SISTEM KONTROL KUNCI PINTU RUMAH BERBASIS *INTERNET OF THINGS* DENGAN PERINTAH SUARA MENGGUNAKAN *NAIVE PATTERN SEARCHING ALGORITHM*

SKRIPSI

Oleh :
DAURIN NABILATUL MUNNA
NIM. 210605110025

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2024

SISTEM KONTROL KUNCI PINTU RUMAH BERBASIS *INTERNET OF THINGS* DENGAN PERINTAH SUARA MENGGUNAKAN *NAIVE PATTERN SEARCHING ALGORITHM*

SKRIPSI

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

Oleh :
DAURIN NABILATUL MUNNA
NIM. 210605110025

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2024

HALAMAN PERSETUJUAN

SISTEM KONTROL KUNCI PINTU RUMAH BERBASIS *INTERNET OF THINGS* DENGAN PERINTAH SUARA MENGGUNAKAN *NAIVE PATTERN SEARCHING ALGORITHM*

SKRIPSI

Oleh :
DAURIN NABILATUL MUNNA
NIM. 210605110025

Telah Diperiksa dan Disetujui untuk Diuji:
Tanggal: 05 Desember 2024

Pembimbing I,	Pembimbing II,
Ajib Hanani, M.T	Fatchurrohman, M.Kom
NIP. 19840731 202321 1 013	NIP. 19700731 200501 1 002

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

Fachrul Kurniawan, M.MT., IPU
NIP. 19771020 200912 1 001

HALAMAN PENGESAHAN

SISTEM KONTROL KUNCI PINTU RUMAH BERBASIS INTERNET OF THINGS DENGAN PERINTAH SUARA MENGGUNAKAN NAÏVE PATTERN SEARCHING ALGORITHM

SKRIPSI

Oleh :
DAURIN NABILATUL MUNNA
NIM. 210605110025

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Tanggal: 12 Desember 2024

Susunan Dewan Penguji

Ketua Penguji	: Johan Ericka Wahyu Prakasa, M.Kom NIP. 19831213 201903 1 004	( )
Anggota Penguji I	: Supriyono, M. Kom NIP. 19841010 201903 1 012	( )
Anggota Penguji II	: Ajib Hanani, M.T NIP. 19840731 202321 1 013	( )
Anggota Penguji III	: Fatchurrohman, M.Kom NIP. 19700731 200501 1 002	( )

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

Dr. Fachrul Kurniawan, M.MT., IPU
NIP. 19771020 200912 1 001

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama	: Daurin Nabilatul Munna
NIM	: 210605110025
Fakultas / Program Studi	: Sains dan Teknologi / Teknik Informatika
Judul Skripsi	: Sistem Kontrol Kunci Pintu Rumah Berbasis *Internet Of Things* dengan Perintah Suara Menggunakan *Naive Pattern Searching Algorithm*

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, 12 Desember 2024
Yang membuat pernyataan,

Daurin Nabilatul Munna
NIM. 210605110025

MOTTO

فَبِأَيِّ ءَالَآءِ رَبِّكُمَا تُكَذِّبَانِ

“so which of the favors of your lord would you deny?”

“Maka nikmat Tuhan kamu yang manakah yang kamu dustakan?”

HALAMAN PERSEMBAHAN

Alhamdulillahirobil Alamin puji syukur atas kehadirat Allah Subhanahu wa ta'ala yang telah melimpahkan segala kasih sayang, rahmat dan petunjuk-Nya, karenanya penulis menyelesaikan skiripsi ini dengan lancar dan baik. Shalawat serta salam senantiasa tercurahkan kepada sang Amirul Mu'minin yakni Rasulullah Shallallahu 'alaihi wasallam, yang telah menata kehidupan umat manusia dari zaman kegelapan hingga zaman yang penuh dengan *Addinul Islam.* Tugas akhir skiripsi ini terselesaikan dengan baik karena adanya kontribusi dan dukungan dari banyak pihak. Halaman persembahasan ini ditulis sebagai bentuk syukur dan terimakasih penulis kepada semua pihak yang telah berkontribusi dan mendukung dalam penelitian ini.

Kepada Ibunda penulis Ani Setiyawati, Ayah kandung penulis Abi Kosim, Ayah sambung penulis Agus Fitriawan yang senantiasa memberikan cinta, kasih sayang, usaha, dukungan, semangat, serta do'a yang tak terhingga sehingga penulis dapat menyelesaikan perkuliahan dan skripsi ini dengan baik. Kemudian, kepada kedua adik penulis yaitu Natasya Nuril Anggraini dan Yusril Fitria Az-zahra yang sangat penulis cintai dan terus memberikan semangat, motivasi, serta tujuan agar penulis dapat sukses sehingga akhirnya skiripsi ini dapat tersusun dengan baik.

Kepada Bapak Ajib Hanani, M.T selaku Dosen Pembimbing I dan Bapak Fatchurrohman, M.Kom selaku Dosen Pembimbing II yang selalu membimbing, memberikan arahan, memberikan motivasi, memberikan masukan, memberikan semangat, dan membantu penulis dalam menyelesaikan skripsi ini. Kepada Bapak Johan Ericka Wahyu Prakasa, M.Kom selaku Dosen Penguji I dan Bapak

Supriyono, M. Kom selaku Dosen Penguji II yang senantiasa memberikan arahan, masukan, dan perbaikan untuk menyempurnakan skiripsi ini serta kepada seluruh Dosen dan Staff Program Studi Teknik Informatika atas ilmu, pengalaman, dan kesempatan yang diberikan selama masa studi penulis dalam bangku perkuliahan dan selama proses penulisan skiripsi ini.

Kepada seluruh saudara dan keluarga seperjuangan Teknik Informatika Angkatan 2021 ASTER yang telah menemani, memberikan dukungan, semangat, motivasi, dan do'a kepada penulis. Dan kepada seluaruh keluarga, teman, sahabat, serta kerabat yang tidak bisa disebutkan satu persatu. Terimakasih banyak atas segala hal baik yang pernah diberikan baik secara sikap, ucapan, semangat, bantuan, dan do'anya.

KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabarakatuh

Alhamdulillah segala puji dan Syukur senantiasa penulis panjatkan pada Allah subhanahu wa ta'ala atas berkat Rahmat, serta hidayah-Nya, sehingga penulis dapat menyelesaikan skripsi ini dengan baik. Sholawat serta salam tetap tercurahkan kepada Nabi Muhammad SAW. Dan semoga kita semua mendapat syafaatnya di hari akhir kelak, Aamiin.

Penulis mengucapkan rasa terima kasih yang begitu besar kepada seluruh pihak yang memberikan dukungan dan motivasi kepada penulis sehingga dapat menyelesaikan skripsi ini. Ucapan terima kasih ini penulis disampaikan kepada:

1. Prof. Dr. M. Zainuddin, M.A., selaku rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang.
2. Prof. Dr. Hj. Sri Harini, M.Si., selaku dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
3. Dr. Ir. Fachrul Kurniawan, M.MT., IPU selaku Ketua Program Studi Teknik Informatika Universitas Islam Negeri Maulana Malik Ibrahim Malang.
4. Ajib Hanani, M.T selaku pembimbing utama yang dengan penuh kesabaran dan ketulusan hati memberikan bimbingan, arahan, serta dorongan dalam setiap tahap penyusunan skripsi ini.
5. Fatchurrohman, M.Kom selaku dosen wali sekaligus pembimbing kedua penulis yang selalu memberikan masukan selama perkuliahan hingga terselesaikannya skripsi ini.

6. Johan Ericka Wahyu Prakasa, M.Kom selaku Dosen Penguji pertama dan Bapak Supriyono, M. Kom selaku penguji kedua yang telah berkenan menguji serta memberikan masukan yang menjadikan sehingga skripsi ini dapat terselesaikan dengan baik.
7. Nia Faricha S, Si selaku admin Program Studi Teknik Informatika yang selalu sabar memberikan informasi, membantu, dan memberikan arahan selama perkuliahan dan proses penulisan skripsi ini.
8. Segenap dosen, laboran, dan jajaran staff Program Studi Teknik Informatika yang telah memberikan ilmu, pengetahuan, dan dukungan selama penulis menjalani studi.
9. Kepada orang tua tercinta, Ibunda Ani Setiyawati, Ayah kandung Abi Kosim, serta Ayah sambung Agus Fitriawan, yang selalu menjadi sumber kekuatan dan inspirasi bagi penulis. Terima kasih atas segala usaha dan doa terbaik yang telah diberikan. Semoga Allah senantiasa melimpahkan kesehatan dan perlindungan, sehingga dapat selalu hadir dalam setiap langkah dan pencapaian penulis.
10. Kedua saudara yang sangat penulis sayangi yaitu Natasya Nuril Anggraini dan Yusril Fitria Az-zahra beserta seluruh keluarga besar yang tiada henti memberikan do'a dan dukungan sehingga penulis mampu menyelesaikan skripsi ini.
11. Kepada sahabat penulis Silvy Anggraeni Cahya Putri, Alif Khunaifah Az-zahro, Ayah Heru Joko Purwanto, dan Ibu Retno yang senantiasa memberikan rumah dan kasih sayang kepada penulis sehingga penulis

tidak pernah kehilangan rumah untuk pulang khususnya di kota Malang ini.

12. Kepada sahabat penulis semenjak kecil, Afif Syihabuddin Rohim, yang selalu hadir di saat dibutuhkan dan menjadi sumber inspirasi bagi penulis, terutama melalui cerita unik "cewek ketan" yang menghibur di saat kebingungan.
13. Teman seperjuangan penulis Imamatul Khoiriyah, Nurul Izzah, Nova Rahma Yunida Putri, Muizzul Azizah Oktavianing Putri, Aldiana Damayanti, dan Agustina Mufidatuzzainiya yang telah menemani, memberikan semangat serta segala bantuan dari awal perkuliahan hingga saat ini.
14. Teman sebimbingan yaitu Heny Rimadana, Rafi Rabani, dan Danendra Farrel. Teman yang pernah tinggal seatap dengan penulis Nurhaliza, Nadhifa, Ninis, Eka, Tina, Dira, Naomi, Aisyah, Nenden Nuraeini, Intan Tiara Dewi, Hamidah Lutfiyanti, Adila Qurrota A'yun, Annisa Fitri Madani, Raniah Oktariza Imani, dan Lailatul Habibah. Teman-teman yang telah berkontribusi banyak dalam perjalanan perkuliahan penulis, Aulia Ilmi Magfira, Putri Purnamasari, Dilla Sofiana, Enggarani Wahyu, dan teman teman yang tidak bisa disebutkan satu persatu kebaikannya.
15. Seluruh warga Teknik Informatika khususnya angkatan 2021 "Aster" yang telah memberikan kehangatan, motivasi, dan dukungan kepada penulis.
16. Teman-teman Mujaahidah khusus nya yang berada di Kota Malang Qonita Afifah, Aisyah Amalia, Ismi , Aditiya Regi, Annisa Rohma, Selvia Nurul,

Fakhira Zahrany, Ayu Dwi, Mawar Nur, Shofia Fardani dan seluruhnya yang tidak bisa disebutkan. Terimakasih atas pengusahaan waktu yang tak pernah mengenal sibuk.

17. Teman-teman Himatif Encoder, Divisi Public Relation ’22 ’23, Komunitas yang berada di Teknik Informatika khusunta ETH-0, dan Volunteer Senyum Anak Nusantara ‘22 yang telah menjadi tempat penulis untuk mengembangkan *skill* selama masa perkuliahan.
18. Seluruh keluarga Kantor Komisi Pemilihan Umum Kota Batu yang sudah berkenan berbagi ilmu dan pengalaman selama PKL.
19. Teman-teman KKM 153 Desa Pagedangan Kecamatan Turen tahun 2023, yang telah memberikan pengalaman dan kenangan yang tak terlupakan.
20. Seluruh pihak yang telah terlibat, baik secara langsung maupun tidak langsung dari awal perkuliahan hingga akhir penulisan skripsi ini.

Penulis berharap semoga skripsi ini dapat memberikan manfaat untuk kedepannya. Penulis menyadari bahwa skripsi ini masih. Oleh karena itu, menerima masukan yang dapat membantu pengembangan lebih lanjut.

Wassalamu’alaikum warahmatullahi wabarakatuh.

Malang, 10 Desember 2024

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Munna, Daurin Nabilatul. 2024. **Sistem Kontrol Kunci Pintu Rumah Berbasis Internet Of Things dengan Perintah Suara Menggunakan Naive Pattern Searching Algorithm.** Skripsi. Program Studi Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Ajib Hanani M.T (II) Fatchurrohman, M.Kom.

.

Kata Kunci: Disabilitas Tuna Netra, Internet Of Things, Kendali Kunci Pintu, Naive Pattern Searching, Perintah Suara

Dalam era globalisasi seperti saat ini, pengembangan sistem berbasis Internet of Things dapat memberikan kenyamanan untuk penyandang disabilitas tuna netra. Penelitian ini akan merancang sistem kontrol kunci pintu rumah berbasis IoT dengan perintah suara dan pengenalan perintah akan dihitung menggunakan Naïve Pattern Searching Algoritm. Sistem ini dirancang dengan tujuan untuk membantu penyandang disabilitas tuna netra dalam mengoperasikan pintu rumah tanpa interaksi fisik dengan pegangan atau kunci pintu. Penelitian ini menggunakan salah satu Algoritma Natural Language Processing yaitu, Naïve Pattern Searching Algoritm untuk mencocokan perintah dengan pattern yang telah ditentukan sebelumnya. Dengan sistem ini, perintah yang akan diproses untuk mengoperasikan hardware atau kunci pintu sehingga dapat membuka dan mengunci pintu secara otomatis. Hasil pengujian menunjukkan tingkat keberhasilan sistem dalam mengenali 643 kalimat perintah atau sebesar 91,86% dari 700 pengujian, dengan tingkat 57 error atau sebesar 8,14%. Penelitian ini membuktikan bahwasanya sistem yang dirancang berhasil dan efektif dalam membantu kenyamanan untuk para penyandang disabilitas tuna netra.

ABSTRACT

Munna, Daurin Nabilatul. 2024. **The Internet-of-Things-Based House Door Control System Using Voice Command of Naive Pattern Searching Algorithm.** Thesis. Informatics Engineriing Faculty of Science and Technology Universitas Islam Negeri Maulana Malik Ibrahim Malang. Advisor: (I) Ajib Hanani M.T (II) Fatchurrohman, M.Kom.

Key words: Door Lock Control, Internet of Things, Naive Pattern Searching, Visual impairment, Voice Command

In the globalization era, the development of Internet-of-Things-Based systems can provide comfort for people with visual impairment. The research will design an IoT-Based House Door Control System Using Voice Command and order recognition counted using the Naive Pattern Searching Algorithm. The system aims to help people with visual impairment operate the doors in their houses without physical interaction with the handle. The research used one of the Natural Language Processing Algorithms, the Naïve Pattern Searching Algorithm, to match order and the set pattern. The system will automatically process an order to operate the hardware or door handle open automatically. The test result shows that the system success level in recognizing 643 commands is 91.86% out of 700 tests, with 57 error levels of 8.14%. The research proves that the designed system is successful and effective in providing comfort for people with visual impairment.

الملخص

المنى، دور نبيلة. 2024. نظام التحكم في قفل الباب المنزلي على أساس إنترنت الأشياء مع الأوامر الصوتية باستخدام خوارزمية البحث عن الأنماط الساذجة. البحث الجامعي. قسم الهندسة المعلوماتية، كلية العلوم والتكنولوجيا بجامعة مولانا مالك إبراهيم الإسلامية الحكومية مالانج. المشرف الأول: عجيب حناني، الماجستير. المشرف الثاني: فتح الرحمن، الماجستير.

.**الكلمات الرئيسية: ذوي إعاقة، كفيف، إنترنت أشياء، تحكم في قفل باب، بحث عن أنماط ساذجة، أوامر صوتية**

في عصر العولمة كما هو الحال اليوم، يمكن لتطوير الأنظمة على أساس إنترنت الأشياء توفير الراحة للأشخاص ذوي الإعاقة أو المكفوفين. سيقوم هذا البحث بتصميم نظام التحكم في قفل الباب المنزلي على أساس إنترنت الأشياء مع الأوامر الصوتية وسيتم حساب التعرف على الأوامر باستخدام خوارزمية البحث عن الأنماط الساذجة. تم تصميم هذا النظام بهدف مساعدة الأشخاص ذوي الإعاقة أو المكفوفين في تشغيل باب المنزل دون تفاعل جسدي مع مقبض الباب أو القفل. استخدم هذا البحث إحدى خوارزميات معالجة اللغة الطبيعية، وهي خوارزمية البحث عن الأنماط الساذجة لمطابقة الأوامر مع الأنماط المحددة مسبقا. باستخدام هذا النظام، ستتم معالجة الأوامر لتشغيل الجهاز أو قفل الباب بحيث يمكن فتح الباب وقفله تلقائيا. أظهرت نتائج الاختبار نسبة نجاح النظام في التعرف على 643 جملة أمر أو 91.86٪ من 700 اختبار، بمعدل 57 خطأ أو 8.14٪. أثبت هذا البحث أن النظام المصمم ناجح وفعال في المساعدة على راحة الأشخاص ذوي الإعاقة أو المكفوفين .

BAB I

PENDAHULUAN

1.1 Latar Belakang

Pada era globalisasi seperti saat ini, teknologi berkembang sangat pesat. Hal ini menyebabkan banyaknya perubahan dalam kehidupan sehari hari, seperti cara bekerja yang harus berdampingan dengan komputer dan gadget, cara berkomunikasi yang dapat dengan mudahnya diakses dari jarak yang jauh menggunakan smartphone, dan cara mendapatkan informasi dengan cepat yang diakses melalui internet. Pada saat ini teknologi juga menjadi sesuatu yang penting dalam berbagai bidang, mulai dari bisnis, pendidikan, pemerintahan, hingga kesehatan. Perkembangan teknologi ini tentunya menjadikan pekerjaan manusia lebih mudah dan cepat (Fitriani, 2023). Perkembangan teknologi yang semakin pesat ini tak lepas dari ridho Allah SWT dalam firmannya yang mengajarkan pentingnya mengenai ilmu pengetahuan dan kemajuan melalui usaha manusia. Dalam surah An-Nahl ayat 78 yang berbunyi:

وَاللّٰهُ اَخْرَجَكُمْ مِّنْۢ بُطُوْنِ اُمَّهٰتِكُمْ لَا تَعْلَمُوْنَ شَيْـًٔاۙ وَّجَعَلَ لَكُمُ السَّمْعَ وَالْاَبْصَارَ وَالْاَفْـِٕدَةَۙ لَعَلَّكُمْ تَشْكُرُوْنَ ﴿٧٨﴾

"Allah mengeluarkan kamu dari perut ibumu dalam keadaan tidak mengetahui sesuatu pun dan Dia menjadikan bagi kamu pendengaran, penglihatan, dan hati nurani agar kamu bersyukur." (QS: An-Nahl: 78)

Ayat ini mengingatkan bahwa segala bentuk pengetahuan dan kemampuan manusia, termasuk dalam menciptakan, mengembangkan dan merasakan teknologi, berasal dari karunia Allah. Hal ini sudah seharusnya membuat manusia untuk

bersyukur atas kemajuan teknologi yang telah memudahkan kehidupan (Amarodin, 2021).

Pada bidang medis *Internet of Things* seringkali digunakan untuk membantu berbagai penyakit, alat bantu penyembuhan, dan alat bantu bagi disabilitas (Hanani & Hariyadi, 2020). Salah satu penyakit atau kelainan yang menjadi perhatian adalah Tuna Netra. Tuna Netra sendiri merupakan sebutan untuk seseorang yang memiliki gangguan terhadap indra penglihatan baik secara total ataupun kurangnya penglihatan (Antonius & Pramida, 2022). Menteri Kesehatan RI pada pembukaan *press Asia-Pacific Academy of Ophthalmology (APAO)* ke-39 di Bali menyatakan bahwasanya Organisasi Kesehetan Dunia atau *World Health Organization (WHO)* menyebutkan terdapat sektar 2,2 miliar orang di seluruh dunia mengalami gangguan penglihatan. Dari jumlah tersebut, 1,1 milyar dengan presentasi 50 persen diantaranya mengalami disabilitas penglihatan total atau buta. Negara Indonesia sendiri menduduki peringkat ketiga dunia setelah India dan China (Tarmizi, 2024).

Bedasarkan data yang dikeluarkan oleh Kementerian Kesehatan Republik Indonesia. Jumlah penyandang disabilitas penglihatan atau tunanetra mencapai angka 1,5 persen dari keseluruhan jumlah penduduk Indonesia. Pertahun 2024 jumlah penduduk di Indonesia mencapai lebih dari 280 jiwa. Maka, dapat dikatakan penyandang disabilitas penglihatan atau tunanetra di Indonesia lebih dari 4 juta jiwa (Imran, 2024). Tingginya angka kebutaan atau tunanetra di Indonesia mendorong terbentuknya berbagai yayasan dan komunitas sosial yang bertujuan untuk membantu para penyandang disabilitas tunanetra. Angka ini juga menunjukkan pentingnya pengunaan teknologi sebagai alat bantu yang sesuai bagi para

penyandang Tuna Netra karena dengan bantuan teknologi dapat menjadikan para penyandang Tuna Netra melakukan aktivitas sehari-hari dengan minim bantuan orang sekitar.

Dalam Upaya membantu dan melindungi sesama, diperlukan teknologi untuk memebrikan kenyamanan para penyandang tuna netra. Membatu sesama merupakan sesuatu hal yang diperintahkan oleh Allah SWT sebagaimana telah difirmakankan dalam Surat Al-Ma'idah Ayat 2 yang berbunyi:

يٰٓاَيُّهَا الَّذِيْنَ اٰمَنُوْا لَا تُحِلُّوْا شَعَاۤىِٕرَ اللّٰهِ وَلَا الشَّهْرَ الْحَرَامَ وَلَا الْهَدْيَ وَلَا الْقَلَاۤىِٕدَ وَلَآ اٰۤمِّيْنَ الْبَيْتَ الْحَرَامَ يَبْتَغُوْنَ فَضْلًا مِّنْ رَّبِّهِمْ وَرِضْوَانًاۗ وَاِذَا حَلَلْتُمْ فَاصْطَادُوْاۗ وَلَا يَجْرِمَنَّكُمْ شَنَاٰنُ قَوْمٍ اَنْ صَدُّوْكُمْ عَنِ الْمَسْجِدِ الْحَرَامِ اَنْ تَعْتَدُوْاۘ وَتَعَاوَنُوْا عَلَى الْبِرِّ وَالتَّقْوٰىۖ وَلَا تَعَاوَنُوْا عَلَى الْاِثْمِ وَالْعُدْوَانِۖ وَاتَّقُوا اللّٰهَ اِنَّ اللّٰهَ شَدِيْدُ الْعِقَابِ ۝٢

“Wahai orang-orang yang beriman, janganlah kamu melanggar syiar-syiar (kesucian) Allah, jangan (melanggar kehormatan) bulan-bulan haram, jangan (mengganggu) hadyu (hewan-hewan kurban) dan qalā’id (hewan-hewan kurban yang diberi tanda), dan jangan (pula mengganggu) para pengunjung Baitulharam sedangkan mereka mencari karunia dan rida Tuhannya! Apabila kamu telah bertahalul (menyelesaikan ihram), berburulah (jika mau). Janganlah sekali-kali kebencian(-mu) kepada suatu kaum, karena mereka menghalang-halangimu dari Masjidilharam, mendorongmu berbuat melampaui batas (kepada mereka). Tolong-menolonglah kamu dalam (mengerjakan) kebajikan dan takwa, dan jangan tolong-menolong dalam berbuat dosa dan permusuhan. Bertakwalah kepada Allah, sesungguhnya Allah sangat berat siksaan-Nya.” (QS: An-Ma’idah ayat 2).

Menurut tafsir Al Maragi ayat ini menerangkan bahwasanya perintah tolong menolong terhadap kebijakan dan takwa dalam ayat ini merupakan suatu bagian dari rukun hidayah yang terkumpul didalam Al-Qur’an (Sintanu, 2023). Rasulullah SAW juga bersabda dalam sebuah hadist shahih yang diriwayatkan oleh Imam Bukhari No. 7376 pada kitab Fathul Bari mengenai pentingnya memberikan kasih sayang dan perhatian terhadap sesama manusia cipataan Allah SWT (Mumtaz dkk., 2022). Berikut merupakan hadis tersebut:

حَدَّثَنَا مُحَمَّدٌ إِبْنُ سَلَامٍ ، أَخْبَرَنَا أَبُو مُعَاوِيَةَ، عَنِ الأَعْمَشِ، عَنْ زَيْدِ بْنِ وَهْبٍ، وَأَبِي، ظَبْيَانَ عَنْ جَرِيرِ بْنِ عَبْدِ اللَّهِ، قَالَ : قَالَ رَسُولُ اللَّهِ صلى الله عليه وسلم " لاَ يَرْحَمُ اللَّهُ مَنْ لاَ يَرْحَمُ النَّاسَ ".

"Telah menceritakan kepada kami Muhammad bin Salam telah menceritakan kepada kami Abu Mu'awiyah dari Al A'masy dari Zaid bin Wahb dan Abu dlabyan dari Jarir bin Abdullah berkata, "Rasulullah shallallahu 'alaihi wasallam bersabda: 'Allah tak bakalan menyayangi siapa saja yang tidak menyayangi manusia.'" (H.R. Bukhari)

Salah satu Solusi dari permasalah ini adalah dengan merancang dan membangun sistem kontrol rumah yang menggunakan perintah suara. Penelitian ini berfokus pada perancangan sistem kontrol kunci pintu rumah berbasis perintah suara, dengan mengangkat permasalahan yang dihadapi oleh penyandang tunanetra. Sistem ini dirancang untuk mengoperasikan pintu rumah secara otomatis hanya dengan menggunakan perintah suara, sehingga mengurangi interaksi secara langsung seperti memegang gagang dan kunci pintu. Hal ini bertujuan untuk mengurangi risiko bagi para penyandang tunanetra, yang mengalami kesulitan dalam mengunci pintu. Sistem ini dapat memberikan kenyamanan dengan memastikan pintu dapat dikunci atau dibuka hanya melalui perintah suara.

Sistem ini dirancang untuk memberikan kemudahan bagi penggunanya, khususnya bagi pengguna yang memiliki keterbatasan fisik atau kebutuhan khusus, seperti penyandang disabilitas tuna netra. Dengan menggunakan teknologi *Internet of Thing (IoT),* sistem dapat memahami dan merespons perintah suara dari pengguna akurat. Ketika pengguna memberikan instruksi seperti "kunci pintu" atau "buka pintu", sistem ini akan memproses perintah tersebut menggunakan *Naive Pattern Searching Algorithm*. Setelah perintah dikenali, sistem ini akan mengaktifkan aktuator yang digunakan mengunci atau membuka kunci pintu. Hal

ini dilakukan secara otomatis tanpa memerlukan kontak fisik dengan gagang dan kunci pintu, sehingga meningkatkan kemudahan dan kenyamanan pengguna.

Penerapan *Naive Pattern Searching Algorithm* dalam penelitian ini bertujuan untuk mengukur performa dalam sistem kendali kunci pintu. *Naive Pattern Searching Algoritm* dipilih setelah beberapa evalusasi dari penelitian sebelumnya seperti pada penelitian yang dilakukan oleh (Retkoceri dkk., 2022) menunjukan bahwa algoritma *Boyer-Moore* mengalami penurunan efisiensi pada pola yang pendek, karena performa algoritma ini bergantung pada heuristik yang diterapkan. Sementara itu, algoritma *Knuth-Morris-Pratt (KMP)* menggunakan informasi dari pencocokan sebelumnya untuk meningkatkan efisiensi, namun implementasinya lebih kompleks, terutama saat membangun tabel lompatan. Dan, algoritma *Rabin-Karp* menggunakan teknik hashing untuk mencari pola dalam teks dengan waktu rata-rata yang efisien, meskipun kinerjanya bisa terpengaruh oleh konflik hash dan memerlukan fungsi hash yang baik untuk efisiensi. Algoritma yang kompleks tidak selalu menghasilkan hasil yang akurat. Oleh karena itu, penelitian ini mengajukan penggunaan algoritma *Naive Pattern Searching* karena algoritma ini memiliki kelebihan, yaitu mudah diimplementasikan, serta tidak memerlukan struktur data tambahan yang kompleks.

1.2 Pernyataan Masalah

Adapun pernyataan masalah yang diangkat dalam penelitian ini adalah mengukur bagaimana tingkat performa *Naive Pattern Searching Algorithm* dalam pengenalan perintah sistem kendali kunci pintu rumah ?

1.3 Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah mengukur performa *Naive Pattern Searching Algorithm* dalam pengenalan perintah sistem kendali kunci pintu rumah.

1.4 Batasan Masalah

Bedasarkan latar belakang yang telah dijabarkan sebelumnya. Adapun batasan masalah dalam penelitian ini adalah sebagai berikut:

1. Penggunaan *Device* yang digunakan dalam sistem kendali merupakan *Smartphone.*
2. Sistem Kendali hanya menerima input suara dalam Bahasa Indonesia.

1.5 Manfaat Penelitian

Bedasarkan latar belakang yang telah dijabarkan sebelumnya. Adapaun manfaat dari penelitian ini adalah sebegai berikut :

1. Penelitian ini dapat membantu para penyandang tuna netra, untuk mengendalikan kunci pintu tanpa perlu interaksi fisik.
2. Sistem ini meningkatkan kemudahan dan kenyamanan pada penyandang disabilitas tuna netra dalam mengunci pintu rumah.
3. Penelitian ini akan memberikan evaluasi pada performa *Naive Pattern Searching Algorithm*.

1.6 Sistematika Penulisan

Adapun sistematika penulisan yang diterapkan dalam penelitian ini adalah sebagai berikut:

1. BAB I: PENDAHULUAN

Berisikan poin - poin pendahuluan penelitian, latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penulisan.

2. BAB II: STUDI PUSTAKA

Berisikan tinjauan teori yang mendukung penelitian yaitu, definisi dan pandangan dari para ahli terkait teori-teori yang dibutuhkan dalam penelitian.

3. BAB III: DESAIN DAN IMPLEMENTASI

Berisikan kerangka pemikiran penelitian dan berbagai desain sistem yang diharapkan dalam penelitian.

4. BAB IV: UJI COBA DAN PEMBAHASAN

Berisi hasil penelitian yang telah dilakukan serta pembahasan mengenai temuan-temuan dari uji coba dalam penelitian.

5. BAB V: KESIMPULAN DAN SARAN

Berisi kesimpulan yang diperoleh dari penelitian serta saran-saran untuk penelitian selanjutnya.

BAB II

STUDI PUSTAKA

2.1 Penelitian Terkait

Naive Pattern Searching Algorithm adalah metode pencarian pola yang berfungsi untuk menemukan kemunculan pola tertentu dalam teks. Algoritma ini bekerja dengan cara membandingkan pola yang dicari dengan *substrings* dari teks, dimulai dari posisi pertama dan berlanjut ke posisi-posisi berikutnya. Setiap kali pola dibandingkan dengan *substrings*, algoritma akan memeriksa kesesuaian karakter demi karakter hingga ditemukan kecocokan pada pola. Algoritma ini memiliki kompleksitas O(m*n), di mana m merupakan panjang pola dan n adalah panjang teks, algoritma ini memiliki keunggulan yaitu mudah diimplementasikan dan berguna dalam aplikasi yang sederhana atau ketika pola yang dicari pendek dibandingkan dengan teks. Dalam penelitian yang berjudul *"Analysis of Pattern Searching Algorithms and Their Application"* membahas tentang penggunaan algoritma *Naive Pattern Searching Algoritm* dalam pemrograman dan pengolahan teks, menunjukkan bahwa meskipun ada algoritma pencarian pola yang lebih efisien (Retkoceri dkk., 2022), *Naive Pattern Searching* tetap relevan dalam situasi di mana kemudahan implementasi lebih penting daripada efisiensi waktu.

Pada penelitan yang dilakukan oleh (Padmaveni & John Aravindhar, 2021) memperkenalkan sebuah algoritma dalam pencocokan string dinamakan *Skip Search* (SS). Algoritma yang diusulkan dalam penelitian ini merupakan hasil evaluasi dari dari algoritma *Naive Searching Algoritm* yang mengakses hanya

setengah karakter dalam string dan memeriksa apakah karakter-karakter tersebut adalah karakter awal atau akhir dari pola. Dengan Algoritma *Skrip Search* yang diusulkan dalam penelitian ini dapat ditentukan panjang lompatan pola dan mengurangi waktu pencarian. Karena tidak memerlukan fase praproses atau ruang memori tambahan, algoritma ini unggul dalam hal kompleksitas ruang. Waktu eksekusi dibandingkan dengan algoritma Knuth-Morris-Pratt (KMP), dan algoritma skip menunjukkan kinerja yang lebih baik dalam sebagian besar kasus uji. Algoritma ini efektif ketika pola terletak di akhir teks atau jika pola tidak ada dalam teks, dan dapat direkomendasikan untuk proyek pencocokan pola.

Banyak algoritma yang dapat dilakukan untuk mengukur kecepatan pola seperti penelitian yang dilakukan oleh (Retkoceri dkk., 2022) yang mana membandingkan kecepatan berbagai algoritma pencarian pola, yaitu *Naive, KMP, Rabin-Karp, Finite Automata, Boyer-Moore, Aho-Corasick, dan Z Algorithm*, dengan menguji kompleksitas waktu mereka menggunakan tiga bahasa pemrograman (C#, Java, dan Python) serta tiga CPU yang berbeda. Penelitian ini mengevaluasi perbandingan kinerja antara bahasa pemrograman dan CPU yang digunakan.algoritma yang diimplementasikan dalam bahasa Java cenderung lebih cepat pada semua ukuran teks, terutama pada CPU i7-2620m. Namun, untuk algoritma tertentu seperti Boyer-Moore Good Suffix dan Algoritma Z, CPU i5-6200U dan AMD A9-9410 menunjukkan performa yang lebih baik dibandingkan CPU lainnya. Hasil penelitian ini menunjukan bahwasanya *Naive Pattern Searching Algorithm* memiliki kinerja yang konsisten dalam uji coba.

String Matching dengan kompleksitas O(m*n) pada penelitian yang dilakukan oleh (Wirawan & Paryatna, 2020) menunjukkan bahwa fitur koreksi menggunakan *String Matching Method* meningkatkan akurasi terjemahan, dan respons mahasiswa terhadap aplikasi ini dalam proses pembelajaran dinilai sangat positif, mendukung pengembangan ilmu pengetahuan, teknologi, dan budaya sosial. Hasil dari penelitian ini menunjukan bahwa metode *String Matching* dengan kompleksitas O(m*n) berhasil memberikan hasil yang akurat dalam menemukan kata yang tepat bedasarkan kata kunci yang digunakan pada aplikasi penerjemah bahasa Bali (Anggah Ungguhing).

Pada penelitian yang dilakukan oleh (Abiodun dkk., 2019) membahas tentang aplikasi *Artificial Neural Network* (ANN) dalam pengenalan pola atau *Pattern Recognition (PR)*. Penelitian ini mengidentifikasi berbagai model ANN yang telah menunjukkan hasil yang baik dalam menjalankan *Pattern Recognition*. Hasil dari penelitian ini memnunjukkan bahwa berbagai model ANN (seperti GAN, SAE, DBN, RBM, RNN, CNN, PNN, dan lainnya) berhasil dengan sangat baik. Mayoritas dari model ANN ini menunjukkan tingkat kesalahan yang rendah dan keakuratan yang tinggi dalam pengenalan pola. Pengujian model *String Matching* dilakukan dengan menggunakan idikator statistik seperti MAPE (Mean Absolute Percentage Error), MAE (Mean Absolute Error), RMSE (Root Mean Squared Error), dan VAPE (Variance of Absolute Percentage Error).

Sistem kendali yang berbasis *IoT* telah banyak digunakan untuk berbagai bidang seperti pada penelitian yang dilakukan oleh (Karamizadeh dkk., 2024) mengenai sistem kontrol berbasis *IoT* dengan deteksi wajah yang menggunakan

model *Deep Learning* (DL) yaitu *Convolutional Neural Networks* (CNN) dan mekanisme perhatian diri (*self-attention*). Sistem ini dirancang untuk meningkatkan keamanan secara signifikan tanpa bergantung pada aturan manual dan mengatasi tantangan dalam deteksi wajah di lingkungan yang tidak terkontrol. Hasil eksperimen menunjukkan akurasi tinggi sebesar 99,7%, menandakan keunggulan model ini dibandingkan model lain dalam hal kecepatan dan akurasi. Skala kompleksitas algoritma. Penelitian ini menggunakan perintah suara sebagai input dengan *Naive Pattern Seraching Algoritm* untuk menggolah perintah sedangkan pada penelitian yang dilakukan oleh Karamizadeh dkk., menggunakan deteksi wajah sebagai input dengan algortima *Convolutional Neural Networks* (CNN) untuk pengolahan akurasi.

Pengenalan perintah suara pada penelitian yang dilakukan oleh (Sasia dkk., 2020) menunjukkan rata-rata waktu untuk menghidupkan dan mematikan lampu berkisar antara 3,28 hingga 3,61 detik, pada intensitas suara minimal sekitar 60-61 dB, dan jangkauan kendali maksimal 1,2 meter. Penelitian ini menggunakan algoritma *Recurrent Neural Network* (RNN) dan *Long-Short-Term Memory Networks* (LSTM) untuk pemrosesan suara yang menggunakan *Recurrent Neural Network* (RNN) dan *Long-Short-Term Memory Networks* (LSTM) yang tersedia pada Google API dalam mengelola perintah dan objek yang digunakan adalah lampu.

Sedangkan, pada penelitian yang dilakukan oleh (Rizky dkk., 2023) dengan menggunakan algoritma *Hidden Markov Model* (HMM) dalam mengenali pola suara dan diimplentasikan dalam sistem kontrol pintu otomatis yang dioperasikan

melalui perintah suara. Alat ini dikembangkan menggunakan bahasa pemrograman C++, dan hasilnya menunjukkan bahwa sistem dapat membuka dan menutup pintu sesuai dengan perintah suara yang diberikan. Hasil dari penelitian menunjukan bahwa algoritma *Hidden Markov Model* (HMM) berhasil mengoperasikan sistem dengan cukup baik dalam mengelola perintah dan objek yang digunakan dalam penelitian adalah pintu.

Pada penelitian yang dilakukan oleh (Fantofani, 2020) dengan menggunakan algoritma Nazief & Adriani dan algoritma *Quick Search* mencatat rata-rata waktu yang diperluhkan 0,01565 detik dengan tingkat error sebesar 28% karena kesulitan mengenali pola kata yang disebabkan oleh kata pemisah dalam kalimat. Ini menunjukkan bahwa meskipun algoritma *Quick Search* cepat dalam hal waktu proses, ia tidak cukup akurat untuk aplikasi perintah suara yang memerlukan pengenalan yang sangat presisi. Penelitian yang dilakukan oleh Fantofani menggunakan algoritma Nazief & Adriani dan algoritma *Quick Search* dalam mengelola perintah dengan objek yang digunakan adalah lampu dan pengujiannya sebanyak 20 kalimat perintah.

Pencocokan pola yang dilakukan pada penelitian (Ulum, 2022) dengan menggunakan menggunakan algoritma *Knuth Morris Pratt* (KMP) untuk pencocokan pola dalam sistem kontrol gorden. Hasil penelitian menunjukkan bahwa algoritma KMP memberikan performa yang sangat baik dengan rata-rata durasi pencarian pola sebesar 513ms dan akurasi mencapai 100% berdasarkan pengujian menggunakan *confusion matrix* dan algoritma *Knuth Morris Pratt* (KMP) dan objek yang digunakan dalam penelitian adalah gorden.

Terakhir, pada penelitian yang dilakukan oleh (Rabbani, 2021) yang menggunakan menggunakan *microcontroller* ESP8266 dan *motor stepper* untuk menggerakkan pagar rumah. Algoritma *Boyer-Moore* digunakan *untuk pattern matching*. Dari pengujian terhadap 50 kalimat perintah dengan 4 pola yang berbeda, hasilnya menunjukkan akurasi 100% dengan waktu proses rata-rata sekitar 0,00672 ms. Adapun perbedaan antara penelitian yang dilakukan oleh Rabbani dengan penelitian ini terletak pada pemilihan algoritma, pemilihan objek, dan alur pengujian dengan menggunakan algoritma *Boyer-Moore* dalam mengelola perintah dengan objek yang digunakan adalah pagar rumah dan pengujiannya sebanyak 50 kalimat perintah.

Tabel 2.1 dibawah ini menyajikan penelitian terdahulu yang telah diuraikan pada paragraf sebelumnya:

Tabel 2. 1 Penelitian Terdahulu

No	Nama Peneliti	Judul Penelitian	Metode	Hasil Penelitian
1.	Padmaveni, K., & John Aravindhar, D. (2021)	*Improved skip algorithm for single pattern searching.*	*Naive Pattern Seraching Algoritm* dan *Skip Search*	Algoritma *Naive Pattern Seraching* digunakan sebagai dasar dari penggembangan metode baru yaitu *Skip Search.* Metode ini menunjukkan kinerja yang lebih baik dalam beberapa uji coba. Algoritma ini sangat efektif ketika pola terletak di akhir teks atau jika pola tidak ada dalam teks, dan dapat direkomendasikan untuk pencocokan pola.
2.	Retkoceri, F., Idrizi, F., Ismaili, S., Imeri, F., & Memeti, A. (2022)	*Analysis of Pattern Searching Algorithms and Their Application.*	Naive, KMP, Rabin-Karp, Finite Automata, Boyer-Moore,	Hasil dari penelitian ini menunjukkan perbandingan kinerja antara bahasa pemrograman dan

No	Nama Peneliti	Judul Penelitian	Metode	Hasil Penelitian
			Aho-Corasick, dan Z Algorithm	CPU yang digunakan. algoritma yang diimplementasikan dalam bahasa Java lebih cepat pada semua ukuran teks, terutama pada CPU i7-2620m. Hasil penelitian ini juga menunjukan bahwasanya *Naive Pattern Searching Algorithm* memiliki kinerja yang konsisten dalam beberapa kali uji coba
3.	Wirawan, I., & Paryatna, I. (2020)	Implementation of the string matching method on anggah-ungguhing balinese language dictionary.	String Matching Method	Hasil penelitian menunjukkan bahwa fitur koreksi menggunakan *String Matching Method* dengan kompleksitas waktu O(M*N) dapat meningkatkan akurasi terjemahan, dan respons mahasiswa terhadap aplikasi ini dalam proses pembelajaran dinilai sangat positif, mendukung pengembangan ilmu pengetahuan, teknologi, dan budaya sosial.
4.	Abiodun, O. I., dkk. (2019)	Comprehensive review of artificial neural network applications to pattern recognition.	Naive Pattern Searching dan ANN	Hasil dari penelitian ini memnunjukkan bahwa berbagai model ANN (seperti GAN, SAE, DBN, RBM, RNN, CNN, PNN, dan lainnya) berhasil dengan sangat baik dengan tingkat kesalahan yang rendah dan keakuratan yang tinggi dalam pengenalan pola.
5.	Karamizadeh, S., Moazen, M., Zamani, M., & Manaf, A. A. (2024)	Enhancing *IoT*-Based Smart Home Security Through a Combination of Deep Learning and	Convolutional Neural Network (CNN)	Dengan menggunakan model *Deep Learning* (DL) yaitu *Convolutional Neural Network* (CNN) dan self-attention. Hasil

No	Nama Peneliti	Judul Penelitian	Metode	Hasil Penelitian
		Self-Attention Mechanism.		eksperimen menunjukkan akurasi tinggi sebesar 99,7%, menandakan keunggulan model ini dibandingkan model lain dalam hal kecepatan dan akurasi.
6.	Sasia, P. I., Murti, M. A., & Setianingsih, C. (2020)	Implementasi Sistem Kendali Lampu dengan Voice Command Menggunakan Google API.	Recurrent Neural Network (RNN) dan Long Short-Term Memory (LSTM)	Hasil penelitian menunjukkan rata-rata waktu untuk menghidupkan dan mematikan lampu antara 3,28 hingga 3,61 detik, pada intensitas suara minimal sekitar 60-61 dB, dan jangkauan kendali maksimal 1,2 meter. Terdaoat 2 algoritma yang digunakan dalam penelitian ini yaitu, *Recurrent Neural Network* RNN dan *Long-Short-Term Memory Networks* LSTM untuk pemrosesan suara.
7.	Rizky, R. F., Zy, A. T., & Sunge, A. S. (2023)	Sistem Smart Door Lock Menggunakan Voice Recognition Berbasis Arduino.	Hidden Markov Model (HMM)	Dengan menggunakan algoritma *Hidden Markov Model* (HMM) dalam mengenali pola suara dan diimplentasikan dalam sistem kontrol pintu otomatis yang dioperasikan melalui perintah suara. Alat ini dikembangkan menggunakan bahasa pemrograman C++, dan hasilnya menunjukkan bahwa sistem dapat membuka dan menutup pintu sesuai dengan perintah suara yang diberikan.
8.	Fantofani, R. (2020)	Smart home berbasis *Internet of things* menggunakan algoritma Nazief & Adriani dan	Nazief dan Quick Search Algoritm	Hasil penelitian menunjukkan bahwa penggunaan algoritma *Quick Search* dalam sistem smart home berbasis *Internet of*

No	Nama Peneliti	Judul Penelitian	Metode	Hasil Penelitian
		algoritma Quick Search		*Things* (*IoT*) dengan perintah suara mencatat rata-rata waktu yang diperluhkan 0,01565 detik dengan tingkat error sebesar 28%. Hal ini disebabkan karena kesulitan mengenali pola kata yang disebabkan oleh kata pemisah dalam kalimat.
9.	Ulum, M. (2022)	Sistem Kontrol Gorden Berbasis *Internet of Things* menggunakan Algoritma Knuth Morris Pratt	Knuth Morris Pratt	Hasil penelitian menunjukkan bahwa algoritma *Knuth Morris Prat* (KMP) memberikan performa yang sangat baik dengan rata-rata durasi pencarian pola sebesar 513ms dan akurasi mencapai 100% berdasarkan pengujian menggunakan *confusion matrix*.
10.	Rabbani, M. K. (2021)	Sistem kendali pagar rumah berbasis *Internet of Things* dengan perintah suara menggunakan algoritma Boyer-Moore	Boyer-Moore	Hasil penelitian ini menunjukan dari pengujian terhadap 50 kalimat perintah dengan 4 pola yang berbeda, hasilnya menunjukkan akurasi 100% dengan waktu proses rata-rata sekitar 0,00672 ms dengan menggunakan Algoritma *Boyer-Moore* digunakan *untuk pattern matching.*

Bedasarkan penelitian terdahulu yang telag dipaparkan sebelumnya perbedaan dengan penelitian ini terletak pada penerapan algoritma *Naive Pattern Searching* dalam sistem kendali kunci pintu rumah berbasis perintah suara. Penelitian ini berfokus pada pemanfaatan algoritma untuk mengenali perintah suara, berbeda dengan penelitian lain yang menggunakan algoritma pencarian pola

lebih kompleks, seperti KMP, *Boyer-Moore*, dan *Hidden Markov Model* (HMM), serta objek pengujian yang beragam, seperti gorden, lampu, atau pagar. Selain itu, penelitian ini menguji performa *Naive Pattern Searching* sebagai input suara tanpa menggunakan algoritma tambahan atau metode praproses yang kompleks. Dengan demikian, penelitian ini menekankan pada efisiensi dan kemudahan implementasi untuk aplikasi kendali pintu rumah.

2.2 *Intenet of Things*

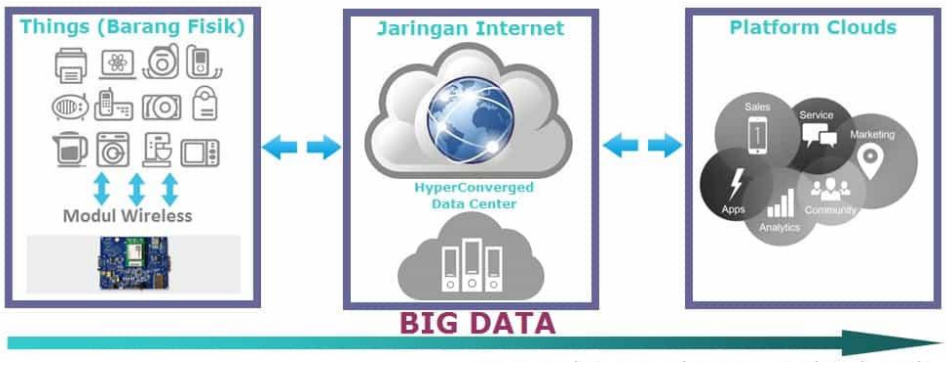
Menurut Prawiyogi *Intermet of Thing (IoT)* merupakan sebuah intfastruktur global yang memiliki jaringan dan tehubung ke internet sehingga, dapat saling komuniasi serta bertukar data tanpa memerlukan kontak fisik dari manusia. *Internet of Thing (IoT)* dilengkapi oleh berbagai perangkat seperti sensor, aktuator, dan modul komunikasi yang berfungsi untuk mengumpulkan, mengirimkan, dan menerima data dari perintah atau input (Prawiyogi dkk., 2023). *IoT* mencakup berbagai aplikasi mulai dari perangkat rumah pintar hingga sistem industri canggih, menawarkan kemampuan otomatisasi dan pengendalian yang lebih canggih.

Bedasarkan penelitian yang dilakukan oleh Selay dkk., munculnya *Internet of Thing(IoT)* berawal pada tahun 1989, internet mulai dikenal secara luas dan memulai era kegiatan online. Penelitian awal mengenai perangkat yang dapat dikendalikan melalui internet dimulai oleh John Romkey pada tahun 1990, ketika ia menciptakan pemanggang roti yang dapat diaktifkan dan dimatikan secara online. Ini menandai langkah awal dalam pengembangan teknologi pengendalian jarak jauh melalui internet. Kemudian, pada tahun 1997, Kevin Ashton, Direktur Eksekutif Auto-ID Lab di MIT, memperkenalkan istilah *"The Internet of Things"* (*IoT*), yang

berbasis pada teknologi *Radio Frequency Identification* (RFID). RFID mulai digunakan dalam skala besar di militer Amerika Serikat pada tahun 2003. Pengembangan Internet Protocol (IP) pada tahun 2008 memberikan dorongan signifikan bagi kemajuan *IoT*, menjadikan perangkat untuk terhubung dan berkomunikasi melalui internet. Seiring waktu, berbagai perusahaan besar mengadopsi teknologi ini, dan berbagai peralatan sehari-hari dengan sensor cerdas dikembangkan untuk dikendalikan secara online. Sensor cerdas ini mengubah data analog menjadi data digital yang kemudian dikirim ke prosesor secara *real-time*, menjadikan otomasi perangkat yang dapat dikendalikan dari jarak jauh dalam arsitektur *IoT* (Selay dkk., 2022).

Setiap perangkat yang ingin terhubung ke internet memerlukan sebuah alamat *Internet Protocol* (IP), yang berfungsi sebagai identitas unik di dalam jaringan. Alamat IP menjadikan perangkat tersebut untuk berkomunikasi dan menerima perintah dari perangkat lain yang juga terhubung dalam jaringan yang sama. Setelah perangkat diberikan alamat IP, ia dapat terhubung ke internet. *IoT* bekerja dengan memanfaatkan instruksi atau perintah pemrograman yang menjadikan perangkat untuk beroperasi secara otomatis dan berkomunikasi dengan perangkat lain tanpa campur tangan manusia, bahkan dari jarak jauh. Faktor utama yang mendukung kelancaran sistem *IoT* adalah koneksi internet, yang menjembatani komunikasi antara sistem dan perangkat. Dalam hal ini, manusia berperan sebagai pemantau yang mengatur dan memberikan perintah untuk proses kerja perangkat. *IoT* memiliki berbagai aplikasi sehari-hari yang, seperti kontrol rumah pintar yang memudahkan pengguna untuk memantau dan mengatur perangkat melalui *remote*

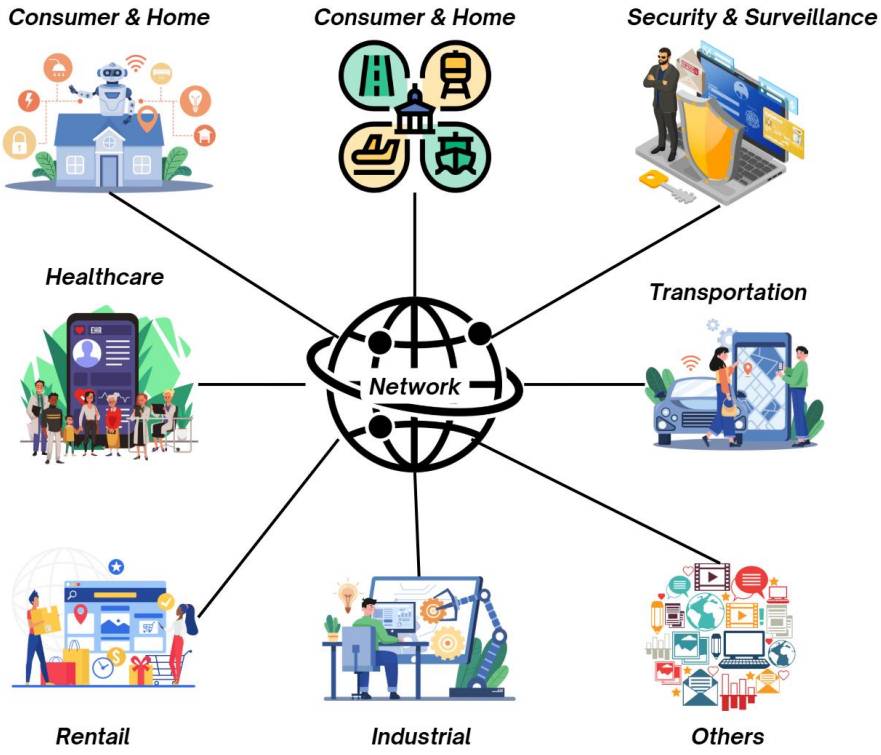
control. Setelah perangkat memiliki alamat IP dan terhubung ke internet, perangkat tersebut biasanya dilengkapi dengan sensor yang mengumpulkan informasi dari lingkungan sekitar. Perangkat dapat mengolah informasi tersebut dan berkomunikasi dengan perangkat lain yang juga terhubung, serta melakukan tindakan otomatis sesuai dengan pengaturan yang telah ditetapkan (Selay dkk., 2022). Gambaran lebih lanjut mengenai konsep dan cara kerja *Internet of Thing (IoT)* dapat dilihat pada Gambar 2.1



Gambar 2. 1 Konsep dan Cara Kerja *Internet of Thing* (Refiansyah Maldini, 2023)

Internet of Things (IoT) mengintegrasikan berbagai perangkat sehari-hari ke dalam jaringan internet, menjadikan mereka untuk berkomunikasi dan beroperasi secara otomatis. Banyak dari kita tidak menyadari bahwa beberapa aktivitas yang kita lakukan sehari-hari, seperti menggunakan peralatan rumah tangga atau sistem keamanan, sudah termasuk dalam cakupan *IoT*. Manfaat *IoT* dapat dirasakan di berbagai sektor, termasuk pembangunan, energi, industri, transportasi, perdagangan, rumah tangga, kesehatan, keamanan, serta teknologi dan jaringan. Misalnya, dalam sektor kesehatan, sensor nirkabel yang dipasang pada tubuh pasien dapat memantau berbagai parameter kesehatan seperti tekanan darah dan detak jantung secara *remote*, menjaga kerahasiaan data pasien. Teknologi *IoT* juga dapat

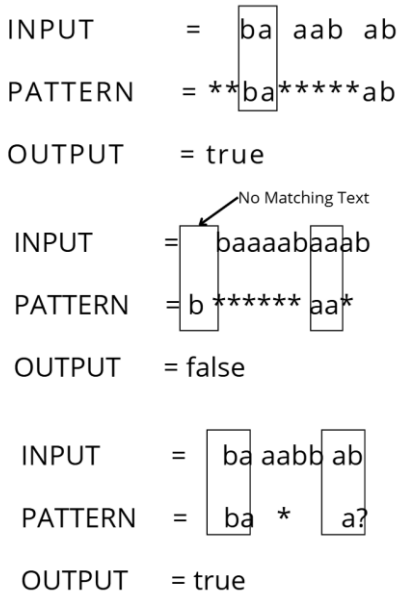
membuat konsultasi medis jarak jauh melalui jaringan WLAN dan internet, meningkatkan aksesibilitas layanan kesehatan. Perkembangan teknologi mobile turut menyokong kemajuan *IoT* dengan mengatasi privasi melalui layanan berbasis lokasi yang mendeteksi dan melindungi privasi pengguna. *IoT* tidak hanya memudahkan kehidupan sehari-hari, seperti mengunci pintu rumah dari jarak jauh melalui smartphone, tetapi juga membantu dalam mengelola berbagai aspek kehidupan seperti menghitung pengunjung toko, membuat sistem alarm rumah, dan mengatur tempat parkir. Misalnya, dengan aplikasi *IoT*, pengguna dapat memesan tempat parkir sebelum tiba di lokasi, serta mendapatkan informasi cuaca secara real-time untuk mengurangi risiko kerugian akibat perubahan cuaca, terutama di daerah pedesaan. Manfaat *IoT* sangat luas dan beragam, memberikan solusi praktis untuk berbagai masalah dalam kehidupan sehari-hari (Susanto dkk., 2022).



Gambar 2. 2 Aktivitas Manusia terhubung dengan Internet

2.3 *Pattern Matching*

Pattern Matching adalah teknik dalam analisis data otomatis yang dilakukan dengan bantuan komputer. Teknik ini menganalisis perbandingan antara sekelompok sifat karakteristik dari suatu objek yang tidak diketahui dengan satu set sifat karakteristik dari objek yang sudah diketahui. Tujuan dari teknik ini adalah untuk menemukan identitas atau klasifikasi yang tepat dari objek yang tidak diketahui berdasarkan kesamaan pola. Dalam konteks pencarian string, *Pattern Matching* digunakan untuk mencari data teks atau biner dengan mencocokkan karakter-karakter tertentu berdasarkan pola pencarian yang telah ditetapkan. Dengan kata lain, teknik ini menjadikan sistem untuk mengidentifikasi dan mengklasifikasikan data yang relevan melalui pencocokan pola yang spesifik (Irmanda dkk., 2021). *Pattern Matching* sering digunakan dalam berbagai aplikasi pemrosesan teks seperti pencarian dokumen, pengenalan pola dalam teks, dan penyaringan spam. Selain itu, teknik ini juga berguna dalam analisis data untuk menemukan pola dalam data numerik atau biner, misalnya dalam deteksi anomali atau dalam pengolahan sinyal. Teknik ini membantu dalam mempercepat pencarian dan ekstraksi informasi dari data besar, yang penting dalam banyak aplikasi teknologi informasi dan data (Rizki dkk., 2023).



Gambar 2. 3 Contoh Proses dalam Teknik Pattern Matching

Pattern Matching memiliki tujuan untuk mempermudah pencarian informasi spesifik dan kompleks dalam database atau file. Misalnya, dalam mesin pencari seperti Google, *Pattern Matching* menjadikan pencarian hasil yang relevan dan efektif berdasarkan kata kunci yang dimasukkan dengan mencocokkan kata kunci tersebut dengan konten yang ada. Teknik ini juga diterapkan dalam aplikasi keamanan seperti antivirus, yang mendeteksi virus atau malware dengan mencocokkan pola tertentu pada file yang diperiksa. Konsep ini merupakan bagian penting dalam bidang seperti *Artificial Intelligence, Machine Learning, dan Data Science,* di mana digunakan untuk pengenalan gambar, pemrosesan bahasa alami, dan deteksi anomali. Dalam pattern matching, istilah-istilah utama meliputi *Pattern* (pola yang dicocokkan), *Text* (data yang diperiksa), dan *Match* (hasil pencocokan antara pola dan data) (Mulyawan, 2024).

2.4 *Naive Pattern Searching Algoritm*

Naive Pattern Searching Algorithm merupakan salah satu algoritma dalam *Natural Language Processing* (NLP). Menurut Supriyona dkk., *Natural Language Processing* adalah cabang dari kecerdasan buatan *Artificial Intelligence* yang berfokus pada kemampuan komputer untuk memahami, menganalisis, dan menghasilkan bahasa manusia secara alami (Supriyono dkk., 2024). Menurut Mukesh T *Naive Pattern Searching Algorithm,* sebagai salah satu metode dasar dalam pencarian pola, digunakan di berbagai bidang seperti pencocokan string, bioinformatika, dan penambangan data. Algoritma ini bekerja dengan cara yang sederhana tetapi efektif dalam mencari pola tertentu dalam data teks, menjadikannya dasar yang penting untuk berbagai aplikasi yang membutuhkan analisis pola secara cepat dan akurat. Algoritma ini, yang juga dikenal sebagai algoritma *Brute Force* atau *Simple Matching,* merupakan pendekatan yang sederhana namun efektif untuk menemukan kemunculan pola dalam teks yang diberikan. Meskipun mudah, algoritma ini merupakan konsep dasar yang penting untuk memahami algoritma pencarian pola yang lebih kompleks. *Naive Pattern Searching Algoritm* bekerja dengan cara mengiterasi teks karakter demi karakter, membandingkan setiap substring dari teks dengan pola yang dicari. Pada setiap posisi dalam teks, algoritma ini memeriksa apakah substring yang dimulai dari posisi tersebut cocok dengan pola. Jika ditemukan kecocokan, algoritma akan mencatat posisi kemunculan pola tersebut. Pendekatan ini, meskipun sederhana, memberikan dasar yang kuat untuk mempelajari dan memahami metode pencarian pola yang lebih efisien (Mukesh T, 2024).

Naive Pattern Searching Algorithm adalah algoritma dasar untuk menemukan kemunculan pola tertentu (suburutan karakter) dalam sebuah string teks yang lebih besar. Algoritma ini cukup sederhana, namun kompleksitas waktunya bisa menjadi tidak efisien untuk dataset yang besar. Dengan persamaan 2.1 dibawah ini:

$$O(M * N) \tag{2.1}$$

Keterangan:
O = Notasi
N = Panjang Teks (String)
M = Panjang Pola (*Pattern)*

Naive Pattern Searching Algoritm melakukan pencarian dengan cara membandingkan pola dengan setiap sub-string yang ada dalam teks utama satu per satu. Pada setiap posisi dalam teks, pola dengan panjang *pattern* dibandingkan dengan sub-string sepanjang *pattern* pada *string*. Proses ini diulang untuk setiap posisi dalam *string* yang dapat menghasilkan kompleksitas waktu seperti pada persamaan 2.1. Dalam scenario terburuk, algoritma melakukan perbandingan $N - M + 1$ masing-masing melibatkan hingga M operasi. Berikut merupakan penjelasan lebih lanjut mengenai *Naive Pattern Searching Algorithm*:

1. Inisialisasi

Algoritma ini menerima dua input, yaitu string teks (txt) dan string pola (pat). Pertama, panjang dari txt (N) dan pat (M) dihitung menggunakan fungsi.

2. Loop Luar (*Outer Loop*)

Loop luar berjalan dari indeks i pada txt hingga $N - M$ untuk memastikan bahwa pola dapat sepenuhnya cocok dalam teks setelah setiap iterasi. Pada setiap iterasi i, loop dalam memeriksa kemungkinan kecocokan pola.

3. Loop Dalam (*Inner Loop*)

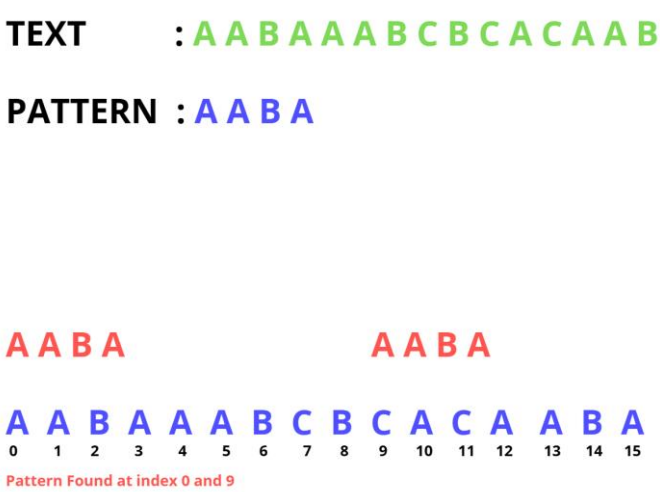
Loop dalam berjalan dari $j = 0$ hingga $M - 1$, membandingkan karakter pada indeks yang sesuai di txt dan pat. Jika ditemukan ketidakcocokan karakter ($txt[i + j] != pat[j]$), loop dalam akan berhenti menggunakan *break*, yang menunjukkan bahwa i saat ini bukan merupakan indeks awal dari pola.

4. Pola Ditemukan

Jika loop dalam selesai tanpa berhenti $j == M$, ini menunjukkan bahwa pat sepenuhnya cocok dengan txt mulai dari indeks i. Algoritma kemudian mencetak *"Pattern found at index -i"*, sebagai indikasi keberhasilan pencocokan.

Naive Searching Algorithm, juga dikenal sebagai *Naïve String Matching Algorithm*, adalah teknik sederhana untuk menemukan kemunculan pola tertentu dalam sebuah string teks yang lebih besar. Algoritma ini mencari semua kejadian string pola (pat) dalam string teks (txt). Dalam kasus terburuk, untuk setiap indeks i dalam loop luar, loop akan berjalan dalam M, (semua karakter dibandingkan) jika tidak ada kecocokan pola sama sekali atau hanya di akhir teks. Hal ini menghasilkan total $(N - M + 1)$*0perbandingan karakter, dengan kompleksitas waktu pada persamaan 2.1 dalam kasus terburuk. Kompleksitas ruang dari *Naive Pattern Searching Algorithm* adalah $O(1)$, yang dianggap konstan. Ini karena algoritma ini terutama menggunakan variabel untuk menyimpan penghitung loop (i, j), panjang string (N, M), dan variabel sementara untuk perbandingan karakter. Ukuran variabel ini tetap konstan tanpa memandang panjang input teks atau pola. Algoritma ini juga tidak membuat struktur data tambahan seperti array atau list untuk

menyimpan hasil sementara atau pola (Ponnaganti, 2024). Dalam gambar 2.4 yang merupakan cara kerja dari *Naive Searching Algorithm* :



Gambar 2. 4 Proses dalam Teknik Pattern Matching

2.5 *Speech Recognizer*

Speech recognition atau pengenalan suara adalah teknologi yang membantu komputer untuk mengenali dan mengidentifikasi suara, kata, atau frasa yang diucapkan oleh manusia, kemudian mengonversi dan menerjemahkannya ke dalam format yang dapat dibaca dan diproses oleh mesin atau komputer. Teknologi ini memiliki beragam aplikasi, seperti dalam penerjemahan bahasa, penyandian data, dan hiburan. *Speech recognition* merupakan bagian dari kemajuan teknologi di bidang kecerdasan mesin, yang semakin terlihat dengan banyaknya perangkat teknologi berbasis komputer yang kini dilengkapi dengan fitur perintah suara atau *voice command* (Riadi, 2023).

Berdasarkan kemampuan *Speech recognition* dalam mengenali kata yang diucapkan, terdapat lima jenis pengenalan kata yang berbeda. Berikut ini

merupakan lima jenis pengenalan kata bedasarkan kemampuan *Speech Recognition* menurut (Riadi, 2023):

1. Kata-kata yang terisolasi, yang mengidentifikasi kata hanya jika ada jeda waktu antar kata yang diucapkan.
2. Kata-kata yang berhubungan, yang serupa dengan kata-kata terisolasi tetapi membutuhkan jeda yang lebih singkat antara kata-kata.
3. Kata-kata yang berkelanjutan, yang menjadikan pengenalan kata yang diucapkan secara terus-menerus dengan jeda waktu yang sangat sedikit atau bahkan tanpa jeda sama sekali, meskipun proses ini lebih rumit karena memerlukan metode khusus untuk membedakan kata-kata yang diucapkan secara berkesinambungan.
4. Kata-kata spontan, yang mengenali kata-kata yang diucapkan secara spontan tanpa jeda waktu antar kata.
5. Verifikasi atau identifikasi suara, yang tidak hanya mengenali kata-kata tetapi juga mampu mengidentifikasi siapa yang berbicara.

2.6 *Stemming*

Stemming adalah suatu proses yang bertujuan untuk menemukan kata dasar dari sebuah kata dengan menghilangkan semua imbuhan, baik berupa awalan (*prefixes*), sisipan (*infixes*), akhiran (*suffixes*), maupun kombinasi awalan dan akhiran (*confixes*) pada kata turunan. *Stemming* merupakan alat dasar dalam pemrosesan teks yang banyak digunakan dalam berbagai aplikasi, seperti text retrieval, mesin penerjemah, meringkas dokumen, dan klasifikasi teks. Berdasarkan asumsi bahwa istilah yang memiliki akar kata yang sama cenderung memiliki

makna yang sama, stemming digunakan dalam information retrieval untuk meningkatkan keakuratan dalam perolehan informasi. Selain itu, proses stemming yang diterapkan saat *indexing* juga dapat mengurangi ukuran dari file indeks, sehingga membuat proses pencarian menjadi lebih efisien (Guterres dkk., 2019).

2.7 *Stopword Removal*

Stopword adalah kata-kata yang dianggap tidak memiliki nilai dalam analisis teks karena tidak memberikan informasi berarti terkait dengan isi sebuah kalimat. Dalam proses *preprocessing* teks, langkah penghapusan *stopword* bertujuan untuk menghilangkan kata-kata yang tidak relevan agar analisis teks lebih fokus pada kata kunci yang signifikan. *Stopword* meliputi kata-kata umum seperti kata penghubung, kata ganti, atau kata-kata lainnya yang sering muncul dalam bahasa tetapi tidak menambah makna kontekstual, misalnya "dan," "di," atau "yang." Daftar *stopword* disimpan dalam bentuk perpustakaan digital yang telah disusun sebelumnya, dan daftar ini sering digunakan dalam berbagai aplikasi pemrosesan teks. Dalam beberapa hal, kata-kata yang dianggap *stopword* secara justru memiliki makna dalam analisis teks. Penghapusan *stopword* membantu meningkatkan efisiensi pemrosesan teks karena mengurangi jumlah kata yang perlu dianalisis lebih lanjut, sehingga mempercepat waktu pemrosesan dan memfokuskan analisis pada kata-kata yang lebih bermakna. *Stopword* yang digunakan harus disesuaikan dengan kebutuhan spesifik agar kata-kata yang relevan dalam konteks tertentu tidak secara tidak sengaja terhapus dari data yang sedang dianalisis (Rinandyaswara dkk., 2022).

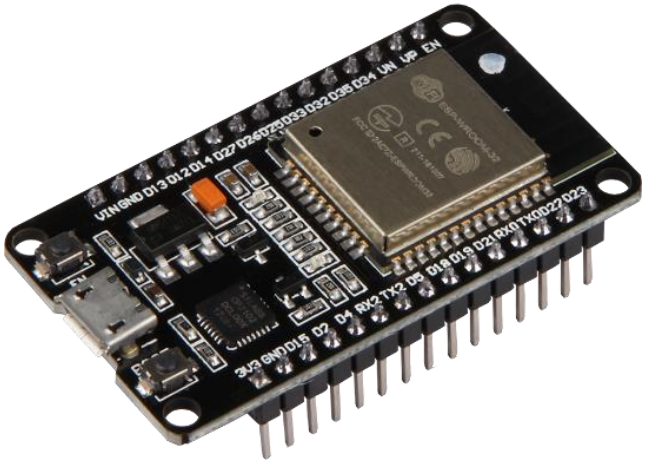
2.8 Arduino IDE

Arduino IDE (*Integrated Development Environment*) adalah perangkat lunak yang digunakan untuk menulis, mengompilasi, dan mengunggah kode ke papan *mikrokontroler* Arduino. Arduino IDE menyediakan antarmuka yang *user-friendly* dengan fitur seperti editor kode, manajer pustaka, manajer papan, monitor serial, serta alat untuk kompilasi dan unggah kode. Ini mendukung berbagai papan Arduino seperti Arduino Uno, Mega, dan papan pihak ketiga seperti ESP32. IDE ini juga dilengkapi dengan berbagai pustaka yang memudahkan integrasi dengan sensor, aktuator, dan komponen lainnya. Dukungan komunitas membuat Arduino IDE menjadi platform yang sangat populer untuk berbagai proyek *IoT* dan robotika (Prianto, 2024). Arduino IDE dapat di dapatkan secara gratis dalam web https://www.arduino.cc/en/software.

2.9 NodeMCU ESP32

NodeMCU adalah platform pengembangan yang berbasis pada modul ESP8266, sebuah mikrokontroler yang dilengkapi dengan kemampuan Wi-Fi. Platform ini dirancang untuk memudahkan pengembangan proyek *Internet of Things* (*IoT*) dengan menyediakan *interface* yang sederhana dan mendukung pemrograman menggunakan bahasa Lua atau Arduino IDE. NodeMCU dapat mempermudah pengguna untuk mengontrol perangkat, mengumpulkan data, dan berkomunikasi dengan server atau perangkat lain melalui jaringan Wi-Fi. Karena kemampuannya yang serbaguna dan harganya yang terjangkau, NodeMCU sering digunakan dalam proyek-proyek DIY, eksperimen *IoT*, dan penelitian akademik

(Phadke & Korde, 2024). NodeMCU ESP32 dapat dilihat lebih jelas pada Gambar 2.5 dibawah ini:



Gambar 2. 5 Hardware NodeMCU ESP32 (Yoga, 2020)

ESP32 merupakan pengembangan hardware dari ESP8266, memperkenalkan sejumlah fitur baru yang semakin cangih, seperti konsumsi daya ultra-rendah dan dukungan untuk Bluetooth serta Wi-Fi secara hybrid. Di pasaran, terdapat berbagai jenis modul ESP32, seperti ESP32 DevKit, Wemos Lolin32, ESP32S Node MCU, dan ESP32 DoIt, masing-masing dengan spesifikasi dan fitur yang berbeda. Modul ESP32 dapat diprogram menggunakan berbagai bahasa pemrograman dan platform, seperti Espressif IDF, LUA, JavaScript, dan Arduino IDE (Radya, 2024). Pada Tabel 2.2 dibawah ini menunjukan spesifikasi dari ESP32:

Tabel 2. 2 Spesifikasi ESP32

No	Fitur	Spesifikasi
1.	Interface ke PC	Micro Usb
2.	Jumlah Core	Dual Core (2)
3.	WiFi	2.4 Ghz up to 150 Mbit/s
4.	Bluetooth	Bluetooth low energy (BLE)
5.	Frekuensi Clock	up to 240 MHz
6.	Pin IO	30 PIN
7.	Fitur	Kapasitif touch
		ADC-DAC
		PWM
		RMII

No	Fitur	Spesifikasi
8.	Komunikasi	IIC, UART, CAN 2.0, SPI, IIS
9.	Arsitektur Programing	32 bits

Dalam Mikrokontroller ESP32 terdapat beberapa pinout yang harus diperhatikan karena bahwa beberapa pin seperti SCK/CLK, SDO/SD0, SDI/SD1, SHD/SD2, SWP/SD3, SCS/CMD, serta GPIO6 hingga GPIO11 tidak disarankan untuk digunakan karena terhubung langsung dengan SPI Flash yang sudah terintegrasi dalam sistem ESP32. Terdapat berbagai varian modul ESP32, sehingga posisi dan jumlah pin dapat bervariasi, dengan umumnya tersedia dalam konfigurasi 30 dan 36 pin. Pinout dalam ESP32 akan disajikan dalam Gambar 2.6 dibawah ini:

Gambar 2. 6 Pinout pada Mikrocontroller ESP32 (Yoga, 2020)

BAB III

DESAIN DAN IMPLEMENTASI

3.1 Desain Sistem

Pada penelitian ini terdapat tiga tahapan dalam pengujian. Tahapan input dalam sistem ini yang berfungsi untuk pengambilan data berupa suara dari pengguna, *button*, dan data *pattern*. Setelah *input,* terdapat tahapan proses, data suara yang diinputkan akan diubah menjadi teks dan melalui tahapan pre-processing, yaitu *Stemming* dan *Stopword Removal*, untuk data. Kemudian, proses *pattern matching* dilakukan menggunakan *Naive Pattern Searching Algorithm* untuk mencocokkan teks dengan *patten* yang sesuai. Hasil dari proses ini diteruskan ke tahapan output, di mana pergerakan servo akan membuka atau mengunci pintu sesuai perintah pengguna. Dibawah ini merupakan penjelasan lebih lanjut mengenai tahapan pada pengujian sistem ini:

3.1.1 Tahapan *Input*

Tahapan input merupakan tahapan penambahan data. *Input* pada sistem ini berupa suara yang bisa diakses *user* melalui perangkat *smartphone*. Selain, input dalam bentuk suara sistem ini juga menyediakan *input* dalam bentuk *button* terdapat 4 *button* yang ada dalam sistem ini yaitu *button* “Buka Pintu Depan”, *button* “Buka Pintu Kamar”, *button* “Kunci Pintu Depan” dan *button* “Kunci Pintu Kamar”.

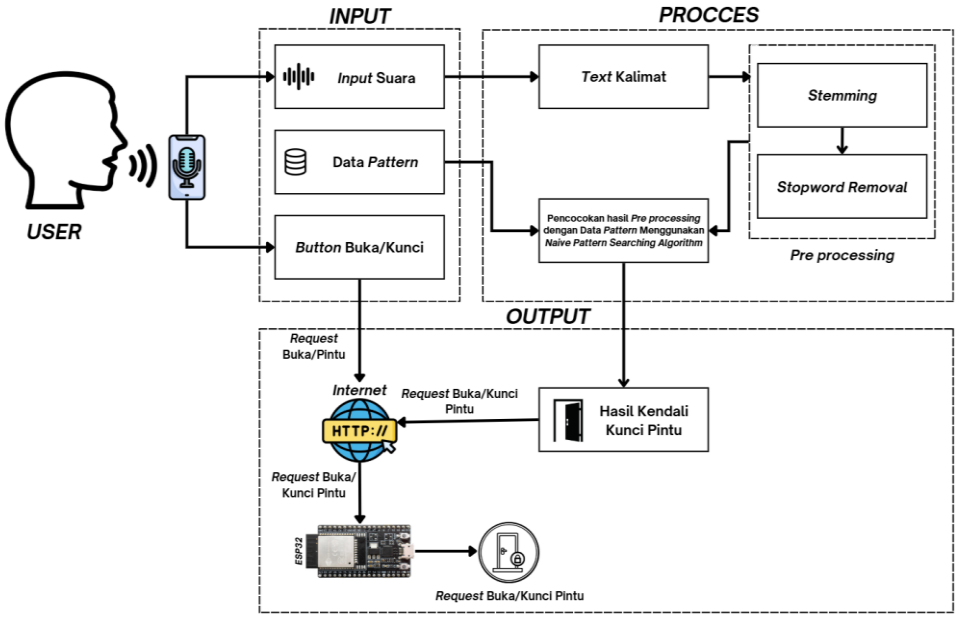
3.1.2 Tahapan *Procces*

Tahapan proses merupakan tahapan yang akan mengolah data yang telah diinputkan dalam bentuk suara dan akan di proses dalam bentuk teks dan setelah teks yang didapatkan akan masuk ketahapan *pre processing*. Terdapat dua tahapan *pre processing* yang digunakan pada penelitian ini yaitu *Stemming* dan *Stopword Removal*. Setelah itu, akan dilakukan proses *pattern matching* menggunakan *Naive Pattern Searching Algoritm* dengan mencocokan teks dan *pattern* untuk mendapatkan hasil dari pengujian sistem.

3.1.3 Tahapan *Output*

Tahapan *output* merupakan tahapan yang menghasilkan hasil berupa pergerakan servo yang akan membuat kunci pintu terbuka atau terkunci sesuai dengan perintah yang telah diinputkan oleh *user* dan diproses oleh *Naive Pattern Searching Algoritm*. Sistem ini nantinya bisa membuka atau mengkunci lebih dari satu pintu dengan *Pattern* yang telah diinputkan sebelumnya. Misalnya, ketika *user* mengucapkan bukakan dua pintu dan *pattern* buka dua pintu merupakan perintah untuk membuka pintu depan dan pintu kamar maka, akan ada dua pintu dalam rumah yang tebuka yaitu, pintu depan dan pintu kamar.

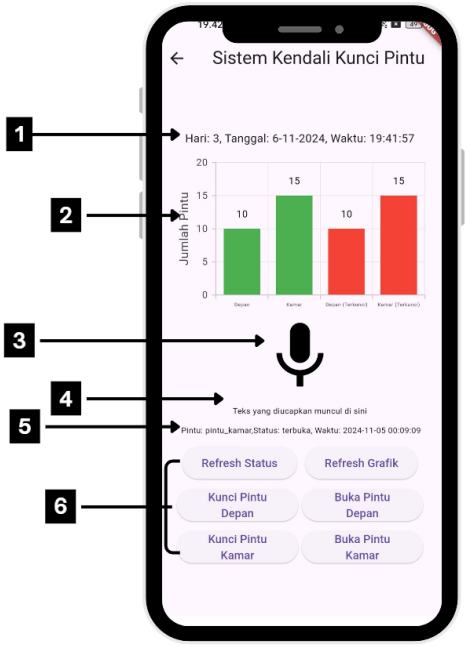
Untuk lebih jelas dapat dilihat pada desain sistem pada penelitian ini pada Gambar 3.1 dibawah ini:



Gambar 3. 1 Desain Sistem

3.2 *User Interface Sistem*

Dalam penelitian ini terdapat tampilan antarmuka (*user interface)* yang bertujuan untuk mempermudah pengguna saat menggunakan sistem ini. setiap tampilan memiliki fungsi, berikut merupakan tampilan dari *user interface* dalam sistem ini:



Gambar 3. 2 Tampilan User Interface dalam Sistem Kendali Kunci Pintu

Keterangan :

1. Waktu
2. Grafik Aktivitas
3. Microphone untuk input suara
4. Suara yang telah diucapkan diproses menjadi teks
5. Status Pintu dengan waktu terakhir
6. Button

3.3 Akusisi Data

Terdapat dua jenis data yang digunakan dalam penelitian ini yaitu, data primer dan data skunder. Data primer digunakan untuk menghasilkan perintah. Sedangkan, data sekunder digunakan untuk kata kunci atau *pattern*. Berikut merupakan penjelasan lebih lanjut mengenai data yang digunakan dalam penelitian ini:

3.2.1 Data Primer

Data primer diperoleh dari suara yang diinputkan oleh *user* pada saat pengujian sistem yang nantinya akan menghasilkan perintah dengan format teks yang nantinya akan di proses menggunakan *Naive Pattern Searching Algoritm*.

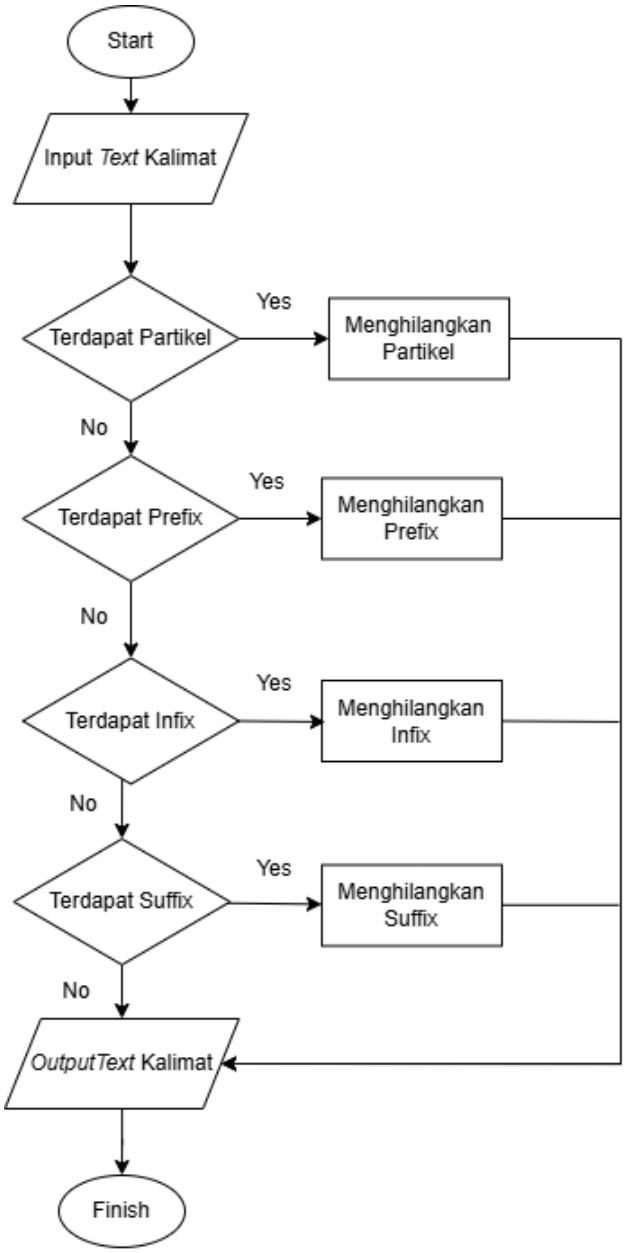
3.2.2 Data Sekunder

Data sekunder diperoleh dari data *pattern* yang diinputkan sebelumnya oleh peneliti. Data sekuender ini dijadikan acuan untuk pengujian sistem data sekunder dalam sistem ini berupa “Buka”, “Kunci”, “Pintu Depan”, “Pintu Kamar”, dan “Semua”

3.4 Proses Stemming

Stemming pada penelitian ini akan memeriksa adanya *partikel* dalam kata, seperti "lah," "kah," atau "pun," yang tidak mengubah arti dasar dari kata tersebut. Partikel ini akan dihapus jika ditemukan. Setelah itu, akan memeriksa apakah kata tersebut memiliki *prefiks*, seperti "ber-," "ter-," atau "se-," yang menunjukkan awalan tertentu. Jika *prefiks* ditemukan maka, akan menghapusnya untuk mendapatkan bentuk dasar kata. Selanjutnya, akan memeriksa adanya *infiks*, yaitu kata yang disisipkan di tengah kata, seperti "-el-," "-er-," atau "-in-," dan menghilangkannya jika ada. Terakhir, akan memeriksa *sufiks* atau akhiran kata, seperti "-an," "-i," atau "-kan." Jika ditemukan *sufiks* maka akan dihapus untuk menyelesaikan proses stemming dan mendapatkan bentuk dasar kata yang diinginkan. Proses ini bertujuan untuk memastikan bahwa kata-kata dalam teks dikembalikan ke bentuk dasarnya, sehingga berbagai bentuk dari kata yang sama

dapat dikenali sebagai satu entitas dalam analisis teks. Proses *Stemming* pada penelitian ini dapat dilihat dalam Gambar 3.2 dibawah ini:



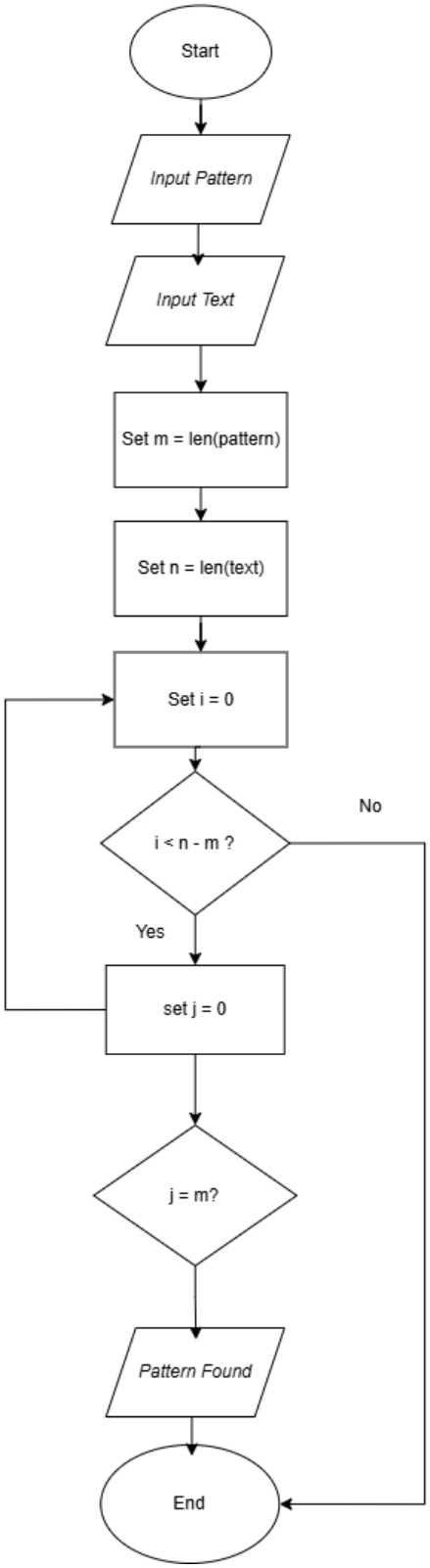
Gambar 3. 3 Proses *Stemming*

3.5 *Stopword Removal*

Pada penelitian ini, salah satu tahapan *pre processing* yang dilakukan adalah proses *Stopword Removal*, yaitu penghapusan kata-kata penghubung seperti "dan," "atau," "yang," serta kata penghubung dan kata ganti lainnya. Kata-kata ini tidak memiliki nilai dalam pengenalan teks, karena tidak menambah informasi atau makna dari pengenalan kalimat. Tujuan dari penghapusan *stopword* adalah untuk menyederhanakan dan mempercepat proses analisis teks, sehingga fokusnya bisa lebih terarah pada kata-kata kunci yang lebih bermakna. Dengan menerapkan S*topword Removal*, proses analisis teks pada penelitian ini dilakukan lebih efisien karena jumlah kata yang dianalisis menjadi lebih sedikit. Ini menjadikan fokus pada bagian teks yang relevan dengan hasil yang diharapkan dari penelitian.

3.6 *Naive Pattern Searching Algoritm*

Naive Pattern Searching Algorithm memiliki cara bekerja dengan memeriksa setiap posisi dalam teks untuk melihat apakah ada kesesuaian antara pola yang dicari dengan segmen teks di posisi tersebut. Karena sifatnya yang sederhana, algoritma ini mudah diimplementasikan tetapi memiliki kekurangan yaitu tidak efisien untuk teks atau pola yang sangat panjang. Pada Gambar 3.3 dibawah ini merupakan *Flowchart* mengenai cara kerja pada *Naïve Pattern Searching Algoritm* :



Gambar 3. 4 Flowchart cara kerja pada Naive Pattern Searching Algoritm

Algoritma *Naive Pattern Searching* bekerja untuk mencocokkan pola (pattern) dalam sebuah teks, seperti yang digambarkan dalam flowchart Gambar 3.3. Proses dimulai dengan langkah Start, di mana algoritma diaktifkan. Selanjutnya, pengguna memasukkan pola (pattern) yang ingin dicari dan teks (text) yang menjadi tempat pencarian. Panjang pola kemudian dihitung dan disimpan dalam variabel m dengan rumus m = len(pattern), sedangkan panjang teks disimpan dalam variabel n dengan n = len(text). Variabel i diinisialisasi ke nilai 0 untuk menandai posisi awal dalam pencarian teks.

Proses selanjutnya adalan *looping* yang kedua yaitu *Outer Loop*, yang memastikan pencarian hanya dilakukan pada bagian teks yang cukup panjang untuk memuat pola. Kondisi ini diperiksa dengan i < n - m. Jika kondisi terpenuhi, algoritma melanjutkan ke langkah berikutnya, tetapi jika tidak, proses pencarian berakhir tanpa menemukan pola. Dalam pencarian pola, variabel j diinisialisasi ke nilai 0 untuk menelusuri karakter dalam pola. Selanjutnya, Inner Loop memeriksa apakah nilai j telah sama dengan m, yang menunjukkan bahwa pola telah ditemukan dalam teks. Jika pola ditemukan, algoritma mencatat keberhasilannya pada langkah Pattern Found. Sebaliknya, jika tidak ditemukan, iterasi dilanjutkan dengan menggeser posisi pencarian di teks. Proses berakhir dengan langkah End setelah seluruh teks diperiksa. Algoritma ini bekerja secara langsung dengan membandingkan setiap karakter pola dengan teks secara berurutan, sehingga sederhana namun kurang efisien untuk teks yang panjang atau pola yang kompleks.

Jika tidak, algoritma kembali ke langkah i < n - m? setelah menginkrementasi nilai i. Algoritma ini memiliki kompleksitas waktu O(N * M),

di mana N adalah panjang teks dan M adalah panjang pola. Ini karena algoritma menggunakan dua loop: loop luar yang beriterasi dari 0 hingga N-M, dan loop dalam yang membandingkan setiap karakter dalam pola dengan segmen teks yang bersesuaian.

Berikut adalah tabel contoh dari langkah-langkah pengujian Naive Pattern Searching Algorithm menggunakan teks "TOLONG BUKA PINTU" dan pola "BUKA PINTU":

Tabel 3. 1 Langkah-Langkah Pengujian *Naive Pattern Searching Algorithm*

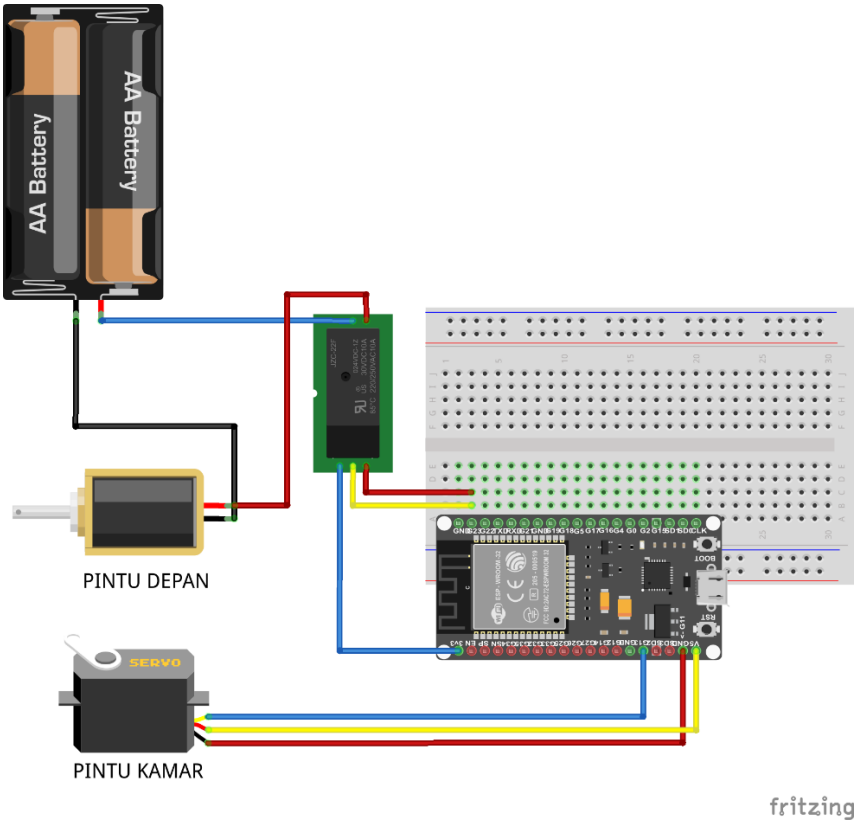
Langkah	**i**	**j**	**Karakter Teks**	**Karakter Pola**	**Keterangan**
1.	0	0	T	B	Tidak cocok, geser i ke 1
2.	1	0	O	B	Tidak cocok, geser i ke 2
3.	2	0	L	B	Tidak cocok, geser i ke 3
4.	3	0	O	B	Tidak cocok, geser i ke 4
5.	4	0	N	B	Tidak cocok, geser i ke 5
6.	5	0	G	B	Tidak cocok, geser i ke 6
7.	6	0	(spasi)	B	Tidak cocok, geser i ke 7
8.	7	0	B	B	Cocok, lanjut ke j = 1
9.	7	1	U	U	Cocok, lanjut ke j = 2
10.	7	2	K	K	Cocok, lanjut ke j = 3
11.	7	3	A	A	Cocok, lanjut ke j = 4
12.	7	4	(spasi)	(spasi)	Cocok, lanjut ke j = 5
13.	7	5	P	P	Cocok, lanjut ke j = 6
14.	7	6	I	I	Cocok, lanjut ke j = 7
15.	7	7	N	N	Cocok, lanjut ke j = 8
16.	7	8	T	T	Cocok, lanjut ke j = 9
17.	7	9	U	U	Cocok, pola ditemukan pada indeks ke-7

Keterangan:

Pada langkah ke-8 hingga ke-17, semua karakter dalam pola "BUKA PINTU" cocok dengan segmen teks yang dimulai pada indeks ke-7. Oleh karena itu, pola ditemukan pada indeks ke-7 dalam teks.

3.7 Desain Komponen

Pada penelitian ini terdapat desain komponen yang bertujuan untuk memberikan gambaran mengenai *hardware* sistem. Dalam penelitian ini menggunakan ESP32 WROOM sebagai *microcontroller, Relay, Selanoid Lock* sebgaia kendali kunci pintu depan, servo sebagai kendali kunci pintu kamar, dan Baterai 12V sebagai sumber daya *selanoid kock.* Desain komponen pada penelitian ini dapat dilihat pada gambar 3.5 dibawah ini:



Gambar 3. 5 Desain Komponen dalam Sistem Kendali Kunci Pintu

3.8 Pengujian Sistem

Pada penelitian ini terdapat tahapan pengujian sistem yang memiliki tujuan untuk mengetahui dan mengevaluasi terkait keakuratan sistem kontrol kunci pintu dengan menggunakan *Naive Pattern Searching Algoritm*. Pengujian dilakukan

dengan menjalankan sistem sebanyak 700 kali percobaan dengan kalimat perintah yang berbeda beda. Skenario uji coba pada penelitian ini dijabarkan pada table 3.2 dibawah ini:

Tabel 3. 2 Skenario Uji Coba Sistem

No	Pengenalan Perintah	Diujikan Oleh	Jumlah Uji Coba
1	Buka Pintu Depan	Peneliti	20 kali
		10 Orang *User*	100 kali
2	Buka Pintu Kamar	Peneliti	20 kali
		10 Orang *User*	100 kali
3	Kunci Pintu Depan	Peneliti	20 kali
		10 Orang *User*	100 kali
4	Kunci Pintu Kamar	Peneliti	20 kali
		10 Orang *User*	100 kali
5	Buka Semua Pintu	Peneliti	10 kali
		10 Orang *User*	100 kali
6	Kunci Semua Pintu	Peneliti	10 kali
		10 Orang *User*	100 kali
Total Uji Coba			700 kali

Pada tabel 3.2 dituliskan bahwasanya terdapat 700 kali percobaan dengan 6 skenario pengenalan perintah. Pada setiap skenario pengenalan perintah akan diuji oleh peneliti dan 10 orang *user*. Sistem yang diuji coba akan dihitung performa keberhasilan atau *succes* dalam merespon *user* dan tingkat *error* atau kegagaglan sistem dalam merespon dengan menggunakan sebuah persamaan. Perhitungan ini dilakukan dengan tujuan untuk mengetahui seberapa besar perbedaan antara hasil yang diharapkan dan hasil aktual yang didapat dari pengujian (Frost, 2021). Berikut merupakan persamaan yang digunakan dalam penelitian ini:

$$\mu = \frac{\sum Ei}{n} \text{ X } 100\% \tag{3.1}$$

Keterangan:
μ : rata-rata dalam bentuk persen (%)
n : jumlah percobaan
∑Ei : jumlah kesalahan dari semua percobaan

Tabel 3. 3 Contoh Alur Pengujian Error

Percobaan	***Input***	***Pattern***	**Hasil Pengujian**	***Error***
1.	Bukain Pintu Depan	Buka, Pintu Depan	*True*	0
2.	Bukakan Pintu Kamar	Buka, Pintu Kamar	*True*	0
3.	Buka Pintu kamarnya dong	Buka Pintu Kamar	*True*	0
4.	Bukain Semua Pintu	Buka, Semua	*False*	1
5.	Membuka Semua Pintu	Buka, Semua	*True*	0
6.	Kunci Semua Pintunya dong	Kunci, Semua	*True*	0
7.	Pintunya Dikunci	Kunci Pintu	*False*	1
…	…	…	…	…
99.	Buka Pintu Kamar	Buka, Pintu Kamar	*False*	1
100.	Kunci Pintu Depan	Kunci Pintu Depan	*False*	1
∑Ei (Jumlah kesalahan dari semua percobaan)				4

Pada tabel 3.3 dapat diketahui bahwa dalam 100 percobaan terdapat 4 *error*. Adapun perhitungan *error* dari pengujian sistem dengan 4 kali *error* adalah sebagai berikut:

$$\mu = \frac{\sum Ei}{n} \text{ X } 100\%$$

$$\mu = \frac{4}{100} \text{ X } 100\% \qquad (3.2)$$

$$\mu = 4\%$$

BAB IV

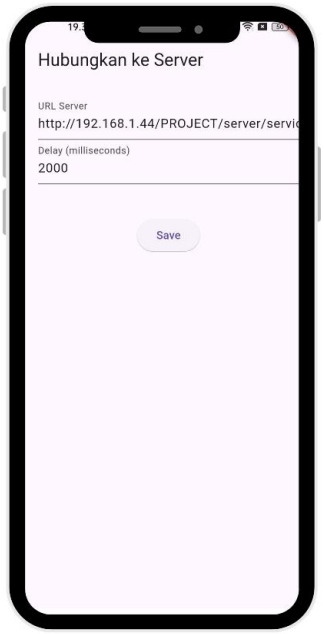
UJI COBA DAN PEMBAHASAN

4.1 Implementasi Sistem

Implmentasi sistem merupakan tahapan dalam pengembangan sistem di mana desain yang telah disusun dalam rancangan diubah menjadi sistem yang siap digunakan. Tujuan dari tahap implementasi ini adalah untuk menguji apakah sistem bekerja sesuai dengan spesifikasi yang diharapkan dan memberikan rekomendasi berdasarkan hasil uji coba (Rabbani, 2021). Terdapat 2 sistem yang dirancang dalam penelitian ini yaitu sistem *Software* dan sistem *Hardware*.

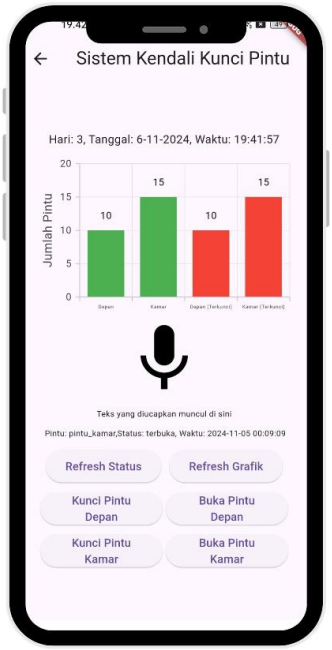
4.2.1 Sistem *Software*

Menurut Hariyadi dan Prakasa sistem *software* merupakan perangkat lunak yang mendukung sistem informasi (Hariyadi & Prakasa, 2023). Sistem *Software* pada penelitian ini dikembangkan dalam penelitian ini merupakan aplikasi berbasis android yang digunakan untuk mengendalikan sistem *hardware*. Aplikasi ini dikembangkan menggunakan *framework* dari google yaitu *flutter*. *Framework flutter* dipilih dalam pengembanngan aplikasi pada penelitian ini karena *flutter* dianggap lebih efisien dan responsif baik dari tampilan *User Interfaces* ataupun fungsi dalam pengenalan suara. Dibawah ini merupakan tampilan-tampilan aplikasi yang telah dirancang dalam penelitian ini dengan menggunakan *framework flutter* dan bahasa pemograman dart.



Gambar 4. 1 Tampilan Awal

Gambar 4.1 merupakan tampilan awal dari aplikasi saat pertama kali aplikasi dibuka. Tampilan awal ini memiliki fungsi untuk mengatur koneksi antara *user* dan *server*. Terdapat *field "URL Server"* yang berfungsi untuk mengisi Alamat IP dan path pada jaringan lokal untuk mengirim dan menerima data. Dibawah *field "URL Server"* terdapat *field "Delay (miliseconds)"* yang berfungsi sebagai interval waktu atau jeda antara pengiriman data atau *request* ke *server* selama dua detik. *Delay* ini berfungsi untuk mengatur frekuensi dengan agar tidak *overload* pada jaringan. Selanjutnya, terdapat tombol *"Save"* yang berfungsi untuk menyimpan konfigurasi *URL Server* dan nilai *delay* yang di isi pada *field*. Jika, data yang diberikan oleh *user* benar dan *user* menekan tombol *"Save"* maka, akan terhubung ke *server* dan akan beralih ke halaman berikut nya yaitu, halaman *Default* seperti pada Gambar 4.2 dibawah ini.



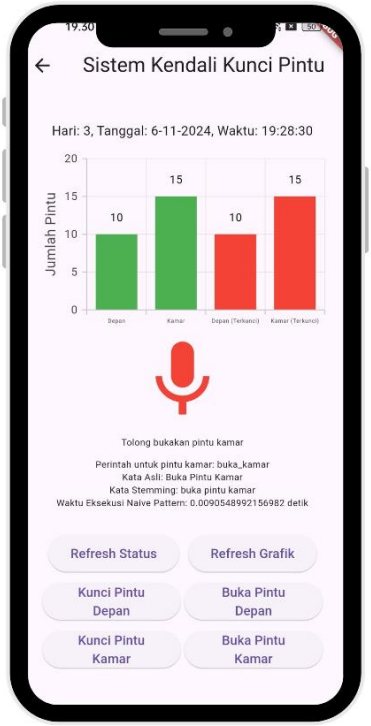
Gambar 4. 2 Tampilan Default

Pada Gambar 4.2 yaitu tampilan *default* terdapat beberapa fitur yang ditampilkan. Berikut merupakan penjelasan mengenai elemen elemen pada tampilan *default* pada sistem ini:

1. Grafik status pintu pada sistem ini ditampilkan dalam grafik batang yang menunjukan status pintu depan dan pintu kamar. Pada sumbu y atau sumbu vertikal digunakan untuk status pintu yaitu, buka pintu depan, buka pintu kamar, tutup pintu depan, dan buka pintu depan. Pada sumbu x atau sumbu horizontal terdapat jumlah status pintu yang terbuka. Grafik ini memiliki fungsi untuk memvisualisasikan informasi dari status pintu berapa kali pintu dibuka dan berapa kali pintu ditutup.
2. Ikon mikrofon pada sistem ini berfungsi untuk tombol memulai perintah suara. *User* menekan tombol ini untuk memberikan perintah dengan suara.

Setelah perintah diberikan maka, akan dikonversi menjadi teks dan akan muncul pada area “Teks yang diucapkan muncul di sini”

3. Area teks status ini berfungsi untuk menampilkan status pintu dari *server*. Data yang ditampilkan pada area teks ini merupakan data status terakhir pada *server*. Misalnya, “Pintu: pintu_kamar; Status: terbuka; Waktu: 2024-11-05 00:09:09,” yang menunjukkan status pintu kamar terbuka pukul 00:09:09 pada 2024-11-05.
4. Button terdapat enam *button* pada sistem ini. Berikut merupakan penjelasan dari masing masing *button*:
5. Refresh status yang berfungsi untuk memperbarui status kunci pintu, memastikan data yang ditampilkan pada aplikasi adalah data terbaru.
    - Refresh grafik berfungsi untuk memperbarui tampilan grafik dengan data terkini dari status kunci pintu.
    - Kunci Pintu Depan *button* ini berfungsi untuk memberikan perintah mengunci pintu depan.
    - Buka Pintu Depan *button* ini berfungsi untuk memberikan perintah membuka pintu depan.
    - Kunci Pintu Kamar *button* ini berfungsi untuk memberikan perintah mengunci pintu kamar.
    - Buka Pintu Kamar *button* ini berfungsi untuk memberikan perintah membuka pintu kamar.



Gambar 4. 3 Tampilan Speech Recognition

Gambar 4.3 merupakan tampilan *Speech Recognition* pada aplikasi sistem ini yang berfungsi untuk mengkonversi suara menjadi menjadi teks dan ditampilkan pada *text field* di bawah ikon mikrofon dan teks tersebut akan dikirimkan ker *server* untuk dikelola. Suara dapat di konversi menjadi teks pada aplikasi menggunakan *package speech_to_text* yang ada pada *framework flutter*.

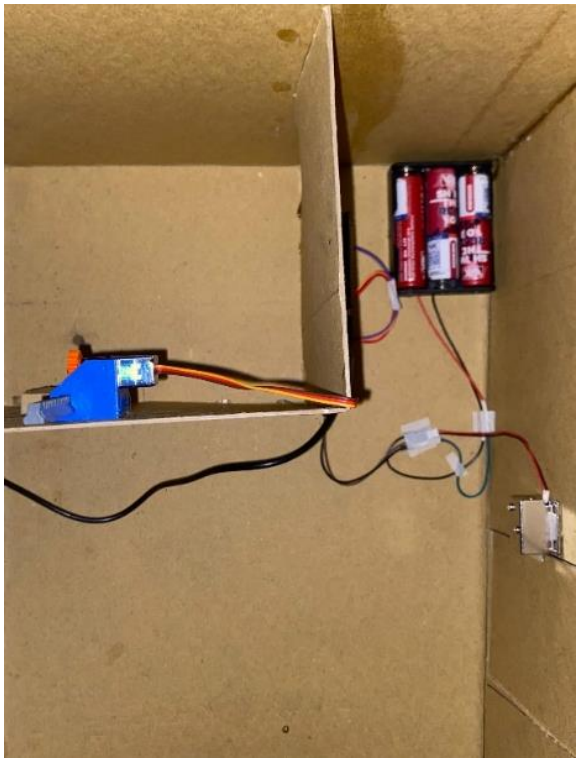
Gambar 4. 4 Tampilan Perintah

Gambar 4.4 merupakan tampilan perintah yang merupakan respon dari *server* setelah perintah dikirimkan. *Server* akan merespon perintah dari *user* dengan menampilkan Kata Asli, Kata *Stemming*, Waktu Eksekusi *Naive Pattern* dalam mencocokan perintah yang diberikan oleh *user* dan data *pattern*, dan Perintah untuk sistem *hardware*.

4.2.2 Sistem *Hardware*

Menurut Hariyadi dan Prakasa sistem *hardware* merupakan perangkat keras yang berupa komponen elektronik pendukung sebuah komputer (Hariyadi & Prakasa, 2023). Sistem *hardware* yang dikembangkan dalam penelitian ini adalah *prototyepe* rumah dengan dua pintu yaitu, pintu depan dengan menggunakan kunci *selanoid lock* dan pintu kamar menggunakan kunci *servo*. Sistem ini menggunakan microkontroler *NodeMCU* ESP32 sebagai perangkat untuk mengendalikan rangkaian *hardware* dan menerima perintah dari *server*. Perintah suara yang

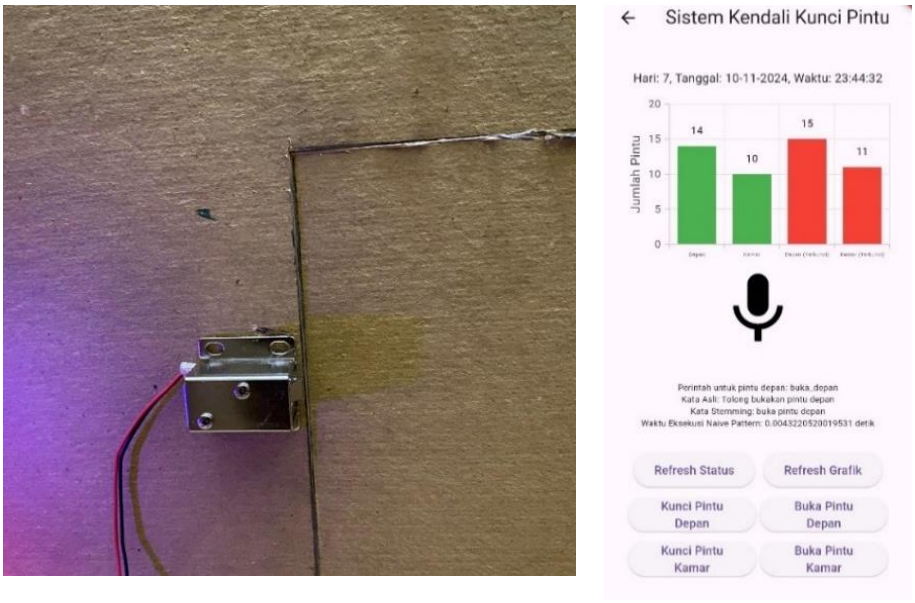
diberikan *user* melalui aplikasi akan diterima oleh *server* dan akan diambil untuk menggerakan *hardware* dengan protokol *Internet of Thing (IoT)* yaitu, *Hypertext Transfer Protocol (HTTP)*. Jika, data berupa perintah untuk pintu depan maka data akan dikirimkan ke *relay* untuk dikendalikan aliran listriknya dan selanjutnya akan mengerakan *selanoid lock*. Dan jika, data berupa perintah untuk menggendalikan pintu kamar maka, data akan langsung dikirimkan ke *servo* dan mengerakan *servo*. Rangkaian *hardware* daripada penelitian ini dapat dilihat pada Gambar 4.5.



Gambar 4. 5 Rangkaian Hardware

Rangakaian *Hardware* yang disajikan pada Gambar 4.5 terdiri dari ESP32 WROOM sebagai *Mikrocontroller*, Baterai 12 V untuk sumber daya *selanoid lock* atau kunci pintu depan, relay, servo untuk kunci pintu kamar. Ketika pintu depan

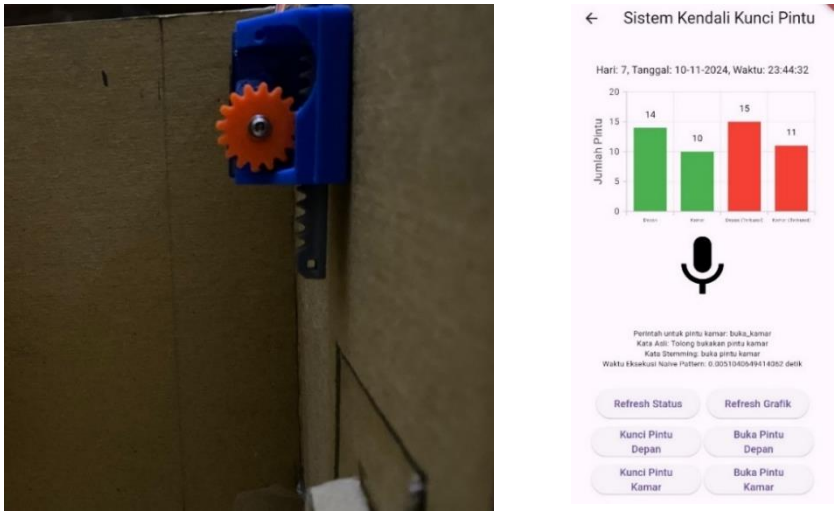
dalam kondisi terbuka maka selanoid lock ada di posisi *"Low"*. Pada Gambar 4.6 merupakan kondisi pintu depan terbuka dan *selanoid lock* dalam posisi *"Low"*.



(a) (b)

Gambar 4. 6 (a) Kondisi Pintu Depan Tebuka (b) Tampilan Aplikasi Pintu Depan Terbuka

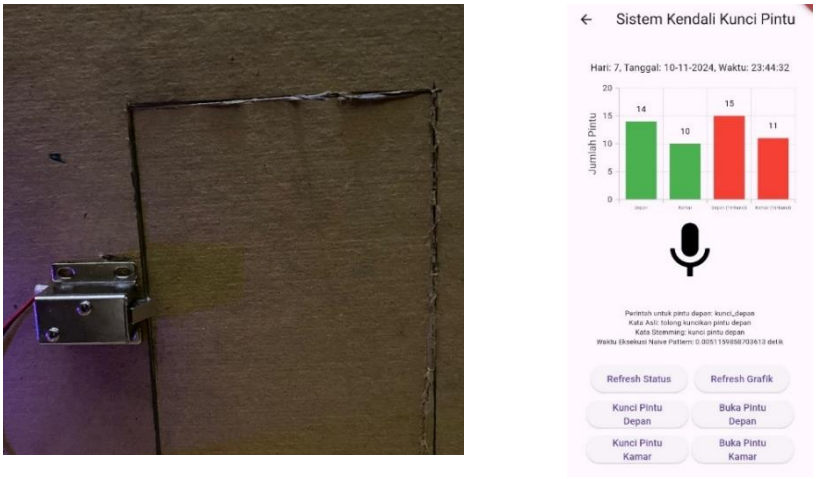
Kemudian, ketika pintu kamar dalam kondisi terbuka maka servo akan ada dalam posisi 180 derajat. Pada Gambar 4.7 merupakan kondisi pintu depan terbuka dan servo dalam posisi 180 derajat.



(a) (b)

Gambar 4. 7 (a) Kondisi Pintu Kamar Tebuka (b) Tampilan Aplikasi Pintu Kamar Terbuka

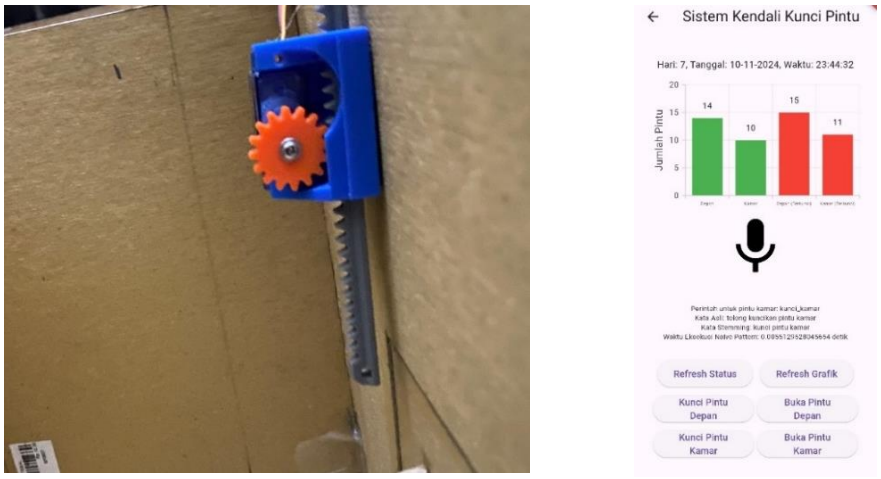
Kemudian, ketika pintu depan dalam kondisi terkunci maka selanoid lock ada di posisi *"High"*. Pada Gambar 4.8 merupakan kondisi pintu depan terbuka dan *selanoid lock* dalam posisi *"High"*.



(a) (b)

Gambar 4. 8 (a) Kondisi Pintu Depan Terkunci (b) Tampilan Aplikasi Pintu Depan Terkunci

Pada Gambar 4.9 merupakan kondisi pintu depan terbuka dan servo dalam posisi 0 derajat.



(a) (b)

Gambar 4. 9 (a) Kondisi Pintu Kamar Terkunci (b) Tampilan Aplikasi Pintu Kamar Terkunci

4.2 Pengujian Sistem dengan *Naive Pattern Searching Algoritm*

Proses pengujian dengan *Naive Pattern Algoritm* merupakan pengujian atau pencocokan kalimat perintah yang telah diberikan oleh *user* pada aplikasi atau sistem *software* dengan *pattern*. Pada *Naive Pattern Searching Algoritm* melakukan pengujian dengan dua kali perulangan yaitu, *Outer loop* dan *Inner loop*. Perulangan pertama atau *Outer loop* merupakan perulangan yang beriterasi dari 0 hingga N-M. Dan untuk perulangan kedua atau *Outer Loop* merupakan perulangan yang mengecek setiap karakter dalam pola dengan segmen teks yang bersesuaian. Pengujian dari *Naive Pattern Algoritm* ini dapat dilihat pada Pseudocode 4.1.

Pseudocode 4. 1 Proses Perhitungan Naive Pattern Searching Algoritm dalam sistem

```
// Fungsi untuk perhitungan Naïve Pattern Searching Algoritm
fungsi search_pattern(patterns, text)

    // Langkah 1: Deklarasi array untuk menyimpan indeks
    found_indices = []

    // Langkah 2: Menentukan nilai panjang pattern dan text
    Untuk setiap pattern dalam patterns:
        m = Panjang dari pattern
        n = Panjang dari text

    // Langkah 3: Memeriksa posisi dalam tekt untuk dicocokkan
dengan pattern
```

```
        Untuk i dari 0 sampai (n - m):
            j = 0

    // Langkah 4: Memeriksa kesamaan antara pattern dan text
        Selama (j < m) dan (text[i + j] == pattern[j]):
           Menambahkan 1 ke j

    // Langkah 5: Jika karakter cocok, maka akan menambahkan
posisi i ke found_indices
        Jika j == m:
           Tambahkan i ke dalam found_indices

    // Langkah 6: Mengembalikan hasil pencarian kedalam array
        Kembalikan found_indices
```

Setelah, dilakukan proses pengujian dengan *Naive Pattern Searching Algoritm* maka, proses selanjutnya adalah proses keputusan perintah. Hasil dari perintah ini akan dikirimkan ke sistem *hardware* dengan prokotol HTTP. Terdapat enam perintah yang bisa dikenali oleh sistem yaitu, “buka_semua” yang berarti akan membuka pintu depan dan pintu kamar, “kunci_semua” yang berarti akan mengunci pintu depan dan pintu kamar, “buka_depan“ yang berarti akan membuka pintu depan, “buka_kamar” yang berarti akan membuka pintu kamar, “kunci_depan” yang berarti akan mengunci pintu kamar, dan “kunci_kamar” yang berarti akan mengunci pintu kamar. Logika keputusan perintah dapat dilihat pada Pseudocode 4. 2.

Pseudocode 4. 2 Logika Keputusan Perintah dalam Sistem

```
// Fungsi untuk keputusan perintah buka dan kunci pintu
Fungsi untuk keputusan perintah buka dan kunci pintu
fungsi proses_perintah(result_indices):
// Langkah 1: Cek apakah terdapat perintah yang menyatakan buka
semua pintu
    Jika result_indices['semua'] != -1 dan
result_indices['buka'] != -1:
http_response = simpanPerintah(["buka_semua"]);
Menampilkan "Perintah: Buka semua pintu"
// Langkah 2: Cek apakah terdapat perintah yang menyatakan kunci
semua pintu
    elseif result_indices['semua'] != -1 dan
result_indices['kunci'] != -1:
http_response = simpanPerintah(["kunci_semua"]);
Menampilkan "Perintah: Kunci semua pintu"
// Langkah 3: Jika tidak ada perintah untuk membuka atau
mengunci semua pintu maka, prohram akan memeriksa pintu depan
atau pintu kamar
else:
p = [] // Inisialisasi array untuk menyimpan perintah
// Langkah 4: Cek perintah untuk pintu depan
       Jika result_indices['pintu_depan'] != -1:
// Langkah 5: Tentukan perintah buka atau kunci untuk pintu
depan
       Jika result_indices['buka'] != -1:
Tambahkan "buka_depan" ke dalam p
       else jika result_indices['kunci'] != -1:
Tambahkan "kunci_depan" ke dalam p
```

```
 Tampilkan
"Perintah untuk pintu depan: " + perintah terakhir dalam p
// Langkah 6: Cek perintah untuk pintu kamar
        Jika result_indices['pintu_kamar'] != -1:
// Langkah 7: Tentukan perintah buka atau kunci untuk pintu
kamar
        Jika result_indices['buka'] != -1:
Tambahkan "buka_kamar" ke dalam p
        else jika result_indices['kunci'] != -1:
Tambahkan "kunci_kamar" ke dalam p
        Tampilkan
"Perintah untuk pintu kamar: " + perintah terakhir dalam p
// Langkah 8: Hapus nilai null dari array p
        p = filterHapusNull(p)
// Langkah 9: Jika ada perintah yang valid, simpan perintah dan
tampilkan respons
        Jika p tidak kosong:
            http_response = simpanPerintah(p)
        else:
            http_response = "Ulangi perintah";
Tampilkan "Output: " + http_response
// Langkah 10: Kembalikan respons HTTP
    Kembalikan http_response
```

4.2.1 Pengenalan Perintah “Buka Pintu Depan”

Proses pengujian pada skenario pertama yaitu pengenalan kalimat perintah “Buka Pintu Depan”. Proses ini dilakukan sebanyak 120 kali yang mana 20 kali

diuji oleh peneliti dan 10 kali diuji oleh 10 orang *user* dengan kalimat perintah yang berbeda. Hasil dari pengujian ini dapat dilihat pada tabel 4.1 dan tabel 4.2 dibawah ini:

Tabel 4. 1 Pengenalan Perintah "Buka Pintu Depan" yang Dilakukan oleh Peneliti.

Perintah "Buka Pintu Depan"						
No	**Perintah**	***Pattern***	**Pergeseran**	**Hasil Pengujian**	***Error***	***Succes***
1	Buka Pintu Depan	Buka, Pintu Depan	0, 5 Karakter	*True*	0	1
2	Tolong bukakan pintu depan.	Buka, Pintu Depan	0, 5 Karakter	*True*	0	1
3	Bisakah anda membantu saya untuk membukakan pintu depan itu saya mau masuk	Buka, Pintu Depan	11, 16 Karakter	*True*	0	1
4	Ayo buka pintu depan secepatnya.	Buka, Pintu Depan	4, 9 Karakter	*True*	0	1
5	Mohon dibukakan pintu depan.	Buka, Pintu Depan	6, 11 Karakter	*True*	0	1
6	Buka pintu depan, ya!	Buka, Pintu Depan	0, 5 Karakter	*True*	0	1
7	Coba buka pintu depan untukku	Buka, Pintu Depan	5, 10 Karakter	*True*	0	1
8	Segera buka pintu depan agar 58itab isa masuk.	Buka, Pintu Depan	7, 12 Karakter	*True*	0	1
9	Hallo smart home tolonglah saya untuk membuka pintu depan	Buka, Pintu Depan	23, 28 Karakter	*True*	0	1
10	Silahkan bukain pintu depan agar angin bisa masuk.	Buka, Pintu Depan	5, 12 Karakter	*True*	0	1
11	Apakah kamu bisa membuka pintu depan untuk saya?	Buka, Pintu Depan	5, 10 Karakter	*True*	0	1
12	Saya minta kamu untuk membuka pintu depan perlahan-lahan.	Buka, Pintu Depan	11, 16 Karakter	*True*	0	1
13	Bukalah pintu depan agar saya dan keluarga bisa masuk.	Buka, Pintu Depan	0, 5 Karakter	*True*	0	1
14	Segeralah membuka pintu depan untuk menyambut tamu.	Buka, Pintu Depan	7, 12 Karakter	*True*	0	1

Perintah "Buka Pintu Depan"						
No	**Perintah**	***Pattern***	**Pergeseran**	**Hasil Pengujian**	***Error***	***Succes***
15	Tolong, bantu saya membuka pintu depan yang terkunci ini.	Buka, Kunci, Pintu Depan	6, 23, 11 Karakter	*True*	0	1
16	Saya memohon kamu untuk membuka pintu depan secepatnya.	Buka, Kunci	11, 16 Karakter	*True*	0	1
17	Bukakkan pintu depan yang terkunci, biarkan semuanya bisa masuk.	Buka, Kunci, Pintu Depan, Semua	0, 17, 5, 28 Karakter	*False*	1	0
18	Bisakah membatu saya untuk mebukakan pintu yang didepan, tolong?	Buka, Pintu Depan	11, 16 Karakter	*True*	0	1
19	Sepertinya didepan itu ada tamu, tolong untuk membukakan pintu depannya agar bisa masuk	Buka, Pintu Depan	19, 24 Karakter	*True*	0	1
20	Saya memohon kamu untuk membuka dipintu depan secepatnya.	Buka, Pintu Depan	11, 16 Karakter	*True*	0	1
∑Ei (Jumlah performa dari semua percobaan)					**1**	**9**

Pada pengujian skenario pertama yaitu pengenalan kalimat perintah "Buka Pintu Depan" seperti pada Tabel 4.1 dapat diketahui bahwasanya dari 20 kali pengujian dengan perintah berbeda yang dilakukan oleh peneliti pada sistem terdapat sembilan perintah yang *succes* dan satu perintah yang mengalami *error* yaitu pada kalimat "Bukakkan pintu depan yang terkunci, biarkan semuanya bisa masuk".

Tabel 4. 2 Pengenalan Perintah "Buka Pintu Depan" yang Dilakukan oleh 10 Orang User.

Perintah "Buka Pintu Depan"							
No	**Perintah**	***Pattern***	**Pergeseran**	**Jumlah *User***	**Hasil Pengujian**	***Jumlah Error***	***JumlahSucces***
1	Buka Pintu Depan	Buka, Pintu Depan	0, 5 Karakter	10	*True*	0	10
2	Tolong bukakan pintu depan.	Buka, Pintu Depan	0, 5 Karakter	10	*True*	0	10

Perintah "Buka Pintu Depan"							
No	**Perintah**	***Pattern***	**Pergeseran**	**Jumlah *User***	**Hasil Pengujian**	***Jumlah Error***	***JumlahSucces***
3	Bisakah anda membantu saya untuk membukakan pintu depan itu saya mau masuk	Buka, Pintu Depan	11, 16 Karakter	10	*True*	0	10
4	Ayo buka pintu depan secepatnya.	Buka, Pintu Depan	4, 9 Karakter	10	*True*	0	10
5	Mohon dibukakan pintu depan.	Buka, Pintu Depan	6, 11 Karakter	10	*True*	0	10
6	Buka pintu depan, ya!	Buka, Pintu Depan	0, 5 Karakter	10	*True*	0	10
7	Coba buka pintu depan untukku	Buka, Pintu Depan	5, 10 Karakter	10	*True*	0	10
8	Segera buka pintu depan agar kita bisa masuk.	Buka, Pintu Depan	7, 12 Karakter	10	*True*	0	10
9	Hallo smart home tolonglah saya untuk membuka pintu depan	Buka, Pintu Depan	23, 28 Karakter	10	*True*	0	10
10	Silahkan bukain pintu depan agar angin bisa masuk.	Buka, Pintu Depan	5, 12 Karakter	10	*True*	0	10
∑Ei (Jumlah performa dari semua percobaan)						**0**	**10**

Pada tabel 4.2 merupakan hasil pengujian pengenalan perintah "Buka Pintu Depan" yang dilakukan oleh 10 orang *user* dengan masing masing memberikan 10 perintah yang berbeda, diketahui bahwasanya sistem berhasil mengenali perintah dengan baik karena dari semua pengujian yang dilakukan *succes* semua atau tidak mengalami *error*. Hal ini menunjukan algoritma dapat merespon semua perintah dengan baik sehingga pintu dapat berjalan sesuai dengan harapan.

4.2.2 Pengenalan Perintah “Buka Pintu Kamar”

Proses pengujian pada skenario kedua yaitu pengenalan kalimat perintah “Buka Pintu Kamar”. Proses ini dilakukan sebanyak 120 kali yang mana 20 kali diuji oleh peneliti dan 10 kali diuji oleh 10 orang *user* dengan kalimat perintah yang berbeda. Hasil dari pengujian ini dapat dilihat pada tabel 4.3 dan tabel 4.4 dibawah ini:

Tabel 4. 3 Pengenalan Perintah "Buka Pintu Kamar" yang Dilakukan oleh Peneliti.

Perintah “Buka Pintu Kamar”						
No	**Perintah**	***Pattern***	**Pergeseran**	**Hasil Pengujian**	***Error***	***Succes***
1	Buka Pintu Kamar	Buka, Pintu Kamar	0, 5 Karakter	*True*	0	1
2	Tolong bukakan pintu kamar.	Buka, Pintu Kamar	0, 5 Karakter	*True*	0	1
3	Bisakah Anda membantu saya untuk membukakan pintu kamar itu? Saya mau masuk.	Buka, Pintu Kamar	11, 16 Karakter	*True*	0	1
4	Ayo, buka pintu kamar secepatnya.	Buka, Pintu Kamar	4, 9 Karakter	*True*	0	1
5	Mohon dibukakan pintu kamar.	Buka, Pintu Kamar	6, 11 Karakter	*True*	0	1
6	Buka pintu kamar, ya!	Buka, Pintu Kamar	0, 5 Karakter	*True*	0	1
7	Coba buka pintu kamar untukku.	Buka, Pintu Kamar	5, 10 Karakter	*True*	0	1
8	Segera buka pintu kamar agar kita bisa masuk.	Buka, Pintu Kamar	7, 12 Karakter	*True*	0	1
9	Halo smart home, tolonglah saya untuk membuka pintu kamar.	Buka, Pintu Kamar	23, 28 Karakter	*True*	0	1
10	Silakan bukain pintu kamar agar angin bisa masuk.	Buka, Pintu Kamar	5, 12 Karakter	*True*	0	1
11	Apakah kamu bisa membuka pintu kamar untuk saya?	Buka, Pintu Kamar	5, 10 Karakter	*True*	0	1

Perintah "Buka Pintu Kamar"						
No	**Perintah**	***Pattern***	**Pergeseran**	**Hasil Pengujian**	***Error***	***Succes***
12	Saya minta kamu untuk membuka pintu kamar perlahan-lahan.	Buka, Pintu Kamar	11, 16 Karakter	*True*	0	1
13	Bukalah pintu kamar agar saya dan keluarga bisa masuk.	Buka, Pintu Kamar	0, 5 Karakter	*True*	0	1
14	Segeralah membuka pintu kamar untuk menyambut tamu.	Buka, Pintu Kamar	7, 12 Karakter	*True*	0	1
15	Tolong, bantu saya membuka pintu kamar yang terkunci ini.	Buka, Kunci, Pintu Kamar	6, 23, 11 Karakter	*True*	0	1
16	Saya memohon kamu untuk membuka pintu kamar secepatnya.	Buka, Pintu Kamar	11, 16 Karakter	*True*	0	1
17	Bukakkan pintu kamar yang terkunci, biarkan semuanya bisa masuk.	Buka, Kunci, Pintu Kamar, Semua	0, 17, 5, 23 Karakter	*False*	1	0
18	Bisakah membantu saya untuk membukakan pintu kamar, tolong?	Buka, Pintu Kamar	11, 16 Karakter	*True*	0	1
19	Sepertinya di depan itu ada orang, tolong bukakan pintu kamar agar bisa masuk.	Buka, Pintu Kamar	20, 25 Karakter	*True*	0	1
20	Saya memohon kamu untuk membuka pintu kamar secepatnya.	Buka, Pintu Kamar	11, 16 Karakter	*True*	0	1
∑Ei (Jumlah performa dari semua percobaan)					**1**	**9**

Pada pengujian skenario kedua yaitu pengenalan kalimat perintah "Buka Pintu Kamar" seperti pada Tabel 4.3 dapat diketahui bahwasanya dari 20 kali pengujian dengan perintah berbeda yang dilakukan oleh peneliti pada sistem terdapat sembilan perintah yang *succes* dan satu perintah yang mengalami *error* yaitu pada kalimat "Bukakkan pintu kamar yang terkunci, biarkan semuanya bisa masuk".

Tabel 4. 4 Pengenalan Perintah "Buka Pintu Kamar" yang Dilakukan oleh 10 Orang User.

Perintah "Buka Pintu Kamar"							
No	**Perintah**	***Pattern***	**Pergeseran**	**Jumlah *User***	**Hasil Pengujian**	***Jumlah Error***	***Jumlah Succes***
1	Apakah kamu bisa membuka pintu kamar untuk saya?	Buka, Pintu Kamar	5, 10 Karakter	10	*True*	0	1
2	Saya minta kamu untuk membuka pintu kamar perlahan-lahan.	Buka, Pintu Kamar	11, 16 Karakter	10	*True*	0	1
3	Bukalah pintu kamar agar saya dan keluarga bisa masuk.	Buka, Pintu Kamar	0, 5 Karakter	10	*True*	0	1
4	Segeralah membuka pintu kamar untuk menyambut tamu.	Buka, Pintu Kamar	7, 12 Karakter	10	*True*	0	1
5	Tolong, bantu saya membuka pintu kamar yang terkunci ini.	Buka, Kunci, Pintu Kamar	6, 23, 11 Karakter	10	*True*	0	1
6	Saya memohon kamu untuk membuka pintu kamar secepatnya.	Buka, Pintu Kamar	11, 16 Karakter	10	*True*	0	1
7	Bukakkan pintu kamar yang terkunci, biarkan semuanya bisa masuk.	Buka, Kunci, Pintu Kamar, Semua	0, 17, 5, 23 Karakter	10	*False*	10	0
8	Bisakah membantu saya untuk membukakan pintu kamar, tolong?	Buka, Pintu Kamar	11, 16 Karakter	10	*True*	0	1
9	Sepertinya di depan itu ada orang, tolong bukakan pintu kamar agar bisa masuk.	Buka, Pintu Kamar	20, 25 Karakter	10	*True*	0	1

Perintah “Buka Pintu Kamar”							
No	**Perintah**	***Pattern***	**Pergeseran**	**Jumlah *User***	**Hasil Pengujian**	***Jumlah Error***	***Jumlah Succes***
10	Saya memohon kamu untuk membuka pintu kamar secepatnya.	Buka, Pintu Kamar	11, 16 Karakter	10	*True*	0	1
∑Ei (Jumlah performa dari semua percobaan)						**10**	**90**

Pada tabel 4.4 merupakan hasil pengujian pengenalan perintah “Buka Pintu Kamar” yang dilakukan oleh 10 orang *user* dengan masing masing memberikan 10 perintah yang berbeda, diketahui bahwasanya sistem *succes* merespon 90 perintah dan mengalami *error* dalam pengenalan perintah sebanyak 10 kali dalam 1 kalimat perintah yang sama yaitu pada kalimat “Bukakkan pintu kamar yang terkunci, biarkan semuanya bisa masuk”.

4.2.3 Pengenalan Perintah “Kunci Pintu Depan”

Proses pengujian pada skenario ketiga yaitu pengenalan kalimat perintah “Kunci Pintu Depan”. Proses ini dilakukan sebanyak 120 kali yang mana 20 kali diuji oleh peneliti dan 10 kali diuji oleh 10 orang *user* dengan kalimat perintah yang berbeda. Hasil dari pengujian ini dapat dilihat pada tabel 4.5 dan tabel 4.6 dibawah ini:

Tabel 4. 5 Perintah "Kunci Pintu Depan" yang Dilakukan oleh Peneliti.

Perintah “Kunci Pintu Depan”						
No	**Perintah**	***Pattern***	**Pergeseran**	**Hasil Pengujian**	***Error***	***Succes***
1	Kunci Pintu Depan	Kunci, Pintu Depan	0, 6 Karakter	*True*	0	1
2	Tolong kunci pintu depan.	Kunci, Pintu Depan	0, 6 Karakter	*True*	0	1

Perintah "Kunci Pintu Depan"						
No	**Perintah**	***Pattern***	**Pergeseran**	**Hasil Pengujian**	***Error***	***Succes***
3	Bisakah Anda membantu saya untuk menguncikan pintu depan itu? Saya mau keluar.	Kunci, Pintu Depan	11, 17 Karakter	*True*	0	1
4	Ayo, kunci pintu depan secepatnya.	Kunci, Pintu Depan	4, 10 Karakter	*True*	0	1
5	Mohon dikuncikan pintu depan.	Kunci, Pintu Depan	6, 12 Karakter	*True*	0	1
6	Kunci pintu depan, ya!	Kunci, Pintu Depan	0, 5 Karakter	*True*	0	1
7	Coba kunci pintu depan untukku.	Kunci, Pintu Depan	5, 11 Karakter	*True*	0	1
8	Segera kunci pintu depan agar kita bisa keluar dengan tenang.	Kunci, Pintu Depan	7, 13 Karakter	*True*	0	1
9	Halo smart home, tolonglah saya untuk mengunci pintu depan.	Kunci, Pintu Depan	23, 29 Karakter	*True*	0	1
10	Silakan kunci pintu depan agar angin tidak masuk.	Kunci, Pintu Depan	5, 11 Karakter	*True*	0	1
11	Apakah kamu bisa mengunci pintu depan untuk saya?	Kunci, Pintu Depan	5, 11 Karakter	*True*	0	1
12	Saya minta kamu untuk mengunci pintu depan perlahan-lahan.	Kunci, Pintu Depan	11, 17 Karakter	*True*	0	1
13	Kuncilah pintu depan agar saya dan keluarga bisa beristirahat.	Kunci, Pintu Depan	0, 6 Karakter	*True*	0	1
14	Segeralah mengunci pintu depan untuk menjaga kenyamanan rumah.	Kunci, Pintu Depan	7, 13 Karakter	*True*	0	1
15	Tolong, bantu saya mengunci pintu depan soalnya saya sudah berada didalam rumah.	Kunci, Pintu Depan	6, 12 Karakter	*True*	0	1
16	Saya memohon kamu untuk mengunci pintu depan secepatnya.	Kunci, Pintu Depan	11, 17 Karakter	*True*	0	1
17	Kuncikan pintu depan yang terbuka, biarkan semuanya ada didalam rumah	Buka, Kunci, Pintu Depan, Semua	22, 0, 11, 32 Karakter	*False*	1	0

Perintah "Kunci Pintu Depan"						
No	**Perintah**	***Pattern***	**Pergeseran**	**Hasil Pengujian**	***Error***	***Succes***
18	Bisakah membantu saya untuk menguncikan pintu depan, tolong?	Kunci, Pintu Depan	11, 21 Karakter	*True*	0	1
19	Kuncilah pintu depan supaya kita bisa merasa nyaman	Kunci, Pintu Depan	0, 6 Karakter	*True*	0	1
20	Saya memohon kamu untuk mengunci pintu depan secepatnya.	Kunci, Pintu Depan	11, 17 Karakter	*True*	0	1
∑Ei (Jumlah performa dari semua percobaan)					**1**	**9**

Pada pengujian skenario ketiga yaitu pengenalan kalimat perintah "Kunci Pintu Depan" seperti pada Tabel 4.5 dapat diketahui bahwasanya dari 20 kali pengujian dengan kalimat perintah berbeda yang dilakukan oleh peneliti pada sistem terdapat sembilan perintah yang *succes* dan satu perintah yang mengalami *error* yaitu pada kalimat "Kuncikan pintu depan yang terbuka, biarkan semuanya ada didalam rumah".

Tabel 4. 6 Pengenalan Perintah "Kunci Pintu Depan" yang Dilakukan oleh 10 Orang User.

Perintah "Kunci Pintu Depan"							
No	**Perintah**	***Pattern***	**Pergeseran**	**Jumlah *User***	**Hasil Pengujian**	***Jumlah Error***	***Jumlah Succes***
1	Kunci Pintu Depan	Kunci, Pintu Depan	0, 6 Karakter	10	*True*	0	10
2	Tolong kunci pintu depan.	Kunci, Pintu Depan	0, 6 Karakter	10	*True*	0	10
3	Bisakah Anda membantu saya untuk menguncikan pintu depan itu? Saya mau keluar.	Kunci, Pintu Depan	11, 17 Karakter	10	*True*	0	10
4	Ayo, kunci pintu depan secepatnya.	Kunci, Pintu Depan	4, 10 Karakter	10	*True*	0	10
5	Mohon dikuncikan pintu depan.	Kunci, Pintu Depan	6, 12 Karakter	10	*True*	0	10

Perintah "Kunci Pintu Depan"							
No	**Perintah**	***Pattern***	**Pergeseran**	**Jumlah *User***	**Hasil Pengujian**	***Jumlah Error***	***Jumlah Succes***
6	Kunci pintu depan, ya!	Kunci, Pintu Depan	0, 5 Karakter	10	*True*	0	10
7	Coba kunci pintu depan untukku.	Kunci, Pintu Depan	5, 11 Karakter	10	*True*	0	10
8	Segera kunci pintu depan agar kita bisa keluar dengan tenang.	Kunci, Pintu Depan	7, 13 Karakter	10	*True*	0	10
9	Halo smart home, tolonglah saya untuk mengunci pintu depan.	Kunci, Pintu Depan	23, 29 Karakter	10	*True*	0	10
10	Silakan kunci pintu depan agar angin tidak masuk.	Kunci, Pintu Depan	5, 11 Karakter	10	*True*	0	10
∑Ei (Jumlah performa dari semua percobaan)						**0**	**100**

Pada tabel 4.6 merupakan hasil pengujian pengenalan perintah "Kunci Pintu Depan" yang dilakukan oleh 10 orang *user* dengan masing masing memberikan 10 perintah yang berbeda, diketahui bahwasanya sistem *succes* dalam merespon semua perinta atau tidak mengalami *error.* Hal ini menunjukan algoritma dapat merespon semua perintah dengan baik sehingga pintu dapat berjalan sesuai dengan harapan.

4.2.4 Pengenalan Perintah "Kunci Pintu Kamar"

Proses pengujian pada skenario keempat yaitu pengenalan kalimat perintah "Kunci Pintu Kamar". Proses ini dilakukan sebanyak 120 kali yang mana 20 kali diuji oleh peneliti dan 10 kali diuji oleh 10 orang *user* dengan kalimat perintah yang

berbeda. Hasil dari pengujian ini dapat dilihat pada tabel 4.7 dan tabel 4.8 dibawah ini:

Tabel 4. 7 Perintah "Kunci Pintu Kamar" yang Dilakukan oleh Peneliti.

Perintah "Kunci Pintu Kamar"						
No	**Perintah**	***Pattern***	**Pergeseran**	**Hasil Pengujian**	***Error***	***Succes***
1	Kunci Pintu Kamar	Kunci, Pintu Kamar	0, 6 Karakter	*True*	0	1
2	Tolong kunci pintu kamar.	Kunci, Pintu Kamar	0, 6 Karakter	*True*	0	1
3	Bisakah Anda membantu saya untuk menguncikan pintu kamar itu? Saya mau tidur.	Kunci, Pintu Kamar	11, 21 Karakter	*True*	0	1
4	Ayo, kunci pintu kamar saya secepatnya.	Kunci, Pintu Kamar	4, 10 Karakter	*True*	0	1
5	Mohon dikunci kan pintu kamar.	Kunci, Pintu Kamar	6, 16 Karakter	*True*	0	1
6	Kunci pintu kamar, ya!	Kunci, Pintu Kamar	0, 6 Karakter	*True*	0	1
7	Coba kunci pintu kamar untukku.	Kunci, Pintu Kamar	5, 11 Karakter	*True*	0	1
8	Segera kunci pintu kamar agar kita bisa keluar dengan tenang.	Kunci, Pintu Kamar	7, 13 Karakter	*True*	0	1
9	Halo smart home, tolonglah saya untuk mengunci pintu kamar.	Kunci, Pintu Kamar	23, 29 Karakter	*True*	0	1
10	Silakan kunci pintu kamar agar debu tidak masuk.	Kunci, Pintu Kamar	5, 11 Karakter	*True*	0	1
11	Apakah kamu bisa mengunci pintu kamar untuk saya?	Kunci, Pintu Kamar	5, 11 Karakter	*True*	0	1
12	Saya minta kamu untuk mengunci pintu kamar perlahan-lahan.	Kunci, Pintu Kamar	11, 17 Karakter	*True*	0	1
13	Kuncilah pintu kamar agar saya dan teman saya bisa beristirahat.	Kunci, Pintu Kamar	0, 6 Karakter	*True*	0	1
14	Segeralah mengunci pintu kamar untuk menjaga kenyamanan rumah.	Kunci, Pintu Kamar	7, 13 Karakter	*True*	0	1

Perintah "Kunci Pintu Kamar"						
No	**Perintah**	***Pattern***	**Pergeseran**	**Hasil Pengujian**	***Error***	***Succes***
15	Tolong, bantu saya mengunci pintu kamar soalnya saya sudah berada didalam kamar.	Kunci, Pintu Kamar	6, 12 Karakter	*True*	0	1
16	Saya memohon kamu untuk mengunci pintu kamar secepatnya.	Kunci, Pintu Kamar	11, 17 Karakter	*True*	0	1
17	Kunci kan pintu kamar yang terbuka, biarkan semuanya ada didalam kamar.	Buka, Kunci, Pintu Kamar, Semua	22, 0, 11, 32 Karakter	*False*	1	0
18	Bisakah membantu saya untuk menguncikan pintu kamar, tolong?	Kunci, Pintu Depan	11, 17 Karakter	*True*	0	1
19	Ternyata mau di luar sudah masuk semua tolong kunci pintu kamar agar aman dan nyaman	Kunci, Pintu Kamar, Semua	27, 33 Karakter	*True*	0	1
20	Saya memohon kamu untuk mengunci pintu kamar secepatnya.	Kunci, Pintu Kamar	11, 17 Karakter	*True*	0	1
∑Ei (Jumlah performa dari semua percobaan)					**1**	**9**

Pada pengujian skenario keempat yaitu pengenalan kalimat perintah "Kunci Pintu Kamar" seperti pada Tabel 4.7 dapat diketahui bahwasanya dari 20 kali pengujian dengan perintah berbeda yang dilakukan oleh peneliti pada sistem terdapat sembilan kalimat perintah yang *succes* merespon dengan baik dan terdapat satu kalimat perintah yang mengalami *error* yaitu pada kalimat ke tujuh belas yang berbunyi "Kunci kan pintu kamar yang terbuka, biarkan semuanya ada didalam kamar".

Tabel 4. 8 Pengenalan Perintah "Kunci Pintu Kamar" yang Dilakukan oleh 10 Orang User.

Perintah "Kunci Pintu Kamar"							
No	**Perintah**	***Pattern***	**Pergeseran**	**Jumlah *User***	**Hasil Pengujian**	**Jumlah *Error***	**Jumlah *Succes***
1	Apakah kamu bisa mengunci pintu kamar untuk saya?	Kunci, Pintu Kamar	5, 11 Karakter	10	*True*	0	10

Perintah "Kunci Pintu Kamar"							
No	**Perintah**	***Pattern***	**Pergeseran**	**Jumlah *User***	**Hasil Pengujian**	**Jumlah *Error***	**Jumlah *Succes***
2	Saya minta kamu untuk mengunci pintu kamar perlahan-lahan.	Kunci, Pintu Kamar	11, 17 Karakter	10	*True*	0	10
3	Kuncilah pintu kamar agar saya dan teman saya bisa beristirahat.	Kunci, Pintu Kamar	0, 6 Karakter	10	*True*	0	10
4	Segeralah mengunci pintu kamar untuk menjaga kenyamanan rumah.	Kunci, Pintu Kamar	7, 13 Karakter	10	*True*	0	10
5	Tolong, bantu saya mengunci pintu kamar soalnya saya sudah berada didalam kamar.	Kunci, Pintu Kamar	6, 12 Karakter	10	*True*	0	10
6	Saya memohon kamu untuk mengunci pintu kamar secepatnya.	Kunci, Pintu Kamar	11, 17 Karakter	10	*True*	0	10
7	Kunci kan pintu kamar yang terbuka, biarkan semuanya ada didalam kamar.	Buka, Kunci, Pintu Kamar, Semua	22, 0, 11, 32 Karakter	10	*False*	10	0
8	Bisakah membantu saya untuk menguncikan pintu kamar, tolong?	Kunci, Pintu Depan	11, 17 Karakter	10	*True*	0	10
9	Ternyata mau di luar sudah masuk semua tolong kunci pintu kamar agar aman dan nyaman	Kunci, Pintu Kamar, Semua	27, 33 Karakter	10	8 *True*, 2 *False*	2	8
10	Saya memohon kamu untuk mengunci pintu kamar secepatnya.	Kunci, Pintu Kamar	11, 17 Karakter	10	*True*	0	10
∑Ei (Jumlah performa dari semua percobaan)						**12**	**88**

Pada tabel 4.8 merupakan hasil pengujian pengenalan perintah "Kunci Pintu Kamar" yang dilakukan oleh 10 orang *user* dengan masing masing memberikan 10

perintah yang berbeda, diketahui bahwasanya sistem *succes* merespon kalimat perintah sebanyak 88 kali dan sistem mengalami 12 *error* yaitu 10 kali *error* pada kalimat “Kunci kan pintu kamar yang terbuka, biarkan semuanya ada didalam kamar” dan 2 kali *error* pada kalimat “Ternyata mau di luar sudah masuk semua tolong kunci pintu kamar agar aman dan nyaman”.

4.2.5 Pengenalan Perintah “Buka Semua Pintu”

Proses pengujian pada skenario kelima yaitu pengenalan kalimat perintah “Buka Semua Pintu”. Proses ini dilakukan sebanyak 110 kali yang mana 10 kali diuji oleh peneliti dan 10 kali diuji oleh 10 orang *user* dengan kalimat perintah yang berbeda. Hasil dari pengujian ini dapat dilihat pada tabel 4.9 dan tabel 4.10 dibawah ini:

Tabel 4. 9 Perintah "Buka Semua Pintu" yang Dilakukan oleh Peneliti.

Perintah “Buka Semua Pintu”						
No	**Perintah**	***Pattern***	**Pergeseran**	**Hasil Pengujian**	***Error***	***Succes***
1.	Buka Semua Pintu	Buka, Semua	0, 5 Karakter	*True*	0	1
2.	Bisakah kamu membukakan semua pintu yaitu kamar dan depan untukku?	Buka, Pintu Kamar, Semua	10, 21, 15 Karakter	*True*	0	1
3.	Buka semua pintu yang terkunci, biarkan semuanya terbuka.	Buka, Kunci, Semua	0, 17 Karakter	*True*	0	1
4.	Bukalah semua pintu agar semua orang bisa masuk dengan nyaman.	Buka, Semua	0, 5 Karakter	*True*	0	1
5.	Mohon dibukakan semua pintu.	Buka, Semua	6, 12 Karakter	*True*	0	1
6.	Ayo, bukalah semua pintu yang terkunci supaya kita tidak terjebak di dalam.	Buka, Kunci, Semua	4, 21, 9 Karakter	*True*	0	1
7.	Coba semua pintu dibuka agar cahaya matahari masuk ke dalam ruangan.	Buka, Semua	17, 15 Karakter	*True*	0	1
8.	Saya ingin bersih bersih rumah bukakkan semua pintu yang ada dirumah	Buka, Semua	13, 18 Karakter	*True*	0	1

Perintah “Buka Semua Pintu”						
No	**Perintah**	***Pattern***	**Pergeseran**	**Hasil Pengujian**	***Error***	***Succes***
9.	Halo smart home, tolonglah saya untuk membuka semua pintu kamar dan depan	Buka, Pintu Kamar, Semua	23, 34, 28 Karakter	*True*	0	1
10.	Saya mohon, segera buka semua pintu agar kita bisa melanjutkan kegiatan.	Buka, Semua	13, 18 Karakter	*True*	0	1
∑Ei (Jumlah performa dari semua percobaan)					**0**	**10**

Pada tabel 4.9 merupakan hasil pengujian pengenalan perintah “Buka Semua Pintu” yang dilakukan oleh peneliti dengan memberikan 10 perintah yang berbeda, diketahui bahwasanya sistem *succes* merespon semua kalimat perintah atau sistem tidak mengalami *error*. Hal ini menunjukan algoritma dapat merespon semua perintah dengan baik sehingga pintu dapat berjalan sesuai dengan harapan.

Tabel 4. 10 Pengenalan Perintah "Buka Semua Pintu" yang Dilakukan oleh 10 Orang User.

Perintah “Buka Semua Pintu”							
No	**Perintah**	***Pattern***	**Pergeseran**	**Jumlah *User***	**Hasil Pengujian**	**Jumlah *Error***	**Jumlah *Succes***
	Buka Semua Pintu	Buka, Semua	0, 5 Karakter	10	*True*	0	10
2.	Bisakah kamu membukakan semua pintu yaitu kamar dan depan untukku?	Buka, Pintu Kamar, Semua	10, 21, 15 Karakter	10	*True*	0	10
3.	Buka semua pintu yang terkunci, biarkan semuanya terbuka.	Buka, Kunci, Semua	0, 17 Karakter	10	*True*	0	10
4.	Bukalah semua pintu agar semua orang bisa masuk dengan nyaman.	Buka, Semua	0, 5 Karakter	10	*True*	0	10
5.	Mohon dibukakan semua pintu.	Buka, Semua	6, 12 Karakter	10	*True*	0	10

Perintah "Buka Semua Pintu"							
No	**Perintah**	***Pattern***	**Pergeseran**	**Jumlah *User***	**Hasil Pengujian**	**Jumlah *Error***	**Jumlah *Succes***
6.	Ayo, bukalah semua pintu yang terkunci supaya kita tidak terjebak di dalam.	Buka, Kunci, Semua	4, 21, 9 Karakter	10	*True*	0	10
7.	Coba semua pintu dibuka agar cahaya matahari masuk ke dalam ruangan.	Buka, Semua	17, 15 Karakter	10	*True*	0	10
8.	Saya ingin bersih bersih rumah bukakkan semua pintu yang ada dirumah	Buka, Semua	13, 18 Karakter	10	*True*	0	10
9.	Halo smart home, tolonglah saya untuk membuka semua pintu kamar dan depan	Buka, Pintu Kamar, Semua	23, 34, 28 Karakter	10	*True*	0	10
10.	Saya mohon, segera buka semua pintu agar kita bisa melanjutkan kegiatan.	Buka, Semua	13, 18 Karakter	10	*True*	0	10
∑Ei (Jumlah performa dari semua percobaan)						**0**	**100**

Pada tabel 4.10 merupakan hasil pengujian pengenalan perintah "Buka Semua Pintu" yang dilakukan oleh 10 orang *user* dengan masing masing memberikan 10 perintah yang berbeda, diketahui bahwasanya sistem *succes* merespon semua kalimat perintah atau sistem tidak mengalami *error*. Hal ini menunjukan algoritma dapat merespon semua perintah dengan baik sehingga pintu dapat berjalan sesuai dengan harapan.

4.2.6 Pengenalan Perintah “Kunci Semua Pintu”

Proses pengujian pada skenario keenam yaitu pengenalan kalimat perintah “Kunci Semua Pintu”. Proses ini dilakukan sebanyak 110 kali yang mana 10 kali diuji oleh peneliti dan 10 kali diuji oleh 10 orang *user* dengan kalimat perintah yang berbeda. Hasil dari pengujian ini dapat dilihat pada tabel 4.1 dan tabel 4.12 dibawah ini:

Tabel 4. 11 Perintah "Kunci Semua Kamar" yang Dilakukan oleh Peneliti.

Perintah “Kunci Semua Pintu”						
No	**Perintah**	***Pattern***	**Pergeseran**	**Hasil Pengujian**	***Error***	***Succes***
1.	Kunci Semua Pintu	Kunci, Semua	0, 6 Karakter	*True*	0	1
2.	Bisakah kamu menguncikan semua pintu yaitu kamar dan depan untukku?	Kunci, Pintu Kamar Semua	10, 22, 16 Karakter	*True*	0	1
3.	Kunci semua pintu yang terbuka, biarkan semuanya aman terkendali.	Buka, Kunci, Semua	18, 0, 6 Karakter	*False*	1	0
4.	Kuncilah semua pintu agar semua orang merasa aman dan nyaman didalam rumah saya.	Kunci, Semua	0, 6 Karakter	*True*	0	1
5.	Mohon dikunci kan semua pintu.	Kunci, Semua	6, 16 Karakter	*True*	0	1
6.	Ayo, kuncilah semua pintu yang terbuka supaya kita tidak terjebak diluar.	Buka, Kunci, Semua	22, 4, 10 Karakter	*False*	1	0
7.	Coba semua pintu yang terbuka dikunci agar tidak ada debu masuk ke dalam ruangan.	Buka, Kunci, Semua	17, 22, 5 Karakter	False	1	0
8.	Saya ingin beristirahat didalam rumah kuncikan semua pintu yang ada dirumah	Kunci, Semua	16, 26 Karakter	*True*	0	1
9.	Halo smart home, tolonglah saya untuk	Kunci, Pintu	23, 35, 29 Karakter	*True*	0	1

Perintah "Kunci Semua Pintu"						
No	**Perintah**	***Pattern***	**Pergeseran**	**Hasil Pengujian**	***Error***	***Succes***
	mengkuncikan semua pintu kamar dan depan	Kamar, Semua				
10.	Saya mohon, segera kunci semua pintu agar kita bisa melanjutkan kegiatan didalam rumah.	Kunci, Semua	13, 19 Karakter	*True*	0	1
∑Ei (Jumlah kesalahan dari semua percobaan)					**3**	**7**

Pada tabel 4.11 merupakan hasil pengujian pengenalan perintah "Kunci Semua Pintu" yang dilakukan oleh peneliti dengan memberikan 10 perintah yang berbeda, diketahui bahwasanya sistem *succes* merespon tujuh kalimat perintah dan mengalami tiga kali *error* pada kalimat "Kunci semua pintu yang terbuka, biarkan semuanya aman terkendali", "Ayo, kuncilah semua pintu yang terbuka supaya kita tidak terjebak diluar", dan "Coba semua pintu yang terbuka dikunci agar tidak ada debu masuk ke dalam ruangan".

Tabel 4. 12 Pengenalan Perintah "Kunci Semua Pintu" yang Dilakukan oleh 10 Orang User.

Perintah "Kunci Semua Pintu"							
No	**Perintah**	***Pattern***	**Pergeseran**	**Jumlah *User***	**Hasil Pengujian**	***Jumlah Error***	**Jumlah *Succes***
1.	Kunci Semua Pintu	Kunci, Semua	0, 6 Karakter	10	*True*	0	10
2.	Bisakah kamu menguncikan semua pintu yaitu kamar dan depan untukku?	Kunci, Pintu Kamar Semua	10, 22, 16 Karakter	10	*True*	9	1
3.	Kunci semua pintu yang terbuka, biarkan semuanya aman terkendali.	Buka, Kunci, Semua	18, 0, 6 Karakter	10	*False*	10	0
4.	Kuncilah semua pintu agar semua orang merasa aman dan nyaman didalam rumah saya.	Kunci, Semua	0, 6 Karakter	10	*True*	0	10
5.	Mohon dikunci kan semua pintu.	Kunci, Semua	6, 16 Karakter	10	*True*	0	10

Perintah "Kunci Semua Pintu"							
No	**Perintah**	***Pattern***	**Pergeseran**	**Jumlah *User***	**Hasil Pengujian**	***Jumlah Error***	**Jumlah *Succes***
6.	Ayo, kuncilah semua pintu yang terbuka supaya kita tidak terjebak diluar.	Buka, Kunci, Semua	22, 4, 10 Karakter	10	1 *True*, 2 *False*	9	1
7.	Coba semua pintu yang terbuka dikunci agar tidak ada debu masuk ke dalam ruangan.	Buka, Kunci, Semua	17, 22, 5 Karakter	10	1 *True*, 2 *False*	9	1
8.	Saya ingin beristirahat didalam rumah kuncikan semua pintu yang ada dirumah	Kunci, Semua	16, 26 Karakter	10	*True*	0	10
9.	Halo smart home, tolonglah saya untuk mengkuncikan semua pintu kamar dan depan	Kunci, Pintu Kamar, Semua	23, 35, 29 Karakter	10	*True*	0	10
10.	Saya mohon, segera kunci semua pintu agar kita bisa melanjutkan kegiatan didalam rumah.	Kunci, Semua	13, 19 Karakter	10	*True*	0	10
∑Ei (Jumlah performa dari semua percobaan)						**28**	**72**

Pada tabel 4.12 merupakan hasil pengujian pengenalan perintah "Kunci Semua Pintu" yang dilakukan oleh 10 orang *user* dengan masing masing memberikan 10 perintah yang berbeda, diketahui bahwasanya sistem *succes* merespon sebanyak 72 kalimat perintah dan sistem mengalami 28 *error* yaitu 10 kali *error* pada kalimat "Kunci semua pintu yang terbuka, biarkan semuanya aman terkendali", 9 kali *error* pada kalimat "Bisakah kamu menguncikan semua pintu yaitu kamar dan depan untukku?", dan 9 kali *error* pada kalimat "Coba semua pintu yang terbuka dikunci agar tidak ada debu masuk ke dalam ruangan".

4.3 Hasil Pengujian Sistem

Pada tahapan pengujian performa dilakukan dengan menjalankan sistem sebanyak 700 kali percobaan dengan akusisi 120 kali untuk pengenalan perintah "Buka Pintu Depan", 120 Kali untuk pengenalan "Buka Pintu Kamar", 120 kali untuk pengenalan perintah "Kunci Pintu Depan", 120 kali untuk pengenalan "Kunci Pintu Kamar", 110 kali untuk pengenalan perintah "Buka Semua Pintu" dan 110 kali untuk pengenalan perintah "Kunci Semua Pintu". Sistem yang diuji coba akan dihitung performa dalam merespon dengan menggunakan persamaan dibawah ini.

$$\mu = \frac{\sum Ei}{n} \text{ X } 100\% \tag{4.1}$$

Tabel 4. 13 Hasil Pengujian Sistem

No	Pengenalan Perintah	Diujikan Oleh	Jumlah Uji Coba	*Error*	*Succes*
1.	Buka Pintu Depan	Peneliti	20 kali	1	19
		10 Orang *User*	100 kali	0	100
2.	Buka Pintu Kamar	Peneliti	20 kali	1	19
		10 Orang *User*	100 kali	10	90
3.	Kunci Pintu Depan	Peneliti	20 kali	1	19
		10 Orang *User*	100 kali	0	100
4.	Kunci Pintu Kamar	Peneliti	20 kali	1	19
		10 Orang *User*	100 kali	12	88
5.	Buka Semua Pintu	Peneliti	10 kali	0	19
		10 Orang *User*	100 kali	0	10
6.	Kunci Semua Pintu	Peneliti	10 kali	3	17
		10 Orang *User*	100 kali	28	72
Total Uji Coba			**700 kali**	**57**	**643**

Pada Tabel 4.13 dapat diketahui bahwa dalam 700 percobaan sistem *success* merespon sebanyak 643 kali dan terdapat 57 *error*. Adapun perhitungan performa dari pengujian sistem dengan 643 kali *succes* adalah sebagai berikut:

$$\mu = \frac{\sum Ei}{n} \text{ X } 100\%$$

$$\mu = \frac{643}{700} \text{ X } 100\% \qquad (4.2)$$

$$\mu = 91.86\%$$

Dapat dilihat bahwasanya performa *Naive Pattern Algoritm* dalam pengenalan perintah sebesar 91.86% dan sisa dari angka tersebut merupakan *error* dari sistem. Adapun perhitungan *error* menggunakan persamaan dari pengujian sistem dengan 57 kali *error* adalah sebagai berikut:

$$\mu = \frac{\sum Ei}{n} \text{ X } 100\%$$

$$\mu = \frac{57}{700} \text{ X } 100\% \qquad (4.3)$$

$$\mu = 8.14\%$$

Pada tabel 4.13 dapat diketahui bahwasanya sistem *succes* merespon perintah sebanyak 643 degan presentase 91.86% dan terdapat 57 kali *error* dengan presentasi 8.14% dalam 700 kali pengujian. *Error* ini disebabkan karena terdapat lebih dari satu *pattern* yang dikenali oleh *Naive Pattern Algoritm.* Hal tersebut menyebabkan tidak dikenalinya perintah keputusan dengan baik sehingga sistem hardware tidak dapat merespon perintah dengan baik.

4.4 Intregrasi Islam

Sistem kendali kunci pintu rumah berbasisis *Internet of Things (IoT)* dengan *Naive Pattern Algoritm* dirancang untuk membantu sesama khususnya untuk para penyandang disabilitas tuna netra agar mereka dapat beraktivitas dengan nyaman

dan mudah. Hal tersebut sesuai dengan perintah Allah yang telah difirmankan dalam kitab suci Al-Qur'an pada surah At-Taubah Ayat 71 yang berbunyi:

وَالْمُؤْمِنُوْنَ وَالْمُؤْمِنٰتُ بَعْضُهُمْ اَوْلِيَآءُ بَعْضٍۘ يَأْمُرُوْنَ بِالْمَعْرُوْفِ وَيَنْهَوْنَ عَنِ الْمُنْكَرِ وَيُقِيْمُوْنَ الصَّلٰوةَ وَيُؤْتُوْنَ الزَّكٰوةَ وَيُطِيْعُوْنَ اللّٰهَ وَرَسُوْلَهٗۗ اُولٰۤىِٕكَ سَيَرْحَمُهُمُ اللّٰهُۗ اِنَّ اللّٰهَ عَزِيْزٌ حَكِيْمٌ ﴿٧١﴾

"Orang-orang mukmin, laki-laki dan perempuan, sebagian mereka menjadi penolong bagi sebagian yang lain. Mereka menyuruh (berbuat) makruf dan mencegah (berbuat) mungkar, menegakkan salat, menunaikan zakat, dan taat kepada Allah dan Rasul-Nya. Mereka akan diberi rahmat oleh Allah. Sesungguhnya Allah Maha perkasa lagi Maha bijaksana." (QS: At-Taubah: 71).

Menurut tafsir Jalalain pada ayat ketujuh puluh satu dalam surah At-Taubah ini merupakan perintah dari Allah kepada orang orang yang beriman baik dari laki laki maupun Perempuan untuk saling tolong menolong dalam kebaikan dan mencegah perbuatan buruk, melaksanakan salat, menunaikan zakat, dan menaati Allah serta Rasul-Nya. Mereka akan dianugerahi rahmat oleh Allah. Sesungguhnya Allah Mahakuasa, sehingga tidak ada yang dapat menghalangi janji atau ancaman-Nya, dan Dia selalu bertindak dengan kebijaksanaan, menempatkan segala sesuatu pada posisi yang tepat (Al-Mahalli & Jalaluddin, 2006). Firman Allah dalam surah At-Taubah ayat 71 relevan dengan penelitian ini karena sistem pada penelitian ini dilatar belakangi untuk membantu tuna netra agar mereka dapat beraktivitas dengan nyaman dan bebas.

Sistem ini dikembangkan dengan teknologi modern baik dari perangkat lunak maupun perangkat kerasnya. Sistem yang canggih dengan menggunakan teknologi modern ini tentunya tidak lepas dari kebesaran dan kuasa Allah SWT yang telah memberikan manusia akal untuk berfikir. Teknologi seharusnya tidak hanya digunakan untuk kemajuan duniawi, tetapi juga diarahkan untuk

kemaslahatan umat manusia, dengan tujuan yang sejalan dengan nilai-nilai Islam. Allah SWT telah berfirman dalam Al-Qur'an yang mengajak umat untuk berpikir tentang tanda-tanda kebesaran-Nya, yang dapat dijadikan landasan dalam pengembangan teknologi dengan memperhatikan manfaatnya bagi umat manusia. Sebagaimana firman-Nya dalam surah Al-Baqarah ayat 164 yang berbunyi:

اِنَّ فِيْ خَلْقِ السَّمٰوٰتِ وَالْاَرْضِ وَاخْتِلَافِ الَّيْلِ وَالنَّهَارِ وَالْفُلْكِ الَّتِيْ تَجْرِيْ فِى الْبَحْرِ بِمَا يَنْفَعُ النَّاسَ وَمَآ اَنْزَلَ اللّٰهُ مِنَ السَّمَآءِ مِنْ مَّآءٍ فَاَحْيَا بِهِ الْاَرْضَ بَعْدَ مَوْتِهَا وَبَثَّ فِيْهَا مِنْ كُلِّ دَآبَّةٍ ۖ وَّتَصْرِيْفِ الرِّيٰحِ وَالسَّحَابِ الْمُسَخَّرِ بَيْنَ السَّمَآءِ وَالْاَرْضِ لَاٰيٰتٍ لِّقَوْمٍ يَّعْقِلُوْنَ ﴿١٦٤﴾

“Sesungguhnya dalam penciptaan langit dan bumi, silih bergantinya malam dan siang, bahtera yang berlayar di laut membawa apa yang berguna bagi manusia, dan apa yang Allah turunkan dari langit berupa air, lalu dengan air itu Dia hidupkan bumi sesudah mati (kering)-nya dan Dia sebarkan di bumi itu segala jenis hewan, dan pengisaran angin dan awan yang dikendalikan antara langit dan bumi; sungguh (terdapat) tanda-tanda (keesaan dan kebesaran Allah) bagi kaum yang memikirkan.” .” (QS: Al-Baqarah: 164)

Menurut tafsir Jalalain mengenai firman Allah dalam surah Al-Baqarah ayat 164 ini menunjukan bahwa sesungguhnya pada penciptaan langit dan bumi, terdapat keajaiban-keajaiban yang menakjubkan, begitu pula dengan pergantian malam dan siang yang datang dan pergi, bertambah dan berkurang. Begitu juga dengan kapal-kapal yang berlayar di lautan tanpa tenggelam atau terhenti, membawa barang-barang yang bermanfaat bagi manusia seperti perdagangan dan angkutan. Allah menurunkan hujan dari langit yang menghidupkan bumi, menyuburkan tanaman setelah kekeringan, dan menyebarkan berbagai jenis hewan yang berkembang biak dengan tumbuhan yang ada. Angin yang berhembus mengalirkan udara ke utara atau selatan, mengubah suhu menjadi panas atau dingin, dan awan yang dikendalikan oleh perintah Allah, bergerak sesuai kehendak-Nya

antara langit dan bumi. Semua ini merupakan tanda-tanda kebesaran Allah yang dapat dilihat dan direnungkan oleh orang-orang yang berpikir (Al-Mahalli & Jalaluddin, 2006).

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan penelitian dan pengujian yang telah dilakukan, Naïve Pattern Searching Algorithm terbukti mampu mengontrol kunci pintu rumah dengan baik. Pengujian dilakukan sebanyak 700 kali oleh 11 orang *user* dengan enam skenario perintah, yaitu “Buka Pintu Depan”, “Buka Pintu Kamar”, “Kunci Pintu Depan”, “Kunci Pintu Kamar”, “Buka Semua Pintu”, dan “Kunci Semua Pintu”. Pengujian ini dibagi menjadi dua tahap yaitu, pengujian oleh peneliti dengan melakukan uji coba sebanyak 100 kali dengan 100 kalimat perintah berbeda dan 10 orang *user* dengan melakukan uji coba sebanyak 600 kali dengan memberikan total 60 kalimat perintah yang berbeda. Dari seluruh pengujian tersebut, sistem *success* merespon perintah sebanyak 643 dengan presentase 91.84% dan mengalami 57 kali error atau sebesar 8,14% dari total 700 pengujian. Angka e*rror* ini disebabkan karena *Naive Pattern Searching Algorithm* mengenali lebih dari satu *pattern* yang sama. Hal tersebut mengakibatkan sistem merespons lebih dari satu perintah dan sistem *hardware* tidak merespon perintah dengan baik dan tidak sesuai harapan.

5.2 Saran

Sistem ini masih memiliki beberapa kekurangan yang perlu diperbaiki untuk meningkatkan kinerjanya. Berikut adalah rekomendasi dari peneliti yang diharapkan dapat menjadi acuan bagi penelitian berikutnya:

1. Pengembangan sistem dengan menambah jumlah pintu atau perangkat dalam rumah lainnya seperti kompor, televisi, dan kipas angin,
2. Meningkatkan aspek keamanan pada sistem kendali, misalnya dengan mengimpentasikan pengenalan suara khusus untuk memastikan hanya suara tertentu yang dapat mengaktifkan sistem.

DAFTAR PUSTAKA

Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Umar, A. M., Linus, O. U., Arshad, H., Kazaure, A. A., Gana, U., & Kiru, M. U. (2019). Comprehensive Review Of Artificial Neural Network Applications To Pattern Recognition. *Ieee Access*, *7*, 158820–158846.

Al-Mahalli, J., & Jalaluddin, J. (2006). *Tafsir Jalalain Berikut Asbabun Nuzul Jilid 1* (B. Abubakar & H. A. Abubakar, Penerj.; 1–2). Pt Sinar Baru Algensindo.

Amarodin, A. (2021). ᵀᴱᴸAâ€^{Tm} Ah Tafsir Qs. An-Nahl Ayat 78 Dan Analisisnya. *Perspektif: Jurnal Program Studi Pendidikan Agama Islam*, *14*(02), 22–61.

Antonius, S., & Pramida, C. (2022). Model Pendampingan Pastoral Dengan Pendekatan Gestalt Terhadap Tuan X Yang Sulit Menerima Kebutaannya Tahun 2021. *Areopagus: Jurnal Pendidikan Dan Teologi Kristen*, *20*(1), 14–29.

Fantofani, R. (2020). *Smart Home Berbasis Internet Of Things Menggunakan Algoritma Nazief & Adriani Dan Algoritma Quick Search* [Skiripsi]. Universitas Islam Negeri Maulana Malik Ibrahim Malang.

Fitriani, R. (2023). Masa Depan Teknologi Digital Identity: Implikasi Dalam Kehidupan Sehari-Hari. *Jurnal Informatika Komputer*, 56–62.

Frost, J. (2021, September 22). *Percent Error: Definition, Formula & Examples*. Statistics By Jim. Https://Statisticsbyjim.Com/Basics/Percent-Error/

Guterres, A., Santoso, J., & Others. (2019). Stemming Bahasa Tetun Menggunakan Pendekatan Rule Based. *Teknika*, *8*(2), 142–147.

Hanani, A., & Hariyadi, M. A. (2020). Smart Home Berbasis Iot Menggunakan Suara Pada Google Assistant. *Jurnal Ilmiah Teknologi Informasi Asia*, *14*(1), 49–56.

Hariyadi, M. A., & Prakasa, J. E. W. (2023). *Manajemen Keamanan Sistem Informasi*.

Imran, M. (2024). Peningkatan Pemberdayaan Penyandang Tunanetra Melalui Perancangan Social Media Newsletter Di Yayasan Sosial Tunanetra. *Jurnal Komunitas : Jurnal Pengabdian Kepada Masyarakat*, *6*(2), 229–239. Https://Doi.Org/10.31334/Jks.V6i2.3587

Irmanda, H. N., Astriratma, R., Chamidah, N., & Santoni, M. M. (2021). Pembuat Sampiran Pantun Otomatis Berbasis Pattern-Matching. *Jurnal Sisfokom (Sistem Informasi Dan Komputer)*, *10*(3), 306–311.

Karamizadeh, S., Moazen, M., Zamani, M., & Manaf, A. A. (2024). Enhancing Iot-Based Smart Home Security Through A Combination Of Deep Learning

And Self-Attention Mechanism. *Arabian Journal For Science And Engineering*, 1–11.

Mukesh T. (2024, April 20). Naive Algorithm On Pattern Searching. *Medium*. Https://Medium.Com/@Mukeshtumapala107/Naive-Algorithm-On-Pattern-Searching-B499aa7fb857

Mulyawan, R. (2024, Agustus). *Pengertian Pattern Matching: Menurut Ahli, Rumus, Dan Paper Terkaitnya!* Rifqi Mulyawan. Https://Rifqimulyawan.Com/Kamus/Pattern-Matching/

Mumtaz, M. R. F., Fikra, H., & Others. (2022). Sikap Menyayangi Sesama Manusia Dalam Perspektif Islam: Studi Takhrij Dan Syarah Hadis. *Gunung Djati Conference Series*, *8*, 609–618.

Padmaveni, K., & John Aravindhar, D. (2021). Improved Skip Algorithm For Single Pattern Searching. *Inventive Communication And Computational Technologies: Proceedings Of Icicct 2020*, 255–267.

Phadke, M., & Korde, M. (2024). Iot-Based Weather Monitoring System Using Nodemcu Esp8266. Dalam T. Senjyu, C. So–In, & A. Joshi (Ed.), *Smart Trends In Computing And Communications* (Hlm. 211–220). Springer Nature. Https://Doi.Org/10.1007/978-981-97-1320-2_18

Ponnaganti, R. (2024, April 20). Naive Algorithm For Pattern Searching. *Medium*. Https://Medium.Com/@Rishitha_Ponnaganti/Naive-Algorithm-For-Pattern-Searching-3dad21e7cd1e

Prawiyogi, A. G., Anwar, A. S., & Others. (2023). Perkembangan Internet Of Things (Iot) Pada Sektor Energi: Sistematik Literatur Review. *Jurnal Mentari: Manajemen, Pendidikan Dan Teknologi Informasi*, *1*(2), 187–197.

Prianto, E. M. (2024). *Rancang Bangun Alat Iot Memberi Pakan Ikan Lele Secara Otomatis Berbasis Telegram* [Skripsi]. Universitas Labuhanbatu.

Rabbani, M. K. (2021). *Sistem Kendali Pagar Rumah Berbasis Internet Of Things Dengan Perintah Suara Menggunakan Algoritma Boyer-Moore* [Skripsi]. Universitas Islam Negeri Maulana Malik Ibrahim.

Radya, M. (2024). *Memahami Dasar Penggunaan Arduino Esp32 Untuk Pemula* [Graphic]. Https://Blog.Indobot.Co.Id/Memahami-Dasar-Penggunaan-Arduino-Esp32-Untuk-Pemula/

Refiansyah Maldini, R. (2023). *Sistem Keamanan Teknologi Untuk Sistem Internet Of Thing*.

Retkoceri, F., Idrizi, F., Ismaili, S., Imeri, F., & Memeti, A. (2022). Analysis Of Pattern Searching Algorithms And Their Application. *International Journal Of Recent Contributions From Engineering, Science & It (Ijes)*, *10*(04), 32–42. Https://Doi.Org/10.3991/Ijes.V10i04.35295

Riadi, M. (2023, November 22). *Speech Recognition—Pengertian, Jenis, Aspek Dan Cara Kerja*. Https://Www.Kajianpustaka.Com/2023/11/Speech-Recognition.Html

Rinandyaswara, R., Sari, Y. A., & Furqon, M. T. (2022). Pembentukan Daftar Stopword Menggunakan Term Based Random Sampling Pada Analisis Sentimen Dengan Metode Naïve Bayes (Studi Kasus: Kuliah Daring Di Masa Pandemi). *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *9*(4), 717–724.

Rizki, M., Fitriansyah, A., & Narji, M. (2023). Aplikasi Chatbot Sebagai Layanan Live Chat Untuk Penerimaan Mahasiswa Baru Menggunakan Metode Word Stemming Dengan Regular Expression Pattern Matching. *Jeis: Jurnal Elektro Dan Informatika Swadharma*, *3*(2), 50–62.

Rizky, R. F., Zy, A. T., & Sunge, A. S. (2023). Sistem Smart Door Lock Menggunakan Voice Recognition Berbasis Arduino. *Bulletin Of Information Technology (Bit)*, *4*(2), 239–244.

Sasia, P. I., Murti, M. A., & Setianingsih, C. (2020). Implementasi Sistem Kendali Lampu Dengan Voice Command Menggunakan Google Api. *Prosiding Seminar Nasional Teknik Elektro Uin Sunan Gunung Djati Bandung*, 77–83.

Selay, A., Andigha, G. D., Alfarizi, A., Wahyudi, M. I. B., Falah, M. N., Khaira, M., & Encep, M. (2022). Internet Of Things. *Karimah Tauhid*, *1*(6), Article 6. Https://Doi.Org/10.30997/Karimahtauhid.V1i6.7633

Sintanu, B. (2023). *Tolong Menolong Dalam Al-Qur'an*.

Supriyono, Wibawa, A. P., Suyono, & Kurniawan, F. (2024). Advancements In Natural Language Processing: Implications, Challenges, And Future Directions. *Telematics And Informatics Reports*, *16*, 100173. Https://Doi.Org/10.1016/J.Teler.2024.100173

Susanto, F., Prasiani, N. K., & Darmawan, P. (2022). Implementasi Internet Of Things Dalam Kehidupan Sehari-Hari. *Jurnal Imagine*, *2*(1), 35–40. Https://Doi.Org/10.35886/Imagine.V2i1.329

Tarmizi, Dr. S. N. (2024, Februari 22). 2,2 Miliar Orang Di Dunia Alami Gangguan Penglihatan, Kongres Apao Ke-39 Diharapkan Lahirkan Solusi. *Sehat Negeriku*. Https://Sehatnegeriku.Kemkes.Go.Id/Baca/Rilis-Media/20240222/0044976/22-Miliar-Orang-Di-Dunia-Alami-Gangguan-Penglihatan-Kongres-Apao-Ke-39-Diharapkan-Lahirkan-Solusi/

Ulum, M. (2022). *Sistem Kontrol Gorden Berbasis Internet Of Things Menggunakan Algoritma Knuth Morris Pratt* [Skripsi]. Universitas Islam Negeri Maulana Malik Ibrahim.

Wirawan, I., & Paryatna, I. (2020). *Implementation Of The String Matching Method On Anggah-Ungguhing Balinese Language Dictionary*.

Yoga, P. (2020, Oktober). *Esp32 Dev Module For Iot (Internet Of Things)* [Www.Arduino.Biz.Id]. Https://Www.Arduino.Biz.Id/2020/10/Esp32-Dev-Module-For-Iot-Internet-Of.Html

LAMPIRAN

Lampiran 1 Hasil Pengenalan Perintah "Buka Pintu Depan" yang Dilakukan oleh Peneliti.

BUKA PINTU DEPAN											
No	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
			Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
1	Buka Pintu Depan	buka pintu depan	0	-	5	-	-	TRUE	0	1	berhasil
2	Tolong bukakan pintu depan.	buka pintu depan	0	-	5	-	-	TRUE	0	1	berhasil
3	Bisakah anda membantu saya untuk membukakan pintu depan itu saya mau masuk	bisa bantu buka pintu depan mau masuk	11	-	16	-	-	TRUE	0	1	berhasil
4	Ayo buka pintu depan secepatnya.	ayo buka pintu depan cepat	4	-	9	-	-	TRUE	0	1	berhasil
5	Mohon dibukakan pintu depan.	mohon buka pintu depan	6	-	11	-	-	TRUE	0	1	berhasil
6	Buka pintu depan, ya!	buka pintu depan	0	-	5	-	-	TRUE	0	1	berhasil
7	Coba buka pintu depan untukku	coba buka pintu depan untuk	5	-	10	-	-	TRUE	0	1	berhasil
8	Segera buka pintu depan agar kita bisa masuk.	segera buka pintu depan masuk	7	-	12	-	-	TRUE	0	1	berhasil
9	Hallo smart home tolonglah saya untuk membuka pintu depan	halo smart home tolong buka pintu depan	23	-	28	-	-	TRUE	0	1	berhasil
10	Silahkan bukain pintu depan agar angin bisa masuk.	sila bukain pintu depan angin masuk	5	-	12	-	-	TRUE	0	1	berhasil
11	Apakah kamu bisa membuka pintu depan untuk saya?	kamu buka pintu depan	5	-	10	-	-	TRUE	0	1	berhasil
12	Saya minta kamu untuk membuka pintu depan perlahan-lahan.	minta kamu buka pintu depan perlahan-lahan	11	-	16	-	-	TRUE	0	1	berhasil
13	Bukalah pintu depan agar saya dan keluarga bisa masuk.	buka pintu depan keluarga masuk	0	-	5	-	-	TRUE	0	1	berhasil
14	Segeralah membuka pintu depan untuk menyambut tamu.	segera buka pintu depan sambut tamu	7	-	12	-	-	TRUE	0	1	berhasil

BUKA PINTU DEPAN											
No	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
			Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
15	Tolong, bantu saya membuka pintu depan yang terkunci ini.	bantu buka pintu depan kunci	6	23	11	-	-	TRUE	0	1	berhasil
16	Saya memohon kamu untuk membuka pintu depan secepatnya.	mohon kamu buka pintu depan cepat	11	-	16	-	-	TRUE	0	0	berhasil
17	Bukakkan pintu depan yang terkunci, biarkan semuanya bisa masuk.	buka pintu depan kunci biar semua masuk	0	17	5	-	28	FALSE	1	0	gagal
18	Bisakah membatu saya untuk mebukakan pintu yang didepan, tolong?	bisa bantu buka pintu depan	11	-	16	-	-	TRUE	0	1	berhasil
19	Sepertinya didepan itu ada tamu, tolong untuk membukakan pintu depannya agar bisa masuk	seperti depan tamu buka pintu depan masuk	19	-	24	-	-	TRUE	0	1	berhasil
20	Saya memohon kamu untuk membuka dipintu depan secepatnya.	mohon kamu buka pintu depan cepat	11	-	16	-	-	TRUE	0	1	berhasil
∑Ei (Jumlah performa dari semua percobaan)									**1**	**19**	**1 Gagal, 19 Berhasil**

Lampiran 2 Hasil Pengenalan Perintah "Buka Pintu Depan" yang Dilakukan oleh 10 Orang User.

BUKA PINTU DEPAN												
No	**Jumlah *User***	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
				Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
1	10	Buka Pintu Depan	buka pintu depan	0	-	5	-	-	TRUE	0	1	berhasil
2	10	Tolong bukakan pintu depan.	buka pintu depan	0	-	5	-	-	TRUE	0	1	berhasil

BUKA PINTU DEPAN												
No	**Jumlah *User***	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
				Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
3	10	Bisakah anda membantu saya untuk membukakan pintu depan itu saya mau masuk	bisa bantu buka pintu depan mau masuk	11	-	16	-	-	TRUE	0	1	berhasil
4	10	Ayo buka pintu depan secepatnya.	ayo buka pintu depan cepat	4	-	9	-	-	TRUE	0	1	berhasil
5	10	Mohon dibukakan pintu depan.	mohon buka pintu depan	6	-	11	-	-	TRUE	0	1	Berhasil
6	10	Buka pintu depan, ya!	buka pintu depan	0	-	5	-	-	TRUE	0	1	Berhasil
7	10	Coba buka pintu depan untukku	coba buka pintu depan untuk	5	-	10	-	-	TRUE	0	1	berhasil
8	10	Segera buka pintu depan agar kita bisa masuk.	segera buka pintu depan masuk	7	-	12	-	-	TRUE	0	1	berhasil
9	10	Hallo smart home tolonglah saya untuk membuka pintu depan	halo smart home tolong buka pintu depan	23	-	28	-	-	TRUE	0	1	berhasil
10	10	Silahkan bukain pintu depan agar angin bisa masuk.	sila bukain pintu depan angin masuk	5	-	12	-	-	TRUE	0	1	berhasil
∑Ei (Jumlah performa dari semua percobaan)										**0**	**100**	**0 Gagal, 100 Berhasil**

Lampiran 3 Hasil Pengenalan Perintah "Buka Pintu Kamar" yang Dilakukan oleh Peneliti.

BUKA PINTU KAMAR										
No	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
			Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
1	Buka Pintu Kamar	buka pintu kamar	0	-	-	5	-	TRUE	0	1	berhasil
2	Tolong bukakan pintu kamar.	buka pintu kamar	0	-	-	5	-	TRUE	0	1	berhasil
3	Bisakah Anda membantu saya untuk membukakan pintu kamar itu? Saya mau masuk.	bisa bantu buka pintu kamar mau masuk	11	-	-	16	-	TRUE	0	1	berhasil
4	Ayo, buka pintu kamar secepatnya.	ayo buka pintu kamar cepat	4	-	-	9	-	TRUE	0	1	berhasil
5	Mohon dibukakan pintu kamar.	mohon buka pintu kamar	6	-	-	11	-	TRUE	0	1	berhasil
6	Buka pintu kamar, ya!	buka pintu kamar	0	-	-	5	-	TRUE	0	1	berhasil
7	Coba buka pintu kamar untukku.	coba buka pintu kamar untuk	5	-	-	10	-	TRUE	0	1	berhasil
8	Segera buka pintu kamar agar kita bisa masuk.	segera buka pintu kamar masuk	7	-	-	12	-	TRUE	0	1	berhasil
9	Halo smart home, tolonglah saya untuk membuka pintu kamar.	halo smart home tolong buka pintu kamar	23	-	-	28	-	TRUE	0	1	berhasil
10	Silakan bukain pintu kamar agar angin bisa masuk.	sila bukain pintu kamar angin masuk	5	-	-	12	-	TRUE	0	1	berhasil
11	Apakah kamu bisa membuka pintu kamar untuk saya?	kamu buka pintu kamar	5	-	-	10	-	TRUE	0	1	berhasil

BUKA PINTU KAMAR											
No	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
			Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
12	Saya minta kamu untuk membuka pintu kamar perlahan-lahan.	minta kamu buka pintu kamar perlahan-lahan	11	-	-	16	-	TRUE	0	1	berhasil
13	Bukalah pintu kamar agar saya dan keluarga bisa masuk.	buka pintu kamar keluarga masuk	0	-	-	5	-	TRUE	0	1	berhasil
14	Segeralah membuka pintu kamar untuk menyambut tamu.	segera buka pintu kamar sambut tamu	7	-	-	12	-	TRUE	0	1	berhasil
15	Tolong, bantu saya membuka pintu kamar yang terkunci ini.	bantu buka pintu kamar kunci	6	23	-	11	-	TRUE	0	1	berhasil
16	Saya memohon kamu untuk membuka pintu kamar secepatnya.	mohon kamu buka pintu kamar cepat	11	-	-	16	-	TRUE	0	1	berhasil
17	Bukakkan pintu kamar yang terkunci, biarkan semuanya bisa masuk.	buka pintu kamar kunci semua masuk	0	17	-	5	23	FALSE	1	0	gagal
18	Bisakah membantu saya untuk membukakan pintu kamar, tolong?	bisa bantu buka pintu kamar	11	-	-	16	-	TRUE	0	1	berhasil
19	Sepertinya di depan itu ada orang, tolong bukakan pintu kamar agar bisa masuk.	seperti depan orang buka pintu kamar masuk	20	-	25	-		TRUE	0	1	berhasil
20	Saya memohon kamu untuk membuka pintu kamar secepatnya.	mohon kamu buka pintu kamar cepat	11	-	-	16	-	TRUE	0	1	berhasil
∑Ei (Jumlah performa dari semua percobaan)									**1**	**19**	**1 Gagal, 19 Berhasil**

Lampiran 4 Hasil Pengenalan Perintah "Buka Pintu Kamar" yang Dilakukan oleh 10 Orang User.

BUKA PINTU KAMAR												
No	**Jumlah *User***	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
				Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
1	10	Apakah kamu bisa membuka pintu kamar untuk saya?	kamu buka pintu kamar	5	-	-	10	-	TRUE	0	10	berhasil
2	10	Saya minta kamu untuk membuka pintu kamar perlahan-lahan.	minta kamu buka pintu kamar perlahan-lahan	11	-	-	16	-	TRUE	0	10	berhasil
3	10	Bukalah pintu kamar agar saya dan keluarga bisa masuk.	buka pintu kamar keluarga masuk	0	-	-	5	-	TRUE	0	10	berhasil
4	10	Segeralah membuka pintu kamar untuk menyambut tamu.	segera buka pintu kamar sambut tamu	7	-	-	12	-	TRUE	0	10	berhasil
5	10	Tolong, bantu saya membuka pintu kamar yang terkunci ini.	bantu buka pintu kamar kunci	6	23	-	11	-	TRUE	0	10	berhasil
6	10	Saya memohon kamu untuk membuka pintu kamar secepatnya.	mohon kamu buka pintu kamar cepat	11	-	-	16	-	TRUE	0	10	berhasil

BUKA PINTU KAMAR												
No	**Jumlah *User***	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
				Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
7	10	Bukakkan pintu kamar yang terkunci, biarkan semuanya bisa masuk.	buka pintu kamar kunci semua masuk	0	17	-	5	23	FALSE	10	0	gagal
8	10	Bisakah membantu saya untuk membukakan pintu kamar, tolong?	bisa bantu buka pintu kamar	11	-	-	16	-	TRUE	0	10	berhasil
9	10	Sepertinya di depan itu ada orang, tolong bukakan pintu kamar agar bisa masuk.	seperti depan orang buka pintu kamar masuk	20	-	25	-		TRUE	0	10	berhasil
10	10	Saya memohon kamu untuk membuka pintu kamar secepatnya.	mohon kamu buka pintu kamar cepat	11	-	-	16	-	TRUE	0	10	berhasil
∑Ei (Jumlah performa dari semua percobaan)										**10**	**90**	**10 Gagal, 90 Berhasil**

Lampiran 5 Hasil Pengenalan Perintah "Kunci Pintu Depan" yang Dilakukan oleh Peneliti.

KUNCI PINTU DEPAN										
No	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
			Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
1	Kunci Pintu Depan	kunci pintu depan	-	0	6	-	-	TRUE	0	1	berhasil
2	Tolong kunci pintu depan.	kunci pintu depan	-	0	6	-	-	TRUE	0	1	berhasil
3	Bisakah Anda membantu saya untuk menguncikan pintu depan itu? Saya mau keluar.	bisa bantu kunci pintu depan mau keluar	-	11	17	-	-	TRUE	0	1	berhasil
4	Ayo, kunci pintu depan secepatnya.	ayo kunci pintu depan cepat	-	4	10	-	-	TRUE	0	1	berhasil
5	Mohon dikuncikan pintu depan.	mohon kunci pintu depan	-	6	12	-	-	TRUE	0	1	berhasil
6	Kunci pintu depan, ya!	kunci pintu depan	-	0	6	-	-	TRUE	0	1	berhasil
7	Coba kunci pintu depan untukku.	coba kunci pintu depan untuk	-	5	11	-	-	TRUE	0	1	berhasil
8	Segera kunci pintu depan agar kita bisa keluar dengan tenang.	segera kunci pintu depan keluar tenang	-	7	13	-	-	TRUE	0	1	berhasil
9	Halo smart home, tolonglah saya untuk mengunci pintu depan.	halo smart home tolong kunci pintu depan	-	23	29	-	-	TRUE	0	1	berhasil
10	Silakan kunci pintu depan agar angin tidak masuk.	sila kunci pintu depan angin masuk	-	5	11	-	-	TRUE	0	1	berhasil
11	Apakah kamu bisa mengunci pintu depan untuk saya?	kamu kunci pintu depan	-	5	11	-	-	TRUE	0	1	berhasil
12	Saya minta kamu untuk mengunci pintu depan perlahan-lahan.	minta kamu kunci pintu depan perlahan-lahan	-	11	17	-	-	TRUE	0	1	berhasil
13	Kuncilah pintu depan agar saya dan keluarga bisa beristirahat.	kunci pintu depan keluarga istirahat	-	0	6	-	-	TRUE	0	1	berhasil

KUNCI PINTU DEPAN											
No	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
			Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
14	Segeralah mengunci pintu depan untuk menjaga kenyamanan rumah.	segera kunci pintu depan jaga nyaman rumah	-	7	13	-	-	TRUE	0	1	berhasil
15	Tolong, bantu saya mengunci pintu depan soalnya saya sudah berada didalam rumah.	bantu kunci pintu depan soal ada rumah	-	6	12	-	-	TRUE	0	1	berhasil
16	Saya memohon kamu untuk mengunci pintu depan secepatnya.	mohon kamu kunci pintu depan cepat	-	11	17	-	-	TRUE	0	1	berhasil
17	Kuncikan pintu depan yang terbuka, biarkan semuanya ada didalam rumah	kunci kan pintu depan buka biar semua rumah	22	0	11	-	32	FALSE	1	0	gagal
18	Bisakah membantu saya untuk menguncikan pintu depan, tolong?	bisa bantu buka pintu depan	-	11	21	-	-	TRUE	0	1	berhasil
19	Kuncilah pintu depan supaya kita bisa merasa nyaman	kunci pintu depan rasa nyaman	-	0	6	-	-	TRUE	0	1	berhasil
20	Saya memohon kamu untuk mengunci pintu depan secepatnya.	mohon kamu kunci pintu depan cepat	-	11	17	-	-	TRUE	0	1	berhasil
∑Ei (Jumlah performa dari semua percobaan)									**1**	**19**	**1 Gagal, 19 Berhasil**

Lampiran 6 Hasil Pengenalan Perintah "Kunci Pintu Depan" yang Dilakukan oleh 10 Orang User.

KUNCI PINTU DEPAN											
No	**Jumlah *User***	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
				Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
1	10	Kunci Pintu Depan	kunci pintu depan	-	0	6	-	-	TRUE	0	10	berhasil
2	10	Tolong kunci pintu depan.	kunci pintu depan	-	0	6	-	-	TRUE	0	10	berhasil
3	10	Bisakah Anda membantu saya untuk menguncikan pintu depan itu? Saya mau keluar.	bisa bantu kunci pintu depan mau keluar	-	11	17	-	-	TRUE	0	10	berhasil
4	10	Ayo, kunci pintu depan secepatnya.	ayo kunci pintu depan cepat	-	4	10	-	-	TRUE	0	10	berhasil
5	10	Mohon dikuncikan pintu depan.	mohon kunci pintu depan	-	6	12	-	-	TRUE	0	10	berhasil
6	10	Kunci pintu depan, ya!	kunci pintu depan	-	0	6	-	-	TRUE	0	10	berhasil
7	10	Coba kunci pintu depan untukku.	coba kunci pintu depan untuk	-	5	11	-	-	TRUE	0	10	berhasil
8	10	Segera kunci pintu depan agar kita bisa keluar dengan tenang.	segera kunci pintu depan keluar tenang	-	7	13	-	-	TRUE	0	10	berhasil
9	10	Halo smart home, tolonglah saya untuk mengunci pintu depan.	halo smart home tolong kunci pintu depan	-	23	29	-	-	TRUE	0	10	berhasil

KUNCI PINTU DEPAN												
No	**Jumlah *User***	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
				Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
10	10	Silakan kunci pintu depan agar angin tidak masuk.	sila kunci pintu depan angin masuk	-	5	11	-	-	TRUE	0	10	berhasil
∑Ei (Jumlah performa dari semua percobaan)										**0**	**100**	**0 Gagal, 100 Berhasil**

Lampiran 7 Hasil Pengenalan Perintah "Kunci Pintu Kamar" yang Dilakukan oleh Peneliti.

KUNCI PINTU KAMAR											
No	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
			Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
1	Kunci Pintu Kamar	buka pintu kamar	-	0	6	-	-	TRUE	0	1	berhasil
2	Tolong kunci pintu kamar.	buka pintu kamar	-	0	6	-	-	TRUE	0	1	berhasil
3	Bisakah Anda membantu saya untuk menguncikan pintu kamar itu? Saya mau tidur.	bisa bantu buka pintu kamar mau masuk	-	11	17	-	-	TRUE	0	1	berhasil
4	Ayo, kunci pintu kamar saya secepatnya.	ayo buka pintu kamar cepat	-	4	10	-	-	TRUE	0	1	berhasil
5	Mohon dikunci kan pintu kamar.	mohon buka pintu kamar	-	6	12	-	-	TRUE	0	1	berhasil
6	Kunci pintu kamar, ya!	buka pintu kamar	-	0	6	-	-	TRUE	0	1	berhasil
7	Coba kunci pintu kamar untukku.	coba buka pintu kamar untuk	-	5	11	-	-	TRUE	0	1	berhasil
8	Segera kunci pintu kamar agar kita bisa keluar dengan tenang.	segera buka pintu kamar masuk	-	7	13	-	-	TRUE	0	1	berhasil
9	Halo smart home, tolonglah saya untuk mengunci pintu kamar.	halo smart home tolong buka pintu kamar	-	23	29	-	-	TRUE	0	1	berhasil

KUNCI PINTU KAMAR										
No	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
			Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
10	Silakan kunci pintu kamar agar debu tidak masuk.	sila bukain pintu kamar angin masuk	-	5	11	-	-	TRUE	0	1	berhasil
11	Apakah kamu bisa mengunci pintu kamar untuk saya?	kamu buka pintu kamar	-	5	11	-	-	TRUE	0	1	berhasil
12	Saya minta kamu untuk mengunci pintu kamar perlahan-lahan.	minta kamu buka pintu kamar perlahan-lahan	-	11	17	-	-	TRUE	0	1	berhasil
13	Kuncilah pintu kamar agar saya dan teman saya bisa beristirahat.	buka pintu kamar keluarga masuk	-	0	6	-	-	TRUE	0	1	berhasil
14	Segeralah mengunci pintu kamar untuk menjaga kenyamanan rumah.	segera buka pintu kamar sambut tamu	-	7	13	-	-	TRUE	0	1	berhasil
15	Tolong, bantu saya mengunci pintu kamar soalnya saya sudah berada didalam kamar.	bantu buka pintu kamar kunci	-	6	12	-	-	TRUE	0	1	berhasil
16	Saya memohon kamu untuk mengunci pintu kamar secepatnya.	mohon kamu buka pintu kamar cepat	-	11	17	-	-	TRUE	0	1	berhasil
17	Kunci kan pintu kamar yang terbuka, biarkan semuanya ada didalam kamar.	buka pintu kamar kunci semua masuk	22	0	11	-	32	FALSE	1	0	gagal
18	Bisakah membantu saya untuk menguncikan pintu kamar, tolong?	bisa bantu buka pintu kamar	-	11	21	-	-	TRUE	0	1	berhasil
19	Ternyata tamu di luar sudah masuk semua tolong kunci pintu kamar agar aman dan nyaman	seperti depan orang buka pintu kamar masuk	-	0	6	-	-	TRUE	0	1	berhasil

KUNCI PINTU KAMAR											
No	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
			Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
20	Saya memohon kamu untuk mengunci pintu kamar secepatnya.	mohon kamu buka pintu kamar cepat	-	11	17	-	-	TRUE	0	1	berhasil
∑Ei (Jumlah performa dari semua percobaan)									**1**	**19**	**1 Gagal, 19 Berhasil**

Lampiran 8 Hasil Pengenalan Perintah "Kunci Pintu Kamar" yang Dilakukan oleh 10 Orang User.

KUNCI PINTU KAMAR												
No	**Jumlah *User***	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
				Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
1	10	Apakah kamu bisa mengunci pintu kamar untuk saya?	kamu kunci pintu kamar	-	5	-	11	-	TRUE	0	10	berhasil
2	10	Saya minta kamu untuk mengunci pintu kamar perlahan-lahan.	minta kamu kunci pintu kamar perlahan-lahan	-	11	-	17	-	TRUE	0	10	berhasil
3	10	Kuncilah pintu kamar agar saya dan teman saya bisa beristirahat.	kunci pintu kamar teman istirahat	-	0	-	6	-	TRUE	0	10	berhasil
4	10	Segeralah mengunci pintu kamar untuk menjaga kenyamanan rumah.	segera kunci pintu kamar jaga nyaman rumah	-	7	-	13	-	TRUE	0	10	berhasil

KUNCI PINTU KAMAR												
No	**Jumlah *User***	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
				Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
5	10	Tolong, bantu saya mengunci pintu kamar soalnya saya sudah berada didalam kamar.	bantu kunci pintu kamar soal ada kamar	-	6	-	13	-	TRUE	0	10	berhasil
6	10	Saya memohon kamu untuk mengunci pintu kamar secepatnya.	mohon kamu kunci pintu kamar cepat	-	11	-	17	-	TRUE	0	10	berhasil
7	10	Kunci kan pintu kamar yang terbuka, biarkan semuanya ada didalam kamar.	kunci kan pintu kamar buka biar semua rumah	22	0	-	10	32	FALSE	10	0	gagal
8	10	Bisakah membantu saya untuk menguncikan pintu kamar, tolong?	bisa bantu kunci pintu kamar	-	11	-	17	-	TRUE	0	10	berhasil
9	10	Ternyata tamu di luar sudah masuk semua tolong kunci pintu kamar agar aman dan nyaman	nyata tamu luar masuk semua kunci pintu kamar aman nyaman	-	27	-	33	21	8 TRUE, 2 FALSE	2	8	gagal
10	10	Saya memohon kamu untuk mengunci pintu kamar secepatnya.	mohon kamu kunci pintu kamar cepat	-	11	-	17	-	TRUE	0	10	berhasil
∑Ei (Jumlah perorma dari semua percobaan)										**12**	**88**	**12 Gagal, 88 Berhasil**

Lampiran 9 Hasil Pengenalan Perintah "Buka Semua Pintu" yang Dilakukan oleh Peneliti.

BUKA SEMUA PINTU										
No	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
			Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
1	Buka Semua Pintu	buka semua pintu	0	-	-	-	5	TRUE	0	1	berhasil
2	Bisakah kamu membukakan semua pintu yaitu kamar dan depan untukku?	bisa kamu buka semua pintu kamar depan untuk	10	-	-	21	15	TRUE	0	1	berhasil
3	Buka semua pintu yang terkunci, biarkan semuanya terbuka.	buka semua pintu kunci biar semua buka	0	17	-	-	5	TRUE	0	1	berhasil
4	Bukalah semua pintu agar semua orang bisa masuk dengan nyaman.	buka semua pintu semua orang masuk nyaman	0	-	-	-	5	TRUE	0	1	berhasil
5	Mohon dibukakan semua pintu.	mohon buka semua pintu	6	-	-	-	11	TRUE	0	1	berhasil
6	Ayo, bukalah semua pintu yang terkunci supaya kita tidak terjebak di dalam.	ayo buka semua pintu kunci jebak	4	21	-	-	9	TRUE	0	1	berhasil
7	Coba semua pintu dibuka agar cahaya matahari masuk ke dalam ruangan.	coba semua pintu buka cahaya matahari masuk ruang	17	-	-	-	5	TRUE	0	1	berhasil
8	Saya ingin bersih bersih rumah bukakkan semua pintu yang ada dirumah	bersih rumah buka semua pintu rumah	13	-	-	-	18	TRUE	0	1	berhasil
9	Halo smart home, tolonglah saya untuk membuka semua pintu kamar dan depan	halo smart home tolong buka semua pintu kamar depan	23	-	-	34	28	TRUE	0	1	berhasil
10	Saya mohon, segera buka semua pintu agar kita bisa melanjutkan kegiatan.	mohon segera buka semua pintu lanjut giat	13	-	-	-	18	TRUE	0	1	berhasil

BUKA SEMUA PINTU										
No	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
			Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
∑Ei (Jumlah kesalahan dari semua percobaan)									**0**	**10**	**0 Gagal, 10 Berhasil**

Lampiran 10 Hasil Pengenalan Perintah "Buka Semua Pintu" yang Dilakukan oleh 10 Orang User.

BUKA SEMUA PINTU											
No	**Jumlah *User***	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	**Error**	***Succes***	**Keterangan**
				Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
1	10	Buka Semua Pintu	buka semua pintu	0	-	-	-	5	TRUE	0	1	berhasil
2	10	Bisakah kamu membukakan semua pintu yaitu kamar dan depan untukku?	bisa kamu buka semua pintu kamar depan untuk	10	-	-	21	15	TRUE	0	1	berhasil
3	10	Buka semua pintu yang terkunci, biarkan semuanya terbuka.	buka semua pintu kunci biar semua buka	0	17	-	-	5	TRUE	0	1	berhasil
4	10	Bukalah semua pintu agar semua orang bisa masuk dengan nyaman.	buka semua pintu semua orang masuk nyaman	0	-	-	-	5	TRUE	0	1	berhasil
5	10	Mohon dibukakan semua pintu.	mohon buka semua pintu	6	-	-	-	11	TRUE	0	1	berhasil

BUKA SEMUA PINTU												
No	**Jumlah *User***	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	**Error**	***Succes***	**Keterangan**
				Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
6	10	Ayo, bukalah semua pintu yang terkunci supaya kita tidak terjebak di dalam.	ayo buka semua pintu kunci jebak	4	21	-	-	9	TRUE	0	1	berhasil
7	10	Coba semua pintu dibuka agar cahaya matahari masuk ke dalam ruangan.	coba semua pintu buka cahaya matahari masuk ruang	17	-	-	-	5	TRUE	0	1	berhasil
8	10	Saya ingin bersih bersih rumah bukakkan semua pintu yang ada dirumah	bersih rumah buka semua pintu rumah	13	-	-	-	18	TRUE	0	1	berhasil
9	10	Halo smart home, tolonglah saya untuk membuka semua pintu kamar dan depan	halo smart home tolong buka semua pintu kamar depan	23	-	-	34	28	TRUE	0	1	berhasil
10	10	Saya mohon, segera buka semua pintu agar kita bisa melanjutkan kegiatan.	mohon segera buka semua pintu lanjut giat	13	-	-	-	18	TRUE	0	1	berhasil
∑Ei (Jumlah performa dari semua percobaan)										**0**	**100**	**0 Gagal, 100 Berhasil**

Lampiran 11 Hasil Pengenalan Perintah "Buka Semua Pintu" yang Dilakukan oleh Peneliti.

KUNCI SEMUA PINTU										
No	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
			Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
1	Kunci Semua Pintu	kunci semua pintu	-	0	-	-	6	TRUE	0	1	berhasil
2	Bisakah kamu menguncikan semua pintu yaitu kamar dan depan untukku?	bisa kamu kunci semua pintu kamar depan untuk	-	10	-	22	16	TRUE	0	1	berhasil
3	Kunci semua pintu yang terbuka, biarkan semuanya aman terkendali.	kunci semua pintu buka biar semua aman kendali	18	0	-	-	6	FALSE	1	0	gagal
4	Kuncilah semua pintu agar semua orang merasa aman dan nyaman didalam rumah saya.	kunci semua pintu semua orang rasa aman nyaman rumah	-	0	-	-	6	TRUE	0	1	berhasil
5	Mohon dikunci kan semua pintu.	mohon kunci kan semua pintu	-	6	-	-	16	TRUE	0	1	berhasil
6	Ayo, kuncilah semua pintu yang terbuka supaya kita tidak terjebak diluar.	ayo kunci semua pintu buka jebak luar	22	4	-	-	10	FALSE	1	0	gagal
7	Coba semua pintu yang terbuka dikunci agar tidak ada debu masuk ke dalam ruangan.	coba semua pintu buka kunci debu masuk ruang	17	22	-	-	5	FALSE	1	0	gagal
8	Saya ingin beristirahat didalam rumah kuncikan semua pintu yang ada dirumah	istirahat rumah kunci kan semua pintu rumah	-	16	-	-	26	TRUE	0	0	berhasil
9	Halo smart home, tolonglah saya untuk mengkuncikan semua pintu kamar dan depan	halo smart home tolong kunci semua pintu kamar depan	-	23	-	35	29	TRUE	0	0	berhasil

KUNCI SEMUA PINTU											
No	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
			Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
10	Saya mohon, segera kunci semua pintu agar kita bisa melanjutkan kegiatan didalam rumah.	mohon segera kunci semua pintu lanjut giat rumah	-	13	-	-	19	TRUE	0	0	berhasil
∑Ei (Jumlah performa dari semua percobaan)									**3**	**7**	**3 Gagal, 7 Berhasil**

Lampiran 12 Hasil Pengenalan Perintah "Kunci Semua Pintu" yang Dilakukan oleh 10 Orang User.

	KUNCI SEMUA PINTU											
No	**Nama**	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
				Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
1	10	Kunci Semua Pintu	kunci semua pintu	-	0	-	-	6	TRUE	0	10	berhasil
2	10	Bisakah kamu menguncikan semua pintu yaitu kamar dan depan untukku?	bisa kamu kunci semua pintu kamar depan untuk	-	10	-	22	16	TRUE	0	10	berhasil
3	10	Kunci semua pintu yang terbuka, biarkan semuanya aman terkendali.	kunci semua pintu buka biar semua aman kendali	18	0	-	-	6	FALSE	10	0	gagal
4	10	Kuncilah semua pintu agar semua orang merasa	kunci semua pintu semua	-	0	-	-	6	TRUE	0	10	berhasil

	KUNCI SEMUA PINTU											
No	**Nama**	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
				Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
		aman dan nyaman didalam rumah saya.	orang rasa aman nyaman rumah									
5	10	Mohon dikunci kan semua pintu.	mohon kunci kan semua pintu	-	6	-	-	16	TRUE	0	10	berhasil
6	10	Ayo, kuncilah semua pintu yang terbuka supaya kita tidak terjebak diluar.	ayo kunci semua pintu buka jebak luar	22	4	-	-	10	1 TRUE, 9 FALSE	9	1	gagal
7	10	Coba semua pintu yang terbuka dikunci agar tidak ada debu masuk ke dalam ruangan.	coba semua pintu buka kunci debu masuk ruang	17	22	-	-	5	1 TRUE, 9 FALSE	9	1	gagal
8	10	Saya ingin beristirahat didalam rumah kuncikan semua pintu yang ada dirumah	istirahat rumah kunci kan semua pintu rumah	-	16	-	-	26	TRUE	0	10	berhasil
9	10	Halo smart home, tolonglah saya untuk mengkuncikan semua pintu kamar dan depan	halo smart home tolong kunci semua pintu kamar depan	-	23	-	35	29	TRUE	0	10	berhasil
10	10	Saya mohon, segera kunci semua pintu agar	mohon segera kunci semua	-	13	-	-	19	TRUE	0	10	berhasil

	KUNCI SEMUA PINTU											
No	**Nama**	**Perintah**	**Hasil Stemming**	**Pattern Ditemukan pada Indeks ke-**					**Hasil Pengujian**	***Error***	***Succes***	**Keterangan**
				Buka	**Kunci**	**Pintu Depan**	**Pintu Kamar**	**Semua**				
		kita bisa melanjutkan kegiatan didalam rumah.	pintu lanjut giat rumah									
∑Ei (Jumlah kesalahan dari semua percobaan)										**28**	**72**	**28 Gagal, 72 Berhasil**

Lampiran 13 Dokumentasi Bukti Pengujian oleh 10 Orang User