LAMPIRAN

UMKT
Program Studi
Teknik Informatika
Fakultas Sains dan Teknologi
Kalimantan Timur
Telp. 0541-748511 Fax. 0541-766832
Website http://informatika.umkt.ac.id
email: informatika@umkt.ac.id

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

Nomor : 056-001/KET/FST.1/A/2024
Lampiran : -
Perihal : **Keterangan Melakukan Penelitian**

Assalamu'alaikum Warrahmatullahi Wabarrakatuh

Puji Syukur kepada Allah Subhanahu wa ta'ala yang senantiasa melimpahkan Rahmat-Nya kepada kita sekalian. Amin.

Dengan surat ini, kami menerangkan bahwa mahasiswa berikut:

No	Nama	NIM
1	Muhammad Fahri Alfianur	1911102441174
2	Rifat Fakhriy Naufal	2011102441041
3	Nur Anjeni Lestari	2011102441024
4	Ridha Anisa Soldzu Parnga	2011102441241
5	Bobli	2011102441069

Melakukan penelitian dengan membuat sebuah alat IoT di Laboratorium Hardware & Networking.

Demikian hal ini disampaikan, atas kerjasamanya kami ucapkan terima kasih.

Wassalamu'alaikum Warrahmatullahi Wabarrakatuh

Samarinda, 18 Dzulhijjah 1445 H
25 Juni 2024 M

Ketua Program Studi S1 Teknik Informatika

Arbansyah, S.Kom., M.TI
NIDN. 1118019203

L.1 Surat Izin Penelitian

KARTU KENDALI BIMBINGAN SKRIPSI

Nama Mahasiswa : Bobli

NIM : 2011102441069

Nama Dosen Pembimbing : Arbansyah, S.Kom, M.T.I.

Judul Penelitian : Implementasi Sistem Keamanan Cerdas Untuk Pencegahan Pencurian Motor Dengan Integritasi ESP32-CAM Dan Sensor Getar

No	Tanggal	Uraian Pembimbingan	Paraf Dosen
1	25 Januari 2024	Pengajuan judul Skripsi	
2	29 Januari	Format Canvas	
3	25 Februari 2024	format pembuatan bab 1 & latarbelakang	
4	20 maret 2024	pembahasan & bab 2	
5	15 April 2024	Membuat Rangkaian flowchat & penulisan	
6	20 April 2024	Membuat skematik rangkaian alat	
7	25 April 2024	membuat Desain Visual	
8	26 April 2024	Uji tes alat	
10	25 Juni 2024	Membuat Program	
11	26 Jun. 2024	Uji tes Kegunaan alat	
12			
13			
14			

Mengetahui,

Dosen Pembimbing

Arbansyah, S.Kom, M.Ti
NIDN. 1118019203

Program Studi

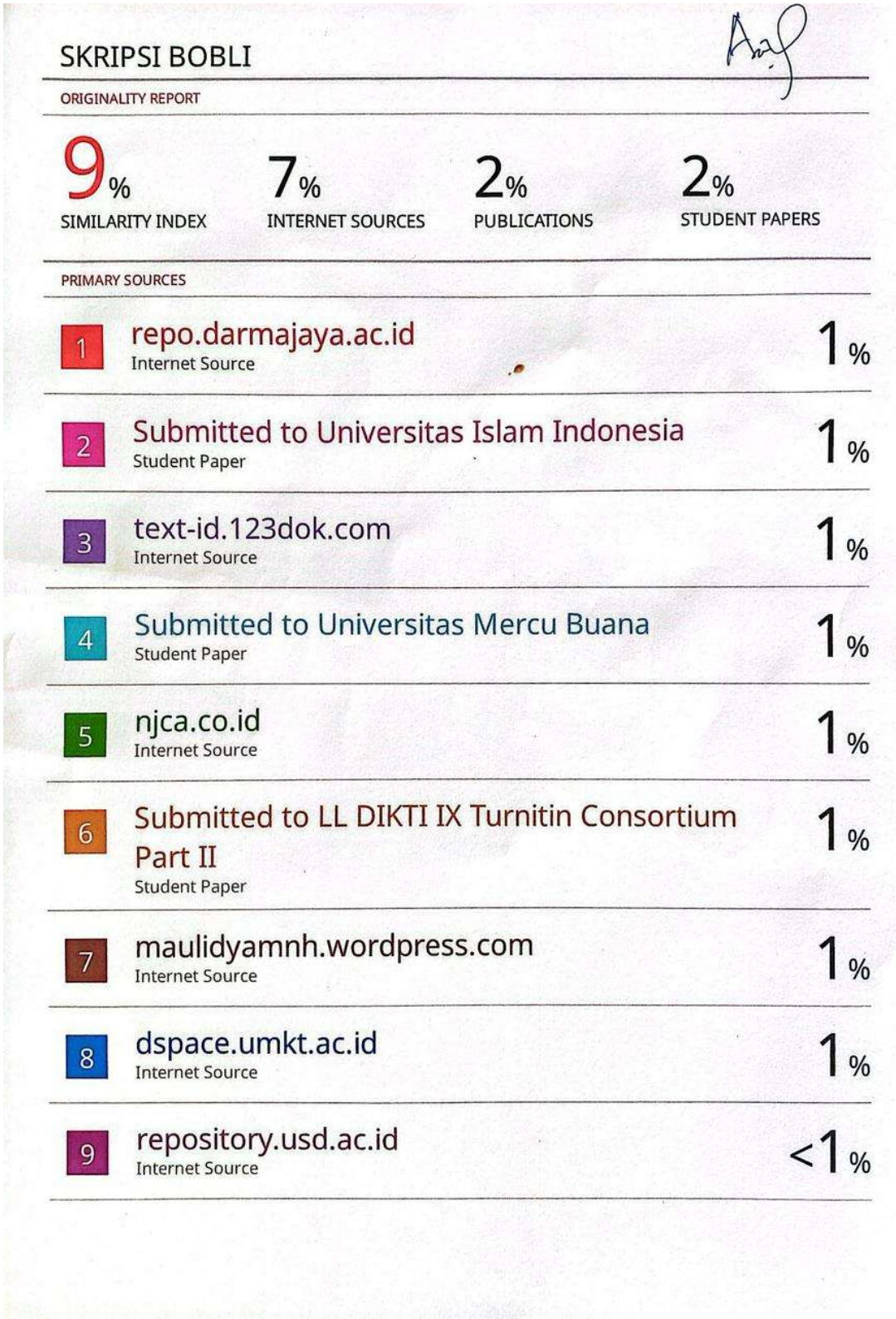
Arbansyah, S.Kom, M.Ti
NIDN. 1118019203

L.2 Kartu Kendali Bimbingan Skripsi

RIWAYAT HIDUP

Nama saya Bobli, lahir pada tanggal 27 juni 2002 di Desa Senyiur, saya memulai pendidikan dasar di SDN 001 Senyiur dari tahun 2008 sampai 2014, kemudian melanjutkan pendidikan menengah pertama di SMPN 004 Senyiur dari tahun 2014 sampai 2017. Setelah itu, saya menempuh pendidikan menengah atas di SMAN 002 Senyiur dari tahun 2017 sampai dengan 2020. Saat ini, saya sedang menyelesaikan studi di Universitas Muhammadiyah Kalimantan Timur . Selama masa kuliah, saya pernah melakukan magang di Universitas Muhamadiyah Kalimantan Timur selama 3 bulan. Pengalaman magang tersebut memberikan saya banyak pengetahuan praktis yang endukung teori yang saya pelajari di bangku kuliah. Penulis sangat mengharapkan kritik dan saran yang membangun dari para pembaca mengenai skripsi ini. Semoga karya ini dapat memberikan manfaat dan kontribusi positif bagi perkembangan ilmu pengetahuan dan teknologi. Terima kasih.

HASIL TURNITIN



SKRIPSI BOBLI

ORIGINALITY REPORT

9%	7%	2%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	repo.darmajaya.ac.id Internet Source	1%
2	Submitted to Universitas Islam Indonesia Student Paper	1%
3	text-id.123dok.com Internet Source	1%
4	Submitted to Universitas Mercu Buana Student Paper	1%
5	njca.co.id Internet Source	1%
6	Submitted to LL DIKTI IX Turnitin Consortium Part II Student Paper	1%
7	maulidyamnh.wordpress.com Internet Source	1%
8	dspace.umkt.ac.id Internet Source	1%
9	repository.usd.ac.id Internet Source	<1%

CODE PROGRAM

```
#include <Arduino.h>
#include <WiFi.h>
#include <WiFiClientSecure.h>
#include "soc/soc.h"
#include "soc/rtc_cntl_reg.h"
#include "esp_camera.h"
#include <UniversalTelegramBot.h>
#include <ArduinoJson.h>

// WiFi credentials
const char* ssid = "Bobli";
const char* password = "apamserawa";

// Telegram Bot Token and Chat ID
String BOTtoken = "7442583602:AAHuLiRU-hAuJbHII1KN8jmhF-uyYvUzxR0";
String CHAT_ID = "6842233029";

// Telegram and WiFi client initialization
WiFiClientSecure clientTCP;
UniversalTelegramBot bot(BOTtoken, clientTCP);

// Global variables
bool sendPhoto = false;
int botRequestDelay = 1000;
unsigned long lastTimeBotRan = 0;
unsigned long lastPhotoTime = 0;
const unsigned long photoInterval = 10000; // Minimum time between photos

// Camera pins definition
#define PWDN_GPIO_NUM     32
#define RESET_GPIO_NUM    -1
#define XCLK_GPIO_NUM      0
#define SIOD_GPIO_NUM     26
```

```
#define SIOC_GPIO_NUM     27
#define Y9_GPIO_NUM       35
#define Y8_GPIO_NUM       34
#define Y7_GPIO_NUM       39
#define Y6_GPIO_NUM       36
#define Y5_GPIO_NUM       21
#define Y4_GPIO_NUM       19
#define Y3_GPIO_NUM       18
#define Y2_GPIO_NUM        5
#define VSYNC_GPIO_NUM    25
#define HREF_GPIO_NUM     23
#define PCLK_GPIO_NUM     22

// Vibration sensor pin definition
#define VIBRATION_SENSOR_PIN 13

// Vibration thresholds
const int vibrationThresholdLow = 1;
const int vibrationThresholdMedium = 2;
const int vibrationThresholdHigh = 3;

void handleNewMessages(int numNewMessages) {
  for (int i = 0; i < numNewMessages; i++) {
    String chat_id = String(bot.messages[i].chat_id);
    String text = bot.messages[i].text;
    String from_name = bot.messages[i].from_name;

    if (text == "/start") {
      String welcome = "Welcome , " + from_name + "\n";
      bot.sendMessage(chat_id, welcome, "");
    } else if (text == "/photo") {
      sendPhoto = true;
      bot.sendMessage(chat_id, "New photo request received.");
    }
  }
```

```
}

String sendPhotoTelegram() {
  const char* myDomain = "api.telegram.org";
  String getAll = "";
  String getBody = "";
  digitalWrite(4, HIGH);
  camera_fb_t * fb = esp_camera_fb_get();
  delay(200);
  digitalWrite(4, LOW);

  if (!fb) {
    Serial.println("Camera capture failed");
    delay(1000);
    ESP.restart();
    return "Camera capture failed";
  }

  if (clientTCP.connect(myDomain, 443)) {
    String head = "--RandomNerdTutorials\r\nContent-Disposition: form-data; name=\"chat_id\"; \r\n\r\n" + CHAT_ID + "\r\n--RandomNerdTutorials\r\nContent-Disposition: form-data; name=\"photo\"; filename=\"esp32-cam.jpg\"\r\nContent-Type: image/jpeg\r\n\r\n";
    String tail = "\r\n--RandomNerdTutorials--\r\n";

    uint16_t imageLen = fb->len;
    uint16_t extraLen = head.length() + tail.length();
    uint16_t totalLen = imageLen + extraLen;

    clientTCP.println("POST /bot" + BOTtoken + "/sendPhoto HTTP/1.1");
    clientTCP.println("Host: " + String(myDomain));
    clientTCP.println("Content-Length: " + String(totalLen));
    clientTCP.println("Content-Type: multipart/form-data; boundary=RandomNerdTutorials");
    clientTCP.println();
    clientTCP.print(head);
```

```
  uint8_t *fbBuf = fb->buf;
  size_t fbLen = fb->len;
  for (size_t n = 0; n < fbLen; n += 1024) {
    size_t chunkSize = (n + 1024 < fbLen) ? 1024 : fbLen % 1024;
    clientTCP.write(fbBuf, chunkSize);
    fbBuf += chunkSize;
  }

  clientTCP.print(tail);
  esp_camera_fb_return(fb);

  int waitTime = 10000;
  long startTimer = millis();
  boolean state = false;

  while ((startTimer + waitTime) > millis()) {
    delay(100);
    while (clientTCP.available()) {
      char c = clientTCP.read();
      if (state) getBody += String(c);
      if (c == '\n') {
        if (getAll.length() == 0) state = true;
        getAll = "";
      } else if (c != '\r') getAll += String(c);
      startTimer = millis();
    }
    if (getBody.length() > 0) break;
  }
  clientTCP.stop();
  Serial.println(getBody);
 } else {
  getBody = "Connection to api.telegram.org failed.";
  Serial.println(getBody);
 }
 return getBody;
```

```
}

void setup() {
  WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0);
  Serial.begin(115200);
  pinMode(VIBRATION_SENSOR_PIN, INPUT);
  pinMode(4, OUTPUT);
  digitalWrite(4, LOW);

  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  clientTCP.setCACert(TELEGRAM_CERTIFICATE_ROOT);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nWiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());

  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
```

```
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sscb_sda = SIOD_GPIO_NUM;
  config.pin_sscb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
  config.xclk_freq_hz = 20000000;
  config.pixel_format = PIXFORMAT_JPEG;

  if (psramFound()) {
    config.frame_size = FRAMESIZE_UXGA;
    config.jpeg_quality = 10;
    config.fb_count = 2;
  } else {
    config.frame_size = FRAMESIZE_SVGA;
    config.jpeg_quality = 12;
    config.fb_count = 1;
  }

  if (esp_camera_init(&config) != ESP_OK) {
    Serial.printf("Camera init failed");
    delay(1000);
    ESP.restart();
  }

  sensor_t * s = esp_camera_sensor_get();
  s->set_framesize(s, FRAMESIZE_CIF);
}

void loop() {
  int vibrationLevel = analogRead(VIBRATION_SENSOR_PIN);

  if (vibrationLevel >= vibrationThresholdHigh && millis() - lastPhotoTime > photoInterval) {
    Serial.println("High vibration detected!");
    sendPhoto = true;
    lastPhotoTime = millis();
```

```
  } else if (vibrationLevel >= vibrationThresholdMedium && millis() - lastPhotoTime > photoInterval) {
    Serial.println("Medium vibration detected!");
    sendPhoto = true;
    lastPhotoTime = millis();
  } else if (vibrationLevel >= vibrationThresholdLow) {
    Serial.println("Low vibration detected but not triggering photo.");
  }

  if (sendPhoto) {
    bot.sendMessage(CHAT_ID, "Sending photo...");
    sendPhotoTelegram();
    sendPhoto = false;
  }

  if (millis() > lastTimeBotRan + botRequestDelay) {
    int numNewMessages = bot.getUpdates(bot.last_message_received + 1);
    while (numNewMessages) {
      handleNewMessages(numNewMessages);
      numNewMessages = bot.getUpdates(bot.last_message_received + 1);
    }
    lastTimeBotRan = millis();
  }
  delay(50);
}
```