RANCANG BANGUN SISTEM PEMANTAU KETERSEDIAAN PARKIR *REAL-TIME* MENGGUNAKAN MIKROKONTROLER BERBASIS *INTERNET OF THINGS*

(Skripsi)

Oleh

Refli Nicholas Hakim
2015031040

PROGRAM STUDI TEKNIK ELEKTRO
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2024

ABSTRAK

RANCANG BANGUN SISTEM PEMANTAU KETERSEDIAAN PARKIR *REAL-TIME* MENGGUNAKAN MIKROKONTROLER BERBASIS *INTERNET OF THINGS*

Oleh

REFLI NICHOLAS HAKIM

Permasalahan dalam mencari slot parkir yang tersedia di pusat perbelanjaan atau mall sering disebabkan oleh kurangnya informasi *real-time* mengenai ketersediaan dan jumlah slot parkir. Penelitian ini bertujuan untuk mempermudah akses informasi tersebut melalui sistem berbasis *Internet of Things* (IoT) yang dapat dipantau secara *real-time*, sehingga pengunjung dapat menghemat waktu dan biaya. Sistem ini akan menampilkan informasi ketersediaan slot parkir dan jumlah slot parkir yang tersedia melalui aplikasi *smartphone*.

Sistem ini dibangun dengan menggunakan sensor ultrasonik HC-SR04 untuk mendeteksi kendaraan yang terparkir dan mikrokontroler ESP32 untuk memproses data dari sensor. Mikrokontroler ESP32 mengirimkan data yang diproses ke *database*, yang kemudian menampilkan informasi posisi dan jumlah slot parkir tersedia secara *real-time* di aplikasi *smartphone*.

Hasil penelitian ini adalah terbangunnya prototipe sistem pemantauan ketersediaan slot parkir. Sistem yang dirancang berfungsi sesuai dengan skenario yang telah ditentukan, dengan akurasi sensor ultrasonik mencapai 100%. Selain itu, latensi pengiriman data berkisar antara 1,74 hingga 3,55 detik, dengan radius jarak antara *smartphone* dan prototipe lokasi parkir di bawah 10 km.

Kata kunci : Ketersediaan Slot Parkir, *Real-time, Internet of Things,* Sensor Ultrasonik, ESP32, Aplikasi Android, Firebase.

ABSTRACT

DESIGN OF REAL-TIME PARKING AVAILABILITY MONITORING SYSTEM USING MICROCONTROLLER BASED ON INTERNET OF THINGS

By

REFLI NICHOLAS HAKIM

The problem of finding available parking slots in shopping centers or malls is often caused by the lack of real-time information about the availability and number of parking slots. This research aims to facilitate access to such information through an Internet of Things (IoT) based system that can be monitored in real-time, so that visitors can save time and money. This system will display information on the availability of parking slots and the number of available parking slots through a smartphone application.

This system is built using HC-SR04 ultrasonic sensors to detect parked vehicles and ESP32 microcontrollers to process data from sensors. The ESP32 microcontroller sends the processed data to the database, which then displays position information and the number of available parking slots in real-time on the smartphone application.

The result of this research is the construction of a prototype parking slot availability monitoring system. The designed system functions according to a predetermined scenario, with ultrasonic sensor accuracy reaching 100%. In addition, the data transmission latency ranges from 1.74 to 3.55 seconds, with a distance radius between the smartphone and the parking lot prototype under 10 km.

Keywords : Parking Slot Availability, Real-time, Internet of Things, Ultrasonic Sensor, ESP32, Android Application, Firebase.

RANCANG BANGUN SISTEM PEMANTAU KETERSEDIAAN PARKIR *REAL-TIME* MENGGUNAKAN MIKROKONTROLER BERBASIS *INTERNET OF THINGS*

Oleh

REFLI NICHOLAS HAKIM

Skripsi

Sebagai salah satu syarat untuk mendapat gelar
SARJANA TEKNIK

pada

Jurusan Teknik Elektro
Fakultas Teknik Universitas Lampung

PROGRAM STUDI TEKNIK ELEKTRO
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2024

Judul Skripsi	**:**	**RANCANG BANGUN SISTEM PEMANTAU KETERSEDIAAN PARKIR *REAL-TIME* MENGGUNAKAN MIKROKONTROLER BERBASIS *INTERNET OF THINGS***
Nama Mahasiswa	:	Refli Nicholas Hakim
Nomor Pokok Mahasiswa	:	2015031040
Program Studi	:	Teknik Elektro
Fakultas	:	Teknik

MENYETUJUI

1. Komisi Pembimbing

Dr. Eng. Ageng Sadnowo R , S.T.,M.T.	**Dr. Sri Purwiyanti, S.T., M.T.**
NIP. 19690228 199803 1 001	NIP. 19731004 199803 2 001

2. Mengetahui

Ketua Jurusan Teknik Elektro	Ketua Program Studi S1 Teknik Elektro
Herlinawati, S.T., M.T.	**Sumadi, S.T., M.T.**
NIP. 19710314 199903 2 001	NIP. 19731104 200003 1 001

MENGESAHKAN

1. Tim Penguji

Ketua	: **Dr. Eng. Ageng Sadnowo R, S.T.,M.T.**	……
Sekretaris	: **Dr. Sri Purwiyanti, S.T., M.T.**	……
Penguji Utama	: **Herlinawati, S.T., M.T.**	……

2. Dekan Fakultas Teknik

Dr. Eng. Ir. Helmy Fitriawan, S.T., M.Sc.
NIP. 19750928 200112 1 002

Tanggal Lulus Ujian Skripsi : **12 Agustus 2024**

SURAT PERNYATAAN

Dengan ini saya menyatakan bahwa dalam skripsi ini yang berjudul "Rancang Bangun Sistem Pemantau Ketersediaan Parkir *Real-time* Menggunakan Mikrokontroler Berbasis *Internet of Things*" tidak terdapat karya yang pernah dilakukan orang lain dan sepanjang pengetahuan saya tidak terdapat atas diterbitkannya oleh orang lain, kecuali secara tertulis diacu dalam naskah ini sebagaimana hyang disebutkan dalam daftar Pustaka. Selain itu, saya menyatakan pula bahwa skripsi ini dibuat oleh saya sendiri. Apabila pernyataan saya tidak benar, maka saya bersedia dikenai sanksi akademik sesuai dengan hukum yang berlaku.

Bandar Lampung, 23 September 2024

Refli Nicholas Hakim
NPM. 2015031040

RIWAYAT HIDUP

Saya lahir di Saribumi, pada tanggal 06 Juli 2002 sebagai anak bungsu dari 2 bersaudara, anak dari Bapak alm. Sukari dan Ibu Rinawati. Pendidikan sekolah dasar diselesaikan di SDN 01 Wates pada tahun 2014, sekolah menengah pertama di SMP Muhammadiyah 1 Gadingrejo diselesaikan pada tahun 2017, dan sekolah menengah kejuruan jurusan Elektronika Industri di SMK Negeri 1 Gadingrejo diselesaikan pada tahun 2020. Pada tahun 2020, saya terdaftar sebagai mahasiswa di Jurusan Teknik Elektro Fakultas Teknik Universitas Lampung melalui jalur SBMPTN. Saya telah aktif terlibat dalam berbagai kegiatan akademik dan organisasi. Selama 2 periode kepengurusan, saya menjabat sebagai Anggota Departemen Komunikasi dan Informasi Periode 2021 s.d. 2022 dan Koordinator Seksi Publikasi Dekorasi dan Dokumentasi di acara Electrical Engineering In Action 2022. Selain itu, saya turut aktif di Laboratorium Teknik Kendali sebagai asisten selama tahun 2022 s.d. 2024.

Pada semester 5, saya memilih mengambil konsentrasi Elektronika dan Kendali. Pencapaian saya yaitu berhasil meraih medali perak pada lomba The 6th Digitalised International Invention, Innovation & Design Competition 2024 yang diadakan oleh Universiti Teknologi MARA Johor bersama Universitas Lampung. Penulis juga melaksanakan Kerja Praktek (KP) di PT. Matanusa Energi Utama yang bergerak di bidang pembangkitan listrik di PLTMG Sutami. Pencapaian ini mencerminkan komitmen saya terhadap pengembangan diri, kontribusi dalam bidang teknologi, dan partisipasi aktif dalam kehidupan kampus. Saya berharap dapat terus berkontribusi dalam meningkatkan kualitas dan eksplorasi di dunia teknologi melalui perjalanan akademis dan kegiatan organisasi yang saya jalani.

PERSEMBAHAN

Dengan Ridho Allah SWT

Teriring shalawat kepada Nabi Muhammad SAW

Karya Tulis ini ku persembahkan untuk:

Ayah dan Ibuku Tercinta

Sukari dan Rinawati

Serta Kakakku Tersayang

Finky Eka Gesta Kharinda, S.A.B.

Dan seluruh keluargaku yang tidak bisa disebutkan satu persatu

Terima kasih untuk semua dukungan dan doa selama ini.

Sehingga aku dapat menyelesaikan hasil karyaku ini.

MOTTO

“Tidaklah Mungkin Bagi Matahari Mengejar Bulan Dan Mallam Pun Tidak Dapat Mendahului Siang. Masing-Masing Beredar Pada Garis Edarnya”

“Disetiap kesulitan pasti ada kemudahan, dan menyerah hanyalah untuk orang yang kalah.”

“ONE DAY OR DAY ONE”

“BELIVE IN YOURSELF”

“Nothing Can Be Gained Without Losing Something Even Heaven Demands Death”

SANWACANA

Alhamdulillah puji syukur kehadirat Allah SWT atas segala karunia, hidayah, serta inayah-Nya kepada penulis, sehingga laporan skripsi ini yang berjudul "Rancang Bangun Sistem Pemantau Ketersediaan Parkir *Real-time* Menggunakan Mikrokontroler Berbasis *Internet of Things*" dapat selesai tepat pada waktunya. Yang merupakan salah satu syarat untuk memperoleh gelar Sarjana Teknik di Jurusan Teknik Elektro Fakultas Teknik Universitas Lampung. Shalawat serta salam selalu tercurah kepada junjungan seluruh alam, Nabi Muhammad SAW. sahabatnya, serta para pengikutnya yang selalu istiqomah diatas jalan agama islam hingga hari akhir zaman. Selama menjalani pengerjaan Skripsi ini, penulis mendapatkan bantuan pemikiran maupun dorongan moril dari berbagai pihak. Oleh karena itu dalam kesempatan kali ini, penulis ingin mengucapkan terima kasih kepada:

1. Ibu Prof. Dr. Ir. Lusmeilia Afriani, D.E.A., I.P.M., selaku Rektor Universitas Lampung.
2. Bapak Dr. Eng. Ir. Helmy Fitriawan, S.T., M.Sc., selaku Dekan Fakultas Teknik Universitas Lampung.
3. Ibu Herlinawati, S.T., M.T. selaku Ketua Jurusan Teknik Elektro Fakultas Teknik Universitas Lampung dan juga sekaligus dosen penguji bagi penulis.
4. Bapak Sumadi, S.T., M.T. selaku Ketua Program Studi Teknik Elektro Fakultas Teknik Universitas Lampung dan telah memberikan motivasi kepada penulis dalam menyelesaikan studi
5. Bapak Dr. Eng. Ageng Sadnowo R., S.T., M.T., selaku Pembimbing Utama, terima kasih atas ilmu, keikhlasan, kesabaran, dan motivasinya selama penyusunan skripsi ini.
6. Ibu Dr. Sri Purwiyanti, S.T., M.T. selaku Pembimbing Pendamping tugas akhir, yang telah membantu, membimbing, dan memberi dukungan kepada penulis.

7. Bapak Dr. Ing. Ardian Ulvan, S.T., M.Sc. sebagai Dosen Pembimbing Akademik, yang telah banyak membimbing dan membantu penulis selama menjalani kuliah.
8. Seluruh Dosen dan karyawan Jurusan Teknik Elektro Universitas Lampung, berkat ilmu yang telah diajarkan kepada penulis selama penulis menjalani masa studi di perkuliahan.
9. Ibu Umi Murdika, S.T., M.T. selaku Kepala Laboratorium Teknik Kendali dan Kak Perdana Agung Nugraha,S.T. selaku PLP Laboratorium Teknik Kendali terima kasih atas diberikannya tempat untuk menyelesaikan tugas akhir ini.
10. Papi Doyok, Mba Debi, Mas Arman, Ibu Nani, Mba Cindy dan seluruh keluarga yang telah mendukung dan memotivasi penulis.
11. Keluarga rekan-rekan di Laboratorium Teknik kendali yang selalu memberikan dukungan, pertolongan, canda tawa, dalam setiap proses apapun selama menjadi asisten laboratorium teknik kendali.
12. Keluarga besar Angkatan 2020, yang telah memberikan banyak motivasi, nilai-nilai sosial, dan bantuan dalam berbagai hal.
13. Keluarga besar HIMATRO UNILA, yang telah menjadi wadah dalam mengembangkan nilai-nilai organisasi bagi penulis.
14. Keluarga kecil PENAMPUNGAN RUMAH AMAL, Ibu, Ayah, Taja, Bang Arif, Amall, Arda, Ahmad, Gus, Saka, Reyzal, Rizki, Alfin, Herly, Zul, Akmall.
15. Semua pihak yang tidak dapat disebutkan satu persatu dan terlibat langsung maupun tidak langsung yang telah membantu penulis dalam pembuatan skripsi.

Akhir kata, semoga Allah membalas semua kebaikan bagi semua yang telah membantu. Semoga skripsi ini dapat bermanfaat bagi kita semua.

Bandar Lampung, 23 September 2024

Penulis,

Refli Nicholas Hakim

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

I. PENDAHULUAN

1.1 Latar Belakang

Pusat perbelanjaan yang sering ramai dikunjungi masyarakat sering menghadapi masalah serius terkait ketersediaan parkir. Pengunjung sering kali harus menghabiskan waktu cukup lama hanya untuk mencari slot parkir yang tersedia. Selain memakan waktu, biaya parkir juga tetap harus dibayar meskipun kendaraan tidak mendapatkan tempat dan akhirnya harus keluar karena lokasi parkir penuh. Waktu yang dihabiskan untuk mencari slot parkir juga berdampak pada peningkatan penggunaan bahan bakar, yang pada akhirnya memperburuk dampak karbon terhadap lingkungan [1].

Salah satu masalah utama dalam mencari slot parkir di pusat perbelanjaan adalah kurangnya informasi *real-time* mengenai ketersediaan slot parkir dan lokasinya. Hal ini menyebabkan pengunjung harus berkeliling untuk mencari tempat parkir yang kosong. Untuk mengatasi masalah ini, diperlukan solusi inovatif yang dapat memberikan informasi terkait kondisi tempat parkir, seperti jumlah slot yang tersedia dan di bagian mana gedung slot tersebut berada. Selain itu, informasi ini juga harus mencakup kondisi tempat parkir di gedung atau pusat perbelanjaan lain yang tergabung dalam sistem informasi yang sama.

Di era digital saat ini, kemajuan teknologi informasi sangat membantu kehidupan manusia, terutama dengan berkembangnya teknologi *Internet of Things* (IoT). IoT memungkinkan perangkat elektronik saling berkomunikasi dan bertukar data melalui internet. Penerapan teknologi IoT memberikan kemudahan signifikan dalam berbagai aspek kehidupan manusia. [2]. Aplikasi berbasis Android dapat menjadi alat yang efektif dan fleksibel untuk memberikan informasi kepada pengguna mengenai ketersediaan slot parkir di pusat perbelanjaan. Pemilihan

platform Android sebagai pengembangan aplikasi didasarkan pada sifatnya yang *open source*, kemudahan dalam pengoperasian, dan fleksibilitas telepon seluler [3]. Di sisi lain, teknologi IoT dapat digunakan untuk memantau dan mengumpulkan data *real-time* terkait ketersediaan slot parkir. Data *real-time* tersebut kemudian dikirimkan ke aplikasi Android.

Penggabungan teknologi Android dan IoT akan menghasilkan aplikasi parkir yang fleksibel dan *real-time*, memberikan manfaat signifikan bagi pengguna ketika merencanakan kunjungan ke pusat perbelanjaan. Aplikasi ini akan memberikan informasi mengenai pusat perbelanjaan mana yang memiliki tempat parkir yang tersedia, sehingga pengguna dapat menghemat waktu dan biaya dalam mencari tempat parkir. Selain itu, aplikasi ini juga dapat membantu mengurangi kemacetan di dalam gedung atau mall yang sering disebabkan oleh pengunjung yang bersamaan mencari tempat parkir.

1.2 Tujuan Penelitian

Adapun tujuan penelitian ini adalah sebagai berikut:

1. Merancang sistem yang dapat memantau ketersediaan slot parkir pada mall atau gedung parkir berbasis *Internet of Things* (IoT).
2. Merancang aplikasi dengan fungsi memantau ketersediaan parkir pada mall atau gedung parkir yang dapat memberikan informasi secara *real-time* kepada pengguna.

1.3 Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan kontribusi dalam menyelesaikan masalah ketidakefisienan pengunjung dari sisi waktu dan biaya.

1.4 Rumusan Masalah

Bagaimana membuat sebuah sistem pemantauan ketersediaan slot parkir dan mengakuisisi datanya pada sebuah *database* serta mengirimkannya melalui *cloud* sehingga dapat dibaca informasinya oleh aplikasi berbasis Android.

1.5 Batasan Masalah

Adapun batasan masalah pada penelitian ini sebagai berikut:

1. Simulasi tempat parkir menggunakan prototipe dengan kapasitas 8 slot parkir mobil. Slot dibagi dalam dua kelompok masing-masing 4 slot mengasumsikan sebuah lokasi dengan dua lantai berkapasitas 4 mobil.
2. Mobil yang digunakan pada prototipe menggunakan mobil berukuran skala 1:100 dari mobil asli.

1.6 Hipotesis

Pada penelitian ini, sistem elektronik yang dibangun dapat memberikan informasi ketersediaan slot parkir secara *real-time* melalui *smartphone*. Perubahan kondisi slot parkir dapat diakses kurang dari 10 detik.

1.7 Sistematika Penulisan

Adapun sistematika penulisan yang digunakan dalam penulisan proposal ini sebagai berikut:

BAB I PENDAHULUAN

Pada bab ini menjelaskan tentang latar belakang penelitian perancangan sistem dan aplikasi pemantau ketersediaan parkir secara *real-time* berbasis android, tujuan penelitian, manfaat penelitian, rumusan masalah, batasan masalah, hipotesis, dan sistematika penulisan laporan tugas akhir.

BAB II TINJAUAN PUSTAKA

Pada bab ini memaparkan tentang keterkaitan antara riset-riset terdahulu dengan riset yang akan dilakukan mengenai *smart parking*, kemudian menjelaskan teori dasar yang terkait dengan rancangan sistem dan aplikasi pemantau ketersediaan tempat parkir berbasis android.

BAB III METODOLOGI PENELITIAN

Pada bab ini, dijelaskan langkah perancangan alat dan aplikasi yang dimulai dari membuat skenario rancangan pemantau ketersediaan parkir berbasis android sampai mengimplementasikannya menjadi sebuah model fisik yang dapat diuji.

BAB IV HASIL DAN PEMBAHASAN

Pada bab ini membahas pengujian model fisik dan aplikasi pemantau ketersediaan parkir kesesuaiannya dengan skenario rancangan.

BAB V KESIMPULAN DAN SARAN

Pada bab ini, berisi kesimpulan dari proses rancang bangun aplikasi dan alat pemantau ketersediaan parkir berbasis android yang menginformasikan capaian tujuan riset yang dilakukan. Selain itu, disampaikan saran-saran yang menjadi pekerjaan mendatang sebagai peningkatan dari capaian rancangan saat ini.

DAFTAR PUSTAKA

II. TINJAUAN PUSTAKA

2.1 Penelitian Terdahulu

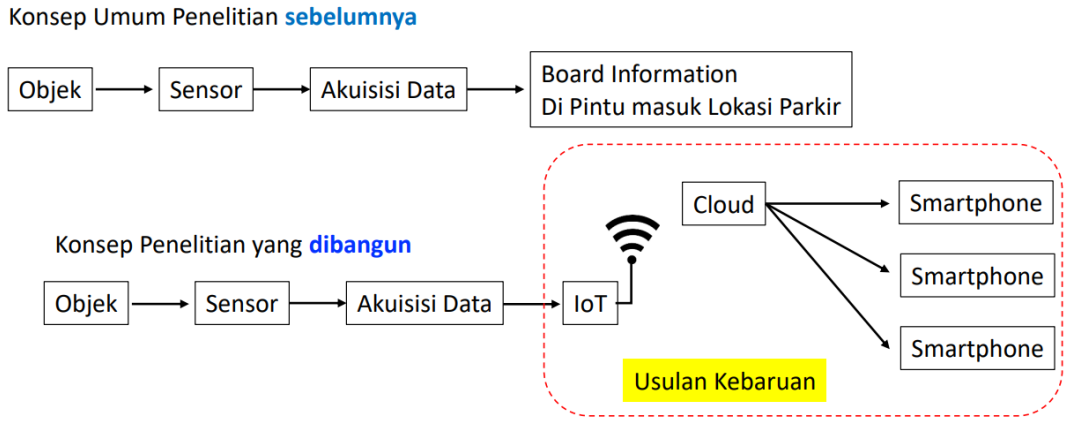
Pada penelitian sebelumnya dilakukan oleh Hendy Tri Laksono dan Zuly Budiarso, Program Studi Teknik Informatika, Universitas Stikubank Semarang. Penelitian ini memiliki judul Rancang Bangun Smart Parkir Berbasis Arduino. Sistem ini menggunakan Arduino Uno dan sensor Ultrasonik HC-SR04 yang dihubungkan dengan LCD 16x2 sebagai tampilan menginformasikan slot kosong dan Motor Servo DC sebagai pembuka pintu palang [4].

Pada penelitian yang dilakukan oleh Ayom Purbo Wiseso, Denny Irawan, dan Rini Puji Astutik, Jurusan Teknik Elektro, Universitas Muhammadiyah Gresik. Penelitian ini memiliki judul Rancang Bangun Sistem Informasi Ketersediaan Slot Parkir Dalam Mall. Penelitian ini mengembangkan sistem parkir dan memantau ketersediaan slot parkir yang tersedia di dalam gedung menggunakan Arduino Mega 2560 sebagai pemrosesan data. Sistem parkir ini dirancang untuk membuka portal pintu masuk secara otomatis dan portal menutup kembali setelah mobil melewati portal tersebut. Sistem ini menggunakan LCD TFT untuk menampilkan informasi lokasi parkir yang tersedia [5].

Pada penelitian yang dilakukan oleh Tri Amelia Dewi Marwan, Tamara Berliana, dan Febrin Aulia Batubara, Politeknik Negeri Medan. Penelitian ini memiliki judul Rancang Bangun Sistem *Smart Parking* Berbasis *Internet of Things* (IoT). Penelitian ini menggunakan NodeMCU ESP8266 sebagai mikrokontroler, Infra Red sensor untuk mendeteksi mobil di slot parkir, ketika masuk dan meninggalkan lahan parkir dan motor servo sebagai palang parkir yang akan terbuka dan tertutup otomatis dan hasil pembacaan sensor akan ditampilkan pada website adafruit IO [6].

Pada penelitian yang dilakukan oleh Firyan Noer Alief Dermawan, Universitas Lampung. Penelitian ini memiliki judul Pendeteksi Lokasi Parkir Kosong secara Otomatis Berbasis Raspberry Pi. Penelitian ini menggunakan Raspberry Pi sebagai mikrokontroler, sensor ultrasonik HC SR04 sebagai pendeteksi lokasi parkir yang kosong. Kemudian hasil pembacaan akan diolah oleh Raspberry Pi untuk menentukan lokasi parkir terdekat, setelah itu ditampilkan pada LCD 16x2.

Secara sederhana, keempat penelitian di atas memiliki konsep rancangan yang sama, seperti terlihat pada Gambar 2.1.



Gambar 2.1. Konsep penelitian sistem informasi ketersediaan slot parkir

Pada Gambar 2.1 Objek terdeteksi oleh sensor, kemudian data sensor diproses apakah ada objek terparkir atau tidak. Jika terbaca adanya objek terdeteksi, maka mikrokontroler akan memberikan sinyal aktif kepada *interface* untuk menampilkan informasi ketersediaan slot parkir.

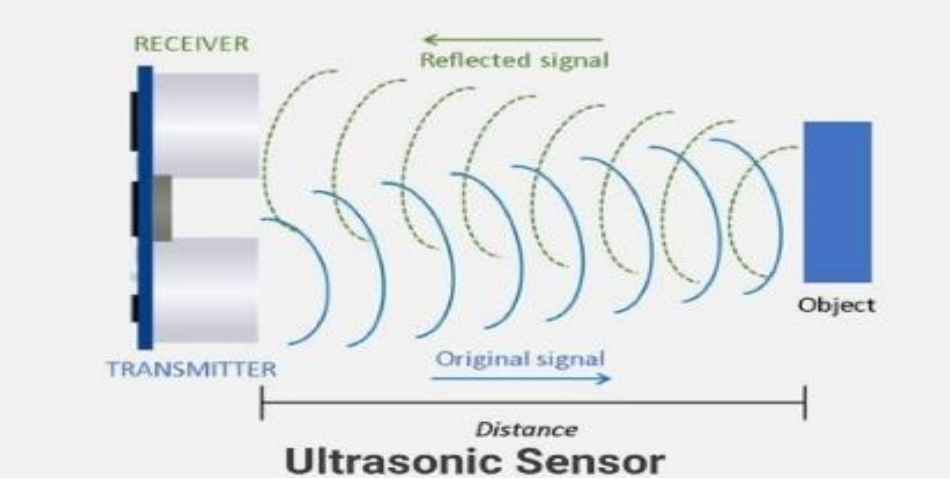
Berdasarkan referensi yang didapatkan, perbedaan pada penelitian ini yaitu berfokus pada pengembangan sistem parkir yang lebih canggih dan terintegrasi dengan aplikasi android. Sistem yang dirancang ini memungkinkan pengguna untuk terhubung langsung melalui aplikasi android yang telah disediakan. Dengan menggunakan aplikasi ini, pengguna dapat dengan mudah melihat ketersediaan slot parkir di berbagai tempat parkir yang terintegrasi dengan perangkat IoT dan aplikasi android. Keunggulan utama dari sistem yang dirancang adalah kemampuannya untuk memberikan informasi secara *real-time* tentang ketersediaan parkir.

Pengguna dapat melihat slot parkir yang tersedia dan memilih tempat parkir yang paling sesuai dengan kebutuhan mereka sebelum tiba di mall, sehingga pengguna dapat menghemat waktu dan biaya dalam mencari tempat parkir yang tersedia.

2.2 Teori Dasar

2.2.1 Prinsip Sensor Ultrasonik

Sensor ultrasonik merupakan sensor yang bekerja dengan memancarkan suatu gelombang dan kemudian menghitung waktu pantulan gelombang tersebut. Sensor ini memiliki 4 pin yang harus dihubungkan ke mikrokontroler, yaitu pin Vcc, pin *ground*, pin *trigger*, dan pin *echo*. Pin Vcc dihubungkan ke sumber tegangan 5V, pin *ground* dihubungkan ke negatif dari sumber tegangan, sedangkan pin *trigger* dan *echo* dihubungkan pada pin digital mikrokontroler [7].



Gambar 2.2. Prinsip sensor ultrasonik HC-SR04

Sensor ultrasonik menggunakan suara untuk menentukan jarak antara sensor dan objek terdekat di jalurnya. Sensor mengirimkan gelombang suara pada frekuensi tertentu melalui transmitter. Kemudian gelombang yang terpantul oleh objek kembali ke sensor melalui receiver seperti pada Gambar 2.2. Sensor melacak waktu antara pengiriman gelombang suara kembali. Bisa dihitung jaraknya perjalanan dengan persamaan 1 [8].

$$S = t \times \frac{340\ m/s}{2} \quad (1).$$

Keterangan:

S = jarak (meter).

t = waktu (detik).

Kecepatan suara dapat dihitung berdasarkan berbagai atmosfer, kondisi, termasuk suhu, kelembaban dan tekanan. Sensor ultrasonic memiliki kerucut deteksi, sudut kerucut ini bervariasi dengan jarak.

Tabel 2.1. Spesifikasi Sensor Ultrasonik HC-SR04

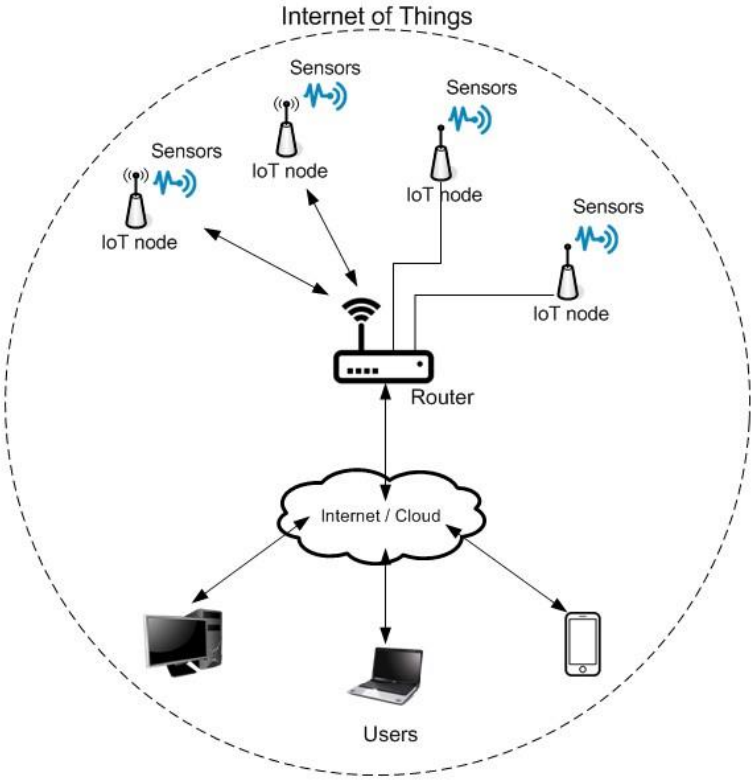
Parameter	*Specifications*
Working Voltage	DC 5 V
Working Current	15mA
Working Frequency	40Hz
Max Range	4m
Min Range	2 cm
MeasuringAngle	15 degree
Trigger Input Signal	10uS TTL pulse
Echo Output Signal	Input TTL lever signal and the range in proportion
Dimension	45*20*15mm

2.2.2 Konsep *Internet of Things*

Internet of Things (IoT) adalah konsep di mana objek-objek fisik yang dilengkapi dengan sensor, perangkat lunak, dan konektivitas internet dapat saling berkomunikasi dan berinteraksi satu sama lain serta dengan lingkungan sekitarnya. Pada dasarnya perangkat IoT terdiri dari sensor sebagai media pengumpul data, sambungan internet sebagai media komunikasi dan server sebagai pengumpul informasi yang diterima sensor untuk analisa. Dari perspektif ini, IoT dapat didefinisikan sebagai kumpulan benda cerdas seperti perangkat rumah, ponsel, laptop, dan lainnya, yang diidentifikasi oleh suatu skema alamat unik dan terhubung ke internet melalui kerangka kerja yang terpadu, yang mungkin merupakan komputasi awan [9].

IoT adalah salah satu teknologi memiliki hubungan erat terhadap istilah M2M (*machine-to-machine*). Alat yang digunakan pada M2M mampu berkomunikasi sehingga disebut *smart devices* atau perangkat cerdas [10]. Pengiriman data dalam *Internet of Things* (IoT) merupakan aspek kunci dalam

menjalankan berbagai aplikasi yang terhubung secara langsung dengan lingkungan fisik. Melalui protokol komunikasi yang sesuai dan menggunakan jaringan nirkabel seperti Wi-Fi, HTTPS, Bluetooth, atau teknologi jaringan khusus seperti LoRa, data yang dikumpulkan oleh sensor-sensor pada objek-objek IoT dapat dikirimkan secara efisien. Gambar 2.3 menggambarkan teknologi IoT.

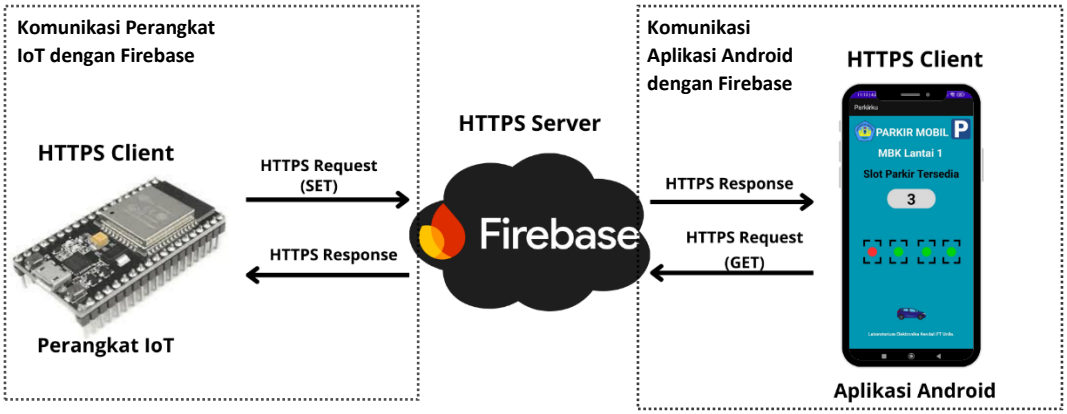


Gambar 2.3. Konsep *Internet of Things* (IoT)

HTTPS adalah protokol komunikasi yang digunakan pada penelitian ini. HTTPS (*Hypertext Transfer Protocol Secure*) adalah protokol komunikasi yang digunakan untuk mengirimkan data melalui internet secara aman. Dengan memanfaatkan *Secure Socket Layer* (SSL) atau *Transport Layer Security* (TLS) sebagai *sublayer* di bawah aplikasi HTTP biasa, teknologi protokol HTTPS mampu mencegah pencurian informasi penting yang dikirimkan selama proses komunikasi antara pengguna dan server web. Secara teknis, situs web yang menggunakan HTTPS akan mengenkripsi data menggunakan teknik enkripsi SSL. Dengan metode ini, meskipun seseorang berhasil mencuri data selama transmisi antara pengguna dan server web, mereka tidak akan dapat membacanya karena data tersebut telah dienkripsi oleh SSL [11]. Dalam konteks aplikasi IoT yang terhubung dengan

Firebase, proses pengiriman dan pengambilan data berlangsung dapat dilihat pada Gambar 2.4 dan dijelaskan sebagai berikut:

1. Pengiriman Data: Perangkat IoT mengirimkan data sensor ke Firebase *Real-time* Database menggunakan HTTPS Request (SET). Setelah permintaan diterima oleh HTTPS Server data ini dienkripsi sebelum dikirim untuk menjaga keamanan selama transit.
2. Pengambilan Data: Aplikasi atau perangkat lainnya dapat mengambil data dari Firebase *Real-time* Database dengan mengirimkan permintaan HTTPS Request (GET) untuk membaca data. Setelah permintaan diterima oleh HTTPS Server data ini dienkripsi sebelum dikirim untuk menjaga keamanan selama transit.



Gambar 2.4. Model Protokol Komunikasi HTTPS

2.2.2 Mikrokontroler

Mikrokontroler memainkan peran sentral dalam mengontrol dan memantau kondisi tempat parkir secara *real-time*. Mikrokontroler yang digunakan dalam sistem ini berfungsi sebagai otak yang mengumpulkan data dari sensor ultrasonik yang dipasang di area parkir, untuk mendeteksi apakah slot parkir tersedia atau tidak. Mikrokontroler juga dapat digunakan sebagai komputasi untuk menghitung jumlah slot parkir yang tersedia dengan cara menambah dan mengurangi jumlah data slot parkir saat terjadi perubahan dari data sensor. Pada penelitian ini menggunakan mikrokontroler ESP32 yang sudah dilengkapi modul WIFI dan Bluetooth yang mana sangat mendukung pembuatan sistem aplikasi *internet of things* (IoT) sehingga data yang telah diproses dapat dikirimkan ke *cloud database*.

III. METODE PENELITIAN

3.1 Waktu dan Tempat Penelitian

Penelitian ini dilaksanakan mulai dari Februari 2024 – Juli 2024 dengan tempat pelaksanaan yang dilakukan di Laboratorium Teknik Kendali Terpadu Teknik Elektro, Jurusan Teknik Elektro, Fakultas Teknik Universitas Lampung.

3.2 Alat dan Bahan

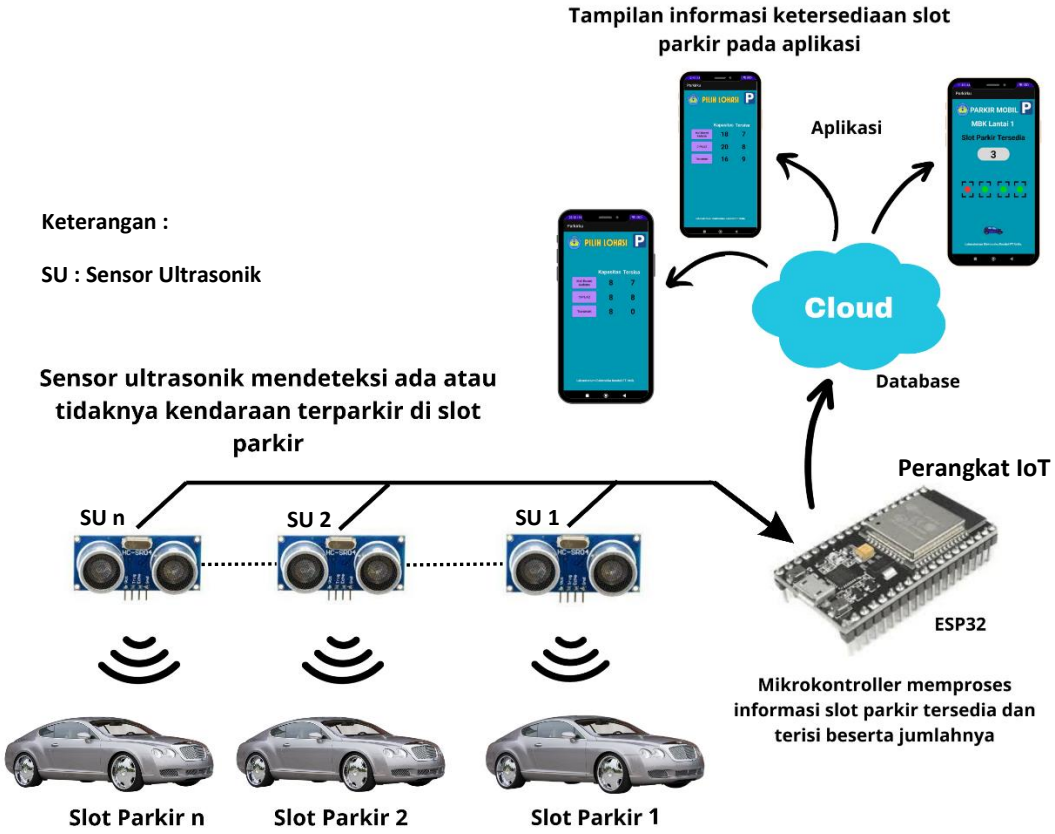
Adapun alat dan bahan digunakan dalam penelitian ini dapat dilihat pada tabel 3.1.

Tabel 3.1. Alat dan Bahan

No.	Nama Komponen dan Perangkat Lunak	Keterangan Penggunaan
1	ESP32	Sebagai pengolah dan pengirim data
2	Sensor Ultrasonik HC-SR04	Sebagai sensor deteksi kendaraan
3	*Arduino* IDE	Sebagai perangkat lunak untuk membangun program yang akan dijalankan pada mikrokontroler
4	Android Studio	Sebagai *platform* dalam pembuatan aplikasi
5	Firebase	Sebagai *platform* dalam pembuatan *Real-time database*
6	Google Sheets	Sebagai penyimpanan data
7	Laptop Asus	Sebagai perangkat keras untuk menjadi wadah dari perancangan sistem
8	*Handphone*	Sebagai perangkat keras untuk menjadi wadah dari hasil sistem

3.3 Metode Penelitian

Pada penelitian ini dilakukan konsep rancangan sistem seperti Gambar 3.1.



Gambar 3.1. Konsep rancangan sistem ketersediaan slot parkir berbasis android

Dari Gambar 3.1 dapat dijelaskan sebagai berikut:

1. ESP32 terhubung dengan koneksi internet menggunakan *id* dan *password* dari *hotspot/wifi* yang digunakan.
2. Setelah terkoneksi maka akan mulai mengaktifkan sensor ultrasonik.
3. ESP32 membaca perubahan nilai pada sensor ultrasonik lalu diolah menjadi data berupa *true*, *false*, slot tersedia, dan slot terpakai.
4. Setelah itu data *output* dikirimkan ke *database* pada platform firebase melalui internet.
5. *Smartphone* dihubungkan pada aplikasi yang telah dirancang melalui internet akan menampilkan informasi ketersediaan slot parkir.

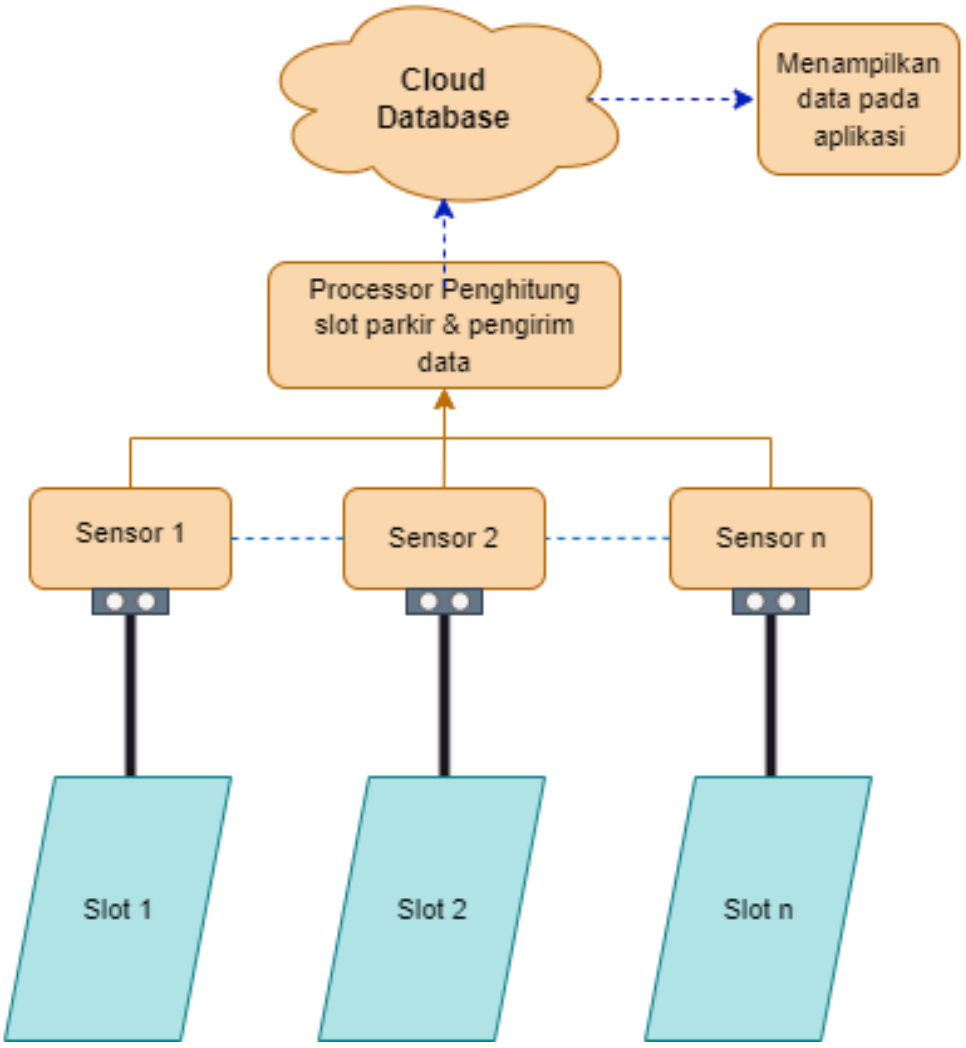
Semua data *output* kemudian akan dipantau melalui *smartphone* secara *real-time* melalui aplikasi yang terintegrasi dengan ESP32.

3.4 Skenario Perancangan Sistem

Skenario perancangan sistem sebagai berikut:

a. Skenario pemasangan sistem sensor

Dapat dilihat pada Gambar 3.2 posisi ultrasonik berada di depan slot parkir kendaraan, semua sensor terhubung dengan prosesor penghitung slot parkir dan pengirim data.



Gambar 3.2. Skenario pemasangan sensor

b. Skenario perhitungan slot parkir

Dalam perhitungan slot parkir saat ada objek kendaraan kurang dari 10 cm pada sensor maka akan memberikan nilai 1 dan saat tidak ada objek kendaraan lebih dari 10 cm pada sensor maka akan memberikan nilai 0. Kemudian untuk perhitungan jumlah slot yang terisi dapat dicari menggunakan persamaan 2.

$$T = N1 + N2 + \ ... + Nn \qquad (2).$$

Keterangan:

T = Jumlah slot terisi.

N = Nilai dari pembacaan sensor ultrasonik.

Kemudian setelah mendapatkan jumlah slot terisi dapat digunakan untuk mencari perhitungan jumlah slot tersedia. Untuk perhitungan jumlah slot yang terisi dapat dicari menggunakan persamaan 3.

$$S = J - T \qquad (3).$$

Keterangan:

S = Jumlah slot tersedia.

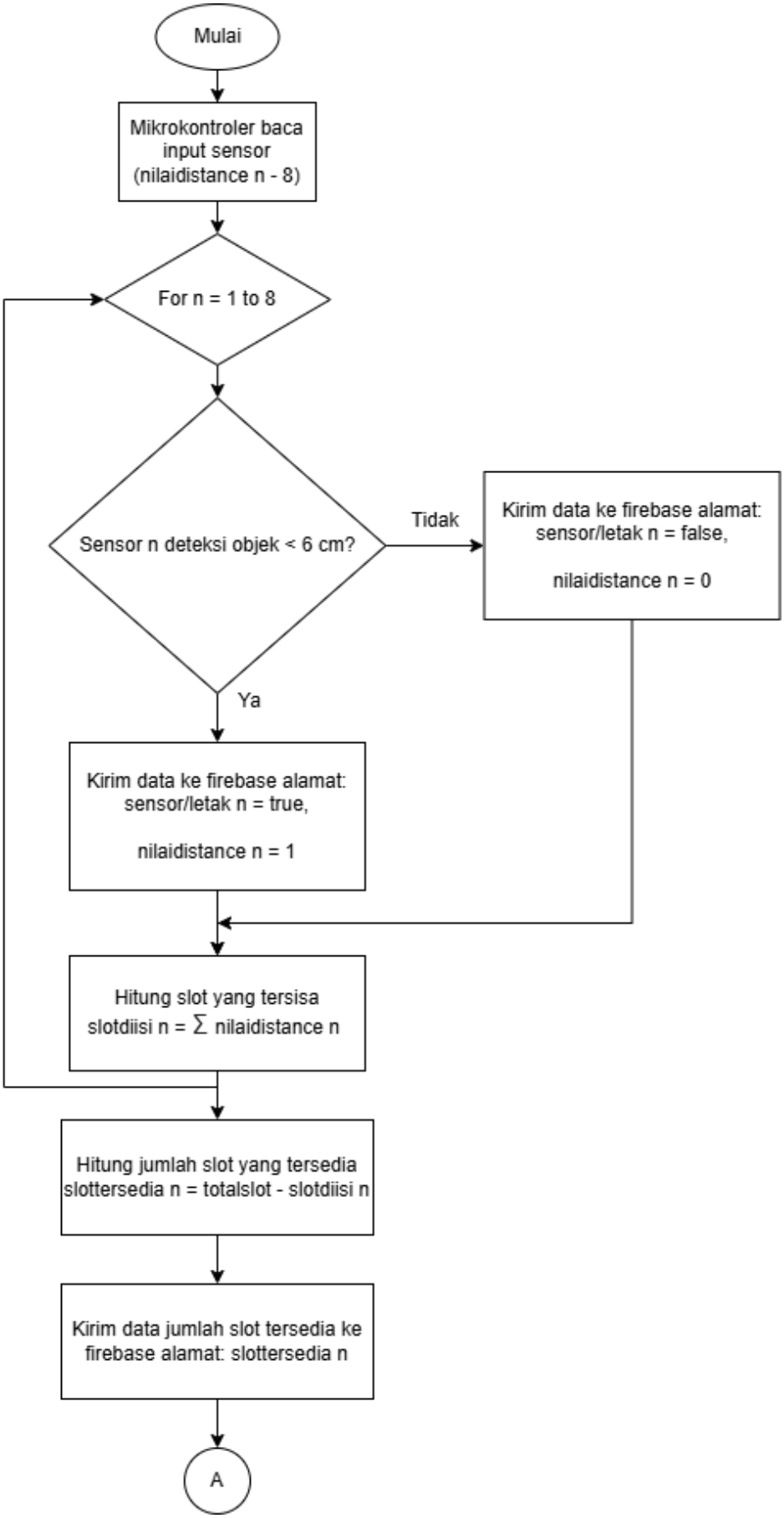
J = Jumlah sensor yang dipakai pada suatu lantai.

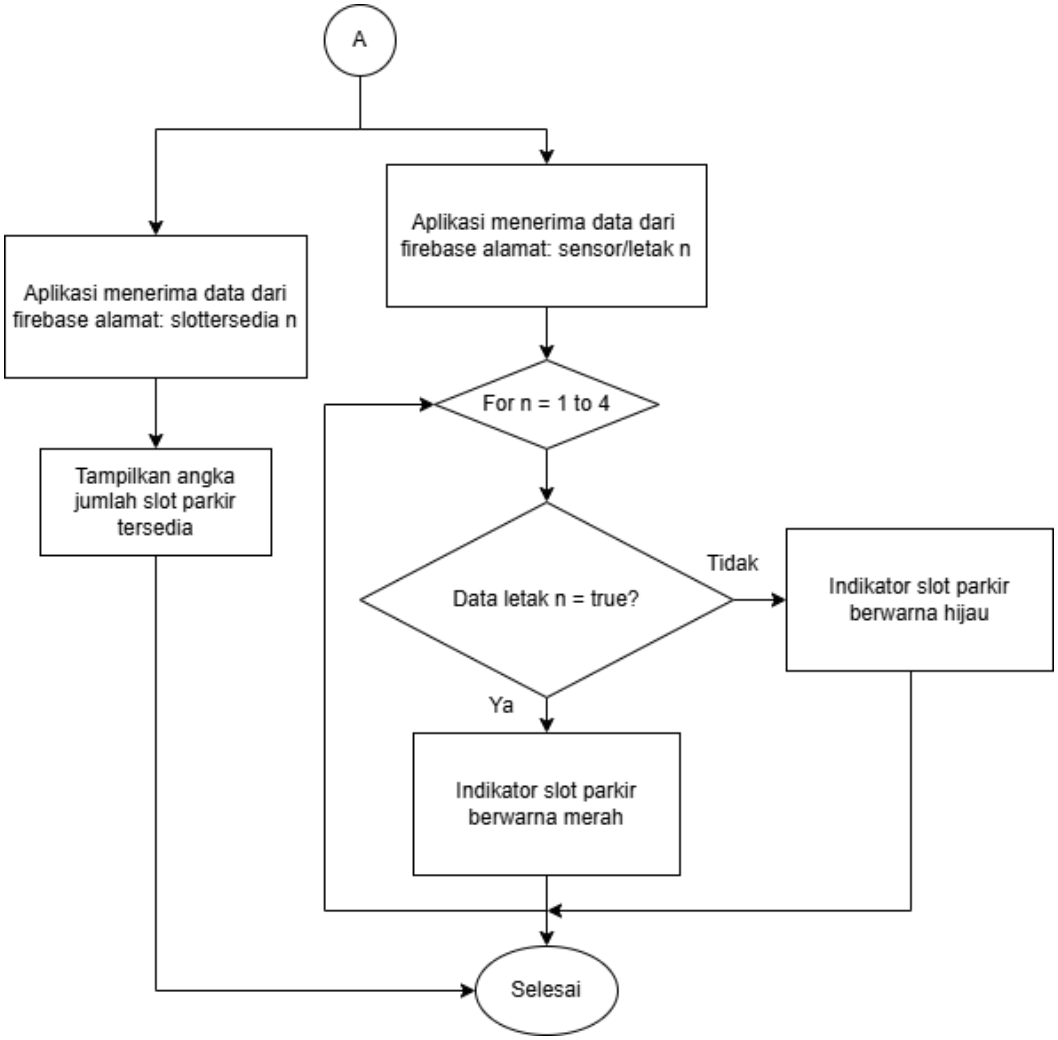
T = Jumlah slot terisi.

Setelah mendapatkan data slot terisi dan data slot tersedia kemudian data tersebut dikirimkan oleh prosesor ke *cloud database. Cloud database* akan memperbarui informasi jumlah slot parkir terisi dan tersedia lalu menampilkannya pada aplikasi.

3.5 Diagram Alir Sistem

Adapun tahapan dari sistem pada penelitian ini dibagi menjadi 2 tahap yaitu tahap perancangan perangkat keras (*hardware*) dan tahap perancangan perangkat lunak (*software*). Diagram alir sistem yang dirancang seperti pada Gambar 3.3.


Mulai
Mikrokontroler baca input sensor (nilaidistance n - 8)
For n = 1 to 8
Sensor n deteksi objek < 6 cm?
Tidak
Kirim data ke firebase alamat: sensor/letak n = false,
nilaidistance n = 0
Ya
Kirim data ke firebase alamat: sensor/letak n = true,
nilaidistance n = 1
Hitung slot yang tersisa
slotdiisi n = ∑ nilaidistance n
Hitung jumlah slot yang tersedia
slottersedia n = totalslot - slotdiisi n
Kirim data jumlah slot tersedia ke firebase alamat: slottersedia n
A



Gambar 3.3. Diagram Alir Sistem

Diagram alur pada Gambar 3.3 menggambarkan proses sistem pendeteksian objek dengan mikrokontroler yang mengukur jarak dari sensor dan kemudian mengirimkan datanya ke Firebase untuk mencatat slot parkir yang terisi dan tersedia. Proses dimulai ketika mikrokontroler membaca input dari sensor yang mendeteksi apakah ada objek dalam jarak kurang dari 6 cm. Jika ada objek yang terdeteksi, data yang dikirim ke Firebase menunjukkan bahwa slot parkir tersebut terisi (nilai `true`), dan menyimpan data slot terisi bertambah 1. Jika tidak ada objek yang terdeteksi, data yang dikirim menunjukkan bahwa slot tersebut tersedia (nilai `false`), dan menyimpan data slot terisi bertambah 0.

Selanjutnya, mikrokontroler menghitung jumlah slot yang terisi dengan menjumlahkan data slot terisi yang telah disimpan dari setiap sensor. Sistem kemudian menghitung jumlah slot yang masih tersedia dengan mengurangi total slot yang ada dengan slot yang sudah terisi. Setelah perhitungan selesai, data jumlah slot yang tersedia dikirim ke Firebase.

Pada tahap berikutnya, aplikasi menerima data dari Firebase, baik untuk jumlah slot yang tersedia maupun status setiap sensor (terisi atau tersedia). Berdasarkan data ini, aplikasi menampilkan jumlah slot parkir yang tersedia kepada pengguna. Selain itu, jika data menunjukkan bahwa suatu slot terisi (`true`), indikator parkir akan berwarna merah. Jika slot kosong (`false`), indikator akan berwarna hijau. Proses ini berulang untuk setiap sensor sampai semua slot parkir diperiksa, dan sistem akan menampilkan informasi secara *real-time* kepada pengguna.

3.5.1 Device ESP32 sebagai prosesor dan perangkat IoT

Fungsi utama esp32 adalah sebagai prosesor untuk menghitung slot parkir yang tersedia dan mengirim data ke database berdasarkan prinsip *Internet of Things.* Untuk program menghubungkan prosesor pada internet serta *database* dan menghitung slot parkir seperti berikut:

1. Program untuk koneksi internet dan *database*

```
#define WIFI_SSID "UNILA-PGN-22"
#define WIFI_PASSWORD ""
#define API_KEY "AIzaSyAASGwEZit_J-Rc1S0yOivXpsR2o5oDw3E"
#define DATABASE_URL “https://parkirslotiot-default-rtdb.asia-southeast1.firebasedatabase.app”
  void setup()
{
WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.print(".");
```

```
  delay(300);
 }
 Serial.println();
 Serial.print("Connected with IP: ");
 Serial.println(WiFi.localIP());
 Serial.println();
 Serial.printf("Firebase Client v%s\n\n", FIREBASE_CLIENT_VERSION);
 config.api_key = API_KEY;
 config.database_url = DATABASE_URL;
 if(Firebase.signUp(&config, &auth, "", "")){
  Serial.println("signUp OK");
  signUpOK = true;
 }else{
  Serial.printf("%s/n", config.signer.signupError.message.c_str());
 }
 config.token_status_callback = tokenStatusCallback;
 Firebase.begin(&config, &auth);
 Firebase.reconnectNetwork(true);
}
```

Program tersebut difungsikan untuk menghubungkan prosesor dengan internet menggunakan wifi UNILA-PGN-22 dan menghubungkan perangkat dengan *database* menggunakan *API KEY* AIzaSyAASGwEZit_J-Rc1S0yOivXpsR2o5oDw3E serta alamat url database https://parkirslotiot-default-rtdb.asia-southeast1.firebasedatabase.app. Kemudian setelah berhasil terhubung dengan *internet* akan menampilkan alamat IP wifi yang digunakan pada serial monitor. Dan setelah itu jika berhasil terhubung dengan *database* akan menampilkan signUp OK pada serial monitor.

2. Program untuk deteksi sensor dan mengirim data letak

```
long duration1, distance1;
digitalWrite(trigPin1, LOW);
delayMicroseconds(2);
digitalWrite(trigPin1, HIGH);
```

```
delayMicroseconds(10);
digitalWrite(trigPin1, LOW);
duration1 = pulseIn(echoPin1, HIGH);
distance1 = (duration1 * 0.0343) / 2;

long duration2, distance2;
digitalWrite(trigPin2, LOW);
delayMicroseconds(2);
digitalWrite(trigPin2, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin2, LOW);
duration2 = pulseIn(echoPin2, HIGH);
distance2 = (duration2 * 0.0343) / 2;

 if (distance1 < 6)
    {Firebase.RTDB.setBool(&fbdo, "sensor/letak1", true);
     nilaidistance1 = 1;
    } else {
       Firebase.RTDB.setBool(&fbdo, "sensor/letak1", false);
       nilaidistance1 = 0;
      }
       Serial.print("Slot 1: ");
       Serial.println(nilaidistance1);

    if (distance2 < 6)
    {Firebase.RTDB.setBool(&fbdo, "sensor/letak2", true);
     nilaidistance2 = 1;
      } else {
       Firebase.RTDB.setBool(&fbdo, "sensor/letak2", false);
       nilaidistance2 = 0;
      }
       Serial.print("Slot 2: ");
```

```
        Serial.println(nilaidistance2);
    }
```

Program tersebut difungsikan untuk sensor ultrasonik mendeteksi kendaraan yang terparkir pada slot parkir. trigPin1 yaitu pin *trigger* sensor ultrasonik 1 dan echoPin1 yaitu pin *receiver* sensor ultrasonik 1, begitupun untuk trigPin2 dan echoPin2 untuk sensor ultrasonik 2. Keadaan awal pin *trigger* LOW atau sinyal rendah sensor ultrasonik belum mengirimkan sinyal *trigger,* lalu pin *trigger* dalam keadaan HIGH atau sinyal tinggi sensor ultrasonik sudah mengirimkan sinyal *trigger,* lalu pin *trigger* dalam keadaan LOW kembali. Kemudian dilakukan pemanggilan pulseln untuk membaca pulsa HIGH atau LOW dan menunggu pulsa dari LOW ke HIGH untuk mulai pemancaran oleh *trigger* dan menunggu diterima oleh *receiver* untuk kembali ke LOW untuk berhenti pemancaran, lalu dibaca jika ada kendaraan kurang dari 6 cm akan mengirimkan data true ke *database* dan menyimpan data bernilai 1 pada sensor ultrasonik. Apabila tidak ada kendaraan kurang dari 6 cm akan mengirimkan data false ke *database* dan menyimpan data bernilai 0 pada sensor ultrasonik pada nilaidistance1, nilaidistance2, nilaidistance3, nilaidistance4, nilaidistance5, nilaidistance6, nilaidistance7, dan nilaidistance8. Lalu menampilkan data tersebut pada serial monitor.

3. Program menghitung jumlah slot parkir terisi dan tersedia

```
slotdiisi1 = ( nilaidistance1 + nilaidistance2 + nilaidistance3 + nilaidistance4);

slottersedia1 = totalslot1 - slotdiisi1;

slotdiisi2 = ( nilaidistance5 + nilaidistance6 + nilaidistance7 + nilaidistance8);

slottersedia2 = totalslot2 - slotdiisi2;

 Serial.print("Slot tersedia 1: ");

 Serial.println(slottersedia1);

 Serial.print("Slot tersedia 2: ");

 Serial.println(slottersedia2);
```

```
Firebase.RTDB.setInt(&fbdo, "slotdiisi1", slotdiisi1);
Firebase.RTDB.setInt(&fbdo, "slotdiisi2", slotdiisi2);
Firebase.RTDB.setInt(&fbdo, "slottersedia1", slottersedia1);
Serial.println("Data slottersedia1 berhasil terkirim.");
Firebase.RTDB.setInt(&fbdo, "slottersedia2", slottersedia2);
Serial.println("Data slottersedia2 berhasil terkirim.");
}
```

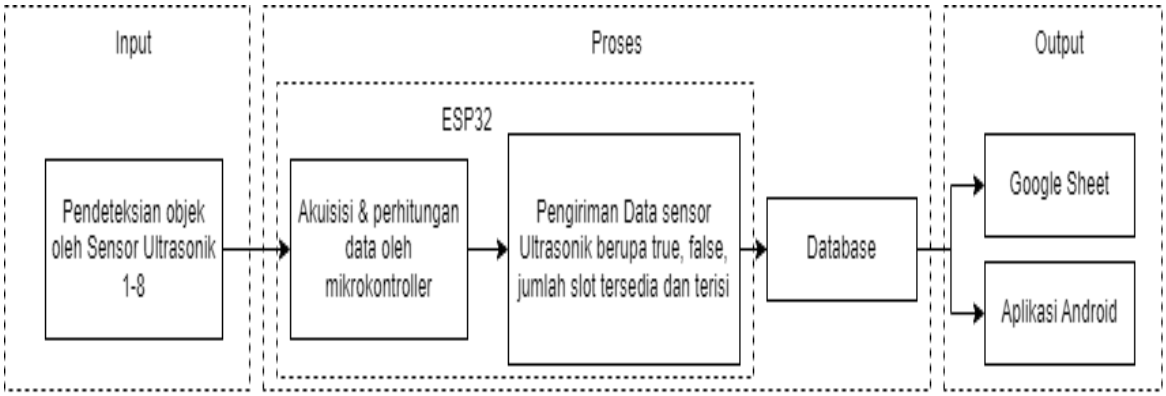
Setelah menyimpan data sensor ultrasonik 1 atau 0 pada program sebelumnya, kemudian dilakukan perhitungan pada program diatas. slotdiisi1 yaitu jumlah slot yang terisi pada lantai 1 dan slottersedia1 yaitu jumlah slot yang tersedia pada lantai 1, begitupun slotdiisi2 yaitu jumlah slot yang terisi pada lantai 2 dan slottersedia2 yaitu jumlah slot yang tersedia pada lantai 2. nilaidistance1 yaitu data 1 atau 0 sensor ultrasonik 1 yang telah disimpan, begitupun nilaidistance2 yaitu data nilai ultrasonik 2, nilaidistance3 yaitu data nilai ultrasonik 3, nilaidistance4 yaitu data nilai ultrasonik 4, nilaidistance5 yaitu data nilai ultrasonik 5, nilaidistance6 yaitu data nilai ultrasonik 6, nilaidistance7 yaitu data nilai ultrasonik 7, dan nilaidistance8 yaitu data nilai ultrasonik 8. Untuk menghitung jumlah slot terisi yaitu dengan menjumlahkan semua data sensor ultrasonik yang digunakan pada lantai tersebut yang mana data telah didapat pada program sebelumnya. Setelah mendapatkan nilai jumlah slot terisi dapat menghitung jumlah slot tersedia dengan mengurangkan total slot dengan jumlah slot terisi. Kemudian data slot terisi dan slot tersedia yang telah didapat dikirim ke *database* dan ditampilkan pada serial monitor.

3.5.2 Perancangan *Hardware*

Perancangan *hardware* merupakan perancangan untuk pembuatan alat pada penelitian. Pada perancangan *hardware* terdapat blok diagram alat dan skema perancangan.

3.5.2.1 Diagram Blok Alat

Diagram blok alat dirancang agar dapat diketahui *input* dan *output* proses pada alat penelitian. Blok diagram alat ditunjukan pada Gambar 3.4.

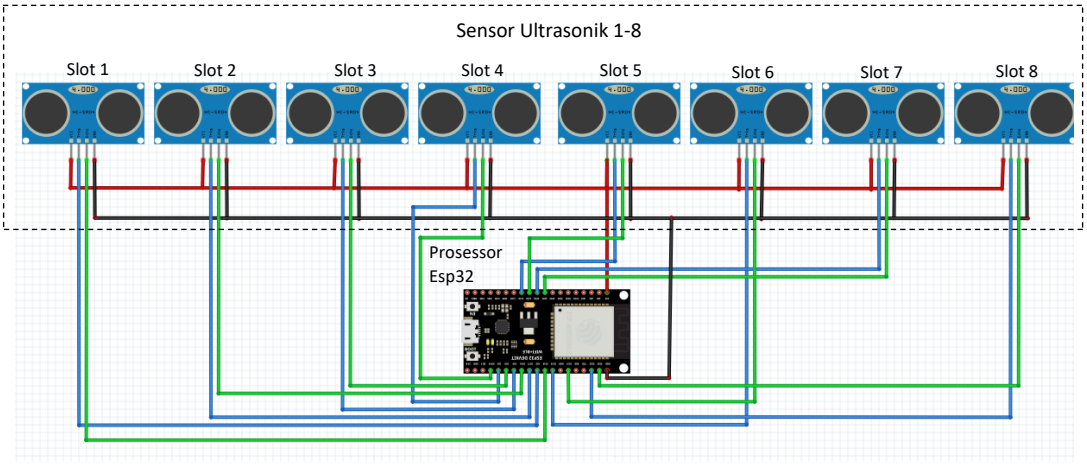


Gambar 3.4. Diagram Blok Alat

Berdasarkan Gambar 3.4 dilihat bahwa sistem memiliki *input* yaitu 8 sensor ultrasonik HC-SR04 yang terbagi menjadi 2 lantai masing-masing terdapat 4 sensor pada setiap lantai. Kemudian data hasil pembacaan sensor akan diproses pada mikrokontroler ESP32. Data yang telah diproses oleh mikrokontroler kemudian dikirimkan ke *database* untuk disimpan pada *Google Sheet* dan ditampilkan pada aplikasi yang telah dibuat. Proses yang dihasilkan akan menginformasikan kepada pengguna *slot* parkir yang tersedia.

3.5.2.2 Skema Perancangan Alat

Skema perancangan alat merupakan perancangan *wiring* yang diterapkan pada alat penelitian. Skema perancangan alat penelitian ditunjukan pada Gambar 3.5.



Gambar 3.5. Skematik Perancangan Alat

Berdasarkan Gambar 3.5 yang merupakan skematik perancangan alat pada sistem parkir. Pada perancangan sistem menggunakan mikrokontroler ESP32 untuk mengolah dan mengirim data dari pembacaan sensor ultrasonik HC-SR04 ke *database*. Data yang diterima *database* akan tertampil pada aplikasi dan dapat dipantau secara *real-time* pada aplikasi yang dibuat sehingga pengguna dapat mengambil tindakan memarkir kendaraan pada gedung parkir yang memiliki slot kosong.

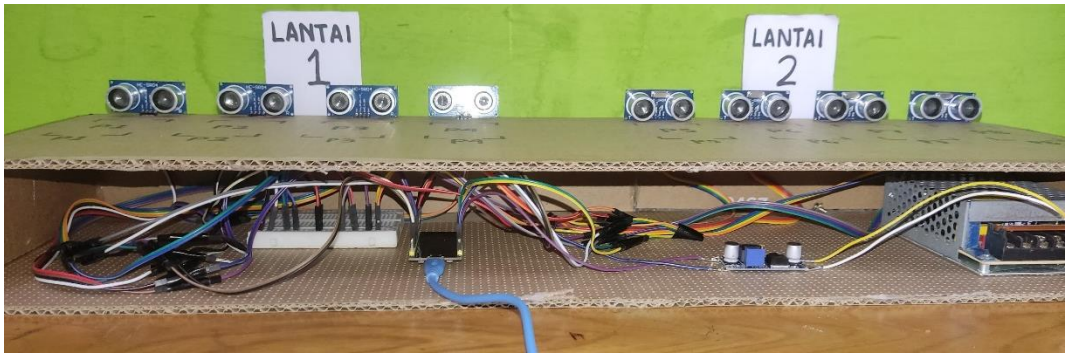
Desain perancangan alat berdasarkan skematik Gambar 3.5. menggunakan 8 buah sensor ultrasonik untuk mendeteksi kendaraan yang parkir pada setiap slot parkir. Apabila terdapat sensor ultrasonik yang mendeteksi adanya kendaraan terparkir maka prosesor akan menghitung jumlah slot terisi dan slot tersedia setelah mendapat data tersebut kemudian dikirim ke *database.*

Dari rangkaian skematik pada Gambar 3.5. didapatkan desain alat seperti pada Gambar 3.6. Pada Gambar 3.6.(a) dapat dilihat terdapat desain keseluruhan alat diasumsikan bahwa 8 buah sensor ultrasonik HC-SR04 yang dipasang terbagi menjadi 2 lantai dengan masing-masing 4 sensor, kemudian pada Gambar 3.6.(b)

dapat dilihat terdapat di dalam alat yang berisikan prosesor untuk menghitung jumlah slot yang dapat terhubung dengan internet sehingga data dapat dikirim ke *database*.

Untuk hasil dari pembuatan alat keseluruhan perangkat keras dapat dilihat pada Gambar 3.6.

(a) Perangkat keras alat tampak luar



(b) Perangkat keras alat tampak dalam

Gambar 3.6. Perangkat keras sistem deteksi kendaraan pada slot parkir (a) tampak luar (b) tampak dalam

Pada Gambar 3.6. (a) Perangkat keras alat terdapat sensor ultrasonik sebagai pendeteksi kendaraan. (b) di dalam alat terdapat prosesor ESP32 sebagai penghitung jumlah slot terisi dan slot tersedia serta sebagai perangkat *internet of things*.

3.5.3 Perancangan Aplikasi

Perancangan aplikasi pada penelitian ini menggunakan Android Studio sebagai *framework* dan Firebase sebagai *database* yang digunakan. Proses perancangan memiliki 3 tahapan seperti ditunjukan pada Tabel 3.2.

Tabel 3.2. Tahapan Perancangan Aplikasi

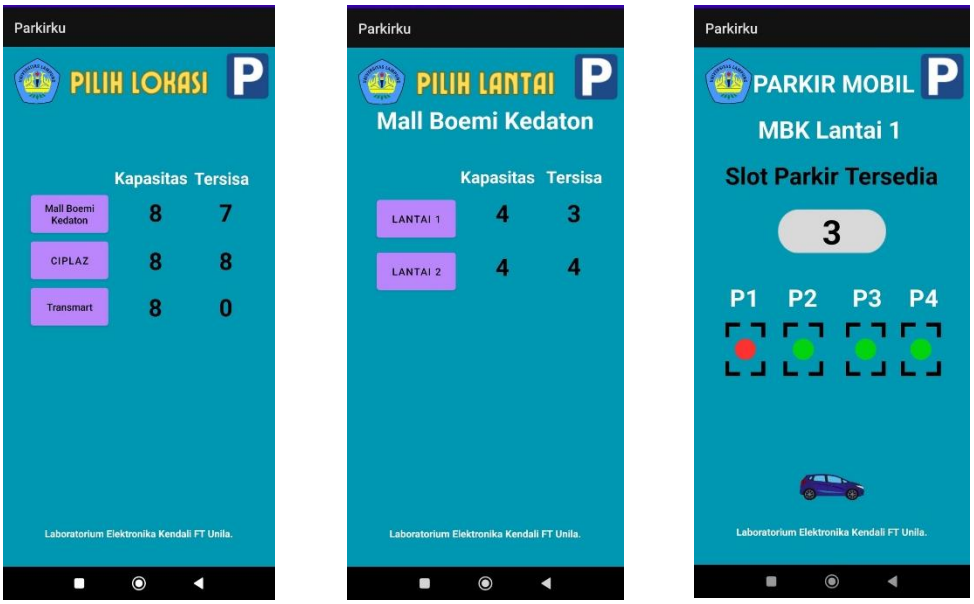
Tahapan	**Hal yang Dilakukan**
Requirements Engineering	Mengidentifikasi semua kebutuhan interface untuk pemantauan slot parkir dan teknologi yang diimplementasikan seperti protokol komunikasi menggunakan HTTPS dan API yang dibutuhkan untuk mengambil dan mengirim data dari sensor ke sistem.
Design Implementation	Menginstal semua kebutuhan serta merancang dan mendesain kebutuhan interface untuk aplikasi yang akan digunakan dengan mempertimbangkan elemen-elemen seperti tata letak, warna, ikon, tombol, dan alur navigasi untuk memastikan antarmuka pengguna yang intuitif dan ramah pengguna.
Testing	Menguji aspek-aspek fungsionalitas sistem tanpa mengetahui detail implementasi internalnya. Pengujian ini dilakukan dari perspektif pengguna untuk memastikan bahwa sistem memenuhi kebutuhan pengguna dengan benar. Pengujian yang dilakukan yaitu, uji splash screen, uji tombol, uji data jumlah slot parkir tersedia, dan uji tata letak slot parkir hijau atau merah.

Berdasarkan Tabel 3.2 yang merupakan tahapan dalam perancangan aplikasi yang berfungsi sebagai *interface* untuk memantau slot parkir yang ada di lahan parkir. Tahap perancangan aplikasi parkir dimulai dengan tahapan *Requirements Engineering*, di mana semua kebutuhan sistem diidentifikasi dengan cermat. Ini mencakup kebutuhan interface dan teknologi, yang akan menentukan bagaimana sistem berinteraksi dengan pengguna, dan teknologi apa yang akan digunakan untuk membangunnya. Tahap berikutnya adalah *Design* dan Implementasi, di mana kebutuhan sistem diterjemahkan menjadi produk yang berfungsi. Ini melibatkan perancangan sistem untuk menentukan arsitektur dan koneksi komponen, pengembangan aplikasi dan sistem dengan menulis kode dan melakukan pengujian,

serta desain interface agar mudah digunakan. Setelah itu, tahap *Testing* dilakukan untuk memastikan bahwa sistem berfungsi dengan benar dan berkinerja efisien tanpa membuang waktu atau sumber daya. Dengan mengikuti ketiga tahap ini, perancangan aplikasi parkir dapat menghasilkan solusi yang memenuhi kebutuhan pengguna dan beroperasi dengan lancar.

3.5.3.1 Desain Tampilan Aplikasi

Desain tampilan aplikasi yang telah dirancang dapat dilihat pada Gambar 3.6.



(a) Halaman pilih lokasi (b) halaman pilih lantai (c) halaman slot tersedia

Gambar 3.7. Desain Tampilan Aplikasi

Pada Gambar 3.7 merupakan desain tampilan aplikasi yang mana terdapat halaman memilih lokasi mall yang akan dilihat ketersediaan slot parkirnya. Setelah memilih lokasi kemudian akan masuk ke halaman memilih lantai. Kemudian setelah memilih lantai akan terdapat halaman slot parkir yang tersedia, pada halaman ini indikator slot parkir yang tersedia akan berwarna merah sedangkan indikator terisi akan berwarna merah dan terdapat jumlah slot parkir yang tersedia.

3.5.3.2 Android Studio

Android Studio adalah sebagai *framework* untuk pembuatan aplikasi yang telah dirancang menggunakan bahasa pemrograman java, kotlin dan lain sebagainya. Untuk program pada aplikasi yang dirancang sebagai berikut:

1. Program java pada halaman pilih lokasi seperti Gambar 3.7. (a).

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    tersisa1 = (TextView) findViewById(R.id.tersisa1);
    tersisa2 = (TextView) findViewById(R.id.tersisa2);
    mRef = new Firebase("https://parkirslotiot-default-rtdb.asia-southeast1.firebasedatabase.app/slotdiisi1");
    mRef2 = new Firebase("https://parkirslotiot-default-rtdb.asia-southeast1.firebasedatabase.app/slotdiisi2");
    mRef.addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {
            String value = dataSnapshot.getValue(String.class);

            if (value != null) {
                try {
                    int intValue = Integer.parseInt(value);
                    int sisa = 8 - intValue;

                    tersisa1.setText(String.valueOf(sisa));
                } catch (NumberFormatException e) {
                    // Handle jika value tidak dapat diubah menjadi integer
                }
            }
        }
```

```
        @Override
        public void onCancelled(FirebaseError firebaseError) {
        }
    });
    mRef2.addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {
            String value2 = dataSnapshot.getValue(String.class);

            if (value2 != null) {
                try {
                    int intValue2 = Integer.parseInt(value2);
                    int sisa2 = 8 - intValue2;
                    tersisa2.setText(String.valueOf(sisa2));
                } catch (NumberFormatException e) {
                    // Handle jika value tidak dapat diubah menjadi integer
                }
            }
        }
        @Override
        public void onCancelled(FirebaseError firebaseError) {
        }
    });
}
public void lokasi(View view){
    Intent intent = new Intent(MainActivity.this,MainActivity3.class);
    startActivity(intent);
}
public void lokasi2(View view){
    Intent intent = new Intent(MainActivity.this,MainActivity5.class);
    startActivity(intent);
}
```

```
public void lokasi3(View view) {
    Intent intent = new Intent(MainActivity.this, MainActivity8.class);
    startActivity(intent);
}
```

Program tersebut difungsikan untuk fungsi pada halaman pilih lokasi, pada program terdapat fungsi untuk menerima data dari *database* berupa data slot terisi pada semua lantai setiap lokasi kemudian diproses dan dihitung untuk ditampilkan jumlah slot tersedia pada lokasi tersebut. Pada program juga terdapat fungsi *button* dan *button* tersebut digunakan pada masing-masing lokasi sebagai pemindah halaman. Jadi saat *button* tersebut ditekan pada salah satu lokasi maka halaman akan berpindah ke halaman selanjutnya yaitu halaman pilih lantai.

2. Program java pada halaman pilih lantai seperti Gambar 3.7. (b).

```
@Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main3);
        tersisa1 = (TextView) findViewById(R.id.tersisa1);
        tersisa2 = (TextView) findViewById(R.id.tersisa2);
        mRef = new Firebase("https://parkirslotiot-default-rtdb.asia-
southeast1.firebasedatabase.app/slotdiisi1");
        mRef2 = new Firebase("https://parkirslotiot-default-rtdb.asia-
southeast1.firebasedatabase.app/slotdiisi2");
        mRef.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(DataSnapshot dataSnapshot) {
                String value = dataSnapshot.getValue(String.class);
                if (value != null) {
                    try {
                        int intValue = Integer.parseInt(value);
                        int sisa = 4 - intValue;
                        tersisa1.setText(String.valueOf(sisa));
                    } catch (NumberFormatException e) {
```

```
                    // Handle jika value tidak dapat diubah menjadi integer
                }
            }
        }
        @Override
        public void onCancelled(FirebaseError firebaseError) {
        }
    });
    mRef2.addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {
            String value = dataSnapshot.getValue(String.class);
            if (value != null) {
                try {
                    int intValue = Integer.parseInt(value);
                    int sisa = 4 - intValue;
                    tersisa2.setText(String.valueOf(sisa));
                } catch (NumberFormatException e) {
                    // Handle jika value tidak dapat diubah menjadi integer
                }
            }
        }
        @Override
        public void onCancelled(FirebaseError firebaseError) {
        }
    });
}
public void lantai(View view){
    Intent intent = new Intent(MainActivity3.this,MainActivity2.class);
    startActivity(intent);
  }
  public void lantai2(View view) {
```

```
        Intent intent = new Intent(MainActivity3.this, MainActivity4.class);
        startActivity(intent);
    }
```

Program tersebut difungsikan untuk fungsi pada halaman pilih lantai, pada program terdapat fungsi untuk menerima data dari *database* berupa data slot tersedia pada lantai tersebut kemudian ditampilkan jumlah slot tersedia pada lokasi dan lantai tersebut. Pada program juga terdapat fungsi *button* dan *button* tersebut digunakan pada masing-masing lantai sebagai pemindah halaman. Jadi saat *button* tersebut ditekan pada salah satu lokasi maka halaman akan berpindah ke halaman selanjutnya yaitu halaman slot tersedia dan letak slot parkir.

3. Program java pada halaman slot tersedia dan letak slot parkir seperti Gambar 3.7. (c).

```
@Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main2);
        //baca komponen nilai texview
        nilai = (TextView) findViewById(R.id.nilai) ;
        ada1 = findViewById(R.id.ada1);
        kosong1 = findViewById(R.id.kosong1);
        ada2 = findViewById(R.id.ada2);
        kosong2 = findViewById(R.id.kosong2);
        ada3 = findViewById(R.id.ada3);
        kosong3 = findViewById(R.id.kosong3);
        ada4 = findViewById(R.id.ada4);
        kosong4 = findViewById(R.id.kosong4);
        mRef = new Firebase("https://parkirslotiot-default-rtdb.asia-
southeast1.firebasedatabase.app/slotdiisi1") ;
        mRef2 = new Firebase("https://parkirslotiot-default-rtdb.asia-
southeast1.firebasedatabase.app/sensor/letak1");
        mRef3 = new Firebase("https://parkirslotiot-default-rtdb.asia-
southeast1.firebasedatabase.app/sensor/letak2");
```

```
mRef4 = new Firebase("https://parkirslotiot-default-rtdb.asia-southeast1.firebasedatabase.app/sensor/letak3");
mRef5 = new Firebase("https://parkirslotiot-default-rtdb.asia-southeast1.firebasedatabase.app/sensor/letak4");
mRef.addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {
        String value = dataSnapshot.getValue(String.class);
        if (value != null) {
            try {
                int intValue = Integer.parseInt(value);
                int sisa = 4 - intValue;
                nilai.setText(String.valueOf(sisa));
            } catch (NumberFormatException e) {
                // Handle jika value tidak dapat diubah menjadi integer
            }
        }
    }
    @Override
    public void onCancelled(FirebaseError firebaseError) {
    }
}) ;
mRef2.addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {
        boolean letak1 = dataSnapshot.getValue(Boolean.class);
        if (letak1 == false) {
            ada1.setVisibility(View.VISIBLE);
            kosong1.setVisibility(View.INVISIBLE);
        } else {
            ada1.setVisibility(View.INVISIBLE);
            kosong1.setVisibility(View.VISIBLE);
```

```
        }
    }
    @Override
    public void onCancelled(FirebaseError firebaseError) {
        // Handle error
    }
});
mRef3.addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {
        boolean letak2 = dataSnapshot.getValue(Boolean.class);
        if (letak2 == false) {
            ada2.setVisibility(View.VISIBLE);
            kosong2.setVisibility(View.INVISIBLE);
        } else {
            ada2.setVisibility(View.INVISIBLE);
            kosong2.setVisibility(View.VISIBLE);
        }
    }
    @Override
    public void onCancelled(FirebaseError firebaseError) {
        // Handle error
    }
});
mRef4.addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {
        boolean letak3 = dataSnapshot.getValue(Boolean.class);
        if (letak3 == false) {
            ada3.setVisibility(View.VISIBLE);
            kosong3.setVisibility(View.INVISIBLE);
        } else {
```

```
                ada3.setVisibility(View.INVISIBLE);
                kosong3.setVisibility(View.VISIBLE);
            }
        }
        @Override
        public void onCancelled(FirebaseError firebaseError) {
            // Handle error
        }
    });
    mRef5.addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {
            boolean letak4 = dataSnapshot.getValue(Boolean.class);
            if (letak4 == false) {
                ada4.setVisibility(View.VISIBLE);
                kosong4.setVisibility(View.INVISIBLE);
            } else {
                ada4.setVisibility(View.INVISIBLE);
                kosong4.setVisibility(View.VISIBLE);
            }
        }
        @Override
        public void onCancelled(FirebaseError firebaseError) {
            // Handle error
        }
    });
}
```

Program tersebut difungsikan untuk fungsi pada halaman slot tersedia dan letak slot parkir, pada program terdapat fungsi untuk menerima data dari *database* berupa data slot tersedia dan data letak slot parkir kemudian ditampilkan jumlah slot tersedia. Dan untuk data letak slot parkir terdapat fungsi untuk mengindikasikan slot tersebut digunakan atau tidak. Saat aplikasi menerima data letak slot parkir

berupa true maka letak slot parkir tersebut akan memiliki indikator berwarna merah yang menandakan bahwa slot parkir tersebut terdapat mobil yang terparkir. Apabila aplikasi menerima data letak slot parkir berupa false maka letak slot parkir tersebut akan memiliki indikator berwarna hijau yang menandakan bahwa slot parkir tersebut tidak terdapat mobil yang terparkir.

3.6 Pengujian Sistem

Pada tahap pengujian sistem dilakukan beberapa pengujian dengan indikator tertentu. Adapun pengujian sistem yang dilakukan dapat dilihat pada Tabel 3.3.

Tabel 3.3. Pengujian Sistem

Pengujian	**Indikator Keberhasilan**	**Keterangan**
Uji Fungsional Deteksi Sensor Ultrasonik	Sensor ultrasonik dapat mendeteksi kendaraan pada jarak tertentu	Menguji keberhasilan pembacaan sensor ultrasonik
Uji Integrasi Sensor ultrasonik dengan Aplikasi	Dapat menampilkan slot parkir terisi dengan indikator merah, slot parkir tersedia dengan indikator hijau dan menampilkan jumlah slot terisi dan tersedia	Melakukan pengujian pada aplikasi pada widget slot parkir
Uji latensi sistem	Dapat mengirimkan perubahan data kurang dari 10 detik	Melakukan pengukuran waktu perubahan data pada perangkat yang kemudian dikirim ke aplikasi

IV. HASIL DAN PEMBAHASAN

4.1 Prinsip Kerja

Penelitian ini merancang suatu alat yang berfungsi untuk mendeteksi kendaraan yang terparkir pada sebuah prototipe gedung parkir. Prototipe gedung parkir memiliki kapasitas 8 slot parkir yang terbagi menjadi 2 lantai dengan kapasitas masing-masing lantai 4 slot parkir. Ketika terdapat kendaraan yang terdeteksi pada slot parkir maka akan menampilkan pada aplikasi yang telah dirancang jumlah slot tersedia dan terisi, serta letak slot parkir yang digunakan dapat diketahui melalui aplikasi. Alat ini dirancang menggunakan 8 sensor Ultrasonik HC-SR04 dan ESP32 yang bekerja secara otomatis sesuai dengan batasan yang telah ditentukan pada program.

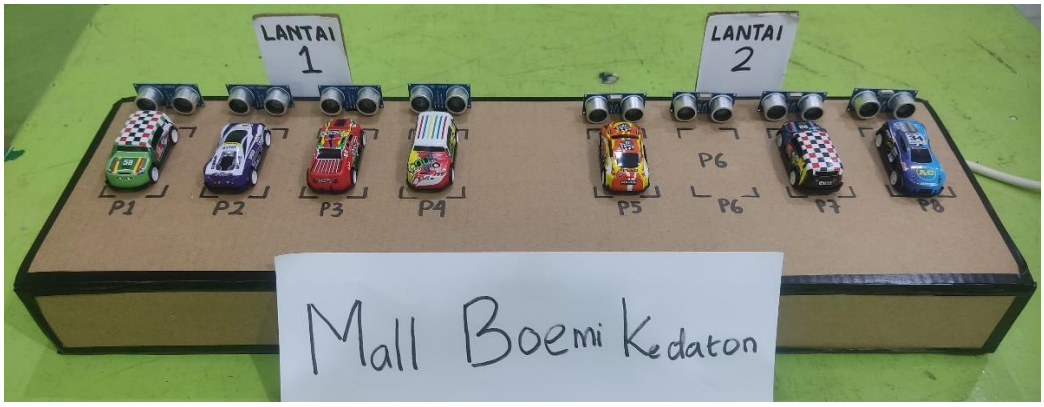
Perancangan aplikasi menggunakan Android Studio, Firebase sebagai *database* serta Google Sheets sebagai penyimpanan data. Perancangan untuk penelitian ini digunakan untuk menampilkan informasi ketersediaan slot parkir pada gedung parkir atau mall secara *real-time.*

4.2 Pengujian Subsistem

Penelitian ini dilakukan beberapa pengujian untuk mengetahui kemampuan dari *hardware* dan *software* yang digunakan pada prototipe lokasi parkir.

4.2.1 Pengujian *Hardware*

Pengujian *hardware* dilakukan terhadap setiap sensor yang digunakan sebagai alat penelitian supaya diketahui kemampuannya. Berikut prototipe lokasi parkir ditunjukan pada Gambar 4.1.



Gambar 4.1. Prototipe Alat Pemantau Ketersediaan Slot Parkir

Berdasarkan Gambar 4.1 merupakan prototipe alat pemantau ketersediaan slot parkir yang telah dibuat dengan terdapat 8 slot parkir dan terdapat 2 lantai yang masing-masing lantai terdapat 4 slot parkir. Pada masing-masing slot parkir ditempatkan sebuah sensor ultrasonik yang terhubung dengan mikrokontroler ESP32.

4.2.1.1 Pengujian Jarak antara Sensor HC-SR04 dan Mobil

Pada pengujian jarak antara sensor dengan mobil mempunyai tujuan untuk mengetahui seberapa jauh sensor HC-SR04 mendeteksi mobil pada penelitian ini. Mobil berukuran 1:100 dari mobil asli digunakan sebagai objek yang akan dideteksi. Pengujian ini dilakukan dengan meletakkan mobil di depan sensor HC-SR04 dengan jarak dimulai dari 1 cm hingga 95 cm. Pada Tabel 4.1 yang merupakan hasil pengujian, dapat dilihat seberapa jauh sensor HC-SR04 dapat mendeteksi mobil. Hasil pengujian dapat dilihat pada Tabel 4.1.

Berdasarkan Tabel 4.1 dapat dilihat bahwa pada jarak 1 cm tidak dapat mendeteksi mobil kemudian pada jarak 2 cm sampai dengan jarak 85cm sensor dapat mendeteksi mobil, sementara pada jarak 90 cm dan 95 cm sensor tidak dapat mendeteksi mobil. Pada jarak 1 cm sensor tidak dapat mendeteksi mobil karena sensor HC-SR04 memiliki titik buta dengan jarak terdekat sensor dengan objek berupa mobil mulai dari jarak 0cm hingga 2 cm. Sedangkan pada jarak 90 cm dan 95 cm sensor tidak dapat mendeteksi karena faktor objek mobil yang terlalu kecil. Karena berdasarkan datasheet sensor HC-SR04 dapat mendeteksi objek hingga 400 cm.

Tabel 4.1. Hasil Pengujian Jarak antara Sensor HC-SR04 dan Mobil

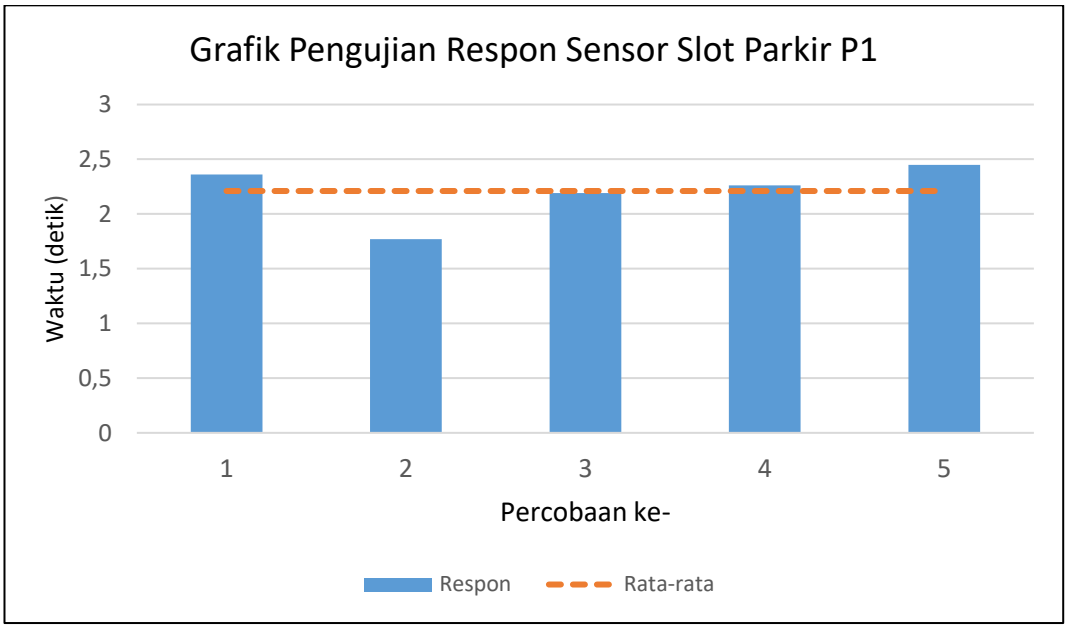
No	Jarak	Keterangan
1	1 cm	Tidak Terdeteksi
2	2 cm	Terdeteksi
3	3 cm	Terdeteksi
4	4 cm	Terdeteksi
5	5 cm	Terdeteksi
6	6 cm	Terdeteksi
7	7 cm	Terdeteksi
8	8 cm	Terdeteksi
9	9 cm	Terdeteksi
10	10 cm	Terdeteksi
11	30 cm	Terdeteksi
12	50 cm	Terdeteksi
13	85 cm	Terdeteksi
14	90 cm	Tidak Terdeteksi
15	95 cm	Tidak Terdeteksi

4.2.1.2 Pengujian Respon Sensor Ultrasonik HC-SR04

Pengujian respon pada sensor ultrasonik berfungsi untuk mengetahui waktu yang dibutuhkan oleh sistem untuk mendeteksi adanya mobil pada slot parkir, dimana *receiver* sensor ultrasonik menangkap pantulan sinyal ultrasound dari mobil yang dikirimkan oleh *transmitter* sensor ultrasonik. Pengujian ini dilakukan pada 8 sensor ultrasonik di setiap zona deteksi slot parkir dengan 5 kali percobaan. Pengukuran dilakukan pada saat mobil diletakkan pertama kali di slot parkir dan menekan mulai pada *stopwatch* secara bersamaan. Pengujian ini dilakukan dengan jarak < 6 cm dari sensor ultrasonik untuk setiap slot parkirnya. Pada saat sensor mendeteksi adanya mobil maka *buzzer* akan berbunyi dalam beberapa detik, waktu ini yang digunakan sebagai pengukuran respon sensor. Hasil pengujian respon sensor ultrasonik HC-SR04 dapat dilihat pada Tabel 4.2 – Tabel 4.9.

Tabel 4.2. Hasil pengujian respon sensor pada slot parkir P1

Percobaan	Waktu (detik)	Kondisi *Buzzer*
1	2,36	Hidup
2	1,77	Hidup
3	2,19	Hidup
4	2,26	Hidup
5	2,45	Hidup
Rata-rata = 2,21 detik		

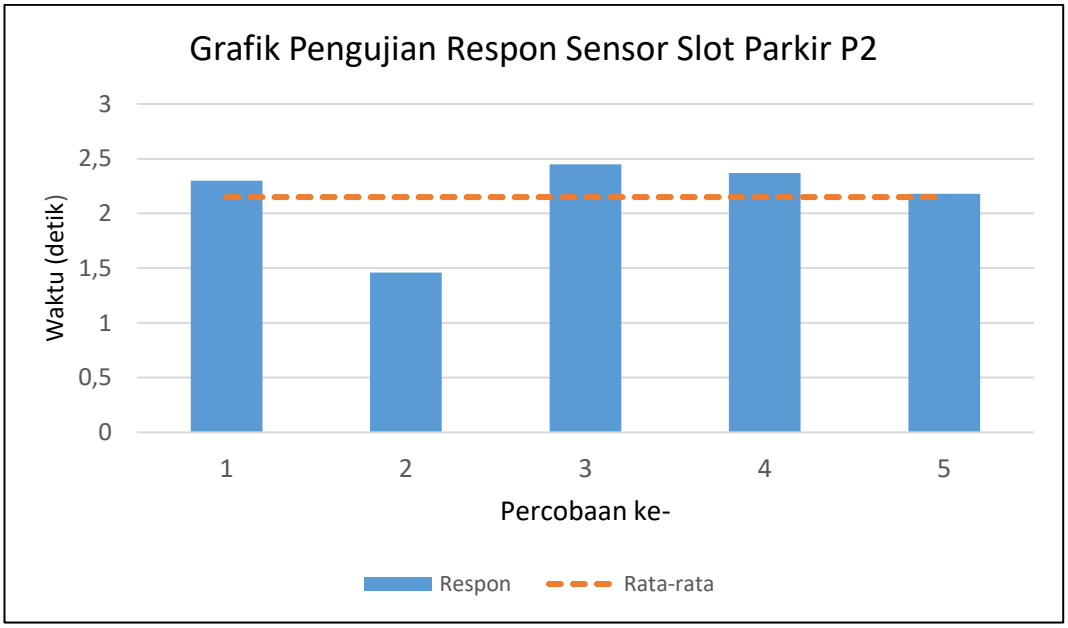


Gambar 4.2. Grafik Pengujian Respon Sensor Slot Parkir P1

Tabel 4.2 dan Gambar 4.2 merupakan hasil dan grafik pengujian respon sensor slot parkir P1 yang dilakukan dengan 5 kali percobaan. Pada percobaan 1 didapatkan respon 2,36 detik ketika *buzzer* hidup. Pada percobaan 2 didapatkan respon 1,77 detik ketika *buzzer* hidup. Pada percobaan 3 didapatkan respon 2,19 detik ketika *buzzer* hidup. Pada percobaan 4 didapatkan respon 2,26 detik ketika *buzzer* hidup. Pada percobaan 5 didapatkan respon 2,45 detik ketika *buzzer* hidup. Dapat disimpulkan bahwa respon sensor slot parkir P1 rata-rata seluruh percobaan yaitu sebesar 2,21 detik.

Tabel 4.3. Hasil pengujian respon sensor pada slot parkir P2

Percobaan	Waktu (detik)	Kondisi *Buzzer*
1	2,3	Hidup
2	1,46	Hidup
3	2,45	Hidup
4	2,37	Hidup
5	2,18	Hidup
Rata-rata = 2,15 detik		

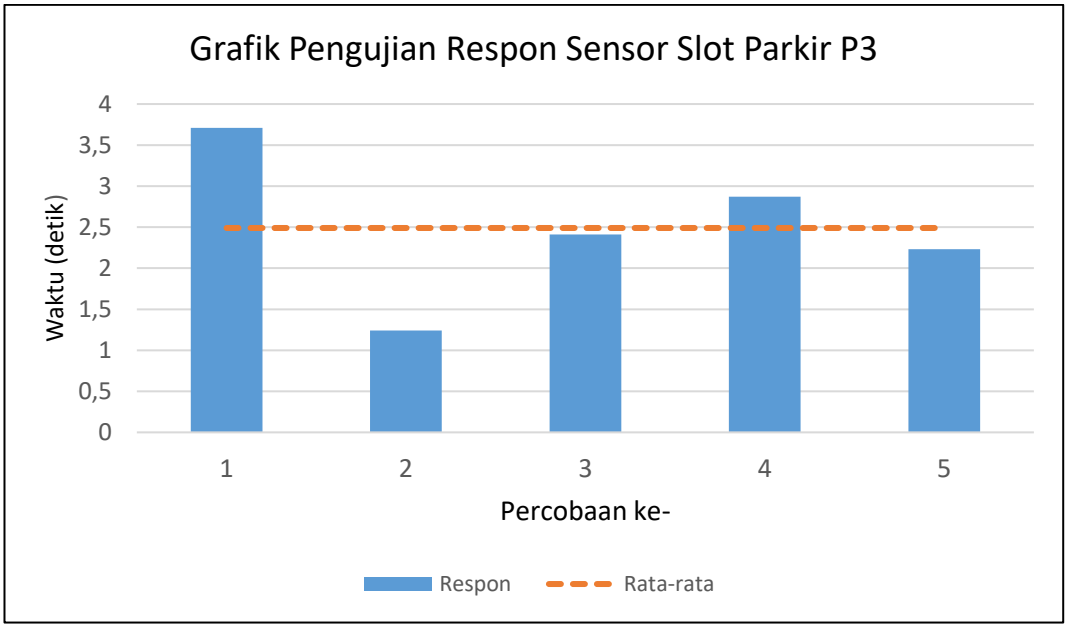


Gambar 4.3. Grafik Pengujian Respon Sensor Slot Parkir P2

Tabel 4.3 dan Gambar 4.3 merupakan hasil dan grafik pengujian respon sensor slot parkir P2 yang dilakukan dengan 5 kali percobaan. Pada percobaan 1 didapatkan respon 2,3 detik ketika *buzzer* hidup. Pada percobaan 2 didapatkan respon 1,46 detik ketika *buzzer* hidup. Pada percobaan 3 didapatkan respon 2,45 detik ketika *buzzer* hidup. Pada percobaan 4 didapatkan respon 2,37 detik ketika *buzzer* hidup. Pada percobaan 5 didapatkan respon 2,18 detik ketika *buzzer* hidup. Dapat disimpulkan bahwa respon sensor slot parkir P2 rata-rata seluruh percobaan yaitu sebesar 2,15 detik.

Tabel 4.4. Hasil pengujian respon sensor pada slot parkir P3

Percobaan	Waktu (detik)	Kondisi *Buzzer*
1	3,71	Hidup
2	1,24	Hidup
3	2,41	Hidup
4	2,87	Hidup
5	2,23	Hidup
Rata-rata = 2,49 detik		

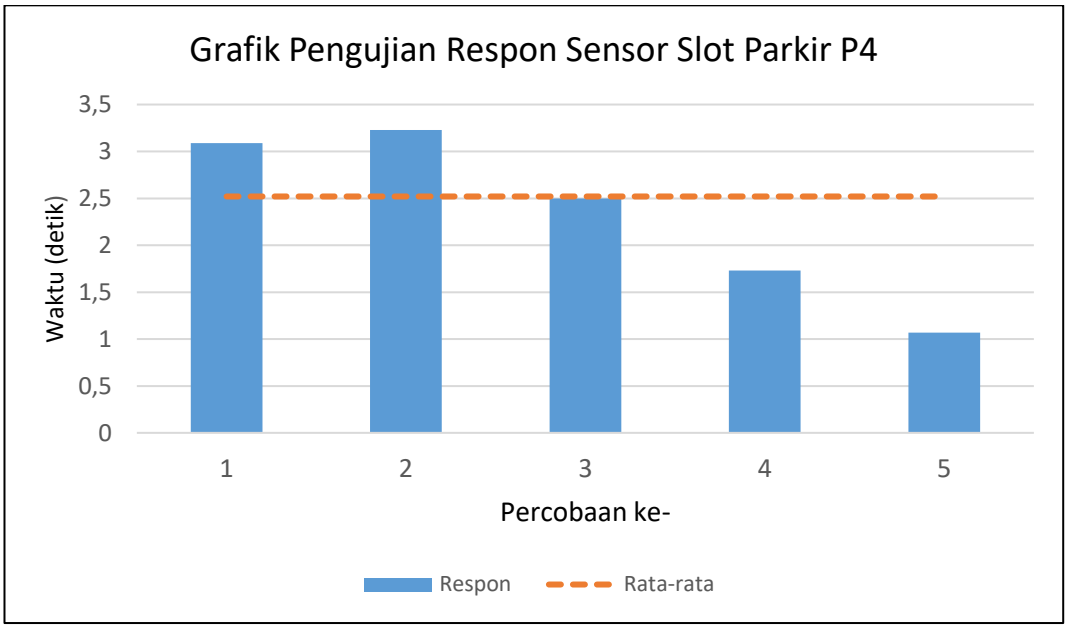


Gambar 4.4. Grafik Pengujian Respon Sensor Slot Parkir P3

Tabel 4.4 dan Gambar 4.4 merupakan hasil dan grafik pengujian respon sensor slot parkir P3 yang dilakukan dengan 5 kali percobaan. Pada percobaan 1 didapatkan respon 3,71 detik ketika *buzzer* hidup. Pada percobaan 2 didapatkan respon 1,24 detik ketika *buzzer* hidup. Pada percobaan 3 didapatkan respon 2,41 detik ketika *buzzer* hidup. Pada percobaan 4 didapatkan respon 2,87 detik ketika *buzzer* hidup. Pada percobaan 5 didapatkan respon 2,23 detik ketika *buzzer* hidup. Dapat disimpulkan bahwa respon sensor slot parkir P3 rata-rata seluruh percobaan yaitu sebesar 2,49 detik.

Tabel 4.5. Hasil pengujian respon sensor pada slot parkir P4

Percobaan	Waktu (detik)	Kondisi *Buzzer*
1	3,09	Hidup
2	3,23	Hidup
3	3,5	Hidup
4	1,73	Hidup
5	1,07	Hidup
Rata-rata = 2,52 detik		

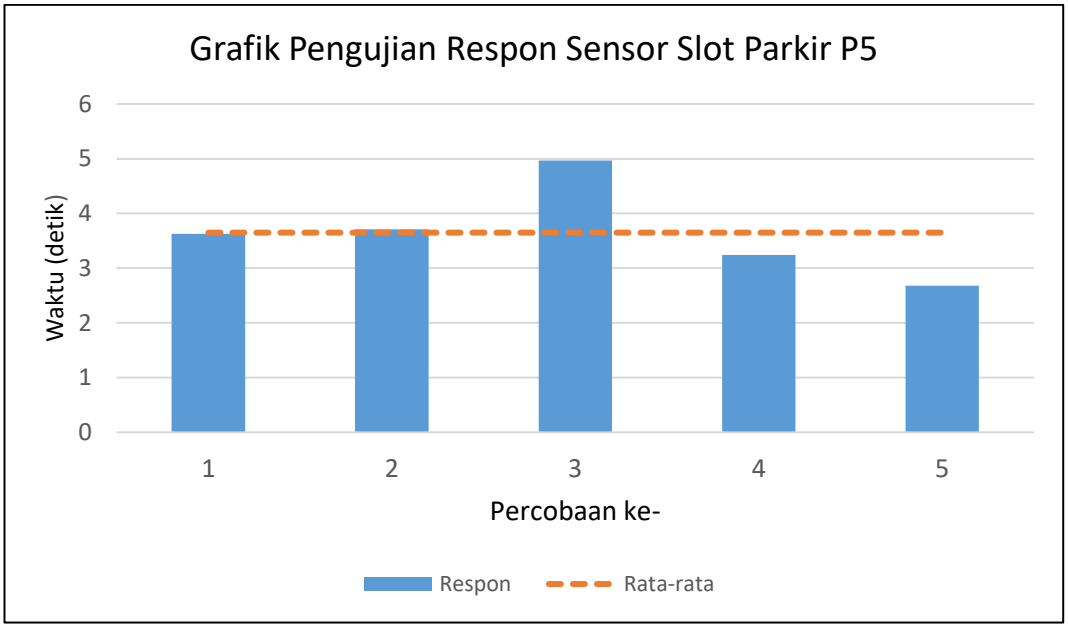


Gambar 4.5. Grafik Pengujian Respon Sensor Slot Parkir P4

Tabel 4.5 dan Gambar 4.5 merupakan hasil dan grafik pengujian respon sensor slot parkir P4 yang dilakukan dengan 5 kali percobaan. Pada percobaan 1 didapatkan respon 3,09 detik ketika *buzzer* hidup. Pada percobaan 2 didapatkan respon 3,23 detik ketika *buzzer* hidup. Pada percobaan 3 didapatkan respon 3,5 detik ketika *buzzer* hidup. Pada percobaan 4 didapatkan respon 1,73 detik ketika *buzzer* hidup. Pada percobaan 5 didapatkan respon 1,07 detik ketika *buzzer* hidup. Dapat disimpulkan bahwa respon sensor slot parkir P4 rata-rata seluruh percobaan yaitu sebesar 2,52 detik.

Tabel 4.6. Hasil pengujian respon sensor pada slot parkir P5

Percobaan	Waktu (detik)	Kondisi *Buzzer*
1	3,63	Hidup
2	3,71	Hidup
3	4,97	Hidup
4	3,24	Hidup
5	2,68	Hidup
Rata-rata = 3,65 detik		

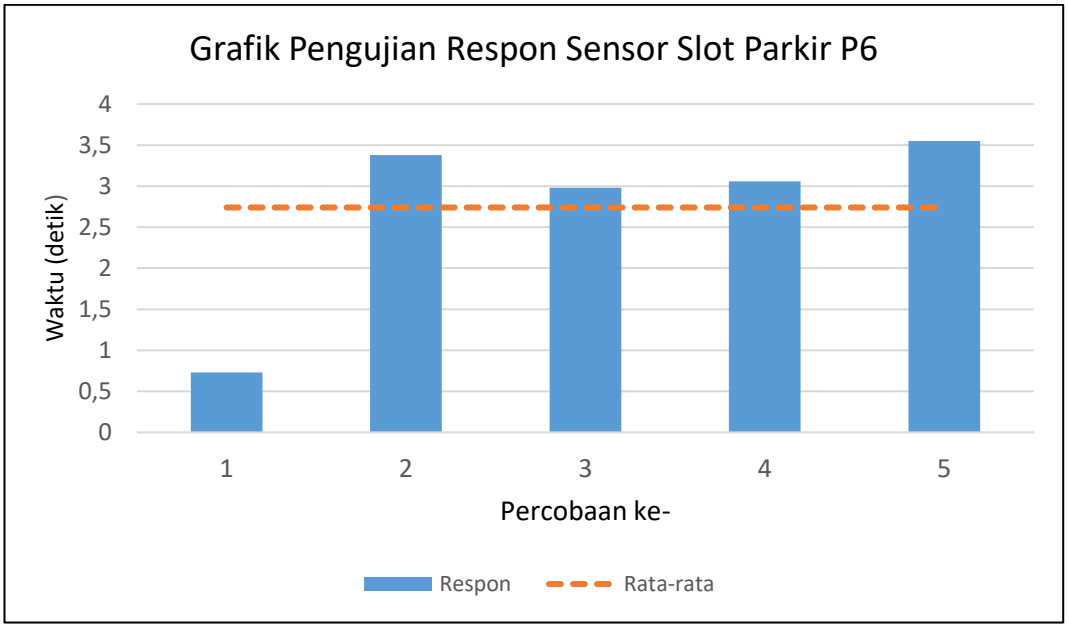


Gambar 4.6. Grafik Pengujian Respon Sensor Slot Parkir P5

Tabel 4.6 dan Gambar 4.6 merupakan hasil dan grafik pengujian respon sensor slot parkir P5 yang dilakukan dengan 5 kali percobaan. Pada percobaan 1 didapatkan respon 3,63 detik ketika *buzzer* hidup. Pada percobaan 2 didapatkan respon 3,71 detik ketika *buzzer* hidup. Pada percobaan 3 didapatkan respon 4,97 detik ketika *buzzer* hidup. Pada percobaan 4 didapatkan respon 3,24 detik ketika *buzzer* hidup. Pada percobaan 5 didapatkan respon 2,68 detik ketika *buzzer* hidup. Dapat disimpulkan bahwa respon sensor slot parkir P5 rata-rata seluruh percobaan yaitu sebesar 3,65 detik.

Tabel 4.7. Hasil pengujian respon sensor pada slot parkir P6

Percobaan	Waktu (detik)	Kondisi *Buzzer*
1	0,73	Hidup
2	3,38	Hidup
3	2,98	Hidup
4	3,06	Hidup
5	3,55	Hidup
Rata-rata = 2,74 detik		

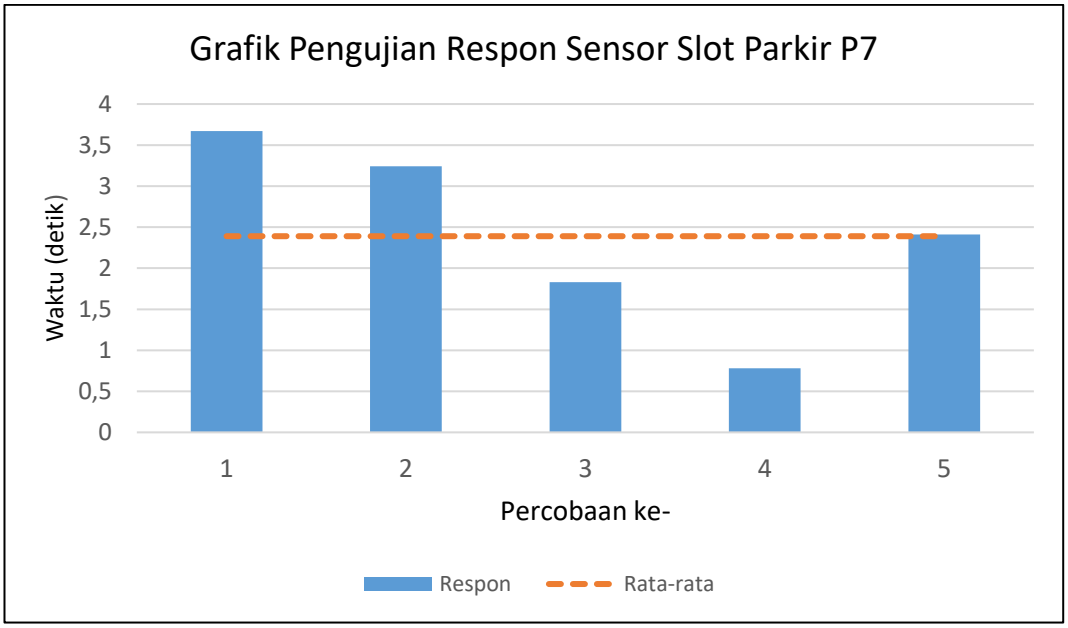


Gambar 4.7. Grafik Pengujian Respon Sensor Slot Parkir P6

Tabel 4.7 dan Gambar 4.7 merupakan hasil dan grafik pengujian respon sensor slot parkir P6 yang dilakukan dengan 5 kali percobaan. Pada percobaan 1 didapatkan respon 0,73 detik ketika *buzzer* hidup. Pada percobaan 2 didapatkan respon 3,38 detik ketika *buzzer* hidup. Pada percobaan 3 didapatkan respon 2,98 detik ketika *buzzer* hidup. Pada percobaan 4 didapatkan respon 3,06 detik ketika *buzzer* hidup. Pada percobaan 5 didapatkan respon 3,55 detik ketika *buzzer* hidup. Dapat disimpulkan bahwa respon sensor slot parkir P6 rata-rata seluruh percobaan yaitu sebesar 2,74 detik.

Tabel 4.8. Hasil pengujian respon sensor pada slot parkir P7

Percobaan	Waktu (detik)	Kondisi *Buzzer*
1	3,67	Hidup
2	3,24	Hidup
3	1,83	Hidup
4	0,78	Hidup
5	2,41	Hidup
Rata-rata = 2,39 detik		

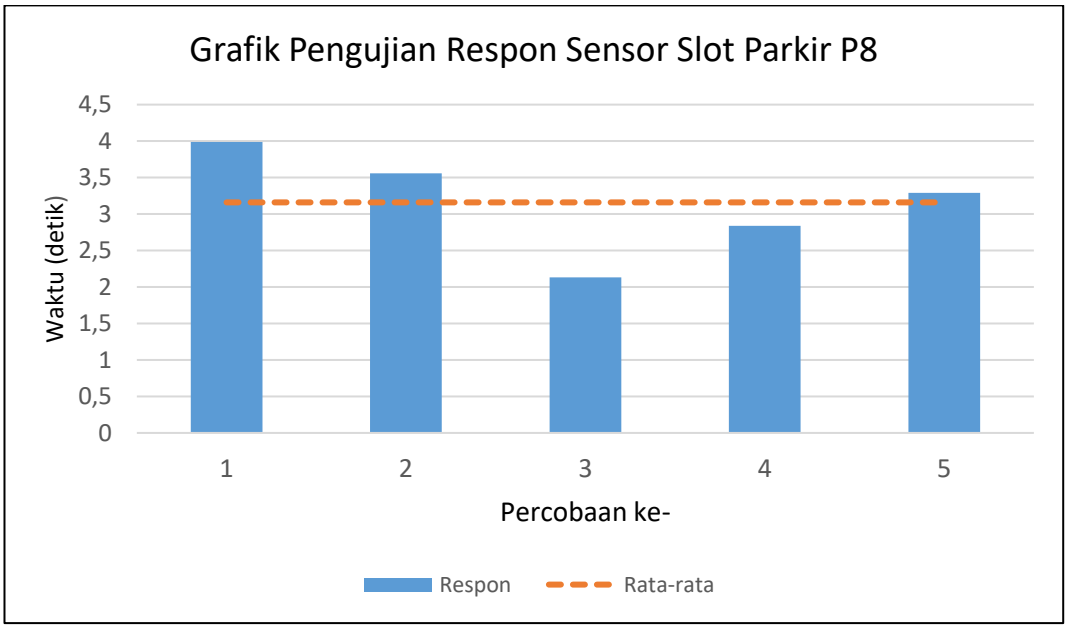


Gambar 4.8. Grafik Pengujian Respon Sensor Slot Parkir P7

Tabel 4.8 dan Gambar 4.8 merupakan hasil dan grafik pengujian respon sensor slot parkir P7 yang dilakukan dengan 5 kali percobaan. Pada percobaan 1 didapatkan respon 3,67 detik ketika *buzzer* hidup. Pada percobaan 2 didapatkan respon 3,24 detik ketika *buzzer* hidup. Pada percobaan 3 didapatkan respon 1,83 detik ketika *buzzer* hidup. Pada percobaan 4 didapatkan respon 0,78 detik ketika *buzzer* hidup. Pada percobaan 5 didapatkan respon 2,41 detik ketika *buzzer* hidup. Dapat disimpulkan bahwa respon sensor slot parkir P7 rata-rata seluruh percobaan yaitu sebesar 2,39 detik.

Tabel 4.9. Hasil pengujian respon sensor pada slot parkir P8

Percobaan	Waktu (detik)	Kondisi *Buzzer*
1	3,99	Hidup
2	3,56	Hidup
3	2,13	Hidup
4	2,84	Hidup
5	3,29	Hidup
Rata-rata = 3,16 detik		



Gambar 4.9. Grafik Pengujian Respon Sensor Slot Parkir P8

Tabel 4.9 dan Gambar 4.9 merupakan hasil dan grafik pengujian respon sensor slot parkir P8 yang dilakukan dengan 5 kali percobaan. Pada percobaan 1 didapatkan respon 3,99 detik ketika *buzzer* hidup. Pada percobaan 2 didapatkan respon 3,56 detik ketika *buzzer* hidup. Pada percobaan 3 didapatkan respon 2,13 detik ketika *buzzer* hidup. Pada percobaan 4 didapatkan respon 2,84 detik ketika *buzzer* hidup. Pada percobaan 5 didapatkan respon 3,29 detik ketika *buzzer* hidup. Dapat disimpulkan bahwa respon sensor slot parkir P8 rata-rata seluruh percobaan yaitu sebesar 3,16 detik.

4.2.1.3 Pengujian Akurasi Sensor Ultrasonik HC-SR04

Pengujian akurasi dilakukan dengan memantau keberadaan kendaraan secara langsung menggunakan mobil berukuran 1:100 dari mobil asli. Pengujian ini dilakukan untuk mengetahui dan mengukur sistem pendeteksian kendaraan dari sistem. Pengujian akurasi dilakukan dengan menggunakan *confusion matrix* dan akurasi sensor dapat dihitung menggunakan Persamaan 4.1. Pengujian ini dilakukan dengan meletakkan mobil pada setiap slot parkir yang dapat dilihat contohnya seperti Gambar 4.10.

$$\text{Akurasi sensor} = \frac{TP+TN}{TP+FP+FN+TN} \quad (4.1)$$

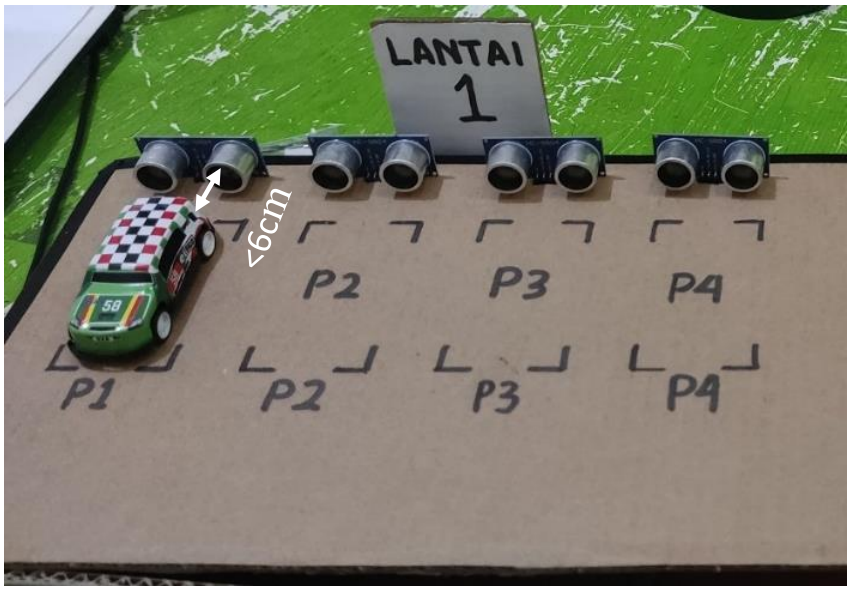
Keterangan :

TP : *True Positive*

FP : *False Positive*

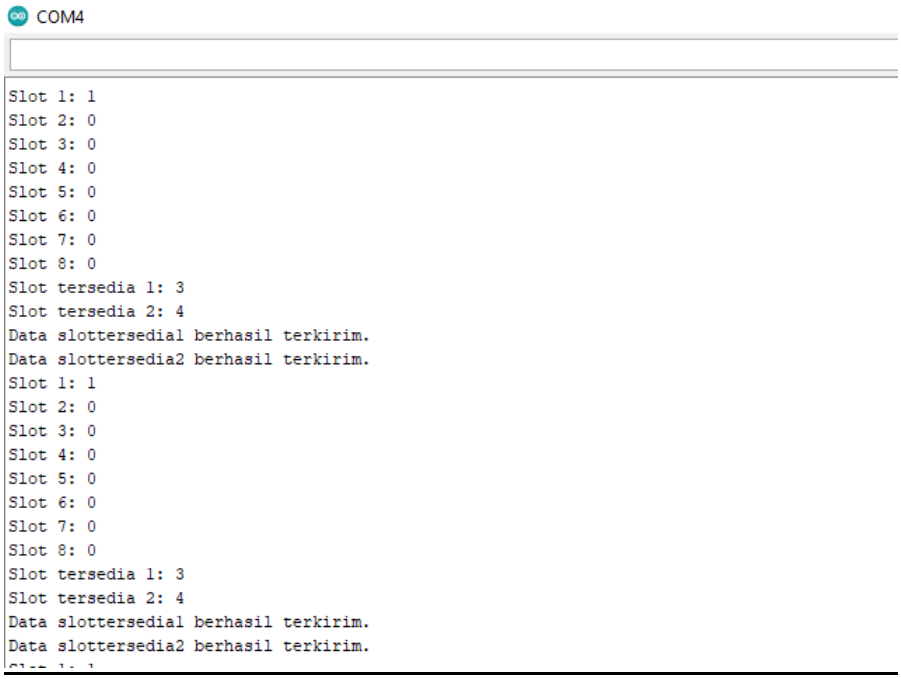
FN : *False Negative*

TN : *True Negative*



Gambar 4.10. Pengujian Akurasi Sensor Ultrasonik

Gambar 4.10 merupakan pengujian sensor ultrasonik untuk mengukur akurasi sensor tersebut dengan meletakkan mobil pada setiap slot parkir. Kemudian setelah meletakkan mobil pada slot parkir yang terdapat sensor ultrasonik dapat melihat keluaran pada serial monitor pada Gambar 4.11 berikut.



Gambar 4.11. *Output* Serial Monitor Pengujian Akurasi Sensor Ultrasonik

Pada Gambar 4.11 merupakan *output* pengujian sensor yang ditampilkan pada serial monitor. Kemudian mencatat hasil keluaran pada serial monitor dengan setiap slot parkir dicatat sebanyak 4 kali percobaan. Berikut Tabel hasil percobaan untuk menghitung nilai akurasi ditunjukan pada Tabel 4.10.

Tabel 4.10. Akurasi Pendeteksian Mobil

Pengujian ke-	**Percobaan Menempatkan Mobil pada Slot Parkir**	**Nilai Deteksi per Slot**							
		Slot 1	**Slot 2**	**Slot 3**	**Slot 4**	**Slot 5**	**Slot 6**	**Slot 7**	**Slot 8**
1	Slot 1	1	0	0	0	0	0	0	0
2	Slot 1	1	0	0	0	0	0	0	0
3	Slot 1	1	0	0	0	0	0	0	0
4	Slot 1	1	0	0	0	0	0	0	0
5	Slot 2	0	1	0	0	0	0	0	0
6	Slot 2	0	1	0	0	0	0	0	0
7	Slot 2	0	1	0	0	0	0	0	0
8	Slot 2	0	1	0	0	0	0	0	0

9	Slot 3	0	0	1	0	0	0	0	0
10	Slot 3	0	0	1	0	0	0	0	0
11	Slot 3	0	0	1	0	0	0	0	0
12	Slot 3	0	0	1	0	0	0	0	0
13	Slot 4	0	0	0	1	0	0	0	0
14	Slot 4	0	0	0	1	0	0	0	0
15	Slot 4	0	0	0	1	0	0	0	0
16	Slot 4	0	0	0	1	0	0	0	0
17	Slot 5	0	0	0	0	1	0	0	0
18	Slot 5	0	0	0	0	1	0	0	0
19	Slot 5	0	0	0	0	1	0	0	0
20	Slot 5	0	0	0	0	1	0	0	0
21	Slot 6	0	0	0	0	0	1	0	0
22	Slot 6	0	0	0	0	0	1	0	0
23	Slot 6	0	0	0	0	0	1	0	0
24	Slot 6	0	0	0	0	0	1	0	0
25	Slot 7	0	0	0	0	0	0	1	0
26	Slot 7	0	0	0	0	0	0	1	0
27	Slot 7	0	0	0	0	0	0	1	0
28	Slot 7	0	0	0	0	0	0	1	0
29	Slot 8	0	0	0	0	0	0	0	1
30	Slot 8	0	0	0	0	0	0	0	1
31	Slot 8	0	0	0	0	0	0	0	1
32	Slot 8	0	0	0	0	0	0	0	1

Keterangan :

1 : Terdeteksi Mobil

0 : Tidak Terdeteksi Mobil

Berdasarkan Tabel 4.10, menampilkan data hasil percobaan pada sistem deteksi kendaraan telah diambil. Percobaan dilakukan sebanyak 4 kali pada setiap slot parkir, dengan total 32 kali percobaan menempatkan mobil pada slot parkir. Hasil dari percobaan tersebut kemudian diolah menjadi tabel confusion matrix, yang

digunakan untuk menguji akurasi deteksi kendaraan pada sistem. Berikut adalah tabel confusion matrix dari hasil percobaan deteksi kendaraan yang ditunjukkan pada Tabel 4.11.

Tabel 4.11. *Confusion Matrix* Hasil Akurasi Sensor Ultrasonik

Percobaan ke …	**TP**	**FP**	**FN**	**TN**
Percobaan ke-1	1	0	0	7
Percobaan ke-2	1	0	0	7
Percobaan ke-3	1	0	0	7
Percobaan ke-4	1	0	0	7
Percobaan ke-5	1	0	0	7
Percobaan ke-6	1	0	0	7
Percobaan ke-7	1	0	0	7
Percobaan ke-8	1	0	0	7
Percobaan ke-9	1	0	0	7
Percobaan ke-10	1	0	0	7
Percobaan ke-11	1	0	0	7
Percobaan ke-12	1	0	0	7
Percobaan ke-13	1	0	0	7
Percobaan ke-14	1	0	0	7
Percobaan ke-15	1	0	0	7
Percobaan ke-16	1	0	0	7
Percobaan ke-17	1	0	0	7
Percobaan ke-18	1	0	0	7
Percobaan ke-19	1	0	0	7
Percobaan ke-20	1	0	0	7
Percobaan ke-21	1	0	0	7
Percobaan ke-22	1	0	0	7
Percobaan ke-23	1	0	0	7
Percobaan ke-24	1	0	0	7
Percobaan ke-25	1	0	0	7

Percobaan ke-26	1	0	0	7
Percobaan ke-27	1	0	0	7
Percobaan ke-28	1	0	0	7
Percobaan ke-29	1	0	0	7
Percobaan ke-30	1	0	0	7
Percobaan ke-31	1	0	0	7
Percobaan ke-32	1	0	0	7
Total	32	0	0	224

Keterangan :

TP : *True Positive*

FP : *False Positive*

FN : *False Negative*

TN : *True Negative*

Berdasarkan Tabel 4.11, yang merupakan confusion matrix hasil pengujian akurasi sensor ultrasonik dari 32 kali percobaan, setiap percobaan melibatkan 4 kondisi yang diperoleh dari 8 slot parkir. Hasil percobaan menunjukkan kondisi *True Positive* (TP) ketika ada mobil terdeteksi dan sensor berhasil mendeteksi keberadaan mobil pada slot parkir tersebut. Kondisi *False Positive* (FP) terjadi ketika sensor mendeteksi adanya mobil, tetapi sebenarnya tidak ada mobil pada slot parkir. Kondisi *False Negative* (FN) terjadi ketika sensor tidak mendeteksi mobil padahal ada mobil yang diparkir. Sementara itu, kondisi *True Negative* (TN) terjadi ketika tidak ada mobil yang terdeteksi dan sensor juga tidak mendeteksi adanya mobil pada slot tersebut. Akurasi sensor dihitung menggunakan rumus yang ditunjukkan pada persamaan 4.1.

$$\text{Akurasi sensor} = \frac{TP+TN}{TP+FP+FN+TN}$$

$$\text{Akurasi sensor} = \frac{32+224}{32+0+0+224}$$

Akurasi sensor = 1

Akurasi sensor = 1*100%

Akurasi sensor = 100%

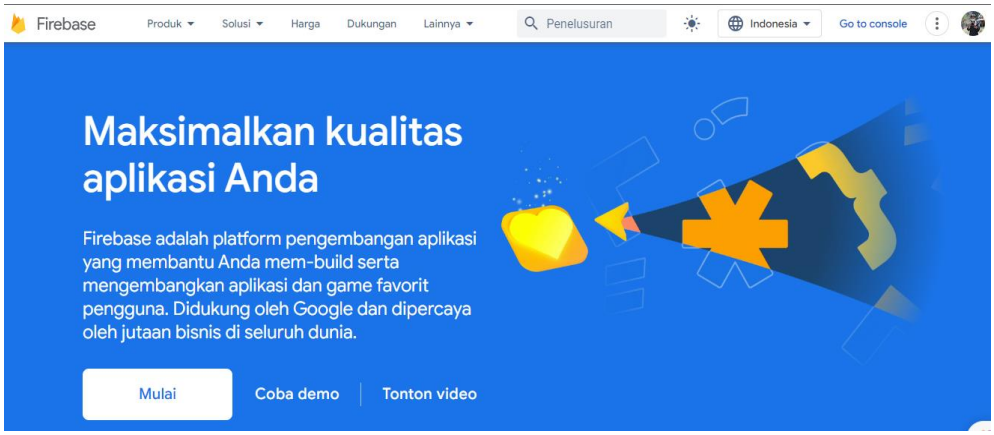
4.2.2 Pengujian *Software*

Pengujian *software* terdiri dari 2 bagian yaitu *database* dan aplikasi. *Database* yang digunakan pada penelitian ini adalah Firebase dan Android Studio sebagai *framework* untuk pembuatan aplikasi.

4.2.2.1 Pengujian *Database*

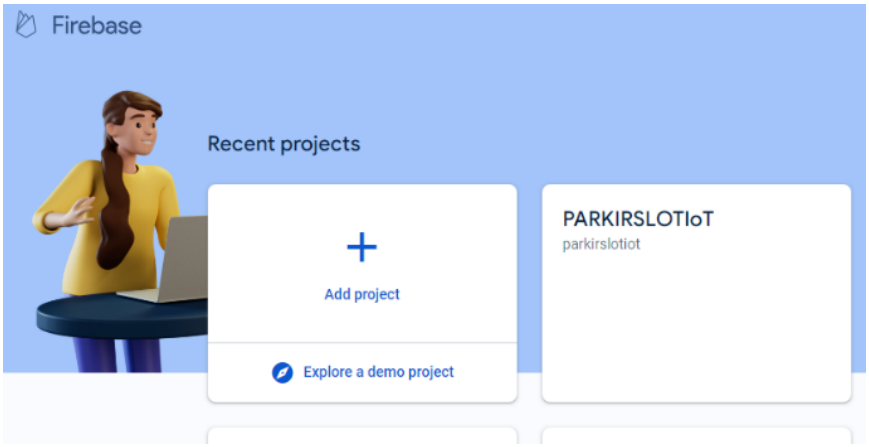
Firebase yang digunakan memanfaatkan fungsi dari *real-time database* sesuai dengan kebutuhan *database* yang digunakan pada penelitian ini. Pengujian *database* bertujuan untuk mengetahui apakah Firebase dapat bekerja dengan baik menangkap data dari alat penelitian. Proses tahapan yang dilakukan untuk membuat *real-time database* sebagai berikut:

1. Membuka situs pada browser dengan link url firebase.google.com dan login menggunakan akun google. Tampilan setelah login dapat dilihat pada Gambar 4.12.



Gambar 4.12. Tampilan Awal Firebase

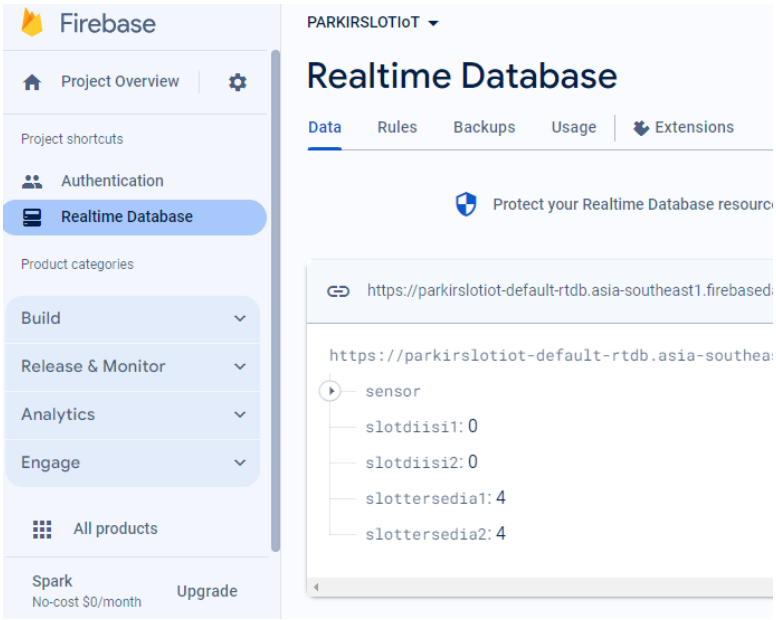
2. Pada tampilan awal Firebase terdapat tombol Mulai untuk masuk ke halaman Firebase konsol. Halaman Firebase konsol mempunyai fungsi untuk membuat *project database* baru maupun melihat *project database* yang telah dibuat sebelumnya. Tampilan pada Firebase konsol dapat dilihat pada Gambar 4.13.



Gambar 4.13. Tampilan Firebase Konsol

Pada Gambar 4.13 merupakan halaman tampilan Firebase konsol yang memiliki tombol *add project*, berfungsi untuk membuat *project database* baru. Langkah yang akan dijalani saat menambahkan *project* baru terdiri dari menentukan nama *project database*, memilih *tools* apa saja yang akan digunakan pada *database* serta mengkonfigurasi google analitis.

3. Proses selanjutnya adalah pembuatan *real-time database* pada menu *Build* di Firebase konsol. Kemudian setelah berhasil membuat *real-time database*, langkah selanjutnya yaitu membuat *path* dalam *database*. Berikut adalah tampilan *real-time database* yang ditunjukan pada Gambar 4.14.



Gambar 4.14. Tampilan *Real-time Database*

4. Membuka *settings* pada *Project Overview* untuk mengetahui nama, id, *nomor project* serta API *keys* yang akan digunakan untuk menghubungkan *database* dengan alat penelitian, sehingga alat dapat mengirimkan data ke Firebase secara *real-time*.
5. Setelah alat dan Firebase telah terhubung dan bisa digunakan, selanjutnya menghubungkan *database* dengan aplikasi android yang akan digunakan.

Setelah proses pembuatan *real-time database* dan berhasil menghubungkan Firebase dengan alat maupun aplikasi, kemudian dilakukan pengujian apakah nilai pada alat sama atau terkonfigurasi dengan *database* yang telah dibuat. Pengujian dilakukan dengan mengamati nilai pada slot parkir tersedia dan posisi slot parkir. Hasil pengujian dapat dilihat pada Tabel 4.12 dan Tabel 4.13.

Tabel 4.12 Hasil Pengujian *Database* pada Lantai 1

Percobaan Menempatkan Mobil pada Slot Parkir	**Nilai pada alat**					**Nilai pada database**					**Kesimpulan**
	Slot 1 (P1)	**Slot 2 (P2)**	**Slot 3 (P3)**	**Slot 4 (P4)**	**Slot tersedia**	**Slot 1 (P1)**	**Slot 2 (P2)**	**Slot 3 (P3)**	**Slot 4 (P4)**	**Slot tersedia**	
1 Mobil											
P1	1	0	0	0	3	1	0	0	0	3	Sesuai
P2	0	1	0	0	3	0	1	0	0	3	Sesuai
P3	0	0	1	0	3	0	0	1	0	3	Sesuai
P4	0	0	0	1	3	0	0	0	1	3	Sesuai
2 Mobil											
P1 & P2	1	1	0	0	2	1	1	0	0	2	Sesuai
P1 & P3	1	0	1	0	2	1	0	1	0	2	Sesuai
P1 & P4	1	0	0	1	2	1	0	0	1	2	Sesuai
P2 & P3	0	1	1	0	2	0	1	1	0	2	Sesuai
P2 & P4	0	1	0	1	2	0	1	0	1	2	Sesuai
P3 &P4	0	0	1	1	2	0	0	1	1	2	Sesuai
3 Mobil											
P1, P2 & P3	1	1	1	0	1	1	1	1	0	1	Sesuai
P1, P2 & P4	1	1	0	1	1	1	1	0	1	1	Sesuai
P1, P3 & P4	1	0	1	1	1	1	0	1	1	1	Sesuai
P2, P3 & P4	0	1	1	1	1	0	1	1	1	1	Sesuai
4 Mobil											
P1, P2, P3 & P4	1	1	1	1	0	1	1	1	1	0	Sesuai

Tabel 4.13. Hasil Pengujian *Database* pada Lantai 2

Percobaan Menempatkan Mobil pada Slot Parkir	Nilai pada alat					Nilai pada database					Kesimpulan
	Slot 1 (P5)	Slot 2 (P6)	Slot 3 (P7)	Slot 4 (P8)	Slot tersedia	Slot 1 (P5)	Slot 2 (P6)	Slot 3 (P7)	Slot 4 (P8)	Slot tersedia	
1 Mobil											
P5	1	0	0	0	3	1	0	0	0	3	Sesuai
P6	0	1	0	0	3	0	1	0	0	3	Sesuai
P7	0	0	1	0	3	0	0	1	0	3	Sesuai
P8	0	0	0	1	3	0	0	0	1	3	Sesuai
2 Mobil											
P5 & P6	1	1	0	0	2	1	1	0	0	2	Sesuai
P5 & P7	1	0	1	0	2	1	0	1	0	2	Sesuai
P5 & P8	1	0	0	1	2	1	0	0	1	2	Sesuai
P6 & P7	0	1	1	0	2	0	1	1	0	2	Sesuai
P6 & P8	0	1	0	1	2	0	1	0	1	2	Sesuai
P7 &P8	0	0	1	1	2	0	0	1	1	2	Sesuai
3 Mobil											
P5, P6 & P7	1	1	1	0	1	1	1	1	0	1	Sesuai
P5, P6 & P8	1	1	0	1	1	1	1	0	1	1	Sesuai
P5, P7 & P8	1	0	1	1	1	1	0	1	1	1	Sesuai
P6, P7 & P8	0	1	1	1	1	0	1	1	1	1	Sesuai
4 Mobil											
P5, P6, P7 & P8	1	1	1	1	0	1	1	1	1	0	Sesuai

Berdasarkan Tabel 4.12 dan 4.13 merupakan hasil pengujian *database* pada lantai 1 dan lantai 2, didapatkan bahwa nilai pada *database* dan alat sama. Hal ini menandakan bahwa *database* yang telah dibuat dan aplikasi yang telah dirancang telah terkonfigurasi dengan baik.

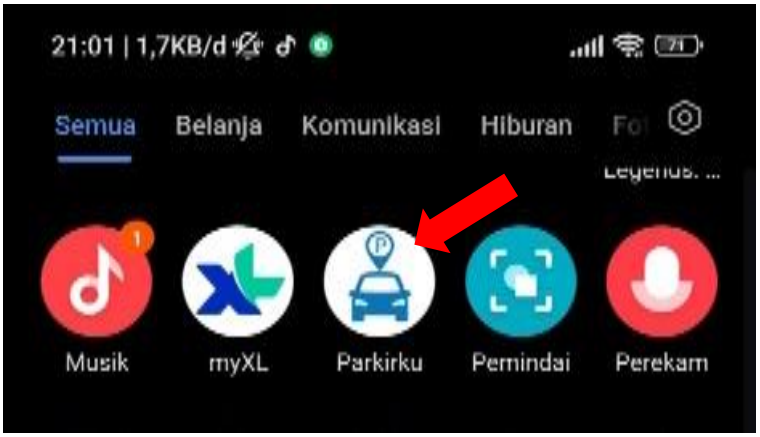
4.2.2.2 Pengujian Aplikasi

Aplikasi dirancang menggunakan Android Studio sebagai *framework* dengan berbagai langkah yang dilalui. Pengujian aplikasi berfungsi untuk mengetahui apakah aplikasi dapat menampilkan data yang dikirim oleh alat ke *database* ditampilkan pada aplikasi secara *real-time.* Pada proses perancangan aplikasi ada beberapa proses dalam pembuatan aplikasi sebagai berikut:

1. Membuka Android Studio dan membuat *project* baru serta memilih jenis *project* lalu mengkonfigurasi *project.*

2. Mengkonfigurasi Firebase dan Android Studio sehingga terhubung dan dapat digunakan.
3. Pada proses running aplikasi, menggunakan HP Poco X3.

Kemudian setelah proses instalasi dan pembuatan *source code* yang diinginkan yaitu menampilkan logo pada aplikasi, membuat *splash screen*, halaman pilih lokasi, halaman pilih lantai, pemantauan slot parkir dan data jumlah slot terisi dan slot tersedia. Implementasi *interface* aplikasi yang dirancang pada *smartphone* terlihat pada Gambar 4.15.



Gambar 4.15. Tampilan Logo Aplikasi Parkirku

Berdasarkan Gambar 4.15 merupakan logo dari aplikasi yang dirancang untuk memonitoring slot parkir yang tersedia dan tersisa pada gedung parkir atau mall secara *real-time.*

Gambar 4.16. *Splash Screen*

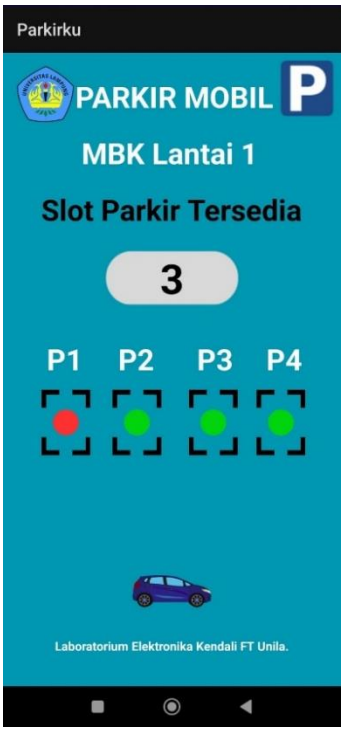
Splash screen merupakan tampilan awal dari aplikasi yang dirancang sesaat setelah aplikasi dibuka dapat dilihat pada Gambar 4.16.

Gambar 4.17. Tampilan Halaman Pilih Lokasi

Pada Gambar 4.17 merupakan halaman pilih lokasi yaitu halaman yang menampilkan lokasi dari gedung parkir yang dapat dipilih dengan menekan tombol pada nama lokasi. Pada halaman ini juga terdapat jumlah kapasitas slot parkir pada suatu lokasi dan jumlah slot parkir yang tersedia.

Gambar 4.18. Tampilan Halaman Pilih Lantai

Pada Gambar 4.18 merupakan halaman pilih lantai yaitu halaman yang menampilkan lantai dari gedung parkir yang dapat dipilih dengan menekan tombol pada nomor lantai. Pada halaman ini juga terdapat jumlah kapasitas slot parkir pada suatu lantai dan jumlah slot parkir yang tersedia.



Gambar 4.19. Tampilan Halaman Posisi dan Data Slot Tersedia

Berdasarkan Gambar 4.19 merupakan halaman yang menampilkan posisi slot parkir pada suatu lantai dan menampilkan jumlah slot parkir yang tersedia pada lantai tersebut. Posisi slot parkir akan memiliki indikator merah apabila terdapat suatu kendaraan terparkir dapat dilihat seperti pada Gambar 4.19 pada P1. Sedangkan posisi slot parkir akan memiliki indikator hijau apabila tidak terdapat suatu kendaraan yang terparkir dapat dilihat seperti pada Gambar 4.19 pada P2, P2 dan P3.

Setelah membuat tampilan dan telah dikonfigurasikan dengan Firebase telah berhasil, kemudian dilakukan pengujian aplikasi parkirku dengan cara mengamati hasil *input* dan *output* dari aplikasi yang telah dibuat tanpa mengetahui struktur kode dari aplikasi tersebut, sehingga dengan pengujian ini dapat mengetahui apakah aplikasi dapat berfungsi dengan baik. Hasil pengujian pada aplikasi parkirku dapat dilihat pada Tabel 4.14.

Tabel 4.14. Hasil Pengujian Integrasi Aplikasi

No	Fitur	Hasil yang diharapkan	Kesimpulan
1	Splash Screen	Menampilkan logo dari aplikasi Parkirku	Berhasil
2	Halaman memilih lokasi parkir	Menampilkan lokasi parkir yang akan dipilih beserta informasi kapasitas dan jumlah slot parkir yang tersedia	Berhasil
3	Tombol lokasi parkir	Setelah disentuh akan berpindah ke halaman memilih lantai pada lokasi tersebut	Berhasil
4	Halaman memilih lantai	Menampilkan lantai pada lokasi parkir yang akan dipilih beserta informasi kapasitas dan jumlah slot parkir yang tersedia	Berhasil
5	Tombol lokasi parkir	Setelah disentuh akan berpindah ke halaman informasi ketersediaan pada lantai tersebut	Berhasil
6	Data ketersediaan slot parkir	Jumlah slot parkir yang tersedia sesuai dengan nilai pada Firebase	Berhasil
7	Data letak slot parkir	Menampilkan indikator berwarna hijau saat slot parkir tidak ada kendaraan terparkir dan akan menampilkan indikator merah saat ada kendaraan terparkir	Berhasil

Berdasarkan Tabel 4.14 Merupakan hasil pengujian aplikasi dapat dilihat bahwa aplikasi dapat melakukan fitur-fitur yang digunakan sebagai sistem informasi ketersediaan slot parkir. Dapat diketahui dari Tabel 4.14 Untuk *hardware* dan *software* terintegrasi dengan baik dan sesuai dengan yang diharapkan.

4.3 Pengujian Latensi Sistem

Pengujian latensi sistem bertujuan untuk mengukur dan menganalisis waktu yang dibutuhkan oleh data untuk berpindah dari perangkat IoT ke aplikasi. Pada pengujian ini mengamati dan mencatat hasil pengukuran latensi pada aplikasi dengan masing-masing provider internet sebanyak 10 kali percobaan yang selanjutnya dicari rata-rata latensi pada masing-masing provider internet yang digunakan. Hal ini bertujuan untuk mengetahui provider internet yang memberikan latensi paling rendah dalam mengoperasikan aplikasi yang telah dibuat. Hasil pengujian dapat dilihat pada Tabel 4.15, Tabel 4.16 dan Tabel 4.17.

Tabel 4.15. Hasil Pengujian Latensi Menggunakan Provider Internet XL

Percobaan	**Latensi Deteksi Mobil pada slot (detik)**							
	P1	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**
1	1,38	2,4	3,64	0,9	1,52	1,22	2,11	1,23
2	2,23	1,87	1,94	5,12	0,82	4,28	3,04	3,05
3	1,83	3,4	0,68	3,53	4,24	1,53	4,93	3,33
4	2,12	2,4	4,03	3,03	0,8	0,72	1,86	3,29
5	1,14	0,82	1,26	1,24	2,6	4,35	1,6	4,53
6	1,14	1,48	0,88	3,35	2,36	0,85	3,67	3,15
7	1,94	1,86	3,67	3,08	1,45	5,03	3,6	4,47
8	1,95	1,93	1,67	1,67	3,88	3,64	3,56	6,97
9	1,32	3,87	2,15	1,46	2,56	3,51	2,4	3,97
10	1,04	0,89	1,55	1,95	1,12	4,46	3,7	0,83
Rata-rata	1,609	2,092	2,147	2,533	2,135	2,959	3,047	3,482
Rata-rata keseluruhan					2,5 detik			

Tabel 4.15 merupakan data hasil pengujian latensi untuk pengiriman data pada alat kepada aplikasi menggunakan provider internet XL disimpulkan bahwa rata-rata latensi yang dihasilkan yaitu 2,5 detik.

Tabel 4.16 merupakan data hasil pengujian latensi untuk pengiriman data pada alat kepada aplikasi menggunakan provider internet Telkomsel disimpulkan bahwa rata-rata latensi yang dihasilkan yaitu 2,7 detik.

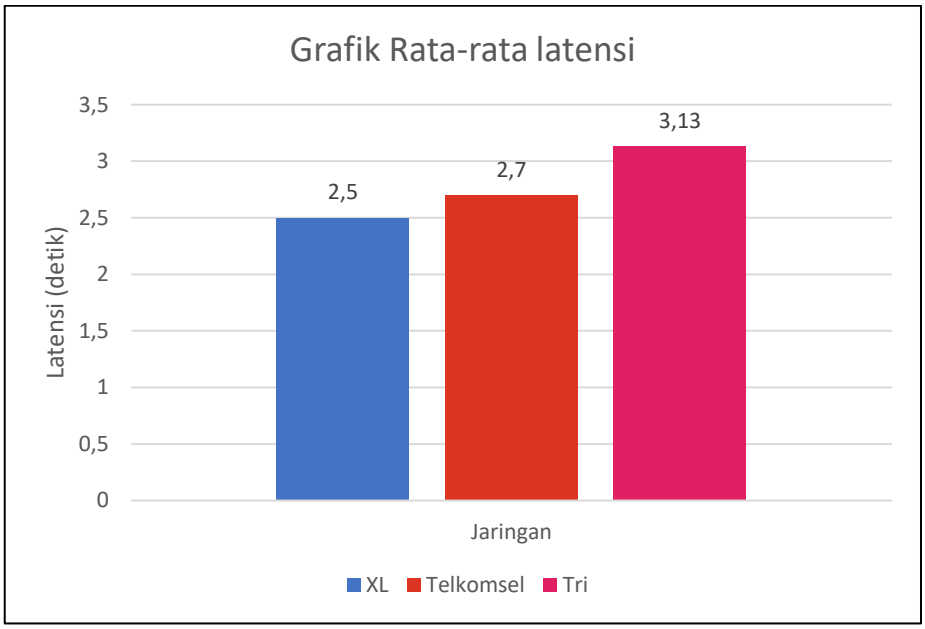
Tabel 4.16. Hasil Pengujian Latensi Menggunakan Provider Internet Telkomsel

Percobaan	Latensi Deteksi Mobil pada slot (detik)							
	P1	P2	P3	P4	P5	P6	P7	P8
1	0,81	2,3	4,28	3,86	1,83	4,08	4,66	3,26
2	2,68	1,56	2,35	1,86	3,45	1,3	3,28	1,76
3	2,24	2,43	2,85	2,85	3,29	5,14	1,9	2,93
4	0,96	2,29	1,76	1,45	2,43	3,61	2,12	3,8
5	2,29	1,17	1,95	1,13	4,64	1,36	4,56	3,85
6	3,72	2,06	2,71	1,55	3,12	4,26	3,28	3,74
7	2,08	2,09	4,88	5,2	2,55	4,15	2,83	2,43
8	1,38	4,04	1,01	4,93	3,02	0,57	2,79	1,81
9	0,66	1,13	2,33	1,87	1,7	1,27	4,39	3,56
10	1,52	3,97	2,48	2,29	3,26	4,83	1,28	3,96
Rata-rata	1,834	2,304	2,66	2,699	2,929	3,057	3,109	3,11
Rata-rata keseluruhan					2,7 detik			

Tabel 4.17. Hasil Pengujian Latensi Menggunakan Provider Internet Tri

Percobaan	Latensi Deteksi Mobil pada slot (detik)							
	P1	P2	P3	P4	P5	P6	P7	P8
1	2,51	4,03	1,65	1,67	6,37	2,84	2,35	2,36
2	1,15	2,56	4,07	1,99	1,66	4,05	0,98	2,96
3	1,44	2,83	1,24	1,88	1,9	3,31	4,77	2,69
4	1,03	4,85	1,92	8,2	1,33	2,17	1,47	4,46
5	4,18	8,29	3,08	3,79	1,06	1,71	4,21	5,51
6	2,79	1,93	3,61	2,9	2,46	5,29	10,91	5,39
7	1,36	3,65	2,4	2,8	1,02	4,12	2,64	3,22
8	3,36	3	4,93	3,96	1,67	9,53	1,61	3,63
9	1,56	2,53	1,48	4,14	1,94	1,9	1,89	4,73
10	2,74	1,07	1,51	1,05	2,07	1,31	6,64	5,25
Rata-rata	2,212	3,474	2,589	3,238	2,148	3,623	3,747	4,02
Rata-rata keseluruhan					3,13 detik			

Tabel 4.17 merupakan data hasil pengujian latensi untuk pengiriman data pada alat kepada aplikasi menggunakan provider internet Tri disimpulkan bahwa rata-rata latensi yang dihasilkan yaitu 3,13 detik.



Gambar 4.20 Grafik rata-rata latensi berdasarkan jaringan yang digunakan

Berdasarkan Gambar 4.20, ditampilkan grafik rata-rata latensi berdasarkan jaringan yang digunakan. Latensi yang diukur dan dicatat adalah waktu yang dibutuhkan perangkat *internet of things* (IoT) untuk mengirimkan data ke aplikasi. Hasil pengujian menunjukkan bahwa latensi pada provider XL sebesar 2,5 detik, pada provider Telkomsel sebesar 2,7 detik, dan pada provider Tri sebesar 3,13 detik. Dari hasil ini, dapat disimpulkan bahwa rata-rata latensi pada setiap jaringan masih tergolong rendah karena berada di bawah prediksi awal 10 detik. Berdasarkan grafik, latensi rata-rata yang dihasilkan oleh jaringan Tri lebih tinggi dibandingkan jaringan XL dan Telkomsel. Hal ini menunjukkan bahwa penggunaan jaringan XL dan Telkomsel lebih baik dibandingkan dengan jaringan Tri.

4.4. Pengujian Latensi dengan Jarak

Pengujian ini dilakukan dengan mengukur latensi sistem pada jarak 3 km, 6 km, dan 10 km menggunakan tiga provider internet yang berbeda yaitu: XL, Telkomsel, dan Tri. Prosedur pengambilan data latensi sistem pada jarak 3 km, 6 km, dan 10 km menggunakan provider internet XL, Telkomsel, dan Tri sebagai berikut:

1. Menyiapkan dan menghidupkan prototipe sistem parkir berbasis IoT.
2. Melakukan pemantauan pada aplikasi parkirku berdasarkan jarak 3 Km, 6 Km, dan 10 Km.
3. Meletakkan mobil pada prototipe dengan kondisi seperti pada Tabel 4.18.

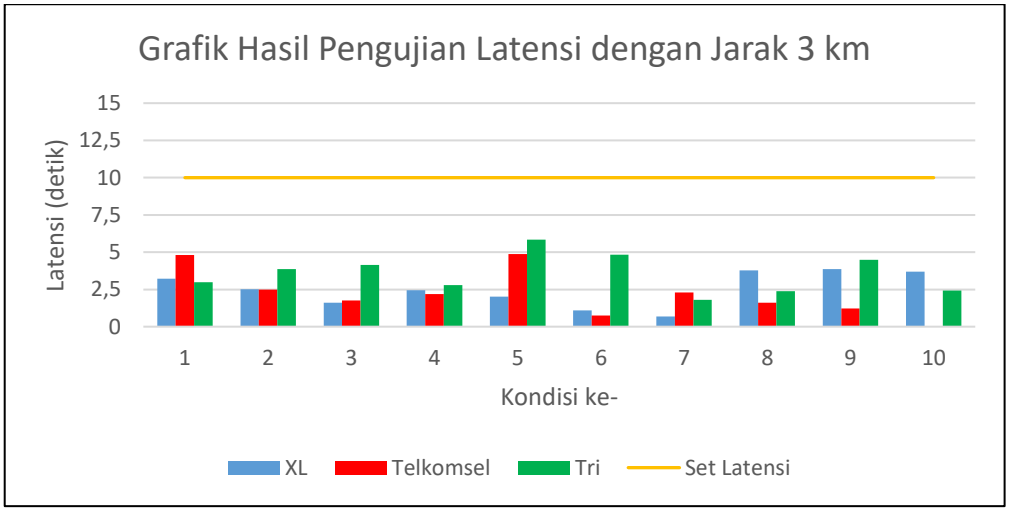
Tabel 4.18. Kondisi pada saat pengujian Latensi dengan Jarak

Kondisi	**Meletakkan mobil pada slot parkir**
Kondisi 1	P1 dan P4
Kondisi 2	P1, P3, dan P4
Kondisi 3	P1, P5, dan P8
Kondisi 4	P1, P2, P4 dan P6
Kondisi 5	P2, P3, P5 dan P7
Kondisi 6	P1, P2, P3, P5 dan P7
Kondisi 7	P2, P3, P4, P6 dan P8
Kondisi 8	P2, P3, P4, P5, P6 dan P7
Kondisi 9	P1, P2, P3, P4, P5, P6 dan P8
Kondisi 10	P1, P2, P3, P4, P5, P6, P7 dan P8

4. Lalu mengukur waktu pengiriman data dari perangkat IoT ke aplikasi android menggunakan *stopwatch*, kemudian mencatat hasilnya pada Tabel 4.19, Tabel 4.20, dan Tabel 4.21.

Tabel 4.19. Hasil Pengujian Latensi dengan Jarak 3 km

No	Waktu		Kondisi	Latensi (detik)		
	Tanggal	Jam		XL	Telkomsel	Tri
1	13/07/2024	15:14:28	Kondisi 1	3,23	4,82	2,99
2	13/07/2024	15:16:29	Kondisi 2	2,52	2,5	3,86
3	13/07/2024	15:17:28	Kondisi 3	1,61	1,75	4,15
4	13/07/2024	15:18:28	Kondisi 4	2,44	2,18	2,8
5	13/07/2024	15:19:28	Kondisi 5	2,01	4,88	5,85
6	13/07/2024	15:20:29	Kondisi 6	1,1	0,74	4,83
7	13/07/2024	15:22:28	Kondisi 7	0,69	2,3	1,81
8	13/07/2024	15:24:29	Kondisi 8	3,78	1,61	2,38
9	13/07/2024	15:25:28	Kondisi 9	3,86	1,23	4,49
10	13/07/2024	15:26:29	Kondisi 10	3,69	0.88	2,42
Rata-Rata				2,49	2,44	3,55

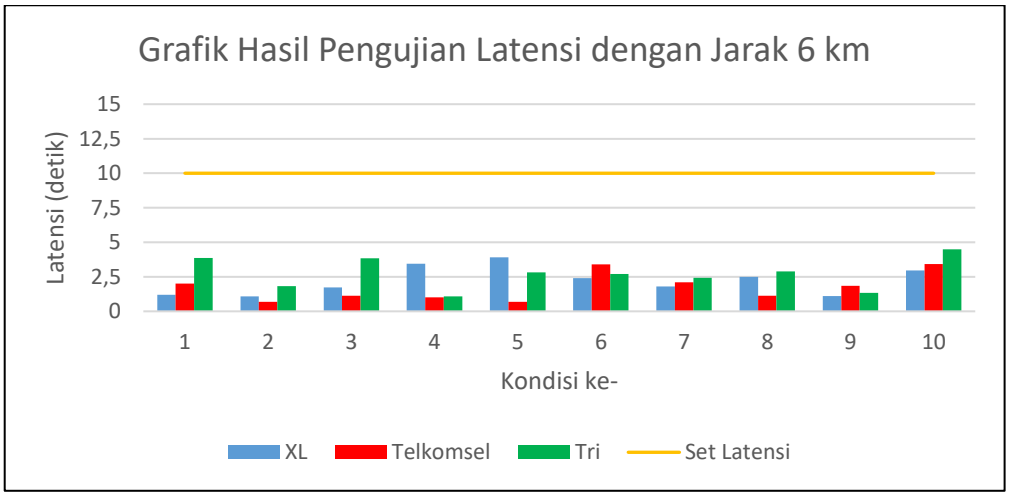


Gambar 4.21. Grafik Hasil Pengujian Latensi dengan Jarak 3 km

Tabel 4.19 dan Gambar 4.21 menampilkan data dan grafik hasil pengujian latensi untuk pengiriman data dari perangkat IoT ke aplikasi pada jarak 3 km. Grafik berwarna biru menunjukkan latensi untuk provider XL, grafik berwarna merah menunjukkan latensi untuk provider Telkomsel, dan grafik berwarna hijau menunjukkan latensi untuk provider Tri. Selain itu, grafik berwarna kuning menandakan batas latensi ideal yang diprediksi awal 10 detik, di mana latensi di bawah batas tersebut dianggap baik untuk sistem. Berdasarkan hasil pengujian, dapat disimpulkan bahwa rata-rata latensi yang dihasilkan oleh provider XL pada jarak 3 km adalah 2,49 detik. Rata-rata latensi untuk provider Telkomsel pada jarak yang sama adalah 2,44 detik, sedangkan rata-rata latensi untuk provider Tri adalah 3,55 detik.

Tabel 4.20. Hasil Pengujian Latensi dengan Jarak 6 km

No	Waktu		Kondisi	Latensi (detik)		
	Tanggal	Jam		XL	Telkomsel	Tri
1	13/07/2024	14:28:28	Kondisi 1	1,19	2,01	3,86
2	13/07/2024	14:29:28	Kondisi 2	1,08	0,68	1,82
3	13/07/2024	14:31:29	Kondisi 3	1,73	1,12	3,85
4	13/07/2024	14:32:28	Kondisi 4	3,44	1,01	1,09
5	13/07/2024	14:34:28	Kondisi 5	3,91	0,68	2,83
6	13/07/2024	14:35:29	Kondisi 6	2,41	3,4	2,71
7	13/07/2024	14:36:28	Kondisi 7	1,79	2,11	2,43
8	13/07/2024	14:37:28	Kondisi 8	2,5	1,12	2,9
9	13/07/2024	14:39:29	Kondisi 9	1,11	1,84	1,34
10	13/07/2024	14:40:28	Kondisi 10	2,95	3,43	4,5
Rata-Rata				2,21	1,74	2,73

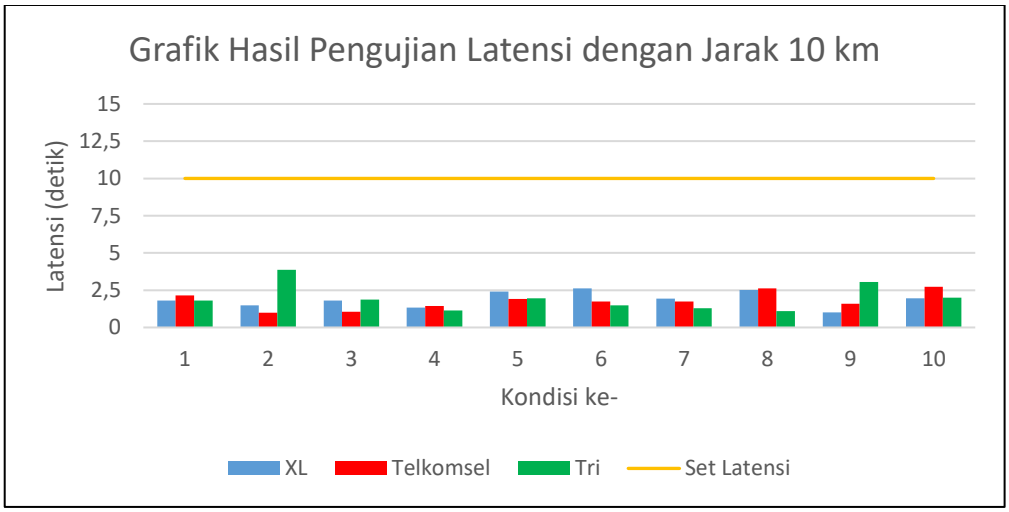


Gambar 4.22. Grafik Hasil Pengujian Latensi dengan Jarak 6 km

Tabel 4.20 dan Gambar 4.22 menampilkan data dan grafik hasil pengujian latensi untuk pengiriman data dari perangkat IoT ke aplikasi pada jarak 6 km. Grafik berwarna biru menunjukkan latensi untuk provider XL, grafik berwarna merah menunjukkan latensi untuk provider Telkomsel, dan grafik berwarna hijau menunjukkan latensi untuk provider Tri. Grafik berwarna kuning menunjukkan batas latensi yang ideal yang diprediksi awal 10 detik, di mana latensi di bawah batas tersebut dianggap baik untuk sistem.Berdasarkan hasil pengujian, dapat disimpulkan bahwa rata-rata latensi yang dihasilkan oleh jaringan XL pada jarak 6 km adalah 2,21 detik. Rata-rata latensi yang dihasilkan oleh jaringan Telkomsel pada jarak yang sama adalah 1,74 detik, sedangkan rata-rata latensi yang dihasilkan oleh jaringan Tri adalah 2,73 detik.

Tabel 4.21. Hasil Pengujian Latensi dengan Jarak 10 km

No	Waktu		Kondisi	Latensi (detik)		
	Tanggal	Jam		XL	Telkomsel	Tri
1	13/07/2024	13:26:28	Kondisi 1	1,81	2,15	1,81
2	13/07/2024	13:28:29	Kondisi 2	1,48	0,98	3,87
3	13/07/2024	13:31:28	Kondisi 3	1,81	1,06	1,86
4	13/07/2024	13:32:28	Kondisi 4	1,34	1,43	1,15
5	13/07/2024	13:34:28	Kondisi 5	2,41	1,92	1,96
6	13/07/2024	13:36:28	Kondisi 6	2,63	1,74	1,48
7	13/07/2024	13:37:28	Kondisi 7	1,93	1,74	1,3
8	13/07/2024	13:38:28	Kondisi 8	2,52	2,62	1,1
9	13/07/2024	13:39:28	Kondisi 9	1,01	1,6	3,06
10	13/07/2024	13:41:28	Kondisi 10	1,96	2,74	2
Rata-Rata				1,89	1,79	1,95



Gambar 4.23. Grafik Hasil Pengujian Latensi dengan Jarak 10 km

Tabel 4.21 dan Gambar 4.23 menampilkan data dan grafik hasil pengujian latensi untuk pengiriman data dari perangkat IoT ke aplikasi pada jarak 10 km. Grafik berwarna biru menunjukkan latensi untuk provider XL, grafik berwarna merah menunjukkan latensi untuk provider Telkomsel, dan grafik berwarna hijau menunjukkan latensi untuk provider Tri. Grafik berwarna kuning menunjukkan batas latensi ideal yang diprediksi 10 detik, di mana latensi di bawah batas tersebut dianggap baik untuk sistem. Berdasarkan hasil pengujian, dapat disimpulkan bahwa rata-rata latensi yang dihasilkan oleh jaringan XL pada jarak 10 km adalah 1,89 detik. Rata-rata latensi yang dihasilkan oleh jaringan Telkomsel pada jarak yang sama adalah 1,79 detik, sedangkan rata-rata latensi yang dihasilkan oleh jaringan Tri adalah 1,95 detik.

Tabel 4.22. Hasil rata-rata latensi informasi data di bawah radius 10 km

Jarak (km)	XL (detik)	Telkomsel (detik)	Tri (detik)
3	2,49	2,44	3,55
6	2,21	1,74	2,73
10	1,89	1,79	1,95

Berdasarkan Tabel 4.22 dapat disimpulkan bahwa latensi rata-rata dari tiga provider dalam jarak di bawah radius 10 km berkisar antara 1,74 detik sampai dengan 3,55 detik. Latensi tersebut masih jauh dibawah prediksi awal 10 detik.

V. KESIMPULAN DAN SARAN

5.1 Kesimpulan

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

1. Sistem pemantau ketersediaan parkir *real-time* berbasis *internet of things* berhasil dibangun dengan dapat diaplikasikan pada *smartphone* dan bekerja sesuai dengan skenario rancangan.
2. Aplikasi dapat memantau ketersediaan slot parkir dengan baik dan dan menampilkan data tersebut pada aplikasi secara *real-time* serta berhasil menyimpan data pada Google Sheets.
3. Untuk jarak antara *smartphone* dan lokasi parkir dalam radius di bawah 10 km, disimpulkan bahwa rata-rata latensi informasi data berkisar antara 1,74 detik sampai dengan 3,55 detik di bawah 10 detik.

5.2 Saran

Adapun saran yang dapat diajukan dari penelitian ini sebagai berikut:

1. Membuat lebih dari satu perangkat IoT sistem ketersediaan slot parkir untuk dikonfigurasikan menjadi satu dan dilihat apakah sistem dapat berjalan dengan baik menggunakan lebih dari satu perangkat IoT.
2. Melakukan pengembangan *interface* pada aplikasi android agar dapat melihat rute dan merekomendasikan slot parkir terdekat terlebih dahulu.

DAFTAR PUSTAKA

[1] L. Hartawan, M. A. Putra, Marsono, S. Rewidyo P and H. Firdaus, "Sistem Monitoring Ruang Parkir Kosong Berbasis Sensor Light Dependent Resistor," *JURNAL REKAYASA ENERGI DAN MEKANIKA,* vol. 2, no. 1, pp. 34-41, 2022.

[2] F. P. Eka Putra, S. Mellyana Dewi, Maugfiroh and A. Hamzah, "Privasi dan Keamanan Penerapan IoT Dalam Kehidupan Sehari-Hari : Tantangan dan Implikasi," *jsisfotek,* vol. 5, no. 2, pp. 26-32, 2023.

[3] A. A. R. Susila, I. Nasrullah, I. Denni and D. D. Bhakti, "Pelatihan Pembuatan Media Pembelajaran Berbasis Mobile Bagi Guru SMP," *BADRANAYA,* vol. 1, no. 2, pp. 60-66, 2023.

[4] H. Tri Laksono and Z. Budiarso, "Rancang Bangun Sistem Smart Parkir Berbasis Arduino," *STRING (Satuan Tulisan Riset dan Inovasi Teknologi),* vol. 7, no. 3, pp. 339-343, 2023.

[5] A. Purbo Wiseso, D. Irawan and R. Puji Astutik, "Rancang Bangun Sistem Informasi Ketersediaan Slot Parkir Dalam Mall," *Jurnal Teknik Elektro dan Informatika,* vol. 17, no. 2, pp. 19-25, 2022.

[6] T. A. D. Marwan, T. Berliana and F. A. Batubara, "Rancang Bangun Sistem Smart Parking Berbasis *Internet of Things* (IoT)," in *Konferensi Nasional Sosial dan Engineering Politeknik Negeri Medan 2021*, Medan, 2021.

[7] E. M. Punuh, "Rancang Bangun Sensor Parkir Kendaraan Roda Empat Berbasis Mikrokontroler Arduino Uno," *Jambura Journal of Electrical and Electronics Engineering,* vol. 6, no. 1, pp. 18-24, 2024.

[8] Cytron Technologies, "HCSR04 User's_Manual.web.eece.maine.edu," May 2013. [Online]. Available: https://web.eece.maine.edu/~zhu/book/lab/HC-SR04%20User%20Manual.pdf. [Accessed 1 April 2024].

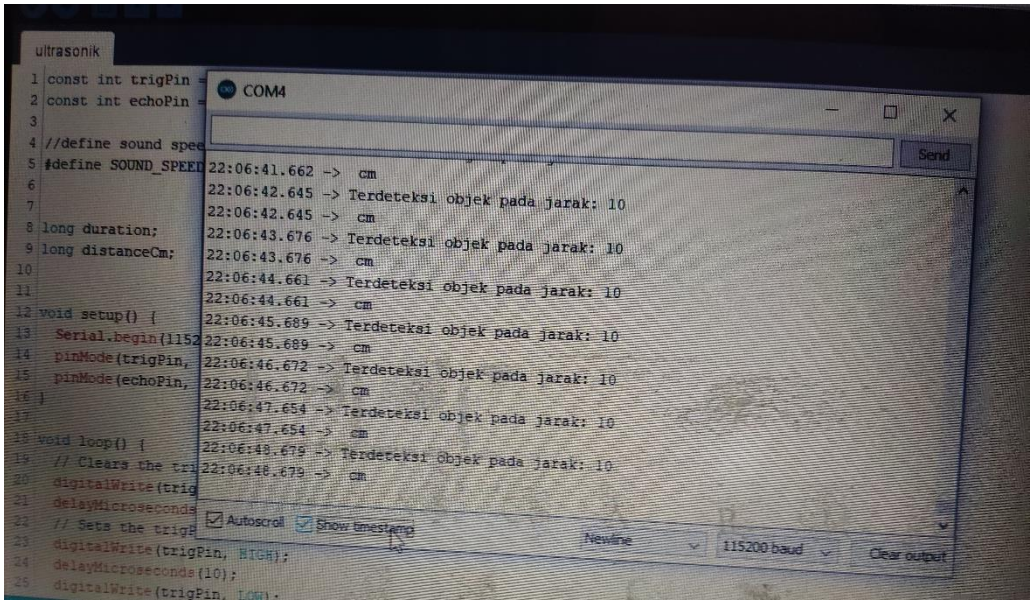
[9] A. Chaudhary and N. Chauhan, "*INTERNET OF THINGS* (IOT): Research Challenges and Future Applications," *International Journal of Emerging Trends in Science and Technology,* vol. 9, no. 6, pp. 1-9, 2022.

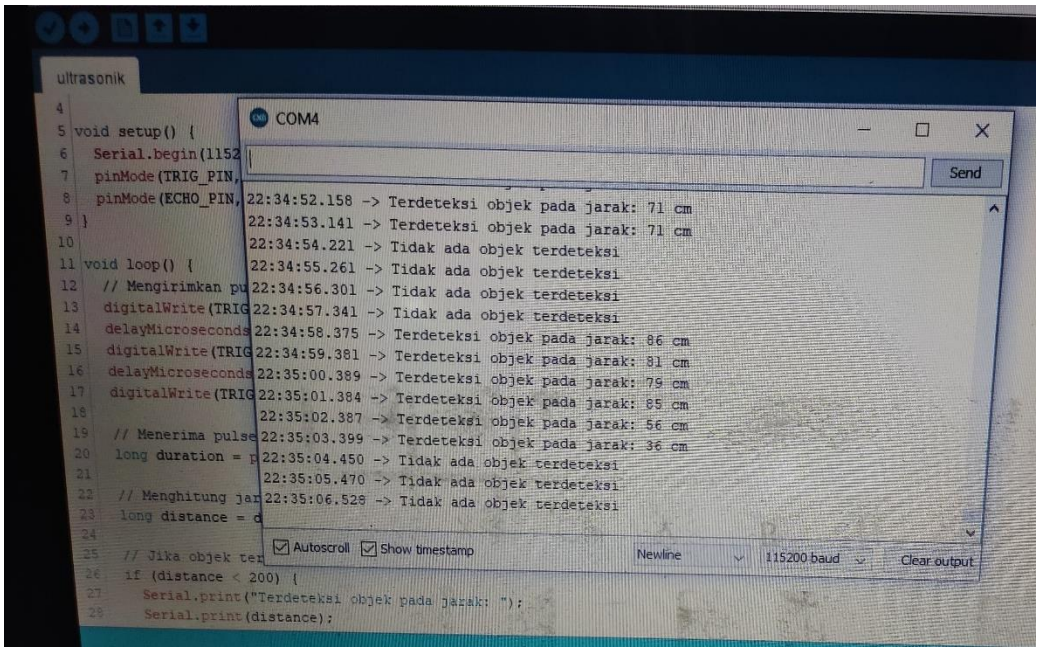
[10] R. Teja, "Getting Started with ESP32 | Introduction to ESP32," Electronics Hub, 17 February 2021. [Online]. Available: https://www.electronicshub.org/getting-started-with-esp32/. [Accessed 1 March 2024].

[11] Niagahoster, "Apa itu HTTPS? Berikut Pengertian dan Manfaatnya!," Niagahoster, 31 Mei 2022. [Online]. Available: https://www.niagahoster.co.id/blog/https-adalah-protokol-versi-aman/. [Accessed 15 Juni 2024].

LAMPIRAN

Lampiran 1. Pengujian Sensor dan Jarak




ultrasonik
void setup() {
Serial.begin(1152
pinMode(TRIG_PIN,
pinMode(ECHO_PIN,
}
void loop() {
// Mengirimkan pu
digitalWrite(TRIG
delayMicroseconds
digitalWrite(TRIG
delayMicroseconds
digitalWrite(TRIG
// Menerima pulse
long duration = p
// Menghitung jar
long distance = d
// Jika objek ter
if (distance < 200) {
Serial.print("Terdeteksi objek pada jarak: ");
Serial.print(distance);
COM4
Send
22:34:52.158 -> Terdeteksi objek pada jarak: 71 cm
22:34:53.141 -> Terdeteksi objek pada jarak: 71 cm
22:34:54.221 -> Tidak ada objek terdeteksi
22:34:55.261 -> Tidak ada objek terdeteksi
22:34:56.301 -> Tidak ada objek terdeteksi
22:34:57.341 -> Tidak ada objek terdeteksi
22:34:58.375 -> Terdeteksi objek pada jarak: 86 cm
22:34:59.381 -> Terdeteksi objek pada jarak: 81 cm
22:35:00.389 -> Terdeteksi objek pada jarak: 79 cm
22:35:01.384 -> Terdeteksi objek pada jarak: 85 cm
22:35:02.387 -> Terdeteksi objek pada jarak: 56 cm
22:35:03.399 -> Terdeteksi objek pada jarak: 36 cm
22:35:04.450 -> Tidak ada objek terdeteksi
22:35:05.470 -> Tidak ada objek terdeteksi
22:35:06.528 -> Tidak ada objek terdeteksi
Autoscroll
Show timestamp
Newline
115200 baud
Clear output

Lampiran 2. Data keseluruhan sistem

Tanggal	Waktu	Slot Tersedia Lantai 1	Slot Diisi Lantai 1	P1	P2	P3	P4	Slot Tersedia Lantai 2	Slot Diisi Lantai 2	P5	P6	P7	P8
11/06/2024	13:58:03	3	1	TRUE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	13:58:28	3	1	FALSE	TRUE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	13:59:28	3	1	FALSE	FALSE	TRUE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	14:00:29	3	1	FALSE	FALSE	FALSE	TRUE	4	0	FALSE	FALSE	FALSE	FALSE
11/6/2024	14:02:28	4	0	FALSE	FALSE	FALSE	FALSE	3	1	TRUE	FALSE	FALSE	FALSE
11/6/2024	14:03:28	4	0	FALSE	FALSE	FALSE	FALSE	3	1	FALSE	TRUE	FALSE	FALSE
11/6/2024	14:04:28	4	0	FALSE	FALSE	FALSE	FALSE	3	1	FALSE	FALSE	TRUE	FALSE
11/6/2024	14:05:28	4	0	FALSE	FALSE	FALSE	FALSE	3	1	FALSE	FALSE	FALSE	TRUE
11/6/2024	14:06:28	3	1	FALSE	TRUE	FALSE	FALSE	2	2	TRUE	FALSE	FALSE	TRUE
11/6/2024	14:07:29	3	1	FALSE	TRUE	FALSE	FALSE	2	2	TRUE	FALSE	FALSE	TRUE
11/6/2024	14:08:28	2	2	TRUE	TRUE	FALSE	FALSE	2	2	TRUE	FALSE	FALSE	TRUE
11/06/2024	14:09:29	1	3	TRUE	TRUE	FALSE	TRUE	1	3	TRUE	FALSE	TRUE	TRUE
11/06/2024	14:10:28	1	3	TRUE	TRUE	FALSE	TRUE	1	3	TRUE	FALSE	TRUE	TRUE
11/06/2024	14:11:29	0	4	TRUE	TRUE	TRUE	TRUE	1	3	TRUE	FALSE	TRUE	TRUE
11/06/2024	14:12:28	0	4	TRUE	TRUE	TRUE	TRUE	1	3	TRUE	FALSE	TRUE	TRUE
11/06/2024	14:13:28	0	4	TRUE	TRUE	TRUE	TRUE	0	4	TRUE	TRUE	TRUE	TRUE
11/06/2024	14:14:29	0	4	TRUE	TRUE	TRUE	TRUE	0	4	TRUE	TRUE	TRUE	TRUE
11/06/2024	14:15:28	1	3	FALSE	TRUE	TRUE	TRUE	0	4	TRUE	TRUE	TRUE	TRUE
11/06/2024	14:16:28	1	3	FALSE	TRUE	TRUE	TRUE	0	4	TRUE	TRUE	TRUE	TRUE
11/06/2024	14:17:28	1	3	FALSE	TRUE	TRUE	TRUE	0	4	TRUE	TRUE	TRUE	TRUE
11/06/2024	14:18:28	2	2	FALSE	TRUE	FALSE	TRUE	0	4	TRUE	TRUE	TRUE	TRUE
11/06/2024	14:19:28	2	2	FALSE	TRUE	FALSE	TRUE	0	4	TRUE	TRUE	TRUE	TRUE
11/06/2024	14:20:28	2	2	FALSE	TRUE	FALSE	TRUE	0	4	TRUE	TRUE	TRUE	TRUE
11/06/2024	14:21:28	2	2	FALSE	TRUE	FALSE	TRUE	0	4	TRUE	TRUE	TRUE	TRUE
11/06/2024	14:22:28	2	2	FALSE	TRUE	FALSE	TRUE	0	4	TRUE	TRUE	TRUE	TRUE
11/06/2024	14:23:28	2	2	FALSE	TRUE	FALSE	TRUE	0	4	TRUE	TRUE	TRUE	TRUE
11/06/2024	14:24:29	2	2	FALSE	TRUE	FALSE	TRUE	0	4	TRUE	TRUE	TRUE	TRUE
11/06/2024	14:25:28	2	2	FALSE	TRUE	FALSE	TRUE	0	4	TRUE	TRUE	TRUE	TRUE
11/06/2024	14:26:28	2	2	FALSE	TRUE	FALSE	TRUE	0	4	TRUE	TRUE	TRUE	TRUE
11/06/2024	14:27:28	2	2	FALSE	TRUE	FALSE	TRUE	0	4	TRUE	TRUE	TRUE	TRUE

11/06/2024	15:36:30	2	2	TRUE	FALSE	FALSE	TRUE	3	1	FALSE	FALSE	FALSE	TRUE
11/06/2024	15:37:28	2	2	TRUE	FALSE	FALSE	TRUE	2	2	TRUE	FALSE	FALSE	TRUE
11/06/2024	15:38:29	2	2	TRUE	FALSE	FALSE	TRUE	2	1	FALSE	FALSE	FALSE	TRUE
11/06/2024	15:39:28	2	2	TRUE	FALSE	FALSE	TRUE	2	2	TRUE	FALSE	FALSE	TRUE
11/06/2024	15:40:28	2	2	TRUE	FALSE	FALSE	TRUE	3	1	FALSE	FALSE	FALSE	TRUE
11/06/2024	15:41:28	2	2	TRUE	FALSE	FALSE	TRUE	3	1	FALSE	FALSE	FALSE	TRUE
11/06/2024	15:42:28	2	2	TRUE	FALSE	FALSE	TRUE	3	1	FALSE	FALSE	FALSE	TRUE
11/06/2024	15:43:28	2	2	TRUE	FALSE	FALSE	TRUE	3	1	FALSE	FALSE	FALSE	TRUE
11/06/2024	15:44:28	3	1	TRUE	FALSE	FALSE	FALSE	2	2	TRUE	FALSE	FALSE	TRUE
11/06/2024	15:45:28	3	1	TRUE	FALSE	FALSE	FALSE	3	1	FALSE	FALSE	FALSE	TRUE
11/06/2024	15:46:28	3	1	TRUE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	15:47:28	4	0	FALSE	FALSE	FALSE	FALSE	3	1	TRUE	FALSE	FALSE	FALSE
11/06/2024	15:48:28	4	0	FALSE	FALSE	FALSE	FALSE	3	1	TRUE	FALSE	FALSE	FALSE
11/06/2024	15:49:28	4	0	FALSE	FALSE	FALSE	FALSE	3	1	TRUE	FALSE	FALSE	FALSE
11/06/2024	15:50:28	4	0	FALSE	FALSE	FALSE	FALSE	3	1	TRUE	FALSE	FALSE	FALSE
11/06/2024	15:51:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	15:52:28	4	0	FALSE	FALSE	FALSE	FALSE	3	1	FALSE	FALSE	FALSE	FALSE
11/06/2024	15:53:31	4	0	FALSE	FALSE	FALSE	FALSE	3	1	TRUE	FALSE	FALSE	FALSE
11/06/2024	15:54:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	15:55:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	15:56:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	15:57:28	4	0	FALSE	FALSE	FALSE	FALSE	3	1	TRUE	FALSE	FALSE	FALSE
11/06/2024	15:58:29	4	0	FALSE	FALSE	FALSE	FALSE	4	0	TRUE	FALSE	FALSE	FALSE
11/06/2024	15:59:28	4	0	FALSE	FALSE	FALSE	FALSE	3	1	TRUE	FALSE	FALSE	FALSE
11/06/2024	16:00:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:01:29	4	0	FALSE	FALSE	FALSE	FALSE	3	1	TRUE	FALSE	FALSE	FALSE
11/06/2024	16:02:28	4	0	FALSE	FALSE	FALSE	FALSE	3	1	TRUE	FALSE	FALSE	FALSE
11/06/2024	16:03:29	4	0	FALSE	FALSE	FALSE	FALSE	4	0	TRUE	FALSE	FALSE	FALSE
11/06/2024	16:04:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:05:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:06:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:07:29	4	0	FALSE	FALSE	FALSE	FALSE	3	1	TRUE	FALSE	FALSE	FALSE
11/06/2024	16:08:29	4	0	FALSE	FALSE	FALSE	FALSE	3	1	TRUE	FALSE	FALSE	FALSE
11/06/2024	16:09:29	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE

11/06/2024	16:14:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:15:28	4	0	FALSE	FALSE	FALSE	FALSE	3	1	TRUE	FALSE	FALSE	FALSE
11/06/2024	16:16:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:17:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:18:30	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:19:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:20:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:21:28	4	0	FALSE	FALSE	FALSE	FALSE	3	1	TRUE	FALSE	FALSE	FALSE
11/06/2024	16:22:30	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:23:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:24:29	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:25:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:26:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:27:28	4	0	FALSE	FALSE	FALSE	FALSE	3	1	TRUE	FALSE	FALSE	FALSE
11/06/2024	16:28:29	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:29:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:30:29	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:31:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:32:29	4	0	FALSE	FALSE	FALSE	FALSE	3	1	TRUE	FALSE	FALSE	FALSE
11/06/2024	16:33:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	TRUE	FALSE	FALSE	FALSE
11/06/2024	16:34:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:35:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:36:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:37:29	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:38:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:39:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:40:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:41:29	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:42:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:43:28	4	0	FALSE	FALSE	FALSE	FALSE	3	1	TRUE	FALSE	FALSE	FALSE
11/06/2024	16:44:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:45:29	4	0	FALSE	FALSE	FALSE	FALSE	4	0	TRUE	FALSE	FALSE	FALSE
11/06/2024	16:46:30	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE
11/06/2024	16:47:28	4	0	FALSE	FALSE	FALSE	FALSE	4	0	FALSE	FALSE	FALSE	FALSE

Lampiran 3. Inisialisasi *Hardware* sistem informasi ketersediaan parkir

```
#include <WiFi.h>
#include <Firebase_ESP_Client.h>
#include <addons/TokenHelper.h>
#include <addons/RTDBHelper.h>

#define WIFI_SSID "UNILA-PGN-22"
#define WIFI_PASSWORD ""
#define API_KEY "AIzaSyAASGwEZit_J-Rc1S0yOivXpsR2o5oDw3E"
#define DATABASE_URL "https://parkirslotiot-default-rtdb.asia-southeast1.firebasedatabase.app"

int totalslot1 = 4;
int slotdiisi1;
int slottersedia1;

int totalslot2 = 4;
int slotdiisi2;
int slottersedia2;

int nilaidistance1 = 0;
int nilaidistance2 = 0;
int nilaidistance3 = 0;
int nilaidistance4 = 0;
int nilaidistance5 = 0;
int nilaidistance6 = 0;
int nilaidistance7 = 0;
int nilaidistance8 = 0;

const int trigPin1 = 5;
const int echoPin1 = 18;

const int trigPin2 = 17;
```

```
const int echoPin2 = 16;

const int trigPin3 = 4;
const int echoPin3 = 0;

const int trigPin4 = 2;
const int echoPin4 = 15;

const int trigPin5 = 14;
const int echoPin5 = 27;

const int trigPin6 = 19;
const int echoPin6 = 21;

const int trigPin7 = 26;
const int echoPin7 = 25;

const int trigPin8 = 22;
const int echoPin8 = 23;

FirebaseData fbdo;
FirebaseAuth auth;
FirebaseConfig config;

unsigned long sendDataPrevMillis = 0;
int ultrasonik = 0;
bool letak1 = false;
bool letak2 = false;
bool letak3 = false;
bool letak4 = false;
bool letak5 = false;
bool letak6 = false;
```

```
bool letak7 = false;
bool letak8 = false;
bool signUpOK = false;

void setup()
{

  Serial.begin(115200);
    // Inisialisasi pin untuk sensor ultrasonik
  pinMode(trigPin1, OUTPUT);
  pinMode(echoPin1, INPUT);
  pinMode(trigPin2, OUTPUT);
  pinMode(echoPin2, INPUT);
  pinMode(trigPin3, OUTPUT);
  pinMode(echoPin3, INPUT);
  pinMode(trigPin4, OUTPUT);
  pinMode(echoPin4, INPUT);
  pinMode(trigPin5, OUTPUT);
  pinMode(echoPin5, INPUT);
  pinMode(trigPin6, OUTPUT);
  pinMode(echoPin6, INPUT);
  pinMode(trigPin7, OUTPUT);
  pinMode(echoPin7, INPUT);
  pinMode(trigPin8, OUTPUT);
  pinMode(echoPin8, INPUT);


  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.print(".");
```

```
  delay(300);
 }
 Serial.println();
 Serial.print("Connected with IP: ");
 Serial.println(WiFi.localIP());
 Serial.println();

 Serial.printf("Firebase Client v%s\n\n", FIREBASE_CLIENT_VERSION);

 config.api_key = API_KEY;
 config.database_url = DATABASE_URL;
 if(Firebase.signUp(&config, &auth, "", "")){
  Serial.println("signUp OK");
  signUpOK = true;
 }else{
  Serial.printf("%s/n", config.signer.signupError.message.c_str());
 }

 config.token_status_callback = tokenStatusCallback;
 Firebase.begin(&config, &auth);
 Firebase.reconnectNetwork(true);
}

void loop()
{
 if (Firebase.ready() && (millis() - sendDataPrevMillis > 5000 || sendDataPrevMillis ==
0))
 {
  sendDataPrevMillis = millis();

  long duration1, distance1;
  digitalWrite(trigPin1, LOW);
  delayMicroseconds(2);
```

```
digitalWrite(trigPin1, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin1, LOW);
duration1 = pulseIn(echoPin1, HIGH);
distance1 = (duration1 * 0.0343) / 2;

long duration2, distance2;
digitalWrite(trigPin2, LOW);
delayMicroseconds(2);
digitalWrite(trigPin2, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin2, LOW);
duration2 = pulseIn(echoPin2, HIGH);
distance2 = (duration2 * 0.0343) / 2;

long duration3, distance3;
digitalWrite(trigPin3, LOW);
delayMicroseconds(2);
digitalWrite(trigPin3, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin3, LOW);
duration3 = pulseIn(echoPin3, HIGH);
distance3 = (duration3 * 0.0343) / 2;

long duration4, distance4;
digitalWrite(trigPin4, LOW);
delayMicroseconds(2);
digitalWrite(trigPin4, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin4, LOW);
duration4 = pulseIn(echoPin4, HIGH);
distance4 = (duration4 * 0.0343) / 2;
```

```
 long duration5, distance5;
digitalWrite(trigPin5, LOW);
delayMicroseconds(2);
digitalWrite(trigPin5, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin5, LOW);
duration5 = pulseIn(echoPin5, HIGH);
distance5 = (duration5 * 0.0343) / 2;

long duration6, distance6;
digitalWrite(trigPin6, LOW);
delayMicroseconds(2);
digitalWrite(trigPin6, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin6, LOW);
duration6 = pulseIn(echoPin6, HIGH);
distance6 = (duration6 * 0.0343) / 2;

long duration7, distance7;
digitalWrite(trigPin7, LOW);
delayMicroseconds(2);
digitalWrite(trigPin7, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin7, LOW);
duration7 = pulseIn(echoPin7, HIGH);
distance7 = (duration7 * 0.0343) / 2;

long duration8, distance8;
digitalWrite(trigPin8, LOW);
delayMicroseconds(2);
digitalWrite(trigPin8, HIGH);
```

```
delayMicroseconds(10);
digitalWrite(trigPin8, LOW);
duration8 = pulseIn(echoPin8, HIGH);
distance8 = (duration8 * 0.0343) / 2;

 if (distance1 < 6)
{Firebase.RTDB.setBool(&fbdo, "sensor/letak1", true);
 nilaidistance1 = 1;

} else {
    Firebase.RTDB.setBool(&fbdo, "sensor/letak1", false);
    nilaidistance1 = 0;

  }
   Serial.print("Slot 1: ");
   Serial.println(nilaidistance1);

if (distance2 < 6)
{Firebase.RTDB.setBool(&fbdo, "sensor/letak2", true);
 nilaidistance2 = 1;

  } else {
    Firebase.RTDB.setBool(&fbdo, "sensor/letak2", false);
    nilaidistance2 = 0;

  }
    Serial.print("Slot 2: ");
    Serial.println(nilaidistance2);

if (distance3 < 6)
{Firebase.RTDB.setBool(&fbdo, "sensor/letak3", true);
 nilaidistance3 = 1;
```

```
} else {
    Firebase.RTDB.setBool(&fbdo, "sensor/letak3", false);
    nilaidistance3 = 0;

  }
    Serial.print("Slot 3: ");
    Serial.println(nilaidistance3);

if (distance4 < 6)
{Firebase.RTDB.setBool(&fbdo, "sensor/letak4", true);
 nilaidistance4 = 1;

  } else {
    Firebase.RTDB.setBool(&fbdo, "sensor/letak4", false);
    nilaidistance4 = 0;

  }
    Serial.print("Slot 4: ");
    Serial.println(nilaidistance4);

if (distance5 < 6)
{Firebase.RTDB.setBool(&fbdo, "sensor/letak5", true);
nilaidistance5 = 1;

} else {
    Firebase.RTDB.setBool(&fbdo, "sensor/letak5", false);
    nilaidistance5 = 0;

  }
    Serial.print("Slot 5: ");
    Serial.println(nilaidistance5);
```

```
if (distance6 < 6)
{Firebase.RTDB.setBool(&fbdo, "sensor/letak6", true);
nilaidistance6 = 1;

  } else {
    Firebase.RTDB.setBool(&fbdo, "sensor/letak6", false);
    nilaidistance6 = 0;

  }
    Serial.print("Slot 6: ");
    Serial.println(nilaidistance6);

if (distance7 < 6)
{Firebase.RTDB.setBool(&fbdo, "sensor/letak7", true);
nilaidistance7 = 1;

} else {
    Firebase.RTDB.setBool(&fbdo, "sensor/letak7", false);
    nilaidistance7 = 0;

  }
    Serial.print("Slot 7: ");
    Serial.println(nilaidistance7);

if (distance8 < 6)
{Firebase.RTDB.setBool(&fbdo, "sensor/letak8", true);
nilaidistance8 = 1;

  } else {
    Firebase.RTDB.setBool(&fbdo, "sensor/letak8", false);
    nilaidistance8 = 0
```

```
  }
    Serial.print("Slot 8: ");
    Serial.println(nilaidistance8);

slotdiisi1 = ( nilaidistance1 + nilaidistance2 + nilaidistance3 + nilaidistance4);
slottersedia1 = totalslot1 - slotdiisi1;

slotdiisi2 = ( nilaidistance5 + nilaidistance6 + nilaidistance7 + nilaidistance8);
slottersedia2 = totalslot2 - slotdiisi2;

 Serial.print("Slot tersedia 1: ");
 Serial.println(slottersedia1);
 Serial.print("Slot tersedia 2: ");
  Serial.println(slottersedia2);
Firebase.RTDB.setInt(&fbdo, "slotdiisi1", slotdiisi1);
Firebase.RTDB.setInt(&fbdo, "slotdiisi2", slotdiisi2);

Firebase.RTDB.setInt(&fbdo, "slottersedia1", slottersedia1);
Serial.println("Data slottersedia1 berhasil terkirim.");
Firebase.RTDB.setInt(&fbdo, "slottersedia2", slottersedia2);
Serial.println("Data slottersedia2 berhasil terkirim.");
}
}
```

Lampiran 4. Program *Software* sistem informasi ketersediaan parkir

Splash screen

```
package com.example.parkirku;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.os.Handler;
public class SplashActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_splash);
        new Handler().postDelayed(new Runnable() {
            @Override
            public void run() {
                Intent intent = new Intent(SplashActivity.this, MainActivity.class);
                startActivity(intent);
                finish();
            }
        }, 3000);
    }
}
```

Halaman Pilih Lokasi

```
package com.example.parkirku;
import androidx.appcompat.app.AppCompatActivity;
import android.annotation.SuppressLint;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;
```

```
import com.firebase.client.DataSnapshot;
import com.firebase.client.Firebase;
import com.firebase.client.FirebaseError;
import com.firebase.client.ValueEventListener;
public class MainActivity extends AppCompatActivity {
    private TextView tersisa1, tersisa2 ;
    private Firebase mRef, mRef2 ;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        tersisa1 = (TextView) findViewById(R.id.tersisa1);
        tersisa2 = (TextView) findViewById(R.id.tersisa2);
        mRef = new Firebase("https://parkirslotiot-default-rtdb.asia-
southeast1.firebasedatabase.app/slotdiisi1");
        mRef2 = new Firebase("https://parkirslotiot-default-rtdb.asia-
southeast1.firebasedatabase.app/slotdiisi2");



        mRef.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(DataSnapshot dataSnapshot) {
                String value = dataSnapshot.getValue(String.class);
                if (value != null) {
                    try {
                        int intValue = Integer.parseInt(value);
                        int sisa = 8 - intValue;
                        tersisa1.setText(String.valueOf(sisa));
                    } catch (NumberFormatException e) {
                        // Handle jika value tidak dapat diubah menjadi integer
                    }
                }
```

```
        }
        @Override
        public void onCancelled(FirebaseError firebaseError) {
        }
    });
    mRef2.addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {
            String value2 = dataSnapshot.getValue(String.class);
            if (value2 != null) {
                try {
                    int intValue2 = Integer.parseInt(value2);
                    int sisa2 = 8 - intValue2;
                    tersisa2.setText(String.valueOf(sisa2));
                } catch (NumberFormatException e) {
                    // Handle jika value tidak dapat diubah menjadi integer
                }
            }
        }
        @Override
        public void onCancelled(FirebaseError firebaseError) {
        }
    });
}
public void lokasi(View view){
    Intent intent = new Intent(MainActivity.this,MainActivity3.class);
    startActivity(intent);
}
public void lokasi2(View view){
    Intent intent = new Intent(MainActivity.this,MainActivity5.class);
    startActivity(intent);
```

```
    }
    public void lokasi3(View view) {
        Intent intent = new Intent(MainActivity.this, MainActivity8.class);
        startActivity(intent);
    }
}
```

Halaman Pilih Lantai

```
package com.example.parkirku;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;
import android.widget.ImageView;
import com.firebase.client.DataSnapshot;
import com.firebase.client.Firebase;
import com.firebase.client.FirebaseError;
import com.firebase.client.ValueEventListener;
public class MainActivity3 extends AppCompatActivity {
    private TextView tersisa1, tersisa2 ;
    //buat refrence firebase koneksi ke server host
    private Firebase mRef, mRef2 ;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main3);
        tersisa1 = (TextView) findViewById(R.id.tersisa1);
        tersisa2 = (TextView) findViewById(R.id.tersisa2);
        mRef = new Firebase("https://parkirslotiot-default-rtdb.asia-
southeast1.firebasedatabase.app/slotdiisi1");
```

```
        mRef2 = new Firebase("https://parkirslotiot-default-rtdb.asia-
southeast1.firebasedatabase.app/slotdiisi2");
        mRef.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(DataSnapshot dataSnapshot) {
                String value = dataSnapshot.getValue(String.class);
                if (value != null) {
                    try {
                        int intValue = Integer.parseInt(value);
                        int sisa = 4 - intValue;
                        tersisa1.setText(String.valueOf(sisa));
                    } catch (NumberFormatException e) {
                        // Handle jika value tidak dapat diubah menjadi integer
                    }
                }
            }
            @Override
            public void onCancelled(FirebaseError firebaseError) {
            }
        });
        mRef2.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(DataSnapshot dataSnapshot) {
                String value = dataSnapshot.getValue(String.class);
                if (value != null) {
                    try {
                        int intValue = Integer.parseInt(value);
                        int sisa = 4 - intValue;
                        tersisa2.setText(String.valueOf(sisa));
                    } catch (NumberFormatException e) {
                        // Handle jika value tidak dapat diubah menjadi integer
```

```
                }
            }
        }
        @Override
        public void onCancelled(FirebaseError firebaseError) {
        }
    });
  }
  public void lantai(View view){
    Intent intent = new Intent(MainActivity3.this,MainActivity2.class);
    startActivity(intent);
  }
  public void lantai2(View view) {
    Intent intent = new Intent(MainActivity3.this, MainActivity4.class);
    startActivity(intent);
  }
}
```

Halaman Slot Tersedia dan Letak Slot Parkir

```
 package com.example.parkirku;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageView;
import android.widget.TextView;
import com.firebase.client.DataSnapshot;
import com.firebase.client.Firebase;
import com.firebase.client.FirebaseError;
import com.firebase.client.ValueEventListener;
public class MainActivity2 extends AppCompatActivity {
  private TextView nilai ;
```

```
    private ImageView ada1, ada2, ada3, ada4, kosong1, kosong2, kosong3, kosong4  ;
    private Firebase mRef, mRef2, mRef3, mRef4, mRef5 ;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main2);
        nilai = (TextView) findViewById(R.id.nilai) ;
        ada1 = findViewById(R.id.ada1);
        kosong1 = findViewById(R.id.kosong1);
        ada2 = findViewById(R.id.ada2);
        kosong2 = findViewById(R.id.kosong2);
        ada3 = findViewById(R.id.ada3);
        kosong3 = findViewById(R.id.kosong3);
        ada4 = findViewById(R.id.ada4);
        kosong4 = findViewById(R.id.kosong4);


        mRef = new Firebase("https://parkirslotiot-default-rtdb.asia-
southeast1.firebasedatabase.app/slotdiisi1") ;
        mRef2 = new Firebase("https://parkirslotiot-default-rtdb.asia-
southeast1.firebasedatabase.app/sensor/letak1");
        mRef3 = new Firebase("https://parkirslotiot-default-rtdb.asia-
southeast1.firebasedatabase.app/sensor/letak2");
        mRef4 = new Firebase("https://parkirslotiot-default-rtdb.asia-
southeast1.firebasedatabase.app/sensor/letak3");
        mRef5 = new Firebase("https://parkirslotiot-default-rtdb.asia-
southeast1.firebasedatabase.app/sensor/letak4");
        mRef.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(DataSnapshot dataSnapshot) {
                String value = dataSnapshot.getValue(String.class);
                if (value != null) {
                    try {
                        int intValue = Integer.parseInt(value);
```

```
            int sisa = 4 - intValue;
            nilai.setText(String.valueOf(sisa));
        } catch (NumberFormatException e) {
            // Handle jika value tidak dapat diubah menjadi integer
        }
    }
  }
  @Override
  public void onCancelled(FirebaseError firebaseError) {
  }
}) ;
mRef2.addValueEventListener(new ValueEventListener() {
  @Override
  public void onDataChange(DataSnapshot dataSnapshot) {
    boolean letak1 = dataSnapshot.getValue(Boolean.class);
    if (letak1 == false) {
      ada1.setVisibility(View.VISIBLE);
      kosong1.setVisibility(View.INVISIBLE);
    } else {
      ada1.setVisibility(View.INVISIBLE);
      kosong1.setVisibility(View.VISIBLE);
    }
  }
  @Override
  public void onCancelled(FirebaseError firebaseError) {
    // Handle error
  }
});
mRef3.addValueEventListener(new ValueEventListener() {
  @Override
  public void onDataChange(DataSnapshot dataSnapshot) {
```

```
        boolean letak2 = dataSnapshot.getValue(Boolean.class);
        if (letak2 == false) {
            ada2.setVisibility(View.VISIBLE);
            kosong2.setVisibility(View.INVISIBLE);
        } else {
            ada2.setVisibility(View.INVISIBLE);
            kosong2.setVisibility(View.VISIBLE);
        }
    }
    @Override
    public void onCancelled(FirebaseError firebaseError) {
        // Handle error
    }
});
mRef4.addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {
        boolean letak3 = dataSnapshot.getValue(Boolean.class);
        if (letak3 == false) {
            ada3.setVisibility(View.VISIBLE);
            kosong3.setVisibility(View.INVISIBLE);
        } else {
            ada3.setVisibility(View.INVISIBLE);
            kosong3.setVisibility(View.VISIBLE);
        }
    }
    @Override
    public void onCancelled(FirebaseError firebaseError) {
        // Handle error
    }
});
```

```
        mRef5.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(DataSnapshot dataSnapshot) {
                boolean letak4 = dataSnapshot.getValue(Boolean.class);
                if (letak4 == false) {
                    ada4.setVisibility(View.VISIBLE);
                    kosong4.setVisibility(View.INVISIBLE);
                } else {
                    ada4.setVisibility(View.INVISIBLE);
                    kosong4.setVisibility(View.VISIBLE);
                }
            }
            @Override
            public void onCancelled(FirebaseError firebaseError) {
                // Handle error
            }
        });
    }
}
```

Plagiarism Checker X Originality Report

Similarity Found: 10%

Date: Wednesday, August 07, 2024
Statistics: 1743 words Plagiarized / 16679 Total words
Remarks: Low Plagiarism Detected - Your Document needs Optional Improvement.

RANCANG BANGUN SISTEM PEMANTAU KETERSEDIAAN TEMPAT PARKIR REAL TIME MENGGUNAKAN MIKROKONTROLER BERBASIS INTERNET OF THINGS (Skripsi) Oleh Refli Nicholas Hakim / PROGRAM STUDI TEKNIK ELEKTRO JURUSAN TEKNIK ELEKTRO FAKULTAS TEKNIK UNIVERSITAS LAMPUNG BANDAR LAMPUNG 2024 RANCANG BANGUN SISTEM PEMANTAU KETERSEDIAAN TEMPAT PARKIR MENGGUNAKAN MIKROKONTROLER BERBASIS INTERNET OF THINGS Oleh REFLI NICHOLAS HAKIM Skripsi Sebagai salah satu syarat untuk mendapat gelar SARJANA TEKNIK pada Jurusan Teknik Elektro Fakultas Teknik Universitas Lampung / PROGRAM STUDI TEKNIK ELEKTRO JURUSAN TEKNIK ELEKTRO FAKULTAS TEKNIK UNIVERSITAS LAMPUNG BANDAR LAMPUNG 2024

DAFTAR ISI Halaman DAFTAR ISI i DAFTAR GAMBAR iii DAFTAR TABEL v I. PENDAHULUAN 1 1.1 Latar Belakang 1 1.2 Tujuan Penelitian 2 1.3 Manfaat Penelitian 2 1.4

Rumusan Masalah 3 1.5 Batasan Masalah 3 1.6 Hipotesis 3 1.7 Sistematika Penulisan 3 II. TINJAUAN PUSTAKA 5 2.1 Penelitian Terdahulu 5 2.2 Teori Dasar 7 2.2.1 Prinsip Sensor Ultrasonik 7 2.2.2 Konsep Internet of Things 8 2.2.2 Mikrokontroler 10 III. METODE PENELITIAN 11 3.1 Waktu dan Tempat Penelitian 11 3.2 Alat dan Bahan 11 3.3 Metode Penelitian 12 3.4 Skenario Perancangan Sistem 13 3.5 Diagram Alir Sistem 14 3.5.1

Device ESP32 sebagai prosesor dan perangkat IoT 16 3.5.2 Perancangan Hardware 21 3.5.2.1 Diagram Blok Alat 21 3.5.2.2 Skema Perancangan Alat 21 3.5.3 Perancangan Aplikasi 23 3.5.3.1 Desain Tampilan Aplikasi 25 3.5.3.2 Android Studio 26 3.6 Pengujian Sistem 34 ?. HASIL DAN PEMBAHASAN 35 4.1 Prinsip Kerja 35 4.2 Pengujian Subsistem 35 4.2.1 Pengujian Hardware 35 4.2.1.1 Pengujian Jarak antara Sensor HC-SR04 dan Mobil 36 4.2.1.2

Pengujian Respon Sensor Ultrasonik HC-SR04 37 4.2.1.3 Pengujian Akurasi Sensor Ultrasonik HC-SR04 46 4.2.2 Pengujian Software 51 4.2.2.1 Pengujian Database 51 4.2.2.2 Pengujian Aplikasi 54 4.3 Pengujian Latensi Sistem 59 4.4. Pengujian Latensi dengan Jarak 62 ?. KESIMPULAN DAN SARAN 67 5.1 Kesimpulan 67 5.2 Saran 67 DAFTAR PUSTAKA 68 LAMPIRAN 70 DAFTAR GAMBAR Gambar 2.1. Konsep penelitian sistem informasi ketersediaan slot parkir 6 Gambar 2.2.

Prinsip sensor ultrasonik HC-SR04 7 Gambar 2.3. Konsep Internet of Things (IoT) 9 Gambar 3.1. Konsep rancangan sistem ketersediaan slot parkir berbasis android 12 Gambar 3.2. Skenario pemasangan sensor 13 Gambar 3.3. Diagram Alir Sistem 15 Gambar 3.4. Diagram Blok Alat 21 Gambar 3.5. Skematik Perancangan Sistem 22 Gambar 3.6.. Perangkat keras sistem deteksi kendaraan pada slot parkir 23 Gambar 3.7.

Desain Tampilan Aplikasi 25 Gambar 4.1. Prototype Alat Pemantau Ketersedian Slot Parkir 36 Gambar 4.2. Grafik Pengujian Respon Sensor Slot Parkir P1 38 Gambar 4.3. Grafik Pengujian Respon Sensor Slot Parkir P2 39 Gambar 4.4. Grafik Pengujian Respon Sensor Slot Parkir P3 40 Gambar 4.5. Grafik Pengujian Respon Sensor Slot Parkir P4 41 Gambar 4.6. Grafik Pengujian Respon Sensor Slot Parkir P5 42 Gambar 4.7.

Grafik Pengujian Respon Sensor Slot Parkir P6 43 Gambar 4.8. Grafik