resisting torque. The mechanical torque τ_m also takes on the shape of

$$\tau_m = K\phi i_a \quad (10)$$

OLTF of the DC motor is defined as below.,

$$H_0(p) = \frac{\frac{K\phi}{fR_a + K_\phi^2}}{\left(\frac{JL_a}{fR_a + K_\phi^2}p^2 + \frac{(R_aJ + fL_a)}{fR_a + K_\phi^2}p + 1\right)} \quad (11)$$

Also, it is possible to write it in a form of the electrical constant τ_e and the mechanical constant τ_m . The function as,

$$H_0(p) = \frac{K}{(1 + \tau_e p)(1 + \tau_m p)} \quad (12)$$

Using a PI regulator, the voltage that has to be supplied to the DC motor armature is adjusted until it reaches the same rotational speed as the wind turbine. The PI regulator was chosen because, unlike the integrator, which decays static error, the proportional operates on speed. The regulator supplied by is $R(p)$,

$$R(p) = K_p + \frac{K_i}{p} = K_p \left(\frac{1 + \tau_p}{\tau_p} \right) \quad (13)$$

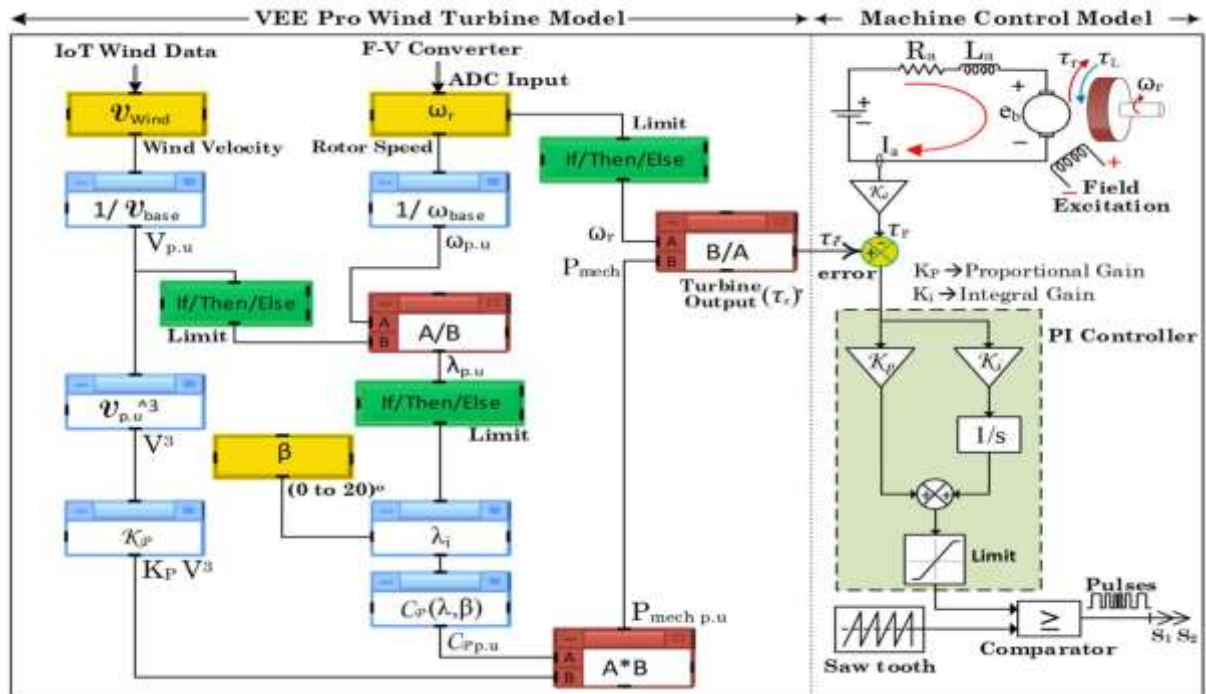


Fig 3. Modelling of wind turbine and DC motor control.

In Figure 3, the PI controller's modeling is shown and explained as,

$$c(t) = k_p e(t) + k_i \int e(t) dt \quad (14)$$

For the comparator, the regulated output (reference voltage) is denoted as $c(t)$, and the equation $e(t) = \tau_r(t) - \tau(t)$ is used. To get the transfer function, we apply the Laplace transform to the above equation, which gives us as,

$$G(s) = \frac{C(s)}{E(s)} = K_p + \frac{K_i}{s} \quad (15)$$

Wind simulation is accomplished with the help of the machine, converter, and controller. In order to implement the intended internet-of-things-based wind simulation system, the laboratory's converter configuration and apparatus have been fine-tuned. In addition, the PI controller is adjusted by using an adaptive trial-and-error tuning technique to choose the W_p and W_i parameters.

IOT CLOUD API BASED DATA FETCHING FOR WIND EMULATION SYSTEM

The kinetic energy of the wind, which is the sum of the motions of billions of air molecules, may be captured by a wind turbine. The concept of wind power seems to be a simple one at first glance: a wind turbine harnesses the energy of the wind to rotate a shaft, which then spins a generator. A wind turbine's rotor blades, gearbox, generator, nacelle, and tower are its essential parts. [24]. The technique may be simple, but there will be substantial material and technical hurdles to overcome when it is put into practice. The primary goals of wind energy engineering are to lengthen the life of systems and components, maximize the collection of wind energy with little maintenance, and reduce equipment costs [25]. The fields of manufacturing process, power electronics, construction, aerodynamics design, materials engineering, endurance, and power electronics will need to work together on this. The basic design of aerodynamic effects on wind turbines is well-documented in the literature [26]–[29]. According to (1), the mechanical power production of a wind turbine is dependent on a number of variables, including wind speed, air density, turbine rotor speed, and swept area. This team is crucial for developing a wind turbine simulator. An electric motor attached directly to the wind generator's shaft is the standard for laboratory wind turbine modeling [30]. Since the armature current of DC motors is regulated in accordance to the torque that the machine generates, DC motor control is both straightforward and has strong dynamic properties. Because of this, a wind turbine simulator with DC motors would be perfect. In this work, a DC motor is powered via a class-A DC-to-DC converter. The author constructed an experimental setup using a DC motor to mimic the static and dynamic characteristics of a genuine wind energy conversion system. With the help of open source servers, visualization dashboards, online development tools, common microelectronics, and light communication protocols, the Internet of Things (IoT) becomes a viable alternative to traditional remote monitoring. Data retrieval, API design, web services, and the actual wind emulation system are all covered in this section, along with the many other processes needed in installing an IoT-based system.

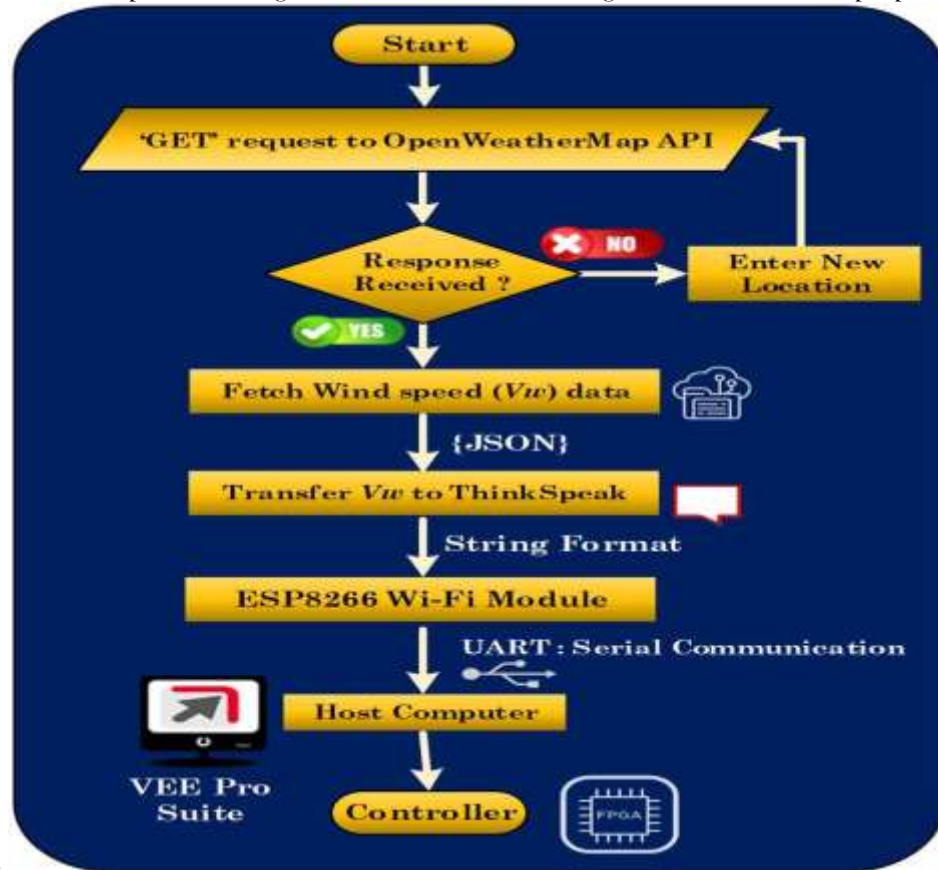
A. IOT WIND DATA FETCHING UNIT

The wind speed information is quickly retrieved from the user's selected global location via the Internet of Things data retrieval technology. The wind data fetching unit is components—the gateway, application programming interface (API), web service, and Internet of Things (IoT) cloud—are shown below.

1) INTERNET OF THINGS

A network of interconnected computing devices, services, and objects that enable intelligent two-way communication between humans and their physical surroundings is known as the "Internet of Things" [31]. The internet of things (IoT) relies on the cloud, which provides specialized services across many different types of applications. You may classify some of the leading IoT cloud providers according to the areas in which their services are most suited, because they have all joined the market to provide IoT-based services. This is seen in the work [32], in which the authors use an Internet of Things (IoT) cloud system to maintain an appropriate temperature and moisture level for rice seedlings after they have passed the critical development phases. Motivated to suggest an Internet of Things (IoT) based system to monitor engine oil level and impurity detection in [33]. Our Internet of Things (IoT) fetching system makes use of a Wi-Fi-based network layer. The following is an illustration of how this design may obtain real-time

wind data from a weather API provider: Figure 4. Wind data fetching is the name of the proposed



system's ending node.

Figure 4 shows the data exchange process between the cloud and the WDFUs, or controller units. This setup uses 2.4 GHz (802.11 n) to analyze location data before sending it to a Wi-Fi gateway. There is just one step in the WLAN implementation between a WDFU and a Wi-Fi gateway. The Wi-Fi access points connect to the internet via a central IoT-WDFU server, which in turn connects to the network switch.

2) ARCHITECTURE OF APPLICATION PROGRAMMING INTERFACE (API)

There are four distinct kinds of application programming interfaces (APIs) on the web, classified by the level of access they grant: internal, partner, open, and composite [34]. Due to the fact that we need access to information that is publicly accessible without charge, one of these open APIs was used to generate this work. Goal 3.2 Everyone, both within and outside the company, has public access to these kinds of open APIs. There are also three type of API designs and protocols, like;. APIs for representational state transfer (REST) allow the server to only respond to requests made by clients; RPC APIs allow users to access specific web service procedures; SOAP APIs allow computers running different operating systems to communicate with one another; The number 35. One example of a RESTful design is the OpenWeatherMap API. Because of their statelessness, REST APIs do not save any information or changes made between requests. Both the XML and JSON data formats are compatible with REST APIs. But in this case, we're receiving JSON object responses from the REST API endpoint.

3) WEATHER WEB SERVICES

A weather API, OpenWeatherMap provides current conditions for any given location.

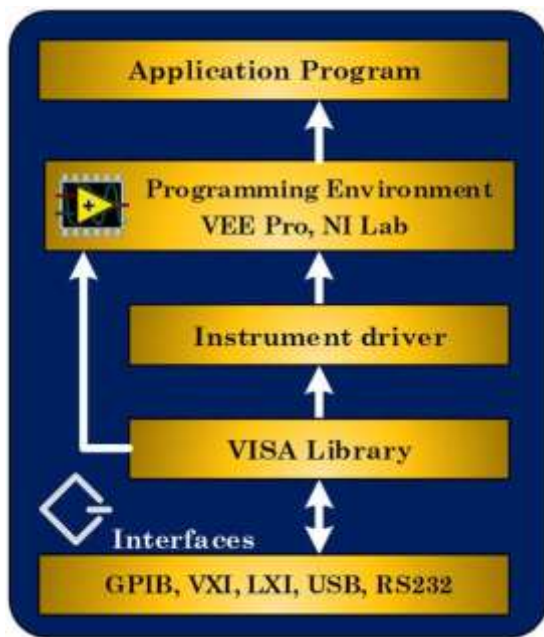


Fig 5. Communication interface between I/O and Application.

During every epidemic, it is possible to gather and transmit 233 possible instances of malaria, 233 possible cases of TB, and prospective rainfall values to epidemiologists, actions of legalizers, and analysts. The data from equipment and sensors like thermometers, hygrometers, and anemometers—which measure wind speed and direction—are sent to the servers of OpenWeatherMap by weather stations located all over the globe [36]. All of the data collected is saved in the OpenWeatherMap database and may be accessed using their website's API. Information retrieved from the openweather map is available in XML and JSON formats. One of the meteorological data parameters in JSON Data is the name of the city or state in question [37]. For reasons of convenience, data is utilized in the JSON format. The data connection path between the cloud and the controller is shown in Figure 4. After a location (city, state, or village) is selected, an API request will be made using the determined wind speed in m/s. In order to request any kind of data via the "API," one must utilize the "GET" HTTP method. The data is retrieved using the 'GET' function when the city name and API key are given to the URL. Tkinter, a Python toolkit for creating user interfaces, was used to develop the graphical user interface.

B. INTEGRATING UNIT

For the purpose of this proposal, the wind speed data will be monitored using ThingSpeak, a cloud platform. A public platform for Internet of Things data is ThingSpeak (<https://thingspeak.com>). Thanks to ThingSpeak's Open API, we can gather and analyze data in real-time. When integrating ThingSpeak with a view toward visualization, monitoring, and data storage, other popular third-party platforms like Matlab, ioBridge, and Arduino might also be considered. The number 38. Users may get data alerts via tweets and other alerting ways after creating channels in 'ThingSpeak.' Data can then be implemented or processed and displayed in Matlab [39]. Figure 5 shows the ThingSpeak cloud channel field receiving real-time wind speed data from the Openweathermap website API. Transferring data security from web-based weather API providers to the host PC is a major factor in the proposed project. There are five main components that make up the system of the proposed work. The data flow starts with a remote computer requesting windspeed for a specified place from the OpenWeatherMap Service using their open API. Afterwards, you may expect a JSON object containing the windspeed from the service. The data is then sent to ThingSpeak.com, a cloud analytics integration platform. The data about the wind speed is retrieved by the ESP8266 Wi-Fi module from ThingSpeak and sent to the host computer over UART-Serial. Using the newly-developed Python interface to access data stored in the cloud and transfer it to a distant PC. In the near future, there will be a remote computing environment that uses Python programs

to gather wind data. The code snippet displays the created wind sift that was acquired from the weather API provider. Using the python environment, the app is built to transfer wind data to the cloud. The host system's Wi-Fi module will communicate with the cloud platform to get wind speed data. The FPGA development board is controlled by sending it to the VEE Pro IDE. The Wi-Fi module is meant to transmit the data of wind speed to the VEE pro software-generated wind turbine model. A collection of input/output (I/O) libraries, application programming interfaces (APIs), and utility apps are available from Keysight. Connecting instruments in various development environments is made easy with the Keysight I/O libraries package. This includes test instruments from several OEMs that are compatible with USB, LAN, PXI, AXIe, and VXI. Figure 5 shows the application's and I/O's communication flow interface.

C. EMULATING UNIT

The purpose of this emulation unit is to make it easier to use characteristic curves to analyze the performance of the wind energy generating system. When it comes to the wind simulation system's control procedure, two options are being considered. the section; Wind energy conversion system analysis may be done in two ways: (i) in a fixed wind mode, where the system parameters are varied for a given wind velocity, or (ii) in a variable wind mode, when the design parameters are varied for a specified wind velocity. Two operational modes are available to the user: fixed wind velocity and variable wind velocity. When using fixed mode, the user must manually enter a wind speed between 8 and 14 meters per second. The wind speeds will be retrieved straight from the clouds if the user selects the variable mode. This model may be used to get the output profile of several factors including speed, torque, and power. When the system detects an external interrupt signal, it will deactivate the chopper's input power and roll back the load.

EXPERIMENTAL SETUP AND GUI INTEGRATION WITH FORECASTING AND SECURITY SERVICES

This section provides a detailed description of the method used to apply the experimental setup and create the graphical user interface.

A. EXPERIMENTAL ARRANGEMENT

The system is made up of a test machine with a 3 kVA AC generator and a primary mover with a 3.7 kW DC driving arrangement. A load cell and a quadrature encoder pulse (QEP) sensor are also included into the machine configuration to measure torque and speed, respectively. The ratings for the machine, converter, and controller are shown in Table 1. The proposed method employs a chopper converter to drive and excite the DC motor using an external excitation mechanism. The converter's power circuit consists of a three-phase diode bridge rectifier coupled to inductors and IGBT switches by DC link capacitors. The converter is also linked to circuits that condition signals, sensors, and drivers. The electrical values of the converter are measured by a multifunction meter (MFM), sensed, and returned to the controller. Pulses for switching are produced by the Xilinx FPGA Spartan 6LX25 controller based on sensor response and data input from the Internet of Things cloud. In particular, controllers that use the VEE (Visual Engineering Environment) Pro software are intended to be equipped with the maximum power point control approach. Agilent Technologies' VEEPro is a graphical data flow programming environment that may be used to automate testing, measurement, data analysis, and reporting. It is compatible with all common measuring and testing devices. A typical application programming interface (API) used for interaction between instruments and computers is the virtual instrument software architecture (VISA). Instrument and software application connection, in addition to VISA architecture. Specifications for communicating with resources are included in the VISA standard, which applies across T&M-specific I/O interfaces like GPIB and VXI. By connecting the Internet of Things (IoT) to the FPGA controller, we can see the results of the wind simulation system and get real-time data on the wind speed at the chosen location. The power meter (WT 333E) is used for calibrating the emulation system's output by measuring the electrical quantities of the AC generator's output. As part of its comprehensive, integrated Wi-Fi networking solution, the ESP8266 may either run alone or delegate all of its Wi-Fi

networking tasks to an additional application processor. It is not about making kits to access those GPIOs on the ESP8266; rather, it finds widespread use in a wide variety of Internet of Things applications. Since the ESP8266 is the only "application processor" in this setup, it boots from an external source as soon as these two components turn on, rather than using its internal SD card. When connected to a computer, the ESP8266's one serial connection allows for data transfer between the two devices. "Universal Serial Bus" (USB) refers to a kind of serial port. The ESP8266's onboard hardware has two digital pins, GPIO 0 and GPIO 1, that may be used in applications that need serial connectivity to other devices, not limited to computers.

B. DEVELOPMENT OF GRAPHICAL USER INTERFACE APPLICATION

An interactive application is built using Tkinter, a Python library. Several node points have been set up in different parts of the world for this application. During the navigation and selection process, data on the current wind speed and time coordinates are collected from the relevant global nodes. A new window appears every time the user clicks the wind emulator icon; the output parameters of the wind emulator, such as the output power, speed, and torque, are displayed in this window. They all have a 15-second refresh time by default. However, the user must click the refresh button in order for the data to be updated anytime the user asks it.

C. FORECASTED WIND DATA

Predicting the weather accurately is of the utmost importance since it is so unpredictable. The user may change the wind turbine to conserve electricity by knowing the estimated windspeed for the following days, which will reveal the power requirement. Here you may find the OpenWeatherMap-retrieved forecasted wind data. With the longitude and latitude of the location supplied in the API URL, the OpenWeatherMap service may also offer weather predictions for any fixed site. We use a for loop to loop over the 'daily' key to collect the wind speed number for every day in the forecast.

D. SECURITY SERVICES

Similar to other online APIs, ThingSpeak encrypts its information by requiring users to authenticate with a unique API key. DDoS assaults will no longer be possible because to API key usage. ThingSpeak API also makes use of API caching to boost application speed. Consequently, the data may be protected from the wind emulation and the most typical attacks on APIs can be mitigated using these approaches. Another aspect that has been protected is the communication between the host computer and the ESP8266 module. An integrated Wi-Fi system with a complete TCP/IP protocol stack is the ESP8266. The data is AES (Advanced Encryption Standard) encrypted to make sure it's safe to send via the ESP8266. It's possible to decode the message, but it's not easy. First, the data is transformed from strings to binary using UTF-8. Then, the binary data is encrypted using AES and last, it is translated to base64 encoding. The original message may be recovered by using the host computer's decrypt functionality, which converts the base64 payload into binary ciphertext.

V. WIND TURBINE EMULATION AND IOT RESULTS

Here we provide the outcomes of the experiments conducted on the suggested wind emulator system. The use of real-time wind data allows for the analysis of wind turbines with novel designs and control.

A. MAXIMUM POWER POINT OF WIND EMULATION SYSTEM

Power curves are useful for evaluating the wind turbine's specs. This involves selecting the fixed wind velocity mode, which means that the load torque (τ_L) may range from zero to maximum, while the wind velocity (V_w) remains constant. Additionally, the pitch angle (β) remains constant during this procedure. Experimenting with other materials involves following the same approach but varying the wind velocities (ranging from 8 to 14 m/sec) at 2-m/sec intervals and the pitch angles (ranging from 0 to 6 degrees) at 1-degree intervals. The data is presented in table 2, which shows the highest power point for various wind and pitch velocities.

B. PARAMETER VARIATION WITH CHANGE IN WIND VELOCITY

Power curves are useful for evaluating the wind turbine's specs. As mentioned earlier, in order to achieve the highest power point of the suggested system, the control method of the wind simulating system is used. This involves selecting the fixed wind velocity mode, which means that the load torque (τ_L) may range from zero to maximum, while the wind velocity (V_w) remains constant. Additionally, the pitch angle (β) remains constant during this procedure. Experimenting with other materials involves following the same approach but varying the wind velocities (ranging from 8 to 14 m/sec) at 2-m/sec intervals and the pitch angles (ranging from 0 to 6 degrees) at 1-degree intervals. The data is presented in table 2, which shows the highest power point for various wind and pitch velocities. The findings indicate that wind turbine output power is directly impacted by wind speed and pitch angle. According to the data, the reference speed in the controller has to be changed in order to reach the maximum power point. This facilitates the development of different control algorithms that strive for the highest possible power output.

C. IOT OUTPUT RESULTS

ThingSpeak is the platform used to monitor wind speed data. ThingSpeak is a publicly available cloud-based IoT data platform (<https://thingspeak.com>). Through an Open API, Thingspeak enables real-time data collection, analysis, and actuation. The term "Channel," which contains the data, location, and status components for different detected data, is the most important component of ThingSpeak's capabilities. Data may be written, analyzed, and displayed using Matlab once the "ThingSpeak" channel has been created. Twitter and others can reply to alarms by sharing the data. Secure data transfers from web-based weather API providers to the host PC are the aim of the proposed effort. The data is sent at a sample rate of 15 seconds from the emulator to the ThingSpeak server via the remote computer over a 10-minute period. For the ThingSpeak API to write the channel fields, a particular Write the key.

CONCLUSION WITH FUTURE SCOPE

Wind turbines' dependable functioning and constant control are crucial to the wind power industry's rapid growth. The suggested method ensures that the Internet of Things (IoT) wind simulator runs on real-time wind speeds from different parts of the world. A hardware prototype has also been created for the purpose of testing the embedded algorithms and controller. Findings demonstrate that the suggested Internet of Things (IoT) based wind emulation system, which both gets cloud-based wind data and transmits the emulator's output parameters to the cloud, is very efficient. In addition to being able to assess the wind emulator's power usage, the system also has predictive power in terms of wind speed. In addition, data sent by the ESP8266 is secured using AES for safe transmission inside the system. The outcomes of the experiments and the simulations are determined to be highly concordant. By keeping an eye on the weather, authorized remote users may quickly and accurately manage the wind energy conversion system's assets to prevent any potential problems. In addition to producing changeable wind energy equivalent, the suggested system is an intelligent decentralized application that has the same mechanical properties as a real wind turbine. Since more complex and preventative measures are required, this study will expand its focus to include improving the security of the Internet of Things (IoT). In order to identify specific patterns of assault in IoT networks, a signature-based intrusion detection system may be used. Data on newly discovered malware assaults will be gathered using this. After the data on the new virus has been gathered, it will be simpler to tell whether an attack is possible. Additional measures include integrating ML technologies with complicated event processing to identify potentially harmful Internet of Things (IoT) devices. Using a private API as opposed to a public one ensures more security. AI methods will be used in the context of wind velocity predictions. Users can better comprehend the power requirements when a wind speed prediction has been set up. Using API, one may get the wind speed and direction at a certain location. Use of private APIs in banking transactions, home IoT devices, and other similar contexts is limited to a select group of individuals or internal organizations. A randomly generated API key is sent together with a secure message and secret key using SHA 256. Client and service will both have access to this private key. A 128-bit or 256-bit secret key encoded in base 64 should be generated using a GUID generator. This is because it is theoretically possible to generate one-of-a-kind

hash values using an n-set of rounds. The opposite is true: energy waste may be reduced by connecting wind turbine outputs to a shared bus. A real-time wind turbine modeling system with regenerative control made possible by the Internet of Things (IoT) can increase safety while reducing power waste. The yaw angle may also be examined in connection with its variation, much like the pitch angle control.

NOMENCLATURE

Symbol	Explanation
τ^*	Turbine Reference Torque
P_m	Mechanical Output Power
c_p	Power Coefficient
v_{wind}	Wind velocity
k_p	Power gain
β	Pitch angle
φ_f	Field winding flux
E	Back emf
U_f, R_f	Constants
I_f	Excitation current
f	Friction
τ_r	Resistant torque
τ_m	Mechanical torque
$H_o(p)$	Open-loop transfer function(OLTF)
τ_e	Electrical constant
R(p)	Regulator

A	Swept area
ρ	Air density
kW	Kilowatt
kVA	Kilovolt-Ampere
Nm	Newton-meter
Hp	Horsepower
p.u	Per-unit
τ_L	Load torque
rpm	Revolution per minute
GHz	Gigahertz

ACKNOWLEDGMENT

This study at the Advanced Drives lab would not have been possible without the 'VIT SEED GRANT,' which the authors are grateful to the VIT in Vellore, India, for providing.

REFERENCES

- [1] Z. Xu, J. Wei, S. Zhang, Z. Liu, X. Chen, Q. Yan, and J. Guo, "A state-of-the-art review of the vibration and noise of wind turbine drivetrains," *Sustain. Energy Technol. Assessments*, vol. 48, Dec. 2021, Art. no. 101629, doi: 10.1016/j.seta.2021.101629.
- [2] S. Dawn, P. K. Tiwari, A. K. Goswami, A. K. Singh, and R. Panda, "Wind power: Existing status, achievements and government's initiative towards renewable power dominating India," *Energy Strategy Rev.*, vol. 23, pp. 178–199, Jan. 2019, doi: 10.1016/j.esr.2019.01.002.
- [3] W. Cao, Y. Xie, and Z. Tan, "Wind turbine generator technologies," in *Advances in Wind Power*. London, U.K.: Intechopen, 2012. [Online]. Available: <https://www.intechopen.com/chapters/38933>, doi: 10.5772/51780.
- [4] J. D. M. De Kooning, A. E. Samani, S. De Zutter, J. De Maeyer, and L. Vandeveld, "Techno-economic optimisation of small wind turbines using co-design on a parametrised model," *Sustain. Energy Technol. Assessments*, vol. 45, Jun. 2021, Art. no. 101165, doi: 10.1016/j.seta.2021.101165.
- [5] V. Sohoni, S. C. Gupta, and R. K. Nema, "A critical review on wind turbine power curve modelling techniques and their applications in wind based energy systems," *J. Energy*, vol. 2016, pp. 1–18, Jun. 2016, doi: 10.1155/2016/8519785.
- [6] K. S. Ajirilo, P. H. Tari, K. Gharali, and M. Zandi, "Development of a wind turbine simulator to design and test micro HAWTs," *Sustain. Energy Technol. Assessments*, vol. 43, Feb. 2021, Art. no. 100900, doi: 10.1016/j.seta.2020.100900.
- [7] D. Zouheyr, B. Lotfi, and B. Abdelmadjid, "Improved hardware implementation of a TSR based MPPT algorithm for a low cost connected wind turbine emulator under unbalanced wind speeds," *Energy*, vol. 232, Oct. 2021, Art. no. 121039, doi: 10.1016/j.energy.2021.121039.
- [8] Dr. S. Prabakaran, V. Shangamithra, G. Sowmiya, R. Suruthi, "Advanced Smart Inventory Management System Using IoT," 2023 IJCRT, Volume 11, DOI: 4 April 2023, ISSN: 2320-2882
- [9] R. Teodorescu, F. Iov, and F. Blaabjerg, "Flexible development and test system for 11 kW wind turbine," in *Proc. IEEE 34th Annu. Conf. Power Electron. Spec.*, Jun. 2003, pp. 67–72, doi: 10.1109/PESC.2003.1218275.
- [10] H. M. Kojabadi, L. Chang, and T. Boutot, "Development of a novel wind turbine simulator for wind energy conversion systems using an inverter-controlled induction motor," *IEEE Trans. Energy Convers.*, vol. 19, no. 3, pp. 547–552, Sep. 2004, doi: 10.1109/TEC.2004.832070.
- [11] B. Rabelo, W. Hofmann, and M. Gluck, "Emulation of the static and dynamic behaviour of a wind-turbine with a DC-machine drive," in *Proc. IEEE 35th Annu. Power Electron. Spec. Conf.*, Jun. 2004, pp. 2107–2112, doi: 10.1109/PESC.2004.1355443.
- [12] P. P. Ray, "A survey on Internet of Things architectures," *J. King Saud Univ., Comput. Inf. Sci.*, vol. 30, no. 3, pp. 291–319, Jul. 2018, doi: 10.1016/j.jksuci.2016.10.003.
- [13] S. Kumar, P. Tiwari, and M. Zymbler, "Internet of Things is a revolutionary approach for future technology enhancement: A review," *J. Big Data*, vol. 6, no. 1, pp. 1–21, Dec. 2019, doi: 10.1186/s40537-019-0268-2.
- [14] L. Tan, K. Yu, F. Ming, X. Cheng, and G. Srivastava, "Secure and resilient artificial intelligence of things: A HoneyNet approach for threat detection and situational awareness," *IEEE Consum. Electron. Mag.*, vol. 11, no. 3, pp. 69–78, May 2022, doi: 10.1109/MCE.2021.3081874.
- [15] Y. Djenouri, G. Srivastava, A. Belhadi, and J. C. Lin, "Intelligent blockchain management for distributed knowledge graphs in IoT 5G environments," *Trans. Emerg. Telecommun. Technol.*, Jul. 2021, Art. no. e4332, doi: 10.1002/ett.4332.
- [16] L. Malina, P. Dzurenda, S. Ricci, J. Hajny, G. Srivastava, R. Matulevicius, A.-A.-O. Affia, M. Laurent, N. H. Sultan, and Q. Tang, "Post-quantum era privacy protection for intelligent infrastructures," *IEEE Access*, vol. 9, pp. 36038–36077, 2021, doi: 10.1109/ACCESS.2021.3062201.
- [17] R. R. Singh, S. M. Yash, S. C. Shubham, V. Indragandhi, V. Vijayakumar, P. Saravanan, and V. Subramaniaswamy, "IoT embedded cloud-based intelligent power quality monitoring system for industrial drive application," *Future Gener. Comput. Syst.*, vol. 112, pp. 884–898, Nov. 2020, doi: 10.1016/j.future.2020.06.032.
- [18] A. B. Letaifa, "SSIM and ML based QoE enhancement approach in SDN context," in *Advances in Computers*, vol. 114, The Netherlands, 2019, ch. 4, pp. 151–196.
- [19] J. Kopecký, P. Fremantle, and R. Boakes, "A history and future of web APIs," *Inf. Technol.*, vol. 56, no. 3, pp. 90–97, Jun. 2014, doi: 10.1515/itit2013-1035.
- [20] W. Alam, D. Sarma, R. J. Chakma, M. J. Alam, and S. Hossain, "Internet of Things based smart vending machine using digital payment system," *Indonesian J. Electr. Eng. Informat.*, vol. 9, no. 3, pp. 719–731, Aug. 2021, doi: 10.52549/V9I3.3133.
- [21] L. C. C. De Biase, P. C. Calcina-Ccori, G. Fedrechski, G. M. Duarte, P. S. S. Rangel, and M. K. Zuffo, "Swarm economy: A model for transactions in a distributed and organic IoT platform," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4561–4572, Jun. 2019, doi: 10.1109/JIOT.2018.2886069.
- [22] P. Anderson and A. Bose, "Stability simulation of wind turbine systems," *IEEE Trans. Power App. Syst.*, vol. PAS-102, no. 12, pp. 3791–3795, Dec. 1983, doi: 10.1109/TPAS.1983.317873.
- [23] B. Amlang, D. Arsurdis, W. Leonhard, W. Vollstedt, and K. Wefelmeier, "Elektrische energieverorgung mit windkraftanlagen," *BMFT Forschungsvorhaben*, Brunswick, Germany, Final Rep. 032-8265-B, 1992.

- [24] M. G. Molina and P. E. Mercado, "Modelling and control design of pitchcontrolled variable speed wind turbines," in *Wind Turbines*. London, U.K.: IntechOpen, 2011. [Online]. Available: <https://www.intechopen.com/chapters/14810>, doi: 10.5772/15880.
- [25] A. Elia, M. Taylor, B. Ó Gallachóir, and F. Rogan, "Wind turbine cost reduction: A detailed bottom-up analysis of innovation drivers," *Energy Policy*, vol. 147, Dec. 2020, Art. no. 111912, doi: 10.1016/j.enpol.2020.111912.
- [26] P. J. Schubel and R. J. Crossley, "Wind turbine blade design," *Energies*, vol. 5, no. 9, pp. 3425–3449, Sep. 2012, doi: 10.3390/en5093425.
- [27] J. J. Chattot, "Wind turbine aerodynamics: Analysis and design," *Int. J. Aerodyn.*, vol. 1, nos. 3–4, p. 404, 2011, doi: 10.1504/ijad.2011.038853.
- [28] T. Mojtaba, S. Tahmine, R. Kiana, and A. Pouria, "Aerodynamic optimal design of wind turbine blades using genetic algorithm," *Energy Equip. Syst.*, vol. 2, no. 2, pp. 185–193, 2014.
- [29] H. Snel, "Review of aerodynamics for wind turbines," *Wind Energy*, vol. 6, no. 3, pp. 203–211, Jul. 2003, doi: 10.1002/we.97.
- [30] K. K. M. S. Kariyawasam, K. K. N. P. Karunarathna, R. M. A. Karunarathne, M. P. D. S. C. Kularathne, and K. T. M. U. Hemapala, "Design and development of a wind turbine simulator using a separately excited DC motor," *Smart Grid Renew. Energy*, vol. 4, no. 3, pp. 259–265, 2013, doi: 10.4236/sgre.2013.43031.
- [31] H. Xu, W. Yu, D. Griffith, and N. Golmie, "A survey on industrial Internet of Things: A cyber-physical systems perspective," *IEEE Access*, vol. 6, pp. 78238–78259, 2018, doi: 10.1109/ACCESS.2018.2884906.
- [32] P. P. Ray, "A survey of IoT cloud platforms," *Futur. Comput. Informat. J.*, vol. 1, nos. 1–2, pp. 35–46, 2016, doi: 10.1016/j.fcij.2017.02.001.
- [33] A. Pantelopoulous and N. G. Bourbakis, "A survey on wearable sensorbased systems for health monitoring and prognosis," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 40, no. 1, pp. 1–12, Jan. 2010, doi: 10.1109/TSMCC.2009.2032660.
- [34] H. Elhoone, T. Zhang, M. Anwar, and S. Desai, "Cyber-based design for additive manufacturing using artificial neural networks for industry 4.0," *Int. J. Prod. Res.*, vol. 58, no. 9, pp. 2841–2861, May 2020, doi: 10.1080/00207543.2019.1671627.
- [35] K. Wagh and R. Thool, "A comparative study of SOAP vs REST web services provisioning techniques for mobile host," *J. Inf. Eng. Appl.*, vol. 2, no. 2, pp. 12–17, 2012.
- [36] M. G. de Vos, W. Hazeleger, D. Bari, J. Behrens, S. Bendoukha, I. Garcia-Marti, R. van Haren, S. E. Haupt, R. Hut, F. Jansson, A. Mueller, P. Neilley, G. van den Oord, I. Pelupessy, P. Ruti, M. G. Schultz, and J. Walton, "Open weather and climate science in the digital era," *Geosci. Commun.*, vol. 3, no. 2, pp. 191–201, Aug. 2020, doi: 10.5194/gc-3-191-2020.
- [37] P. Bourhis, J. L. Reutter, and D. Vrgoč, "JSON: Data model and query languages," *Inf. Syst.*, vol. 89, Mar. 2020, Art. no. 101478, doi: 10.1016/j.is.2019.101478.
- [38] A. Giessmann and C. Legner, "Designing business models for cloud platforms," *Inf. Syst. J.*, vol. 26, no. 5, pp. 551–579, Sep. 2016, doi: 10.1111/isj.12107.
- [39] A. A. H. Mohamad, N. K. Jumaa, and S. H. Majeed, "ThingSpeak cloud computing platform based ECG diagnose system," *Int. J. Comput. Digit. Syst.*, vol. 8, no. 1, pp. 11–18, Jan. 2019, doi: 10.12785/ijcds/080102.