



SOMS CA1 (Part One) Step-by-step Tutorial

Table of Contents

Section 1 Overview of Smart Office Management System (SOMS).....	2
A. What is Smart Office Management System (SOMS)?.....	2
B. Summary of the steps that will be described	4
C. How does the final RPI set-up looks like?	5
D. How does the schematics look like?	11
E. How does the SOMS web application look like?	13
Section 2 Hardware requirements	20
Hardware checklist.....	20
Section 3 Setup Hardware (Raspberry Pi)	21
Section 4 Setup Hardware (Arduino)	26
Section 5 What is MEN stack?	29
Section 6 Setup Node.js development environment	31
Section 7 Setup required libraries and dependencies	33
Section 8 Creating the SOMS web application (Backend).....	36
Section 9 Creating the SOMS web application (Frontend)	50
Section 10 Running SOMS	103

Section 1

Overview of Smart Office Management System (SOMS)

A. What is Smart Office Management System (SOMS)?

Smart Office Management System (SOMS) is a web application that is used to manage certain processes required by people in an office environment. The web application interfaces with the Raspberry Pi 3 and various modules connected to it to carry out its functionalities.

Imagine that you are the director of an IT company called Encode IT Solutions Pte Ltd. You want to have a robust system to manage your employee's daily attendance, as well as to monitor and control the lights and camera in the office. The following are four issues that you are having now and how having the SOMS can help to resolve these issues.

Issue #1 – Lack of efficiency in monitoring employee's attendance

The current situation is that when employees arrive in the office front desk, they will have to sign their attendance on a sheet of paper to clock in the morning and to clock out in the evening. The employees have to search for their own names in the attendance sheet to sign. During peak hours in the morning and evening, a large number of employees will head to the front desk to queue in line to sign their attendance at the same time. This results in inefficiency and time loss as the time spent waiting in line could be time spent productively on their work.

With SOMS, the human resource (HR) can simply create a new user for each employee and enroll their NFC-enabled employee card in the system. Employees can simply wave their employee card in front of the Radio-frequency identification (RFID) reader to clock in and clock out in mere seconds. Upon waving their card in front of the RFID reader, the red led light and buzzer will light up and beep simultaneously for 5 times to create visual and auditory impact on the user.

Issue #2 – No single point of access to the office light switches

In an office, there are multiple lights that have to be controlled via multiple light switches at various locations. Employees may not be able to find the light switches and even if they found the switch, they may feel that it is inconvenient to walk from their desk to the light switch to control the lights.

SOMS addresses the issue of lack of single point of access to the office light switches. With SOMS, users can simply access the web application and navigate to the Lights page to turn on and off the lights in the comfort of his own desk. Other than turning on and off the lights, users can turn on different light colours they desire and make the led light blink.

Issue #3 – Lack of an integrated and controllable surveillance system

The office may have installed multiple Closed-Circuit Television (CCTV) cameras inside and outside of the office. However, these CCTVs are often bought from external vendors and they are only compatible with its own bundled software. Users who want to access the CCTV may have to do so through another application. Furthermore, these CCTVs can only capture footage at a certain angle, depending on how it is mounted and the type of CCTV.

SOMS resolves this problem by using the Raspberry PiCam. Users can access and view the PiCam's live footage through the web application. One of the potential pitfalls of the PiCam is that once it is mounted on the wall, it can only capture footage at a limited range of angle. To resolve this, the PiCam can be mounted on 2 servos.

The 1st servo is mounted on the wall which provides rotation along the X-axis, left and right. The 2nd servo is mounted on the first servo and attached to the PiCam. Users can push the joystick left and right to turn the camera in that direction. When the joystick is pushed up and down, it can rotate the PiCam's orientation. This greatly increases the range of angles the PiCam can capture.

Issue #4 – Lack of contact point for visitors to the office

There can be visitors who wants to visit the office, such as delivery persons. They cannot enter the office since the front door only allows entry for authorized employees. These visitors have difficulty trying to get the attention of the people inside of the office to attend to them.

With SOMS, visitors can simply push the big white button that is mounted on the wall outside of the office. Upon pushing the button, the buzzer makes an iconic beep sound to alert the front desk person that there is a visitor outside to attend to.

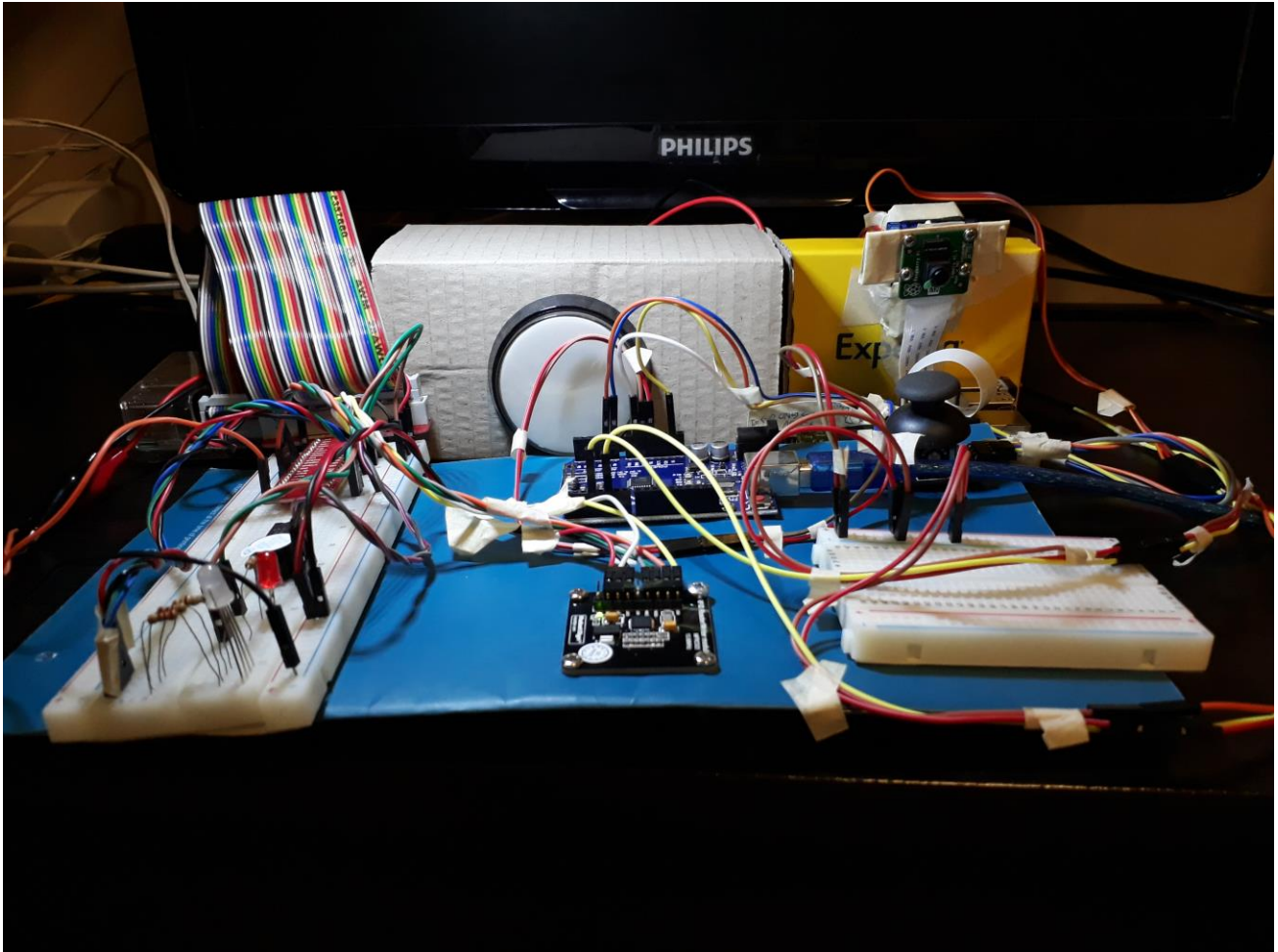
B. Summary of the steps that will be described

Provide a bullet list of the steps that will be covered in the other parts of this tutorial

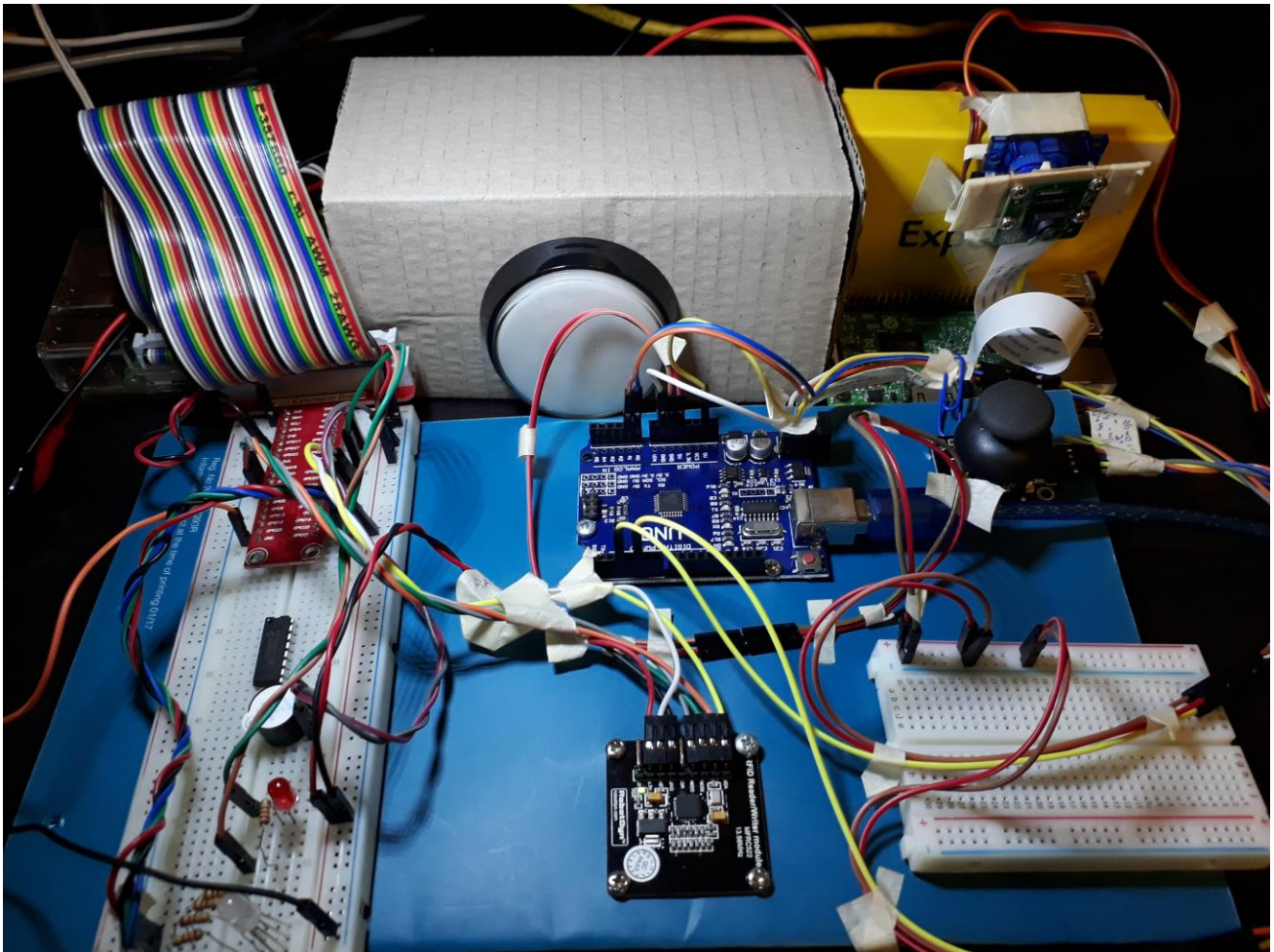
	Section	Description
1)	Overview	Provides an overview of SOMS
Sections 2 to 10 provides the step-by-step instructions to set up the entire application		
2)	Hardware requirements	Provides overview of hardware required
3)	Setup Hardware (Raspberry Pi)	Provides instructions on how to set up the hardware used for Raspberry Pi according to Fritzing diagram
4)	Setup Hardware (Arduino)	Provides instructions on how to set up the hardware used for Arduino according to Fritzing diagram
5)	What is MEN stack?	Provides a brief introduction to the MEN stack which SOMS runs on
6)	Setup Node.js development environment	Provides instructions on how to set up the Node.js development environment on the Raspberry Pi
7)	Setup required libraries and dependencies	Provides instructions on how to set up the required libraries and dependencies for developing SOMS
8)	Creating the SOMS web application (Backend)	Provides instructions on how to set up the backend for SOMS
9)	Creating the SOMS web application (Frontend)	Provides instructions on how to set up the frontend for SOMS
10)	Running SOMS	Provides instructions on how to run SOMS

C. How does the final RPI set-up looks like?

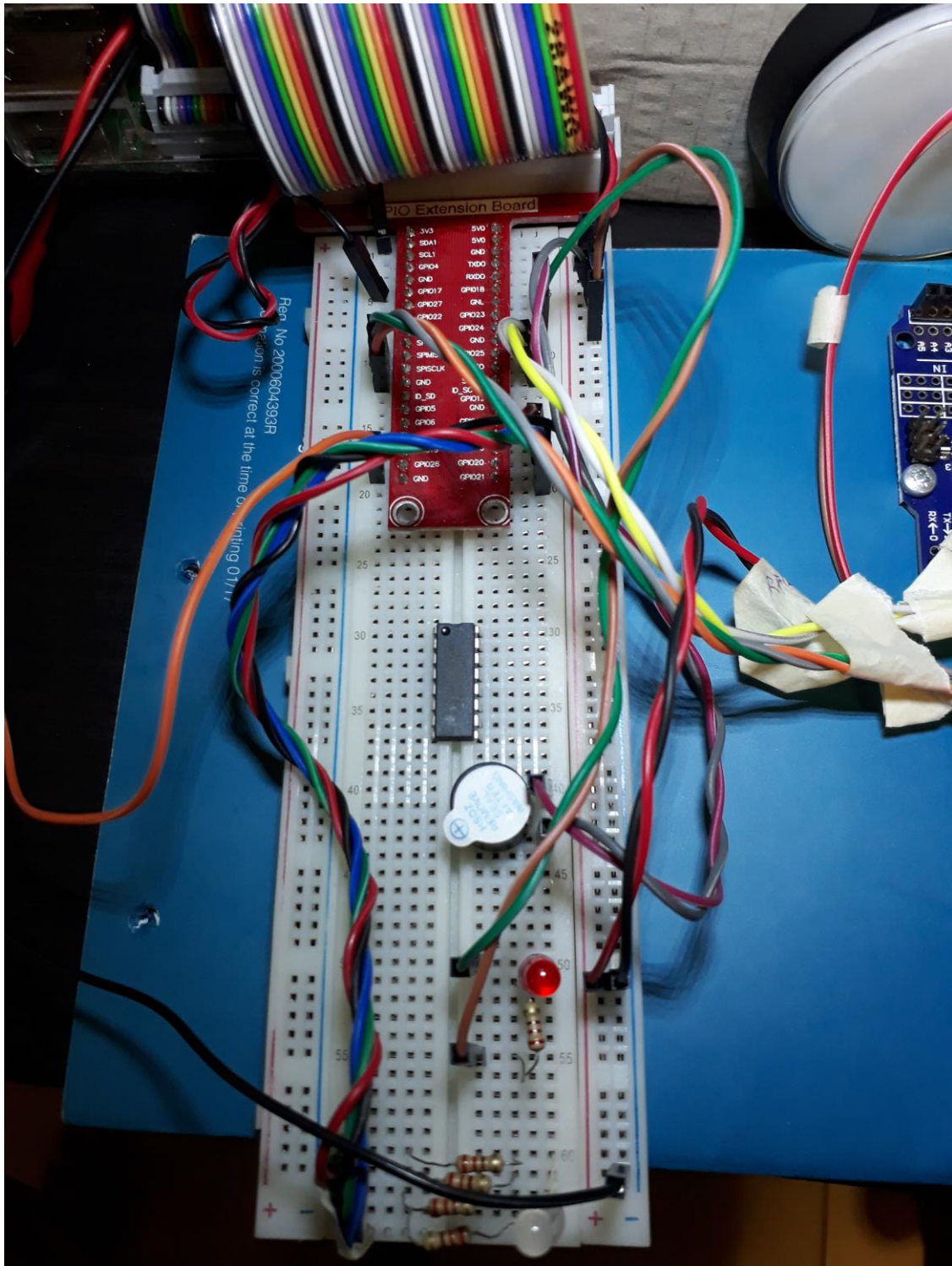
The image below shows the final RPI set-up in normal view.



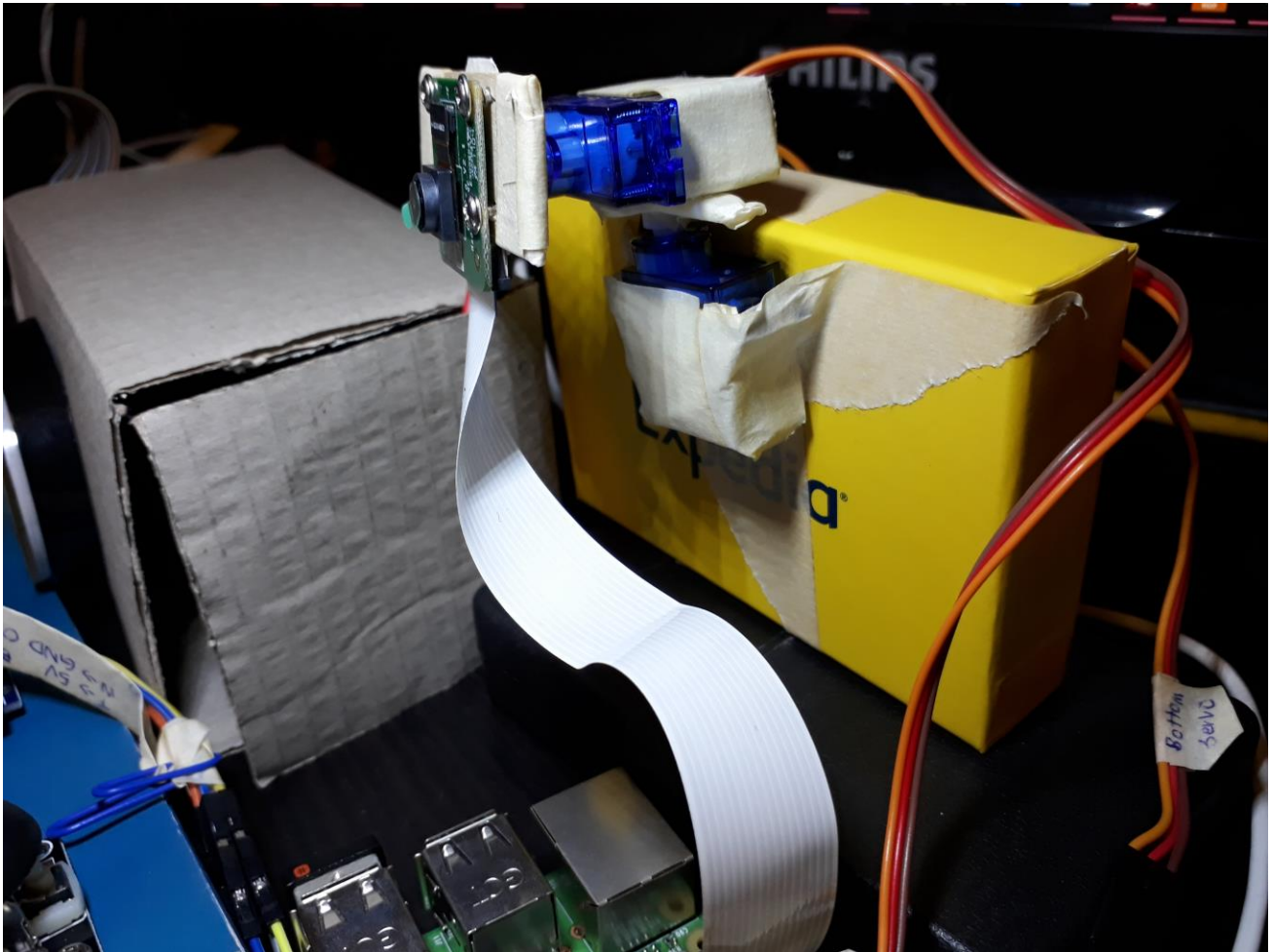
The image below shows the final RPI set-up in top-down view.



The image below in detail shows the breadboard connections for the buzzer, RFID reader, red LED and RGB LED.

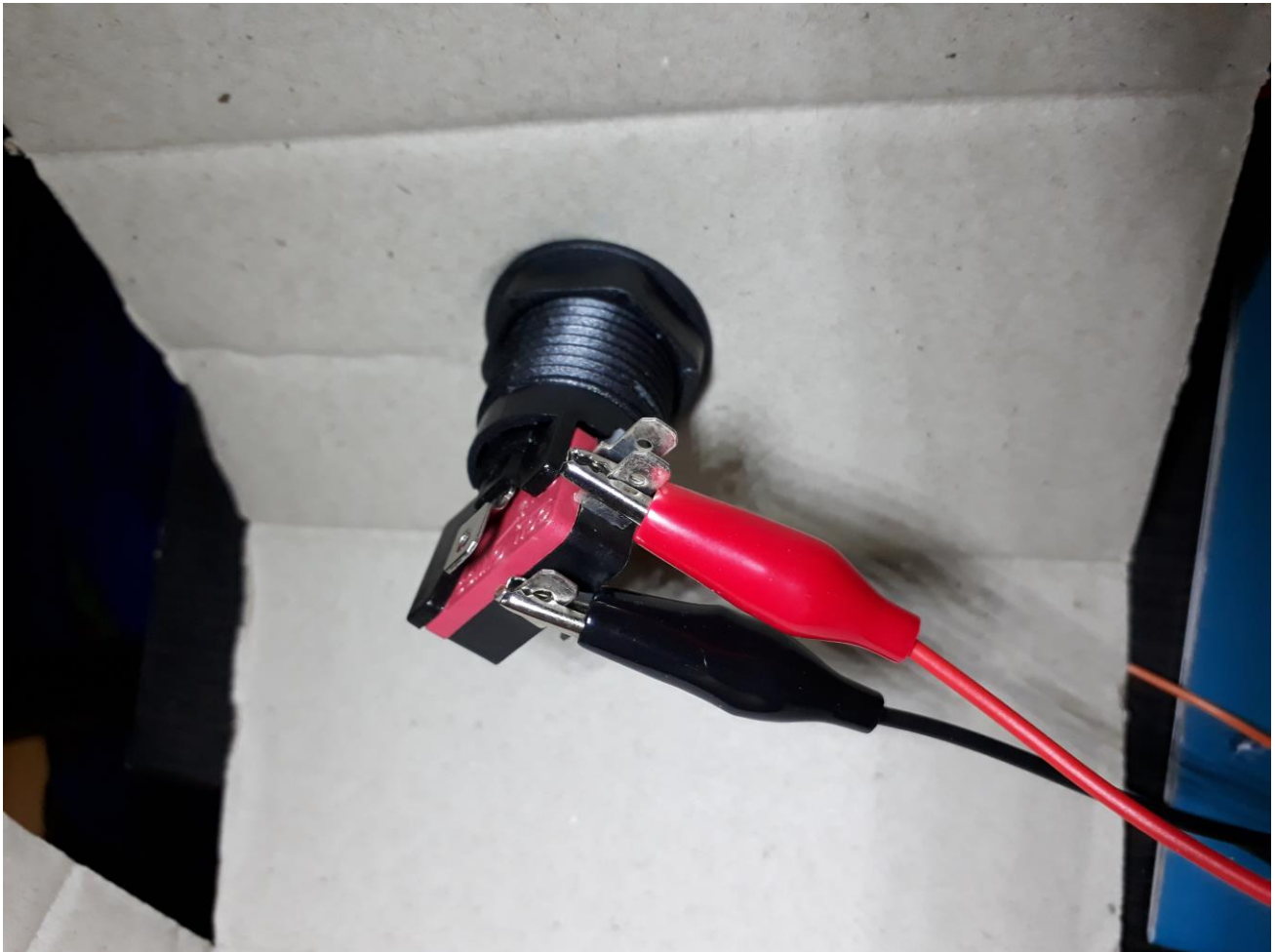


The image below shows in detail how the two servos and the PiCam which is connected to another Raspberry Pi are mounted. The yellow box represents the wall.



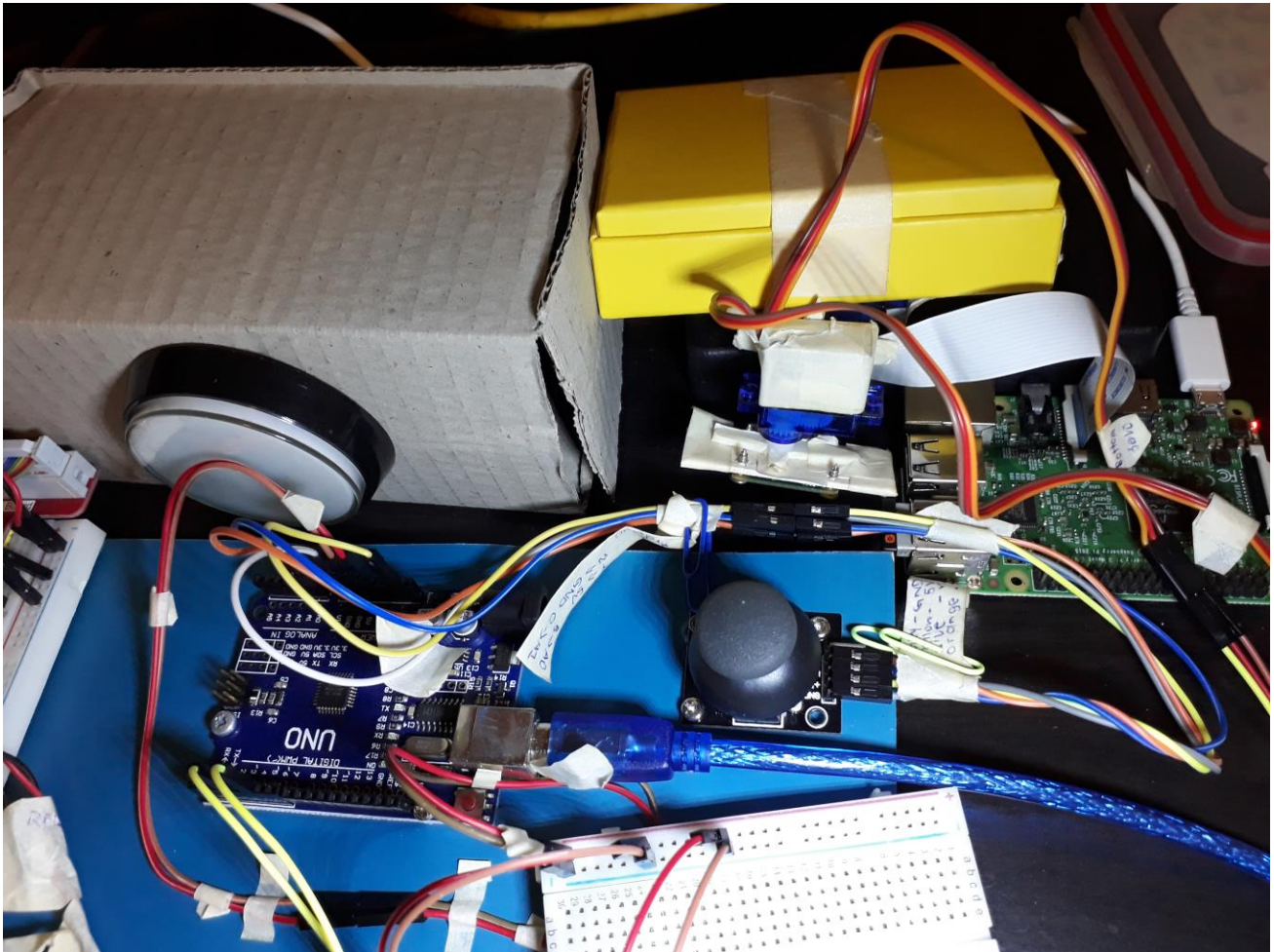
The image below shows in detail how the big white button is connected to the wall, which is represented by the cardboard box.

The black and red alligator clips are connected to black and red jumper wires respectively. Then, the black jumper wire is connected to GND and red jumper wire is connected to GPIO26.



The image below shows in detail how the Arduino UNO, joystick, breadboard and the two servos are connected.

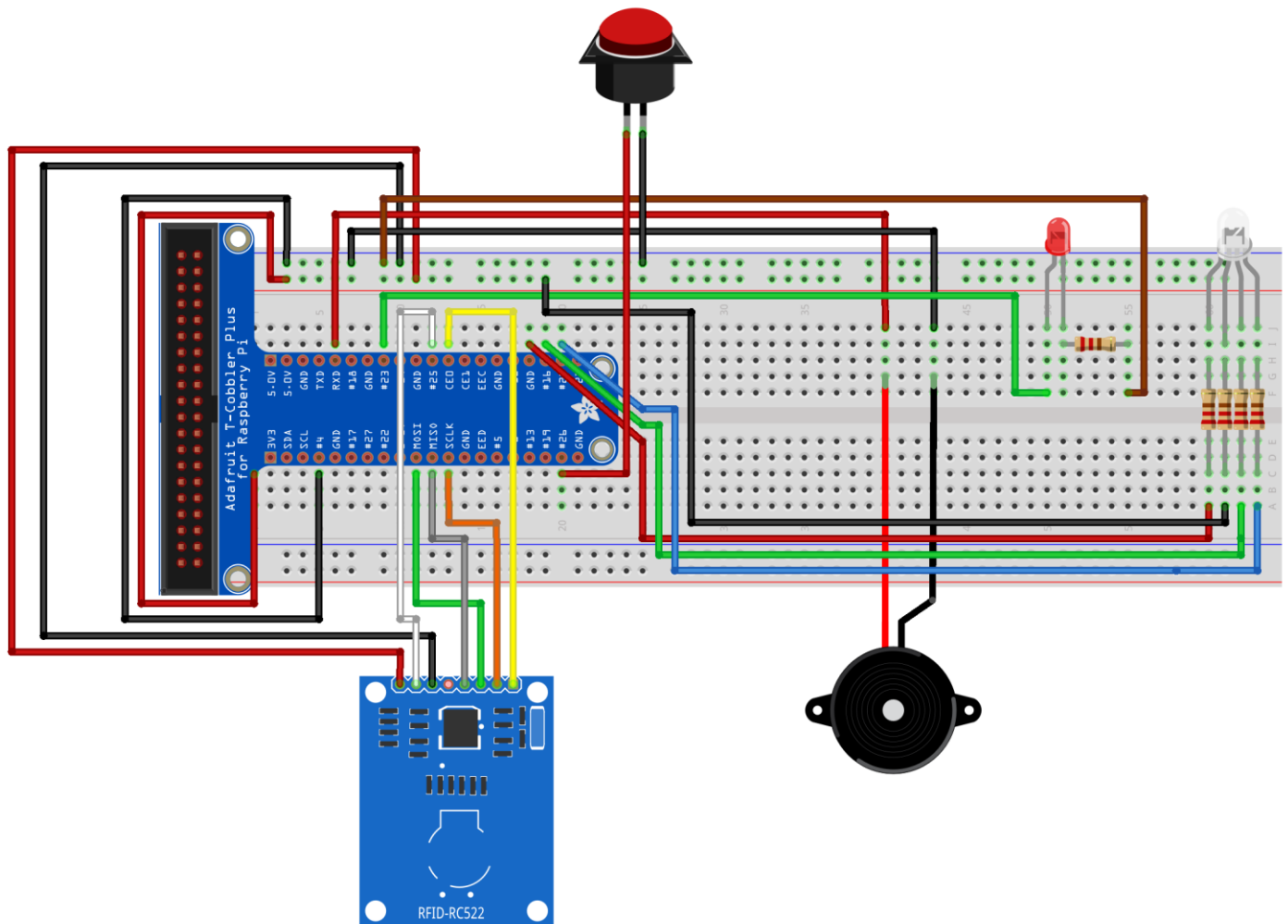
In order to not overload the Raspberry Pi, the Arduino's power source is obtained by connecting the USB A-B cable from the Arduino USB port to the laptop USB port. The Arduino provides power for the joystick and the two servos.



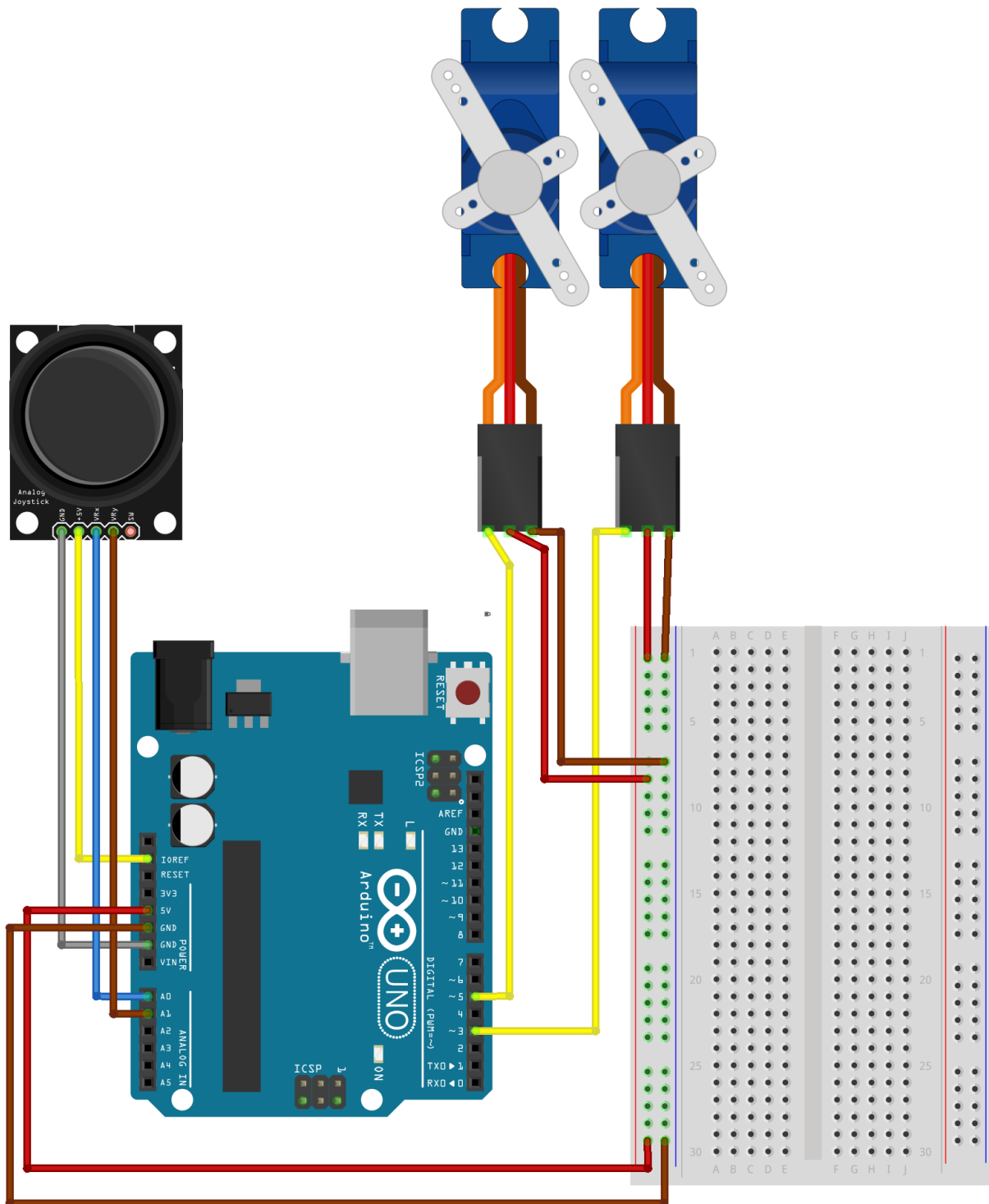
That's it! 😊

D. How does the schematics look like?

The image below illustrates the schematics for the Raspberry Pi and its breadboard connections.



The image below illustrates the schematics for the Arduino UNO and its breadboard connections.



fritzing

E. How does the SOMS web application look like?

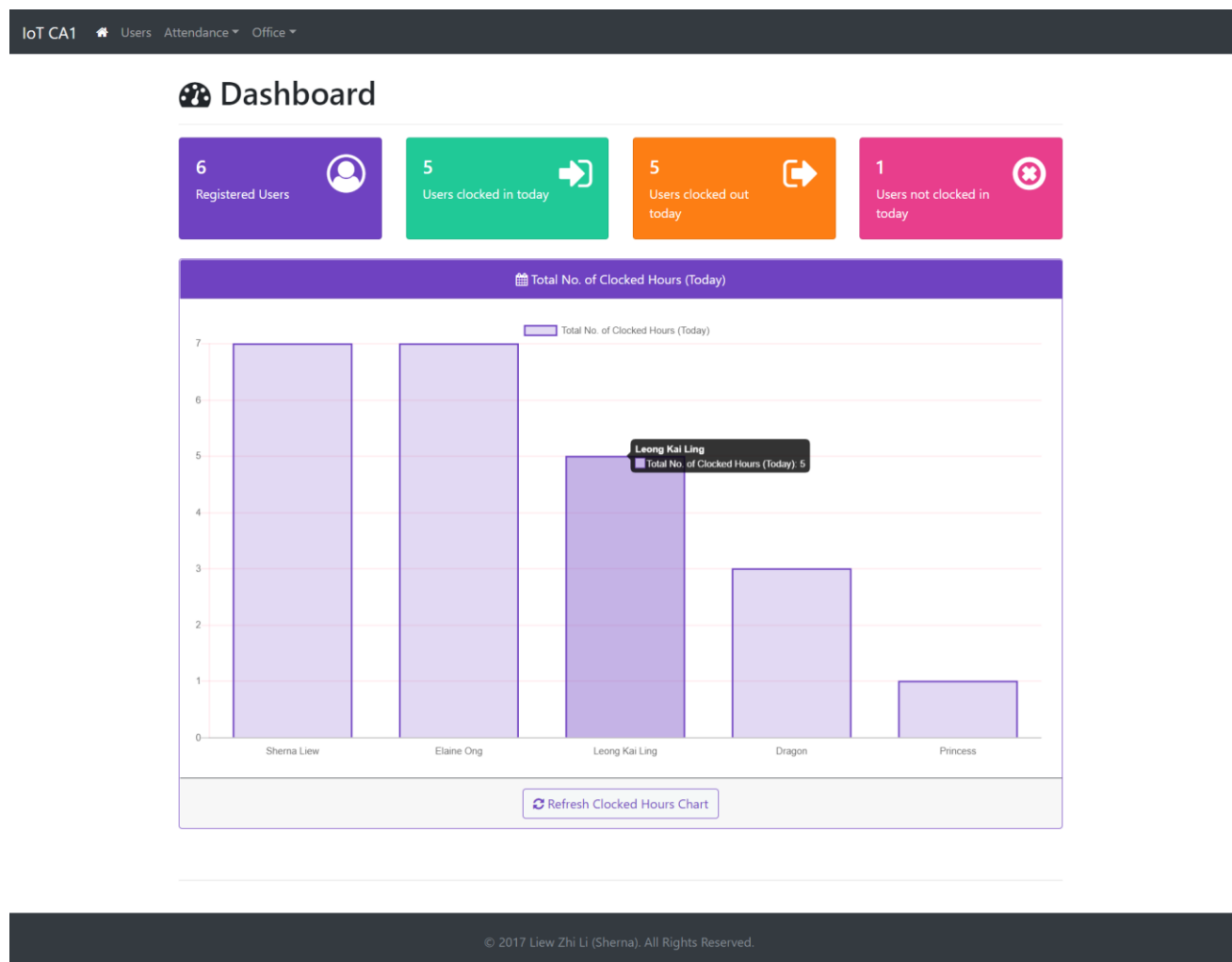
The image below shows the SOMS' dashboard.

The dashboard allow users to have an overview of the number of:

- Registered users
- Clocked in users
- Clocked out users
- Users not clocked in today

The bar graph shows the historical data of the total number of hours clocked by each user or employee. For example, we can see that the employee "Leong Kai Ling" has clocked in a total of 5 hours for today.

There is a "Refresh Clocked Hours Chart" at the bottom to refresh the bar chart.



The next image below shows the Users management page.

All Create, Read, Update, Delete (CRUD) functionalities for User can be performed in this page.

In the Users table, under the “Actions” column, the blue, yellow and red buttons are “Enroll NFC Mifare Card”, “Edit User” and “Delete User” respectively.

IoT CA1 [Users](#) [Attendance](#) [Office](#)

Users

Search

Any Keyword

Show 10 entries

Full Name	Username	Email	NRIC	NFC Mifare Card UID	Actions
Dragon	dragon	dragon@email.com	S0000001D	04c8e412a13580	
Elaine Ong	elaine	elaine@email.com	S0000001C	0439a612a13581	
Leong Kai Ling	kailing	kailing@email.com	S0000001B	04c424c2c44880	
Mimi	mimi	mimi@email.com	S0000001F	04aa50c2c44880	
Princess	princess	princess@email.com	S0000001E	2068a989	
Sherna Liew	shernaliu	shernaliu@gmail.com	S0000001A	04fc8912a13580	

Showing 1 to 6 of 6 entries

Previous 1 Next

Refresh

Add New User

Full Name

Enter Full Name

Username

Enter Username

Email address

Enter Email

NRIC

Enter NRIC

Add New User

Edit Existing User

Full Name

Enter Full Name

Username

Enter Username

Email address

Enter Email

NRIC

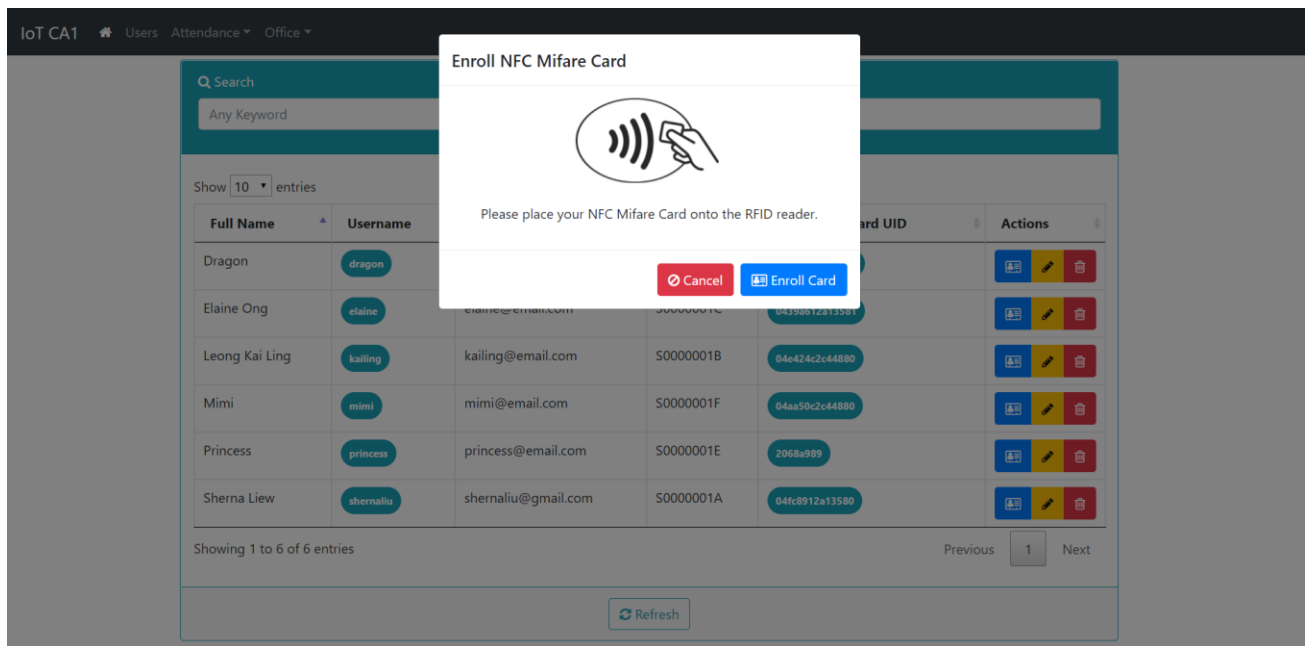
Enter NRIC

Cancel Save

© 2017 Liew Zhi Li (Sherna). All Rights Reserved.

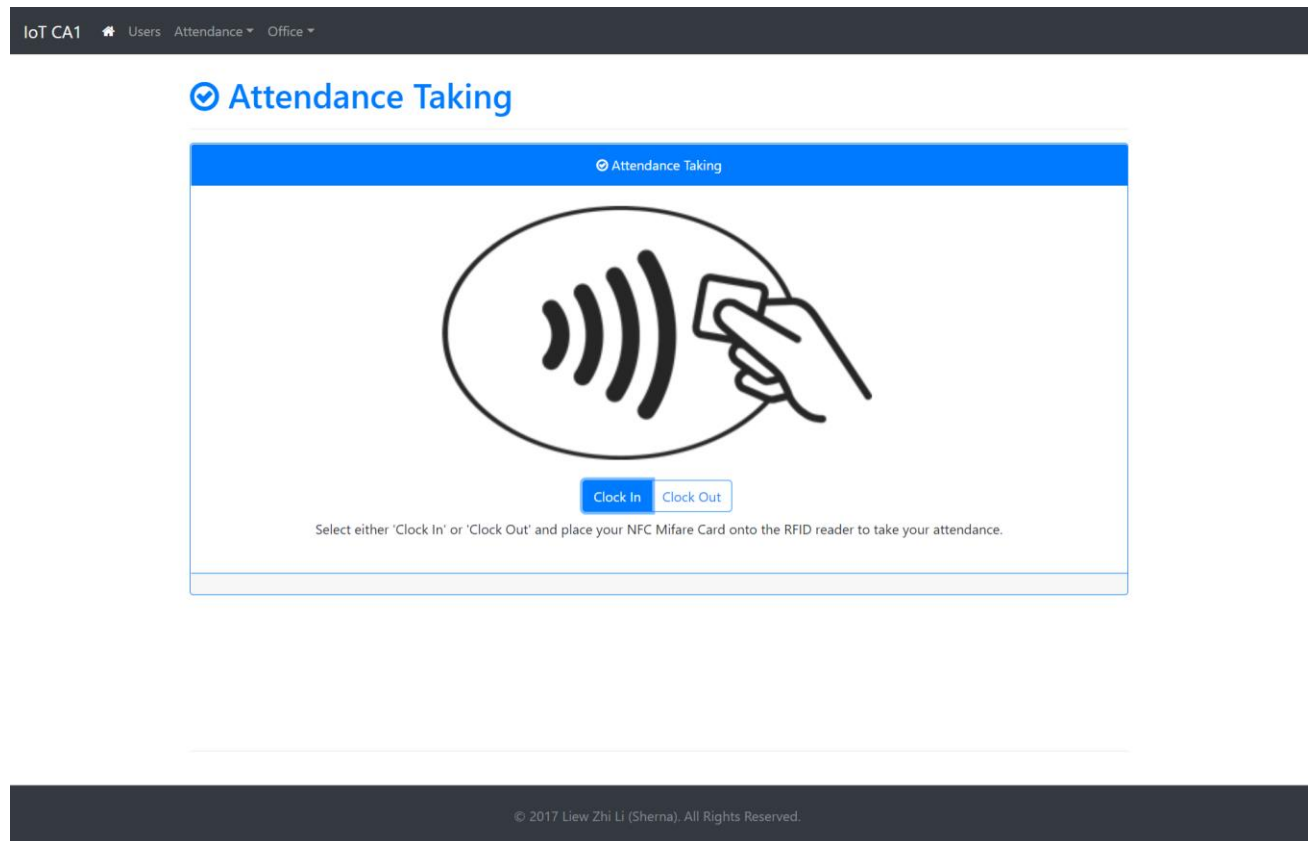
Upon clicking on the blue “Enroll NFC Mifare Card”, a modal pops up to prompt the user to place the employee card onto the RFID reader.

Once the UID for the card has been captured, it will be displayed. The user then clicks on “Enroll Card” button.



The next image shows the Attendance Taking page.

Users can navigate to this page, toggle whether to Clock In or Clock Out and tap their employee card onto the RFID reader.



The next image shows the Attendance Log page.

The Attendance Log table shows a historical view of all users's clock in timestamp, clock out timestamp and the total hours clocked.


The user can also search for specific records using the Search bar and type in any search keyword. The "Refresh" button at the bottom allows the user to refresh the Attendance Log table.

IoT CA1

Users

Attendance

Office

 Attendance Log

Search

Any Keyword

Show 10 entries

Full Name	Clock In Timestamp	Clock Out Timestamp	Total Hours Clocked
Dragon	Dec 28, 2017 11:00:00 AM	Dec 28, 2017 02:12:45 PM	03 hours 13 mins
Elaine Ong	Dec 28, 2017 07:00:00 AM	Dec 28, 2017 02:11:56 PM	07 hours 12 mins
Leong Kai Ling	Dec 28, 2017 09:00:00 AM	Dec 28, 2017 02:12:20 PM	05 hours 12 mins
Princess	Dec 28, 2017 01:00:00 PM	Dec 28, 2017 02:13:09 PM	01 hour 13 mins
Sherna Liew	Dec 28, 2017 07:00:00 AM	Dec 28, 2017 02:11:41 PM	07 hours 12 mins

Showing 1 to 5 of 5 entries

Previous1Next

Refresh

© 2017 Liew Zhi Li (Sherna). All Rights Reserved.

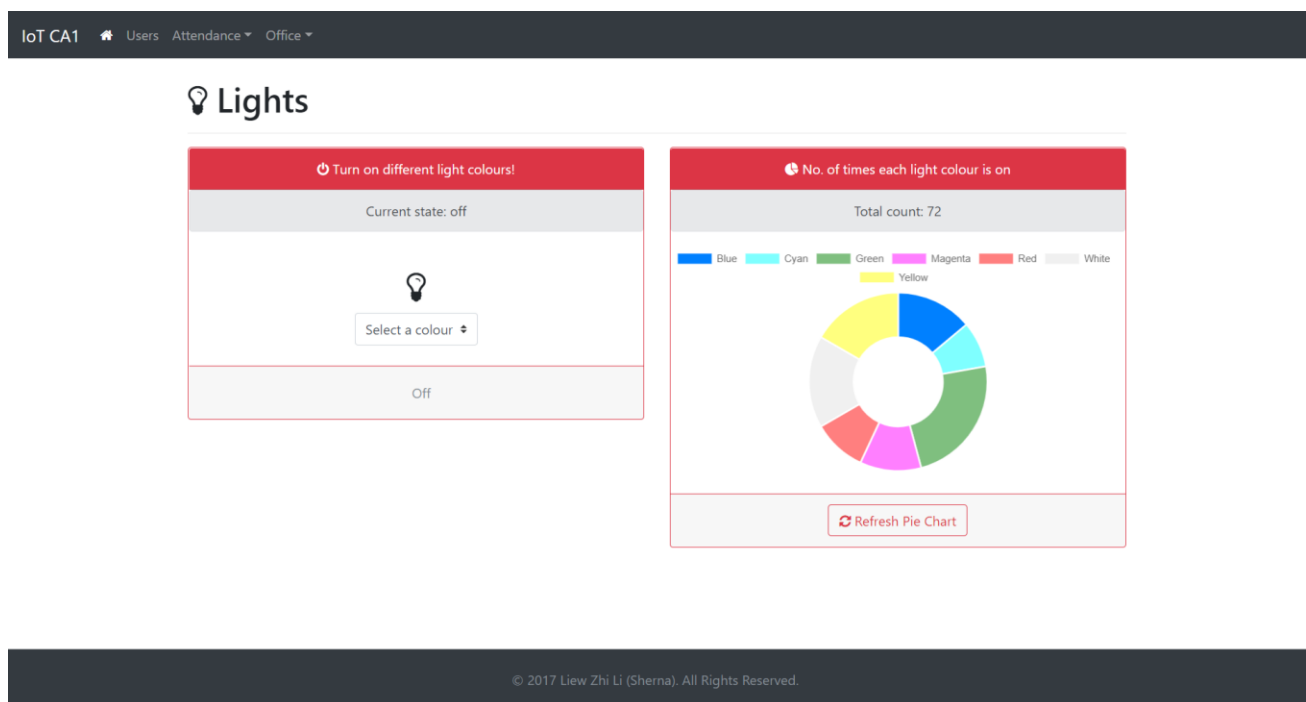
The next image shows the Lights page.

This page allows the user to view the current and historical state of the RGB LED. The current state of the RGB LED can be “off”, “blue”, or “red + blinking” depending on what the user toggles.

The pie or doughnut chart on the right shows the historical data of the number of times each light colour is on. The total count for all colours is stated clearly as well.

The user can hover their mouse pointer over each colour to see the number of times the colour was turned on.

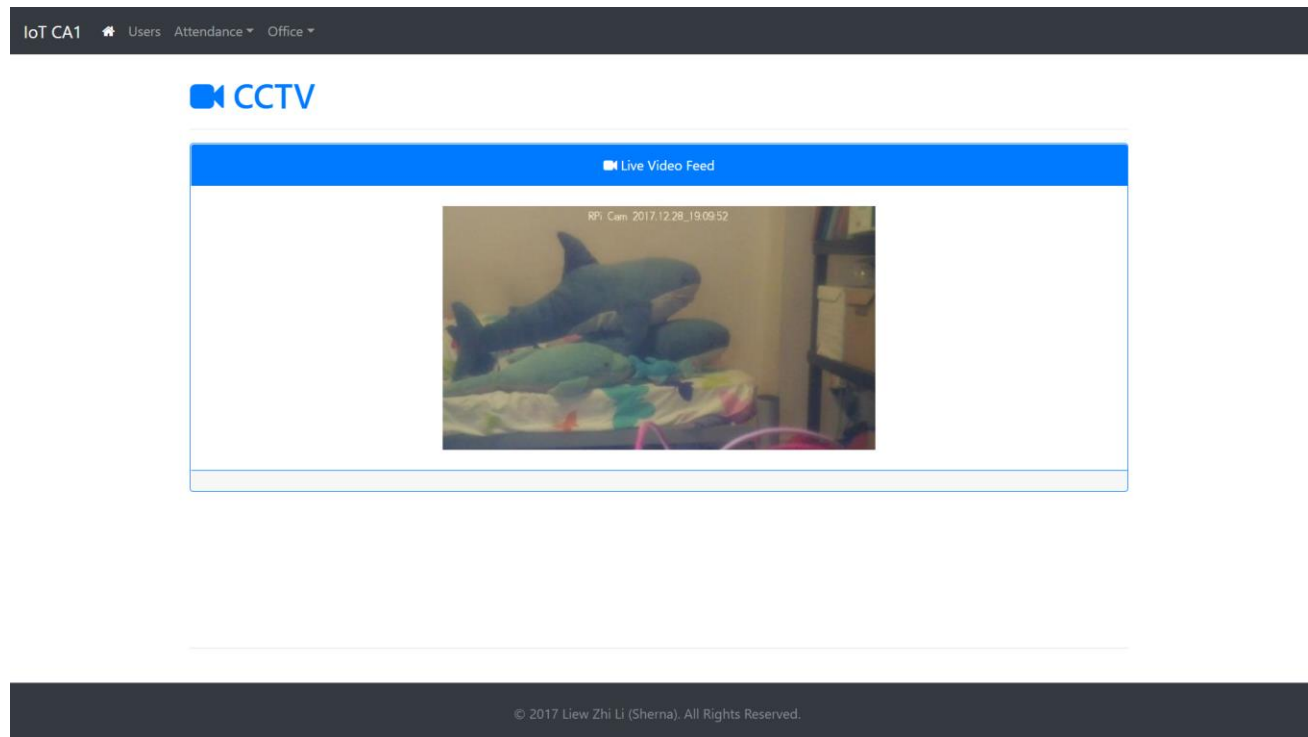
The “Refresh Pie Chart” button allows users to refresh the chart. By default, when the user toggles the RGBLED, the pie chart will be automatically refreshed, and this allows greater convenience for the user. 😊



Lastly, the image below shows the CCTV page.

This page shows the live video feed of the camera that is mounted outside of the office. The date and timestamp of the camera operating is superimposed on the video itself.

PS: It looks like there is a family of sharks and whales visiting my office this evening! 😊



Section 2

Hardware requirements

Hardware checklist

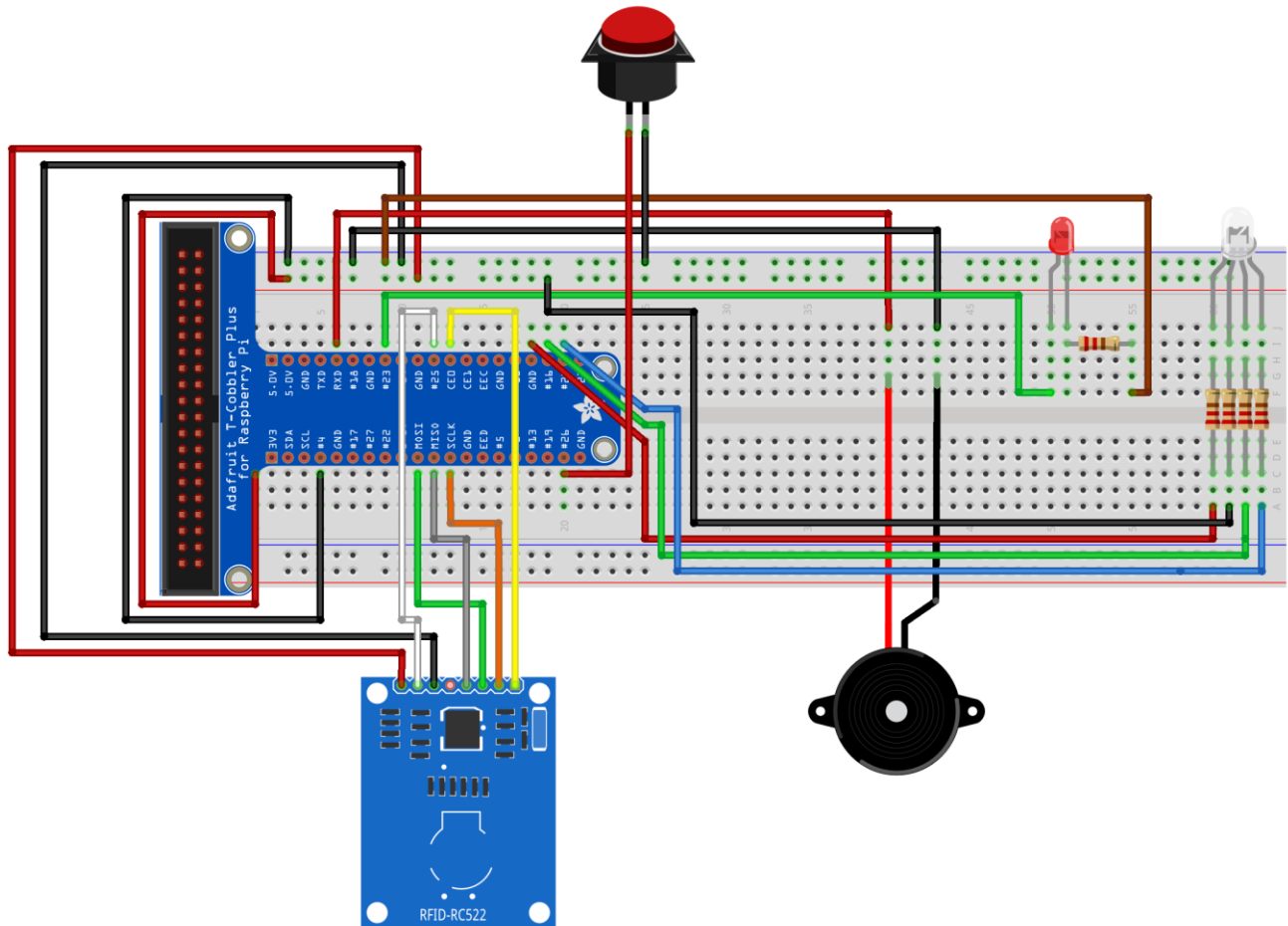
The following table shows the hardware required to replicate my project.

No.	Hardware Module	Model Name	Quantity	Price/unit (\$)	Where to buy?
1	Raspberry Pi 3	Model B	2	52.21	Element14
2	PiCam	Version 2	1	37.29	Element14
2	Active Buzzer		1	1.08	Banggood
3	Common Anode RGB LED		1	0.06	AliExpress
4	Red LED		1	0.02	AliExpress
5	RFID Reader	MFRC522	1	5.43	AliExpress
6	Arduino	UNO R3		4.57	AliExpress
7	Analog Joystick	KY-023	1	0.73	AliExpress
8	Servo	TowerPro SG90	2	1.49	AliExpress
9	Arcade Button (any colour)		1	3.00	AliExpress
10	Full-length Breadboard		1	2.24	AliExpress
11	Half-length Breadboard		1	1.23	AliExpress
12	Jumper Wires (M-to-M, M-to-F, F-to-F)		Up to you	4.70	AliExpress
13	220Ω Resistors		5	0.02	AliExpress

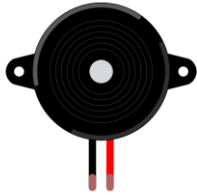
Section 3

Setup Hardware (Raspberry Pi)


The following are detailed instructions on how to set up the project according to the schematics.



Buzzer


Image	
Description	<p>A buzzer is an audio signaling device which is commonly found in circuits to create a buzzing or beeping noise.</p> <p>Buzzers can be categorized as active buzzers and passive ones. In our project, we will be using an active buzzer since they are simpler to use albeit more expensive.</p> <p>An active buzzer can be connected just like an LED but they are even easier to use because a resistor is not needed.</p> <p>A buzzer typically has 2 pins</p> <ul style="list-style-type: none"> - VOUT – Connect this to a GPIO pin to control its value - GND – Connect this to ground
Instruction	<p>Connect jumper wires from the longer leg (Anode) to GPIO18.</p> <p>Connect jumper wires from the the shorter leg (Cathode) to GND.</p>

Red LED

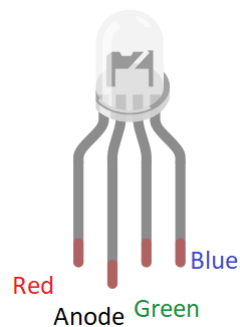
Image	
Description	<p>LED stands for Light Emitting Diode. It lights up when electricity is passed through it in the right direction. One leg of the LED is always longer than the other. This is denoted with a kink in one of its legs in the above image.</p> <p>The longer leg (Anode) is always connected to the positive supply of the circuit.</p> <p>The shorter leg (Cathode) is connected to the negative side of the power supply, which is Ground.</p> <p>LEDs will only light up if the power is supplied in the right direction.</p>

	If you do not connect them correctly, they simply won't light up. So, if you find them not lighting up, it may be that you have not connected them in the right direction.
Instruction	Place the Red LED onto the breadboard first, making note of where you insert the longer leg.

Resistor for LED

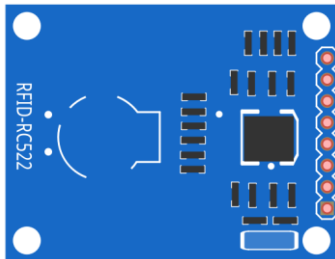
Image	
Description	<p>Resistors are used to limit the amount of electricity flowing in a circuit. It works by limiting the amount of current flowing in the circuit.</p> <p>Resistance are measured with the unit called Ohm (Ω). The higher the resistance, the more it limits the current.</p> <p>Resistor values are marked by the coloured bands.</p> <p>In our project, we will be using a 220Ω resistor. This is identified by the four coloured bands in this sequence: Orange, Orange, Brown, and then Gold.</p> <p>When you connect a resistor, the direction of the resistor can be disregarded as current flows in both ways through them.</p>
Instruction	<p>Place one end of the resistor to the shorter leg of the LED and the other onto an empty hole.</p> <p>Connect jumper wires from the longer leg of the LED to GPIO24.</p> <p>Connect jumper wires from the free end of the resistor to GND.</p>

Common Anode RGB LED

Image	
Description	Red, Green, Blue (RGB) LED is made up of three tiny LEDs of 3 primary colors red, green and blue with a common terminal.

	<p>There are 2 types of RGB LEDs:</p> <ul style="list-style-type: none"> - Common Anode - Common Cathode <p>The difference between them is that they either can have a positive terminal (Anode) or a negative terminal (Cathode). Similar to regular LED, you can find one of the legs longer than the other.</p> <p>For a Common Anode RGB LED, you connect the Anode to 3.3V or 5V. Then, we write a LOW to that pin to turn the it ON and a HIGH to turn it off.</p> <p>For a Common Cathode RGB LED, you connect the Cathode to GND. Then, we write a HIGH to turn it on and a LOW to turn it off.</p> <p>When varying voltage is supplied to the RGB LED, we can produce lots of different colours. For example, if we supply 1 to all Red, Green and Blue LEDs, we can make white light!</p>
Instruction	<p>Place the RGB LED onto the breadboard.</p> <p>Connect 4 pieces of 220Ω resistors for each leg.</p> <p>Connect jumper wires for Red, Green, Blue to GPIO16, GPIO20, GPRI021 respectively.</p>

MFRC522 RFID Reader

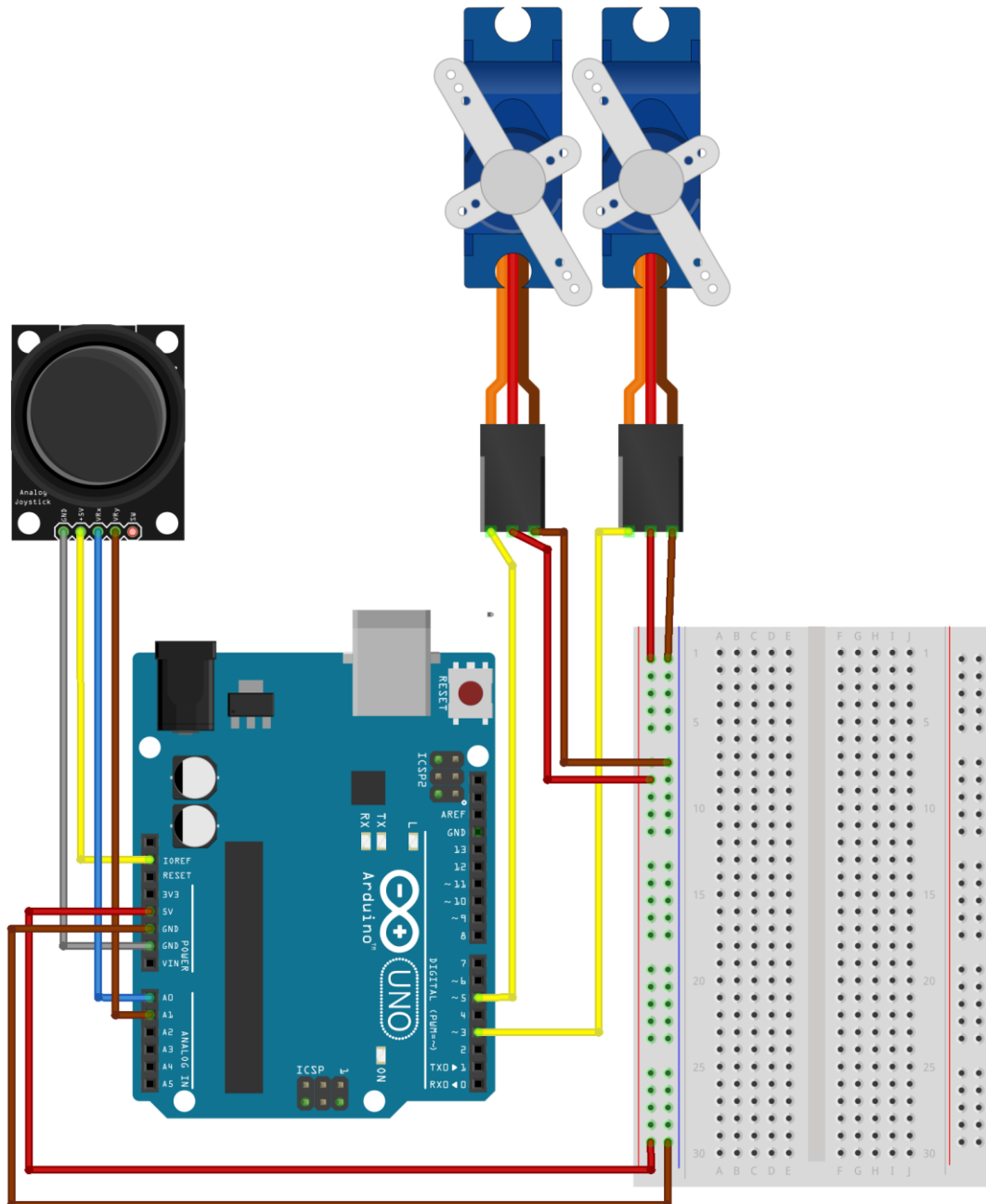
Image	
Description	<p>Radio-frequency identification (RFID) is commonly used in many applications such as unlocking specific doors as well as tagging items to perform tracking or identification.</p> <p>RFID tags contain electronic data and radio-waves are used to transfer these data from the RFID tag to the reader to identify an item.</p> <p>If you've been to the public library, the books are actually tagged with a paper thin RFID tag. When you borrow a book, you would place the book onto the RFID reader, and the RFID reader reads the data in the RFID tag and identifies that particular book you are borrowing against its database.</p>

	This RFID reader is capable of reading Near-field Communication (NFC) Mifare Classic cards. NFC is based on RFID protocols. The difference between NFC is that its capable of bi-directional communication, whereas RFID is capable of one-directional communication.
Instruction	Connect a yellow jumper wire from SDA of the RFID to SPICE0 on the RPi. Connect an orange jumper wire from SCK of the RFID to SCLK on the RPi. Connect a green jumper wire from MOSI of the RFID to MOSI on the RPi. Connect a blue jumper wire from MISO of the RFID to MSIO on the RPi. Connect a black jumper wire from GND of the RFID to GND on the RPi. Connect a white jumper wire from RST of the RFID to GPIO25 on the RPi. Connect a red jumper wire from 3.3V of the RFID to 3.3V on the RPi.

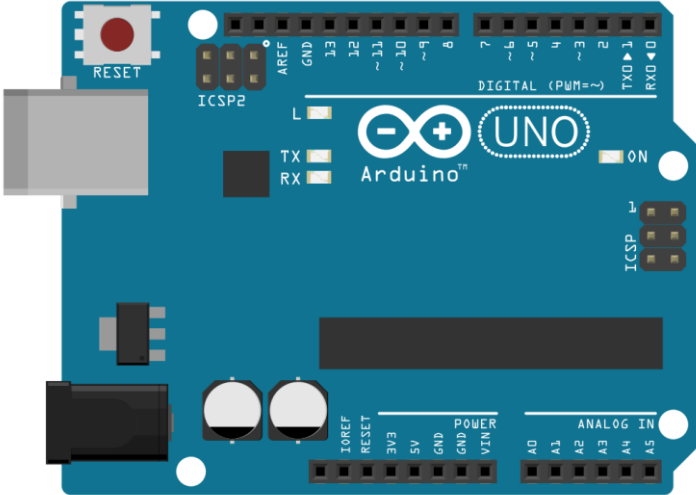
Section 4

Setup Hardware (Arduino)

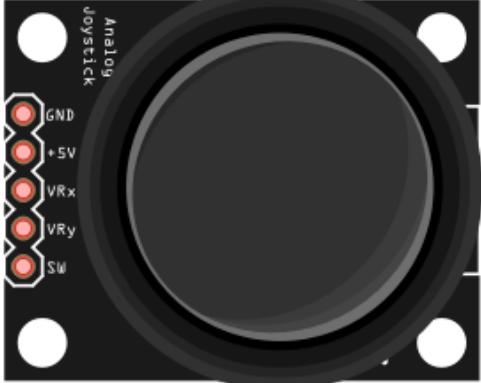
The following are detailed instructions on how to set up the project according to the schematics.



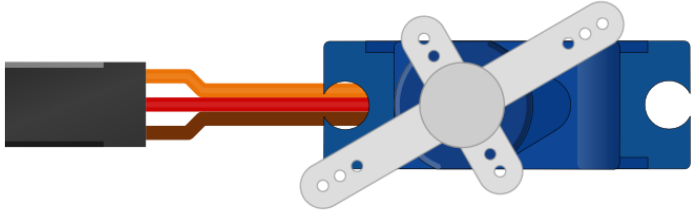
fritzing

Arduino UNO R3	
Image	
Description	<p>The Arduino has a total of 20 digital input and output pins. Out of the 20, 6 of these pins can be used as PWM outputs and 6 can be used as analog inputs.</p> <p>The difference between an Arduino and a Raspberry Pi is that the Arduino is a micro-controller, not a full-fledged computer like Raspberry Pi.</p> <p>They do not run on any Operating System (OS). We simply upload the codes on the Arduino using the Arduino IDE, and the Arduino executes the codes in the program in a loop.</p> <p>You will need a foundational understanding of C or C++ as most of the codes written for Arduino are either in C or C++.</p> <p>The Arduino is perfect for simple logics and applications. If you require more diverse and complex logics, the Raspberry Pi is a better choice.</p>
Instruction	-

KY-023 Joystick

Image	
Description	<p>The KY-023 is an analog joystick for Arduino. The joystick can move in both the X-Axis and Y-Axis and their positions can be measured as an analog voltage at an output pin.</p> <p>The joystick has two potentiometer and a push button switch.</p>
Instruction	<p>Connect a jumper wire from the GND of the joystick to the GND of the Arduino.</p> <p>Connect a jumper wire from the 5V of the joystick to the 5V of the Arduino.</p> <p>Connect a jumper wire from the VRx of the joystick to the A0 of the Arduino.</p> <p>Connect a jumper wire from the VRy of the joystick to the A1 of the Arduino.</p>

TowerPro SG90 Servo

Image	
Description	<p>The servo is a tiny device with an output shaft that has a range of rotation angle of 0° to 180°. It has high output power. Servos require a lot of current to operate as there is a motor inside. This type of servo is good for beginners.</p>
Instruction	<p>Connect a jumper wire from the GND (brown) of the servo to the GND of the Arduino.</p> <p>Connect a jumper wire from the 5V (red) of the servo to the 5V of the Arduino.</p> <p>Connect a jumper wire from the signal (orange) of the servo to the PWM3 of the Arduino.</p> <p>Note: For the second servo, do the same for GND and 5V. For the signal, connect it to PWM5.</p>

Section 5

What is MEN stack?

The technology stack that I have chosen to use is a hybrid type. The main bulk of the application that SOMS runs on is the MEN stack.

MEN stack stands for:



MongoDB is a popular non-relational database, it also called a NoSQL database. It is different from a traditional relational database like MySQL. Data in MongoDB is stored using JSON-structured documents with schemas.

Mongoose is a MongoDB Object Document Mapper (ODM) which helps to translate the objects in code with the document representation of the data. In contrast, MySQL would use a Object Relational Mapper (ORM) such as Hibernate, Entity Framework, and others.

express

Express.js is a light-weight yet flexible server-side web application framework for Node.js. With Express, you can build Application Programming Interfaces (APIs) and single-page or multi-page web applications. Express supports the Model-View-Controller (MVC) architecture which provides a clean separation of concerns among the UI, Business Logic and Model.



Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine. This means that it converts the JavaScript code to machine code extremely fast.

Node.js uses a programming model that is a non-blocking event-driven I/O. What this means is that for example, your web browser makes a request for index.html.

Node.js accepts the request and executes a function to retrieve the file. While it is retrieving, Node.js can proceed to service subsequent web requests. Once the file has been retrieved, a callback function will be added to the Node.js server queue. Node.js will then execute the function to render the index.html and send it back to the client's web browser.

Other than the MEN stack, we will also be using Flask, which is a Python micro web framework to build APIs. We can write Python codes to communicate with the Raspberry Pi modules and expose them to the APIs.

In conclusion, you may find other popular variants of the tech stack such as MEAN, MEVN, MERN. The 'A', 'V' and 'R' in each of these acronyms are Angular, Vue.js and React.js. These are presentation frameworks or view engines to help to make DOM manipulations easier.

The reason I decided not to use a view engine is because I am a beginner who just started learning about this tech stack. I feel that including a view engine will make the learning curve much steeper and there is always the issue of time constraint. In order to manipulate DOM elements, I used Javascript and JQuery, which I am familiar with.

Section 6

Setup Node.js development environment

Step 1: Install MongoDB

Let's get started with the MongoDB installation on Raspberry Pi which is pretty simple. Execute the following commands line by line.

```
sudo apt-get update
sudo apt-get upgrade
sudo apt-get install mongodb-server
```

To start the MongoDB on the Raspberry Pi, type this command.

```
mongod
```

The binaries are stored in the `/usr/bin/` folder while the datas are in the `/var/lib/mongodb/` folder. You can check everything is ok by using the mongo shell:

```
mongo
```

In order to create a database, you actually don't need to do anything because MongoDB does not have any commands to create database. 😊

In MongoDB, we just need to "use" database of whatever name you want and just save a single record in database to create it.

Let's say we want to "create" a database called `iot_ca1`. Just run the following commands to save a single record which will automatically create the database and table for you.

```
use iot_ca1
s = { Name : "Sherna Liew" }
db.test.insert(s);
```

The result of running this command is shown in the screenshot below:

```
MongoDB Enterprise > s = { Name : "Sherna Liew" }
{ "Name" : "Sherna Liew" }
MongoDB Enterprise > db.test.insert(s);
WriteResult({ "nInserted" : 1 })
MongoDB Enterprise > db.test.find()
{ "_id" : ObjectId("5a46284959fe82f3c5739dcf"), "Name" : "Sherna Liew" }
MongoDB Enterprise > |
```

When you do `db.test.find()` and you are able to get back the data you just saved, then it is working properly. 😊

Step 2: Install Node.js

Let's proceed with an installation of the latest version of Node at the moment which is Node 9.3.0.

```
curl -sL https://deb.nodesource.com/setup_9.x | sudo -E bash -
```

Now that we have added the NodeSource package repository, we can move on and install Node.js!

```
sudo apt install nodejs
```

We can then test and see what version of Node we are running and launch the Node REPL as we discussed in the previous article as a quick test to confirm the installation was successful.

```
$ node -v
v9.3.0
$ node
> 1 + 3
4
> # we can hit Ctrl-C twice to exit the REPL and get back to the bash
(shell) prompt.
```

Congratulations! You are now ready to start development! 😊

Section 7

Setup required libraries and dependencies

Now, we will be creating the directory structure to store our project in. This is the tree view of the entire directory which is what we will be achieving at the end.

```
├── arduino
│   └── 2_servos.ino
├── back_end
│   ├── config.js
│   ├── models
│   │   ├── Attendance.js
│   │   ├── Light.js
│   │   └── User.js
│   ├── package.json
│   ├── package-lock.json
│   ├── routes
│   │   ├── attendance.js
│   │   ├── light.js
│   │   └── user.js
│   └── server.js
├── documentation
│   ├── iot_ca1_final_fritzing_diagram.fzz
│   └── iot_ca1_final_fritzing_diagram.png
├── front_end
│   ├── attendance.html
│   ├── cctv.html
│   ├── colours.css
│   ├── index.html
│   ├── lights.html
│   ├── no_image.jpg
│   ├── package.json
│   ├── package-lock.json
│   ├── rfid.png
│   ├── starter-template.css
│   ├── take_attendance.html
│   └── user_management.html
├── python
│   └── server.py
└── README.md
```

First, execute the following command to create the “soms” directory.

```
mkdir soms && cd soms
```

Execute the subsequent command to create these 4 directories.

```
mkdir back_end front_end python arduino
```

As you can see, we have separate folders for storing back end and front end files, as well as separate folders for python and arduino codes. This will make our life easier. 😊

Execute the subsequent commands to make these 2 directories inside back_end folder.

```
cd back_end  
mkdir models routes
```

The models and routes will be used to store all our Data Model and Routes respectively.

Now, since the project will use quite a number of dependencies for back end and front end, we need a way to store them. To do that, we need a package.json file to keep a record of all the dependencies we installed and used.

Ensure you are in the folder back_end.

Execute the subsequent command to create the package.json file

```
npm init
```

It will ask you a bunch of questions and you can just type whatever you'd like!

This is what I have told npm and they will include these information inside the file.

```
{  
  "name": "shernaliew-iot",  
  "version": "1.0.0",  
  "description": "IOT CA1 Assignment",  
  "main": "server.js",  
  "scripts": {  
    "test": "echo \\\"Error: no test specified\\\" && exit 1"  
  },  
  "repository": {  
    "type": "git",  
    "url": "git+https://github.com/shernaliu/soms.git"  
  },  
  "author": "Sherna Liew",  
  "license": "ISC",  
  "bugs": {  
    "url": "https://github.com/shernaliu/soms/issues"  
  },  
  "homepage": "https://github.com/shernaliu/soms#readme"  
}
```

Once you have completed this step, you should be able to find a package.json file inside back_end. After that, we will be installing the dependencies needed and save them inside package.json.

Execute the following commands to install all the dependencies.

```
npm install --save body-parser express ip moment mongoose rc522 rprio  
socket.io whatwg-fetch
```

The `--save` command is important as this will record down all the installed dependencies inside package.json.

Later on, when we set up version control, we will not be committing all the dependencies because it will be too much. Instead, anyone who clones this project in the future can install all the dependencies by themselves by using “npm install”.

Once you are done, you can find a folder called node_modules inside back_end. This is where npm has installed all the dependencies!

Congratulations! Now you have installed the dependencies required for back end. But wait! We still need to install dependencies required for the front end as well.

At this point, the steps will be largely similar since we have gone through it once. In terminal, simply cd into front_end and run the command to create the package.json file.

Next, execute the following command to install these front end dependencies.

```
npm install --save bootstrap@4.0.0-beta.2 bootstrap-toggle chart.js font-  
awesome jquery socket.io sweetalert2 whatwg-fetch
```

Once you are done, you can find a folder called node_modules inside front_end. This is where npm has installed all the dependencies!

Congratulations! Now you have installed the dependencies required for front end.

That's it! 😊

Section 8

Creating the SOMS web application (Backend)

We are now ready to create the back end files needed for our project!

At this point, you can simply copy and paste the codes. Not much explanation will be needed since I have inserted the explanation inside the codes as comments. ☺

In back_end, create a new file called server.js

Server.js

```
var express = require('express')
var bodyParser = require('body-parser')
var app = express()
var http = require('http').Server(app)
var io = require('socket.io')(http)
var ip = require("ip")
var mongoose = require('mongoose')
var rc522 = require("rc522")
var rpio = require('rpio')
mongoose.Promise = Promise

//
// Register Node.js middleware
// -----
app.disable('etag'); // Resolve HTTP status of 304 Modified due to caching
https://stackoverflow.com/questions/18811286/nodejs-express-cache-and-304-status-code
app.use(express.static('../front_end'))
app.use(bodyParser.json())
app.use(bodyParser.urlencoded({ extended: false }))

var dbUrl = 'mongodb://localhost:27017/iot_ca1'

//
// Import routes
// -----
var user = require('./routes/user');
var light = require('./routes/light');
var attendance = require('./routes/attendance');

//
// Use routes
// -----
app.use('/api', user);
app.use('/api', light);
app.use('/api', attendance);
```

```
// --- Database Connection to MongoDB ---
mongoose.connect(dbUrl, { useMongoClient: true }, (err) => {
  console.log('MongoDB connection err: ', err)
}))

io.on('connection', (socket) => {
  console.log('a user connected')
  // emit ip address
  io.sockets.emit("ip", ip.address())
})

rc522(function (rfidSerialNumber) {
  io.sockets.emit("rfid", rfidSerialNumber); // Sends the RFID Serial Number
  through Socket.IO
  buzz_rfid();
  console.log(rfidSerialNumber);
});

/*
 * Init device (Buzzer) on Pin 12 (GPIO18)
 * Init device (Green LED) on Pin 18 (GPIO24)
 * Set the initial state to low. The state is set prior to the pin becoming
 * active, so is safe for devices which require a stable setup.
 */
var buzzer_pin = 12;
var led_pin = 18;
rpio.open(buzzer_pin, rpio.OUTPUT, rpio.LOW);
rpio.open(led_pin, rpio.OUTPUT, rpio.LOW);

// Function that beeps the buzzer & lit the green LED for 0.5 seconds
function buzzer_ring() {
  console.log("buzz")
  /* On buzzer for 50ms */
  rpio.write(buzzer_pin, rpio.HIGH);
  rpio.msleep(100);

  /* Off both devices for 50ms */
  rpio.write(buzzer_pin, rpio.LOW);
  rpio.msleep(100);

  rpio.msleep(250);

  rpio.write(buzzer_pin, rpio.HIGH);
  rpio.msleep(150);

  /* Off both devices for 500ms */
  rpio.write(buzzer_pin, rpio.LOW);
```

```
    rpio.msleep(150);

}

// Function that beeps the buzzer & lit the green LED in a cute pattern
function buzz_rfid() {
    for (x = 0; x <= 5; x++) {
        /* On both devices for half of a half of a second (50ms) */
        rpio.write(buzzer_pin, rpio.HIGH);
        rpio.write(led_pin, rpio.HIGH);
        rpio.msleep(50);

        /* Off both devices for half of a half of a second (50ms) */
        rpio.write(buzzer_pin, rpio.LOW);
        rpio.write(led_pin, rpio.LOW);
        rpio.msleep(50);
    }
}

var server = http.listen(3000, () => {
    console.log('NodeJS server is running on ' + ip.address() + ':' +
server.address().port)
    console.log(io.sockets.name)
})
```

Let's create our models!

In back_end/models, create a new file called User.js

User.js

```
var mongoose = require('mongoose');

var UserSchema = new mongoose.Schema({
  fullName: {
    type: String,
    required: [true, "Cannot be blank"]
  },
  username: {
    type: String,
    lowercase: true,
    required: [true, "Cannot be blank"],
    match: /^[a-zA-Z0-9]+$/, 'is invalid',
    index: true
  },
  email: {
    type: String,
    lowercase: true,
    required: [true, "Cannot be blank"],
    match: [/^S+@S+\.S+/, 'is invalid'],
    index: true
  },
  nric: {
    type: String,
    required: [true, "Cannot be blank"],
    match: /^[STFG]\d{7}[A-Z]$/, 'is invalid'
  },
  nfcCard: {
    type: String
  }
}, { timestamps: true });

const User = mongoose.model('User', UserSchema);
module.exports = User;
```

In back_end/models, create a new file called Attendance.js

Attendance.js

```
var mongoose = require('mongoose');

var AttendanceSchema = new mongoose.Schema({
  clockInTimestamp: {
    type: Date,
    required: true
  },
  clockOutTimestamp: {
    type: Date
  },
  totalClockedHours: {
    type: Number
  },
  user: {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'User'
  }
}, { timestamps: true });

const Attendance = mongoose.model('Attendance', AttendanceSchema);
module.exports = Attendance;
```

In back_end/models, create a new file called Light.js

Light.js

```
var mongoose = require('mongoose');

var LightSchema = new mongoose.Schema({
  colour: {
    type: String
  },
  timestamp:{
    type: Date
  }
});

const Light = mongoose.model('Light', LightSchema);
module.exports = Light;
```

Congratulations! You have set up all the models. Let's move on with routes!

Routes are simply where the API endpoints are written.

Note that all routes have the same name as their model itself, but they start with a lowercase letter.

In `back_end/routes`, create a new file called `user.js`

`user.js`

```
var express = require('express');
var router = express.Router();
var User = require('../models/User')
var ObjectId = require('mongoose').Types.ObjectId;

// Get all users
router.route('/users').get(function (req, res) {
  User.find({}, (err, users) => {
    res.send(users)
  })
})

// Get all users count
router.route('/usersCount').get(function (req, res) {
  User.count({}, (err, count) => {
    var response = {
      count: count,
    };
    res.status(200).send(response);
  })
})

// Get a user by id
router.route('/users/:id').get(function (req, res) {
  User.find({ _id: new ObjectId(req.params.id) }, (err, users) => {
    res.send(users)
  })
})

// Get a user by nfcCard
router.route('/users/getUserByNfcCard/:id').get(function (req, res) {
  User.find({ nfcCard: (req.params.id) }, (err, users) => {
    if (users.length === 0) { // users is return as array
      // Return 404 resouce not found
      res.sendStatus(404)
    } else {
      res.status(200).send(users)
    }
  })
})
})
```

```
// Create a new user
router.route('/users').post(function (req, res) {
  var user = new User(req.body)
  user.save((err) => {
    if (err) {
      res.sendStatus(500)
    }
    res.sendStatus(200)
  })
})

// Update a user
router.route('/users/:id').put(function (req, res) {

  // Find the existing resource by ID
  User.findById(req.params.id, (err, currentUser) => {
    if (err) {
      res.status(500).send(err);
    } else {
      console.log(currentUser);
      var newUser = new User(req.body)

      // Update each attribute
      currentUser.fullName = newUser.fullName;
      currentUser.username = newUser.username;
      currentUser.email = newUser.email;
      currentUser.nric = newUser.nric;

      // Save the updated document back to the database
      currentUser.save((err, updatedUser) => {
        if (err) {
          res.status(500).send(err);
        }
        res.status(200).send(updatedUser);
      })
    }
  })
})

// Update nfcCard for a user
router.route('/users/updateNfcCard/:id').put(function (req, res) {

  // Find the existing resource by ID
  User.findById(req.params.id, (err, currentUser) => {
    if (err) {
      res.status(500).send(err);
    } else {
      console.log(currentUser);
```



```
    // Update nfcCard attribute
    currentUser.nfcCard = req.body.nfcCard;

    // Save the updated document back to the database
    currentUser.save((err, updatedUser) => {
      if (err) {
        res.status(500).send(err);
      }
      res.status(200).send(updatedUser);
    })
  }
})
})

// Delete an existing user
router.route('/users/:id').delete(function (req, res) {
  User.findByIdAndRemove(req.params.id, (err, user) => {
    var response = {
      message: "User successfully deleted",
      id: user._id
    };
    res.status(200).send(response);
  })
})

module.exports = router;
```

In back_end/routes, create a new file called attendance.js

attendance.js

```
var express = require('express');
var router = express.Router();
var Attendance = require('../models/Attendance')
var ObjectId = require('mongoose').Types.ObjectId;
var moment = require('moment');

// Get all attendance
router.route('/attendance').get(function (req, res) {
  Attendance.find({}).populate('user').exec((err, attendances) => {
    res.send(attendances)
  })
})

// Get all attendance for a specific date
router.route('/attendance/searchByDate/:date').get(function (req, res) {
  console.log("searchByDate called")
  // Query by date: Get start of date & end of date
  // Example:
  // startDate: Sat Dec 23 2017 00:00:00 GMT+0800 (+08)
  // endDate: Sun Dec 24 2017 00:00:08 GMT+0800 (+08)
  var date = new Date(req.params.date);
  var startDate = new Date(date.getFullYear(), date.getMonth(), date.getDate());
  var endDate = new Date(date.getFullYear(), date.getMonth(), date.getDate(), 23,
59, 59, 9999);

  console.log(date)
  console.log(startDate)
  console.log(endDate)

  Attendance
    .find(
      {
        clockInTimestamp: { $gte: startDate, $lt: endDate }
      }
    )
    .populate('user')
    .exec((err, attendances) => {
      res.status(200).send(attendances);
    })
  })

// Get all clock in count for today
router.route('/attendance/clockInCount').get(function (req, res) {
  // Query by today's date: Get start of today & end of today
  // Example:
```

```
// startOfToday: Sat Dec 23 2017 00:00:00 GMT+0800 (+08)
// endOfToday: Sun Dec 24 2017 00:00:08 GMT+0800 (+08)
var now = new Date();
var startOfToday = new Date(now.getFullYear(), now.getMonth(), now.getDate());
var endOfToday = new Date(now.getFullYear(), now.getMonth(), now.getDate(), 23,
59, 59, 9999);

Attendance.count(
  {
    clockInTimestamp: { $gte: startOfToday, $lt: endOfToday }
  }, (err, count) => {
    var response = {
      count: count,
    };
    res.status(200).send(response);
  })
})

// Get all clock out count for today
router.route('/attendance/clockOutCount').get(function (req, res) {
  // Query by today's date: Get start of today & end of today
  // Example:
  // startOfToday: Sat Dec 23 2017 00:00:00 GMT+0800 (+08)
  // endOfToday: Sun Dec 24 2017 00:00:08 GMT+0800 (+08)
  var now = new Date();
  var startOfToday = new Date(now.getFullYear(), now.getMonth(), now.getDate());
  var endOfToday = new Date(now.getFullYear(), now.getMonth(), now.getDate(), 23,
59, 59, 9999);

  Attendance.count(
    {
      clockOutTimestamp: { $gte: startOfToday, $lt: endOfToday }
    }, (err, count) => {
      var response = {
        count: count,
      };
      res.status(200).send(response);
    })
  })

// Get an attendance record where user._id = 'xxxxx' & clockedInTimestamp == today
router.route('/attendance/checkIfClockedIn/:id').get(function (req, res) {
  // Query by today's date: Get start of today & end of today
  // Example:
  // startOfToday: Sat Dec 23 2017 00:00:00 GMT+0800 (+08)
  // endOfToday: Sun Dec 24 2017 00:00:08 GMT+0800 (+08)
  var now = new Date();
  var startOfToday = new Date(now.getFullYear(), now.getMonth(), now.getDate());
```

```

    var endOfToday = new Date(now.getFullYear(), now.getMonth(), now.getDate(), 23,
59, 59, 9999);

    Attendance
        .find(
            {
                clockInTimestamp: { $gte: startOfToday, $lt: endOfToday },
                "user": new ObjectId(req.params.id)
            })
        .populate('user')
        .exec((err, attendances) => {
            // There can be only 1 attendance record for a user per day
            res.send({ "count": attendances.length })
        })
    })

// Get an attendance record where user._id = 'xxxxx' & clockedOutTimestamp == null
router.route('/attendance/checkIfClockedOut/:id').get(function (req, res) {
    Attendance
        .find(
            {
                clockOutTimestamp: null,
                "user": new ObjectId(req.params.id)
            })
        .populate('user')
        .exec((err, attendances) => {
            // There can be only 1 attendance record for a user per day
            res.send({ "count": attendances.length })
        })
    })

// Clock in a user
router.route('/attendance/clockIn').post(function (req, res) {

    // TODO: Codes to turn back the time for testing purposes
    // You can uncomment them for testing purposes
    // var now = new Date();
    // var yesterday = new Date(now.getFullYear(), now.getMonth(), now.getDate(),
now.getHours() - 1);
    // req.body.clockInTimestamp = yesterday;

    var attendace = new Attendance(req.body)
    attendace.save((err, createdAttendance) => {
        if (err) {
            res.status(500).send(err);
        }
    })
})

```

```

        res.status(200).send(createdAttendance);
    })
})

// Clock out a user
// Update clockOutTimestamp & totalClockedHours for an attendance
router.route('/attendance/clockOut/:id').put(function (req, res) {
    // Find the attendance document where user._id = 'xxxxx' & clockedOutTimestamp
    == null
    Attendance
    .find(
    {
        clockOutTimestamp: null,
        "user": new ObjectId(req.params.id)
    }, (err, attendances) => {
        if (err) {
            console.log(err)
        } else {
            console.log(attendances);

            // Update clockOutTimestamp attribute
            attendances[0].clockOutTimestamp = req.body.clockOutTimestamp;

            // Calculate number of clocked hours
            var clockInTime = moment(new Date(attendances[0].clockInTimestamp));
            var clockOutTime = moment(new
            Date(attendances[0].clockOutTimestamp));

            console.log("clockInTime: " + clockInTime);
            console.log("clockOutTime: " + clockOutTime);

            var duration = moment.duration(clockOutTime.diff(clockInTime));
            var hours = duration.asMilliseconds();
            console.log("hours: " + hours)

            // Update totalClockedHours attribute
            attendances[0].totalClockedHours = hours;

            // Save the updated document back to the database
            attendances[0].save((err, updatedAttendance) => {
                if (err) {
                    res.status(500).send(err);
                }
                res.status(200).send(updatedAttendance);
            })
        }
    })
})
})
})

```

```
module.exports = router;
```

In back_end/routes, create a new file called light.js

light.js

```
var express = require('express');
var router = express.Router();
var Light = require('../models/Light')

router.route('/lights').get(function (req, res) {
  // Get an aggregate total count of each colours
  Light.aggregate(
    // Grouping pipeline
    {
      $group:
      { _id: '$Colour', total_count: { $sum: 1 } }
    },
    // Sorting pipeline (sort by colour name alphabetically)
    { "$sort": { _id: 1 } },
    function (err, data) {
      if (err) return res.send(err);
      res.status(200).send(data);
    }
  );
})

module.exports = router;
```

That's it! You have completed set up for routes. 😊

Section 9

Creating the SOMS web application (Frontend)

In front_end, create a new file called index.html

index.html

```
<!doctype HTML>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta name="description" content="IoT CA1 Assignment">
  <meta name="author" content="Sherna Liew">
  <title>SOMS</title>
  <!-- CSS -->
  <link href="node_modules/bootstrap/dist/css/bootstrap.min.css" rel="stylesheet">
  <link href="colours.css" rel="stylesheet">
  <link href="starter-template.css" rel="stylesheet">
  <link href="node_modules/font-awesome/css/font-awesome.min.css"
rel="stylesheet">
  <link href="node_modules/bootstrap-toggle/css/bootstrap-toggle.css"
rel="stylesheet">
  <link href="node_modules/sweetalert2/dist/sweetalert2.min.css" rel="stylesheet">
  <!-- Javascript libraries -->
  <script src="node_modules/jquery/dist/jquery.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.13.0/umd/popper.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/tether/1.4.0/js/tether.min.js"
integrity="sha384-DztdAPBWPRXSA/3eYEEUWrWCy7G5KFbe8fFjk5JAIxUYHKkDx6Qin1DkWx51bBrb"
crossorigin="anonymous"></script>
  <script src="node_modules/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
  <script src="node_modules/bootstrap/dist/js/bootstrap.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.20.1/moment.min.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/moment-duration-
format/2.1.0/moment-duration-format.min.js"></script>
  <script src="node_modules/chart.js/dist/Chart.min.js"></script>
  <script src="node_modules/bootstrap-toggle/js/bootstrap-toggle.min.js"></script>
  <script src="node_modules/sweetalert2/dist/sweetalert2.all.min.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/core-
js/2.4.1/core.js"></script>
  <script src="/socket.io/socket.io.js"></script>
```



```

    <script src="node_modules/whatwg-fetch/fetch.js"></script>
</head>

<body>
    <nav class="navbar navbar-expand-md navbar-dark bg-dark fixed-top">
        <a class="navbar-brand" href="index.html">SOMS</a>
        <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarsExampleDefault" aria-controls="navbarsExampleDefault"
        aria-expanded="false" aria-label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
        </button>

        <div class="collapse navbar-collapse" id="navbarsExampleDefault">
            <ul class="navbar-nav mr-auto">
                <li class="nav-item active">
                    <a class="nav-link" href="index.html">
                        <i class="fa fa-home" aria-hidden="true"></i>
                        <span class="sr-only">(current)</span>
                    </a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="user_management.html">Users</a>
                </li>
                <li class="nav-item dropdown">
                    <a class="nav-link dropdown-toggle" id="dropdown01" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">Attendance</a>
                    <div class="dropdown-menu" aria-labelledby="dropdown01">
                        <a class="dropdown-item"
href="take_attendance.html">Attendance Taking</a>
                        <a class="dropdown-item" href="attendance.html">Attendance
Log</a>
                    </div>
                </li>
                <li class="nav-item dropdown">
                    <a class="nav-link dropdown-toggle" id="dropdown01" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">Office</a>
                    <div class="dropdown-menu" aria-labelledby="dropdown01">
                        <a class="dropdown-item" href="lights.html">Lights</a>
                        <a class="dropdown-item" href="cctv.html">CCTV</a>
                    </div>
                </li>
            </ul>
        </div>
    </nav>

    <main role="main" class="container">
        <h1>
            <i class="fa fa-tachometer" aria-hidden="true"></i>&nbsp;&nbsp;&nbsp;Dashboard</h1>

```

```

    <hr />
  </main>

  <div class="container">
    <div class="row">
      <div class="col-sm-3">
        <div class="card text-white bg-purple" style="min-height: 8em">
          <div class="card-body pb-0">
            <div class="d-flex justify-content-between">
              <div>
                <h4 id="registeredUsersCount">0</h4>
                <p>Registered Users</p>
              </div>
              <div>
                <i class="fa fa-user-circle-o fa-3x" aria-
hidden="true"></i>
              </div>
            </div>
          </div>
        </div>
      </div>
      <div class="col-sm-3">
        <div class="card text-white bg-teal" style="min-height: 8em">
          <div class="card-body pb-0">
            <div class="d-flex justify-content-between">
              <div>
                <h4 id="clockedInUsersCount">0</h4>
                <p>Users clocked in today</p>
              </div>
              <div>
                <i class="fa fa-sign-in fa-3x" aria-
hidden="true"></i>
              </div>
            </div>
          </div>
        </div>
      </div>
      <div class="col-sm-3">
        <div class="card text-white bg-orange" style="min-height: 8em">
          <div class="card-body pb-0">
            <div class="d-flex justify-content-between">
              <div>
                <h4 id="clockedOutUsersCount">0</h4>
                <p>Users clocked out today</p>
              </div>
              <div>

```

```

                <i class="fa fa-sign-out fa-3x" aria-
hidden="true"></i>
            </div>
        </div>
    </div>
</div>
</div>

<div class="col-sm-3">
    <div class="card text-white bg-pink" style="min-height: 8em">
        <div class="card-body pb-0">
            <div class="d-flex justify-content-between">
                <div>
                    <h4 id="notClockedInUsersCount">0</h4>
                    <p>Users not clocked in today</p>
                </div>
                <div>
                    <i class="fa fa-times-circle-o fa-3x" aria-
hidden="true"></i>
                </div>
            </div>
        </div>
    </div>
</div>
<br />

<!-- User's clocked in time -->
<div class="card border-purple text-center">
    <div class="card-header text-white bg-purple">
        <i class="fa fa-calendar" aria-hidden="true"></i>&nbsp;Total No. of
Clocked Hours (Today)
    </div>
    <div class="card-block">
        <br />
        <canvas id="clockedHoursChart"></canvas>
        <br />
    </div>
    <div class="card-footer border-dark text-muted">
        <button id="refreshClockedHoursChart" type="button" class="btn btn-
outline-purple">
            <i class="fa fa-refresh" aria-hidden="true"></i>&nbsp;Refresh
Clocked Hours Chart</button>
        </div>
</div>
<br />
<br />

```

```

        <hr />
        <br />
    </div>
    <!-- End of container -->

    <footer class="footer bg-dark text-center">
        <div class="container">
            <br />
            <span class="text-muted">&copy; 2017 - 2018 Liew Zhi Li (Sherna). All
Rights Reserved.</span>
            <br />
            <br />
        </div>
    </footer>
    <script>
        var ipAddress;
        var $colourOptions = $("#colourOptions");
        var $lightToggle = $("#lightToggle");
        var lightPieChart;
        var $totalCount = $("#totalCount");
        var $currentState = $("#currentState");

        // Init Socket.IO to make connection to the Socket.IO server at the same URL
the page is being hosted on
        var socket = io()

        // Register Socket.IO event when rfid is emitted
        socket.on('ip', function (ip) {
            ipAddress = ip;
        });

        $((() => {
            // Call init methods on document ready
            initRegisteredUsersCount();
            initClockedHoursChart();

            // Refresh Attendance Chart Button onclick handler
            $("#refreshClockedHoursChart").click(function () {
                initClockedHoursChart();
                swal("Refresh", "Clocked Hours chart refreshed!", "success");
            });
        }

        // Function to initialize the registered users count
        function initRegisteredUsersCount() {
            console.log("initRegisteredUsersCount")

            fetch('api/usersCount')

```

```
.then((response) => {
    return response.json();
})
.then((responseData) => {
    $('#registeredUsersCount').text(responseData.count);
}).then(() => {
    initClockedInUsersCount();
})
}

// Function to initialize the clocked in users count
function initClockedInUsersCount() {
    console.log("initClockedInUsersCount")

    fetch('api/attendance/clockInCount')
        .then((response) => {
            return response.json();
        })
        .then((responseData) => {
            $('#clockedInUsersCount').text(responseData.count);
        })
        .then(() => {
            initClockedOutUsersCount();
        })
    }

// Function to initialize the clock out users count
function initClockedOutUsersCount() {
    console.log("initClockedOutUsersCount")

    fetch('api/attendance/clockOutCount')
        .then((response) => {
            return response.json();
        })
        .then((responseData) => {
            $('#clockedOutUsersCount').text(responseData.count);
        })
        .then(() => {
            initNotClockedInUsersCount();
        })
    }

// Function to initialize the not clocked in users count
function initNotClockedInUsersCount() {
    console.log("initNotClockedInUsersCount")

    var registeredUsers = $('#registeredUsersCount').text();
    var clockedInUsers = $('#clockedInUsersCount').text();
}
```

```

    var notClockedInUsers;

    notClockedInUsers = registeredUsers - clockedInUsers;
    console.log(notClockedInUsers)
    $('#notClockedInUsersCount').text(notClockedInUsers);
}

// Function to initialize the clocked hours chart
function initClockedHoursChart() {
    // Get Attendance data
    fetch('api/attendance/searchByDate/' + new Date())
        .then((response) => {
            return response.json();
        })
        .then((responseData) => {
            // console.log(responseData)
            drawClockedHoursChart(responseData);
        })
}

// Function to draw the clocked hours chart based on JSON data
function drawClockedHoursChart(data) {
    var dataArr = data;
    var users = [];
    var totalClockedHours = [];

    // Populate the data in each array
    for (var i in dataArr) {
        var duration =
moment.duration(data[i].totalClockedHours).format('HH');
        totalClockedHours.push(duration)

        // Error handling for when a user has been deleted
        if (!dataArr[i].user) {
            users.push('[Deleted User]');
        } else {
            users.push(dataArr[i].user.fullName);
        }
    }

    var canvas = document.getElementById('clockedHoursChart');
    var data = {
        labels: users,
        datasets: [
            {
                label: "Total No. of Clocked Hours (Today)",
                backgroundColor: "rgba(111, 66, 193, 0.2)",
                borderColor: "rgba(111, 66, 193, 1)",
            }
        ]
    }
}

```

```
        borderWidth: 2,
        hoverBackgroundColor: "rgba(111, 66, 193, 0.4)",
        hoverBorderColor: "rgba(111, 66, 193, 1)",
        data: totalClockedHours,
    }
]
};
var option = {
    scales: {
        yAxes: [{
            stacked: true,
            gridLines: {
                display: true,
                color: "rgba(255,99,132,0.2)"
            }
        }],
        xAxes: [{
            gridLines: {
                display: false
            }
        }]
    }
};

var totalClockedHoursChart = Chart.Bar(canvas, {
    data: data,
    options: option
});
}
</script>
</body>
</html>
```

In front_end, create a new file called user_management.html

user_management.html

```
<!doctype HTML>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta name="description" content="IoT CA1 Assignment">
  <meta name="author" content="Sherna Liew">
  <title>SOMS</title>
  <!-- CSS -->
  <link href="node_modules/bootstrap/dist/css/bootstrap.min.css" rel="stylesheet">
  <link href="starter-template.css" rel="stylesheet">
  <link href="node_modules/font-awesome/css/font-awesome.min.css"
rel="stylesheet">
  <link href="node_modules/bootstrap-toggle/css/bootstrap-toggle.css"
rel="stylesheet">
  <link href="https://cdn.datatables.net/1.10.16/css/jquery.dataTables.min.css"
rel="stylesheet">
  <link href="node_modules/sweetalert2/dist/sweetalert2.min.css" rel="stylesheet">
  <!-- Javascript libraries -->
  <script src="node_modules/jquery/dist/jquery.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.13.0/umd/popper.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/tether/1.4.0/js/tether.min.js"
integrity="sha384-DztdAPBWPRXSA/3eYEEUWrWCy7G5KFbe8fFjk5JAIxUYHKkDx6Qin1DkWx51bBrb"
crossorigin="anonymous"></script>
  <script src="node_modules/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
  <script src="node_modules/bootstrap/dist/js/bootstrap.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.20.1/moment.min.js"></script>
  <script src="node_modules/bootstrap-toggle/js/bootstrap-toggle.min.js"></script>
  <script src="node_modules/sweetalert2/dist/sweetalert2.all.min.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/core-
js/2.4.1/core.js"></script>
  <script
src="https://cdn.datatables.net/1.10.16/js/jquery.dataTables.min.js"></script>
  <script src="/socket.io/socket.io.js"></script>
</head>

<body>
  <nav class="navbar navbar-expand-md navbar-dark bg-dark fixed-top">
```



```

    <a class="navbar-brand" href="index.html">SOMS</a>
    <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarsExampleDefault" aria-controls="navbarsExampleDefault"
    aria-expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
</button>

<div class="collapse navbar-collapse" id="navbarsExampleDefault">
  <ul class="navbar-nav mr-auto">
    <li class="nav-item active">
      <a class="nav-link" href="index.html">
        <i class="fa fa-home" aria-hidden="true"></i>
      </a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="user_management.html">Users</a>
    </li>
    <li class="nav-item dropdown">
      <a class="nav-link dropdown-toggle" id="dropdown01" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">Attendance</a>
      <div class="dropdown-menu" aria-labelledby="dropdown01">
        <a class="dropdown-item"
href="take_attendance.html">Attendance Taking</a>
        <a class="dropdown-item" href="attendance.html">Attendance
Log</a>
      </div>
    </li>
    <li class="nav-item dropdown">
      <a class="nav-link dropdown-toggle" id="dropdown01" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">Office</a>
      <div class="dropdown-menu" aria-labelledby="dropdown01">
        <a class="dropdown-item" href="lights.html">Lights</a>
        <a class="dropdown-item" href="cctv.html">CCTV</a>
      </div>
    </li>
  </ul>
</div>
</nav>

<main role="main" class="container">
  <h1 class="text-info">
    <i class="fa fa-user-circle text-info" aria-
hidden="true"></i>&nbsp;Users</h1>
  <hr />
</main>

<div class="container">
  <!-- Users -->

```

```

        <div class="card border-info">
            <div class="card-header text-white bg-info">
                <div class="form-group">
                    <label for="searchInput">
                        <i class="fa fa-search" aria-
hidden="true"></i>&nbsp;Search</label>
                    <input type="text" name="searchInput" class="form-control"
id="searchInput" placeholder="Any Keyword">
                </div>
            </div>
            <div class="card-block">
                <br />
                <div class="col-sm-12 col-lg-12">
                    <table id="userTable" class="table table-bordered table-hover">
                        <thead>
                            <tr>
                                <th>Full Name</th>
                                <th>Username</th>
                                <th>Email</th>
                                <th>NRIC</th>
                                <th>NFC Mifare Card UID</th>
                                <th>Actions</th>
                            </tr>
                        </thead>
                    </table>
                </div>
                <br />
            </div>
            <div class="card-footer border-info text-muted text-center">
                <button id="refreshUsers" type="button" class="btn btn-outline-
info">
                    <i class="fa fa-refresh" aria-
hidden="true"></i>&nbsp;Refresh</button>
            </div>
        </div>
        <br>
    </div>
    <!-- End of container -->

    <div class="container">
        <!-- Add New User -->
        <div class="row">
            <div class="col-sm-6">
                <div class="card border-info">
                    <div class="card-header text-white bg-info">
                        <i class="fa fa-user-plus" aria-hidden="true"></i>&nbsp;Add
New User

```

```

        </div>
        <div class="card-block">
            <br />
            <!-- Add User form -->
            <div class="col-md-12">
                <form id="addUserForm" class="form-horizontal">
                    <div class="form-group">
                        <label for="fullNameAdd">Full Name</label>
                        <input type="text" class="form-control"
id="fullNameAdd" placeholder="Enter Full Name" pattern="^[a-zA-Z\s]*$"
title="Alphabetical characters only."
                            required>
                    </div>
                    <div class="form-group">
                        <label for="userNameAdd">Username</label>
                        <input type="text" class="form-control"
id="userNameAdd" placeholder="Enter Username" pattern=".{5,20}" required title="5 to
20 characters only."
                            required>
                    </div>
                    <div class="form-group">
                        <label for="emailAdd">Email address</label>
                        <input type="email" class="form-control"
id="emailAdd" placeholder="Enter Email" required>
                    </div>
                    <div class="form-group">
                        <label for="nricAdd">NRIC</label>
                        <input type="text" class="form-control"
id="nricAdd" placeholder="Enter NRIC" pattern="^[STFG]\d{7}[A-Z]$" title="First
letter can be S/F/T/G. Followed by 7 digits. Ends with an alphabet from A-Z. All
letters are in uppercase."
                            required>
                    </div>
                </div>
            </div>
            <br />
            <div class="card-footer border-info text-muted text-center">
                <button id="addUserBtn" type="button" class="btn btn-
outline-info">
                    <i class="fa fa-user-plus" aria-
hidden="true"></i>&nbsp;Add New User</button>
            </div>
        </form>
    </div>
</div>
<br />
<br />

```

```

    <!-- Edit Existing User -->
    <div class="col-sm-6">
        <div class="card border-info">
            <div class="card-header text-white bg-info">
                <i class="fa fa-pencil" aria-hidden="true"></i>&nbsp;Edit
Existing User
            </div>
            <div class="card-block">
                <br />
                <!-- Edit User form -->
                <div class="col-md-12">
                    <form id="addUserForm" class="form-horizontal">
                        <div class="form-group" style="display:none">
                            <label for="idUpdate">Id</label>
                            <input type="text" class="form-control"
id="idUpdate">
                        </div>
                        <div class="form-group">
                            <label for="fullNameUpdate">Full Name</label>
                            <input type="text" class="form-control"
id="fullNameUpdate" placeholder="Enter Full Name" pattern="^[a-zA-Z\s]*$"
title="Alphbetical characters only."
                                required disabled>
                        </div>
                        <div class="form-group">
                            <label for="userNameUpdate">Username</label>
                            <input type="text" class="form-control"
id="userNameUpdate" placeholder="Enter Username" pattern=".{5,20}" required title="5
to 20 characters only."
                                required disabled>
                        </div>
                        <div class="form-group">
                            <label for="emailUpdate">Email address</label>
                            <input type="email" class="form-control"
id="emailUpdate" placeholder="Enter Email" required disabled>
                        </div>
                        <div class="form-group">
                            <label for="nricUpdate">NRIC</label>
                            <input type="text" class="form-control"
id="nricUpdate" placeholder="Enter NRIC" pattern="^[STFG]\d{7}[A-Z]$" title="First
letter can be S/F/T/G. Followed by 7 digits. Ends with an alphabet from A-Z. All
letters are in uppercase."
                                required disabled>
                        </div>
                    </div>
                </div>
            </div>
            <br />
            <div class="card-footer border-info text-muted text-center">

```

```

        <button id="cancelUpdateBtn" type="button" class="btn btn-
outline-info" disabled>
            <i class="fa fa-ban" aria-
hidden="true"></i>&nbsp;Cancel</button>
        <button id="updateUserBtn" type="button" class="btn btn-
outline-info" disabled>
            <i class="fa fa-floppy-o" aria-
hidden="true"></i>&nbsp;Save</button>
    </div>
</form>
</div>
</div>
<br />
<hr />
<br />

<!-- Enroll NFC Modal -->
<div class="modal fade" id="enrollNfcModal" tabindex="-1" role="dialog"
aria-labelledby="exampleModalLabel" aria-hidden="true">
    <div class="modal-dialog" role="document">
        <!-- Modal content-->
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title" id="exampleModalLabel">Enroll NFC
Mifare Card</h5>
            </div>
            <div class="modal-body text-center">
                
                <br />
                <br />
                <input id="capturedUserId" type="text"
style="display:none;">
                <input id="capturedNfcCardId" type="text"
style="display:none;">
                <p id="modalMessage">Please place your NFC Mifare Card onto
the RFID reader.</p>
            </div>
            <div class="modal-footer">
                <button type="button" class="btn btn-danger" data-
dismiss="modal">
                    <i class="fa fa-ban" aria-
hidden="true"></i>&nbsp;Cancel</button>
                <button id="saveNfcCardForUser" type="button" class="btn
btn-primary" data-dismiss="modal">
                    <i class="fa fa-address-card-o" aria-
hidden="true"></i>&nbsp;Enroll Card</button>
            </div>

```

```

        </div>
    </div>
</div>
<!-- End of container -->

<footer class="footer bg-dark text-center">
    <div class="container">
        <br />
        <span class="text-muted">&copy; 2017 - 2018 Liew Zhi Li (Sherna). All
Rights Reserved.</span>
        <br />
        <br />
    </div>
</footer>
<script>
    // Init Socket.IO to make connection to the Socket.IO server at the same URL
the page is being hosted on
    var socket = io()

    // Init variables
    var table;
    var $searchInput = $('#searchInput');
    var $userTable = $('#userTable');
    var $addUserBtn = $('#addUserBtn');
    var $updateUserBtn = $('#updateUserBtn');
    var $cancelUpdateBtn = $('#cancelUpdateBtn');
    var $refreshUsers = $('#refreshUsers');
    var $saveNfcCardForUser = $('#saveNfcCardForUser');

    // Register Socket.IO event when rfid is emitted
    socket.on('rfid', function (data) {
        console.log("NFC Serial number: " + data);
        $('#capturedNfcCardId').val(data);
        $('#modalMessage').text("NFC Mifare Card ID: " + data);
    });

    // Datatables's Custom Renders are defined here
    var CustomRenderers = {
        username: function (data, type, row, meta) {
            return '<span class="badge badge-pill badge-info" style="padding:
10px">' + data + '</span>';
        },
        nfcCard: function (data, type, row, meta) {
            if (data == '') {
                return '<span class="badge badge-pill badge-secondary"
style="padding: 10px">Not Registered</span>';
            } else {

```

```

        return '<span class="badge badge-pill badge-info"
style="padding: 10px">' + data + '</span>';
    },
    actionButtons: function (data, type, row, meta) {
        return '<div class="btn-group" role="group">'
            + '<button type="button" onclick="enrollNFC(\'' + data + '\')"'
class="btn btn-primary btn-md" data-toggle="tooltip" data-placement="top"
title="Enroll NFC Mifare Card"><i class="fa fa-address-card-o" aria-
hidden="true"></i></button>'
            + '<button type="button" onclick="prepUpdate(\'' + data + '\')"'
class="btn btn-warning btn-md" data-toggle="tooltip" data-placement="top"
title="Edit User"><i class="fa fa-pencil" aria-hidden="true"></i></button>'
            + '<button type="button" onclick="deleteUser(\'' + data + '\')"'
class="btn btn-danger btn-md" data-toggle="tooltip" data-placement="top"
title="Delete User"><i class="fa fa-trash-o" aria-hidden="true"></i></button>'
            + '</div>';
    }
};

$(() => {
    // Call init methods on document ready
    table = initUserDataTable();

    // Onclick event handler function for $refreshUsers
    $refreshUsers.click(function () {
        table.ajax.reload();
        swal("Refresh", "Users table refreshed!", "success");
    })

    // Onclick event handler function for $addUserBtn
    $addUserBtn.click(function () {
        // Get the User details as object
        var user = {
            fullName: $('#fullNameAdd').val(),
            username: $('#userNameAdd').val(),
            email: $('#emailAdd').val(),
            nric: $('#nricAdd').val(),
            nfcCard: ""
        }

        // AJAX call to make a POST request to api/users
        $.ajax({
            url: 'api/users',
            type: 'POST',
            data: JSON.stringify(user),
            contentType: "application/json",
            success: function (data) {

```

```

        $('#addUserForm')[0].reset();

        // reload the all users table again
        table.ajax.reload();
        swal("User", "Successfully added new user!", "success");
    }
});

// Onclick event handler function for $updateUserBtn
$updateUserBtn.click(function () {
    var id = $('#idUpdate').val()
    // Get the User details as object
    var user = {
        fullName: $('#fullNameUpdate').val(),
        username: $('#userNameUpdate').val(),
        email: $('#emailUpdate').val(),
        nric: $('#nricUpdate').val()
    }

    // AJAX call to make a PUT request api/users/<id>
    $.ajax({
        url: 'api/users/' + id,
        type: 'PUT',
        data: JSON.stringify(user),
        contentType: "application/json",
        success: function (data) {
            $('#form')[1].reset();
            $('#fullNameUpdate').prop('disabled', true);
            $('#userNameUpdate').prop('disabled', true);
            $('#emailUpdate').prop('disabled', true);
            $('#nricUpdate').prop('disabled', true);
            $('#cancelUpdateBtn').prop('disabled', true);
            $('#updateUserBtn').prop('disabled', true);

            // reload the all users table again
            table.ajax.reload();
            swal("User", "Successfully updated the user!", "success");
        }
    });
});

// Onclick event handler function for $cancelUpdateBtn
$cancelUpdateBtn.click(function () {
    $('#form')[1].reset();
    $('#fullNameUpdate').prop('disabled', true);
    $('#userNameUpdate').prop('disabled', true);
    $('#emailUpdate').prop('disabled', true);

```



```

        $('#nricUpdate').prop('disabled', true);
        $('#cancelUpdateBtn').prop('disabled', true);
        $('#updateUserBtn').prop('disabled', true);
        swal("User", "User update cancelled", "info");
    })

    // Onclick event handler function for $saveNfcCardForUser
    $saveNfcCardForUser.click(function () {
        var id = document.getElementById('capturedUserId').value;
        var nfcCardId = document.getElementById('capturedNfcCardId').value;

        var item = {
            nfcCard: nfcCardId
        }

        // AJAX call to make a PUT request to api/users/updateNfcCard/<id>
        $.ajax({
            url: 'api/users/updateNfcCard/' + id,
            type: 'PUT',
            data: JSON.stringify(item),
            contentType: "application/json",
            success: function (data) {
                // reload the all users table again
                table.ajax.reload();
                swal("User", "Successfully saved the NFC Mifare UID for the
user!", "success");
            }
        });
    })

    }) // end of document.ready

    // Function to initialize the user datatable
    function initUserDataTable() {
        // Define the data to render for each column and the render style
        var columns = [
            { "data": "fullName" },
            { "data": "username", "render": CustomRenderers.username },
            { "data": "email" },
            { "data": "nric" },
            { "data": "nfcCard", "render": CustomRenderers.nfcCard },
            { "data": "_id", "render": CustomRenderers.actionButtons }
        ];

        // Define column width
        var columnDefs = [
            { "width": "25%", "targets": 4 },
            { "width": "13%", "targets": 5 }

```

```

];

// Define datatables options
var options = {
  "processing": false,
  "autoWidth": true,
  "serverSide": false,
  "searching": true,
  "columns": columns,
  "columnDefs": columnDefs,
  "ajax": { "url": "api/users", "dataSrc": "" },
  "dom": "<'row'<'col-sm-6'l>>" + "<'row'<'col-sm-12'tr>>" +
    "<'row'<'col-sm-5'i><'col-sm-7'p>>",
  "order": [
    [0, "asc"]
  ]
};

// Initialize datatables
var dataTableApi = $userTable.DataTable(options);

// On key up event handler for $searchInput
$searchInput.keyup(function () {
  dataTableApi.search(this.value).draw();
});

// Trigger the tooltips when table is redrawn
$userTable.on('draw.dt', function () {
  $('[data-toggle="tooltip"]').tooltip({
    container: 'body'
  });
});

return dataTableApi;
}

// Function to prepare the Update form
function prepUpdate(id) {
  console.log("id is: " + id)

  // AJAX call to make a GET request to api/users/<id>
  $.ajax({
    url: 'api/users/' + id,
    type: 'GET',
    success: function (data) {
      // Populate values
      $('#idUpdate').val(data[0]._id);
      $('#fullNameUpdate').val(data[0].fullName);
    }
  });
}

```

```

        $('#userNameUpdate').val(data[0].username);
        $('#emailUpdate').val(data[0].email);
        $('#nricUpdate').val(data[0].nric);

        // Enable inputs
        $('#fullNameUpdate').prop('disabled', false);
        $('#userNameUpdate').prop('disabled', false);
        $('#emailUpdate').prop('disabled', false);
        $('#nricUpdate').prop('disabled', false);
        $('#cancelUpdateBtn').prop('disabled', false);
        $('#updateUserBtn').prop('disabled', false);
    }
    });
}

// Function to delete a specified user
function deleteUser(id) {
    swal({
        title: "Are you sure?",
        text: "Once its deleted, it will be gone forever! :(",
        icon: "warning",
        buttons: true,
        dangerMode: true,
    })
        .then((willDelete) => {
            if (willDelete) {
                $.ajax({
                    type: 'DELETE',
                    url: "api/users/" + id,
                }).done(function () {
                    // reload the all user table again
                    table.ajax.reload();
                })

                swal("Success", "Poof! The user has been deleted!",
"success")
            }
        });
}

// Function to enroll NFC card
function enrollNFC(id) {
    // Reset the text & value first
    $("#capturedNfcCardId").val("");
    $("#modalMessage").text("Please place your NFC Mifare Card onto the RFID
reader.")

    // Set the capturedUserId

```

```
        $("#capturedUserId").val(id);  
        $("#enrollNfcModal").modal('show');  
    }  
    </script>  
</body>  
</html>
```

In front_end, create a new file called take_attendance.html

take_attendance.html

```
<!doctype HTML>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta name="description" content="IoT CA1 Assignment">
  <meta name="author" content="Sherna Liew">
  <title>SOMS</title>
  <!-- CSS -->
  <link href="node_modules/bootstrap/dist/css/bootstrap.min.css" rel="stylesheet">
  <link href="starter-template.css" rel="stylesheet">
  <link href="node_modules/font-awesome/css/font-awesome.min.css"
rel="stylesheet">
  <link href="node_modules/bootstrap-toggle/css/bootstrap-toggle.css"
rel="stylesheet">
  <link href="https://cdn.datatables.net/1.10.16/css/jquery.dataTables.min.css"
rel="stylesheet">
  <link href="node_modules/sweetalert2/dist/sweetalert2.min.css" rel="stylesheet">
  <!-- Javascript libraries -->
  <script src="node_modules/jquery/dist/jquery.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.13.0/umd/popper.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/tether/1.4.0/js/tether.min.js"
integrity="sha384-DztdAPBWPRXSA/3eYEEUWrWCy7G5KFbe8fFjk5JAIxUYHKkDx6Qin1DkWx51bBrb"
crossorigin="anonymous"></script>
  <script src="node_modules/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
  <script src="node_modules/bootstrap/dist/js/bootstrap.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.20.1/moment.min.js"></script>
  <script src="node_modules/bootstrap-toggle/js/bootstrap-toggle.min.js"></script>
  <script src="node_modules/sweetalert2/dist/sweetalert2.all.min.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/core-
js/2.4.1/core.js"></script>
  <script
src="https://cdn.datatables.net/1.10.16/js/jquery.dataTables.min.js"></script>
  <script src="/socket.io/socket.io.js"></script>
  <script src="node_modules/whatwg-fetch/fetch.js"></script>
</head>

<body>
```

```

<nav class="navbar navbar-expand-md navbar-dark bg-dark fixed-top">
  <a class="navbar-brand" href="/">SOMS</a>
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarsExampleDefault" aria-controls="navbarsExampleDefault"
  aria-expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>

  <div class="collapse navbar-collapse" id="navbarsExampleDefault">
    <ul class="navbar-nav mr-auto">
      <li class="nav-item active">
        <a class="nav-link" href="/">
          <i class="fa fa-home" aria-hidden="true"></i>
          <span class="sr-only">(current)</span>
        </a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="user_management.html">Users</a>
      </li>
      <li class="nav-item dropdown">
        <a class="nav-link dropdown-toggle" id="dropdown01" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">Attendance</a>
        <div class="dropdown-menu" aria-labelledby="dropdown01">
          <a class="dropdown-item"
href="take_attendance.html">Attendance Taking</a>
          <a class="dropdown-item" href="attendance.html">Attendance
Log</a>
        </div>
      </li>
      <li class="nav-item dropdown">
        <a class="nav-link dropdown-toggle" id="dropdown01" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">Office</a>
        <div class="dropdown-menu" aria-labelledby="dropdown01">
          <a class="dropdown-item" href="lights.html">Lights</a>
          <a class="dropdown-item" href="cctv.html">CCTV</a>
        </div>
      </li>
    </ul>
  </div>
</nav>

<main role="main" class="container">
  <h1 class="text-primary">
    <i class="fa fa-check-circle-o" aria-hidden="true"></i>&nbsp;Attendance
Taking</h1>
  <hr />
</main>

```

```
<div class="container">  
    <!-- Attendance Taking -->  
    <div class="card border-primary text-center">  
        <div class="card-header text-white bg-primary">  
            <i class="fa fa-check-circle-o" aria-  
hidden="true"></i>&nbsp; Attendance Taking  
        </div>  
        <div class="card-block">  
            <br />  
              
            <input id="userId" type="text" style="display:none">  
            <br />  
            <br />  
            <div class="btn-group" data-toggle="buttons">  
                <label class="btn btn-outline-primary active">  
                    <input type="radio" name="options" id="clockIn"  
value="clockIn" checked> Clock In  
                </label>  
                <label class="btn btn-outline-primary">  
                    <input type="radio" name="options" id="clockOut"  
value="clockOut"> Clock Out  
                </label>  
            </div>  
            <br />  
            <p>Select either 'Clock In' or 'Clock Out' and place your NFC Mifare  
Card onto the RFID reader to take your  
attendance.  
            </p>  
            <br />  
        </div>  
        <div class="card-footer border-primary text-muted">  
        </div>  
    </div>  
    <br>  
    <br />  
    <br />  
    <br />  
    <br />  
    <br />  
    <br />  
    <br />  
    <br />  
    <br />  
    <br />  
    <br />  
    <br />  
    <br />  
    </div>  
    <!-- End of container -->  
    <footer class="footer bg-dark text-center">  
        <div class="container">  
            <br />
```

```

        <span class="text-muted">&copy; 2017 - 2018 Liew Zhi Li (Sherna). All
Rights Reserved.</span>
        <br />
        <br />
    </div>
</footer>
<script>
    // Init Socket.IO to make connection to the Socket.IO server at the same URL
the page is being hosted on
    var socket = io()

    // Register Socket.IO event when rfid is emitted
    socket.on('rfid', function (nfcCard) {
        console.log("NFC Serial number: " + nfcCard);

        // If 'Clock In' is checked, run clock in function
        if ($("#clockIn").is(':checked')) {

            // 1st layer
            // Fetch the user associated to the NFC Card ID
            fetch('api/users/getUserByNfcCard/' + nfcCard)
                .then((response) => {
                    return response.json();
                })
                .then((responseData) => {
                    var scannedUserId = responseData[0]._id;
                    $("#userId").val(scannedUserId) // store in a hidden input
cause i cant save it in fetches x.x
                    console.log("success. scannedUserId is: ") // get user id
                    console.log(scannedUserId)

                    console.log("call checkIfClockedIn")

                    // 2nd layer
                    // Check if user has clocked in today
                    fetch('api/attendance/checkIfClockedIn/' + scannedUserId)
                        .then((response) => {
                            return response.json();
                        })
                        .then((responseData) => {
                            var count = responseData.count
                            console.log("success. count is: ")
                            console.log(count)

                            if (count == 0) {
                                // 3rd layer yay
                                // Save today's attendance for the user
                                clockIn($("#userId").val())

```



```

        .then(() => {
            swal("Success", "User has successfully
clocked in!", "success");
        })
    } else {
        swal("Error", "User has clocked in today
already.", "error");
    }
})
.catch((error) => {
    console.log("idk what error.")
})

})
.catch((error) => {
    console.log("Could not find the user associated to the NFC
Mifare Card you just scanned.")
    swal("Error", "Could not find the user associated to the NFC
Mifare Card you just scanned.", "error");
})

} else {
    console.log("run clock out")

    // here we go again
    // 1st layer
    // Fetch the user associated to the NFC Card ID
    fetch('api/users/getUserByNfcCard/' + nfcCard)
        .then((response) => {
            return response.json();
        })
        .then((responseData) => {
            var scannedUserId = responseData[0]._id;
            $("#userId").val(scannedUserId) // store in a hidden input
cause i cant save it in fetches x.x
            console.log("success. scannedUserId is: ") // get user id
            console.log(scannedUserId)

            console.log("call checkIfClockedIn")

            // 2nd layer
            // Check if user has clocked in today
            fetch('api/attendance/checkIfClockedIn/' + scannedUserId)
                .then((response) => {
                    return response.json();
                })
        })
    }
}

```

```

        .then((responseData) => {
            var count = responseData.count
            console.log("success. count is: ")
            console.log(count)

            // must be strictly 1.
            if (count == 1) {
                // 3rd layer
                // Check if user has clockked out today
                fetch('api/attendance/checkIfClockedOut/' +
scannedUserId)

                    .then((response) => {
                        return response.json()
                    })
                    .then((responseData) => {
                        var clockOutCount = responseData.count;
                        console.log("checkIfClockedOut success.
clockOutCount is:")

                        console.log(responseData.count)

                        // 4th layer
                        // If yet to clock out, then clock out
the user.

                        if (clockOutCount == 1) {
                            clockOut($("#userId").val())
                                .then(() => {
                                    swal("Success", "User has
successfully clocked out!", "success");
                                })
                        } else {
                            swal("Error", "User has clocked out
today already.", "error");
                        }

                    })
                    .catch((error) => {
                        console.log("i dont know what happened")
                    })
            } else {
                swal("Error", "User has not clocked in today.",
"error");
            }
        })
        .catch((error) => {
            console.log("idk what error.")
        })
    }
}

```

```
        })
        .catch((error) => {
            console.log("Could not find the user associated to the NFC
Mifare Card you just scanned.")
            swal("Error", "Could not find the user associated to the NFC
Mifare Card you just scanned.", "error");
        })
    }
});

// Function to fetch the user associated to the NFC Card ID
function getUserByNfcCard(nfcCardId) {
    return $.ajax({
        url: 'api/users/getUserByNfcCard/' + nfcCardId,
        type: 'GET'
    })
}

// Function to check if the user has clocked in today
// User can only clock in once per day
function checkIfClockedIn() {
    return $.ajax({
        url: 'api/users/getUserByNfcCard/' + nfcCardId,
        type: 'GET'
    })
}

// Function to create clock in user aka create attendance record
function clockIn(id) {
    console.log("id is: " + id)
    // Get the Attendance details as object
    var attendance = {
        clockInTimestamp: new Date(),
        clockOutTimestamp: null,
        totalClockedHours: null,
        verificationImage: null,
        user: id
    }

    return $.ajax({
        url: 'api/attendance/clockIn',
        type: 'POST',
        data: JSON.stringify(attendance),
        contentType: "application/json",
    })
}

// Function to create clock out user aka update attendance record
```

```
function clockOut(id) {  
    console.log("id is: " + id)  
    // Get the Attendance details as object  
    var attendance = {  
        clockOutTimestamp: new Date(),  
    }  
  
    return $.ajax({  
        url: 'api/attendance/clockOut/' + id,  
        type: 'PUT',  
        data: JSON.stringify(attendance),  
        contentType: "application/json",  
    })  
}  
  
$(() => {  
  
    })  
</script>  
</body>  
</html>
```

In front_end, create a new file called attendance.html

attendance.html

```
<!doctype HTML>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta name="description" content="IoT CA1 Assignment">
  <meta name="author" content="Sherna Liew">
  <title>SOMS</title>
  <!-- CSS -->
  <link href="node_modules/bootstrap/dist/css/bootstrap.min.css" rel="stylesheet">
  <link href="starter-template.css" rel="stylesheet">
  <link href="node_modules/font-awesome/css/font-awesome.min.css"
rel="stylesheet">
  <link href="node_modules/bootstrap-toggle/css/bootstrap-toggle.css"
rel="stylesheet">
  <link href="https://cdn.datatables.net/1.10.16/css/jquery.dataTables.min.css"
rel="stylesheet">
  <link href="node_modules/sweetalert2/dist/sweetalert2.min.css" rel="stylesheet">
  <!-- Javascript libraries -->
  <script src="node_modules/jquery/dist/jquery.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.13.0/umd/popper.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/tether/1.4.0/js/tether.min.js"
integrity="sha384-DztdAPBWPRXSA/3eYEEUWrWCy7G5KFbe8fFjk5JAIxUYHKkDx6Qin1DkWx51bBrb"
crossorigin="anonymous"></script>
  <script src="node_modules/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
  <script src="node_modules/bootstrap/dist/js/bootstrap.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.20.1/moment.min.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/moment-duration-
format/2.1.0/moment-duration-format.min.js"></script>
  <script src="node_modules/bootstrap-toggle/js/bootstrap-toggle.min.js"></script>
  <script src="node_modules/sweetalert2/dist/sweetalert2.all.min.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/core-
js/2.4.1/core.js"></script>
  <script
src="https://cdn.datatables.net/1.10.16/js/jquery.dataTables.min.js"></script>
  <script src="/socket.io/socket.io.js"></script>
</head>
```

```

<body>
  <nav class="navbar navbar-expand-md navbar-dark bg-dark fixed-top">
    <a class="navbar-brand" href="/">SOMS</a>
    <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarsExampleDefault" aria-controls="navbarsExampleDefault"
    aria-expanded="false" aria-label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>
    </button>

    <div class="collapse navbar-collapse" id="navbarsExampleDefault">
      <ul class="navbar-nav mr-auto">
        <li class="nav-item active">
          <a class="nav-link" href="/">
            <i class="fa fa-home" aria-hidden="true"></i>
            <span class="sr-only">(current)</span>
          </a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="user_management.html">Users</a>
        </li>
        <li class="nav-item dropdown">
          <a class="nav-link dropdown-toggle" id="dropdown01" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">Attendance</a>
          <div class="dropdown-menu" aria-labelledby="dropdown01">
            <a class="dropdown-item"
href="take_attendance.html">Attendance Taking</a>
            <a class="dropdown-item" href="attendance.html">Attendance
Log</a>
          </div>
        </li>
        <li class="nav-item dropdown">
          <a class="nav-link dropdown-toggle" id="dropdown01" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">Office</a>
          <div class="dropdown-menu" aria-labelledby="dropdown01">
            <a class="dropdown-item" href="lights.html">Lights</a>
            <a class="dropdown-item" href="cctv.html">CCTV</a>
          </div>
        </li>
      </ul>
    </div>
  </nav>

  <main role="main" class="container">
    <h1 class="text-primary">
      <i class="fa fa-calendar text-primary" aria-
hidden="true"></i>&nbsp;&nbsp;&nbsp;Attendance Log</h1>
    <hr />
  </main>

```

```

<div class="container">
  <!-- Attendance -->
  <div class="card border-primary">
    <div class="card-header text-white bg-primary">
      <div class="form-group">
        <label for="searchInput">
          <i class="fa fa-search" aria-
hidden="true"></i>&nbsp;Search</label>
          <input type="text" name="searchInput" class="form-control"
id="searchInput" placeholder="Any Keyword">
        </div>
      </div>
      <div class="card-block">
        <br />
        <div class="col-sm-12 col-lg-12">
          <table id="attendanceTable" class="table table-bordered table-
hover">
            <thead>
              <tr>
                <th>Full Name</th>
                <th>Clock In Timestamp</th>
                <th>Clock Out Timestamp</th>
                <th>Total Hours Clocked</th>
              </tr>
            </thead>
          </table>
        </div>

        <br />
      </div>
      <div class="card-footer border-primary text-muted text-center">
        <button id="refreshUsers" type="button" class="btn btn-outline-
primary">Refresh</button>
      </div>
    </div>
    <br />
    <hr />
    <br />
  </div>
  <!-- End of container -->

  <footer class="footer bg-dark text-center">
    <div class="container">
      <br />
      <span class="text-muted">&copy; 2017 - 2018 Liew Zhi Li (Sherna). All
Rights Reserved.</span>
      <br />
    </div>
  </div>

```

```

        <br />
    </div>
</footer>
<script>
    var $searchInput = $('#searchInput');
    var $attendanceTable = $("#attendanceTable");

    // Datatables render
    var CustomRenderers = {
        user: function (data, type, row, meta) {
            if (data === null) {
                return "[Deleted User]"
            }
            return data.fullName;
        },
        clockInTimestamp: function (data, type, row, meta) {
            return moment(data).format("MMM DD, YYYY hh:mm:ss A")
        },
        clockOutTimestamp: function (data, type, row, meta) {
            if (data === null) {
                return 'Not clocked out yet';
            } else {
                return moment(data).format("MMM DD, YYYY hh:mm:ss A");
            }
        },
        totalClockedHours: function (data, type, row, meta) {
            if (data === null) {
                return 'Not calculated yet';
            } else {
                var duration = moment.duration(data);
                return duration.format('HH [hours] mm [mins]');
            }
        }
    };

    $((() => {
        var table = initAttendanceDataTable();

    }

    // Function to initialize the attendance datatable
    function initAttendanceDataTable() {
        // Define the data to render for each column and the render style
        var columns = [
            { "data": "user", "render": CustomRenderers.user },
            { "data": "clockInTimestamp", "render":
CustomRenderers.clockInTimestamp },

```



```

        { "data": "clockOutTimestamp", "render":
CustomRenderers.clockOutTimestamp },
        { "data": "totalClockedHours", "render":
CustomRenderers.totalClockedHours }
    ];

    // Define datatables options
    var options = {
        "processing": false,
        "autoWidth": true,
        "serverSide": false,
        "searching": true,
        "columns": columns,
        "ajax": { "url": "api/attendance", "dataSrc": "" },
        "dom": "<'row'<'col-sm-6'l>>" + "<'row'<'col-sm-12'tr>>" +
"<'row'<'col-sm-5'i><'col-sm-7'p>>",
        "order": [
            [0, "asc"]
        ]
    };

    var dataTableApi = $attendanceTable.DataTable(options);

    // On key up event handler for $searchInput
    $searchInput.keyup(function () {
        dataTableApi.search(this.value).draw();
    });

    // Trigger tooltips when table is redrawn
    $attendanceTable.on('draw.dt', function () {
        $('[data-toggle="tooltip"]').tooltip({
            container: 'body'
        });
    });

    return dataTableApi;
}
</script>
</body>
</html>

```

In front_end, create a new file called lights.html

lights.html

```
<!doctype HTML>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta name="description" content="IoT CA1 Assignment">
  <meta name="author" content="Sherna Liew">
  <title>SOMS</title>
  <!-- CSS -->
  <link href="node_modules/bootstrap/dist/css/bootstrap.min.css" rel="stylesheet">
  <link href="colours.css" rel="stylesheet">
  <link href="starter-template.css" rel="stylesheet">
  <link href="node_modules/font-awesome/css/font-awesome.min.css"
rel="stylesheet">
  <link href="node_modules/bootstrap-toggle/css/bootstrap-toggle.css"
rel="stylesheet">
  <link href="node_modules/sweetalert2/dist/sweetalert2.min.css" rel="stylesheet">
  <!-- Javascript libraries -->
  <script src="node_modules/jquery/dist/jquery.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.13.0/umd/popper.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/tether/1.4.0/js/tether.min.js"
integrity="sha384-DztdAPBWPRXSA/3eYEEUWrWCy7G5KFbe8fFjk5JAIxUYHKkDx6Qin1DkWx51bBrb"
crossorigin="anonymous"></script>
  <script src="node_modules/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
  <script src="node_modules/bootstrap/dist/js/bootstrap.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.20.1/moment.min.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/moment-duration-
format/2.1.0/moment-duration-format.min.js"></script>
  <script src="node_modules/chart.js/dist/Chart.min.js"></script>
  <script src="node_modules/bootstrap-toggle/js/bootstrap-toggle.min.js"></script>
  <script src="node_modules/sweetalert2/dist/sweetalert2.all.min.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/core-
js/2.4.1/core.js"></script>
  <script src="/socket.io/socket.io.js"></script>
  <script src="node_modules/whatwg-fetch/fetch.js"></script>
</head>

<body>
```

```

<nav class="navbar navbar-expand-md navbar-dark bg-dark fixed-top">
  <a class="navbar-brand" href="index.html">SOMS</a>
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarsExampleDefault" aria-controls="navbarsExampleDefault"
  aria-expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>

  <div class="collapse navbar-collapse" id="navbarsExampleDefault">
    <ul class="navbar-nav mr-auto">
      <li class="nav-item active">
        <a class="nav-link" href="index.html">
          <i class="fa fa-home" aria-hidden="true"></i>
          <span class="sr-only">(current)</span>
        </a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="user_management.html">Users</a>
      </li>
      <li class="nav-item dropdown">
        <a class="nav-link dropdown-toggle" id="dropdown01" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">Attendance</a>
        <div class="dropdown-menu" aria-labelledby="dropdown01">
          <a class="dropdown-item"
href="take_attendance.html">Attendance Taking</a>
          <a class="dropdown-item" href="attendance.html">Attendance
Log</a>
        </div>
      </li>
      <li class="nav-item dropdown">
        <a class="nav-link dropdown-toggle" id="dropdown01" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">Office</a>
        <div class="dropdown-menu" aria-labelledby="dropdown01">
          <a class="dropdown-item" href="lights.html">Lights</a>
          <a class="dropdown-item" href="cctv.html">CCTV</a>
        </div>
      </li>
    </ul>
  </div>
</nav>

<main role="main" class="container">
  <h1>
    <i class="fa fa-lightbulb-o" aria-hidden="true"></i>&nbsp;Lights</h1>
  <hr />
</main>

<div class="container">

```

```

<!-- Lights -->
<div class="row">
  <div class="col-sm-6">
    <div class="card border-danger text-center">
      <div class="card-header text-white bg-danger">
        <i class="fa fa-power-off" aria-hidden="true"></i>&nbsp;Turn
on different light colours!
      </div>
      <div class="card-block">
        <div id="currentState" class="alert alert-secondary"
role="alert">
          Current state: off
        </div>
        <br />
        <h1>
          <i class="fa fa-lightbulb-o" aria-hidden="true"></i>
        </h1>
        <!-- Select colour options -->
        <select class="custom-select" id="colourOptions">
          <option value="none" selected hidden>Select a
colour</option>
          <option value="red">Red</option>
          <option value="green">Green</option>
          <option value="blue">Blue</option>
          <option value="yellow">Yellow</option>
          <option value="magenta">Magenta</option>
          <option value="cyan">Cyan</option>
          <option value="white">White</option>
        </select>
      </div>
      <br />
      <div class="card-footer border-danger text-muted">
        <input id="lightToggle" type="checkbox" checked data-
toggle="toggle" data-onstyle="danger">
        <button id="blinkLedBtn" type="button" class="btn btn-
outline-danger">Blink LED</button>
      </div>
    </div>
  </div>
  <div class="col-sm-6">
    <div class="card border-danger text-center">
      <div class="card-header text-white bg-danger">
        <i class="fa fa-pie-chart" aria-hidden="true"></i>&nbsp;No.
of times each light colour is on
      </div>

```

```

        <div class="card-block">
            <div id="totalCount" class="alert alert-secondary"
role="alert">
                Total count: 5
            </div>
            <canvas id="lightPieChart" />
        </div>
        <br />
        <div class="card-footer border-danger text-muted">
            <button id="refreshLightChartBtn" type="button" class="btn
btn-outline-danger">
                <i class="fa fa-refresh" aria-
hidden="true"></i>&nbsp;Refresh Pie Chart</button>
            </div>
        </div>
    </div>
</div>
<br />
<br />
<br />
<br />
<br />
<br />
</div>
<!-- End of container -->

<footer class="footer bg-dark text-center">
    <div class="container">
        <br />
        <span class="text-muted">&copy; 2017 - 2018 Liew Zhi Li (Sherna). All
Rights Reserved.</span>
        <br />
        <br />
    </div>
</footer>
<script>
    var ipAddress;
    var $colourOptions = $("#colourOptions");
    var $lightToggle = $("#lightToggle");
    var lightPieChart;
    var $totalCount = $("#totalCount");
    var $currentState = $("#currentState");
    var $blinkLedBtn = $("#blinkLedBtn");

    // Init Socket.IO to make connection to the Socket.IO server at the same URL
the page is being hosted on
    var socket = io()

    // Register Socket.IO event when rfid is emitted

```

```

socket.on('ip', function (ip) {
  ipAddress = ip;
});

$(() => {
  // Initialize the light doughnut chart
  initLightPieChart();

  // Refresh Light Doughnut Chart Button onclick handler
  $("#refreshLightChartBtn").click(function () {
    // Destroy the lightPieChart instance before re-init
    lightPieChart.destroy();
    initLightPieChart();
    swal("Refresh", "Light chart refreshed!", "success");
  });

  // Onclick event handler function for $blinkLedBtn
  $blinkLedBtn.click(() => {
    fetch('http://' + ipAddress + ':8001/led/blink')
      .then((response) => {
        return response.json()
      })
      .then((responseData) => {
        // Append the 'blinking' to current state
        $currentState.append(' + ' + responseData.ledStatus)
        $blinkLedBtn.prop('disabled', true);
      })
  })

  // Toggle the lightToggle to off first
  $lightToggle.bootstrapToggle('off');

  // Hide the 'Blink LED' button first
  $blinkLedBtn.hide();

  // onChange event handler for lightToggle
  $lightToggle.on('change', function () {
    // console.log($(this).is(':checked'))

    if ($(this).is(':checked')) {
      // On
      if ($colourOptions.val() == "none") {
        console.log("none selected")
      } else {
        // Re-enable the button
        $blinkLedBtn.prop('disabled', false);

        swal("Light", "Light is on!", "info", { button: "YAY!" });
      }
    }
  });

```

```
        // Toggle the LED light
        toggleLed($colourOptions.val());

        // Display the 'Blink LED' button
        $blinkLedBtn.show();
    }
} else {
    // Off
    swal("Light", "Light is off", "info");
    toggleLed("off");
    // Hide the 'Blink LED' button
    $blinkLedBtn.hide();
}
});

}))

// Function to initialize light doughnut/pie chart
function initLightPieChart() {
    // Get light data
    var jqXHR = $.ajax({
        type: "GET",
        url: '/api/lights',
        async: false
    });
    jqXHR.done(function (data, textStatus, jqXHR) {
        drawLightPieChart(data);
    });
    jqXHR.fail(function (jqXHR, textStatus, errorThrown) {
    });
}

// Function to draw the light pie chart based on JSON data
function drawLightPieChart(data) {
    // console.log(data)

    var countArr = [];
    var totalCount = 0;

    // Populate the count
    for (var i in data) {
        countArr.push(data[i].total_count);
    }

    countArr.forEach(function (entry) {
        totalCount += entry;
    });
}
```

```

    $totalCount.text('Total count: ' + totalCount);

    var config = {
        type: 'doughnut',
        data: {
            labels: ["Blue", "Cyan", "Green", "Magenta", "Red", "White",
"Yellow"],
            datasets: [{
                backgroundColor: ["#0080ff", "#7ffffff", "#7FBF7F",
"#ff7fff", "#FF7F7F", "#F0F0F0", "#FFFF7F"],
                data: countArr
            }]
        },
        options: {
            responsive: true
        }
    };

    // Get context & init chart
    var ctx = document.getElementById("lightPieChart").getContext("2d");
    lightPieChart = new Chart(ctx, config);
} // end of drawLightPieChart()

// Function to toggle the LED colour
function toggleLed(colour) {
    // TODO: Hardcoded URL string
    var jqXHR = $.ajax({
        type: "GET",
        url: "http://" + ipAddress + ':8001/led/' + colour,
        async: false
    });
    jqXHR.done(function (data, textStatus, jqXHR) {
        if (colour != "off") {
            // Only init light pie chart if colour != off
            // Destroy the lightPieChart instance before re-init
            lightPieChart.destroy();
            initLightPieChart();
        }
        // Update the current state of LED
        $currentState.text('Current state: ' + data.ledStatus);
    });
    jqXHR.fail(function (jqXHR, textStatus, errorThrown) {
    });
}

</script>
</body>

```



```
</html>
```

In front_end, create a new file called cctv.html

cctv.html

```
<!doctype HTML>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta name="description" content="IoT CA1 Assignment">
  <meta name="author" content="Sherna Liew">
  <title>SOMS</title>
  <!-- CSS -->
  <link href="node_modules/bootstrap/dist/css/bootstrap.min.css" rel="stylesheet">
  <link href="starter-template.css" rel="stylesheet">
  <link href="node_modules/font-awesome/css/font-awesome.min.css"
rel="stylesheet">
  <link href="node_modules/bootstrap-toggle/css/bootstrap-toggle.css"
rel="stylesheet">
  <link href="https://cdn.datatables.net/1.10.16/css/jquery.dataTables.min.css"
rel="stylesheet">
  <link href="node_modules/sweetalert2/dist/sweetalert2.min.css" rel="stylesheet">
  <!-- Javascript libraries -->
  <script src="node_modules/jquery/dist/jquery.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.13.0/umd/popper.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/tether/1.4.0/js/tether.min.js"
integrity="sha384-DztdAPBWPRXSA/3eYEEUWrWCy7G5KFbe8fFjk5JAIxUYHKkDx6Qin1DkWx51bBrb"
crossorigin="anonymous"></script>
  <script src="node_modules/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
  <script src="node_modules/bootstrap/dist/js/bootstrap.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.20.1/moment.min.js"></script>
  <script src="node_modules/bootstrap-toggle/js/bootstrap-toggle.min.js"></script>
  <script src="node_modules/sweetalert2/dist/sweetalert2.all.min.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/core-
js/2.4.1/core.js"></script>
  <script
src="https://cdn.datatables.net/1.10.16/js/jquery.dataTables.min.js"></script>
  <script src="/socket.io/socket.io.js"></script>
  <script src="node_modules/whatwg-fetch/fetch.js"></script>
</head>

<body>
```

```

<nav class="navbar navbar-expand-md navbar-dark bg-dark fixed-top">
  <a class="navbar-brand" href="/">SOMS</a>
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarsExampleDefault" aria-controls="navbarsExampleDefault"
  aria-expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>

  <div class="collapse navbar-collapse" id="navbarsExampleDefault">
    <ul class="navbar-nav mr-auto">
      <li class="nav-item active">
        <a class="nav-link" href="/">
          <i class="fa fa-home" aria-hidden="true"></i>
          <span class="sr-only">(current)</span>
        </a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="user_management.html">Users</a>
      </li>
      <li class="nav-item dropdown">
        <a class="nav-link dropdown-toggle" id="dropdown01" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">Attendance</a>
        <div class="dropdown-menu" aria-labelledby="dropdown01">
          <a class="dropdown-item"
href="take_attendance.html">Attendance Taking</a>
          <a class="dropdown-item" href="attendance.html">Attendance
Log</a>
        </div>
      </li>
      <li class="nav-item dropdown">
        <a class="nav-link dropdown-toggle" id="dropdown01" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">Office</a>
        <div class="dropdown-menu" aria-labelledby="dropdown01">
          <a class="dropdown-item" href="lights.html">Lights</a>
          <a class="dropdown-item" href="cctv.html">CCTV</a>
        </div>
      </li>
    </ul>
  </div>
</nav>

<main role="main" class="container">
  <h1 class="text-primary">
    <i class="fa fa-video-camera" aria-hidden="true"></i>&nbsp;  CCTV</h1>
  <hr />
</main>

<div class="container">

```

```

    <!-- CCTV -->
    <div class="card border-primary text-center">
        <div class="card-header text-white bg-primary">
            <i class="fa fa-video-camera" aria-hidden="true"></i>&nbsp;Live
Video Feed
        </div>
        <div class="card-block">
            <br />
            <!-- TODO: Hardcoded URL link to access IP address of 2nd raspberry
pi -->
            
        </div>
        <br />
        <div class="card-footer border-primary text-muted">
        </div>
    </div>
    <br>
    <br />
    <br />
    <br />
    <br />
    <br />
    <br />
    <br />
    <hr />
    <br />
</div>
<!-- End of container -->
<footer class="footer bg-dark text-center">
    <div class="container">
        <br />
        <span class="text-muted">&copy; 2017 - 2018 Liew Zhi Li (Sherna). All
Rights Reserved.</span>
        <br />
        <br />
    </div>
</footer>
<script>
    // Init Socket.IO to make connection to the Socket.IO server at the same URL
the page is being hosted on
    var socket = io()

    $((() => {

        })

    </script>
</body>

```

```
</html>
```

If you have a second Raspberry Pi and PiCam lying around, you can use that to perform live streaming.

You can simply clone the project from [here](#) onto the second Raspberry Pi. Once you have done that, simply open Terminal and execute the following command.

```
./start.sh
```

You are executing this start.sh shell script to start up the web application for video streaming.

.sh file are called “shell script” which is for Linux environment. The Window’s equivalent to this would be the .bat file called “batch script”.

This is the only solution that I found that has the lowest latency when streaming across the network, thus I used it.

Note: You will need to change the IP address to the IP of the second Raspberry Pi that you are using.

In front_end, create a new file called colours.css

colours.css

```
.bg-purple {
    background-color: #6f42c1 !important;
}

.bg-pink {
    background-color: #e83e8c !important;
}

.bg-orange {
    background-color: #fd7e14 !important;
}

.bg-teal {
    background-color: #20c997 !important;
}

.bg-indigo {
    background-color: #6610f2 !important;
}

.btn-outline-purple{
    border-color: #6f42c1 !important;
    background-color: #F5F8F8 !important;
    color: #6f42c1 !important;
}

.btn-outline-purple:hover {
    border-color: currentcolor;
    background-color: #6f42c1 !important;
    color: #FFFFFF !important;
}

.border-purple{
    border-color: #6f42c1 !important;
    color: #FFFFFF !important;
}

.bg-purple{
    color: #6f42c1 !important;
    color: #FFFFFF !important;
}
```

In `front_end`, create a new file called `starter_template.css`

starter_template.css

```
body {  
  padding-top: 5rem;  
}  
.starter-template {  
  padding: 3rem 1.5rem;  
  text-align: center;  
}
```

In python, create a new file called server.py

server.py

```
import datetime
import gevent
import gevent.monkey
from gevent.pywsgi import WSGIServer
import pymongo

from gpiozero import Button, Buzzer
from picamera import PiCamera
import time
from time import sleep
from signal import pause

gevent.monkey.patch_all()

from flask import Flask, request, Response, render_template, jsonify
from flask_cors import CORS, cross_origin
from gpiozero import RGBLED

##### Pin Declarations #####
button = Button(26)
buzzer = Buzzer(18)
led = RGBLED(red=16, green=20, blue=21, pwm=False)

##### Connect to MongoDB #####
try:
    conn = pymongo.MongoClient('mongodb://localhost:27017/')
    print("Successfully connected to MongoDB")
except pymongo.errors.ConnectionFailure, e:
    print("Could not connect to MongoDB: {}".format(e))

# MongoDB collections
lightsCollection = conn.iot_ca1.lights

# Default led to 'off' state
led.color = (1, 1, 1)

# Function to sound buzzer to represent door bell ringing when visitor presses the
# arcade button
def buzzer_open_door():
    buzzer.on()
    sleep(0.25)
    buzzer.off()
    sleep(0.25)

    buzzer.on()
```



```
    sleep(0.50)
    buzzer.off()
    sleep(0.50)

# When button is pressed, sound the buzzer
button.when_pressed = buzzer_open_door

# Functions to store a toggled light colour
def addLight(colour):
    try:
        lightsCollection.insert_one({
            "Colour": colour,
            "Timestamp": datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")
        })
        print("Successfully added 1 light record to MongoDB")
    except pymongo.errors.OperationFailure, e:
        print("Error attempting to add 1 light record to MongoDB: {}".format(e))

# Functions to on different light colours
def onRed():
    led.color = (0, 1, 1) # full red
    addLight("red")
    return "red"

def onGreen():
    led.color = (1, 0, 1) # full green
    addLight("green")
    return "green"

def onBlue():
    led.color = (1, 1, 0) # full blue
    addLight("blue")
    return "blue"

def onYellow():
    led.color = (0, 0, 1) # yellow
    addLight("yellow")
    return "yellow"

def onMagenta():
    led.color = (0, 1, 0) # magenta
    addLight("magenta")
    return "magenta"

def onCyan():
    led.color = (1, 0, 0) # cyan
    addLight("cyan")
    return "cyan"
```

```
def onWhite():
    led.color = (0, 0, 0) # white
    addLight("white")
    return "white"

def blink():
    currentColour = led.color
    led.blink(on_color = currentColour, off_color=(1, 1, 1)) # blink the led with
currentColour && off colour
    return "blinking"

def offLed():
    led.color = (1, 1, 1) # off
    return "off"

# define app
app = Flask(__name__)
CORS(app, support_credentials=True)

@app.route("/led/<colour>")
@cross_origin(supports_credentials=True)
def onLED(colour):
    # lowercase the string for comparison
    colour = colour.lower()

    if colour == "red":
        response = onRed()
    elif colour == "green":
        response = onGreen()
    elif colour == "blue":
        response = onBlue()
    elif colour == "yellow":
        response = onYellow()
    elif colour == "magenta":
        response = onMagenta()
    elif colour == "cyan":
        response = onCyan()
    elif colour == "white":
        response = onWhite()
    elif colour == "blink":
        response = blink()
    else:
        response = offLed()

    responseData = {
        'ledStatus': response
    }
```

```
# Send the response back
return jsonify(**responseData)

# Main method
def main():
    try:
        http_server = WSGIServer(('0.0.0.0', 8001), app)
        app.debug = True
        print("Server is running on port: 8001")
        http_server.serve_forever()
    except:
        print("Exception running server")

if __name__ == '__main__':
    main()
```

In arduino, create a new file called 2_servos.ino

2_servos.ino

```
#include <Servo.h>

// define our servos
Servo servo1;
Servo servo2;

// define joystick pins (Analog)
int joyX = 0;
int joyY = 1;

// variable to read the values from the analog pins
int joyVal;

void setup() {
  // put your setup code here, to run once:
  // attaches our servos on pins PWM 3-5
  servo1.attach(3);
  servo2.attach(5);
}

void loop() {
  // put your main code here, to run repeatedly:
  joyVal = analogRead(joyX);
  joyVal = map (joyVal, 0, 1023, 0, 180); // servo value between 0-180
  servo1.write(joyVal); // set the servo position according to the joystick value

  joyVal = analogRead(joyY);
  joyVal = map(joyVal, 0, 1023, 0, 180);
  servo2.write(joyVal);
  delay(15);
}
```

Here are the image assets you need:

Download the rfid image from [here](#) and drop it in the front_end folder.

That's it! You have successfully set up the entire project!

Section 10

Running SOMS

Open up Terminal on your Raspberry Pi.

Open up 3 tabs in the Terminal by using Ctrl + Shft + T.

In the 1st tab, we need to first start up the MongoDB server.

Run the following command to start up the MongoDB server.

```
mongod
```

In the 2nd tab, cd into the soms/back_end

Run the following command to start up the Node.js server.

```
sudo nodemon server.js
```

In the 3rd tab, cd into soms/python folder

Run the following command to start up the Python Flask server.

```
sudo python server.py
```

On your extra Raspberry Pi which is responsible for the live streaming, run the following command to start up the live stream.

```
./start.sh
```

That's it! You have successfully started up the SOMS application! 😊

-- End of CA1 Step-by-step tutorial --