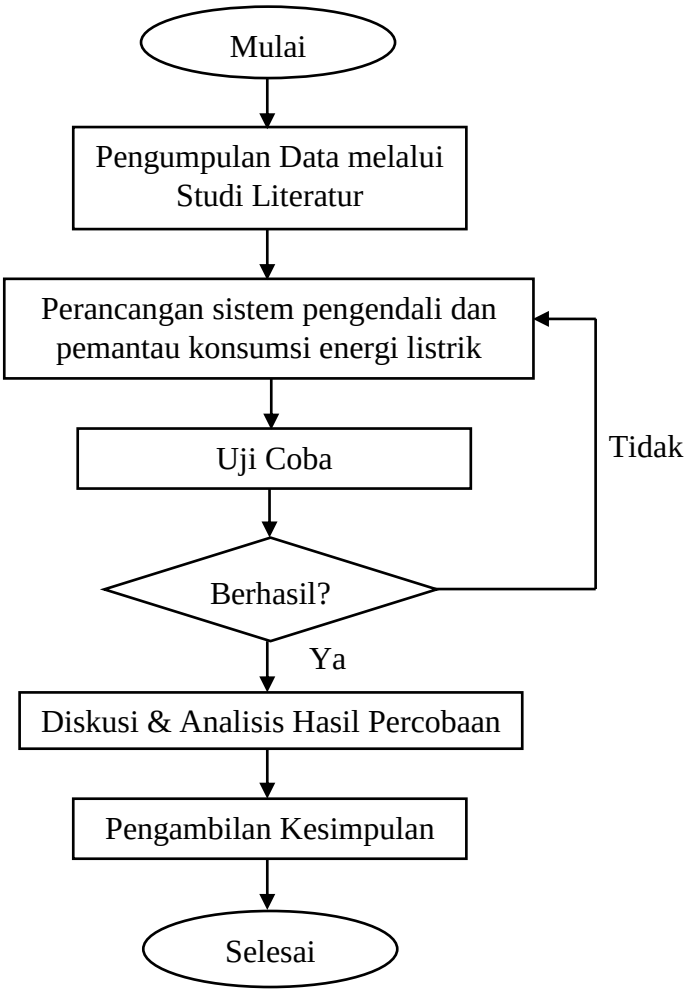
BAB III
METODE PENELITIAN

3.1 Metode Penelitian

Metode yang digunakan untuk melaksanakan penelitian ini adalah metode eksperimen. Metode eksperimen digunakan untuk merancang bangun sistem pengendali dan pemantau konsumsi energi listrik berbasis ESP-12E dengan menggunakan IoT. Pada Gambar 3.1 diperlihatkan diagram alur yang dilakukan.



Gambar 3.1 Diagram alur penelitian

Pada tahap awal, studi literatur dilakukan untuk mendapatkan teori-teori yang dibutuhkan berupa data maupun informasi yang mendukung akan adanya penelitian ini. Data yang dikumpulkan berupa materi-materi tentang

mikrokontroller ESP-12E dan penerapannya serta penggunaan *software* Android Studio.

Kemudian selanjutnya dilakukan perancangan. Perancangan yang dilakukan adalah untuk merancang bangun sistem pengendali dan pemantau konsumsi energi listrik berbasis ESP-12E dengan menggunakan IoT secara keseluruhan, baik perangkat keras maupun perangkat lunak pada prototipe *smart socket* dan Android. Perancangan pcb dibuat dengan menggunakan *software* Eagle yang kemudian akan di cetak pada pcb tipe FR4. Kemudian dilakukan *soldering* komponen pada pcb yang telah dirancang. Kemudian dilakukan peng-*upload*-an program pada mikrokontroler ESP8266E dengan menggunakan FTDI USB to TTL.

Pada prototipe dilakukan perancangan *User Interface* (UI) sebagai tampilan antarmuka pada saat melakukan konfigurasi pada prototipe *Smart Socket.* Sedangkan pada aplikasi androidnya diperlukan perancangan *layout*, contohnya seperti tombol *on-off*, input alamat tujuan & fitur lainnya. Dari *layout* yang sudah dibuat akan diberikan perintah-perintah dengan menggunakan Javascript.

Setelah dilakukan perancangan, selanjutnya hasil perancangan akan diimplemantasikan pada perangkat keras dan perangkat lunak yang ada. Setelah sistem pengendali dan pemantau konsumsi energi listrik berbasis ESP-12E dengan menggunakan IoT telah berhasil dibuat, dilakukan pembahasan pada hasil uji coba perangkat dan juga prinsip kerja dari prototipe *Smart Socket* yang telah dibuat. Dari masing-masing hasil diskusi, dibuat analisisnya.

Tahap akhir dari penelitian ini adalah pengambilan kesimpulan berdasarkan tujuan dan rumusan masalah penelitian serta dari pembahasan dan analisis dari perangkat yang telah dibuat.

3.2 Perancangan Sistem Pengendali dan Pemantau Konsumsi Energi Listrik Berbasis ESP12E dengan Menggunakan IoT

Sistem pengendali dan pemantau konsumsi energi listrik berbasis ESP-12E dengan menggunakan IoT terdiri dari prototipe *smart socket* dan aplikasi Android.

Berikut adalah perancangan sistem pengendali dan pemantau konsumsi energi listrik berbasis ESP-12E dengan menggunakan IoT.

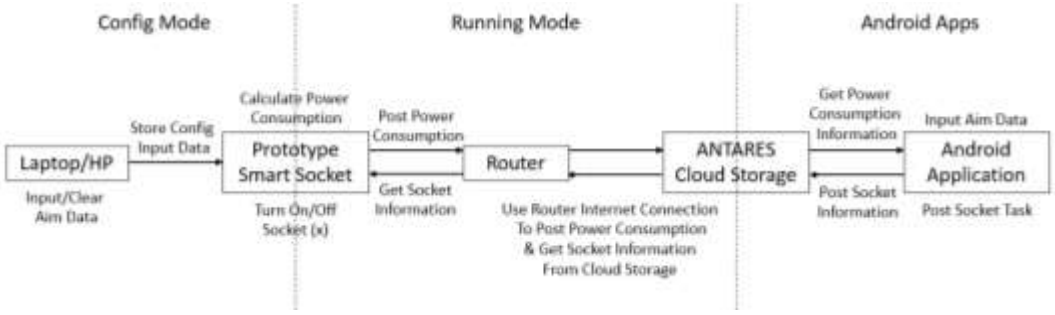
3.2.1 Perangkat penunjang penelitian

Perangkat keras dan perangkat lunak penunjang penelitian yang digunakan terbagi menjadi 2 bagian, yaitu perangkat keras dan perangkat lunak. Adapun perangkat keras yang digunakan di antaranya adalah ponsel Android OS 5.0 Lollipop, FTDI USB to TTL, jumper, mini USB, modul ESP-12E, Konverter AC 220V to DC 5V, *Voltage Regulator* AMS1117, IC CD4052, Sensor arus ACS712 058 5V, relay AC 250V, buzzer aktif 5V, transitor NPN SMD 4096, Dioda scotthcy SS34, kontak terminal 4 slot, terminal block, kabel tembaga diameter 1.5mm, steker 220V, on off switch, on button, LED, trimpot, resistor 100Ω, 1kΩ & 10kΩ, kapasitor, dan *spicer*.

Semetara itu, perangkat lunak yang digunakan terdiri dari Android Studio, POSTMAN, Big NOX, Arduino IDE, Eagle, Adobe Reader pdf, Google chrome (Browser) dan ANTARES Cloud Storage.

3.2.2 Prinsip kerja

Prinsip kerja sistem pengendali dan pemantau konsumsi energi listrik berbasis ESP-12E dengan menggunakan IoT digambarkan dengan menggunakan diagram blok yang ditunjukkan pada Gambar 3.2.



Gambar 3.2 Diagram blok prinsip kerja prototipe *smart socket*

Diagram blok pada Gambar 3.2 menjelaskan bahwa prototipe *smart socket* dibagi menjadi 2 mode kerja yaitu *config mode* dan *running mode*. Pada *config mode*, dilakukan komunikasi secara lokal antara prototipe s*mart socket* dengan Laptop/HP. Pada saat *config mode*, prototipe *smart socket* akan menjadi *host*

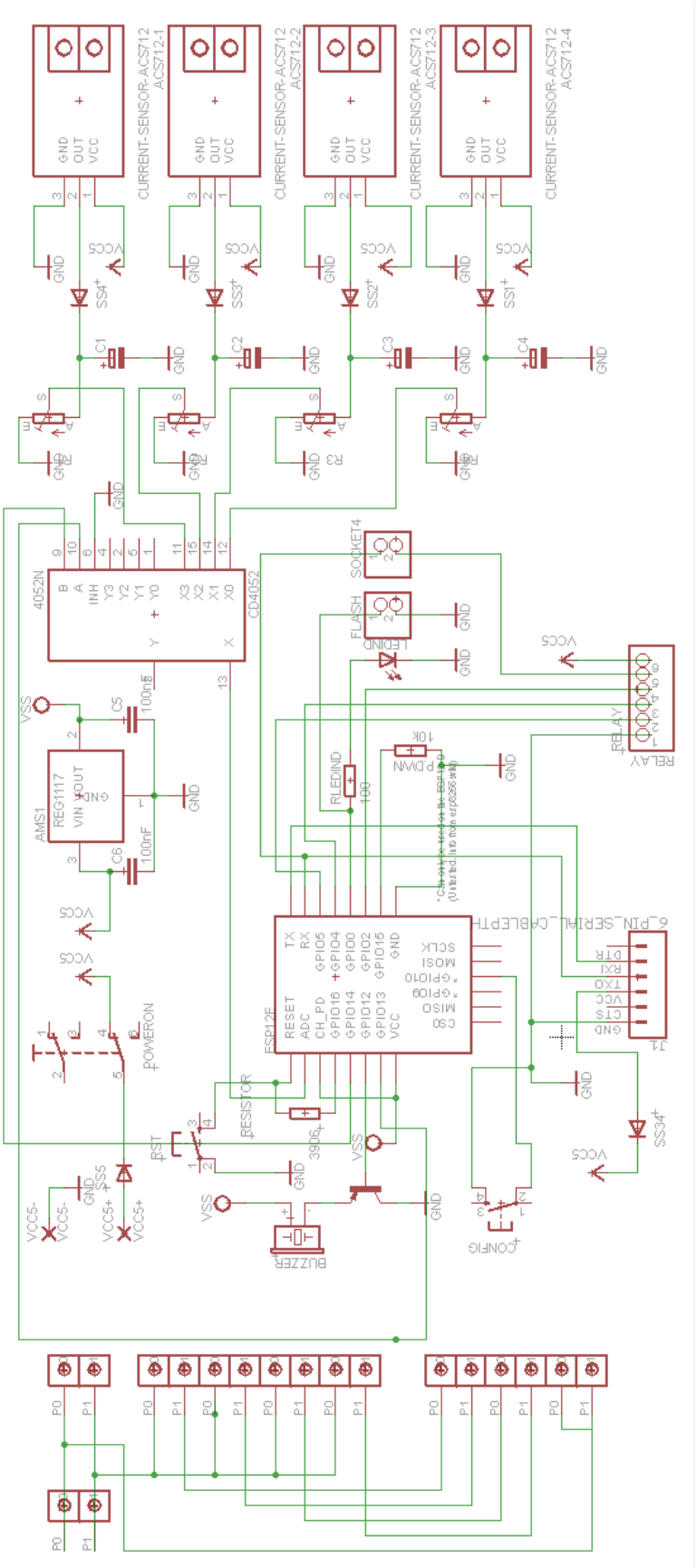
(access point). Config mode dibuat dengan tujuan untuk melakukan pengaturan prototipe. *Config mode* dilengkapi dengan webserver lokal dengan alamat urlnya menggunakan alamat ip gateaway. Webserver lokal di desain dengan menggunakan bahasa HTML, CSS dan Javascript dengan format file .txt.

Pada *running mode*, prototipe *smart socket* akan mengambil data dari ANTARES *Platform* dan melakukan perhitungan total konsumsi energi listrik pada *socket*. Protokol HTTP digunakan untuk melakukan komunikasi data antara prototipe *smart socket* dengan *server*. Untuk mengambil informasi tentang status *socket* digunakan metode GET. Informasi status *socket* akan langsung diterapkan pada masing-masing *socket* dengan mengontrol relay. Sedangkan untuk mengirim informasi konsumsi energi listrik digunakan metode POST. Perhitungan konsumsi energi listrik dihitung tiap 13 detik sekali. Energi listrik akan dihitung hanya pada saat *socket* dalam keadaan aktif dan sedang digunakan.

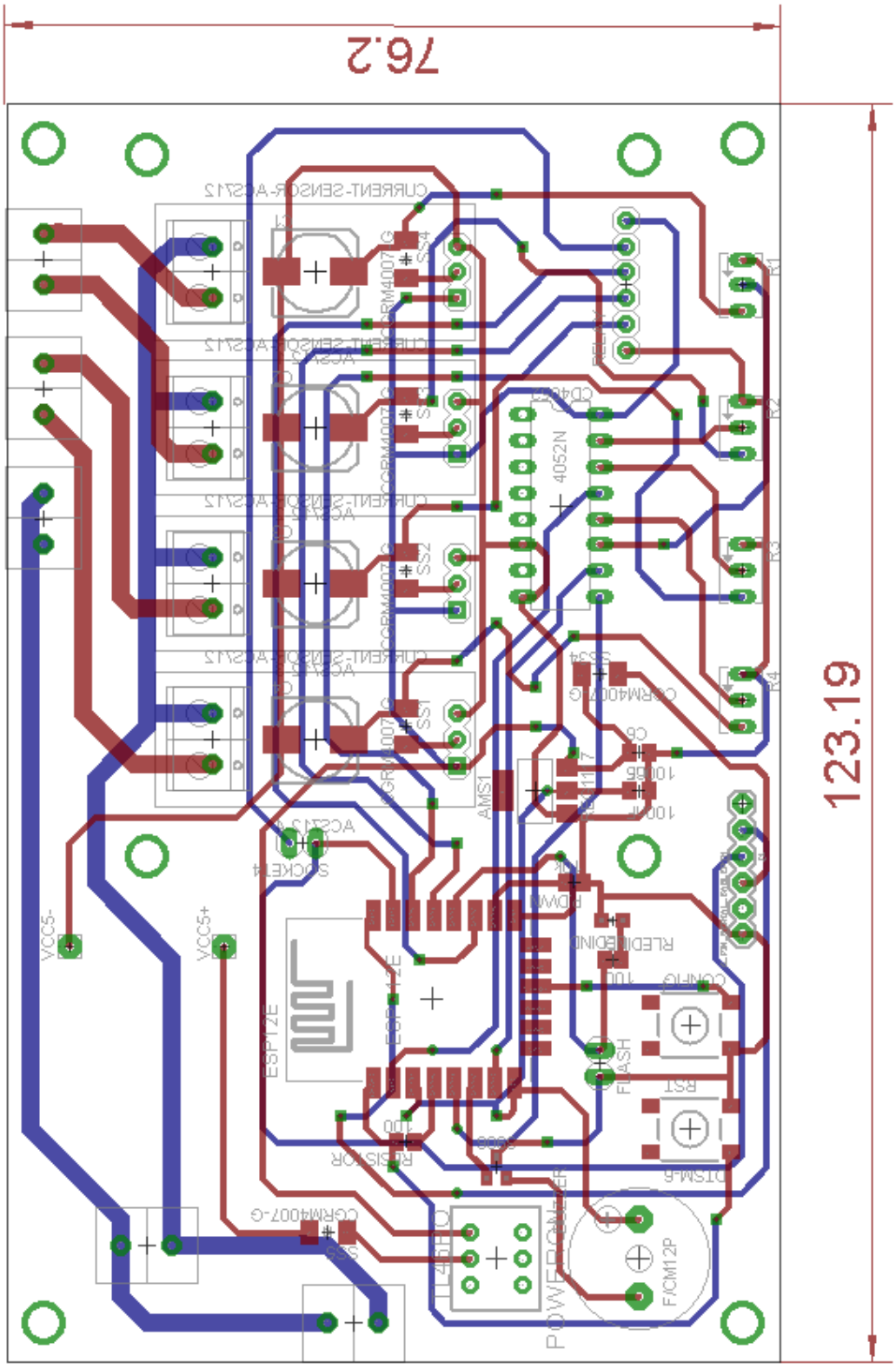
Aplikasi android digunakan untuk melakukan kontrol relay pada prototipe *smart socket* dan mengambil informasi total konsumsi energi listrik pada masing-masing *socket.* Protokol HTTP digunakan untuk melakukan komunikasi antara telepon pintar dan *server*. Untuk mengirimkan informasi status *socket* metode POST dengan dilengkapi konten data dengan format XML. Untuk melakukan POST status *socket*, digunakan tombol *switch* pada *main activity* aplikasi Android. Sedangkan untuk mengambil informasi total konsumsi energi listrik menggunakan metode GET. Informasi yang diambil akan ditampilkan pada *main activity* aplikasi Android dengan satuan mWh. Selain itu, digunakan server ANTARES sebagai media perantara antara aplikasi Andorid dan prototipe *smart socket*.

3.2.3 Desain skematik & PCB prototipe *smart socket*

Pada perancangan skematik dan desain PCB digambarkan susunan komponen dan jalur yang digunakan untuk membuat prototipe *smart socket.* Perancangan dilakukan menggunakan *software* Eagle. Perancangan skematik rangkaian dan desain PCB prototipe *smart socket* ditunjukkan pada Gambar 3.3 dan Gambar 3.4.



Gambar 3.3 Skematik Rangkaian prototipe *smart socket*

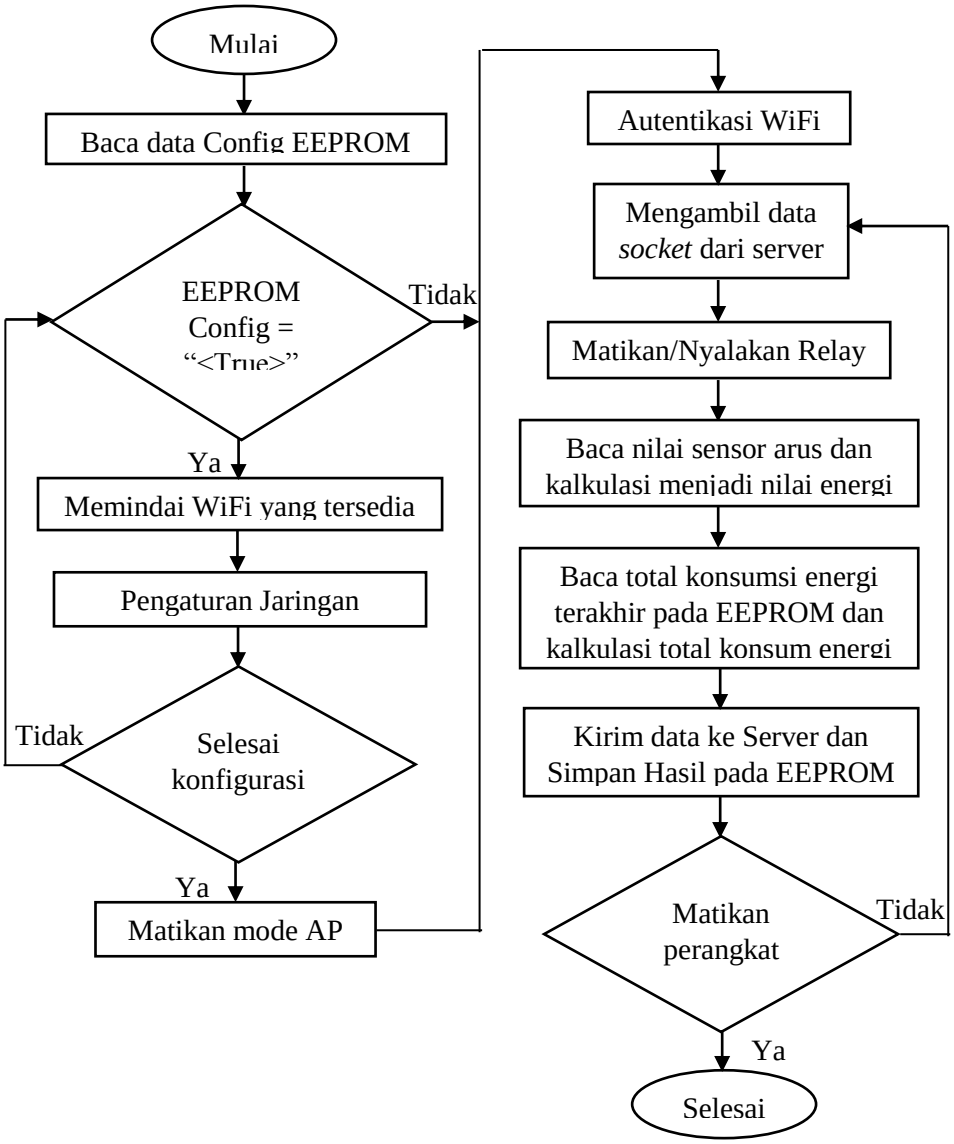


Gambar 3.4 Desain PCB prototipe *smart socket*

3.2.4 Algoritma

1. Algoritma prototipe *smart socket*

Algortima prototipe *smart socket* digambarkan dengan menggunakan diagram blok yang ditunjukkan pada Gambar 3.5.



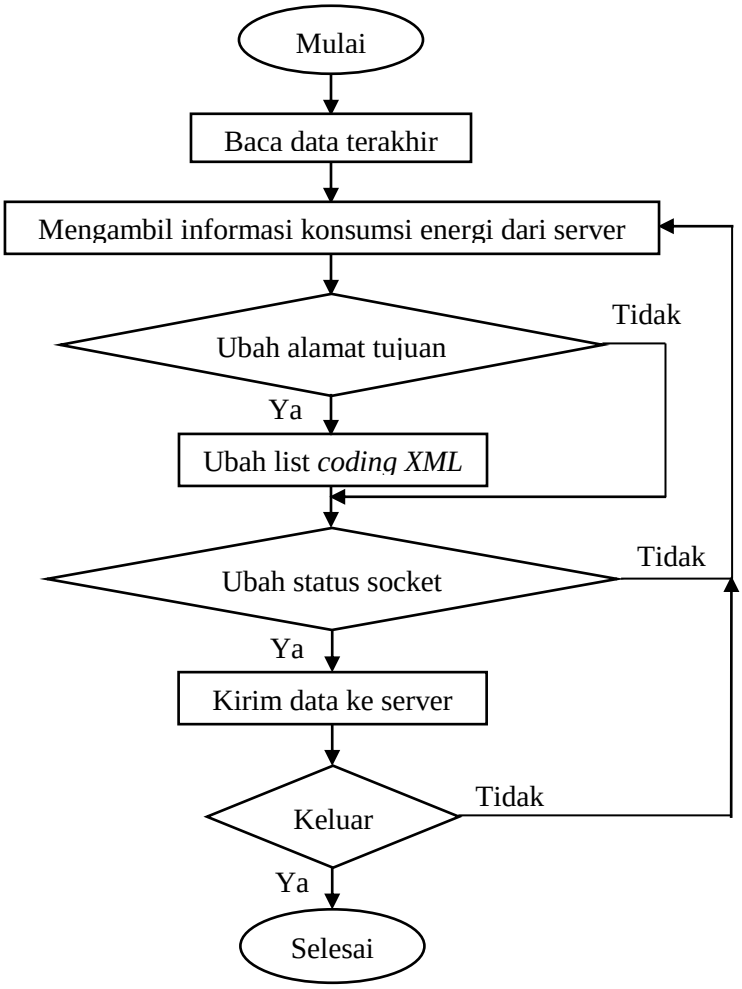
Gambar 3.5 Diagram blok algoritma prototipe *smart socket*

Berikut adalah penjelasan mengenai urutan algoritma pada prototipe *smart socket* yang dirancang:

1. Tahap inisialisasi, pada tahap ini, pertama-tama *smart socket* akan membaca data untuk konfigurasi yang ada pada memori EEPROM. Data hasil pembacaan data konfigurasi EEPROM akan menentukan langkah selanjutnya. Jika data data konfigurasi berisikan “true”, akan dilanjutkan ke langkah 5. Sedangkan jika data konfigurasi berisikan “false”, akan lanjut ke langkah 2.
2. Melakukan *scanning* WiFi pada area yang terjangkau.
3. Melakukan konfigurasi pada data kebutuhan komunikasi atau melakukan pembacaan energi secara lokal.
4. Jika data sudah selesai dikonfigurasi maka *smart socket* akan mematikan mode *access point* dan akan menjadi mode *client.*
5. Melakukan autentikasi pada WiFi yang telah dikonfigurasi.
6. Setelah terhubung pada WiFi, akan didapatkan koneksi internet. Kemudian akan digunakan untuk mengambil data tentang status *socket*.
7. Jika data yang terbaca <onn(x)> maka *smart socket* akan menyalakan *socket* nomor (x). Jika yang terbaca <off(x)> maka *smart socket* akan mematikan *socket* nomor (x).
8. Melakukan kalkulasi konsumsi energi listrik yang digunakan tiap 13 detik.
9. Melakukan *posting* total konsumsi energi listrik yang digunakan selama *smart socket* bekerja.
10. Akhir *looping*, jika data pada konfigurasi ingin dirubah dapat diberikan input pada GPIO10 untuk kembali pada langkah 1. Jika tidak ada perubahan, *smart socket* akan terus melakukan *looping* dari langkah 6 sampai 10.

2. Algoritma aplikasi Android

Algortima aplikasi android untuk mengontrol prototipe s*mart socket* digambarkan dengan menggunakan diagram blok yang ditunjukkan pada Gambar 3.6.



Gambar 3.6 Diagram blok algortima aplikasi Android

Berikut adalah penjelasan mengenai urutan algoritma pada aplikasi Android *smart socket* yang dirancang:

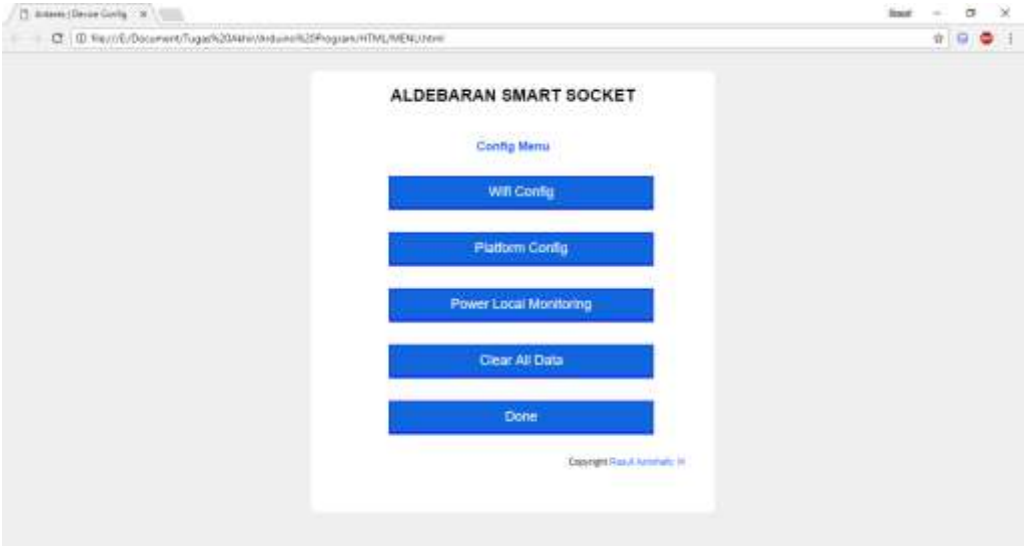
1. Tahap inisialisasi, pertama-tama aplikasi akan membaca input data terakhir akan nama socket yang diberikan.
2. Dengan menggunakan *permission*, didapatkan koneksi internet pada aplikasi Android *smart socket* yang kemudian akan digunakan untuk mengambil informasi tentang total konsumsi energi listrik pada tiap *socket.* Informasi konsumsi energi akan diambil secara otomatis tiap 10 detik sekali.

3. Jika data server tujuan tidak dirubah, akan dilanjutkan ke langkah 6. Jika data diubah akan dilanjutkan ke langkah 4.
4. Jika data ingin diubah, dapat diakses melalui menu setting yang terletak pada pojok kanan atas *layout main activity*. Pada bagian setting, dapat diubah data tentan *server* tujuan.
5. Jika data *server* tujuan diubah, secara otomatis, aplikasi android *smart socket* akan merubah barisan *coding* pada bagian HTTP *Post* XML.
6. Jika pengguna menekan tombol *switch* pada *layout* aplikasi android, akan dilanjutkan ke langkah 7. Jika tidak akan kembali ke langkah 3.
7. Jika tombol *switch* pada *layout* aplikasi android ditekan, secara otomatis aplikasi akan melakukan *posting* data pada *server* yang dituju.

3.2.5 Perancangan *User Interface* (UI)

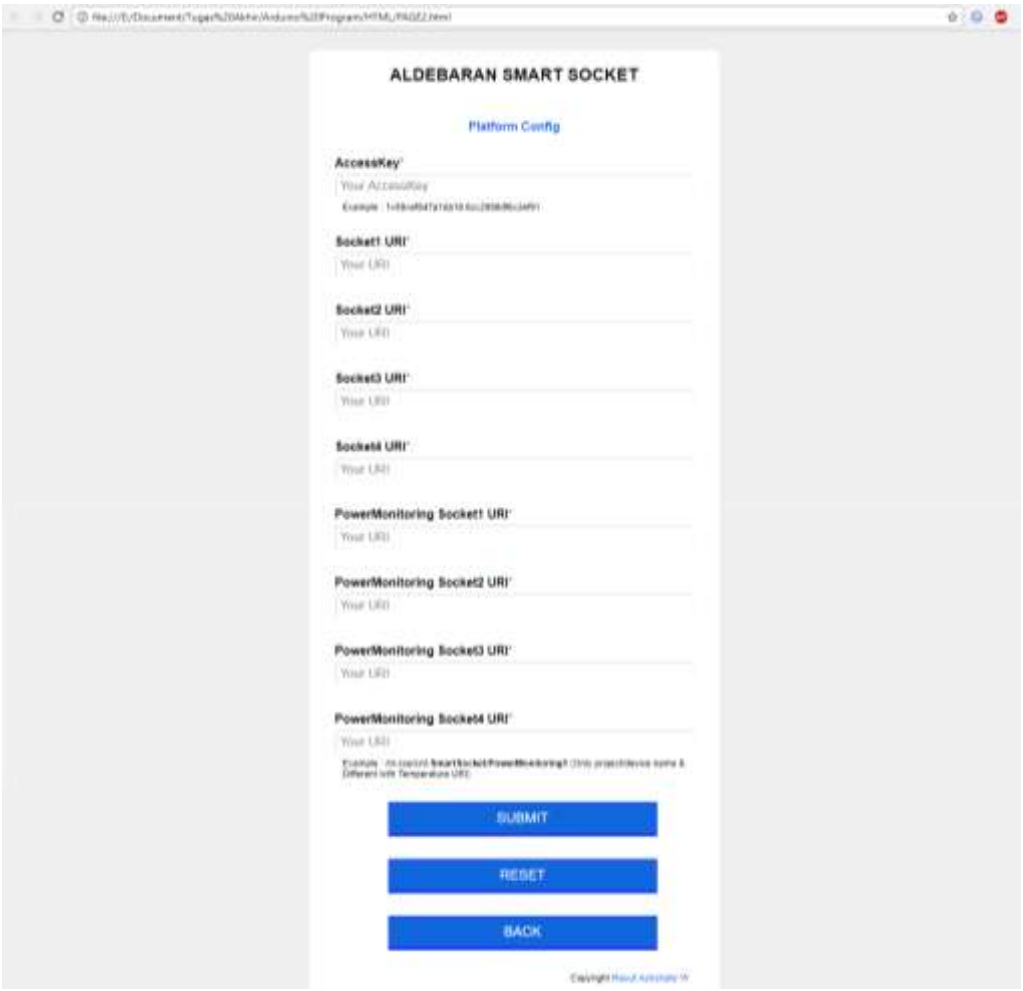
1. *User Interface* (UI) untuk *config* mode pada prototipe *smart socket*

Untuk membuat UI pada prototipe s*mart socket* digunakan bahasa *markup* HTML untuk membuat desain *web*. Untuk membuat validasi data, digunakan tambahan *coding* Javascript. Dan untuk membuat tampilan menjadi lebih menarik digunakan *coding* CSS. Agar dapat mencakup beberapa *menu config*, halaman *web* dibuat sebanyak 4 halaman yaitu *main menu page, WiFi config page, Platform config page* dan *Local Power Monitoring page.* Masing-masing halaman tersebut ditunjukkan pada Gambar 3.7, Gambar 3.8, Gambar 3.9 dan Gambar 3.10.

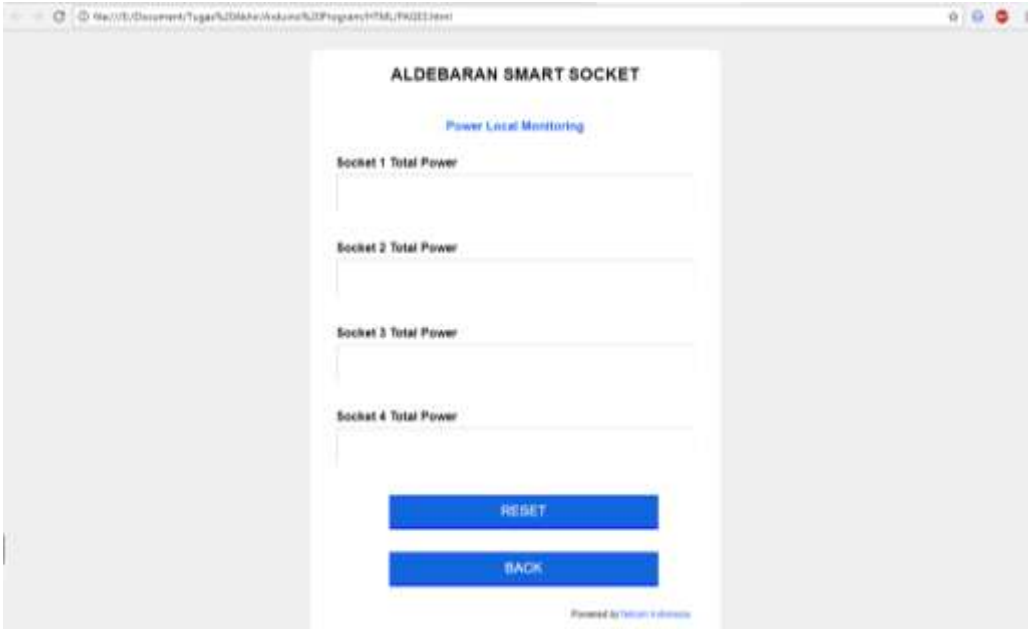


Gambar 3.7 *Menu* utama prototipe *smart socket*

Gambar 3.8 *Menu* WiFi *Config* pada prototipe *smart socket*



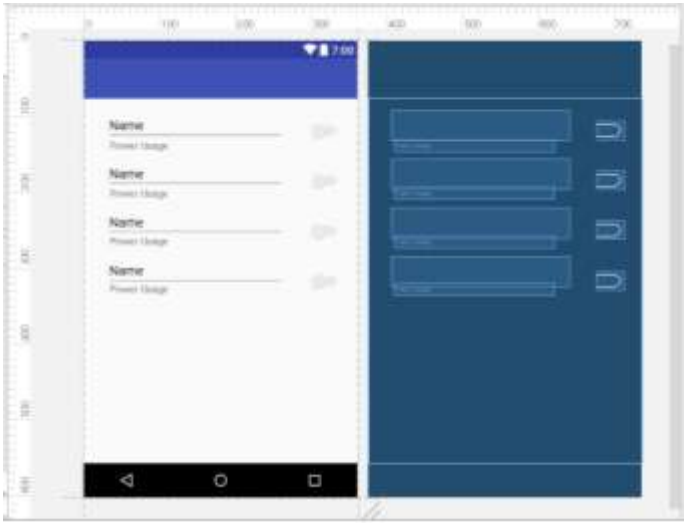
Gambar 3.9 *Menu Platform Config* pada prototipe *smart socket*



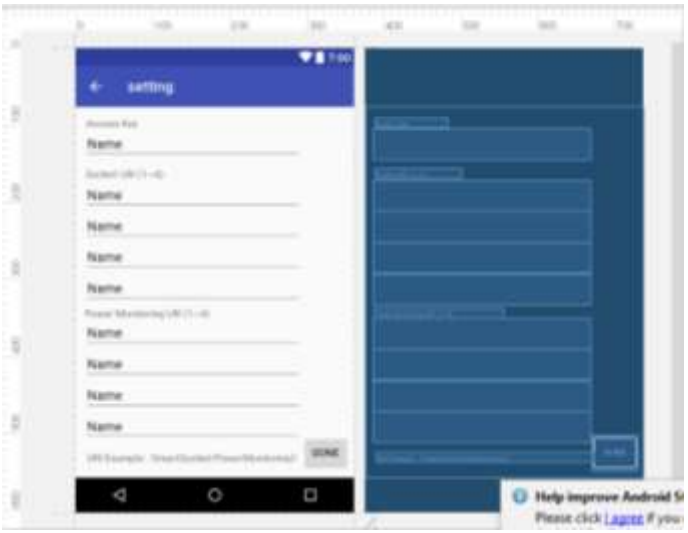
Gambar 3.10 *Menu Power Local Monitoring* prototipe *smart socket*

2. *User Interface* (UI) untuk Aplikasi Android

Untuk membuat UI pada aplikasi Android digunakan raw data XML untuk layout pada *software* Android Studio. Untuk membuat perintah-perintah pada masing-masing komponen digunakan *coding* Javascript. Aktivitas dibuat sebanyak 2 halaman, yang mencakup *main activity dan setting activity* yang ditunjukkan pada Gambar 3.11 dan 3.12.



Gambar 3.11 *Layout main cctivity* aplikasi Android



Gambar 3.12 *Layout setting activity* aplikasi Android

3.2.6 Perancangan s*ketch* program Arduino IDE

Sketch program arduino IDE merupakan *coding* berbasis bahasa C yang dijadikan *firmware* dari prototipe *smart socket.* Untuk meng-*upload coding* pada modul ESP-12E, digunakan FTDI USB to TTL.

3.2.7 Perancangan *sketch* javascript Android Studio

Sketch program Android Studio merupakan *coding* berbasis Javascript yang dijadikan perintah-perintah untuk aplikasi Android yang dirancang. Pada Android Studio *coding* yang dibuat terdiri dari beberapa *activity*. Masing-masing dari *activity* memiliki *coding*-nya masing-masing. Hasil *coding* yang dibuat dapat langsung dibangun menjadi aplikasi pada Android atau Android SDK *Simulator*.