TAF Journal of Advances in Technology and Engineering Research JATER
2019, **5**(6): 231-242

PRIMARY RESEARCH

An AI-based web surveillance system using raspberry Pi

P. V. Luu [1] , J. Weed [2], S. Rodriguez [3], S. Akhtar [4*]
[1, 2, 3, 4] Department of CS/IT, Clayton State University, Morrow, GA, USA

Keywords
Artificial Intelligence (AI)
Web
Surveillance
Raspberry costing
Pi
USB webcam

Received: 3 October 2019
Accepted: 7 November 2019
Published: 23 December 2019

Abstract
We present details of a project to build an inexpensive AI-based home surveillance system using Raspberry Pi. We used a Raspberry Pi and a USB webcam to record a live environment and take photos when motion is detected. The proposed security system allows multiple cameras and users. Users have access to a control center where they can login and select the appropriate camera. The website interface encrypts the user's account information and stores them in a database. Further, an improved setup using Pi-based camera is shown and discussed. The proposed system uses IBM's Internet of Things software library to implement several features via node-red flows. Two flows are created. The first flow triggers the camera to take pictures when motion is detected. The picture is then sent to visual recognition node for processing. If there are people detected, an email is sent with the link to the streaming. If people are not detected, there is no further action. The second flow is used to analyze the picture with AI-based visual recognition node. The picture is passed to the visual recognizer for analysis. Then the result is sent to a template node for display as a table.

I. INTRODUCTION

Security has always been a major concern for homes, businesses, and organizations. With several surveillance systems offered today one can keep a place secured and monitored. However, installing and monitoring surveillance systems require expertise and security personnel, which involve additional cost. The early camera systems were primarily used for real-time viewing, because of the lack of reliable video recording systems. The introduction of Videocassette Recorders (VCR) in 1970's led to the increased popularity of video surveillance.

A simpler residential security system is becoming more important due to the advancement in technology and increasing crime and theft. These systems can provide video footage, whether live or recorded, within a property. In fact, modern surveillance systems providing undeniable video evidence have led to the incarceration of many criminals. This newer technology offers simpler home security system, but it could be expensive if one needs to buy products and pay a company to monitor it. One has to consider if the data is stored locally or remotely on cloud and if the feature of live monitoring is needed.

Early video surveillance systems consist of a video camera connected to a video recording system and/or live video monitoring [1]. The technology has been advancing to include computer-based video processing including implementation of intelligent systems [2], and recent implementations of deep learning techniques including edge detection [3]. A gender recognition using neural network was implemented by Mittal [4]. Various face recognition algorithms involving video surveillance have been studied and compared in [5] and [6].

In [7], an AI-based trajectory analysis has been proposed to automatically gather and analyze the transfer and change behavior of passengers at transportation nodes within the public transport system. A component-based face detection using some early AI techniques was proposed in [8]. A review on the application of AI in smart homes that include face recognition is given in [9]. Since recording of all surveillance activities involve large storage, AI-based human identification is used in [10] to record only when humans are identified.

We propose to build an inexpensive surveillance camera system using a Raspberry Pi. We propose a secure system

*Corresponding author: S. Akhtar
†email: ShakilAkhtar@clayton.edu

by enhancing the existing surveillance system via a control center where users have to login to manage the system. Our enhancement includes a web-based control center for surveillance cameras. The web-based control center can view the live feed of the surveillance camera and provides the ability to analyze the images. The control panel also implements security management by encrypting users' confidential information such as usernames and passwords.

With the development of technology, surveillance cameras nowadays are vulnerable to attackers. It has been reported that hackers can abuse cameras in a variety of locations frequented by people, from public spaces, such as stores, bars, and restaurants, to more intimate places, such as dressing rooms at swimming pools and in fitness centers, saunas in spa salons, and even operating rooms in hospitals [11]. With the access to surveillance cameras, an attacker can record video for their own, sell access to other parties, use the camera to snoop around stores or shops, or steal credit card information. In addition, an attacker may want to use the surveillance camera just to play pranks. Others reason for attackers to attack surveillance camera includes doing distribute denial-of-service attacks, convert cryptocurrency, and financial crime. An attacker may use the camera data to acquire privacy information, financial transaction, and even critical infrastructure for their own purpose such as asking ransom.

Surveillance camera hacking can be done by people who are not experts in hacking because there are plenty of software and tools in public domain that can be used to gain access to the camera. For example, a popular software that recognizes known people in images is SquardCam. Hackers may use software with some well-known penetration testing tools [12], like masscan [13] and RouterScan [14]. A default password is normally used to login to the cameras. A hacker uses the default password to access the cameras instead of creating their own passwords. In addition, sometimes the streaming video does not require any authentication, which increases the chance for attackers to easily get the video. Therefore, just owning an advanced surveillance system without good security may have an adverse effect on the security information of owner. With the security measures in place the safety and reliability of the system is greatly enhanced.

Next section provides the detail of our proposed surveillance camera setup using Raspberry Pi [15, 16]. Section III describes the control panel of our security system. Section IV outlines the proposed AI enhancement of our system that includes the feature of facial recognition.

II. SURVEILLANCE CAMERA SETUP

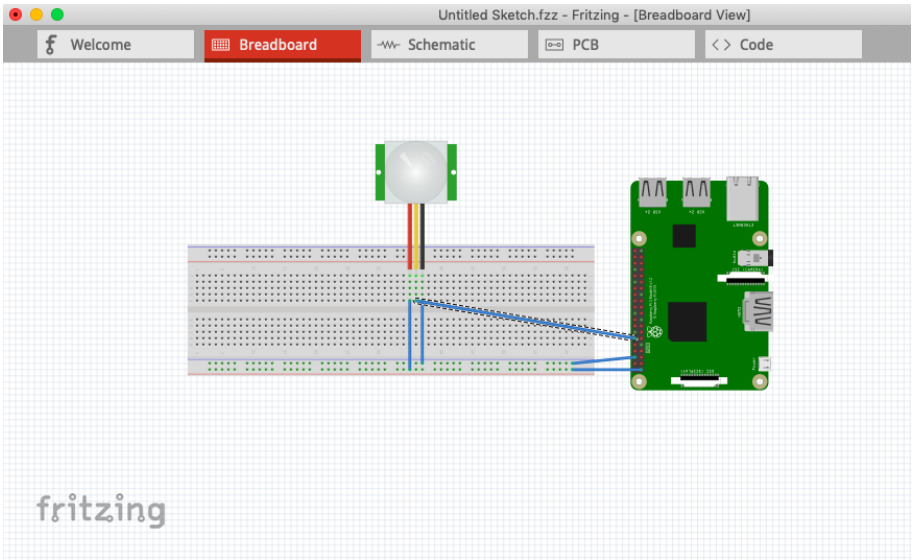
We propose to first utilize an external camera that connects to the Raspberry Pi through a USB port. The system also uses the sensor function to detect the movements. The movements trigger the camera to take a picture and live streaming over the web when the motion is detected. In addition, the system may also support AI for visual recognition specifically face detection as shown later in this paper. If the picture contains people, an email will be sent to the owner. The content of the email includes a link to the stream and the AI page. On the AI page, the visual recognition analyzes the taken image and displays the result about gender and approximate age of the person.

The first function of the surveillance system is to capture pictures and then stream video. To capture a picture, a library named "fswebcam" is installed on Raspberry Pi. For streaming video, another library named "motion" is installed. A video streaming is implemented based upon [17], which provides a tutorial on implementation of streaming system. Next, the sensor to the breadboard is connected to detect the motion. This setup is shown in Figure 1. A tutorial by Short and Joel talks about how to use a breadboard [18].

Next, we set up a node-red program to allow image processing when motion is detected. We create a bash command (in command mode) and an execute node-red to execute it [19]. When a motion is detected, command first stops the streaming. Then it takes a picture and enable the streaming again. Node-red requires an installation of Javascript called Nodejs, within node-red [20, 21]. Finally, we implement the visual recognition within node-red to do the face analysis.

The node-red flows are setup as follows. There are two flows that are created. The first flow will trigger the camera to take a picture when motion is detected (Figure 2). The picture will then be sent to visual recognition node to process. If there are people, an email will be sent with the link to the streaming and analyzing the page. If people are not detected, there is no further action. The second flow is used to analyze the picture with visual recognition node (Figure 3). A picture will be passed to the visual recognition to analyze. Then the result will be sent to a template node which will display as a table on the website.

ISSN: 2414-4592
DOI: 10.20474/jater-5.6.2



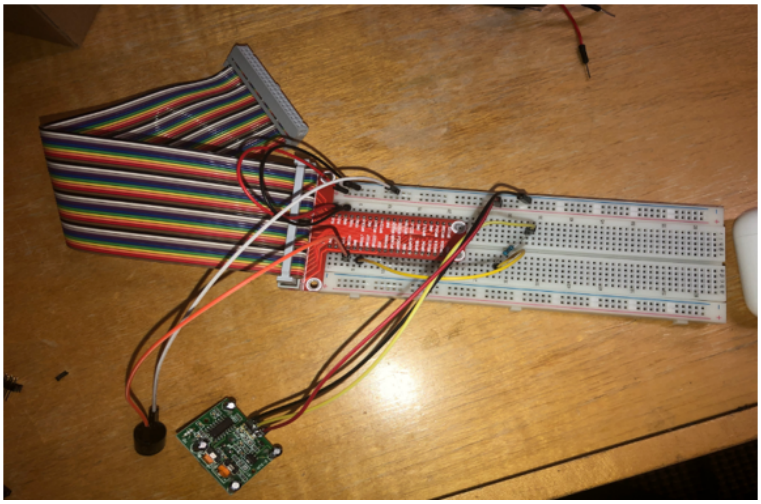


Fig. 1. Breadboard connections to detect and video stream movements

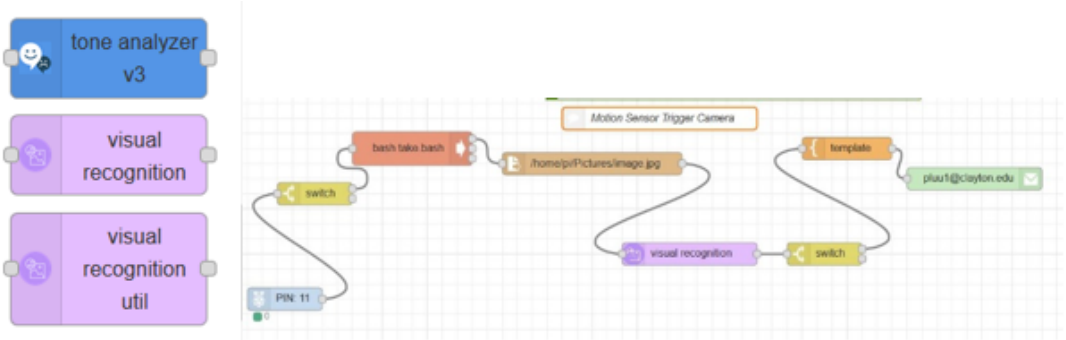


Fig. 2. Node-Red elements and flow for motion sensing

ISSN: 2414-4592
DOI: 10.20474/jater-5.6.2

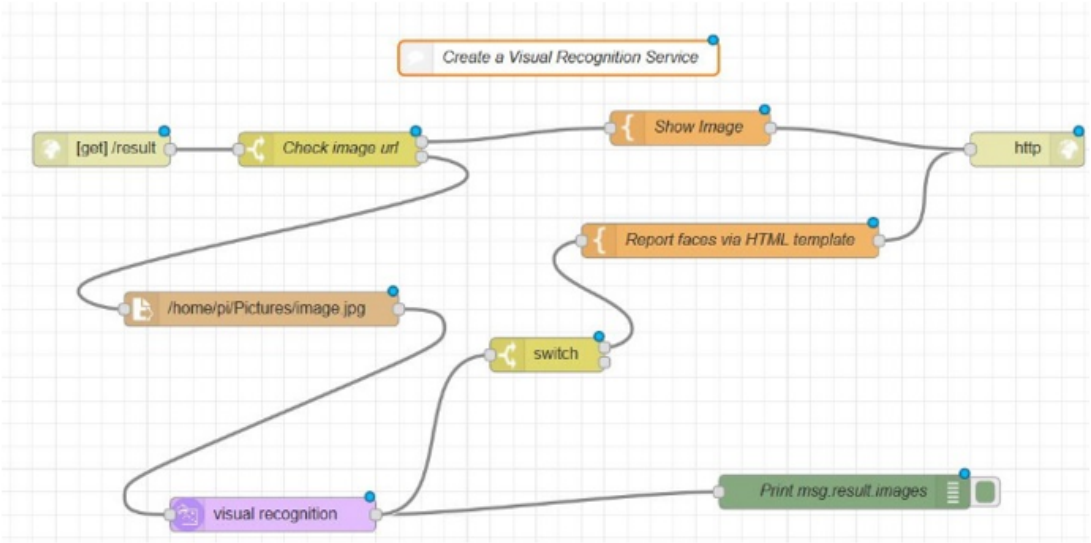


Fig. 3. Node-Red flow for visual recognition

A. *Enhanced Surveillance Camera System*

Node-red programming allows further enhancement of surveillance system by improving the streaming quality and security features. The current system produces the streaming video with a big delay in movement display. In order to improve the quality, we switch from an external camera with a Pi camera. Next, the security features to the surveillance system is added. We implement a control center through web interface for the management of various cameras in the system including the streaming. Only the authorized users can access the system. In addition, to web interface, we create a database to store the camera information, user information, and users who own the camera. In addition, to secure the confidential information for the users, we use node-red element BCrypt to encrypt the sensitive information such as password before inserting into the database.

B. *Enhanced Streaming Quality*

To enhance the quality of the streaming, we replace the external camera with a Pi camera for better quality. In Raspberry Pi, there is a port which is used for connecting Pi camera. After attaching the ribbon into the port. Next, we need to enable the Camera option in the Raspberry Pi configuration to finish the basic set-up process for the Pi camera to work with Raspberry Pi as shown in Figure 4.

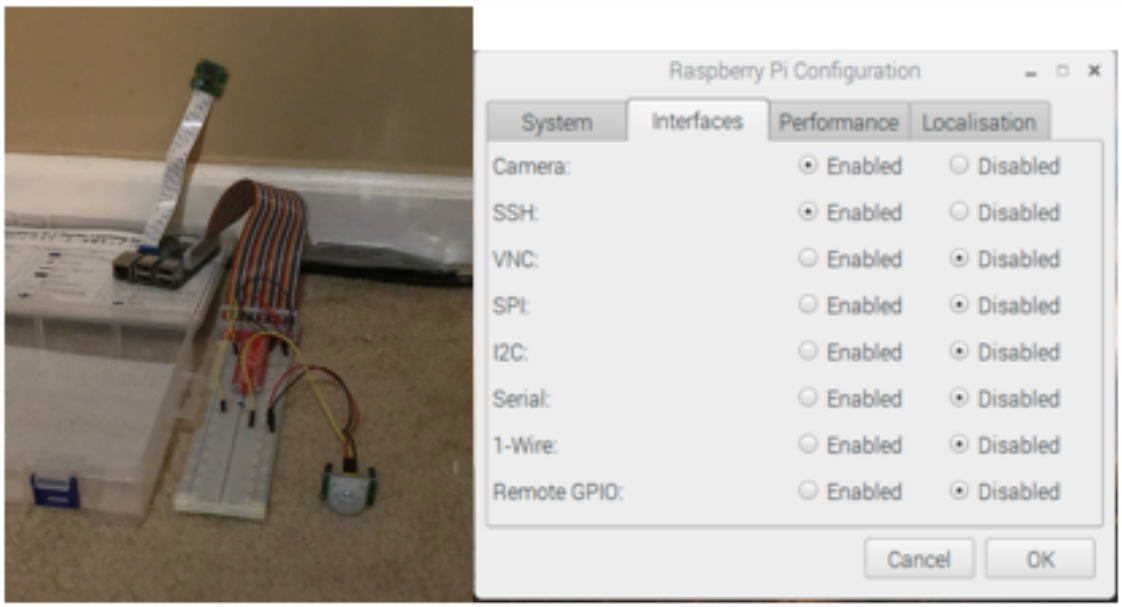


Fig. 4. Pi camera setup and configuration

ISSN: 2414-4592
DOI: 10.20474/jater-5.6.2

Note that after replacing the external camera with Pi camera, the controllers for picture shooting and streaming needs update. For both controllers, we used a Python program.

C. Taking Picture and Streaming Video

In Python, there is a package called picamera (Picamera). This package provides a Python interface to the Raspberry Pi camera module [22] based upon various recipes provided by IBM Watson labs [23]. There are several tools within the picamera package. However, we used only the primary tool. This class contains a basic function that a Pi camera needs to operate such as "capture," for taking a picture and "recording," for recording a video (Class Programming) [24]. With recording, there are two states, "start_recording" and "stop_recording." In addition, the class also has some useful functions such as start_preview() which is used for previewing the camera. Figure 6a shows the implementation code in Python for capturing an image and saving its location [25].

```
from picamera import PiCamera

camera = PiCamera()

camera.capture('/home/pi/Pictures/image.jpg')
```

Fig. 5. Python code for image capturing

```
class StreamingServer(socketserver.ThreadingMixIn, server.HTTPServer):
    socketserver.TCPServer.allow_reuse_address = True
    daemon_threads = True
```

Fig. 6. Streaming handler class to create a URL

The second function we used is recording, which is also used for streaming video over the internet. For streaming, we used the available Python code from picamera [26].readthedocs.io under the section called Web-Streaming. For this script, there are three different classes. They are "StreamingOutput()," "StreamingHandler()," and "StreamingServer()." StreamingServer() class creates the server for the streaming.

Figure 6b shows the snippet of StreamingHandler() class that is used to create the website URL and get the content of the video from the streamingOutput(). Figure 7 shows the set-up of the website is passed to the class StreamingHandler() which is used to create the website for streaming. The streaming website operates on port 8000.

```
class StreamingHandler(server.BaseHTTPRequestHandler):
    def do_GET(self):
        if self.path == '/':
            self.send_response(301)
            self.send_header('Location', '/camera.html')
            self.end_headers()
        elif self.path == '/camera.html':
            content = PAGE.encode('utf-8')
            self.send_response(200)
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(content))
            self.end_headers()
            self.wfile.write(content)
```

Fig. 7. Website setup for streaming

D. Setting Up Service for Streaming

After setting up the Python code for streaming, we create a system service which runs the streaming at startup allowing the streaming to run when the Raspberry Pi boots up.

In Linux system, service is one of the systemd units. Systemd is "systemd is a Linux initialization system and service manager that includes features like on-demand starting of daemons, mount and automount point maintenance,

ISSN: 2414-4592
DOI: 10.20474/jater-5.6.2

snapshot support, and processes tracking using Linux control groups" [27]. To define a service, there are three items namely, Unit, Service, and Install that must be included. The website setup code is used to set up a service which runs at startup as shown in Figure 8.

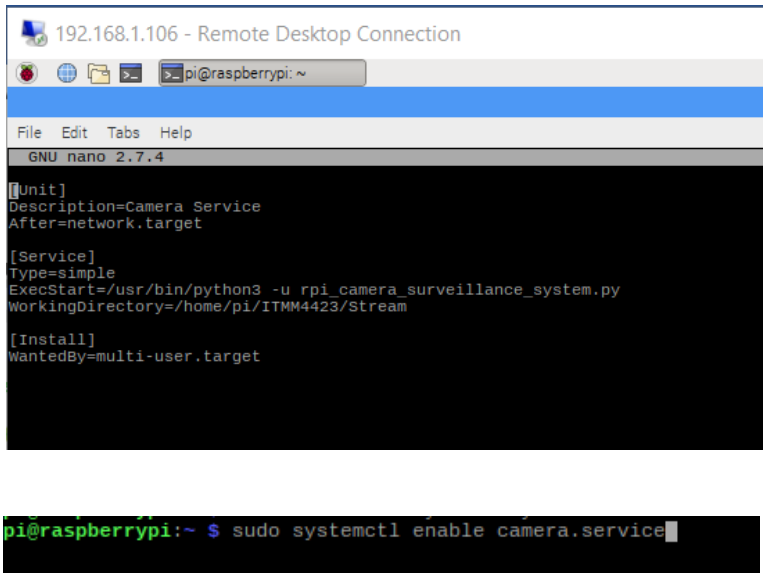


Fig. 8. System startup code

The condition for the code to run is after the Raspberry Pi connects to the network. The "After" keyword in the [Unit] section defines the condition. The [Service] section is where the code gets executed with the keyword "ExecStart". "WorkingDirectory" is used to define where the file can be found. Systemctl is the command that is used to manage the service such as enable, disable, start, or stop the service. The start is used to start the service immediately, and stop is used for immediate stop as opposed to using enable, which is used to set the service to start only at boot time.

III. CONTROL CENTER FOR THE SURVEILLANCE SYSTEM

The best way to provide a secure channel for clients to experience their Raspberry Pi product is through a website as the security center (Figure 9). The website makes use of a secure login, a dashboard of the Raspberry Pi Cameras, and a live stream area. If the client does not have an account with the Raspberry Pi Security Center, they can create one to access their product's live stream.

Fig. 9. Security center login

ISSN: 2414-4592
DOI: 10.20474/jater-5.6.2

After the client logs into the website, a session is created to add another layer of security. A session is a way of tracking who is logged into the website, and to prevent anyone logged out from accessing any pages without logging in first.

However, if the user enters an incorrect email/password combination a message is displayed and they are directed back to the login page, and a session is not created.

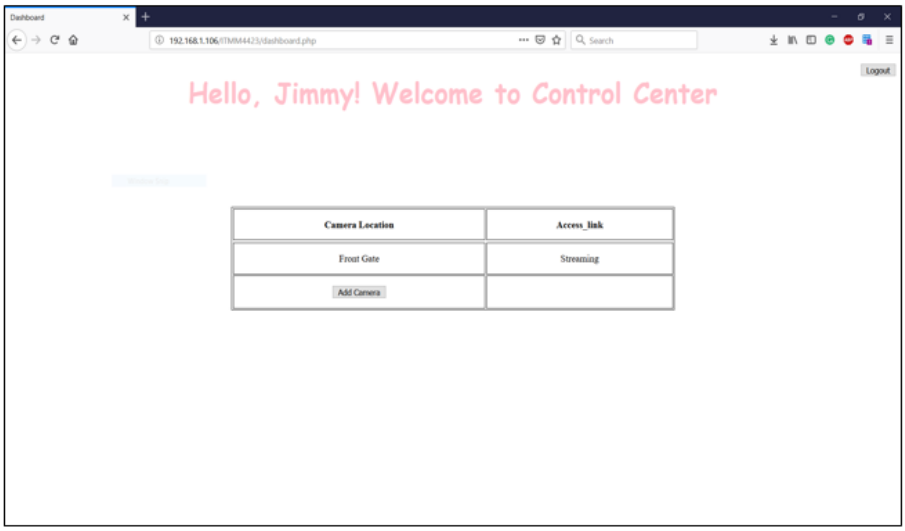


Fig. 10. Control center to access security cameras

Once login is complete, the user will see a list of all their Security Center cameras on the dashboard and the location of the product (Figure 10). Also, there is a link to take to a live video stream. At the stream page, users watch the video and can see the AI statistics regarding any human motion detected by the Raspberry Pi's.

To add login security, the client's passwords are hashed when an account is created and stored in a PostgreSQL database (PostgreSQL). The passwords in the database are stored in their hashed form for further protection. When a user goes to sign in, the form [8] what the user enters and goes through a function to compare what they entered to what is in the database. The Hashing method that we used is called BCrypt.

A. Password Security using BCrypt

To allow for login security in our Raspberry Pi Security Center, BCrypt is used to hash the client's passwords. Hashing is the process of securing information by creating a random value of alphanumeric characters strung together after inputting such information into a function. Bcrypt is a hashing function that is designed based on the Blowfish block cipher cryptomatic algorithm and takes the form of an adaptive hash function.

BCrypt guarantees three core properties of a secure password function:

1. It's preimage resistant.
2. The salt space is large enough to mitigate precomputation attacks, such as rainbow table. 3. It has an adaptable cost.

Figure 11 shows an example of the algorithm that accepts a text password and outputs a hash using BCrypt.

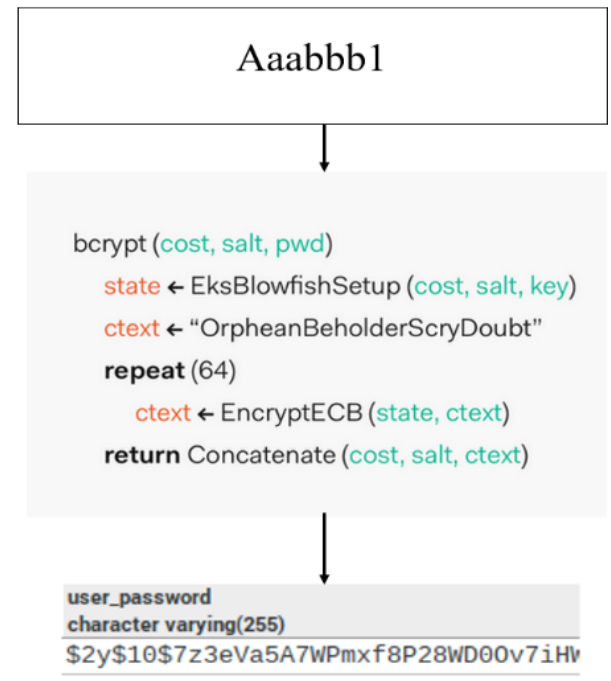


Fig. 11. Control center to access security cameras

ISSN: 2414-4592
DOI: 10.20474/jater-5.6.2

B. Webserver Setup

To allow accessing the website from Raspberry Pi from machines in the same network, it requires a web server to be installed. The first application that needs to be installed is the server application Apache (Apache). "Apache is a popular web server application you can install on the Raspberry Pi to allow it to serve web pages." (Setting up an Apache Web Server on a Raspberry Pi). Apache can serve HTML files over HTTP. It also has modules that allow dynamic web pages using PHP, scripting language. Here are two steps to install Apache into the Raspberry Pi.

After installing the Apache, the website now can be accessed by another machine that is in the same network. The machine will access the website by the IP address of the Raspberry Pi. The location that is used to add files for the website is "/var/www/html/".

Aside from HTML, we also use PHP to access the data from the database. Therefore, we also install database into the Raspberry Pi.

C. Database Setup

The next setup to support the control center is the database. For this project, we use PostgreSQL as our database. PostgreSQL is "PostgreSQL is a powerful, open source object-relational database system with over 30 years of active development that has earned it a strong reputation for reliability, feature robustness, and performance" [28]. Figure 12 shows the command that is used to install PostgreSQL.

After installing PostgreSQL, we continue installing pgAdmin which is a graphical tool that allows access to the database. It allows users to create a database and manage it. It also allows users to write and execute SQL statement.

```
sudo apt-get update
```

Then, install the `apache2` package with this command:

```
sudo apt-get install apache2 -y
```

```
sudo apt-get install php libapache2-mod-php -y
```

```
sudo apt install postgresql libpq-dev postgresql-client
postgresql-client-common -y
```

```
sudo apt install pgadmin3
```

Fig. 12. Control center to access security cameras

IV. AI ENHANCEMENT IN NODE-RED

AI enhancement requires an update in node-red flows due to replacement of external camera by Pi camera. First, the streaming needs to be paused in flow to be able to take a picture. Next, the Python code is activated to enable picture taking, followed by activation of streaming. We resume the service by the command systemctl start as before. The changed first flow is shown in Figure 13.

ISSN: 2414-4592
DOI: 10.20474/jater-5.6.2

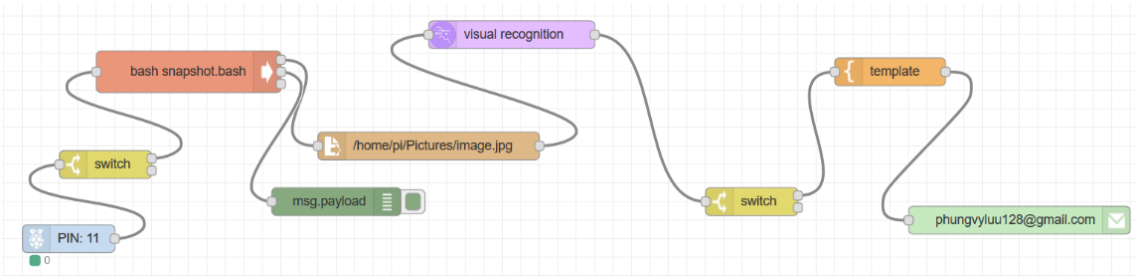


Fig. 13. Improved first node-red flow to implement AI-based visual recognition

A. Application Design

Mobile application is taking over desktop applications because mobile devices are more actively connected to the internet. Therefore, an application could utilize a software more effectively resulting in higher revenues. We used the following steps to create our application.

1. Application goals/design:

a) The application will serve as the user interface and controller between the homeowner and the raspberry pi surveillance camera system.

b) The system will send a notification to the home-owner's email when it detects motion.

c) The homeowner will login.

d) Once logged in, the user will have the ability to select a live feed from each camera.

e) After selecting the camera, user will have the ability to take a picture from live feed.

f) User will be able to select library from home screen.

g) System has a settings menu at the home screen that will have the options to add/remove cameras, adjust sensitivity, change account information.

h) Logoff.

2. Research similar applications:

Comparing with other applications, our application goals were refined as follows:

a) System must be user friendly and easy to use

b) System should allow an ability to watch live video stream

c) Motion detection and configuration of the motion sensor should be implemented

d) Data save/delete should be possible

e) Notify user via email motion is detected

f) Snap photos from live video stream

g) Work with any (or most) webcams

h) It should be easy to add/remove camera

3. Create a use case diagram:

A case diagram for the software design as explained above is shown in Figure 14. Use Case Diagrams represents the behavior diagram of our system. It models the functions of supported by the application using actors and use cases. In our Raspberry Pi Home Security System, we will only have one actor and that would be the homeowner. Our use cases include login, settings, add/remove camera, configure motion sensor, view live feed, capture snapshot, and logout.

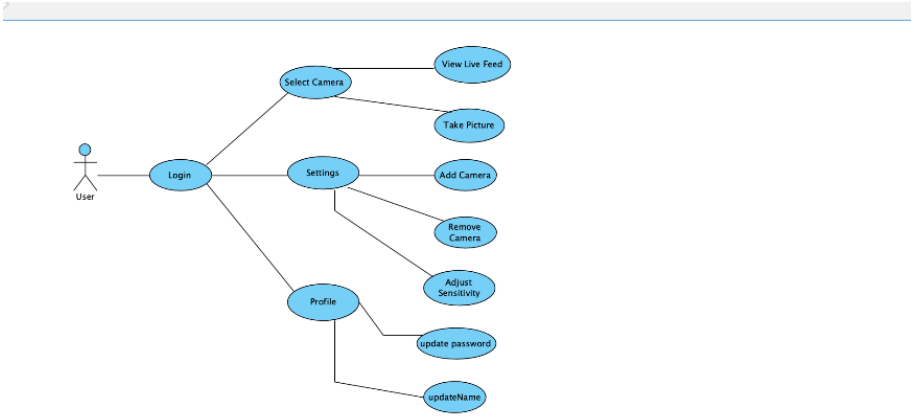
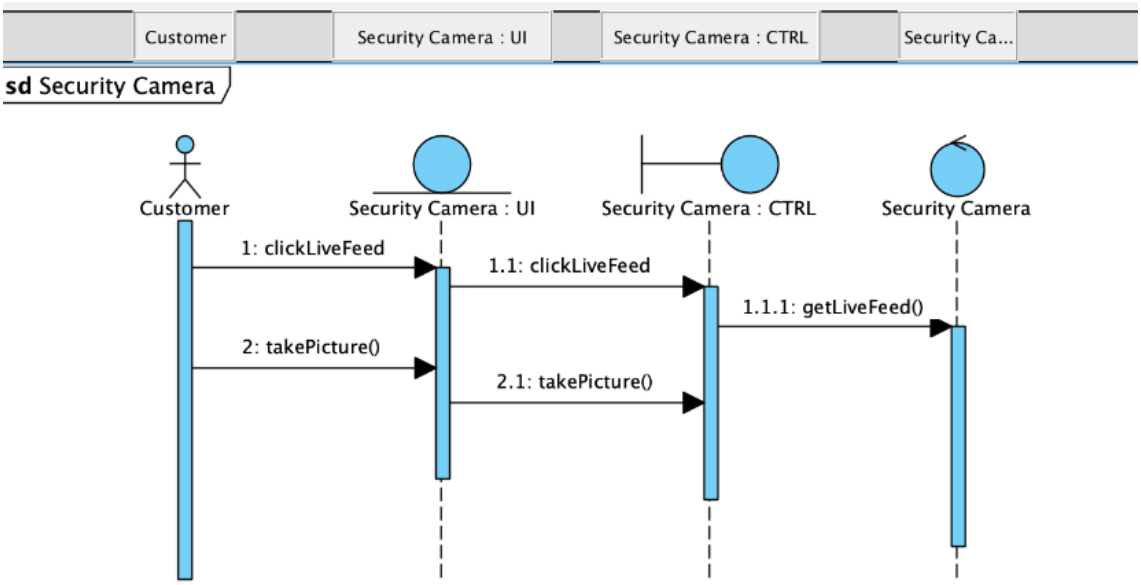


Fig. 14. A case diagram for the software design

ISSN: 2414-4592
DOI: 10.20474/jater-5.6.2

4. Create a Class diagram:
The class diagram (Class Diagram) shows the structure of the system or the blueprint (Figure 15). It shows the relationship between the classes, objects, attributes, and operations. There are active classes that control the flow of activity. A class is a template definition of a method and corresponding variable in an object. The visibility markers show who can access the information in a class.
a) Create Sequence diagram
b) Create a user interface
c) Choose program development path
d) Build and test application using a programming language (we used Java).
e) Debug and finalize application



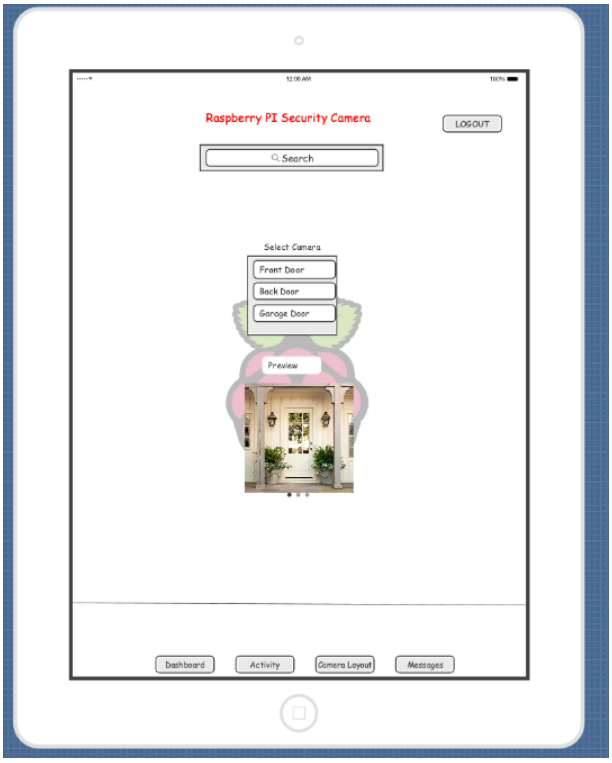


Fig. 15. Class diagram for the web interface program

ISSN: 2414-4592
DOI: 10.20474/jater-5.6.2

V. CONCLUSION

The Raspberry Pi is a great way to create a home surveillance system for not much money. It can be created as a D.I.Y project with not many parts and can offer the same or better features than many retail security cameras can. All one needs is a PI, a USB camera or Pi camera, and optionally a motion sensor (w/breadboard and wires). This can change the landscape of home security and put more power in the hands of the consumer rather than the manufacturer.

REFERENCES

[1] V. Tsakanikas and T. Dagiuklas, ``Video surveillance systems-current status and future trends,'' *Computers & Electrical Engineering*, vol. 70, no. 7, pp. 736-753, 2018. doi: https://doi.org/10.1016/j.compeleceng.2017.11.011

[2] C.-S. Fan, J.-M. Liang, Y.-T. Lin, K.-R. Wu, K.-Y. Li, T.-Y. Lin, and Y.-C. Tseng, ``A survey of intelligent video surveillance systems: History, applications and future,'' in *Intelligent Systems and Applications*. Californi, CA: IOS Press, 2015, pp. 1479-1488.

[3] J. Chen, K. Li, Q. Deng, K. Li, and S. Y. Philip, ``Distributed deep learning model for intelligent video surveillance systems with edge computing,'' *IEEE Transactions on Industrial Informatics*, vol. 4, no. 7, pp. 45-60, 2019. doi: https://doi.org/10.1109/TII.2019.2909473

[4] S. Mittal and S. Mittal, ``Gender recognition from facial images using convolutional neural network,'' in *Fifth International Conference on Image Information Processing (ICIIP),* Bangkok, Thailand, 2019.

[5] A. R. Shinwari, A. J. Balooch, A. A. Alariki, and S. A. Abdulhak, ``A comparative study of face recognition algorithms under facial expression and illumination,'' in *21st International Conference on Advanced Communication Technology (ICACT),* Boston, MA, 2019.

[6] E. Jose, M. Greeshma, M. T. Haridas, and M. Supriya, ``Face recognition based surveillance system using facenet and mtcnn on jetson tx2,'' in *5th International Conference on Advanced Computing & Communication Systems (ICACCS),* New Jersy, NJ, 2019.

[7] L. Paletta, S. Wiesenhofer, N. Brandle, O. Sidla, and Y. Lypetskyy, ``Visual surveillance system for monitoring of passenger flows at public transportation junctions,'' in *Proceedings of IEEE Intelligent Transportation Systems,* California, CA, 2005.

[8] B. Heiselet, T. Serre, M. Pontil, and T. Poggio, ``Component-based face detection,'' in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* New York, NY, 2001.

[9] X. Guo, Z. Shen, Y. Zhang, and T. Wu, ``Review on the application of artificial intelligence in smart homes,'' *Smart Cities*, vol. 2, no. 3, pp. 402-420, 2019. doi: https://doi.org/10.3390/smartcities2030025

[10] E. Alajrami, H. Tabash, Y. Singer, and M.-T. El Astal, ``On using ai-based human identification in improving surveillance system efficiency,'' in *International Conference on Promising Electronic Technologies (ICPET),* London, UK, 2019.

[11] T. M. Team, ``Exposed video streams: How hackers abuse surveillace cameras,'' 2018. [Online]. Available: https://bit.ly/3xZ4Geg

[12] G. Jevtic, ``17 best security penetration testing tools the pros use,'' 2019. [Online]. Available: https://bit.ly/2V8TxJd

[13] D. Miessler, ``Masscan examples: From installation to everyday use.'' 2019. [Online]. Available: https://bit.ly/3BtYH3k

[14] Kali Tools, ``Router scan,'' 2016. [Online]. Available: https://bit.ly/3x1bO8y

[15] M. Parreno, ``How to access your raspberry pi camera from anywhere,'' 2018. [Online]. Available: https://bit.ly/3hUeNeN

[16] O. A. Osahenvemwen and O. F. Odiase, ``Effective management of handover process in mobile communication network,'' *Journal of Advances in Technology and Engineering Studies*, vol. 2, no. 6, pp. 176-182, 2016. doi: https://doi.org/10.20474/jater-2.6.1

[17] M. West, ``Building a motion activated security camera with the raspberry pi zero,'' 2017. [Online]. Available: https://bit.ly/2VfKDtL

[18] M. Short and E. Joel, ``How to use a breadboard,'' 2018. [Online]. Available: https://bit.ly/2UyXXcK

[19] D. Albano, `` RPi node-red: PIR RGB LED or buzzer,'' 2018. [Online]. Available: https://bit.ly/2V3yFDh

[20] Nodejs, ``Node-versions,'' n.d. [Online]. Available: https://bit.ly/3eJR2Es

[21] H. S. Alam, D. Soetraprawata, Bahrudin, T. Haiyunnisa, T. I. Salim, A. Munandar, and D. Setiawan, ``Design of stand-alone irrigation system on strawberry cultivation powered by wind turbine and photovoltaics,'' *International Journal of Technology and Engineering Studies*, vol. 2, no. 5, pp. 154-163, 2016. doi: https://doi.org/10.20469/ijtes.2.40005-5

ISSN: 2414-4592
DOI: 10.20474/jater-5.6.2

[22] D. Jones, ``Picamera - web-streaming,'' 2019. [Online]. Available: https://bit.ly/3Bv01mt
[23] IBM, ``Enterprise ready AI,'' n.d. [Online]. Available: https://ibm.co/3iyWX02
[24] Wikipedia, ``Class (programming),'' n.d. [Online]. Available: https://bit.ly/3hYdEmH
[25] Smartdraw, ``Class diagram,'' n.d. [Online]. Available: https://bit.ly/3ixhi5W
[26] Picamera, ``Getting started with picamera,'' n.d. [Online]. Available: https://bit.ly/3zx9c4h
[27] Linode, ``What is systemd?'' 2018. [Online]. Available: https://bit.ly/2UE7Ci6
[28] Postgre SQL, ``Postgresql: The world's most advanced open source relational database,'' n.d. [Online]. Available: https://bit.ly/3rtpzfa

ISSN: 2414-4592
DOI: 10.20474/jater-5.6.2