E-ISSN: 3025-5481
DINASTI INFORMATION AND TECHNOLOGY
DINASTI RESEARCH
https://dinastires.org/DIT dinasti.info@gmail.com +62 811 7404 455

DOI: https://doi.org/10.38035/dit.v4i2.3958
https://creativecommons.org/licenses/by/4.0/
Received: 10 October 2026 | **Revised:** 04 November 2026 | **Accepted:** 08 November 2026

Rancang Bangun Sistem Running Text Dua Panel Berbasis Internet of Things (IOT) Menggunakan Aplikasi Mobile

Nur Shita Harata[1], Seliwati[2]
[1]Teknik Komputer, Politeknik Piksi Ganesha, Indonesia, itaharata@gmail.com
[2]Teknik Komputer, Politeknik Piksi Ganesha, Indonesia, seliwati88@gmail.com

Corresponding Author: itaharata@gmail.com[1]

Abstract: *The advancement of digital technology has increased the need for information media that can be easily updated and controlled. This study aims to design and develop a two-panel running text system based on the Internet of Things (IoT) using a mobile application. The system uses two ESP32 microcontrollers as controllers, with each device controlling one RGB P10 LED panel. The mobile application was developed using Flutter to manage messages, display modes, text speed, brightness level, and power status through a local WiFi network. Data is transmitted from the application to each ESP32 using the HTTP POST method in JSON format. The test results show that both panels can receive and display different configurations and operate independently. The running, static, and blinking display modes operate according to the given commands, while the speed, brightness, and power settings function as configured. Communication range testing achieved a success rate of 100% at distances of 5, 10, and 15 meters, while the connection was lost at distances beyond 15 meters, resulting in an overall average success rate of 75%. The system also continues to display the running text after the mobile application is disconnected, as the received data is processed and executed locally by the ESP32. These results demonstrate that the system can be used to update information on two panels wirelessly without reprogramming the devices and without relying on cloud services or public internet connections.*

Keywords: *Running text, Internet of Things (IoT), ESP32, LED P10, Flutter.*

Abstrak: Kemajuan teknologi digital meningkatkan kebutuhan terhadap media informasi yang dapat diperbarui dan dikendalikan secara mudah. Penelitian ini bertujuan untuk merancang dan membangun sistem running text dua panel berbasis Internet of Things (IoT) menggunakan aplikasi mobile. Sistem menggunakan dua ESP32 sebagai pengendali masing-masing panel LED P10 RGB. Aplikasi mobile dikembangkan menggunakan Flutter untuk mengatur pesan, mode tampilan, kecepatan teks, tingkat kecerahan, dan status daya melalui jaringan WiFi lokal. Data dikirim dari aplikasi ke masing-masing ESP32 menggunakan metode HTTP POST dalam format JSON. Hasil pengujian menunjukkan bahwa kedua panel dapat menerima dan menampilkan konfigurasi yang berbeda serta bekerja secara independen. Fitur mode berjalan, diam, dan berkedip dapat dijalankan sesuai perintah, sedangkan pengaturan kecepatan, kecerahan, dan daya dapat berfungsi sesuai konfigurasi yang diberikan. Pengujian jangkauan

komunikasi menunjukkan tingkat keberhasilan sebesar 100% pada jarak 5 meter, 10 meter, dan 15 meter, sedangkan pada jarak lebih dari 15 meter koneksi terputus, sehingga diperoleh rata-rata tingkat keberhasilan sebesar 75%. Sistem juga tetap menjalankan running text setelah koneksi aplikasi terputus, karena data yang telah diterima diproses dan dijalankan secara lokal oleh ESP32. Hasil tersebut menunjukkan bahwa sistem dapat digunakan untuk memperbarui informasi pada dua panel secara nirkabel tanpa pemrograman ulang perangkat dan tanpa ketergantungan terhadap layanan cloud atau koneksi internet publik.

Kata Kunci: *Running text, Internet of Things (IoT)*, ESP32, LED P10, Flutter.

PENDAHULUAN

Perkembangan teknologi informasi membawa perubahan dalam berbagai cara penyampaian informasi. Penggunaan media konvensional seperti papan pengumuman, poster, dan tulisan cetak secara bertahap mulai dilengkapi dengan media digital yang lebih mudah diperbarui. Salah satu keunggulan media digital adalah informasi yang ditampilkan dapat diubah sesuai kebutuhan tanpa harus mengganti media secara fisik. Salah satu bentuk media informasi digital yang cukup banyak digunakan adalah running text. Running text merupakan media elektronik yang menampilkan informasi dalam bentuk tulisan secara dinamis melalui susunan LED sehingga dapat digunakan sebagai media penyampaian informasi yang mudah diperbarui (Triwahyuni & Beta, 2022; Yuhanas et al., 2021). Penggunaannya dapat ditemukan di berbagai tempat, seperti lingkungan pendidikan, perkantoran, pusat pelayanan, fasilitas umum, serta tempat kegiatan masyarakat. Kemampuan menampilkan informasi secara bergerak membuat running text dapat menjadi salah satu media penyampaian informasi yang lebih menarik dan mudah diperhatikan.

Panel LED P10 merupakan salah satu jenis LED matrix yang dapat dimanfaatkan dalam pembuatan running text. Panel tersebut tersusun dari sejumlah LED yang membentuk matriks sehingga dapat digunakan untuk menampilkan teks, warna, maupun pola sederhana pada sistem informasi digital (Yuhanas et al., 2021). Agar informasi dapat ditampilkan sesuai kebutuhan, panel LED membutuhkan perangkat pengendali yang dapat mengatur data piksel, warna, posisi tulisan, serta pergerakan teks pada layar. Pengendalian tampilan LED matriks melalui jaringan Wi-Fi nirkabel juga dapat mempercepat transmisi data dan respons perintah (Nataprawira et al., 2020). Dalam penelitian ini, dua ESP32 DevKit V1 digunakan sebagai perangkat edge controller pada sistem. Selain memiliki kemampuan pemrosesan yang cukup untuk mengendalikan panel LED, ESP32 juga telah dilengkapi dengan fitur WiFi. Fitur tersebut memungkinkan proses komunikasi dengan aplikasi dilakukan secara nirkabel. Dengan demikian, perubahan informasi yang ditampilkan dapat dilakukan melalui aplikasi tanpa harus menghubungkan perangkat menggunakan kabel atau melakukan pemrograman ulang pada mikrokontroler setiap kali pesan diganti.

Komunikasi pada sistem menerapkan konsep Internet of Things (IoT), yaitu konsep yang memungkinkan perangkat fisik saling terhubung dan berkomunikasi untuk melakukan pertukaran informasi serta mengoordinasikan suatu tindakan (Watmah, 2025). Pada penelitian ini, penerapan IoT menggunakan jaringan lokal yang menghubungkan aplikasi mobile dengan ESP32 sebagai pengendali panel LED P10. Penerapan IoT pada sistem running text memungkinkan perangkat melakukan komunikasi dan pertukaran data sehingga informasi dapat diperbarui tanpa melakukan pemrograman ulang pada perangkat (Budioko & Bednar, 2025; Syahputra et al., 2024). Penggunaan jaringan lokal membuat komunikasi dapat berlangsung secara langsung tanpa membutuhkan layanan cloud maupun koneksi internet publik. Penerapan jaringan nirkabel lokal pada sistem IoT dirancang untuk mengoptimalkan komunikasi data serta meningkatkan efisiensi transmisi antar-perangkat (Asvial et al., 2021). Selain komunikasi IoT lokal, penelitian ini juga menerapkan konsep edge computing. Dalam

penerapannya, ESP32 tidak hanya bertugas menerima data dari aplikasi, tetapi juga melakukan pengolahan data secara langsung pada perangkat. Data yang diterima digunakan untuk menentukan isi pesan, posisi tulisan, kecepatan pergerakan, mode tampilan, serta pengendalian panel LED. Oleh karena itu, proses pengolahan utama tidak memerlukan server eksternal.

Pendekatan edge computing memungkinkan pengolahan data dilakukan dekat dengan sumber data sehingga dapat mengurangi ketergantungan terhadap server eksternal dan meningkatkan respons sistem (Jebamani & Winster, 2022). Setelah perintah diterima, ESP32 dapat melakukan pengolahan data dan pengendalian tampilan secara lokal. Sistem yang dirancang dalam penelitian ini terdiri atas dua panel LED P10 RGB yang masing-masing dikendalikan oleh satu ESP32 sebagai edge controller. Masing-masing panel dirancang untuk menampilkan informasi yang berbeda sehingga pengguna memiliki fleksibilitas dalam mengatur isi tampilan. Pengaturan pesan, mode tampilan, dan kecepatan dapat dilakukan secara terpisah untuk setiap panel melalui satu aplikasi mobile. Sementara itu, pengaturan tingkat kecerahan dan daya digunakan untuk mengendalikan sistem secara keseluruhan.

Untuk memudahkan proses pengendalian, aplikasi mobile dikembangkan menggunakan framework Flutter. Untuk memudahkan proses pengendalian, aplikasi mobile dikembangkan menggunakan framework Flutter. Framework Flutter memberikan fleksibilitas tinggi dalam pembuatan aplikasi pengendali IoT yang responsif dan multi-platform (Mulyani et al., 2024). Penggunaan aplikasi mobile sebagai media pengendali memungkinkan pengguna melakukan perubahan informasi pada running text melalui perangkat smartphone (Triwahyuni & Beta, 2022). Aplikasi tersebut menyediakan antarmuka yang memungkinkan pengguna memasukkan pesan, memilih mode tampilan seperti berjalan, diam, atau berkedip, serta mengatur kecepatan dan tingkat kecerahan panel. Setelah pengaturan ditentukan, data dikirimkan dari aplikasi menuju ESP32 melalui jaringan WiFi lokal. ESP32 kemudian memproses data tersebut dan menerapkannya pada panel LED sesuai dengan perintah yang diterima. Penggunaan ESP32 sebagai pengontrol informasi running text menunjukkan keandalan tinggi dalam pemrosesan data dan sinkronisasi tampilan secara real-time (Ramadanti et al., 2026). Berdasarkan uraian tersebut, penelitian ini dilakukan dengan judul "Rancang Bangun Sistem Running Text Dua Panel Berbasis Internet Of Things (IoT) Menggunakan Aplikasi Mobile". Penelitian ini bertujuan merancang dan membangun sistem running text dua panel yang dapat dikendalikan secara nirkabel melalui satu aplikasi mobile menggunakan jaringan WiFi lokal. Masing-masing panel dikendalikan oleh satu ESP32 sebagai edge device sehingga dapat menerima konfigurasi dan melakukan pemrosesan tampilan secara independen.

METODE

Penelitian ini menggunakan metode rancang bangun dengan pendekatan *Prototyping*. Metode tersebut dipilih karena pengembangan sistem dilakukan melalui beberapa tahapan, mulai dari perancangan perangkat keras, pembuatan program, pengujian fungsi, hingga penyempurnaan sistem berdasarkan hasil pengujian. Sistem yang dikembangkan terdiri dari aplikasi *mobile*, ESP32 DevKit V1, dua panel LED P10 RGB, serta sumber daya sebagai komponen utama.

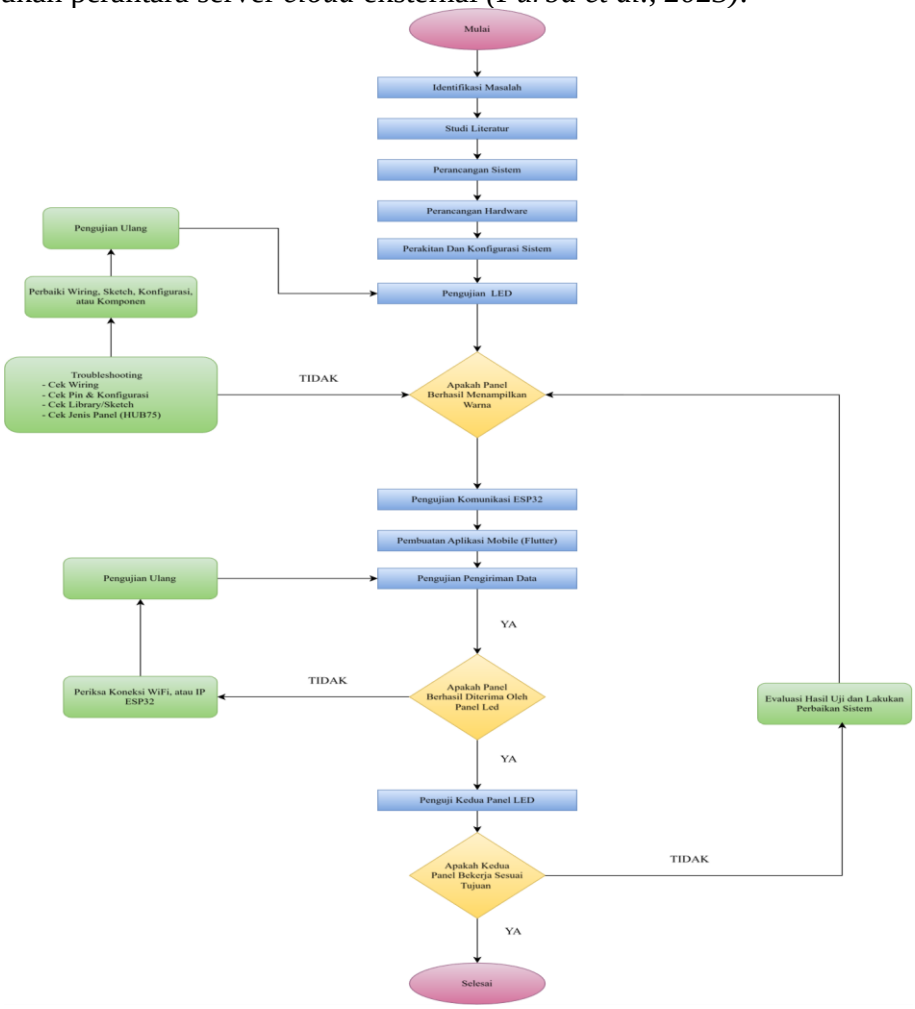
1. Tahapan Penelitian

   Proses penelitian dilakukan secara bertahap agar setiap bagian sistem dapat diuji sebelum seluruh perangkat diintegrasikan. Tahapan penelitian meliputi identifikasi kebutuhan, perancangan sistem, perakitan perangkat keras, pengembangan perangkat lunak, integrasi sistem, serta pengujian. Tahap identifikasi kebutuhan dilakukan dengan menentukan fungsi utama sistem, yaitu menampilkan informasi melalui dua panel LED P10 yang dapat dikendalikan menggunakan aplikasi *mobile*. Setiap panel dirancang agar dapat menampilkan pesan yang berbeda serta memiliki pengaturan mode dan kecepatan teks secara terpisah.

Tahap berikutnya adalah perancangan perangkat keras dengan menentukan hubungan antara ESP32 dan panel LED melalui antarmuka HUB75. Setelah rangkaian perangkat keras dapat bekerja dengan baik, dilakukan pembuatan program ESP32 untuk mengolah data dari aplikasi dan mengendalikan tampilan LED. Aplikasi *mobile* kemudian dikembangkan menggunakan *Flutter* sebagai antarmuka pengguna. Setelah bagian perangkat keras dan perangkat lunak selesai, seluruh komponen diintegrasikan dan diuji untuk mengetahui kesesuaian fungsi sistem dengan rancangan yang telah dibuat.

2. Arsitektur Sistem

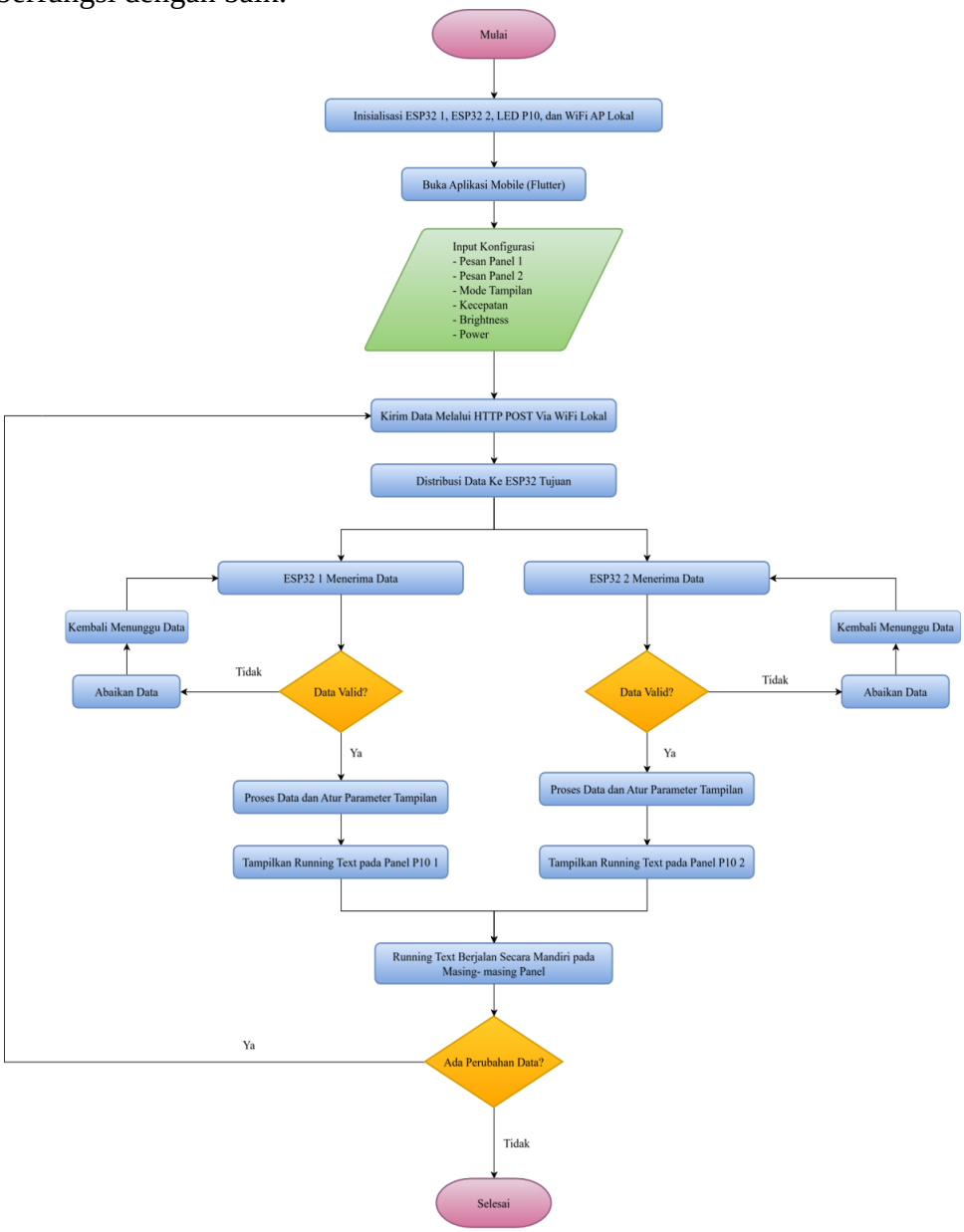
Sistem dirancang menggunakan komunikasi nirkabel dalam satu jaringan lokal yang sama. ESP32 Panel 1 dikonfigurasi pada mode *Access Point* (AP) untuk memancarkan sinyal *WiFi* sekaligus menjalankan *Web server*. Sementara itu, ESP32 Panel 2 dikonfigurasi dalam mode *Station* (STA) yang terhubung ke jaringan *WiFi* ESP32 Panel 1 dan menjalankan *Web server*-nya sendiri. Smartphone pengguna kemudian dihubungkan ke jaringan *WiFi* yang dipancarkan oleh ESP32 Panel 1 sehingga aplikasi dapat mengirimkan konfigurasi ke kedua ESP32. Komunikasi berbasis Web Server pada mikrokontroler memungkinkan penerimaan perintah HTTP POST secara langsung di jaringan lokal tanpa memerlukan perantara server *cloud* eksternal *(Purba et al., 2023)*.



Gambar 1. Flowchart Sistem

Berdasarkan Flowchart sistem Gambar 1, menunjukkan alur penelitian yang dilakukan dalam proses pengembangan sistem running text berbasis ESP32. Tahapan penelitian diawali dengan identifikasi masalah dan studi literatur untuk memperoleh dasar teori serta referensi yang berkaitan dengan sistem yang akan dibuat. Selanjutnya dilakukan perancangan sistem yang mencakup perancangan perangkat keras dan perangkat lunak.

Setelah rancangan selesai, komponen dirakit dan dikonfigurasi sesuai kebutuhan sistem. Pengujian awal dilakukan pada panel LED untuk memastikan panel dapat berfungsi dan menampilkan data dengan baik. Apabila hasil pengujian belum sesuai, dilakukan pemeriksaan dan perbaikan pada bagian *Wiring*, konfigurasi pin, maupun jenis konfigurasi panel HUB75, kemudian pengujian diulangi. Setelah panel dapat bekerja dengan baik, tahap berikutnya adalah pengujian komunikasi ESP32 dan pembuatan aplikasi *mobile* menggunakan *Flutter*. Aplikasi tersebut digunakan untuk mengirimkan data yang akan ditampilkan pada panel LED. Pengujian pengiriman data dilakukan untuk memastikan data dari aplikasi dapat diterima oleh ESP32. Jika data belum diterima dengan baik, dilakukan pemeriksaan terhadap koneksi *WiFi* dan alamat IP ESP32 sebelum pengujian diulang. Setelah komunikasi berhasil, pengujian dilanjutkan pada dua panel LED untuk mengetahui kemampuan sistem dalam menampilkan data secara bersamaan. Hasil pengujian kemudian dievaluasi untuk memastikan seluruh bagian sistem telah bekerja sesuai dengan rancangan. Jika masih ditemukan kendala, dilakukan perbaikan dan pengujian kembali hingga sistem dapat berfungsi dengan baik.



Gambar 2. Flowchart Cara Kerja Sistem

Berdasarkan flowchart cara kerja sistem pada Gambar 2, proses kerja sistem diawali dengan inisialisasi ESP32 1, ESP32 2, panel LED P10, dan jaringan *WiFi* lokal. Setelah

perangkat terhubung, pengguna membuka aplikasi *mobile* berbasis *Flutter* yang digunakan untuk mengatur tampilan pada panel LED. Pada aplikasi, pengguna dapat memasukkan konfigurasi berupa pesan untuk Panel 1 dan Panel 2, mode tampilan, kecepatan teks, *brightnes*s, serta power. Data yang telah dimasukkan kemudian dikirim melalui metode *HTTP POST* menggunakan jaringan *WiFi* lokal dan diteruskan ke ESP32 sesuai dengan panel yang dituju.

Setelah menerima data, masing-masing ESP32 melakukan pemeriksaan untuk menentukan apakah data yang diterima valid. Jika data tidak valid, maka data akan diabaikan. Jika data valid, ESP32 memproses data dan mengatur parameter tampilan pada panel yang terhubung. ESP32 Panel 1 digunakan untuk mengendalikan Panel LED P10 1, sedangkan ESP32 Panel 2 digunakan untuk mengendalikan Panel LED P10 2. Setelah proses pengolahan selesai, masing-masing panel menampilkan *running text* sesuai dengan konfigurasi yang diberikan melalui aplikasi. Sistem kemudian melakukan pengecekan terhadap perubahan data. Apabila terdapat konfigurasi baru, data akan kembali diproses dan diterapkan pada panel. Jika tidak terdapat perubahan, *running text* tetap berjalan secara mandiri pada masing-masing panel sesuai dengan pengaturan terakhir yang diterima.

3. Perancangan Perangkat Keras

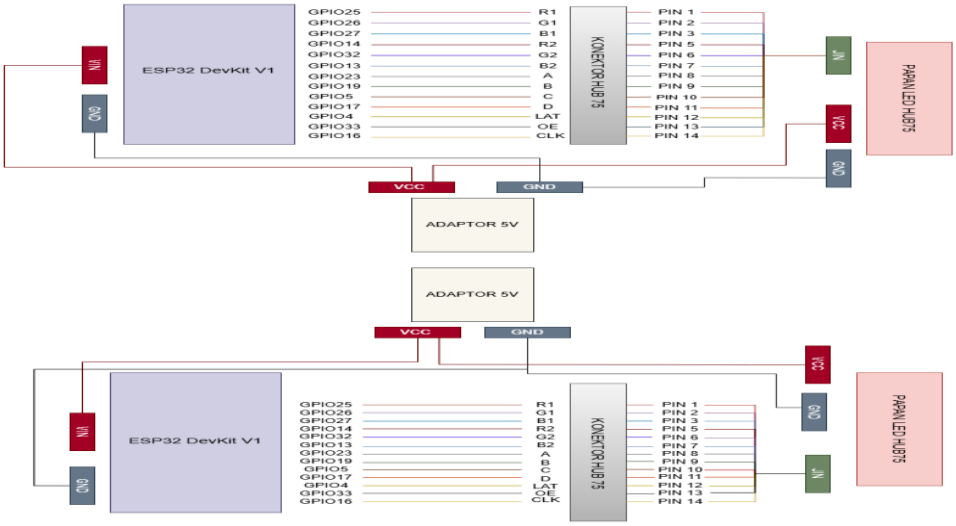
Perangkat keras yang digunakan terdiri atas dua ESP32 DevKit V1 sebagai *edge controller* yang masing-masing berfungsi untuk mengendalikan satu panel LED P10 RGB, dua panel LED P10 RGB sebagai media tampilan, serta *Power supply* 5 V sebagai sumber daya sistem. Panel LED P10 menggunakan antarmuka HUB75 yang terhubung dengan pin GPIO pada ESP32.

Panel LED yang digunakan memiliki resolusi 32 × 16 piksel dengan sistem pemindaian 1/4 *Scan*. Kedua panel ditempatkan dalam jarak berdekatan, namun masing-masing dikendalikan oleh satu ESP32 sebagai *edge controller*. Konfigurasi ini memungkinkan setiap panel melakukan proses pengolahan data secara independen. Dalam penerapannya, setiap panel memiliki area tampilan tersendiri sehingga dapat menampilkan informasi yang berbeda sesuai dengan konfigurasi yang diberikan melalui aplikasi *mobile*.

Tabel 1. Konfigurasi Pin HUB75 dan ESP32

SINYAL HUB75	GPIO ESP32 DEV KIT V1	Fungsi
R1	GPIO25	Data warna merah 1
G1	GPIO26	Data warna hijau 1
B1	GPIO27	Data warna biru 1
R2	GPIO14	Data warna merah 2
G2	GPIO32	Data warna hijau 2
B2	GPIO13	Data warna biru 2
A	GPIO23	Pemilihan baris LED
B	GPIO19	Pemilihan baris LED
C	GPIO5	Pemilihan baris LED
D	GPIO17 (TX2)	Pemilihan baris LED
LAT	GPIO4	Penguncian data
OE	GPIO33	Pengaktifan output LED
CLK	GPIO16 (RX2)	Sinyal *Clock*

Berdasarkan Tabel 1 Konfigurasi Pin HUB75 dan ESP32, koneksi antara panel LED HUB75 dan ESP32 DevKit V1 terdiri atas jalur data warna RGB, jalur pemilihan baris, serta jalur pengendalian tampilan. Jalur R1, G1, B1, R2, G2, dan B2 digunakan untuk mengirimkan data warna RGB, sedangkan jalur A, B, C, dan D digunakan untuk pemilihan baris LED. Sementara itu, LAT berfungsi untuk mengunci data, OE untuk mengaktifkan atau menonaktifkan output LED, dan CLK sebagai sinyal *Clock* dalam proses pengiriman data. Ground dan VCC pada ESP32 dan panel LED dihubungkan pada referensi yang sama. Panel memperoleh daya dari *Power supply* 5 V dengan kapasitas arus yang disesuaikan dengan kebutuhan dua modul LED.



Gambar 3. Skematik

Berdasarkan Gambar 3, skematik sistem menunjukkan hubungan antara dua ESP32 DevKit V1 dengan masing-masing panel LED P10 RGB melalui konektor HUB75. Setiap ESP32 berfungsi sebagai *edge controller* yang mengendalikan satu panel LED secara independen. Jalur data warna R1, G1, B1, R2, G2, dan B2 masing-masing dihubungkan ke GPIO25, GPIO26, GPIO27, GPIO14, GPIO32, dan GPIO13 pada ESP32. Jalur pemilihan baris A, B, C, dan D masing-masing terhubung ke GPIO23, GPIO19, GPIO5, dan GPIO17. Sementara itu, jalur LAT, OE, dan CLK masing-masing terhubung ke GPIO4, GPIO33, dan GPIO16.

Pada bagian catu daya, setiap ESP32 dan panel LED menggunakan adaptor 5 V sebagai sumber daya. Tegangan 5 V untuk ESP32 dihubungkan melalui pin VIN, sedangkan panel LED memperoleh tegangan melalui pin VCC. Pin GND pada ESP32 dan panel LED dihubungkan dengan ground yang sama pada masing-masing rangkaian. Dengan konfigurasi tersebut, ESP32 dapat mengirimkan data dan sinyal pengendalian melalui antarmuka HUB75, sedangkan panel LED menerima data tersebut untuk menampilkan informasi sesuai dengan konfigurasi yang diberikan melalui aplikasi *mobile*.

4. Perancangan Perangkat Lunak

Program pada ESP32 dikembangkan menggunakan Arduino IDE. Pengendalian panel LED dilakukan menggunakan *library* ESP32-HUB75-MatrixPanel-I2S-DMA dan ESP32-HUB75-VirtualMatrixPanel_T**.** Karakteristik panel 1/4 *Scan* memerlukan penyesuaian pemetaan piksel agar susunan baris dan warna dapat ditampilkan sesuai dengan posisi fisik LED. Oleh karena itu, sistem menggunakan *virtual mapping* untuk menyesuaikan keluaran data dari ESP32 dengan susunan panel.

Program ESP32 dirancang untuk menerima beberapa parameter dari aplikasi, yaitu teks, mode tampilan, kecepatan, tingkat kecerahan, dan status daya. Setelah data diterima, ESP32 memproses parameter tersebut dan memperbarui tampilan tanpa melakukan proses unggah program ulang. Mode tampilan yang digunakan terdiri atas tiga jenis, yaitu berjalan**,** diam**,** dan berkedip. Pada mode berjalan, posisi teks diubah secara berkala sehingga menghasilkan efek *running text*. Mode diam mempertahankan posisi teks pada area panel, sedangkan mode berkedip mengatur kondisi tampil dan tidak tampil berdasarkan interval waktu tertentu.

5. Perancangan Aplikasi *Mobile*

Aplikasi *mobile* dikembangkan menggunakan framework *Flutter*. Aplikasi berfungsi sebagai antarmuka utama untuk mengatur informasi yang ditampilkan pada kedua panel. Pada aplikasi tersedia dua bagian kontrol yang mewakili Panel 1 dan Panel 2. Masing-masing bagian menyediakan masukan pesan, pilihan mode tampilan, dan pengaturan kecepatan. Dengan rancangan tersebut, pengguna dapat memberikan konfigurasi yang

berbeda pada setiap panel. Selain pengaturan masing-masing panel, aplikasi menyediakan pengaturan umum berupa tingkat kecerahan dan status daya LED. Aplikasi juga menampilkan informasi status koneksi ESP32 sehingga pengguna dapat mengetahui apakah perangkat sudah terhubung dengan sistem.

6. Komunikasi Data

Pertukaran data antara aplikasi *mobile* dan ESP32 dilakukan melalui jaringan *WiFi* lokal berbasis topologi *Access Point* (AP) - *Station* (STA). ESP32 Panel 1 bertindak sebagai penyedia jaringan (*Access Point*) dengan alamat IP lokal 192.168.4.1, sedangkan ESP32 Panel 2 bertindak sebagai klien (*Station*) yang terhubung ke ESP32 Panel 1 dan mendapatkan alamat IP statis 192.168.4.20. Smartphone bertindak sebagai perangkat pengguna yang terhubung ke SSID ESP32 Panel 1, sehingga aplikasi *mobile* dapat mendistribusikan data konfigurasi ke masing-masing alamat IP tujuan sesuai dengan panel yang ingin dikendalikan.

Format data dikirim menggunakan struktur JSON yang berisi parameter sesuai kebutuhan masing-masing panel. Parameter yang dikirim meliputi pesan, mode tampilan, kecepatan pergerakan teks, tingkat kecerahan, dan status daya.

Tabel 2. Format Data JSON untuk Pengendalian ESP32

Data untuk ESP32 Panel 1	Data untuk ESP32 Panel 2
{	{
"text": "TEKNIK",	"text": "KOMPUTER",
"mode": "move",	"mode": "move",
"speed": 5,	"speed": 5,
"brightness": 5,	"brightness": 5,
"power": true	"power": true
}	}

Setelah data diterima, masing-masing ESP32 melakukan proses validasi dan pengolahan parameter sesuai dengan fungsi yang diberikan. ESP32 Panel 1 hanya memproses data untuk Panel LED P10 1, sedangkan ESP32 Panel 2 hanya memproses data untuk Panel LED P10 2. Hasil pengolahan tersebut kemudian diterapkan pada panel LED yang terhubung sehingga kedua panel dapat bekerja secara independen.

7. Penerapan *Edge Computing*

Pendekatan *edge computing* pada sistem diterapkan dengan menjadikan ESP32 sebagai perangkat yang melakukan pemrosesan langsung di sisi perangkat. Aplikasi *mobile* berfungsi untuk memberikan data dan perintah, sedangkan proses pengendalian tampilan dilakukan oleh ESP32. Penerapan *edge computing* pada perangkat IoT lokal memungkinkan distribusi data yang stabil, terisolasi dari gangguan jaringan publik, serta mengurangi ketergantungan pada server pusat *(Cao et al., 2020; Jebamani & Winster, 2022)*. Pemrosesan lokal tersebut meliputi pembacaan pesan, penentuan mode tampilan, pengaturan kecepatan pergerakan teks, pengaturan tingkat kecerahan, perhitungan posisi karakter, dan pengiriman data piksel menuju panel LED.

Dengan mekanisme tersebut, aplikasi tidak perlu mengirimkan data untuk setiap perubahan posisi piksel. Aplikasi hanya mengirimkan konfigurasi, sedangkan proses animasi selanjutnya dijalankan secara mandiri oleh ESP32. Pendekatan ini mengurangi pertukaran data yang tidak diperlukan dan membuat sistem tidak bergantung pada server eksternal untuk menjalankan fungsi utama.

8. Pengujian Sistem

Pengujian dilakukan untuk mengetahui apakah seluruh fungsi sistem dapat bekerja sesuai dengan perancangan. Pengujian dilakukan secara bertahap pada perangkat keras, komunikasi, dan tampilan panel.

Pengujian yang dilakukan meliputi:

1. Pengujian koneksi aplikasi *mobile* dengan ESP32.
2. Pengujian pengiriman dan perubahan pesan pada Panel 1.
3. Pengujian pengiriman dan perubahan pesan pada Panel 2.
4. Pengujian mode berjalan, diam, dan berkedip.
5. Pengujian pengaturan kecepatan teks.
6. Pengujian tingkat kecerahan panel.
7. Pengujian fungsi daya LED.
8. Pengujian dua panel dengan pesan dan mode tampilan yang berbeda.

Setiap pengujian dilakukan dengan membandingkan perintah yang diberikan melalui aplikasi dengan keluaran yang ditampilkan pada panel LED. Sistem dinyatakan berhasil apabila panel memberikan respons sesuai dengan konfigurasi yang dikirimkan oleh pengguna.

HASIL DAN PEMBAHASAN

Hasil penelitian diperoleh melalui proses implementasi perangkat keras, pengembangan aplikasi *mobile*, konfigurasi komunikasi jaringan lokal, serta pengujian fungsi sistem *running text*.

1. Hasil Implementasi Perangkat Keras

Implementasi perangkat keras terdiri atas dua ESP32 DevKit V1 sebagai *edge controller* yang masing-masing mengendalikan satu panel LED P10 RGB. Panel LED yang digunakan memiliki resolusi 32 × 16 piksel sebagai media tampilan informasi. Kedua panel menggunakan antarmuka HUB75 dengan karakteristik pemindaian 1/4 *Scan*. Pada tahap awal pengujian ditemukan beberapa ketidaksesuaian tampilan berupa warna yang tidak sesuai dan susunan piksel yang belum terbentuk dengan benar. Permasalahan tersebut disebabkan oleh konfigurasi pin GPIO dan pemetaan piksel yang belum sesuai dengan karakteristik panel LED P10. Setelah dilakukan penyesuaian konfigurasi GPIO serta penerapan *virtual mapping* untuk panel dengan sistem 1/4 *Scan*, panel mampu menampilkan teks dan warna dengan susunan piksel yang sesuai.

Penggunaan *virtual mapping* diperlukan karena struktur pemindaian panel LED P10 tidak dapat diperlakukan sama dengan panel HUB75 standar. ESP32 melakukan penyesuaian koordinat piksel sebelum data dikirimkan menuju panel LED. Hasil tersebut menunjukkan bahwa kesesuaian konfigurasi perangkat lunak terhadap karakteristik fisik panel merupakan faktor penting dalam keberhasilan pengendalian LED P10. Pada pengembangan sistem dua panel, kedua panel dirancang untuk ditempatkan berdekatan dengan masing-masing panel dikendalikan oleh satu ESP32 sebagai *edge controller*. Konfigurasi ini memungkinkan proses pengolahan data dilakukan secara independen pada setiap perangkat. Panel pertama dan panel kedua dapat menerima konfigurasi yang berbeda melalui satu aplikasi *mobile* sehingga masing-masing mampu menampilkan informasi sesuai dengan konfigurasi yang diberikan.



Gambar 4. **Tampilan Panel 1**



Gambar 5. Tampilan Panel 2

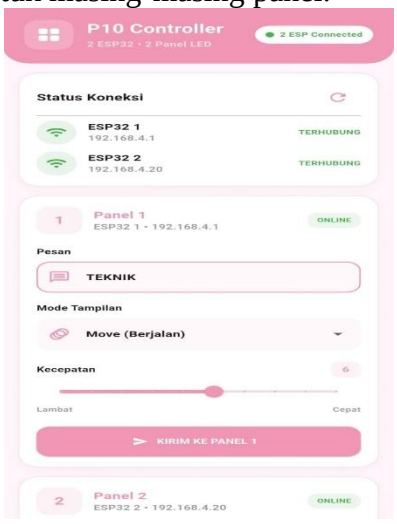
Berdasarkan Gambar 4, ditunjukkan hasil tampilan pada Panel 1 dengan teks "TEKNIK", sedangkan Gambar 5 menunjukkan hasil tampilan pada Panel 2 dengan teks "KOMPUTER". Setiap panel memiliki kapasitas tampilan sebanyak 4 karakter dalam satu waktu. Pada mode diam dan berkedip, panel hanya dapat menampilkan 4 karakter yang terlihat pada area panel, sedangkan teks yang memiliki lebih dari 4 karakter perlu menggunakan mode *running text* agar seluruh karakter dapat ditampilkan secara bergantian. Hasil pengujian menunjukkan bahwa masing-masing panel dapat menerima data teks dari aplikasi *mobile* melalui ESP32 dan menampilkan teks sesuai dengan perintah yang dikirimkan.

2. Hasil Implementasi Aplikasi *Mobile*

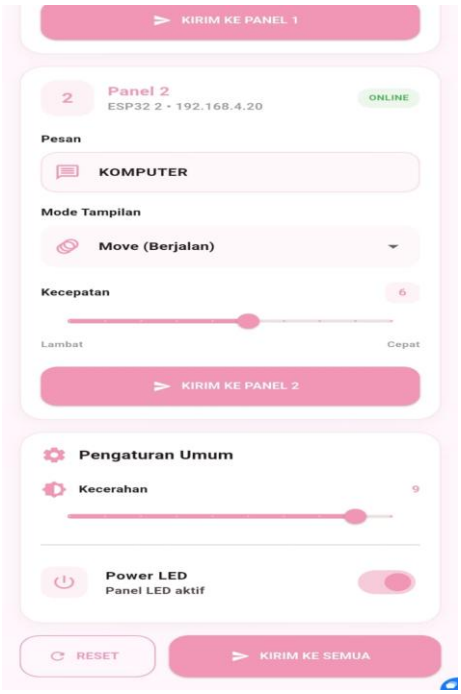
Aplikasi pengendali dikembangkan menggunakan *framework Flutter*. Antarmuka aplikasi dirancang agar pengguna dapat mengendalikan dua panel melalui satu aplikasi. Aplikasi menyediakan bagian kontrol yang terpisah untuk Panel 1 dan Panel 2. Setiap panel memiliki kolom masukan pesan, pilihan mode tampilan, dan pengaturan kecepatan. Sementara itu, tingkat kecerahan dan daya LED ditempatkan pada bagian pengaturan umum.

Mode tampilan yang tersedia meliputi mode berjalan, diam, dan berkedip. Pemisahan kontrol pada masing-masing panel memungkinkan pengguna memberikan konfigurasi yang berbeda tanpa harus menggunakan aplikasi atau perangkat pengendali tambahan. Aplikasi juga menyediakan indikator status koneksi ESP32. Informasi tersebut digunakan untuk membantu pengguna mengetahui apakah aplikasi telah berhasil berkomunikasi dengan perangkat.

Hasil pengembangan antarmuka menunjukkan bahwa seluruh komponen kontrol dapat ditampilkan dan dioperasikan pada lingkungan pengembangan *Flutter*. *Slider* kecepatan dan kecerahan dapat diubah, pilihan mode dapat dipilih, serta masukan pesan dapat diberikan secara terpisah untuk masing-masing panel.



Gambar 6. Tampilan Aplikasi Flutter



Gambar 7. Tampilan Aplikasi Flutter

Berdasarkan Gambar 6 dan gambar 7 menunjukkan tampilan aplikasi *mobile* yang digunakan sebagai antarmuka pengendalian sistem *running text*. Aplikasi menyediakan pengaturan untuk Panel 1 dan Panel 2 berupa input pesan, pemilihan mode tampilan, serta pengaturan kecepatan tampilan teks. Selain itu, tersedia pengaturan kecerahan dan kontrol daya untuk mengatur kondisi tampilan LED. Aplikasi juga menampilkan status koneksi sebagai informasi mengenai keterhubungan aplikasi dengan perangkat ESP32. Setiap pengaturan yang diberikan melalui aplikasi selanjutnya dikirimkan ke ESP32 untuk diproses dan diterapkan pada panel LED sesuai konfigurasi yang dipilih.

Tabel 3. Hasil Pengujian Fungsional Aplikasi *Mobile*

No.	Fungsi Yang Diuji	Kondisi Pengujian	Hasil Yang Diharapkan	Hasil Aktual	Status
1.	Input pesan Panel 1	Memasukkan teks	Teks dapat dimasukkan	Berhasil	Berhasil
2.	Input pesan Panel 2	Memasukkan teks	Teks dapat dimasukkan	Berhasil	Berhasil
3.	Pemilihan mode	Memilih mode tampilan	Mode dapat dipilih	Berhasil	Berhasil
4.	Pengaturan kecepatan	Mengubah nilai kecepatan	Nilai kecepatan berubah	Berhasil	Berhasil
5.	Pengaturan kecerahan	Mengubah nilai kecerahan	Nilai kecerahan berubah	Berhasil	Berhasil
6.	Kontrol daya	Menekan tombol ON/OFF	Status daya berubah	Berhasil	Berhasil
7.	Status koneksi	Menghubungkan aplikasi dengan ESP32	Status koneksi ditampilkan	Berhasil	Berhasil
8.	Pengaturan timer	Menentukan penggunaan timer atau tanpa timer	Panel dapat menyala sesuai durasi yang ditentukan pada mode timer dan tetap menyala pada mode tanpa timer	Berhasil	Berhasil
9.	Penyimpanan data terakhir	Memutuskan koneksi aplikasi dengan	Data terakhir tetap tersimpan pada ESP32	Berhasil	Berhasil

	ESP32 setelah data berhasil dikirim	dan tetap ditampilkan pada panel	

3. Hasil Pengujian Komunikasi IoT Lokal

Komunikasi antara aplikasi *mobile* dan ESP32 dilakukan menggunakan jaringan *WiFi* lokal. ESP32 Panel 1 berfungsi sebagai *Access Point* (AP) sekaligus *Web server*, sedangkan ESP32 Panel 2 berfungsi sebagai *Station* (STA) yang terhubung ke jaringan *WiFi* ESP32 Panel 1 dan menjalankan *Web server* untuk menerima konfigurasi dari aplikasi *mobile*. ESP32 Panel 1 menggunakan alamat IP lokal 192.168.4.1, sedangkan ESP32 Panel 2 menggunakan alamat IP lokal 192.168.4.20. Pengiriman data konfigurasi dilakukan menggunakan metode *HTTP POST* menuju *Endpoint* /data.

Data konfigurasi dikirim menggunakan format JSON yang berisi parameter sesuai dengan panel tujuan, seperti pesan, mode tampilan, kecepatan pergerakan teks, tingkat kecerahan, dan status daya. Aplikasi *mobile* menentukan tujuan pengiriman berdasarkan alamat IP ESP32 yang terhubung dengan panel LED P10 yang akan dikendalikan. Pengujian komunikasi dilakukan untuk mengetahui keberhasilan aplikasi dalam mengirimkan perintah kepada masing-masing ESP32. Sistem dinyatakan berhasil apabila data konfigurasi dapat diterima dengan baik dan diterapkan sesuai dengan panel yang dituju tanpa terjadi kehilangan maupun pertukaran data antar panel.

Tabel 4. Hasil Pengujian Komunikasi Jaringan dan Konektivitas Dua Panel

No.	Pengujian	Hasil yang Diharapkan	Hasil Aktual	Status
1	Koneksi aplikasi dengan ESP32	Aplikasi dapat terhubung ke ESP32	Aplikasi berhasil terhubung ke *WiFi* ESP32 dan indikator status koneksi menunjukkan “Terhubung”	Berhasil
2	Pengiriman data melalui /data	ESP32 menerima data JSON	ESP32 menerima data JSON secara utuh melalui *Endpoint* /data tanpa ada data yang hilang	Berhasil
3	Perubahan pesan Panel 1	Pesan Panel 1 berubah	Tampilan pada panel segera berganti menjadi teks baru sesuai dengan pesan yang dikirim melalui aplikasi	Berhasil
4	Perubahan pesan Panel 2	Pesan Panel 2 berubah	Tampilan Panel 2 berhasil memperbarui teks sesuai dengan data baru yang dikirim dari aplikasi tanpa terjadi kesalahan tampilan	Berhasil
5	Pengiriman konfigurasi kedua panel	Kedua panel merespons sesuai perintah	Kedua panel berhasil menerima konfigurasi melalui satu aplikasi dan menampilkan pesan sesuai pengaturan masing-masing tanpa terjadi pertukaran data antar panel	Berhasil

Berdasarkan pengujian tersebut, keberhasilan komunikasi dapat dinilai dari kesesuaian antara perintah yang dikirim melalui aplikasi dan respons yang diberikan oleh ESP32.

4. Hasil Pengujian Dua Panel

Pengujian dua panel dilakukan untuk mengetahui kemampuan dua ESP32 dalam mengendalikan masing-masing panel LED P10 secara independen. Pengujian ini bertujuan untuk memastikan bahwa setiap panel mampu menerima dan menjalankan konfigurasi yang berbeda melalui satu aplikasi *mobile* tanpa memengaruhi tampilan pada panel lainnya. Hasil

pengujian menunjukkan bahwa masing-masing ESP32 mampu menerima, mengolah, dan menerapkan data konfigurasi sesuai dengan parameter yang dikirimkan oleh aplikasi. Setiap ESP32 berperan sebagai *edge controller* yang melakukan proses pengolahan data secara lokal sebelum meneruskan keluaran tampilan menuju panel LED P10 yang terhubung.

Pada tahap pengujian, Panel 1 dan Panel 2 diberikan pesan serta parameter tampilan yang berbeda melalui aplikasi *mobile*. Sistem dinyatakan berhasil apabila masing-masing panel mampu mempertahankan tampilan sesuai konfigurasi yang diberikan, meliputi isi pesan, mode tampilan, dan kecepatan pergerakan teks, tanpa terjadi perpindahan data maupun pencampuran tampilan antar panel.

Tabel 5. Hasil Pengujian Independensi Parameter Tampilan Dua Panel

Pengujian	Panel 1	Panel 2	Hasil
Pesan berbeda	TEKNIK	KOMPUTER	Berhasil
Mode berbeda	Berjalan	Diam	Berhasil
Kecepatan berbeda	5	8	Berhasil
Mode berkedip	Berkedip	Diam	Berhasil
Kedua panel aktif bersamaan	Aktif	Aktif	Berhasil

Hasil pengujian ini menunjukkan bahwa sistem mampu memberikan kontrol independen terhadap dua panel LED P10, dengan masing-masing panel dikendalikan oleh satu ESP32 melalui satu aplikasi *mobile*.

5. Hasil Pengujian Mode Tampilan

Tiga mode tampilan diterapkan pada sistem, yaitu berjalan, diam, dan berkedip. Pengujian dilakukan pada masing-masing panel untuk mengetahui apakah perubahan mode dapat dilakukan melalui aplikasi. Pada mode berjalan, ESP32 memperbarui posisi horizontal teks secara berkala sehingga menghasilkan efek pergerakan. Pada mode diam, koordinat teks dipertahankan pada posisi tertentu. Sementara itu, mode berkedip dilakukan dengan mengatur kondisi tampil dan tidak tampil berdasarkan interval waktu.

Tabel 6. Hasil Pengujian Mode Tampilan Running Text

No.	Mode	Hasil yang Diharapkan	Hasil Aktual	Status
1	Berjalan	Teks bergerak secara horizontal	Teks bergeser secara horizontal dari kanan ke kiri secara terus-menerus pada area panel sesuai dengan pesan yang dimasukkan	Berhasil
2	Diam	Teks tidak bergerak	Teks muncul dan menetap pada koordinat statis di tengah panel tanpa ada pergerakan piksel sama sekali	Berhasil
3	Berkedip	Teks tampil dan menghilang berkala	Teks muncul dan menghilang (padam) secara bergantian	Berhasil

Pengolahan animasi dilakukan secara lokal oleh ESP32. Aplikasi hanya memberikan jenis mode yang dipilih, sedangkan perubahan posisi dan waktu tampilan diproses langsung pada mikrokontroler

6. Hasil Pengujian Kecepatan dan Kecerahan

Pengaturan kecepatan digunakan untuk mengubah laju pergerakan *running text*. Aplikasi memberikan nilai kecepatan dalam rentang 1 sampai 10. Nilai tersebut kemudian dikonversi oleh ESP32 menjadi interval pergeseran teks. Pengujian dilakukan menggunakan nilai kecepatan rendah, sedang, dan tinggi untuk mengetahui perubahan yang terjadi pada tampilan.

Tabel 7. Hasil Pengujian Kecepatan Pergerakan Running Text

Nilai Kecepatan	Kondisi Yang Diharapkan	Hasil Aktual
1	Pergerakan paling lambat	Teks bergerak dengan kecepatan rendah sesuai pengaturan
5	Pergerakan sedang	Teks bergerak dengan kecepatan sedang sesuai pengaturan
10	Pergerakan paling cepat	Teks bergerak dengan kecepatan tinggi sesuai pengaturan

Selain kecepatan, aplikasi menyediakan pengaturan tingkat kecerahan. Perubahan nilai kecerahan dikirim menuju ESP32 kemudian diterapkan pada driver panel LED.

Tabel 8. Hasil Pengujian Tingkat Kecerahan Running Text

Nilai Kecerahan	Kondisi Yang Diharapkan	Hasil Aktual
1	Cahaya rendah	Berhasil, intensitas cahaya LED berkurang sesuai pengaturan
5	Cahaya sedang	Berhasil, intensitas cahaya LED berada pada tingkat sedang
10	Cahaya tinggi	Berhasil, intensitas cahaya LED meningkat sesuai pengaturan

Pengaturan tersebut memberikan fleksibilitas kepada pengguna untuk menyesuaikan tampilan berdasarkan kondisi lingkungan dan kebutuhan penggunaan.

7. Hasil Pengujian Daya LED

Fungsi daya digunakan untuk mengaktifkan atau menonaktifkan tampilan panel melalui aplikasi. Saat kondisi daya aktif, panel menampilkan teks sesuai dengan konfigurasi yang telah diterima. Ketika daya dinonaktifkan, tampilan LED pada panel berhenti dan tidak menampilkan informasi. Penggunaan mikrokontroler ESP32 terbukti efektif dalam melakukan pemrosesan data serta manajemen daya secara stabil pada sistem berbasis IoT (Yasa et al., 2023). Pengujian dilakukan dengan mengubah status daya pada masing-masing panel melalui aplikasi dan mengamati respons yang dihasilkan.

Tabel 9. Hasil Pengujian Fungsi Power On dan Power Off Panel

Panel	Kondisi Perintah	Hasil Yang Diharapkan	Hasil Aktual
Panel 1	Perangkat dinyalakan	Panel 1 menampilkan informasi	Berhasil, Panel 1 menampilkan teks sesuai konfigurasi
Panel 1	Perangkat dimatikan	Tampilan Panel 1 mati	Berhasil, tampilan LED Panel 1 berhenti dan tidak menampilkan informasi
Panel 1	Perangkat dinyalakan kembali	Panel 1 kembali aktif	Berhasil, Panel 1 aktif kembali dan menjalankan tampilan terakhir
Panel 2	Perangkat dinyalakan	Panel 2 menampilkan informasi	Berhasil, Panel 2 menampilkan teks sesuai konfigurasi

Panel 2	Perangkat dimatikan	Tampilan Panel 2 mati	Berhasil, tampilan LED Panel 2 berhenti dan tidak menampilkan informasi
Panel 2	Perangkat dinyalakan kembali	Panel 2 kembali aktif	Berhasil, Panel 2 aktif kembali dan menjalankan tampilan terakhir

Berdasarkan hasil pengujian, fungsi power pada masing-masing panel dapat berjalan sesuai dengan perintah yang diberikan. Ketika perangkat dinyalakan, panel menampilkan teks sesuai konfigurasi. Saat perangkat dimatikan, tampilan LED berhenti dan panel tidak menampilkan informasi. Setelah perangkat dinyalakan kembali, panel dapat aktif dan menjalankan tampilan terakhir yang tersimpan.

8. Hasil Pengujian Penerapan *Edge Computing*

Penerapan *edge computing* pada sistem terlihat dari proses pengolahan data yang dilakukan langsung oleh ESP32. Aplikasi *mobile* tidak mengirimkan data piksel secara terus-menerus, tetapi hanya mengirimkan parameter yang dibutuhkan oleh sistem. Sebagai contoh, ketika pengguna mengirimkan pesan dengan mode berjalan, aplikasi hanya mengirim isi pesan, mode, dan nilai kecepatan. Setelah konfigurasi diterima, ESP32 secara mandiri memproses isi pesan, menghitung posisi karakter, mengatur perubahan tampilan, dan memperbarui panel LED secara terus-menerus tanpa memerlukan pengiriman data piksel dari aplikasi pada setiap perubahan tampilan. Dengan demikian, proses animasi tidak bergantung pada komunikasi antara aplikasi dan ESP32 pada setiap perubahan piksel.

Karakteristik tersebut dapat dibuktikan melalui pengujian dengan memutus koneksi aplikasi setelah konfigurasi berhasil dikirim. Apabila *running text* tetap berjalan pada panel, maka proses utama telah berlangsung secara lokal pada perangkat *edge*.

Tabel 10. Hasil Pengujian Kestabilan Sistem terhadap Gangguan Koneksi

Kondisi Pengujian	Hasil Yang Diharapkan	Hasil Aktual
Aplikasi terhubung	Panel menerima konfigurasi	Berhasil, panel menerima data dari aplikasi
Konfigurasi berhasil diterima	*Running text* berjalan	Berhasil, teks tampil sesuai konfigurasi
Aplikasi terputus dari jaringan *WiFi* lokal	*Running text* tetap berjalan	Berhasil, tampilan tetap berjalan meskipun koneksi aplikasi terputus
Aplikasi terhubung kembali	Konfigurasi baru dapat dikirim	Berhasil, panel dapat menerima konfigurasi baru

Mekanisme tersebut menunjukkan bahwa ESP32 berperan lebih dari sekadar sebagai penghubung komunikasi. ESP32 bertindak sebagai *edge device* karena melakukan pengolahan informasi dan pengendalian perangkat output secara lokal. Pendekatan ini juga mengurangi kebutuhan pertukaran data secara terus-menerus dengan aplikasi.

Tabel 11. Hasil Pengujian Keberhasilan Pengiriman Data

No.	Percobaan	Panel 1	Panel 2
1.	1	Berhasil	Berhasil
2.	2	Berhasil	Berhasil
3.	3	Berhasil	Berhasil
4.	4	Berhasil	Berhasil
5.	5	Berhasil	Berhasil
Panel	**Berhasil**	**Gagal**	**Persentase**

1	5	0	100 %
2	5	0	100 %

Berdasarkan hasil pengujian pada Tabel 11, pengiriman data dari aplikasi ke Panel 1 dan Panel 2 dilakukan sebanyak lima kali percobaan. Seluruh percobaan menunjukkan bahwa data berhasil diterima dan ditampilkan pada kedua panel. Panel 1 memperoleh lima keberhasilan dari lima percobaan tanpa adanya kegagalan, sedangkan Panel 2 juga memperoleh lima keberhasilan tanpa kegagalan. Berdasarkan hasil tersebut, tingkat keberhasilan pengiriman data pada masing-masing panel mencapai 100% dari lima kali percobaan. Hasil pengujian menunjukkan bahwa sistem mampu melakukan pengiriman data dari aplikasi ke kedua panel secara konsisten sesuai dengan data yang diberikan.

Tabel 12. Hasil Pengujian Sistem Berdasarkan Jarak

No.	Jarak	Percobaan	Berhasil	Gagal	Persentase
1	5 m	5	5	0	100%
2	10 m	5	5	0	100%
3	15 m	5	5	0	100%
4	>15 m	5	0	5	0%
Rata-rata					75%

Berdasarkan hasil pengujian pada Tabel 12, sistem berhasil melakukan komunikasi pada jarak 5 meter, 10 meter, dan 15 meter dengan tingkat keberhasilan masing-masing sebesar 100%. Pada jarak lebih dari 15 meter, koneksi terputus sehingga seluruh percobaan mengalami kegagalan dengan persentase keberhasilan sebesar 0%. Dari seluruh pengujian diperoleh rata-rata tingkat keberhasilan sebesar 75%. Dengan demikian, pada kondisi pengujian yang dilakukan, sistem dapat berkomunikasi dengan baik hingga jarak 15 meter, sedangkan pada jarak lebih dari 15 meter koneksi tidak dapat dipertahankan.

Tabel 13. Hasil Pengukuran Waktu Respons Sistem

No.	Percobaan	Waktu Respons
1.	1	18 detik
2.	2	17 detik
3.	3	17 detik
4.	4	17 detik
5.	5	15 detik
Rata-rata		16,8 detik

Berdasarkan hasil pengukuran pada Tabel 13, waktu respons sistem diuji sebanyak lima kali untuk mengetahui waktu yang dibutuhkan sistem dalam merespons perintah yang diberikan melalui aplikasi. Hasil pengujian menunjukkan waktu respons berturut-turut sebesar 18 detik, 17 detik, 17 detik, 17 detik, dan 15 detik. Berdasarkan hasil tersebut, diperoleh rata-rata waktu respons sistem sebesar 16,8 detik. Hasil pengukuran menunjukkan bahwa sistem mampu memberikan respons terhadap perintah dari aplikasi dengan waktu respons yang relatif konsisten.

Tabel 14. Hasi Pengujian Timer

No.	panel	Waktu mulai	Waktu selesai	Hasil pengujian	Keterangan
1	Panel 1	14.00 WIB	14.30 WIB	Berhasil menyala sesuai jadwal	Teks tampil pada pukul 14.00 dan berhenti pada pukul 14.30

2	Panel 2	14.30 WIB	15.00 WIB	Berhasil menyala sesuai jadwal	Teks tampil pada pukul 14.30 dan berhenti pada pukul 15.00

Berdasarkan hasil pengujian, fungsi power pada masing-masing panel dapat berjalan sesuai dengan perintah yang diberikan. Ketika perangkat dinyalakan, panel menampilkan teks sesuai konfigurasi. Saat perangkat dimatikan, tampilan LED berhenti dan panel tidak menampilkan informasi. Setelah perangkat dinyalakan kembali, panel dapat aktif dan menjalankan tampilan terakhir yang tersimpan.

Tabel 145. State Of Art

No	Peneliti	Judul	Teknologi	Hasil	Perbedaan & Kebaruan Penelitian Ini
1	Yuhanas et al. (2021)	Rancang Bangun Running Text Menggunakan Modul LED Matrix P10 Berbasis Arduino Uno di Fakultas Teknik Universitas PGRI Banyuwangi	Arduino Uno ATmega328, ESP8266, RTC DS3231, Android, Wi-Fi, LED Matrix P10 96×32 cm	Sistem memungkinkan perubahan informasi *running text* melalui aplikasi Android dengan koneksi Wi-Fi. ESP8266 menerima data dari smartphone dan meneruskannya ke Arduino Uno untuk ditampilkan pada LED P10. RTC DS3231 digunakan untuk menampilkan waktu, hari, tanggal, bulan, dan tahun.	Penelitian ini menggunakan dua ESP32 DevKit V1 dan dua panel LED P10 yang dikendalikan melalui satu aplikasi mobile Flutter pada jaringan Wi-Fi lokal. Masing-masing panel dapat memiliki konfigurasi teks, mode, kecepatan, kecerahan, dan daya yang berbeda.
2	Triwahyuni & Beta (2022)	Running Text Information System Design Internet-Based for Small Outlets	ESP32, MQTT, Android, DMD LED Panel P10, Internet	Sistem dirancang untuk mengendalikan informasi *running text* melalui internet menggunakan ESP32 dan protokol MQTT sehingga pembaruan informasi tidak terbatas pada jangkauan	Penelitian ini menggunakan dua ESP32 dan dua panel LED P10 yang dikendalikan melalui satu aplikasi Flutter pada jaringan Wi-Fi lokal. Setiap panel dapat dikonfigurasi secara berbeda

				access point lokal.	meliputi teks, mode, kecepatan, kecerahan, dan daya, tanpa menggunakan protokol MQTT atau koneksi internet.
3	Syahputra et al. (2024)	Design and Build Running Text Controls Using IoT	Mikrokontroler, modul *WiFi*, LED P10, sensor DHT22, aplikasi Blynk	Sistem dapat mengendalikan pesan pada LED P10 melalui aplikasi Blynk serta menampilkan data suhu dan kelembapan secara *real-time* menggunakan sensor DHT22. Hasil pengujian menunjukkan teks dan data suhu dapat ditampilkan dengan latensi rendah.	Penelitian ini menggunakan dua ESP32 dan dua panel LED P10 yang dikendalikan melalui satu aplikasi Flutter pada jaringan Wi-Fi lokal. Sistem berfokus pada pengendalian dua panel dengan konfigurasi tampilan yang berbeda, tanpa menggunakan sensor DHT22 untuk monitoring lingkungan.

Berdasarkan Tabel State of the Art, penelitian terdahulu telah menerapkan berbagai teknologi dalam pengembangan sistem *running text*, seperti Arduino Uno, ESP8266, ESP32, Android, MQTT, IoT, dan aplikasi Blynk. Yuhanas et al. (2021) menggunakan Arduino Uno ATmega328 dan ESP8266 untuk mengendalikan LED Matrix P10 melalui aplikasi Android dengan koneksi Wi-Fi. Triwahyuni & Beta (2022) menggunakan ESP32, MQTT, Android, dan DMD LED Panel P10 berbasis internet untuk mengendalikan informasi *running text* sehingga pembaruan informasi tidak terbatas pada jangkauan *access point* lokal. Sementara itu, Syahputra et al. (2024) menerapkan mikrokontroler, modul Wi-Fi, LED P10, aplikasi Blynk, dan sensor DHT22 untuk mengendalikan pesan serta menampilkan data suhu dan kelembapan secara *real-time*.

Berdasarkan penelitian tersebut, penelitian ini mengembangkan sistem *running text* dengan dua panel LED P10 yang dikendalikan menggunakan dua ESP32 melalui satu aplikasi mobile berbasis Flutter pada jaringan Wi-Fi lokal. Setiap panel dapat memiliki konfigurasi tampilan yang berbeda, meliputi teks, mode, kecepatan, kecerahan, dan daya. Dengan demikian, penelitian ini memiliki perbedaan pada penggunaan dua panel yang dikendalikan dalam satu aplikasi serta pengaturan masing-masing panel dalam satu sistem.

Pembahasan

Secara keseluruhan, sistem yang dikembangkan mengintegrasikan komunikasi Internet of Things (IoT) lokal dengan pendekatan *edge computing*. Komunikasi IoT lokal digunakan sebagai media pertukaran data antara aplikasi *mobile* dan ESP32 melalui jaringan *WiFi*, sedangkan pendekatan *edge computing* diterapkan dengan melakukan pemrosesan utama pada masing-masing ESP32. Penggunaan dua ESP32 sebagai *edge controller* memungkinkan setiap panel LED P10 menerima konfigurasi dan menjalankan tampilan secara independen. Konsep komunikasi tersebut sesuai dengan prinsip IoT yang memungkinkan perangkat fisik terhubung dan melakukan pertukaran data melalui jaringan komunikasi (Watmah, 2025).

Hasil implementasi perangkat keras menunjukkan bahwa keberhasilan pengendalian panel LED P10 RGB dipengaruhi oleh kesesuaian konfigurasi antara ESP32 dan karakteristik panel yang digunakan. Pada tahap awal implementasi ditemukan ketidaksesuaian warna dan susunan piksel. Setelah dilakukan penyesuaian konfigurasi GPIO serta penerapan *virtual mapping* untuk panel dengan sistem pemindaian 1/4 *Scan*, teks dan warna dapat ditampilkan dengan susunan piksel yang sesuai. Hasil tersebut menunjukkan bahwa keberhasilan integrasi perangkat keras tidak hanya ditentukan oleh hubungan kelistrikan antara ESP32 dan panel LED, tetapi juga oleh kesesuaian pemetaan data piksel pada perangkat lunak. Dengan demikian, penerapan *virtual mapping* menjadi bagian penting dalam menyesuaikan keluaran ESP32 dengan karakteristik pemindaian panel LED P10 yang digunakan.

Dari sisi komunikasi, hasil pengujian menunjukkan bahwa aplikasi *mobile* mampu berkomunikasi dengan kedua ESP32 melalui jaringan *WiFi* lokal. ESP32 Panel 1 berperan sebagai *Access Point* sekaligus *Web server*, sedangkan ESP32 Panel 2 berfungsi sebagai *Station* yang terhubung pada jaringan tersebut. Data konfigurasi dikirim melalui metode *HTTP POST* dalam format JSON yang memuat parameter pesan, mode tampilan, kecepatan, tingkat kecerahan, dan status daya. Hasil pengujian memperlihatkan bahwa data dapat diterima secara utuh dan diterapkan pada panel yang dituju tanpa terjadi pertukaran data antar panel. Hal tersebut menunjukkan bahwa jaringan lokal dapat digunakan untuk mendukung proses pengendalian running text secara langsung tanpa bergantung pada layanan *cloud* maupun koneksi internet publik. Pemanfaatan IoT untuk memperbarui informasi running text secara nirkabel juga sejalan dengan penelitian Budioko dan Bednar (2025), yang menunjukkan bahwa teknologi IoT dapat digunakan untuk mendukung perubahan informasi pada sistem running text tanpa melakukan pemrograman ulang perangkat. Kemampuan ESP32 dalam menangani transmisi data JSON melalui metode HTTP POST secara konsisten sejalan dengan penelitian terdahulu yang menunjukkan stabilitas tinggi ESP32 pada aplikasi kendali IoT (Yazid & Permana, 2022)**.**

Penggunaan dua ESP32 sebagai pengendali masing-masing panel memberikan kemampuan kontrol yang bersifat independen. Berdasarkan hasil pengujian, Panel 1 dan Panel 2 dapat menerima pesan, mode tampilan, serta nilai kecepatan yang berbeda melalui satu aplikasi *mobile*. Ketika Panel 1 diberikan mode berjalan dan Panel 2 menggunakan mode diam, kedua konfigurasi dapat dijalankan secara bersamaan tanpa saling memengaruhi. Hal yang sama juga terlihat pada pengujian pesan dan kecepatan yang berbeda. Hasil tersebut menunjukkan bahwa pembagian proses pengendalian pada dua ESP32 dapat memisahkan proses pengolahan data untuk setiap panel. Masing-masing ESP32 bertanggung jawab terhadap penerimaan konfigurasi, pemrosesan data, dan pengendalian keluaran pada panel LED P10 yang terhubung dengannya.

Hasil pengujian mode tampilan menunjukkan bahwa tiga mode yang dirancang, yaitu berjalan, diam, dan berkedip, dapat dijalankan sesuai dengan perintah dari aplikasi. Pada mode berjalan, teks bergerak secara horizontal dari kanan ke kiri secara terus-menerus. Pada mode diam, teks dipertahankan pada posisi tertentu tanpa pergerakan piksel, sedangkan pada mode berkedip teks ditampilkan dan dihilangkan secara bergantian berdasarkan interval waktu. Keberhasilan ketiga mode tersebut menunjukkan bahwa ESP32 mampu mengolah parameter

mode yang diterima dari aplikasi menjadi proses pengendalian tampilan pada panel LED. Aplikasi hanya mengirimkan jenis mode yang dipilih, sedangkan proses perubahan posisi teks maupun interval tampilan diproses secara lokal oleh ESP32.

Selain mode tampilan, hasil pengujian kecepatan dan kecerahan menunjukkan bahwa parameter tampilan dapat disesuaikan berdasarkan kebutuhan pengguna. Nilai kecepatan dalam rentang 1 sampai 10 dikonversi oleh ESP32 menjadi interval pergeseran teks, sehingga nilai rendah menghasilkan pergerakan teks yang lebih lambat dan nilai tinggi menghasilkan pergerakan yang lebih cepat. Pengaturan kecerahan juga mampu menghasilkan perubahan intensitas cahaya LED pada tingkat rendah, sedang, dan tinggi. Pengaturan tersebut memberikan fleksibilitas kepada pengguna untuk menyesuaikan tampilan running text dengan kondisi lingkungan maupun kebutuhan penyampaian informasi. Penggunaan aplikasi *mobile* sebagai media pengendali juga mendukung kemudahan perubahan informasi tanpa harus melakukan pengaturan langsung pada perangkat, sebagaimana pemanfaatan aplikasi mobile dalam sistem *running text* yang telah dikembangkan pada penelitian sebelumnya (Triwahyuni & Beta, 2022)**.**

Penerapan *edge computing* pada sistem dapat dilihat secara jelas dari hasil pengujian ketika koneksi aplikasi dengan jaringan *WiFi* diputus setelah konfigurasi berhasil dikirimkan. Pada kondisi tersebut, running text tetap berjalan sesuai dengan konfigurasi terakhir yang diterima. Ketika aplikasi kembali terhubung, konfigurasi baru dapat dikirim dan diterapkan kembali pada panel. Kondisi tersebut menunjukkan bahwa aplikasi tidak harus terus-menerus terhubung selama proses tampilan berlangsung. Aplikasi hanya berfungsi untuk mengirimkan konfigurasi awal atau perubahan konfigurasi, sedangkan proses pengolahan teks, perhitungan posisi karakter, pengaturan animasi, dan pembaruan tampilan dilakukan secara lokal oleh ESP32.

Hasil pengujian tersebut memperkuat penerapan konsep *edge computing* pada sistem yang dikembangkan. ESP32 tidak hanya berfungsi sebagai penerima data atau penghubung komunikasi, tetapi juga bertindak sebagai *edge device* yang melakukan pemrosesan langsung di sisi perangkat. Pendekatan tersebut sesuai dengan konsep *edge computing* yang menempatkan proses pengolahan data lebih dekat dengan sumber data atau perangkat pengguna sehingga dapat mengurangi ketergantungan terhadap server eksternal (Jebamani & Winster, 2022)**.** Penerapan mobile *edge computing* terbukti efektif dalam menjaga kelangsungan operasional perangkat IoT secara mandiri, sehingga running text tetap berjalan normal meskipun koneksi aplikasi terputus (Gutub, 2022).

Berdasarkan seluruh hasil pengujian, integrasi aplikasi *mobile*, komunikasi IoT lokal, dua ESP32, dan dua panel LED P10 telah mampu memenuhi fungsi utama sistem yang dirancang. Pengguna dapat mengubah pesan, mode tampilan, kecepatan, kecerahan, dan status daya melalui aplikasi tanpa harus melakukan unggah ulang program pada ESP32. Selain itu, setiap panel dapat menerima konfigurasi yang berbeda dan tetap bekerja secara independen. Dengan demikian, sistem yang dikembangkan tidak hanya memberikan kemudahan dalam pembaruan informasi secara nirkabel, tetapi juga menerapkan pemrosesan lokal pada masing-masing ESP32 sehingga fungsi utama running text tidak bergantung pada layanan *cloud* maupun koneksi internet publik.

KESIMPULAN DAN SARAN

Kesimpulan

Berdasarkan hasil perancangan, implementasi, dan pengujian, sistem running text dua panel berhasil dikembangkan menggunakan dua ESP32 sebagai edge controller yang berkomunikasi melalui jaringan WiFi lokal. Aplikasi mobile digunakan sebagai antarmuka untuk mengirimkan pesan dan pengaturan tampilan kepada masing-masing ESP32 secara nirkabel, sehingga perubahan informasi dapat dilakukan tanpa melakukan pemrograman ulang

pada perangkat. Penggunaan dua ESP32 memungkinkan setiap panel LED P10 memiliki perangkat pengendali dan pemrosesan lokal secara mandiri. Masing-masing ESP32 bertanggung jawab terhadap satu panel LED P10 yang ditempatkan dalam satu sistem, sehingga setiap panel dapat menerima konfigurasi pesan, mode tampilan, dan kecepatan yang berbeda melalui satu aplikasi mobile . Sementara itu, pengaturan kecerahan dan daya dapat diterapkan untuk mendukung pengoperasian kedua panel. Mode tampilan yang digunakan meliputi teks berjalan, teks diam, dan teks berkedip.

Pendekatan edge computing diterapkan dengan menjadikan masing-masing ESP32 sebagai perangkat yang melakukan pemrosesan utama secara lokal. Setelah konfigurasi diterima dari aplikasi, ESP32 mengolah teks, menentukan mode tampilan, mengatur kecepatan pergerakan, menghitung perubahan posisi karakter, serta mengendalikan keluaran menuju panel LED secara mandiri. Dengan mekanisme tersebut, aplikasi tidak perlu melakukan komunikasi secara terus-menerus selama proses running text berlangsung. Penerapan komunikasi IoT lokal memungkinkan aplikasi mobile mengirimkan konfigurasi kepada ESP32 melalui jaringan WiFi lokal tanpa menggunakan layanan cloud atau koneksi internet publik, sedangkan pendekatan edge computing memungkinkan proses pengolahan tampilan dilakukan langsung pada masing-masing ESP32. Sistem ini dapat menjadi dasar pengembangan papan informasi digital yang lebih fleksibel melalui penambahan jumlah panel, penyimpanan konfigurasi, maupun integrasi dengan jaringan yang lebih luas.

REFERENSI

Asvial, M., Cracias, A., Laagu, M., & Arifin, A. S. (2021). Design and analysis of low power and lossy network routing system for Internet of Things network. International Journal of Intelligent Engineering and Systems, 14(4), 48–58. https://doi.org/10.22266/IJIES2021.0831.48

Budioko, T., & Bednar, B. (2025). Sistem IoT pengubah informasi running text berbasis MQTT dan FreeRTOS. JIKO (Jurnal Informatika dan Komputer), 9(2), 463–473[2]. https://doi.org/10.26798/jiko.v9i2.1934

Cao, K., Liu, Y., Meng, G., & Sun, Q. (2020). An overview on edge computing research. IEEE Access, 8, 85714–85728[2][3]. https://doi.org/10.1109/ACCESS.2020.2991734

Gutub, A. (2022). AI-based mobile edge computing for IoT: Applications, challenges, and future scope. Arabian Journal for Science and Engineering, 47(8), 9801–9831[3]. https://doi.org/10.1007/s13369-021-06348-2

Jebamani, S. A., & Winster, S. G. (2022). Localization of IoT edge devices. In Internet of Everything Smart Sensing Technologies (pp. 115–130)[3]. CRC Press.

Mulyani, Y., Gusti, K. S., Septiana, T., & Septama, H. D. (2024). Rancang bangun aplikasi pengelolaan perkuliahan tatap muka berbasis mobile menggunakan framework Flutter. Jurnal Informatika dan Teknik Elektro Terapan, 12(1), 645–654[3][4]. https://doi.org/10.23960/jitet.v12i1.3892

Nataprawira, A. S., Rizal, A., & Wibowo, A. S. (2020). Perancangan display LED dot matrix via Wi-Fi menggunakan aplikasi mobile Android. INTECH: Informational Technology Journal, 1(1), 1–7[4]. https://doi.org/10.54895/intech.v1i1.240

Purba, S., Hariri, M., Banjarnahor, R. J., & Siregar, S. N. (2023). LED control system using Arduino Wemos D1 R1 based on web server communication via Internet of Things (IoT). Formosa Journal of Science and Technology, 2(6), 1397–1408[4][5]. https://doi.org/10.55927/fjst.v2i6.4436

Ramadanti, E. F., Taqwa, A., & Adewasti, A. (2026). Website-based information management and temperature and air quality monitoring system using ESP32. SISFO: Jurnal Ilmiah Sistem Informasi, 10(2), 471–481[5][6]. https://doi.org/10.29103/sisfo.v10i2.27852

Syahputra, A. T., Fauzi, A., & Syari, M. A. (2024). Design and build running text controls using IoT. Journal of Artificial Intelligence and Engineering Applications, 4(1), 295–301[6]. https://doi.org/10.59934/jaieA.v4i1.625

Triwahyuni, N., & Beta, S. (2022). Running text information system design internet-based for small outlets. JAICT (Journal of Applied Informatics and Communication Technology), 7(2), 97–101[6][7]. https://doi.org/10.32497/jaict.v7i2.3270

Watmah, S. (2025). Pemantauan dan pengendalian perangkat elektronik berbasis Internet of Things (IoT) menggunakan ESP32. IMTechno: Journal of Industrial Management and Technology, 6(2), 94–99. https://doi.org/10.31294/imtechno.v6i2.9518

Yasa, I. K. A., Purbhawa, I. M., Yasa, I. M. S., Teresna, I. W., Nugroho, A., & Winardi, S. (2023). IoT-based electrical power recording using ESP32 and PZEM-004T microcontrollers. Journal of Computer Science Technology and Studies, 5(4), 62–68. https://doi.org/10.32996/jcsts.2023.5.4.7

Yazid, Y. A. M., & Permana, R. A. (2022). Rancang bangun prototype monitoring lampu jalan secara otomatis menggunakan mikrokontroller ESP32 dan API Bot Telegram. Jurnal Teknik Informatika, 8(1), 12–19[7]. https://doi.org/10.51998/jti.v8i1.477

Yuhanas, M., Hadi, C. F., & Lestari, R. F. (2021). Rancang bangun running text menggunakan modul LED matrix P10 berbasis Arduino Uno di Fakultas Teknik Universitas PGRI Banyuwangi. Journal Zetroem, 3(2), 16–22[7][8]. https://doi.org/10.36526/ztr.v3i2.1479.