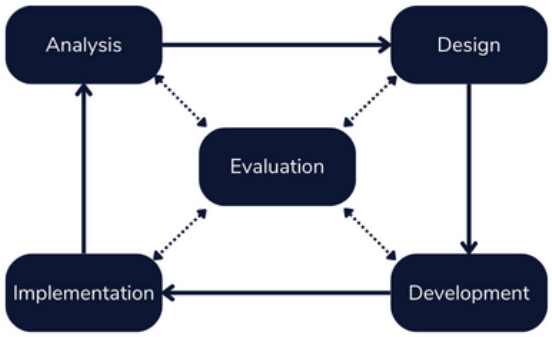
# BAB III
# METODE PENELITIAN

Bab ini menjelaskan pendekatan dan langkah-langkah yang digunakan dalam pelaksanaan penelitian.

## 3.1 Jenis Penelitian

Penelitian ini menggunakan metode *Research and Development* (R&D) dengan pendekatan penelitian ADDIE *(Analysis, Design, Development, Implementation, Evaluation)*. Penelitian R&D digunakan untuk merancang dan membangun *Smart Energy Meter* untuk pemantauan penggunaan kWh listrik dan memperamalan penggunaan kWh listrik berdasarkan pola penggunaan sebelumnya. Metode penelitian R&D bertujuan untuk menghasilkan suatu produk melalui tahapan identifikasi potensi permasalahan, perancangan, serta pengembangan produk yang dirancang sebagai Solusi paling efektif terhadap permasalahan tersebut [55]. Adapun pendekatan penelitian yang digunakan pada penelitian ini adalah metode ADDIE. Metode ADDIE menawarkan keunggulannya dalam menghasilkan produk inovatif serta efektif yang sistematis dan terstruktur [55]. Tahapan penelitian menggunakan pendekatan ADDIE dalam pengembangan *Smart Energy Meter* berbasis IoT disajikan pada Gambar 3.1.

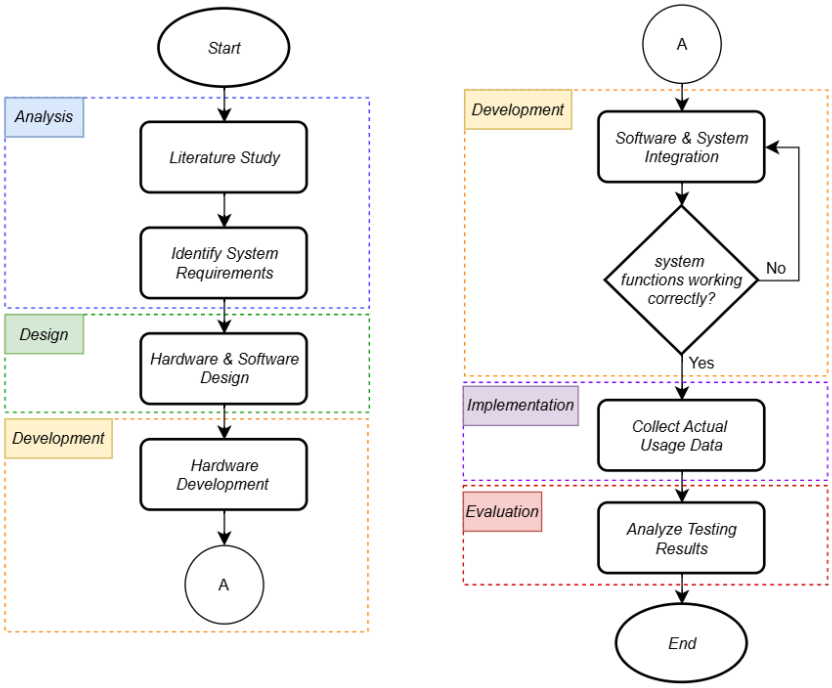



**Gambar 3. 1** Desain Penelitian

## 3.2 Alur Penelitian

Alur penelitian berfungsi sebagai dasar dan panduan sistematis bagi peneliti dalam melaksanakan seluruh proses penelitian. Tujuan dari penyusunan alur ini adalah untuk memastkan bahwa setiap tahapan penelitian bejalan secara terarah dan

konsisten dengan tujuan yang telah ditetapkan. Mengacu pada model pengembangan ADDIE yang digunakan dalam pendekatan prosedural, tahapan-tahapan yang dilalui dalam penelitian ini dirancang secara berurutan dan saling berkesinambungan. Alur proses tersebut dapat dilihat melalui diagram alir pada Gambar 3.2.




**Gambar 3. 2** Alur Penelitian

1. *Analysis*

   Tahap *analysis* dilakukan untuk mengidentifikasi kebutuhan dalam pengembangan produk atau model, serta menilai kelayakan dari produk yang akan dikembangkan. Pada penelitian ini, proses analisis diawali dengan melakukan studi literatur dan pengumpulan data yang relevan untuk mengidentifikasi kebutuhan penelitian *Smart Energy Meter*.

2. *Design*

   Pada tahap ini mencakup proses perancangan *hardware* dan *software*. Perancangan *hardware* meliputi pemilihan komponen yang akan digunakan, seperti modul ESP32, sensor PZEM-004T, LCD 16x2, dan Oled *display*. Selain itu, pada tahap ini juga dilakukan penyusunan skema perakitan antar komponen serta perancangan tata letak pada papan sirkuit (PCB). Sementara itu,

perancangan *software* mencakup pengembangan sejumlah program yang berfungsi untuk membaca data dari sensor, mengirimkan data tersebut ke layanan *cloud*, serta menampilkan informasi hasil pemantauan dan peramalan konsumsi energi melalui antarmuka yang tersedia.

3. *Development*

   Tahap *development* merupakan proses pengembangan dari rancangan sistem yang telah dibuat sebelumnya. Pada tahap ini dilakukan perakitan *hardware* menggunakan ESP32, sensor PZEM-004T, LCD 16x2, dan Oled *display*. Mikrokontroler diprogram untuk membaca data sensor, menampilkan data ke *display* (LCD dan Oled), serta mengirimkan data secara *realtime* ke Firebase. Selanjutnya, dilakukan integrasi model peramalan berbasis XGBoost yang telah dilatih menggunakan data historis konsumsi energi dari Firebase. Model ini dijalankan melalui API Flask, dan hasil peramalannya digunakan untuk menjadi batas acuan penggunaan listrik perharinya.

4. *Implementation*

   Sistem yang telah dikembangkan dan diuji melalui serangkaian pengujian fungsional untuk memastikan semua fitur berjalan sesuai rencana. Selanjutnya dilakukan proses pengumpulan data sebagai bagian dari validasi sistem dan analisis model peramalan.

5. *Evaluation*

   Hasil pengujian dan data yang terkumpul dianalisis untuk menilai efektivitas sistem, termasuk keakuratan pembacaan sensor, kestabilan komunikasi data, serta performa peramalan konsumsi energi. Hasil evaluasi ini menjadi dasar perbaikan sistem secara berkelanjutan.

### 3.3 Deskripsi Umum Penelitian

Penelitian ini berfokus pada pengembangan *Smart Energy Meter* berbasis *Internet of Things* (IoT), algoritma *machine learning*, dan *web dashboard* untuk memantau sekaligus meramalkan konsumsi energi listrik secara cerdas dan *realtime*. Sistem dirancang menggunakan mikrokontroler ESP32 sebagai unit kendali utama, yang terintegrasi dengan sensor PZEM-004T untuk mengukur parameter kelistrikan seperti arus, frekuensi, tegangan, daya, dan faktor daya. Data

yang diperoleh diproses oleh ESP32 untuk menghitung konsumsi energi dalam satuan *kilowatt-hour* (kWh). Untuk mempermudah pemantauan secara local, system dilengkapi dengan dua jenis tampilan: LCD Oled yang menyajikan grafik konsumsi energi dan LCD 16x2 untuk menampilkan data sensor secara *realtime.*

Sebagai bagian dari fitur prediktif, sistem ini menerapkan model *machine learning* XGBoost yang digunakan untuk meramal konsumsi energi listrik dalam periode mendatang berdasarkan pola historis. Jika konsumsi aktual melampaui ambang batas hasil peramalan, system akan secara otomatis mengirim notifikasi melalui Telegram sebagai bentuk peringatan dini kepada pengguna. Seluruh data pembacaan sensor disimpan di *platform* Firebase dan disajikan di web *dashboard* beserta nilai hasil peramalan, yang berfungsi sebagai media pemantauan dan analisis daring.

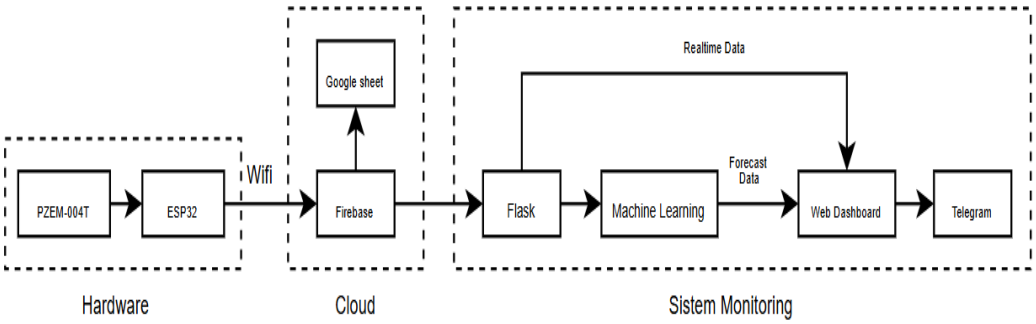
Dalam proses pengembangan sistem, *software* EasyEda digunakan untuk merancang skematik dan *printed circuit board* (PCB), Canva dimanfaatkan untuk mendesain jalur pengkabelan antar komponen, dan Draw.io digunakan dalam pembuatan diagram alur sistem, blok diagram, serta elemen visual pendukung lainnya.

## 3.4 Perancangan Sistem

Perancangan sistem merupakan tahapan krusial dalam pengembangan *Smart Energy Meter* ini, yang bertujuan untuk memastikan seluruh komponen dapat terintegrasi dan berfungsi sesuai dengan tujuan penelitian. Proses perancangan ini mencakup aspek arsitektur sistem secara keseluruhan, perancangan *hardware* (alat), perancangan *software*, perancangan model XGBoost.

### 3.4.1 Arsitektur Sistem Keseluruhan

Arsitektur sistem secara keseluruhan dirancang untuk memberikan gambaran komprehensif mengenai komponen-komponen utama dan interaksi antar-komponen dalam sistem *Smart Energy Meter*. Arsitektur sistem disajikan pada Gambar 3.3.

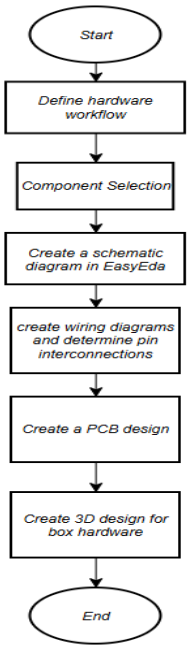



**Gambar 3. 3** Arsitektur Sistem

Gambar 3.3 mengilustrasikan komponen utama sistem yang terdiri dari unit pengukuran (PZEM-004T), unit kendali (ESP32), platform *cloud* (Firebase), model *machine learning* (XGBoost), *web dashboard*, dan sistem notifikasi (Telegram). Masing-masing komponen memiliki peran spesifik dan saling berinteraksi dalam alur data yang terstruktur.

Pada bagian *hardware*, sensor PZEM-004T digunakan untuk membaca parameter kelistrikan seperti tegangan, arus, daya, dan energi listrik. Data tersebut dikirim melalui ESP32 menggunakan koneksi WiFi ke layanan Firebase. Firebase bertindak sebagai media penyimpanan data secara *realtime*, yang juga terhubung ke Google Sheet untuk pencatatan dan keperluan dokumentasi. Selanjutnya, data dari Firebase diteruskan ke server Flask, yang kemudian mengirimkan data tersebut ke model *machine learning* untuk dilakukan proses peramalan konsumsi energi. Hasil peramalan serta data *realtime* ditampilkan pada *web dashboard,* yang memungkinkan pengguna untuk memantau kondisi energi secara interaktif. Selain itu, sistem juga terintegrasi dengan Telegram sebagai media notifikasi otomatis untuk mengirim peringatan atau inforamsi penting kepada pengguna.

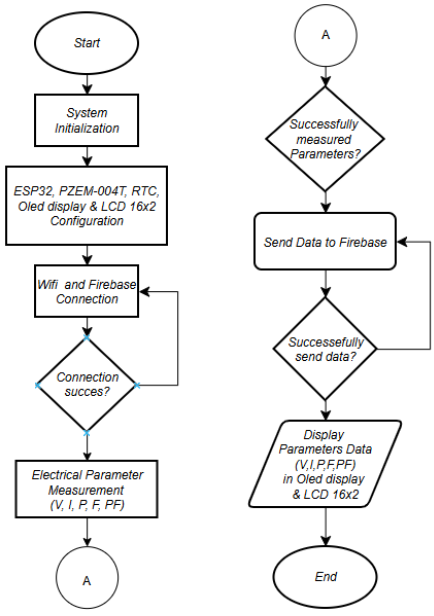
### 3.4.2 Perancangan *Hardware*




**Gambar 3. 4** *Flowchart* Perancangan *Hardware*

Gambar 3.4 menampilkan *flowchart* perancangan *hardware* secara sistematis untuk penelitian ini. *Flowchart* ini membantu menggambarkan bahwa proses perancangan *hardware* dilakukan secara berurutan dan terstruktur, guna memastikan keterpaduan dan fungsionalitas sistem secara menyeluruh.

#### 3.4.2.1 *Flowchart Hardware System*

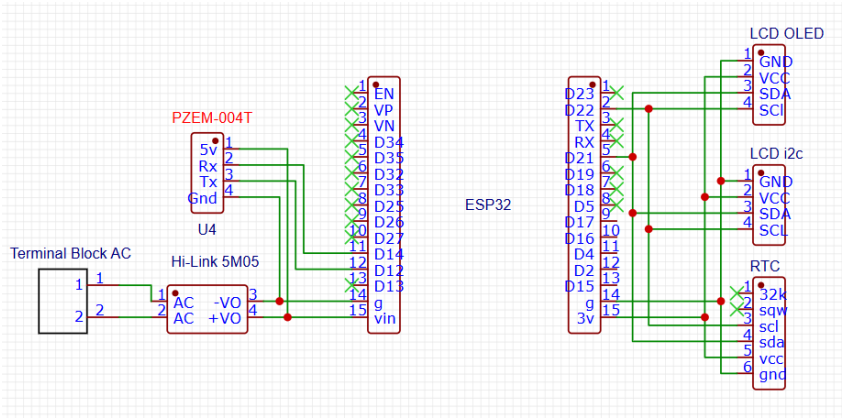



**Gambar 3. 5** *Flowchart Hardware System*

Gambar 3.5 menggambarkan proses kerja sistem *hardware* secara bertahap. Proses dimulai dengan inisialisasi sistem untuk memastikan seluruh perangkat siap beroperasi. Selanjutnya dilakukan konfigurasi pada ESP32, sensor PZEM-004T, modul RTC, serta tampilan OLED dan LCD 16x2 agar semuanya dapat berfungsi sesuai fungsinya. Setelah konfigurasi, sistem mencoba terhubung ke jaringan WiFi dan Firebase sebagai media penyimpanan data di cloud. Jika koneksi berhasil, ESP32 akan mulai melakukan pengukuran parameter kelistrikan seperti tegangan, arus, daya, frekuensi, dan faktor daya menggunakan sensor PZEM-004T. Data yang berhasil diukur kemudian dikirim ke Firebase. Jika proses pengiriman data berjalan sukses, maka hasil pengukuran ditampilkan secara *realtime* melalui layar Oled dan LCD 16x2 sebagai visualisasi lokal yang memudahkan pengguna dalam memantau penggunaan energi secara langsung.

### 3.4.2.2 Skema Rangkaian

Skema rangkaian adalah gambaran visual dari suatu sistem rangkaian listrik atau elektronik yang menunjukkan hubungan antar komponen menggunakan simbol-simbol standar. Pada penelitian ini menggunakan skema seperti pada Gambar 3.6.



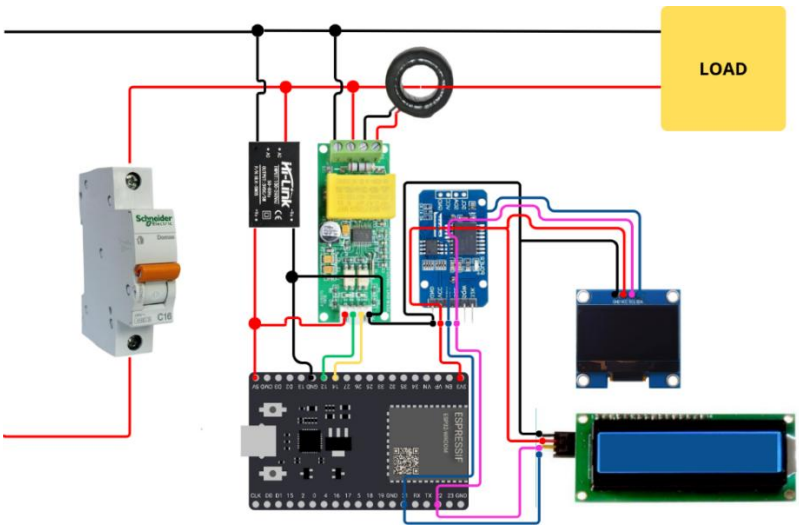

**Gambar 3. 6** *Schematic diagram*

Pada Gambar 3.6, ESP32 berfungsi sebagai mikrokontroler utama yang menangani komunikasi dengan berbagai sensor, termasuk sensor PZEM-004T yang digunakan untuk mengukur parameter listrik pada instalasi rumah seperti tegangan, arus, daya, dan energi. Data dari sensor ini menjadi input penting dalam sistem pemantauan untuk memperoleh informasi penggunaan energi secara *realtime* dan

akurat. Selain itu, ESP32 juga terhubung dengan modul RTC (*Real Time Clock*) yang berperan dalam mencatat informasi waktu pengambilan data, sehingga setiap data yang terekam dapat ditelusuri berdasarkan timestamp-nya dan mendukung analisis konsumsi energi yang lebih informatif dan historis. ESP32 juga dihubungkan dengan modul *display*, seperti LCD 16x2 dan LCD Oled. Kedua modul *display* ini berfungsi sebagai *interface* pengguna lokal, memungkinkan pengguna untuk mengetahui informasi data pemantauan energi secara instan tanpa perlu mengakses *web dashboard*.

#### 3.4.2.3 *Wiring Diagram* dan Interkoneksi Pin

Wiring diagram berfungsi untuk menunjukkan jalur fisik dari kabel atau koneksi antar komponen elektronik, posisi koneksi, dan urutan pin. Sedangkan, interkoneksi pin berguna untuk menjelaskan hubungan antar pin dari berbagai komponen untuk memudahkan dalam proses perakitan




**Gambar 3. 7** *Wiring diagram*

*Wiring diagram* pada Gambar 3.7 menjadi acuan utama dalam proses perakitan *hardware*. Perancangan rangkaian ini dilakukan dengan mengacu pada datasheet masing-masing komponen, guna memastikan bahwa koneksi antar perangkat telah sesuai dengan spesifikasi teknis dan tidak menimbulkan kesalahan fungsi. Setiap jalur komunikasi dan suplai daya ditentukan berdasarkan karakteristik dan kebutuhan masing-masing modul yang digunakan. Untuk mempermudah

pemahaman terhadap konfigurasi sistem, interkoneksi antar pin ESP32 dan komponen pendukung disajikan dalam Tabel 3.1 berikut.

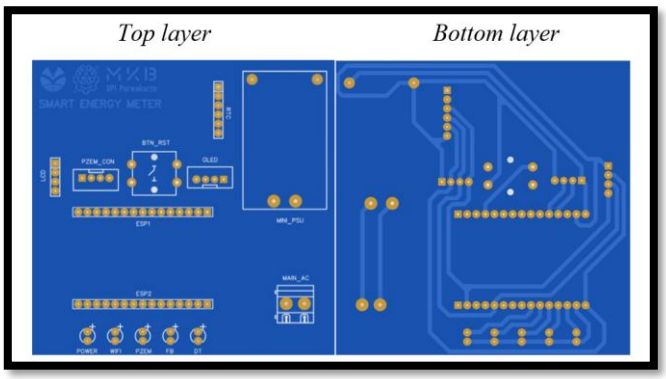
**Tabel 3. 1** Interkoneksi Pin

| Modul | Pin Modul | Pin ESP32 | Deskripsi |
|---|---|---|---|
| PZEM-004T | TX | D12 | Sebagai pin komunikasi untuk sensor mengirim data ke ESP32 |
| | RX | D14 | Sebagai pin komunikasi untuk sensor menerima data dari ESP32 |
| RTC, LCD Oled, LCD 16x2 | SCL | D22 | Sebagai jalur sinyal *clock* untuk sinkronisasi komunikasi |
| | SDA | D21 | Sebagai jalur transmisi data pada komunikasi $I^2C$ |
| Hi-Link HLK-5M05 | Vo+ | VIN | Sebagai *input* tegangan 5V DC |
| | Vo- | GND | Sebagai *input ground* tegangan |

Tabel 3.1 menjadi acuan untuk tahap awal perancangan sistem. Sistem dirancang menggunakan media *breadboard* untuk memudahkan proses pengujian dan identifikasi kesalahan rangkaian secara cepat. Penggunaan *breadboard* memungkinkan penyesuaian ulang jika terjadi ketidaksesuaian antar komponen atau *error* saat proses pembacaan dan pengiriman data.

### 3.4.2.4 Desain PCB

Setelah seluruh komponen berhasil berjalan sesuai perancangan, rangkaian dialihkan ke dalam bentuk *Printed Circuit Board* (PCB) untuk memastikan keandalan dan kestabilan koneksi, serta untuk memperbaiki tata letak fisik komponen agar lebih terorganisir. Proses ini juga bertujuan untuk mengurangi potensi gangguan yang timbul akibat koneksi kabel longgar atau noise antarkabel.

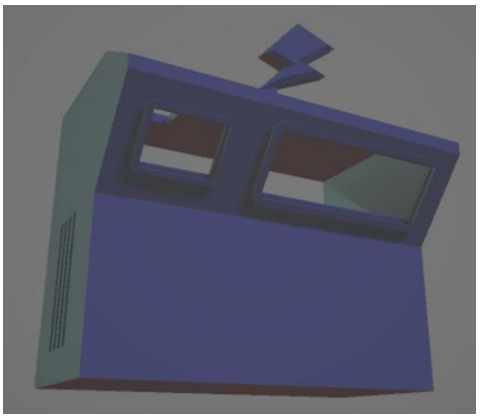



**Gambar 3. 8** Desain PCB

Desain PCB pada Gambar 3.8, merupakan desain yang dikembangkan dari *wiring diagram* yang sudah dibuat, dengan tujuan untuk merapihkan koneksi antar komponen serta meningkatkan kestabilan sistem secara keseluruhan. Desain ini disusun berdasarkan konfigurasi pin yang telah diuji pada tahap perakitan awal menggunakan *breadboard*.

#### 3.4.2.5 Desain Box 3D

Sebagai bagian dari perancangan fisik perangkat, dilakukan pula desain box 3D sebagai wadah utama untuk tempat seluruh komponen elektronik seperti mikrokontroler, sensor, dan modul *display*.

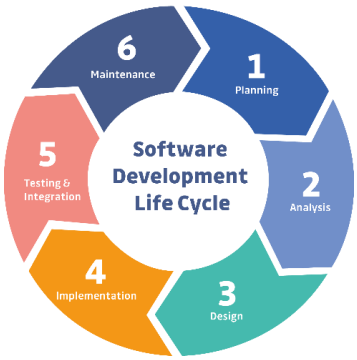


**Gambar 3. 9** Desain Box 3D

Gambar 3.9 menampilkan desain box 3d untuk alat *Smart Energy Meter* pada penelitian ini, desain box dibuat menyesuaikan dengan dimensi komponen agar tidak terjadi tumpang tindih antar perangkat, serta memperhatikan sirkulasi udara untuk mencegah panas berlebih. Selain itu, desain juga mempertimbangkan kemudahan perakitan.

### 3.4.3 Perancangan *Software*

Perancangan *software* pada penelitian ini menggunakan pendekatan *Software Development Life Cycle* (SDLC), yang terdiri dari enam tahapan berurutan seperti pada Gambar 3.10.




**Gambar 3. 10** *Software Development Life Cycle*

Pendekatan SDLC digunakan agar sistem yang dikembangkan berjalan secara terstruktur dan sistematis mulai dari perencanaan hingga evaluasi sistem.

#### 3.4.3.1 *Planning*

Tahap perencanaan merupakan fase awal yang bertujuan untuk merumuskan latar belakang, tujuan pengembangan sistem, serta ruang lingkup secara menyeluruh. Fokus utama dari perencanaan ini adalah membangun *dashboard* berbasis *website* untuk pemantauan energi listrik yang dapat bekerja secara *realtime*, informatif, memiliki kemampuan peramalan konsumsi listrik berbasis model *machine learning,* serta memberikan notifikasi otomatis apabila konsumsi aktual melebihi nilai *forecast*. Perencanaan juga meliputi pemilihan teknologi yang akan digunakan selama proses pengembangan, baik dari sisi antarmuka pengguna (HTML, CSS, JavaScript), server aplikasi (Flask), hingga integrasi dengan sensor dan *cloud database* (Firebase Realtime Database).

#### 3.4.3.2 *Analysis*

Tahap analisis dilakukan untuk mengidentifikasi kebutuhan sistem secara menyeluruh, baik dari aspek fungsional maupun non-fungsional. Hasil analisis ini menjadi dasar dalam proses perancangan dan pengujian sistem di tahap selanjutnya. Kebutuhan fungsional dirancang untuk mencerminkan bagaimana sistem harus bekerja dari perspektif pengguna, kebutuhan fungsional pada sistem ini adalah:

1. Sistem mampu menampilkan parameter kelistrikan secara *realtime*.
2. Pengguna dapat melihat data historis konsumsi listrik berdasarkan tanggal tertentu.

3. Sistem dapat menghasilkan dan menampilkan hasil peramalan energi listrik.
4. Sistem dapat mengirimkan pesan notifikasi melalui Telegram kepada pengguna ketika penggunaan aktual melebihi nilai *forecast*.

Lalu kebutuhan non-fungsional mencakup aspek performa dan pengalaman pengguna seperti:

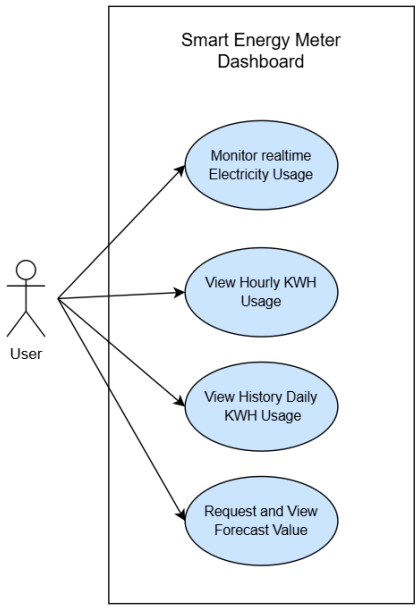
1. Sistem responsif dan dapat diakses.
2. Waktu respon cepat, terutama saat memuat data dan melakukan peramalan.
3. Stabilitas koneksi ke Firebase sebagai sumber data utama.

#### 3.4.3.3 *Design*

Pada tahap ini dilakukan proses desain sistem baik secara struktural maupun visual.

**A.) *Use Case Diagram***

*Use case diagram* pada Gambar 3.11 digunakan untuk memvisualisasikan interaksi antara pengguna dengan sistem.




**Gambar 3. 11** *Use Case Diagram Dashboard*

Gambar 3.11 menggambarkan fitur-fitur utama yang dapat diakses pengguna pada *dashboard* seperti:

- *Monitor realtime electricity usage*: Menampilkan data pemantauan konsumsi listrik secara *realtime*
- *View hourly KWH usage*: Menyajikan grafik konsumsi energi listrik per jam dalam satu hari, yang berguna untuk mengetahui pola penggunaan listrik secara mendetail setiap jamnya.
- *View history daily KWH usage*: Menyediakan histori penggunaan listrik harian yang dapat ditampilkan berdasarkan rentang tanggal tertentu yang dipilih oleh pengguna.
- *Request and view forecast value*: Meminta menampilkan hasil prediksi konsumsi energi listrik untuk periode mendatang, berdasarkan analisis data historis dengan menggunakan metode prediktif.

**B.) *Dashboard Design***

Desain antarmuka pengguna dirancang menggunakan Figma untuk memastikan tampilan sistem menarik dan informatif. Desain *dashboard* yang dibuat seperti pada Gambar 3.12.


**Gambar 3. 12** *Dashboard Design*

Desain yang dikembangkan pada Gambar 3.12 terdiri dari tiga panel utama:

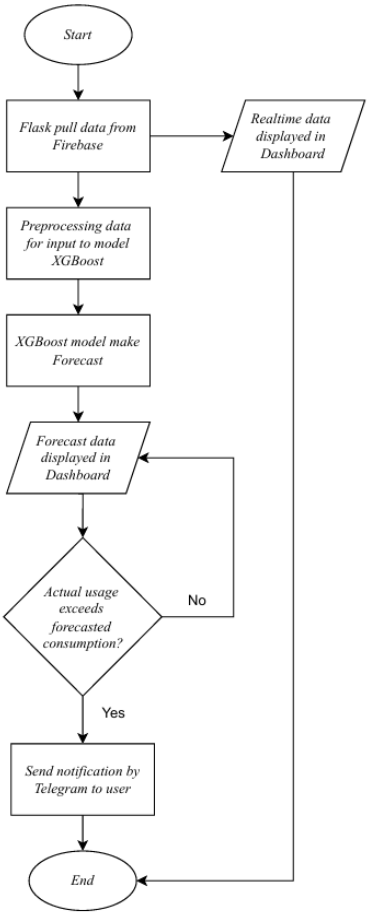
1. *Current Status Panel* : Menampilkan parameter kelistrikan secara *realtime* seperti daya, tegangan, arus, frekuensi, faktor daya, dan konsumsi energi per jam.
2. *24-Hour Energy Consumption* : Grafik untuk menampilkan tren penggunaan listrik selama 24 jam terakhir.

3. *Weekly Report Panel* : Menyajikan ringkasan statistik konsumsi energi dalam periode mingguan.

Desain ini kemudian akan diimplementasikan menggunakan HTML, CSS, dan JavaScript.

**C.) *Backend Architecture***

Bagian *backend* dari sistem ini dirancang untuk menangani seluruh logika pemrosesan data dan integrasi layanan pihak ketiga. *Backend* dikembangkan menggunakan Flask sebagai web server. Alur sistem *backend* seperti pada Gambar 3.13.




**Gambar 3. 13** Alur Kerja *Backend*

Gambar 3.13 menunjukkan Sistem *Smart Energy Meter* berbasis IoT ini bekerja dengan alur terintegrasi antara *backend* Flask dan Firebase, di mana *backend* secara periodik menarik data konsumsi energi dari Firebase dan langsung menampilkannya di *dashboard* sebagai data *realtime*. Data ini kemudian melalui

tahap *preprocessing* sebelum diproses oleh model XGBoost untuk menghasilkan peramalan konsumsi energi pada periode tertentu. Hasil *forecast* ditampilkan berdampingan dengan data aktual di *dashboard*, dan *backend* secara otomatis membandingkan keduanya. Apabila konsumsi aktual melebihi nilai *forecast*, sistem akan mengirimkan notifikasi langsung kepada pengguna melalui Telegram Bot API. Jika tidak, proses terus berulang hingga sistem dihentikan. Integrasi Telegram sebagai saluran notifikasi dipilih karena kemudahan integrasi, kecepatan pengiriman, serta kemampuannya dalam menyampaikan informasi secara instan dan efisien, yang memungkinkan pengguna segera merespons potensi anomali konsumsi listrik secara preventif maupun korektif.

#### 3.4.3.4 *Implementation*

Tahap ini merupakan realisasi teknis dari desain sistem ke dalam bentuk kode program. Implementasi dilakukan secara modular agar mudah dalam proses *debugging* dan pengembangan lanjutan.

#### 3.4.3.5 *Testing & Integration*

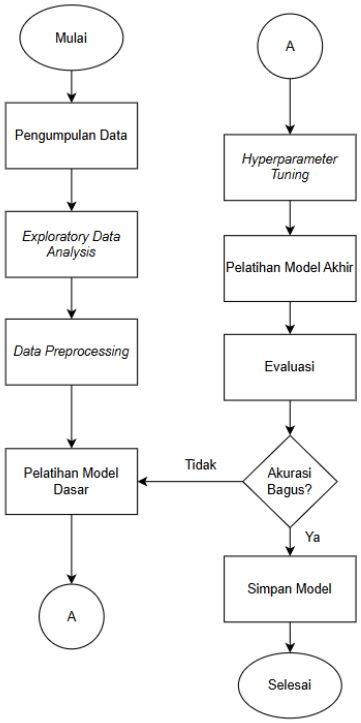
Proses pengujian dilakukan untuk memastikan bahwa sistem berjalan sesuai dengan fungsi yang telah ditentukan. Pengujian yang dilakukan seperti menguji fungsionalitas *button* yang dibuat, grafik yang ditampilkan, dan notifikasi Telegram ketika penggunaan aktual melebihi nilai *forecast*.

#### 3.4.3.6 *Maintenance*

*Maintenance* dilakukan secara terbatas, pemeliharaan hanya mencakup pengecekan koneksi antar komponen seperti Firebase, Flask, dan Telegram Bot untuk memastikan program berjalan tanpa *error*, serta melakukan *update* kode jika ditemukan *bug* atau perlu adanya perbaikan pada logika pemrosesan data.

### 3.4.4 Perancangan Sistem Peramalan

Pada penelitian ini, perancangan sistem peramalan konsumsi energi listrik dilakukan menggunakan algoritma XGBoost yang dirancang secara bertahap dan sistematis seperti pada Gambar 3.14.




**Gambar 3. 14** Alur Pembuatan Model

Setiap tahapan pada Gambar 3.14 dirancang untuk memastikan bahwa model yang dihasilkan memiliki tingkat akurasi yang tinggi serta mampu menangkap pola konsumsi energi secara temporal. Alur lengkap tahapan pembuatan model mencakup proses pengumpulan data hingga evaluasi akhir terhadap performa model.

### 3.4.4.1 Pengumpulan Data

Langkah awal dalam proses pembangunan model XGBoost untuk *forecast* konsumsi energi listrik adalah pengumpulan data historis. Data yang dikumpulkan berupa data historis konsumsi energi listrik rumah tangga dalam satuan *kilowatt-hour* (kWh). Data ini berasal dari pembacaan alat *Smart Energy Meter* yang dikirim ke Firebase. Setelah itu, data energi per jam dari Firebase akan diteruskan ke Google Sheet dan disimpan dalam format *excel* untuk memudahkan proses analisis awal. Data ini dikumpulkan selama lebih dari 30 hari yang dimulai dari tanggal 26 April 2025 hingga 1 Juni 2025, data terdiri dari 2 kolom yaitu waktu dan kWh dan 888 baris data. Setiap baris data memuat informasi waktu dalam format *YYYY_MM_DD_HH* serta jumlah konsumsi listrik pada jam tersebut.

#### 3.4.4.2 *Exploratory Data Analysis*

Tahap ekspolarasi data (EDA) dilakukan untuk memahami struktur, kualitas, dan karakteristik distribusi data berdasarkan data historis yang telah dikumpulkan. Tujuan utama dari tahap ini adalah untuk memastikan bahwa data yang akan digunakan dalam proses pelatihan model berada dalam kondisi bersih, terstruktur, dan representatif terhadap fenomena yang diamati. Proses yang akan dilakukan pada eksplorasi data ini meliputi pemeriksaan struktur data dan tipe data, *parsing* tipe data, pengecekan *missing value*, deskripsi statistik data, visualisasi distribusi data, deteksi *outlier*, dan analisis tren dan pola waktu.

#### 3.4.4.3 *Data Preprocessing*

Tahap *data preprocessing* yang bertujuan untuk memastikan bahwa data dalam kondisi bersih, terstruktur, dan siap digunakan dalam pelatihan model. Langkah-langkah yang dilakukan pada tahap *data preprocessing* berdasarkan hasil eksplorasi data (EDA). Tahapan-tahapan yang akan dilakukan apabila pada proses EDA ditemukan beberapa temuan seperti:

5. *Missing value*, apabila terdapat *missing value* bisa melakukan beberapa teknik seperti menghapus baris data yang hilang atau imputasi data dengan mengisi nilai menggunakan nilai *mean, median*, dan k-NN.
6. *Outlier*, apabila ada nilai data yang berada jauh dari mayoritas data lainnya, data bisa dihapus apabila data tersebut merupakan kesalahan *input* data tetapi apabila *outlier* tersebut mewakili nilai asli berdasarkan fenomena yang terjadi maka data dipertahankan.

Selanjutnya, dilakukan proses *features engineering* yang bertujuan untuk pembuatan beberapa kelompok fitur untuk meningkatkan kemampuan model dalam memahami pola data konsumsi listrik berbasis waktu. *Temporal features* seperti (*hour, day_of_week, month, is_weekend*) diekstrak dari atribut waktu untuk merepresentasikan konteks waktu yang berpengaruh terhadap perilaku konsumsi. Fitur-fitur ini penting karena konsumsi listrik cenderung menunjukkan pola musiman atau siklus harian dan mingguan yang khas, misalnya peningkatan konsumsi pada malam hari atau perbedaan antara hari kerja dan akhir pekan. Selanjutnya, untuk menangani sifat waktu yang bersifat siklikal maka dibuat

*cyclical features* menggunakan transformasi sinus dan kosinus pada fitur (*hour, day_of_week, month*). Fitur ini memungkinkan model untuk mengenali transisi alami antar periode waktu, seperti dari pukul 23 ke pukul 0 atau dari Desember ke Januari, yang tidak bisa ditangkap dengan representasi numerik biasa. Selain itu, *lag features* disusun untuk memasukkan informasi historis konsumsi listrik, karena nilai masa lalu sering kali menjadi indikator kuat terhadap nilai saat ini. Kemudian, disusun juga *rolling statistics* seperti *rolling_mean, rolling_std, rolling_max,* dan *rolling_min* pada jendela waktu tertentum yang bertujuan untuk menangkap tren lokal serta fluktuasi dalam periode pendek. Fitur *differencing* digunakan untuk menangkap arah perubahan dan pola tren antar waktu. Keseluruhan fitur ini dirancang untuk memperkaya representasi data, memungkinkan model XGBoost mengenali hubungan temporal secara lebih akurat dan efektif dalam melakukan peramalan konsumsi listrik.

Setelah proses *features engineering*, dilakukan proses seleksi fitur untuk menyaring fitur yang relevan terhadap target. Pendekatan yang digunakan adalah visualisasi matriks korelasi menggunakan *heatmap*. *Heatmap correlation* digunakan untuk mengevaluasi hubungan linier antara fitur dan target. Fitur yang memiliki nilai korelasi positif atau negatif yang cukup tinggi terhadap target dianggap memiliki kontribusi yang signifikan dalam membentuk prediksi.

#### 3.4.4.4 Pelatihan Model Dasar

Pada proses pelatihan model dasar dilakukan beberapa strategi pelatihan. Pertama melakukan pelatihan model dengan *basic temporal features time series* seperti (*hour, day_of_week, day_of_month, month, is_weekend*). Setelah itu, proses pelatihan dengan menggunakan seluruh fitur hasil *features engineering* dan yang terakhir adalah proses pelatihan dengan fitur-fitur pilihan yang didapatkan dari proses seleksi fitur dengan *heatmap*. Beberapa percobaan ini bertujuan untuk menguji apakah dengan semakin banyak fitur model akan semakin bagus atau sebaliknya.

#### 3.4.4.5 *Hyperparameter Tuning*

Proses *hyperparameter tuning* bertujuan untuk mencari kombinasi parameter terbaik yang dapat meningkatkan kinerja model XGBoost secara optimal. Dalam

penelitian ini digunakan sebuah *framework hyperparameter optimization* berbasis Python yaitu Optuna. Optuna dipilih karena mampu melakukan pencarian nilai *hyperparameter* secara adaptif menggunakan pendekatan *Bayesian optimization* dengan algoritma *Tree-structured Parzen Estimator* (TPE). Pada tahap ini, tuning dilakukan terhadap beberapa *hyperparameter* penting dari algoritma XGBoost, yang secara langsung memengaruhi kompleksitas model dan akurasi *forecast*. Adapun parameter yang dioptimasi beserta rentang pencariannya seperti yang tercantum pada Tabel 3.2.

**Tabel 3. 2** Rentang pencarian *hyperparameter*

| *Hyperparameters* | *Search Range* |
|---|---|
| *n_estimator* | 50 - 1000 |
| *learning_rate* | 0,01 - 1 |
| *max_depth* | 2 – 6 |
| *min_child_weight* | 1 – 10 |
| *gamma* | 0 – 1 |
| *subsample* | 0,1 – 1 |
| *colsample_bytree* | 0 – 1 |
| *reg_alpha* | 0 – 1 |
| *reg_lambda* | 0 – 1 |

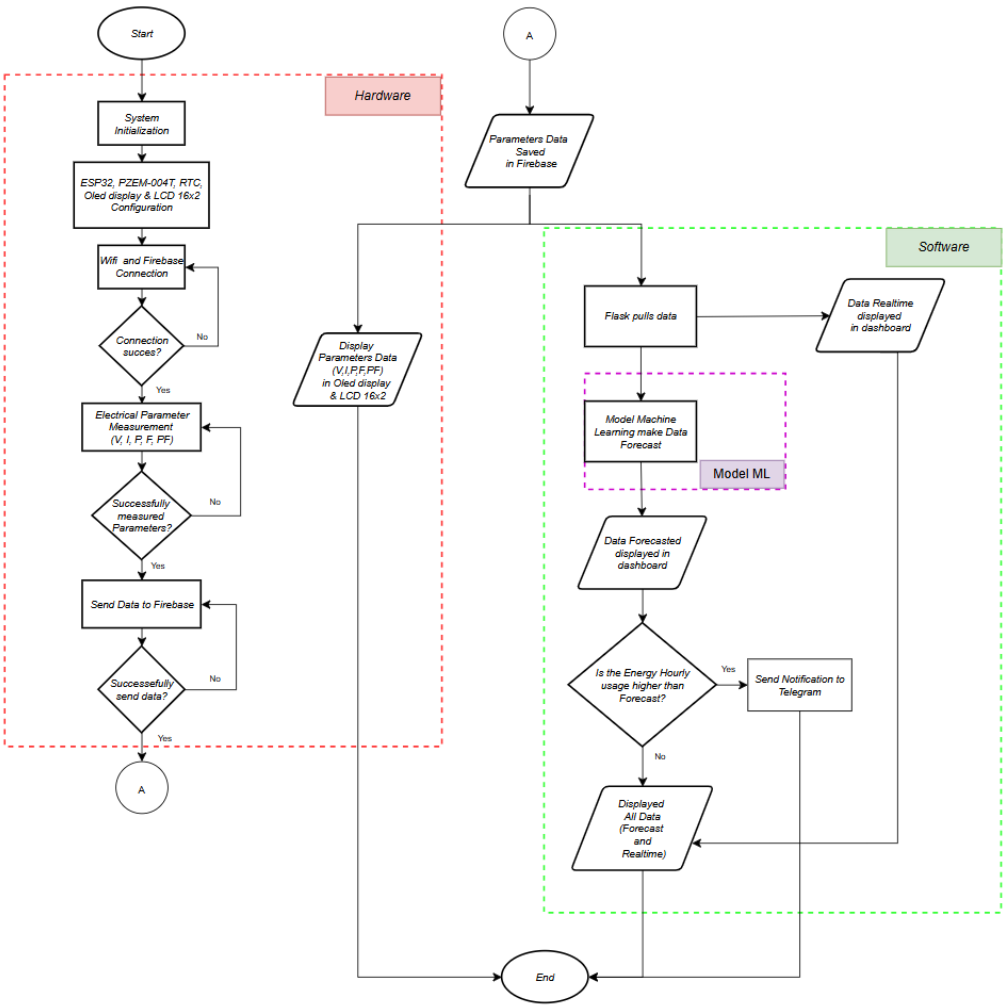
*Search range* dari parameter yang ada pada Tabel 3.2 telah disesuaikan berdasarkan dokumentasi resmi parameter XGBoost. Parameter *n_estimator* mengatur jumlah pohon yang digunakan dan mempengaruhi performa serta waktu pelatihan, nilai yang lebih tinggi dapat meningkatkan akurasi namun memperpanjang waktu pelatihan. Parameter *max_depth* berfungsi untuk membatasi kedalaman pohon dan mencegah *overfitting*, sementara *min_child_weight* dan *gamma* berperan sebagai parameter regularisasi yang mengontrol pembentukan *node* baru berdasarkan bobot minimum dan penurunan *loss* yang dihasilkan. Parameter *subsample* dan *colsample_bytree* digunakan untuk memperkenalkan elemen acak dalam pemilihan data dan fitur, sehingga membantu mengurangi risiko *overfitting* dan meningkatkan generalisasi model. Selain itu, *reg_alpha* (L1 regularization) dan *reg_lambda* (L2 regularization) memberikan penalti terhadap bobot model guna menjaga kesederhanaan struktur dan menghindari kompleksitas

berlebih. Keseluruhan parameter ini perlu dioptimasi dengan mempertimbangkan *trade-off* antara *bias* dan *varians*, serta disesuaikan dengan karakteristik data untuk menghasilkan model yang efektif dan *robust*.

**3.4.4.6 Pelatihan Model Akhir dan Evaluasi**

Setelah parameter dan kombinasi fitur terbaik didapatkan, tahapan selanjutnya adalah pelatihan model menggunakan parameter dan fitur terbaik tersebut. Proses pelatihan dan evaluasi model dilakukan dengan pendekatan *TimeSeriesSplit* sebanyak lima *fold* (*n_splits* = 5), yang secara khusus dirancang untuk menangani data *time series* dengan menjaga urutan temporal data sehingga tidak terjadi *data leakage* antar lipatan. Setiap fold terdiri dari bagian data pelatihan dan data validasi yang beruntun berdasarkan waktu, dan model akan dievaluasi pada *fold* terakhir (*fold* ke 5), yang dianggap mewakili kondisi data terbaru dan mendekati skenario nyata. Selanjutnya, dilakukan evaluasi kinerja model pada data uji menggunakan tiga metrik evaluasi, yaitu MAE, RMSE, dan $R^2$.

## 3.5 Integrasi Sistem Keseluruhan




**Gambar 3. 15** *Flowchart Smart Energy Meter*

*Flowchart* pada Gambar 3.15 menggambarkan alur integrasi sistem *Smart Energy Meter* yang terdiri dari tiga komponen utama, yaitu IoT *hardware, software* (*dashboard dan backend*), serta model *machine learning* (XGBoost). Integrasi antar ketiga bagian ini dilakukan melalui jalur komunikasi berbasis internet dengan memanfaatkan Firebase *Realtime* Database sebagai pusat pertukaran data. Proses dimulai dari sisi *hardware*, dimana ESP32 berperan sebagai mikrokontroler yang membaca parameter kelistrikan dari sensor PZEM-004T (tegangan, arus, daya, frekuensi, dan faktor daya). Setelah data diperoleh, ESP32 mengirimkannya ke Firebase menggunakan koneksi WiFi melalui protokol HTTPS.

Data yang tersimpan di Firebase kemudian diakses oleh *backend* yang dikembangkan menggunakan *framework* Flask. Flask bertugas mengambil (*pull*) data secara *realtime* dari Firebase melalui Firebase REST API, lalu mendistribusikannya ke dua jalur utama, yaitu menampilkan data parameter *realtime* di *dashboard* dan memproses data energi setiap jam sebagai input untuk model peramalan energi berbasis XGBoost. Model XGBoost ini tidak berjalan secara terus-menerus, melainkan di *load* ke dalam memori Flask menggunakan *library* seperti joblib atau pickle saat server dijalankan, dan proses prediksi hanya dilakukan saat pengguna mengakses *endpoint* tertentu (misalnya dengan menekan tombol pada *dashboard*). Hal ini memberikan fleksibilitas dan efisiensi, karena proses inferensi dilakukan secara *on-demand* berdasarkan permintaan pengguna.

Setelah tombol ditekan, Flask memproses data historis dari Firebase, menjalankan model XGBoost untuk menghasilkan *forecast* konsumsi energi, lalu mengembalikan hasil *forecast* tersebut ke *dashboard*. Selanjutnya, sistem akan membandingkan antara hasil *forecast* dengan data aktual konsumsi energi secara *realtime*. Jika konsumsi aktual lebih tinggi dari data *forecast*, sistem akan secara otomatis mengirimkan notifikasi ke Telegram menggunakan Telegram Bot API, untuk memberikan peringatan kepada pengguna.

Dengan demikian, integrasi sistem ini berjalan melalui komunikasi terstruktur yang melibatkan WiFi, HTTPS, dan API Firebase sebagai media pertukaran data. Flask berperan sebagai pengendali utama yang menjembatani data dari *hardware*, mengelola model *machine learning*, serta menyajikan informasi ke pengguna melalui *dashboard* dan notifikasi Telegram. Alur integrasi ini memungkinkan sistem berjalan secara efisien, cerdas, dan responsif terhadap situasi konsumsi energi aktual.

### 3.6 Metode Pengujian *Prototype Smart Energy Meter*

Pengujian *prototype Smart Energy Meter* dilakukan untuk menguji tingkat keberhasilan sistem yang dirancang dengan melakukan pengujian fungsionalitas. Adapun tahapan pengujian yang dilakukan sebagai berikut:

### 3.6.1 Pengujian Fungsionalitas Sistem

Pengujian sistem *prototype Smart Energy Meter* menggunakan pengujian *black box* untuk menguji sistem yang telah dirancang seperti pada Tabel 3.3.

**Tabel 3. 3** Metode Pengujian Fungsionalitas *Smart Energy Meter*

| Fitur | Fungsi | Hasil yang diharapkan | Hasil Pengujian | Status |
|---|---|---|---|---|
| Koneksi internet | Sistem *Smart Energy Meter* terhubung internet melalui WiFi | Berhasil terhubung ke internet dan mengirim data ke Firebase | | |
| Pembacaan sensor | Membaca parameter listrik | Data terbaca dan tampil di LCD dan Firebase | | |
| Tampilan LCD | Menampilkan data *realtime* | Data tampil sesuai nilai aktual sensor | | |
| Pengiriman ke Firebase | Data dikirim ke *cloud* secara *realtime* | Firebase menerima data dengan struktur yang sesuai | | |
| *Web dashboard* | Menampilkan data dan grafik | Data *realtime* dan grafik tampil responsif dan *realtime* | | |
| Notifikasi Telegram | Mengirim peringatan jika konsumsi melebihi nilai *forecast* | User berhasil menerima pesan peringatan | | |

### 3.6.2 Pengujian Pencatatan Data Energi dari *Smart Energy Meter*

Pengujian ini bertujuan untuk memastikan bahwa sistem pencatatan data konsumsi energi listrik melalui perangkat *Smart Energy Meter* berjalan dengan baik. Fokus utama dari pengujian adalah memverifikasi apakah data konsumsi energi yang terbaca oleh perangkat berhasil dikirim ke Firebase *Realtime* Database, serta secara otomatis tercatat di dalam Google Spreadsheet sebagai data historis. Pengujian dilakukan dengan mengamati proses perekaman data setiap satu jam,

kemudian mencocokkan waktu dan nilai pembacaan antara Firebase dan Spreadsheet.

### 3.6.3 Pengujian Hasil *Forecasting* Model

Pengujian ini dilakukan untuk menilai performa model XGBoost yang telah dilatih, dengan cara membandingkan hasil *forecast* terhadap data konsumsi energi aktual yang diperoleh setelah model selesai dilatih. Pengujian dilakukan pada data terbaru yang belum pernah digunakan dalam proses pelatihan model, untuk mengukur kemampuan generalisasi model dalam meramal pola konsumsi energi yang baru.

## 3.7 Metode Penentuan Penggunaan Listrik yang Efisien

Penggunaan listrik dikatakan efisien apabila tidak melebihi ambang batas Intensitas Konsumsi Energi (IKE) yang telah ditentukan berdasarkan standar atau target tertentu. Dalam penelitian ini, pengujian efisiensi energi dilakukan secara mingguan selama bulan Juni, dengan membandigkan antara penggunaan aktual, *forecast*, dan hasil perhitungan IKE. Pengujian penentuan efisiensi ini diawali dengan pembuatan *forecast* data untuk satu minggu yang dijadikan acuan penggunaan, lalu data aktual penggunaan dikumpulkan selama periode data *forecast* tersebut. Setelah data aktual dan *forecast* selama periode tertentu didapat, kedua nilai tersebut diubah ke dalam bentuk nilai IKE Perhitungan IKE seperti pada persamaan (3.1).

$$IKE = \frac{Pemakaian\ Listrik\ Aktual\ (kWh)}{Luas\ Bangunan\ (m^2)} \tag{3.1}$$

Dari persamaan (3.1) nantinya akan didapatkan nilai IKE untuk menentukan apakah penggunaan energi tergolong efisien atau tidak.