Project #5
Computer Science 4366 Senior Capstone:
Validation and Full Demo (VD)

Big Blue Button
Alarm System Project

Team Members:
Bipin Chhetri
Breanae Campbell-Rivera
Cameron Gibson
Sally Comer
Saroj Gopali

Table of Contents

1. Introduction

1.1 Project Description

The goal of this project is to create a system to prevent intruders from walking into the user's room and steal any valuable item. The project named "Big Blue Button", is inspired by blue (color) which often symbolizes serenity, stability. Besides being a calming color, blue also symbolizes reliability (A sign showing being safe and secured from being a victim of property crime). In the project, we are implementing an **Alarm System** for home security and detection of any trespassers in the room. The final product will consist of both software and hardware part connected via Ethernet/Wi-Fi. The hardware consists of Raspberry pi connected with a motion sensor and a camera.

The Software side will be connected and accessed via a Mobile application which can be used remotely. Some preconditions that should be met include activating the device and registering the user in that device. Initially, when the user is registered in the system and his/her login details are also registered. After the registration, the user will have access to turn the system on/off which will activate the alarm, this process can be done on the android mobile application or the website for the system. The Big blue button will then detect any intruder in the room and notify the user with text message or by e-mail.

1.2 Intended Audience and Reading Suggestions

This document is intended both for the designers of the system mentioned here in as well as for users of said system. However, the main target audience is for students sharing their room with other students living in dorm or any shared apartments.

1.3 Product Scope

The product's name will be the Big Blue Button, we decided to name the system this because the color blue often symbolizes serenity, stability. Besides being a calming color, blue also symbolizes reliability (A sign showing being safe and secured from being a victim of property crime). The Big Blue Button will monitor the intended range that it is placed in, and when motion is detected in the area it will send off an alarm to the user's phone and /or email. Included in the email and/or text ...will be a picture of the intruder along with a message that will advise the user to open the application and proceed with telling the hardware what to do next. The hardware will not act without instruction from the user, which gives the user the power over the situation even when they are not home.

1.4 References

Hardware: https://www.raspberrypi.org/

Programming language: https://www.python.org/

Integrated Development Environment: https://developer.android.com/studio

Twilio (text and email alert): https://www.twilio.com/

AWS: https://aws.amazon.com/

1.5 Definitions, Acronyms and Abbreviations.

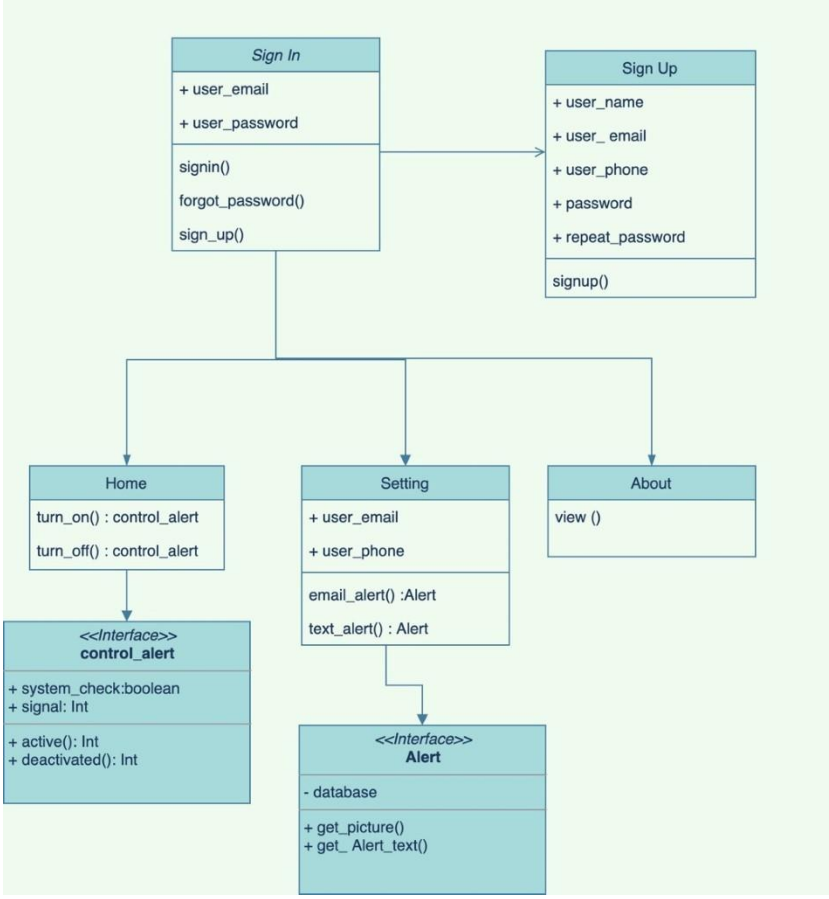
1. **User:** The person who runs the Big Blue Button alarm system.
2. **System:** The system means the Big Blue Button alarm device.
3. **Mobile Application:** The android mobile software that runs on phone.
4. **Raspberry pi:** A small low-cost single board computer which runs on Raspbian, the Debian-based Linux OS
5. **API- (Application programming interface):** A computing interface that defines interactions between multiple software intermediaries.
6. **Open CV- (Open-Source Vision Library):** An open-source computer vision and machine learning software library.
7. **AWS S3 bucket- (Amazon Simple Storage Service)**: An object storage service that offers industry-driving versatility, information accessibility, security, and performance.
8. **Twilio**: Twilio is a Cloud communication platform. Twilio permits developers to programmatically make and receive phone calls, send, and get instant messages, and perform other correspondence functions using its web service APIs.
9. **AWS Rekognition:** Amazon Rekognition is a cloud-based Software as a service computer vision platform. Amazon Rekognition makes it easy to add image and video analysis to an application using proven, highly scalable, deep learning technology.
10. **UML Diagram- (Unified Modeling Language):** A UML diagram is a diagram with the end goal of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

2. Design Details

For our project we used three UML diagram namely Class Diagram, Sequence Diagram and Use Case Diagram.

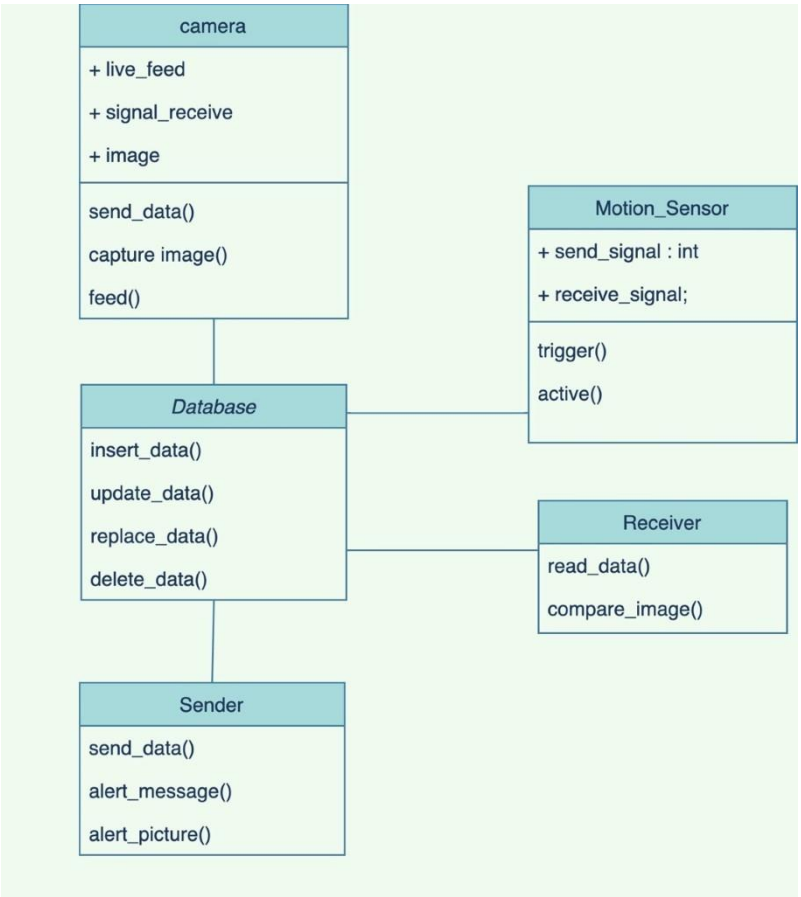
2.1 Class Diagram

The software class diagram contains the classes of mobile application of the system. Sign in to get access to the application using a username and a password. The sign-in also contains a "forgot password" section to assist the user in resetting their password. The signup will collect all the user's information which will be stored in a database for further user interaction. The user should provide permission through the settings to receive alerts from the system. The home class contains an option to toggle the device on or off.



The hardware class diagram holds the classes of hardware and the database of the system. The database contains insert data, update data, replace data and delete data of

the user for the system. The camera class contains capture image and send data functions which captures an image if the trigger from motion senor sends to the signal. The motion sensor class also contains an active function to begin detecting motion. The receiver and sender classes hold read data, compare image and send data, alert message, alert picture simultaneously.



2.2 Sequence Diagram

The figure 1 image will display the system's overall sequence diagram where the application, database, and hardware communicate based on the user instruction. The application sends a signal to turn on /off the big blue button system through the database to the hardware. Whereas the hardware passed the image to the database, and

the database communicates with AWS Rekognition to compare images if available previously and Twilio for alert text and email message.

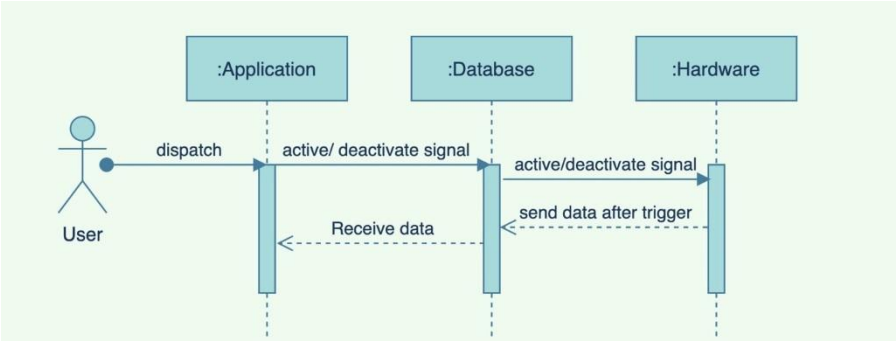


Figure :1

Figure 2 diagram shows a diagram where the sign-in sends the data to the database to authenticate the user's information. The user also has the option to sign up or reset the password on the sign-in page. The sign up will send all user information to the database to store or create new user accounts for further interaction.

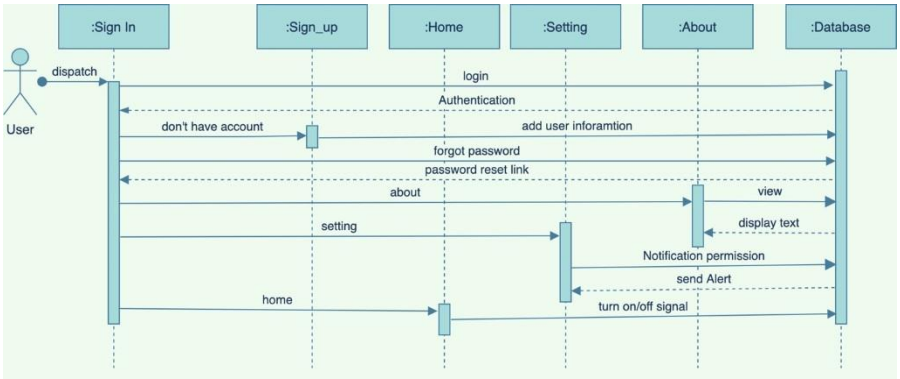
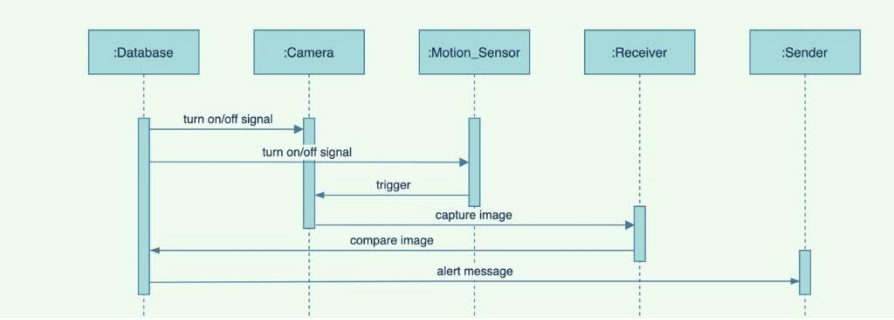


Figure: 2

The sequence diagram figure 3 of hardware shows the signal to toggle the alerts for the camera and motion sensor will be sent through the database which the user has provided on their mobile application. If there is movement, it will trigger the motion sensor to send a signal to the camera to a capture image. The image will be sent to the receiver from database. The image from database will send to AWS Rekognition and compare if the image in recognized using Amazon web server as

host. If the image is not recognized, then the sender will receive an alert through database to the user in email or text or both.



2.2.1 Use Case Diagram

These figures contain the use case diagrams of Big Blue Button User, Big Blue Button Hardware, and Big Blue Button User/ Hardware.

In figure 1, the user will interact with the Big Blue Button application to log in / sign up. It will also provide the feature for the user to choose alert notification and toggle the device on/off.

Figure 2 shows the interaction between hardware and motion detected, the camera begins photo capture, recognizes the image, and sends alerts to the user.

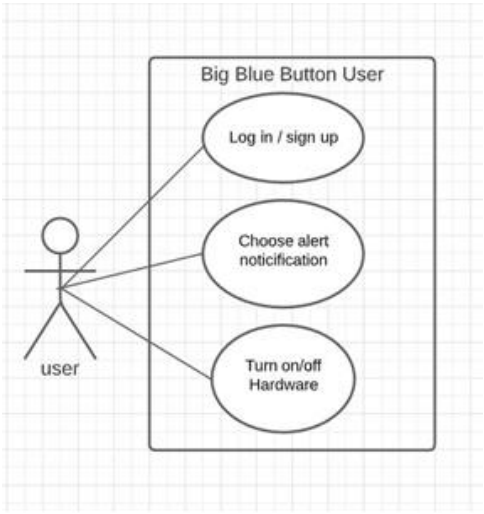


Figure 1 Use Case

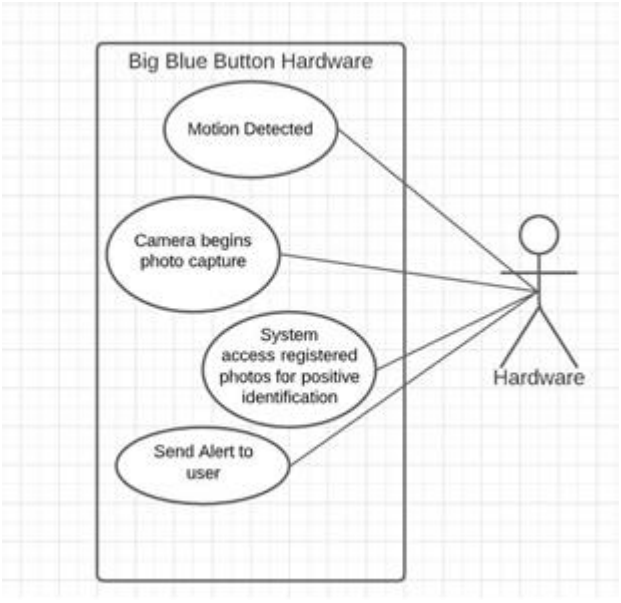


Figure 2 Use Case

Figure 3 shows the interaction of the user with the Big Blue Button System. The user receives the alert message, and the user can also log in to the application if need to control the system where the Big Blue button can only send an alert message to the user.

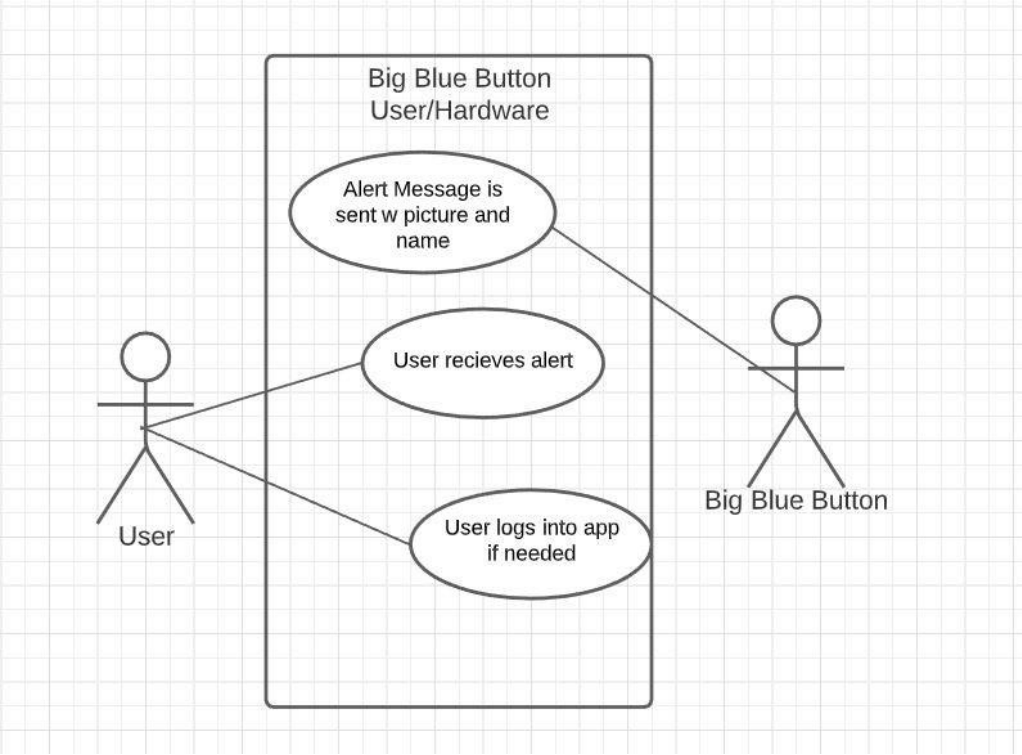


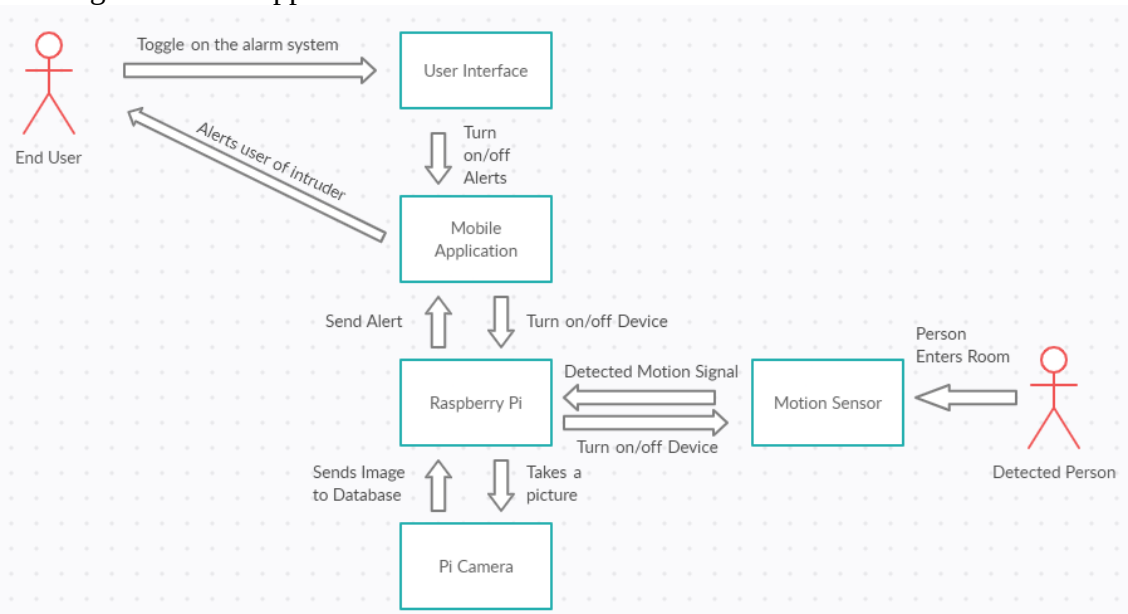
Figure 3 Use Case

3. Implementation Details

3.1 Implementation Overview

To give a more concise view of our system, we have broken our sections into Software implementation, Hardware implementation and implementation issues. Here is an overall

use diagram for our application.

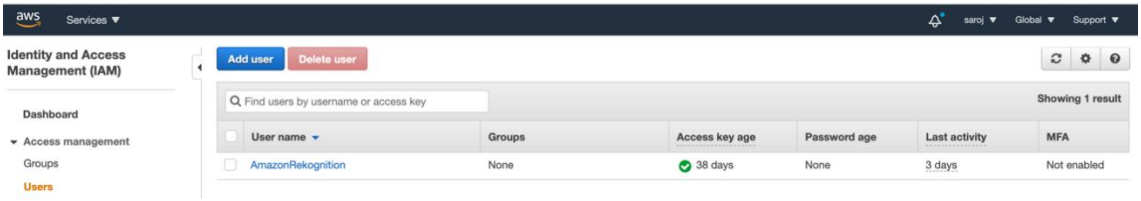


3.2 Software Implementation

On the software end, our team has developed an android mobile application using Java programming language on the android studio Integrated Development Environment. To host the system, we use Amazon Web Service. To send the alert message, we used Twilio API.

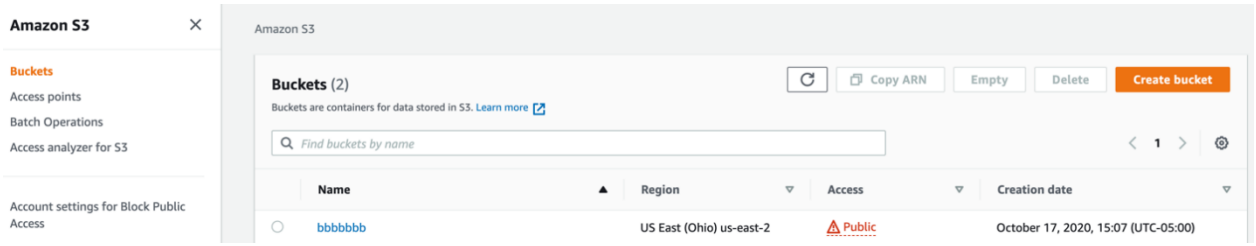
The following shows services and software our team has used to implement parts of the Big Blue Button system:

- **AWS Rekognition**: - To compare the faces to check match



To setup AWS Rekognition:
https://docs.aws.amazon.com/rekognition/latest/dg/what-is.html

- **AWS S3 bucket**: - To store image from Pi Camera



To setup AWS S3 Bucket:
https://docs.aws.amazon.com/AmazonS3/latest/gsg/GetStartedWithS3.html

- **Twilio**: - To send an alert to the user
  To set up Twilio: https://www.twilio.com/docs

- **Open CV**: - To detect image and send to AWS Rekognition
  To set up Open CV: https://opencv-python-tutroals.readthedocs.io/en/latest/

3.2.1 Feature Implementation

3.2.1.1 Add Image to Collection in AWS

```
parser = argparse.ArgumentParser(description='Facial recognition')
parser.add_argument('--collection', help='Collection Name',
    default='bb-faces')
parser.add_argument('--face_cascade', help='Path to face cascade.',
    default='../opencv/data/haarcascades/haarcascade_frontalface_alt2
    .xml')
parser.add_argument('--camera', help='Camera device number.', type=int,
    default=0)
args = parser.parse_args()
```

The feature will take image from pi camera and send to open cv it pass to aws collection for the AWS Recognition.

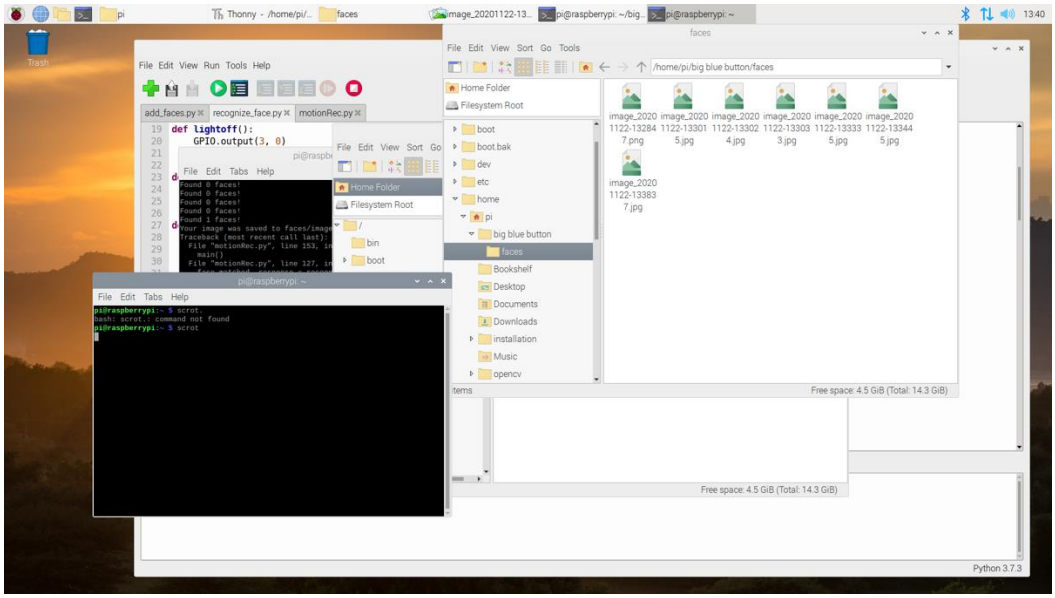
3.2.1.2 Face detects function in Big Blue Button

```
def detectFace(frame,face_cascade):
    face_detected = False
    #Detect faces
    faces = face_cascade.detectMultiScale(frame,
        scaleFactor=1.1,
        minNeighbors=5,
        minSize=(30, 30),
        flags = cv.CASCADE_SCALE_IMAGE)
    print("Found {0} faces!".format(len(faces)))
    timestr = time.strftime("%Y%m%d-%H%M%S")
    image = '{0}/image_{1}.jpg'.format(directory, timestr)
    if len(faces) > 0 :
        face_detected = True
        cv.imwrite(image,frame)
        print 'Your image was saved to %s' % image

    return face_detected, image
```

The function will take frame camera and pass to open cv and print if the face is detected or not. It will also store the time stamp of image taken.

Example from Big Blue Button



3.2.1.3 Face Recognition function in Big Blue Button

```
def recognizeFace(client,image,collection):
    face_matched = False
    with open(image, 'rb') as file:
        response = client.search_faces_by_image(CollectionId=collection,
            Image={'Bytes': file.read()}, MaxFaces=1, FaceMatchThreshold=85)
        if (not response['FaceMatches']):
            face_matched = False
        else:
            face_matched = True
    return face_matched, response
```

The function will take an image and check if the image is in the collection, if it is not, it will return false, if recognized return true.

3.2.1.4 Upload Image function in Big Blue Button

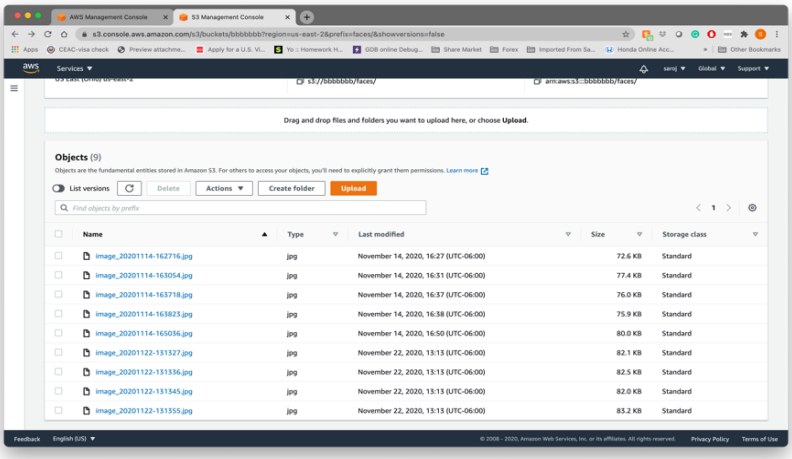
```
def uploadPhoto(fName):
    s3=boto3.resource('s3')
    data = open(fName, 'rb')
    s3.Bucket(bucketName).put_object(Key=fName, Body=data, ContentType =
        'image/jpeg')

    # makes uploaded image link public
    object_acl = s3.ObjectAcl(bucketName, fName)
    response = object_acl.put(ACL='public-read')

    link = 'https://'+bucketName+'.s3.us-east-2.amazonaws.com/'+fName
    return link
```

The function will take the file name from the directory in Raspberry where it stores the images and pass to AWS s3 to store and it will return an image link.

Example from Big Blue Button



3.2.1.5 Send Alert function in Big Blue Button

```
def twilio(message, imageLink):
    client = Client(account_sid, auth_token)

    message = client.messages.create(to='+18062392436', from_='+12513134607', body
        = message, media_url=imageLink)
```

The function will take message and image link from s3 bucket to send an alert to the user.

Example from Big Blue Button

```
if (face_matched):
     print ' Image matched'
    twilio("!!! Attention !! !Big Blue Button  Has Detected a
        Known Person ", uploadPhoto(image))
else:
    print 'Unknown Human Detected!'
    twilio("!!!!! Intruder !!!! Big Blue Button Has Sensed
        Unknown Person ", uploadPhoto(image))
```

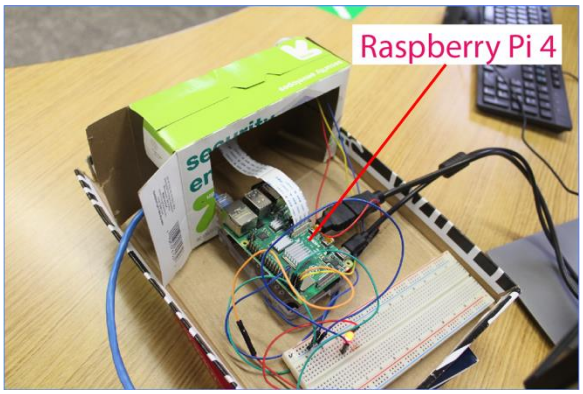
Alert from Big Blue Button

3.3 Hardware Implementation

Overview

The Project's hardware consists of a Raspberry Pi 4, a RGB LED, an Infrared PIR Motion Sensor to detect motion, a Raspberry Pi Camera Module V2 to take pictures, and a Raspberry Pi expansion kit for breadboard and expansion board.

3.3.1 Raspberry Pi 4



Big Blue Button System

```
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(11, GPIO.IN)
GPIO.setup(3, GPIO.OUT)

def lightoff():
    GPIO.output(3, 0)
    time.sleep(.3)

def lightOn():
    GPIO.output(3, 1)
    time.sleep(.3)
```

3.3.2 Motion Senor

```
while True:
    i=GPIO.input(11)
    if i==0:                #When output from motion sensor is LOW
        print ("No intruders")
        lightoff() #Turn off LED

    elif i==1:              #When output from motion sensor is HIGH
        print ("Intruder detected")
        lightOn()  #Turn ON LED
```

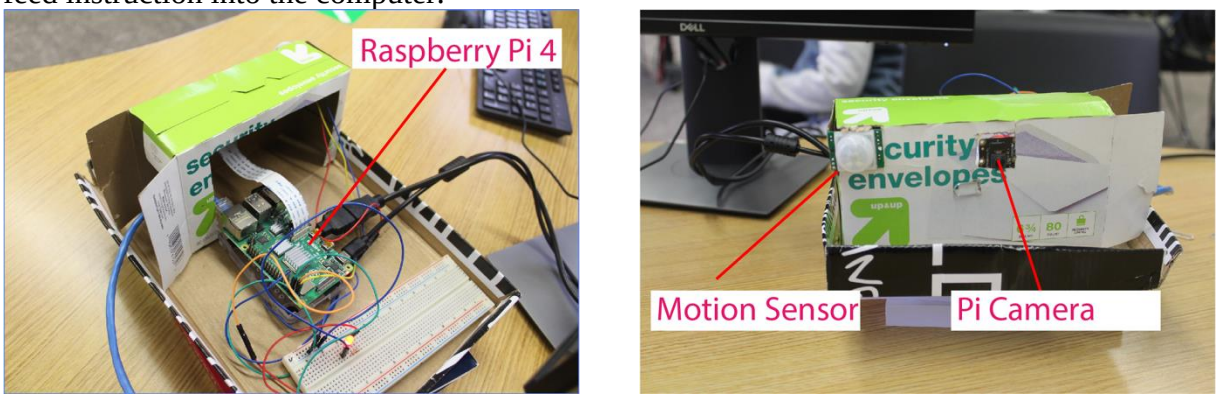
3.4 Implementation Issues

- An implementation issue that was quite noticeable was on the hardware part of the Big blue button. The camera connected to our raspberry pi 4 has a long ribbon which makes it harder to stay put. Having a rigid stand to mount the device and fixing the spot was an obstacle. In addition, the motion sensor had a similar issue to remain stable. To fix the issue a temporary holding box was made to hold the raspberry pi n motion sensor and the camera.
- Another issue with the camera involved running too many jobs at once, this would cause it to become stuck in running processes and produce errors.
- While working with Raspberry we had a problem accessing AWS, therefore, causing an error while executing the code. The issue was the ethernet for the internet which did not update the current time. Temporarily, we connected the hardware through a mobile hotspot to connect the time and able to work with AWS as well as the text alert part. After fixing on the External end connection the ethernet is now fixed and working.

4. Functionalities

4.1 Hardware Functionalities

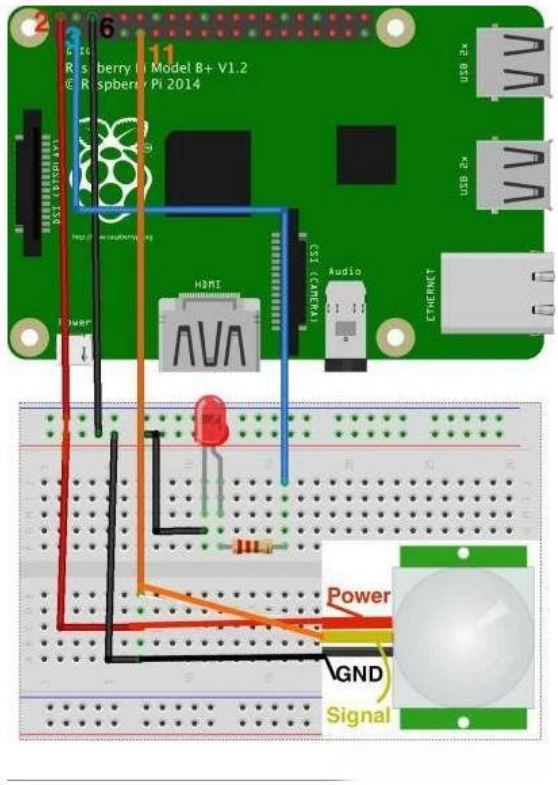
The hardware has much of the functionality for the project that can be carried out using a stable Internet connection. The hardware functionality uses the Raspberry Pi 4 subsystem. The Raspberry pi system consists of the motion sensor, Pi camera, which is then connected to a monitor to display and feed instruction into the computer.

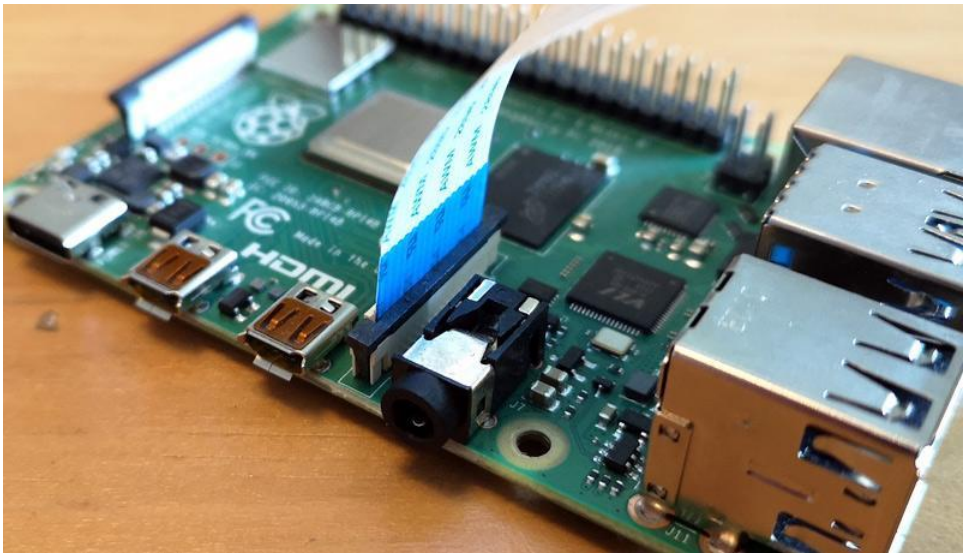


Big Blue Button System

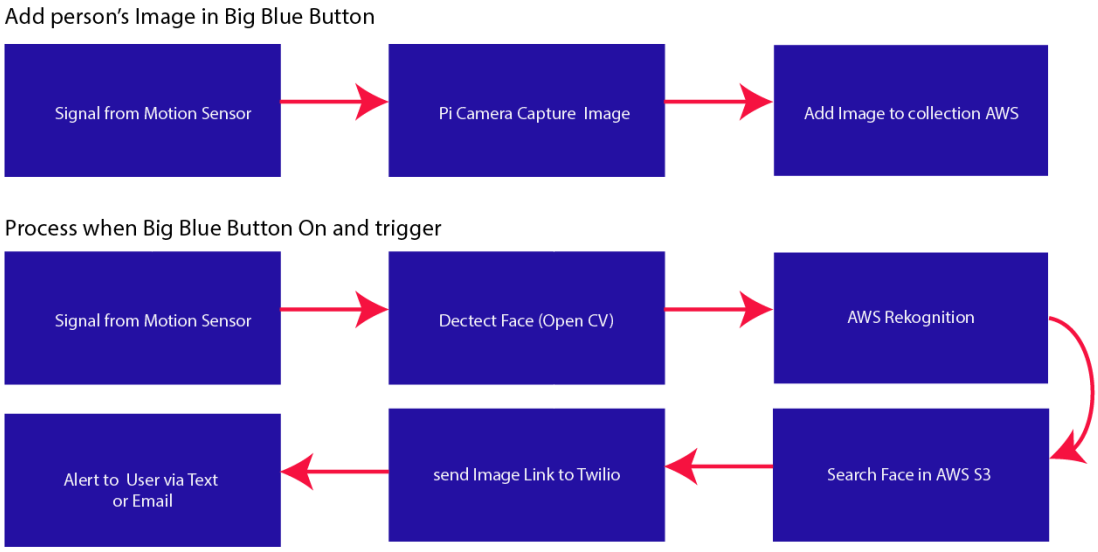
The above figure displays the hardware setup done during the testing of the Raspberry Pi. A temporary cardboard stand is set up to hold the motion sensor which separate 8 cm away from the pi Camera.

The hardware was created using multiple resources! The description is as followed...





Our Motion Sensor was hooked up to the raspberry pi by using a breadboard, cords and the motion sensor... Along with the resistor and led light of choice to make sure the connections were plugged in correctly. Our Camera was plugged into the camera port and was tested for functionality through the terminal in python.



When a signal is detected in the motion sensor the pi camera triggers and takes the picture which is sent to AWS database. In a similar case, if the face is detected the face is then matched with one in AWS S3 to see if the face matches. If the match is found or not it will

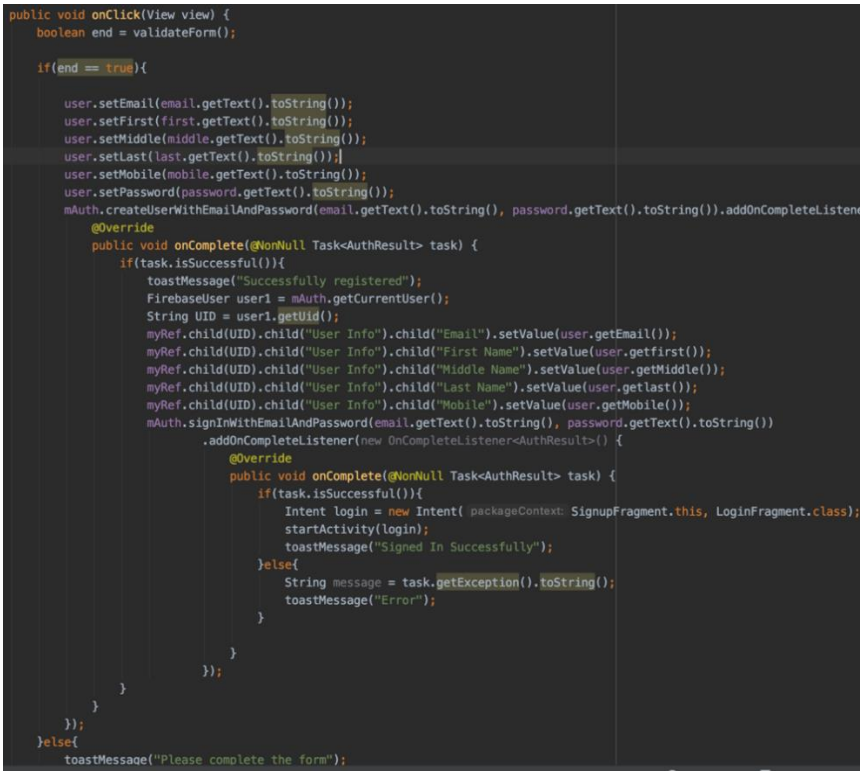
send an image to Twilio and send a text or an email alert to the registered user depending on their device settings.

4.2 Application Functionalities Overview

The Big Blue Button android application was developed using android studio programmed with java programming language. The application has a connection with firebase which stores user data, using firebase authentication user's login credentials i.e. email addresses can be stored. The stored information the user provided during the signup period will be matched in that of the firebase to successfully log in the system application.

4.3 Application Functionalities

The Application functionalities include being able to sign in the Big Blue Button application, turn on/off the alarm, and either enable or disable the text and email alert into the user associated phone or email. Figures 1 and 2 show the signup and sign in page of the big blue button application. Figure 2 show the email address store in the firebase after signing up for the application. Here is the snip of the code which we used for the application to do sign up in the android application.



```
public void onClick(View view) {
    boolean end = validateForm();

    if(end == true){

        user.setEmail(email.getText().toString());
        user.setFirst(first.getText().toString());
        user.setMiddle(middle.getText().toString());
        user.setLast(last.getText().toString());
        user.setMobile(mobile.getText().toString());
        user.setPassword(password.getText().toString());
        mAuth.createUserWithEmailAndPassword(email.getText().toString(), password.getText().toString()).addOnCompleteListene
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if(task.isSuccessful()){
                    toastMessage("Successfully registered");
                    FirebaseUser user1 = mAuth.getCurrentUser();
                    String UID = user1.getUid();
                    myRef.child(UID).child("User Info").child("Email").setValue(user.getEmail());
                    myRef.child(UID).child("User Info").child("First Name").setValue(user.getfirst());
                    myRef.child(UID).child("User Info").child("Middle Name").setValue(user.getMiddle());
                    myRef.child(UID).child("User Info").child("Last Name").setValue(user.getlast());
                    myRef.child(UID).child("User Info").child("Mobile").setValue(user.getMobile());
                    mAuth.signInWithEmailAndPassword(email.getText().toString(), password.getText().toString())
                            .addOnCompleteListener(new OnCompleteListener<AuthResult>() {
                                @Override
                                public void onComplete(@NonNull Task<AuthResult> task) {
                                    if(task.isSuccessful()){
                                        Intent login = new Intent( packageContext: SignupFragment.this, LoginFragment.class);
                                        startActivity(login);
                                        toastMessage("Signed In Successfully");
                                    }else{
                                        String message = task.getException().toString();
                                        toastMessage("Error");
                                    }

                                }
                            });
                }
            }
        });
    }else{
        toastMessage("Please complete the form");
```

Code snip for Sign Up

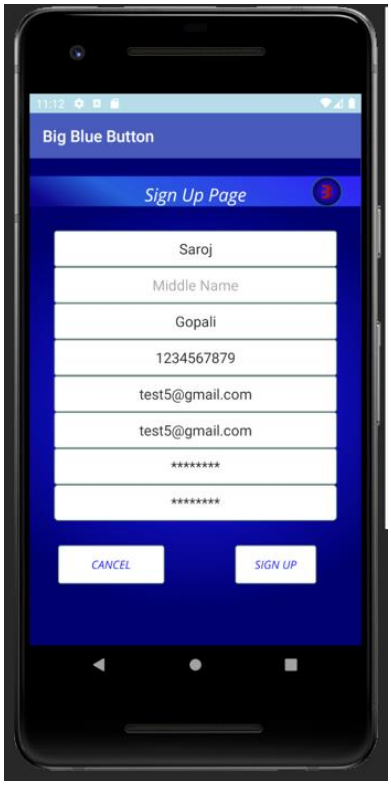


Figure 4 *and Figure 5*

The figure above displays the sign-up and sign-in page in The Big Blue Button application page. The sign-up page requires the user's details which include, name, phone number, an email address and password. Then the next, step requires the user to use credentials to login into the user portal to access the function to activate the alert system from his application.

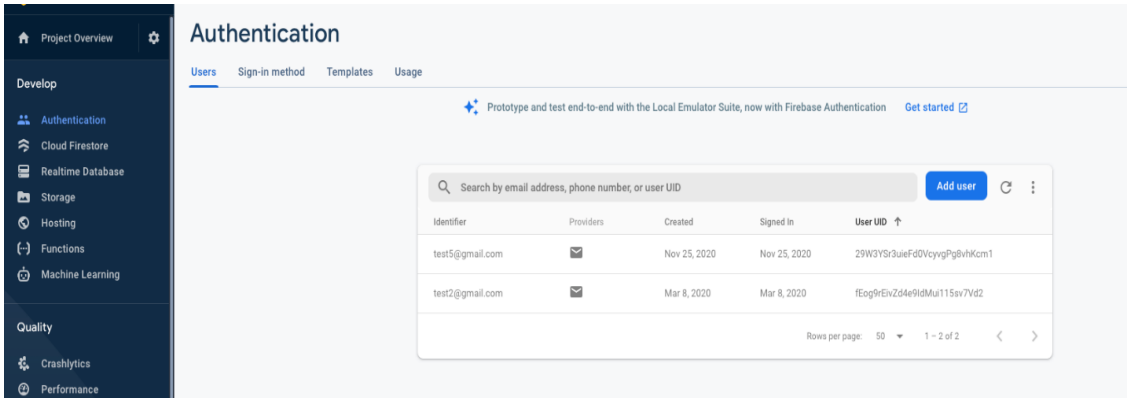


Figure 6

Figure 6 displays that the user has signed up for the application and the credentials are saved in the database. While logging in the system will match if the credential in the database is found and matches the user will be able to login.

4.3.1 Completed Functionalities

Figures 4 and 5 are the main functions of the application. Figure 4 is the home button which sends a signal to turn on or off the hardware, and figure 5 is the alert where user has the option to select email alert, phone or both.

Figure 7 *Figure 8*

The figure displays a big button which is found on the home page as soon as the user logs in for quick access to turn in and off the Alarm system. When the menu option is clicked on the top right the user can access the settings where they have option to edit with their specific preferences.

4.3.2 Remaining Functionalities

Figure 6 is the additional functions which we proposed for future work. The feature will provide users the ability to open a live feed for the area where the Big Blue Button device is installed.

One of the vital functions also a live feed option which can be used to capture the intruder in real-time and record the intruder in the room. The figure 6 above

displays a screenshot of how the interface of live feed looks in the mobile application.

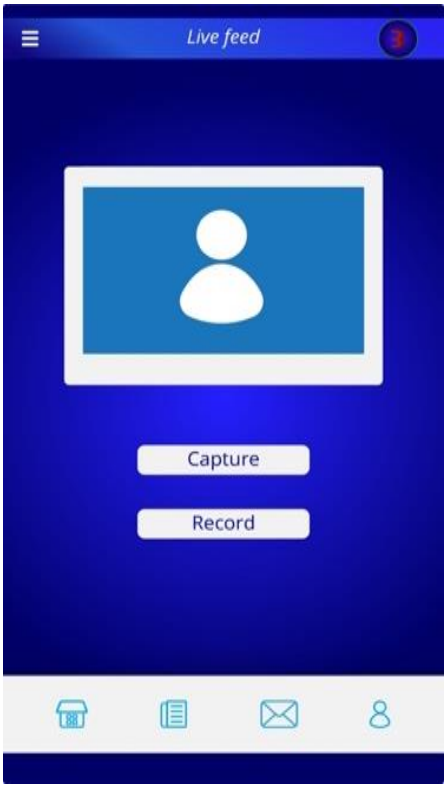


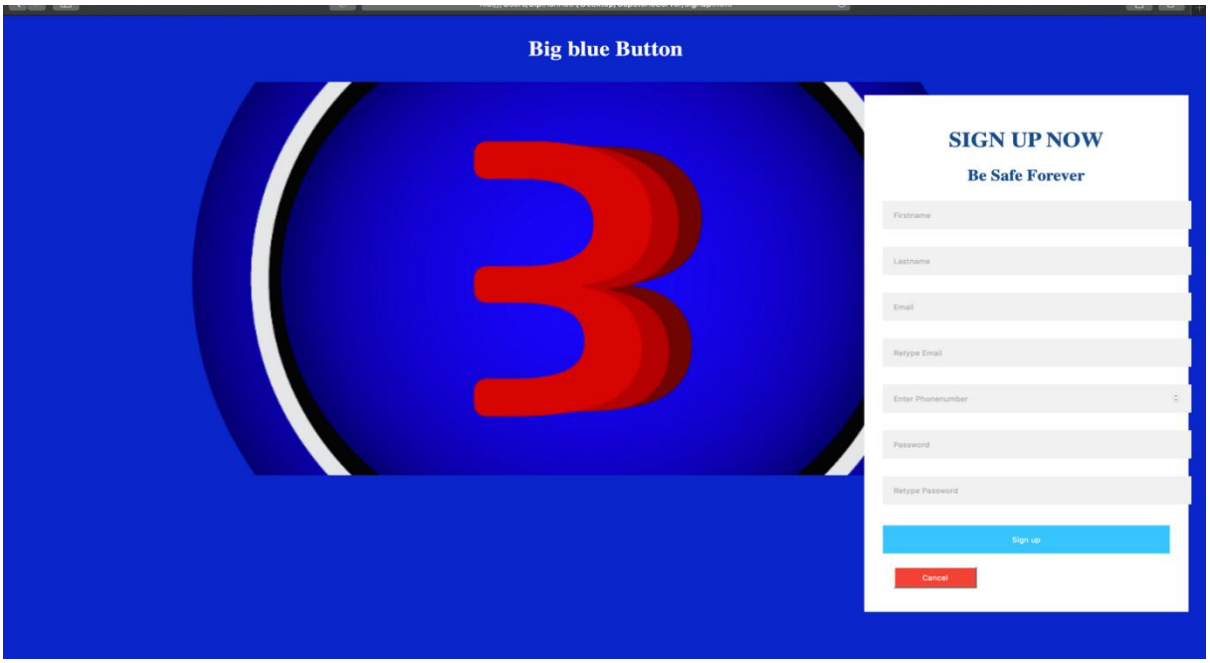
Figure 10

4.4 Website Functionalities:

The big blue button website was developed using HTML and CSS. The website enables the user to log in or signup. Also, the other functionalities of the website include viewing the picture of the intruder that comes into the restricted area.

Home page

The above figure displays the home page of the website. Which has the function to enable the user to login and sign up. The figure below shows the snippet of the sign-up and login page.



Sign up page

Validation and Full Demo for Big Blue Button

Login in page

Validation and Full Demo for Big Blue Button

homepage.html — CapstoneServer

about.html | homepage.css | homepage.html | index.html | login.html | styles.css | styles.css (Working Tree) | logins

homepage.html

```
<!DOCTYPE html>
<head>
    <meta charset="utf-8" />
    <title>Big blue Button </title>
    <link rel= "stylesheet" href="homepage.css">
</head>
<body style="background-color:#0B26CA;">
<div  class= "container">
<h1 style="padding-left:500px"><p style="color:white;">Big blue Button</p> </h1>
</div>
<div class="topnav" id="myTopnav">
  <div class= "nav-wrap">

                <div class= "nav-wrap-link">
                  <a href="index.html class="active">Home</a>
                 </div>
                 <div class= "nav-wrap-link">
                      <a href="setting.html">Setting </a>
                    </div>
                    <div class= "nav-wrap-link">
                        <a href="login.html">Log in </a>
                    </div>
                    <div class= "nav-wrap-link">
                        <a href="signup.html">Sign up</a>
                    </div>
                    <div class= "nav-wrap-link">
                        <a href="about.html">About</a>
                    </div>

            </div>
        </div>
</div>
<div class= "Center-side;" style="padding-top: 100px;padding-left:250px">
                <img  align="center" src="logo.png" alt="Big blue button" >

        </div>


</body>
</html>
```

Homepage front end HTML

EXPLORER

OPEN EDITORS

CAPSTONESERVER

- images copy
- about.html
- bluebackground.jpeg
- chirp.jpg
- homepage.css M
- homepage.html
- image1.jpeg
- index.html
- login.html
- loginstyle.css
- logo.png
- server.py
- setting.html
- signup.css
- signup.html
- styles.css

about.html | homepage.css | logo.png | homepage.html | index.html

homepage.css > {} @media screen and (max-width: 600px)

```
body {
    margin: 0;
    font-family: Arial, Helvetica, sans-serif;
}

.topnav {
    overflow: hidden;
    background-color: rgba(136, 136, 136, 0.533);
}

.topnav a {
    float: left;
    display: block;
    color: #f2f2f2;
    text-align: center;
    padding: 14px 16px;
    text-decoration: none;
    font-size: 17px;
}


/* For grid system for hovering effect */

.topnav a:hover {
    background-color: #ddd;
    color: black;
}

.topnav a.active {
    background-color: #4CAF50;
    color: white;
}

.topnav .icon {
    display: none;
}
```

Homepage front end CSS

The two snippet shows the design layout of the home page for the big blue button. The base of the website design was done in CSS.