INTERNET-BASED REMOTE CONTROL AND MONITORING SYSTEM FOR COMMERCIAL DOORS USING MOBILE DEVICES

A Thesis by

Mohammad Hossein Erjaei

Bachelor of Computer Science, Shiraz University, Iran, 2017

Submitted to the Department of Electrical Engineering and Computer Science
and the faculty of the Graduate School of
Wichita State University
in partial fulfillment of
the requirements for the degree of
Master of Science

May 2019

INTERNET-BASED REMOTE CONTROL AND MONITORING SYSTEM
FOR COMMERCIAL DOORS USING MOBILE DEVICES

The following faculty members have examined the final copy of this thesis for form and content, and recommend that it be accepted in partial fulfillment of the requirement for the degree of Master of Science with a major in Electrical Engineering.

_____
Ali Eslami, Committee Chair

_____
Abu Asaduzzaman, Committee Member

_____
Hamid Lankarani, Committee Member

DEDICATION

To my parents

“If I had asked people what they wanted, they would have said ‘faster horses’.”

– Henry Ford

ACKNOWLEDGMENTS

I would like to thank my advisor, Professor Ali Eslami, for his support and advice during the time of my Master's degree research. His guidance and patience made it possible for me to develop this thesis. I am also grateful to my committee members, Professor Hamid Lankarani and Professor Abu Asaduzzaman, for generously taking the time to evaluate my thesis.

My appreciation goes out to all my friends, particularly Pouya Ammari Azar and Farshad Mashhadi, for their support when I needed it. I am also thankful for the encouragement of my siblings.

This thesis is dedicated to my parents and my wife for all their sacrifices, endless love, and support along the way.

ABSTRACT

The ability to remotely control a commercial door unit is typically provided by a dedicated remote control. In some cases, local control may also be available via a panel mounted near the door. Such systems allow for controlling (e.g., opening and closing) the door unit from short distances while a user is on-site and actively utilizing the door. However, all remote mobile applications have focused only on non-commercial garage doors. It would be desirable to provide means for enabling the user to control a commercial door unit without using a mounted door panel or dedicated remote control device. This thesis provides such a unit that enables the user to control and monitor the commercial door unit from any distance using a smartphone. Users are able to securely manage multiple doors from one user application loaded on their smartphones. A log including the history of each door's status is also provided. In addition, the unit is easily scalable so that an unlimited number of users could control and monitor many doors on their mobile devices using the same application.

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CSS3	Cascading Style Sheet 3
ERD	Entity Relationship Diagram
HTML5	Hypertext Markup Language 5
ID	Identification
iOS	Apple IPhone Operating System
IP	Internet Protocol
LTE	Long-Term Evolution
MD4	Message Digest 4
MD5	Message Digest 5
MySQL	My Structured Query Language
OS	Operating System
PHP	Personal Home Page
SD	Secure Digital
SHA2	Secure Hash Algorithm 2
SHA3	Secure Hash Algorithm 3
SQL	Structured Query Language
TCP	Transmission Control Protocol
UI	User Interface
Wi-Fi	Wireless Fidelity

CHAPTER 1

INTRODUCTION

1.1 Motivation

Currently all hydraulic doors use a dedicated remote control. Such systems allow for controlling (e.g., opening and closing) the door unit from short distances while a user is on-site and actively utilizing the door. Customers may not have control of the door while it is in operation and thereby unable to stop the door immediately if an emergency occurs. Also, if a customer has more than one hydraulic door, then at least one remote control for each door is needed.

All remote mobile applications have focused on non-commercial garage doors only. However, there is a need for a remote mobile application for commercial doors. This kind of application can reduce cost and improve security for controlling these doors. All of these issues provide the motivation to create a device that allows customers to control their commercial doors by using their smartphone.

1.2 Market Review

Existing devices for controlling commercial hydraulic doors typically use Bluetooth communication, whereby each remote system enables customers to operate their doors from approximately 200–300 feet away [1]. Figure 1 shows four different types of Bluetooth remote controls for Powerlift hydraulic doors. These devices are quite expensive (e.g., about $300 for each remote per door). In the case of a moderately sized business, several remotes are often needed for the employees, which increases the cost significantly.

Figure 1. Powerlift hydraulic door remote controls [1]

Some mobile applications enable customers to control residential garage doors. However, these applications are only for residential garage doors, not commercial doors. Moreover, residential garage door applications do not include any livestream video from the door side. Based on regulations, opening or closing a commercial door should not be allowed without monitoring. Table 1 is showing similar exciting device and application including their positive and negative features.

Table 1. Exciting device and application

Product Name	Powerlift hydraulic door controls	Tap It Open
Advantages	- Compatible with commercial doors - High security	- Simple user interface - Compatible with garage doors
Disadvantages	- Expensive - Operate doors from 300 feet away	- Poor security - Works only with garage doors

1.3 Prior Attempts

Prior to this thesis, many efforts have been made regarding controlling commercial doors and internet-based remote controls for garage doors, some of which resulted in patents such as US20140171032A1, US20150137941A1, US7635966B2, US7224275B2, US3831158A, US4954810A, US6049289A, US20030076235A1 and US20020180600A [2, 3, 4, 5, 6, 7, 8, 9, 10]. We review few of these patents below.

Patent US20140171032A1 provides a method and system for remotely monitoring the open and close status of a garage door via the internet of things and NFC (Near Field Communication) technology. The system includes a Door Monitoring & Controlling Device (DMCD) and a web based Advanced Door Management System [2]. Unlike this method, we are using internet as communication channel which gives us ability of controlling doors over internet from anywhere.

Patent US20150137941A1 discloses an internet-connected garage door control system that includes a garage door opener for opening and closing a garage door in response to signals received through the internet, and an in-vehicle remote garage door opener integrated into a vehicle for transmitting the signals through the internet to the garage door opener [3]. This patent is using in-vehicle remote garage door opener. However, in our introduced unit we are using a mobile application connected to internet to communicate with the door control system.

1.4 Problem Description

The literature indicates that for controlling commercial doors, customers must use Bluetooth remotes, which forces them to be close to the door. The work in this thesis focuses on creating a new control unit that allows customers to manage their commercial doors by using their smartphones from any location.

1.5 Contribution

The main aim of this thesis was to create a unit that gives customers the ability to control their commercial doors with their smartphones. This unit comprises of two components: 1) a mobile application that enables remote control and real-time camera monitoring of commercial doors from a mobile device such as a cell phone or tablet over the Internet from anywhere that provides access to a central server (with an integrated or separate back-end database), and 2) a door controller unit at each controlled door closure. The mobile application would be universal and could be seamlessly used on any Android or Apple IPhone operating system (iOS) device.

The back-end database provides clients with an online management panel (i.e., dashboard) where multiple doors are to be controlled and monitored, and provides report logs for managers who are in charge. The system can handle as many cameras per door as needed for monitoring. It also includes security features for both users and managers, and employs a secure communication protocol to carry out these operations.

The main contributions of this thesis are as follows:

- Development of a mobile application that can be installed on smartphones and can give the user the ability to control the door.
- Configuration of the Raspberry Pi, a tiny computer, on the door side, which connects to the network and allows it to receive commands from the mobile application.
- Setup of a main server that acts as a controller and ensures that both the mobile application and Raspberry Pi are synchronized and secure.
- Application of different security techniques, both on the mobile application and the Raspberry Pi, to ensure that both are secure and persistent.

1.6 Organization

This thesis is organized as follows: Chapter 2 explains the development of the mobile application and different features of the application. Chapter 3 discusses in detail the configuration of the Raspberry Pi, in order to make it ready to receive commands from the mobile application. Chapter 4 provides steps in the main server setup and details of the duties of this part of the unit. Chapter 5 discusses the security concerns of the unit and proposed solutions. Chapter 6 concludes this thesis, including suggestions for future work.

CHAPTER 2

MOBILE APPLICATION DEVELOPMENT

The main steps of any application development are prototyping, programming language selection, user interface (UI) design, and back-end development. Each of these steps are discussed in this chapter.

2.1 Prototype

After collecting user requirements, the first step in application development is creating a prototype. The software prototyping is about building a software application prototype that displays the functionality of the final product but may not actually hold the exact logic of the original software. The software prototype aids in obtaining valuable feedback from the customer, and helps software designers and developers understand exactly what is expected from the product under development [11].

As mentioned above, the software prototype does not need to have all logic of the final product. Therefore, the prototype involves a mobile application consisting of a fake login page and a page to open and close the door. As shown in Figure 2, the second page of the prototype involves two buttons that play a video of the door forward and backward.

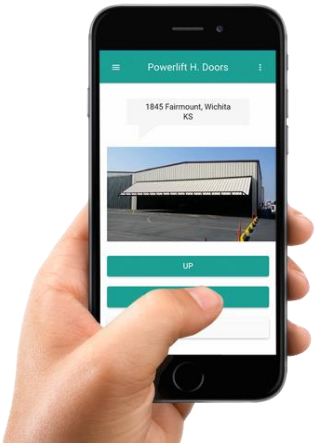

Figure 2. Mobile application prototype

2.2 Language Selection

Selecting the correct programming language to develop an application is one of the most important steps in software development. After analyzing customer requirements, the Apache Cordova, an open-source mobile development framework, was selected. It allows developers to use standard web technologies: Hypertext Markup Language 5 (HTML5), Cascading Style Sheet 3 (CSS3), and JavaScript for cross-platform development [12]. Cordova uses HTML5 and CSS3 for front-end development and JavaScript for back-end development. Since applications built using Cordova can run on both Android and iOS smartphones, this framework was selected for this study.

2.3 User Interface Design

Even the best functionality of an application is not accessible to end-users without a user interface. Therefore, a user-friendly UI is one of the most important factors of a useful application. As mentioned in section 2.2, HTML5 and CSS3 were used to design the UI in the Cordova framework.

As shown in Figure 3, the mobile application here includes seven pages: (1) a login page, which asks the user for a username and password; (2) a dashboard page, which contains two, three, or four livestream videos from the selected door, two buttons for opening and closing the selected door, and a third button for recording sound; the elected door can be changed by using a left menu, and the user can manage all doors in one application; (3) a page for adding new users and assigning them specific permission; (4) a page to manage and remove users that have been created; (5) two report pages containing graphs of door activities and user activities; and (6) a profile management page that gives the user the ability to personalize the account and change the password.

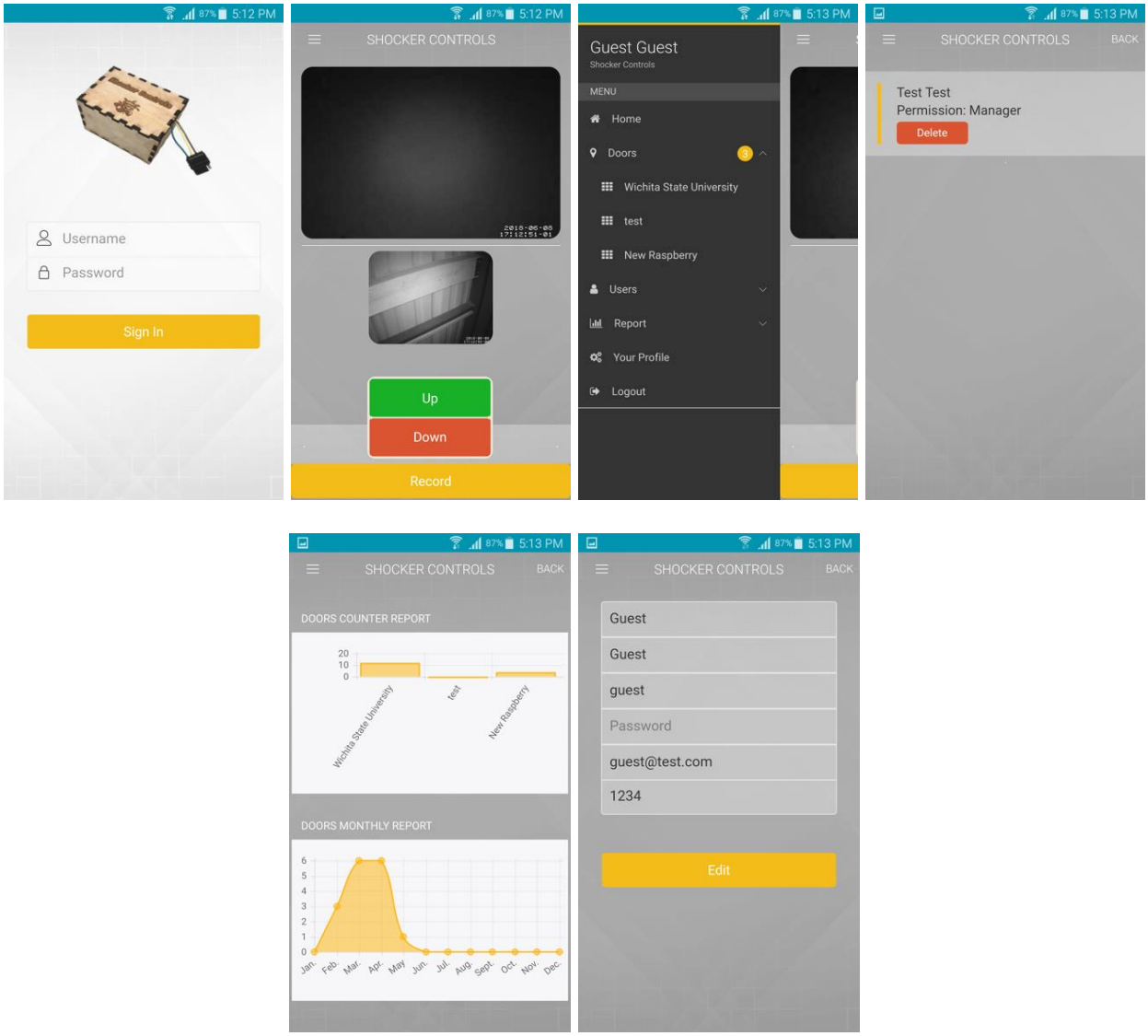

Figure 3. Mobile application user interface design

2.4 Back-End Development

An application does not have any functionality without back-end development, which is the soul of the application. As mentioned in section 2.2, JavaScript was used for the back-end development of this application. Also, several libraries were used to improve the application features. These libraries help in different aspects of development, such as showing a message to the user, opening a socket to specific port to the door side, and drawing graphs on the report page.

On the login page, the application receives the entered username and password. Then the application converts the plain text password to the Message Digest 5 (MD5) hashed password. Finally, the application posts both the username and the hashed password to a personal home page (PHP) file on the server side. Based on the response of the server side, the application either allows or disallows the user to enter the application.

On the dashboard page, based on information about the selected door received from the server side, the application will show two to four livestream videos. The application receives the livestream videos on ports 5000 and 5001 from the Raspberry Pi on the door side. Moreover, on the dashboard page, the user can open or close the selected door using two buttons. When the user clicks on one of the two buttons, the application creates a Transmission Control Protocol (TCP) socket directly connected to the related Raspberry Pi on the door side, and the application sends controlling commands using this TCP socket (more details in Chapter 3). Communication between the mobile application, central server, and Raspberry Pi is shown in detail in Figure 4.

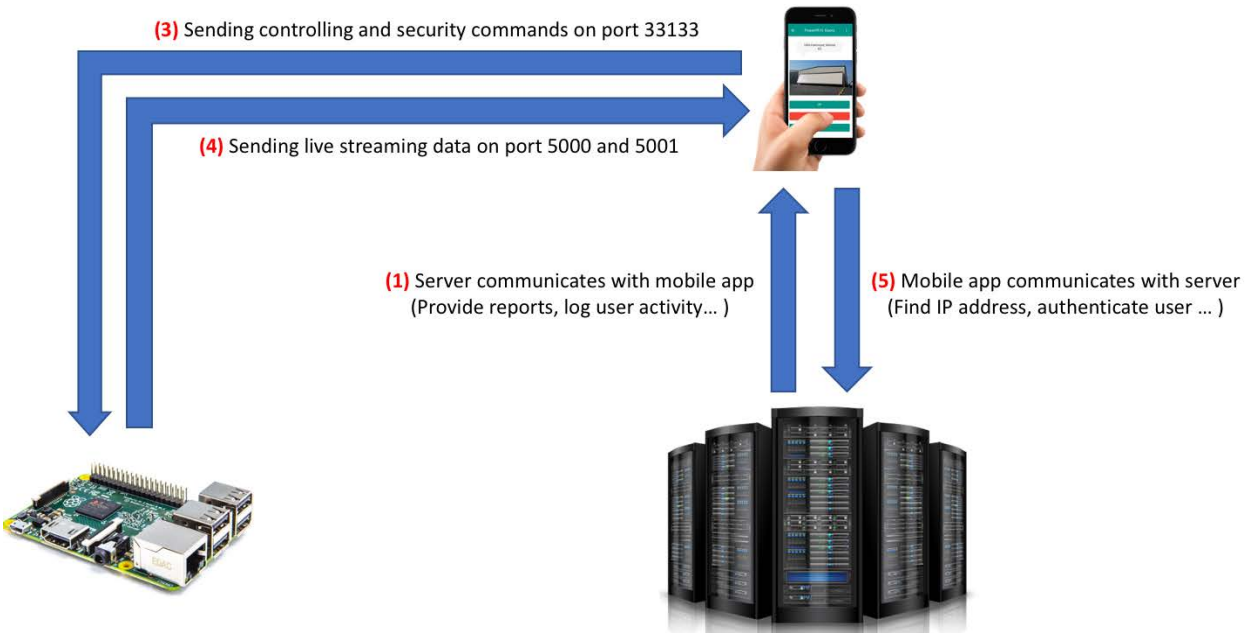

Figure 4. Communication between mobile application, central server, and Raspberry Pi

Moreover, in the background, the application updates its status every two seconds with the central database by calling a PHP file on the server side. In other words, the application will report every two seconds to the server if the user clicks on the open or close button. This feature will help the central server to synchronize the mobile application and the Raspberry Pi (more details in Chapter 3).

In addition, the mobile application will notify the user if the cameras have captured any new motion. The user can mute this notification or set the application to provide notification at a specific hour (for example, at midnight). If any motion is captured and the application is closed at that time, then the user will receive a notification at the next application startup.

Also, on the left menu, the user can select different doors to control. By selecting each door, the application sends a request to a PHP file on the server side and asks for information about the selected door. This information includes the door's Raspberry Pi Internet Protocol (IP) address, number of livestream cameras, door latitude, and door longitude.

In the mobile application, users can have different types of access and roles. By default, the manager role has top-level access. Manager users can use the left menu to add new users and their information, which is then sent to the server side. If the information is valid, then the new user will be added. Manager users can also manage all added users, with the ability to edit or remove each user as necessary.

The mobile application also includes two report pages with information received from the server side. This information is categorized based on different doors and different users. Each category is shown on two separate graphs. Graphs are drawn using the Chart.js library, which is a simple yet flexible JavaScript charting tool for designers and developers.

Finally, users can edit their personal information by using the profile page in the application. They can change their name, email, and phone number as well as update their password. The application will send updated information to a PHP file on the server side, and if this information is valid, then the PHP file will update the database.

CHAPTER 3

RASPBERRY PI SETUP AND CONFIGURATION

We need a computer on the door side, through which the mobile application can send the control command to that computer. On the other side, the computer should send signals to the door relay to open, close, or stop the door and broadcast the livestream videos to the application. Therefore, it was decided to use the Raspberry Pi as the door-side computer. The Raspberry Pi is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation to promote the teaching of basic computer science in schools and in developing countries [13].

Figure 5 shows how the Raspberry Pi fits inside a wooden box. The main steps involved in the Raspberry Pi configuration are as follows: installing Raspberry Pi operating system (OS), selecting programming language, writing required codes, and installing motion software to enable Raspberry Pi to stream video.

Figure 5. Raspberry Pi box and wired connection to door relay

3.1 Installing Raspberry Pi Operating System

For the first step, the Raspbian operating system was used for the Raspberry Pi. This open source OS allows users to download it from the Raspberry Pi website. To install Raspbian, the user can unzip the downloaded zip file and copy the ISO file inside the secure digital (SD) card of the Raspberry Pi. If there is enough space on the SD card, the Raspberry Pi will take care of the rest of the installation steps on the first reboot.

After installing Raspbian OS, two configurations must be done. The first configuration is connecting the OS to the wireless network, which needs some changes on two files: */etc/dhcpcd.conf* and */etc/wpa_supplicant/wpa_supplicant.conf*. The second configuration is to set the Raspberry Pi to send its valid IP address to the central server upon each reboot. The mobile application will use this IP address to send the control command directly to the door side.

3.2 Select Programming Language

The next step after installing the OS is to select an appropriate programming language to write necessary codes on the Raspberry Pi. Since Raspbian OS supports Python, an interpreted high-level programming language for general-purpose programming, and since Python provides useful libraries in network programming, this language was chosen as the door side programming language. Python, created by Guido van Rossum and first released in 1991, has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales [14].

3.3 Writing Required Codes

The Raspberry Pi application here always listens on port 33133 and is ready to receive the control command from a mobile application. The door-side application will send an open, close,

or stop signal to the door relay as soon as it receives a control command from the mobile application. This control application automatically runs by each reboot.

The Raspberry Pi updates the status of the door every two seconds and receives the status of the mobile application every two seconds. If the statuses of the Raspberry Pi and mobile application are not synchronized for four seconds, then the Raspberry Pi will stop the door immediately. For example, if the Raspberry Pi sends an open signal to the door relay but receives a stop status as status of the mobile application, then the Raspberry Pi will stop the door immediately, thereby ensuring that the mobile application and the Raspberry Pi are in the same status. If the direct communication between the mobile application and the Raspberry Pi breaks for any reason, then the Raspberry Pi will detect a problem by checking the central database. Communication between the Raspberry Pi, mobile application, central server, and door relay are presented in Figure 6.

Figure 6. Communication between Raspberry Pi, application, central server, and door relay

3.4 Installing Motion Software

The last step of the Raspberry Pi configuration is installing motion software, which enables the Raspberry Pi to broadcast livestream video of the connected cameras on specific ports. Moreover, motion software monitors the video signal from one or more cameras and can detect if a significant part of the picture has changed. Or in other word, it can detect motion [15]. Installing motion software is possible on the Raspbian OS by using the "apt-get" command. The desired configurations must be set for each camera. These configurations include the streaming port, format of streaming, and configuration of the motion detection.

3.5 Programming Motion Detection

By using the motion-detection feature, the motion software will detect any movement on the front of any camera and will start recording video until the motion stops. By detecting motion, the Raspberry Pi calls a PHP file on the central server. This PHP file adds information of the detected motion on the database and marks it as a new detected motion whereby the mobile application notifies the user. The Raspberry Pi keeps recording the video until the motion ends. At the end of the motion, the Raspberry Pi will call another PHP file on the central server, and this time it will update the database with the end time of the motion. The mobile application will check the central database for any new detected motion every four seconds, and it will notify the user if there is any new detected motion. The Raspberry Pi keeps recorded video for 30 days.

CHAPTER 4

CENTERAL SERVER CONFIGURATION

A central server was used to synchronize the mobile application and the Raspberry Pi. Figure 7 shows communication between the mobile application, Raspberry Pi, door relay, and central server. Considering that the My Structured Query Language (MySQL) database system and PHP were already installed on the server, the main steps in the server configuration were creating a new database, creating the required tables for the database, and writing PHP codes to update the database.

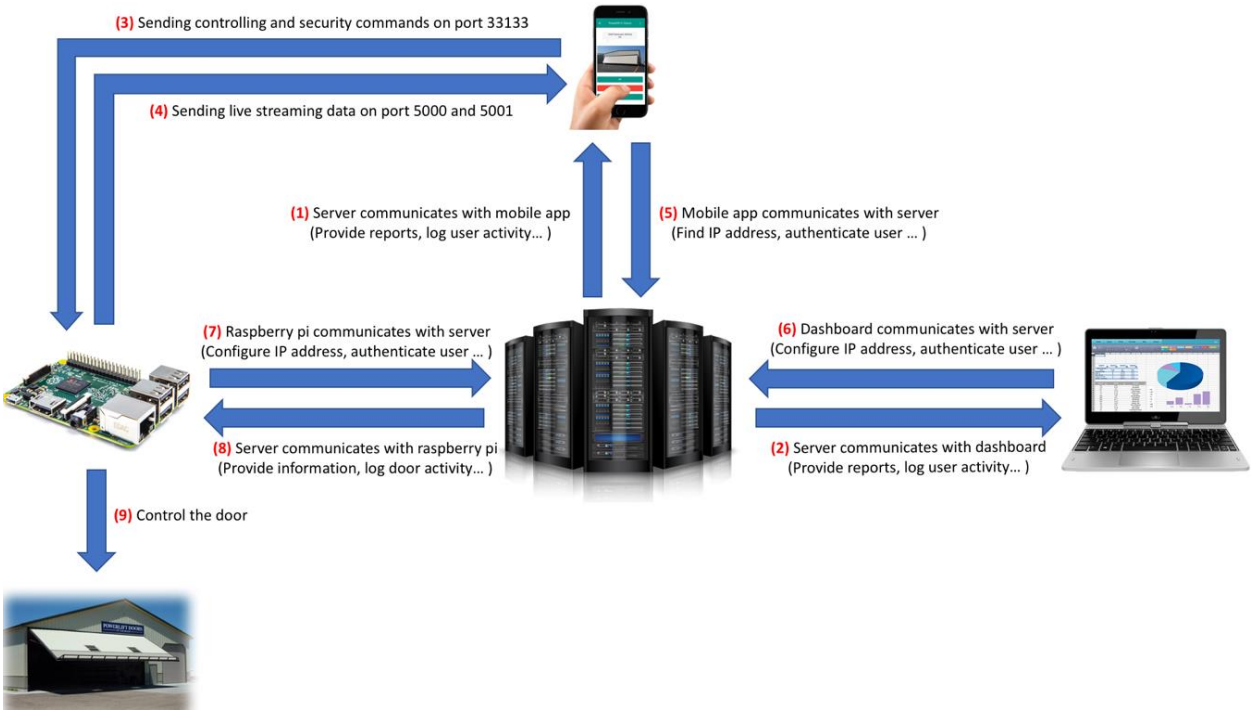

Figure 7. Communication between mobile application, Raspberry Pi, and central server

4.1 Creating New Database

The first step here was to create a new database on MySQL, which was already installed on the server. The software phpMyAdmin was used to create the new database. This free software tool written in the PHP, is intended to handle the administration of MySQL over the Web. The

phpMyAdmin software supports a wide range of operations on the MySQL and MariaDB database systems. Frequently used operations (managing databases, tables, columns, relations, indexes, users, permissions, etc.) can be performed via the user interface, while the user can still directly execute any Structured Query Language (SQL) statement [16].

4.2 Creating Required Tables

Tables are necessary to store information about different doors and users. The database here contains six tables: doors, history, IP address, motion, user, and role. Figure 8 shows the entity relationship diagram (ERD) created for this database, including the table names and their relations.

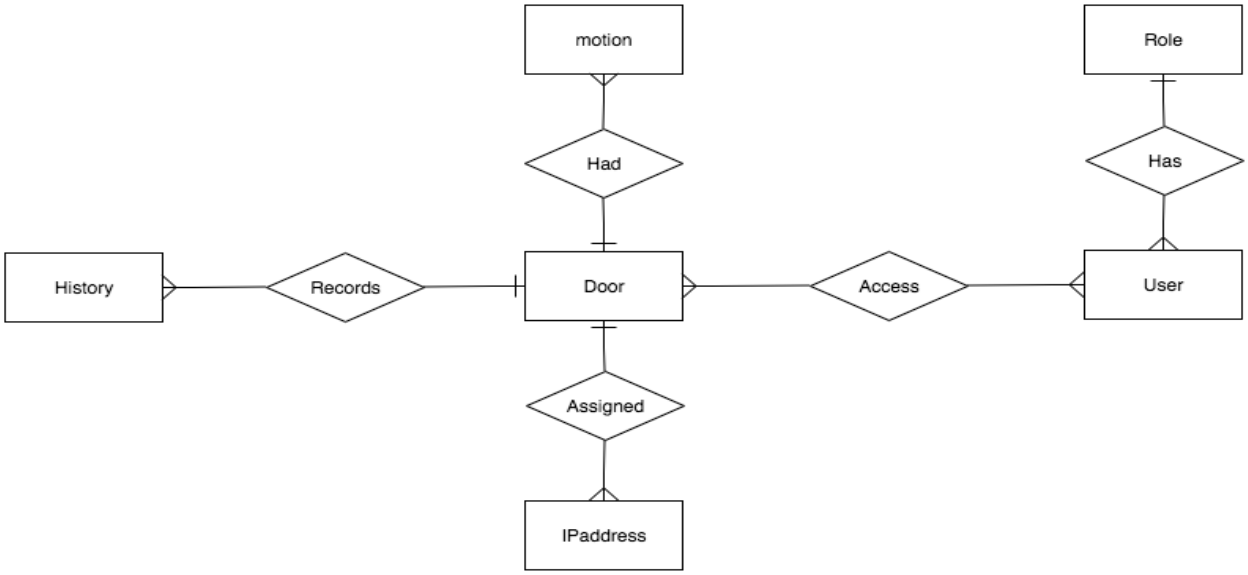

Figure 8. Entity relationship diagram of central database

Each table contains different columns and one unique identification (ID). The unique ID is an integer and increments automatically. The door table retains the information of each door, such as the door's nickname, location of the door, connected Raspberry Pi IP address, status of the door, and status of the mobile application. The history table keeps track of operations of each door. This table can be used to provide useful reports for managers. The IP address table keeps track of those IP addresses that have access to a particular door. This table is used for security purposes (more

details in Chapter 5). The motion table retains information about the motions captured by the door cameras, including start time and end time of each motion. The user table stores the information of each user in the system. Each user can be related to one or more doors, and each user can have a specific type of access, which is indicated in the role table.

4.3 Creating Server-Side Personal Home Page Codes

After configuring the database, personal home page files are created, which are the connection between the mobile application (or Raspberry Pi) and the central server. Therefore, there are two categories of PHP files: one for connecting the mobile application to the central server and one to connect the Raspberry Pi to the central server.

4.3.1 Mobile Application PHP Files

Ten different PHP files have been implemented in the mobile application:

- **loginCheck.php**: This file will receive a set of username and hashed password, check to determine if they are valid or not, and if they are valid, then return information about all doors to which the user has access.
- **checkAndReport.php**: This file will set the status of the application on the door table. It will report if a user clicked on the open or close buttons in the application.
- **checkMotion.php**: This file will check the motion table and determine if there is any new motion, returning information of the new motion to the mobile application.
- **addUser.php** and **editUser.php**: These files will receive the necessary information to create or edit a user on the user table.
- **deleteUser.php**: This file will receive a username and remove that particular user from the database.

- **getUsers.php**: This file will receive a manager username and return a list of all users created by that particular manager.
- **getDoorsReport.php** and **getDoorsReportMonthly.php**: These files will receive a manager username and return necessary information about door usage to that particular manager .
- **getUserReport.php**: This file will receive a manager username and return necessary information about users operations to that particular manager.

4.3.2 Raspberry Pi PHP Files

Four different PHP files have been implemented in the Raspberry Pi application:

- **loginCheck.php**: This file will receive a set of username and hashed password, and check to determine if that particular user has access to operate the door. This file can be used for security purposes (more details in Chapter 5).
- **reportRaspberryIP.php**: This file will receive an IP address and update the IP address of the door.
- **checkAndReport.php**: This file will update the status of the door and see if it is operating or not. This file will also return the status of the mobile application to the Raspberry Pi.
- **reportMotion.php**: This file will add any new motion, including start time and end time of the captured motion, to the database.

CHAPTER 5

UNIT SECURITY CONCERNS AND SOLUTIONS

Using the Internet as the communication channel between the mobile application, Raspberry Pi, and central server causes some security concerns. This chapter covers these types of concerns and provides some effective solutions.

5.1 Login to Mobile Application

For user authentication, both the username and password should be sent to the central server. However, sending the password in plain text can cause "password sniffing," an attack on the Internet that is used to steal usernames and passwords from the network. This issue is mostly of historical interest, because today most protocols use strongly encrypted passwords. However, in the 1990s, this was the worst security problem on the Internet, with news of major password sniffing attacks reported almost weekly [17].

Solving this problem involves saving passwords in hash format in the central database. Here the mobile application converts the plain password to a hash format and sends the hashed password to the central server; the central server then authenticates the user by means of the username and hashed password. The MD5 algorithm has been used to convert plain passwords to a hash format. MD5 is simply the name for a type of cryptographic hashing function that takes an arbitrary block of data and returns a fixed-size "hash" value [18].

5.2 Sending Control Commands to Raspberry Pi

The mobile application creates a TCP socket at a specific port on the Raspberry Pi to send control commands to open or close the door. Anyone knowing the IP address and that specific port of the Raspberry Pi can send any control commands to the Raspberry Pi. To solve this problem,

the username and hashed password are sent along with the control command to the Raspberry Pi. Therefore, whenever the Raspberry Pi receives a control command, first it will authenticate the sender by sending username and hashed password to the server, and then if the sender is determined to be valid, the Raspberry Pi will send the control command to the door relay.

5.3 Broadcast Livestream Video

Broadcast livestream video from cameras on specific ports will begin when the Raspberry Pi starts up. Anyone knowing the Raspberry Pi IP address and streaming ports can easily access the livestream video of the doors. To solve this problem, it was decided to restrict broadcasting to the connected mobile application IP addresses. The steps in this process are as follows:

1. The user tries to log in through the mobile application. If the input username and password are valid, then the central server saves the mobile application IP address into an IP address table in the central database. This specific IP address is related to the specific door that the user requests for obtaining access.
2. On the door side, a firewall restricts broadcasting to specific IP addresses. The Raspberry Pi updates these IP addresses every 5 seconds. The Raspberry Pi calls a PHP file on the central server and checks for any new connected IP address on the database. If there is a new IP address, then the Raspberry Pi will add it to the white list of the firewall and allow the motion software to broadcast livestream videos to that IP address.

5.4 Different Types of User Access

As mentioned in Chapter 2, the mobile application developed here provides different features to customers. For example, it is possible to add new users and give them access to specific doors, or it is possible to see a report from different door operations or user operations. However, it is not acceptable to provide all type of access to all users. To solve this problem, a role table was

added to the central database, whereby each user has a specific type of access. The basic version of the application involves three types of access: employee, manager, and system administrator. The employee type of user has limited access—simply controlling and monitoring specific doors. The manager type of user has access to control and monitor all of his/her doors. Moreover, the manager user can add new employee users to the system and assign them to specific doors, as well as view reports provided in the application. The system administrator type of user has access to all doors in the system, including being able to control and monitor all doors, add a new manager and employee users, and access full reports of the system.

5.5 Future Works

In this section we are providing two future works which can improve this product. The first improvement is adding Long-Term Evolution (LTE) connectivity to the unit. Currently broadband internet connection is in use for Raspberry Pi internet connection. However, such these commercial doors can be placed in farms and areas which have no broadband internet access. But still there is mobile network in such these areas and the unit can connect to internet by using LTE technology. AT&T and Verizon introduced a new technology called LTE Category M in 2018. LTE Cat M providing cheap LTE internet connection for commercial usage. For adding LTE Cat M to Raspberry Pi, we need to add a module and config the Raspberry Pi to use LTE when it cannot find any broadband internet access.

The second improvement is using more secure hash functions to convert customers passwords to hash strings. Currently Message Digest 5 (MD5) algorithm is in use to generate hash passwords. However, MD5 function is breakable with powerful computers. Therefore, By replacing MD5 function with newer functions such as SHA2 (Secure Hash Algorithm 2) or SHA3 (Secure Hash Algorithm 3) we can improve security of unit.

CHAPTER 6

CONCLUSION

Designing a complete unit to control a commercial door by using a mobile application could be very useful in industry, reducing the cost of commercial door owners and improving security and reliability. This unit includes a mobile application, a Raspberry Pi on the door side, and a central server.

On the mobile application side, the Cordova development framework was used to develop a user-friendly mobile application that should work on both Android and iOS devices. Moreover, this mobile application provides a variety of features to end-users, such as controlling an unlimited number of doors in one application, and adding, editing, and removing users.

On the door side, the Raspberry Pi is used as a minicomputer, responsible for receiving control commands from the application and sending control signals to the door relay. Moreover, the Raspberry Pi broadcasts livestream video from multiple cameras connected to it.

Finally, the unit has a central server to synchronize the mobile application and the Raspberry Pi. Moreover, this central server is used for security purposes, such as checking user authentication, keeping track of valid connected IP addresses to each Raspberry Pi, and keeping track of a user's type of access.

In the future, adding Long-Term Evolution (LTE) to the Raspberry Pi could be a big improvement for the unit. Some areas do not have Wi-Fi coverage. Therefore, the best solution could be replacing Wi-Fi with LTE. Moreover, security will always be one of the most important aspects of this unit. Therefore, improving unit security (for example, replacing the MD5 hash algorithm with a newer hash algorithm) could make a very large impact on the unit.

REFERENCES

REFERENCES

[1] Powerlift Hydraulic Doors, “Powerlift Accessories,” URL: https://powerliftdoors.com/products/powerlift-accessories/ [November 20, 2018].

[2] Yue Huang, “Automatic garage door automatic monitoring and controlling system based on the internet of things concept and near field communication (nfc) technology,” United States Patent Application Publication, June 19, 2014, p. 1

[3] Gentex Corporation, “Internet-connected garage door control system,” United States Patent Application Publication, May 21, 2015, p. 1

[4] Brian Frederic Butler, “Barrier movement operator battery backup and power equipment battery charging center,” United States Patent Application Publication, December 22, 2009

[5] James J. Fitzgibbon, “Movable barrier operators status condition transception apparatus and method,” United States Patent Application Publication, May 29, 2007

[6] J BulgerR Rempel, “Self-levelling motion detecting device and alarm system incorporating the same,” United States Patent Application Publication, Aguste 20, 1974

[7] Theodore E. Llewellyn, “Garage door openers,” United States Patent Application Publication, September 04, 1990

[8] Dennis W. Waggamon, “Remote controlled garage door opening system,” United States Patent Application Publication, April 11, 2000

[9] Gallen Tsui, “Garage door monitoring system,” United States Patent Application Publication, April 24, 2003

[10] Ronnie Kirkland, “Garage door remote monitoring system,” United States Patent Application Publication, December 05, 2002

[11] Tutorials Point, “SDLC - Software Prototype Model,” URL: https://www.tutorialspoint.com/sdlc/sdlc_software_prototyping.htm [March 27, 2017].

[12] The Apache Software Foundation, “Apache Cordova Overview,” URL: https://cordova.apache.org/docs/en/latest/guide/overview/ [November 30, 2018]

[13] Wikipedia, “Raspberry Pi,” URL: https://en.wikipedia.org/wiki/Raspberry_Pi [December 10, 2018]

[14] Dave Kuhlman, *A Python Book: Beginning Python, Advanced Python, and Python Exercises*, Platypus Global Media, 2012.

[15] GitHub, Inc., “Motion-Project/Motion,” URL: https://github.com/Motion-Project/motion [November 25, 2018]

[16] phpMyAdmin, “Bringing MySQL to the web,” URL: https://www.phpmyadmin.net/ [December 20, 2018]

[17] SSH Communications Security, Inc., “Password Sniffing,”, URL: https://www.ssh.com/attack/password-sniffing [November 1, 2018]

[18] Tim Brookes, “What All This MD5 Hash Stuff Actually Means,” Make Use Of, February 10, 2011, URL: https://www.makeuseof.com/tag/md5-hash-stuff-means-technology-explained/ [January 21, 2019]

APPENDICES

APPENDIX A

MOBILE APPLICATION PSEUDOCODE

```
Login:
    HashPassword=HashFunction(Password);
    ServerAnswer=SendUsernamePasswordToServer(Username, Password);
    if(ServerAnswer==true):
        Let user access the application
    else:
        Ask user to try another username or password

Communicate with Raspberry Pi:

    if (user select UP Button):
        SendCodeToRaspberryPi(OpenDoor);
    else if(user select DOWN Button):
        SendCodeToRaspberryPi(CloseDoor);
    else:
        SendCodeToRaspberryPi(STOP);

Communicate with central server:

    while(application is sending commands to Raspberry Pi):
        if (user selected UP Button):
            SendStatusToCentralServerEvery2Second(AppOpeningDoor);
        else if(user select DOWN Button):
            SendStatusToCentralServerEvery2Second (AppClosingDoor);
        else:
            SendStatusToCentralServerEvery2Second (AppStopOperationDoor);
```

APPENDIX B

RASPBERRY PI PSEUDOCODE

Door Operation:

```
ReceivedDate=ListenOnPort33133();
ServerAnswer=SendUsernamePasswordToServer(ReceivedDate.Username,
ReceivedDate.Password);
if(ServerAnswer==true):
    SendControlCommandToDoorRely(ReceivedDate.ControlCommand);
else:
    SendErrorMessageToMobileApplication();
```

Communicate with central server:

```
While(true):
    if(CheckRaspberryPiStatusEvery2Second(CurrentStatus)==false):
        StopDoorUpdateCentralServer();
```

Motion detection:

```
if(MotionDetected()=true):
    SendMotionStartTimeToCentralServer();
    WaitUntilMotionStops();
    SendMotionEndTimeToCentralServer();
```

APPENDIX C

CENTERAL SERVER PSEUDOCODE

Login:

```
Username, HashPassword=ReceiveUsernamePassword();
if (ValidInDatabase(Username, HashPassword)):
    return true;
else:
    return false;
```

Communicate with mobile application:

```
ReceivedStatus=ReceiveMobileApplicationStatus();
UpdateDatabase(ReceivedStatus);
```

Communicate with raspberry pi:

```
ReceivedRaspberryPiStatus=ReceiveRaspberryPiStatus();
UpdateDatabase(ReceivedStatus);
MobileApplicationStatus=GetMobileApplicationStatus();
if(ReceivedRaspberryPiStatus==MobileApplicationStatus):
    return false;
else:
    return false;
```