SISTEM BEL DAN JADWAL PELAJARAN DI SEKOLAH BERBASIS IOT

PROYEK AKHIR

Laporan akhir ini dibuat dan diajukan untuk memenuhi salah satu syarat kelulusan Sarjana Terapan/Diploma III Politeknik Manufaktur Negeri Bangka Belitung

Disusun Oleh:

Ariza Prasetya NIM 0032234

Hardiansyah NIM 0032240

POLITEKNIK MANUFAKTUR NEGERI

BANGKA BELITUNG

2025

LEMBAR PENGESAHAN

SISTEM BEL DAN JADWAL PELAJARAN DI SEKOLAH BERBASIS IOT

Oleh:

Ariza Prasetya 0032234

Hardiansyah 0032240

Laporan akhir ini telah disetujui dan disahkan sebagai salah satu syarat kelulusan Program Sarjana Terapan/Diploma III Politeknik Manufaktur Negeri Bangka Belitung

Menyetujui,

Pembimbing 1	Pembimbing 2
Surojo,M.T	Dr. Parulian Silalahi,M.Pd.
Penguji 1	**Penguji 2**
Daya Wulandari,M.Tr.T	**Enggar Hero Istoto,S.Si.,M.En**

PERNYATAAN BUKAN PLAGIAT

Yang bertanda tangan di bawah ini:

Nama Mahasiswa 1 : Ariza Prasetya NIM : 0032234

Nama Mahasiswa 2 : Hardiansyah NIM : 0032240

Dengan Judul : Sistem Bel dan Jadwal Pelajaran di Sekolah Berbasis IoT

Menyatakan bahwa laporan akhir ini adalah hasil kerja kami sendiri dan bukan merupakan plagiat. Pernyataan ini kami buat dengan sebenarnya dan bila ternyata dikemudian hari ternyata melanggar pernyataan ini, kami bersedia menerima sanksi yang berlaku.

Sungailiat, 26 Juni 2025

Mahasiswa 1

Ariza Prasetya

Mahasiswa 2

Hardiansyah

ABSTRAK

Implementasi sistem otomatis berbasis Internet of Things (IoT) dalam dunia pendidikan menjadi solusi strategis untuk meningkatkan efisiensi pengelolaan waktu. Penelitian ini menyajikan perancangan dan implementasi sistem bel dan jadwal pelajaran otomatis berbasis IoT yang terintegrasi dengan kontrol jarak jauh melalui aplikasi mobile. Sistem ini dikembangkan menggunakan NodeMCU ESP32 sebagai unit kendali utama, DFPlayer Mini sebagai pemutar audio otomatis, LED Dot Matrix untuk tampilan informasi, dan speaker sebagai keluaran suara. Antarmuka pengguna dikembangkan melalui platform Blynk, sementara proses pemrograman dilakukan dengan Arduino IDE. Hasil pengujian menunjukkan bahwa sistem mampu beroperasi secara stabil, membunyikan bel sesuai dengan jadwal, dan menampilkan informasi secara real-time. Selain itu, integrasi antara kontrol manual dan otomatis melalui aplikasi mobile meningkatkan fleksibilitas dan kenyamanan pengguna. Sistem ini berkontribusi pada peningkatan efisiensi manajemen waktu dan otomasi operasional di lingkungan pendidikan.

Kata Kunci: Internet of Things, ESP32, Bel Otomatis, Jadwal Pelajaran, Blynk.

ABSTRACT

The integration of Internet of Things (IoT)-based systems in educational environments represents a strategic approach to improving time management efficiency. This study presents the design and implementation of an IoT-based automatic school bell and class scheduling system, controllable remotely via a mobile application. The system utilizes a NodeMCU ESP32 microcontroller as the main control unit, a DFPlayer Mini for automated audio playback, an LED Dot Matrix display for schedule visualization, and a speaker for audio output. The user interface is developed using the Blynk platform, and all programming is conducted through the Arduino IDE. Experimental results demonstrate that the system operates reliably, rings the bell in accordance with predefined schedules, and displays real-time schedule information. Furthermore, the integration of mobile control enhances user flexibility and operational convenience. This system contributes to the automation and optimization of scheduling tasks within educational institutions.

Keywords: Internet of Things, ESP32, Automatic Bell, Class Schedule, Blynk.

KATA PENGANTAR

Rasa syukur tiada hentinya penulis panjatkan kehadirat Allah SWT atas segala nikmat dan karunia-Nya sehingga penulis dapat menyelesaikan Makalah Proyek Akhir ini tepat pada waktunya.

Makalah ini disusun sebagai salah satu persyaratan penting untuk menyelesaikan program Diploma III di Politeknik Manufaktur Negeri Bangka Belitung. Dengan adanya makalah ini, diharapkan pembaca akan memperoleh pemahaman yang jelas mengenai proyek akhir yang telah dikerjakan oleh penulis. Dalam pembuatan proyek akhir ini, penulis mengaplikasikan pengetahuan yang telah diperoleh selama kurang lebih 3 tahun menempuh pendidikan di Politeknik Manufaktur Negeri Bangka Belitung. Selain itu, penulis juga memanfaatkan informasi dan data dukungan dari makalah-makalah proyek akhir mahasiswa Politeknik Manufaktur Negeri Bangka Belitung dari tahun-tahun sebelumnya.

Selama menyusun makalah proyek akhir ini penulis mendapatkan banyak bantuan dan bimbingan dari berbagai pihak sehingga penulisan makalah ini dapat diselesaikan dengan baik. Oleh karena itu, penulis ingin mengucapkan terima kasih kepada:

1. Orang tua dan keluarga tercinta yang selalu memberikan dukungan baik secara moral maupun materil sehingga penulis dapat memberikan hasil yang terbaik.
2. Bapak I Made Andik Setiawan, M.Eng., Ph.D. selaku Direktur Politeknik Manufaktur Negeri Bangka Belitung.
3. Bapak Zanu Saputra, M.Tr.T selaku Ketua Jurusan Teknik Elektro dan Informatika Politeknik Manufaktur Negeri Bangka Belitung.
4. Ibu Novitasari, M.Pd. Selaku Ketua Prodi Elektronika di Politeknik Manufaktur Negeri Bangka Belitung.

5. Pembimbing 1 Bapak Surojo, S.T., M.T. dalam proyek akhir ini.
6. Bapak Dr. Parulian Silalahi, M.Pd. selaku dosen pembimbing 2 dalam proyek akhirini.
7. Seluruh tenaga pendidik dan kependidikan Politeknik Manufaktur Negeri Bangka Belitung.
8. Rekan-rekan mahasiswa/i Politeknik Manufaktur Negeri Bangka Belitung yang telah membantu dalam penyelesaian proyek akhir.
9. Rekan-rekan mahasiswa/i jurusan Teknik Elektro dan Informatika angkatan 2025 yang telah berperan penting dalam memberikan pengalaman dan pembelajaran selama di bangku perkuliahan.
10. Sahabat-sahabat penulis yang selalu memberikan canda tawa, menemami, dan merangkul penulis selama proses pembelajaran dari awal sampai akhir sehingga penulis dapat menyelesaikan proyek akhir ini.
11. Kepada diri sendiri karena telah bertanggung jawab untuk menyelesaikan apa yang telah dimulai dan selalu berusaha untuk tidak menyerah, serta senantiasa menikmati setiap prosesnya.

Penulis menyadari bahwa ada kekurangan dalam laporan ini dikarenakan terbatasnya pengetahuan dan keterampilan penulis. Oleh karena itu, penulis mengharapkan kritik dan saran yang membangun dari semua pihak untuk memperbaiki dan mengembangkan laporan ini di masa yang akan datang. Penulis berharap laporan ini dapat memberikan manfaat bagi para pembaca yang berkepentingan, terutama, dan juga berkontribusi pada perkembangan ilmu pengetahuan dan teknologi secara umum.

Sungailiat, 26 Juni 2025

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Bel sekolah merupakan sarana penting dalam mendukung keteraturan kegiatan belajar mengajar di lingkungan pendidikan. Fungsi utamanya adalah sebagai penanda waktu pergantian jam pelajaran, istirahat, dan kepulangan. Namun, pada banyak institusi pendidikan, sistem bel yang digunakan masih bersifat manual atau menggunakan *timer* sederhana yang tidak fleksibel dan rawan kesalahan operasional [1]. Ketergantungan pada operator atau keterbatasan perangkat non-digital ini menyebabkan efektivitas pengelolaan waktu di sekolah menjadi kurang optimal [2].

Seiring perkembangan teknologi *Internet of Things* (IoT), sistem kendali otomatis kini dapat diterapkan untuk menggantikan sistem konvensional yang terbatas. IoT memungkinkan perangkat fisik seperti *mikrokontroler*, sensor, dan aktuator untuk terhubung ke jaringan internet dan dikendalikan secara *real-time* dari jarak jauh [3]. Pemanfaatan teknologi ini dalam sistem bel sekolah memberikan kemudahan dalam pengaturan jadwal, pemantauan sistem, serta integrasi dengan aplikasi *mobile* yang dapat diakses oleh operator sekolah [4].

Proyek akhir ini bertujuan merancang sistem bel dan jadwal pelajaran otomatis berbasis IoT menggunakan *mikrokontroler* ESP32 sebagai pusat kendali, *DFPlayer Mini* sebagai pemutar suara bel, *speaker* sebagai *output* suara, dan *Led dot matrix* untuk menampilkan informasi waktu atau jadwal pelajaran. Sistem ini tidak menggunakan RTC, melainkan dikendalikan secara dinamis melalui aplikasi *Blynk*, sehingga jadwal dapat diatur langsung melalui *smartphone* tanpa memodifikasi perangkat keras. Dengan pendekatan ini, diharapkan sistem bel

sekolah menjadi lebih fleksibel, efisien, dan sesuai dengan kebutuhan institusi pendidikan modern [5][6].

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, maka rumusan masalah dalam penelitian ini adalah sebagai berikut:

1. Bagaimana merancang sistem bel sekolah otomatis berbasis *Internet of Things* (IoT) menggunakan *mikrokontroler* ESP32 tanpa modul RTC?
2. Bagaimana mengintegrasikan *DFPlayer Mini* dan *speaker* sebagai media *output* suara dalam sistem bel otomatis?
3. Bagaimana merancang*Led dot matrix* sebagai media informasi penjadwalan pelajaran di lingkungan sekolah?
4. Bagaimana merancang sistem pengendalian bel sekolah yang dapat diakses dan dikonfigurasi melalui aplikasi mobile *Blynk*?

1.3 Tujuan Proyek Akhir

Tujuan dari proyek akhir ini adalah untuk merancang dan mengimplementasikan sistem bel sekolah otomatis berbasis *Internet of Things* (IoT) dengan memanfaatkan *mikrokontroler* ESP32 sebagai unit kendali utama. Secara khusus, tujuan yang ingin dicapai adalah:

1. Merancang sistem bel otomatis tanpa menggunakan modul RTC, dengan kendali terpusat melalui koneksi internet.
2. Mengintegrasikan *DFPlayer Mini* dan *speaker* sebagai komponen utama dalam pemutaran suara bel secara otomatis.
3. Merancang tampilan informasi jadwal pelajaran melalui *Led dot matrix* yang dikendalikan oleh *mikrokontroler* ESP 32.
4. Merancang antarmuka pengendali sistem bel berbasis aplikasi *mobileBlynk* untuk mempermudah konfigurasi dan pemantauan jadwal secara fleksibel.

1.4 Batasan Masalah

Agar proyek akhir ini terfokus dan berjalan sesuai dengan tujuan yang telah ditetapkan, maka ruang lingkup kajian dibatasi pada aspek-aspek teknis dan fungsional tertentu. Batasan ini juga diperlukan untuk menghindari perluasan topik di luar kemampuan waktu, sumber daya, dan perangkat keras yang digunakan. Beberapa batasan dalam proyek akhir ini adalah sebagai berikut:

a) Sistem dikembangkan menggunakan *mikrokontroler*ESP32 sebagai inti kendali, dengan koneksi Wi-Fi untuk sinkronisasi waktu, tanpa penggunaan modul RTC.
b) Tampilan informasi hanya difokuskan pada *Led dot matrix*, yang menampilkan waktu dan jadwal pelajaran secara bergilir dan tidak menggunakan display grafis lainnya.
c) Sinyal bel atau audio yang dikeluarkan berasal dari *DFPlayer Mini* yang diprogram memutar file MP3, tanpa integrasi suara dinamis atau TTS (*Text-to-Speech*).
d) Sistem kontrol dan pemantauan hanya dapat diakses melalui aplikasi *Blynk* yang terhubung ke jaringan internet, tanpa pengembangan aplikasi khusus atau integrasi dengan sistem akademik sekolah.
e) Pemrograman seluruh sistem dilakukan menggunakan Arduino IDE, sehingga batasan logika dan performa disesuaikan dengan kapabilitas platform tersebut.

Dengan batasan-batasan ini, project akhir akan lebih terarah pada pengembangan prototipe sistem bel otomatis berbasis IoT yang sederhana namun fungsional, dan dapat diterapkan di lingkungan sekolah secara efektif.

BAB II
LANDASAN TEORI

2.1. *Internet of Things* (IoT)

Internet of Things (IoT) yang ditunjukkan pada gambar 2.1. adalah konsep teknologi yang memungkinkan berbagai perangkat elektronik terhubung melalui jaringan internet dan saling berkomunikasi tanpa intervensi manusia secara langsung. Sistem ini dapat menghubungkan *mikrokontroler*, sensor, dan aktuator dalam satu kesatuan yang dikendalikan secara otomatis melalui jaringan[7]. Dalam implementasinya, IoT dapat digunakan untuk mengontrol sistem bel secara otomatis berdasarkan jadwal yang telah ditentukan, sehingga proses pengaturan waktu menjadi lebih efisien dan terstruktur[7].

Gambar 2.1. *Internet Of Things* (IOT)

Sebaliknya, pada sistem otomatis yang belum menerapkan IoT, perangkat masih bergantung pada pengaturan lokal melalui *mikrokontroler* dengan jadwal tetap yang tertanam dalam kode program[8]. Sistem seperti ini belum memungkinkan kontrol jarak jauh, sehingga perubahan jadwal harus dilakukan secara manual langsung pada perangkat. Perbandingan ini menunjukkan bahwa integrasi IoT menawarkan nilai tambah yang signifikan dalam hal kemudahan pengelolaan dan adaptabilitas sistem otomatis[8].

2.2.*Mikrokontroler* ESP32

Mikrokontroler ESP32 yang ditunjukkan pada gambar 2.2. merupakan otak dari sistem tertanam (*embedded system*) yang berfungsi untuk memproses *input* dan menghasilkan *output* sesuai dengan logika program yang telah ditanamkan. Dalam pengembangan sistem bel otomatis, *mikrokontroler* berperan penting dalam mengatur jadwal, memicu suara bel, serta mengendalikan tampilan informasi pada perangkat yang terhubung. Salah satu jenis *mikrokontroler* yang banyak digunakan adalah ESP32 karena memiliki konektivitas Wi-Fi dan Bluetooth bawaan, serta kemampuan pemrosesan yang lebih tinggi dibandingkan *mikrokontroler* generasi sebelumnya.

Gambar 2.2. Nodemcu ESP32

Pada penelitian sebelumnya, pengendalian sistem bel otomatis dilakukan menggunakan mikrokontroler Arduino Uno yang terintegrasi dengan modul RTC sebagai penentu waktu bel berbunyi[9]. Namun, sistem tersebut masih bergantung pada pengaturan waktu lokal dan belum mendukung kontrol jarak jauh. Berbeda dengan ESP32 yang memiliki konektivitas nirkabel, *mikrokontroler* ini memungkinkan sistem untuk terhubung dengan internet dan dikontrol melalui aplikasi seperti *Blynk*, sehingga pengelolaan jadwal menjadi lebih fleksibel dan efisien.

2.3.*DFPlayer Mini*

DFPlayer Mini yang ditunjukkan pada gambar 2.3. adalah modul pemutar file audio berukuran kecil yang mendukung format MP3 dan WAV melalui penyimpanan eksternal seperti *microSD*. Modul ini banyak digunakan dalam sistem berbasis *mikrokontroler* karena dapat dioperasikan dengan komunikasi serial yang sederhana. Dalam sistem bel otomatis, *DFPlayer Mini* berfungsi untuk memutar suara bel secara otomatis berdasarkan instruksi dari *mikrokontroler*, tanpa memerlukan perangkat pemroses audio tambahan[10]. Modul ini mampu menyimpan dan memainkan banyak file suara, sehingga memungkinkan variasi suara bel sesuai dengan kebutuhan sistem.

Gambar 2.3.*Dfplayer Mini*

DFPlayer Mini memiliki kemampuan pengaturan volume dan pemilihan file audio melalui perintah digital, yang memudahkan integrasi dengan sistem kendali otomatis. Pengguna cukup mengatur pemutaran berdasarkan jadwal yang telah diprogram, dan modul akan menjalankan suara secara tepat waktu. Dengan konsumsi daya rendah dan dimensi yang ringkas, *DFPlayer Mini* menjadi pilihan ideal dalam sistem bel otomatis modern yang membutuhkan efisiensi ruang dan daya[10].

2.4.*Speaker*

Speaker yang ditunjukkan pada gambar 2.4. merupakan komponen *output* audio yang berfungsi untuk mengubah sinyal listrik dari modul audio seperti *DFPlayer Mini* menjadi gelombang suara yang dapat didengar oleh manusia. Dalam sistem bel otomatis, *speaker* digunakan sebagai perangkat akhir untuk menyampaikan informasi bunyi, seperti tanda masuk, istirahat, atau pulang sekolah. Kualitas *speaker* yang digunakan memengaruhi kejernihan dan kekuatan suara yang dihasilkan, sehingga pemilihan impedansi dan daya *output* yang sesuai sangat penting agar suara tidak mengalami distorsi[11].

Gambar 2.4.*Speaker*

Penggunaan *speaker* dalam sistem berbasis *mikrokontroler* perlu mempertimbangkan sumber daya dan kebutuhan volume. Biasanya, *speaker* 8Ω dengan daya 3W atau 5W digunakan dalam sistem sederhana. Pada kondisi tertentu, *speaker* juga dapat dikombinasikan dengan *amplifier* tambahan apabila dibutuhkan keluaran suara yang lebih besar. Dalam sistem bel otomatis, *speaker* bekerja secara otomatis berdasarkan sinyal dari modul pemutar suara seperti *DFPlayer Mini* yang dikendalikan oleh *mikrokontroler* utama[11].

2.5. Aplikasi *Blynk*

Aplikasi *Blynk* yang ditunjukkan pada gambar 2.5. merupakan platform *Internet of Things* (IoT) yang memungkinkan pengguna untuk mengontrol dan memantau perangkat berbasis *mikrokontroler* secara *real-time* melalui antarmuka aplikasi *mobile*. Dalam sistem bel otomatis, *Blynk* digunakan sebagai penghubung antara *mikrokontroler* dan pengguna, di mana pengguna dapat mengatur jadwal bel, mengaktifkan atau menonaktifkan sistem, serta memantau status perangkat dari jarak jauh menggunakan koneksi internet[12]. Kelebihan utama dari *Blynk* adalah antarmuka yang sederhana, fleksibel, dan dapat dikonfigurasi tanpa perlu mengubah kode program secara keseluruhan.

Gambar 2.5. Aplikasi *Blynk*

Integrasi *Blynk* dengan *mikrokontroler* seperti ESP32 mempermudah pengguna dalam mengelola sistem bel otomatis tanpa harus berinteraksi langsung dengan perangkat keras. Platform ini juga mendukung penggunaan berbagai jenis widget, seperti tombol, slider, dan notifikasi, yang dapat disesuaikan sesuai kebutuhan sistem. Dengan adanya *Blynk*, sistem menjadi lebih efisien karena pengguna tidak perlu melakukan pemrograman ulang hanya untuk mengatur ulang jadwal atau mengganti mode operasi perangkat secara manual[12].

2.6. Aplikasi Arduino IDE

Arduino IDE (*Integrated Development Environment*) yang ditunjukkan pada gambar 2.6. adalah perangkat lunak yang digunakan untuk menulis, mengunggah, dan menguji program ke dalam papan *mikrokontroler* seperti ESP32. Dalam sistem bel otomatis, Arduino IDE memegang peranan penting karena menjadi tempat utama untuk menyusun logika pemrograman seperti penjadwalan bel, pemutaran audio, serta integrasi dengan modul-modul seperti *DFPlayer Mini* dan *LED dot matrix*[13]. Bahasa pemrograman yang digunakan pada Arduino IDE berbasis C/C++, sehingga memudahkan pengembangan sistem yang bersifat modular dan terstruktur.

Gambar 2.6. Aplikasi Arduino IDE

Arduino IDE mendukung berbagai pustaka (*library*) yang dapat diinstal sesuai dengan kebutuhan komponen yang digunakan dalam proyek. Kemudahan antarmuka pengguna dan kompatibilitasnya dengan banyak jenis *mikrokontroler* menjadikan Arduino IDE sebagai platform yang populer di kalangan pengembang sistem otomatisasi. Dalam proyek bel otomatis berbasis IoT, Arduino IDE juga digunakan untuk mengintegrasikan mikrokontroler dengan aplikasi pihak ketiga seperti *Blynk* untuk pengendalian jarak jauh[13].

2.7. Led Dot Matrix

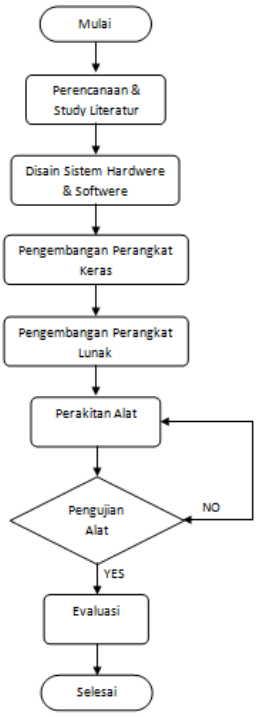
LED dot matrix yang ditunjukkan pada gambar 2.7. merupakan tampilan visual yang tersusun dari kumpulan *LED* dalam bentuk matriks baris dan kolom, yang memungkinkan penyajian informasi berupa teks atau simbol secara dinamis. Dalam sistem bel otomatis, *LED dot matrix* digunakan untuk menampilkan informasi penting seperti jam digital, jadwal pelajaran, atau pemberitahuan yang berkaitan dengan aktivitas sekolah[14]. Komponen ini dikendalikan langsung oleh mikrokontroler melalui antarmuka serial atau paralel, dan dapat diprogram untuk menampilkan data secara bergulir (*running text*) maupun statis.

Gambar 2.7.*Led Dot Matrix*

Pemanfaatan *LED dot matrix* dalam sistem otomatisasi berbasis IoT mendukung penyampaian informasi secara *real-time*. Melalui pemrograman yang dilakukan pada *mikrokontroler* dan *software* pendukung, tampilan dapat disesuaikan dengan waktu tertentu atau diperbarui dari jarak jauh. Dengan integrasi ini, sistem bel otomatis menjadi tidak hanya sebagai pemicu suara, tetapi juga sebagai media informasi visual yang informatif dan responsif terhadap jadwal yang telah ditentukan[15].

BAB III
METODE PELAKSANAAN

Bab ini membahas metode pelaksanaan yang akan dilakukan selama proyek akhir yang berjudul "SISTEM BEL DAN JADWAL PELAJARAN DI SEKOLAH BERBASIS IOT" bertujuan untuk membuatnya lebih mudah untuk diselesaikan. Metode pelaksanaan atau tahapan proses pengerjaan proyek digambarkan dalam diagram *flowchart* berikut:



Gambar 3.1.*Flowcart* Metode Pelaksanaan

Berikut adalah beberapa tahapan dalam proses pembuatan proyek akhir, seperti yang ditunjukkan pada gambar3.1. *flowcart* metode pelaksanaan di atas:

3.1. Study literatur

Study literatur dilakukan untuk memperoleh pemahaman yang mendalam mengenai komponen-komponen utama serta konsep teknologi yang digunakan dalam perancangan sistem bel dan jadwal pelajaran otomatis berbasis IOT. Literatur yang dikaji meliputi penelitian-penelitian terdahulu yang membahas penerapan *mikrokontroler* ESP32 dalam sistem kendali otomatis, penggunaan modul *DFPlayer Mini* sebagai pemutar audio, serta implementasi *LED dot matrix* untuk tampilan informasi.

Selain itu, studi literatur juga mencakup pemahaman terhadap platform aplikasi *mobile Blynk* sebagai media pengendalian jarak jauh sistem IOT, serta penggunaan Arduino IDE sebagai alat bantu dalam pemrograman *mikrokontroler* dan integrasi berbagai perangkat keras. Kajian literatur ini bertujuan untuk mengidentifikasi metode terbaik, kelebihan, serta kendala yang mungkin dihadapi dalam pengembangan sistem, sehingga dapat menjadi dasar dalam perancangan dan implementasi proyek ini.

Studyliteratur dilakukan secara komprehensif, penelitian ini diharapkan dapat mengadopsi teknologi yang tepat guna menghasilkan sistem bel otomatis yang handal, efisien, dan mudah dikontrol sesuai kebutuhan sekolah modern.

3.2.Desain Sistem*Hardware* dan *Software*

Desain sistem *hardware* dalam penelitian ini melibatkan pemilihan dan pengintegrasian komponen utama yang terdiri dari *mikrokontroler* ESP32 sebagai pusat kendali, modul *DFPlayer Mini* sebagai pemutar suara bel, *speaker* sebagai *output* audio, dan *LED dot matrix* sebagai media tampilan informasi jadwal pelajaran. ESP32 bertugas menerima perintah melalui koneksi WiFi yang dikendalikan oleh aplikasi *mobile Blynk*, kemudian memproses data jadwal dan mengaktifkan modul *DFPlayer Mini* serta *LED dot matrix* sesuai dengan waktu yang telah ditentukan.

Rangkaian *hardware* dirancang agar komponen-komponen tersebut dapat saling terhubung dengan baik, memperhatikan kebutuhan sumber daya, komunikasi antar modul, serta kestabilan sistem. Penggunaan ESP32 memungkinkan sistem untuk beroperasi tanpa modul RTC dengan mengandalkan sinkronisasi waktu melalui internet.

Sementara itu, desain *software* meliputi pembuatan program menggunakan Arduino IDE yang mengatur logika pengendalian jadwal bel, pemutaran audio, dan tampilan *LED dot matrix*. Program juga mengimplementasikan komunikasi dengan aplikasi *Blynk* untuk menerima update jadwal secara *real-time*. Desain *software* dibuat modular agar memudahkan pemeliharaan dan pengembangan sistem di masa depan.

Desain *hardware* dan *software* yang terintegrasi secara baik, sistem diharapkan mampu berfungsi secara otomatis dan responsif terhadap perubahan jadwal yang dilakukan melalui aplikasi *mobile*.

3.3. Pengembangan Perangkat Keras

Pengembangan perangkat keras dalam sistem bel dan jadwal pelajaran otomatis ini dimulai dengan pemilihan komponen utama yang sesuai dengan kebutuhan fungsional sistem. *Mikrokontroler* ESP32 dipilih sebagai otak pengendali karena kemampuannya untuk terhubung ke jaringan WiFi dan kompatibilitasnya dengan platform Arduino IDE. Modul *DFPlayer Mini* digunakan untuk memutar file audio bel secara otomatis, sedangkan *speaker* dipasang sebagai *output* suara utama.

Selanjutnya, *LED dot matrix* diintegrasikan untuk menampilkan informasi jadwal pelajaran secara visual. Semua komponen ini dirakit pada papan *prototipe* (*breadboard*) terlebih dahulu untuk memastikan konektivitas dan fungsi masing-masing modul berjalan dengan baik. Pengkabelan dilakukan sesuai dengan

diagram rangkaian yang telah dirancang untuk meminimalkan gangguan sinyal dan memastikan kestabilan sistem.

Sumber daya yang digunakan disesuaikan dengan kebutuhan masing-masing komponen agar sistem dapat berjalan dengan optimal tanpa mengalami gangguan daya. Setelah tahap perakitan selesai, dilakukan pengujian awal untuk memastikan semua perangkat keras bekerja sesuai dengan fungsi yang diinginkan, khususnya komunikasi antara ESP32 dengan modul *DFPlayer Mini* dan *LED dot matrix*, serta *output* suara melalui *speaker*.

Tahap pengembangan perangkat keras ini menjadi fondasi penting bagi keberhasilan integrasi dengan perangkat lunak dalam menciptakan sistem bel otomatis yang handal dan efisien.

3.4. Pengembangan Perangkat Lunak

Pengembangan perangkat lunak pada sistem bel dan jadwal pelajaran otomatis berbasis IOT ini dilakukan dengan menggunakan Arduino IDE sebagai lingkungan pemrograman utama. Program yang dibuat berfungsi untuk mengatur logika kendali jadwal bel, komunikasi dengan aplikasi *mobile Blynk*, serta pengelolaan *output* suara melalui *DFPlayer Mini* dan tampilan informasi pada *LED dot matrix*.

Langkah pertama dalam pengembangan perangkat lunak adalah pembuatan kode program yang mampu menghubungkan *mikrokontroler* ESP32 dengan jaringan WiFi dan menerima perintah dari aplikasi *Blynk* secara *real-time*. Selanjutnya, program mengelola data jadwal yang diterima untuk mengaktifkan modul *DFPlayer Mini* sesuai waktu yang telah ditentukan agar suara bel dapat diputar otomatis.

Selain itu, perangkat lunak juga mengendalikan tampilan *LED dot matrix* yang menampilkan jadwal pelajaran atau informasi penting lainnya, sehingga pengguna dapat memantau jadwal dengan mudah. Penggunaan *library* yang

kompatibel dengan komponen seperti *DFPlayer Mini* dan *LED dot matrix* mempermudah pengembangan dan pengujian perangkat lunak.

Setelah tahap pemrograman selesai, dilakukan pengujian dan *debugging* untuk memastikan seluruh fungsi berjalan dengan baik dan sistem dapat merespon perintah dari aplikasi *Blynk* secara tepat waktu. Pengembangan perangkat lunak yang terstruktur dan modular diharapkan dapat memudahkan perawatan dan pengembangan sistem di masa mendatang.

seluruh fungsi berjalan stabil dan responsif sesuai kebutuhan operasional di lingkungan sekolah.

3.5. Perakitan Alat

Perakitan alat dimulai dengan pengaturan komponen-komponen utama, yaitu *mikrokontroler*ESP32, modul *DFPlayer Mini*, speaker, dan *LED dot matrix*, pada papan *prototipe* atau *casing* yang telah disiapkan. Setiap komponen dihubungkan sesuai dengan diagram rangkaian yang telah dirancang sebelumnya, memastikan koneksi antara pin *input* dan *output* berjalan dengan baik.

Proses perakitan memperhatikan penataan kabel agar rapi dan menghindari gangguan sinyal serta kerusakan mekanis. *Power supply* disambungkan untuk menyediakan daya yang stabil sesuai kebutuhan setiap modul. *Speaker* diposisikan sedemikian rupa agar suara bel dapat terdengar jelas di area yang diinginkan.

Setelah perakitan fisik selesai, dilakukan pengecekan koneksi dan pengujian fungsi masing-masing komponen secara individual sebelum diintegrasikan menjadi satu kesatuan sistem. Pengujian ini meliputi verifikasi komunikasi antara ESP32 dengan *DFPlayer Mini*, respons suara bel, dan tampilan pada *LED dot matrix*.

3.6. Pengujian Alat

Setelah proses perakitan selesai, langkah selanjutnya adalah melakukan pengujian alat untuk memastikan seluruh komponen bekerja dengan baik dan sistem berjalan sesuai dengan fungsi yang diharapkan. Pengujian dilakukan secara bertahap, mulai dari pengujian konektivitas jaringan, pemutaran suara bel, hingga tampilan jadwal pada *LED dot matrix*.

Pertama, dilakukan pengujian konektivitas ESP32 dengan jaringan WiFi serta komunikasi antara ESP32 dan aplikasi *Blynk*. Tujuannya adalah untuk memastikan bahwa perangkat dapat menerima perintah secara *real-time* dari aplikasi *mobile*. Selanjutnya, dilakukan pengujian pemutaran file audio menggunakan *DFPlayer Mini* dan *speaker* untuk memastikan suara bel terdengar jelas dan aktif sesuai waktu yang dijadwalkan.

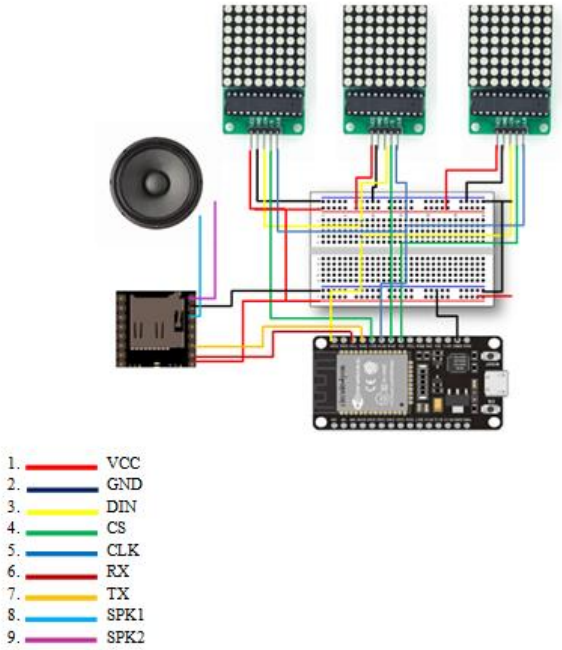
Tahapan berikutnya adalah pengujian tampilan informasi jadwal pelajaran pada *LED dot matrix*. Tampilan harus menyesuaikan dengan jadwal yang telah dikonfigurasi melalui aplikasi *Blynk*, dan informasi yang ditampilkan harus terbaca dengan jelas. Selain itu, pengujian dilakukan terhadap kemampuan sistem dalam merespons perubahan jadwal yang dikirim dari aplikasi secara cepat dan tepat.

Hasil dari pengujian ini digunakan untuk mengevaluasi performa sistem secara keseluruhan serta sebagai dasar dalam melakukan perbaikan atau penyesuaian jika ditemukan kendala teknis. Dengan pengujian yang menyeluruh, sistem bel otomatis berbasis IOT ini diharapkan dapat berfungsi secara handal dalam lingkungan operasional sekolah.

BAB IV
PEMBAHASAN

4.1. Desain Sistem

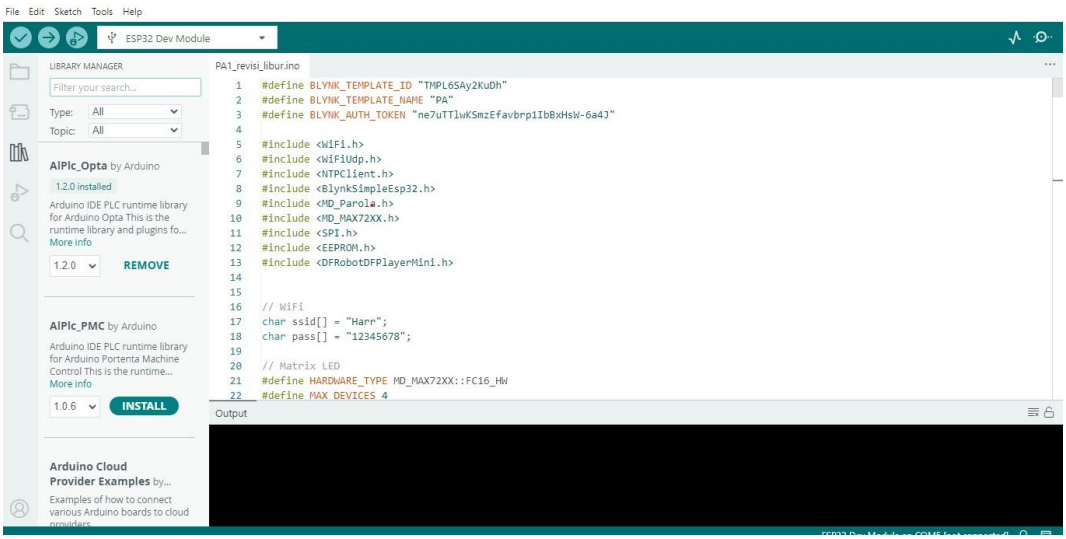
Desain Sistem Bel dan Jadwal Pelajaran di Sekolah Berbasis *IoT* ini melibatkan dua komponen utama, yaitu perangkat keras (*hardware*) dan perangkat lunak (*software*). Sistem ini menggunakan *mikrokontroler* ESP32 sebagai pengendali utama, modul *DFPlayer Mini* untuk memutar suara bel dan *Blynk* sebagai antarmuka pengguna yang berbasis web dan *mobile*. Seperti pada gambar 4.1. dibawah ini:



Gambar 4.1. Desain Perangkat Keras

Perangkat keras terdiri dari beberapa komponen, yaitu:

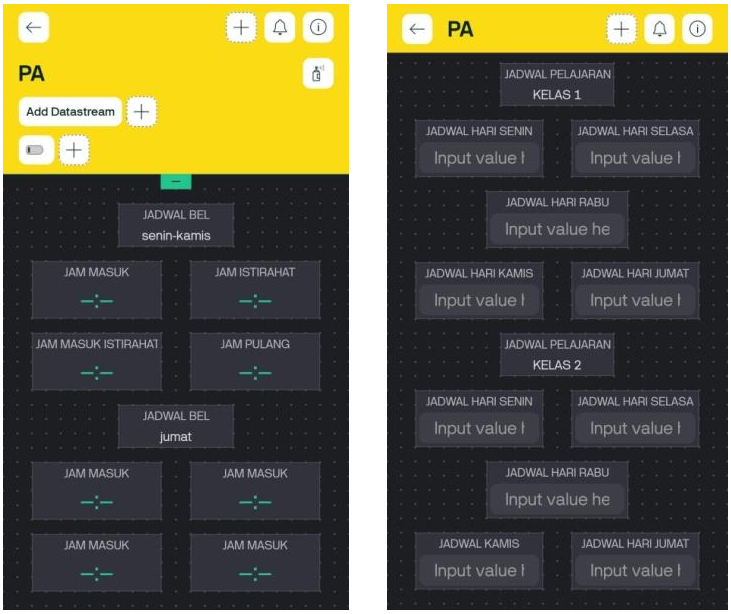
1. Esp32: *Mikrokontroler* utama untuk mengendalikan perangkat, mengakses internet, mengatur waktu, dan terhubung ke *Blynk* secara nirkabel.
2. *Led Dot Metrix*: Menampilkan informasi jadwal pelajaran, jam digital, atau notifikasi bel sekolah secara visual dan *real-time*.
3. *DFPlayer Mini*: Memutar file audio MP3 dari *microSD* sebagai suara bel otomatis sesuai jadwal pelajaran yang telah ditentukan.
4. PAM8403: *Amplifier* audio kecil yang memperkuat sinyal dari *DFPlayer* ke *Speaker* agar suara bel terdengar jelas dan nyaring.
5. *Speaker*: Menghasilkan *output* suara dari *DFPlayer*, berfungsi untuk menyuarakan bel otomatis pada pergantian jam pelajaran.
6. *Blynk*: Aplikasi *IoT* yang digunakan untuk mengatur dan mengubah jadwal bel secara remote melalui smartphone dengan koneksi internet.
7. Arduino IDE: Lingkungan pemrograman untuk menulis, mengunggah, dan *debugging* kode ESP32 yang mengatur seluruh sistem bel otomatis.



Gambar 4.1. Desain Perangkat Lunak

Perangkat lunak terdiri dari dua bagian yaitu:

1. *Nodemcu* ESP32 menjalankan *firmware* yang terhubung ke WiFi dan *Blynk*. Setelah mendapatkan waktu aktual dari NTP, ESP32 mencocokkan waktu tersebut dengan jadwal yang dikirim dari aplikasi *Blynk*. Jika waktunya sesuai, ESP32 akan menampilkan informasi pada *LED Dot Matrix* dan memicu *DFPlayer* untuk memutar suara bel melalui *Speaker*.
2. Aplikasi *Blynk* berfungsi sebagai antarmuka pengguna berbasis *IoT*. Melalui *Blynk*, pengguna dapat memasukkan, mengubah, atau menghapus jadwal bel sekolah secara *real-time*. Data dikirim langsung ke ESP32, memungkinkan kontrol dan pembaruan sistem tanpa perlu memprogram ulang perangkat.



Gambar 4.2. Antarmuka Aplikasi Web/Mobile

Blynk yang terdapat pada gambar 4.3. adalah platform *IoT* berbasis *cloud* yang digunakan untuk mengontrol dan memantau perangkat seperti ESP32 melalui aplikasi di *smartphone*. Dalam penelitian ini, *Blynk* berperan sebagai antarmuka pengguna (*user interface*) untuk mengatur jadwal bel otomatis sekolah secara fleksibel dan *real-time*.

Pengguna dapat menggunakan *widget Time Input* pada aplikasi *Blynk* untuk mengatur waktu bel masuk, istirahat, dan pulang. Jadwal yang telah ditentukan akan dikirim ke ESP32 melalui koneksi internet menggunakan server *Blynk*. Data tersebut kemudian disimpan dan dibandingkan oleh ESP32 dengan waktu aktual yang diperoleh melalui server NTP.

Kelebihan penggunaan *Blynk* adalah kemudahannya dalam mengubah jadwal tanpa perlu memprogram ulang *mikrokontroler*. Misalnya, saat terdapat perubahan jam pelajaran, pengguna cukup membuka aplikasi *Blynk*, mengatur waktu baru, dan sistem langsung diperbarui secara otomatis.

4.2. Hasil Pengujian

Pada tahap pengujian, seluruh sistem diuji dengan mengatur jadwal bel secara manual melalui aplikasi *blynk*. Pengujian ini bertujuan untuk memeriksa integrasi antara perangkat keras dan perangkat lunak, serta memastikan bahwa semua komponen dapat berfungsi dengan baik. Berikut berberapa pengujian yang dilakukan antara lain:

1. Pengaturan jadwal melalui *Blynk*: Aplikasi *Blynk* mampu mengirim data jadwal ke ESP32 dengan baik. Saat pengguna memasukkan atau mengubah jadwal bel melalui widget *Time Input*, data langsung dikirim ke ESP32 dan tersimpan dengan benar. Tidak ditemukan kendala koneksi selama ESP32 terhubung ke jaringan WiFi.
2. Pemutaran bel otomatis sesuai jadwal: Setelah jadwal diterima dari *Blynk*, ESP32 mencocokkannya dengan waktu aktual dari NTP. Saat waktu sesuai, *DFPlayer Mini* secara otomatis memutar file audio MP3 melalui PAM8403 dan *Speaker*. Suara bel terdengar dengan jelas, tanpa keterlambatan atau kesalahan waktu.
3. Tampilan waktu pada *LED dot matrix*: ESP32 berhasil menampilkan jam digital secara *real-time* pada *LED dot matrix*. Tampilan berjalan stabil dan dapat dibaca dengan jelas. Informasi juga dapat diperbarui jika

diperlukan, dan tidak terjadi flicker atau gangguan pada tampilan selama sistem aktif.

4.3 Pembahasan

A. Kelebihan Sistem

1. Pengaturan jadwal fleksibel dan jarak jauh: Dengan aplikasi *Blynk*, pengguna dapat mengatur atau mengubah jadwal bel dari mana saja secara *real-time* tanpa harus memprogram ulang perangkat.
2. Otomatisasi bel tanpa RTC eksternal: Sistem menggunakan waktu internet (NTP) sehingga tidak memerlukan modul RTC tambahan, membuat rangkaian lebih sederhana dan biaya lebih efisien.
3. Tampilan jam dan informasi *real-time*: *LED dot matrix* menampilkan jam digital yang terus diperbarui secara otomatis dan stabil, sehingga pengguna dapat memantau waktu saat ini dengan jelas.

B. Kekurangan dan Tantangan

1. Ketergantungan pada koneksi internet dan waktu NTP: Sistem sangat bergantung pada koneksi WiFi untuk mengakses aplikasi *Blynk* dan sinkronisasi waktu dari server NTP. Jika koneksi terputus, jadwal tidak dapat diperbarui dan waktu tidak tersinkron, sehingga bel otomatis tidak berfungsi dengan akurat.
2. Gangguan kinerja karena manajemen daya yang belum optimal: Penggunaan *LED dot matrix* dan *DFPlayer Mini* secara bersamaan menyebabkan konsumsi daya meningkat, yang memicu distorsi suara atau penurunan performa. Tantangan ini memerlukan penambahan kapasitor dan pengaturan catu daya yang stabil agar semua komponen dapat bekerja secara bersamaan tanpa gangguan.
3. Minimnya sistem monitoring dan notifikasi kesalahan: Sistem belum dilengkapi fitur notifikasi apabila terjadi kesalahan seperti gagalnya pemutaran bel, gangguan koneksi, atau sinkronisasi waktu.

BAB V
KESIMPULAN DAN SARAN

5.1. Kesimpulan

Sistem bel otomatis ini dirancang untuk mampu beroperasi secara mandiri tanpa memerlukan intervensi manual dari pengguna setiap hari. Proses penjadwalan dilakukan melalui aplikasi *Blynk* yang terhubung dengan *mikrokontroler* ESP32 sebagai unit pengendali utama. Dengan pengaturan waktu yang tersimpan dalam sistem dan dukungan koneksi internet, perangkat dapat mengaktifkan bel secara otomatis sesuai dengan jadwal yang telah ditentukan. Hal ini memungkinkan sistem bekerja secara konsisten, efisien, dan fleksibel dalam berbagai kondisi penggunaan, menjadikannya solusi ideal untuk lingkungan sekolah atau institusi lain yang memerlukan pengingat waktu terjadwal secara otomatis.

Dalam perancangan sistem bel dan jadwal pelajaran di sekolah berbasis IoT ini, sejumlah komponen penting seperti ESP32 sebagai pusat kendali dan penghubung ke internet, *DFPlayer Mini* untuk memutar file audio sebagai suara bel, *Speaker* sebagai *output* suara dari *DFPlayer Mini*, *LED Dot Matrix* untuk menampilkan waktu dan informasi visual lainnya aplikasi *Blynk* untuk Mengatur dan memonitor sistem dari jarak jauh, Arduino IDE digunakan untuk menulis dan mengunggah program ke ESP32.

Hasil pengujian menunjukkan bahwa sistem bekerja secara efektif dan akurat tanpa adanya kesalahan. Meskipun tidak menggunakan modul RTC (*Real Time Clock*), sistem tetap dapat menjalankan fungsi waktunya dengan akurat melalui sinkronisasi internet menggunakan NTP. Bel berbunyi sesuai jadwal yang ditentukan dan informasi waktu tampil dengan tepat melalui *LED dot matrix*.

Seluruh komponen saling terintegrasi dengan baik dan menunjukkan kestabilan selama pengoperasian. Sistem bel otomatis berbasis ESP32 berhasil dirancang tanpa menggunakan modul RTC, namun tetap akurat karena sinkronisasi waktu melalui internet. Seluruh komponen seperti *DFPlayer Mini*, *Speaker*, *LED dot matrix*, dan *Blynk* berfungsi stabil. Sistem ini efektif, berjalan mandiri, dan layak diterapkan dalam lingkungan sekolah.

5.2 Saran

Berdasarkan hasil penelitian dan implementasi sistem bel otomatis berbasis IoT**,** berikut adalah beberapa saran yang dapat memperbaiki dan mengembangkan sistem lebih lanjut:

1. **Perawatan Komponen Secara Berkala**

   Komponen seperti *LED dot matrix*, *DFPlayer Mini*, *Speaker*, dan koneksi kabel perlu dirawat secara berkala untuk menjaga performanya tetap optimal. Perawatan dapat dilakukan dengan membersihkan debu yang menempel, mengecek kekencangan kabel *jumper*, serta memastikan tidak ada korosi pada konektor. Suhu lingkungan juga perlu diperhatikan agar tidak terjadi overheat, terutama pada ESP32 dan *DFPlayer Mini*. *Speaker* yang digunakan sebaiknya diuji secara berkala agar suara bel tetap terdengar jernih. Dengan perawatan yang tepat dan rutin, sistem akan lebih tahan lama, stabil, dan tidak mengalami gangguan saat digunakan dalam kegiatan belajar mengajar.

2. **Penggunaan Catu Daya Stabil dan Aman**

   Kestabilan suplai daya sangat berpengaruh terhadap kinerja sistem. Disarankan untuk menggunakan *adaptor* yang memiliki arus dan tegangan yang sesuai dengan kebutuhan ESP32 dan modul lainnya, misalnya *adaptor* 5V 2A dengan kualitas yang baik. Selain itu, sebaiknya digunakan modul proteksi seperti *fuse*, *diode*, atau modul *step-downregulator* dengan *filter kapasitor* untuk menghindari lonjakan tegangan yang dapat merusak

komponen. Gangguan listrik seperti *noise* atau *drop voltage* dapat menyebabkan bel tidak berbunyi tepat waktu atau bahkan sistem mengalami restart. Oleh karena itu, penggunaan *power supply* yang andal menjadi salah satu aspek penting dalam menjaga keandalan sistem.

3. Penyediaan Panduan Operasional Sistem

Sistem ini sebaiknya dilengkapi dengan buku panduan atau petunjuk penggunaan singkat yang menjelaskan langkah-langkah pengoperasian, pengaturan jadwal melalui aplikasi *Blynk*, serta solusi saat terjadi kendala umum. Panduan ini berguna bagi pengguna non-teknis seperti guru atau staf sekolah agar dapat mengelola sistem tanpa kesulitan. Penyediaan panduan juga membantu dalam proses serah terima sistem ke pihak pengguna akhir. Panduan sebaiknya mencakup ilustrasi, tampilan aplikasi, penjelasan fungsi tombol, dan prosedur perawatan. Dengan adanya dokumentasi tersebut, sistem akan lebih mudah diadopsi, digunakan secara maksimal, dan tidak bergantung terus-menerus pada pengembang.

4. Pengembangan Antarmuka Aplikasi *Blynk*

Aplikasi *Blynk* sudah memudahkan pengguna dalam mengatur jadwal bel secara jarak jauh, namun antarmukanya masih dapat dikembangkan agar lebih intuitif dan ramah pengguna. Penambahan label yang jelas, ikon waktu, status koneksi, serta fitur notifikasi dapat membantu pengguna dalam memantau dan mengatur sistem dengan lebih baik. Misalnya, pengguna bisa diberikan notifikasi saat bel berbunyi atau ketika koneksi internet terputus. Selain itu, visualisasi jadwal dalam bentuk kalender atau daftar akan meningkatkan pengalaman pengguna. Dengan antarmuka yang lebih baik, sistem tidak hanya fungsional tetapi juga lebih mudah digunakan oleh siapa saja, termasuk pengguna tanpa latar belakang teknis.

DATAR PUSTAKA

[1]Imran, Muhammad Ali; Fauzi, Achmad; Khair, Husnul. Rancang Bangun Kontrol Bel Otomatis Berdasarkan Jadwal Perkuliahan Menggunakan Internet of Things (IoT). *Modem: Jurnal Informatika dan Sains Teknologi.*, 2024, 2.4: 21-32.

[2]Agung, Raka; Janardana, Ngurah; Ardiansyah, Ferry. Rancang Bangun Bel Sekolah Otomatis Berbasis Mikrokontroler AVR ATMEGA8. *Teknologi Elektro*, 2011, 10.2: 11-17.

[3]Hidayati, Qory; Aziz, Ahmad Nurul. Rancang Bangun Bel Otomatis Berbasis RTC DS3231 Menggunakan Arduino Uno R3 Sebagai Tanda Pergantian Jadwal. *JREC (Journal of Electrical and Electronics)*, 2018, 6.1: 1-8.

[4]Pauzan, Muh; Yanti, Indri. Bel sekolah otomatis berbasis Arduino yang dikontrol menggunakan aplikasi mobile. *Jurnal Nasional Teknik Elektro dan Teknologi Informasi| Vol*, 2021, 10.2.

[5]Juwita, Rena Amelia. *Rancang Bangun Bel Sekolah Otomatis Berbasis Arduino ESP8266*. 2024. PhD Thesis. STMIK Widya Cipta Dharma.

[6]Hutasuhut, Indah Khairunnisa Ahmad. Prototype Smart Alarm Automated System Berbasis DFPlayer Mini untuk Mengefisiensikan Jadwal Waktu. *Jurnal Teknik Informatika*, 2023, 9.2: 34-41.

[7]Hasby, Fachrul, et al. Rancang Bangun Bel Otomatis Di STIKOM Tunas Bangsa Berbasis Arduino Dilengkapi Dengan Output Suara. *Jurnal Teknologi Informasi Dan Komunikasi*, 2021, 12.2: 58-68.

[8]Fajar Taufik, Agustian. *Sistem Penjadwalan Bel Sekolah Otomatis Berbasis Internet Of Things (IOT) Menggunakan Mobile App*. 2021. PhD Thesis. Nusa Putra University.

[9]Patriya, Panji. Rancang bangun jam digital dengan kelengkapan alarm bel sekolah otomatis berbasis mikrokontroler AT89S52. 2007.

[10]Afriza, Wahyu Dwi Mohamad. *Rancang Bangun Sistem Kontrol Running Text Dari Jarak Jauh Menggunakan Nodemcu ESP8266*. 2024. Phd Thesis. Universitas Muhammadiyah Ponorogo.

[11]Suardin, Alfiah; Elviralita, Yoan; Fauziah, Fauziah. Sistem Perancangan Alarm Otomatis Menggunakan Sirine Berbasis Arduino. *Mechatronics Journal In Professional And Entrepreneur (MAPLE)*, 2023, 5.2: 60-65.

[12]Khaerudin, Muhammad; Warta, Joni; Mahbub, Asep Ramdhani. Mikrokontroler Untuk Sistem Penjadwalan Penyiraman Otomatis Tanaman Aeroponik Pada Kebun Hidroponik. *IKRAM: Jurnal Ilmu Komputer Al Muslim*, 2025, 4.1: 47-53.

[13]Zikri[1], Mhd Zahir Az; Apdillah, Dicky. Tren Terkini Dalam Pengembangan IOT (Internet Of Things) Dan Mobile Integration.

[14]Asyiah, Nilovar. Perancangan Sistem Bel Otomatis Dan Informasi Waktu Belajar Di Sekolah Berbasis Internet Of Things:(Studi Kasus: SMK Bina Mandiri). *Spectrum: Multidisciplinary Journal*, 2024, 1.3: 169-178.

[15]Susilo, Dody; Laksono, Ridam Dwi; Ardiansyah, Yovie Eri. Rancang Bangun Sistem Bel Sekolah Otomatis Berbasis Mikrokontroller Menggunakan ISD 4003. *ELECTRA Electr. Eng. Artic*, 2022, 2.2: 12.

LAMPIRAN

Lampiran 1 : Daftar Riwayat Hidup

Nama : Ariza Prasetya

Tempat, Tanggal Lahir: Sungailiat, 28 November 2003

Alamat : Jl.Batin Tikal No.09 RT 02 Desa Air RuayKecamatan Pemali

Jenis Kelamin : Laki-laki

Agama : Islam

Status : Mahasiswa

Email : arizaprasetya028@gmail.com

No. Telepon : 083170754816

Nama : Hardiansyah

Tempat, Tanggal Lahir : Mengkubang, 12 Juli 2004

Alamat : DSN Damai Baru

Jenis Kelamin : Laki-laki

Agama : Islam

Status : Mahasiswa

Email : hardiansyah56473@gmail.com

No. Telepon : 083892451595

Lampiran 2 : Program Softwere Arduino IDE

```
#define BLYNK_TEMPLATE_ID "TMPL6SAy2KuDh"

#define BLYNK_TEMPLATE_NAME "PA"

#define BLYNK_AUTH_TOKEN "ne7uTTlwKSmzEfavbrp1IbBxHsW-6a4J"


#include <WiFi.h>

#include <WiFiUdp.h>

#include <NTPClient.h>

#include <BlynkSimpleEsp32.h>

#include <MD_Parola.h>

#include <MD_MAX72XX.h>

#include <SPI.h>

#include <EEPROM.h>

#include <DFRobotDFPlayerMini.h>



// WiFi

char ssid[] = "Harr";

char pass[] = "12345678";
```

```
// Matrix LED
#define HARDWARE_TYPE MD_MAX72XX::FC16_HW
#define MAX_DEVICES 4
#define DATA_PIN 23
#define CLK_PIN  18
#define CS1 5   // Matrix Guru
#define CS2 19  // Matrix Kelas 1
#define CS3 21  // Matrix Kelas 2

MD_Parola matrixGuru(HARDWARE_TYPE, CS1, MAX_DEVICES);
MD_Parola matrixK1(HARDWARE_TYPE, CS2, MAX_DEVICES);
MD_Parola matrixK2(HARDWARE_TYPE, CS3, MAX_DEVICES);

bool tampilkanPesanBelGuru = false;
bool tampilkanPesanBelK1 = false;
bool tampilkanPesanBelK2 = false;

String pesanBelGuru = "";
String pesanBelK1 = "";
String pesanBelK2 = "";
```

```
unsigned long waktuPesanBelGuru = 0;
unsigned long waktuPesanBelK1 = 0;
unsigned long waktuPesanBelK2 = 0;
const unsigned long durasiPesanBel = 6000; // 10 detik


int volumeDF = 15; // default volume DFPlayer

// Timer dan NTP
BlynkTimer timer;
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "pool.ntp.org", 25200); // UTC+7

// DFPlayer
HardwareSerial mySerial(1); // RX, TX
DFRobotDFPlayerMini dfplayer;

// EEPROM
#define EEPROM_SIZE 1024
int timeInputEEPROM[8]; // 8 jadwal, disimpan sebagai jam*100+menit
```

```
// Flag bel harian
bool belHarian[8] = {false};

// Jadwal
String jadwalK1[5] = {
  "IPS (07:00-09:00), IPA (09:15-10:00), B.Inggris (10:00-11:30)",
  "PENJAS (07:00-09:00), SENI (09:15-10:00), B.Indonesia (10:00-11:30)",
  "PENJAS (07:00-09:00), SENI (09:15-10:00), B.Indonesia (10:00-11:30)",
  "PENJAS (07:00-09:00), SENI (09:15-10:00), B.Indonesia (10:00-11:30)",
  "PENJAS (07:00-09:00), SENI (09:15-10:00), B.Indonesia (10:00-11:30)"
};
String jadwalK2[5] = {
  "B.Indonesia (07:00-09:00), Matematika (09:15-10:00), IPA (10:00-11:30)",
  "B.Inggris (07:00-09:00), IPA (09:15-10:00), IPS (10:00-11:30)",
  "PENJAS (07:00-09:00), SENI (09:15-10:00), B.Indonesia (10:00-11:30)",
  "PENJAS (07:00-09:00), SENI (09:15-10:00), B.Indonesia (10:00-11:30)",
  "PENJAS (07:00-09:00), SENI (09:15-10:00), B.Indonesia (10:00-11:30)"
};

// Mode tampilan
enum Mode { TAMPIL_JADWAL, TAMPIL_JAM };
```

```
Mode modeGuru = TAMPIL_JADWAL;
Mode modeK1 = TAMPIL_JADWAL;
Mode modeK2 = TAMPIL_JADWAL;

bool matrixGuruDone = false;
bool matrixK1Done = false;
bool matrixK2Done = false;

// Bel ke-1 (Masuk), ke-2 (Istirahat), ke-3 (Masuk lagi), ke-4 (Pulang)
uint8_t suaraPerJadwal[4] = {1, 2, 3, 4}; // Mengacu pada 0001.mp3 dst


// Variabel jam per matrix
unsigned long lastUpdateGuru = 0;
unsigned long lastUpdateK1 = 0;
unsigned long lastUpdateK2 = 0;

char jamStrGuru[6];
char jamStrK1[6];
char jamStrK2[6];
```

```
void setup() {
  Serial.begin(115200);
  mySerial.begin(9600, SERIAL_8N1, 16, 17);
  EEPROM.begin(EEPROM_SIZE);
  WiFi.begin(ssid, pass);
  while (WiFi.status() != WL_CONNECTED) {
    delay(300); Serial.print(".");
  }

  Blynk.config(BLYNK_AUTH_TOKEN);
  Blynk.connect();
  timeClient.begin();
  mySerial.begin(9600);
  dfplayer.begin(mySerial);
  dfplayer.volume(volumeDF);

  for (auto& m : { &matrixGuru, &matrixK1, &matrixK2 }) {
    m->begin(); m->setIntensity(3); m->displayClear();
  }

  timer.setInterval(17000L, toggleGuru);
```

```
  timer.setInterval(10000L, toggleK1);
  timer.setInterval(10000L, toggleK2);
  timer.setInterval(1000L, cekBel);

  // Baca jadwal dari EEPROM
for (int i = 0; i < 5; i++) {
  jadwalK1[i] = bacaStringEEPROM(i * 100);
  jadwalK2[i] = bacaStringEEPROM((i + 5) * 100);
}


  for (int i = 0; i < 8; i++) {
    EEPROM.get(i * 2, timeInputEEPROM[i]);
  }
}

void tampilkanJikaSelesai(MD_Parola &matrix, const String &text, bool &doneFlag, bool pesanBelAktif = false) {
  if (pesanBelAktif) {
    // Paksa hentikan animasi sebelumnya dan langsung tampilkan pesan bel
    if (!doneFlag) {
```

```
      matrix.displayClear();
      matrix.displayText(text.c_str(), PA_CENTER, 100, 0, PA_SCROLL_LEFT, PA_SCROLL_LEFT);
      matrix.displayReset();
      doneFlag = true;
    }
  } else {
    // Normal: tunggu animasi selesai baru reset
    if (!doneFlag && matrix.displayAnimate()) {
      matrix.displayText(text.c_str(), PA_CENTER, 100, 0, PA_SCROLL_LEFT, PA_SCROLL_LEFT);
      matrix.displayReset();
      doneFlag = true;
    }
  }
}



void loop() {
  Blynk.run();
  timer.run();
```

```
 timeClient.update();


 int day = timeClient.getDay();
 int idx = day - 1;
 bool isLibur = (day == 0 || day == 6);


  if (isLibur) {
   matrixGuruDone = false;
   matrixK1Done = false;
   matrixK2Done = false;
  tampilkanJikaSelesai(matrixGuru, "LIBUR", matrixGuruDone);
  tampilkanJikaSelesai(matrixK1, "LIBUR", matrixK1Done);
  tampilkanJikaSelesai(matrixK2, "LIBUR", matrixK2Done);
 } else {
  unsigned long nowMillis = millis();


     // Guru
if (tampilkanPesanBelGuru && nowMillis - waktuPesanBelGuru <
durasiPesanBel) {
 tampilkanJikaSelesai(matrixGuru, pesanBelGuru, matrixGuruDone, true);
} else {
```

```
  tampilkanPesanBelGuru = false;
  if (modeGuru == TAMPIL_JADWAL) {
    String semuaJadwal = "K1: " + jadwalK1[idx] + " | K2: " + jadwalK2[idx];
    tampilkanJikaSelesai(matrixGuru, semuaJadwal, matrixGuruDone);
  } else tampilkanJam(matrixGuru, matrixGuruDone, lastUpdateGuru, jamStrGuru);
}

if (tampilkanPesanBelK1 && nowMillis - waktuPesanBelK1 < durasiPesanBel) {
  tampilkanJikaSelesai(matrixK1, pesanBelK1, matrixK1Done, true);
} else {
  tampilkanPesanBelK1 = false;
  if (modeK1 == TAMPIL_JADWAL) tampilkanJikaSelesai(matrixK1, jadwalK1[idx], matrixK1Done);
  else tampilkanJam(matrixK1, matrixK1Done, lastUpdateK1, jamStrK1);
}

if (tampilkanPesanBelK2 && nowMillis - waktuPesanBelK2 < durasiPesanBel) {
  tampilkanJikaSelesai(matrixK2, pesanBelK2, matrixK2Done, true);
} else {
  tampilkanPesanBelK2 = false;
```

```
  if (modeK2 == TAMPIL_JADWAL) tampilkanJikaSelesai(matrixK2, jadwalK2[idx], matrixK2Done);
  else tampilkanJam(matrixK2, matrixK2Done, lastUpdateK2, jamStrK2);
}
  }
  matrixGuru.displayAnimate();
  matrixK1.displayAnimate();
  matrixK2.displayAnimate();
}


void toggleGuru() {
  if (matrixGuru.displayAnimate()) {
    matrixGuruDone = false;
    modeGuru = (modeGuru == TAMPIL_JADWAL) ? TAMPIL_JAM : TAMPIL_JADWAL;
  }
}
void toggleK1() {
  if (matrixK1.displayAnimate()) {
    matrixK1Done = false;
```

```
  modeK1 = (modeK1 == TAMPIL_JADWAL) ? TAMPIL_JAM : TAMPIL_JADWAL;
 }
}
void toggleK2() {
 if (matrixK2.displayAnimate()) {
  matrixK2Done = false;
  modeK2 = (modeK2 == TAMPIL_JADWAL) ? TAMPIL_JAM : TAMPIL_JADWAL;
 }
}

void tampilkanJam(MD_Parola &matrix, bool &doneFlag, unsigned long &lastUpdate, char jamStr[]) {
 sprintf(jamStr, "%02d:%02d", timeClient.getHours(), timeClient.getMinutes());
 if (!doneFlag && matrix.displayAnimate()) {
  matrix.displayText(jamStr, PA_CENTER, 150, 0, PA_SCROLL_LEFT, PA_SCROLL_LEFT);
  matrix.displayReset();
  doneFlag = true;
  lastUpdate = millis();
 }
```

```
}

void simpanStringEEPROM(int start, String data) {
  for (int i = 0; i < 100; i++) {
    char c = (i < data.length()) ? data[i] : '\0';
    EEPROM.write(start + i, c);
  }
  EEPROM.commit();
}

String bacaStringEEPROM(int start) {
  char buf[101];
  for (int i = 0; i < 100; i++) {
    buf[i] = EEPROM.read(start + i);
  }
  buf[100] = '\0';
  return String(buf);
}

void tampilkanSituasiBel(String teks) {
```

```
  for (auto& matrix : { &matrixGuru, &matrixK1, &matrixK2 }) {
    matrix->displayClear();
    matrix->displayText(teks.c_str(), PA_CENTER, 100, 0, PA_SCROLL_LEFT,
PA_SCROLL_LEFT);
    matrix->displayReset();
  }

  // Set mode tampilan kembali ke JADWAL agar setelah itu balik normal
  modeGuru = TAMPIL_JADWAL;
  modeK1 = TAMPIL_JADWAL;
  modeK2 = TAMPIL_JADWAL;

  // Reset flag supaya animasi tampil ulang
  matrixGuruDone = false;
  matrixK1Done = false;
  matrixK2Done = false;
}

// Fungsi cek bel dan mainkan suara sesuai waktu
void cekBel() {
  int day = timeClient.getDay();  // 0 = Minggu, 6 = Sabtu
```

```
if (day == 0 || day == 6) return;  // Lewati jika hari libur (Minggu atau Sabtu)

int now = timeClient.getHours() * 100 + timeClient.getMinutes();
for (int i = 0; i < 8; i++) {
  if (now == timeInputEEPROM[i] && !belHarian[i]) {
    belHarian[i] = true;
    uint8_t suara = suaraPerJadwal[i % 4];
    dfplayer.volume(volumeDF);
    dfplayer.play(suara);

    // Tentukan pesan teks berdasarkan jenis bel
    String pesan = "";
    switch (suara) {
      case 1: pesan = "MASUK"; break;
      case 2: pesan = "ISTIRAHAT"; break;
      case 3: pesan = "MASUK KEMBALI"; break;
      case 4: pesan = "PULANG"; break;
      default: pesan = "BEL"; break;
    }

    tampilkanPesanBelGuru = true;
```

```
      tampilkanPesanBelK1 = true;
      tampilkanPesanBelK2 = true;
      pesanBelGuru = pesan;
      pesanBelK1 = pesan;
      pesanBelK2 = pesan;

      waktuPesanBelGuru = millis();
      waktuPesanBelK1 = millis();
      waktuPesanBelK2 = millis();

      matrixGuruDone = false;
      matrixK1Done = false;
      matrixK2Done = false;
    }

    if (now != timeInputEEPROM[i]) belHarian[i] = false;
  }
}

// Input Time dari Blynk (Senin–Kamis V20–V23, Jumat V24–V27)
BLYNK_WRITE(V20){ setTimeEEPROM(0, param); }
```

```
BLYNK_WRITE(V21){ setTimeEEPROM(1, param); }
BLYNK_WRITE(V22){ setTimeEEPROM(2, param); }
BLYNK_WRITE(V23){ setTimeEEPROM(3, param); }
BLYNK_WRITE(V24){ setTimeEEPROM(4, param); }
BLYNK_WRITE(V25){ setTimeEEPROM(5, param); }
BLYNK_WRITE(V26){ setTimeEEPROM(6, param); }
BLYNK_WRITE(V27){ setTimeEEPROM(7, param); }
BLYNK_WRITE(V50) {
  volumeDF = param.asInt();
  dfplayer.volume(volumeDF);
}



void setTimeEEPROM(int idx, BlynkParam param) {
  TimeInputParam t(param);
  int waktu = t.getStartHour() * 100 + t.getStartMinute();
  timeInputEEPROM[idx] = waktu;
  EEPROM.put(idx * 2, waktu);
  EEPROM.commit();
}
```

```
// Input dari Blynk untuk jadwal kelas 1

BLYNK_WRITE(V28){ jadwalK1[0] = param.asStr(); simpanStringEEPROM(0
* 100, jadwalK1[0]); matrixK1Done = false; }

BLYNK_WRITE(V29){ jadwalK1[1] = param.asStr(); simpanStringEEPROM(1
* 100, jadwalK1[1]); matrixK1Done = false; }

BLYNK_WRITE(V1){ jadwalK1[2] = param.asStr(); simpanStringEEPROM(2 *
100, jadwalK1[2]); matrixK1Done = false; }

BLYNK_WRITE(V2) { jadwalK1[3] = param.asStr(); simpanStringEEPROM(3 *
100, jadwalK1[3]); matrixK1Done = false; }

BLYNK_WRITE(V6) { jadwalK1[4] = param.asStr(); simpanStringEEPROM(4 *
100, jadwalK1[4]); matrixK1Done = false; }


// Input dari Blynk untuk jadwal kelas 2

BLYNK_WRITE(V11) { jadwalK2[0] = param.asStr(); simpanStringEEPROM(5
* 100, jadwalK2[0]); matrixK2Done = false; }

BLYNK_WRITE(V12) { jadwalK2[1] = param.asStr(); simpanStringEEPROM(6
* 100, jadwalK2[1]); matrixK2Done = false; }

BLYNK_WRITE(V13) { jadwalK2[2] = param.asStr(); simpanStringEEPROM(7
* 100, jadwalK2[2]); matrixK2Done = false; }

BLYNK_WRITE(V7) { jadwalK2[3] = param.asStr(); simpanStringEEPROM(8 *
100, jadwalK2[3]); matrixK2Done = false; }

BLYNK_WRITE(V10){ jadwalK2[4] = param.asStr(); simpanStringEEPROM(9
* 100, jadwalK2[4]); matrixK2Done = false; }
```