SISTEM KENDALI JEMURAN PINTAR BERBASIS *INTERNET OF THINGS* DENGAN PERINTAH SUARA MENGGUNAKAN ALGORITMA *RAITA*

SKRIPSI

Oleh:
ACHMAD FAHRY BAIHAKI
NIM. 220605110100

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2026

SISTEM KENDALI JEMURAN PINTAR BERBASIS *INTERNET OF THINGS* DENGAN PERINTAH SUARA MENGGUNAKAN ALGORITMA *RAITA*

SKRIPSI

Diajukan kepada:
Universitas Islam Negeri Maulana Malik Ibrahim Malang
Untuk memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh:
ACHMAD FAHRY BAIHAKI
NIM. 220605110100

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2026

HALAMAN PERSETUJUAN

SISTEM KENDALI JEMURAN PINTAR BERBASIS *INTERNET OF THINGS* DENGAN PERINTAH SUARA MENGGUNAKAN ALGORITMA *RAITA*

SKRIPSI

Oleh :
ACHMAD FAHRY BAIHAKI
NIM. 220605110100

Telah Diperiksa dan Disetujui untuk Diuji:
Tanggal: 22 April 2026

Pembimbing I,	Pembimbing II,
Ajib Hanani, M.T	Dr. Ir. M. Amin Hariyadi, M.T
NIP. 19840731 202321 1 013	NIP. 19670018 200501 1 001

Mengetahui,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Supriyono, M. Kom
NIP. 19841010 201903 1 012

HALAMAN PENGESAHAN

SISTEM KENDALI JEMURAN PINTAR BERBASIS *INTERNET OF THINGS* DENGAN PERINTAH SUARA MENGGUNAKAN ALGORITMA *RAITA*

SKRIPSI

Oleh :
ACHMAD FAHRY BAIHAKI
NIM. 220605110100

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Tanggal: 04 Mei 2026

Susunan Dewan Penguji

Ketua Penguji	: Prof. Dr. Muhammad Faisal, M.T NIP. 19740510 200501 1 007	( )
Anggota Penguji I	: A'la Syauqi, M.Kom NIP. 19771201 200801 1 007	( )
Anggota Penguji II	: Ajib Hanani, M.T NIP. 19840731 202321 1 013	( )
Anggota Penguji III	: Dr. Ir. M. Amin Hariyadi, M.T NIP. 19670018 200501 1 001	( )

Mengetahui dan Mengesahkan,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Supriyono, M. Kom
NIP. 19841010 201903 1 012

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama	: Achmad Fahry Baihaki
NIM	: 220605110100
Fakultas / Program Studi	: Sains dan Teknologi / Teknik Informatika
Judul Skripsi	: Sistem Kendali Jemuran Berbasis *Internet of Things* dengan Perintah Suara Menggunakan Algoritma *Raita*

Menyatakan dengan sebenarnya bahwa Skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri, bukan merupakan pengambil alihan data, tulisan, atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila dikemudian hari terbukti atau dapat dibuktikan skripsi ini merupakan hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 04 Mei 2026
Yang membuat pernyataan,

Achmad Fahry Baihaki
NIM. 220605110100

MOTTO

"*Memento mori*"
"*Carpe diem*"

HALAMAN PERSEMBAHAN

Puji dan syukur penulis panjatkan ke hadirat Allah Subhanahu wa ta'ala atas segala limpahan anugerah dan rahmat-Nya sehingga penulis dapat menyelesaikan skripsi ini dengan baik. Tak lupa, shalawat serta salam selalu tercurah kepada Nabi Muhammad Shalallaahu 'Alaihi Wasallam yang telah menata kehidupan umat manusia dari zaman kegelapan hingga zaman kebenaran yang penuh. Halaman persembahan ini dirangkai sebagai ucapan syukur dan terima kasih penulis atas semua pihak yang telah berkontribusi dalam penelitian ini sehingga dapat terselesaikan dengan baik.

Karya ini penulis persembahkan kepada Ibunda dan Ayahanda tercinta yang senantiasa memanjatkan do'a nya serta memberikan dukungan, kasih sayang, dan pengorbanan yang tak terhingga. Tidak lupa, kepada kedua adik penulis dan seluruh keluarga besar yang sangat penulis cintai yang senantiasa memberikan motivasi dan perhatian dalam setiap langkah dalam perjalanan akademik penulis. Kontribusi dan dukungan tulus mereka menjadi ombak besar yang mendorong penulis hingga mencapai tepi keberhasilan dalam penyusunan skripsi ini.

Persembahan ini juga turut penulis sampaikan kepada seluruh Dosen dan staf akademik dalam memberikan pengetahuan, bimbingan, kritik dan saran, serta bantuan yang menjadi landasan penting dalam proses akdemik penulis. Ucapan terima kasih juga penulis utarakan kepada rekan-rekan seperjuangan yang senantiasa menemani dalam suka maupun duka, memberikan dukungan, serta menjadi wadah bertukar ide dan berbagi pengalaman selama proses pendidikan ini berlangsung.

Sebagai akhir, penulis persembahkan karya ini untuk diri sendiri sebagai ungkapan penghargaan atas dedikasi, ketahanan, dan komitmen yang terus dipelihara hingga skripsi ini dapat diselesaikan dengan baik. Keinginan untuk terus bertahan, belajar, dan memperbaiki diri telah menjadi elemen penting dalam perjalanan yang dilalui penulis. Penghargaan ini diharapkan menjadi motivasi untuk terus maju dan memberikan kontribusi yang baik di masa depan.

KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabarakatuh

Alhamdulillahirabbil'alamin segala puji dan syukur penulis panjatkan ke hadirat Allah Subhanahu wa ta'ala atas rahmat, taufik, dan hidayah-Nya yang selalu menyertai setiap langkah penulis dalam menyelesaikan skripsi ini. Atas kemampuan yang diberikan oleh Allah Subhanahu wa ta'ala kepada penulis untuk menempuh perjalanan yang cukup panjang ini sehingga karya yang berjudul "Sistem Kendali Jemuran Pintar Berbasis Internet of Things dengan Perintah Suara Menggunakan Algoritma *Raita*" dapat terwujud sesuai dengan waktu yang telah ditetapkan.

Proses penyusunan penelitian ini membawa penulis melalui berbagai tahapan yang cukup kompleks, mulai dari perencanaan sistem, pengembangan sistem, analisis hasil, hingga tahap akhir penulisan yang penuh dinamika. Beragam pengalaman yang dilalui memberikan banyak pelajaran, di mana setiap kendala menjadi sarana untuk meningkatkan ketekunan, dan setiap dukungan menjadi sumber kekuatan bagi penulis. Berkat kontribusi dan perhatian dari berbagai pihak, skripsi ini dapat diselesaikan dengan baik. Untuk itu, penulis dengan segala kerendahan hati menyampaikan terima kasih kepada:

1. Prof. Dr. Hj. Ilfi Nur Diana, M.Si., selaku Rektor Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang.
2. Dr. H. Agus Mulyono, S.Pd., M.Kes, selaku Dekan Fakultas Sains dan Teknologi UIN Maulana Malik Ibrahim Malang.

3. Supriyono, M. Kom, selaku Ketua Program Studi Teknik Informatika UIN Maulana Malik Ibrahim Malang.
4. Ajib Hanani, M.T., dan Dr. M. Amin Hariyadi, M.T., selaku Dosen Pembimbing yang telah membimbing sekaligus memberikan arahan, saran, dan motivasi dengan penuh kebasaran kepada penulis sehingga bisa menuntaskan skripsi ini.
5. Prof. Dr. Muhammad Faisal, M.T., dan A'la Syauqi, M.Kom., selaku Dosen Penguji yang telah memberikan kritik dan saran yang membangun selama proses penulisan skripsi ini berlangsung.
6. Nia Faricha, S.Si., selaku admin Program Studi Teknik Informatika, atas waktu dan tenaganya dalam memberikan bantuan, informasi, dan dukungan administratif dari proses awal masa studi hingga proses penyusunan skripsi ini terselesaikan.
7. Seluruh dosen, laboran, dan staf Program Studi Teknik Informatika, atas ilmu, bimbingan, semangat, dan dukungan yang diberikan kepada penulis selama masa studi.
8. Ayahanda, Acep Sarifudin dan Ibunda, Mimi Maliayanah, terima kasih yang sebesar-besarnya atas segala do'a, motivasi, kasih sayang, dan segala bentuk usaha yang tak terhingga yang menjadi sumber kekuatan serta inspirasi bagi penulis. Segala perhatian dan dukungan yang diberikan menjadi landasan penulis untuk tetap kuat dan terus berjuang dalam menyelesaikan masa pendidikan hingga akhir.

9. Kedua adik yang sangat penulis banggakan, yaitu Jajang dan Faraaz yang menjadi sumber kekuatan bagi penulis untuk senantiasa berjuang sehingga penulis mampu menyelesaikan pendidikan ini.
10. Seluruh anggota dari keluarga besar di Cakung dan Rawa Kuning, terima kasih atas perhatian, semangat, dan dukungan dalam bentuk apapun yang senantiasa menguatkan penulis dalam menyelesaikan masa studi.
11. Para anggota keluarga "GoosyDude", yaitu Ammar Caplang, Adam Bewok, Ilyas Botek, Guntur Maji, dan Saddam Hadi yang menjadi rumah kedua bagi penulis. Terima kasih atas segala kebersamaan, kesetiaan, canda tawa, motivasi, serta bantuan yang telah disalurkan dari masa SMP hingga masa sekarang. Semoga kesuksesan senantiasa mengiringi dan hubungan baik ini tetap terjalin dengan baik hingga maut memisahkan.
12. Teman-teman "Kontrakan Penghuni Surga", yaitu Adi, Alfred, Gumilang, Syamsul, Fajar, Farros, Rama, Rheza, Afif, Maulana, Muzakky, serta seluruh teman yang tidak dapat disebutkan satu per satu. Karya ini penulis persembahkan untuk kalian yang selama ini menjadi sumber kekuatan, semangat, dan motivasi bagi penulis. Terima kasih atas kebersamaannya dalam situasi suka maupun duka.
13. Bu Pri dan Pak Pri yang sudah penulis anggap sebagai orang tua kedua selama masa perantauan penulis di Kota Malang. Terima kasih atas perhatian dan bantuan yang diberikan. Terima kasih juga atas kenyamanan dan keamanan yang diberikan selama penulis tinggal di "Kontrakan Penghuni Surga".

14. Rekan-rekan Ma'had Ibnu Rusyd Angkatan 2022, khususnya Kamar 20 yaitu, Puji, Farhan, Thoriq, Quthob, Bintang, dan Bang Nawal, yang telah membersamai penulis semenjak masa perkuliahan dimulai. Terima kasih atas segala dukungan dan pengalaman yang diberikan kepada penulis selama di Ma'had.
15. Seluruh warga Program Studi Teknik Informatika, khususnya Angkatan 2022 "Infinity", terima kasih atas kebersamaan, motivasi, pengalaman, dan dukungan yang diberikan kepada penulis selama masa perkuliahan.
16. Teruntuk diri sendiri, terima kasih sudah bertahan dan berjuang sejauh ini untuk tidak menyerah dan berani melangkah menghadapi segala rintangan yang terjadi, walaupun tidak selalu berjalan dengan mudah. Semoga hal baik senantiasa menyertai langkah-langkah selanjutnya di masa depan.

Penulis menyadari bahwa skripsi ini belum mencapai kata sempurna dan memiliki beberapa keterbatasan. Dengan begitu, penulis terbuka untuk menerima saran dan kritik konstruktif demi perbaikan serta pengembangan di masa yang akan datang. Penulis berharap bahwa karya ini bermanfaat tidak hanya bagi pembaca, tetapi juga dapat memberikan sumbangsih yang signifikan untuk masyarakat secara keseluruhan.

Wassalamu'alaikum warahmatullahi wabarakatuh.

Malang, 04 Mei 2026

Penulis

DAFTAR ISI

HALAMAN PERSETUJUAN ..... **iii**
HALAMAN PENGESAHAN ..... **iv**
PERNYATAAN KEASLIAN TULISAN ..... **v**
MOTTO ..... **vi**
HALAMAN PERSEMBAHAN ..... **vii**
KATA PENGANTAR ..... **ix**
DAFTAR ISI ..... **xiii**
DAFTAR GAMBAR ..... **xv**
DAFTAR TABEL ..... **xvi**
ABSTRAK ..... **xvii**
ABSTRACT ..... **xviii**
الملخص ..... **xix**
BAB I PENDAHULUAN **1**
1.1 Latar Belakang 1
1.2 Rumusan Masalah 5
1.3 Tujuan Penelitian 6
1.4 Batasan Penelitian 6
1.5 Manfaat Penelitian 6
BAB II STUDI PUSTAKA **8**
2.1 Penelitian Terkait 8
2.2 *Internet of Things* 12
2.3 *Speech Recognition* 14
2.4 *String Matching* 16
2.5 Algoritma *Raita* 17
2.6 MIT App Inventor 20
BAB III DESAIN DAN IMPLEMENTASI **21**
3.1 Pengumpulan Data 21
3.1.1 Data Primer 22
3.1.2 Data Sekunder 22
3.2 Desain Sistem 22
3.2.1 Tahapan *Input* 23
3.2.2 Tahapan *Process* 24
3.2.3 Tahapan *Output* 31
3.2.4 Desain Komponen 32
3.3 Implementasi Sistem 33
3.4 Pengujian Sistem 33
3.5 Hasil Analisa Sistem 35
BAB IV HASIL DAN PEMBAHASAN **37**
4.1 Implementasi Sistem Kendali 37
4.1.1 Halaman *Software* 37
4.1.2 Implementasi Komunikasi IoT 40
4.1.3 Kendali Jemuran Menggunakan *Speech Recognition* 43
4.1.4 Rangkaian *Hardware* 46

4.2 Hasil Pengujian Sistem 49
4.2.1 Pengujian Sistem Kendali 50
4.2.2 Pengujian Kendali Jemuran Otomatis 52
4.2.3 Pengujian *Error* Sistem 56
4.3 Pembahasan 61
BAB V KESIMPULAN DAN SARAN 66
5.1 Kesimpulan 66
5.2 Saran 67
DAFTAR PUSTAKA
LAMPIRAN

DAFTAR GAMBAR

Gambar 2.1 Contoh Proses *String Matching* 16
Gambar 2.2 Contoh *Bad Character Shift Table* 18
Gambar 2.3 Contoh Proses *Searching* Algoritma *Raita* 19
Gambar 3.1 Desain Penelitian 21
Gambar 3.2 Desain Sistem Kendali Jemuran Pintar 23
Gambar 3.3 Alur Proses *Stemming* 25
Gambar 3.4 Ilustrasi Pencocokan Algoritma *Raita* 31
Gambar 3.5 Desain Komponen Jemuran Pintar 32
Gambar 4.1 *Splash Screen* Aplikasi 38
Gambar 4.2 Halaman Utama Aplikasi 39
Gambar 4.3 Fitur *Google Speech Recognition* Aktif 40
Gambar 4.4 *Dashboard* Cloudflare Tunnel 41
Gambar 4.5 Akses *Endpoint* API status.php dengan Internet 41
Gambar 4.6 *Output* Pesan *Subscribe* di Command Prompt 45
Gambar 4.7 Rangkaian *Hardware* 46
Gambar 4.8 *Motor Stepper* 28BYJ-48 47
Gambar 4.9 Sensor Cahaya LDR LM393 48
Gambar 4.10 Sensor Hujan YL-83 49
Gambar 4.11 (a) Tampilan Aplikasi Perintah Jemuran Keluar
(b) Kondisi Jemuran Keluar 50
Gambar 4.12 (a) Tampilan Aplikasi Perintah Jemuran Masuk
(b) Kondisi Jemuran Masuk 51
Gambar 4.13 Tampilan Aplikasi Perintah Tidak Terdaftar 52
Gambar 4.14 Kondisi Cerah dan Kering 53
Gambar 4.15 Kondisi Gelap dan Kering 54
Gambar 4.16 Kondisi Gelap dan Basah 55

DAFTAR TABEL

Tabel 2.1 Penelitian Terkait 10
Tabel 3.1 Ilustrasi *Stemming* 26
Tabel 3.2 Panjang Kata Kunci *Pattern* "masuk" 28
Tabel 3.3 Indeks Kata Kunci *Pattern* "masuk" 28
Tabel 3.4 *Bad Character Shift Table* 29
Tabel 3.5 Skenario Pengujian Sistem 34
Tabel 3.6 Ilustrasi Alur Pengujian *Error* Sistem 35
Tabel 4.1 Hasil Pengujian Pola Perintah "Masuk" oleh 8 Orang Pengguna 56
Tabel 4.2 Hasil Pengujian Pola Perintah "Keluar" oleh 8 Orang Pengguna 57
Tabel 4.3 Hasil Pengujian Pola Perintah "Jangan Masuk"
oleh 8 Orang Pengguna 58
Tabel 4.4 Hasil Pengujian Pola Perintah "Jangan Keluar"
oleh 8 Orang Pengguna 59

ABSTRAK

Baihaki, Achmad Fahry. 2026. **Sistem Kendali Jemuran Pintar Berbasis *Internet of Things* dengan Perintah Suara Menggunakan Algoritma *Raita*.** Skripsi. Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Ajib Hanani, M.T (II) Dr. Ir. M. Amin Hariyadi, M.T.

Kata kunci: Internet of Things, Jemuran Pintar, MQTT, Raita, Speech Recognition.

Perubahan cuaca yang tidak menentu sering menyebabkan pakaian yang dijemur basah kembali karena hujan datang tiba-tiba saat pemilik rumah tidak berada di rumah. Penelitian ini bertujuan merancang dan mengimplementasikan sistem kendali jemuran pintar berbasis Internet of Things (IoT) dengan perintah suara menggunakan algoritma *Raita* serta mengetahui tingkat *error* sistem. Sistem menggunakan aplikasi Android berbasis MIT App Inventor dengan Google Speech Recognition untuk mengubah suara menjadi teks, kemudian diproses menggunakan algoritma *Raita* untuk mencocokkan empat pola perintah utama, yaitu masuk, keluar, jangan masuk, dan jangan keluar. Perangkat keras menggunakan ESP32, motor stepper 28BYJ-48, sensor LDR, dan sensor raindrop. Komunikasi data dilakukan melalui MQTT. Hasil pengujian sebanyak 480 kali menunjukkan sistem berhasil mengenali 434 perintah dengan tingkat *error* sebesar 9,58%, sehingga sistem memiliki performa yang baik dalam memproses perintah suara sederhana.

ABSTRACT

Baihaki, Achmad Fahry. 2026. **An Internet of Things-Based Smart Clothesline Control System with Voice Commands Using the *Raita* Algorithm.** Thesis. Informatics Engineering Departement. Faculty of Science and Technology Universitas Islam Negeri Maulana Malik Ibrahim Malang. Advisor: (I) Ajib Hanani, M.T (II) Dr. Ir. M. Amin Hariyadi, M.T.

Unpredictable weather changes often cause clothes left out to dry to get wet again due to sudden rain when the homeowner is not at home. This study aims to design and implement a smart clothesline control system based on the Internet of Things (IoT) with voice commands using the Raita algorithm, as well as to determine the system's error rate. The system uses an Android application built with MIT App Inventor and Google Speech Recognition to convert speech into text, which is then processed using the Raita algorithm to match four main command patterns: enter, exit, do not enter, and do not exit. The hardware uses an ESP32, a 28BYJ-48 stepper motor, an LDR sensor, and a raindrop sensor. Data communication is conducted via MQTT. Test results from 480 trials showed that the system successfully recognized 434 commands with an error rate of 9.58%, indicating that the system performs well in processing simple voice commands.
.

Keywords: Internet of Things, MQTT, Raita, Smart Clothesline, Speech Recognition

الملخص

بيهاكي، أحمد فاهري. 2026. نظام التحكم الذكي في مجفف الملابس القائم على إنترنت الأشياء مع الأوامر الصوتية باستخدام خوارزمية رايتا. البحث الجامعي. قسم الهندسة المعلوماتية، كلية العلوم والتكنولوجيا بجامعة مولانا مالك إبراهيم الإسلامية الحكومية مالانج. المشرف الأول: أجيب حناني، ماجستير في الهندسة؛ المشرف الثاني: د. إر. محمد أمين هاريادي، م.ت.
.

الكلمات الرئيسية: MQTT، رايتا، التعرف على الكلام، إنترنت الأشياء، حبل الغسيل الذكي.

غالبًا ما تؤدي التقلبات المناخية إلى عودة الملابس المعلقة إلى حالة الرطوبة بسبب هطول الأمطار المفاجئ في غياب صاحب المنزل. تهدف هذه الدراسة إلى تصميم وتنفيذ نظام تحكم ذكي في حبل الغسيل يعتمد على إنترنت الأشياء (IoT) مع أوامر صوتية باستخدام خوارزمية Raita، بالإضافة إلى تحديد معدل خطأ النظام. يستخدم النظام تطبيق Android يعتمد على MIT App Inventor مع Google Speech Recognition لتحويل الصوت إلى نص، ثم تتم معالجته باستخدام خوارزمية Raita لمطابقة أربعة أنماط رئيسية للأوامر، وهي: الدخول، والخروج، وعدم الدخول، وعدم الخروج. تستخدم الأجهزة ESP32 ومحرك متدرج 28BYJ-48 ومستشعر LDR ومستشعر قطرات المطر. تتم عملية تبادل البيانات عبر MQTT. أظهرت نتائج الاختبار التي أجريت 480 مرة أن النظام نجح في التعرف على 434 أمرًا بمعدل خطأ بلغ 9.58%، مما يعني أن النظام يتمتع بأداء جيد في معالجة الأوامر الصوتية البسيطة.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Kemajuan teknologi *Internet of Things* (IoT) telah mendorong berbagai inovasi dalam otomatisasi aktivitas rumah tangga, termasuk dalam meningkatkan efisiensi dan kenyamanan pengguna (Kaur, 2023). Namun, dalam konteks Indonesia sebagai negara tropis, kondisi cuaca yang tidak menentu, seperti perubahan dari panas ke hujan secara tiba-tiba (BMKG, 2024), masih menjadi permasalahan yang sering dihadapi dalam aktivitas sehari-hari. Ketidakpastian cuaca ini berdampak pada berbagai kegiatan rumah tangga, salah satunya adalah proses menjemur pakaian yang sangat bergantung pada kondisi lingkungan.

Fenomena hujan dalam Islam juga dipandang sebagai salah satu tanda kebesaran Allah Subhanahu wa ta'ala yang menunjukkan kekuasaan-Nya dalam mengatur alam semesta. Sebagaimana yang termaktub dalam *QS. An-Nur (24:43)*:

اَلَمْ تَرَ اَنَّ اللّٰهَ يُزْجِيْ سَحَابًا ثُمَّ يُؤَلِّفُ بَيْنَهٗ ثُمَّ يَجْعَلُهٗ رُكَامًا فَتَرَى الْوَدْقَ يَخْرُجُ مِنْ خِلٰلِهٖۚ وَيُنَزِّلُ مِنَ السَّمَآءِ مِنْ جِبَالٍ فِيْهَا مِنْۢ بَرَدٍ فَيُصِيْبُ بِهٖ مَنْ يَّشَآءُ وَيَصْرِفُهٗ عَنْ مَّنْ يَّشَآءُۗ يَكَادُ سَنَا بَرْقِهٖ يَذْهَبُ بِالْاَبْصَارِ

"Tidakkah engkau melihat bahwa sesungguhnya Allah mengarahkan awan secara perlahan, kemudian mengumpulkannya, lalu menjadikannya bertumpuk-tumpuk. Maka, engkau melihat hujan keluar dari celah-celahnya. Dia (juga) menurunkan (butiran-butiran) es dari langit, (yaitu) dari (gumpalan-gumpalan awan seperti) gunung-gunung. Maka, Dia menimpakannya (butiran-butiran es itu) kepada siapa yang Dia kehendaki dan memalingkannya dari siapa yang Dia kehendaki. Kilauan kilatnya hampir-hampir menghilangkan penglihatan." (QS. An-Nur: 43)

Ayat tersebut menjelaskan proses terbentuknya hujan melalui penggiringan awan oleh angin, penggabungan awan menjadi tumpukan tebal, hingga keluarnya

butiran air hujan dari celah-celahnya. Menurut kajian ilmiah dalam Tafsir Al-Marāghī, ayat ini menggambarkan tiga tahap ilmiah turunnya hujan, antara lain kondensasi, koalesensi, dan presipitasi, yang sejalan dengan konsep meteorologi modern. Hal ini menunjukkan bahwa Al-Qur'an telah menyinggung mekanisme turunnya hujan secara ilmiah sebagai tanda kekuasaan Allah Subhanahu wa ta'ala (Khasanah, 2023). Dengan demikian, pengembangan teknologi jemuran pintar bebasis IoT yang dapat menyesuaikan diri terhadap kondisi cuaca dapat dipandang sebagai implementasi ilmu pengetahuan untuk menjaga kemaslahatan hidup sesama.

Kondisi cuaca yang tidak menentu tersebut seringkali menyebabkan permasalahan dalam proses menjemur pakaian, seperti pakaian yang telah dijemur menjadi basah kembali akibat hujan yang turun secara tiba-tiba. Selain itu, pengguna harus secara manual memantau kondisi cuaca dan memindahkan jemuran, yang tentunya kurang efisien dan menyita waktu. Permasalahan ini menjadi semakin kompleks bagi pengguna yang memiliki keterbatasan waktu atau sedang tidak berada di rumah (BPS, 2025), sehingga meningkatkan risiko pakaian tidak kering secara maksimal.

Permasalahan yang muncul akibat cuaca tidak menentu dan terbatasnya waktu masyarakat untuk tetap di rumah pada akhirnya menuntut adanya solusi yang mampu meringankan pekerjaan rumah tangga. Dalam perspektif Islam, telah dianjurkan untuk membantu dan memudahkan urusan orang lain. Seperti sabda Rasulullah Shalallaahu 'Alaihi Wasallam dalam hadis sahih riwayat Imam Muslim, No. 2699:

، عَنْ أَبِيْ هُرَيْرَةَ رَضِيَ اللَّهُ عَنْهُ عَنِ النَّبِيِّ صَلَّى اللَّهُ عَلَيْهِ وَسَلَّمَ قَالَ : مَنْ نَفَّسَ عَنْ مُؤْمِنٍ كُرْبَةً مِنْ كُرَبِ الدُّنْيَا نَفَّسَ اللهُ عَنْهُ كُرْبَةً مِنْ كُرَبِ يَوْمِ الْقِيَامَةِ، وَمَنْ يَسَّرَ عَلَى مُعْسِرٍ ، يَسَّرَ اللهُ عَلَيْهِ فِي الدُّنْيَا وَالْآخِرَةِ ، وَمَنْ سَتَرَ مُسْلِمًا ، سَتَرَهُ اللهُ فِي الدُّنْيَا وَالْآخِرَةِ ، وَاللهُ فِي عَوْنِ الْعَبْدِ مَا كَانَ الْعَبْدُ فِي عَوْنِ أَخِيهِ ، وَمَنْ سَلَكَ طَرِيقًا يَلْتَمِسُ فِيهِ عِلْمًا ، سَهَّلَ اللهُ لَهُ بِهِ طَرِيقًا إِلَى الْجَنَّةِ ، وَمَا اجْتَمَعَ قَوْمٌ فِي بَيْتٍ مِنْ بُيُوتِ اللهِ يَتْلُونَ كِتَابَ اللهِ ، وَيَتَدَارَسُونَهُ بَيْنَهُمْ ، إِلَّا نَزَلَتْ عَلَيْهِمُ السَّكِينَةُ ، وَغَشِيَتْهُمُ الرَّحْمَةُ ، وَحَفَّتْهُمُ الْمَلاَئِكَةُ ، وَذَكَرَهُمُ اللهُ فِيمَنْ عِنْدَهُ ، وَمَنْ بَطَّأَ بِهِ عَمَلُهُ ، لَمْ يُسْرِعْ بِهِ نَسَبُهُ

“Dari Abu Hurairah Radhiallahu’anhu, dari Rasulullah Shallallahu’alaihi wasallam bersabda : “Siapa yang menyelesaikan kesulitan seorang mu’min dari berbagai kesulitan-kesulitan dunia, niscaya Allah akan memudahkan kesulitan kesulitannya hari kiamat. Dan siapa yang memudahkan orang yang sedang kesulitan niscaya akan Allah mudahkan baginya di dunia dan akhirat dan siapa yang menutupi (aib) seorang muslim Allah akan tutupkan aibnya di dunia dan akhirat. Allah selalu menolong hambanya selama hambanya menolong saudaranya. Siapa yang menempuh jalan untuk mendapatkan ilmu, akan Allah mudahkan baginya jalan ke syurga. Sebuah kaum yang berkumpul di salah satu rumah Allah membaca kitab-kitab Allah dan mempelajarinya di antara mereka, niscaya akan diturunkan kepada mereka ketenangan dan dilimpahkan kepada mereka rahmat, dan mereka dikelilingi malaikat serta Allah sebut-sebut mereka kepada makhluk disisi-Nya. Dan siapa yang lambat amalnya, hal itu tidak akan dipercepat oleh nasabnya.” (HR. Muslim, no. 2699)

Hadis tersebut menyatakan bahwa Allah Subhanahu wa ta'ala akan selalu membantu hamba-Nya yang membantu saudaranya. Kemudian di dalamnya juga disebutkan bagaimana tindakan seseorang terhadap orang lain dan apa yang Allah Subhanahu wa ta'ala akan berikan kepada mereka (Ginting, 2019). Dengan demikian, pengembangan teknologi yang bertujuan memudahkan urusan rumah tangga dan membantu meringankan pekerjaan dapat dipandang sebagai wujud nyata dari pengindahan nilai tolong-menolong dalam kebajikan dan ketakwaan.

Sejalan dengan upaya untuk mempermudah aktivitas manusia, penelitian oleh (Syahputra & Maulana, 2025) telah mengembangkan prototipe jemuran otomatis berbasis IoT yang memanfaatkan sensor hujan, cahaya, dan kelembapan

untuk mengontrol pergerakan jemuran. Meskipun sistem tersebut efektif dalam merespons perubahan kondisi lingkungan, pendekatan yang sepenuhnya bergantung pada sensor masih memiliki keterbatasan. Sensor tidak selalu mampu mendeteksi kondisi cuaca ekstrem secara akurat dan berpotensi mengalami kegagalan fungsi, sehingga dapat mengurangi keandalan sistem dalam situasi tertentu (Gaddam et al., 2020). Oleh karena itu, diperlukan mekanisme kendali alternatif yang tetap dapat digunakan oleh pengguna ketika sistem otomatis tidak berjalan optimal.

Salah satu pendekatan yang dapat digunakan adalah dengan memanfaatkan perintah suara (*speech recognition*) sebagai media interaksi antara pengguna dan sistem. Dalam penelitian ini, perintah suara diproses menjadi teks menggunakan *Google Speech Recognition* pada *smartphone* Android. Selanjutnya, teks hasil pengenalan suara tersebut perlu dicocokkan dengan pola perintah tertentu menggunakan algoritma *string matching* agar sistem dapat mengenali instruksi pengguna secara cepat dan akurat (Hakak et al., 2019). Hal ini menjadikan pemilihan algoritma *string matching* sebagai aspek penting dalam pengembangan sistem.

Algoritma *Raita* dipilih dalam penelitian ini karena memiliki performa yang efisien dalam proses pencocokan *string*. Raita, 1992 menunjukkan bahwa algoritma ini mampu meningkatkan kecepatan pencarian dengan melakukan pemeriksaan karakter pada bagian tertentu dari pola, sehingga mempercepat proses deteksi ketidakcocokan. Penelitian lain oleh (Jain & Rao, 2013) juga menyatakan bahwa *Raita* memiliki efisiensi yang baik dibandingkan algoritma lain seperti Knuth-

Morris-Pratt (KMP), Boyer-Moore (BM), dan Boyer-Moore-Horspool (BMH). Selain itu, (Maesyaroh et al., 2023) membuktikan bahwa *Raita* memiliki waktu pencarian yang lebih cepat dibandingkan BMH dalam konteks sistem informasi. Namun, penerapan algoritma *Raita* pada sistem IoT berbasis perintah suara masih belum banyak diteliti, sehingga menjadi celah yang ingin diisi dalam penelitian ini.

Berdasarkan uraian tersebut, penelitian ini mengusulkan pengembangan sistem jemuran pintar berbasis IoT yang dilengkapi dengan kendali perintah suara menggunakan algoritma *string matching Raita*. Sistem ini diharapkan dapat memberikan fleksibilitas bagi pengguna dalam mengendalikan jemuran, khususnya bagi mereka yang memiliki mobilitas tinggi. Selain itu, sistem yang dikembangkan diharapkan mampu meningkatkan efisiensi serta mengurangi risiko pakaian basah akibat perubahan cuaca yang tidak menentu, sekaligus memberikan kontribusi dalam pengembangan teknologi *smart home* berbasis IoT yang adaptif.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, rumusan masalah dalam penelitian ini adalah bagaimana merancang dan mengimplementasikan sistem kendali jemuran pintar berbasis IoT dengan perintah suara menggunakan algoritma *Raita* sebagai metode pencocokan *string* pada teks hasil konversi *speech recognition* serta bagaimana tingkat kesalahan sistem tersebut dalam memproses pola perintah yang telah ditentukan berdasarkan pengujian *error* sistem.

1.3 Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah untuk merancang dan mengimplementasikan sistem kendali jemuran pintar berbasis IoT dengan perintah suara menggunakan algoritma *Raita* sebagai metode pencocokan *string* pada teks hasil konversi *speech recognition* serta mengevaluasi tingkat kesalahan sistem dalam mengenali dan memproses pola perintah berdasarkan hasil pengujian *error* sistem.

1.4 Batasan Penelitian

Batasan penelitian dalam penelitian ini dibuat untuk mencegah melebarnya topik dalam penelitian. Adapun batasan-batasan masalah pada penelitian ini adalah:

1. Penggunaan gawai dalam sistem *speech recognition* berupa *smartphone* berbasis Android.
2. Aplikasi sistem kendali dibuat menggunakan MIT App Inventor.
3. Kendali hanya untuk memasukkan dan mengeluarkan jemuran pintar menggunakan bahasa Indonesia dengan *keyword* “masuk”, “keluar”, “jangan masuk”, dan “jangan keluar”.
4. Jemuran pintar berupa prototipe sederhana dengan mekanisme tarik-ulur menggunakan tali dan *motor stepper*.
5. Jemuran pintar hanya menggunakan sensor cahaya (LDR) dan hujan (*raindrop*) untuk membaca kondisi lingkungan.

1.5 Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan manfaat sebagai berikut:

1. Membantu masyarakat, khususnya yang memiliki mobilitas tinggi dalam mengendalikan jemuran secara lebih mudah dan andal, serta menghadirkan solusi alternatif ketika sensor tidak berfungsi optimal.
2. Memberikan kontribusi ilmiah dalam bidang IoT dengan menghadirkan kajian baru mengenai penerapan algoritma *Raita* dalam *speech recognition* pada sistem jemuran pintar.

BAB II

STUDI PUSTAKA

2.1 Penelitian Terkait

Penelitian yang dilakukan oleh (Nindyawati et al., 2021) membahas pengembangan aplikasi Kamus Bahasa Lembak Bengkulu berbasis Android dengan menerapkan algoritma *Raita*. Dalam penelitian ini, *Raita* digunakan untuk mempercepat proses pencarian kosakata pada kamus digital sehingga pengguna dapat menemukan terjemahan dengan lebih cepat. Hasil uji coba menunjukkan bahwa algoritma *Raita* mampu memberikan hasil pencarian dengan waktu yang cukup efisien yaitu sekitar 14,9 *ms*. Kontribusi utama penelitian ini terletak pada pembuktian bahwa *Raita* efektif diterapkan dalam sistem berbasis *mobile* dengan kebutuhan pencarian kata yang presisi.

Keandalan algoritma *Raita* diperkuat oleh (Maesyaroh et al., 2023) yang membandingkan BMH dan *Raita* pada sistem pencarian data mahasiswa. Hasilnya, *Raita* lebih unggul dengan rata-rata waktu pencarian 4,307 *ms* dibandingkan BMH 9,268 *ms*. Hal ini sejalan dengan penelitian (Rozy, 2024) yang mengimplementasikan *Raita* pada Sistem Informasi Manajemen Penjualan Sparepart Mobil, di mana algoritma ini mampu mempercepat pencarian data produk dalam *database* yang besar. Kedua penelitian tersebut menegaskan efektivitas *Raita* untuk skala data besar, meskipun masih terbatas pada sistem informasi. Berbeda dengan penelitian ini yang menerapkan *Raita* dalam teknologi *speech recognition* untuk mengendalikan perangkat IoT.

Dalam konteks *speech recognition*, penelitian oleh (Rimadana et al., 2025) mengembangkan sistem QIRABOX, yaitu *speaker murottal* Al-Qur'an yang dikendalikan melalui perintah suara IoT. Sistem ini memanfaatkan *Google Speech Recognition* untuk mengubah perintah suara menjadi teks secara *real-time*, kemudian diproses lebih lanjut menggunakan algoritma Aho-Corasick untuk pencocokan pola. Dengan dukungan mikrokontroler ESP32 dan protokol komunikasi MQTT, sistem mampu merespons instruksi pengguna secara akurat dan efisien. Hasil pengujian menunjukkan tingkat kesalahan hanya sebesar 2% dalam mengenali perintah suara, serta memperoleh skor *System Usability Scale* (SUS) sebesar 78,75% yang termasuk dalam kategori "Baik". Temuan ini menunjukkan bahwa QIRABOX efektif sebagai solusi teknologi asistif yang mendukung aksesibilitas penyandang tunanetra dalam mengakses murottal Al-Qur'an.

Di sisi lain, terdapat penelitian yang membahas tentang perancangan jemuran otomatis berbasis IoT yang dilakukan oleh (Syahputra & Maulana, 2025). Penelitian tersebut mirip dengan penelitian ini dalam hal pengembangan sistem jemuran pintar yang dapat bergerak otomatis berdasarkan *input* sensor. Namun, penelitian tersebut tidak menerapkan teknologi *speech recognition* maupun algoritma *string matching*. Selain itu, terdapat perbedaan yang mencolok terkait perangkat keras yang digunakan, yaitu NodeMCU ESP8266 dengan *raindrop sensor*, LDR, dan DHT11, sedangkan penelitian ini menggunakan ESP WROOM-32 dengan *raindrop sensor* dan LDR serta menambahkan kendali berbasis suara dengan algoritma *Raita*.

Tabel 2.1 Penelitian Terkait

No	Nama Peneliti	Judul Penelitian	Metode	Hasil Penelitian
1	Nindyawati	Aplikasi Bahasa Lembak Bengkulu Berbasis Android Menggunakan Algoritma *Raita*	Algoritma *Raita*	Penelitian ini menunjukkan bahwa algoritma *Raita* mampu mempercepat proses pencarian kosakata pada kamus digital dua arah (Indonesia–Lembak). Pada panjang kata yang dipasangkan dengan *pattern* (KPP) sejumlah 3 hingga 6 karakter, hasil rata-rata durasi pencarian sebesar 14,9 *ms*, jauh lebih efisien dibandingkan pencarian manual, sehingga pengguna dapat memperoleh hasil terjemahan dengan cepat dan akurat.
2	Siti Maesyaroh, Daswa, Gentur Priguna Suwarto	Perbandingan Algoritma Horspool dan *Raita* pada Pencarian Data Mahasiswa	Algoritma Horspool dan *Raita*	Penelitian ini membandingkan performa Horspool dan *Raita* pada *database* berisi 1003 data mahasiswa. Hasil uji coba memperlihatkan bahwa algoritma *Raita* lebih unggul dengan rata-rata waktu pencarian 4,307 *ms*, sedangkan Horspool membutuhkan 9,268 *ms*. Selain itu, pencarian menggunakan *Raita* lebih konsisten dalam menampilkan hasil yang tepat meski jumlah data besar.
3	Ahmad Rozy	Implementasi *Raita* pada Sistem Informasi Manajemen Penjualan Sparepart Mobil	Algoritma *Raita*	Algoritma *Raita* diimplementasikan untuk mempercepat pencarian data *sparepart* dan kode produk dalam *database* toko. Hasil penelitian menunjukkan proses

No	Nama Peneliti	Judul Penelitian	Metode	Hasil Penelitian
				pencarian lebih cepat dan efisien dibandingkan metode pencarian konvensional, karena jumlah perbandingan karakter yang dilakukan lebih sedikit. Hal ini berdampak pada peningkatan kinerja sistem informasi dalam melayani transaksi penjualan.
4	Heny Rimadana, Ajib Hanani, Agung Teguh Wibowo Almais, Shoffin Nahwa Utama, Johan Ericka Wahyu Prakasa, M. Amin Hariyadi, Yunifa Miftachul Arif	QIRABOX: Kontrol Speaker Murrotal Al-Qur'an denga Perintah Suara Berbasis *Internet of Things*	Algoritma Aho-Corasick	Penelitian ini berhasil mengembangkan sistem QIRABOX, yaitu *speaker murottal* Al-Qur'an berbasis IoT yang dikendalikan melalui perintah suara. Dengan memanfaatkan *Google Speech Recognition* dan algoritma Aho-Corasick, sistem mampu mengenali perintah dengan tingkat kesalahan hanya 2% dan memperoleh skor *System Usability Scale* (SUS) sebesar 78,75% dalam kategori "Baik".
5	Abdillah Syahputra, Halim Maulana	Design and Construction of Automatic Clothes Drying Rack Prototype Based on IoT (Internet of Things)	-	Jemuran otomatis merespons kondisi cuaca berdasarkan sensor. Penelitian ini menghasilkan prototipe jemuran otomatis yang mampu merespons kondisi cuaca secara *real time*. Saat hujan terdeteksi oleh sensor, jemuran secara otomatis ditarik masuk, sedangkan saat cerah jemuran akan dikeluarkan. Sistem juga dilengkapi sensor suhu dan kelembaban untuk membantu mengatur kondisi pengeringan pakaian.

No	Nama Peneliti	Judul Penelitian	Metode	Hasil Penelitian
				Hasil pengujian menunjukkan sistem dapat bekerja stabil dengan respon cepat sesuai perubahan cuaca.

Berdasarkan uraian penelitian-penelitian terkait di atas, dapat disimpulkan bahwa algoritma *Raita* telah terbukti efisien dalam mempercepat proses pencocokan *string* pada berbagai konteks sistem informasi, sementara *Google Speech Recognition* menunjukkan keandalannya dalam mengonversi perintah suara menjadi teks pada sistem berbasis IoT. Di sisi lain, sistem jemuran otomatis berbasis sensor juga telah berhasil diimplementasikan untuk merespons perubahan kondisi cuaca secara *real-time*. Penelitian (Rimadana et al., 2025) bahkan membuktikan bahwa integrasi antara *Google Speech Recognition* dengan algoritma *string matching* dapat meningkatkan akurasi pengenalan perintah dalam sistem berbasis IoT. Namun demikian, hingga saat ini belum ada penelitian yang secara khusus mengintegrasikan algoritma *Raita*, *Google Speech Recognition*, dan sistem jemuran otomatis berbasis IoT. Oleh karena itu, penelitian ini mengusulkan integrasi tersebut guna menghadirkan sistem yang tidak hanya adaptif terhadap kondisi lingkungan melalui sensor, tetapi juga fleksibel melalui kendali suara sebagai alternatif ketika sensor tidak berfungsi secara optimal.

2.2 *Internet of Things*

Menurut (Selay et al., 2022), *Internet of Things* (IoT) adalah teknologi yang memperluas manfaat konektivitas internet dengan menghubungkan berbagai benda di sekitar agar aktivitas sehari-hari menjadi lebih mudah dan efisien. IoT

memungkinkan perangkat mengumpulkan dan mengirim data melalui jaringan internet, lalu berinteraksi tanpa campur tangan manusia secara langsung. Pada awalnya IoT erat kaitannya dengan *Radio Frequency Identification* (RFID), namun kini juga mencakup teknologi sensor, nirkabel, maupun kode QR (*Quick Response*). Secara etimologi, IoT sendiri terdiri dari dua kata, yaitu "Internet" yang berarti mengatur konektivitas dan "Things" yang berarti objek atau perangkat yang dapat saling bertukar data secara otomatis.

Berdasarkan jejak historis nya, IoT bermula pada awal 1980-an ketika *programmer* di Carnegie Mellon University menghubungkan mesin penjual Coca Cola ke internet untuk memantau suhu dan stok minuman. Pada tahun 1990, John Romkey menciptakan pemanggang roti yang dapat dikendalikan melalui internet, disusul penggunaan *webcam* di University of Cambridge pada tahun 1991 untuk memantau persediaan kopi. Perkembangan berikutnya ditandai dengan munculnya WearCam oleh Steve Mann pada 1994 dan penjelasan Paul Saffo pada 1997 mengenai potensi sensor di masa depan. Istilah IoT sendiri diperkenalkan oleh Kevin Ashton pada tahun 1999 melalui penelitian tentang *Radio Frequency Identification* (RFID) di Auto-ID Centre MIT. Memasuki abad ke-21, IoT semakin dikenal luas, ditandai dengan konferensi internasional pertama di Swiss pada 2008 serta hadirnya produk komersial seperti kulkas pintar LG pada tahun 2000 dan robot Nabaztag pada tahun 2005 yang mampu memberikan informasi secara interaktif (Selay et al., 2022).

Menurut (Manuhutu et al., 2025), IoT merupakan perangkat yang terhubung melalui internet untuk mengumpulkan, menganalisis, dan bertukar data secara *real-*

time, sehingga memberikan efisiensi dan kenyamanan dalam kehidupan manusia. IoT kini banyak diterapkan pada bidang *smart home*, kesehatan, transportasi, hingga industri. Dalam penelitian ini, IoT digunakan sebagai dasar perancangan jemuran pintar, di mana sensor hujan dan cahaya terhubung dengan mikrokontroler serta dipadukan dengan kendali suara untuk menciptakan sistem *smart home* yang lebih adaptif dan efisien.

2.3 *Speech Recognition*

Dalam penelitiannya, (Mohamad Salman Farizi et al., 2022) mengemukakan bahwa *speech recognition* atau pengenalan suara adalah proses otomatis untuk mengenali kata-kata yang diucapkan manusia berdasarkan informasi dari sinyal suara. Teknologi ini memungkinkan sistem komputer untuk mengonversi sinyal suara menjadi teks sehingga dapat diproses lebih lanjut. Implementasinya cukup luas, mulai dari aplikasi *smart home* hingga kendali perangkat elektronik rumah tangga berbasis IoT. Dengan adanya *speech recognition*, interaksi manusia dan mesin dapat lebih natural karena pengguna cukup memberikan perintah suara tanpa harus melakukan *input* manual.

Berdasarkan penelitian yang dilakukan oleh (Wisnu, 2024), dijelaskan bahwa sistem *speech recognition* terbagi atas dua bagian, yaitu *front-end* dan *back-end*. Bagian *front-end* berfungsi mengubah sinyal suara menjadi sinyal digital dengan bantuan *analog-digital converter*, sedangkan *back-end* bertugas memproses sinyal tersebut untuk dikenali sebagai kata atau kalimat tertentu. Berdasarkan fungsinya, bagian *back-end* memiliki komponen-komponen utama sebagai berikut:

1. *Acoustic Model*

   Proses ini menghubungkan suara berupa kata-kata yang diucapkan dengan satuan bunyi bahasa yang serupa (fonem).

2. *Language Model*

   Mengubah suku kata yang diidentifikasi oleh *acoustic model* menjadi kata-kata yang sesuai dengan apa yang diucapkan oleh pengguna. Suku kata yang dihasilkan menjadi teks dapat dicari polanya dan dicocokkan dengan menggunakan kamus atau *lexicon*.

3. *Lexicon*

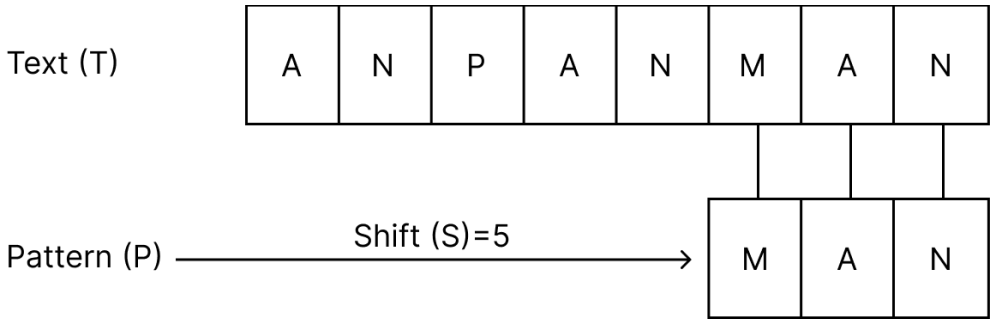
   Lexicon berperan sebagai kamus (*dictionary*) yang menyediakan daftar kata serta cara pengucapannya.

Kombinasi dari ketiga komponen ini menggunakan pendekatan *string matching* yang memungkinkan kemunculan teks dapat sesuai dengan ucapan dari pengguna.

Dalam penelitian ini, sistem tidak membangun mekanisme *speech recognition* secara mandiri, melainkan memanfaatkan layanan *Google Speech Recognition* pada perangkat Android untuk mengonversi suara pengguna menjadi teks secara *real-time*. Teks hasil konversi tersebut kemudian diproses lebih lanjut menggunakan algoritma *string matching*, yaitu *Raita*, untuk menentukan kesesuaian dengan pola perintah yang telah ditentukan. Dengan demikian, peran *string matching* dalam penelitian ini berada pada tahap pemetaan teks hasil pengenalan suara, bukan pada proses ekstraksi fitur suara itu sendiri.

2.4 *String Matching*

Menurut (Mesi & Oktarina, 2021), *string matching* atau pencocokan *string* adalah pendekatan untuk menemukan kesesuaian antara dua *string* yang berbeda, yaitu *pattern* dan *text*. *Pattern* adalah rangkaian karakter dengan panjang tertentu yang ingin dicari, sedangkan *text* merupakan rangkaian karakter yang lebih panjang sebagai tempat pencarian. Dalam prosesnya, algoritma *string matching* bekerja dengan cara membandingkan karakter demi karakter dari *pattern* (*P*) terhadap *text* (*T*) hingga ditemukan kecocokan. Sejalan dengan itu, (Rasool et al., 2012) menjelaskan bahwa permasalahan *string matching* adalah menemukan semua kemunculan *pattern* dalam sebuah *text* dengan cara menggeser (*shift*) *pattern* sepanjang *text* untuk mengevaluasi kecocokan. Dalam Gambar 2.1 merupakan contoh dari hasil proses *string matching*.



Gambar 2.1 Contoh Proses String Matching

Sementara itu, (Sari & Ginting, 2021) menjelaskan bahwa teknik *string matching* terbagi menjadi dua jenis, yaitu *exact string matching* dan *inexact string matching*. *Exact string matching* merupakan pencocokan *string* secara tepat, di mana susunan karakter harus identik antara *pattern* dan *text*. Algoritma yang termasuk dalam kategori ini adalah Brute Force, Knuth-Morris-Pratt (KMP),

Boyer-Moore, dan sebagainya. Sedangkan *inexact string matching* merupakan pencocokan *string* secara samar, di mana *string* yang dibandingkan tidak harus identik, tetapi memiliki kemiripan tekstual maupun fonetik. Teknik ini bermanfaat ketika pencarian tidak hanya membutuhkan hasil cocok atau tidak cocok, melainkan juga mencari pola dengan tingkat kesamaan tertentu.

Berdasarkan kedua kajian tersebut, dapat dipahami bahwa *string matching* memiliki peran penting dalam mendukung proses pencarian data maupun pola teks. Dalam penelitian ini, konsep *exact string matching* digunakan sebagai dasar untuk penerapan algoritma *Raita*, yang mampu memberikan kecepatan dan efisiensi lebih tinggi dalam mengenali pola kata hasil dari *speech recognition*.

2.5 Algoritma *Raita*

Menurut (Harahap & Ginting, 2020), *Raita* merupakan salah satu algoritma dalam kategori *exact string matching* yang berfungsi untuk mencocokkan pola berdasarkan urutan teks secara tepat. Algoritma ini pertama kali diusulkan oleh Timo Raita pada tahun 1992 sebagai bentuk penyempurnaan dari algoritma sebelumnya yaitu Boyer-Moore-Horspool (BMH) (Timanta Tarigan et al., 2019). Selanjutnya, (Hudaa et al., 2020) menyatakan bahwa *Raita* dikenal bekerja cepat dalam melakukan pencarian *string*, baik pada teks pendek maupun panjang, dengan strategi pencocokan yang dimulai dari karakter terakhir pola, kemudian karakter tengah, dan dilanjutkan pada karakter kedua hingga sebelum terakhir. Urutan pemeriksaan ini membuat algoritma *Raita* lebih efisien dalam menemukan kecocokan dibandingkan beberapa algoritma *exact string matching* lain.

Lebih lanjut, (Hudaa et al., 2020) juga menyatakan bahwa algoritma *Raita* secara umum terdiri dari dua tahap utama, yaitu *preprocessing* dan *searching*. Pada tahap *preprocessing*, dibentuk tabel pergeseran karakter buruk (*bad character shift table*) yang prinsipnya diadopsi dari algoritma Boyer-Moore. Nilai pergeseran tersebut dihitung menggunakan persamaan 2.1 di bawah ini:

$$BmBc[x[i]] = m - i - 1 \qquad (2.1)$$

Di mana:
$x[i]$ = karakter pola pada indeks ke-i
m = panjang karakter pola
i = indeks masing-masing karakter

Persamaan tersebut digunakan untuk menentukan sejauh mana pola (*pattern*) dapat digeser jika terjadi ketidakcocokan pada saat proses pencarian. Dalam Gambar 2.2 merupakan contoh *bad character shift table* (BmBc *table*) yang didapatkan dari persamaan 2.1.

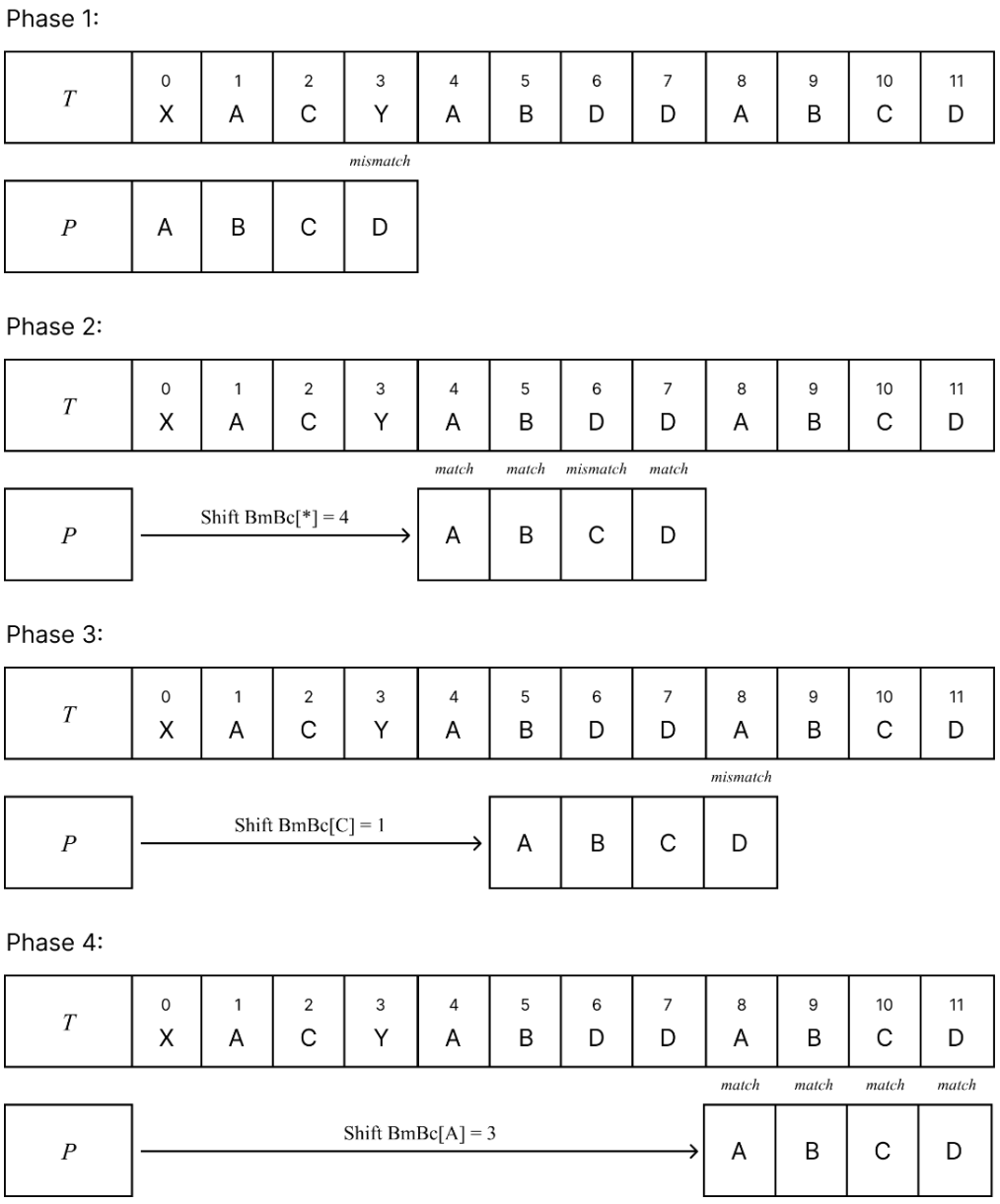
i	0	1	2	*
$x[i]$	A	B	C	D
$bmBc[x[i]]$	3	2	1	4

*D and other characters = 4

Gambar 2.2 Contoh *Bad Character Shift Table*

Selanjutnya, pada tahap *searching*, pencocokan dilakukan dengan strategi khas *Raita*, yaitu dimulai dari karakter terakhir *pattern*, kemudian karakter pertama, dilanjutkan ke karakter tengah, dan baru ke karakter lain yang tersisa. Urutan pemeriksaan ini memungkinkan deteksi ketidakcocokan lebih cepat, sehingga

proses pencarian lebih efisien. Dalam Gambar 2.3 merupakan contoh tahapan *searching* pada algoritma *Raita*.



Gambar 2.3 Contoh Proses *Searching* Algoritma *Raita*

Untuk tahap *preprocessing*, *Raita* memiliki kompleksitas waktu $O = (m + \sigma)$, sedangkan untuk tahap *searching*, kompleksitas waktunya adalah $O =$

$(m * n)$ dengan m adalah panjang pola, n adalah panjang teks, dan σ adalah ukuran alfabet yang digunakan (Bawanto & Rosmawanti, 2017).

Dalam konteks penelitian ini, panjang pola perintah yang digunakan relatif pendek, seperti “masuk”, “keluar”, “jangan masuk”, dan “jangan keluar”, sehingga algoritma *Raita* dinilai sesuai untuk diterapkan pada sistem kendali berbasis IoT yang membutuhkan respons cepat. Dengan karakteristik tersebut, *Raita* mampu melakukan pencocokan pola secara efisien pada teks hasil konversi *speech recognition* tanpa menambah beban komputasi yang signifikan pada sistem.

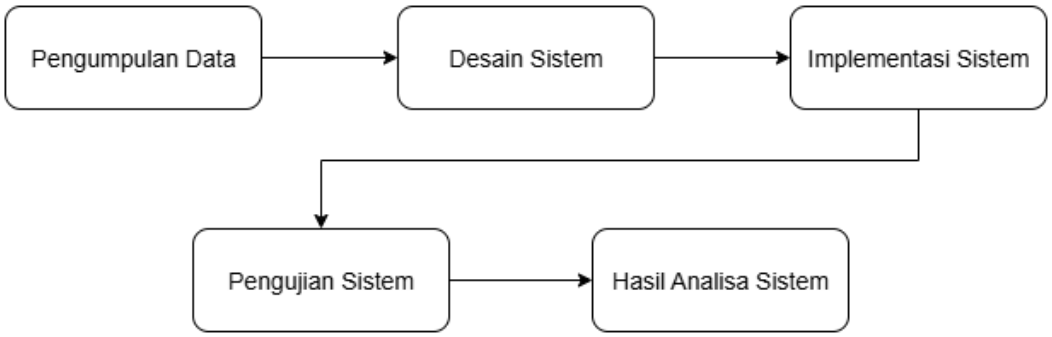
2.6 MIT App Inventor

MIT App Inventor adalah sebuah media berbasis aplikasi web untuk mengembangkan aplikasi Android atau *mobile* yang didesain khusus untuk pengguna tanpa pengalaman dalam pemrograman. Selain bersifat *open source*, aplikasi ini menerapkan konsep *drag and drop* berbentuk blok-blok perintah, sehingga memudahkan pengguna dalam mengembangkan aplikasi Android tanpa perlu lagi menulis instruksi dalam bahasa pemrograman (Nugroho, 2019). Sejalan dengan itu, menurut (Rimadana, 2024), aplikasi ini memiliki keunggulan lain yaitu dapat diintegrasikan oleh *hardware* dan API, seperti mikrofon, kamera, sensor, dan fitur-fitur Android lainnya. Dalam penelitian ini, MIT App Inventor digunakan sebagai alat untuk mengembangkan aplikasi sistem kendali dengan perintah suara yang diterjemahkan menjadi sebuah instruksi untuk mengeluarkan dan memasukkan jemuran pada jemuran pintar.

BAB III

DESAIN DAN IMPLEMENTASI

Pada bab ini akan menjelaskan tahapan-tahapan yang dilakukan dalam penelitian mulai dari pengumpulan data, desain sistem, implementasi sistem, pengujian sistem, hingga analisis hasil. Alur penelitian ini disusun untuk memberikan gambaran sistematis mengenai langkah-langkah yang ditempuh agar penelitian dapat berjalan sesuai dengan tujuan yang telah ditetapkan. Setiap tahapan saling berkesinambungan dan menjadi dasar dalam pengembangan sistem jemuran pintar berbasis IoT dengan perintah suara menggunakan algoritma *Raita*. Adapun alur penelitian yang digunakan dalam penelitian ini dapat dilihat pada Gambar 3.1.



Gambar 3.1 Desain Penelitian

3.1 Pengumpulan Data

Dalam penelitian ini terdapat dua jenis data yang digunakan, yaitu data primer dan data sekunder. Data primer digunakan untuk menghasilkan perintah kepada sistem. Sedangkan data sekunder, digunakan untuk kata kunci atau pola (*pattern*) dalam sistem ini. Lebih lanjut, di bawah ini akan dijelaskan mengenai kedua jenis data yang digunakan dalam penelitian ini.

3.1.1 Data Primer

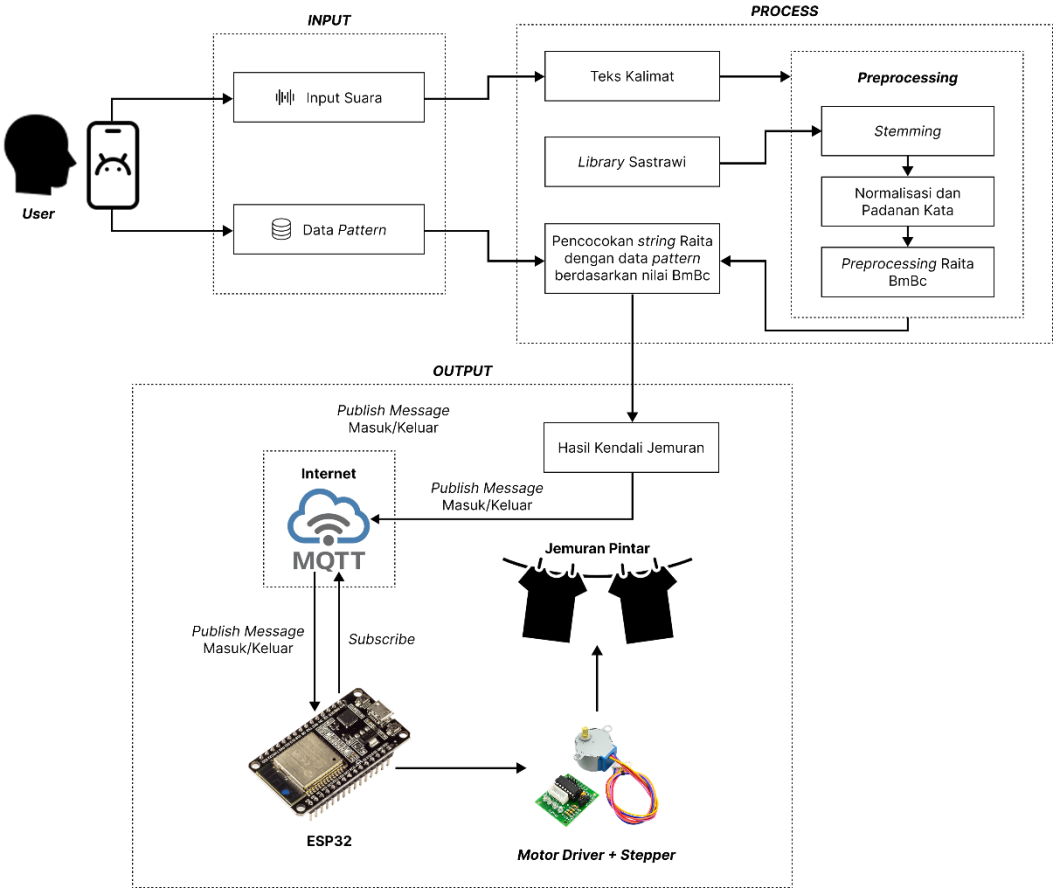
Data primer diperoleh melalui *input* suara pengguna menggunakan *Google Speech Recognition* dalam bahasa Indonesia yang nantinya dikonversi ke teks, kemudian diproses menggunakan algoritma *Raita*.

3.1.2 Data Sekunder

Data sekunder diperoleh melalui *pattern* yang disiapkan sebelumnya oleh peneliti. Data sekunder digunakan sebagai sampel kata kunci untuk pengujian dalam sistem ini, yaitu berupa kata “masuk”, “keluar”, “jangan masuk”, dan “jangan keluar”.

3.2 Desain Sistem

Dalam penelitian ini terdapat tiga tahapan utama untuk sistem kendali jemuran pintar dengan perintah suara. Tahapan-tahapan tersebut antara lain *input*, *process*, dan *output*. Selain itu, disajikan desain sistem dalam penelitian ini yang dapat dilihat pada Gambar 3.2.



Gambar 3.2 Desain Sistem Kendali Jemuran Pintar

3.2.1 Tahapan *Input*

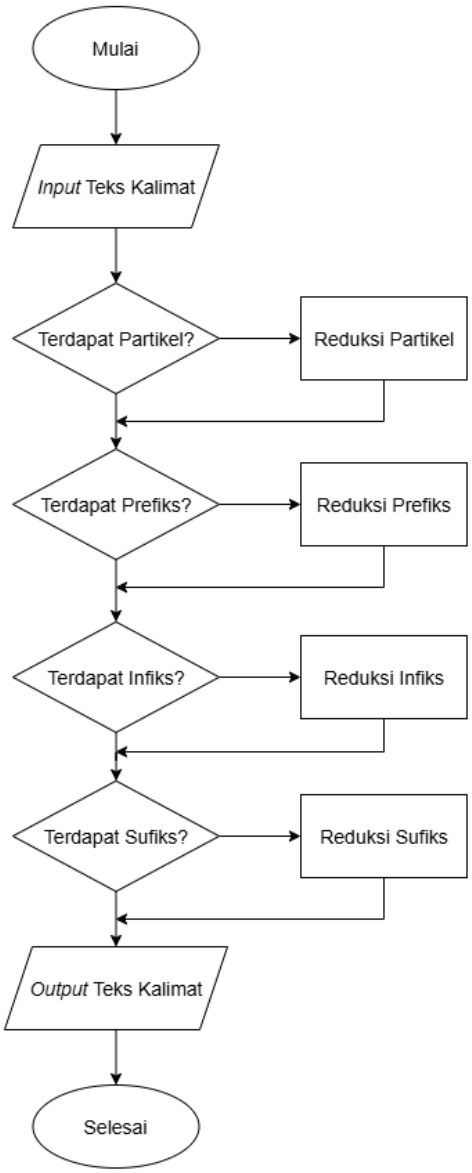
Tahapan *input* merupakan tahap awal di mana pengguna memberi perintah pada sistem melalui perintah suara pada aplikasi Android sistem kendali. Jika menggunakan perintah suara, sistem akan mengaktifkan *mic smartphone* pengguna dan memanfaatkan fitur *Google Speech Recognition* yang disediakan oleh MIT App Inventor untuk mengubah perintah suara menjadi teks yang akan diproses oleh algoritma *Raita*.

3.2.2 Tahapan *Process*

Perancangan sistem kendali ini melalui beberapa tahapan proses yang mencakup rangkaian *text preprocessing* untuk menyiapkan hasil teks dari pengenalan suara sebelum dilakukan pencocokan pola menggunakan algoritma *Raita*. Pada penelitian ini, *text preprocessing* terdiri dari tiga tahap utama, yaitu *stemming* untuk mengubah kata menjadi bentuk dasarnya, normalisasi dan padanan kata untuk membersihkan hasil teks serta menyeragamkan variasi frasa yang bermakna sama, dan *preprocessing* algoritma *Raita* untuk menentukan nilai BmBc sebagai dasar proses pencocokan. Setelah ketiga tahap tersebut selesai, sistem akan melakukan pencocokan teks dengan algoritma *Raita* terhadap pola perintah yang telah disimpan di dalam *database*. Berikut ini merupakan penjelasan dari masing-masing tahapan proses yang digunakan dalam perancangan sistem kendali pada penelitian ini.

3.2.2.1 *Stemming*

Stemming adalah proses mereduksi imbuhan seperti prefiks, infiks, dan sufiks yang ada pada tiap kata sehingga berubah ke bentuk dasarnya. Hal ini dilakukan untuk menyeragamkan varian kata yang berbeda agar dapat dianggap sama (Ardiani et al., 2020). Penelitian ini melakukan *stemming* berdasarkan Bahasa Indonesia, yang pemrogramannya diproses menggunakan *library* Sastrawi dengan bahasa pemrograman PHP. Alur proses *stemming* dalam penelitian ini dapat dilihat pada Gambar 3.3.



Gambar 3.3 Alur Proses *Stemming*

Proses *stemming* pada penelitian ini diawali dengan pemeriksaan partikel seperti “-lah”, “-kah”, atau “-pun”, yang akan dihapus karena tidak mengubah makna dasar kata. Selanjutnya sistem memeriksa adanya prefiks (awalan) seperti “ber-“, “ter-“, atau “se-“ untuk kemudian direduksi jika ditemukan. Setelah itu dilakukan deteksi terhadap infiks (sisipan) seperti “-el-“ atau “-er-“ yang juga akan

dihapus. Tahap terakhir adalah reduksi sufiks (akhiran) seperti “-an”, “-i”, atau “-kan” agar kata dikembalikan ke bentuk dasarnya. Dengan tahapan ini, semua variasi morfologis kata dapat dipetakan menjadi kata dasar sehingga memudahkan proses pencocokan *string* dalam sistem. Dalam Tabel 3.1 dipaparkan ilustrasi dari hasil proses *stemming*.

Tabel 3.1 Ilustrasi *Stemming*

Sebelum *Stemming*	**Sesudah *Stemming***
keluarkan jemurannya sekarang	keluar jemur sekarang
tolong jemurannya dimasukkan dong	tolong jemur masuk dong
jangan masukkan jemuran	jangan masuk jemur
jangan masukin jemurannya sekarang	jangan masuk jemur sekarang
jemurannya jangan dimasukkan dulu	jemur jangan masuk dulu

Berdasarkan hasil pengujian awal, sistem pengenalan suara bawaan *Google Speech Recognition* sudah memiliki tingkat akurasi yang cukup tinggi dalam menangkap pelafalan pengguna, termasuk pada variasi pengucapan ringan seperti “masok” atau “masuh”. Oleh karena itu, sistem tidak menambahkan modul koreksi ejaan berbasis fonetik, melainkan memanfaatkan tahap normalisasi berikutnya untuk menjaga konsistensi hasil teks agar tetap sesuai dengan format yang dapat dikenali oleh sistem sebelum dilakukan proses lebih lanjut.

3.2.2.2 Normalisasi dan Padanan Kata

Tahap normalisasi dan padanan kata merupakan langkah lanjutan setelah proses *stemming*, yang berfungsi untuk memastikan hasil teks dari proses pengenalan suara memiliki bentuk yang bersih, konsisten, dan sesuai dengan konteks perintah pengguna. Tahapan ini dibagi menjadi dua proses utama, yaitu normalisasi ejaan dan pemetaan padanan kata (*synonym mapping*).

Pada proses normalisasi ejaan, sistem melakukan pembersihan teks dari karakter non-huruf, tanda baca, atau spasi berlebih yang dapat mengganggu proses analisis kata. Selain itu, dilakukan pula penyesuaian terhadap bentuk kata tidak baku atau bentuk *slang* yang umum digunakan dalam percakapan sehari-hari, seperti mengganti "masukin" menjadi "masuk" atau "keluarin" menjadi "keluar". Tahap ini bertujuan untuk menjaga konsistensi hasil teks dari *Google Speech Recognition* agar siap digunakan dalam tahap pencocokan pola berikutnya.

Sementara itu, proses pemetaan padanan kata berfungsi untuk menyeragamkan frasa-frasa yang memiliki makna sama dengan empat jenis perintah utama, yaitu "masuk", "keluar", "jangan masuk", dan "jangan keluar". Pemetaan ini dilakukan dengan membuat kamus padanan kata eksternal berbentuk berkas JSON yang memuat berbagai variasi frasa yang umum digunakan pengguna. Misalnya, frasa "amankan jemurannya", "bawa jemurannya masuk", atau "tarik jemurannya ke dalam" semuanya akan dikenali sebagai perintah "masuk", sedangkan frasa "taruh jemurannya di luar", "gantung jemurannya di luar", atau "jemur di luar" akan dikenali sebagai perintah "keluar".

Dengan penerapan tahap normalisasi dan padanan kata ini, sistem menjadi lebih adaptif terhadap cara pengguna memberikan perintah secara alami menggunakan bahasa sehari-hari, tanpa perlu mengubah logika utama algoritma *Raita* yang digunakan sebagai metode pencocokan pola berbasis *exact string matching*. *Output* dari tahap ini berupa teks yang telah diseragamkan dan siap diproses lebih lanjut pada tahap *preprocessing* algoritma *Raita*.

3.2.2.3 *Preprocessing* Algoritma *Raita*

Setelah melalui kedua tahap di atas, selanjutnya adalah melakukan *preprocessing* berdasarkan algoritma *Raita* untuk menentukan nilai pergeseran (BmBc) tiap karakter *pattern*. Langkah-langkah yang perlu dilakukan adalah sebagai berikut:

1. Menentukan kata kunci *pattern*. Contohnya menggunakan kata kunci *pattern* "masuk".
2. Menghitung panjang kata kunci *pattern* "masuk".

   Contoh:

   Tabel 3.2 Panjang Kata Kunci *Pattern* "masuk"

M	A	S	U	K
1	2	3	4	5

   Jadi, panjang kata kunci *pattern* "masuk" adalah 5.
3. Menghitung indeks dari tiap karakter pada kata kunci *pattern* "masuk".

   Tabel 3.3 Indeks Kata Kunci *Pattern* "masuk"

M	A	S	U	K
0	1	2	3	4

   Jadi, indeks kata kunci *pattern* "masuk" adalah 4.
4. Hitung nilai BmBc dari masing-masing karakter kata kunci *pattern*, dengan pengecualian untuk karakter indeks terakhir dan karakter yang tidak ada pada *pattern* diberikan nilai sesuai dengan panjang karakter kata kunci *pattern*. Untuk menentukan nilai BmBc, digunakan persamaan 3.1 di bawah ini:

$$BmBc[x[i]] = m - i - 1 \qquad (3.1)$$

Keterangan:
$x[i]$ = karakter *pattern* pada indeks ke-i
m = panjang karakter *pattern*

i = indeks masing-masing karakter

Misalkan *pattern* "masuk" dihitung menggunakan persamaan 3.1 di atas.

$BmBc[M] = 5 - 0 - 1 = 4$

$BmBc[A] = 5 - 1 - 1 = 3$

$BmBc[S] = 5 - 2 - 1 = 2$

$BmBc[U] = 5 - 3 - 1 = 1$

$BmBc[K] = 5$ (karakter indeks terakhir)

5. Setelah nilai BmBc tiap karakter *pattern* didapatkan, langkah terakhir adalah membuat *bad character shift table* yang berisi karakter *pattern* dan nilai pergeserannya pada tiap *window*. Untuk ilustrasi *bad character shift table* dapat dilihat pada Tabel 3.4.

Tabel 3.4 *Bad Character Shift Table*

i	0	1	2	3	*
$[x[i]]$	M	A	S	U	K
$BmBc[x[i]]$	4	3	2	1	5

Keterangan:
(*) = karakter yang tidak ada pada *pattern*
Nilai untuk karakter K (indeks terakhir) sesuai dengan panjang *pattern*

3.2.2.4 Pencocokan Algoritma *Raita*

Tahap selanjutnya adalah implementasi algoritma *Raita* untuk mencocokkan *pattern* dengan karakter pada teks atau kalimat yang berasal dari perintah suara pengguna. Proses pencocokan algoritma *Raita* dijelaskan melalui beberapa langkah berikut:

1. Pemeriksaan dimulai dengan membandingkan karakter terakhir pada *pattern* dengan karakter teks pada posisi yang sejajar.

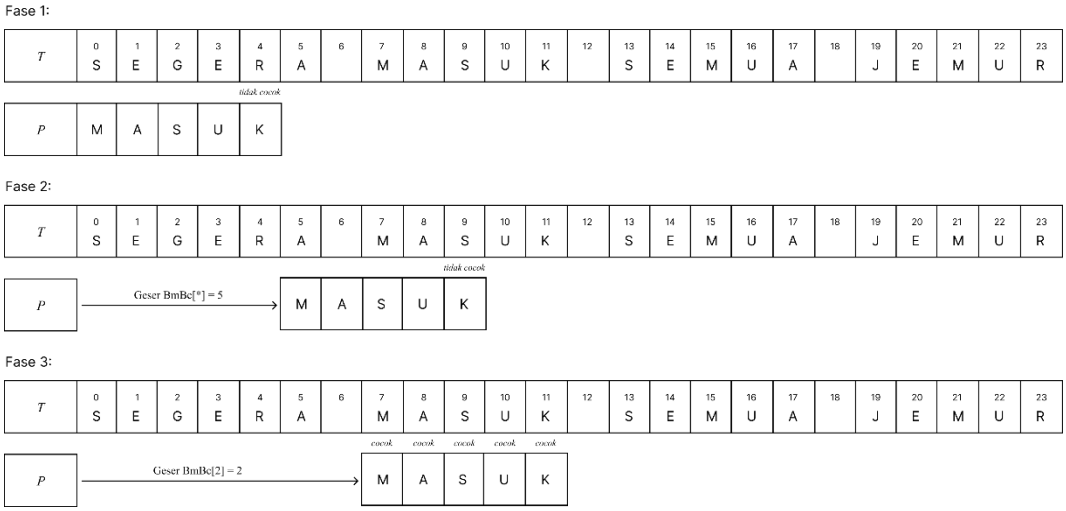
2. Apabila terjadi ketidakcocokan, *pattern* langsung digeser dengan melihat nilai pada *bad character shift table* (BmBc) berdasarkan karakter teks yang dibandingkan, kemudian perbandingan diulangi.
3. Jika karakter terakhir cocok, perbandingan dilanjutkan pada karakter pertama *pattern* dengan karakter teks pada posisi sejajar.
4. Setelah itu, dilakukan perbandingan pada karakter tengah pola (indeks $[m/2]$).
5. Jika salah satu dari tiga tahap awal (karakter terakhir, pertama, atau tengah) tidak sesuai, *pattern* kembali digeser sesuai nilai pada *bad character shift table*.
6. Apabila ketiga karakter awal (terakhir, pertama, dan tengah) cocok, dilanjutkan dengan pemeriksaan menyeluruh terhadap seluruh karakter *pattern*.
7. Jika ditemukan kecocokan penuh, *pattern* dianggap sesuai pada posisi tersebut.
8. Jika pada pemeriksaan penuh terjadi ketidakcocokan, *pattern* digeser kembali berdasarkan aturan pergeseran *bad character shift table*.

Berikut ilustrasi proses pencocokan algoritma *Raita* berdasarkan *preprocessing* sebelumnya dengan contoh kasus di bawah ini:

Text : Segera masukkan semua jemuran

: segera masuk semua jemur (setelah *stemming*)

Pattern : masuk



Gambar 3.4 Ilustrasi Pencocokan Algoritma *Raita*

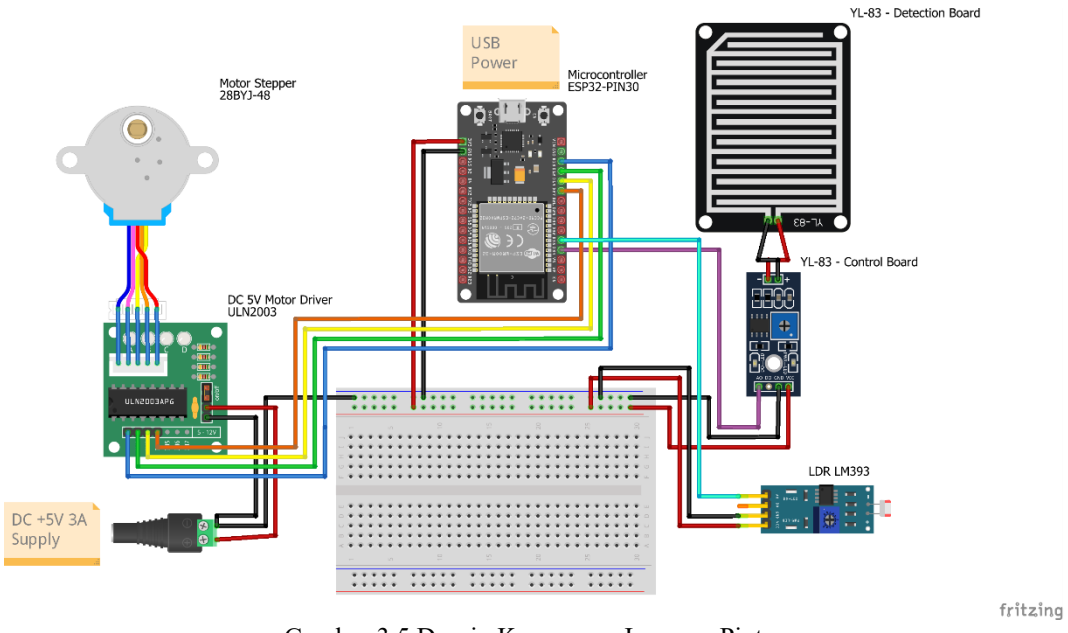
Proses pencocokan pada Gambar 3.4 merepresentasikan bahwa *pattern* “masuk” ditemukan dalam *text* “Segera masukkan semua jemuran” pada iterasi atau fase ke-3 pencocokan.

3.2.3 Tahapan *Output*

Tahapan output pada rancangan sistem ini merupakan hasil akhir dari keseluruhan proses pengolahan perintah suara. Instruksi yang telah melalui tahap konversi suara ke teks, *preprocessing*, dan pencocokan dengan algoritma *Raita* diteruskan sebagai sinyal kendali menuju aktuator. Perintah yang telah dikenali akan dikirim melalui protokol komunikasi MQTT menuju mikrokontroler ESP32 yang berfungsi sebagai pengendali utama. Selanjutnya, ESP32 akan mengaktifkan *motor stepper* melalui *driver motor* sesuai instruksi, baik untuk menarik jemuran ketika perintah “masuk” dikenali, maupun mengeluarkan jemuran saat perintah “keluar” terdeteksi.

3.2.4 Desain Komponen

Desain komponen pada penelitian ini dirancang untuk mengintegrasikan berbagai perangkat keras yang berfungsi mendukung sistem jemuran pintar berbasis IoT. Mikrokontroler ESP32 berperan sebagai pusat kendali yang menerima *input* dari dua sensor, yaitu YL-83 sebagai sensor hujan dan LDR LM393 sebagai sensor intensitas cahaya. Kedua sensor ini memberikan sinyal kondisi lingkungan yang kemudian diproses oleh ESP32 untuk menentukan instruksi kendali. Hasil pemrosesan selanjutnya dikirimkan ke *motor stepper* 28BYJ-48 melalui *driver motor* ULN2003, yang berfungsi sebagai penguat arus agar *motor* dapat bekerja secara stabil dalam menggerakkan jemuran. Sumber daya utama berasal dari DC 5V 3A untuk *driver motor* dan *motor stepper*, USB power untuk ESP32, serta distribusi tegangan diatur melalui *breadboard* agar seluruh komponen terhubung dengan baik. Desain komponen dalam yang dijelaskan di atas dapat dilihat pada Gambar 3.5 di bawah ini.



Gambar 3.5 Desain Komponen Jemuran Pintar

3.3 Implementasi Sistem

Sistem jemuran pintar berbasis IoT pada penelitian ini dibangun melalui integrasi antara *software* dan *hardware*. Aplikasi Android yang dirancang dengan MIT App Inventor berfungsi sebagai antarmuka pengguna untuk memberikan instruksi berbasis suara. Instruksi tersebut diproses menggunakan layanan *Google Speech Recognition* agar dapat diubah menjadi teks. Selanjutnya, teks hasil konversi diperiksa menggunakan algoritma *Raita*, yang dipilih karena memiliki kinerja pencocokan pola teks yang cepat dan efisien sehingga mampu mengenali kata kunci sederhana seperti “masuk” dan “keluar” secara akurat. Setelah perintah dikenali, instruksi diteruskan melalui *broker* MQTT sehingga dapat dijalankan dalam lingkungan *Internet of Things* (IoT).

Pada sisi perangkat keras, sistem memanfaatkan mikrokontroler ESP32 sebagai pusat kendali yang terhubung dengan *driver motor* ULN2003 dan *motor stepper* 28BYJ-48 untuk menggerakkan jemuran. Selain itu, sensor YL-83 digunakan untuk mendeteksi kondisi hujan, sedangkan LDR LM393 dipakai untuk memantau intensitas cahaya. Perintah dari aplikasi yang masuk ke ESP32 akan diproses sesuai dengan kondisi sistem, kemudian *motor stepper* digerakkan untuk menarik jemuran ke dalam atau mendorong jemuran ke luar. Dengan cara ini, pengguna dapat mengendalikan jemuran pintar yang bergerak secara otomatis melalui perintah suara.

3.4 Pengujian Sistem

Dalam penelitian ini dilakukan tahap pengujian sistem yang mengevaluasi tingkat kesalahan (*error*) pemrosesan perintah suara pada sistem. Tujuan dari

pengujian ini adalah untuk mengetahui seberapa besar *error* sistem yang terjadi ketika sistem digunakan untuk menjalankan fungsi memasukkan dan mengeluarkan jemuran berdasarkan perintah suara yang diproses dengan algoritma *Raita*. Pengujian dilakukan sebanyak 480 kali percobaan pada sistem dengan pola perintah yang berbeda-beda. Skenario pengujian sistem dalam penelitian ini dipaparkan dalam Tabel 3.5.

Tabel 3.5 Skenario Pengujian Sistem

No	Pola Perintah (Setelah di-*Stemming*)	Jumlah Pengujian
1.	Masuk (memasukkan jemuran)	120 kali
2.	Keluar (mengeluarkan jemuran)	120 kali
3.	Jangan masuk (jangan memasukkan jemuran)	120 kali
4.	Jangan keluar (jangan mengeluarkan jemuran)	120 kali
Total Pengujian		**480 kali**

Pada Tabel 3.5, dipaparkan skenario pengujian yang di dalamnya terdapat 480 kali pengujian dengan 4 skenario pengenalan perintah berdasarkan pola yang sudah ditentukan. Hasil pengujian sistem akan dikalkulasi berdasarkan jumlah kesalahannya dalam merespon perintah pengguna. Hal tersebut dapat dihitung menggunakan persamaan persentase *error* untuk mengetahui seberapa besar perbedaan antara hasil pengujian yang sebenarnya dengan yang diharapkan (Munna, 2024). Berikut ini merupakan persamaan persentase *error* yang digunakan:

$$\mu_{error} = \frac{\sum Ei}{n} \times 100\% \qquad (3.2)$$

Di mana:
μ_{error} = rata-rata *error* dalam bentuk persen (%)
n = jumlah pengujian
$\sum Ei$ = jumlah *error* dari semua pengujian

Tabel 3.6 Ilustrasi Alur Pengujian *Error* Sistem

Pengujian	***Input* Teks**	***Pattern***	***Success***	***Error***
1.	Tolong jemurannya dimasukkan dong	masuk	1	0
2.	Jemuran segera masuk	masuk	1	0
3.	Ayo jemurannya masukkan sekarang	masuk	1	0
4.	Masukkan jemuran cepat	masuk	1	0
5.	Jemurannya jangan dimasukkan dulu	jangan masuk	1	0
6.	Jangan masukkan jemuran ya	jangan masuk	1	0
7.	Tolong jemurannya jangan dimasukkan	jangan masuk	0	1
8.	Jemuran segera dikeluarkan	keluar	1	0
9.	Ayo jemurannya dikeluarkan sekarang	keluar	1	0
10.	Tolong keluarkan jemurannya	keluar	1	0
11.	Keluarin jemuran ke halaman	keluar	0	1
12.	Jangan keluarkan jemuran ya	jangan keluar	1	0
13.	Tolong jemurannya jangan dikeluarkan	jangan keluar	0	1
14.	Jemurannya jangan dikeluarkan	jangan keluar	1	0
15.	Jangan keluarin jemurannya sekarang	jangan keluar	0	1
$\sum Ei$ (Jumlah *error* dari semua pengujian)				**4**

Berdasarkan Tabel 3.6, disajikan hasil dari simulasi pengujian *error* sistem yang diperoleh sebanyak 4 *error* dan 11 *success* dari total 15 kali pengujian. Adapun untuk mendapatkan persentase *error* dari pengujian tersebut digunakan persamaan 3.2 yang menghasilkan persentase sebesar 26.6% seperti yang dapat dilihat pada persamaan 3.3 di bawah ini:

$$\mu_{error} = \frac{\sum Ei}{n} \times 100\%$$

$$\mu_{error} = \frac{4}{15} \times 100\% \qquad (3.3)$$

$$\mu_{error} = 26.6\%$$

3.5 Hasil Analisa Sistem

Setelah dilakukan serangkaian pengujian terhadap sistem yang telah dibangun, peneliti menganalisis hasil sistem untuk memahami kinerjanya,

menemukan elemen-elemen yang mempengaruhi kesalahan sistem, dan menentukan bagian mana yang perlu diperbaiki pada penelitian berikutnya.

BAB IV

HASIL DAN PEMBAHASAN

4.1 Implementasi Sistem Kendali

Sistem kendali yang sudah dibuat diimplementasi dan diuji berdasarkan rancangan penelitian yang telah diuraikan pada bab sebelumnya. Sistem kendali dalam penelitian ini terdiri dari dua elemen utama, antara lain *software* yang merupakan aplikasi sistem kendali berbasis Android dengan fitur *speech recognition* sebagai media dalam proses mengirim dan menerima perintah suara pengguna, serta *hardware* yang terdiri atas *motor stepper* dan beberapa komponen lainnya. Berikut ini diuraikan secara rinci implementasi sistem kendali dalam penelitian ini mulai dari aplikasi atau *software*, konektivitas, serta rangkaian *hardware* yang digunakan.

4.1.1 Halaman *Software*

Pada awal aplikasi dibuka, akan tampil berupa *splash screen* logo dari aplikasi sistem kendali jemuran yang diberi nama Dijemurin. Halaman ini ditampilkan dengan tujuan supaya pengguna mengetahui dengan jelas aplikasi apa yang sedang dibuka. Ketika pengguna pertama kali membuka aplikasi, halaman ini akan ditampilkan selama beberapa detik yang kemudian pengguna akan diarahkan ke halaman utama.

Gambar 4.1 *Splash Screen* Aplikasi

Seperti yang dapat dilihat pada Gambar 4.1, merupakan tampilan *splash screen* aplikasi yang tampil dalam beberapa detik pada layar *smartphone* pengguna ketika aplikasi baru dibuka. Kemudian setelah *splash screen* ditampillkan, pengguna akan diarahkan ke halaman utama dari aplikasi Dijemurin.

Gambar 4.2 Halaman Utama Aplikasi

Dalam Gambar 4.2 ditampilkan halaman utama aplikasi yang terdiri dari beberapa komponen, bagian paling atas layar merupakan judul aplikasi, di bawahnya terdapat ikon jemuran yang menggantung dan label bertuliskan "Respons Sistem:" untuk memberi tahu pengguna *feedback* dari sistem setelah diberikan perintah, Selanjutnya, pada bagian tengah terdapat tombol mikrofon yang digunakan untuk mengaktifkan fitur *Google Speech Recognition* dan terdapat label "Teks Perintah:" untuk menampilkan perintah yang diberikan melalui perintah suara. Lebih lanjut, pada bagian bawah layar ditampilkan status sistem terbaru supaya pengguna mengetahui posisi jemuran terkini sedang berada di dalam atau di luar.

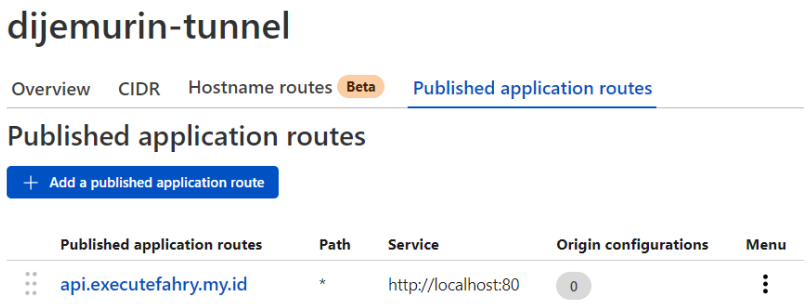
Gambar 4.3 Fitur *Google Speech Recognition* Aktif

Pada Gambar 4.3 ditunjukkan fitur *Google Speech Recognition* yang dapat aktif secara otomatis ketika pengguna pertama kali buka aplikasi dan ketika pengguna menekan tombol mikrofon untuk memberikan perintah kepada sistem jemuran pintar. Aplikasi ini dirancang sedemikian rupa dengan mempertimbangkan kemudahan fungsional supaya pengguna mendapatkan akses cepat dalam mengendalikan jemurannya. Selain itu, desain dari tiap halaman aplikasi dibuat sesederhana mungkin untuk memastikan interaksi pengguna dengan sistem berlangsung mudah dan lancar.

4.1.2 Implementasi Komunikasi IoT

Supaya *software* dan *hardware* dapat saling berinteraksi, digunakan komunikasi melalui internet dengan menerapkan platform Cloudflare Tunnel dan

PHP MQTT Client. Komunikasi IoT pada sistem ini dirancang agar aplikasi Android dan perangkat ESP32 dapat saling bertukar data secara otomatis dan *real-time*.



Gambar 4.4 *Dashboard* Cloudflare Tunnel

Seperti yang dapat dilihat pada Gambar 4.4, platform Cloudflare Tunnel digunakan untuk menyediakan akses *online* dari aplikasi menuju *backend* yang berjalan pada server lokal. Dengan penerapan Cloudflare Tunnel, *endpoint* API pada server lokal dapat diakses secara publik menggunakan jaringan internet dengan domain https://www.api.executefahry.my.id, sehingga tidak memerlukan konfigurasi jaringan yang rumit.

Gambar 4.5 Akses *Endpoint* API status.php dengan Internet

Dapat dilihat pada Gambar 4.5, setelah *endpoint* API dapat diakses melalui internet, proses komunikasi IoT dilanjutkan dengan menghubungkan server ke *broker* MQTT menggunakan *library* PHP MQTT Client. *Library* ini digunakan untuk melakukan *publish* perintah dari server menuju topik yang telah ditentukan. Pada sistem jemuran pintar ini digunakan topik “jemuran/cmd” untuk mengirimkan instruksi pergerakan jemuran seperti “masuk” atau “keluar”. Pesan yang dikirim melalui topik ini akan diterima oleh ESP32 yang berperan sebagai perangkat *subscriber* dan selanjutnya menjalankan *motor stepper* sesuai perintah yang diberikan.

Pembaruan status jemuran pada sistem dilakukan secara terintegrasi melalui komunikasi MQTT antara ESP32 dan server. Setiap kali jemuran bergerak, baik karena perintah suara (mode manual) maupun keputusan otomatis berdasarkan sensor hujan dan cahaya (mode auto), ESP32 akan mempublikasikan status terbaru ke topik “jemuran/status" dalam format JSON. Pesan tersebut diterima oleh *script* “status_listener.php” yang berjalan di *backend* sebagai *subscriber*, kemudian data status diperbarui pada tabel “system_status” di basis data. Selanjutnya, sistem mengakses *endpoint* “status.php” seperti pada Gambar 4.5, untuk mengambil data terbaru sehingga informasi posisi jemuran yang ditampilkan selalu sesuai dengan kondisi aktual perangkat jemurannya.

Implementasi komunikasi IoT ini memastikan bahwa seluruh komponen, mulai dari aplikasi, *backend*, hingga perangkat ESP32, dapat saling bertukar data dengan stabil. Kombinasi Cloudflare Tunnel sebagai jalur komunikasi internet dan PHP MQTT Client sebagai *messagging protocol* memungkinkan sistem kendali

jemuran dapat dioperasikan dari mana saja tanpa bergantung pada jaringan lokal. Dengan demikian, fitur kendali jarak jauh pada jemuran dapat berjalan secara fleksibel dan efisien.

4.1.3 Kendali Jemuran Menggunakan *Speech Recognition*

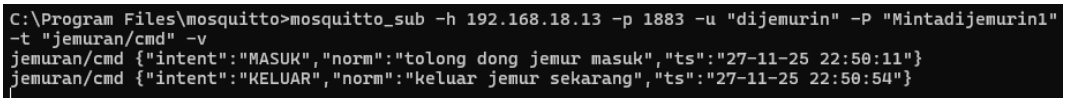
Kendali jemuran dengan perintah suara pada aplikasi Android dibuat menggunakan komponen pada MIT App Inventor seperti SpeechRecognizer, WebService, dan TextToSpeech. Fitur ini memungkinkan pengguna dapat memberikan instruksi dengan suara untuk memasukkan dan mengeluarkan jemuran. Perintah suara yang diterima oleh aplikasi akan dikonversi menjadi teks, kemudian dikirim ke server melalui *endpoint* "command.php" untuk diproses lebih lanjut menggunakan algoritma *Raita*. Hasil pemrosesan tersebut dikembalikan kepada aplikasi dan ditampilkan kepada pengguna sebagai *feedback*. Tahapan kendali jemuran dengan perintah suara pada aplikasi Android ditunjukkan melalui *pseudocode* berikut.

```
// Langkah 1: Mendengarkan perintah suara
Jika tombol mic ditekan
    Panggil SpeechRecognizer.GetText()
// Langkah 2: Mengambil teks hasil pengenalan suara
teks_perintah = SpeechRecognizer.Result
// Langkah 3: Mengirim teks perintah ke server API
URL =
"https://api.executefahry.my.id/dijemurin/app/api/command.php"
body = {"cmd_text": teks_perintah, "source": "app"}
Web.PostText(URL, body)
```

```
// Langkah 4: Menerima dan memproses respons dari server
respons_server = Web1.Response
intent_output = respons_server[“intent”]
decision_info = respons_server[“decision”]
// Langkah 5: Evaluasi respons server untuk menentukan tindakan
If intent_output == “MASUK”
    (Tampilkan “Jemuran sudah dimasukkan”)
    Panggil TextToSpeech(“Jemuran sudah dimasukkan”)
Else if intent_output == “KELUAR”
    (Tampilkan “Jemuran sudah dikeluarkan”)
    Panggil TextToSpeech(“Jemuran sudah dikeluarkan”)
Else if intent_output == “None” AND decision_info == “no_match”
    (Tampilkan “Perintah tidak dikenali”)
    Panggil TextToSpeech(“Perintah tidak dikenali”)
Else
    (Tampilkan “Perintah sebelumnya dibatalkan”)
    Panggil TextToSpeech(“Perintah sebelumnya dibatalkan”)
// Langkah 6: Update status jemuran setelah eksekusi perintah
URL_status =
“https://api.executefahry.my.id/dijemurin/app/api/status.php”
Web1.Get(URL_status)
// Langkah 7: Menampilkan status jemuran di aplikasi
respons_status = Web1.Response
state_jemuran  = respons_status[“status”]
last_cmd       = respons_status[“last_cmd”]
update_time    = respons_status[“updated_at_human”]
Jika state_jemuran == “MASUK”
    Tampilkan “Di dalam”
```

```
Else if state_jemuran == “KELUAR”
    Tampilkan “Di luar”
Else
    Tampilkan “Tidak diketahui”
Tampilkan last_cmd
Tampilkan update_time
```

Setelah server memproses perintah suara dan menentukan tujuan (*intent*) akhir, sistem akan *publish* pesan ke broker MQTT melalui topik “jemuran/cmd”. Pesan ini kemudian diterima oleh ESP32 yang melakukan *subscribe* pada topik tersebut. Tampilan *output* pesan MQTT pada Command Prompt ditunjukkan pada Gambar 4.6.

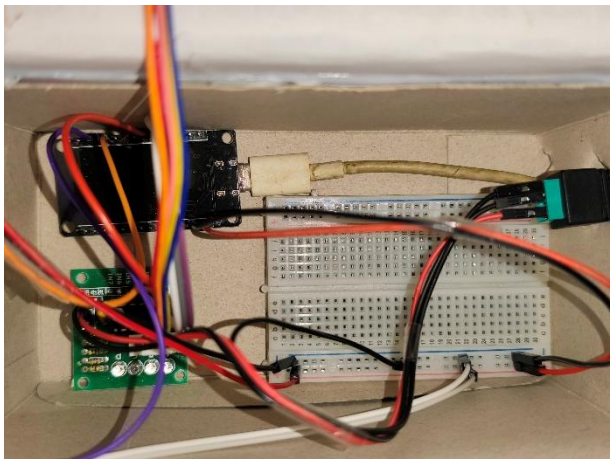


Gambar 4.6 *Output* Pesan *Subscribe* di Command Prompt

Pada Gambar 4.6 dapat dilihat bahwa setiap perintah suara yang dikirim menghasilkan *payload* berupa objek JSON yang berisi tujuan (*intent*), teks perintah yang telah dinormalisasi (*norm*), serta detail waktu pengiriman (ts). Pesan dengan intent “MASUK” dan “KELUAR” yang tampil pada *subscriber* membuktikan bahwa proses *speech recognition*, pemrosesan algoritma *Raita*, dan publikasi MQTT berjalan dengan benar serta diterima secara *real-time* oleh ESP32.

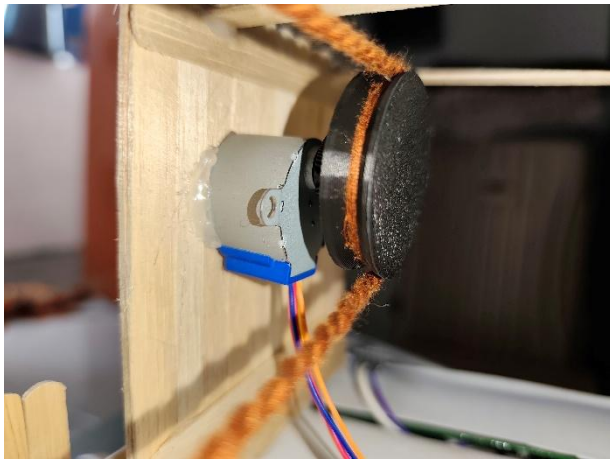
4.1.4 Rangkaian *Hardware*

Sistem kendali jemuran pintar pada penelitian ini menggunakan beberapa komponen *hardware* yang dirangkai secara terintegrasi untuk menghasilkan fungsi kendali otomatis maupun manual berbasis perintah suara.



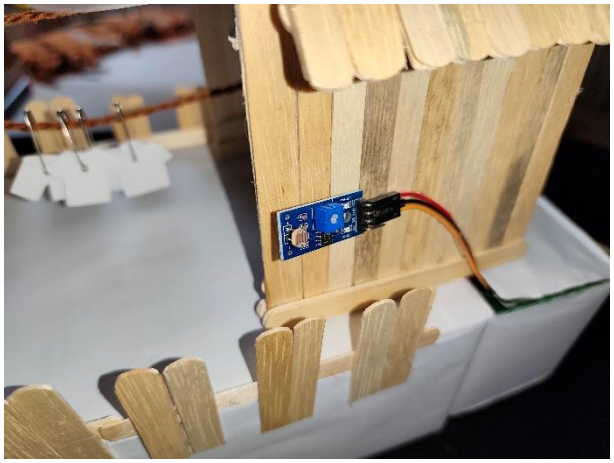
Gambar 4.7 Rangkaian *Hardware*

Berdasarkan pada Gambar 4.7, beberapa komponen utama ditempatkan di dalam sebuah kotak kontrol yang terdiri dari mikrokontroler ESP32 sebagai pusat kendali utama, *driver motor* ULN2003 sebagai pengendali putaran dari *motor stepper* 28BYJ-48, dan *barrel jack* sebagai penyambung daya listrik adaptor DC 5V 3A untuk daya dari *driver motor* ULN2003. Komponen-komponen di atas terhubung ke sebuah *breadboard* supaya distribusi tegangan dapat lebih teratur.



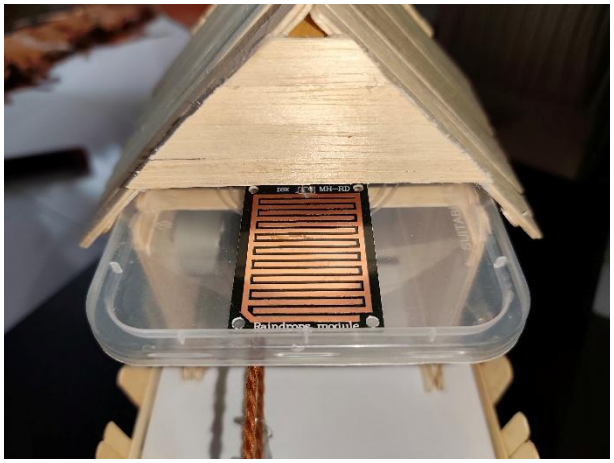
Gambar 4.8 *Motor Stepper* 28BYJ-48

Untuk proses menggerakkan jemuran, ESP32 berperan sebagai penerima instruksi yang dikirimkan dari *backend* melalui pesan MQTT pada topik "jemuran/cmd". Setelah sebuah perintah suara seperti "MASUK" atau "KELUAR" diterima, ESP32 menafsirkan nilai *intent* tersebut dan mengaktifkan *driver* ULN2003 untuk mengatur putaran *motor stepper* 28BYJ-48. *Motor stepper* kemudian memutar *pulley* yang terhubung dengan tali jemuran, sehingga menghasilkan gerakan menarik atau mengulur jemuran sesuai instruksi. Posisi *motor stepper* dan *pulley* diletakkan di dalam rumah jemuran seperti yang tertera pada Gambar 4.8.



Gambar 4.9 Sensor Cahaya LDR LM393

Sensor cahaya (LDR) digunakan untuk mendeteksi tingkat intensitas cahaya yang memancar di lingkungan sekitar jemuran. Pada prototipe, modul sensor LDR diletakkan pada bagian samping kiri rumah jemuran menghadap area tempat pakaian digantung, sebagaimana ditunjukkan pada Gambar 4.9. Sensor ini akan mendeteksi tingkat intensitas cahaya dengan memberikan sinyal digital ke ESP32 untuk diidentifikasi apakah cahaya sedang terang atau gelap. Hal tersebut memungkinkan sistem jemuran pintar dapat mengambil keputusan secara otomatis, seperti menarik jemuran apabila cahaya di lingkungan sekitar jemuran mulai meredup.



Gambar 4.10 Sensor Hujan YL-83

Selain sensor cahaya, sistem jemuran pintar juga dilengkapi dengan sensor hujan YL-83 yang berfungsi untuk mendeteksi adanya tetesan air di lingkungan sekitar jemuran. Seperti yang ditunjukkan pada Gambar 4.10, sensor ini ditempatkan di bagian atap rumah jemuran supaya dapat mendeteksi tetesan air secara langsung ketika hujan terjadi. Sensor ini menghasilkan *output* berupa sinyal digital yang akan dikirimkan ke ESP32 sebagai indikator untuk menarik jemuran ke dalam guna melindungi pakaian dari cuaca hujan supaya tidak basah kembali.

4.2 Hasil Pengujian Sistem

Dalam memastikan semua fungsi berjalan sesuai dengan rancangan, dilakukan pengujian sistem untuk mengevaluasi kinerja dari sistem jemuran pintar yang dibangun. Pada penelitian ini, meliputi pengujian sistem kendali melalui perintah suara, pengujian kendali jemuran secara otomatis berbasis sensor cahaya dan sensor hujan, serta pengujian *error* sistem pada proses pengenalan dan pemrosesan perintah suara.

4.2.1 Pengujian Sistem Kendali

Pengujian sistem kendali bertujuan untuk memvalidasi keberhasilan atas berjalannya proses masuk dan keluar jemuran yang diperoleh melalui perintah suara dengan integrasi antara sistem *software* dan *hardware*. Pengujian dilakukan dengan kondisi perangkat kendali (HP Android) terhubung ke internet dengan jaringan seluler untuk memastikan sistem tetap dapat berjalan pada jaringan internet yang bervariatif. Dalam pengujian ini, ESP32 terhubung ke *broker* MQTT dengan jaringan lokal, lalu server *backend publish* pesan ke *broker* MQTT berdasarkan perintah suara yang diperoleh dari aplikasi Dijemurin dengan topik "jemuran/cmd".

(a)

(b)

Gambar 4.11 (a) Tampilan Aplikasi Perintah Jemuran Keluar (b) Kondisi Jemuran Keluar

Pengujian diawali dengan kondisi *motor stepper* berada di posisi masuk (jemuran di dalam). Dalam skenario ini, pengujian ditujukan untuk mengeluarkan

jemuran melalui perintah suara yang diberikan oleh pengguna melalui aplikasi sistem kendali. Pada Gambar 4.11, dapat dilihat bahwa sistem berhasil memvalidasi perintah suara pengguna dengan memberikan *output* berupa teks dan suara respons, dan *motor stepper* juga merespons perintah tersebut dengan bergerak ke arah luar sesuai perintah yang diberikan.

(a)

(b)

Gambar 4.12 (a) Tampilan Aplikasi Perintah Jemuran Masuk (b) Kondisi Jemuran Masuk

Pengujian berlanjut dengan menguji proses sebaliknya, yaitu memasukkan jemuran ke dalam menggunakan perintah suara yang diberikan oleh pengguna melalui aplikasi sistem kendali. Dalam Gambar 4.12, sistem berhasil memvalidasi perintah pengguna untuk memasukkan jemuran ke dalam dengan memberikan *output* berupa teks dan suara respons, dan *motor stepper* juga berhasil merespons perintah tersebut dengan bergerak masuk.

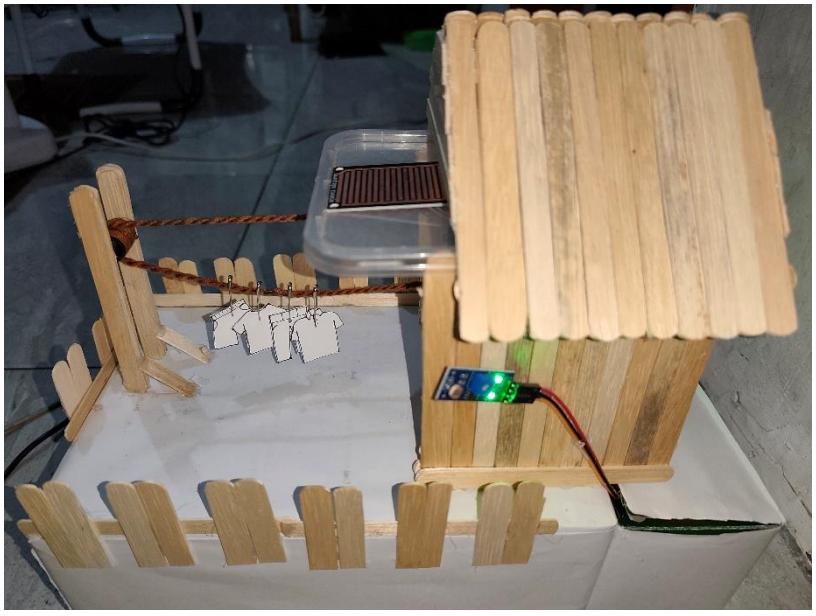
Gambar 4.13 Tampilan Aplikasi Perintah Tidak Terdaftar

Kemudian pada Gambar 4.13 memperlihatkan hasil respons ketika pengguna memberikan perintah suara yang tidak terdaftar dalam sistem. Dalam pengujian ini, sistem memberikan respons berupa teks dan suara untuk memberitahu pengguna bahwa perintah yang diberikan tidak dikenali. Hal ini disebabkan karena setiap kata yang diucapkan dalam perintah tersebut tidak terdapat pola perintah (“Masuk” dan “Keluar”) yang telah didefinisikan dalam sistem. Karena hal tersebut, sistem tidak mengirimkan pesan apapun ke ESP32 dan *motor stepper* tidak bergerak.

4.2.2 Pengujian Kendali Jemuran Otomatis

Pengujian ini dilakukan untuk memastikan sistem jemuran dapat dikendalikan secara otomatis berdasarkan hasil deteksi dari sensor. Sensor cahaya

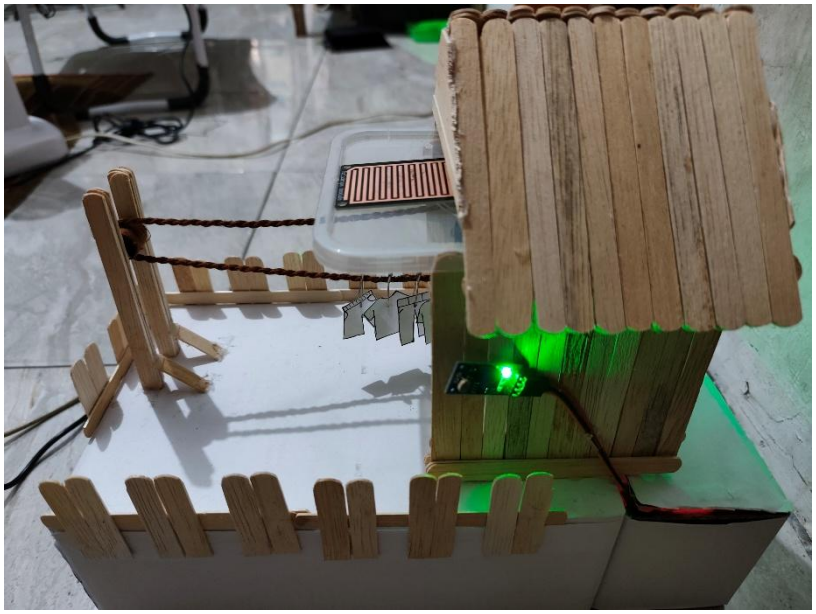
(LDR) dan sensor hujan (*raindrop*) dalam penelitian ini berperan untuk mendeteksi kondisi cuaca di sekitar jemuran. Untuk memastikan sensor berfungsi dengan baik, pengujian dilakukan pada kondisi yang berbeda-beda untuk tiap sensor. Sensor cahaya (LDR) akan diuji pada kondisi cahaya cerah dan gelap, sedangkan sensor hujan akan diuji di kondisi ketika basah atau kering. Dalam pengujian ini, tiap sensor tidak diuji secara langsung di lingkungan *outdoor* dengan tujuan demi keamanan alat jemuran, mengingat jemuran masih berupa protipe dan terbuat dari material-material yang cukup ringan, serta tidak anti air.



Gambar 4.14 Kondisi Cerah dan Kering

Pengujian dimulai dengan menguji sensor cahaya (LDR) dan sensor hujan (*raindrop*) untuk mendeteksi kondisi ketika sedang cerah dan kering (tidak hujan). Sensor LDR bekerja berdasarkan level intensitas cahaya yang mengenainya, sedangkan sensor *raindrop* bekerja berdasarkan prinsip perubahan nilai resitansi akibat adanya air yang menyentuh papan sensor. Nilai resistansi tiap sensor akan

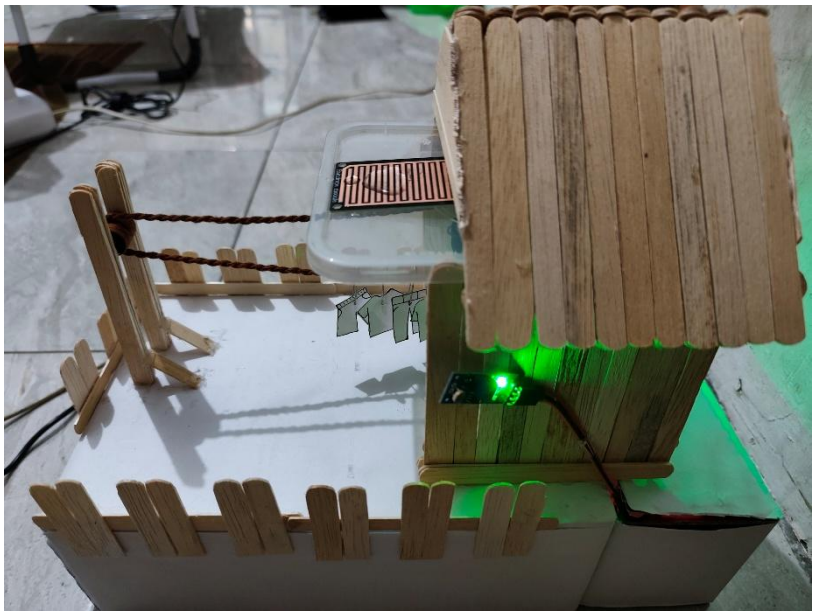
berubah berdasarkan kondisi tertentu. Dalam skenario pengujian ini, sensor LDR mendeteksi adanya cahaya dan mengirimkan sinyal digital LOW atau 0 ke ESP32, sedangkan sensor *raindrop* di sini mendeteksi tidak adanya air yang mengenai papan sensor dan mengirimkan sinyal digital HIGH atau 1 ke ESP32. Seperti yang dapat dilihat pada Gambar 4.14, sistem jemuran berhasil merespon hasil deteksi sensor dengan mengeluarkan jemuran keluar dari atap rumah jemuran melalui *motor stepper*.



Gambar 4.15 Kondisi Gelap dan Kering

Dalam Gambar 4.15 merupakan hasil dari pengujian sensor LDR dan *raindrop* pada kondisi ketika gelap dan kering. Sensor LDR mendeteksi intensitas cahaya yang rendah yang menandakan kondisi cuaca gelap atau mendung, lalu sensor mengirimkan sinyal digital HIGH atau 1 ke ESP32. Kemudian sensor *raindrop* berada dalam kondisi papan sensor yang kering, menandakan bahwa kondisi cuaca tidak hujan, sehingga mengirimkan sinyal digital HIGH atau 1 ke

ESP32. Hasilnya, sistem jemuran berhasil menggerakkan *motor stepper* ke dalam atap rumah jemuran.



Gambar 4.16 Kondisi Gelap dan Basah

Gambar 4.16 menunjukkan hasil pengujian untuk sensor LDR dan *raindrop* pada kondisi ketika gelap dan basah. Dalam kondisi cahaya yang gelap, sensor LDR mendeteksi intensitas cahaya menurun, hal ini menandakan kondisi cuaca yang gelap atau mendung, lalu mengirimkan sinyal digital HIGH atau 1 ke ESP32. Kemudian sensor *raindrop* mendeteksi adanya air yang menyebabkan papan sensor menjadi basah, ini menjadi tanda kondisi cuaca sedang hujan, lalu sensor mengirimkan sinyal LOW atau 0 ke ESP32. Sebagai hasilnya, *motor stepper* aktif dengan bergerak berdasarkan instruksi dari sistem untuk memindahkan jemuran ke dalam atap rumah jemuran.

4.2.3 Pengujian *Error* Sistem

Pengujian *error* sistem dilakukan untuk mengetahui tingkat kesalahan sistem jemuran pintar dalam memproses perintah suara pengguna menggunakan algoritma *Raita* sebagai metode pencocokan pola. Pengujian ini melibatkan 8 orang pengguna sebagai responden, di mana masing-masing pengguna mengucapkan berbagai variasi kalimat yang merepresentasikan empat pola perintah, yaitu "masuk", "keluar", "jangan masuk", dan "jangan keluar". Setiap pola perintah diuji menggunakan 15 variasi kalimat, sehingga total pengujian untuk setiap pola adalah 120 percobaan (15 variasi × 8 pengguna), dengan total keseluruhan 480 percobaan.

Tujuan dari pengujian ini adalah untuk mengukur tingkat kesalahan sistem dalam mengenali pola perintah berdasarkan hasil konversi suara menjadi teks (*speech recognition*) yang kemudian diproses menggunakan algoritma *Raita*. Hasil pengujian disajikan dalam bentuk tabel untuk masing-masing pola perintah, sehingga dapat dianalisis perbedaan tingkat kesalahan pada setiap pola perintah.

Tabel 4.1 Hasil Pengujian Pola Perintah "Masuk" oleh 8 Orang Pengguna

Pola Perintah "Masuk"					
No.	***Input* Teks**	***Pattern***	**Jumlah Pengguna**	***Success***	***Error***
1.	"tolong masukin jemuran"	masuk	8	8	0
2.	"masukkan jemuran sekarang"	masuk	8	8	0
3.	"tarik jemuran ke dalam"	masuk	8	8	0
4.	"bawa jemuran masuk sekarang"	masuk	8	8	0
5.	"amankan jemurannya"	masuk	8	8	0
6.	"simpan jemurannya"	masuk	8	8	0
7.	"pindahkan jemuran ke dalam"	masuk	8	8	0
8.	"tolong masukin jemuran"	masuk	8	8	0
9.	"segera dimasukin jemurannya"	masuk	8	8	0
10.	"masukin dulu jemurannya"	masuk	8	8	0

Pola Perintah "Masuk"					
No.	***Input* Teks**	***Pattern***	**Jumlah Pengguna**	***Success***	***Error***
11.	"angkat jemuran"	masuk	8	8	0
12.	"ambil jemuran ke dalam"	masuk	8	1	7
13.	"tarik masuk jemuran"	masuk	8	8	0
14.	"masukin dulu sebelum hujan"	masuk	8	8	0
15.	"masukkan dulu sebelum keluar"	masuk	8	8	0
$\sum Ei$ (Jumlah *error* dari semua pengujian)					**7**

Pengujian pertama diawali dengan menguji sistem dalam mengenali pola perintah "Masuk" yang dilakukan oleh 8 orang pengguna sebagai responden, masing-masing pengguna memberikan 15 perintah dengan kalimat yang bervariasi. Berdasarkan hasil pengujian pada Tabel 4.1, diperoleh total *error* pada sistem sebanyak 7 kesalahan dalam 1 teks perintah yang sama, yaitu "ambil jemuran ke dalam".

Tabel 4.2 Hasil Pengujian Pola Perintah "Keluar" oleh 8 Orang Pengguna

Pola Perintah "Keluar"					
No.	***Input* Teks**	***Pattern***	**Jumlah Pengguna**	***Success***	***Error***
1.	"tolong dong keluarin jemurannya"	keluar	8	8	0
2.	"keluarkan jemurannya dong"	keluar	8	8	0
3.	"jemur di luar rumah"	keluar	8	8	0
4.	"taruh jemuran di luar"	keluar	8	8	0
5.	"gantung jemuran di luar"	keluar	8	8	0
6.	"jemurkan pakaian di luar"	keluar	8	8	0
7.	"bawa keluar jemuran"	keluar	8	8	0
8.	"keluarkan cucian sekarang"	keluar	8	8	0
9.	"tolong jemur sekarang"	keluar	8	8	0
10.	"bentangkan jemuran"	keluar	8	8	0
11.	"taruh cucian di luar"	keluar	8	8	0
12.	"gantungin cucian"	keluar	8	1	7
13.	"keluarkan dulu sebelum masuk"	keluar	8	8	0
14.	"pindahin jemuran keluar"	keluar	8	8	0

Pola Perintah "Keluar"					
No.	***Input* Teks**	***Pattern***	**Jumlah Pengguna**	***Success***	***Error***
15.	"bisa tolong jemur sekarang"	keluar	8	8	0
$\sum Ei$ (Jumlah *error* dari semua pengujian)					**7**

Pengujian berikutnya dilakukan untuk menguji tingkat kesalahan sistem dalam mengenali pola perintah "Keluar" yang dilakukan oleh 8 orang pengguna, masing-masing pengguna memberikan perintah sebanyak 15 kali dengan variasi kalimat yang berbeda. Pada Tabel 4.2 merupakan hasil pengujian pada sistem dengan perolehan *error* sebanyak 7 kesalahan dalam 1 teks perintah yang sama, yaitu teks perintah "gantungin cucian".

Tabel 4.3 Hasil Pengujian Pola Perintah "Jangan Masuk" oleh 8 Orang Pengguna

Pola Perintah "Jangan Masuk"					
No.	***Input* Teks**	***Pattern***	**Jumlah Pengguna**	***Success***	***Error***
1.	"jangan masuk dulu sekarang"	jangan masuk	8	8	0
2.	"tolong jemurannya jangan dimasukkan"	jangan masuk	8	8	0
3.	"jangan masukin jemuran dulu"	jangan masuk	8	8	0
4.	"jangan sampai jemurannya masuk"	jangan masuk	8	8	0
5.	"jangan dimasukkan sekarang"	jangan masuk	8	8	0
6.	"jangan masuk sebelum hujan"	jangan masuk	8	8	0
7.	"jangan masuk ya nanti saja"	jangan masuk	8	8	0
8.	"jangan sampai jemuran masuk"	jangan masuk	8	0	8
9.	"tolong jangan masukin dulu"	jangan masuk	8	8	0
10.	"biarkan di luar"	jangan masuk	8	8	0
11.	"nggak usah masukin jemurannya"	jangan masuk	8	0	8
12.	"jangan tarik jemuran ke dalam"	jangan masuk	8	8	0
13.	"jangan angkat jemurannya dulu"	jangan masuk	8	8	0
14.	"jangan simpan jemuran dulu"	jangan masuk	8	8	0

Pola Perintah "Jangan Masuk"					
No.	***Input* Teks**	***Pattern***	**Jumlah Pengguna**	***Success***	***Error***
15.	"jangan ditarik masuk dulu"	jangan masuk	8	8	0
$\sum Ei$ (Jumlah *error* dari semua pengujian)					**16**

Pengujian selanjutnya dilakukan untuk menguji tingkat kesalahan sistem dalam mengenali pola perintah negasi "Jangan Masuk" yang dilakukan oleh 8 orang pengguna, masing-masing memberikan perintah sebanyak 15 kali dengan teks perintah yang bervariasi. Dalam Tabel 4.3 menunjukkan hasil pengujian dengan total *error* sistem sebanyak 16 kesalahan pada 2 teks perintah yang berbeda. Diperoleh 8 *error* untuk teks perintah "jangan sampai jemuran masuk" dan 8 *error* untuk teks perintah "nggak usah masukin jemurannya".

Tabel 4.4 Hasil Pengujian Pola Perintah "Jangan Keluar" oleh 8 Orang Pengguna

Pola Perintah "Jangan Keluar"					
No.	***Input* Teks**	***Pattern***	**Jumlah Pengguna**	***Success***	***Error***
1.	"jangan keluar dulu sekarang"	jangan keluar	8	8	0
2.	"jangan keluarin jemuran dulu"	jangan keluar	8	8	0
3.	"jangan sampai jemuran keluar"	jangan keluar	8	0	8
4.	"jangan taruh jemuran di luar"	jangan keluar	8	8	0
5.	"jangan dikeluarkan sekarang"	jangan keluar	8	8	0
6.	"jangan keluar sebelum cerah"	jangan keluar	8	8	0
7.	"jangan keluar ya nanti saja"	jangan keluar	8	8	0
8.	"nggak usah keluarin jemurannya"	jangan keluar	8	0	8
9.	"jangan keluarkan jemuran"	jangan keluar	8	8	0
10.	"jangan taruh jemuran di luar"	jangan keluar	8	8	0
11.	"jangan gantung jemuran dulu"	jangan keluar	8	8	0
12.	"jangan bentangkan jemurannya dulu"	jangan keluar	8	8	0

Pola Perintah "Jangan Keluar"					
No.	***Input* Teks**	***Pattern***	**Jumlah Pengguna**	***Success***	***Error***
13.	"jangan jemur dulu"	jangan keluar	8	8	0
14.	"jangan taruh di luar dulu"	jangan keluar	8	8	0
15.	"tolong jangan keluarkan jemurannya"	jangan keluar	8	8	0
$\sum Ei$ (Jumlah *error* dari semua pengujian)					**16**

Pengujian diakhiri dengan menguji tingkat kesalahan sistem dalam mengenali pola perintah "Jangan Keluar" yang dilakukan oleh 8 orang pengguna sebagai responden, yang masing-masing memberikan perintah sebanyak 15 kali dengan teks perintah yang bervariasi. Hasil pengujian pada Tabel 4.4 diperoleh total *error* sebanyak 16 kesalahan dalam 2 teks perintah yang berbeda-beda. Hasil pengujian untuk teks perintah "jangan sampai jemuran keluar" memperoleh kesalahan sebanyak 8 *error* dan teks perintah "nggak usah keluarin jemurannya" memperoleh kesalahan sebanyak 8 *error*.

Berdasarkan keseluruhan pengujian terhadap empat pola perintah dengan total 480 percobaan, diperoleh jumlah *error* sistem sebanyak 46 kesalahan. Dengan demikian, jumlah perintah yang berhasil dikenali oleh sistem adalah sebanyak 434 percobaan. Tingkat *error* atau kesalahan sistem dihitung menggunakan persamaan 4.1 di bawah ini:

$$\mu_{error} = \frac{\sum Ei}{n} \times 100\%$$

$$\mu_{error} = \frac{46}{480} \times 100\% \qquad (4.1)$$

$$\mu_{error} = 9.58\%$$

Apabila dianalisis lebih lanjut, pola perintah afirmatif seperti “masuk” dan “keluar” menunjukkan tingkat kesalahan yang lebih rendah dibandingkan pola perintah negasi “jangan masuk” dan “jangan keluar”. Distribusi *error* yang masih muncul pada kalimat-kalimat tertentu menunjukkan bahwa kinerja algoritma *Raita* sangat dipengaruhi oleh keberadaan dan struktur kata kunci dalam teks hasil pengenalan suara. Kesalahan cenderung terjadi pada variasi kalimat yang menggunakan bentuk sinonim atau konstruksi bahasa yang lebih kompleks.

Temuan ini menunjukkan bahwa performa sistem tidak hanya bergantung pada algoritma pencocokan pola, tetapi juga dipengaruhi oleh variasi linguistik dari pengguna serta hasil konversi *speech recognition*. Oleh karena itu, diperlukan pembahasan lebih lanjut untuk menganalisis faktor-faktor yang menyebabkan terjadinya *error* serta implikasinya terhadap pengembangan sistem ke depan.

4.3 Pembahasan

Berdasarkan hasil pengujian *error* sistem yang telah dilakukan terhadap empat pola perintah dengan total 480 percobaan, sistem jemuran pintar memperoleh tingkat *error* sebesar 9,58%. Hasil ini menunjukkan bahwa sistem telah mampu mengenali sebagian besar perintah suara pengguna dengan baik melalui proses *speech recognition* yang kemudian diproses lebih lanjut menggunakan algoritma *Raita* sebagai metode pencocokan pola.

Jika ditinjau berdasarkan jenis pola perintah, sistem menunjukkan performa yang lebih stabil pada pola perintah afirmatif seperti “masuk” dan “keluar”. Kedua pola tersebut memiliki jumlah *error* yang relatif rendah dibandingkan dengan pola perintah negasi “jangan masuk” dan “jangan keluar”. Hal ini menunjukkan bahwa

algoritma *Raita* bekerja lebih optimal ketika mendeteksi kata kunci utama yang muncul secara eksplisit dan langsung di dalam kalimat.

Algoritma *Raita* merupakan algoritma pencocokan *string* berbasis *pattern matching* yang melakukan perbandingan karakter untuk menemukan kecocokan pola di dalam teks. Dalam konteks penelitian ini, algoritma bekerja dengan mencari kemunculan pola tertentu pada teks hasil *speech recognition*. Oleh karena itu, keberhasilan sistem sangat bergantung pada keberadaan kata kunci yang identik dengan pola yang telah ditentukan sebelumnya. Variasi kalimat yang menggunakan sinonim atau struktur bahasa yang berbeda berpotensi menyebabkan ketidaksesuaian pencocokan, meskipun secara makna perintah tersebut masih relevan.

Selain faktor struktur kalimat, hasil pengujian juga menunjukkan bahwa unsur negasi menjadi tantangan tersendiri dalam sistem berbasis pencocokan pola sederhana. Pada beberapa teks perintah negasi, sistem mengalami kesalahan karena pola yang dicari tidak selalu berada dalam posisi atau bentuk kata yang konsisten. Hal ini menunjukkan bahwa pendekatan algoritma *exact string matching* belum sepenuhnya mampu menangani kompleksitas bahasa alami yang memiliki variasi bentuk kata, imbuhan, dan konteks semantik.

Selain dipengaruhi oleh mekanisme pemrosesan teks pada sistem, tingkat *error* yang terjadi selama pengujian juga dipengaruhi oleh beberapa variabel yang tidak dapat dikontrol secara penuh. Variabel tersebut seperti kondisi lingkungan yang bising yang mempengaruhi proses penangkapan suara oleh mikrofon pada perangkat *smartphone* sehingga hasil transkripsi yang dihasilkan oleh sistem *speech*

recognition tidak selalu sesuai dengan perintah yang dimaksud. Selain itu, pengucapan perintah yang terlalu cepat, terlalu pelan, maupun variasi pelafalan antar pengguna juga dapat menyebabkan perubahan hasil transkripsi teks yang pada akhirnya berpotensi menimbulkan kesalahan dalam *pattern matching* pada sistem.

Namun demikian, dengan tingkat *error* sebesar 9.58%, sistem dapat dikategorikan memiliki performa yang baik untuk implementasi kendali perangkat IoT berbasis perintah suara sederhana. Hasil ini menunjukkan bahwa algoritma *Raita* sudah dapat dikatakan efektif untuk digunakan dalam sistem *Internet of Things* (IoT) yang memerlukan proses pencocokan teks secara cepat dan ringan dari sisi komputasi.

Hasil penelitian ini mengindikasikan bahwa pendekatan pencocokan pola menggunakan algoritma *Raita* mampu memberikan performa yang memadai untuk sistem jemuran pintar berbasis perintah suara, namun masih memiliki keterbatasan dalam menangani variasi linguistik yang lebih kompleks. Oleh karena itu, pengembangan lebih lanjut dapat mempertimbangkan integrasi teknik pemrosesan bahasa alami (*Natural Language Processing*) yang lebih adaptif untuk meningkatkan akurasi sistem di masa mendatang.

Selain menunjukkan capaian kinerja sistem secara teknis, hasil penelitian ini juga merefleksikan integrasi nilai-nilai keislaman dalam proses perancangan hingga pengujiannya. Prinsip *ikhtiar* tercermin dari upaya sistematis dalam merancang, mengimplementasikan, serta menguji sistem jemuran pintar melalui 480 kali pengujian terstruktur. Proses tersebut menunjukkan bahwa keberhasilan sistem bukanlah hasil kebetulan, melainkan buah dari usaha yang direncanakan dan

dievaluasi secara ilmiah. Hal ini sejalan dengan firman Allah Subhanahu wa ta'ala dalam *QS. Ar-Ra'd (13:11)*:

لَهٗ مُعَقِّبٰتٌ مِّنْۢ بَيْنِ يَدَيْهِ وَمِنْ خَلْفِهٖ يَحْفَظُوْنَهٗ مِنْ اَمْرِ اللّٰهِ اِنَّ اللّٰهَ لَا يُغَيِّرُ مَا بِقَوْمٍ حَتّٰى يُغَيِّرُوْا مَا بِاَنْفُسِهِمْ وَاِذَآ اَرَادَ اللّٰهُ بِقَوْمٍ سُوْۤءًا فَلَا مَرَدَّ لَهٗ وَمَا لَهُمْ مِّنْ دُوْنِهٖ مِنْ وَّالٍ

"Baginya (manusia) ada (malaikat-malaikat) yang menyertainya secara bergiliran dari depan dan belakangnya yang menjaganya atas perintah Allah. Sesungguhnya Allah tidak mengubah keadaan suatu kaum hingga mereka mengubah apa yang ada pada diri mereka. Apabila Allah menghendaki keburukan terhadap suatu kaum, tidak ada yang dapat menolaknya, dan sekali-kali tidak ada pelindung bagi mereka selain Dia." (QS. Ar-Ra'd: 11)

Ayat tersebut menegaskan bahwa perubahan dan perbaikan memerlukan usaha aktif. Dalam konteks penelitian ini, usaha tersebut terwujud melalui perancangan algoritma, pengujian berulang, serta analisis berbasis data untuk memastikan sistem bekerja sesuai tujuan.

Selanjutnya, penerapan algoritma *Raita* dalam proses pencocokan pola perintah serta evaluasi tingkat kesalahan melalui perhitungan persentase menunjukkan adanya komitmen terhadap ketelitian dan profesionalisme. Nilai ini selaras dengan konsep *itqan* (sungguh-sungguh dan profesional), sebagaimana sabda Rasulullah Shalallaahu 'Alaihi Wasallam yang diriwayatkan oleh Thabrani, No. 891, Baihaqi, No. 334:

عَنْ عَائِشَةَ رَضِيَ اللهُ عَنْهَا قَالَتْ: قَالَ رَسُوْلُ اللهِ صَلَّى اللهُ عَلَيْهِ وَسَلَّمَ: إِنّ اللَّهَ تَعَالى يُحِبّ إِذَا عَمِلَ أَحَدُكُمْ عَمَلاً أَنْ يُتْقِنَهُ (رواه الطبرني والبيهقي)

Dari Aisyah Radhiyallahu Anhu., sesungguhnya Rasulullah Shalallaahu 'Alaihi Wasallam. bersabda: "Sesungguhnya Allah mencintai seseorang yang apabila bekerja, mengerjakannya secara profesional". (HR. Thabrani, no. 891, Baihaqi, no. 334).

Ketelitian dalam menyusun logika sistem, mengontrol variabel pengujian, serta menyajikan hasil secara kuantitatif mencerminkan penerapan prinsip tersebut dalam konteks pengembangan teknologi.

Lebih lanjut, aspek *amanah* juga tercermin dalam penelitian ini, baik sebagai tanggung jawab ilmiah maupun sebagai tanggung jawab sistem terhadap pengguna. Data hasil pengujian disajikan secara objektif sesuai kondisi aktual tanpa manipulasi, dan sistem dirancang untuk menjalankan fungsinya secara konsisten dalam membantu menjaga pakaian pengguna dari risiko cuaca. Hal ini sejalan dengan firman Allah Subhanahu wa ta'ala dalam *QS. An-Nisa (4:58)*:

اِنَّ اللّٰهَ يَأْمُرُكُمْ اَنْ تُؤَدُّوا الْاَمٰنٰتِ اِلٰٓى اَهْلِهَاۙ وَاِذَا حَكَمْتُمْ بَيْنَ النَّاسِ اَنْ تَحْكُمُوْا بِالْعَدْلِۗ اِنَّ اللّٰهَ نِعِمَّا يَعِظُكُمْ بِهٖۗ
اِنَّ اللّٰهَ كَانَ سَمِيْعًاۢ بَصِيْرًا

"Sesungguhnya Allah menyuruh kamu menyampaikan amanah kepada pemiliknya. Apabila kamu menetapkan hukum di antara manusia, hendaklah kamu tetapkan secara adil. Sesungguhnya Allah memberi pengajaran yang paling baik kepadamu. Sesungguhnya Allah Maha Mendengar lagi Maha Melihat."(QS. An-Nisa: 58)

Dengan demikian, kesesuaian antara rancangan sistem dan hasil pengujian tidak hanya menunjukkan validitas dan kinerja teknis, tetapi juga mencerminkan nilai usaha maksimal (*ikhtiar*), profesionalisme (*itqan*), serta tanggung jawab (*amanah*) dalam proses penelitian. Integrasi nilai-nilai tersebut memperlihatkan bahwa pengembangan teknologi dapat berjalan selaras dengan prinsip-prinsip etika dan spiritual dalam Islam.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Penelitian ini bertujuan untuk merancang dan mengimplementasikan sistem kendali jemuran pintar berbasi IoT menggunakan algoritma *Raita* dengan konsep *speech recognition*, serta melakukan pengujian *error* untuk mengetahui tingkat kesalahan sistem. Sistem berhasil diimplementasikan melalui integrasi aplikasi Android dan perangkat keras berbasis ESP32, *motor stepper* 28BYJ-48, serta sensor LDR dan *raindrop* yang mendukung mode manual dan otomatis. Algoritma *Raita* digunakan untuk mengenali empat pola perintah utama setelah melalui *text preprocessing* hasil *speech recognition*. Berdasarkan 480 kali pengujian yang telah dilakukan, sistem berhasil mengenali 434 perintah dengan tingkat *error* sebesar 9,58%, sehingga menunjukkan performa yang tergolong baik dalam memproses perintah suara sederhana pada sistem IoT. Namun, pendekatan *exact string matching* yang digunakan masih bergantung pada kesesuaian *keyword* dengan *pattern* yang telah ditentukan. Selain itu, hasil pengujian juga dipengaruhi oleh beberapa variabel eksternal, seperti tingkat kebisingan lingkungan, variasi pelafalan dan intonasi pengguna, serta kecepatan pengucapan. Faktor-faktor tersebut dapat mempengaruhi hasil transkripsi yang berdampak pada proses *pattern matching* oleh algoritma. Dengan demikian, meskipun tujuan penelitian telah tercapai, sistem masih memerlukan pengembangan lebih lanjut untuk meningkatkan fleksibilitas dan akurasi dalam menangani variasi bahasa alami pada perintah suara.

5.2 Saran

Penelitian ini masih memiliki beberapa keterbatasan yang bisa diperbaiki pada pengembangan selanjutnya untuk meningkatkan kinerja sistem supaya lebih optimal. Adapun beberapa saran pengembangan yang dapat dipertimbangkan adalah sebagai berikut:

1. Mengembangkan metode pengenalan perintah dengan pendekatan yang lebih fleksibel, seperti penerapan *Natural Language Processing* (NLP) atau metode berbasis *machine learning*, agar sistem tidak hanya bergantung pada pencocokan *string literal* dan mampu menangani variasi bahasa yang lebih kompleks.
2. Mengembangkan sistem dari tahap prototipe sederhana ke implementasi skala penuh serta melakukan pengujian langsung pada lingkungan *outdoor*, sehingga performa dan ketahanan sistem dapat dievaluasi dalam kondisi penggunaan yang sebenarnya.

DAFTAR PUSTAKA

Ardiani, L., Sujaini, H., & Tursina, T. (2020). Implementasi Sentiment Analysis Tanggapan Masyarakat Terhadap Pembangunan di Kota Pontianak. *Jurnal Sistem Dan Teknologi Informasi (Justin)*, *8*(2), 183. https://doi.org/10.26418/justin.v8i2.36776

Bawanto, D. R., & Rosmawanti, N. (2017). Perbandingan Algoritma Binary Search Dan *Raita* Dalam Pencarian Data. *Jutisi : Jurnal Ilmiah Teknik Informatika Dan Sistem Informasi*, *6*(1), 1311–1317. https://doi.org/https://dx.doi.org/10.35889/jutisi.v6i1.225

BMKG. (2024). *Catatan Iklim Dan Kualitas Udara Indonesia 2024*. https://iklim.bmkg.go.id/bmkgadmin/storage/buletin/Catatan Iklim dan Kualitas Udara 2024 BMKG.pdf

BPS. (2025). *Indikator Pasar Tenaga Kerja Indonesia Februari 2025* (Vol. 16, Issue 1). https://www.bps.go.id/id/publication/2025/06/30/7cc0d8a5ea09515a0b2a2729/indikator-pasar-tenaga-kerja-indonesia-februari-2025.html

Gaddam, A., Wilkin, T., Angelova, M., & Gaddam, J. (2020). Detecting sensor faults, anomalies and outliers in the internet of things: A survey on the challenges and solutions. *Electronics (Switzerland)*, *9*(3), 1–15. https://doi.org/10.3390/electronics9030511

Ginting, R. (2019). Refleksi Hadits terhadap Kualitas Pelayanan Referensi dalam Membantu Memenuhi Kebutuhan Informasi Pemustaka di Perguruan Tinggi. *Pustakaloka*, *11*(1), 131. https://doi.org/10.21154/pustakaloka.v11i1.1565

Hakak, S. I., Kamsin, A., Shivakumara, P., Gilkar, G. A., Khan, W. Z., & Imran, M. (2019). Exact String Matching Algorithms: Survey, Issues, and Future Research Directions. *IEEE Access*, *7*, 69614–69637. https://doi.org/10.1109/ACCESS.2019.2914071

Harahap, A. F., & Ginting, G. L. (2020). Penerapan Algoritma *Raita* pada Kamus Akronim Bahasa Indonesia Berbasis Android. *Terapan Informatika Nusantara*, *1*(3), 126–132. https://ejurnal.seminar-id.com/index.php/tin/article/view/426/276

Hudaa, S., Nguyen, P. T., Lestari, S. P., Gunawan, G., & Supiyandi, S. (2020). Data search using raita algorithm. *Journal of Critical Reviews*, *7*(1), 72–75. https://doi.org/10.22159/jcr.07.01.14

Jain, S., & Rao, A. L. N. (2013). A comparative performance analysis of approximate string matching. *International Journal of Innovative Technology and*

Exploring Engineering, *3*(5), 123–128. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=aaadcdc2f1964957f587e780d4a66fe8770fa6e6

Kaur, N. (2023). Development of a Smart Home Automation System Using IoT Technology. *International Journal of Mechanical Engineering*, *8*(1), 106–112. https://doi.org/10.56452/6-14

Khasanah, U. (2023). Proses Turunnya Hujan dalam Tafsir Al-Maraghi (Kajian Saintifik Ayat-Ayat Hujan). *Al-Muntaha: Jurnal Ilmu Al-Qur'an Dan Tafsir*, *4*(2), 21–36. https://ojs.unsiq.ac.id/index.php/mth/article/view/8914/3665

Maesyaroh, S., Daswa, & Suwarto, G. P. (2023). Perbandingan Algoritma Horspool dan *Raita* Pada Pencarian Data Mahasiswa. *Jurnal Buffer Informatika*, *9*(2), 1–9. https://doi.org/https://doi.org/10.25134/buffer.v9i2.9331

Manuhutu, A., Warella, S., Likliwatil, C., Sasauw, C., & Sitopu, J. W. (2025). Mengubah Kehidupan Sehari-hari: Dampak Implementasi Internet of Things (IoT). *Indonesian Research Journal on Education : Jurnal Ilmu Pendidikan*, *5*(1), 1072–1078. https://doi.org/https://doi.org/10.31004/irje.v5i1.2019

Mesi, E., & Oktarina, D. (2021). Penerapan Algoritma Horspool Pada Sistem Pendataan Obat Pada Apotek Fajar Mas. *Seminar Nasional Informatika*, 79–85. http://www.ejournal.pelitaindonesia.ac.id/ojs32/index.php/SENATIKA/article/view/1138

Mohamad Salman Farizi, Somantri, S., & Yustiana, I. (2022). Implementasi Speech Recognition Pada Sistem Kendali Perangkat Elektronik Rumah Berbasis Iot (Internet Of Things) dan Mobile Application. *ZONAsi: Jurnal Sistem Informasi*, *4*(2), 157–166. https://doi.org/10.31849/zn.v4i2.10662

Munna, D. N. (2024). *Sistem Kontrol Kunci Pintu Rumah Berbasis Internet of Things dengan Perintah Suara Menggunakan Naive Pattern Searching Algorithm* [Universitas Islam Negeri Maulana Malik Ibrahim Malang]. http://etheses.uin-malang.ac.id/70991/1/210605110025.pdf

Nindyawati, N., Toyib, R., Rifqo, M. H., & Sahputra, E. (2021). Aplikasi Kamus Bahasa Lembak Bengkulu Berbasis Android Menggunakan Algoritma *Raita*. *JUSIBI (Jurnal Sistem Informasi Dan E-Bisnis)*, *3*(2), 54–64. https://doi.org/10.54650/jusibi.v3i2.357

Nugroho, K. (2019). Implementasi Sistem Speech to Text Berbasis Android Menggunakan APP Inventor Speech Recognition. *Jurnal Ilmiah Infokam*, *15*(1), 38–43. https://amikjtc.com/jurnal/index.php/jurnal/article/view/176

Raita, T. (1992). Tuning the boyer-moore-horspool string searching algorithm. *Software: Practice and Experience*, *22*(10), 879–884. https://doi.org/10.1002/spe.4380221006

Rasool, A., Tiwari, A., Singla, G., & Khare, N. (2012). String Matching Methodologies: A Comparative Analysis. *(IJCSIT) International Journal of Computer Science and Information Technologies*, *3*(2), 3394–3397. https://pdfs.semanticscholar.org/5ed4/52b42f81b6efb9e68b8428c5e992b4d4d143.pdf

Rimadana, H. (2024). Sistem Kontrol Speaker Murottal Al-Qur'an Berbasis Internet Of Things Menggunakan Algoritma Aho-Corasick [Universitas Islam Maulana Malik Ibrahim Malang]. In *Etheses Universitas Islam Negeri Maulana Malik Ibrahim*. http://etheses.uin-malang.ac.id/71067/1/210605110009.pdf

Rimadana, H., Hanani, A., Almais, A. T. W., Utama, S. N., Prakasa, J. E. W., Hariyadi, M. A., & Arif, Y. M. (2025). QIRABOX: Kontrol Speaker Murottal Al-Qur'an dengan Perintah Suara Berbasis Internet of Things. *Multinetics*, *11*(1), 85–93. https://doi.org/10.32722/multinetics.v11i1.7553

Rozy, A. (2024). Implementasi *Raita* Pada Sistem Informasi Manajemen Penjualan Sparepart Mobil. *FIMERKOM: Journal of Information Systems and Technology*, *1*(2), 53–56. https://ejournal.katersipublisher.com/index.php/FIMERKOM/article/view/96/41

Sari, S. S., & Ginting, G. (2021). Implementasi Algoritma Boyer Moore pada kamus perbedaan kata dalam Bahasa Inggris British dan Bahasa Inggris America. *Journal of Informatics Management and Information Technology*, *1*(2), 74–78.

Selay, A., Andgha, G. D., Alfarizi, M. A., Bintang, M. I., Falah, M. N., Khaira, M., & Encep, M. (2022). Internet of Things. *Karimah Tauhid*, *1*(6), 860–868. https://doi.org/https://doi.org/10.30997/karimahtauhid.v1i6.7633

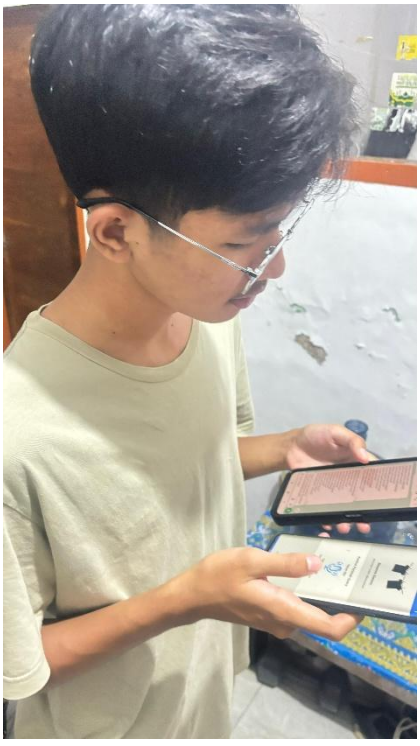
Syahputra, A., & Maulana, H. (2025). Design and Construction of Automatic Clothes Drying Rack Prototype Based on IoT (Internet of Things). *Tsabit Journal of Computer Science*, *1*(2), 52–59. https://doi.org/10.56211/tsabit31

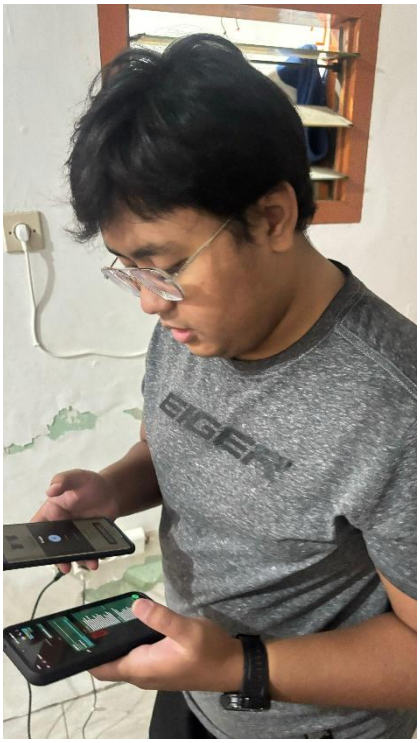
Timanta Tarigan, J., Fajar, R., & Sharif, A. (2019). Comparing the Performance of Reverse Colussi and *Raita* in Finding Indonesian Text. *Journal of Physics: Conference Series*, *1235*(1), 1–6. https://doi.org/10.1088/1742-6596/1235/1/012092

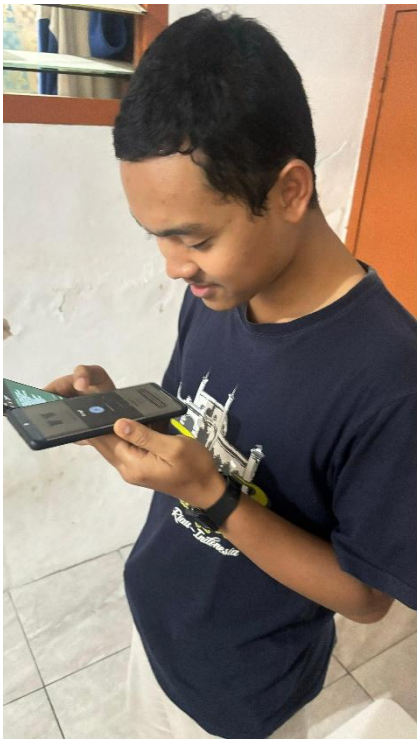
Wisnu, A. (2024). *Sistem Kendali Pintu Rumah Berbasis Internet of Things Menggunakan Algoritma Rabin-Karb* [Universitas Islam Negeri Maulana Malik Ibrahim Malang]. http://etheses.uin-malang.ac.id/65986/

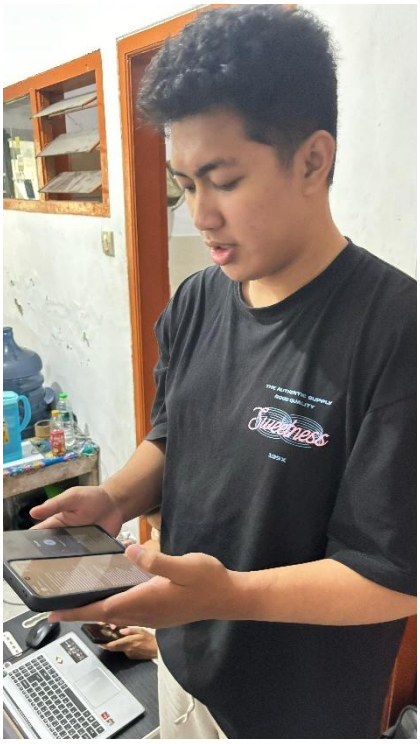
LAMPIRAN

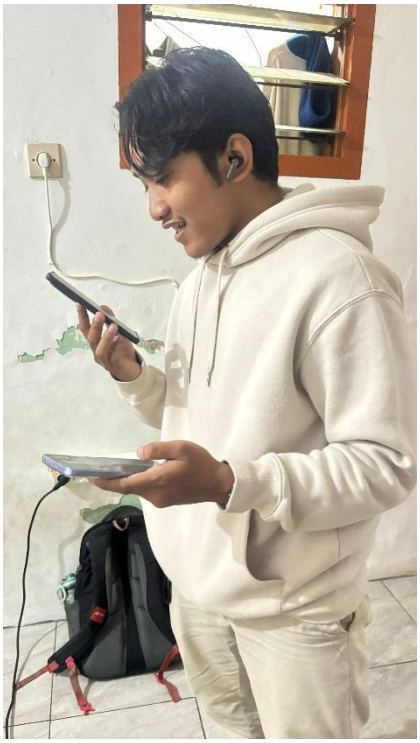
Lampiran 1. Dokumentasi Pengujian *Error* Sistem











Lampiran 2. Video Demonstrasi Sistem Jemuran Pintar

LINK GOOGLE DRIVE:

https://tinyurl.com/DemoSistemJemuranPintar