BAB I

PENDAHULUAN

A. Latar Belakang

Pembangkit listrik tenaga surya (PLTS) merupakan sumber energi listrik yang tidak akan pernah habis dan tidak merusak lingkungan. Komponen penyusun PLTS yaitu panel surya sebagai perangkat yang dapat mengubah cahaya matahari menjadi energi listrik, kontroller untuk mengontrol arus ke baterai, baterai sebagai tempat penyimpanan daya dan *inverter* sebagai pengubah arus DC menjadi arus AC (Rauf, R. 2023).PLTS merupakan sumber energi alternatif yang bisa digunakan saat suatu daerah belum terjangkau pasokan listrik atau belum memiliki sumber pembangkit listrik yang umum digunakan seperti gas atau air (Melly, W. A. 2024) . Pemanfaatan energi terbarukan, seperti Pembangkit Listrik Tenaga Surya (PLTS), telah menjadi fokus utama dalam upaya mengurangi ketergantungan pada sumber energi fosil yang semakin terbatas dan mereduksi dampak negatif terhadap lingkungan (Darmana, T., Pambudi 2024). PLTS merupakan salah satu teknologi yang menjanjikan untuk memenuhi kebutuhan energi dengan cara yang berkelanjutan dan ramah lingkungan. Dalam konteks skala laboratorium, pengembangan sistem kendali yang efisien dan terintegrasi menjadi krusial untuk meningkatkan kinerja dan pengelolaan PLTS (Mahendra, G. S 2024).Teknologi *Internet of Things*(IOT) menawarkan potensi besar untuk meningkatkan kinerja pemantauan pembangkit listrik tenaga surya(PLTS) dengan

internet of things(IOT), Berbagai perangkat dapat terhubung dan berkomunikasi satu sama lain memunkingkan pemantauan dan pegendalian secara *real time.* Integrasi *Internet of things* (IOT) dalam sistem kendali Pembangkit listrik tenaga surya (PLTS) skala laboratoruim dapat membantu megoptimalkan penggunaan energi, medeteksi dan mendiagnosis masalah operasional lebih cepat, serta menigkatkan efesiensi secara keseluruhan pemanfaatan energi terbarukan, seperti Pembangkit Listrik Tenaga Surya (PLTS). Pembangkit listrik tenaga surya (PLTS) merupakan salah satu teknologi yang menjanjikan untuk memenuhi kebutuhan energi dengan cara yang berkelanjutan dan ramah lingkungan (Layela, G. I. 2023). Dalam konteks skala laboratorium, pengembangan sistem kendali yang efisien dan terintegrasi menjadi krusial untuk meningkatkan kinerja pegelolaan pembangkit listrik tenaga surya (PLTS) (David, D. 2024). Sistem kendali pembangkit listrik tenaga surya skala lab berbasis *Internet of thing* adalah solusi modern untuk meningkatkan efisiensi dan efektivitas pengelolaan energi surya di laboratorium. Sistem ini memungkinkan pemantauan dan pengendalian yang real-time, mempermudah analisis data, dan mendukung pengambilan keputusan yang lebih cepat dan akurat dalam manajemen energi(Mahendra, G. S., 20024). Penggunaan *Internet of thing* diharapkan dapat memperbaiki proses pengumpulan data, monitoring, dan kendali secara signifikan dibandingkan dengan sistem tradisional. Penggunaan *Internet of thing* dalam kendali pembangkit listrik tenaga surya memungkinkan pemantauan dan pengendalian yang lebih terpusat dan terotomatisasi, meningkatkan efisiensi dan kestabilan sistem secara keseluruhan pengawasan dan analisis performa panel surya menjadi lebih mudah dan akurat.

Sistem ini diharapkan dapat memberikan data *real-time* yang penting untuk optimalisasi dan pemeliharaan panel surya.

B. Rumusan Masalah

Bagaimana merancang arsitektur sistem yang dapat mengendalikan pengisian baterai menggunakan aplikasi iot?

C. Tujuan Masalah

Untuk mengetahui arsitektur sistem yang dapat mengendalikan pengisian baterai menggunakan aplikasi iot.

D. Batasan Masalah

1. Sensor yang digunakan adalah Sensor tegangan.
2. Data yang akan di uji hanya pengendalian pengisian baterai menggunkan aplikasi iot.

E. Manfaat masalah

1. Memudahkan untuk mengamati pegisian daya pada baterai.
2. Dapat memantau kinerja sistem dari jarak jauh, sehingga memudahkan pengelolaan dan pengawasan tanpa harus berada di lokasi fisik panel surya.

BAB II

TINJAUAN PUSTAKA

A. Kajian Teori

Untuk membuat rancangan sistem kendali pembangkit listrik tenaga surya skala laboratorium berbasis *internet of thing* sebelumnya dapat dijelaskan tinjauan pustaka dan sistem yang terkait didalamnya yang dapat dijelaskan sebagai berikut

1. Panel Surya

Pembangkit listrik tenaga surya merupakan sistem yang mampu mengonversi energi dari cahaya matahari menjadi energi listrik melalui prinsip efek *fotovoltaik*. Energi listrik yang dihasilkan akan disimpan dalam baterai, lalu digunakan untuk mengoperasikan perangkat elektronik sesuai dengan kebutuhan daya listriknya(Muslim, S. 2020).



Gambar 2. 1 panel surya
sumber:(https://solarsuryaindotama.co.id)

Tabel 2. 1 spesifikasi panel surya

No	Spesifikasi	
1	Daya Puncak Maksimum (Pmax)	50w
2	Tegangan pada Daya Maksimum (Vmp)	17.5 - 18.2V
4	Arus pada Daya Maksimum (Imp)	2.75 - 2.85A
5	Tegangan Sirkuit Terbuka (Voc)	21.5 - 22.1V
6	Tipe Sel Surya	Polycrystalline
7	Dimensi (Panjang x Lebar x Tebal)	610-700 x 510-670 x 25-35mm
8	Berat	4.0 - 5.5KG

Secara sederhana, prinsip operasi panel surya melibatkan penyerapan cahaya matahari dan akumulasi energinya dalam sebuah baterai. Dengan demikian, sistem dapat beroperasi pada saat senja, malam hari, atau bahkan dalam kondisi hujan. Selain itu, terdapat kabel yang terhubung dengan sistem listrik di dalam rumah. Ketika terjadi konsumsi listrik oleh perangkat elektronik, seperti saat menghidupkan televisi, tenaga listrik secara otomatis diambil dari baterai(Rudiyanto, B 2023) . Adapun Fungsi atau manfaat panel surya adalah sebagai alat untuk mengubah energi matahari menjadi energi listrik. beberapa fungsi panel surya antara lain:

1. Konversi Energi Matahari

   Panel surya berfungsi untuk menangkap cahaya matahari dengan menggunakan sel *fotovoltaik.* Sel surya ini terbuatt dari bahan semikonduktor, seperti slikon, yang mampu mengubah energi matahari menjadi energi listrik melalui efek *fotovoltaik* (Muner, M. 2021).

2. Penyerapan Cahaya Matahari

Ketika sinar matahari mengenai panel surya, sel *fotovoltaik* menyerap foton dari cahaya tersebut. Proses ini merangsang pelepasan elektron dalam sel, menciptakan arus listrik.

3. Penghasilan Arus Listrik

   Proses penyerapan cahaya matahari dan pelepasan *elektron* menghasilkan arus listrik searah (DC) dalam panel surya. Arus inilah yang dapat digunakan sebagai sumber energi listrik.

4. Inversi Arus Listrik

   Arus listrik searah yang dihasilkan kemudian diubah menjadi arus bolak-balik (AC) oleh *inverter*. Arus bolak-balik adalah jenis arus listrik yang umumnya digunakan di rumah dan bisnis (Idris, M. 2019).

2. Skala Laboratorium

Skala laboratorium merujuk pada ukuran atau skala eksperimen yang dilakukan dalam lingkungan laboratorium untuk tujuan penelitian dan pengembangan. Pada skala ini, parameter dan variabel dapat dikendalikan dengan lebih baik untuk menguji hipotesis atau mengembangkan teknologi sebelum diaplikasikan dalam skala yang lebih besar.

3. Sistem Rangkaian pembaca tegangan

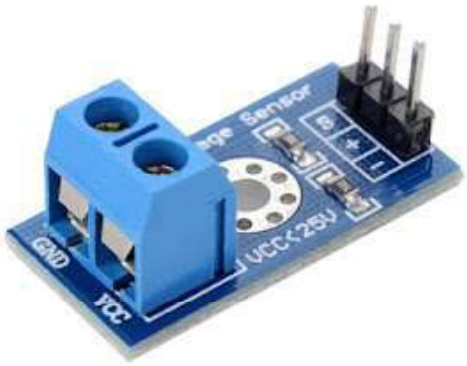
Rangkaian pembaca pembaca tegangan yang digunakan adalah Sensor tegangan. Sensor tegangan adalah modul yang dirancang untuk mengukur, memonitor, dan menghitung nilai tegangan pada suatu rangkaian elektronika. Sensor tegangan memiliki input berupa tegangan listrik yang diambil dari sumber

daya yang ingin diukur. Sementara itu, *output* dari sensor ini dapat bervariasi, meliputi *switch*, sinyal analog, atau modul alarm .

Cara kerja sensor tegangan dapat bervariasi tergantung pada jenis sensor tersebut dan tujuan penggunaannya. Berikut adalah penjelasan lebih lanjut tentang beberapa cara kerja sensor tegangan yang mencakup beberapa deteksi spesifik.

1. Medan magnet Sensor tegangan yang mendeteksi medan magnet bekerja dengan mengukur fluks magnet, arah, dan kekuatan medan magnet antara dua komponen. Sensor ini dapat digunakan dalam berbagai aplikasi, seperti peralatan navigasi, industri, atau pengukuran ilmiah. Jika sensor mendeteksi medan magnet yang lemah, peringatan dapat dikirimkan ke sistem manajemen pemeliharaan terkomputerisasi (CMMS) untuk memulai tindakan pemeliharaan yang diperlukan.
2. Medan elektromagnetik Sensor medan elektromagnetik mampu mendeteksi partikel bermuatan yang mengalami percepatan, memantau kekuatan gelombang elektromagnetik pada aset-aset kritis. Sama seperti sensor medan magnet, sensor ini juga digunakan dalam aplikasi navigasi, industri, dan ilmiah. Jika medan elektromagnetik melemah, sensor dapat memberikan peringatan kepada CMMS untuk melakukan pemeliharaan yang diperlukan.
3. Tegangan kontak Sensor tegangan kontak dirancang khusus untuk mengukur tegangan pada titik-titik kontak dalam berbagai aplikasi industri. Salah satu aplikasi umumnya adalah pemantauan baterai. Sensor ini dapat mendeteksi penurunan tegangan kontak, seperti saat baterai

terlepas dari tempatnya setelah beberapa waktu. Dengan mendeteksi perubahan ini, sensor akan mengirim peringatan ke CMMS, memungkinkan teknisi pemeliharaan untuk menindaklanjuti dan merestorasi kontak yang terputus (Pratama, M. F. 2021).



Gambar 2. 2 Sensor tegangan

Sumber:(https://www-datasheethub-com./0-to-25v-dc-voltage-sensor-module)

Tabel 2. 2 Datasheet sensor tegangan (0 to 25v voltage sensor)

Nama pin	Keterangan
VCC	Terminal positif sumber tegangan eksternal (0-25V)
GND	Terminal negatif dari sumber tegangan eksternal
S	Analog pin connected to Analog pin of Arduino
+	Tidak terhubung
_	Pin Ground terhubung ke GND Arduino

Sumber:(https://www-datasheethub-com./0-to-25v-dc-voltage-sensor-module)

Tabel 2. 3 Spesifikasi sensor tegangan (0 to 25v voltage sensor)

No	Spesifikasi	
1	Ukuran	28 x 14 x 13 mm (PxLxT)
2	Berat	4 gram
4	kisaran tegangan masukan	0 hingga 25V DC
5	Rentang deteksi tegangan	DC 0,02445 V hingga 25 V
6	Resolusi Tegangan Analog	0,00489V

Aplication

1. Memantau tegangan baterai: Modul sensor tegangan DC dapat digunakan untuk memantau tegangan baterai, memberikan informasi real-time tentang status pengisian dayanya. Hal ini sangat berguna dalam aplikasi seperti kendaraan listrik, sistem tenaga surya, atau perangkat elektronik portabel di mana mengetahui voltase baterai sangat penting untuk pengoperasian yang efisien dan mencegah pengosongan daya yang berlebihan.
2. Pemantauan catu daya: Modul sensor tegangan DC dapat digunakan untuk memantau tegangan keluaran catu daya di berbagai sistem elektronik. Dengan mengukur tegangan secara terus-menerus, modul dapat mendeteksi fluktuasi atau penyimpangan apa pun dari nilai yang diinginkan, sehingga memungkinkan pemecahan masalah dan pemeliharaan dengan cepat. Hal ini penting pada peralatan sensitif, di mana tegangan catu daya yang stabil dan akurat sangat penting agar dapat berfungsi dengan baik.
3. Otomasi industri: Modul sensor tegangan DC banyak digunakan dalam aplikasi otomasi industri. Mereka dapat diintegrasikan ke dalam sistem

kontrol untuk memantau dan mengatur tingkat tegangan sumber daya DC, memastikan pengoperasian mesin dan peralatan yang stabil dan andal. Sensor tegangan berperan penting dalam mendeteksi kesalahan, penurunan tegangan, atau kondisi abnormal, memicu alarm, atau memulai tindakan perbaikan untuk mencegah kerusakan atau waktu henti (Ardiansyah, A. 2020).

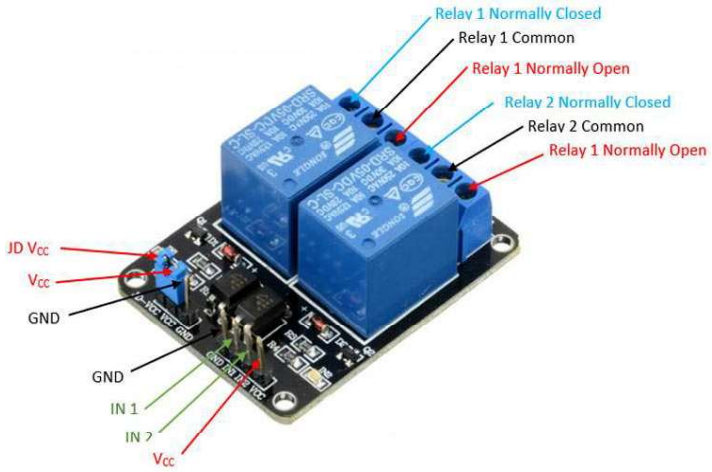
4. Rangkaian Kendali Otomatis

Rangkaian Kendali Otomatis yang digunkan adalah relay. Modul relay adalah perangkat, komponen, atau alat yang berfungsi sebagai saklar elektronik yang dikendalikan secara otomatis oleh sinyal listrik dari mikrokontroler seperti Arduino atau ESP32. Modul ini dapat digunakan untuk mengontrol aliran arus listrik pada perangkat bertegangan tinggi (seperti 220V AC) dengan hanya menggunakan sinyal kecil dari sistem kontrol digital.

Saat menerima sinyal input dari mikrokontroler, modul relay akan mengaktifkan elektromagnet di dalam relay, yang kemudian akan mengubah posisi kontak saklar di dalamnya. Proses ini memungkinkan relay untuk menyambungkan atau memutuskan aliran listrik pada rangkaian luar. Modul ini menghasilkan kontak fisik antara terminal NO (*Normally Open*), NC (*Normally Closed*), dan COM (*Common*), yang digunakan untuk mengatur status *ON/OFF* perangkat eksternal.

Modul relay menghasilkan aksi nyata berdasarkan input logika (*HIGH* atau *LOW*) dan sering digunakan dalam aplikasi otomasi rumah, sistem proteksi, pengendalian beban listrik, hingga monitoring daya. Modul ini juga dapat

terintegrasi dengan sistem penyimpanan data dan kontrol otomatis melalui *Internet of Things* (IoT) untuk memudahkan analisa dan pengendalian jarak jauh.



Gambar 2. 3 Relay

Sumber:(https://components101.com.)

Tabel 2. 4 Spesifikasi Relay

Nomor	Nama	Deskripsi
1	COM1	Terminal umum untuk channel 1
2	COM2	Terminal umum untuk channel 2
3	NO1	Output Normally Open chanel 1
4	NO2	Output Normally Open chanel 2
5	NC1	Output Normally Close chanel 1
	NC2	Output Normally Close chanel 2
6	IN1	Pin input kendali pertama untuk relay
7	IN2	Pin input kendali Kedua untuk relay
7	GND	Terminal sinyal ground
8	VCC	terminal catu daya perangkat

Sumber:(https://components101.com.)

5. Baterai dan Solar Charger Controler

a. Baterai

Baterai adalah perangkat penyimpan energi listrik yang bekerja dengan cara mengubah energi listrik menjadi energi kimia saat proses pengisian, dan mengubah kembali energi kimia tersebut menjadi energi listrik saat proses pemakaian (*discharging*). Dalam konteks sistem kelistrikan, baterai berfungsi sebagai sumber energi cadangan yang dapat menyediakan listrik secara mandiri tanpa harus bergantung pada pasokan dari jaringan utama. (Iqram, 2025).

Contoh jenis Baterai, adalah bateri tipe AGM (*Absorbent Glass Mat*) atau VRLA (*Valve Regulated Lead Acid*) adalah jenis baterai timbal-asam tertutup yang dirancang bebas perawatan *(maintenance-free*) karena elektrolitnya tidak berbentuk cair bebas, melainkan diserap sepenuhnya dalam lapisan serat kaca. Teknologi ini membuat baterai tidak mudah tumpah, tahan terhadap guncangan, dan dapat dipasang dalam berbagai posisi, sehingga sangat cocok digunakan pada kendaraan bermotor, sistem tenaga surya, UPS, dan perangkat elektronik lainnya. Katup tekanan otomatis (*valve*) pada baterai VRLA memungkinkan gas yang dihasilkan saat pengisian untuk direkombinasi menjadi air, sehingga tidak perlu pengisian ulang elektrolit seperti pada aki basah. Dibandingkan aki basah, baterai AGM memiliki efisiensi lebih tinggi, daya starter lebih besar, umur pakai lebih lama, serta lebih aman digunakan di lingkungan tertutup karena tidak menghasilkan uap berbahaya. Namun, harga awalnya lebih

mahal dan sensitif terhadap kesalahan pengisian. Secara umum, baterai AGM/VRLA merupakan pilihan ideal untuk sistem penyimpanan energi yang mengutamakan keandalan, keamanan, dan kemudahan penggunaan.



Gambar 2. 4 Baterai AGM/VRLA

Sumber:(https://www.yuasabattery.co.id)

Tabel 2. 5 Spesifikasi Baterai AGM/VRLA

No	**Spesifikasi**	
1	Tegangan nominal	12Volt
2	Kapsitas	3.5 Ah
3	CCA (Cold Cranking Ampere)	$\pm 65 - 80\ A$
4	Dimensi	113 × 70 × 85 mm
5	Berat	±1.6 kg
6	Tipe terminal	Nut & Bolt (T1/T2)

Sumber:(https://www.yuasabattery.co.id)

b. Solar charger controler

Solar Charge Controller (SCC) adalah perangkat elektronik yang berfungsi untuk mengatur dan mengontrol proses pengisian daya dari panel surya ke baterai, sehingga baterai dapat terisi secara optimal dan

aman. SCC bertugas menjaga agar baterai tidak mengalami *overcharging* (pengisian berlebih) maupun *overdischarging* (pengosongan berlebih) yang dapat merusak umur pakai baterai. Prinsip kerja SCC didasarkan pada pemantauan tegangan dan arus dari panel surya serta kondisi baterai, lalu mengatur kapan arus harus dialirkan, dibatasi, atau diputus. Ketika tegangan baterai rendah, SCC akan mengalirkan arus dari panel ke baterai; namun saat baterai telah terisi penuh, SCC akan memutus atau membatasi aliran arus untuk mencegah kerusakan. SCC juga biasanya memiliki terminal khusus untuk beban (*LOAD*), sehingga dapat langsung digunakan untuk menyalakan perangkat tanpa membebani baterai secara langsung. Terdapat dua jenis utama SCC berdasarkan metode pengisiannya, yaitu PWM (*Pulse Width Modulation*) dan MPPT (*Maximum Power Point Tracking*), di mana MPPT memiliki efisiensi pengisian yang lebih tinggi, terutama dalam kondisi pencahayaan yang tidak stabil. Dengan SCC, sistem tenaga surya menjadi lebih andal, efisien, dan aman dalam jangka panjang.



Gambar 2. 5 Solar charger controller

Sumber:(https://atonergi.com)

Tabel 2. 6 Spesifikasi *Solar charger controller* (pwm)

No	Spesifikasi	
1	Tegangan sistem	12v/ 24v
2	Arus maksimum	10/20/30 Ampare
3	Tegangan pegisian float dan boost	13,6v dan 14,4v
4	Tegangan Proteksi Overcharge, Undervoltage	14,5v dan 11,0v
5	Tipe Baterai yang Didukung	SLA ,AGM, GEL
6	Suhu Operasional	-20°C hingga +60°C

Sumber:(https://atonergi.com)

6. Sistem Rangkaian Tampilan

Sistem Rangkaian Tampilan ini menggunakan LCD I2C 16x2. LCD *(Liquid Crystal Display*) merupakan jenis media tampilan yang memanfaatkan kristal cair sebagai elemen utama penampil. LCD (*Liquid Crystal Display*) dapat menampilkan gambar atau karakter karena terdapat banyak titik cahaya (*piksel*), di mana setiap titik terdiri dari satu kristal cair. Meski disebut sebagai titik cahaya, Kristal cair ini tidak memancarkan cahaya sendiri. LCD 16x2 dapat menampilkan total 32 karakter yang terbagi dalam 2 baris, dengan masing-masing baris berisi 16 karakter. (Simarmata, f. A. 2024).

LCD 16×2 umumnya menggunakan 16 pin untuk kontrolnya, dan tentunya akan sangat boros jika semua 16 pin tersebut digunakan. Oleh karena itu, dipakai driver khusus agar LCD dapat dikendalikan dengan modul I2C (*Inter-Integrated Circuit*). Modul I2C memungkinkan LCD 16x2 menggunakan hanya dua pin untuk mengirim data dan dua pin sebagai sumber tegangan. Oleh karena itu, hanya diperlukan empat pin yang harus dihubungkan ke Esp 32 Dev kit 1yaitu:

Tabel 2. 7 Spesifikasi LCD I2C 16x2

No	Spesifikasi	
1	Protokol Komunikasi	I2C
2	VCC	3,3- 5v
3	Jumlah Pin I2C	SDA,SCL
4	Alamat I2C Default	0x27 atau 0x3



Gambar 2. 6 LCD I2C 16x2

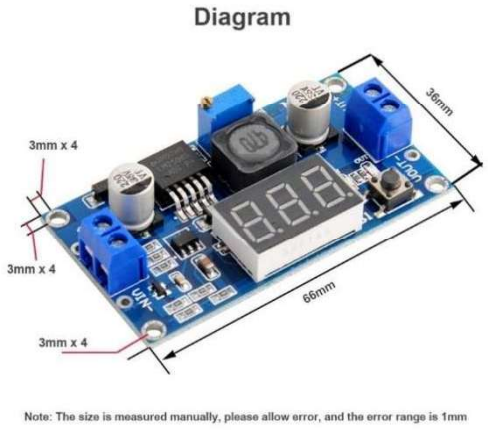
(Sumber: https://shopee.co.id/LCD-Display-1602-With-I2C)

7. Sistem Rangkain Penurun Tegangan

Rangkaian modul konverter DC/DC dengan frekuensi tetap 150 kHz dan fixed-voltage *(PWM step-down*) menggunakan IC Regulator LM2596 adalah Modul Regulator LM 2596. Modul ini mampu menggerakkan beban 5A dengan efisiensi tinggi, derek rendah, serta regulasi garis dan beban yang sangat baik. Regulator ini mudah digunakan, memerlukan jumlah minimum komponen eksternal dan mencakup osilator frekuensi tetap serta kompensasi frekuensi internal.

Modul regulator LM2596 dapat beroperasi dengan suplai tegangan antara 4V hingga 40V dan memiliki arus keluaran sebesar 3A. Module regulator LM2596 menggunakan IC SMD (*Surface Mount Device*) dan dilengkapi dengan sebuah

potensio untuk mengatur tegangan masukannya dari 1,25V hingga 37V DC pada frekuensi kerja 150 kHz, sehingga dapat disesuaikan dengan kebutuhan tegangan .



Gambar 2. 7 Stepdown LM 2596

(Sumber : https://shopee.co.id/product)

8. Mikrokontroler dan Aplikasi Sistem Kendali

a. Mikrokontroler

Mikrokontroler merupakan chip kecil yang berfungsi sebagai otak di berbagai perangkat elektronik. Mikrokontroler dirancang untuk mengontrol dan menjalankan perangkat atau sistem tertentu dengan cara memproses data dan menjalankan perintah yang sudah diprogram sebelumnya. Mikrokontroler sering digunakan dalam banyak aplikasi, seperti perangkat rumah tangga, alat kesehatan, robotika, otomotif, dan berbagai perangkat IoT (*Internet of Things*). (Widharma & Wiranata, 2022).

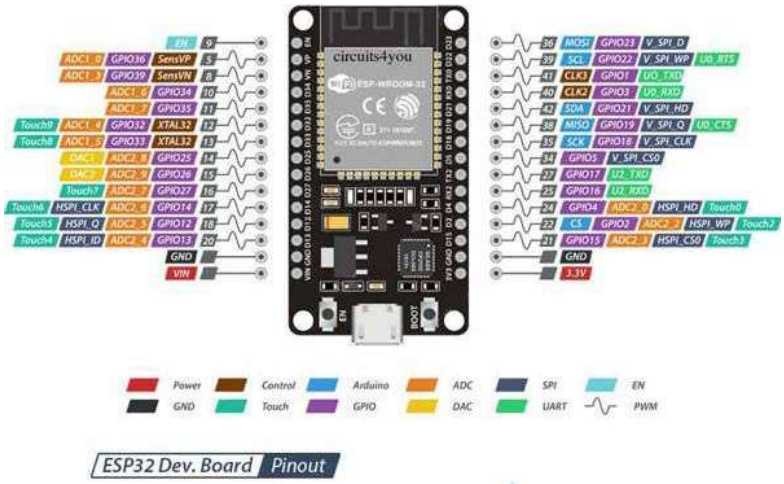
ESP32 merupakan sebuah mikrokontroler yang dikembangkan oleh Espressif Systems, berbasis prosesor Xtensa® 32-bit LX6 dual-core dengan kecepatan hingga 240 MHz. Mikrokontroler ini dirancang khusus untuk

mendukung aplikasi berbasis *Internet of Things* (IoT), karena telah dilengkapi dengan fitur Wi-Fi dan *Bluetooth* secara onboard. Kombinasi fitur ini menjadikan ESP32 sebagai pilihan populer dalam berbagai proyek otomasi, pengendalian jarak jauh, dan monitoring real-time, termasuk dalam sistem pengisian baterai yang membutuhkan kendali otomatis dan pelaporan status secara berkala.

Secara teknis, ESP32 memiliki spesifikasi yang cukup mumpuni untuk menangani berbagai tugas kendali dan komunikasi data. Mikrokontroler ini memiliki RAM sebesar 520 KB dan ROM sebesar 448 KB, serta mendukung protokol komunikasi seperti UART, SPI, I2C, PWM, dan CAN. Selain itu, ESP32 juga dilengkapi dengan ADC (*Analog to Digital Converter*) 12-bit, DAC

(*Digital to Analog Converter*), serta GPIO yang dapat digunakan untuk mengakses berbagai sensor dan aktuator. Dengan tegangan kerja sebesar 3.3V, ESP32 sangat cocok digunakan dalam sistem elektronik berdaya rendah.

Keunggulan utama ESP32 dalam sistem pengisian baterai terletak pada kemampuannya mengakses sensor secara real-time dan mengirimkan data ke cloud atau antarmuka pengguna melalui Wi-Fi. Selain itu, mikrokontroler ini juga mendukung mode hemat daya seperti *deep sleep* dan *light sleep*, yang memungkinkan sistem tetap efisien dalam penggunaan energi. Dengan demikian, ESP32 sangat sesuai untuk digunakan sebagai otak sistem pengendali pengisian baterai yang cerdas, efisien, dan dapat diakses dari jarak jauh.



Gambar 2. 8 Esp 32

Sumber:(https://www.ardutech.com)

Tabel 2. 8 spesifikasi Esp 32

No	Spesifikasi	Keterangan
1	Mikrokontroller	Tensilica Xtensa LX6 Dual-Core
2	Tegangan Operasional	3.3V
3	Tegangan Input	5V (melalui port USB) atau 3.3V (langsung ke pin 3.3V)
4	Pin I/O Digital	34 Pin
5	Pin Analog	18 Pin
6	SRAM	520 KB
7	EEPROM	4 KB
8	Kecepatan Clock	240 MHz

Sumber:(https://www.ardutech.com)

b. Aplikasi sistem kendali

Sistem kendali merupakan bagian penting dalam pengelolaan Pembangkit Listrik Tenaga Surya (PLTS), khususnya pada skala laboratorium. Dalam konteks ini, sistem kendali digunakan untuk mengatur berbagai parameter operasional, seperti intensitas cahaya yang diterima panel surya, tegangan keluaran, serta pengaturan beban yang terhubung dengan sistem. Pada skala laboratorium, sistem

kendali memungkinkan pengujian dan pengembangan teknologi yang lebih presisi, karena semua variabel dapat dikontrol dan dimanipulasi dengan lebih akurat (Rauf, R. 2023). Hal ini memberikan kemudahan bagi peneliti untuk memvalidasi hipotesis dan mengidentifikasi parameter optimal yang dapat diterapkan pada PLTS dalam skala yang lebih besar.

Penggunaan teknologi *Internet of Things* (IoT) dalam sistem kendali PLTS pada skala laboratorium memberikan banyak keuntungan, terutama dalam hal monitoring dan pengendalian jarak jauh (Rauf, R. 2023).. Dengan menggunakan sensor dan perangkat IoT, data dari PLTS dapat dikumpulkan secara *real-time* dan dianalisis untuk memberikan umpan balik kepada sistem kendali. Misalnya, sensor cahaya dapat mendeteksi perubahan intensitas cahaya matahari dan secara otomatis menyesuaikan posisi panel surya untuk mendapatkan eksposur maksimal. Selain itu, IoT memungkinkan peneliti untuk memantau kondisi operasional sistem, seperti suhu, kelembaban, dan arus listrik, dari jarak jauh melalui jaringan internet. Dengan demikian, sistem kendali berbasis IoT tidak hanya meningkatkan efisiensi operasional tetapi juga membantu dalam pemeliharaan prediktif dan pengurangan risiko kerusakan sistem.

Integrasi sistem kendali berbasis IoT dalam PLTS skala laboratorium juga membuka peluang untuk pengembangan fitur-fitur cerdas, seperti optimasi daya secara otomatis dan pengaturan beban dinamis. Misalnya, dalam kondisi di mana output daya surya berkurang akibat cuaca mendung, sistem kendali dapat secara otomatis mengurangi beban atau mengalihkan daya dari sumber cadangan untuk menjaga stabilitas sistem.

B. **Kajian Hasil Penelitian**

Adapun penelitian terdahulu yang menjadi salah satu acuan penulis dalam melakukan penelitian, sehigga penulis dapat memperbanyak teori yang digunakan dalam mengkaji penelitian yang dilakukan. Penulis sengaja mengangkat beberapa penelitian sebelumnya untuk dijadikan sebagai referensi dalam memperkaya bahan kajian pada penelitian ini. Berikut ini yang merupakan penelitian terdahulu berupa jurnal yang terkait dengan penelitian yang dilakukan oleh penulis dalam penelitian yang pertama dilakukan oleh (Sari, L. O.,2024).dengan judul "Sistem Monitoring Arus Listrik Berbasis *Internet of Things* (IoT) pada Solar Panel di Laboratorium Pembangkit Listrik Tenaga Surya (PLTS)"penelitian ini mencakup pemantauan Sistem Tenaga Surya Berbasis IoT, Optimasi Kinerja Panel Surya dengan Sensor IoT,dan Implementasi IoT dalam Monitoring Energi Terbarukan. ini menunjukkan bahwa penggunaan IoT dapat secara signifikan meningkatkan monitoring, optimasi, dan pemeliharaan sistem PLTS, baik dalam lingkungan laboratorium maupun untuk aplikasi yang lebih luas.

Penelitian yang kedua dilakukan oleh (Anggoro, G. P. T. N. 2023). dengan judul penelitian "kendali pembangkit listrik tenaga surya untuk ladang surya berbasis *internet of things*"penelitian ini mencakup Sistem Kendali Cerdas Berbasis IoT, Pemantauan dan Kendali Otomatis pada PLTS.

Penelitian yang ketiga dilakukan oleh (Setiawan, D. K., Widjonarko, W., 2022).dengan judul penelitian "Sistem Monitoring Panel Surya Berbasis Android Secara *Real-Time*" Penelitian terkait sistem monitoring panel surya berbasis

Android secara *real-time* menunjukkan bahwa aplikasi mobile dapat digunakan untuk memantau kinerja panel surya secara langsung. Sistem ini memanfaatkan perangkat *Android* untuk menampilkan data dari sensor panel surya, memungkinkan pemantauan dan penanganan masalah dari jarak jauh. Teknologi ini meningkatkan efisiensi pemantauan dengan menyediakan akses mudah dan responsif terhadap perubahan kondisi sistem tenaga surya. Integrasi aplikasi mobile mempermudah pengelolaan dan analisis performa sistem secara *real-time*.

Penelitian keempat dilakukan oleh Hendra Sahroni & Arnita, 2023 dengan judul "Perancangan Sistem Kontrol dan Monitoring Sistem Proteksi *Over Charging* Baterai pada PLTS Berbasis *Internet of Things* (IoT)". Penelitian ini mengembangkan sistem kendali dan pemantauan PLTS berbasis IoT yang dilengkapi mekanisme proteksi untuk mencegah terjadinya pengisian berlebih (*over charging*) pada baterai. Sistem tersebut memanfaatkan sensor INA219 untuk mengukur arus dan tegangan dengan tingkat akurasi yang baik berdasarkan hasil perbandingan dengan alat ukur konvensional. Mikrokontroler ESP32 digunakan sebagai pengendali utama yang mengatur proses pengisian serta menjaga kestabilan koneksi data selama sistem beroperasi. Hasil penelitian menunjukkan bahwa integrasi IoT dalam sistem proteksi baterai mampu meningkatkan keamanan, memperpanjang umur pakai baterai, dan menjaga kinerja PLTS agar tetap optimal.

BAB III

METODE PENELITIAN

A. Jenis Penelitian

Jenis penelitian yang dilakukan adalah dengan merancang sistem kendali Pembangkit Listrik Tenaga Surya (PLTS) berbasis *Internet of Things* (IoT). Sistem dirancang untuk melakukan kendali pada Panel Surya dan Proses Charging Baterai secara otomatis melalui aplikasi berbasis *Internet of Things.*

B. Lokasi dan waktu

1. Penelitian ini dilakukan di Laboratorium Teknik Elektro Universitas Muhammadiyah Parepare.

2. Waktu

Untuk waktu penelitian dapat dilihat pada tabel di bawah ini :

Tabel 3. 1 Jadwal pelaksanaan penelitian

No	Kegiatan	Bulan					
		1	2	3	4	5	6
1	Perancangan perangkat keras						
2	Perancangan perangkat lunak						
4	Realisasi perangkat keras dan perangkat lunak						
6	Pengujian tahap awal (validasi dan kalibrasi) komponen)						
8	Pengujian tahap kedua (Blok diagram)						
	Pembuatan aplikasi web						
9	Pengujian pengedalin proses charging pada baterai						
10	Pengujian sistem secara keseluruhan						
11	Analisis dan evaluasi						
12	Penyusunan laporan						

C. Alat dan bahan

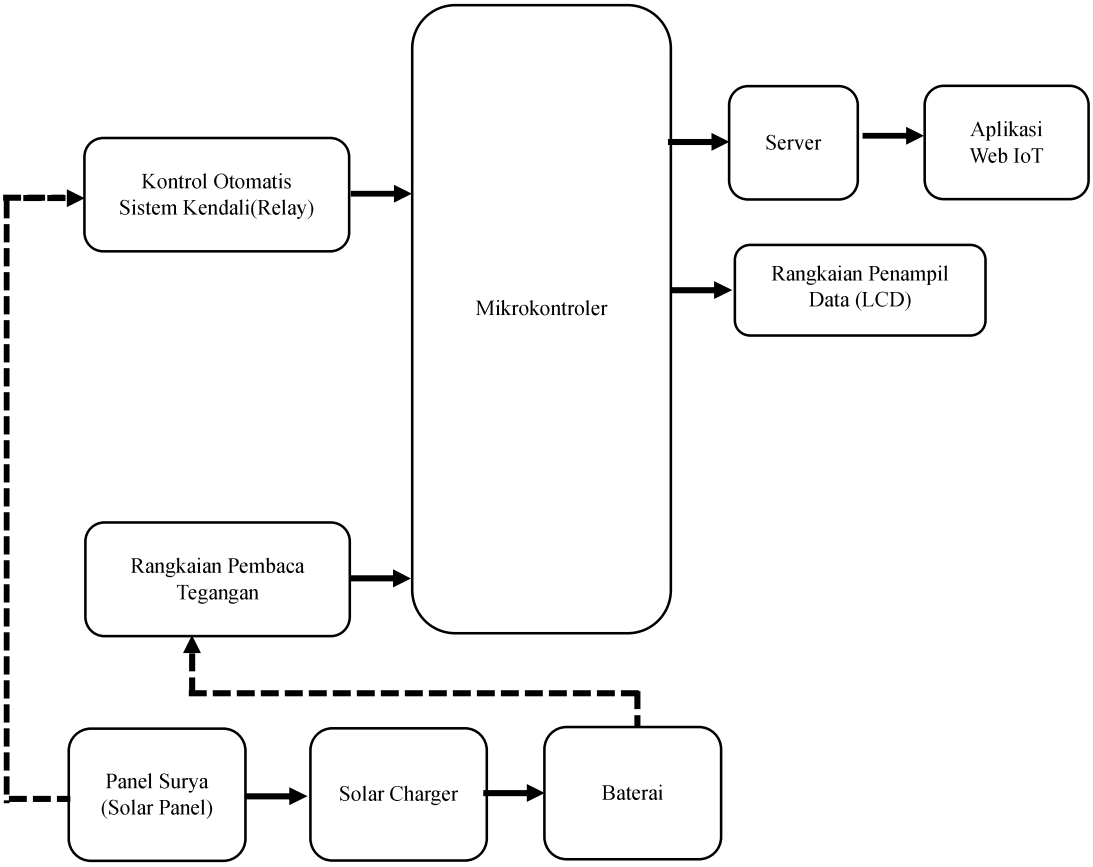
Alat dan bahan yang digunakan dalaam penelitan ini yaitu:

Tabel 3. 2 Alat dan Bahan beserta jumlahnya

No	Uraian	Jumlah
1	Panel Surya	1
2	Sensor Tegangan	1
3	Relay	1
4	Baterai	1
5	LCD	1
6	Esp 32	1
7	Solar charge	1
8	Step down	1

D. Rancangan Sistem

Pada bagian ini akan dibahas suatu pernyataan rangkaian alat yang akan dirancang. Berikut blok diagram dan perancangan sistem kendali pembangkit listrik tenaga surya skala laboratorium berbasis *internet of thing.*



Gambar 3. 1 Blok diagram

Blok diagram sistem keseluruhan dari alat ini terdiri dari : panel surya, solar charge, baterai, beban (*load*), sensor tegangan dan mikrokontroler, sistem kendali otomatis (Relay), LCD, database, aplikasi web IOT.

Penjelasan pada bagian blok diagram sebagai berikut:

1. **Panel surya**

Sistem ini memiliki1 panel surya yang berfungsi untuk menangkap energi matahari dan mengubahnya menjadi energi listrik

2. **Solar charge**

Bekerja untuk menerima energi listrik dari panel surya dan mengatur pengisian baterai untuk memastikan baterai diisi dengan aman dan efisien

3. **Baterai**

Untuk menyimpan energi listrik yang dihasilkan oleh panel surya melalui solar charger, energi yang disimpan di baterai dan dapat digunakan saat dibutuhkan.

4. **Kontrol otomatis sistem kendali (Relay)**

Kontrol manual sistem kendali, ini memungkinkan pengguna untuk mengendalikan beberapa fungsi pada sistem secara otomatis.

5. **Sensor tegangan**

Sensor tegangan yang digunakan untuk memantau kondisi tegangan pada berbagai titik dalam sistem seperti di output panel surya dan baterai.

6. **Mikrokontroler**

Mikrokontroler bertindak sebagai otak dari sistem kendali, menerima data dari sensor, mengontrol pengisian baterai, dan mengirimkan data ke *server*.

7. **LCD**

Digunakan untuk menampilkan informasi seperti status kapasitas,baterai, tegangan baterai, kondisi *charging*.

8. Server

Untuk menyimpan data yang dikirim oleh mikrokontroler. Data ini kemudian dapat diakses melalui aplikasi web IOT.

9. Aplikasi web

Digunakan untuk memantau dan mengendalikan sistem PLTS dari jarak jauh melalui internet.

E. Teknik Pengumpulan Data

1. Validasi dan Kalibrasi Sensor

Tahap ini dilakukan untuk mempelajari Validasi dan kalibrasi sensor untuk proses memastikan bahwa sensor memenuhi spesifikasi yang diperlukan dan berfungsi dengan benar dalam kondisi yang diinginkan. Tujuannya adalah untuk memastikan bahwa sensor dapat memberikan hasil data yang akurat dan konsisten.

2. Pengujian Blok Diagram

Pengujian setiap blok dilakukan untuk mempermudah analisis sistem. Pengujian dibagi beberapa bagian, yaitu:

a. Pengujian sensor untuk memastikan sensor mengukur dengan akurat dan mengirim data yang benar.
b. Pengujian komunikasi data untuk memverifikasi bahwa data dikirim dari pengontrol ke server dengan benar.
c. Pengujian server untuk memastikan server menerima dan menyimpan data dengan benar serta mengelola permintaan dari antarmuka web.

d. Pengujian antarmuka web untuk memastikan apakah data yang ditampilkan dengan benar di web browser dan pembaruan real time berjalan dengan baik.

3. Pengujian Sistem Secara Keseluruhan

Tahap ini dilakukan untuk menguji sistem secara keseluruhan setelah semua sub – blok diagram telah terintegrasi. Pengujian sistem keseluruhan memverifikasi bahwa perancangan ini menjalnakan tugas sesuai rancangan. Pengujian sistem dapat memeriksa bahwa setiap jenis masukan menghasilkan keluaran yang diinginkan di seluruh rancangan.

BAB IV

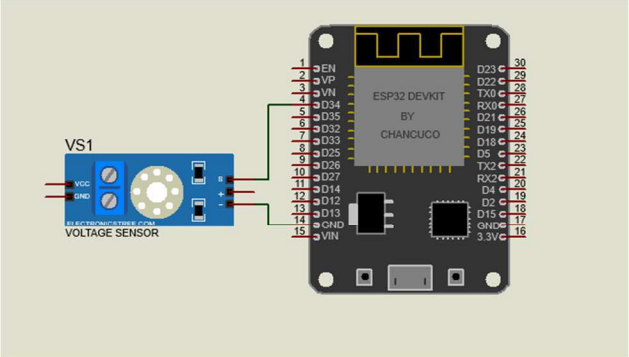
HASIL DAN PEMBAHASAN

A. Perancangan Perangkat Keras

Pada perancangan sistem perangkat keras, dilakukan analisis terhadap kebutuhan teknis dan fungsional yang diperlukan untuk memastikan perangkat keras yang dikembangkan dapat mendukung kinerja sistem secara optimal, dengan mempertimbangkan aspek efesiensi, dan kemampuan dalam pembacaan hasil sensor. Rancangan ini menjelaskan tentang sistem rangkaian yang di buat secara keseluruhan dan rangkaian sistem per bagian dari sistem rangkaian keseluruhan.

1. Rankaian Pembaca tegangan

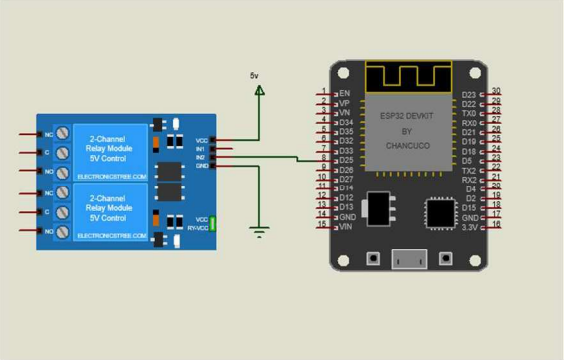
Sensor tegangan pada sistem ini berfungsi untuk mendeteksi besarnya tegangan dari baterai 12V. Modul sensor yang digunakan adalah *voltage* sensor berbasis pembagi tegangan (*voltage divider*), yang dirancang untuk membaca tegangan hingga 25V dan menurunkannya ke level yang aman agar dapat diterima oleh pin analog mikrokontroler. Dalam rangkaian ini, sensor dihubungkan ke *board* ESP32 melalui pin analog IO34, sementara jalur ground sensor disambungkan ke pin *GND* mikrokontroler. Nilai tegangan yang terbaca pada pin analog tersebut kemudian diolah oleh ESP32 untuk mengetahui kondisi baterai secara real-time. Hasil olahan data ini dapat digunakan untuk keperluan monitoring, proteksi sistem, atau sebagai logika pengambilan keputusan seperti mengatur proses pengisian atau pemutusan beban secara otomatis melalui sistem kontrol.



Gambar 4. 1 Rangkaian Pembaca tegangan

2. Rangkaian Sistem kendali

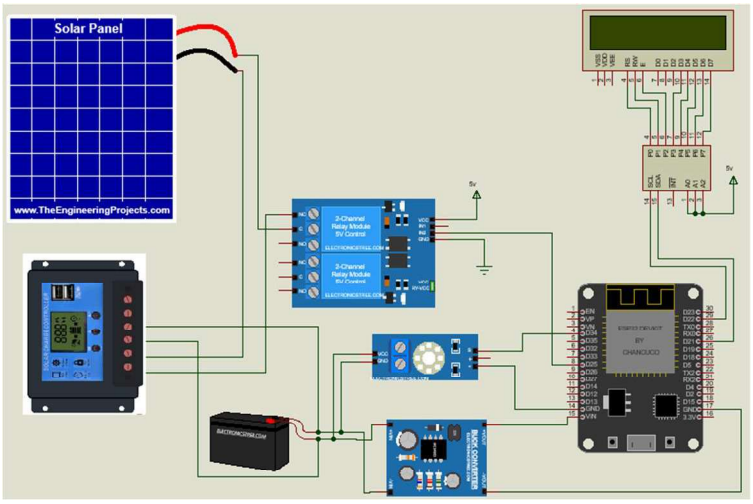
Pada rangkaian ini digunakan modul relay 2 channel yang berfungsi sebagai saklar elektronik untuk mengendalikan perangkat bertegangan tinggi, seperti beban AC atau DC, dengan sinyal kontrol dari mikrokontroler. Modul relay dihubungkan ke board ESP32, di mana pin IN1 dan IN2 pada relay masing-masing tersambung ke pin digital IO25 dan IO26 pada ESP32. Selain itu, pin *VCC* pada modul relay dihubungkan ke pin 5V pada ESP32, sedangkan pin *GND* relay dihubungkan ke GND ESP32 agar keduanya memiliki referensi tegangan yang sama. Ketika ESP32 mengirimkan logika *HIGH* atau *LOW* ke pin kendali relay, maka relay akan aktif atau nonaktif, sehingga dapat memutus atau menghubungkan rangkaian daya yang dikontrol. Sistem ini sangat berguna dalam aplikasi otomatisasi, seperti mengontrol pengisian baterai, memutus beban saat tegangan rendah, atau menyalakan peralatan secara otomatis berdasarkan kondisi tertentu yang dibaca oleh sensor.



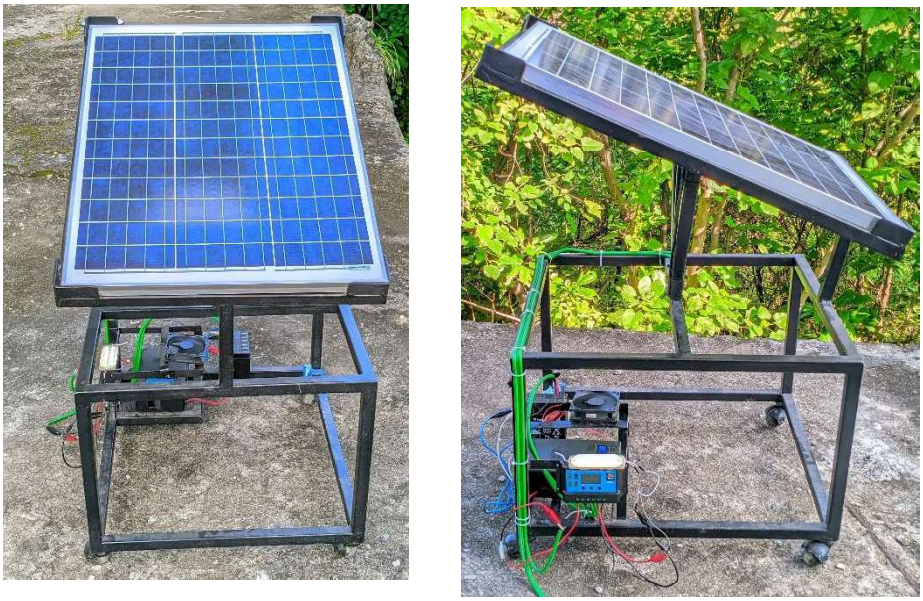
Gambar 4. 2 Rangkaian pegendali

3. Rankaian sistem keseluruhan

Dalam perancangan sistem kendali panel surya ini, dilakukan integrasi antara panel surya, *solar charge controller*, baterai 12V, sensor, modul relay, dan mikrokontroler ESP32. Panel surya menghasilkan energi listrik yang disalurkan ke *solar charge controller* (PWM) untuk mengatur aliran arus pengisian ke baterai. Sensor tegangan digunakan untuk memantau level tegangan baterai secara *real-time*, dan datanya dikirim ke mikrokontroler ESP32, yang berfungsi sebagai pusat pengendali logika sistem. Modul relay 2 channel digunakan untuk mengatur sirkuit pengisian atau beban secara otomatis, berdasarkan tegangan yang terdeteksi oleh sistem. Informasi sistem ditampilkan secara langsung melalui LCD 16x2, yang menunjukkan status tegangan dan proses pengisian. Sistem ini juga dilengkapi dengan koneksi antarmuka I2C dan logika digital antara sensor, relay, dan ESP32 untuk menjamin kendali dan pemantauan yang efisien. Dengan desain ini, sistem mampu bekerja secara mandiri.



Gambar 4. 3 Rangkaian Keseluruhan



Gambar 4. 4 Rancangan Alat

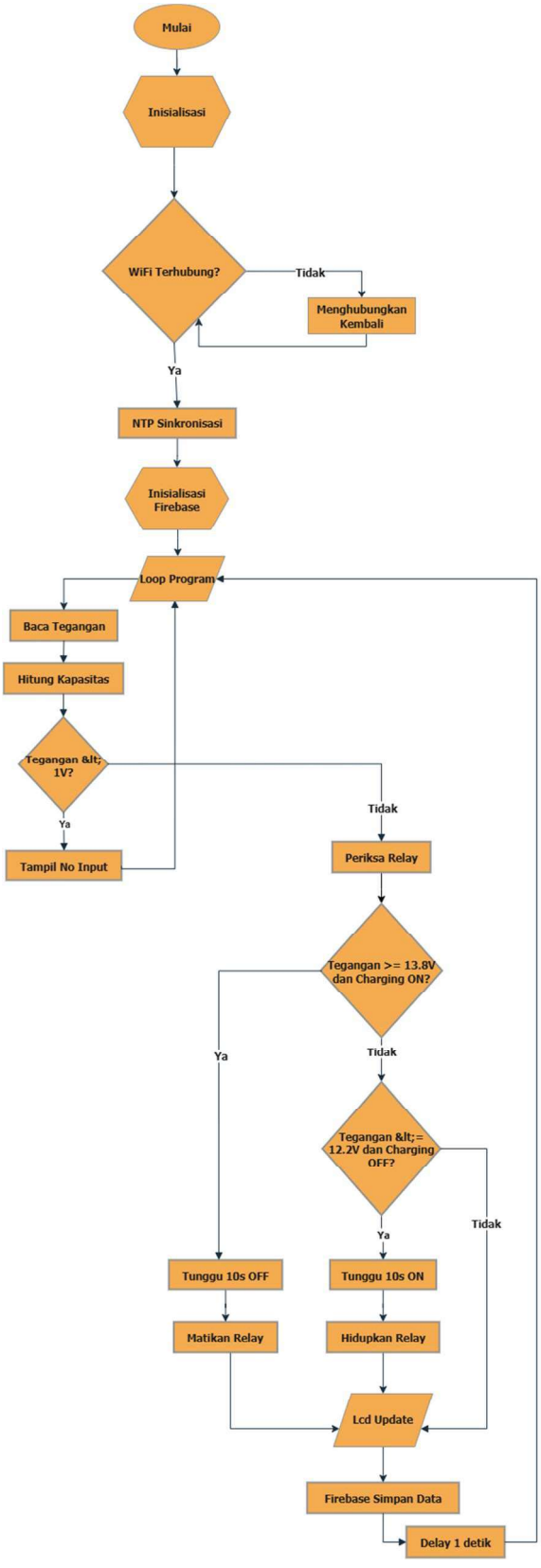
B. Perancangan Perangkat Lunak

Perancangan perangkat lunak untuk sistem kendali pembangkit listrik tenaga surya skala laboratorium berbasis *Internet of Things* bertujuan untuk mengontrol dan memantau seluruh perangkat keras yang terhubung, seperti sensor tegangan,

modul relay, dan LCD I2C, melalui instruksi yang ditanamkan dalam mikrokontroler ESP32. Pengembangan program dilakukan menggunakan software Arduino IDE dengan bahasa pemrograman berbasis C/C++, di mana barisan kode dibuat untuk membaca data sensor tegangan yang memantau kondisi pegisian baterai 12V secara real-time. Selain itu, perangkat lunak ini dirancang untuk mengatur kerja modul relay secara otomatis agar dapat menghubungkan atau memutus rangkaian pengisian baterai berdasarkan nilai tegangan yang terdeteksi. Data yang diolah oleh mikrokontroler kemudian ditampilkan pada layar LCD I2C dan di kirim ke aplikasi web agar pengguna dapat memantau status sistem secara langsung. Dengan rancangan ini, sistem kendali mampu bekerja secara otomatis dan efisien, serta memberikan kemudahan dalam pengawasan kondisi PLTS.

1. *Flowchart*

Flowchart adalah grafik yang digunakan untuk menggambarkan alur kerja atau proses dalam bentuk simbol-simbol grafis untuk menunjukkan Langkah-langkah aliran kontrol yang berperan dalam proses.*Flowchart* memudahakan penerima pegrtian suatu sistem dalam cara yg lebih visual dan tersutruktur. Diagram alir sistem kerja di tunjukan pada gambar 4.5



Gambar 4. 5 *Flowchart*

Flowchart pada Gambar 4.5 menggambarkan alur kerja sistem kendali otomatis pada pembangkit listrik tenaga surya skala laboratorium yang terintegrasi dengan teknologi *Internet of Things* (IoT). Sistem ini dirancang untuk memantau kondisi tegangan baterai serta mengendalikan proses pengisian (charging) secara otomatis melalui modul relay.

Proses dimulai dengan tahap inisialisasi, di mana sistem menyiapkan koneksi WiFi, sensor, relay, serta berbagai modul pendukung, termasuk koneksi ke layanan *cloud (Firebase)* untuk keperluan penyimpanan data secara daring (*online*).

Setelah inisialisasi selesai, sistem melakukan pengecekan apakah koneksi WiFi tersedia. Jika koneksi belum terhubung, sistem akan berupaya untuk menghubungkan kembali hingga berhasil. Apabila *WiFi* berhasil terhubung, langkah selanjutnya adalah melakukan sinkronisasi waktu melalui NTP (*Network Time Protocol*). Hal ini penting agar data yang dicatat dan dikirim ke cloud memiliki informasi waktu yang akurat.

Berikutnya, sistem melakukan inisialisasi *Firebase* sebagai media penyimpanan data secara *online*. Data-data seperti tegangan baterai, kapasitas, status relay, dan waktu pencatatan akan disimpan ke *Firebase*, sehingga memudahkan proses monitoring dari jarak jauh.

Sistem kemudian memasuki proses utama yaitu *loop* program, yang dijalankan secara terus-menerus. Di dalam *loop* ini, sistem akan membaca tegangan baterai yang dihasilkan oleh pembangkit listrik tenaga surya. Data tegangan tersebut kemudian digunakan untuk menghitung kapasitas baterai agar diketahui kondisi daya yang tersimpan.

Jika terdeteksi bahwa tegangan baterai kurang dari 1 Volt, maka sistem menganggap tidak ada input tegangan, misalnya akibat sensor terlepas, koneksi kabel terputus, atau kondisi panel surya tidak menghasilkan daya. Dalam kondisi ini, sistem akan menampilkan pesan "*No Input*" pada LCD sebagai penanda adanya gangguan atau ketidaknormalan.

Namun, jika tegangan baterai berada di atas atau sama dengan 1 Volt, sistem melanjutkan ke tahap pemeriksaan kondisi relay. Pada tahap ini, sistem menentukan apakah relay perlu dihidupkan atau dimatikan, berdasarkan nilai tegangan baterai dan status proses pengisian. Apabila tegangan baterai ≥ 13,8 Volt dan proses pengisian masih aktif (*Charging ON*), maka sistem akan menunggu selama 10 detik sebelum memutuskan untuk mematikan relay. Tujuannya agar pengisian baterai dihentikan guna menghindari risiko *overcharge* yang dapat merusak baterai.Sebaliknya, apabila tegangan baterai ≤ 12,2 Volt dan proses pengisian sedang tidak aktif (*Charging OFF*), sistem akan menunggu selama 10 detik sebelum memutuskan untuk menghidupkan relay. Hal ini dilakukan agar proses pengisian daya kembali aktif sehingga baterai tidak mengalami kondisi kekurangan daya (*undercharge*).

Setelah relay dihidupkan atau dimatikan sesuai dengan kondisi, sistem akan melakukan pembaruan tampilan pada LCD untuk menampilkan informasi terbaru, seperti tegangan baterai, kapasitas, serta status relay (*ON/OFF*).

Data terbaru ini selanjutnya akan dikirim dan disimpan ke *Firebase*, sehingga operator atau pengguna dapat memantau kondisi pembangkit listrik tenaga surya secara *real-time* dari jarak jauh.

Setelah semua langkah selesai, sistem memberikan jeda waktu (*delay*) selama 1 detik, kemudian kembali mengulangi proses loop secara terus-menerus. Dengan demikian, kondisi baterai selalu terpantau dan proses pengisian dapat dikendalikan secara otomatis sesuai kebutuhan.

Melalui alur kerja ini, sistem kendali pembangkit listrik tenaga surya skala laboratorium dapat bekerja secara otomatis, efisien, dan terintegrasi dengan teknologi IoT, sehingga mempermudah monitoring, pencatatan data, serta pengambilan keputusan untuk menjaga performa baterai dan keselamatan sistem.

2. Uraian Program

Adapun sketch program yang akan digunakan untuk menjalankan fungsi dari sistem yang dibuat sesuai dengan *flowchart* yaitu:

a. Inisialisasi objek dan menentukan *library* yang akan digunakan.

1. Penjelasan library program

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <WiFi.h>
#include <FirebaseESP32.h>
#include <time.h>
```

Program kendali sistem Pembangkit Listrik Tenaga Surya (PLTS) skala laboratorium berbasis Internet of Things (IoT) ini dirancang dengan memanfaatkan beberapa library utama yang mendukung fungsi komunikasi, tampilan, dan pengelolaan data secara daring. Terdapat lima library penting yang digunakan dalam program ini, yaitu *Wire.h, LiquidCrystal_I2C.h, WiFi.h, FirebaseESP32.h*, dan t*ime.h*.Library *Wire.h* merupakan pustaka dasar yang digunakan untuk komunikasi antar perangkat melalui protokol I2C (*Inter-Integrated Circuit*). Dalam konteks sistem ini, *Wire.h* sangat berperan dalam memastikan koneksi antara

mikrokontroler ESP32 dan perangkat I2C lainnya seperti sensor atau layar LCD berjalan dengan baik.Selanjutnya, LiquidCrystal_I2C.h digunakan untuk mengendalikan tampilan pada modul LCD yang menggunakan antarmuka I2C. Dengan library ini, informasi seperti tegangan baterai, status relay dapat ditampilkan secara langsung pada layar LCD, sehingga pengguna dapat memantau sistem secara lokal dengan mudah.*Library WiFi.h* berfungsi sebagai pengelola konektivitas nirkabel ESP32 ke jaringan *Wi-Fi*. Dengan koneksi ini, ESP32 dapat terhubung ke internet dan mengirim atau menerima data dari server, dalam hal ini Firebase.Untuk integrasi dengan layanan *cloud*, digunakan *library* FirebaseESP32.h. Library ini memungkinkan ESP32 untuk berkomunikasi langsung dengan *Firebase Realtime Database*, yaitu platform penyimpanan data berbasis *cloud* yang dikembangkan oleh Google. Melalui *Firebase*, data hasil pemantauan sistem dapat disimpan, diakses, dan dipantau secara *real-time* dari jarak jauh, memberikan kemudahan dalam proses monitoring dan kendali.Terakhir, library *time.h* digunakan untuk melakukan sinkronisasi waktu antara mikrokontroler dan server waktu global (NTP *server*). Hal ini penting agar setiap data yang dicatat dan dikirimkan memiliki stempel waktu (*timestamp*) yang akurat dan sesuai dengan waktu nyata, sehingga analisis data dapat dilakukan dengan lebih tepat.

2. Penjelasan fungsi *define* program

```
#define WIFI_SSID "Vio"
#define WIFI_PASSWORD "adl220180020"
#define FIREBASE_HOST
"https://panelsurya-f15ad-default-rtdb.firebaseio.com"
#defineFIREBASE_AUTH
"7ae7WV2viAk73oNbq5t1NuJXOZ0WMdT3fo81Uc9c"
#define ANALOG_IN_PIN   34
```

```
#define RELAY_CHARGE    25
#define REF_VOLTAGE     3.3
#define ADC_RESOLUTION  4096.0
#define R1              30000.0
#define R2              7500.0

LiquidCrystal_I2C lcd(0x27, 16, 2);
```

Dalam program ini, terdapat beberapa konstanta yang didefinisikan sebagai parameter pengaturan sistem, seperti *WIFI_SSID* dan *WIFI_PASSWORD* yang menyimpan nama hotspot dan kata sandi WiFi yang akan digunakan. Selain itu, terdapat konstanta *FIREBASE_HOST* dan *FIREBASE_AUTH* yang berfungsi sebagai alamat database Firebase dan token autentikasi agar ESP32 dapat terhubung ke layanan cloud. Beberapa nilai teknis lainnya seperti *REF_VOLTAGE*, *ADC_RESOLUTION,* R1, dan R2 digunakan dalam perhitungan konversi data ADC menjadi nilai tegangan baterai yang sebenarnya, dengan mempertimbangkan faktor pembagi tegangan (*voltage divider*) yang digunakan dalam rangkaian sensor tegangan.Seluruh sistem monitoring tegangan baterai berbasis IoT ini memanfaatkan input data dari pin *ANALOG*_IN_PIN (pin 34) sebagai input tegangan analog hasil pembacaan sensor *voltage divider*. Selain itu, terdapat pin kendali relay yang terhubung ke pin *RELAY_CHARGE* (pin 25), yang bertugas memutus atau menyambung jalur pengisian baterai dari panel surya ke baterai sesuai dengan logika kendali sistem.Untuk menampilkan status sistem secara lokal, digunakan LCD I2C 16x2 yang diinisialisasi melalui objek LiquidCrystal_I2C lcd(0x27, 16, 2). LCD ini memiliki alamat I2C 0x27 dan resolusi 16 karakter dalam 2 baris. LCD digunakan untuk menampilkan pesan-pesan penting, misalnya status koneksi WiFi,tegangan baterai hingga status pengisian baterai.

3. penjelasan fungsi variabel program

```
// Firebase objects
FirebaseData fbdo;
FirebaseAuth auth;
FirebaseConfig config;

unsigned long lastSwitchTime = 0;
const unsigned long delayRelay = 10000;

bool charging = true;

const float teganganOff = 13.8;
const float teganganOn  = 12.2;
// Faktor koreksi linearconst
float koreksi = 1.087;
```

Pada bagian ini, program mendefinisikan sejumlah objek dan variabel global yang berfungsi untuk mendukung proses komunikasi dengan Firebase, pengaturan logika kendali relay, serta pengolahan data tegangan baterai.

Pertama, dideklarasikan tiga objek utama yang berkaitan dengan Firebase, yaitu *FirebaseData fbdo, FirebaseAuth auth*, *dan FirebaseConfig config*. Objek fbdo digunakan sebagai wadah data untuk mengelola transaksi komunikasi antara ESP32 dan *Firebase Realtime Database*, baik untuk pengiriman maupun penerimaan data. Sementara itu, *auth* dan *config* berfungsi sebagai komponen autentikasi dan konfigurasi layanan, yang memastikan bahwa ESP32 dapat terhubung dengan database *cloud* secara aman dan sesuai dengan parameter yang telah didefinisikan sebelumnya pada bagian *FIREBASE_HOST* dan *FIREBASE_AUTH.*

Selanjutnya, variabel *lastSwitchTime* bertipe *unsigned long* digunakan untuk menyimpan waktu terakhir relay mengalami perubahan status, dalam satuan milidetik. Variabel ini bekerja sama dengan konstanta *delay*Relay yang bernilai 10000 (10 detik), yang berfungsi sebagai jeda minimum untuk mencegah relay

berpindah kondisi terlalu cepat atau berulang kali dalam waktu singkat. Dengan pengaturan ini, sistem lebih stabil dan komponen relay menjadi lebih awet karena tidak sering mengalami switching yang berlebihan.

Variabel charging bertipe *bool* digunakan sebagai penanda status pengisian baterai. Nilai *true* menandakan bahwa relay dalam kondisi aktif sehingga baterai sedang dalam proses pengisian, sedangkan nilai *false* menandakan bahwa relay dalam kondisi mati dan pengisian dihentikan.

Kemudian, terdapat dua konstanta bertipe *float* yang mendefinisikan tegangan batas untuk logika pengendalian relay, yaitu tegangan*Off* sebesar 13,8 V dan tegangan*On* sebesar 12,2 V. Tegangan 13,8 V digunakan sebagai batas atas yang menandakan baterai telah mencapai sekitar 85% kapasitas, sehingga pengisian dihentikan untuk menghindari risiko *overcharging*. Sementara itu, tegangan 12,2 V digunakan sebagai batas bawah yang menandakan baterai berada pada sekitar 30% kapasitas, sehingga pengisian perlu diaktifkan kembali untuk mencegah *deep discharging*. Logika ini membentuk mekanisme histeresis, yang bertujuan menjaga kesehatan baterai dan memperpanjang umur pakainya.

Terakhir, didefinisikan konstanta koreksi dengan nilai 1.087, yang berperan sebagai faktor koreksi linear untuk hasil pembacaan tegangan dari sensor analog. Faktor ini digunakan karena pembacaan tegangan dari rangkaian pembagi tegangan dan ADC ESP32 tidak selalu akurat secara mutlak, sehingga penyesuaian diperlukan agar nilai yang ditampilkan maupun dikirim ke *cloud* mendekati kondisi tegangan aktual baterai.

4. Penjelasan fungsi program konversi tegangan ke kapasitas persen

```
// Konversi tegangan → kapasitas persen
int teganganToPersen(float tegangan) {
 if (tegangan >= 14.4) return 100;
 else if (tegangan >= 13.8)
  return map(tegangan * 100, 1380, 1440, 85, 100);
 else if (tegangan >= 12.2)
  return map(tegangan * 100, 1220, 1380, 30,
 else if (tegangan >= 11.0)
  return map(tegangan * 100, 1100, 1220, 0,
 else
  return 0;
}
```

Selanjutnya, fungsi tegangan *ToPersen*() dibuat untuk mengkonversi nilai tegangan baterai menjadi persentase kapasitas baterai. Fungsi ini memiliki logika pembagian range, misalnya tegangan di atas 14.4V dianggap penuh 100%, sedangkan tegangan antara 12.2V hingga 13.8V dianggap memiliki kapasitas antara 30% hingga 85%. Fungsi map() digunakan untuk menghasilkan skala nilai persentase secara linear antara batas bawah dan batas atas tegangan.

5. Penjelasan fungsi *getAverageVoltage*

```
float getAverageVoltage(int pin) {
long total = 0;
for (int i = 0; i < 25; i++) {
  total += analogRead(pin);
  delay(20);
}
float voltageADC = ((float)total / 25.0) * REF_VOLTAGE /
ADC_RESOLUTION;
float teganganBaterai = voltageADC * ((R1 + R2) / R2) *
koreksi;
 return teganganBaterai;
}
```

Fungsi *getAverageVoltage* pada program ini digunakan untuk memperoleh nilai tegangan baterai yang lebih stabil dan akurat melalui proses pembacaan sinyal analog dari pin sensor. Proses diawali dengan melakukan pembacaan nilai ADC pada pin yang ditentukan sebanyak 25 kali secara berulang, kemudian setiap hasil

pembacaan dijumlahkan ke dalam variabel total dengan jeda waktu sekitar 20 milidetik di antara setiap pengukuran. Metode pengambilan sampel berganda ini bertujuan untuk meminimalkan fluktuasi data akibat noise atau gangguan listrik yang mungkin muncul pada pembacaan analog. Setelah seluruh sampel terkumpul, program menghitung rata-rata nilai ADC dengan cara membagi total pembacaan dengan jumlah sampel, lalu mengonversinya menjadi tegangan ADC dalam satuan volt menggunakan persamaan dasar ADC, yaitu membandingkan hasil pembacaan dengan tegangan referensi (*REF_VOLTAGE*) dan resolusi ADC(ADC_*RESOLUTION*). Nilai tegangan ADC ini selanjutnya dikonversi menjadi tegangan baterai aktual melalui persamaan pembagi tegangan yang melibatkan resistor R1 dan R2, yang berfungsi menurunkan tegangan agar sesuai dengan batas input analog ESP32. Hasil akhir kemudian dikalikan dengan faktor koreksi linear koreksi untuk menyesuaikan pembacaan agar mendekati nilai tegangan nyata baterai.

6. Fungsi *get DateTime*

```
String getDateTime() {
time_t now = time(nullptr);
struct tm* t = localtime(&now);
char buf[25];
sprintf(buf, "%04d-%02d-%02d %02d:%02d:%02d",
  1900 + t->tm_year, 1 + t->tm_mon, t->tm_mday,
  t->tm_hour, t->tm_min, t->tm_sec);
return String(buf);
}
```

Fungsi *getDateTime*() pada program ini digunakan untuk menghasilkan informasi waktu dalam format yang rapi dan mudah dibaca, yang nantinya dapat digunakan sebagai penanda waktu (*timestamp*) pada setiap data yang dikirim atau disimpan. Proses dimulai dengan pemanggilan fungsi *time(nullptr)* untuk memper oleh waktu saat ini dari sistem, yang telah disinkronisasi sebelumnya dengan server

NTP melalui *library time.h*. Nilai waktu ini disimpan dalam variabel bertipe *time_t* bernama *now*. Selanjutnya, fungsi *localtime(&now)* digunakan untuk mengonversi nilai waktu mentah tersebut menjadi struktur tm yang berisi komponen waktu lengkap, seperti tahun, bulan, hari, jam, menit, dan detik sesuai zona waktu lokal.Hasil konversi ini kemudian diformat ke dalam bentuk string menggunakan sprintf() dan disimpan pada *buffer* karakter buf dengan panjang 25 karakter. Format yang digunakan adalah *YYYY-MM-DD HH:MM:SS,* di mana tahun dihitung dengan menambahkan 1900 ke *t->tm_year* dan bulan dimulai dari 1 karena *tm_mon* menggunakan indeks mulai dari nol. Setelah itu, *buffer* karakter ini dikembalikan sebagai objek String agar mudah digunakan di seluruh program, misalnya untuk dicatat di LCD, disertakan dalam log data, atau dikirimkan ke *Firebase Realtime Database.*

7. Penjelasan fungsi *setup* program

```
void setup() {
Serial.begin(9600);
WiFi.mode(WIFI_STA);
WiFi.disconnect();
 delay(1000);
analogSetAttenuation(ADC_11db);
pinMode(RELAY_CHARGE, OUTPUT);
digitalWrite(RELAY_CHARGE, LOW);
lcd.init     ();
lcd.backlight();
lcd.setCursor(0, 0);
lcd.print("Sistem Kendali");
Serial.println();
Serial.println();
Serial.print("Connecting to: ");
lcd.print("Hubungkan ke Hotspot");
Serial.println(WIFI_SSID);
WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
while (WiFi.status() != WL_CONNECTED) {
Serial.print("-");
lcd.print("menghubungkan......");
delay(500);
}
```

```
Serial.println();
Serial.println("WiFi Connected");
lcd.print("OK WiFi Terhubung");
Serial.println();
lcd.setCursor(0, 1);
lcd.print("Pengisian bat.");
delay(2000);
lcd.clear();
config.host = FIREBASE_HOST;
config.signer.tokens.legacy_token = FIREBASE_AUTH;
Firebase.begin(&config, &auth);
Firebase.reconnectWiFi(true);
configTime(8*3600, 0, "pool.ntp.org", "time.nist.gov");
Serial.println("Awaiting NTP sync...");
while (!time(nullptr)) {
Serial.print(".");
delay(500);
}
Serial.println("\nNTP synced");
}
```

Bagian setup pada program ini memiliki peran penting dalam melakukan inisialisasi perangkat keras, koneksi jaringan, tampilan LCD, serta sinkronisasi waktu untuk mendukung sistem kendali dan pemantauan Pembangkit Listrik Tenaga Surya (PLTS) skala laboratorium berbasis IoT. Proses inisialisasi ini hanya dijalankan satu kali ketika perangkat pertama kali dinyalakan atau di-reset..

Proses dimulai dengan mengaktifkan komunikasi serial melalui Serial.begin(9600) yang digunakan untuk menampilkan log dan status sistem pada Serial Monitor, sehingga mempermudah proses debugging. Selanjutnya, modul *Wi-Fi* pada ESP32 diatur dalam mode stasiun dengan perintah *WiFi*.mode(*WIFI*_STA) dan dilakukan pemutusan koneksi sebelumnya menggunakan *WiFi.disconnect*() agar koneksi baru dapat berjalan dengan bersih. *Delay* selama satu detik diberikan untuk memastikan modul *Wi-Fi* siap sebelum melanjutkan proses berikutnya.

Setelah itu, dilakukan konfigurasi perangkat keras terkait sistem pengisian . Pin ADC disesuaikan menggunakan *analogSetAttenuation*(ADC_11db) agar ESP32 mampu membaca tegangan yang lebih tinggi dari input sensor. Pin relay

yang terhubung ke jalur pengisian didefinisikan sebagai output melalui *pinMode*(*RELAY_CHARGE, OUTPUT)* dan diberi logika awal *LOW* untuk memastikan pengisian dalam kondisi mati saat pertama kali sistem dijalankan.

Selanjutnya, inisialisasi LCD dilakukan dengan *lcd.init()* dan *lcd.backlight* untuk menyalakan layar beserta lampu latar. Pesan awal seperti "Sistem Kendali" dan "Hubungkan ke Hotspot" ditampilkan sebagai indikasi proses startup dan pen ghubungan ke jaringan *WiFi*.Proses koneksi ke jaringan dilakukan dengan *WiFi*.be gin(*WIFI_SSID,WIFI_PASSWORD*),kemudian program menunggu hingga status *WiFi* berubah menjadi *WL_CONNECTED*. Selama proses ini,simbol penghubung dicetak di Serial Monitor, dan LCD menampilkan pesan "menghubungkan……" sebagai umpan balik visual bagi pengguna. Setelah koneksi berhasil, pesan "*WiFi Connected*" dan "OK *WiFi* Terhubung" ditampilkan, menandakan bahwa ESP32 siap untuk melakukan komunikasi berbasis internet.

Tahap berikutnya adalah konfigurasi *Firebase.* Alamat *host database* dan token autentikasi ditetapkan ke dalam variabel *config,* kemudian inisialisasi Firebase dilakukan melalui *Firebase.begin(&config,&auth).* Fungsi *Firebase.reco nnectWiFi(true)* juga diaktifkan agar sistem dapat otomatis menyambung kembali apabila koneksi Wi-Fi terputus.

Terakhir, sistem melakukan sinkronisasi waktu dengan server *NTP* menggunakan fungsi configTime(8*3600, 0*, "pool.ntp.org", "time.nist.gov*") di mana offset waktu 8 jam digunakan untuk menyesuaikan zona waktu lokal WITA. Program akan menunggu hingga data waktu berhasil diperoleh, yang ditandai dengan pesan "*NTP synced*" pada Serial Monitor.

b. Penjelasan bagian-bagian fugsi *loop*

1. Fungsi *loop*

```
void loop() {
  float teganganBaterai = getAverageVoltage(ANALOG_IN_PIN);
  unsigned long now = millis();
  String waktu = getDateTime();
  int kapasitas = teganganToPersen(teganganBaterai);

  if (teganganBaterai < 1.0) {
    lcd.setCursor(0, 0);
    lcd.print("Teg:0.00V       ");
    lcd.setCursor(0, 1);
    lcd.print("No Input        ");
    Serial.println("Tegangan: 0.00 V | No Input");
    delay(1000);
    return;
  }

  }
```

Bagian awal dari fungsi *loop*() ini berperan sebagai inti dari proses pemantauan sistem PLTS berbasis IoT yang berjalan secara berulang dan terus-menerus selama perangkat aktif. Fungsi ini mengatur proses pembacaan sensor, perhitungan kapasitas baterai, penentuan kondisi input daya, serta penyajian informasi ke LCD maupun Serial Monitor.

Pertama, sistem membaca tegangan baterai melalui pemanggilan fungsi *getAverageVoltage(ANALOG_IN_PIN)* dan menyimpannya ke variabel teganganB aterai bertipe *float.* Fungsi ini telah dirancang untuk melakukan pengambilan sampel berulang dan perhitungan rata-rata guna memperoleh nilai tegangan yang lebih stabil dan akurat. Nilai waktu dalam satuan milidetik sejak perangkat dinyalakan juga diambil menggunakan *millis*() dan disimpan pada variabel *now*. Kemudian, untuk mencatat penanda waktu aktual, sistem memanggil fungsi *getDateTime*() yang mengembalikan informasi waktu dalam format YYYY-MM-

DD HH:MM:SS. Selain itu, kapasitas baterai dalam bentuk persentase dihitung melalui fungsi tegangan*ToPersen*(teganganBaterai), yang berperan mengonversi nilai tegangan hasil pembacaan menjadi estimasi tingkat pengisian baterai.

Setelah nilai tegangan diperoleh, program memeriksa kondisi awal untuk mendeteksi apakah terdapat input daya yang valid. Apabila nilai tegangan baterai yang terbaca kurang dari 1,0 volt, sistem menganggap bahwa tidak ada sumber daya atau sensor tidak menerima input tegangan yang memadai. Dalam kondisi ini, LCD akan menampilkan pesan "Teg:0.00V" pada baris pertama dan "*No Input*" pada baris kedua, yang memberikan informasi visual kepada pengguna bahwa tidak ada tegangan yang masuk ke sistem. Pesan yang sama juga dikirim ke Serial Monitor dalam format "Tegangan: 0.00 V | *No Input*" sebagai *log* monitoring untuk pemantauan jarak dekat atau proses debugging.

Setelah menampilkan informasi tersebut, program memberikan jeda selama satu detik melalui *delay*(1000) untuk mencegah pembaruan tampilan terlalu cepat, kemudian menjalankan perintah return untuk mengulang siklus *loop*() berikutnya. Mekanisme ini memastikan bahwa sistem tidak melakukan perhitungan kapasitas maupun pengendalian relay jika sumber daya tidak tersedia, sekaligus menjaga kestabilan pemantauan pada kondisi tanpa input tegangan.

Apabila tegangan baterai berada di atas ambang ini, baris kode selanjutnya yang belum ditampilkan akan mengatur logika pengendalian relay pengisian, pembaruan tampilan LCD, dan pengiriman data ke *Firebase* secara *real-time*.

2. Fungsi bagain kendali Relay

```
//funsi kendali relay
if (teganganBaterai >= teganganOff && charging) {
  static unsigned long highVoltageStart = 0;
  if (highVoltageStart == 0) highVoltageStart = now;
  if ((now - highVoltageStart) >= delayRelay) {
    digitalWrite(RELAY_CHARGE, HIGH);
    charging = false;
    lastSwitchTime = now;
Serial.println("Charging OFF (tegangan tinggi stabil)");
        }
} else if (teganganBaterai <= teganganOn && !charging &&
(now - lastSwitchTime > delayRelay)) {
 digitalWrite(RELAY_CHARGE, LOW);
charging = true;
lastSwitchTime = now;
Serial.println("Charging ON (tegangan rendah)");
 } else {
  static unsigned long highVoltageStart = 0;
  highVoltageStart = 0;
}
```

Fungsi kendali relay merupakan inti dari logika pengendalian relay, pengisian baterai pada sistem PLTS berbasis IoT. Logika ini dirancang untuk mengatur kapan pengisian baterai harus dihentikan atau diaktifkan kembali, berdasarkan nilai tegangan yang terukur, dengan mempertimbangkan batas atas (tegangan*Off*) dan batas bawah (tegangan*On*) yang telah ditentukan sebelumnya. Prinsip ini membentuk sistem histeresis, yang bertujuan menjaga kesehatan baterai agar terhindar dari kondisi *overcharging* maupun *deep discharging.*

Proses dimulai dengan pemeriksaan kondisi tegangan tinggi, yaitu ketika nilai teganganBaterai lebih besar atau sama dengan tegangan*Off* dan variabel *charging* bernilai *true*, menandakan bahwa pengisian sedang berlangsung. Dalam kondisi ini, digunakan variabel statis *highVoltageStart* untuk mendeteksi apakah tegangan tinggi ini bertahan stabil dalam periode tertentu. Jika nilai variabel tersebut masih 0, maka waktu saat ini (*now*) disimpan sebagai titik awal. Apabila

tegangan tinggi ini bertahan hingga melewati durasi delayRelay (10 detik), maka relay akan diputus dengan perintah *digitalWrite*(RELAY_*CHARGE, HIGH*), yang menandakan pengisian dihentikan. Variabel *charging* diubah menjadi *false,* l*astSwitchTime* diperbarui dengan waktu saat ini, dan pesan log "*Charging OFF* (tegangan tinggi stabil)" dikirim ke Serial Monitor. Mekanisme ini memastikan relay tidak memutus pengisian hanya karena fluktuasi tegangan sesaat.

Sebaliknya, kondisi tegangan rendah diperiksa pada blok *else if*. Jika teganganBaterai turun sama atau di bawah tegangan*On*, status charging saat ini *false*, dan selisih waktu sejak terakhir *switching* lebih besar dari *delay*Relay, maka relay diaktifkan kembali dengan perintah digitalWrite(RELAY_*CHARGE, LOW*). Hal ini menandakan proses pengisian diaktifkan kembali untuk mencegah baterai memasuki kondisi *deep discharge*. Variabel *charging* dikembalikan ke **true**, *lastSwitchTime* diperbarui, dan pesan *"Charging ON* (tegangan rendah)" dicetak ke Serial Monitor sebagai indikasi proses pengisian telah dimulai kembali.

Pada blok else terakhir, variabel statis *highVoltageStart* direset ke 0 setiap kali kondisi tegangan tidak memenuhi syarat pengisian atau penghentian. Hal ini memastikan penghitungan waktu stabil tegangan tinggi hanya berlaku saat tegangan benar-benar berada di atas batas yang ditentukan.

3. Fungsi bagian tampilan LCD

```
// LCD display
lcd.setCursor(0, 0);
lcd.print("Teg:");
lcd.print(teganganBaterai, 2);
lcd.print("V     ");
lcd.setCursor(0, 1);
lcd.print(charging ? "Chg:ON " : "Chg:OFF");
lcd.setCursor(9, 1);
lcd.print(kapasitas);
lcd.print("% ");
```

Fungsi Tampilan LCD bertanggung jawab untuk menampilkan informasi status sistem secara real-time melalui LCD 16x2 yang terhubung menggunakan komunikasi I2C. Tujuan utama dari tampilan ini adalah memberikan umpan balik visual bagi pengguna agar kondisi baterai dan status pengisian dapat dipantau secara langsung di lokasi tanpa memerlukan koneksi ke *Firebase* atau Serial Monitor.

Proses tampilan diawali dengan pengaturan kursor LCD ke posisi awal baris pertama menggunakan lcd.setCursor(0, 0). Selanjutnya, teks “Teg:” dicetak sebagai label untuk tegangan baterai. Nilai tegangan aktual yang telah diperoleh dari fungsi *getAverageVoltag*e() kemudian ditampilkan menggunakan lcd.print(teganganBaterai, 2) dengan dua angka di belakang koma untuk menampilkan hasil yang lebih presisi dalam satuan *volt*. Setelah itu, beberapa spasi tambahan dicetak untuk membersihkan sisa karakter dari tampilan sebelumnya, memastikan nilai baru selalu terbaca jelas oleh pengguna.

Pada baris kedua, program menampilkan status pengisian. Kursor terlebih dahulu diarahkan ke awal baris kedua dengan lcd.s*etCursor*(0, 1). Status relay ditampilkan menggunakan ekspresi ternary charging ? "*Chg:ON* " : "*Chg:OFF*", yang berarti jika variabel charging bernilai *true,* LCD akan menampilkan teks “*Chg:ON*”, menandakan proses pengisian sedang aktif. Sebaliknya, jika bernilai *false*, LCD menampilkan *“Chg:OFF*”, menunjukkan bahwa pengisian sedang dihentikan karena tegangan baterai telah mencapai batas atas.

Selanjutnya, kursor dipindahkan ke kolom ke 9 baris kedua (lcd.*setCursor*(9, 1)) untuk menampilkan kapasitas baterai dalam persentase. Nilai kapasitas yang

telah dihitung oleh fungsi tegangan*ToPerse*n() ditampilkan diikuti dengan simbol **“%”** untuk memberikan konteks yang jelas bagi pengguna. Dengan pengaturan tampilan ini, LCD secara simultan menampilkan tiga informasi penting: tegangan baterai, status pengisian relay, dan persentase kapasitas baterai.

4. Fungsi bagian *Firebase* pengirim

```
// Simpan data realtime (update)
Firebase.setString(fbdo,
"/solar_system/realtime_data/kapasitas_baterai",
String(kapasitas));
Firebase.setString(fbdo,
"/solar_system/realtime_data/status_charging", charging ?
"on" : "off");
```

Bagian kode ini berfungsi untuk menyimpan dan memperbarui data sistem secara real-time pada *Firebase Realtime Database*, yang menjadi inti dari kemampuan IoT (*Internet of Things*) pada sistem PLTS skala laboratorium ini. Data yang dikirim meliputi kapasitas baterai dan status pengisian, sehingga pengguna dapat memantau kondisi sistem dari jarak jauh melalui internet tanpa harus berada di lokasi.

Proses dimulai dengan pemanggilan fungsi *Firebase.setString*() yang mengirimkan data ke server *Firebase* melalui objek komunikasi fbdo. Baris pertama mengirimkan nilai kapasitas yang sebelumnya telah dihitung dari tegangan baterai ke *path /solar_system/realtime*_data/kapasitas_baterai. Nilai ini dikonversi terlebih dahulu menjadi String agar kompatibel dengan format data yang diterima oleh *Firebase Realtime Database*. Informasi kapasitas ini penting untuk mengetahui tingkat pengisian baterai dalam persentase, yang membantu pengguna menentukan kondisi cadangan energi pada sistem PLTS.

Baris kedua mengirimkan status pengisian baterai dalam bentuk string *"on"* atau *"off",* yang ditentukan berdasarkan nilai *boolean charging*. Apabila relay dalam kondisi aktif, data yang dikirim adalah "*on*", menandakan bahwa baterai sedang diisi. Sebaliknya, jika relay mati, data yang dikirim adalah "*off*", yang berarti proses pengisian dihentikan karena tegangan telah mencapai batas aman. Data ini disimpan pada *path /solar_system/realtime*_data/status_*charging* di *Firebase*, yang dapat ditampilkan dalam aplikasi pemantauan atau dashboard berbasis web.

5. Fungsi bagian *Firebase Json*

```
// Simpan data historis (push)
 FirebaseJson json;
json.set("waktu", String(waktu ));
json.set("tegangan_baterai",
String(teganganBaterai, 2));
json.set("kapasitas_baterai", String(kapasitas));
json.set("status_charging", charging ? "on" : "off");

if (Firebase.pushJSON(fbdo, "/solar_system/logs", json))
{
Serial.println("✓ Data berhasil dipush ke
/solar_system/logs");
  } else {
    Serial.print("✗ Gagal push Firebase: ");
    Serial.println(fbdo.errorReason());
  }
  delay(1000);
}
```

Bagian akhir dari fungsi loop() ini berfungsi untuk menyimpan data historis ke *Firebase Realtime Database*, yang memungkinkan seluruh aktivitas sistem PLTS terekam dengan rapi untuk keperluan pemantauan jangka panjang dan analisis performa. Mekanisme yang digunakan adalah push data dengan format *JSON,* sehingga setiap entri baru akan tersimpan sebagai catatan terpisah di bawah *path /solar_system/logs,* tanpa menimpa data sebelumnya.

Proses dimulai dengan pembuatan objek *FirebaseJson json*, yang menjadi wadah untuk menyusun data dalam format *JSON.* Data yang dikirim terdiri dari empat parameter utama. Pertama, “waktu” yang diisi dengan nilai waktu dari fungsi *getDateTim*e(), berfungsi sebagai timestamp untuk setiap catatan log. Kedua, “tegangan_baterai” yang berisi nilai tegangan hasil pembacaan ADC, dikonversi menjadi String dengan dua angka di belakang koma untuk menjaga presisi. Ketiga, “kapasitas_baterai” yang menampilkan persentase pengisian baterai, sehingga memudahkan pengguna mengetahui kondisi cadangan energi. Keempat, “status_charging” yang menunjukkan kondisi relay pengisian, ditulis sebagai “*on*” jika baterai sedang diisi, atau “o*ff*” jika pengisian dihentikan.

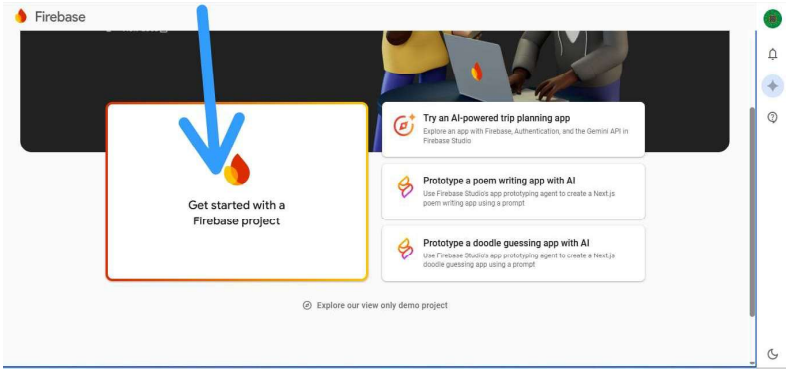
Setelah data *JSON* lengkap, program memanggil fungsi *Firebase.pushJSON*(fbdo, "/*solar_system/logs", json*) untuk mengirimkannya ke *Firebase.* Jika pengiriman berhasil, Serial Monitor akan menampilkan pesan “✓ Data berhasil dipush ke /*solar_system/logs*” sebagai indikasi keberhasilan pencatatan data. Sebaliknya, jika terjadi kegagalan, sistem akan menampilkan pesan “✗ Gagal *push Firebase*” diikuti dengan alasan error dari fbdo.*errorReason*(). Mekanisme ini membantu proses debugging dan memastikan integritas pengiriman data ke *cloud.*

Akhirnya, diberikan jeda delay(1000) selama satu detik sebelum siklus loop() diulang kembali. Penundaan ini berfungsi untuk mengatur frekuensi pengiriman data agar tidak terlalu cepat, sehingga penggunaan *bandwidth internet* dan beban *Firebase* tetap efisien.

c. pembuatan dan Stuktur *firebase database*

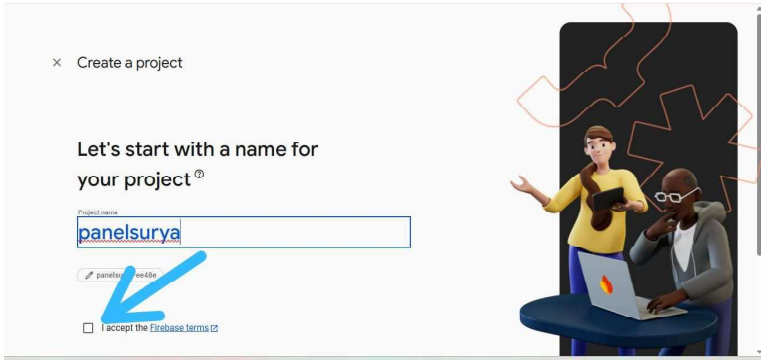
Beberapa langkah pembuatan *firebase database*

1. Klik kotak "*Get started with a Firebase project*."



Gambar 4. 6 *Get started with a Firebase project*

2. *Create a Project* – dan Isi Nama Project



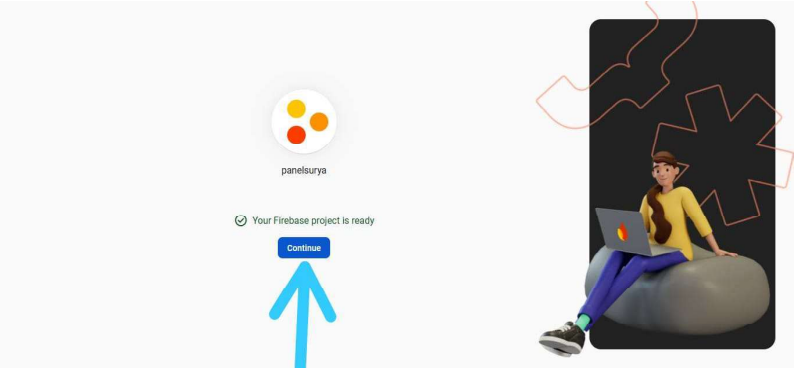
Gambar 4. 7 *Create a Project* dan Nama Project

3. *Create a Project* – Konfigurasi Opsional



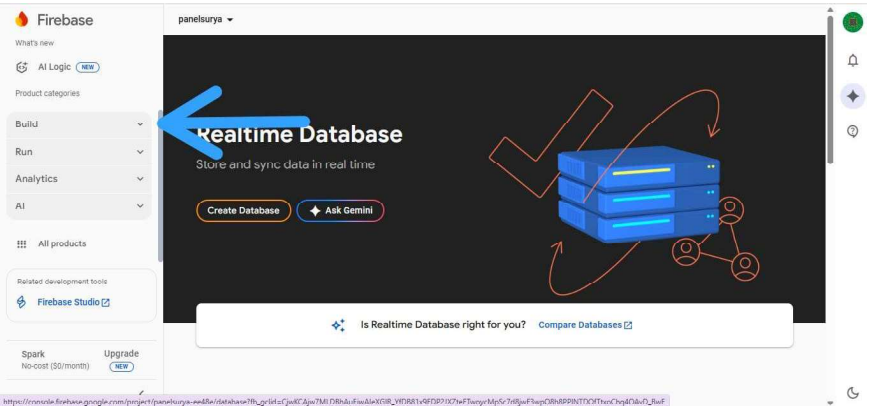
Gambar 4. 8 *Create a Project* dan Konfigurasi Opsional

4. *Firebase project* siap



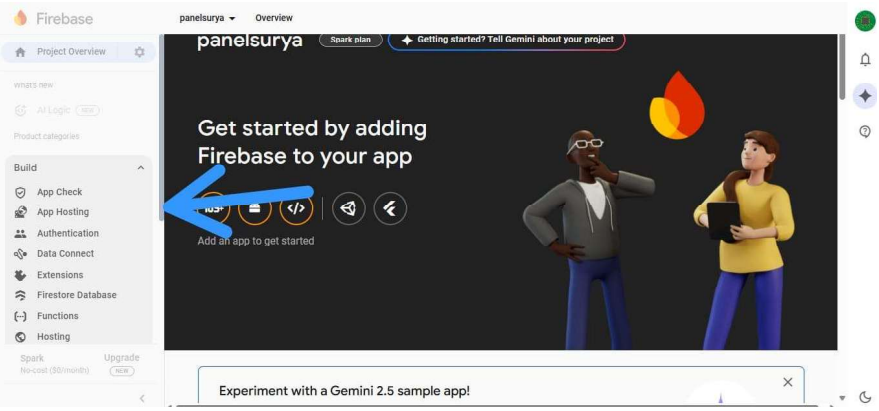
Gambar 4. 9 *Project* Siap

5. Masuk ke *Realtime Database*



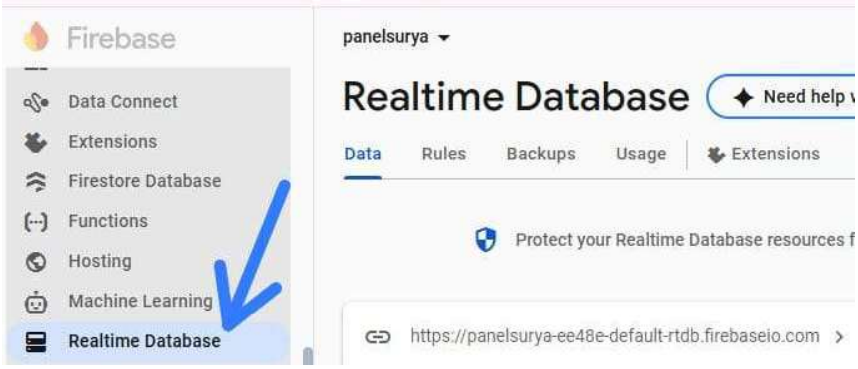
Gambar 4. 10 Tampilan *Firebase Console*

6. Cari Realtime *database* di *Firebase console*



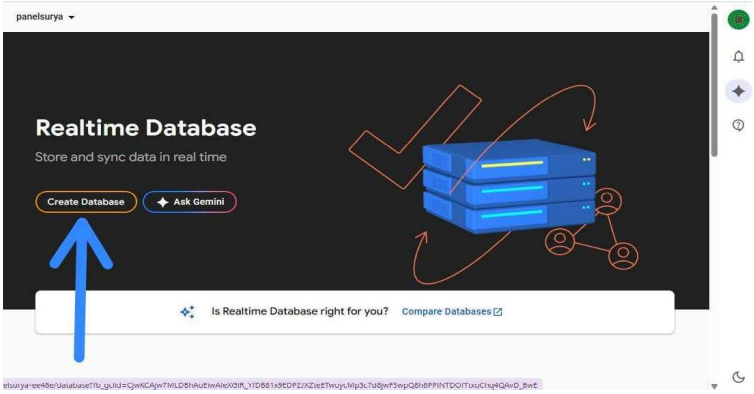
Gambar 4. 11 Tampilan *Firebase Console*

7. Piih dan dan klik *Realtime Database*



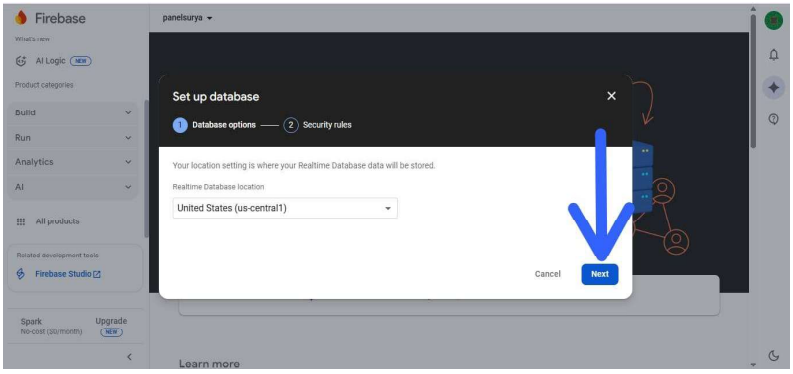
Gambar 4. 12 Tampilan *Firebase console*

8. Lalu Klik *Create Database*



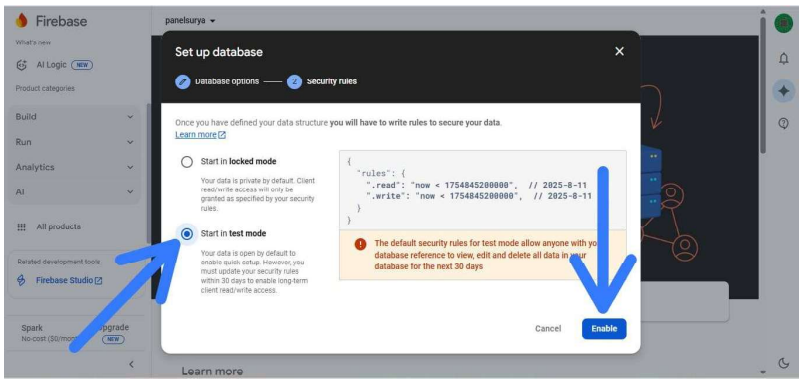
Gambar 4. 13 *Create Database*

9. Pilih Region Dan Klik *Next*



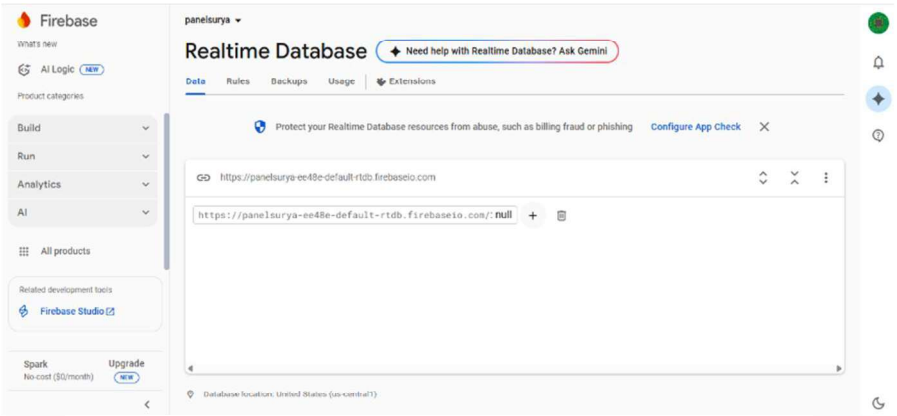
Gambar 4. 14 *Region firebase*

10. Klik *Start in test* dan klik *Enable*



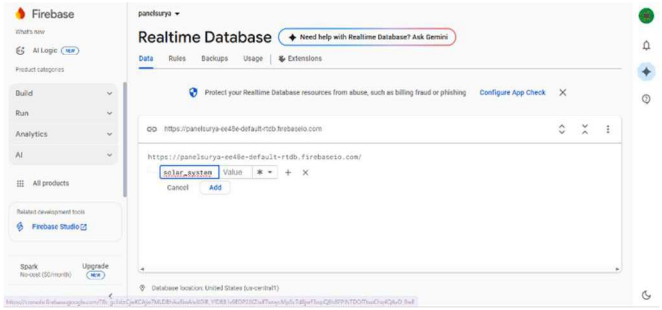
Gambar 4. 15 *Start* dan *enable Firebas console*

11. Setelah itu kita akan membuat susunan tampilan *firebase database realtime* degan megklik simbol +



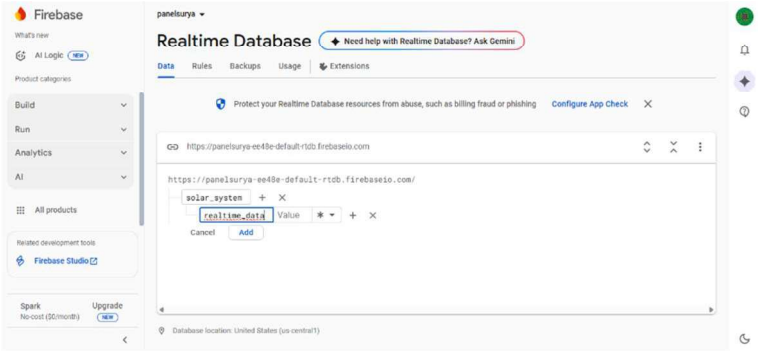
Gambar 4. 16 *Realtime database* Tampilan Awal

12. Tekan simbol + lalu ketik *solar system*



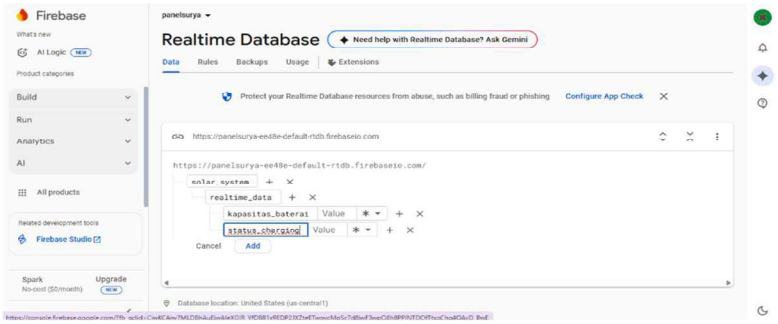
Gambar 4. 17 Tampilan *Realtime Firebase solar system*

13. Lalu klik simbol + yg ada di samping solar system lalu ketik *Realtime_* data



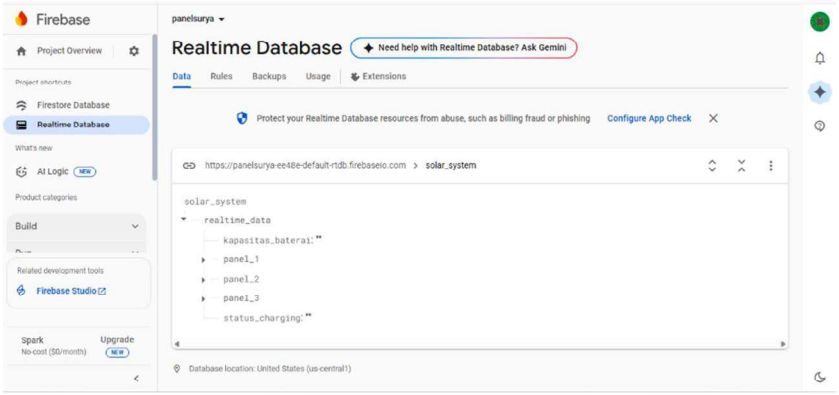
Gambar 4. 18 Tampilan Peyimpanan Realtime data

14. Klik kembali simbol + lalu ketik kapasitas_baterai dan *System_charging*



Gambar 4. 19 Tampilan Peyimpanan data Kapasitas baterai dan Status *charging*

15. Tampilan Akhir *firebase database*



Gambar 4. 20 Tampilan Akhir Susunan peyimpanan data *firebase*

d. Fungsi Program Tampilan aplikasi web

Program ini merupakan antarmuka web untuk sistem monitoring dan kendali Pembangkit Listrik Tenaga Surya (PLTS) skala laboratorium yang dibangun berbasis *Internet of Things* (IoT). Sistem ini dirancang untuk memberikan akses *real-time* terhadap data penting seperti tegangan baterai, status pengisian (*charging*), dan kapasitas daya tersimpan.Antarmuka web ini terintegrasi langsung dengan *Firebase Realtime Database*, memungkinkan pembaruan data secara otomatis dan tanpa jeda. Melalui koneksi ini, setiap perubahan nilai sensor yang terjadi di sisi perangkat keras akan langsung ditampilkan di halaman web. Dengan demikian, pengguna dapat memantau kondisi sistem PLTS secara akurat dan terus-menerus dari jarak jauh.Tampilan web dirancang responsif dan ramah pengguna, dilengkapi elemen visual seperti ikon status dan penggunaan gaya CSS modern. Hal ini dilakukan agar informasi yang disajikan tidak hanya akurat tetapi juga mudah dipahami secara visual. Elemen ikon diperoleh dari pustaka *Font Awesome* untuk memperjelas status sistem seperti mode pengisian aktif atau tidak aktif.Secara

keseluruhan, program ini berfungsi sebagai sarana visualisasi data sistem PLTS yang informatif dan efisien, serta mendukung proses pengambilan keputusan dalam pengendalian sistem berbasis IoT yang sedang dikembangkan.

1. Fungsi file HTML

```
<!DOCTYPE html>
<html lang="id">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
  <title>Monitoring Panel Surya IoT</title>
  <link   href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.0.0-beta3/css/all.min.css" rel="stylesheet">
  <script
src="https://www.gstatic.com/firebasejs/5.0.4/firebase.js"><
/script>
  <link rel="stylesheet" href="style.css" />
  <script defer src="script.js"></script>
</head>

<body>
  <h1>Sistem Monitoring & Kendali Panel Surya</h1>

  <div class="grid-row">
    <div class="card">
      <h2>Monitoring Panel 1</h2>
      <i class="fas fa-solar-panel"></i>
      <div class="sensor">
        <div    class="tegangan"><p    id="teganganPanel1">0
V</p></div>
        <div class="suhu"><p id="suhuPanel1">0 °C</p></div>
        <div class="lux"><p id="luxPanel1">0 lux</p></div>
      </div>
    </div>

    <div class="card">
      <h2>Monitoring Panel 2</h2>
      <i class="fas fa-solar-panel"></i>
      <div class="sensor">
        <div    class="tegangan"><p    id="teganganPanel2">0
V</p></div>
        <div class="suhu"><p id="suhuPanel2">0 °C</p></div>
        <div class="lux"><p id="luxPanel2">0 lux</p></div>
      </div>
    </div>

    <div class="card">
      <h2>Monitoring Panel 3</h2>
      <i class="fas fa-solar-panel"></i>
      <div class="sensor">
```

```
        <div    class="tegangan"><p    id="teganganPanel3">0
V</p></div>
        <div class="suhu"><p id="suhuPanel3">0 °C</p></div>
        <div class="lux"><p id="luxPanel3">0 lux</p></div>
      </div>
    </div>
  </div>

  <div class="grid-row">
    <div class="card">
      <h2>Kapasitas Baterai</h2>
      <i class="fas fa-battery-half"></i>
      <div class="sensor">
        <div    class="battery"><p    id="kapasitasBaterai">0
%</p></div>
      </div>
    </div>

    <div class="card">
      <h2>Kondisi Charger</h2>
      <i class="fas fa-plug"></i>
      <div class="sensor">
        <div                              class="status"><p
id="statusCharging">off</p></div>
      </div>
    </div>
  </div>

  <footer>
    <p>© 2025 Monitoring System</p>
  </footer>
</body>
</html>
```

Fungsi program HTML in adalah Tampilan antarmuka dari sistem monitoring dan kendali panel surya ini dirancang menggunakan struktur dasar HTML5 dengan bahasa Indonesia sebagai pengaturan lokalnya. Penggunaan elemen-elemen HTML pada halaman ini bertujuan untuk menyajikan data pemantauan dari tiga panel surya sekaligus, disertai informasi pendukung seperti kapasitas baterai dan status pengisian daya secara *real-time*. Di bagian awal, halaman ini memuat pengaturan karakter UTF-8 dan konfigurasi viewport yang

membuat tampilan dapat menyesuaikan secara otomatis dengan ukuran layar perangkat pengguna, baik pada komputer maupun perangkat mobile.

Judul halaman ditetapkan sebagai “Monitoring Panel Surya IoT”, dan dalam <head> juga disisipkan pustaka eksternal seperti *Font Awesome* untuk ikon visual, serta pustaka Firebase versi 5.0.4 yang digunakan untuk koneksi dengan Firebase Realtime Database. File CSS eksternal (style.css) dan JavaScript (script.js) juga diimpor untuk mengatur tampilan visual dan fungsionalitas halaman. Seluruh struktur dirancang agar sederhana namun informatif, sehingga pengguna dapat dengan mudah memahami data yang disajikan.

Di dalam elemen <*body*>, antarmuka dimulai dengan judul utama yang menunjukkan fungsi sistem, yaitu pemantauan dan pengendalian panel surya. Tampilan data disusun dalam format grid responsif yang terbagi ke dalam beberapa kartu (card). Setiap kartu mewakili satu komponen sistem, seperti Panel 1, Panel 2, Panel 3, kapasitas baterai, dan kondisi pengisian daya. Masing-masing kartu dilengkapi ikon pendukung dan menampilkan data dalam satuan yang sesuai, misalnya volt (V) untuk tegangan, derajat *Celcius* (°C) untuk suhu, serta *lux* untuk intensitas cahaya. Selain itu, status pengisian daya ditampilkan dalam bentuk teks “*on”* atau *“off”,* sedangkan kapasitas baterai ditampilkan dalam persen (%), yang memberikan gambaran visual terhadap kondisi sistem secara menyeluruh.

Setiap nilai pada halaman ini akan diperbarui secara otomatis melalui koneksi ke *Firebase,* sehingga memungkinkan pengguna untuk memantau kondisi sistem secara *real-time* tanpa perlu memuat ulang halaman. Pada bagian akhir

halaman terdapat footer sederhana yang mencantumkan hak cipta sebagai identitas sistem.

2. Fungsi file CSS

```
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: sans-serif;
}

body {
  background: #f1f1f1;
  padding: 20px;
}

h1 {
  text-align: center;
  margin-bottom: 20px;
  color: #4CAF50;
  font-size: 2em;
}

.grid-row {
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));
  gap: 20px;
  max-width: 1200px;
  margin: 0 auto 20px auto;
}

.card {
  background-color: #fff;
  border-radius: 10px;
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.15);
  padding: 20px;
  text-align: center;
  background: linear-gradient(135deg, #e0f7fa, #fce4ec);
  transition: transform 0.2s;
}

.card:hover {
  transform: scale(1.05);
}

.card i {
  font-size: 3em;
  color: #00796B;
  margin-bottom: 10px;
}

h2 {
  font-size: 1.5em;
```

```
  color: #333;
  margin-bottom: 10px;
}

.sensor {
  display: flex;
  flex-wrap: wrap;
  justify-content: center;
  gap: 10px;
  margin-top: 10px;
}

.sensor div {
  background-color: #fff;
  color: #333;
  padding: 10px;
  border-radius: 10px;
  width: 45%;
  box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
  text-align: center;
}

.tegangan {
  background-color: #e0f7fa;
  color: #00796B;
}

.suhu {
  background-color: #ffeef2;
  color: #e91e63;
}

.lux {
  background-color: #ffe0b2;
  color: #f57c00;
}

.status {
  background-color: #f3e5f5;
  color: #6A1B9A;
}

.battery {
  background-color: #fff3e0;
  color: #ff9800;
}
p {
  font-size: 1.5em;
  margin: 0;
}
footer {
  text-align: center;
  margin-top: 30px;
  color: #888;
  font-size: 0.9em;
}
```

Fungsi program CSS adalah Tampilan visual dari sistem monitoring dan kendali panel surya ini diatur menggunakan file CSS eksternal yang dirancang untuk menghasilkan antarmuka yang responsif, rapi, dan mudah dipahami oleh pengguna. Pada bagian awal pengaturan, digunakan reset gaya dengan menghilangkan margin dan padding default, serta menerapkan model *box-sizing*: border-box agar semua elemen memiliki penghitungan ukuran yang konsisten. Seluruh elemen teks menggunakan font standar sans-serif untuk kesan bersih dan modern.

Latar belakang halaman diberikan warna abu-abu terang (#f1f1f1) untuk mengurangi ketegangan mata saat pengguna mengamati data dalam waktu lama. Judul utama (<h1>) ditempatkan di tengah halaman dengan ukuran font yang cukup besar dan warna hijau cerah, mencerminkan tema ramah lingkungan dari sistem berbasis energi surya. Tata letak data disusun dalam grid yang adaptif, memungkinkan setiap kartu informasi (*card*) menyesuaikan diri terhadap lebar layar perangkat, baik itu desktop maupun ponsel.

Setiap kartu memiliki gaya yang seragam: latar belakang berwarna putih, sudut membulat, dan efek bayangan (*box-shadow*) untuk memberikan kesan mengambang. Warna latar kartu menggunakan gradasi lembut yang memberikan sentuhan estetika tanpa mengganggu keterbacaan. Saat pengguna mengarahkan kursor ke salah satu kartu, diterapkan efek perbesaran (*transform*: s*cale*(1.05))

sebagai umpan balik visual interaktif. Ikon dalam setiap kartu diberikan warna hijau kebiruan yang konsisten dengan tema lingkungan dan energi bersih.

Komponen data sensor disusun dalam elemen flex, memungkinkan penataan elemen dalam baris yang fleksibel. Masing-masing parameter seperti tegangan, suhu, intensitas cahaya (l*ux*), status *charging*, dan kapasitas baterai diberikan warna latar yang berbeda-beda untuk memudahkan identifikasi visual secara cepat. Misalnya, warna biru muda untuk tegangan, merah muda untuk suhu, oranye terang untuk lux, ungu untuk status pengisian, dan oranye pucat untuk baterai. Gaya paragraf dalam kartu juga disesuaikan dengan ukuran besar agar mudah dibaca.

Terakhir, bagian footer disusun dengan gaya sederhana, menampilkan informasi hak cipta sistem, dan diletakkan secara terpusat di bagian bawah halaman.

3. Fungsi file *Javascipt*

```
var config = {
  apiKey: "AIzaSyAiBrr9yTwIdIyKxy337PdkGkjXZVM_lNs",
  databaseURL:            "https://panelsurya-f15ad-default-
rtdb.firebaseio.com/"
};
firebase.initializeApp(config);
var                          db                              =
firebase.database().ref("solar_system/realtime_data");

const timeout = 15000; // 15 detik
let lastUpdate = {
  panel_1: 0,
  panel_2: 0,
  panel_3: 0,
  status: 0
};

function updatePanel(panelID, snapshot) {
  const p = snapshot.child(panelID);
  document.getElementById("teganganPanel" + panelID.slice(-
1)).innerHTML = p.child("tegangan_panel").val() + " V";
  document.getElementById("suhuPanel"   +   panelID.slice(-
1)).innerHTML = p.child("suhu_panel").val() + " °C";
  document.getElementById("luxPanel"    +    panelID.slice(-
1)).innerHTML = p.child("lux_panel").val() + " lux";
```

```
  lastUpdate[panelID] = Date.now();
}

function resetPanel(panelID) {
  document.getElementById("teganganPanel" + panelID.slice(-
1)).innerHTML = "0 V";
  document.getElementById("suhuPanel"   +   panelID.slice(-
1)).innerHTML = "0 °C";
  document.getElementById("luxPanel"    +    panelID.slice(-
1)).innerHTML = "0 lux";
}

function updateStatus(snapshot) {
  document.getElementById("kapasitasBaterai").innerHTML    =
snapshot.child("kapasitas_baterai").val() + " %";
  document.getElementById("statusCharging").innerHTML      =
snapshot.child("status_charging").val();
  lastUpdate.status = Date.now();
}

function resetStatus() {
  document.getElementById("kapasitasBaterai").innerHTML = "0
%";
  document.getElementById("statusCharging").innerHTML      =
"off";
}
// Firebase listener
db.on("value", function(snapshot) {
  if                    (snapshot.child("panel_1").exists())
updatePanel("panel_1", snapshot);
  if                    (snapshot.child("panel_2").exists())
updatePanel("panel_2", snapshot);
  if                    (snapshot.child("panel_3").exists())
updatePanel("panel_3", snapshot);
  if    (snapshot.child("kapasitas_baterai").exists()    &&
snapshot.child("status_charging").exists()) {
    updateStatus(snapshot);
  }
});
// Timeout pengecekan tiap 5 detik
setInterval(() => {
  const now = Date.now();
  ["panel_1", "panel_2", "panel_3"].forEach(panel => {
    if (now - lastUpdate[panel] > timeout) resetPanel(panel);
  });
  if (now - lastUpdate.status > timeout) resetStatus();
}, 5000);
```

Fungsi program *JavaScript* adalah berperan sebagai penghubung antara antarmuka pengguna (*frontend*) dengan *Firebase Realtime Database*, khususnya pada bagian pemantauan data panel surya secara langsung. Pertama-tama, sistem melakukan inisialisasi koneksi *Firebase* menggunakan API key dan URL *database*

yang telah dikonfigurasi. Referensi utama data disimpan dalam variabel db, yang mengarah langsung ke jalur *solar_system/realtime*_data di dalam struktur Firebase.

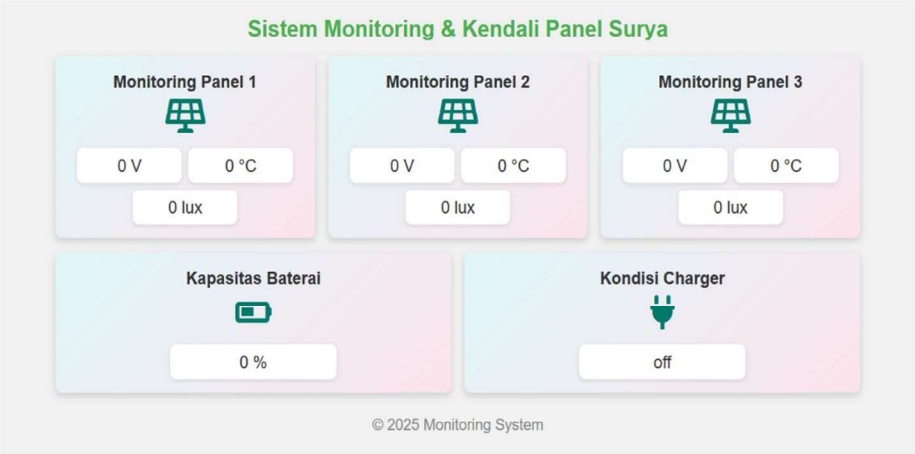
Untuk menghindari penampilan data yang tidak akurat akibat keterlambatan pengiriman dari perangkat, sistem menetapkan batas waktu (*timeout*) selama 15 detik. Jika dalam periode tersebut tidak ada pembaruan data, maka nilai sensor akan di-reset secara otomatis ke angka nol. Hal ini dilakukan untuk menjaga validitas informasi yang ditampilkan di halaman.

Fungsi updatePanel bertugas memperbarui data sensor dari masing-masing panel surya, yaitu tegangan (dalam *volt*), suhu (dalam derajat *Celsius*), dan intensitas cahaya (dalam satuan *lux*). Data ini akan ditampilkan secara real-time ke dalam elemen HTML yang sesuai. Sebagai tambahan, fungsi resetPanel akan menampilkan angka nol ketika data dari salah satu panel tidak diperbarui dalam waktu tertentu, memberikan indikator visual bahwa data tersebut tidak lagi valid.

Selanjutnya, fungsi updateStatus digunakan untuk menampilkan status kapasitas baterai dalam bentuk persentase serta kondisi pengisian (charging), apakah dalam keadaan aktif (*on*) atau tidak (*off*). Bila data ini tidak diperbarui tepat waktu, maka fungsi resetStatus akan secara otomatis mengatur kembali tampilan ke nilai default, yaitu "0 %" dan "off".

Sistem juga menggunakan mekanisme listener dari *firebase* (db.on("value", ...)) yang akan aktif setiap kali terjadi perubahan data pada jalur yang dipantau. Dengan demikian, setiap pembaruan dari perangkat ESP32 akan langsung muncul di antarmuka pengguna tanpa perlu melakukan penyegaran halaman secara manual. Untuk memastikan konsistensi data, interval pengecekan dijalankan setiap lima

detik, yang akan memverifikasi apakah data masih dalam rentang waktu yang valid, serta menjalankan fungsi reset jika dibutuhkan.

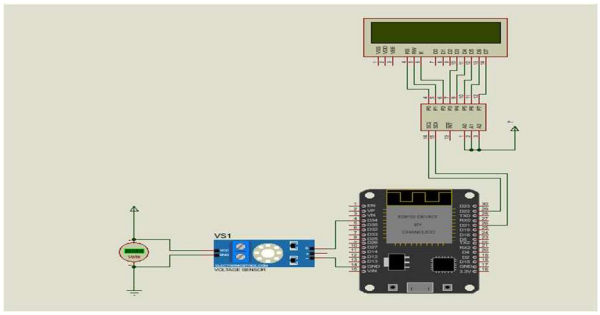


Gambar 4. 21 Tampilan antarmuka web

C. Pengujian

1. Pengujian sesnsor tegangan

Pengambilan data pada sensor tegangan dilakukan untuk mengetahui kemampuan pembacaan sensor, sensor tegangan yang digunakan terhubung dengan pada pin analog mikrokontroler dan dilakukan dengan menyambungkan secara seri dengan power suply dengan mengatur tegangan yang secara bertahap dari 0 – 9,7volt dan untuk memudahlan pengamatan secara langsung terhadap pembacaan sensor digunkan lcd .pengujian sensor tegangan dlihat pada gambar dibawah ini.



Gambar 4. 22 Pengujian Sensor Tegangan

Tabel 4. 1 Pengujian sensor Tegangan

no	power suply	Pegujian	Hasil (volt)
1	1	1	1,085
		2	1,088
		3	1,091
2	1,3	1	1,350
		2	1,352
		3	1,353
3	3	1	3,116
		2	3,116
		3	3,083
4	3,4	1	3,444
		2	3,446
		3	3,447
5	5	1	5,117
		2	5,121
		3	5,123
6	5,5	1	5,595
		2	5,605
		3	5,616
7	7	1	7,09
		2	7,09
		3	7,10
8	7,6	1	7,70
		2	7,72
		3	7,76
9	9	1	9,09
		2	9,14
		3	9,14
10	9,7	1	9,81
		2	9,82
		3	9,82

Pengujian dilakukan untuk mengetahui performa sensor dalam merespons variasi tegangan input dari sumber daya listrik (*power supply*). Sensor diuji pada sepuluh tingkat tegangan berbeda, mulai dari 1 volt hingga 9,7 volt. Setiap titik pengujian dilakukan sebanyak tiga kali guna memastikan kestabilan serta konsistensi data yang diperoleh.

Berdasarkan data pada Tabel,diperoleh bahwa sensor menunjukkan respon yang konsisten dan stabil pada seluruh variasi tegangan. Contohnya, pada tegangan input sebesar 1 volt, output sensor tercatat sebesar 1,085 V, 1,088 V, dan 1,091 V, dengan deviasi maksimal sebesar 0,006 V. Begitu pula pada tegangan 9,7 volt, output sensor berada pada kisaran 9,81 V hingga 9,82 V. Nilai deviasi antar pengulangan yang sangat kecil ini mengindikasikan bahwa sensor memiliki presisi tinggi dan kinerja yang stabil, bahkan dalam pengukuran berulang.

Selain kestabilan, sensor juga menunjukkan karakteristik linier terhadap input yang diberikan. Kenaikan tegangan input menghasilkan kenaikan output sensor secara proporsional. Misalnya, tegangan input sebesar 3,4 volt menghasilkan output sekitar 3,446 V, sedangkan input 7,6 volt menghasilkan output sekitar 7,727 V. Tren linier ini sangat penting dalam sistem kendali dan pemantauan, karena memudahkan proses kalibrasi dan interpretasi data sensor.

2. Pengujian sistem keseluruhan

    a. Pengujian Kondisi *Charging*

Pengujian sistem dilakukan untuk mengevaluasi kinerja dari sistem kendali otomatis yang dirancang pada pembangkit listrik tenaga surya (PLTS) skala laboratorium berbasis *Internet of Things* (IoT). Fokus utama pengujian adalah

melihat bagaimana sistem membaca data tegangan baterai, menentukan kapasitas, serta mengatur kondisi relay sebagai pemutus atau penghubung antara panel surya dan sistem pengisian baterai.

Sistem dirancang untuk bekerja secara otomatis berdasarkan nilai ambang batas tegangan baterai yang telah ditentukan. Relay akan dinyalakan *(ON*) jika baterai berada di bawah ambang batas bawah (≤12.2V), dan akan dimatikan (*OFF*) jika tegangan baterai telah mencapai ambang batas atas (≥13.8V). Seluruh data disimpan secara real-time ke *Firebase* dan ditampilkan melalui LCD, sesuai rancangan sistem pada *flowchart* yang telah dijelaskan sebelumnya.

Pengujian dilakukan selama tiga hari berturut-turut dengan mencatat nilai tegangan baterai (dalam *Volt*), kondisi relay (*ON/OFF*), dan kapasitas baterai (dalam persen). Hasil pengujian disajikan dalam bentuk tabel dan dianalisis sebagai berikut:

1. Pengujian Hari Rabu, 9 Juli 2025

Pengujian pertama dilakukan pada hari Rabu dimulai pukul 10.02 WITA. Pada saat itu, sistem mencatat tegangan awal sebesar 12.80V dengan kapasitas baterai 55%. Relay dalam keadaan *ON* karena sistem mendeteksi bahwa baterai masih dalam keadaan belum penuh.

Seiring waktu, terjadi peningkatan tegangan secara bertahap yang menunjukkan proses pengisian berjalan efektif. Pada pukul 11.58 WITA, tegangan baterai mencapai 14.12V dengan kapasitas 92%. Sesuai dengan logika kendali pada sistem, relay kemudian secara otomatis dimatikan (*OFF*) karena tegangan sudah melampaui ambang batas atas.

Hal ini menunjukkan bahwa sistem bekerja sesuai yang diharapkan. Pengendalian relay berjalan otomatis berdasarkan pembacaan nilai tegangan baterai, dan sistem berhasil menghindari *overcharging* yang dapat merusak baterai.

Tabel 4. 2 Pegujian Rabu 9 Juli 2025

No	Waktu	Tegangan baterai(V)	Kondisi Relay	Kapasitas baterai(%)
1	10.02	12.80	ON	55
2	10.22	13.22	ON	65
3	10.38	13.28	ON	66
4	10.48	13.33	ON	68
5	11.26	13.62	ON	78
6	11.47	13.66	ON	80
7	11.58	14.12	OFF	92

Berdasarkan hasil pengujian, pengisian baterai dilakukan dari kondisi awal pada pukul 10:02 dengan tegangan baterai 12,80 V atau setara dengan 55% kapasitas. Proses pengisian berlangsung hingga sistem mencapai kondisi *OFF* pada pukul 11:58 dengan tegangan baterai 14,12 V atau sekitar 92% kapasitas. Dengan demikian, terjadi kenaikan kapasitas baterai sebesar:

$$\Delta\% = 92\% - 55\% = 37\%$$

Baterai yang digunakan memiliki kapasitas nominal 12 V – 3 Ah, sehingga total energi yang dapat disimpan adalah:

$$E_{baterai} = 12 \times 3 = 36 \text{ Wh}$$

Energi yang benar-benar tersimpan selama periode pengisian adalah:

$$\Delta E = 37\% \times 36 \text{ Wh} = 13{,}32 \text{ Wh}$$

Waktu pengisian dari awal hingga sistem *OFF* adalah 1 jam 56 menit atau 1,933 jam. Dengan demikian, daya rata-rata yang benar-benar masuk ke baterai dapat dihitung sebagai:

$$Pavg = \frac{\Delta E}{\Delta t} - \frac{13{,}32}{1{,}993} = 6{,}89W$$

Padahal, panel surya yang digunakan memiliki kapasitas nominal 50 Wp. Jika diasumsikan panel menghasilkan daya penuh selama periode tersebut, energi yang seharusnya tersedia adalah:

$$\text{Epanel}=50\times1{,}933=96{,}67 \text{ Wh}$$

Maka efisiensi efektif sistem selama proses pengisian dapat dihitung sebagai:

$$\boldsymbol{\eta} efektif = \frac{\Delta E}{Epanel} \times 100\% = \frac{13{,}32}{96{,}67} \times 100\% = 13{,}8\%$$

Secara teoritis, waktu yang dibutuhkan untuk mengisi energi sebesar 13,32 Wh menggunakan panel 50 W dengan asumsi efisiensi 70% hanya:

$$\boldsymbol{\tau teori} = \frac{13{,}32}{50 - 0{,}7} = 0{,}3806 \approx 22{,}8\, menit$$

2. Pengujian Hari Kamis, 10 Juli 2025

Pada pengujian hari Kamis, sistem mulai bekerja pada tegangan baterai sebesar 12.58V dan kapasitas 43%. Sama seperti hari sebelumnya, relay tetap dalam kondisi *ON* selama proses pengisian. Terjadi peningkatan tegangan hingga mencapai 13.85V dengan kapasitas 86% pada pukul 11.39 WITA.Pada titik tersebut, sistem memutuskan untuk mematikan relay secara otomatis. Hal ini menunjukkan bahwa sistem telah mencapai ambang tegangan atas dan menonaktifkan pengisian untuk menjaga kestabilan sistem.

Dibandingkan dengan hari Rabu, proses pengisian hari Kamis menunjukkan kinerja yang sedikit lebih rendah dari sisi kapasitas akhir (92% → 86%). Hal ini dapat disebabkan oleh intensitas cahaya matahari yang lebih rendah, mengingat sistem mengandalkan sumber energi dari panel surya.Meskipun demikian, sistem tetap mampu melakukan pengendalian relay secara mandiri dan mencatat seluruh data ke *Firebase*.

Tabel 4. 3 Pegujian Kamis 10 Juli 2025

No	Waktu	Tegangan baterai(V)	Kondisi Relay	Kapasitas baterai(%)
1	10.13	12.58	ON	43
2	10.33	12.84	ON	52
3	10.54	13.08	ON	60
4	11.06	13.30	ON	67
5	11.23	13.49	ON	74
6	11.31	13.70	ON	81
7	11.39	13.85	OFF	86

Berdasarkan hasil pengujian, pengisian baterai dilakukan dari kondisi awal pada pukul 10:13 dengan tegangan baterai 12,58 V atau setara dengan 43% kapasitas. Proses pengisian berlangsung hingga sistem mencapai kondisi *OFF* pada pukul 11:39 dengan tegangan baterai 13,85 V atau sekitar 86% kapasitas.

Dengan demikian, terjadi kenaikan kapasitas baterai sebesar:

$$\Delta\% = 86\% - 43\% = 43\%$$

Baterai yang digunakan memiliki kapasitas nominal 12 V – 3 Ah, sehingga total energi yang dapat disimpan adalah:

$$E_{baterai} = 12 \times 3 = 36 \text{ Wh}$$

Energi yang benar-benar tersimpan selama periode pengisian adalah:

$$\Delta E = 43\% \times 36 \text{ Wh} = 15{,}48 \text{ Wh}$$

Waktu pengisian dari awal hingga sistem *OFF* adalah 1 jam 26 menit atau 1,433 jam. Dengan demikian, daya rata-rata yang benar-benar masuk ke baterai dapat dihitung sebagai:

$$Pavg = \frac{\Delta E}{\Delta t} - \frac{15{,}48}{1{,}433} = 10{,}8W$$

Padahal, panel surya yang digunakan memiliki kapasitas nominal 50 Wp. Jika diasumsikan panel menghasilkan daya penuh selama periode tersebut, energi yang seharusnya tersedia adalah:

$$Epanel = 50 \times 1{,}433 = 71{,}65 \text{ Wh}$$

Maka efisiensi efektif sistem selama proses pengisian dapat dihitung sebagai

$$\boldsymbol{\eta} efektif = \frac{\Delta E}{Epanel} \times 100\% = \frac{15{,}48}{71{,}65} \times 100\% = 21{,}6\%$$

Secara teoritis, waktu yang dibutuhkan untuk mengisi energi sebesar 15,48 Wh menggunakan panel 50 W dengan asumsi efisiensi 70% hanya:

$$\boldsymbol{\tau teori} = \frac{15{,}48}{50 - 0{,}7} = 0{,}442 jam \approx 26{,}5 \, menit$$

3. Pengujian Hari Jumat, 11 Juli 2025

Pengujian terakhir dilakukan pada hari Jumat, 11 Juli 2025, dimulai pada pukul 13.53 WITA dengan tegangan awal baterai sebesar 12,41 V dan kapasitas sebesar 36%. Sejak awal pengujian hingga pukul 15.18 WITA, relay berada dalam kondisi *ON*, yang menandakan sist em masih berada dalam mode pengisian karena tegangan belum mencapai batas atas yang telah ditentukan. Tegangan dan kapasitas baterai meningkat secara bertahap seiring waktu, mencerminkan bahwa proses pengisian daya dari panel surya berjalan secara normal. Pada pukul 14.47 WITA, tegangan tercatat sebesar 13,46 V dengan kapasitas baterai 72%, dan terus meningkat hingga mencapai 13,87 V dengan kapasitas 86% pada pukul 15.18 WITA. Pada titik ini, sistem secara otomatis memutus pengisian dengan

mengubah status relay menjadi *OFF*, karena tegangan telah mendekati ambang batas atas. Meskipun waktu pengujian hari Jumat berlangsung lebih singkat dibandingkan hari sebelumnya, sistem tetap menunjukkan performa yang baik dengan mengendalikan pengisian secara otomatis dan mencatat seluruh data sensor ke *Firebase* secara *real-time*. Hal ini menegaskan bahwa sistem tetap berjalan efektif dalam kondisi pencahayaan terbatas sekalipun.

Tabel 4. 4 Pegujian Jum'at 11 Juli 2025

No	Waktu	Tegangan baterai(V)	Kondisi Relay	Kapasitas baterai(%)
1	13.53	12.41	ON	36
2	14.03	12.67	ON	46
3	14.13	12.95	ON	55
4	14.23	13.20	ON	64
5	14.35	13.38	ON	70
6	14.47	13.46	ON	72
7	15.02	13.57	ON	76
8	15.12	13.70	ON	81
9	15.18	13.87	OFF	86

Berdasarkan hasil pengujian, pengisian baterai dilakukan dari kondisi awal pada pukul 13: 53 dengan tegangan baterai 12,41 V atau setara dengan 36% kapasitas. Proses pengisian berlangsung hingga sistem mencapai kondisi *OFF* pada pukul 15: 18 dengan tegangan baterai 13,87 V atau sekitar 86% kapasitas.

Dengan demikian, terjadi kenaikan kapasitas baterai sebesar:

$$\Delta\% = 86\% - 36\% = 50\%$$

Baterai yang digunakan memiliki kapasitas nominal 12 V – 3 Ah, sehingga total energi yang dapat disimpan adalah:

$$E_{baterai} = 12 \times 3 = 36 \text{ Wh}$$

Energi yang benar-benar tersimpan selama periode pengisian adalah:

$$\Delta E = 43\% \times 36 \text{ Wh} = 18{,}00 \text{ Wh}$$

Waktu pengisian dari awal hingga sistem *OFF* adalah 1 jam 26 menit atau 1,417 jam. Dengan demikian, daya rata-rata yang benar-benar masuk ke baterai dapat dihitung sebagai:

$$Pavg = \frac{\Delta E}{\Delta t} - \frac{18{,}00}{1{,}417} = 12{,}70W$$

Padahal, panel surya yang digunakan memiliki kapasitas nominal 50 Wp. Jika diasumsikan panel menghasilkan daya penuh selama periode tersebut, energi yang seharusnya tersedia adalah:

$$\text{Epanel}=50\times1{,}417=70{,}85\ \text{Wh}$$

Maka efisiensi efektif sistem selama proses pengisian dapat dihitung sebagai

$$\boldsymbol{\eta} efektif = \frac{\Delta E}{Epanel} \times 100\% = \frac{18{,}00}{70{,}85} \times 100\% = 25{,}4\%$$

Secara teoritis, waktu yang dibutuhkan untuk mengisi energi sebesar 18,00 Wh menggunakan panel 50 W dengan asumsi efisiensi 70% hanya:

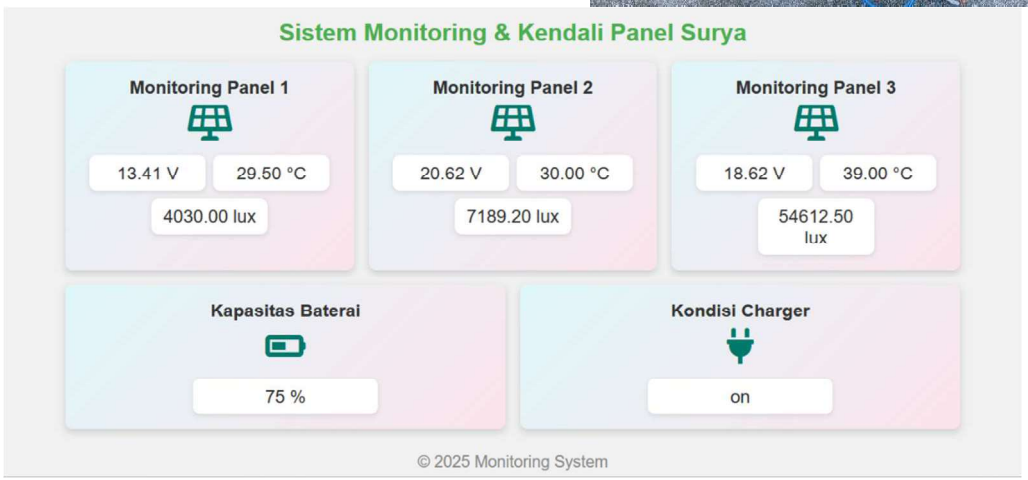
$$\boldsymbol{\tau teori} = \frac{18{,}00}{50 - 0{,}7} = 0{,}514 jam \approx 30{,}8\ menit$$

Hasil dari ketiga hari pengujian menunjukkan bahwa sistem kendali pengisian baterai pada PLTS skala laboratorium berbasis IoT ini berfungsi dengan baik, akurat, dan adaptif terhadap kondisi lapangan. Sistem tidak hanya bekerja secara otomatis berdasarkan nilai tegangan baterai, tetapi juga dapat memberikan data *real-time* yang dapat dipantau dari jarak jauh melalui jaringan internet.

Dengan implementasi seperti ini, sistem mampu meningkatkan efisiensi pengisian daya, memperpanjang umur baterai, dan memberikan kontrol penuh kepada pengguna, tanpa perlu intervensi manual.







Gambar 4. 23 Pegujian Kondisi *Charging*

b. Pegujian kondisi *discharging*

Selain dilakukan pengujian pada kondisi pengisian (*charging*), sistem juga diuji untuk memantau perilaku discharging baterai. Proses discharging terjadi saat beban mengambil daya dari baterai, sementara suplai pengisian dari panel surya dihentikan atau relay berada dalam kondisi *OFF*. Pengujian ini bertujuan untuk melihat bagaimana sistem memantau penurunan tegangan baterai dan bagaimana sistem melakukan respons otomatis saat kondisi baterai mencapai tegangan batas bawah (*cut-off*).

Pada sistem yang dirancang, batas bawah tegangan baterai ditetapkan sebesar 12.20V. Ketika tegangan baterai turun mencapai atau di bawah nilai ini, maka sistem dirancang untuk mengaktifkan relay kembali (*ON*) agar proses pengisian dimulai. Sistem secara otomatis memutus atau menyambungkan relay sesuai logika yang tercantum pada *flowchar*t, memastikan baterai tetap berada dalam rentang tegangan aman.

Selain faktor waktu dan intensitas matahari, beban yang terpasang juga sangat memengaruhi kecepatan penurunan tegangan baterai. Pada pengujian ini, beban yang digunakan berbeda di masing-masing hari pengujian, sehingga memberikan karakteristik penurunan tegangan yang juga bervariasi.Berikut analisis pengujian selama tiga hari berturut-turut

1. Pengujian Rabu 9 Juli 2025

Pada hari Rabu, pengujian proses discharging dilakukan mulai pukul 12.02WITA dengan kondisi relay *OFF*, menunjukkan bahwa sistem sedang tidak melakukan pengisian daya ke baterai. Pada pengujian ini, baterai diberikan beban lampu AC 20W . Hal ini memberikan penarikan arus yang cukup besar, sehingga mempercepat proses penurunan tegangan baterai.

Tegangan awal baterai tercatat sebesar 13.31V dengan kapasitas 64%. Seiring waktu, nilai tegangan baterai mengalami penurunan signifikan, dari 13.31V menjadi 12.20V, dan kapasitas turun dari 64% menjadi 29%.

Pada pukul 13.08 WITA, tercatat tegangan 12.20V, yang merupakan batas bawah tegangan baterai. Sistem merespons otomatis dengan mengaktifkan relay (*ON*), yang artinya sistem kembali memulai proses pengisian baterai. Hal ini

menunjukkan bahwa logika *cut-off* bekerja sesuai desain, melindungi baterai dari kondisi *over-discharging* yang dapat merusak sel baterai.

Tabel 4. 5 Pengujian Rabu 9 Juli 2025

No	Waktu	Tegangan baterai(V)	Kondisi Relay	Kapasitas baterai(%)
1	12.02	13.31	OFF	64
2	12.06	12.91	OFF	54
3	12.12	12.50	OFF	40
4	12.18	12.45	OFF	38
5	12.35	12.37	OFF	35
6	12.52	12.30	OFF	33
7	13.08	12.20	ON	29

Berdasarkan pengujian rabu 9 juli 2025 dengan kondisi *discharging* sistem, baterai 12 V 3 Ah digunakan sebagai sumber energi untuk menyalakan beban lampu AC 25 W melalui *inverter* berkapasitas 220 W Perhitungan kebutuhan daya DC dapat dihitung dengan:

- Waktu awal pengamatan: 12:02 — Kapasitas = 64 %
- Waktu akhir pengamatan: 13:08 — Kapasitas = 29 %
- Selisih waktu: 13:08 − 12:02 = 66 menit = 1,10 jam

Spesifikasi baterai: $V_{Bat} = 12V\ C_{Bat} = 3Ah\ \ Energi\ nominal$

$$E_{bat,nom} = 12Vx3Ah = 36Wh$$

Selisih kapasitas:

$$\Delta\% = 64\% - 29\% = 35\%$$

Energi yang “keluar” (tersedot) dari baterai selama periode:

$$\Delta E = \frac{\Delta\%}{100} x E_{Bat,nom} = 0{,}35\ x\ 36 = 12{,}6Wh$$

Daya rata-rata yang keluar dari baterai selama periode (berdasarkan ΔE dan Δt):

$$P_{AVG,means} = \frac{\Delta E}{\Delta t} = \frac{12{,}6}{1{,}10 jam} = 11{,}45W$$

kebutuhan DC dari baterai ketika mensuplay beban 25 W melalui inverter:

$$\mathcal{P}DC\frac{\mathcal{P}AC}{\eta inv} + \mathcal{P}idle$$

Dengan

P_{DC}= daya yang diambil dari baterai (W)

P_{AC}= daya beban AC (25 W),

η_{inv}= efisiensi inverter(2w).

P_{idle}= daya komsumsi 85%

Hasil perhitungan:

$$\mathcal{P}DC\frac{25}{0{,}85} + 2 = 31{,}41W$$

Sehingga arus yang disuplai dari baterai:

$$I_{Bat}\frac{P_{DC}}{V_{Bat}} = \frac{31{,}41}{12} = 2{,}62A$$

2. Pengujian kamis 10 juli 2025

Pada hari Kamis, pengujian dilakukan dengan kondisi awal baterai pada tegangan 12.99V dengan kapasitas 56%, dalam kondisi relay *OFF*. Sama seperti hari Rabu,degan beban yang dihubungkan ke baterai,adalah lampu AC 25W. ini juga menyebabkan penurunan tegangan relatif cepat karena konsumsi arus yang lebih tinggi.

Selama proses pengujian, tegangan terus mengalami penurunan secara bertahap hingga mencapai 12.20V pada pukul 12.27, dengan kapasitas turun menjadi 29%.

Pada titik ini, sistem secara otomatis mengaktifkan relay (*ON*). Hal ini sesuai dengan algoritma kontrol yang dirancang dalam sistem, di mana relay akan aktif kembali ketika tegangan baterai mencapai batas bawah. Pengujian hari kedua ini menunjukkan bahwa sistem memiliki kemampuan respon cepat dan tepat dalam mendeteksi kondisi drop tegangan, sekaligus menjaga baterai tetap berada dalam rentang operasi aman.

Tabel 4. 6 Pengujian Kamis 10 Juli 2025

No	Waktu	Tegangan baterai(V)	Kondisi Relay	Kapasitas baterai(%)
1	11.46	12.99	OFF	56
2	11.54	12.76	OFF	48
3	12.04	12.47	OFF	39
4	12.13	12.37	OFF	35
5	12.23	12.24	OFF	31
6	12.27	12.20	ON	29

Berdasarkan pengujian kamis 10 juli 2025 dengan kondisi *discharging* sistem, baterai 12 V 3 Ah digunakan sebagai sumber energi untuk menyalakan beban lampu AC 25 W melalui inverter berkapasitas 220 W Perhitungan kebutuhan daya DC dapat dihitung dengan:

- Waktu awal pengamatan: 11,58 — Kapasitas = 56%
- Waktu akhir pengamatan: 12,27 — Kapasitas = 29 %
- Selisih waktu: 12,27− 11:58 = 41 menit = 0,68 jam

Spesifikasi baterai: $V_{Bat} = 12V\ C_{Bat} = 3Ah\ \ Energi\ nominal$

$$E_{bat,nom} = 12x3 = 36Wh$$

Selisih kapasitas:

$$\Delta\% = 56\% - 29\% = 27\%$$

Energi yang “keluar” dari baterai selama periode:

$$\Delta E = \frac{\Delta\%}{100} x E_{Bat,nom} = 0{,}27\ x\ 36 = 9{,}72Wh$$

Daya rata-rata yang keluar dari baterai selama periode (berdasarkan ΔE dan Δt):

$$P_{AVG,means} = \frac{\Delta E}{\Delta t} = \frac{9{,}72}{0{,}68} = 14{,}29W$$

kebutuhan DC dari baterai ketika mensuplay beban 25 W melalui inverter:

$$\mathcal{P}DC\frac{\mathcal{P}AC}{\eta inv} + \mathcal{P}idle$$

Dengan

P_{DC}= daya yang diambil dari baterai (W)

P_{AC}= daya beban AC (25 W),

η_{inv}= efisiensi inverter(2w).

P_{idle}= daya komsumsi 85%

Hasil perhitungan:

$$\mathcal{P}DC\frac{25}{0{,}85} + 2 = 31{,}41W$$

Sehingga arus yang disuplai dari baterai:

$$I_{Bat}\frac{P_{DC}}{V_{Bat}} = \frac{31{,}41}{12} = 2{,}62A$$

3. Pengujian jum'at 11 juli 2025

Pada hari Jum'at, pengujian dilakukan dengan kondisi awal baterai pada tegangan 12.98V dengan kapasitas 56%, Seluruh data menunjukkan relay dalam kondisi *OFF* sejak awal, menunjukkan proses discharging berlangsung penuh.

Tegangan baterai menurun perlahan hingga mencapai 12.20V pada pukul 16.28, dengan kapasitas tersisa 29%.

Proses penurunan tegangan pada hari Jumat terlihat lebih lambat dibandingkan dua hari sebelumnya, karena beban lebih kecil. Sistem kemudian mengaktifkan relay (*ON*) secara otomatis ketika tegangan menyentuh batas bawah. Hal ini membuktikan konsistensi sistem dalam memantau kondisi baterai dan mengambil tindakan yang diperlukan secara otomatis.

Tabel 4. 7 Pengujian Jum'at 11 Juli 2025

No	Waktu	Tegangan baterai(V)	Kondisi relay	Kapasitas baterai(%)
1	15.24	12.98	OFF	56
2	15.34	12.58	OFF	43
3	15.44	12.52	OFF	41
4	15.54	12.50	OFF	39
5	16.04	12.36	OFF	35
6	16.14	12.31	OFF	33
7	16.24	12.23	OFF	31
8	16.28	12.20	ON	29

Berdasarkan pengujian jum'at 11 juli 2025 dengan kondisi *discharging* sistem, baterai 12 V 3 Ah digunakan sebagai sumber energi untuk menyalakan beban lampu AC 25 W melalui inverter berkapasitas 220 W Perhitungan kebutuhan daya DC dapat dihitung dengan:

- Waktu awal pengamatan: 15,24 — Kapasitas = 56%
- Waktu akhir pengamatan: 16,28 — Kapasitas = 29 %
- Selisih waktu: 15,24− 16:28 = 41 menit = 1,07 jam

Spesifikasi baterai: $V_{Bat} = 12V\ C_{Bat} = 3Ah\ \ Energi\ nominal$

$$E_{bat,nom} = 12x3 = 36Wh$$

Selisih kapasitas:

$$\Delta\% = 56\% - 29\% = 27\%$$

Energi yang “keluar” dari baterai selama periode:

$$\Delta E = \frac{\Delta\%}{100} x E_{Bat,nom} = 0{,}27\ x\ 36 = 9{,}72Wh$$

Daya rata-rata yang keluar dari baterai selama periode (berdasarkan ΔE dan Δt):

$$P_{AVG,means} = \frac{\Delta E}{\Delta t} = \frac{9{,}72}{1{,}07} = 9{,}08W$$

kebutuhan DC dari baterai ketika mensuplay beban 25 W melalui inverter:

$$\mathcal{P}DC\frac{\mathcal{P}AC}{\eta inv} + \mathcal{P}idle$$

Dengan

P_{DC}= daya yang diambil dari baterai (W)

P_{AC}= daya beban AC (25 W),

η_{inv}= efisiensi inverter(2w).

P_{idle}= daya komsumsi 85%

Hasil perhitungan:

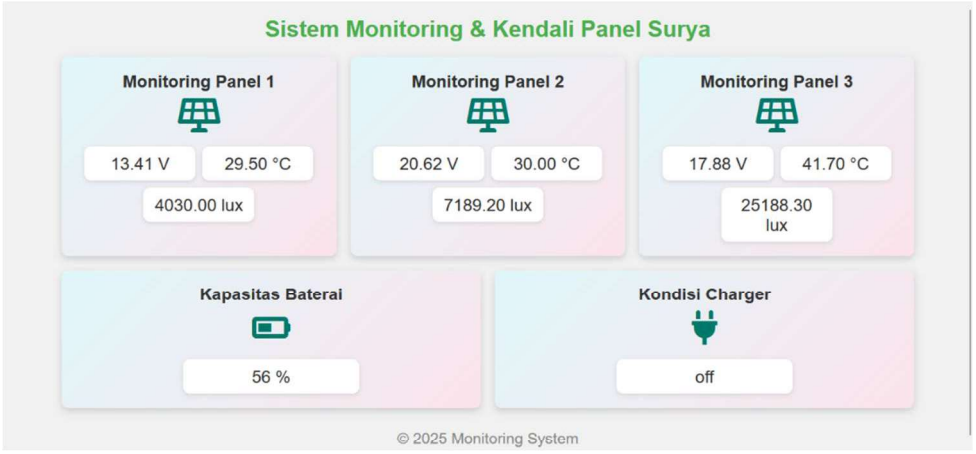
$$\mathcal{P}DC\frac{25}{0{,}85} + 2 = 31{,}41W$$

Sehingga arus yang disuplai dari baterai:

$$I_{Bat}\frac{P_{DC}}{V_{Bat}} = \frac{31{,}41}{12} = 2{,}62A$$







Gambar 4. 24 Pegujian kondisi *discharging*

Dari hasil pengujian tiga hari berturut-turut, dapat disimpulkan bahwa sistem kendali pembangkit listrik tenaga surya skala laboratorium berbasis IoT berfungsi dengan baik pada kondisi *discharging*. Sistem mampu menjaga tegangan baterai tidak turun di bawah batas aman, serta secara otomatis memulai kembali pengisian saat diperlukan. Perbedaan laju penurunan tegangan pada masing-masing hari menunjukkan bahwa faktor beban sangat berpengaruh terhadap performa baterai, yang menjadi pertimbangan penting dalam perancangan sistem PLTS.

Selain itu, integrasi dengan *Firebase* memberikan keunggulan tambahan berupa kemudahan monitoring jarak jauh, serta memudahkan analisis historis kondisi sistem. Dengan demikian, sistem ini sangat layak digunakan sebagai prototipe atau model pengendalian pembangkit listrik tenaga surya skala laboratorium yang modern, efisien, dan berbasis teknologi *Internet of Things.*

BAB V

PENUTUP

A. Kesimpulan

Berdasarkan perancangan, implementasi, dan pengujian yang telah dilakukan terhadap sistem kendali Pembangkit Listrik Tenaga Surya (PLTS) skala laboratorium berbasis *Internet of Things* (IoT). Bahwa Arsitektur sistem kendali pengisian baterai berbasis IoT pada PLTS skala laboratorium berhasil dirancang dan diimplementasikan dengan baik menggunakan mikrokontroler ESP32 sebagai unit utama pengolah data dan pengendali logika sistem**.**Sistem ini mampu mengendalikan proses pengisian baterai secara otomatis berdasarkan nilai tegangan yang terukur oleh sensor *voltage* . Ketika tegangan mencapai ambang batas atas 13,8V maka sistem memutus pengisian dengan mengnonaktifkan relay dan saat tegangan turun ke ambang batas bawah 12,2V maka sistem secara otomatis menghubungkan kembali proses pengisian Daya pemantauan seperti tegangan baterai, kapasitas baterai dan status pengisian ditampilkan melalui LCD dan dikirim secara *real-time* ke *Firebase Realtime Database* sehingga memungkinkan pemantauan jarak jauh melalui aplikasi berbasis web. Sistem juga mampu menyimpan log data historis secara otomatis, termasuk waktu, nilai tegangan, kapasitas baterai dan status pengisian, yang dapat dimanfaatkan untuk analisis performa lebih lanjut.

B. Saran

Sebagai bentuk evaluasi dan pengembangan lebih lanjut terhadap sistem kendali PLTS skala laboratorium ini, beberapa saran berikut dapat dijadikan pertimbangan:

1. Sistem pemantauan berbasis *Firebas*e dapat dikembangkan lebih lanjut dengan fitur tambahan seperti grafik tren data historis, notifikasi peringatan jika kapasitas baterai rendah atau tegangan melebihi batas.
2. Uji coba sebaiknya dilakukan dalam berbagai kondisi pencahayaan, untuk mengetahui sejauh mana kestabilan sistem dalam situasi nyata dan mengevaluasi durabilitas dari perangkat keras.
3. Untuk keperluan skala laboratorium yang ingin meniru kondisi di lapangan, disarankan untuk menggunakan jenis baterai yang sesuai seperti tipe deep cycle bukan aki kendaraan, agar hasil pengujian lebih representatif terhadap sistem PLTS sesungguhnya.

DAFTAR PUSTAKA

Anggoro, G. P. T. N. (2023). *Kendali Pembangkit Listrik Tenaga Surya untuk Ladang Surya Berbasis Internet of Things* (Doctoral dissertation, Universitas Gadjah Mada).

Ardiansyah, A. (2020). *Monitoring Daya Listrik Berbasis IoT (Internet of Things)* (Doctoral dissertation, Universitas Islam Indonesia).

Darmana, T., Pambudi, I. T., Suyanto, H., & Rahayu, S. (2024). *Sistem Hybrid Pembangkit Listrik Tenaga Surya (PLTS), Pembangkit Listrik Tenaga Bayu (PLTB), Pembangkit Listrik Tenaga Mikrohidro (PLTMH), Pembangkit Listrik Tenaga Panas Bumi (PLTP)*. Uwais Inspirasi Indonesia.

Daya, B. T. S. B. (2025). Pelatihan Pembangkit Listrik Tenaga Surya Jenis Off-Grid di BLK Don.

David, D. (2024). POTENSI PEMBANGKIT LISTRIK TENAGA SURYA (PLTS) ON-GRID UNTUK KEBUTUHAN LISTRIK LABORATORIUM TEKNIK UNIVERSITAS JAMBI (Doctoral dissertation, Teknik Elektro).

Idris, M. (2019). Rancang panel surya untuk instalasi penerangan rumah sederhana daya 900 watt. *Jurnal Elektronika Listrik Dan Teknologi Informasi Terapan*, *1*(1), 17-22.

IQRAM, M. (2025). *SISTEM OTOMATIS PENGISIAN BATERAIDENGAN SUMBER TENAGASURYA DAN ANGIN* (Doctoral dissertation, Universitas Muhammadiyah Parepare).

Layela, G. I. (2023). *Perancangan Pembangkit Listrik Tenaga Surya (PLTS) Sistem Off-Grid untuk Memenuhi Kebutuhan Listrik pada Kopi Kapal Api UKI Berbasis HOMER* (Doctoral dissertation, Universitas Kristen Indonesia).

Mahendra, G. S., Judijanto, L., Tahir, U., Nugraha, R., Dwipayana, A. D., Nuryanneti, I., ... & Rakhmadani, D. P. (2024). *Green Technology: Panduan Teknologi Ramah Lingkungan*. PT. Sonpedia Publishing Indonesia.

Mahendra, G. S., Judijanto, L., Tahir, U., Nugraha, R., Dwipayana, A. D., Nuryanneti, I., ... & Rakhmadani, D. P. (2024). *Green Technology: Panduan Teknologi Ramah Lingkungan*. PT. Sonpedia Publishing Indonesia.

Melly, W. A. (2024). Prototype Sistem Monitoring Penyerapan Daya dan Perawatan Panel Surya pada PLTS Berbasis Website (Doctoral dissertation, Universitas Andalas).

Muner, M. (2021). *pemanfaatan energi matahari melalui panel surya dengan beban mesin pengering larva BSF* (Doctoral dissertation, Universitas Yudharta).

Muslim, S., Khotimah, K., & Azhiimah, A. N. (2020). analisis kritis terhadap perencanaan pembangkit listrik tenaga surya (PLTS) tipe photovoltaic (PV) sebagai energi alternatif masa depan. *Rang Teknik Journal*, *3*(1), 119-130.

Pratama, M. F. (2021). *Sistem Monitoring dan Kontrol Daya PLTS Menggunakan IoT Berbasis Fuzzy Logic* (Master's thesis, Universitas Islam Sultan Agung (Indonesia)).

Rauf, R. (2023). Matahari sebagai Energi Masa Depan Panduan Lengkap Pembangkit Listrik Tenaga Surya (PLTS).

Rauf, R. (2023). Matahari sebagai Energi Masa Depan Panduan Lengkap Pembangkit Listrik Tenaga Surya (PLTS).

Rauf, R. (2023). Optimalisasi Pembangkit Listrik Tenaga Surya di Daerah Kepulauan.

Rudiyanto, B., Rachmanita, R. E., & Budiprasojo, A. (2023). Dasar-Dasar Pemasangan Panel Surya. *Dasar Dasar Pemasangan Panel Surya*.

Sari, L. O., Saputra, M. F. E., & Safrianti, E. (2024). Sistem Monitoring Arus Listrik Berbasis Internet of Things (IoT) pada Solar Panel di Laboratorium Pembangkit Listrik Tenaga Surya (PLTS) UIN Suska Riau: Electric Current Monitoring System Based on IoT (Internet of Things) On Solar Panel In Solar Electric Power Plant (PLTS) Laboratory of UIN Suska Riau. *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, *4*(1), 205-211.

Setiawan, D. K., Widjonarko, W., & Firdaus, A. (2022). Sistem Monitoring Panel Surya Berbasis Android Secara Real-Time. *Jurnal FORTECH*, *3*(1), 7-16.

SIMARMATA, F. A. (2024). Rancang Bangun Prototipe Sistem Kendali Ketukan Lonceng Berbasis Internet of Things (IoT).

Widharma, I. G. S., & Wiranata, L. F. (2022). *Mikrokontroler dan aplikasi*. wawasan Ilmu.