Design of Electronic Volume Corrector (EVC) Using Web-Based Microcontroller

Aliefa Indira Nugraheni[1,*], Roni Heru Triyanto[2], Suka Handaja[3]
[1,2,3] Program Studi Teknik Instrumentasi Kilang, Politeknik Energi dan Mineral Akamigas, Cepu Jawa Tengah, Indonesia 55312

[*]Corresponding Author: aliefaindira077@gmail.com (Aliefa Indira Nugraheni)

Info Artikel

Histori Artikel:
Diterima berkas: 5 Mei 2024
Direvisi: 20 Mei 2024
Disetujui terbit 1 Juni 2024

Keyword:
Electronic volume corrector
Web server
AGA 7
Kelton

ABSTRACT

In the metering industry, an electronic volume corrector (EVC) is essential for collecting measurement data and compensating for standard readings, as mandated by the directorate general of oil and gas. Over time, advancements in data delivery systems have increased the demand for data accessibility and affordability. To address this, a prototype EVC with web-based reading capabilities was developed, allowing users to access and manage data efficiently without needing to visit a physical location. This research aims to implement the EVC not only to maintain accurate corrections but also to enhance ease of access for users in the metering industry. The web server is accessible only to authorized users in the database. Testing using the REST API protocol demonstrated appropriate accuracy for both local readings and those on the web server. In addition to being sent to the web server, data was also sent via Telegram. The average errors produced were 0.0132%, 0.0052%, and 0.0690% compared to the AGA 7 standard at Kelton. This indicates that the EVC prototype has a response system that meets a tolerance of ±1%, matches the capabilities of standard reading correctors, and facilitates data accessibility.

1. INTRODUCTION

The emergence of the term Industrial Revolution 4.0 and Japan's vision of Society 5.0, which combines technology and social aspects, has led to an increase in people's desire for convenience and instant gratification[1]. This encourages research and development, especially in technology (super smart society), to make human life easier[2]. This industrial revolution has had a major impact in various fields, including automation and digitalization of the work sector with the use of Artificial Intelligence, Big Data, and the Internet of Things (IoT)[3],[4].

Digitalization simplifies human tasks through interconnection and transparency of information. According to Angela Merkel, the basis of IoT is the combination of digital technology, the internet, and conventional industry[5]. Kang Ermann added that the industrial revolution involves the integration of Cyber-Physical Systems (CPS) and IoT[6]. Indonesia has prepared a national priority roadmap, including the development of digital infrastructure for energy security with a focus on availability, accessibility, affordability, and acceptability[7]. The main aspect related to the Industrial Revolution is accessibility, which allows easy access to energy sources through network infrastructure and wireless communications.

National energy policy aims to enhance people's welfare by meeting energy needs such as gas, water, and electricity. However, energy distribution often faces challenges such as human error causing measurement inaccuracies. It is necessary to develop an IoT-based gas distribution monitoring system that can minimize

errors and fraud. This system uses a turbine meter to measure flow pressure and temperature in accordance with AGA Report 7 standards, and the results are processed by a microcontroller and stored in a database.

EVC or what is usually called electronic volume corrector is a device used to measure volume. There are several variables used to measure volume flow rate, namely pressure and temperature which are converted into pulses to be received by EVC. Usually EVC is located in a metering system which is used to correct the value of volume during base conditions with several important parts in EVC, namely pressure transmitter, temperature transmitter, recorder, corrector, display.

2. RESEARCH METHOD

2.1 Hardware design

In the data collection process, use a simulator. The turbine meter utilizes a pulse injector simulator. The pressure transmitter uses current injector 1, and the temperature transmitter uses current injector 2. Each reading from the current injector is processed by the ADS1115, which converts it into a digital signal. Meanwhile, the pulse injector simulator is processed by the Arduino Uno microcontroller. This setup prevents the ESP32 from being exposed to electromagnetic effects, which can disrupt the processing of data from the current injectors. To maintain the accuracy of the reading results, the injection outputs from the pulse injector are processed by the Arduino Uno. The PC817 facilitates the isolation of two different electrical circuits between the pulse injector and the Arduino Uno.

2.2 Software design

The software design begins by programming the ESP32 microcontroller using the Arduino IDE to process monitoring data from pressure, temperature, and flow variables in flowing conditions. The Arduino Uno R3 is also programmed to process data from the pulse injector simulator, with the variables displayed as pressure (PSIG), temperature (Fahrenheit), and flow (ft^3/h). Data from the ESP32 is displayed on the Nextion HMI, including five variables: pressure, temperature, flow, total flow, and flow at base condition (Qb). Conversion of readings from flowing conditions to standard conditions is carried out by the AGA 7 standard. In addition to the local display on the Nextion HMI, data is also sent online using the sendData() function to a server, utilizing a JSON object library for data storage. The local interface uses a Nextion HMI designed with the HMI Nextion Editor software, while the web interface was developed with Next.js and hosted on Vercel to display real-time data in line graphs. Data is also sent to Telegram via the gateway to display time, date, pressure, temperature, Qf, Qb, and total flow.

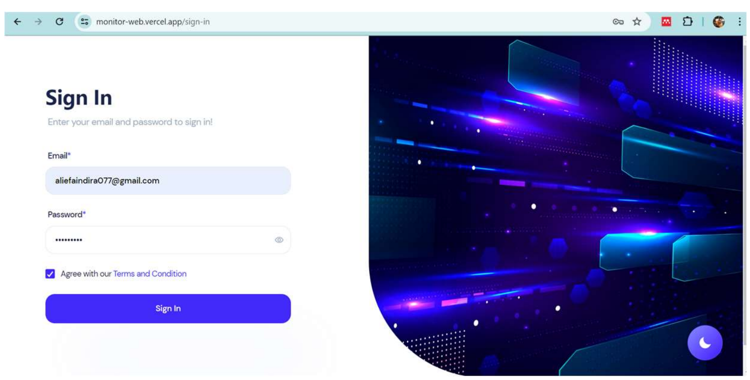

Fig 1. Software Design

3. RESULT

In the implementation of the electronic volume corrector prototype, both hardware and software designs were developed. The hardware design involved wiring various modules, including the microcontroller, RTC DS3231, ADS1115, simulator injector, SD card module, and HMI Nextion. The software design focused on creating the interface for the web server. Following the design phase, the system underwent testing, and monitoring of measurement results was conducted through both local and remote HMIs.

3.1 Implementation system

In the hardware implementation, the modules were arranged inside a panel box measuring 30x40x20 cm. Externally, the panel box features an HMI Nextion used to display the measurement results. Additionally, there is a simulator injector that provides signals in the form of current and pulses to simulate flow.

Fig 2. Hardware Implementation

The wiring implementation involved connecting each component to be installed inside the panel box. A power supply was used to provide power to all the components. Additionally, a relay was included to protect the components of the electronic volume corrector prototype in case of a current surge. Each simulator injector is connected through terminals to the control system, which includes an ESP32 and an Arduino Uno. These microcontrollers are linked using serial communication protocol. Several other modules, such as the RTC DS3231, SD card module, ADS1115, and PC817, are also integrated into the system.

The software implementation began with programming the ESP32 and Arduino Uno in C++ using the Arduino IDE. This programming converts values from the simulator injector into variables such as pressure, temperature, and flow. Additionally, the program handles displaying these variable readings on the HMI Nextion for local monitoring. The displayed variables include pressure, temperature, flow, and total flow readings. The flow variable must be compensated to standard readings because actual conditions like ambient pressure and temperature can affect measurements. Standard compensation calculations follow AGA 7 guidelines since a turbine meter is used.

Beyond local HMI readings, remote readings are accessed via a web application developed with Node.js, which handles database connections and HTTP requests, and builds the server-side of the web app. The web framework relies on Next.js for rendering JSON data sent from the controller processes. The system also includes an API service that sends data to Telegram, providing notification of the readings.

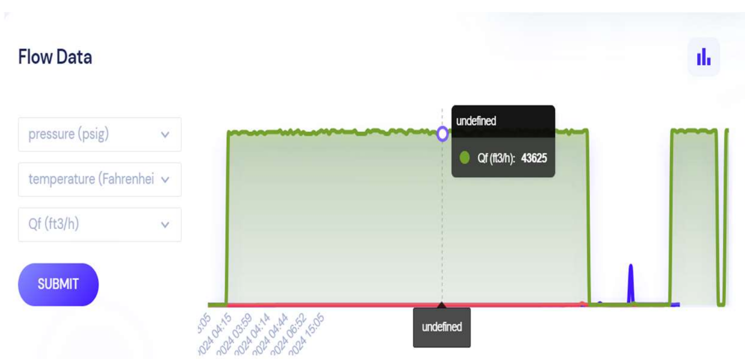

Fig 3. Graphic Implementation

On the initial page, users must enter an email and password stored in the database to log in and access the measurement results page, as shown in Figure. If the credentials do not match the database records, the user cannot log in. The displayed variables include pressure (in PSIG), temperature (in Fahrenheit), and flow (in ft^3/h), along with two other variables: standard condition flow and total flow. The continuous graph displays three variables: pressure, temperature, and Qf. Once the readings are displayed on the web server, data notifications are also sent to Telegram. The transmitted data includes the time and date, along with other variables used in the electronic volume corrector prototype, such as pressure, temperature, Qf, Qb, and total flow, sent as a single, consolidated message.

Fig 4. Telegram Notification

3.2 Equations

Berikan pernyataan apa yang diharapkan seperti yang tertera pada bagian "PENDAHULUAN" dan apa yang diperoleh yang menghasilkan bagian "HASIL DAN PEMBAHASAN", sehingga terdapat kesesuaian. Selain itu juga dapat ditambahkan prospek pengembangan hasil penelitian dan prospek penerapan studi lanjutan ke depan (berdasarkan hasil dan pembahasan).

In this study, the AGA 7 standard is used to calculate the volume of the flow rate. There are several things that must be considered in calculating volumetric flow rates including temperature and pressure as well as compressibility. These three components can greatly affect the volumetric flow rate. A turbine meter is a flow rate measuring device that depends on the flow of gas which causes the meter rotor to rotate at a speed similar to the flow rate

$$Q_b = Q_f \left(\frac{P_f}{P_b}\right)\left(\frac{T_b}{T_f}\right)\left(\frac{Z_b}{Z_f}\right) \quad (1)$$

Next, the error value between Kelton's Qb and the Qb displayed on the prototype is calculated.

$$error = \frac{Q_{b\ Prototype} - Q_{b\ Kelton}}{Q_{b\ Kelton}} \times 100\% \quad (2)$$

3.3 Measurement result data

In this case, the compressibility factor uses the assumption of 1.08980. Apart from that, the temperature factor also uses Rankine units which have been compensated for standard calculations. By setting two variables to fixed values from the 3 existing variables, namely pressure and temperature with fixed values which change Qf.

Table 1. Data P,T Constant

Pressure **(PSIG)**	***Temperature*** **(Fahrenheit)**	**Qf (ft^3/h)**	**Qb Prototype (MSCFH)**
92	37	4451	36,77
92	37	13354	110,33
92	37	22258	183,89
92	37	31161	257,45
92	37	40064	367,78

The second static test experiment is by determining the values of pressure and Qf.

Table 2. Data P,Qf Constant

Pressure (PSIG)	Temperature (Fahrenheit)	Qf (ft3/h)	Qb Prototype (MSCFH)
92	0	13354	119,21
92	37	13354	110,33
92	75	13354	102,68
92	112	13354	96,03
92	149	13354	90,19

The third static test experiment is by determining the values of temperature and Qf.

Table 3. Data T,Qf Constant

Pressure (PSIG)	Temperature (Fahrenheit)	Qf (ft3/h)	Qb Manual (MSCFH)
0	37	13354,8	15,23
375	37	13354,8	402,87
750	37	13354,8	790,52
1125	37	13354,8	1178,16
1500	37	13354,8	1565,8

4. DISCUSSION

Data sent to the web server has the same value as data sent to Telegram. With this, it can be seen that the error amount contained in the Telegram notification has the same value as the error magnitude contained in the web server reading results as shown in the picture. The detailed digits that are read in the numbers after the comma are because the data originating from the injector simulator is processed first by the 16 bit ADS1115 so that the reading is more accurate.

Fig 1. Database and Notification

The data sent from the microcontroller used in the signal injector output process uses the implementation of JSON (Javascript Object Notation) which is used to describe data from the web server to the microcontroller. In sending data to the web server, code rules are used which are called API (Application Programming Interface) so that both the web server framework build program and the program embedded in the ESP32 microcontroller can communicate with the request and response cycle. Meanwhile, the protocol used is REST API (Representational State Transfer) with the definition of how API communication works by using HTTP/HTTPS to send data using the REST protocol. The data transmission format used is JSON format. Later the data will be stored in a data storage service, in this research supabase is used to store the results of data processing.

4.1 Data conversion

Before processing the data on the microcontroller, it is first converted into voltage so that it can be processed by the ADC. In this experiment a 250 Ohm resistor was used. With that, the results obtained range from 0-5 V. Then, the results from the ADC are sent via serial communication protocol. This shows that the ESP 32 does not directly receive a voltage of 5V. In this experimental test, there were 2 transmitters used, namely a pressure transmitter with a range of 0-1500 PSIG and a temperature transmitter with a range of 0-149°F.

By using the pressure transmitter output equation it can be calculated as follows:

$$mA = \left(\frac{measurement\ input - LRV}{input\ span} \times output\ span\right) + live\ zero \quad (3)$$

$$mA = \left(\frac{0-0}{1500} \times 16\right) + 4 \quad (4)$$
$$mA = 4 \quad (5)$$

After the transmitter output is produced, it can be continued with conversion to voltage using the following equation:

$$V = I \times R \quad (6)$$

$$V = 4\ mA \times 250 \quad (7)$$

$$V = 1000\ mV = 1\ V \quad (8)$$

Continue by converting the voltage value into a digital value using the following comparison equation:

$$Number\ of\ step = \frac{V_{in}}{resolusi} \quad (9)$$

$$Number\ of\ step = \frac{1000\ mV}{0{,}076295\ mV} \quad (10)$$

$$Number\ of\ step = 13107 \approx 11001100110011_2 \quad (11)$$

In the conversion process to digital, 16 bits are used in accordance with the ADS1115 specification which indicates that the range ranges from 0 to 65535.

$$Number\ of\ step = (1 \times 2^{13}) + (1 \times 2^{12}) + (0 \times 2^{11}) + (0 \times 2^{10}) + (1 \times 2^{9}) + (1 \times 2^{8}) + (0 \times 2^{7}) + (0 \times 2^{6}) + (1 \times 2^{5}) + (1 \times 2^{4}) + (0 \times 2^{3}) + (0 \times 2^{2}) + (1 \times 2^{1}) + (0 \times 2^{0}) \quad (12)$$

$$Number\ of\ step = 8192 + 4096 + 0 + 0 + 512 + 256 + 0 + 0 + 32 + 16 + 0 + 0 + 2 + 0 \quad (13)$$

$$Number\ of\ step = 13106 \quad (14)$$

After that, the decimal number is then converted back to voltage with the following equation:

$$V_{in} = Number\ of\ step \times resolution \quad (15)$$

$$V_{in} = 13106\ \times 0{,}076295 \quad (16)$$

$$V_{in} = 999{,}9237\ mV = 0{,}9\ V \quad (17)$$

Because the conversion process uses a 250 Ohm resistor, it can be converted to standard current as follows:

$$I = \frac{V}{R} = \frac{0{,}9}{250} = 3{,}99\ mA \quad (18)$$

Continue by converting to the original quantity using the following equation:

$$measurement\ display = \left(\frac{current\ input\quad mA}{output\ span}\right) \times\ input\ span \quad (19)$$

$$measurement\ display = \left(\frac{3{,}99\ mA - 4\ mA}{16\ mA}\right) \times 1500 \quad (20)$$

$$measurement\ display = -0{,}02861\ PSIG \quad (21)$$

The explanation above is the data processing process that can be read from the ADC to the ESP32 microcontroller. The voltage conversion can be up to 5V because a separate ADC module from the ESP32 is used.

4.2 Verification data

Data verification is used for the author to find out that the equation used to calculate the flow rate at base standard is in accordance with AGA 7. The steps used in data verification are by using a comparison of the Kelton software displayed in the attachment with the data that has been obtained from telegram notifications. or with data that has been stored in the database. The data verification process is carried out using a static test, namely by setting two variables to fixed values from the 3 existing variables. The first thing to do is set the pressure and temperature values, then the value that changes is the flow rate at flowing condition.

It can be seen from table 1 that as the Qf value increases, Qb (flow rate at base condition) will also increase. As the calculation formula contained in the theoretical basis states that Qf is proportional to Qb. Manual Qb calculations can be done using the equation below:

$$Q_b = Q_f \left(\frac{P_f}{P_b}\right)\left(\frac{T_b}{T_f}\right)\left(\frac{Z_b}{Z_f}\right) \quad (22)$$

$$Q_b = 4451 \left(\frac{92+\quad ,73}{14,73}\right)\left(\frac{60+4}{37+}\right)(1{,}08980) \quad (23)$$

$$Q_b = 36773{,}51 \text{ SCFH} \quad (24)$$

$$Q_b = 36{,}773 \text{ MSCFH} = 36{,}77 \text{ MSCFH} \quad (25)$$

Next, the error value between Qb Kelton and Qb displayed on the prototype is calculated. This is done using the cross validation method to obtain consistent error values.

Fig 2. Output Kelton

The allowable error threshold is 1% for the flow variable, in figure 7 the error value is still within the appropriate threshold.

$$error = \frac{Q_{b\ Prototype} - Q_{b\ Kelton}}{Q_{b\ Kelton}} \times 100\% \quad (26)$$

$$error = \frac{36{,}77-3\quad ,78}{36{,}78} \times 100\% \quad (27)$$

$$error = 0{,}027\% \quad (28)$$

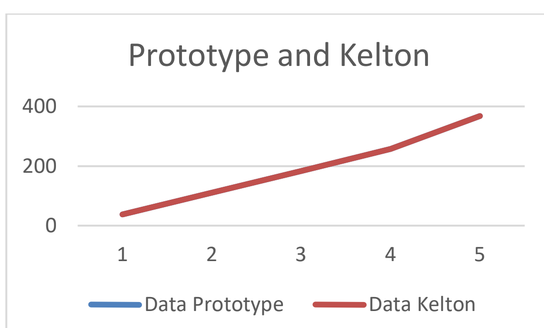

Fig 7. Difference Between Prototype and Kelton

In this case, Qb Kelton is used as a reference because basically the software has been designed and developed with an algorithm that has gone through rigorous testing so that it has a higher level of accuracy and precision compared to the results of manual Qb calculations and Qb prototypes. The results displayed on the prototype also produce values that are the same as the program for calculating flow rate values under standard conditions. This is in accordance with Figure 8. Both from the local HMI and notifications sent to Telegram have the same value. However, sending notification of measurement results to reach a stable value takes some time. This is because the ESP32 is not only used for the process of sending data but also processing the converted data so that adjusting the results between the local HMI and telegram notifications takes quite a long time.

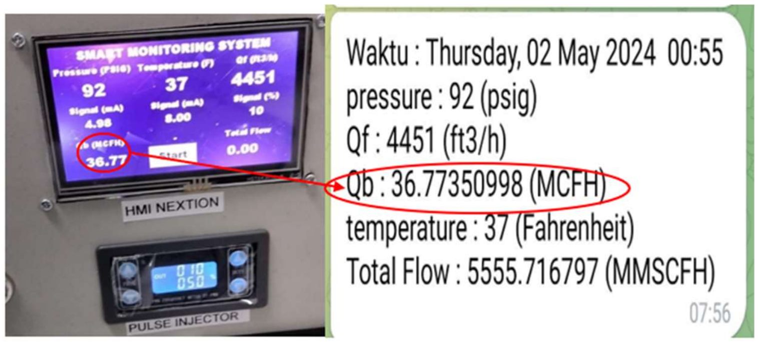

Fig 8. HMI Local

Likewise, the reading results will be displayed on the WEB display. In stabilizing the reading results, it also takes some time to match the standards. This is due to the process from the database to reading the web. As shown in Figure 9, the results displayed are the same as the results sent to Telegram. All data read on the web and also sent to Telegram comes from the database and then communicates with the API method to send the reading data results. In accordance with the program embedded in the ESP32, data transmission to the database will occur every 1 minute. The existence of a pulse injector which has an output in the form of pulses results in a stable Qf value in accordance with the duty and frequency settings. This is also the reason why the value sent to Telegram or the web takes a long time.

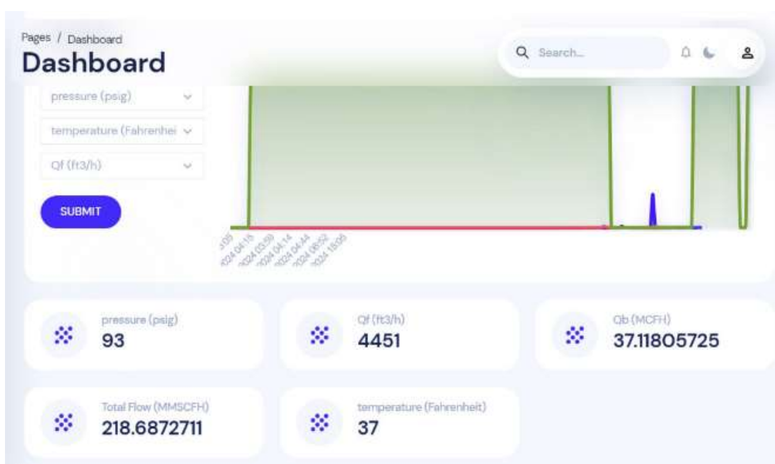

Fig 3. Web Server

Meanwhile in table 2 is a static test experiment by determining the values of pressure and Qf. The results of the table calculation show that the higher the temperature value, the lower the Qb value will be. This is in accordance with the equation in the Qb calculation which is inversely proportional to the temperature when in flowing conditions. Next, the error value between Qb Kelton and Qb displayed on the prototype is calculated.

Table 4. Error P,Qf Constant

Qb Prototype (MSCFH)	Qb Kelton (MSCFH)	Error (%)
119.22	119,22	0,0168
110,33	110,34	0,0091
102,68	102,68	0,0000
96,03	96,03	0,0000
90,19	90,19	0,0000
Average Error		0,0052

Table 3 shows that if the pressure value increases, the Qb value will also increase. This is in accordance with the equation contained in the Qb calculation. With this, pressure is directly proportional to Qb. Next, the error value between Qb Kelton and Qb displayed on the prototype is calculated.

Table 5. Error T,Qb Constant

Qb Prototype (MSCFH)	Qb Kelton (MSCFH)	Error (%)
15,23	15,20	0,1974
402,87	403	0,0323
790,52	790,81	0,0367
1178,16	1178,62	0,0390
1565,8	1566,42	0,0396
Average Error		0,0690

5. CONCLUSSION

1. The system is designed to facilitate users in observing necessary data from the metering system. It sends notification readings to Telegram, allowing users to monitor data flexibly without needing to visit the site.
2. The EVC system effectively monitors essential metering variables such as pressure, temperature, and flow in flowing conditions, converting them to standard readings. The system uses both local and remote HMIs for displaying readings. Local readings are stored on an SD card module as a data logger, while remote HMI readings are saved in a Supabase database. Both readings are consistent, though the SD card module cannot display as many digits as the web server and Telegram.
3. Monitoring can be done locally and via web readings and Telegram notifications using the REST API protocol. The API enables direct data storage from the microcontroller to the database, with React managing the display of this data.
4. Tests show that the accuracy of data transmission is good, with an error rate of less than 1%. Data sent to both the web server and Telegram is consistent.
5. The data has been validated using Kelton software, confirming that the programming equations used for calculating flow rate at base standard comply with AGA 7 standards.

REFERENCES

[1] l. A. Adha, “digitalisasi industri dan pengaruhnya terhadap ketenagakerjaan dan hubungan kerja di indonesia,” j. Kompil. Huk., vol. 5, no. 2, pp. 267–298, 2020, doi: 10.29303/jkh.v5i2.49.

[2] k. Hassan, m. Sanusi, a. Kolawole, and a. Abdulkareem, internet of things based smart energy meter for electrical load management system. 2023.

[3] l. Cao, w. Jiang, and z. Zhang, “automatic meter reading system based on wireless mesh networks and sopc technology,” intell. Networks intell. Syst. Int. Work., vol. 0, pp. 142–145, nov. 2009, doi: 10.1109/icinis.2009.45.

[4] h. U. Fajar, a. P. Kharisma, and a. Bhawiyuga, “pengembangan aplikasi layanan kesehatan berbasis web untuk skrining pendengaran menggunakan arsitektur clean (studi kasus: fakultas kedokteran universitas brawijaya),” j. Pengemb. Teknol. Inf. Dan ilmu komput., vol. 6, no. 4, pp. 1645–1652, 2022, [online]. Available: https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/10893.

[5] b. Pranata, a. Hijriani, and a. Junaidi, “perancangan application programming interface (api) berbasis web menggunakan gaya arsitektur representational state transfer (rest) untuk pengembangan sistem informasi administrasi pasien klinik perawatan kulit,” j. Komputasi, vol. 6, pp. 33–42, apr. 2018, doi: 10.23960/komputasi.v6i1.1554.

[6] y. Efendi, “internet of things (iot) sistem pengendalian lampu menggunakan raspberry pi berbasis mobile,” j. Ilm. Ilmu komput., vol. 4, pp. 21–27, sep. 2018, doi: 10.35329/jiik.v4i2.41.
[7] n. Helda and s. Suryadi, “koneksi tanpa batas: membangun portfolio web interaktif dengan vue, nuxt, dan api,” j. Minfo polgan, vol. 12, no. 1, pp. 1557–1568, 2023, doi: 10.33395/jmp.v12i1.12892.