Journal of Soft Computing Exploration

Homepage: https://shmpublisher.com/index.php/joscex

p-ISSN: 2746-7686
e-ISSN: 2746-0991

.

Digital image based IoT intelligent fire detection with telegram notification

Rizqi Sukma Kharisma[1], Malik Ibrahim[2]
[1,2]Department of Informatic Engineering, Universitas AMIKOM Yogyakarta, Indonesia

Article Info

Article history:

Received May 15, 2025
Revised May 20, 2025
Accepted June 18, 2025

Keywords:

IoT
Digital image
Detection
YOLO
Fire

ABSTRACT

Due to inadequate handling, fire disasters often result in significant losses and even loss of life. A fire detection system is essential, especially in places prone to fire. In this study, a digital image-based IoT system was built using the YOLO (You Only Look Once) algorithm to detect and provide fire warnings quickly and accurately. This research was conducted to develop a fire detection system from existing research on IoT devices by combining it with digital image processing technology with the YOLOv8 algorithm, as well as integrating the IoT system into the Telegram instant messaging application. This study also combines a fire detection system with a fire sensor, MQ-2 temperature sensor, and MQ-2 smoke sensor. The study results show that the YOLOv8 nano model with ESP32-CAM can detect small flames from candles up to a distance of 220 cm. The ESP32 fire sensor can detect small flames up to a distance of 90 cm and large flames up to a distance of 140 cm. VPS can be sent to the Telegram application, just as the LM35 temperature sensor detects temperatures above 50ºC and the MQ-2 smoke sensor detects smoke levels above 450 ppm. All data obtained can be displayed on the VPS dashboard and the Telegram application.

Corresponding Author:

Rizqi Sukma Kharisma,
Department of Informatics Engineering,
Universitas AMIKOM Yogyakarta,
Padjajaran Street, Ring Road Utara, Condongcatur, Depok, Sleman, Yogyakarta 55283, Indonesia.
Email: lsntl@ccu.edu.tw
https://doi.org/10.52465/joscex.v6i2.567

1. INTRODUCTION

Fire is a non-natural disaster that often causes losses. Fires can be caused by several things, namely electrical short circuits, lightning strikes, gas leaks, burning rubbish, cigarettes, etc. [1]. Data from BPS DKI in 2022 states that, fires due to electrical short circuit are the main cause of fire disasters in Jakarta, as many as 65.82% of fire disasters occur due to electrical short circuit, 11.41% due to gas leakage, and the rest are other causes such as burning garbage, cigarette butts, and candles. While data from BPS Surakarta City, throughout 2023 the cause of fires occurred due to electricity as many as 39 cases, gas stoves 26 cases and other causes 75 cases [2], [3].

IoT (Internet of Things) is a concept of various devices or everyday objects connected to the internet between one another, which allows data exchange and interaction between devices without direct human intervention [4]. The purpose of IoT technology is to extend internet connectivity from computers or cell phones to other devices used at home or for business purposes. IoT allows users to control a device remotely through the internet network infrastructure. Basically, IoT technology can reduce human work by giving access to devices connected via the internet network easily [5].

Digital image is the result of the digitization process of an image that is represented numerically [6]. Digital image processing itself is a type of processing on images, to enhance or extract information from images, where this processing has input in the form of images and outputs that can be in the form of characteristics or features associated with the image [7]. Digital image processing technology requires an algorithm to work. The digital image algorithm that will be used in this research is the YOLO algorithm. The YOLO (You Only Look Once) algorithm is an algorithm used to detect an object in real-time [8]. The use of the YOLO algorithm itself is because this algorithm has superior performance compared to other algorithms for detection [9].

From the explanation above, the author sees that there is an opportunity to solve the fire problem by combining IoT technology and digital image processing. This system will work by connecting the ESP32-CAM camera with a local server on a computer containing the YOLO model that has been created. If the camera detects a fire, the data will be sent to the VPS along with data from the ESP32 which will control the sensor, the system will decide whether there is a fire or not. If the system detects a fire, it will send a message to the user's telegram instant messaging application. Hopefully, this system can prevent material losses and casualties, which can be prevented if we have a good and accurate warning and detection system.

There are also several research journals that discuss fire detection systems, for example Integrated Fire Detection System using ML and IOT by Swapnil Sawant, et al., in 2024. They created an integrated fire detection system that utilizes IoT technology by combining smoke analysis, thermal, and image analysis technology using the YOLOv8 model for real-time fire detection [10]. An Improved Fire Detection Approach Based on YOLO-v8 for Smart Cities by Fatma M. Talaat and Hanaa ZainEldin, in 2023 they are introducing a fire detection system called the Smart Fire Detection System (SFDS), which is based on the YOLOv8 algorithm. This system uses deep learning, to identify fire quickly and accurately. This system can detect fires in real-time, has a precision level of 97.1% which shows effectiveness in distinguishing fire objects from others [11]. Fire Detection System At Labuhanbatu University Based On Internet Of Things (IoT) by Iwan Purnama, et al. in 2023 The fire detection system developed in this study obtained a fairly high level of effectiveness, namely 90%. This study uses 3 sensors, namely fire, gas, and temperature sensors. The system is made using the ESP8266 microcontroller, and is connected to the blynk server and warnings using alarms and LED lights [12]. IoT Based Fire Detector System Using MQ-4 and LM35 Sensor by Md. Mehedi Hasan Shuvo and Suman Chowdhury in 2024 Developing a prototype fire detection system that utilizes modern technology to alert property owners of fire hazards. This system aims to detect changes in temperature and the presence of gas (especially methane, CH4). The features of this system are able to send warnings to property owners via phone calls when a fire or gas leak is detected [13]. This research is the basis for the author in creating the YOLO IoT fire detection system by combining fire sensors, MQ-2 temperature sensors, and MQ-2 smoke sensors. Not only that, the system will be connected to VPS and Telegram applications that display data and warnings in real time

2. METHOD

YOLOv8 serves as the foundation of a highly efficient and fast object detection algorithm, renowned for its capability to accurately identify objects—specifically fire—from digital images. This algorithm employs a single-shot detection approach, meaning that it processes the entire input image in one step, without separating the task into multiple phases such as feature extraction followed by classification. The image is divided into a grid structure, where each grid cell simultaneously predicts bounding boxes that may contain objects along with the corresponding class probabilities. This simultaneous prediction allows YOLOv8 to recognize complex object patterns with impressive precision and speed, making it particularly well-suited for applications requiring real-time responsiveness, such as monitoring hazardous situations. Figure 1 shows the main architecture of YOLOv8.

YOLOv8 nano is an optimized, compact variant of this algorithm designed to be lightweight and resource-efficient, making it ideal for deployment in indoor IoT systems focused on fire detection. This version uses a convolutional neural network architecture that has been specifically trained to identify distinctive fire patterns, including the challenging task of detecting small flames even at considerable distances. The compact size and computational efficiency of YOLOv8 nano enable it to operate in real-time environments with minimal latency, a critical feature for early fire warning systems where every second counts. By delivering rapid and accurate detection, YOLOv8 nano maximizes the potential for timely alerts and interventions, thereby significantly reducing the risk of extensive damage and enhancing safety measures indoors.

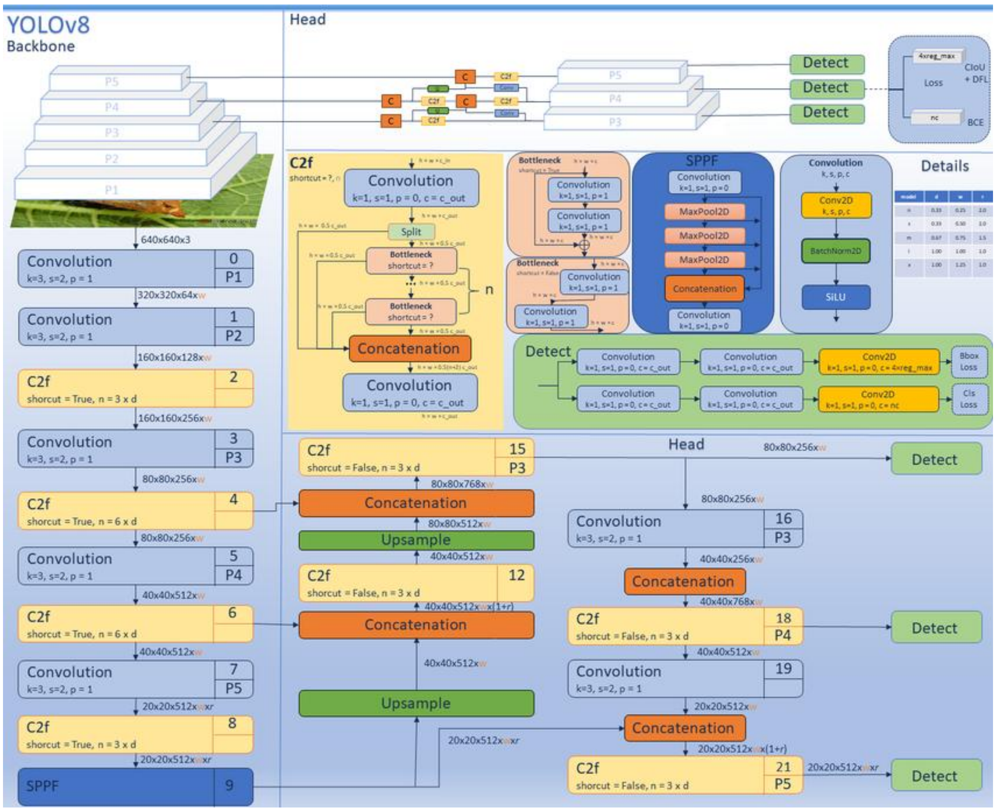


Figure 1. YOLOv8 main architecture

This system uses an ESP32 microcontroller and ESP32-CAM to capture images and several sensors, such as fire, temperature, and smoke. This research uses a quantitative approach to develop and test a fire detection system based on digital images and sensors. This quantitative approach measures sensor data from the surrounding environment, such as temperature and smoke concentration. Data will be collected through temperature, smoke, and fire sensor devices, and ESP32-CAM will be used to capture video streams that detect fires using the YOLOv8 nano algorithm. This research has procedures and steps for creating the system. The Waterfall method is a very common method in developing a system, the stages of the waterfall method are requirements, design, implementation, testing and maintenance [14]. The following is the flow of this research according to the waterfall method, which:

a. *Requirement*
   The author seeks information about the needs in making the system, which includes literature study and needs analysis.
b. *Design*
   Designing system design, including architecture design and VPS dashboard design.
c. *Implementation*
   Code generation for system devices, including the YOLOv8 nano model, ESP32-CAM camera, and ESP32.
d. *Testing*
   Integrate devices including camera and sensors in the system to the VPS and Telegram application.
e. *Maintenance*
   Conduct testing and ensure system reliability and performance.

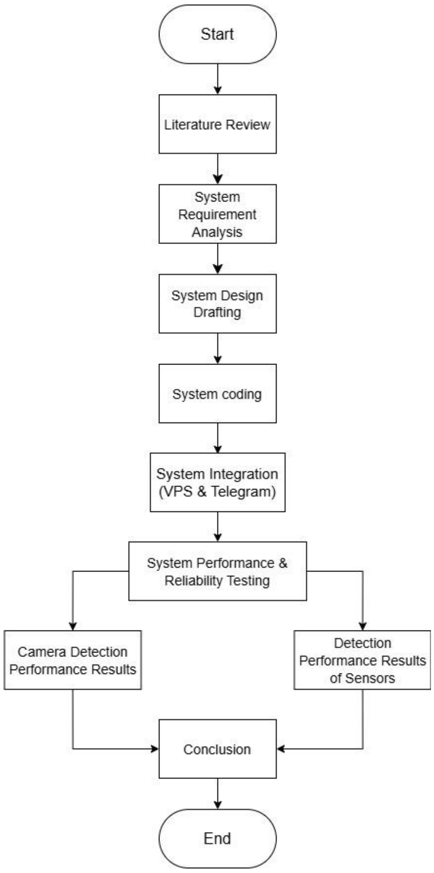


Figure 2. Research flow

Figure 2 shows the research flow for system development. The following is an explanation of the research flow in the diagram above.

a. *Literature Review:*
At this stage, the author will collect data from sources of books, articles, and research journals that discuss similar research.

b. *System Requirement Analysis:*
In the system requirement analysis, the author collects information about hardware and software specifications.

c. *System Design Drafting*
At this stage, the author makes a design for both the circuit form of the hardware and the system architecture.

d. *System Coding*
Perform program code generation for both ESP32 and ESP32-CAM hardware, VPS dashboard, and VPS backend for integration.

e. *System Integration*
Set up a Telegram bot to then be integrated with the VPS backend so that it can send warning messages.

f. *System Performance and Reliability Testing*
Conducting a trial of the entire system, including hardware, VPS, and Telegram, by testing them one by one, starting from detection distance, detection time, and the VPS's ability to display detection results on the dashboard and send them to the Telegram application.

g. *Conclusion*
The author summarizes the points obtained during the research and provides suggestions to further researchers.

.

3. RESULTS AND DISCUSSIONS

A. Design and Development

Device interaction in this system runs in a way: ESP32-CAM will use a computer as a local server to run the YOLOv8 nano model, and then streaming and detection images from the YOLO model will be sent to the VPS. ESP32 will collect data from sensors and send the data obtained to the VPS; the data in the VPS will be sent to the Telegram instant messaging application using a chatbot. Figure 3 shows the design of this fire detection system in the form of an architectural diagram.

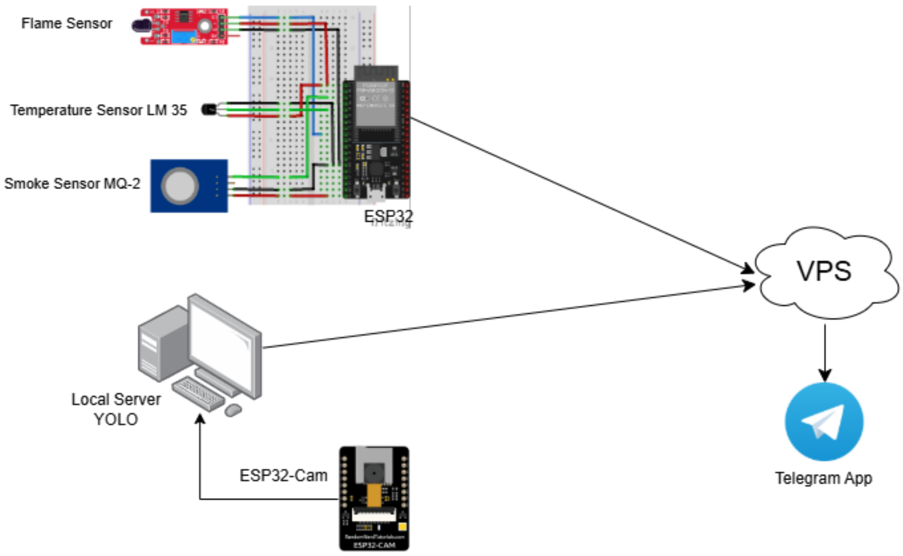


Figure 3. System architecture diagram

The ESP32 needs to be assembled so that this microcontroller can run with the connected sensors and receive data from the sensors. Figure 4 is a picture of the circuit of the ESP32 and ESP32-CAM made with the Fritzing application.

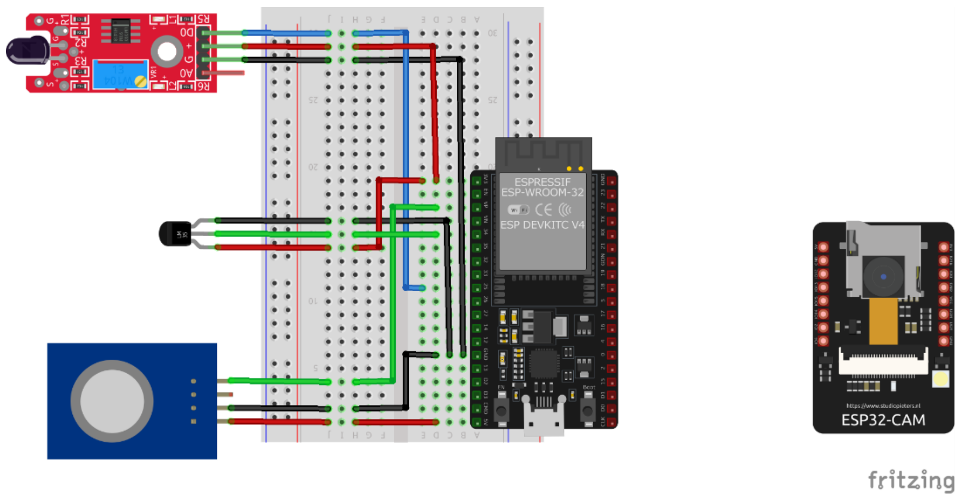


Figure 4. ESP32 and ESP32-CAM circuit design

VPS (Virtual Private Server) is used to run the dashboard, receive data, and also send data with alert messages to the telegram application. To do this, VPS requires a backend configuration. VPS uses the backend to receive data from ESP32-CAM and ESP32. Later, the data obtained by the backend will be displayed on the dashboard page. If any data needed is obtained, the backend will immediately send it to the telegram application in the form of a chat message controlled by the bot. Figure 5 shows the form of a dashboard that has been made.

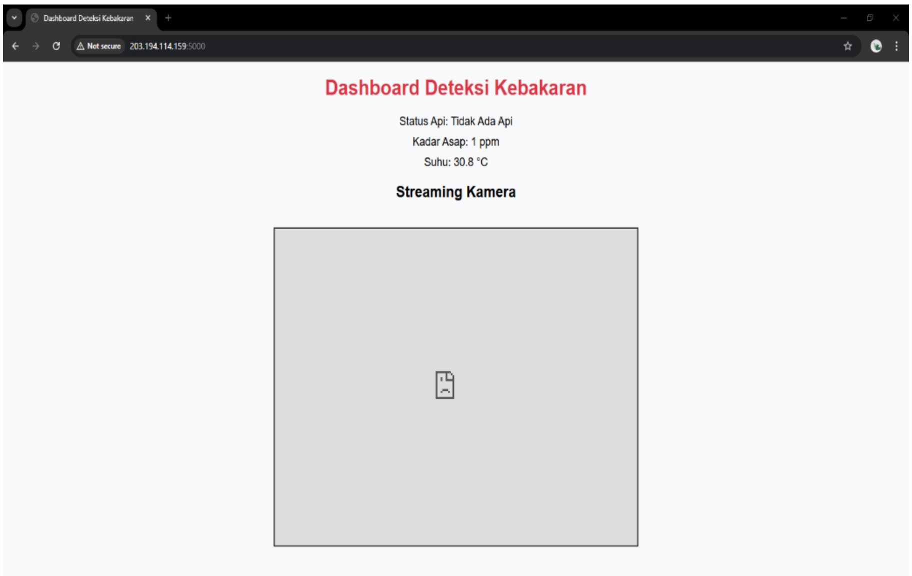


Figure 5. Dashboard design

The backend and chat bot are integrated using the telegram token and chat id obtained when creating the bot. Tokens and chat ids are written in the backend python program code. Telegram bots in this system are made using BotFather which is provided directly by telegram. This bot can send messages according to the data obtained by the backend in real-time as shown in Figure 6.

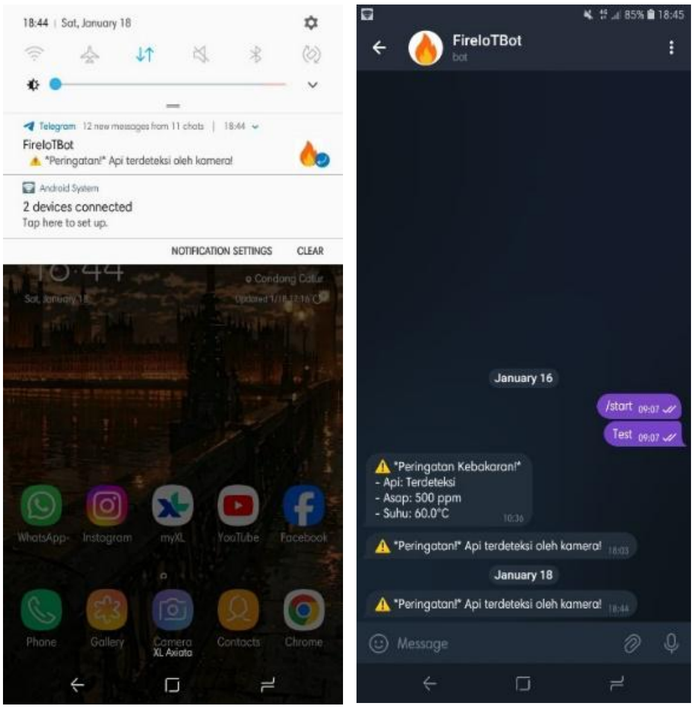


Figure 6. Telegram notification

The stages of how the system works can be seen in Figure 7 which is depicted in the form of a flowchart.

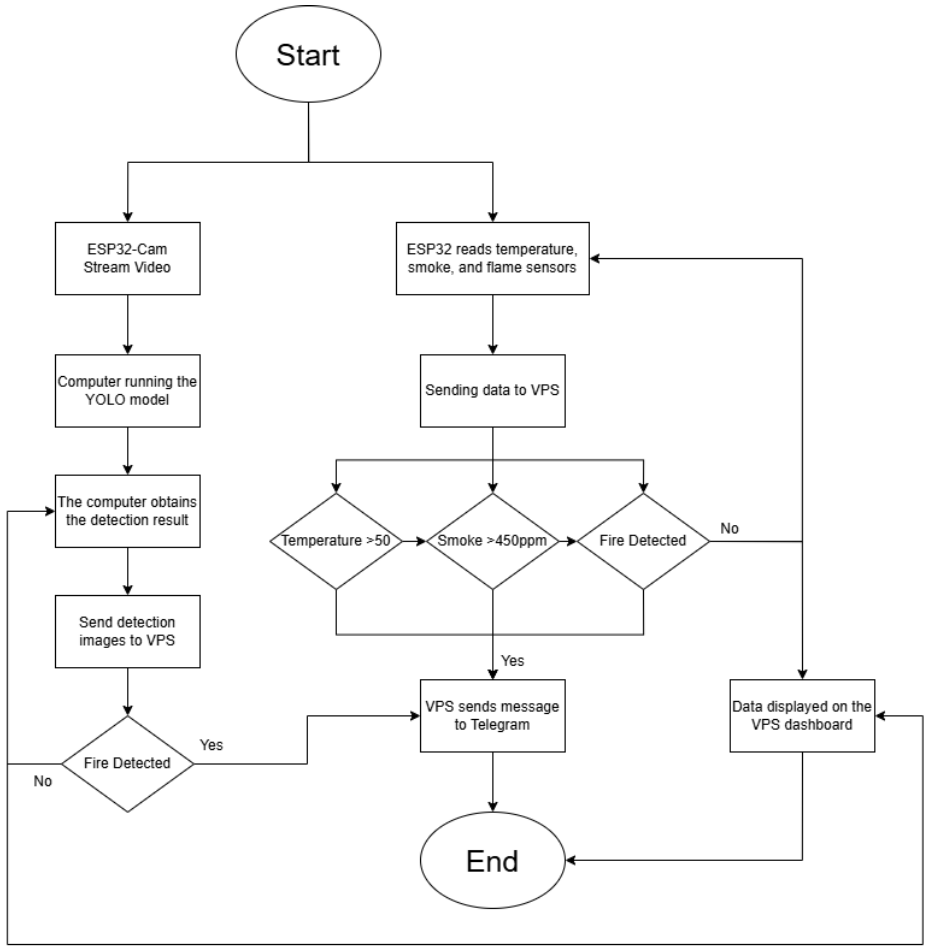


Figure 7. System flowchart

The following is an explanation of how the system works from the flowchart in Figure 7 :

a. *ESP32-CAM*

Once the ESP32-CAM gets power by connecting it to a computer port, it will start running the video streaming, and the computer also acts as a server to run the YOLOv8 nano model on the ESP32-CAM. The YOLO model will read each image frame, and the computer as a server will send the data to the VPS backend. If the YOLO model reads a fire, the computer will immediately send the detection data to the VPS backend, and the VPS will send a Telegram message to the user. All data obtained by the server computer is displayed by the backend to the web dashboard.

b. *ESP32*

When the ESP32 gets power, the sensors will start working to read data. Just like the ESP32-CAM, the data obtained by the sensors will be sent in real time to the VPS backend, which is then also displayed on the web dashboard. When one of the sensors reads a significant change, such as a temperature sensor reading a temperature of more than 50 degrees Celsius, a smoke sensor detecting smoke levels of more than 450 ppm, or a fire sensor detecting a fire, the ESP32 will send the data to the VPS backend and then the backend will send it to the Telegram application.

B. *System Testing*

System testing aims to ensure that the system made can match the expectations of the research objectives. The achievements to be made in testing this system contain system performance testing, including detection distance and response time of each sensor and camera. Testing for sensors is done one by one sensor.

a. *Camera Testing*

Camera testing is done by increasing the fire distance from the camera every 20 cm. Seen in Figure 8 is a simulation of testing the ESP32-CAM YOLO camera.

Testing is carried out to determine that the YOLO detection results in the VSCode terminal when running the model can be displayed on the dashboard shown in the VPS dashboard table column and the results can be sent to the telegram application in accordance with the provisions, which can be sent to the

Telegram application when the camera detects the fire shown in the telegram notification table column, and for the response time table column is the time it takes for the camera to detect the fire after the fire is lit.

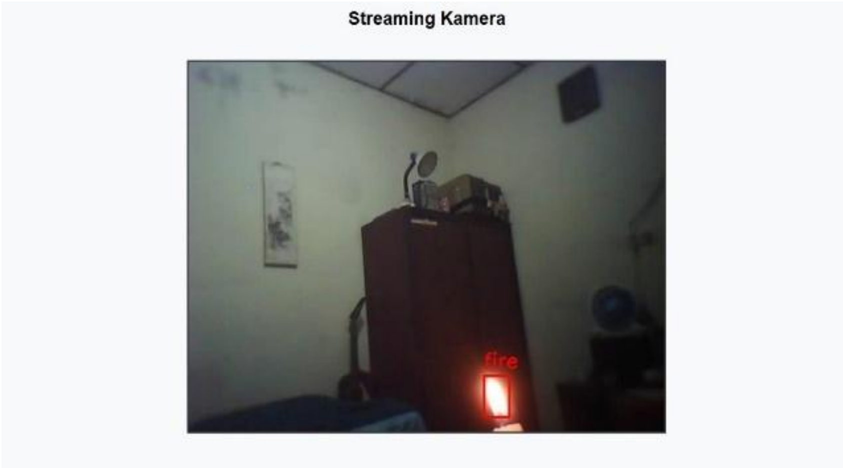


Figure 8. Camera testing

Tabel 1. Camera testing

No	Distance (cm)	VPS Dashboard	Telegram Notification	Response Time (Second)
1	20	Detected	Yes	2
2	40	Detected	Yes	3
3	60	Detected	Yes	2
4	80	Detected	Yes	3
5	100	Detected	Yes	2
6	120	Detected	Yes	4
7	140	Detected	Yes	3
8	160	Detected	Yes	6
9	180	Detected	Yes	5
10	200	Detected	Yes	3
11	220	Detected	Yes	9
12	240	No	No	-

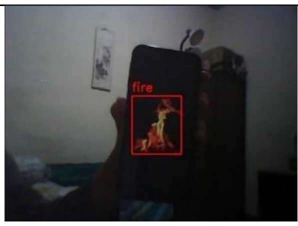
In table 1 above it can be seen, ESP32-CAM with YOLO model can detect fire from candles up to 220 cm, the backend system will immediately send a notification to the Telegram application when it detects the first fire, Telegram notification can be seen in Figure 5.

This result can be influenced by the size of the fire, the bigger the fire, the easier the camera will detect. This result can also be affected by the speed of the image frame, if the camera frame drops then the camera cannot detect the fire quickly.

The YOLO model was also tested to determine whether the YOLO model would detect objects other than fire or not. This test was carried out by displaying an image of a fire from a cell phone, incandescent lamp, a cell phone flash light, and an LED light. The results of this test can be seen in table 2 below.

.

Tabel 2. Testing using another object

No	Object	Result
1	 Picture of fire from cell phone	Detected
2	 Cell phone flash light	Not Detected
3	 Incandescent lamp	Detected
4	 LED Lamp	Not Detected

From the above tests, it can be seen that the YOLO model used can detect fire images and incandescent light bulbs that are lit as fire, this can be a weakness of the system, because users cannot use incandescent lamps at the same time as using this fire detection system. It can also be noted that the YOLO model does not detect LED lights and cell phone flash lights as fire.

b. *Flame Sensor Testing*

Fire sensor testing is taken by changing the distance every 10 cm until the sensor cannot detect fire. Fire sensor testing is done using fire from a candle, the test data can be seen in table 4.3. The VPS dashboard table column is the result of fire sensor detection on the Arduino IDE serial monitor which can be displayed on the VPS dashboard, while the Telegram notification table column is when the sensor detects a fire VPS can send a message to the Telegram application.

Tabel 3. Flame sensor testing (small fire)

No	Distance (cm)	VPS Dashboard	Telegram Notification
1	10	Detected	Yes
2	20	Detected	Yes
3	30	Detected	Yes
4	40	Detected	Yes
5	50	Detected	Yes
6	60	Detected	Yes
7	70	Detected	Yes
8	80	Detected	Yes
9	90	Not Detected	No

In the data above, it can be seen that the maximum distance that can be detected by the sensor in detecting fire from a candle is 90 cm. Data from the detection results can be displayed on the VPS dashboard and can also be sent via message to the Telegram application.

Testing of the fire sensor was also carried out using a larger fire than burning paper waste, this test also increased the distance from the sensor to the fire every 20 cm. The test results can be seen in table 4 below.

Tabel 4. Flame Sensor testing (big fire)

No	Distance (cm)	VPS Dashboard	Telegram Notification
1	20	Detected	Yes
2	40	Detected	Yes
3	60	Detected	Yes
4	80	Detected	Yes
5	100	Detected	Yes
6	120	Detected	Yes
7	140	Not Detected	No

Testing using a fire that is larger than burning paper waste, it is found that the fire sensor on the ESP32 can detect up to a distance of 140 cm, from these tests it can be seen that the size of the fire can affect the results of sensor detection.

c. MQ-2 Smoke Sensor Testing

The results of the MQ-2 sensor test using paper burning smoke are taken based on data sent to the VPS 80 times. In the graph below is the detection result of the MQ-2 smoke sensor from the Arduino IDE serial monitor which can be displayed on the VPS dashboard, the blue bar is the detection result sent to the telegram application when the sensor detects smoke levels of more than 450 ppm.

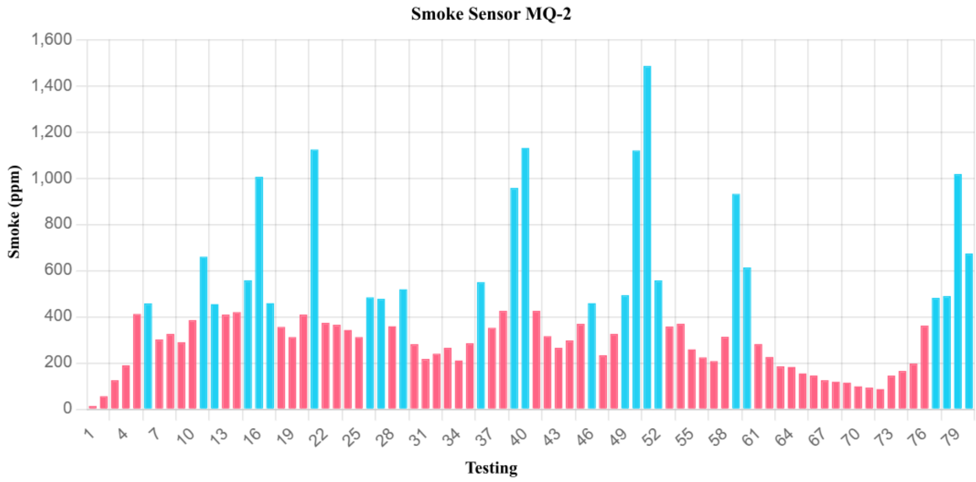


Figure 9. MQ-2 smoke sensor testing

From the data above, it can be seen that the smoke sensor works as expected, data from the Arduino IDE serial monitor can be displayed in real-time to the VPS dashboard when the sensor reads smoke above 450 ppm, data from the VPS will be sent to the Telegram application.

d. LM35 Temperature Sensor Testing

The test results can be seen in Figure 10. The test on the LM-35 temperature sensor on the graph is the data in the Arduino IDE serial monitor which can be displayed on the VPS dashboard. Data is taken based on data sent to the VPS 80 times. The blue bar on the graph is the detection result that can be sent to the telegram application when the sensor detects a temperature of more than 50 ° C.

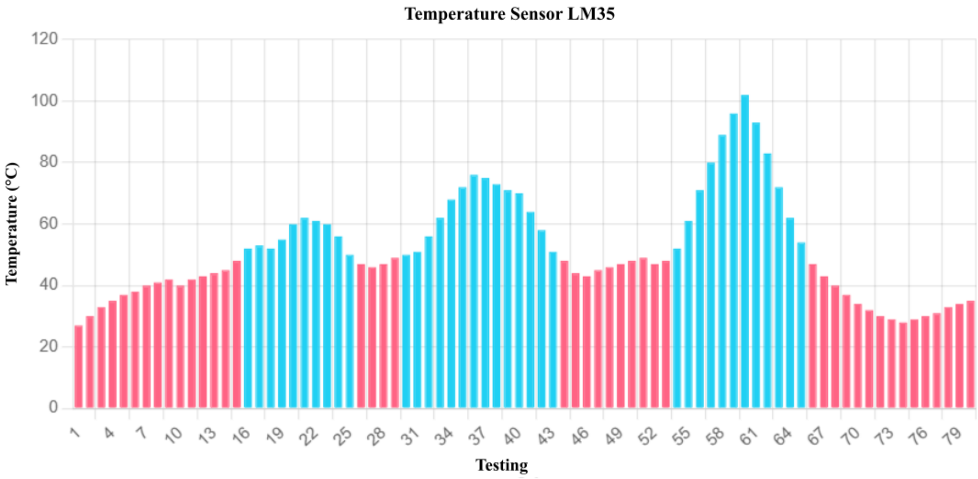


Figure 10. LM35 temperature sensor testing

From the data obtained above, it can be seen that the temperature sensor can work as expected. Data from the Arduino IDE serial monitor can be sent to the VPS dashboard in real-time, and when the temperature sensor reads a temperature of more than 50 ° C, the VPS that receives the data, the VPS can immediately send a message to Telegram.

4. CONCLUSION

Based on the results of making and testing the system that has been carried out, the author has concluded
the fire detection system using the YOLOv8 nano model and ESP32-CAM, can detect small fires from a distance of 220 cm. The size of the fire, and camera frames per second can affect the detection speed. The use of the private ip address of the PC to display streaming YOLO detection results on the VPS dashboard causes streaming to be inaccessible from outside the network (public). The camera can read fire images and incandescent lamps as fire. VPS is proven to be able to send an alert message to the telegram application when the camera detects a fire, sending the message is only done when the camera detects the first fire, to avoid spam.

In the use of ESP32, the fire sensor can detect small fires up to 90 cm, and can detect larger fires up to 140 cm. The system has also been proven to be able to send messages to the Telegram application when the sensor detects a fire. After testing the LM35 temperature sensor and MQ-2 smoke sensor 80 times, it can be concluded that the sensors can work as expected, by sending a message to Telegram, when the temperature sensor detects more than 50ºC, and the smoke sensor is more than 450 ppm. Sensor data displayed on the Arduino IDE serial monitor is also proven to be sent to the dashboard in real-time.

Distance testing using fire from candles can only be detected by cameras and fire sensors, for MQ-2 smoke sensors and LM35 temperature sensors cannot detect smoke or temperature significantly from burning candles, but the VPS will still send potential fire alerts to the Telegram application even though the fire is only detected from one of the devices, either the camera or from one of the sensors.

REFERENCES

[1] Lathifah, *Panduan Keselamatan Saat Kebakaran*. Yogyakarta: DIVA Press, 2021.
[2] Satu Data Jakarta, "Satu Data Jakarta." 2023.
[3] Badan Pusat Statistik, "Badan Pusat Statistik." 2024.
[4] E. A. I. Datya, S. W. I. Adhicandra, E. Budihartono, and N. W. Purnawati, *Pengantar dan Penerapan Internet of Things*. Jambi: Sonpedia Publishing Indonesia, 2023.
[5] L. P. I. Kharisma *et al.*, *Internet of Things (Pengenalan dan Penerapan Teknologi IoT)*. Jambi: Sonpedia Publishing Indonesia, 2024.
[6] M. Rometdo *et al.*, *Pengantar Dasar Deep Learning*. Payakumbuh: Serasi Media Teknologi, 2024.
[7] A. F. Marpaung, F. Aulia, N. Suryani, and R. C. Nabila, *Computer Vision dan Pengolahan Citra Digital*. Surabaya: Pustaka Aksara, 2022.
[8] A.-K. and A. R. Lubis, *Artificial Intelegence*. Medan: UMSU PRESS, 2023.
[9] E. R. Setyanigsih and M. S. Edy, "YOLOv4 dan MaskR-CNN Untuk Deteksi Kerusakan Pada Karung

Komoditi," *TEKNIKA*, vol. 11, no. 1, pp. 45–52, 2022.

[10] S. Sawant, B. Chauhan, S. Kumbhar, G. Chaudhari, and P. Thakar, "Integrated Fire Detection System using ML and IOT," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 12, no. 5, pp. 2091–2100, 2024.

[11] F. M. Talaat and H. ZainEldin, "An improved fire detection approach based on YOLO-v8 for smart cities," *Neural Comput. Appl.*, vol. 35, no. 28, pp. 20939–20954, 2023.

[12] I. Purnama, I. R. Munthe, K. R. Watrianthos, and Z., "Fire Detection System At Labuhanbatu University Based On Internet Of Things (IoT)," *J. RESTI (Rekayasa Sist. dan Teknol. Informasi)*, vol. 7, no. 4, pp. 865–869, 2023.

[13] M. M. H. Shuvo and S. Chowdhury, "IoT Based Fire Detector System Using MQ-4 and LM35 Sensor," in *2024 2nd International Conference on Sustainable Computing and Smart Systems (ICSCSS)*, 2024, pp. 400–403.

[14] R. A. Putra, "Pengantar Teknologi Informasi," in *Buku Referensi*, Jambi: Sonpedia Publishing Indonesia, 2024.

.