DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

DAFTAR ARTI LAMBANG DAN SINGKATAN

Lambang/Singkatan	Arti dan Keterangan
RF	Radio Frequency
IoT	Internet of Things
MFA	Multi-Factor Authentication
RFID	Radio Frequency Identification
GPS	Global Positioning System
Hz	Satuan Frekuensi Hertz
TSMC	Taiwan Semiconductor Manufacturing Company
WIFI	Wireless Fidelity
PCB	Printed Circuit Board
CCTV(?)	Closed Circuit Television
BT	BlueTooth
BLE	Bluetooth Low Energy
SoC	System-on-Chip
CPU	Central Processing Unit
DMIPS	Dynamic Microscopic Image Processing Scanner
KB	KiloByte
SRAM	Static Random Access Memory
PSRAM	Pseudo-Static Random Access Memory
UART	Universal Asynchronous Receiver-Transmitter
SPI	Serial Peripheral Interface
I2C	Inter Integrated Circuit
PWM	Pulse Width Modulation
ADC	Analog to Digital Converter
DAC	Digital to Analog Converter

TF	TransFlash
STA	Station
AP	Access Point
FOTA	Firmware Over-The-Air
Lambang/Singkatan	**Arti dan Keterangan**
IDE	Integrated Development Environment
OS	Operating System
VDC	Satuan Tegangan Volt Listik Searah
mV	Satuan Tegangan milliVolt
mA	Satuan Arus milliAmpere
mm	Satuan Panjang milliMeter
ISO	International Organization for Standardization
IEC	International Electrotechnical Commission
UID	User Identification
NUID	New User Identification
IFF	Identification Friend or Foe
°C	Satuan Suhu Celcius
LCD	Liquid Crystal Display
IR	Instruction Register
DR	Data Register
BF	Busy Flag
AC	Address Counter
DDRAM	Display Data Random Access Memory
CGRAM	Character Generator Random Access Memory
CGROM	Character Generator Read Only Memory
ROM	Read Only Memory
ASCII	American Standard Code for Interchange Information
BaaS	Backend as a Service
SQL	Stuctured Query Language

noSQL	Not only SQL
JSON	Javascript Object Notation
PaaS	Platform as a Service
I/O	Input / Output
HTTP	Hypertext Tranfer Protocol

Lambang/Singkatan	**Arti dan Keterangan**
API	Application Programming Interface
CNN	Convolutional Neural Network
EM	Electromagnetic

BAB I

PENDAHULUAN

I.1. Latar Belakang

Tingkat kejahatan di Indonesia mengalami peningkatan tahun ke tahun. Jenisnya semakin beragam, ada spesialis pencuri kendaraan, pencurian toko, pencurian rumah. Beberapa metode dikembangkan untuk meningkatkan keamanan suatu tempat agar terhindar dari tindak pencurian. Salah satunya dengan memanfaatkan teknologi komputer.

Teknologi komputer pada saat ini berkembang dengan sangat pesatnya dan merupakan salah satu bidang yang mempunyai peran yang sangat penting dibeberapa aspek kehidupan manusia, termasuk pada bidang *security.* Saat ini telah banyak dikembangkan sebuah sistem pengamanan akses masuk ke sebuah rumah atau ruangan dengan beberapa verifikasi identitas, baik dengan menggunakan kunci, kartu, *password*, dan sebagainya. Namun metode ini masih memiliki kekurangan seperti keterbatasan manusia dalam mengingat benda dan kombinasi angka yang menyebabkan tidak dapatnya diakses pintu tersebut. Salah satu solusi untuk mengatasi kekurangan tersebut adalah memanfaatkan teknologi biometrik dalam melakukan identifikasi ataupun verifikasi.

Kehandalan sebuah sistem keamanan bergantung pada sistem autentikasi yang digunakan. Autentikasi tetap menjadi perlindungan mendasar terhadap akses tidak sah ke perangkat atau aplikasi sensitif lainnya, baik offline maupun online. Pada awalnya hanya satu faktor yang digunakan untuk melakukan autentikasi pada subjek. Contohnya ialah penggunaan password untuk mengonfirmasi kepemilikan

dari ID pengguna. Tapi, ini adalah level terlemah dari sistem autentikasi. Dengan membagikan password, seseorang langsung dapat mengakses akun tersebut. Oleh karena itu, Autentikasi Multi-Faktor diajukan untuk menyediakan keamanan dengan level tinggi untuk melindungi perangkat dari akses yang tidak sah dengan menggunakan lebih dari dua faktor pengenal [1].

Kevin Dwi Septian, Setia Juli Irzal Ismail, dan Anang Sularsa membuat prototipe sistem keamanan pintu berbasis *face recognition*. Sistem ini lebih fleksibel dan bersifat otomatis yang dapat mencegah tindak pencurian [2].

Sedangkan Banu Santoso dan Ryan Putranda Kristianto memanfaatkan *face recognition* untuk sistem presensi perkuliahan mahasiswa. Sehingga memudahkan pihak dosen dan staff kampus untuk memonitor kehadiran mahasiswa yang begitu banyak. Hal ini dapat mengurangi tingkat kecurangan dalam pengisian daftar presensi dan meningkatkan efektifitas pengolahan data mahasiswa [3].

Aji Shofiudin berhasil membuat sistem pengaman ganda pada sepeda motor menggunakan *Fingerprint Identification* dan *Remote Control RF* yang berbasis pada arduino. Sistem ini dapat mengunci stang motor tanpa menggunakan kunci, tapi menggunakan *remote control* sehingga bisa diakses dari jarak jauh [4].

Wahyu Kusuma Raharja dan Bagas Santoso memanfaatkan *Fingerprint Identification* dan *Internet of Things* untuk membuat alat monitoring keamanan ruangan. Sidik jari yang diidentifikasi menggunakan sensor sidik jari akan dikirim ke web database menggunakan jaringan internet. Hasil informasi rekam data pengguna ruangan akan ditampilkan pada website [5].

Cahya Rezky Prihatmoko membuat *Smart Hybrid Reader* yang beroperasi dengan dua mode yaitu menggunakan *smartcard* dan aplikasi android pada *smartphone* untuk keamanan ruangan. *Smartcard* yang dibaca oleh modul RFID akan dikirimkan ke database untuk diverifikasi [6].

Dari beberapa jurnal terkait maka pada skripsi ini akan dibuat *Smart Key Reader* dengan *Multi-Factor Authentication* (MFA) yang menggabungkan teknologi biometrik, *smartcard*, dan web browser. Sistem ini akan mengidentifikasi beberapa faktor autentikasi yang nantinya akan dilakukan proses verifikasi dan pencocokan antara *sample* dan *database*. Sistem dalam skripsi ini memiliki keunggulan dalam hal keamanan dan fleksibilitas untuk ruangan yang membutuhkan tingkat keamanan yang tinggi.

I.2. Rumusan Masalah

Berdasarkan uraian pada latar belakang, maka rumusan masalah pada skripsi ini adalah :

1. Bagaimana membuat sistem *reader* berbasis MFA dengan pengenal wajah, sidik jari, *smartcard*, dan *web browser*?
2. Bagaimana unjuk kerja akses pintu dengan menggunakan *smart key reader* berbasis MFA?

I.3. Tujuan dan Manfaat Penelitian

Tujuan yang ingin dicapai dari skripsi ini adalah :

1. Mampu membuat sistem *reader* berbasis MFA menggunakan teknologi biometrik, *smartcard*, dan *web browser*.

2. Mengetahui dan mengimplementasikan unjuk kerja akses pintu dengan menggunakan *smart key reader* berbasis MFA.

I.4. Manfaan Penelitian

Skripsi ini memiliki beberapa manfaat seperti yang diuraikan berikut ini :

1. Bagi masyarakat, mahasiswa, dan staf akademik skripsi ini diharapkan kedepannya dapat memberikan manfaat bagi siapa saja yang membutuhkan sistem keamanan ruangan yang tinggi.
2. Bagi institusi Universitas Hasanuddin, Departemen Teknik Elektro, dan pada bidang Teknologi Telekomunikasi dan Informasi, skripsi ini dapat berguna sebagai referensi ilmiah untuk pengembangan penelitian yang berhubungan dengan topik *smart reader* khususnya dalam pengembangan *smart reader* berbasis MFA dan teknologi biometrik.
3. Bagi peneliti, skripsi ini memiliki manfaat untuk menambah wawasan dan kemampuan penulis, serta menjadi sumber data dalam pembuatan *smart key reader* berbasis MFA.

I.5. Batasan Masalah

Dalam pengerjaan skripsi ini, perancangan yang akan dibuat dibatasi pada hal-hal sebagai berikut :

1. Sistem keamanan berbasis MFA untuk ruangan dibuat menggunakan ESP32 CAM, ESP32, sensor sidik jari AS608, modul RFID, dan Electromagnetik Lock Door.

2. Sistem keamanan berbasis MFA untuk ruangan diuji coba pada Computer & PABX Room Laboratorium Telekomunikasi, Radio, dan Gelombang Pendek.
3. Sistem pengenal wajah dibuat menggunakan library javascript face-api.
4. Pengambilan gambar wajah untuk sistem pengenal wajah dilakukan pada jarak 50cm dengan menggunakan cahaya flash dari modul esp32-cam.
5. *Web Server* di-*deploy* menggunakan layanan *Platform As A Service* (PaaS) HEROKU
6. Database menggunakan layanan dari Firebase.
7. Parameter kinerja yang dianalisis adalah respon Elektromagnetik Door Lock terhadap data wajah, sidik jari, dan *smartcard* yang diperoleh.

I.6. Metode Penelitian

Adapun metode penulisan yang digunakan pada skripsi ini guna menyelesaikan masalah, antara lain :

1. Studi Literatur

   Tahap awal dari penelitian ini yaitu mencari sumber-sumber referensi serta materi pendukung yang dijadikan sebagai acuan dalam penyelesaian skripsi dimana merujuk pada jurnal-jurnal nasional maupun internasional seperti yang tertera pada daftar tinjauan pustaka sehingga bisa dipelajari dalam pengerjaan dan penulisan skripsi sebelum melakukan implementasi dan pengujian secara langsung.

2. Desain dan Pembuatan Perangkat Keras

   Tahap kedua dari penelitian ini yaitu merakit dan mengintegrasikan komponen-komponen penyusun reader agar siap untuk diuji.

3. Desain dan Pembuatan Perangkat Lunak

   Tahap ketiga dari penelitian ini yaitu merancang dan membuat program yang akan dimasukkan dalam mikrokontroller dan web server untuk memproses, mengolah, dan menginterpretasikan data ke layar dan EM Door Lock.

4. Pengujian Sistem

   Tahap keempat dari penelitian ini yaitu melakukan pengujian untuk mengetahui kinerja alat yang telah dirancang baik dari perangkat lunak maupun perangkat keras agar sesuai dengan yang direncanakan.

5. Penyusunan Laporan

   Tahap terkahir dari penelitian ini yaitu menyusun laporan skripsi setelah tahapan-tahapan sebelumnya telah diselesaikan sehingga diperoleh hasil dari pembuatan sistem secara rinci dari beberapa pengujian yang telah dilakukan.

I.7. Sistematika Penulisan

BAB I PENDAHULUAN

Bab ini menguraikan tentang Latar Belakang, Tujuan dan Manfaat, Perumusan Masalah, Batasan Masalah, dan Sistematika Penulisan mengenai *Smart Key Reader* dengan *Multi-Factor Authentication*.

BAB II LANDASAN TEORI

Bab ini berisi tentang teori dasar yang berhubungan dengan penulisan laporan yang menunjang penelitian tentang akses pintu yang memanfaatkan teknologi biometrik, *smartcard*, dan *web browser*.

BAB III METODOLOGI PENELITIAN

Bab ini berisi proses perancangan sistem baik *hardware* maupun *software*, serta menjelaskan langkah-langkah pengujian sistem alat yang dirancang menggunakan teknologi biometrik, *smartcard*, dan *web browser* untuk mengakses pintu ruangan.

BAB IV HASIL DAN PEMBAHASAN

Bab ini berisi pengujian menyeluruh dari alat *Smart Key Reader* yang dirancang dan menganalisis hasil pengujian setiap faktor autentikasi untuk mencapai kesimpulan.

BAB V PENUTUP

Bab ini merupakan penutup yang berisi kesimpulan tentang hasil pemecahan masalah yang diperoleh selama penyusunan skripsi, serta tambahan beberapa saran untuk pengembangan penelitian lebih lanjut terkhusus pada bidang *Smart Reader*.

BAB II

TINJAUAN PUSTAKA

II.1. *Internet Of Things* (IoT)

Internet of Things (IoT), juga disebut Internet Segalanya atau komputasi *Cloud* atau *Industrial Internet*, merupakan paradigma teknologi baru yang dibayangkan sebagai jaringan global IoT, mesin dan perangkat yang mampu berinteraksi satu sama lain. IoT diakui sebagai frekuensi Radio dan juga salah satu bidang terpenting dari teknologi masa depan dan mendapatkan perhatian luas dari berbagai industri. Nilai sebenarnya dari IoT untuk perusahaan dapat sepenuhnya terwujud ketika perangkat yang terhubung dapat saling berkomunikasi dan berintegrasi dengan sistem inventaris yang dikelola vendor, sistem dukungan pelanggan, aplikasi intelijen bisnis, dan analitik bisnis. Memperkirakan bahwa IoT akan mencapai 26 miliar unit pada tahun 2020, naik dari 0,9 miliar pada tahun 2009, dan akan berdampak pada informasi yang tersedia untuk mitra rantai pasokan dan bagaimana rantai pasokan beroperasi. Dari lini produksi dan pergudangan hingga pengiriman ritel dan rak toko, IoT mengubah proses bisnis dengan memberikan visibilitas yang lebih akurat dan *real-time* ke dalam aliran bahan dan produk. Perusahaan akan berinvestasi dalam IoT untuk mendesain ulang alur kerja pabrik, meningkatkan pelacakan bahan, dan mengoptimalkan biaya distribusi [7].

IoT merupakan segala aktifitas yang pelakunya saling berinteraksi dan dilakukan dengan memanfaatkan internet. Dalam penggunaan nya IoT banyak ditemui dalam berbagai aktifitas, contohnya : banyaknya transportasi *online*, *e-commerce*, pemesanan tiket secara *online*, *live streaming*, *e-learning* dan lain-lain

bahkan sampai alat-alat untuk membantu dibidang tertentu seperti *remote temperature sensor*, GPS *tracking*, and sebagainya yang menggunakan internet atau jaringan sebagai media untuk melakukannya. Dengan banyaknya manfaat dari IoT maka membuat segala sesuatu nya lebih mudah, dalam bidang pendidikan IoT sangat diperlukan untuk melakukan segala aktifitas dengan menggunakan sistem dan tertata serta sistem pengarsipan yang tepat [8].

IoT dapat digambarkan sebagai penghubung benda-benda seperti telepon pintar, televisi

Internet, sensor dan aktuator ke Internet dimana perangkat tersebut digabungkan menjadi bentuk baru yang memungkinkan adanya komunikasi antara seseorang dan benda tersebut. IoT memiliki konsep yang bertujuan untuk memperluas manfaat yang tersambung dalam koneksi Internet secara terus menerus [9].

II.2. *Multi-Factor Authentication* (MFA)

Autentikasi adalah proses dimana "pengguna mengidentifikasi dirinya dengan mengirimkan x ke sistem; sistem mengautentikasi identitasnya dengan menghitung F(x) dan memeriksa bahwa itu sama dengan nilai yang tersimpan y". Definisi ini tidak berubah secara signifikan dari waktu ke waktu meskipun fakta bahwa *password* sederhana tidak lagi menjadi satu-satunya faktor untuk memvalidasi pengguna dari perspektif teknologi informasi.

Berikut adalah tiga tipe umum informasi yang tersedia untuk dijadikan faktor-faktor pada metode autentikasi:

1. *Knowledge factor* adalah sesuatu yang diketahui oleh *user*, seperti *password* atau “rahasia”;
2. *Ownership factor* adalah sesuatu yang dimiliki oleh *user*, seperti kartu atau *smartphone*;
3. *Biometric factor* adalah *user* itu sendiri, seperti data biometrik atau pola kebiasaan.

Multi-Factor Authentication (MFA) adalah metode autentikasi yang memerlukan *user* untuk menyediakan dua atau lebih faktor verifikasi untuk mendapatkan akses ke sebuah perangkat tertentu. Sebagian besar MFA berbasis pada biometrik, yang secara automatis mengenali individu berdasarkan perilakunya dan karakteristik biologi. Metode ini menawarkan peningkatan level keamanan karena bergantung pada dua atau lebih faktor yang berbeda.

II.3. Penguncian Cerdas

Penguncian cerdas merupakan tambahan atau bahkan pengganti penguncian konvensional dikarenakan sistem ini dapat mengatasi kelemahan bahkan memberikan kemudahan yang lebih baik dibandingkan penguncian konvesional. Fitur yang diberikan penguncian cerdas seperti memberikan informasi berupa alarm ketika penguncian berusaha dibobol, dapat melacak siapa yang masuk dan keluar dari suatu wilayah atau area serta peningkatan pengontrolan sehingga pengguna dapat menutup dan membuka akses dari jarak jauh. Hal ini membuat sistem penguncian cerdas merupakan hal yang diminati untuk diteliti lebih lanjut demi perkembangan sistem penguncian cerdas. Penguncian cerdas merupakan salah satu produk dari rumah cerdas yang masih banyak perlu dikembangkan untuk mengatasi

kelemahan yang ada seperti kurang responsif dalam mengontrol, permasalahan daya yang digunakan dan sebagainya [6].

II.4. ESP32

ESP32 dikenalkan oleh Espressif System yang merupakan penerus dari mikrokontroler ESP8266. Mikrokontroler ESP32 memiliki keunggulan yaitu sistem berbiaya rendah, dan juga berdaya rendah dengan modul WiFi yang terintegrasi dengan chip mikrokontroler serta memiliki bluetooth dengan mode ganda dan fitur hemat daya menjadikannya lebih fleksibel. ESP32 kompatibel dengan perangkat seluler dan aplikasi IoT (*Internet of Things*). Mikrokontroler ini dapat digunakan sebagai sistem mandiri yang lengkap atau dapat dioperasikan sebagai perangkat pendukung mikrokontroler *host*.

Gambar II. 1 ESP32

ESP32 yang diperlihatkan pada gambar II.1 adalah chip combo 2,4 GHz WiFi dan Bluetooth tunggal yang dirancang dengan daya ultra rendah TSMC 40 nm. Perangkat ini dirancang untuk mencapai daya dan kinerja RF yang terbaik,

menunjukkan ketahanan, keserbagunaan dan keandalan dalam berbagai macam aplikasi dan skenario daya [6].

II.5. ESP32-CAM

ESP32-CAM adalah papan pengembangan mode ganda WIFI + bluetooth yang menggunakan antena dan inti papan PCB berbasis chip ESP32. Modul ini dapat bekerja secara independen sebagai sistem minimum. Modul ini merupakan sebuah modul WiFi yang sudah dilengkapi dengan kamera ov2640. Dari modul ini bisa digunakan untuk berbagai keperluan, contoh untuk CCTV, mengambil gambar dan sebagainya. Fitur lain yaitu kita bisa mendeteksi wajah (*face detection*) dan pengenal wajah (*face recognition*). Maka demikian, modul ESP32-CAM ini dapat digunakan untuk mengambil gambar, dan juga dapat digunakan sebagai modul WiFi untuk mengirim data [10].

Gambar II. 2 ESP32-CAM

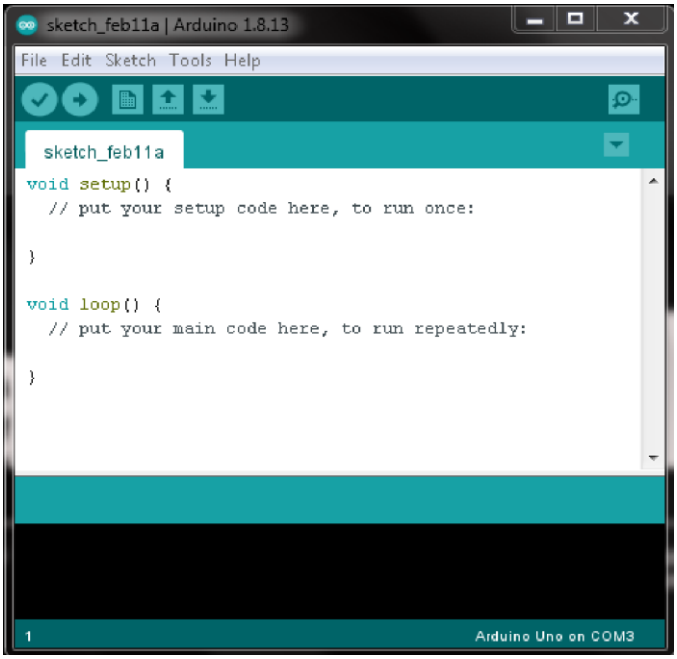
Fitur dari modul ESP32-CAM adalah [10]:

a. Modul Ultra-small 802.11b / g / n WiFi + BT / BLE SoC
b. Daya rendah dual-core 32-bit CPU untuk prosesor aplikasi

c. Hingga 240MHz, hingga 600 DMIPS
d. Built-in 520 KB SRAM, eksternal 4M PSRAM
e. Mendukung antarmuka seperti UART / SPI / I2C / PWM / ADC / DAC
f. Mendukung kamera OV2640 dan OV 7670 dengan flash built-in
g. Dukungan untuk upload gambar WiFi
h. Dukungan kartu TF
i. Mendukung beberapa mode tidur
j. Tertanam Lwip dan FreeRTOS
k. Mendukung mode kerja STA / AP / STA + AP
l. Dukungan smart config / AirKiss jaringan distribusi sekali klik
m. Dukungan untuk peningkatan lokal serial dan peningkatan firmware jarak jauh (FOTA)
n. Mendukung pengembangan sekunder

II.6. *Software* Arduino IDE (*Integrated Development Environment*)

Arduino memakai *Software processing* untuk diaplikasikan dalam menulis program kedalam *Arduino processing* ini sendiri merupakan penggabungan antara bahasa C++ dan bahasa Java. *Software Arduino* dapat di install di berbagai operating sistem (OS) 9 Linux, Mac OS dan Windows. *Software arduino* yang biasa digunakan adalah software IDE yang tampilannya dapat dilihat pada gambar II.3.



Gambar II. 3 *Software* Arduino IDE (*Integrated Development Environment*)

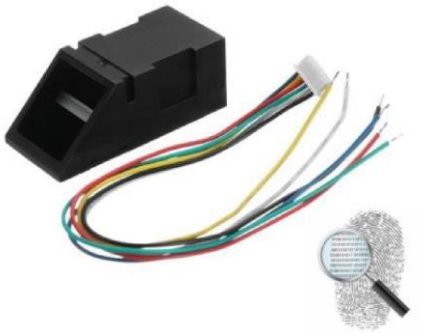
IDE Arduino adalah software yang sangat canggih dan dapat di program menggunakan Java. IDE Arduino terdiri dari [11]:

1. Editor program adalah jendela yang memungkinkan pengguna untuk menulis dan mengedit program dalam Bahasa *Processing*.
2. *Compiler* adalah fitur untuk mengubah kode program menjadi kode biner. Compiler perlu dilakukan dalam hal ini. Karena sebuah mikrokonroler tidak bisa memahami Bahasa *Processing.*
3. *Uploader* adalah fitur untuk memuat kode biner dari komputer yang diteruskan ke memori pada *board Arduino.*

II.7. Sensor *Fingerprint* AS608

Fingerprint adalah sebuah alat elektronik yang menerapkan sensor *scanning* untuk mengetahui sidik jari seseorang guna keperluan verifikasi identitas. Sensor

fingerprint seperti ini digunakan pada beberapa peralatan elektronik seperti smartphone, pintu masuk, alat absensi karyawan dan berbagai macam peralatan elektronik yang membutuhkan tingkat keamanan yang tinggi, dan hanya bisa di akses oleh orang - orang tertentu saja [12].



Gambar II. 4 Sensor Sidik Jari AS608

Sensor *fingerprint* merupakan sensor yang digunakan untuk mendeteksi sidik jari dengan menggunakan sistem optik. Deteksi dilakukan dengan membaca kontur (tinggi dan rendah permukaan) sidik jari dan listrik statis tubuh. Hal ini menghasilkan tingkat keamanan yang tinggi karena tidak bisa dipalsukan dengan fotokopi sidik jari ataupun sidik jari palsu. Sensor ini memiliki lapisan kaca yang tahan lama dan juga memiliki sensor gerak, yaitu jika jari ditempelkan pada sensor maka sensor akan langsung menyala untuk mengambil sidik jari [13].

Spesifikasi dari modul AS608:

- *Supply Voltage*: 3.6-6.0 VDC
- *Operating Current*: 120mA max
- *Peak Current*: 150mA max
- *Fingeprint Imaging Time*: <1.0 Seconds

- Luas Permukaan Sensor: 14mm x 18mm
- Kapasitas Penyimpanan: 162 Sidik Jari

II.8. Smartcard

Smartcard adalah anggota termuda dan terpandai dari keluarga kartu identitas dalam format ID-1. Fitur karakteristiknya adalah sirkuit terintegrasi yang tertanam dalam kartu, yang memiliki komponen untuk mentransmisikan, menyimpan, dan memproses data. Data dapat ditransmisikan menggunakan kontak pada permukaan kartu atau medan elektromagnetik, tanpa kontak apa pun. *Smartcard* menawarkan beberapa keunggulan dibandingkan dengan kartu magnetic-stripe. Misalnya, kapasitas penyimpanan maksimum smartcard berkali-kali lebih besar daripada kartu strip magnetik. Chip dengan lebih dari 256 kB memori saat ini tersedia, dan angka ini akan berlipat ganda dengan setiap generasi chip baru [14].

Adapun *smartcard* yang dikembangkan oleh NXP Semiconductors, Mifare Classic EV1 yang ditunjukkan pada gambar II.5 merupakan *contactless smart card* berbasis ISO/IEC 14443 Type A. Mifare Classic EV1 yang digunakan adalah dengan memori 1K [15].

Gambar II. 5 Tag Mifare Classic 1K.

Fitur dan keunggulan:

a. Transmisi data dan suplay energi *contactless*.
b. Jarak pengoperasian mencapai 100 mm, tergantung dari geometri antena dan konfigurasi *reader*.
c. Bekerja pada frekuensi 13.56 MHz.
d. Transfer data 106 kbit/s.
e. *Data integrity of 16-bit CRC, parity, bit coding, bit counting*.
f. Anticollision.
g. Tipikal waktu transaksi tiket < 100 ms (termasuk *backup management*).
h. 7 Byte UID atau 4 Byte NUID.
i. Mendukung *Random ID* (7 Byte UID *version*).

II.9. Radio Frequency Identification (RFID)

Radio Frequency Identification (RFID) adalah sebuah teknologi yang menggunakan frekuensi radio untuk mengindentifikasi suatu barang atau manusia. Sejarah perkembangan *radio frequency identification* dimulai sejak tahun 1920,

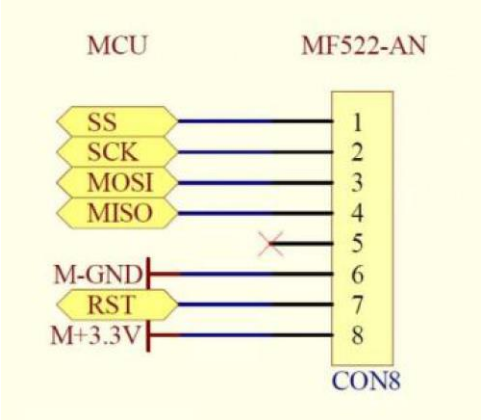
tetapi berkembang menjadi IFF transponder pada tahun 1939, dapat dilihat pada gambar II.6.

Gambar II. 6 *Radio Frequency Identification* (RFID)

Yang waktu itu berfungsi sebagai alat identifikasi pesawat musuh, dipakai oleh militer Inggris pada perang dunia II. Sejak tahun 1945 beberapa orang berfikir bahwa perangkat pertama RFID ditemukan oleh Leon Theremin sebagai suatu tool untuk pemerintahan Rusia. Sistem RFID terbagi menjadi 3 komponen, yaitu: RFID Tag, RFID Terminal Reader, dan Middleware. Sedangkan untuk jenisnya RFID terbagi, berdasarkan frekuensi, berdasarkan sumber energi, dan berdasarkan bentuk [16].

Identifikasi data pada RFID tag dilakukan melalui frekuensi radio yang merambat melalui media udara pada jangkauan tertentu sesuai dengan fitur yang dimiliki oleh setiap modul RFID (terdiri dari RFID *Reader* dan RFID tag) yang digunakan. Pada umumnya, data RFID tag yang bersifat unik tersimpan atau tertanam dalam sebuah kartu chip sehingga pengaruh kondisi alam seperti debu, kotoran ataupun temperature udara tidak akan mengurangi kualitas komunikasi data yang terjadi. Fitur-fitur yang dimiliki oleh teknologi RFID ini menjadi keunggulan

dari teknologi RFID. Namun keunggulan ini akan bersifat relatif karena akan tergantung dari pemanfaatan suatu teknologi identifikasi pada suatu aplikasi yang akan diimplementasikan. Teknologi ini telah dimanfaatkan pada berbagai aplikasi yang berhubungan dengan sistem identifikasi objek pada beberapa penelitian sebelumnya, seperti membuka pintu, mengakses *computer*, menyalakan sepeda motor, serta mengontrol peralatan di ruangan kantor seperti lampu, *computer* dan lampu penerangan [17].



Gambar II. 7 Konfigurasi pin modul MRFC522 RFID

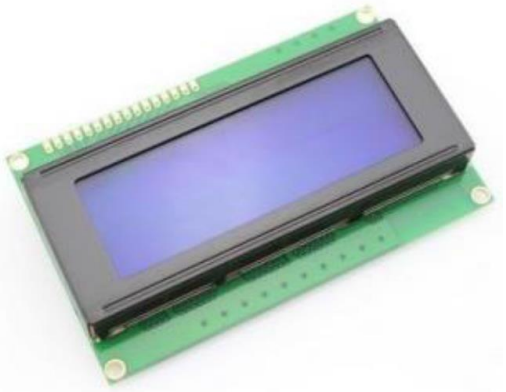
Spesifikasi dari modul RFID MIFARE RC522 [18]:

- *Chipset*: MFRC522 *Contactless Reader/Writer IC*
- Frekuensi: 13,56 MHz
- Jarak pembacaan kartu: < 50mm
- Protokol akses: SPI (*Serial Peripheral Interface*) @ 10 Mbps
- Kecepatan transmisi RF: 424 kbps (dua arah / *bi-directional*) / 848 kbps (*unidirectional*)

- Mendukung kartu MIFARE jenis Classic S50 / S70, UltraLight, dan DESFire
- *Framing & Error Detection* (parity+CRC) dengan 64 byte internal I/O *buffer*
- Catu Daya: 3,3 Volt
- Konsumsi Arus: 13-26 mA pada saat operasi baca/tulis, < 80µA saat modus siaga.
- Suhu operasional: -20°C s.d. +80°C
- Dimensi: 40 x 50 mm

II.10. LCD

LCD (*Liquid Crystal Display*) adalah suatu jenis media tampilan yang menggunakan kristal cair sebagai penampil utama, LCD *dot matrix* berfungsikan untuk menampilkan tulisan berupa angka, huruf, dan grafik sesuai dengan yang diinginkan (sesuai dengan program yang digunakan untuk mengontrolnya). Pada penelitian ini menggunakan LCD *dot matrix* dengan karakter 16x2, yang kaki-kakinya berjumlah 20 pin, seperti yang terlihat pada gambar II.8.



Gambar II. 8 LCD (*Liquid Crystal Display*)

LCD adalah suatu jenis media tampilan yang menggunakan kristal cair sebagai penampil utama. LCD terdiri dari lapisan-lapisan cairan kristal diantara dua pelat kaca. Film transparan yang dapat menghantarkan listrik atau *back plane*, diletakkan pada lembaran belakang kaca kemudian bagian trasparan dari film yang dapat menghantarkan arus listrik pada bagian luar dari karakter yang diinginkan dilapiskan pada pelat bagian depan. Pada saat terdapat tegangan antara segmen dan *back plane*, bagian yang berarus listrik ini mengubah transmisi cahaya melalui daerah di bawah segmen film. Berdasarkan jenis tampilan, LCD dapat dikelompokkan menjadi beberapa jenis, yaitu:

1. *Segment* LCD jenis ini terbentuk dari beberapa *seven-segment display* atau *sixteen segment display*, tetapi ada juga yang menggunakan gabungan dari keduanya. LCD jenis ini sering dipakai pada jam digital dan alat ukur digital.
2. *Dot Matrix Character* LCD jenis ini terbentuk dari beberapa *dot matrix* display berukuran 5x7 atau 5x9, yang membentuk sebuah matriks yang lebih besar dengan berbagai kombinasi jumlah kolom dan baris. Kombinasi ini menentukan jumlah karakter yang dapat ditampilkan oleh LCD tersebut, seperti 2 baris x 20 karakter atau 4 baris x 20 karakter.
3. Graphic LCD jenis ini masih terus berkembang sampai saat ini. Resolusi LCD jenis ini bervariasi, diantaranya 128x64, 128x128, 240x64, 240x128. Sekarang ini, graphic LCD banyak dipakai pada handycam, laptop, telepon selular (*cellphone*), monitor komputer, dan lain-lain.

Adapun register-register yang terdapat dalam LCD adalah sebagai berikut [6]:

1. IR (*Instruction Register*), digunakan untuk menentukan fungsi yang harus dikerjakan oleh LCD serta pengalamatan DDRAM atau CGRAM.
2. DR (data register), digunakan sebagai tempat data DDRAM atau CGRAM yang akan dituliskan ke atau dibaca oleh komputer atau sistem minimum.
3. BF (*Busy Flag*), digunakan untuk memberi tanda bahwa LCD dalam keadaan siap atau sibuk. Apabila LCD sedang melakukan operasi internal, BF di-set menjadi 1, sehingga tidak akan menerima perintah dari luar. Jadi, BF harus dicek apakah telah direset menjadi 0 ketika akan menulis LCD (memberi data pada LCD). Cara untuk menulis LCD adalah dengan mengeset RS menjadi 0 dan mengeset R/W menjadi 1.
4. AC (*address counter*), digunakan untuk menunjuk alamat pada DDRAM atau CGRAM dibaca atau ditulis, maka AC secara otomatis menunjukkan alamat berikutnya. Alamat yang disimpan AC dapat dibaca bersamaan dengan BF.
5. DDRAM (*Display Data Random Access Memory*), digunakan sebagai tempat penyimpanan data sebesar 80 byte. AC menunjukan alamat karakter yang sedang ditampilkan.
6. CGROM (*Character Generator Read Only Memory*), pada LCD telah terdapat ROM untuk menyimpan karakter-karakter ASCII (*American Standard Code for Interchange Information*), sehingga cukup memasukan kode ASCII untuk menampilkannya.
7. CGRAM (*Character Generator Random Access Memory*), sebagai data storage untuk merancang karakter yang dikehendaki. Untuk CGRAM

terletak pada kode ASCII dari 00h sampai 0Fh, tetapi hanya delapan karakter yang disediakan. Alamat CGRAM hanya 6 bit, 3 bit untuk mengatur tinggi karakter dan 3 bit tinggi menjadi 3 bit rendah DDRAM yang menunjukan karakter, sedangkan 3 bit rendah sebagai posisi data CGRAM untuk membuat tampilan satu baris dalam dot matrix 5x7 karakter tersebut dimulai dari atas, sehingga 25 karakter untuk kode ASCII 00h sama dengan 09h sampai 07h dengan 0Fh. Oleh karena itu untuk perancangan satu karakter memerlukan penulisan data ke CGRAM sampai delapan kali.

II.11. *Web Browser*

Web Browser merupakan nama penelusuran yaitu dengan perangkat lunak yang mempunyai fungsi untuk melakukan dan berhubungan dengan dokumen yang berada di *web server* atau secara sederhana. *Browser* adalah suatu program yang digunakan untuk menjelajahi dunia Internet atau sebagai alat untuk mencari informasi tentang suatu halaman web yang tersimpan di komputer [19]. Jenis-jenis *web browser* ditunjukkan pada Gambar 2.9.

Gambar II. 9 Jenis-jenis *Web Browser*

II.12. Firebase

Firebase adalah *Backend as a Service* (BaaS) yang saat ini dimiliki oleh Google ditunjukkan pada gambar II.9 Firebase merupakaan solusi yang ditawarkan oleh Google untuk mempermudah pengembangan aplikasi mobile. Dua fitur menarik dari Firebase adalah *Firebase Remote Config* dan *Firebase Real Time Database*. Selain itu juga terdapat fitur pendukung untuk aplikasi yang memerlukan *push notification* yaitu *Firebase Notification Console*. *Firebase Database* merupakan penyimpanan basis data noSQL yang memungkinan untuk menyimpan beberapa tipe data. Tipe data itu antara lain String, Long, dan Boolean. Data pada *Firebase Database* disimpan sebagai objek JSON tree. Tidak seperti basis data SQL, tidak ada tabel dan baris pada basis data noSQL. Ketika ada penambahan data, data tersebut akan menjadi node pada struktur JSON. Node merupakaan simpul yang berisi data dan bisa memiliki cabang-cabang berupa node lainnya yang berisi data pula. Proses pengisian suatu data ke Firebase Database dikenal dengan istilah push. Selain Firebase Database, Firebase menyediakan beberapa layanan lainnya yang juga dimanfaatkan dalam pengembangan aplikasi ini. Layanan tersebut antara lain *Firebase Authentication*, *Storage,* dan *Cloud Messaging*. Pada pengembangan aplikasi, layanan lainnya yang digunakan pada pengembangan aplikasi adalah *Firebase Storage*. Layaknya sebuah penyimpanan awan, *Firebase Storage* memungkin pengembang untuk mengunggah atau mengunduh sebuah berkas pada pengembangan aplikasi [6].

II.13. Heroku

Heroku adalah sebuah *cloud platform* yang menjalankan bahasa pemrograman tertentu, Heroku mendukung bahasa pemrograman seperti Ruby, Node.js, Python, Java, PHP, dan lain-lain.

Heroku termasuk ke dalam kriteria *Platform As A Service* (PaaS), sehingga bagi anda yang ingin melakukan *deploy* aplikasi ke heroku cukup hanya dengan melakukan konfigurasi aplikasi yang ingin di *deploy* dan menyediakan platform yang memungkinkan pelanggan untuk mengembangkan, menjalankan, dan mengelola aplikasi tanpa kompleksitas membangun dan memelihara infrastruktur yang biasanya terkait dengan pengembangan dan peluncuran aplikasi.

Manfaat menggunakan Heroku adalah layanannya yaitu menjalankan script app langsung tanpa memerlukan setting yang sangat rumit, memungkinkan pengembang aplikasi lebih fokus pada kode aplikasi mereka, tanpa terlalu dipusingkan dengan arsitektur dan server [20].

II.14. Node.js

Node.js adalah sistem perangkat lunak yang didesain untuk pengembangan aplikasi web. Aplikasi ini ditulis dalam bahasa JavaScript, menggunakan basis *event* dan *asynchrounous* I/O. Tidak seperti kebanyakan bahasa JavaScript yang dijalankan pada web browser, Node.js dieksekusi sebagai aplikasi server. Aplikasi ini terdiri dari V8 JavaScript Engine buatan Google dan beberapa modul bawaan yang terintegrasi [21].

Modul-modul yang digunakan dalam pembuatan web server untuk smart reader ini antara lain *Face-api.js* sebagai implementasi tensorflow untuk mendeteksi wajah, *Multer* untuk menangani multipart/form data, dan *Express* yang merupakan kerangka kerja HTTP pada Node.js

II.15. Tensorflow.js

Tensorflow merupakan *open source framework* yang dapat digunakan untuk mengembangkan, melatih, dan menggunakan model deteksi objek. Sistem ini sudah banyak diterapkan pada berbagai produk google anatar lain pencarian image, deteksi wajah, dan plat nomor kendaraan pada *google street view, Google assistant, way mo* atau *self driving car,* dan lain-lain [22].

TensorFlow.js adalah *library* untuk membuat dan menjalankan algoritme pembelajaran mesin dalam JavaScript. Model TensorFlow.js berjalan di browser web dan di lingkungan Node.js. TensorFlow.js dapat berjalan di sisi klien dan server untuk menjalankan *Machice Learning* menggunakan JavaScript dengan performa tinggi. *Library* adalah bagian dari ekosistem TensorFlow, yang menyediakan serangkaian API yang kompatibel dengan yang ada di Python, memungkinkan model untuk di-porting antara ekosistem Python dan JavaScript [23].

TensorFlow.js bukan satu-satunya *library* Javascript untuk *deep learning*, dan juga bukan yang pertama kali muncul sebelumnya sudah ada brain.js dan ConvNetJS. Alasan mengapa TensorFlow.js sering digunakan dibandingkan dengan *library* yang lain adalah kelengkapannya. TensorFlow.js adalah satu-

satunya perpustakaan yang tersedia saat ini yang mendukung semua bagian penting dari alur kerja dalam *deep learning* [22].

II.15.1. Face-api.js

Face-api.js adalah modul JavaScript, dibangun di atas inti tensorflow.js, dan mengimplementasikan beberapa *Convolutional Neural Networks* (CNNs) untuk melakukan deteksi wajah dan pengenalan wajah, dan telah dioptimalkan untuk bekerja di web dan perangkat seluler [24].

BAB III

METODE PENELITIAN

III.1. Jenis Penelitian

Jenis penelitian ini merupakan penelitian berbentuk eksperimen, dimana penelitian ini dilakukan untuk merancang dan menguji sistem keamanan berbasis MFA untuk ruangan yang menggabungkan teknologi biometrik, *smartcard*, dan *web browser*.

III.2. Tempat dan Waktu Penelitian

Penelitian ini dilakukan di Laboratorium Telekomunikasi Radio dan Gelombang Pendek Departemen Teknik Elektro Fakultas Teknik Universitas Hasanuddin. Waktu penelitian dijadwalkan pada periode November 2021 hingga Juli 2022.

III.3. Spesifikasi Rancangan

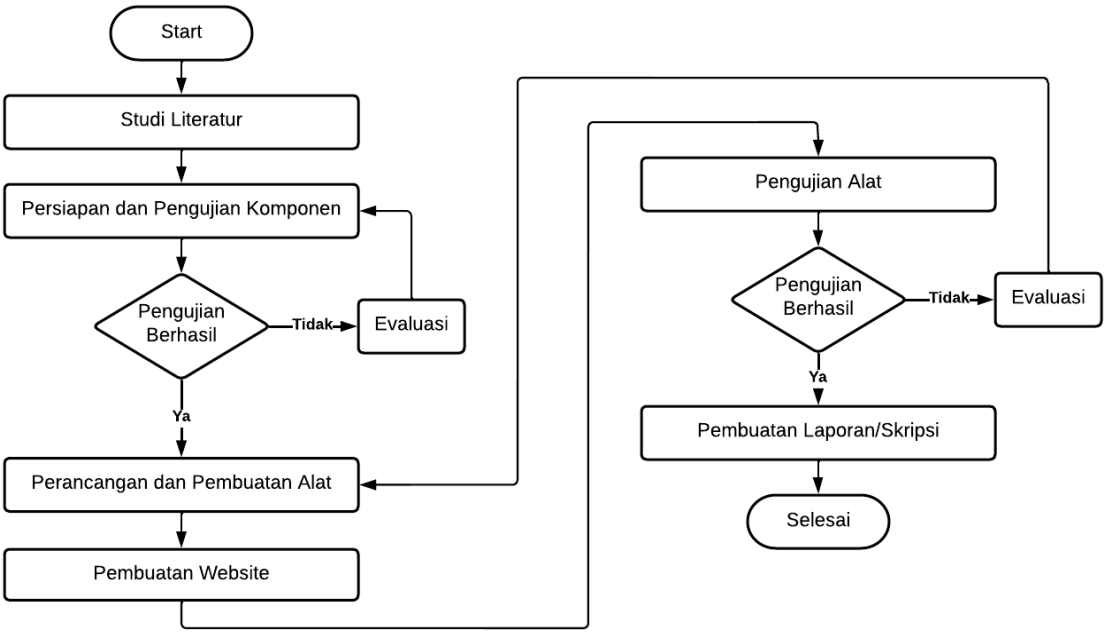
Untuk merancang suatu alat dan sistem keamanan reader berbasis MFA, maka diperlukan beberapa komponen dengan spesifikasi khusus yang memenuhi kriteria perancangan prototipe. Tabel 3.2 merupakan komponen utama yang digunakan dalam penelitian ini :

Tabel III. 1 Daftar Komponen

NO	JENIS	JUMLAH	KETERANGAN
1.	ESP32	1	Sebagai pembaca dan pengontrol *Smartcard*, *Fingerprint* dan modul WiFi
2.	ESP32-CAM	1	Sebagai mikrokontroler dan modul WiFi
3.	AS608	1	Sebagai pembaca sidik jari
4.	RFID RC522	1	Sebagai pembaca *Smartcard*
5.	*Smartcard*	1	Sebagai alat yang dibaca oleh RFID RC522
6.	USB TTL	1	Sebagai alat penghubung komputer ke ESP32-CAM
7.	Adaptor 12v	1	Sebagai sumber listrik
8.	LM2596	1	Sebagai pengatur tegangan yang masuk dari adaptor
9.	Kabel Jumper	1	Sebagai penghubung komponen elektronika
10.	LCD 2x16 + i2c	1	Sebagai media penampil yang dihasilkan oleh alat
11.	Elektromagnetik Door Lock 12v	1	Sebagai pengunci pintu
12.	Relay 5v	1	Sebagai pengontrol EM Door Lock

III.4. Diagram Alir Perencanaan

Dalam perancangan *Smart Key Reader* dengan MFA, terdapat beberapa tahapan yang harus dilakukan mulai dari studi literatur sampai pembuatan laporan. Gambar 3.1 memperlihatkan diagram alir perencanaan *Smart Key Reader* dengan MFA.



Gambar III. 1 Diagram Alir Penelitian

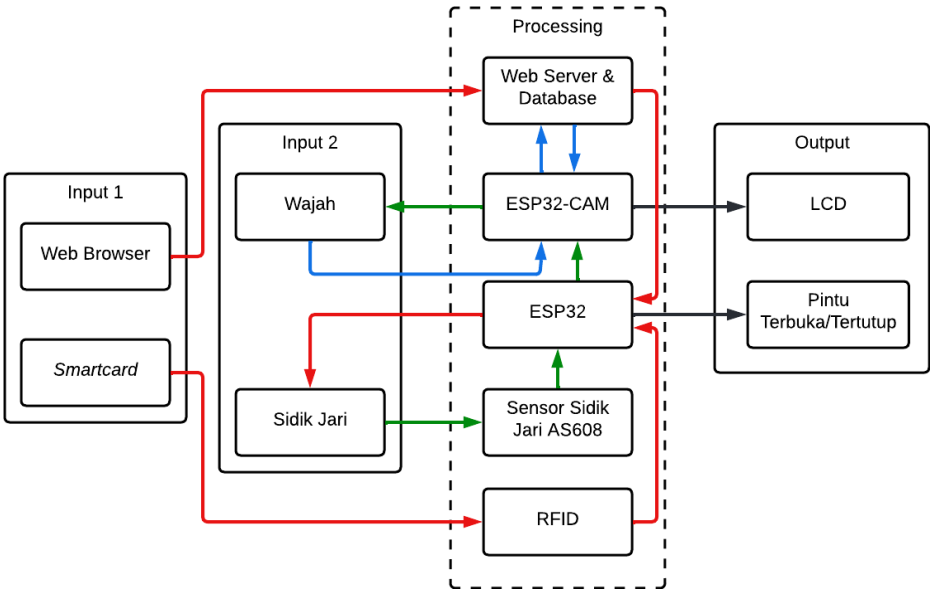
Penjelasan mengenai setiap langkah kegiatan penelitian diuraikan pada item-item berikut ini :

1. Studi Literatur: Studi literatur merupakan kajian penulis atas referensi-referensi yang ada baik berupa buku, jurnal, internet maupun melalui media massa yang berhubungan dengan penulisan laporan penelitian ini.
2. Persiapan dan Pengujian Komponen: Dilakukan dua tahap, tahap pertama adalah persiapan dilakukan dengan membeli komponen yang akan digunakan, tahap kedua adalah pengujian komponen yang akan digunakan untuk mengetahui data dan kelayakan komponen.
3. Perancangan dan Pembuatan Alat: Pada tahap perancangan, dilakukan desain dan perancangan alat yang akan dibuat. Pada tahap pembuatan alat, dilakukan proses pembuatan alat dengan merangkai komponen yang sudah dipersiapkan sebelumnya sesuai rancangan alat.

4. Pembuatan Website: Pada tahap ini, dilakukan pembuatan aplikasi berbasis web untuk sistem *face recognition*.
5. Pengujian Alat: Pengujian dilakukan dalam kondisi ideal dengan menggunakan parameter-parameter pengujian yang telah ditetapkan. Sistem diuji coba untuk mendapatkan hasil yang sesuai dengan parameter yang diujikan.
6. Pengambilan Data: Pada tahap ini, seluruh hasil dari evaluasi kinerja alat dicatat dan didokumentasikan sebagai data hasil penelitian.
7. Pembuatan Laporan: Pada tahap ini, diperoleh hasil yang dapat menyelesaikan masalah yang diteliti, sehingga hasil ini ditetapkan sebagai simpulan dari penelitian ini. Pada simpulan ini tujuannya untuk membuat laporan hasil akhir yang telah didapatkan.

III.5. Diagram Blok Sistem Kerja Smart Key Reader

Sistem kerja *Smart Key Reader* dengan MFA ditunjukkan pada gambar 3.2 yang bekerja dengan dua kondisi yaitu menggunakan *smartcard* atau web browser untuk mengakses *reader* yang kemudian dilanjutkan dengan verifikasi sidik jari dan wajah.



Gambar III. 2 Blok Diagram Sistem Kerja *Smart Key Reader*

1. Blok *Input*

Blok *input* terdiri dari *web browser* atau *smartcard* yang merupakan blok input pertama untuk mengakses wajah dan sidik jari yang berfungsi sebagai parameter blok input kedua yang akan digunakan untuk mengontrol *reader* agar dapat mengakses ruangan. Jika blok *input* pertama berhasil diakses dan selama 10 detik blok *input* kedua tidak terbaca atau gagal terbaca maka sistem akan kembali untuk meminta blok *input* yang pertama.

2. Blok *Processing*

Blok *processing* merupakan suatu bagian utama pada sistem *smart key reader*. Blok *processing* juga bias disebut otak dari rangkaian *smart key reader* ini, untuk komponen intinya terdiri dari RFID, *fingerprint sensor* AS608, ESP32, ESP32-CAM, dan *web server*. Pada RFID terjadi proses pembacaan ID pada *smartcard*, kemudian data ID akan dikirim ke ESP32. Pada ESP32 terjadi pengolahan data. Data yang terdaftar akan mengaktifkan *fingerprint sensor* AS608

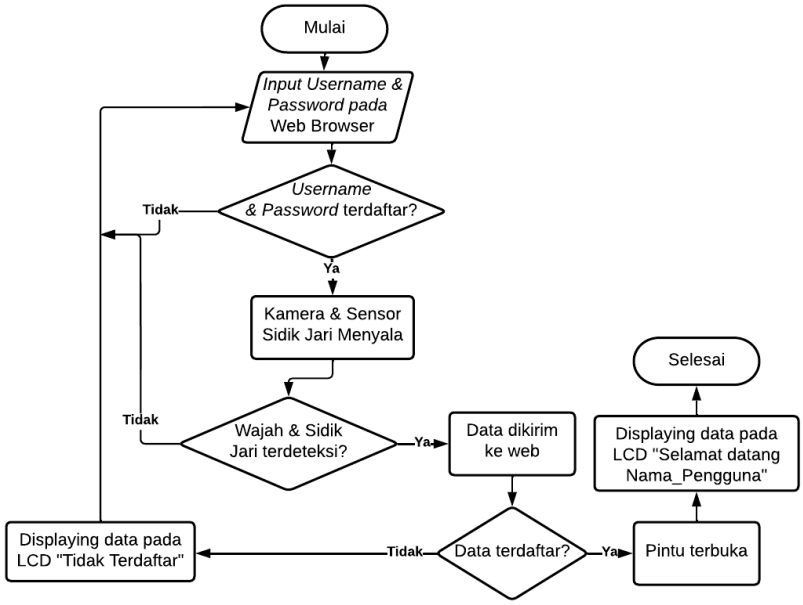
untuk membaca sidik jari dan mengirim data sidik jari ke ESP32-CAM yang akan mengaktifkan kameranya untuk mengambil gambar wajah. ESP32-CAM kemudian mengirim seluruh data yang diperoleh ke *web server*. *Web server* bertugas untuk mengolah data wajah, sidik jari, dan data akses *reader* dari *web browser*.

3. Blok *Output*

Blok *output* merupakan bagian akhir dari sistem *smart key reader* yang berfungsi menampilkan hasil pengolahan data dari *web server* ke LCD sebagai perintah terbuka atau tertutupnya pintu.

Sistem keamanan berbasis MFA ini berkerja dengan dua mode, adapun rinciannya dapat dilihat dalam masing-masing *flowchart* dibawah ini.

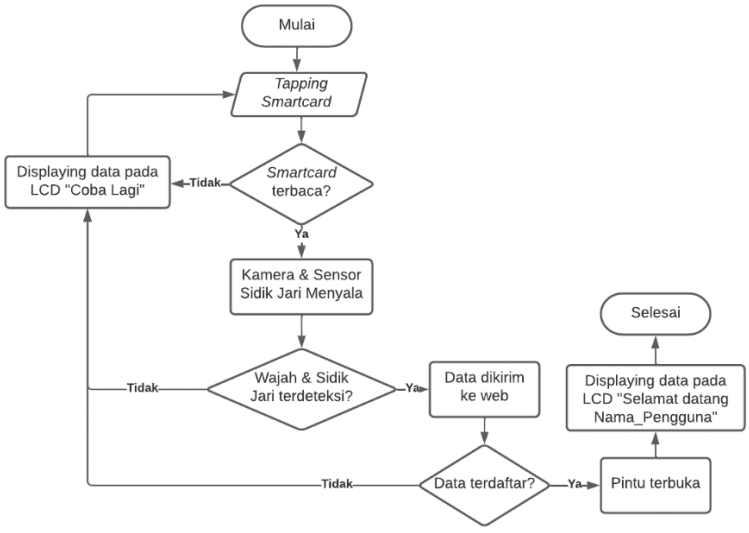
a. *Flowchart* Proses Akses Reader MFA Menggunakan *Web Browser*, Sidik Jari dan *Face Recognition*



Gambar III. 3 *Flowchart* Proses Akses Reader MFA Menggunakan *Web Browser*, Sidik Jari, dan *Face Recognition*

Dari *flowchart* pada gambar 3.3 dapat dilihat cara mengakses reader dengan menggunakan *web browser* untuk mengakses halaman *login*. Halaman *login website* dapat diakses melalui *smart device* apapun yang mempunyai *web browser*. *Web server* akan menerima data *username* dan *password* untuk divalidasi dengan *database*. Sehingga dapat mengaktifkan sensor sidik jari AS608 pada ESP32. ID jari yang terdeteksi akan dikirim ke ESP32-CAM sebagai isyarat agar kamera mengambil gambar. Selanjutnya ID jari dan gambar wajah yang ditangkap kamera secara bersamaan dikirim ke server melalui modul *WiFi* untuk dilakukan validasi dan verifikasi. Jika ID jari dan wajah cocok maka server akan merespon agar ESP32 dapat mengontrol *Electromagnetic Door Lock* agar pintu dapat terbuka selama 5 detik. sebaliknya pintu akan tetap terkunci jika ID jari dan wajah tidak sesuai.

b. *Flowchart* Proses Akses Reader MFA Menggunakan *Smartcard*, Sidik Jari, dan *Face Recognition*



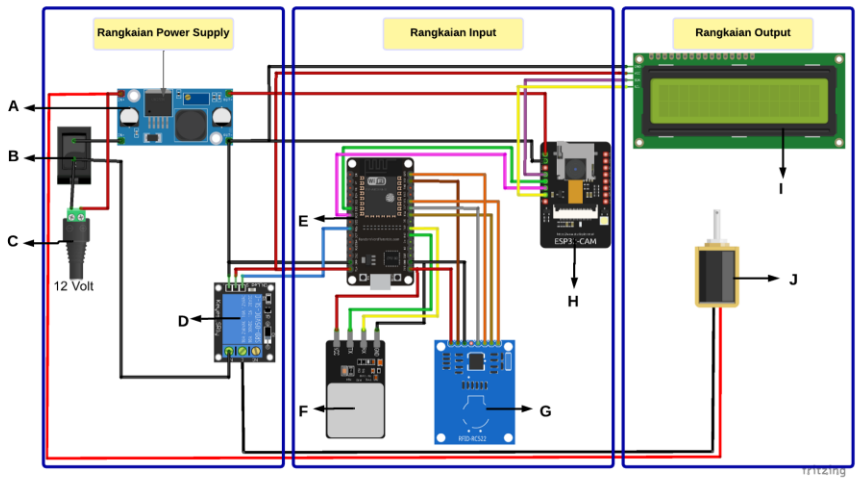
Gambar III. 4 Flowchart Proses Akses Reader MFA Menggunakan *Smartcard*, Sidik Jari, dan *Face Recognition*

Dari *flowchart* pada gambar 3.4 dapat dilihat cara mengakses reader menggunakan *smartcard* yang berbasis *contactless* sebagai tahap awal proses autentikasi. User yang membawa *smartcard* melakukan *tapping* pada *reader*. Kemudian *Radio Frequency Identification* (RFID) akan mengaktifkan sensor sidik jari AS608 pada ESP32. ID jari yang terdeteksi akan dikirim ke ESP32-CAM sebagai isyarat agar kamera mengambil gambar. Selanjutnya ID jari dan gambar wajah yang ditangkap kamera secara bersamaan dikirim ke *server* melalui modul *WiFi* untuk dilakukan validasi dan verifikasi. Jika ID jari dan wajah cocok maka server akan merespon agar ESP32 dapat mengontrol *Electromagnetic Door Lock* agar pintu dapat terbuka selama 5 detik. sebaliknya pintu akan tetap terkunci jika ID jari dan wajah tidak sesuai.

III.6. Perancangan dan Pembuatan *Reader*

III.6.1. Perancangan *Hardware*

Perancangan *Hardware* pada reader dilakukan dengan membuat skematik dari komponen-komponen elektronika yang sesuai dengan perancangan reader yang diinginkan seperti terlihat pada gambar 3.5



Gambar III. 5 Skematik *Hardware*

Keterangan gambar:

A. LM2596 Voltage Regulator
B. SPST Switch
C. Power DC Jack Female
D. Relay 5v
E. ESP32
F. AS608
G. RFID RC522
H. ESP32-CAM
I. LCD I2C 16x2
J. Elektromagnetik Door Lock 12 Volt

Perancangan *Hardware* bertujuan untuk merancang *reader* dan rangkaian yang akan digunakan pada sistem *Smart Key Reader*. Berikut ini penjelasan setiap skematik diagram:

1. Rangkaian *Power Supply*

Rangkaian *Supply* merupakan sumber daya pada rangkaian. Daya dari sumber ini diperuntukkan untuk mengontrol sebuah *elektromagnetik*

door lock sebesar 12 volt dan komponen lainnya sebesar 5v dan 3.3v yang membutuhkan daya. Daya dari adaptor 12v disalurkan ke relay dan ke regulator sesuai dengan keperluan.

2. Rangkaian *Input*

Rangkaian *Input* berupa RFID, *Fingerprint Sensor* AS608, dan ESP32-CAM sebagai kamera. RFID merupakan teknologi yang digunakan untuk melakukan identifikasi dan pengambilan data pada *smartcard*. AS608 merupakan sensor sidik jari yang berkerja dengan menangkap gambar sidik jari yang menempel dipermukaan sensor dan mencocokkan dengan data sidik jari yang tersimpan. RFID dan AS608 terhubung dengan sebuah pengendali yaitu ESP32. ESP32 juga terhubung dengan ESP32-CAM sebagai modul pengendali untuk memotret gambar wajah. Seluruh data yang diperoleh akan dikirim ke *web server* dan *database* menggunakan modul WiFi yang tersedia pada ESP32-CAM.

3. Rangkaian *Output*

Pada rangkaian *output* terdapat LCD 16x2 yang terhubung dengan ESP32-CAM dan berfungsi sebagai tampilan yang membantu user melihat keadaan reader saat ini. Sedangkan *elektromagnetik door lock* terhubung dengan ESP32 dan berfungsi sebagai pengunci magnet agar pintu dapat terbuka selama 5 detik atau tidak.

III.6.2. Perancangan *Software* untuk mikrokontroller

Perancangan *software* untuk mikrokontroller menggunakan text editor Arduino IDE (*Integrated Development Environment*). Mikrokontroller yang digunakan adalah ESP32 dengan bahasa pemrograman C/C++. Adapun tahap perancangannya dimulai dengan menghubungkan program dengan library disetiap komponen yang dibutuhkan. Kemudian mendefinisikan variable-variabel program seperti wifi, kamera, dan database. Menentukan peran dan kondisi inisiasi komunikasi serial yang menjadi penghubung antar mikrokontroller, wifi, dan database. Membuat fungsi loop seperti fungsi pembacaan kartu, pembacaan sidik jari, pembacaan *database*, dan lain-lain. Untuk lebih rincinya, program *software* mikrokontroller yang dibuat dapat dilihat pada lampiran I.

III.6.3. Perancangan *Web Server* dan *Website Login*

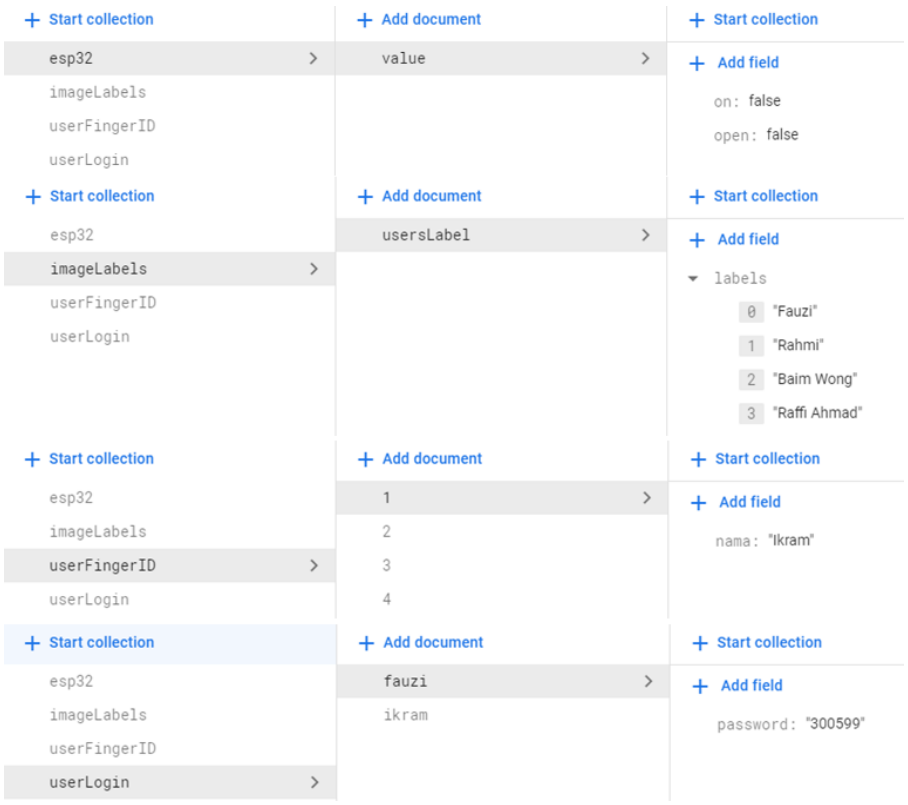
Web server dan *website* halaman *login* dirancang untuk melayani *request* dari *client* yang ingin melakukan proses pengenalan wajah pada gambar wajah yang dipotret ataupun melakukan proses validasi data user pada database. *Software* yang digunakan dalam pembuatan *web server* adalah Node.js dengan menggunakan bahasa pemrograman Javascript.

Web server ini dibangun menggunakan beberapa modul javascript yang tersedia di npm (node package manager) yaitu *express.js, multer, face-api.js* dan ejs. *Express.js* menyediakan kerangka kerja kecil dan kuat untuk mengatur jalur request HTTP, sehingga dapat digunakan untuk pembuatan *web application* dan *website*. *Multer* adalah middleware dari node.js

digunakan untuk menangani multipart/form-data, yang biasa digunakan untuk peng-*upload*-an file - file. Pada web server yang dibuat *multer* digunakan untuk menangani POST *request* terhadap gambar dan ID jari yang dikirim dari ESP32-CAM dan menyimpannya ke folder imageUpload. *Face-api.js* adalah modul pengenal wajah javascript untuk browser dan node.js dan digunakan untuk mengolah gambar wajah yang tersimpan di folder imageUpload dan dicocokkan terhadap data gambar user terdaftar yang tersimpan di folder image. Modul *face-api.js* bekerja dengan mendeteksi wajah pada gambar input dan gambar referensi wajah yang tersimpan kemudian memetakan beberapa titik yang mendeskripsikan wajah, mata, alis, hidung, dan mulut, setelah itu menghitung jarak antara titik yang bersesuaian pada kedua gambar yang dibandingkan. Semakin kecil jarak yang dihasilkan maka semakin akurat hasil pengenalan wajahnya. Kemudian membuat halaman *login website* menggunakan *template engine* ejs sebagai sarana akses *user* terhadap *reader*. *Web server* yang telah bekerja selanjutnya di-*deploy* ke sebuah *cloud platform* Heroku. Untuk lebih rincinya program *web server* dan *website login* dapat dilihat pada lampiran 3 sampai lampiran 9.

III.6.4. Perancangan *Database*

Perancangan *database* untuk *reader* ini menggunakan firebase sebagai tempat media penyimpanan database yang akan diintegrasikan dengan program pada Arduino IDE untuk mikrokontroller dan program untuk *web server* yang akan dibangun.



Gambar III. 6 Tampilan *Database*

Tampilan *database* yang dirancang pada firebase dapat dilihat pada gambar 3.6 yang menggunakan model perancangan *database* dalam bentuk *document*, yang disusun menjadi *collection*. Setiap *document* berisi kumpulan pasangan *key-value*. Adapun penjelasannya sebagai berikut:

a. esp32, sebagai format perintah untuk mengakses dan mengontrol *reader* yang berisi *value* dan terdiri dari dua pasang *key-value*:

- *on*, merupakan *key* yang dirancang dengan nilai Boolean “true” untuk mengakses reader dan “false” untuk menolak akses reader.
- *open*, merupakan *key* yang dirancang dengan nilai Boolean “true” untuk membuka akses pintu dan “false” untuk menutup pintu.

b. imageLabels, sebagai format data user yang menyimpan array *labels* yang memuat nama dari user yang data gambar wajahnya tersimpan di server.
c. userFingerID, sebagai format data yang menyimpan ID jari user dalam bentuk *document* dan setiap *document* berisi pasangan *key-value* nama user.
d. userLogin, sebagai format dari data pengguna yang menyimpan *username* sebagai *document* dan pasangan *key-value password* untuk melakukan login melalui halaman web.

III.7. Pengujian Kinerja *Reader*

Setelah *reader* dibuat sesuai dengan rancangan yang telah disempurnakan, maka perlu diadakan pengujian kinerja *reader*. Jika *reader* dapat menyelesaikan permasalahan yang telah dirumuskan dan luaran yang diharapkan dapat tercapai, maka proses pembuatan *reader* selesai. Tetapi jika *reader* belum bias menyelesaikan permasalahan yang telah dirumuskan dan luaran yang diharapkan belum bisa tercapai, maka akan dilakukan analisis ulang. Secara garis besar ada dua pengujian yaitu pengujian *hardware* dan pengujian *software*.

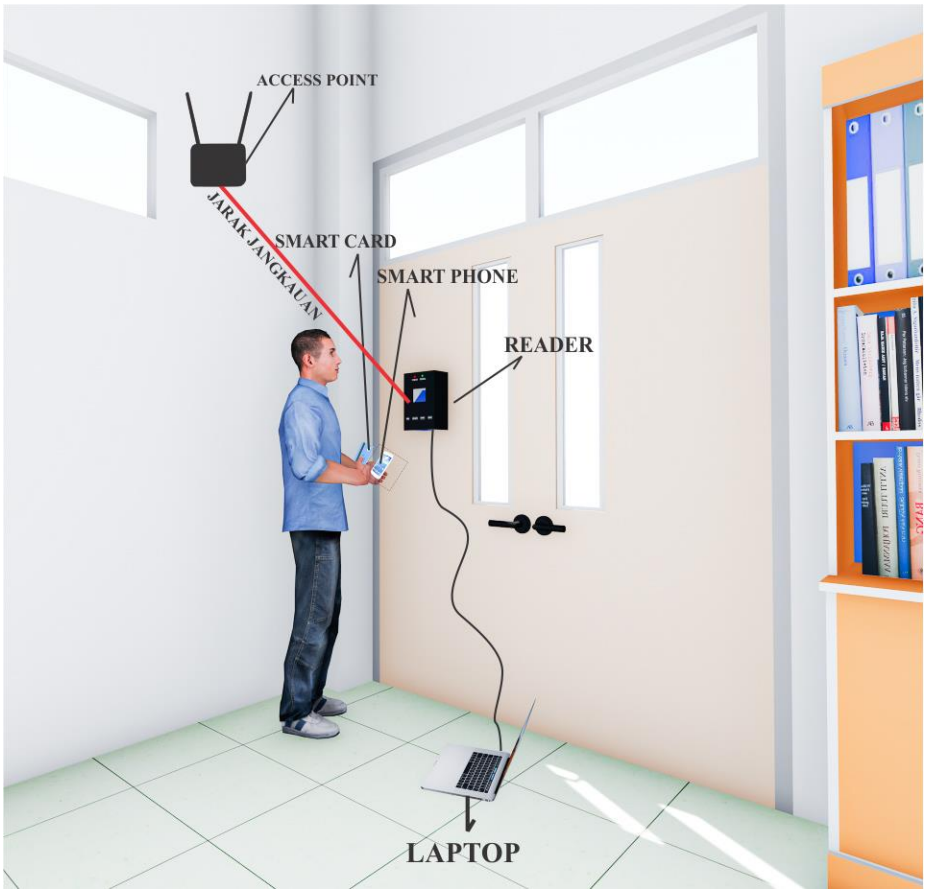
Langkah pengujian *reader* :

1. Memastikan perangkat terhubung ke sumber listrik.
2. Mengaktifkan WiFi atau Hotspot.
3. Membuka aplikasi Arduino IDE dan menyambungkan ESP32 yang ada pada *reader* sehingga dapat menampilkan *serial monitor* untuk melihat *output* saat program sedang berjalan.
4. Mengaktifkan perangkat melalui *switch*.

5. Memastikan tampilan awal LCD *reader* “Ready” yang artinya perangkat telah terhubung ke WiFi atau Hotspot.
6. Melakukan *tapping smartcard* pada reader dan akan muncul hasil dari serial monitor, dan melakukan pencocokan data dari data yang telah disimpan.
7. Muncul tampilan pada LCD dengan ucapan “Akses diterima” dan “Masukkan sidik jari dan wajah”.
8. Menempelkan sidik jari ke *fingerprint sensor* pada reader dan akan muncul hasil dari *serial monitor*, dan melakukan pencocokan data dari data yang telah disimpan.
9. Menempatkan wajah didepan reader setelah meletakkan sidik jari, data wajah dan ID sidik jari yang dibaca akan dikirim bersamaan ke *web server* untuk dilakukan pencocokan data.
10. Muncul tampilan pada LCD ucapan “Selamat datang” dan “Nama Pengguna” lalu kunci pintu terbuka ketika data valid. Sedangkan ketika data tidak valid reader akan melakukan penolakan dan tampilan di LCD muncul kalimat “Tidak terdafatar”.

III.7.1. Pengujian Kinerja *Hardware*

Pada gambar 3.7 merupakan proses pengujian *hardware* menggunakan laptop dan smartphone di depan pintu Computer & PABX Room Lab. Telekomunikasi, Radio, dan Microwave untuk menemukan *error* pada masing-masing perangkat keras. Adapun macam-macam pengujian yang akan diuraikan sebagai berikut.



Gambar III. 7 Pengujian *Hardware*

1. Pengujian Jarak Baca *Reader* dengan *Smartcard*

Pengujian ini dilakukan dengan cara melakukan *tapping smartcard* pada *reader* khususnya pada bagian RFID, kemudian memberikan penghalang berupa kertas karton dengan ketebalan 3 mm sampai dengan *smartcard* tidak terbaca. Hal ini bertujuan untuk mengetahui mejauh mana *reader* dapat membaca *smartcard* dengan atau tanpa penghalang.

2. Pengujian *Delay Scan Smartcard* pada *Reader*

Pengujian ini dilakukan dengan menyambungkan *reader* pada laptop dan membuka *serial monitor* pada *software* Arduino IDE, kemudian melakukan *tapping smartcard* pada *reader* sehingga dapat melihat setiap

kondisi yang diperintahkan pada *reader* setiap saat melalui *serial monitor*. Untuk membaca besar *delay* pada tampilan *serial monitor* yaitu ketika *serial monitor* menunjukkan tampilan pada keadaan "Kartu Terdeteksi" sampai dengan "Selamat datang" atau "Akses ditolak". Dari selisih waktu yang didapatkan pada tampilan *serial monitor* akan menunjukkan waktu tunda *reader* pada saat beroperasi. Hal ini bertujuan untuk mengetahui seberapa lama waktu tunda (*delay*) pada saat *reader* beroperasi setiap sekali perintah dilakukan.

3. Pengujian *Biometric Reader*

    a. Pengujian *Fingerprint Sensor*

        Pengujian ini dilakukan dengan menyambungkan *reader* pada laptop dan membuka *serial monitor* pada software Arduino IDE, kemudian meletakkan sidik jari pada *fingerprint sensor* sehingga dapat melihat setiap kondisi pembacaan melalui *serial monitor*. Untuk menguji keandalan *fingerprint sensor* maka dilakukan pengujian pada delapan sidik jari berbeda dan melihat keberhasilan sensor dalam membaca sidik jari. Untuk membaca besar *delay* pada tampilan *serial monitor* yaitu ketika *serial monitor* menunjukkan tampilan pada keadaan "Image taken" sampai dengan "Found a print match!" atau "Did not find a match". Dari selisih waktu yang didapatkan pada *serial monitor* akan menunjukkan waktu tunda (*delay*) sensor pada saat beroperasi.

b. Pengujian *Face Recognition*

Pengujian ini dilakukan tepat setelah sidik jari terbaca yang mengirim sinyal agar kamera pada ESP32-CAM memotret gambar lalu mengirimkannya ke *web server* dan *web server* mengirim respon ke ESP32-CAM. Untuk menguji keandalan *face recognition* maka dilakukan pengujian pada enam wajah berbeda yang telah terdaftar di *web server* dengan dua kondisi berbeda yaitu saat membuka mata dan menutup mata. Kinerja *face recognition* dapat dinilai melalui keberhasilan sistem dalam mengenali wajah. *Delay* dibaca saat keadaan LCD pada *reader* menampilkan "Image Taken" sampai dengan "Selamat datang [nama pengguna]" atau "Tidak terdaftar". Untuk menghitung selisih waktu antara keadaan tersebut maka digunakan *stopwatch* yang akan menunjukkan waktu tunda (*delay*) saat sistem *face recognition* bekerja.

c. Pengujian Akurasi *Face Recognition*

Pengujian ini dilakukan untuk mengetahui pengaruh banyaknya data gambar referensi wajah yang tersimpan terhadap akurasi sistem pengenal wajah. Pengujian ini akan menghasilkan nilai *distance* dari dua gambar wajah yang dibandingkan. Jika data gambar referensi wajah yang tersimpan lebih dari satu, maka sistem akan membandingkan gambar wajah input terhadap setiap data gambar referensi wajah yang tersimpan dan menghasilkan nilai

distance rata-rata. Semakin kecil nilai *distance* yang diperoleh maka akan semakin akurat hasil pengenalan wajahnya.

d. Pengujian *Delay Biometric Reader*

Pengujian ini dilakukan dengan meletakkan sidik jari pada *fingerprint sensor* dan mengarahkan wajah sejajar dengan kamera. Untuk membaca besar *delay biometric reader* dapat dilihat pada saat jari diletakkan pada permukaan *fingerprint sensor* sampai dengan LCD reader menunjukkan tampilan "Selamat datang [nama pengguna]" atau "Tidak terdaftar". Untuk menghitung selisih waktu antara keadaan tersebut maka digunakan *stopwatch* yang akan menunjukkan waktu tunda (delay) saat *biometric reader* bekerja secara keseluruhan.

4. Pengujian Kekuatan Sinyal WiFi Pada *Reader*

Pengujian ini dilakukan dengan menempatkan *access point* wifi dengan jarak berbeda-beda dari penempatan *reader*. Alat ukur yang digunakan mengukur jarak penempatan *access point* yaitu dengan menggunakan meteran. Hal ini bertujuan untuk mengetahui seberapa jauh *reader* dapat menangkap pancaran sinyal wifi dari *access point*.

III.7.2. Pengujian Kinerja *Software Website Login* melalui *Web Browser*

Pengujian perangkat lunak merupakan proses untuk menemukan *error* pada *software* website halaman *login* dan halaman *user* sebelum dikirim kepada pengguna. Pengujian *software* adalah kegiatan yang ditujukan untuk mengevaluasi atribut atau kemampuan program dan memastikan bahwa itu

memenuhi hasil yang dicari, atau suatu investigasi yang dilakukan untuk mendapatkan informasi mengenai kualitas dari produk atau layanan yang sedang diuji (*under test*). Teknik pengujian sistem yang digunakan pada penelitian ini adalah *blackbox testing*. *Blackbox testing* adalah metode pengujian perangkat lunak yang menguji fungsionalitas aplikasi yang bertentangan dengan struktur internal atau kerja.

Uji kasus dibangun disekitar spesifikasi dan persyaratan aplikasi, yakni, apa yang seharusnya dilakukan. Perancangan uji memilih input yang valid dan tidak valid dan menentukan output yang benar dan tidak benar. Hal yang diuji adalah seluruh tahapan tampilan antarmuka dari halaman login website yang dirancang untuk *user*. Hal ini bertujuan untuk menilai apakah halaman login website yang dirancang sudah sesuai atau belum dengan yang diinginkan.

BAB IV

HASIL DAN PEMBAHASAN

Pada bab ini dijelaskan hasil dan pembahasan berdasarkan perencanaan sistem yang telah dilakukan untuk mengetahui unjuk kerja sistem dan apakah sistem yang telah dibuat bekerja sesuai dengan rancangan. Adapun hasil perakitan dapat terlihat pada gambar.

(a) (b)

Gambar IV. 1 Hasil Perakitan Hardware, (a) Akses Masuk, (b) Akses Keluar

Dari Gambar IV.1 (a) dan (b), *reader* telah dipasang pada pintu Computer dan PABX Room di Laboratorium Telekomunikasi, Radio, dan *Microwave*.

IV.1. Hasil Pengujian *Hardware*

Pada pengujian *hardware* dilakukan pengujian dan analisa dari keseluruhan *reader* yang dibuat sebagai berikut.

IV.1.1. Pengujian Jarak Baca *Reader* dengan *Smartcard*

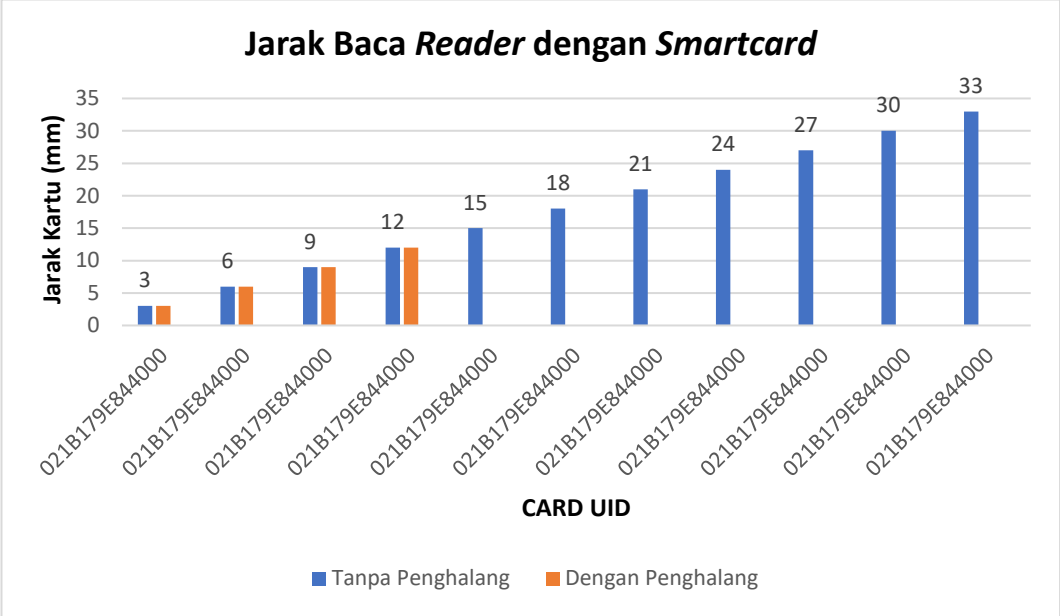
(a) (b)

Gambar IV. 2 Pengujian Jarak Baca *Reader* dengan *Smartcard.* (a) *Tapping Smartcard* Langsung, (b) *Tapping Smartcard* dengan penghalang

Pada pengujian jarak baca *reader* dengan *smartcard* yang ditunjukkan pada Gambar IV.2 dan Tabel IV.1. Pengukuran dilakukan menggunakan penggaris untuk menambah jarak *smartcard* ke *reader* dan menggunakan potongan kardus yang memiliki tebal 3 mm. Pengujian ini bertujuan untuk mengetahui *smartcard* tersebut masih dapat terbaca atau tidak, kemudian akan ditunjukkan pada indikator layar LCD.

Tabel IV. 1 Pengujian Jarak Baca *Reader* dengan cara *Tapping Smartcard* Tanpa Penghalang dan *Tapping Smartcard* dengan Penghalang Kardus

NO.	CARD UID	JARAK (mm)	TAPPING SMARTCARD TANPA PENGHALANG	TAPPING SMARTCARD DENGAN PENGHALANG
1	021B179E844000	0	TERBACA	TERBACA
2	021B179E844000	3mm	TERBACA	TERBACA
3	021B179E844000	6mm	TERBACA	TERBACA
4	021B179E844000	9mm	TERBACA	TERBACA
5	021B179E844000	12mm	TERBACA	TERBACA
6	021B179E844000	15mm	TERBACA	TIDAK TERBACA
7	021B179E844000	18mm	TERBACA	TIDAK TERBACA
8	021B179E844000	21mm	TERBACA	TIDAK TERBACA
9	021B179E844000	24mm	TERBACA	TIDAK TERBACA
10	021B179E844000	27mm	TERBACA	TIDAK TERBACA
11	021B179E844000	30mm	TERBACA	TIDAK TERBACA
12	021B179E844000	33mm	TERBACA	TIDAK TERBACA
13	021B179E844000	36mm	TIDAK TERBACA	TIDAK TERBACA
14	021B179E844000	39mm	TIDAK TERBACA	TIDAK TERBACA
15	021B179E844000	42mm	TIDAK TERBACA	TIDAK TERBACA

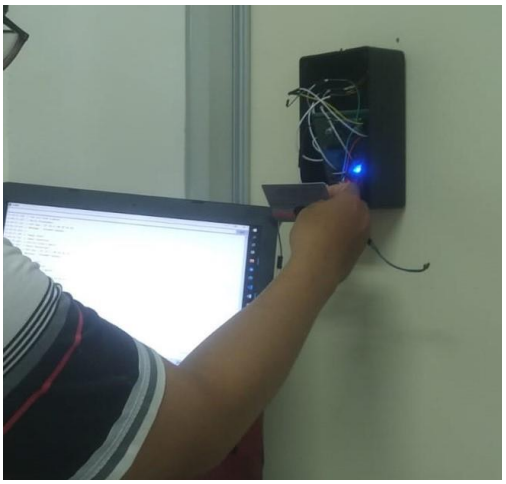


Gambar IV. 3 Jarak Baca *Reader* dengan *Smartcard*

Pada Gambar IV.3 menunjukkan hasil pengujian yang telah dilakukan, dimana *smartcard* dapat terbaca pada jarak maksimum 33mm saat di-*tapping* tanpa penghalang, jika *smartcard* di-*tapping* dengan adanya penghalang yaitu kardus maka *reader* dapat membaca *smartcard* dengan jarak maksimum 12mm, terjadinya perbedaan jarak baca disebabkan dari jenis *smartcard* yang digunakan dan adanya penghalang. Dalam pengujian ini *smartcard* yang digunakan merupakan jenis *smartcard* pasif yang tidak memiliki daya sendiri, sehingga hanya memperoleh daya dari medan gelombang elektromagnetik yang dihasilkan oleh *reader* RFID. Gelombang elektromagnetik RFID mampu menembus material yang bersifat isolator, tetapi karena adanya bahan tersebut maka sebagian daya gelombang elektromagnetik yang dipancarkan oleh *reader* RFID diserap dan dipantulkan oleh bahan tersebut sehingga terjadi pengurangan jarak baca *reader* dari

33mm menjadi 21mm. Pengurangan jarak baca tersebut tidak mempengaruhi sistem secara keseluruhan.

IV.1.2. Pengujian *Delay Scan Smartcard* pada *Reader*



Gambar IV. 4 Pengujian *Delay Scan Smartcard* pada *Reader*

Pengujian *delay scan smartcard* pada *reader* terlihat pada Gambar IV.4. Pengujian ini dilakukan bertujuan untuk mengetahui *delay* yang dibutuhkan oleh *reader* untuk membaca dan memproses *smartcard*. Pengujian dilakukan dengan melakukan *tapping smartcard* ke RFID pada *reader* dengan memperhatikan *output* dari *serial monitor* yang ada pada *software* Arduino IDE.

```
14:31:16.558 -> Masukkan ID Card/Login
14:31:16.558 ->
14:31:27.401 -> Kartu Terdeteksi
14:31:27.401 -> UID tag : D5 06 0C AD
14:31:27.401 -> Message : Akses ditolak
14:31:27.401 ->
14:31:39.962 -> Kartu Terdeteksi
14:31:39.962 -> UID tag : 95 0B AD AC
14:31:39.962 -> Message : Selamat Datang
14:31:39.962 ->
```

Gambar IV. 5 *Serial Monitor* Pengujian *Delay Scan Smartcard* pada *Reader*.

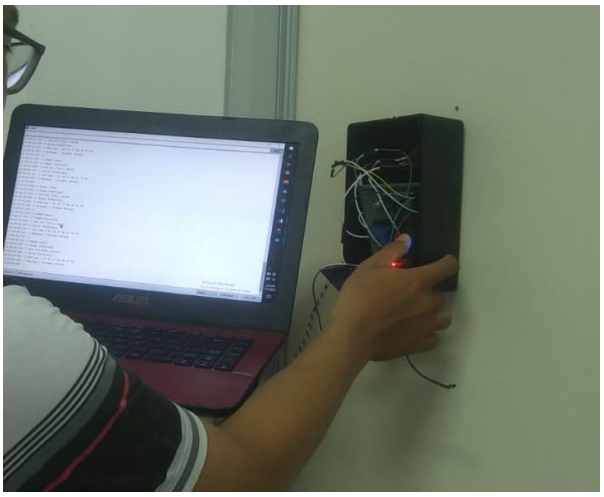
Tabel IV. 2 Pengujian *Delay Scan* Kartu pada *Reader*

NO.	UID KARTU	DELAY
1	158810AD	0ms
2	D5060CAD	0ms
3	85E8A1AC	0ms
4	950BADAC	0ms
5	B171461B	0ms
6	C5F2A8AC	0ms
7	C5D708AD	0ms
8	3519A6AC	0ms

Dari data hasil pengujian dapat dilihat pada tabel diketahui bahwa *reader* dapat bekerja dengan maksimal dalam membaca kartu dengan tidak menghasilkan *delay* sama sekali. Proses pembacaan *delay* dapat dilihat dari *serial monitor* pada Gambar IV.5, dimana *serial monitor* akan menampilkan nilai *delay scan smartcard* yaitu mulai dari waktu terdeteksinya *smartcard* sampai dengan *serial monitor* menampilkan waktu saat kartu terdaftar atau tidak. *Delay* yang diperoleh dalam pengujian ini tergolong sangat bagus.

IV.1.3. Pengujian *Biometric Reader*

a. Pengujian *Fingerprint Sensor*



Gambar IV. 6 Pengujian *Fingerprint Sensor.*

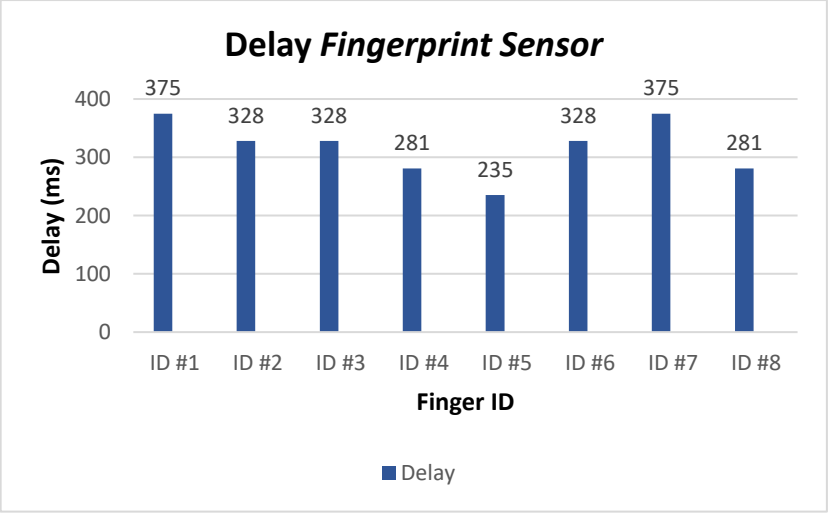
Pengujian *fingerprint sensor* dapat dilihat pada Gambar IV.6. Pengujian ini dilakukan bertujuan untuk mengetahui kemampuan *fingerprint sensor* dalam mengenali sidik jari user dan waktu (*delay*) yang dibutuhkan sensor untuk bekerja. Pengujian dilakukan dengan menempelkan jari pada permukaan *fingerprint sensor* lalu melihat hasil pembacaan pada *serial monitor* yang ada pada *software* Arduino IDE.

```
14:34:04.560 ->
14:34:07.701 -> Image taken
14:34:07.982 -> Image converted
14:34:08.076 -> Found a print match!
14:34:08.076 -> Found ID #1 with confidence of 175
14:34:08.124 ->
14:34:23.450 -> Image taken
14:34:23.637 -> Image converted
14:34:23.684 -> Found a print match!
14:34:23.684 -> Found ID #2 with confidence of 76
14:34:23.716 ->
```

Gambar IV. 7 *Serial Monitor* Pengujian *Fingerprint Sensor*.

Tabel IV. 3 Pengujian Kinerja dan *Delay Fingerprint Sensor*.

No.	Finger ID	Reading Result (Terbaca/Tidak Terbaca)	Delay
1	1	Terbaca	375ms
2	2	Terbaca	328ms
3	3	Terbaca	328ms
4	4	Terbaca	281ms
5	5	Terbaca	235ms
6	6	Terbaca	328ms
7	7	Terbaca	375ms
8	8	Terbaca	281ms



Gambar IV. 8 *Delay Fingerprint Sensor*

Dari data hasil pengujian dapat dilihat pada Tabel IV.3 dan Gambar IV.8 diketahui bahwa *fingerprint sensor* dapat membaca semua sidik jari *user* yang terdaftar dengan baik dan tidak terjadi *error*. Waktu pembacaan dari *fingerprint sensor* bervariasi mulai dari 235ms hingga 375ms. Hasil pengujian dapat dilihat dari serial monitor pada Gambar

IV.7, dimana serial monitor akan menampilkan hasil pembacaan dan delay dari *fingerprint sensor*. Perhitungan delay dimulai dari waktu diambilnya gambar sidik jari sampai dengan menampilkan Finger ID yang terdaftar. *Delay* yang diperoleh tergolong bagus, semuanya kurang dari 1 detik sesuai dengan spesifikasi *fingerprint sensor* AS608.

b. Pengujian *Face Recognition*

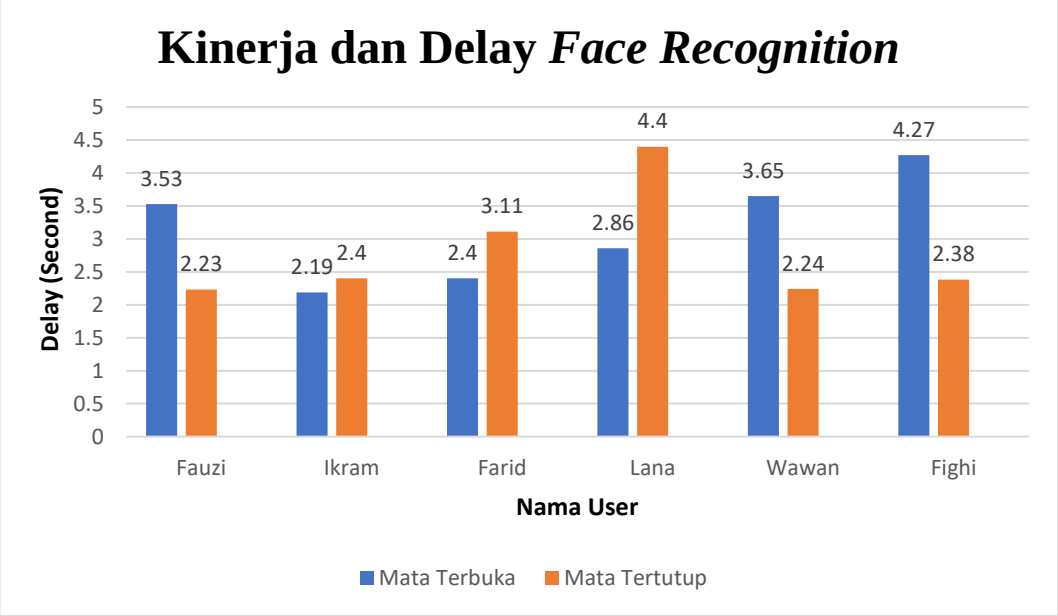
(a) (b)

Gambar IV. 9 Pengujian *Face Recognition*. (a) Mata Terbuka, (b) Mata Tertutup

Pengujian *face recognition* dapat dilihat pada Gambar IV.9. Pengujian ini dilakukan untuk mengetahui kinerja dari sistem *face recognition* dalam mengenali wajah user terdaftar melalui dua kondisi yaitu saat mata user terbuka atau tertutup dan menghitung waktu yang dibutuhkan oleh sistem untuk mengenali wajah. Pengujian dilakukan dengan menempatkan wajah user didepan kamera lalu melihat hasil pengenalan wajah di LCD *reader* dan delay yang dihasilkan menggunakan *stopwatch*.

Tabel IV. 4 Pengujian Kinerja dan Delay *Face Recognition*

No.	Nama User	Scan Wajah Mata Terbuka		Scan Wajah Mata Tertutup	
		Hasil Scan	Delay	Hasil Scan	Delay
1	Fauzi	Terbaca	3.53s	Terbaca	2.23s
2	Ikram	Terbaca	2.19s	Terbaca	2.40s
3	Farid	Terbaca	2.40s	Terbaca	3.11s
4	Lana	Terbaca	3.86s	Terbaca	4.40s
5	Wawan	Terbaca	3.65s	Terbaca	2.24s
6	Fighi	Terbaca	4.27s	Terbaca	2.38s

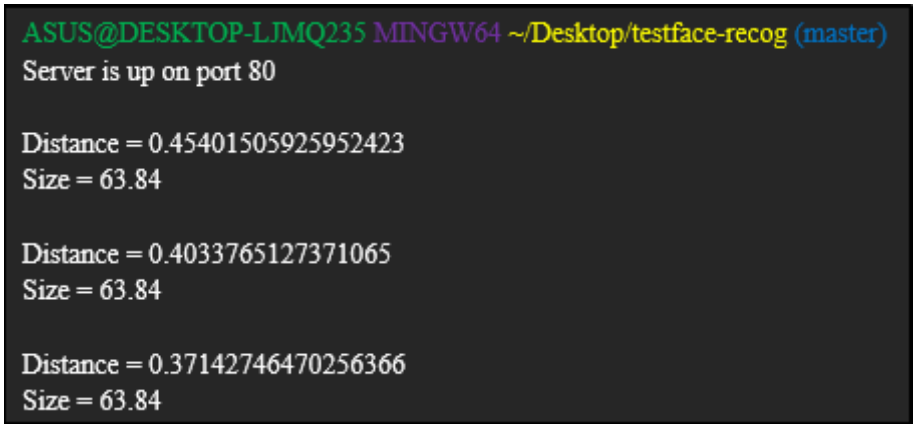


Gambar IV. 10 Pengujian Kinerja dan *Delay Face Recognition*

Setelah pengujian dilakukan, telah didapatkan data hasil pengujian yang dapat dilihat pada Tabel IV.4 dan Gambar IV.10. Pada data tersebut dapat diketahui bahwa tidak terdapat kegagalan pada pengujian kondisi mata terbuka maupun tertutup. Hal ini menunjukkan bahwa pada kondisi wajah dengan mata terbuka maupun tertutup tidak mempengaruhi kinerja

sistem dalam mengenali wajah. Delay dari *face recognition* beragam mulai dari 2.19 detik sampai 4.4 detik. Pengujian delay dilakukan menggunakan *stopwatch* dan menghitung rentang waktu saat LCD *reader* menampilkan "Image taken" sampai dengan menampilkan '"Selamat datang [nama user]". Tampilan "Image taken" menandakan wajah telah terpotret dan siap di-*upload* ke *web server*, sedangkan tampilan "Selamat datang [nama user]" menandakan *reader* telah menerima *respon* dari *web server*. Nilai delay yang dihasilkan sangat bergantung pada kecepatan/kualitas pengiriman data gambar ke *web server*.

c. Pengujian Akurasi *Face Recognition*



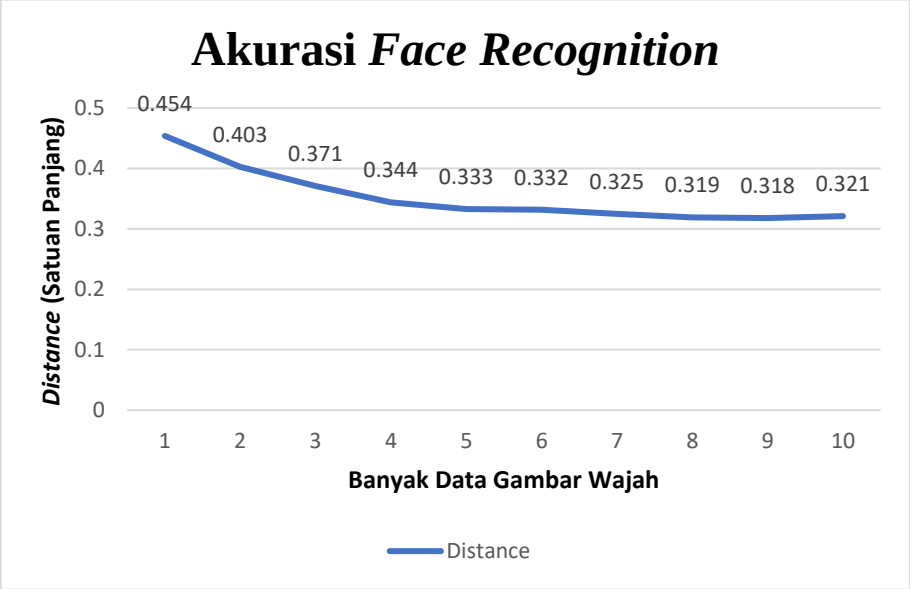
Gambar IV. 11 Tampilan Terminal Pengujian Akurasi *Face Recognition*

Pengujian akurasi *face recognition* dapat dilihat pada tampilan terminal pada Gambar IV.11. Pengujian ini dilakukan untuk mengetahui pengaruh banyak data gambar referensi wajah yang tersimpan terhadap nilai *distance* yang dihasilkan sistem *face recognition*. Pengujian dilakukan dengan menggunakan satu gambar wajah input yang sama kemudian akan dibandingkan terhadap data gambar referensi wajah yang

tersimpan dengan jumlah yang berbeda-beda dari satu sampai 10. Hasil yang didapatkan dapat dilihat pada tampilan terminal.

Tabel IV. 5 Pengujian Akurasi *face recognition*

Jumlah Data Gambar Wajah	Distance
1	0.454
2	0.403
3	0.371
4	0.344
5	0.333
6	0.332
7	0.325
8	0.319
9	0.318
10	0.321



Gambar IV. 12 Pengujian Akurasi *Face Recognition*

Dari data hasil pengujian yang ditampilkan pada Gambar IV. 12, dapat dilihat pengaruh banyak data gambar referensi wajah terhadap nilai *distance* yang dihasilkan oleh *face recognition* mulai saat data gambar referensi wajah hanya satu sampai dengan sepuluh. Saat data gambar referensi wajah hanya satu, nilai *distance* yang dihasilkan relatif besar yaitu 0.454 dan saat data gambar referensi wajah ada sepuluh, nilai *distance* yang dihasilkan relatif kecil yaitu 0.321. Pada grafik dapat dilihat bahwa nilai *distance* relatif mengecil seiring dengan bertambahnya data gambar wajah yang tersimpan. Hal ini disebabkan karena banyak nya gambar referensi wajah sehingga sistem dapat membandingkan wajah input tidak hanya pada satu gambar sehingga didapatkan nilai-nilai *distance* yang beragam yang menyebabkan nilai rata-rata *distance* mengecil. Nilai rata-rata *distance* yang mendekati nol menandakan semakin akurat hasil pembacaan dari sistem *face recognition*.

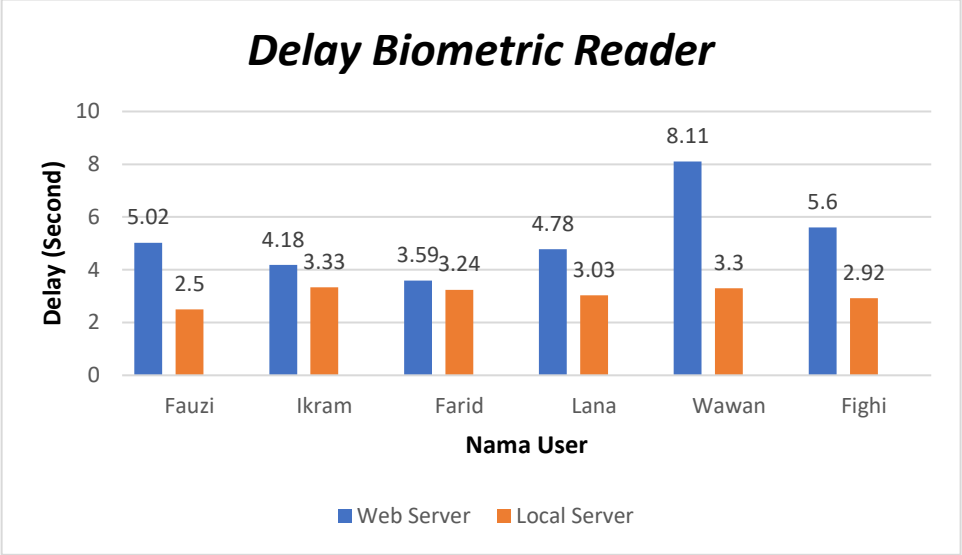
d. Pengujian Delay *Biometric Reader*

Gambar IV. 13 Pengujian *Delay Biometric Reader*.

Pengujian *Delay Biometric Reader* dapat dilihat pada Gambar IV.13. Pengujian ini dilakukan untuk mengetahui waktu kerja dari *Biometric Reader* saat menggunakan *web server* dan *local server* mulai dari pembacaan sidik jari sampai dengan pengenalan wajah. Delay dihitung menggunakan *stopwatch* saat user pertama kali menempelkan sidik jari ke *fingerprint sensor* lalu menempatkan wajah didepan kamera yang akan segera memotret jika sidik jari terdaftar. *Stopwatch* dihentikan saat LCD *reader* menampilan "Selamat datang [nama user]" atau "Tidak terdaftar" yang artinya proses autentikasi biometrik telah selesai.

Tabel IV. 6 Pengujian *Delay Biometric Reader*

No.	Nama User	Delay	
		Web Server	Local Server
1	Fauzi	5.02s	2.50s
2	Ikram	4.18s	3.33s
3	Farid	3.59s	3.24s
4	Lana	8.11s	3.03s
5	Wawan	6.41s	3.30s
6	Fighi	5.60s	2.92s

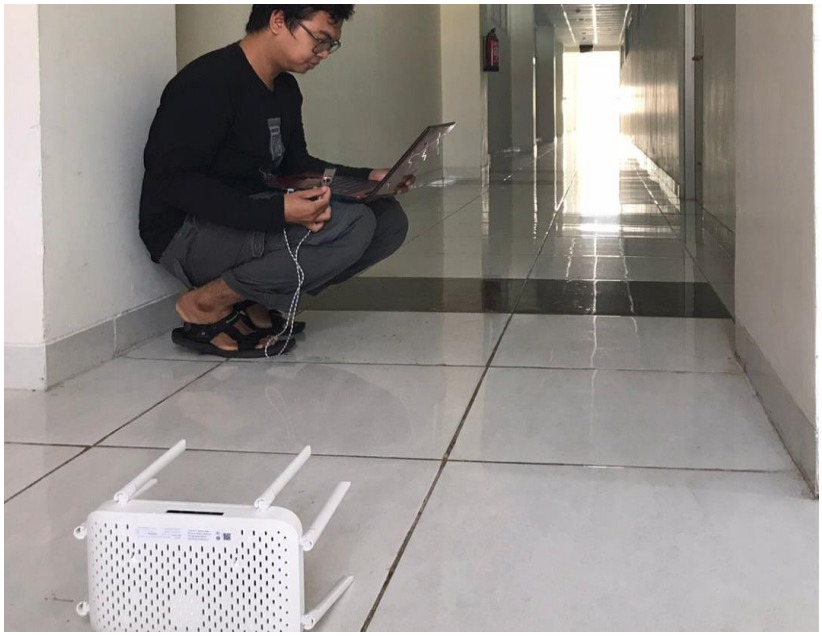


Gambar IV. 14 Pengujian *Delay Biometric Reader*

Dari data hasil pengujian yang dapat dilihat pada Gambar IV.14 dapat dilihat perbandingan delay yang dihasilkan oleh *biometric reader* saat menggunakan *web server* atau *local server*. Nilai delay *biometric reader* saat menggnakan *web server* bervariasi mulai dari 3.59 detik hingga 8.11 detik. Sedangkan saat menggunakan *local server* nilai delay mulai dari 2.50 detik hingga 3.33 detik. Nilai delay tertinggi yaitu 8.11 detik disebabkan oleh banyak faktor. Beberapa faktor yang menyebabkan besarnya delay adalah kualitas dan kecepatan dari mikrokontroller dan jaringan internet dalam mengirim data gambar yang tidak stabil. Nilai delay saat menggunakan *web server* lebih tinggi dibanding saat menggunakan *local server*. Hal ini disebabkan karena program yang di-*deploy* pada web server berada di jaringan yang berbeda. Sehingga data gambar yang dikirim oleh *reader* harus melewati beberapa jaringan untuk sampai ke *web server* yang mengakibatkan munculnya *delay*. Berbeda

saat menggunakan *local server* yang berada di jaringan yang sama dengan *reader* sehingga *delay* yang timbul tidak terlalu besar.

IV.1.4. Pengujian Kekuatan Sinyal WiFi Pada *Reader*

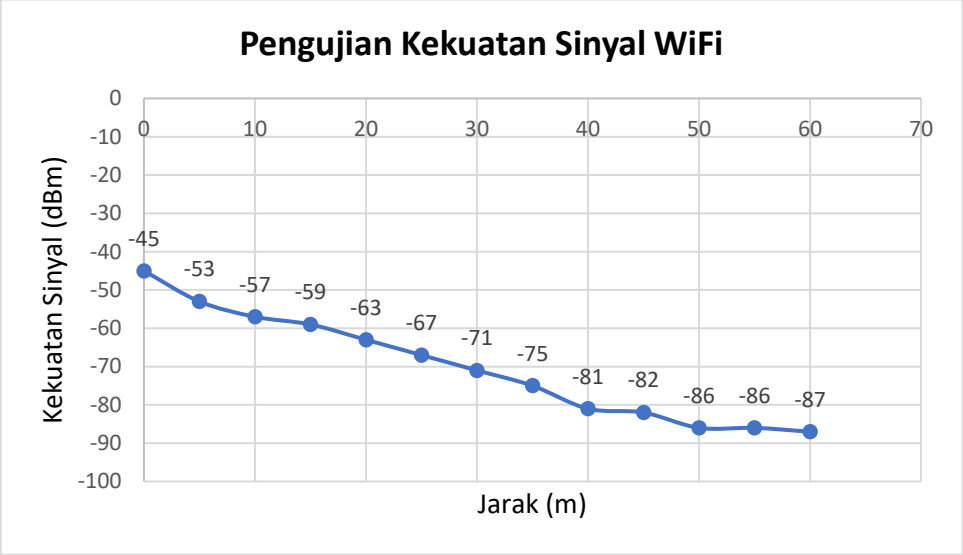


Gambar IV. 15 Pengujian Kekuatan Sinyal WiFi pada *Reader*

Pada Gambar IV.15 dan Tabel IV.7 dapat dilihat proses pengujian kekuatan sinyal, pengujian ini dilakukan untuk mengetahui cakupan jarak terjauh *reader* dapat diletakkan dari *access point*, karena dalam proses pengiriman data dari *reader* menggunakan WiFi.

Tabel IV. 7 Pengujian Kekuatan Sinyal WiFi

No.	JARAK	KEKUATAN SINYAL (dBm)	KETERANGAN
1	0 meter	-45	Terbaca
2	5 meter	-53	Terbaca
3	10 meter	-57	Terbaca
4	15 meter	-59	Terbaca
5	20 meter	-63	Terbaca
6	25 meter	-67	Terbaca
7	30 meter	-71	Terbaca
8	35 meter	-75	Terbaca
9	40 meter	-81	Terbaca
10	45 meter	-82	Terbaca
11	50 meter	-86	Terbaca
12	55 meter	-86	Terbaca
13	60 meter	-87	Terbaca



Gambar IV. 16 Pengujian Kekuatan Sinyal WiFi

Dari Gambar IV.16 dapat dilihat besaran kekuatan sinyal WiFi dari *reader* yang diuji di koridor Laboratorium Telekomunikasi, Radio, dan *Microwave* sampai Laboratorium Listrik Dasar tanpa halangan (*Line of Sight*) dengan jarak 0 meter sampai dengan 60 meter. Terdapat variasi kekuatan sinyal yaitu antara -45 dBm sampai dengan -87 dBm. Dari data kekuatan sinyal yang diperoleh masih tergolong cukup baik. Hal lain yang mempengaruhi kecepatan/kualitas pengiriman data pada saat mengoperasikan *reader* adalah jaringan wifi (*provider*) yang digunakan.

IV.2. Hasil Pengujian *Software*

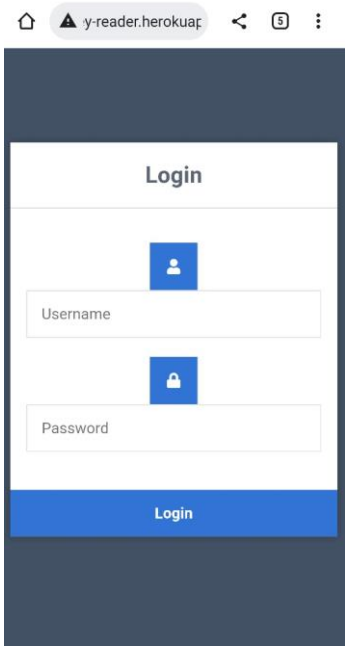
Pada pengujian *software* dilakukan pengujian dan analisa dari keseluruhan halaman *web login* yang dibuat. Halaman web dapat diakses melalui *web browser* dengan alamat url *smart-key-reader.herokuapp.com*.

IV.2.1. Pengujian Antarmuka *Login*

Pengujian yang ditunjukkan pada Tabel IV.8 Dilakukan untuk mengetahui apakah antarmuka *log in* berjalan dengan baik dan dapat memunculkan halaman *login* pada web broswer, dan muncul perintah untuk memasukkan *username* dan *password* untuk pindah ke antarmuka halaman *user* saat menekan *login*.

Tabel IV. 8 Pengujian Antarmuka *Login*

Test Factor	**Hasil**	**Keterangan**
Memunculkan halaman untuk *login* pada web browser saat url *smart-key-reader.herokuapp.com* diakses	√	Web browser menampilkan halaman web untuk *login*
Muncul perintah untuk memasukkan *username* dan *password* untuk pindah ke antarmuka *user* setelah url diakses	√	Muncul perintah untuk memasukkan *username* dan *password* dapat tampil dengan baik dan berpindah ke antarmuka *user* saat menekan *login*
Memasukkan *username* dan *password* yang benar	√	Halaman *login* akan berpindah ke halaman *user*
Memasukkan *username* dan *password* yang salah	√	Halaman *login* menolak untuk berpindah ke halaman *user*
Screenshot		



Ket: √ = Berhasil X = Tidak Berhasil

Antarmuka halaman *login* akan menampilkan halaman untuk *login* saat web browser mengakses url *smart-key-reader.herokuapp.com.* Dengan memasukkan *username* dan *password* lalu menekan tombol *login* maka web browser menuju ke antarmuka halaman *user*.

Berdasarkan pengujian yang dilakukan pada antarmuka halaman *login* terlihat pada Tabel IV.8 Web browser berhasil menampilkan antarmuka halaman *login* saat url *smart-key-reader.herokuapp.com* diakses. Selain itu tombol *login* dapat berfungsi dengan baik dengan menampilkan antarmuka *user* setelah memasukkan *username* dan *password* yang benar lalu menekan *login* dan keadaan sebaliknya akan terjadi ketika memasukkan *username* dan

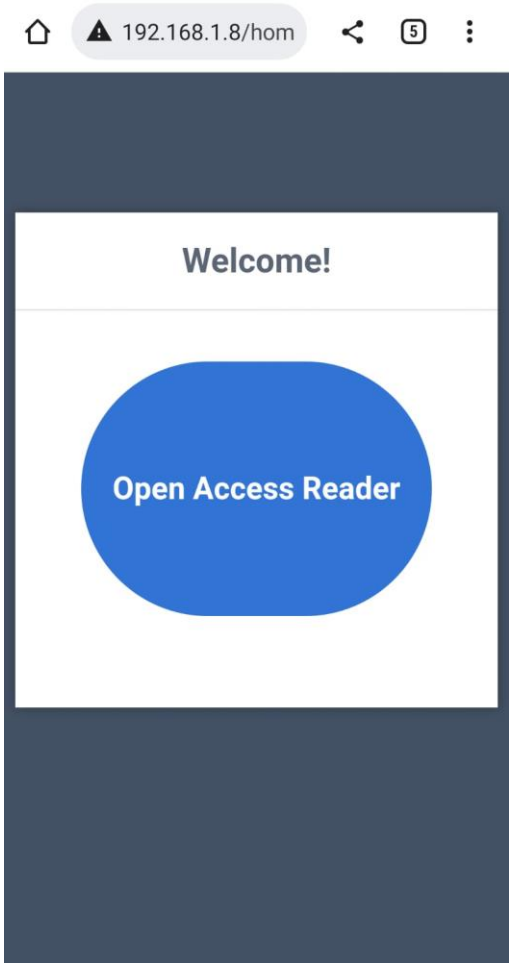
password yang salah lalu menekan tombol *login* web browser akan menolak berpindah ke halaman *user*.

IV.2.2. Pengujian Antarmuka *User*

Pengujian yang ditunjukkan pada Tabel IV.9 Dilakukan untuk mengetahui apakah antarmuka *user* berjalan dengan baik dan menampilkan tulisan "Welcome!". Menampilkan *button Open Access Reader* dan melakukan perintah ke *reader* (akses biometrik terbuka) setelah menekan *button Open Access Reader*. Jika *button Open Access Reader* telah ditekan atau selama lima detik *button Open Access Reader* tidak ditekan, maka *user* secara otomatis akan *logout* dan halaman *user* akan berpindah ke halaman *login*.

Tabel IV. 9 Pengujian Antarmuka *User*

Test Factor	**Hasil**	**Keterangan**
Menampilkan tulisan "Welcome!".	√	Halaman *user* menampilkan tulisan "Welcome!" pada pengguna yang *login*
Menampilkan *button Open Access Reader* dan melakukan perintah ke *reader* (akses biometrik terbuka) setelah menekan *button Open Access Reader*.	√	Halaman web *user* akan melakukan perintah ke *reader* (akses biometrik terbuka) dan akan *logout* secara otomatis dari *user* ke halaman *login* setelah menekan *button Open Access Reader*.
Tidak menekan *button Open Access Reader* selama lima detik.	√	Halaman web *user* secara otomatis berpindah ke halaman *login* jika

		selama lima detik *button Open Access Reader* tidak ditekan.
Screenshot		
		
Ket: √ = Berhasil X = Tidak Berhasil		

Halaman *login* terdiri dari beberapa fitur:

1. *Button Open Access Reader* melakukan perintah ke *reader* (akses biometrik terbuka) setelah ditekan.
2. Halaman *user* secara otomatis *logout* dan berpindah ke halaman *login* setelah *button open access reader* ditekan.

3. Halaman *user* secara otomatis *logout* dan berpindah ke halaman *login* jika selama lima detik *button open access reader* tidak ditekan.

Berdasarkan pengujian yang dilakukan pada antarmuka halaman *user* terlihan pada Tabel IV.9 Antarmuka halaman *user* berhasil dapat berfungsi dengan baik dengan menjalankan masing-masing fitur pada halaman tersebut.

BAB V

PENUTUP

V.1. Kesimpulan

Berdasarkan hasil pengujian yang telah dilakukan, maka dapat disimpulkan sebagai berikut:

1. Sistem dari *Smart Key Reader* dengan *Multi-Factor Authentication* untuk akses pintu di Computer & PABX Room di Laboratorium Telekomunikasi, Radio, dan *Microwave* dapat menggunakan dua mode akses yaitu menggunakan *Smartcard*, *Fingerprint Sensor*, dan *Face Recognition* atau *Web Browser, Fingerprint Sensor,* dan *Face Recognition* untuk mengoperasikan perangkat tersebut.
2. Hasil unjuk kerja dari *Smart Key Reader* dengan *Multi-Factor Authentication* terdiri dari:

a. *Smartcard* dapat terbaca pada jarak maksimum 33mm saat di-*tapping* tanpa penghalang, dan jarak maksimum 12mm saat di-*tapping* dengan penghalang kardus. Terjadinya pengurangan jarak baca RFID dikarenakan adanya daya gelombang yang terserap dan dipantulkan oleh material kardus sehingga daya pancar dari gelombang radio tersebut berkurang.

b. RFID pada *Smart Key Reader* dapat bekerja dengan sangat baik dalam melakukan pembacaan *smartcard*. Data hasil pengukuran menunjukkan nilai *delay* 0ms untuk semua pengukuran pembacaan *smartcard*. Hal ini menandakan *reader* bekerja sangat optimal saat membaca *smartcard*.

c. *fingerprint sensor* dapat membaca semua sidik jari *user* yang terdaftar dengan baik dan tidak terjadi error. Data hasil pengukuran *delay fingerprint sensor* bervariasi mulai dari 235ms hingga 375ms. *Delay* yang diperoleh tergolong bagus, semuanya kurang dari 1 detik sesuai dengan spesifikasi *fingerprint sensor* AS608.

d. Kondisi mata terbuka atau mata tertutup tidak mempengaruhi kinerja dari *face recognition*. Delay dari *face recognition* beragam mulai dari 2.19 detik sampai 4.4 detik. Nilai delay yang dihasilkan sangat bergantung pada kecepatan/kualitas pengiriman data gambar ke *web server*.

e. Semakin banyak gambar referensi wajah user yang tersimpan maka semakin kecil nilai *distance* yang dihasilkan. 1 gambar referensi wajah yang tersimpan menghasilkan nilai *distance* 0.454 satuan panjang. Sedangkan 10 gambar referensi wajah yang tersimpan menghasilkan nilai *distance* 0.321 satuan panjang. nilai *distance* yang semakin mendekati nol maka semakin akurat hasil pembacaan wajahnya.

f. Nilai delay *biometric reader* saat menggunakan *web server* bervariasi mulai dari 3.59 detik hingga 8.11 detik. Sedangkan saat menggunakan *local server* nilai delay mulai dari 2.5 detik hingga 3.33 detik. Hal ini terjadi karena *local server* berada dalam satu jaringan yang sama dengan *reader* sedangkan *web server* berada di jaringan yang berbeda.

g. Besaran kekuatan sinyal WiFi yang diterima *reader* dari *access point* saat dilakukan pengujian di depan koridor Laboratorium Telekomunikasi, Radio, dan *Microwave* sampai Laboratorium Listrik Dasar tanpa halangan

(*Line of Sight*) dengan jarak 0 meter sampai 60 meter. Terdapat variasi kekuatan sinyal yaitu -45 dBm sampai dengan -87 dBm. Kekuatan sinyal WiFi yang diterima tergolong cukup baik.

V.2. Saran

1. Sistem dapat dikembangkan dengan menambahkan berbagai fitur autentikasi yang memanfaatkan *cloud computing* untuk melakukan komputasi data.
2. Jaringan internet yang digunakan pada *reader* sebaiknya menggunakan jaringan yang lebih cepat agar komunikasi sistem ke *server* dan *database* lebih maksimal.
3. Sebaiknya menambahkan indikator peringatan saat pintu tidak tertutup kembali setelah user berhasil mengakses ruangan.

DAFTAR PUSTAKA

[1] A. Ometov, S. Bezzateev, N. Mäkitalo, S. Andreev, T. Mikkonen, and Y. Koucheryavy, "Multi-factor authentication: A survey," *Cryptography*, vol. 2, no. 1, pp. 1–31, 2018, doi: 10.3390/cryptography2010001.

[2] K. D. Septian, S. J. I. Ismail, and A. Sularsa, "Prototipe Sistem Keamanan Face Recognition Berbasis Principal Component Analisis (Pca)," *e-Proceeding Appl. Sci.*, vol. 5, no. 2, pp. 1340–1349, 2019.

[3] B. Santoso and R. P. Kristianto, "Implementasi Penggunaan Opencv Pada Face Recognition Untuk Sistem Presensi Perkuliahan Mahasiswa," *Sistemasi*, vol. 9, no. 2, p. 352, 2020, doi: 10.32520/stmsi.v9i2.822.

[4] A. Shofiudin, *Sistem Pengamanan Ganda Pada Sepeda Motor Menggunakan Sensor Fingerprint Dan Remote Control Rf Berbasis Arduino*. 2020.

[5] W. K. Raharja and B. Santoso, "Purwarupa Alat Telemonitoring Keamanan Ruangan Menggunakan Identifikasi Sidik Jari Berbasis Internet of Things," *Electro Luceat*, vol. 6, no. 2, pp. 156–168, 2020, doi: 10.32531/jelekn.v6i2.227.

[6] C. R. Prihatmoko, "Pengembangan Teknologi Smart Hybrid Reader Untuk Sistem Smart Campus Unhas," p. 6, 2021.

[7] I. Lee and K. Lee, "The Internet of Things (IoT): Applications, investments, and challenges for enterprises," *Bus. Horiz.*, vol. 58, no. 4, pp. 431–440, Jul. 2015, doi: 10.1016/J.BUSHOR.2015.03.008.

[8] O. K. Sulaiman and A. Widarma, "Sistem Internet of Things (Iot) Berbasis Cloud Computing Dalam Campus Area Network," 2017, doi: 10.31227/osf.io/b6m79.

[9] M. A. Amrullah, K. M. Lhaksmana, and D. Adytia, "Pembangunan dan pengujian protokol MQTT & WebSocket untuk Aplikasi IoT Rumah Cerdas

berbasis Android," *e-Proceeding Eng.*, vol. 5, no. 2, pp. 3760–3769, 2018, [Online]. Available: https://libraryeproceeding.telkomuniversity.ac.id/index.php/engineering/article/download/6713/6610

[10] Y. Fauzan, "Kotak Penerima Paket Berbasis IoT Menggunakan Modul ESP32-CAM," 2020.

[11] D. Wahyuni, "Perancangan Prototype Smart Parking System Sebagai Informasi Ketersediaan Tempat Parkir Berbasis Arduino Mega 2560," pp. 1–19, 2019.

[12] F. N. A. Wijaya, Sidik Noertjahjono, and Yosep Agus Pranoto, "Rancang Bangun Sistem Keamanan Pada Sepeda Motor Menggunakan Sms Gateway Berbasis Mikrokontroller," *JATI (Jurnal Mhs. Tek. Inform.*, vol. 4, no. 2, pp. 113–119, 2020, doi: 10.36040/jati.v4i2.2658.

[13] U. Hasanah, "Rancang Bangun Prototype Fingerprint Doorlock Dan Notification Getaran Berbasis Mikrokontroler Arduino," pp. 1–85, 2020.

[14] W. Rankl and W. Effing, *Smart Card Handbook*. 2020. doi: 10.51888/phj.v12i1.59.

[15] M. R. Adhitama and I. G. E. W. Putra, "Analisis Pembacaan dan Penulisan Data Buku Menggunakan RFID Mifare RC-522 untuk Perpustakaan," pp. 23–34, 2019, doi: 10.30864/jsi.v14i1.228.

[16] T. Santoso, "Rancang Bangun Sistem Pembayaran Non Tunai Menggunakan RFID Berbasis Internet of Things," 2019.

[17] A. S. Sudiar, "Sistem Cerdas Keamanan Ruangan Berbasis RFID Dan MAC Address Handphone," 2017, [Online]. Available: file:///C:/Users/User/Downloads/339-879-1-SM (1).pdf

[18] R. H. Suki, A. Zainuri, T. Elektro, and U. Brawijaya, "Implementasi RFID Sebagai Pengaman Pada Sepeda Motor Untuk Mengurangi Tindak

Pencurian," pp. 1–5.

[19] D. Susilo, C. Sari, and G. W. Krisna, "Sistem Kendali Lampu Pada Smart Home Berbasis IOT (Internet of Things)," *ELECTRA Electr. Eng. Artic.*, vol. 2, no. 1, p. 23, 2021, doi: 10.25273/electra.v2i1.10504.

[20] Herianto and K. Pradityo, "Sistem Chatbot untuk Membantu Diagnosa Kerusakan Sistem Komputer," *J. Sains Teknol. Fak. Tek. Univ. Darma Persada*, vol. V, no. 2, 2015.

[21] H. Studiawan, M. C.R., Iqbal, and M. Husni, "Implementasi Klien SIP Berbasis Web Menggunakan HTML5 dan Node.js," *J. Tek. Its*, vol. 1, pp. 242–245, 2012.

[22] Yakip, "Rancang Bangun Sistem Klasifikasi Mineral dan Batuan Menggunakan Tensorflow.js," 2020.

[23] D. Smilkov *et al.*, "TensorFlow.js : Machine Learning For The Web And Beyond".

[24] C. Gabriel and E. Sandoval, "Multiple Face Detection and Recognition System Design Applying Deep Learning in Web Browsers using JavaScript Multiple Face Detection and Recognition System Design Applying Deep Learning in Web Browsers using JavaScript," 2019.

LAMPIRAN

Lampiran 1 Program Arduino untuk ESP32

```
//-----WiFi-------

#include <WiFi.h>

#include <Arduino.h>


//-----Firebase-----

#include <Firebase_ESP_Client.h>

#include <addons/TokenHelper.h>


#define API_KEY "AIzaSyAE-t5wcjetrrM6CKVIoXp0Yb0OIxOGEwo"


#define FIREBASE_PROJECT_ID "userid-db"


#define USER_EMAIL "nurfauzi1999@gmail.com"

#define USER_PASSWORD "fauzi300599"


//-----HardwareSerial---

#define RXD1 32

#define TXD1 33


//------SOLENOID PIN-----

#define SOLENOID 26
```

```
//------TimeElapse
#include <elapsedMillis.h>
//-----RFID-------
#include <SPI.h>
#include <MFRC522.h>
#define SS_PIN 21
#define RST_PIN 22
//------AS608------
#include <Adafruit_Fingerprint.h>
//------PUSH BUTTON PIN------
#define BUTTON_PIN 25

FirebaseData fbdo;
FirebaseAuth auth;
FirebaseConfig config;

elapsedMillis timeElapsedrr;
elapsedMillis timeElapsedread;
elapsedMillis timeElapsedUnread;
elapsedMillis timeElapsedbs;

Adafruit_Fingerprint finger = Adafruit_Fingerprint(&Serial2);
```

```
MFRC522 rfid(SS_PIN, RST_PIN);
WiFiClient client;

const char* ssid = "Lab Telkom";
const char* password = "elektromagnetik";

String webResponse = "";
String strID;
int readCard;
int readDB;
int readState;
int readCardUndetect;
int queue;

void setup() {
  // put your setup code here, to run once:
  SPI.begin();
  rfid.PCD_Init();
  Serial.begin(115200);
  Serial1.begin(115200,SERIAL_8N1, RXD1, TXD1);
  while (!Serial);  // For Yun/Leo/Micro/Zero/...
  delay(100);
```

```
//  Serial.println("I am waiting for card...");
  WiFi.mode(WIFI_STA);
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }
  Serial.println();
  Serial.print("ESP32 IP Address: ");
  Serial.println(WiFi.localIP());

  pinMode(SOLENOID, OUTPUT);
  pinMode(BUTTON_PIN, INPUT_PULLUP);
  digitalWrite(SOLENOID, LOW);

   finger.begin(57600);
  delay(5);
  if (finger.verifyPassword()) {
    Serial.println("Found fingerprint sensor!");
```

```
  } else {
    Serial.println("Did not find fingerprint sensor :(");
    while (1) { delay(1); }
  }

  finger.getTemplateCount();
  Serial.print("Sensor contains "); Serial.print(finger.templateCount); Serial.println(" templates");
  Serial.println("Waiting for valid finger...");

  config.api_key = API_KEY;
  auth.user.email = USER_EMAIL;
  auth.user.password = USER_PASSWORD;
  config.token_status_callback = tokenStatusCallback;
  Firebase.begin(&config, &auth);
  Firebase.reconnectWiFi(true);

  readDB = 1000;
  readCard = 200;
  readState = 100;
  readCardUndetect = 10000;
  queue = 0;
```

```
}

void loop() {
  // put your main code here, to run repeatedly:
  if(Firebase.ready() && timeElapsedrr > readDB && queue == 0){
    //code untuk read database web login
    String documentPath = "esp32/value";
    String mask = "";
    if (Firebase.Firestore.getDocument(&fbdo, FIREBASE_PROJECT_ID,"", documentPath.c_str(), mask.c_str() ))
    {
      FirebaseJson payload;
      payload.setJsonData(fbdo.payload().c_str());
        // Get the data from FirebaseJson object
      FirebaseJsonData jsonData;
      FirebaseJsonData resData;

      payload.get(jsonData, "fields/on/booleanValue", true);
      payload.get(resData, "fields/open/booleanValue", true);

      int buttonState = digitalRead(BUTTON_PIN);
```

```
if(resData.stringValue == "true" || buttonState == 0) {

  Serial.println("Open");
  digitalWrite(SOLENOID, HIGH);
  delay(5000);
  digitalWrite(SOLENOID, LOW);
  timeElapsedrr = 0;
  return;

} else if(jsonData.stringValue == "true"){

  Serial.println("Selamat datang");
  Serial.println();
  Serial1.print("true");
  queue = 1;
  timeElapsedUnread = 0; //penambahan timeElapsedUnread = 0
  timeElapsedrr = 0;

} else {
  timeElapsedrr = 0;
  return;
```

```
    }
   }
   else {
       Serial.println(fbdo.errorReason()+ " 1");
       timeElapsedrr = 0;
   }
}

if(timeElapsedread > readCard && queue == 0){
  //code untuk read RFID
  if ( ! rfid.PICC_IsNewCardPresent()){
    return;
  }
  Serial.println("Kartu Terdeteksi"); //ditambahi
  // Select one of the cards
  if ( ! rfid.PICC_ReadCardSerial()) {
    return;
  }
  //Show UID on serial monitor
  Serial.print("UID tag :");
  String content= "";
  byte letter;
```

```
    for (byte i = 0; i < rfid.uid.size; i++)
    {
      Serial.print(rfid.uid.uidByte[i] < 0x10 ? " 0" : " ");
      Serial.print(rfid.uid.uidByte[i], HEX);
      content.concat(String(rfid.uid.uidByte[i] < 0x10 ? " 0" : " "));
      content.concat(String(rfid.uid.uidByte[i], HEX));
    }
    Serial.println();
    Serial.print("Message : ");
    content.toUpperCase();
    if (content.substring(1) == "02 1B 17 9E 84 40 00") //TUKAR UID INI SESUAI
RFID TAG
    {
      Serial.println("Selamat datang");
      Serial.println();
      Serial1.print("true");
      queue = 1;
      timeElapsedUnread = 0; //penambahan timeElapsedUnread = 0
      timeElapsedread = 0;
    } else {
      Serial.println("Akses ditolak");
      Serial.println();
      Serial1.print("false");
```

```
      delay(3000);
      timeElapsedread = 0;
    }
  }

  if(queue == 1) {
    while (true){
       if (getFingerprintID() == 0 || timeElapsedUnread > readCardUndetect){
//penambahan code "|| timeElapsedUnread > readCardUndetect"
         break;
       }
      }
    delay(2000);
    queue = 0;
  }
}
uint8_t getFingerprintID() {
  uint8_t p = finger.getImage();
  switch (p) {
    case FINGERPRINT_OK:
      Serial.println("Image taken");
      break;
    case FINGERPRINT_NOFINGER:
```

```
//      Serial.println("No finger detected");
      return p;
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("Communication error");
      return p;
    case FINGERPRINT_IMAGEFAIL:
      Serial.println("Imaging error");
      return p;
    default:
      Serial.println("Unknown error");
      return p;
  }

  // OK success!

  p = finger.image2Tz();
  switch (p) {
    case FINGERPRINT_OK:
      Serial.println("Image converted");
      break;
    case FINGERPRINT_IMAGEMESS:
      Serial.println("Image too messy");
```

```
    return p;
  case FINGERPRINT_PACKETRECIEVEERR:
    Serial.println("Communication error");
    return p;
  case FINGERPRINT_FEATUREFAIL:
    Serial.println("Could not find fingerprint features");
    return p;
  case FINGERPRINT_INVALIDIMAGE:
    Serial.println("Could not find fingerprint features");
    return p;
  default:
    Serial.println("Unknown error");
    return p;
}


// OK converted!
p = finger.fingerSearch(); //fingerFastSearch diganti dengan fingerSearch()
if (p == FINGERPRINT_OK) {
  Serial.println("Found a print match!");
} else if (p == FINGERPRINT_PACKETRECIEVEERR) {
  Serial.println("Communication error");
  return p;
```

```
  } else if (p == FINGERPRINT_NOTFOUND) {
    Serial.println("Did not find a match");
//    Serial1.print("unmatch"); //Ditambahi
    return 0;
  } else {
    Serial.println("Unknown error");
    return p;
  }

  // found a match!
  Serial.print("Found ID #"); Serial.print(finger.fingerID);
  Serial.print(" with confidence of "); Serial.println(finger.confidence);
  Serial.println();
  //---Serial Com----
  Serial1.print(finger.fingerID);

//  return finger.fingerID;
    return 0;
}
```

Lampiran 2 Program Arduino untuk ESP32-CAM

```
#include <WiFi.h>

#include <Arduino.h>

#include "esp_camera.h"

#include "soc/soc.h"

#include "soc/rtc_cntl_reg.h"

#include <LiquidCrystal_I2C.h>

#include <Wire.h>


#define RXD1 14

#define TXD1 15

#define SDA 13

#define SCL 2

#define COLUMNS 16

#define ROWS 2

#define LED 4



LiquidCrystal_I2C lcd(0x27, COLUMNS, ROWS);


String fingerReceive = "";//int fingerReceive = 0;

String dataReceive = "";

String getAll;
```

```
String getBody;



const char* ssid = "Lab Telkom";
const char* password = "elektromagnetik";

String serverName = "smart-key-reader.herokuapp.com";
String serverPath = "/uploadImage";
const int serverPort = 80;


WiFiClient client;


// CAMERA_MODEL_AI_THINKER
#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

void setup()
{
  WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0);
  Serial.begin(115200);   // Initiate a serial communication
  Serial1.begin(115200, SERIAL_8N1, RXD1, TXD1);

  WiFi.mode(WIFI_STA);
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
```

```
while (WiFi.status() != WL_CONNECTED) {
  Serial.print(".");
  delay(500);
}
Serial.println();
Serial.print("ESP32-CAM IP Address: ");
Serial.println(WiFi.localIP());

Wire.begin(SDA, SCL);
lcd.init();
lcd.backlight();

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;

// init with high specs to pre-allocate larger buffers
if(psramFound()){
  config.frame_size = FRAMESIZE_VGA;
  config.jpeg_quality = 8;  //0-63 lower number means higher quality
  config.fb_count = 2;
} else {
  config.frame_size = FRAMESIZE_CIF;
  config.jpeg_quality = 10;  //0-63 lower number means higher quality
```

```
    config.fb_count = 1;
  }

// camera init
  esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK) {
    Serial.printf("Camera init failed with error 0x%x", err);
    delay(1000);
    ESP.restart();
  }

  pinMode(LED, OUTPUT);

    lcd.setCursor(0, 0);
    lcd.print("Ready");
    delay(3000);
}
void loop() {

    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Tempelkan ID");
```

```
lcd.setCursor(0, 1);
lcd.print("Card / Login");
Serial.println("Tempelkan ID Card / Login");

while (Serial1.available() > 0) {
   char d = Serial1.read();
   dataReceive += d;
}

if (dataReceive.length() > 0) {
   Serial.println(dataReceive);
   if (dataReceive == "true"){

      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("Akses diterima");
      delay(1000);
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("Masukkan Sidik");
      lcd.setCursor(0, 1);
      lcd.print("Jari dan Wajah");
```

```
        delay(3000);
        Serial.println("Kartu dikenali, Akses Diterima");

    } else if (dataReceive == "false") {

        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Akses ditolak");
        delay(1000);
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("ID Card / User");
        lcd.setCursor(0, 1);
        lcd.print("Tidak Terdaftar");
        delay(3000);

        Serial.println("Kartu tidak dikenali, Akses ditolak");

    } else if (dataReceive == "1" || dataReceive == "2" || dataReceive == "3" ||
dataReceive == "4" || dataReceive == "5"
            || dataReceive == "6" || dataReceive == "7" || dataReceive == "8" ||
dataReceive == "9" || dataReceive == "10"
```

```
                || dataReceive == "11" || dataReceive == "12" || dataReceive == "13" ||
dataReceive == "14" || dataReceive == "15") {
                  fingerReceive = dataReceive;
                  Serial.println(fingerReceive);
                  lcd.clear();
                  lcd.setCursor(0, 0);
                  lcd.print("Scanning...");
                  sendPhoto();
                  lcd.clear();

                  if (getBody == "\nfalse" || getBody =="\n"){

                     lcd.setCursor(0 ,0);
                     lcd.print("Tidak terdaftar");
                     lcd.setCursor(0 ,1);
                     delay(3000);
                     getBody = "";

                     Serial.println("Data Tidak Terdaftar");

                  } else if (getBody == "Connection to " + serverName + " failed.") {

                     lcd.setCursor(0, 0);
```

```
        lcd.print("Server Not");
        lcd.setCursor(0, 1);
        lcd.print("Respond");
        delay(3000);
        getBody = "";
        Serial.println("Server Not Response");

      } else {

        int stringLength = getBody.length();
        String newgetBody = getBody.substring(1, stringLength);
        lcd.setCursor(0,0);
        lcd.print("Selamat Datang");
        lcd.setCursor(0,1);
        lcd.print(newgetBody);
        delay(6000);
        getBody = "";

        Serial.println("Welcome " + getBody);

    }
fingerReceive = "";
```

```
            }


        dataReceive = "";
    }
delay(1000);
}

String sendPhoto() {

//  String getAll;
//  String getBody;

  digitalWrite(LED, HIGH);
  delay(800);
  camera_fb_t * fb = NULL;
  fb = esp_camera_fb_get();
  if(!fb) {
    Serial.println("Camera capture failed");
    delay(1000);
    ESP.restart();
  }
  delay(100);
```

```
  digitalWrite(LED, LOW);
//-----ditambahi-------------
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Image Taken");
//-----ditambahi-------------

  Serial.println("Connecting to server: " + serverName);

  if (client.connect(serverName.c_str(), serverPort)) {
    Serial.println("Connection successful!");
    String head = "--RandomNerdTutorials\r\nContent-Disposition: form-data; name=\"avatar\"; filename=\"esp32-cam.jpg\"\r\nContent-Type: image/jpeg\r\n\r\n";
    String fingerId = "--RandomNerdTutorials\r\nContent-Disposition: form-data; name=\"fingerId\"\r\nContent-Type: text/plain\r\n\r\n";
    String tail = "\r\n--RandomNerdTutorials--\r\n";

    uint32_t dataLen = fingerReceive.length();
    uint32_t imageLen = fb->len;
    uint32_t extraLen = head.length() + tail.length();
    uint32_t totalLen = dataLen + fingerId.length() + imageLen + extraLen;
```

```
    client.println("POST " + serverPath + " HTTP/1.1");
    client.println("Host: " + serverName);
    client.println("Content-Length: " + String(totalLen));
    client.println("Content-Type: multipart/form-data; boundary=RandomNerdTutorials");
    client.println();
    client.print(fingerId);
    char dataBuff[fingerReceive.length() + 1];
    fingerReceive.toCharArray(dataBuff, fingerReceive.length() + 1);
    client.write(dataBuff);

    client.println();
    client.print(head);

    uint8_t *fbBuf = fb->buf;
    size_t fbLen = fb->len;

    for (size_t n=0; n<fbLen; n=n+1024) {
      if (n+1024 < fbLen) {
        client.write(fbBuf, 1024);
        fbBuf += 1024;
      }
      else if (fbLen%1024>0) {
```

```
    size_t remainder = fbLen%1024;
    client.write(fbBuf, remainder);
  }
}
client.print(tail);

esp_camera_fb_return(fb);

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

while ((startTimer + timoutTimer) > millis()) {
  Serial.print(".");
  delay(100);
  while (client.available()) {
    char c = client.read();
    if (c == '\n') {
      if (getAll.length()==0) { state=true; }
      getAll = "";
    }
    else if (c != '\r') { getAll += String(c); }
```

```
        if (state==true) { getBody += String(c); }
        startTimer = millis();
      }
      if (getBody.length()>0) { break; }
    }
    Serial.println();
    client.stop();
//    Serial.print(getBody);
  }
  else {
    getBody = "Connection to " + serverName +  " failed.";
//    Serial.println(getBody);
  }
  return getBody;
}
```

Lampiran 3 Program Nodejs untuk Server

```
const faceRecognition = require('./faceRecog/faceRecognition')
const express = require('express')
const path = require('path')
const db = require('./config')
const session = require('express-session')
const imageUpload = require('./faceRecog/imagePostHandle')
const app = express()
const port = process.env.PORT || 80

faceRecognition.loadAll()
console.log('Loaded');
```

```
//-----------login----------
app.set('view engine', 'ejs')

app.use(session({
  secret: 'secret',
  resave: false,
  saveUninitialized: false,
  cookie: {
    sameSite: true,
    maxAge:5000
  }
}))

app.use(express.json())
app.use(express.urlencoded({ extended: true }))
app.use(express.static(path.join(__dirname, 'static')))

app.get('/', (req, res) => {
    res.render('login')
})

app.post('/auth', async (req, res) => {
  const username = req.body.username
  const password = req.body.password

  const userLoginRef = db.userLogin.doc(`${username}`)
  const doc = await userLoginRef.get()

  if (doc.exists) {
    if (doc.data().password == password) {
      req.session.loggedin = true
      req.session.username = username
      res.redirect('/home')
    } else {
      res.redirect('/home')
    }
  } else {
      res.redirect('/home')
  }
}, (error, req, res, next) => {
  res.status(400).send({ error: error.message})
})

app.get('/home', (req, res) => {
  if (req.session.loggedin) {
```

```
    res.render('user')
  } else {
    res.render('login')
  }
  res.end()
})

app.post('/open', async (req, res) => {
  const esp32Ref = await db.esp32.doc('value').set({on: true, open:
false})
  setTimeout(() => {
    db.esp32.doc('value').set({on: false, open: false})
  },2000)
  req.session.loggedin = false
  res.redirect('/home')
})


app.listen(port, () => {
    console.log('Server is up on port ' + port)
})

//---------------login--------------

//------------ImagePostHandler------
app.post('/uploadImage', imageUpload.single('avatar'), async(req,
res) => {
    const fingerId = req.body.fingerId
    console.log(fingerId)
    const usersRef = db.users.doc(`${fingerId}`)
    const doc = await usersRef.get()
    const faceRecog = await
faceRecognition.run(`./${req.file['destination']}/${req.file['filena
me']}`)
    console.log(faceRecog)
    if (doc.exists) {
      if (doc.data().nama == faceRecog) {
        const esp32 = await db.esp32.doc('value').set({on:false,
open: true});
        res.send(faceRecog)
        setTimeout(() => {
          db.esp32.doc('value').set({on:false, open: false})
        },2000)
      } else {
        res.send(false)
```

```
      }
    } else {
      console.log('No such document!')
      res.send(false)
    }
 }, (error, req, res, next) => {
     res.status(400).send({ error: error.message })
 })

app.get("/viewimage", (req, res) => {
  res.render('imagetaken')
})
```

Lampiran 4 Program Nodejs untuk Face Recognition

```
require('@tensorflow/tfjs-node')
const faceapi = require('./face-api-node.js')
const canvas = require('canvas')
const db = require('../config')

const { Canvas, Image, ImageData } = canvas
faceapi.env.monkeyPatch({ Canvas, Image, ImageData })
var faceMatcher

async function loadAll() {
    await faceapi.nets.ssdMobilenetv1.loadFromDisk('./weights')
    await faceapi.nets.faceLandmark68Net.loadFromDisk('./weights')
    await faceapi.nets.faceRecognitionNet.loadFromDisk('./weights')
    const labeledFaceDescriptors = await loadLabeledImage()
    faceMatcher = new faceapi.FaceMatcher(labeledFaceDescriptors,
0.6)
}

async function run(qry_Image) {

    const qryImage = await canvas.loadImage(qry_Image)
    const resultQry = await
faceapi.detectSingleFace(qryImage).withFaceLandmarks().withFaceDescr
iptor()

    if (resultQry) {
```

```
        const bestMatch =
faceMatcher.findBestMatch(resultQry.descriptor)
        return bestMatch._label
    }
}

async function loadLabeledImage() {
    const doc = await db.imageLabels.doc('usersLabel').get()
    const labels = doc.data().labels
    // const labels = ['Fauzi', 'Rahmi', 'Baim Wong', 'Raffi Ahmad']
    return Promise.all(
        labels.map(async label => {
            const descriptions = []
            for (let i = 1; i <= 2; i++){
                const ref_Img = await
canvas.loadImage(`./image/${label}/${i}.jpeg`)
                const refResult = await
faceapi.detectSingleFace(ref_Img).withFaceLandmarks().withFaceDescri
ptor()
                descriptions.push(refResult.descriptor)
            }

            return new faceapi.LabeledFaceDescriptors(label,
descriptions)
        })
    )
}

// run('./image/qryimg2.jpeg');
module.exports = {run, loadAll}
```

Lampiran 5 Program Nodejs untuk Upload File Image

```
const multer = require('multer')
const path = require('path')

const imageStorage = multer.diskStorage({
    // Destination to store image
    // destination: 'imageUpload',
          destination: 'static',
      filename: (req, file, cb) => {
          cb(null, file.fieldname + '_'
            + path.extname(file.originalname))
            // file.fieldname is name of the field (image)
```

```
            // path.extname get the uploaded file extension
    }
})

const imageUpload = multer({
    storage: imageStorage,
    limits: {
      fileSize: 1000000 // 1000000 Bytes = 1 MB
    },
    fileFilter(req, file, cb) {
      if (!file.originalname.match(/\.(png|jpeg|jpg)$/)) {
         // upload only png and jpg format
         return cb(new Error('Please upload a Image'))
       }
     cb(undefined, true)
  }
})

module.exports = imageUpload
```

Lampiran 6 Program Antarmuka Halaman Login

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width,minimum-
scale=1.0">
        <meta Http-Equiv="Cache-Control" Content="no-cache"/>
        <meta Http-Equiv="Pragma" Content="no-cache"/>
        <meta Http-Equiv="Expires" Content="0"/>
        <title>Smart Key Reader | Login</title>
        <!-- the form awesome library is used to add icons to our
form -->
        <link rel="stylesheet"
href="https://use.fontawesome.com/releases/v5.7.1/css/all.css">
        <!-- include the stylesheet file -->
        <link href="/style.css" rel="stylesheet" type="text/css">
    </head>
    <body>
        <div class="login">
            <h1>Login</h1>
```

```
            <form action="/auth" method="post">
                    <label for="username">
                        <!-- font awesome icon -->
                        <i class="fas fa-user"></i>
                    </label>
                    <input type="text" name="username"
placeholder="Username" id="username" required>
                    <label for="password">
                        <i class="fas fa-lock"></i>
                    </label>
                    <input type="password" name="password"
placeholder="Password" id="password" required>
                <input type="submit" value="Login">
            </form>
        </div>
    </body>
</html>
```

Lampiran 7 Program Antarmuka Halaman User

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8"/>
        <meta name="viewport" content="width=device-width,minimum-
scale=1.0"/>
        <meta http-equiv="refresh" content="5"/>
        <meta Http-Equiv="Cache-Control" Content="no-cache"/>
        <meta Http-Equiv="Pragma" Content="no-cache"/>
        <meta Http-Equiv="Expires" Content="0"/>
        <title>Door Lock User Accesses</title>
        <!-- the form awesome library is used to add icons to our
form -->
        <link rel="stylesheet"
href="https://use.fontawesome.com/releases/v5.7.1/css/all.css">
        <!-- include the stylesheet file -->
        <link href="/style.css" rel="stylesheet" type="text/css">
    </head>
    <body>
        <div class="control">
            <h1>Welcome!</h1>
            <form action="/open" method="post">
                <input type="submit" value="Open Access Reader">
            </form>
```

```
        </div>
    </body>
</html>
```

Lampiran 8 Program Style Halaman Web

```
* {
    box-sizing: border-box;
    font-family: -apple-system, BlinkMacSystemFont, "segoe ui",
roboto, oxygen, ubuntu, cantarell, "fira sans", "droid sans",
"helvetica neue", Arial, sans-serif;
    font-size: 16px;
}
body {
    background-color: #435165;
}
.login, .success {
    max-width: 400px;
    background-color: #ffffff;
    box-shadow: 0 0 9px 0 rgba(0, 0, 0, 0.3);
    margin: 100px auto;
}
.login h1 {
    text-align: center;
    color: #5b6574;
    font-size: 24px;
    padding: 20px 0 20px 0;
    border-bottom: 1px solid #dee0e4;
}
.login form {
    display: flex;
    flex-wrap: wrap;
    justify-content: center;
    padding-top: 20px;
}
.login form label {
    display: flex;
    justify-content: center;
    align-items: center;
    width: 50px;
    height: 50px;
    background-color: #3274d6;
    color: #ffffff;
}
```

```
.login form input[type="password"], .login form input[type="text"] {
    width: 310px;
    height: 50px;
    border: 1px solid #dee0e4;
    margin-bottom: 20px;
    padding: 0 15px;
}


.login form input[type="submit"] {
    width: 100%;
    padding: 15px;
    margin-top: 20px;
    background-color: #3274d6;
    border: 0;
    cursor: pointer;
    font-weight: bold;
    color: #ffffff;
    transition: background-color 0.2s;
}
.login form input[type="submit"]:hover {
  background-color: #2868c7;
    transition: background-color 0.2s;
}

.control {
    max-width: 400px;
    height: 350px;
    background-color: #ffffff;
    box-shadow: 0 0 9px 0 rgba(0, 0, 0, 0.3);
    margin: 100px auto;
}

.control h1 {
    text-align: center;
    color: #5b6574;
    font-size: 24px;
    padding: 20px 0 20px 0;
    border-bottom: 1px solid #dee0e4;
}

.control form {
    display: flex;
    flex-wrap: wrap;
    justify-content: center;
```

```
    padding-top: 20px;
}

.control form input[type="submit"] {
    width: 250px;
    height: 180px;
    padding: 15px;
   margin-top: 0px;
    background-color: #3274d6;
    border: 0;
    border-radius: 100px;
    cursor: pointer;
    font-weight: bold;
    color: #ffffff;
    transition: background-color 0.2s;
}

.control form input[type="submit"]:hover {
    background-color: #2868c7;
      transition: background-color 0.2s;
}

.control form input[value="Open Access Reader"] {
    font-size: 22px;
}

.success h1 {
    text-align: center;
    color: #5b6574;
    font-size: 24px;
    padding: 20px 0 20px 0;
    border-bottom: 1px solid #dee0e4;
}

.imageTaken {
    width: 800px;
    height: 800px;
}
```

Lampiran 9 Program Konfigurasi Firebase

```
const {initializeApp, applicationDefault, cert} = require('firebase-
admin/app')
const {getFirestore} = require('firebase-admin/firestore')

const serviceAccount = require('./serviceAccount.json')

// Initialize Firebase
initializeApp({
    credential: cert(serviceAccount)
})

const db = getFirestore()
const users = db.collection('userFingerID') // users diubah menjadi
userFingerID
const userLogin = db.collection('userLogin') // Smartphone diubah
menjadi userLogin
const esp32 = db.collection('esp32')
const imageLabels = db.collection('imageLabels')
module.exports = {users, userLogin, esp32, imageLabels}
```