CHAPTER 5
IMPLEMENTATION AND TESTING

5.1 Implementation

In this project a series of microcontroller devices is programmed through the Arduino IDE which has been installed on the board as a program controller. The programming language used is the C programming language. The following is a brief description of the code.

Lines 1-8 are libraries used in the project so that they can carry out their functions to connect to Wifi and send notifications to telegram, namely the wifi library, ESP32-Cam and telegram.

```
1. #include <WiFi.h>
2. #include <WiFiClientSecure.h>
3. #include "soc/soc.h"
4. #include "soc/rtc_cntl_reg.h"
5. #include "esp_camera.h"
6. #include <UniversalTelegramBot.h>
7. #include <ArduinoJson.h>
8. #include <Wire.h>
```

Lines 10 – 14 are authentication for connecting to wifi and telegram bots, 10-11 is authentication for WiFi settings by setting the SSID and WiFi password to be used, 12-13 is the setting for the telegram bot. Previously bots were created by searching for the username @BotFather in the search field will then get a special token to be able to access the HTTP API and to control the Telegram BOT, line 14 is a condition for sending photos to position 0 or off.

```
9.
10.  const char* ssid = "iphonefitria";
11.  const char* password = "12345678";
12.  String chatId = "1160943523";
13.  String BOTtoken = "1705881550:AAGjOm-wCGODKxoyBhbCS4Jt0P_pg_mnQQE";
14.  bool sendPhoto = false;
```

In lines 58- 69 is a declaration to try to connect a wifi connection

```
58.  //library
59.    WiFi.mode(WIFI_STA);
60.    Serial.println();
61.    Serial.print("Connecting to ");
62.    Serial.println(ssid);
63.    WiFi.begin(ssid, password);
```

```
64.    while (WiFi.status() != WL_CONNECTED) {
65.      Serial.print(".");
66.      delay(500);
67.  }
68.  Serial.println();
69.  Serial.print("ESP32-CAM IP Address: ");
70.  Serial.println(WiFi.localIP());
```

Lines 126 – 132 are loops for flash and photo sending process.

```
126.
127.   if (sendGambar){
128.     Serial.println("Preparing photo");
129.     sendGambarTelegram();
130.     sendGambar = false;
131.     flashLed = !flashLed;
132.     digitalWrite(FLASH_LAMPU, flashLed);
133.   }
```

Lines 134 – 147 are loops for the PIR sensor.

```
134.
135.   if (sensor){
136.     delay(1000);
137.     if(digitalRead(pir) == 1){
138.       flashLed = !flashLed;
139.       digitalWrite(FLASH_LAMPU, flashLed);
140.       Serial.print("Motion Detected, Value = ");
141.       Serial.println(digitalRead(pir));
142.       String motion = "Terdeteksi gerakan!!\n";
143.       motion += "Foto akan segera dikirim\n";
144.       bot.sendMessage(chatId, motion, "");
145.       sendGambarTelegram();
146.       flashLed = !flashLed;
147.       digitalWrite(FLASH_LAMPU, flashLed);
148.     }
149.   }
```

Lines 162 – 191 encode the process of sending photos to the telegram bot. Image capture on the ESP32-CAM is then connected to the “api.telegram.org” domain. After that, you can send notifications to Telegram via the Telegram bot token that has been created.

```
162.
163.
164. String sendGambarTelegram(){
165.   const char* myDomain = "api.telegram.org";
166.   String getAll = "";
167.   String getBody = "";
168.
169.   camera_fb_t * fb = NULL;
170.   fb = esp_camera_fb_get();
171.   if(!fb) {
172.     Serial.println("Camera capture failed");
173.     delay(1000);
```

```
174.     ESP.restart();
175.     return "Camera capture failed";
176.   }
177.
178.   Serial.println("Connect to " + String(myDomain));
179.
180.   if (clientTCP.connect(myDomain, 443)) {
181.     Serial.println("Connection successful");
182.
183.     String head = "--RandomNerdTutorials\r\nContent-Disposition: form-
   data;    name=\"chat_id\";    \r\n\r\n"    +    chatId    +    "\r\n--
   RandomNerdTutorials\r\nContent-Disposition: form-data; name=\"photo\";
   filename=\"esp32-cam.jpg\"\r\nContent-Type: image/jpeg\r\n\r\n";
184.     String tail = "\r\n--RandomNerdTutorials--\r\n";
185.
186.     uint16_t imageLen = fb->len;
187.     uint16_t extraLen = head.length() + tail.length();
188.     uint16_t totalLen = imageLen + extraLen;
189.
190.     clientTCP.println("POST /bot"+BOTtoken+"/sendPhoto HTTP/1.1");
191.     clientTCP.println("Host: " + String(myDomain));}
```

5.2 Testing

This study uses the RnD or Research and Development method to develop a tool and test the effectiveness of the tool. So in this study several experiments were carried out with different lighting. The test was carried out 10 times for bright and low light places.

Table 5.1 : Test scenario in brigt places

Distance (meters)	**System Response**			**Buzzer Response Time (second)**	**Phone Response Time(second)**
	Detection	***Notification***	***Photo Notification***		
48	detected	Yes	succeed	4.16	16.85
58	detected	Yes	succeed	2.45	12.07
65	detected	Yes	succeed	2.57	13.88
84	detected	Yes	succeed	5.22	16.46
120	detected	Yes	succeed	5.55	13.11
166	detected	Yes	succeed	6.48	14.77
185	detected	Yes	succeed	8.26	18.30
206	detected	Yes	succeed	7.36	20.48
270	detected	Yes	succeed	7.33	15.56
300	detected	Yes	succeed	9.74	19.80
330	detected	Yes	succeed	10.87	21.33
400	detected	Yes	succeed	12.90	22.41

430	not detected	No	not successful	X	X
Average				**6.90**	**17.08**

Table 5.1 shows that the maximum distance the tool can work to detect motion in bright places is about 400 cm. This is because at a distance of more than 400 cm the tool does not detect the movement that occurs.

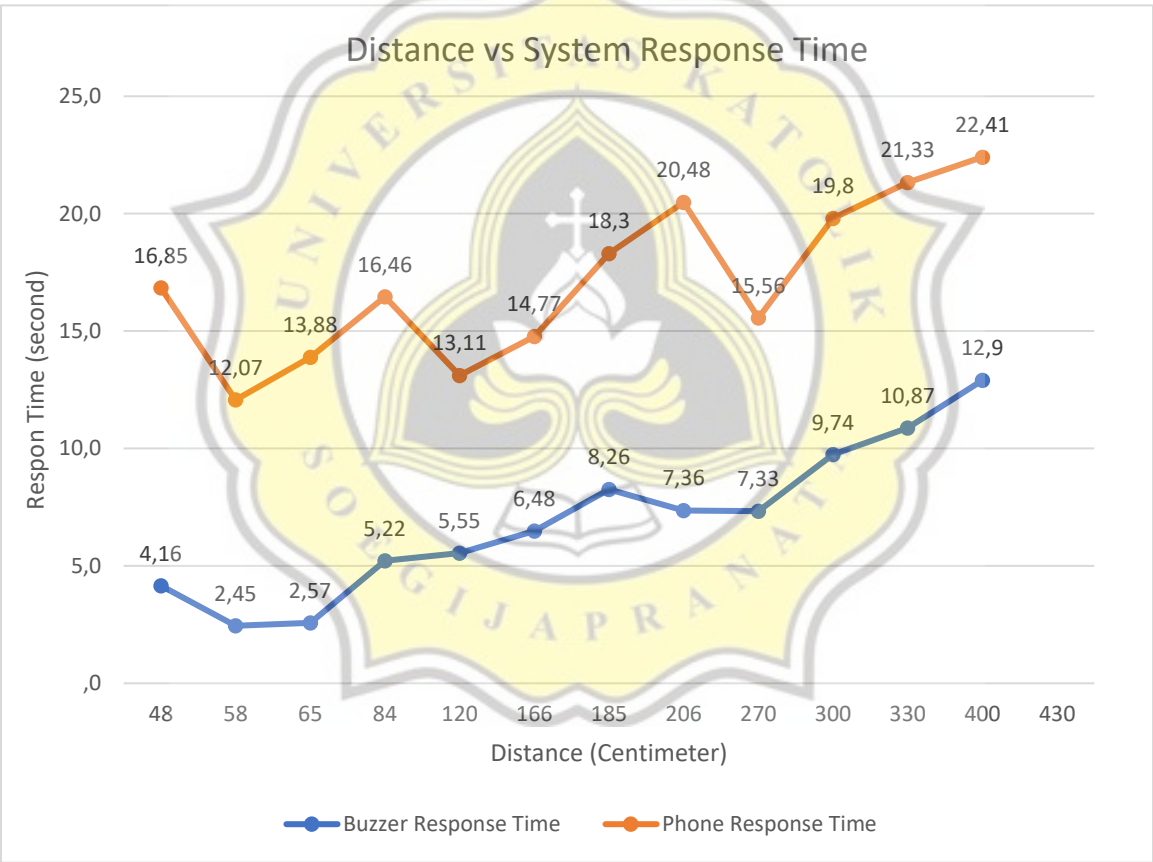

Figure 5.1. System response in bright place

From the graph above shows that the system can work well in bright places, it can be seen from the time the alarm sounds until the time it receives a text notification to the owner of the telegram house, the fastest results are obtained when 0.5-1 meters.

Table 5.2 : Test scenario in low light

Distance (meters)	**System Response**			**Buzzer Response Time (second)**	**Phone Response Time(second)**
	Detection	***Notification***	***Photo Notification***		
20	detected	Yes	succeed	7.11	13.03
60	detected	Yes	succeed	4.46	15.94
100	detected	Yes	succeed	5.47	18.82
150	detected	Yes	succeed	6,26	17.14
200	detected	Yes	succeed	9.77	21.11
240	detected	Yes	succeed	7.35	15.04
310	detected	Yes	succeed	7.64	12.22
350	detected	Yes	succeed	7.50	13.55
380	detected	Yes	succeed	7.54	13.77
400	detected	Yes	succeed	9.68	22.50
430	not detected	No	not successful	X	X
Average				**7,27**	**16.31**

Table 5.2 above shows that the tool can detect well at a distance of approximately 400 cm in low light conditions. This test condition is carried out from a distance of 0.2-4 meters. This is proven because at a distance of more than 4 meters the tool does not detect it. It can be seen when the object is within the range of the sensor and the camera successfully detects it, a notification message is sent to the user. Telegram Messenger users can also receive results on connected telegrams. The average buzzer time is 7.27 seconds and the average first notification received by the homeowner is around 16.31 seconds for low light.

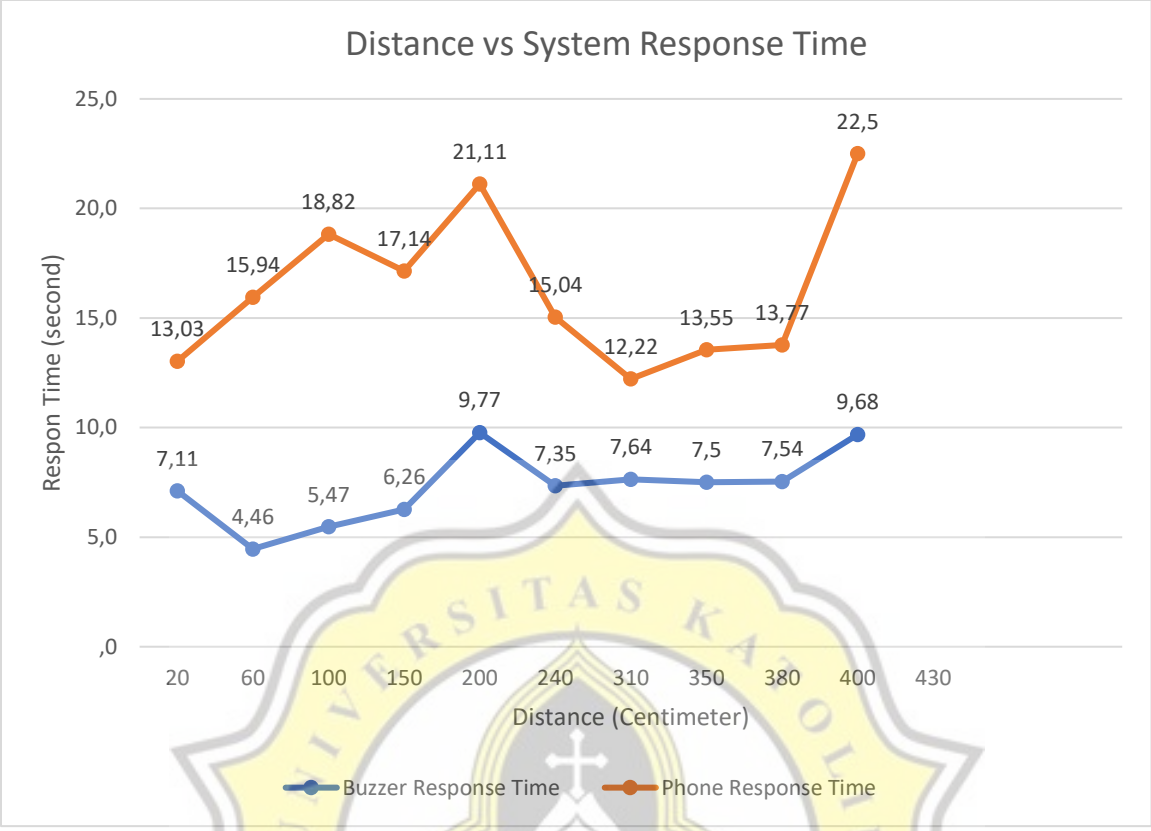

Figure 5.2. System response in low light

Testing of the entire security system is carried out when the sensor detects human movement, the buzzer turns on and when it sends a notification to the user's Telegram. The test conditions were carried out from a distance of 0.2-4 meters. A good response time by the sensor and providing the furthest distance is approximately 4 meters. From the results of the system response test, seen from receiving text notifications to the homeowner's telegram, the fastest results were obtained when they were 0.5-1 meters away.

From the table and graph data, it can be concluded that the system can respond well to movement in bright and low light places. For the average delivery time, the test results are 6.90 for sufficient light and 7.27 for low light. From these results, it can be seen that the delay for bright light is smaller than in low light. The distance test above is done so that the sensor can detect movement and the camera can take pictures from a predetermined distance.

Table 5.3 : Test scenario angle

Object Position Angle	Test Result
30^o	not detected
45^o	detected
60^o	detected
90^o	detected
120^o	detected
150^o	not detected

In the table above, tests are carried out to determine the optimal angle of the PIR sensor in the safety system to detect movement. This shows that humans will traverse the system by forming various angles. The distance of this test is carried out with objects that pass as far as 1-2 meters. This distance is still within the range of the PIR sensor and indicates that the detected object is not always the same at the end of the corner. the table and graph data, it can be concluded that the system can respond well to movement in bright and low light places.