Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 165 (2019) 624–630

Procedia Computer Science

www.elsevier.com/locate/procedia

INTERNATIONAL CONFERENCE ON RECENT TRENDS IN ADVANCED COMPUTING 2019, ICRTAC 2019

Image-Based Smart Surveillance and Remote Door Lock Switching System for Homes

Jay Patel*, Sundar Anand, Rohan Luthra

Vellore Institute of Technology, Vandalur-Kelambakkam Road, Chennai 600127, India

Abstract

Internet of Things (IoT) has found multiple use-cases when it comes to our homes. Smart home-surveillance is one of them. This work presents a prototype of a smart surveillance system that works with photos instead of videos. Not only such a system is more cost-efficient, but it is also one that limits internet traffic. On particular triggers, the system clicks a photo, and in case if it is some person who triggered the system, the taken image is labelled using a facial recognition service that labels the person based on the images uploaded by the system's user/s via a mobile application beforehand. This paper also presents a secure and reliable mechanism for remote switching of the house's doors using the same mobile application. An added feature in the door-lock switching mechanism is that of an onsite greeting system.

Keywords: Smart surveillance; Remote door lock switching; Greeting system

1. Introduction

1.1. Need for the proposed system

The world evolving towards smart cities certainly promises to make people's lives easier. While speaking of smart cities and homes, smart security and surveillance also come into the picture[1]. It is important to understand that not every security use-case requires a bulky, costly, sophisticated, and heavy internet traffic generating surveillance system. Most existing security and surveillance systems consisting of closed-circuit television (CCTV)

*Corresponding author: Jay Patel; jaypatel.01@zoho.com

10.1016/j.procs.2020.01.056

cameras, though sophisticated, require human supervision to achieve real-time capabilities; not to mention the high power consumption[2]. Besides, smart cities are bound to bring their own inherent challenges-extremely high internet traffic being one of them. The problem owes itself to a myriad of devices that are to connect to the internet. The issue of real-time home security needs to be tackled whilst keeping in mind the internet traffic challenge.

1.2. System overview

Home security and surveillance, as well as remote door switching with a greeting system are two aspects of this work. The journey of a user of the system starts with the installation of the hardware parts of the system for the security and surveillance aspect. Note that for the remote door unlocking aspect, the presence of the door is assumed, and there is no physical lock in our prototype. However, an LCD screen as the greeting system for the house is included. The internet can be used for the remote control of devices[3]. For the remote door lock switching aspect, the main task in this work is to propose a secure and reliable mechanism; its implementation requires highly sophisticated and advanced locks. Now, after the hardware installation, the users at their discretion need to upload a few pictures of people who they wish to mark as 'known' (along with a name-tag). There are options as to click the photos and upload or choose the photos from the phone gallery to upload. The user uses the mobile application for this purpose. In the prototype presented in this work, it is an Android application. A facial recognition service extracts and stores the facial features present in these uploaded images along with the name-tag.

The system prototype consists of a small computer (Raspberry Pi Model 3B+), a camera (Raspberry Pi Camera Module v2), a cloud service - Amazon Web Services (AWS), a proximity sensor (infrared sensor), a switch (to mimic a doorbell), an LED bulb (to mimic the door), an LCD screen, and an Android application [4][5][6]. The camera is triggered on two events – either when someone presses the doorbell or when someone is in proximity for more than a predefined threshold of time period. The photo clicked is uploaded to the cloud, wherein the facial features are extracted and compared with the 'known' faces. The results – (whether known or unknown and if known then who) are stored on the cloud, besides simultaneously being sent to the user application and registered email/s. On the application, the user is alerted via notification, and they can view the image as well as the associated label. The user also has an option to view the entire history. The advantage of labelling the image is that sometimes the person at the door or in the proximity may have their face covered, but with state-of-the-art facial recognition algorithms, partially covered faces which may not be identified by humans, are identified very accurately. Our test results support this. Numerous tasks can be achieved for people visiting the house[7]. For switching the door remotely, the user has an option in the mobile application itself. If the user unlocks the door for a known person, an LCD screen displays a personalized welcome message.

1.3. Service Oriented Architecture overview

In the prototype, the Raspberry Pi, the Raspberry Pi camera, the proximity sensor, and the LCD module are the physical components present on-site. At the backend, besides the script which is responsible for tasks like taking inputs from the doorbell and the proximity sensor, triggering the camera at appropriate times, relaying images to the cloud, and switching the door lock on receiving a command, there are a number of services involved. They are provided by the AWS, which employs Service Oriented Architecture (SOA).

SOA is an architectural paradigm in the design of software wherein application components provide services to other components using communication protocols over a network. As a result, a number of services can be used in conjunction to form a larger software application. In AWS, the services have Application Programming Interfaces (APIs) available, which can be used programmatically via AWS Software Development Kits (SDKs). In the presented prototype, Amazon S3, Amazon DynamoDB, Amazon Rekognition, and AWS Lambda are the services that are used.

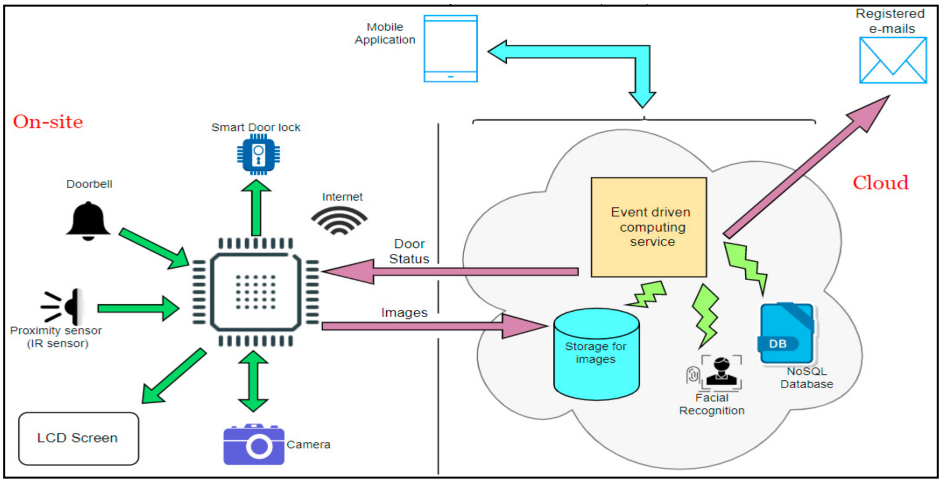


Fig. 1. Architecture of the system

2. System architecture

The prototype only serves to be a Proof of Concept (PoC) representative of an architecture that is proposed here. Fig. 1. depicts the overall architecture of the system. The on-site unit consists of a computer with an active internet connection, besides a proximity sensor, a doorbell, a smart door lock, a camera, and an LCD screen – all of them interfaced with the computer. The cloud-side unit in broad sense consists of four services – storage areas for images, a facial recognition service or program, NoSQL databases – one for storing facial features and another for storing door lock's status, and an event-driven computing service which performs tasks (Table 1) on particular triggers (Table 2).

There are two storage units for images – one for storing the images that are uploaded by the user through the application (Image storage 1), and another one for storing the images uploaded by the on-site computer (Image storage 2). The event-driven service helps upload images from the application, and upon arrival of an image to storage 1, transfers a copy of the image to the facial recognition service, which extracts and stores the facial features in a NoSQL database. These features are tagged so that features extracted from multiple images of the same person pertain to that single person only. On arrival of an image to storage 2, that is when an image arrives from the on-site computer, the same event-driven computing service first transfers the image to the facial recognition service, which extracts the facial features and then makes a comparison with the features stored in storage 1. If a confidence value of 90% or more occurs, then a match is made (that is, it is a known person around the house), else it is an unknown person.

For door lock switching, a slider button on the application is used. For storing the status of it, a NoSQL database providing key-value storage is used. The key here is 'door_status' and the value is kept binary – 0 or 1, with 0 indicating that the door is locked and 1 indicating that the door is unlocked. This change is realized in the database, and an event-driven service monitoring the database notices the change and relays it to the on-site processing unit, which in turn switches the lock into the appropriate state. If the main door's lock is switched open for a known person, then an LCD screen at the door greets the person with a message "Welcome 'person'", where 'person' represents the name-tag with which the image of the person at the door is stored.

Table 1. Triggers for the event-driven computing service

Triggers
1. Hitting of upload button on the application
2. Successful storage of an image to storage 1
3. Successful storage of an image to storage 2
4. End of matching of facial features extracted from an image in storage 2 to the ones stored in the NoSQL database.
5. Switching the door lock's slider button on the application
6. Successful change of the value of 'door_status' in another NoSQL database.

Table 2. Tasks done by the event-driven computing service

Task	From	To
Transfer image	Application	Image Storage 1
Transfer image on arrival	Image storage 1	Facial recognition service or program
Transfer image features	Facial recognition service or program	NoSQL database 1
Transfer image on arrival	Image storage 2	Facial recognition service or program
Alert on app and e-mail	Face match results	App and e-mail
Transfer door lock's status	Application	NoSQL database 2
Transfer door lock's status	NoSQL database 2	On-site computer

3. System prototype and working

3.1. System prototype

Table 3. Hardware components in the prototype

Hardware
Raspberry Pi Model 3B+
Raspberry Pi camera module v2
Proximity sensor (IR sensor)
LCD module

Table 4. Software components in the prototype

Software
Raspbian Stretch OS (kernel v4.14)
Python IDLE (for program development)
AWS Services – S3, DynamoDB, Rekognition, Lambda
Android Studio

The connections of the hardware components in the prototype are depicted in Fig. 2. A Raspberry Pi model 3B+ is used as the main computer at the site, that is at home. In the presented prototype it runs Raspbian Stretch OS. Before Pi's use in the development process, it is accessed using the Secure Shell (SSH) protocol, and its Virtual Network Computing (VNC) server is enabled. After that, anytime this Pi is to be worked with, it is accessed using the VNC viewer software. The main advantage of this is that Pi can be used with the peripherals of a desktop computer or a laptop, without having to externally attach an HDMI screen, a mouse, and a keyboard to it. In the security system, the Pi has multiple tasks. It is responsible for monitoring the proximity sensor, and on particular triggers make the Pi-camera capture an image. The image is to be relayed to an appropriate cloud service. Besides these tasks, it has to listen to another cloud service that informs Pi as to whether to open or close the door's lock and if the door is to be opened, it has to fetch a name-tag to be displayed on the LCD screen message. The Raspberry Pi constantly needs to be connected to the internet. A camera module connected to the Pi is responsible for capturing an image on either of the two triggers – the first one is the pressing of the doorbell which is also interfaced with the Raspberry Pi. In the prototype the doorbell is represented using a push-button. The second trigger is the presence of anyone in the vicinity of the house (proximity sensor) for more than a pre-defined threshold of time. An IR sensor is used as the proximity sensor in the presented prototype. For the door-switching system, an LED bulb represents the switching of the door that happens after the command is relayed to the Raspberry Pi from the mobile application via the cloud. The LCD screen is represented using an LCD module.

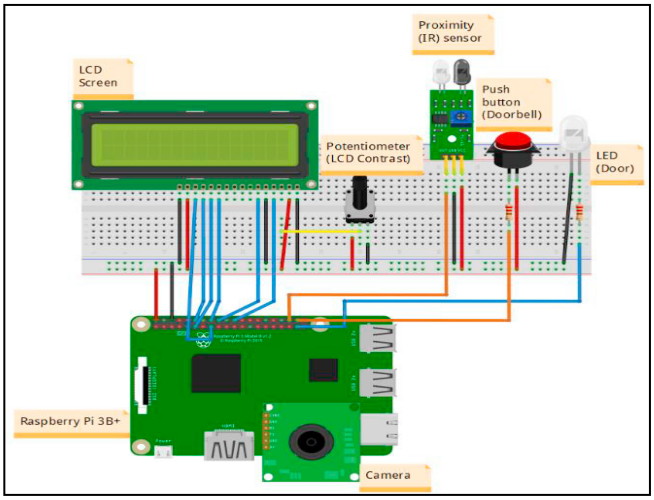


Fig. 2. System prototype

3.2. Working and testing

Table 5 shows the services employed from AWS for the development of the system. Figures 3 and 4 depict the workflow of the prototype system.

Table 5. Services employed from AWS

Service	Purpose	Analogy to architecture
Amazon S3	Storage of images	Image storage (1 and 2)
Amazon DynamoDB	Storage of facial features and door lock's status	NoSQL database (1 and 2)
Amazon Rekognition	Facial features' extraction	Facial recognition service
AWS Lambda	*Refer to Table 2*	Event-driven computing service

Different AWS Lambda are employed to serve our purpose. Each contains a program to run when triggered.
The user in the mobile application enters e-mail/s to which the notification is to be sent to (besides the mobile application). Say the user wants to upload three pictures of a known person who we call Mr. X. The user then either selects three pictures from the photo gallery or they click three pictures via the mobile application window. The user then enters a name-tag and hits 'UPLOAD IMAGE'. Now, a Lambda connected to the application is triggered, and it transfers the photos to an Amazon S3 bucket called 'Known'. As soon as the transfer is done, another Lambda is triggered which takes the images one by one and passes them onto Rekognition, which extracts the facial features, and stores them to a NoSQL database DynamoDB with the tag the same as the name-tag entered by the user before uploading the image from the application.

To understand the working of the facial recognition and alert system, say, a person lurks around the house for more than 10 seconds which is the pre-defined threshold of time. The IR sensor senses this and as a result the Raspberry Pi checks if a presence is sensed for more than 10 seconds, in which case the camera is triggered to take a picture. This picture is named with the date and time stamp and uploaded to an S3 bucket called 'Visitors'. Just as the upload is done, a Lambda realizes this and transfers the image to Rekognition. After Rekognition is done extracting features, another Lambda matches them to the ones present in the 'Known' database. If a confidence value of 90% or over is achieved, a match is made, and the name-tag of the match is extracted. If the confidence value is under 90%, no match is made. Either way, the results are sent to the registered e-mail/s. On the application, a notification is sent, and in a window the user can see the photo taken by the Raspberry Pi camera, as well as the associated tag which is either 'unknown' or the name-tag if the person outside the house is known. This same working occurs if the person at the door presses the doorbell. Just as a person presses the doorbell, a picture is clicked, and a time buffer of 5 seconds is given so that multiple pictures are not taken in duration during which the doorbell switch is kept pressed.

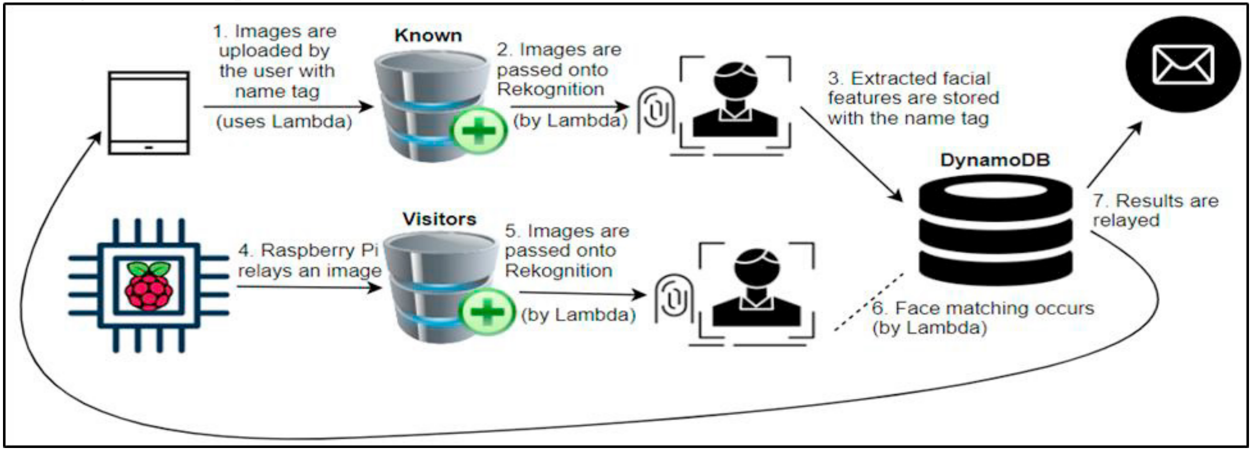

1. Images are uploaded by the user with name tag
(uses Lambda)
Known
2. Images are passed onto Rekognition
(by Lambda)
3. Extracted facial features are stored with the name tag
DynamoDB
7. Results are relayed
Visitors
4. Raspberry Pi relays an image
5. Images are passed onto Rekognition
(by Lambda)
6. Face matching occurs (by Lambda)

Fig. 3. Working of security system

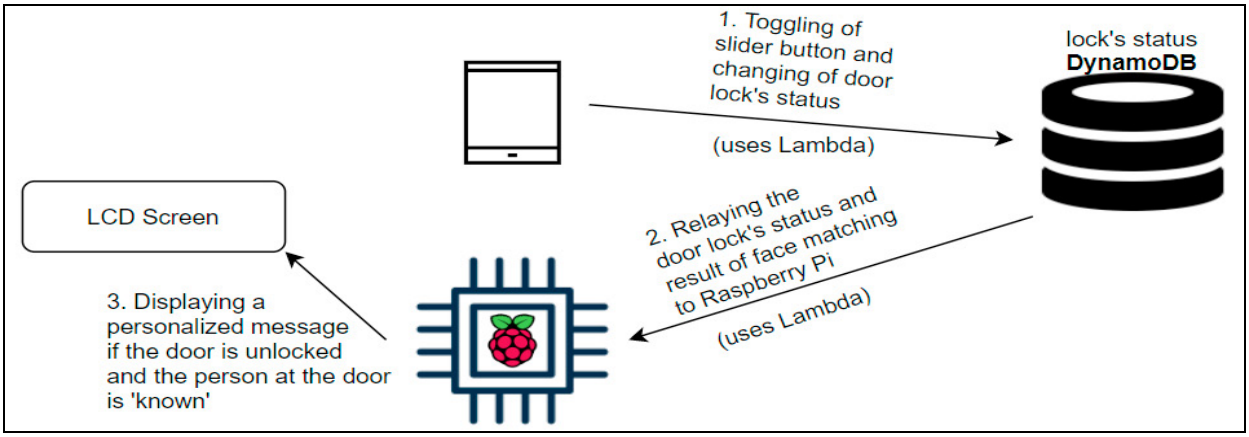

1. Toggling of slider button and changing of door lock's status
(uses Lambda)
lock's status
DynamoDB
LCD Screen
2. Relaying the door lock's status and result of face matching to Raspberry Pi
(uses Lambda)
3. Displaying a personalized message if the door is unlocked and the person at the door is 'known'

Fig. 4. Working of remote door lock switching system

Now, say the person at the door is Mr. X, and the alerted users want to let him in. For this, the user switches the slider button on the app. After the user tries to toggle it, a prompt comes up asking if the user is sure of the operation in order to avoid slip-of-hand mistakes. Only after the user agrees to the prompt, the button toggles. This button on toggling triggers a Lambda which takes the status value and updates it on another DynamoDB database which only contains a key-value pair. The key is ‘door_status’, and the value is either 0 or 1. 1 denotes open and 0 denotes closed. As soon as this value update occurs, another Lambda takes this value (1) from the database and transfers it to the Raspberry Pi, which switches the LED bulb on, meaning that the door lock is switched open. The same Lambda also carries with itself the result of the face-match, that is, it carries a name-tag with itself if the door is unlocked for a known person or carries the string ‘unknown’ if the door is unlocked for a person unknown to the system. Since, in this case the person is known and the name-tag is ‘Mr. X’, the LCD module displays “Welcome, Mr. X”. If the person at the door is unknown to the system, but the door is unlocked for them, then no greeting message would show up on the LCD module.

The facial recognition service needed to be checked for the flowing aspects –
1) Whether it worked for a full picture and not just a zoomed-in picture of a face.
2) Whether it worked even when the face of a person known to the system is partially covered.

We tested the system directly at the cloud, whose screenshots are the Fig. 5(b) and 5(c). We uploaded three pictures with the name-tag ‘Sundar Anand’ in order for Rekognition to extract and store the facial features. We then uploaded some other images of ‘Sundar Anand’ shown in Fig. 5(b) and 5(c) in order to test the service and found that the images were labelled correctly by the service.

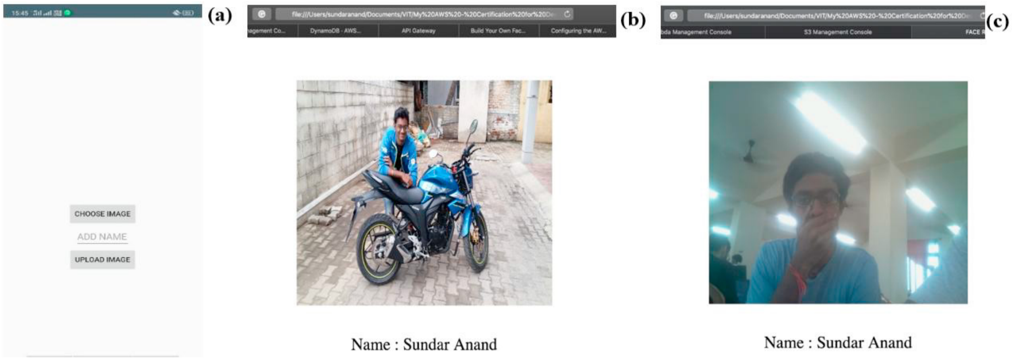
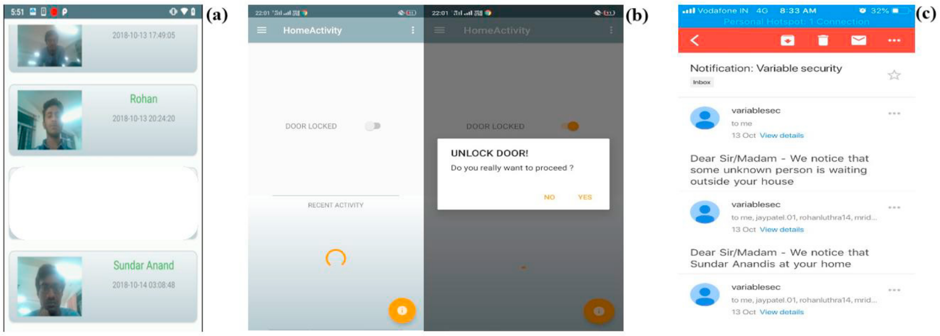


Fig. 5. (a) Upload window in the app; (b) Zoomed out picture taken by the camera; (c) Partially covered face of a person known to the system



UNLOCK DOOR!
Do you really want to proceed ?

Notification: Variable security

variablesec
to me
13 Oct View details

Dear Sir/Madam - We notice that some unknown person is waiting outside your house

variablesec
to me, jaypatel.01, rohanluthra14, mrid...
13 Oct View details

Dear Sir/Madam - We notice that Sundar Anandis at your home

variablesec
to me, jaypatel.01, rohanluthra14, mrid...
13 Oct View details

Fig. 6. (a) Visitor history window in app; (b) Door lock switching button and prompt; (c) E-mail alerts

4. Conclusion

An architecture for a system providing home surveillance by working with images besides providing a remote door lock switching mechanism has been proposed. A prototype utilizing the same architecture was successfully developed as well as tested against a number of test cases. The system achieved a high level of reliability. It can be used in other settings and for multiple door locks, and can also be extended to other use-cases which involve safeguarding and/or surveillance.

References

[1] Anitha, A. (2017, November). “Home security system using internet of things” in *Materials Science and Engineering Conference Series* (Vol. 263, No. 4, p. 042026)

[2] Keat, L. H., & Wen, C. C. (2018). “Smart Indoor Home Surveillance Monitoring System Using Raspberry Pi” in *JOIV: International Journal on Informatics Visualization*, 2(4-2), 299-308

[3] Vaishnavi S. Gunge and Pratibha S. Yalagi, “Smart Home Automation: A Literature Review”, *National Seminar on Recent Trends in Data Mining- RTDM 2016.*

[4] AWS documentation. “https://docs.aws.amazon.com” [Online]

[5] Raspberry Pi documentation. “https://www.raspberrypi.org/documentation/” [Online]

[6] Android documentation for app developers. “https://developer.android.com/docs” [Online]

[7] Kodali, R. K., Jain, V., Bose, S., & Boppana, L. (2016, April). “IoT based smart security and home automation system”. In *2016 international conference on computing, communication and automation (ICCCA)* (pp. 1286-1289). IEEE.