DAFTAR PUSTAKA

[1] S. Sadi, "Sistem Keamanan Buka Tutup Kunci Brankas Menggunakan Bluetooth Hc – 05 Berbasis Arduino Mega 2560," *J. Tek.*, vol. 6, no. 2, pp. 1–9, 2017, doi: 10.31000/jt.v6i2.457.

[2] Y. Irawan, R. Wahyuni, D. Rahmawati, and H. T. Saputra, "Sistem Keamanan Smart Brankas Menggunakan Fingerprint Android," *J. Jar. Sist. Inf. Robot.*, vol. 6, no. 1, pp. 14–19, 2022, [Online]. Available: http://ojsamik.amikmitragama.ac.id

[3] A. Fauzan, N. Rofi, S. Fahrezi, B. S. Hadi, and M. Humam, "Sistem keamanan brankas berbasis raspberry pi dengan metode *face recognition*".

[4] M. D. Wuryandari and I. Afrianto, "Backpropagation Dan Learning Vector Quantization Program Studi Teknik Informatika Jurnal Komputer dan Informatika (KOMPUTA)," *Komputa*, vol. 1, pp. 45–51, 2012.

[5] T. Susim and C. Darujati, "Pengolahan Citra untuk Pengenalan Wajah (*Face Recognition*) Menggunakan OpenCV," *J. Syntax Admiration*, vol. 2, no. 3, pp. 534–545, 2021, doi: 10.46799/jsa.v2i3.202.

[6] M. W. Septyanto, H. Sofyan, H. Jayadianti, O. S. Simanjuntak, and D. B. Prasetyo, "Aplikasi Presensi Pengenalan Wajah Dengan Menggunakan Algoritma Haar Cascade Classifier," *Telematika*, vol. 16, no. 2, p. 87, 2020, doi: 10.31315/telematika.v16i2.3182.

[7] S. Ratna, "Pengolahan Citra Digital Dan Histogram Dengan

Phyton Dan Text Editor Phycharm," *Technol. J. Ilm.*, vol. 11, no. 3, p. 181, 2020, doi: 10.31602/tji.v11i3.3294.

[8] M. Zufar, "Introductory Computer Vision and Image Processing," *Sens. Rev.*, vol. 18, no. 3, pp. 2–4, 1998, doi: 10.1108/sr.1998.08718cae.001.

[9] Budi Utami Fahnun and Reza Pangestu, "Sistem Remote Kontrol Pada Robot Mobil Via Web Berbasis Raspberry Pi," *J. Ilm. Tek.*, vol. 1, no. 2, pp. 143–153, 2022, doi: 10.56127/juit.v1i2.204.

[10] R. Pi, B. Girsang, I. D. A. M, M. Ak, B. Apriyanto, and M. Si, "Automasi Pencatatan Hasil Timbangan Dengan Menggunakan," no. 5.

[11] A. H. M. Nasution, S. Indriani, N. Fadhilah, C. Arifin, and S. P. Tamba, "Pengontrolan Lampu Jarak Jauh Dengan Nodemcu Menggunakan Blynk," *J. TEKINKOM*, vol. 2, pp. 93–98, 2019.

[12] R. P. Pratama, "APLIKASI WEBSEaRVER ESP8266 UNTUK PENGENDALI PERALATAN LISTRIK," *INVOTEK J. Inov. Vokasional dan Teknol.*, vol. 17, no. 2, pp. 39–44, 2017, doi: 10.24036/invotek.v17i2.87.

[13] U. Figa, "Prototype Sistem Keamanan Pintu Menggunakan Radio Frequency Identification (Rfid) Dengan Kata Sandi Berbasis Mikrokontroler," *J. Coding, Sist. Komput. Untan*, vol. 03, no. 1, pp. 30–40, 2015.

[14] F. Wazir, "RABIT : Jurnal Teknologi dan Sistem Informasi Univrab Volume 1 No . 2 | Juli 2016 : 61-77 ISSN CETAK :

2477-2062 ISSN ONLINE : 2502-891X RANCANG BANGUN SISTEM PENGENALAN WAJAH DENGAN METODE RABIT : Jurnal Teknologi dan Sistem Informasi Univrab Volume 1 No," *J. Teknol. dan Sist. Inf. Univrab*, vol. 1, no. 2, pp. 61–77, 2016, [Online]. Available: http://jurnal.univrab.ac.id/index.php/rabit/article/view/23/10

[15] R. Suwartika and G. Sembada, "Perancangan Sistem Keamanan Menggunakan Solenoid Door Lock Berbasis Arduino Uno pada Pintu Laboratorium di PT. XYZ," *J. E-Komtek*, vol. 4, no. 1, pp. 62–74, 2020, doi: 10.37339/e-komtek.v4i1.217.

[16] R. Mardiati, F. Ashadi, and G. F. Sugihara, "Rancang Bangun Prototipe Sistem Peringatan Jarak Aman pada Kendaraan Roda Empat Berbasis Mikrokontroler ATMEGA32," *TELKA - Telekomun. Elektron. Komputasi dan Kontrol*, vol. 2, no. 1, pp. 53–61, 2016, doi: 10.15575/telka.v2n1.53-61.

[17] H. Soeroso *et al.*, "Penggunaan Bot Telegram Sebagai Announcement Systempada Intansi Pendidikan," *Semin. Master PPNS*, vol. Vol 2 No 1, pp. 45–48, 2017.

~ Halaman Ini Sengaja Dikosongkan ~

LAMPIRAN

A. Listing Program

1. Program Raspberry Pi 4B

```
import time
import datetime
import os
import cv2
import telepot
from telepot.loop import MessageLoop
import pickle
import RPi.GPIO as GPIO
from imutils.video import VideoStream, FPS
import face_recognition
import imutils
from pad4pi import rpi_gpio

# GPIO setup for the relay and buzzer
RELAY = 17
BUZZER = 18  # Define the GPIO pin for the buzzer
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(RELAY, GPIO.OUT)
GPIO.setup(BUZZER, GPIO.OUT)
GPIO.output(RELAY, GPIO.LOW)
GPIO.output(BUZZER, GPIO.LOW)

# GPIO setup for the keypad
KEYPAD = [
    [1, 2, 3, 'A'],
    [4, 5, 6, 'B'],
    [7, 8, 9, 'C'],
    ['*', 0, '#', 'D']
]
ROW_PINS = [5, 6, 13, 19]  # BCM numbering
```

```
COL_PINS = [12, 16, 20, 21]  # BCM numbering

factory = rpi_gpio.KeypadFactory()
keypad = factory.create_keypad(keypad=KEYPAD,
row_pins=ROW_PINS, col_pins=COL_PINS)

chat_id = 5992431594  # Initialize chat_id as a known value

code_entered = ""
correct_code_file = "correct_code.txt"
correct_code = ""
unlock_code = "A1234B"
wrong_attempts = 0
is_blocked = False
reset_mode = False

# Load correct code from file
def load_correct_code():
    global correct_code
    try:
        with open(correct_code_file, "r") as file:
            correct_code = file.read().strip()
    except FileNotFoundError:
        correct_code = "1234"  # Default correct code

# Save correct code to file
def save_correct_code():
    global correct_code
    with open(correct_code_file, "w") as file:
        file.write(correct_code)

# Load the correct code at the start of the program
load_correct_code()

# Function to initialize VideoStream
```

```
def initialize_videostream():
    for i in range(4):  # Try camera indices 0 to 3
        try:
            vs = VideoStream(src=i).start()
            time.sleep(2.0)  # Allow camera sensor to warm up
            frame = vs.read()
            if frame is not None:
                print(f"Camera index {i} initialized successfully.")
                return vs
            vs.stop()
        except:
            print(f"Camera index {i} not available.")
    raise Exception("No available camera found.")

# Initialize VideoStream globally
try:
    vs = initialize_videostream()
except Exception as e:
    print(f"Failed to initialize VideoStream: {e}")
    vs = None

# Function to control the buzzer
def buzz(times, duration=0.1):
    for _ in range(times):
        GPIO.output(BUZZER, GPIO.HIGH)
        time.sleep(duration)
        GPIO.output(BUZZER, GPIO.LOW)
        time.sleep(duration)

def process_key(key):
    global code_entered, chat_id, wrong_attempts, is_blocked,
reset_mode, correct_code
    print(f"Key pressed: {key}")
    buzz(1)  # Buzz once for each key press

    if is_blocked:
```

```
        code_entered += str(key)
        if code_entered == unlock_code:
            print("Unlock code entered. Access restored.")
            is_blocked = False
            wrong_attempts = 0
            code_entered = ""
        elif len(code_entered) >= len(unlock_code):
            print("Incorrect unlock code.")
            code_entered = ""
        return

    if key == 'D':
        print("Resetting code.")
        reset_mode = True
        code_entered = ""
        return

    if reset_mode:
        code_entered += str(key)
        if len(code_entered) >= 4:
            print(f"New correct code set to: {code_entered}")
            correct_code = code_entered
            save_correct_code()
            buzz(4)  # Buzz 4 times for reset code success
            reset_mode = False
            code_entered = ""
        return

    code_entered += str(key)

    if len(code_entered) == 4:
        if code_entered == correct_code:
            if not reset_mode and not is_blocked:
                print("Correct code entered. Starting face recognition.")
                start_face_recognition()
                wrong_attempts = 0
```

```
            else:
                print("System is in reset mode or blocked. Access
denied.")
        else:
            print("Incorrect code. Access denied.")
            wrong_attempts += 1
            if wrong_attempts == 1:
                buzz(4)  # Buzz 4 times for the first incorrect attempt
            elif wrong_attempts >= 3:
                print("Too many wrong attempts. System blocked.")
                is_blocked = True
                buzz(10)  # Buzz 10 times for blocking the system
            image_path = '/home/pi/Desktop/unknown_capture.jpg'
            capture_image_from_stream(image_path, vs)
            timestamp = datetime.datetime.now().strftime("%Y-%m-
%d %H:%M:%S")
            try:
                telegram_bot.sendPhoto(chat_id,
photo=open(image_path, 'rb'), caption=f"Keypad pressed by
Unknown at {timestamp}")
                print(f"Photo sent to Telegram: {image_path}")
            except Exception as e:
                print(f"Failed to send photo to Telegram: {e}")
        code_entered = ""

keypad.registerKeyPressHandler(process_key)

# Get the current time
now = datetime.datetime.now()

# Function to capture an image from the webcam
def capture_image(file_path):
    cap = cv2.VideoCapture(0)
    ret, frame = cap.read()
    if ret:
        cv2.imwrite(file_path, frame)
```

```
    cap.release()

# Function to find the latest image in a directory
def find_latest_image(directory):
    files = [os.path.join(directory, f) for f in os.listdir(directory) if
f.endswith(('.jpg', '.jpeg', '.png'))]
    latest_file = max(files, key=os.path.getmtime)
    return latest_file

# Function to send photo to Telegram bot
def send_photo_to_telegram(photo_path, caption):
    try:
        bot.sendPhoto(chat_id='5992431594',
photo=open(photo_path, 'rb'), caption=caption)
        print(f"Photo sent to Telegram: {photo_path}")
    except Exception as e:
        print(f"Failed to send photo to Telegram: {e}")

# Function to capture an image from the existing video stream
def capture_image_from_stream(file_path, video_stream):
    try:
        frame = video_stream.read()
        os.makedirs(os.path.dirname(file_path), exist_ok=True)
        cv2.imwrite(file_path, frame)
        print(f"Image captured from stream and saved to
{file_path}")
    except Exception as e:
        print(f"Error capturing image from stream: {e}")

# Function to handle incoming messages
def action(msg):
    global chat_id
    chat_id = msg['chat']['id']  # Extract chat_id from the incoming
message
    command = msg['text']  # Extract the text of the command
    print('Received: %s' % command)
```

```
    # Respond to different commands
    if command == '/hi':
        telegram_bot.sendMessage(chat_id, str("Hi Ziddan, How are
you today?"))
    elif command == '/time':
        telegram_bot.sendMessage(chat_id, str(now.hour) + str(":")
+ str(now.minute))
    elif command == '/capture':
        image_path = '/home/pi/Desktop/captured_image.jpg'
        capture_image_from_stream(image_path, vs)
        telegram_bot.sendPhoto(chat_id, photo=open(image_path,
'rb'))
    elif command.startswith('/keypad '):
        keypad_code = command.split(' ')[1]
        if keypad_code == correct_code:
            telegram_bot.sendMessage(chat_id, "Keypad code
accepted")
        else:
            # Capture image and send with "Unknown" message
            image_path = '/home/pi/Desktop/unknown_capture.jpg'
            capture_image_from_stream(image_path, vs)
            timestamp = datetime.datetime.now().strftime("%Y-%m-
%d %H:%M:%S")
            try:
                telegram_bot.sendPhoto(chat_id,
photo=open(image_path, 'rb'), caption=f"Keypad pressed by
Unknown at {timestamp}")
                print(f"Photo sent to Telegram: {image_path}")
            except Exception as e:
                print(f"Failed to send photo to Telegram: {e}")

# Function to start face recognition
def start_face_recognition():
    print("[INFO] loading encodings + face detector...")
    encodingsP = "encodings.pickle"
```

```
    data = pickle.loads(open(encodingsP, "rb").read())
    cascade = "haarcascade_frontalface_default.xml"
    detector = cv2.CascadeClassifier(cascade)

    print("[INFO] starting video stream...")
    fps = FPS().start()

    prevTime = 0
    doorUnlock = False

    while True:
        frame = vs.read()
        frame = imutils.resize(frame, width=700)
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

        rects = detector.detectMultiScale(gray, scaleFactor=1.1,
minNeighbors=5, minSize=(30, 30),
flags=cv2.CASCADE_SCALE_IMAGE)
        boxes = [(y, x + w, y + h, x) for (x, y, w, h) in rects]
        encodings = face_recognition.face_encodings(rgb, boxes)
        names = []

        for encoding in encodings:
            matches =
face_recognition.compare_faces(data["encodings"], encoding)
            name = "Unknown"

            if True in matches:
                matchedIdxs = [i for (i, b) in enumerate(matches) if b]
                counts = {}

                for i in matchedIdxs:
                    name = data["names"][i]
                    counts[name] = counts.get(name, 0) + 1
```

```
            name = max(counts, key=counts.get)
            print(f"Recognized: {name}")

            GPIO.output(RELAY, GPIO.HIGH)
            time.sleep(5)
            prevTime = time.time()
            doorUnlock = True
            print("door unlock")
            # Capture the image
            image_path =
'/home/pi/facial_recognition/unlock_capture.jpg'
            capture_image_from_stream(image_path, vs)
            timestamp = datetime.datetime.now().strftime("%Y-
%m-%d %H:%M:%S")
                # Send the captured image to Telegram
            try:
                telegram_bot.sendPhoto(chat_id,
photo=open(image_path, 'rb'), caption=f"Door unlocked for
{name}  at {timestamp}")
                print(f"Photo sent to Telegram: {image_path}")
            except Exception as e:
                print(f"Failed to send photo to Telegram: {e}")


        else:
            # Capture image and send with "Unknown" message
            image_path = '/home/pi/Desktop/unknown_capture.jpg'
            capture_image_from_stream(image_path, vs)
            timestamp = datetime.datetime.now().strftime("%Y-
%m-%d %H:%M:%S")
            try:
                telegram_bot.sendPhoto(chat_id,
photo=open(image_path, 'rb'), caption=f"Door locked for
Unknown at {timestamp}")
                print(f"Photo sent to Telegram: {image_path}")
            except Exception as e:
```

```
                print(f"Failed to send photo to Telegram: {e}")

            names.append(name)

        if doorUnlock and time.time() - prevTime > 10:
            doorUnlock = False
            GPIO.output(RELAY, GPIO.LOW)
            print("door lock")
            break

        for ((top, right, bottom, left), name) in zip(boxes, names):
            cv2.rectangle(frame, (left, top), (right, bottom), (0, 255,
0), 2)
            y = top - 15 if top - 15 > 15 else top + 15
            cv2.putText(frame, name, (left, y),
cv2.FONT_HERSHEY_SIMPLEX, .8, (0, 255, 255), 2)

            if name == "Unknown":
                GPIO.output(RELAY, GPIO.LOW)
                print("Unknown face detected. Door locked.")

        cv2.imshow("Facial Recognition is Running", frame)
        key = cv2.waitKey(1) & 0xFF

        if key == ord("q"):
            break

        fps.update()

    fps.stop()
    print("[INFO] elapsed time: {:.2f}".format(fps.elapsed()))
    print("[INFO] approx. FPS: {:.2f}".format(fps.fps()))

    cv2.destroyAllWindows()

if __name__ == "__main__":
```

```
    print("Enter the 4-digit code on the keypad.")
    telegram_bot =
telepot.Bot('7163580664:AAFZbEwyG4JSDVmf1dBW8Tw71fH
4KfuI5JY')  # Replace with your actual bot API key

    MessageLoop(telegram_bot, action).run_as_thread()
    print('Let's have a chat session...')

    try:
        while True:
            time.sleep(1)
    except KeyboardInterrupt:
        print("Program stopped by User")
        GPIO.cleanup()
        vs.stop()
```

B. Desain Alat

BIODATA PENULIS

Nama	: Ziddan Fahrezky
Tempat/Tanggal Lahir	: Cilacap, 20 Maret 2003
Kelas	: Teknik Elektronika B (Angkatan 2021)
E-mail	: zfahrezky@gmail.com
No. telp	: +6282133244378
Hobi	: Bermimpi
Moto	: Kunci sukses adalah TANI (Tawakal, Amanah, Niatnya Ibadah

Riwayat Pendidikan

1. SD Negeri 4 Kranji Purwokerto — Tahun 2009 – 2011
2. SD Petra Mandiri Cilacap — Tahun 2011 – 2015
3. SMP Negeri 8 Cilacap — Tahun 2015 – 2018
4. SMK Boedi Oetomo Cilacap
   Jurusan Teknik Instalasi Tenaga Listrik — Tahun 2018 – 2021
5. Politeknik Negeri Cilacap
   Prodi D3 - Teknik Elektronika — Tahun 2021 – 2024